latent-GLAT: Glancing at Latent Variables for Parallel Text Generation

Yu Bao[♠♢], **Hao Zhou**[♣], **Shujian Huang**[♠♢*], **Dongqi Wang**[♠♢]
Lihua Qian[♣], **Xinyu Dai**[♠♢], **Jiajun Chen**[♠♢], **Lei Li**[♡†]
[♠]National Key Laboratory for Novel Software Technology, Nanjing University, China
[♢]Collaborative Innovation Center of Novel Software Technology and Industrialization, China
[♣]Bytedance AI Lab, China [♡]University of California Santa Barbara, USA
{baoy,wangdq}@smail.nju.edu.cn
{zhouhao.nlp,qianlihua}@bytedance.com
{huangsj,daixinyu,chenjj}@nju.edu.cn leili@cs.ucsb.edu

Abstract

Recently, parallel text generation has received widespread attention due to its success in generation efficiency. Although many advanced techniques are proposed to improve its generation quality, they still need the help of an autoregressive model for training to overcome the one-to-many multi-modal phenomenon in the dataset, limiting their applications. In this paper, we propose *latent*-GLAT, which employs the discrete latent variables to capture word categorical information and invoke an advanced curriculum learning technique, alleviating the multi-modality problem. Experiment results show that our method outperforms strong baselines without the help of an autoregressive model, which further broadens the application scenarios of the parallel decoding paradigm. [‡]

1 Introduction

Non-autoregressive Transformer (NAT, Gu et al., 2018) introduce a parallel decoding paradigm with higher decoding efficiency ($> 10\times$) than autoregressive models (Bahdanau et al., 2015; Gehring et al., 2017; Vaswani et al., 2017). Unlike autoregressive models, NAT models impose conditional independence assumptions in words to support parallel decoding of sentences during inference. It attracts many researchers to explore NAT in machine translation (Gu et al., 2018; Lee et al., 2018; Kaiser et al., 2018) and text-to-speech tasks (Chen et al., 2019; Peng et al., 2020).

Amount of researchers devoted themselves to improve the NATs' inferior generation quality. Such as modeling word inter-dependencies by curriculum learning (Guo et al., 2020a; Liu et al., 2020) or iterative refinements mechanism (Ghazvininejad et al., 2019; Guo et al., 2020b), introducing latent variables to decompose target sentences and serve as the springboard for decoding (Shu et al., 2019; Ma et al., 2019; Bao et al., 2021), and introduce inductive bias for models' training (Wei et al., 2019; Li et al., 2019). The most successful method is the glancing transformer (GLAT, Qian et al., 2021a), which trains the NAT model by sampling partial target words as inputs to predict the remaining target words, explicitly building dependencies between the observed and unobserved words. Qian et al. (2021b) employ GLAT to achieve impressive results on the translation task of WMT21[1], even outperforming many strong autoregressive translation systems in BLEU score (Papineni et al., 2002).

Although existing NAT models achieve competitive results compared to autoregressive models in translation tasks, it is not negligible that they still need the help of an autoregressive Transformer (AT, Vaswani et al., 2017) as a teacher for training, i.e., sequence-level knowledge distillation (Kim and Rush, 2016). A well-recognized explanation is a *multi-modality problem* (Zhou et al., 2020; Sun and Yang, 2020): each input may have multiple valid outputs in datasets, which will prevent NAT models from learning to organize consistent outputs. Training with the outputs of an AT can directly bypass the multi-modal phenomenon in the dataset, effectively improving the models' performances.

However, training NAT models by knowledge distillation are limited. First, it needs to train an extra AT model, which inevitably enlarges the training cost. Second, it is hard to promise that the teacher (or AT) model can be accurate enough in all text generation settings, which will become the bottleneck for its student NAT model. Therefore, training a model from scratch without the help of an AT model is still an open and interesting problem.

In this paper, we propose *latent*-GLAT, which can directly learn from the raw dataset. It alleviates

[*]Shujian Huang is the corresponding author.
[†]Work is done while at ByteDance AI Lab.
[‡]The implementation of *latent*-GLAT will be released at https://github.com/baoy-nlp/Latent-GLAT.
[1]http://statmt.org/wmt21/

the multi-modality problem following a divide-and-conquer spirit, introducing a small set of discrete latent variables to capture the target word categorical information and divide the origin goal into latent variables modeling and sentence reconstruction. First, the categorical information may have fewer multi-modality phenomena than the original words, thus can be learned directly without the help of knowledge distillation. Second, the word categorical information is informativeness to the sentence reconstruction. We can extend glancing training with these discrete latent variables for modeling the sentence, encouraging the model to build dependencies on word categorical information rather than words, which works more robustly.

Experiment results on WMT14, Quora, and DailyDialog datasets show that *latent*-GLAT achieves remarkable improvements over several strong baselines, verifying the effectiveness of *latent*-GLAT. More impressively, *latent*-GLAT even outperforms autoregressive models in Quora and DailyDialog datasets, further validating our motivation for removing knowledge distillation. In-depth analyses indicate that the introduced discrete latent variables are helpful to alleviate the multi-modality problem and are necessary for performance improvement.

2 Background

For a sequence-to-sequence task of predicting sequence $Y = (y_1, y_2, \cdots, y_m)$ given its input sequence $X = (x_1, x_2, \cdots, x_n)$, the classical autoregressively factorization decomposes the $p(Y|X)$ with a series of conditional probability:

$$p_{\text{AT}}(Y|X) = \prod_{t=1}^{m} p(y_t|y_{<t}, X), \tag{1}$$

where $y_{<t} = (y_1, y_2, \cdots, y_{t-1})$.

Although such factorization achieved great success in previous studies (Bahdanau et al., 2015; Gehring et al., 2017; Vaswani et al., 2017), they predict each word[2] based on the prefix words, which may suffer from the issues of error accumulation and slow decoding during inference.

Non-autoregressive Transformer. To tackle the above problems, Gu et al. (2018) firstly propose non-autoregressive Transformer (NAT), introducing a non-autoregressive factorization as:

$$p_{\text{NAT}}(Y|X) = \prod_{t=1}^{m} p(y_t|X), \tag{2}$$

where each word y_t are modeled independently. During inference, the NAT model can decode the word simultaneously by $\arg\max_{y_t} p(y_t|X)$ for each y_t, remarkably improving the efficiency (15× speedups to an autoregressive Transformer).

However, the independence assumption may prevent the NAT model from leveraging the inherent word dependencies to organize consistent outputs. Due to this,the efficiency improvements of NAT are at the cost of its quality, e.g., the performance degradation by more than 10.0 BLEU (Papineni et al., 2002) points in machine translation tasks (Gu et al., 2018). Besides, recent studies (Zhou et al., 2020; Sun and Yang, 2020) point out that the multi-modality phenomenon in the dataset aggravates the challenge of NAT models.

Glancing Transformer. To mitigate the issue of missing word dependency in NAT models, Qian et al. (2021a) propose Glancing Transformer (GLAT), introducing glancing training (GLT) and sampling partial target tokens for training NAT:

$$\begin{aligned}\mathcal{L}_{\text{GLAT}} &= -\log p(\overline{Y_{\text{obs}}}|Y_{\text{obs}}, X) \\ &= -\sum_{y_i \in \overline{Y_{\text{obs}}}} \log p(y_i|Y_{\text{obs}}, X),\end{aligned} \tag{3}$$

where Y_{obs} is the partial target tokens, and $\overline{Y_{\text{obs}}}$ is its complements set. It progressively decreases the sampling ratio and obtains better performances in machine translation tasks.

Nevertheless, we find that GLAT in experiments still has a multi-modality problem[3]: First, its sampling rate cannot be decreased to zero during training, which exists the issue of *exposure bias*. Second, it still heavily relies on a teacher model for further improvements (Qian et al., 2021a).

Latent Transformer. To alleviate the multi-modality problem, Kaiser et al. (2018); Shu et al. (2019); Ma et al. (2019); Bao et al. (2021) propose Latent Transformer (LT), introducing latent variables $\boldsymbol{z}$ for NAT predictions as:

$$p_{\text{LT}}(Y|X) = \int_{\boldsymbol{z}} p(\boldsymbol{z}|X) \cdot p(Y|\boldsymbol{z}, X). \tag{4}$$

[2]We use BPE segmentation in our experiments, and they are strictly tokens. For clarity, we use words and tokens interchangeably in the paper.

[3]We include details of GLAT in Appendix A.

where $p_{\text{LT}}(Y|X)$ is always trained by variational inference (Ma et al., 2019) or discretization techniques (Kaiser et al., 2018). Such latent variables are decomposed from the target sentence, which is informative to determine the mode of the sentence and alleviates the multi-modality problems.

Although Latent Transformer models improve performance in terms of BLEU score, their used autoregressive predictor (Kaiser et al., 2018; Bao et al., 2021) or deep iterative transformation (Shu et al., 2019; Ma et al., 2019) for predicting latent variables unavoidable sacrifice the overall decoding efficiency. Besides, they do not explicitly build the interdependencies among the outputs.

3 Proposed Method: *latent*-GLAT

In this section, we present *latent*-GLAT. *latent*-GLAT follows Latent Transformer models (Kaiser et al., 2018; Bao et al., 2021) but introduces glancing training (Qian et al., 2021a) with the discrete latent variables. Our intuitions are as follows:

First, compared to the words, the introduced discrete latent variables may have fewer modes than words and be informative to determine the modes of the sentences. In such a case, we can directly learn the discrete latent variables by the Glancing Transformer (Qian et al., 2021a), keeping competitive inference efficiency. More importantly, we can employ the latent variables to invoke glancing training for modeling the target sentences, which is informative enough to reduce the multi-modality problem of original sentences. Besides, glancing at latent variables also works robustly due we can obtain the latent variables during inference.

3.1 Introducing Discrete Latent Variables for Modeling Target Categorical Information

In this part, we state the structure of *latent*-GLAT, which introduces a small set of discrete latent variables for a NAT model, basically following Kaiser et al. (2018); Roy et al. (2018); Bao et al. (2021).

Let K be the size of the discrete latent space and let $[K]$ denote the set $\{1, 2, \cdots, K\}$. For each target sentence $Y = (y_1, y_2, \cdots, y_m)$, we use a same-length latent variable sequence for modeling it as:

$$p(Y|X) = \sum_{\boldsymbol{z}} p_\theta(\boldsymbol{z}|X) \cdot \prod_{t=1}^{m} p_\theta(y_t|\boldsymbol{z}, X), \quad (5)$$

where $\boldsymbol{z} = (z_1, z_2, \cdots, z_m)$ and $z_i \in [K]$, θ is the model parameters.

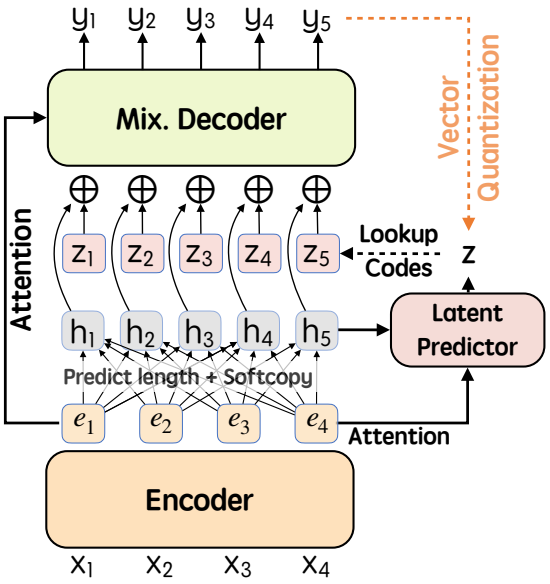


Figure 1: **Model architecture of *latent*-GLAT.** $\bigoplus$: Position-wise mix h_i and representation of z_i by a gated neural network.

Discretization. For discretizing target sentences to latent variables, we use *vector quantization* (Roy et al., 2018), which works by dividing a large set of origin vector representations into small groups. We assign each token y_i with a group $j \in [K]$ that has the nearest distance to its representation:

$$z_i = \arg\min_{j \in [K]} ||\operatorname{repr}(y_i) - \boldsymbol{q}_j||_2, \quad (6)$$

where $\boldsymbol{q} \in \mathbb{R}^{\text{K} \times d_{\text{model}}}$ is the maintained representations and d_{model} is its dimension. We use the embedding as $\operatorname{repr}(y_i)$, refer to Bao et al. (2021). Finally, the model is trained to minimize

$$\mathcal{L}_{\text{LT}} = \mathcal{L}_{\text{LP}} + \mathcal{L}_{\text{WP}}, \quad (7)$$

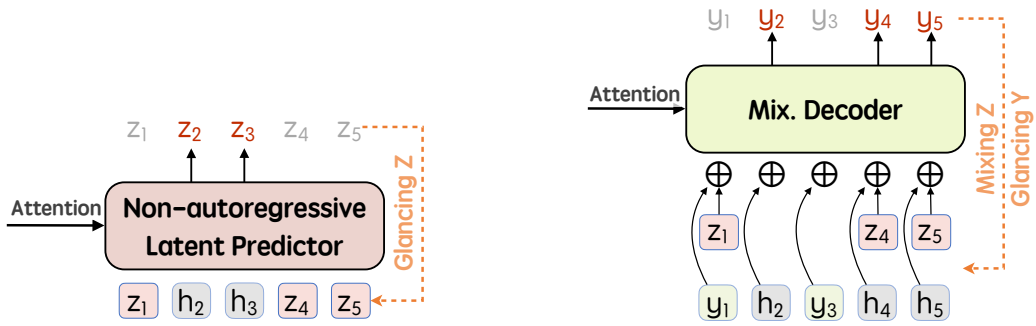
where $\mathcal{L}_{\text{WP}}$ and $\mathcal{L}_{\text{LP}}$ are the prediction loss for words Y and latent variables $\boldsymbol{z}$, respectively.

The maintained representations $\boldsymbol{q}$ are updated with an exponential moving average over a mini-batch of target tokens $\{y_1, \cdots, y_i, \cdots\}$:

$$\begin{aligned} c_j &\leftarrow \lambda c_j + (1-\lambda) \sum_i \mathbb{1}[z_i = j], \\ \boldsymbol{q}_j &\leftarrow \lambda \boldsymbol{q}_j + (1-\lambda) \sum_i \frac{\mathbb{1}[z_i = j] \operatorname{repr}(y_i)}{c_j} \end{aligned} \quad (8)$$

where c_j is assigned count for group j, and we set decay parameter $\lambda = 0.999$ in our experiments.

Architecture. As shown in Figure 1, *latent*-GLAT mainly consists of an encoder $\mathbb{F}_{\text{ENC}}$ (NAT



(a) $\mathcal{L}_{\text{LP}}^{\text{GLT}}$ for Latent Predictor (b) $\mathcal{L}_{\text{WP}}^{\text{GLT}}$ for Mixture Decoder

Figure 2: **Training the latent predictor and mixture decoder by glancing at discrete latent variables.**

Encoder), a latent predictor $\mathbb{F}_{\text{LP}}$ (NAT Predictor), and a decoder $\mathbb{F}_{\text{DEC}}$ (Mix. Decoder). We parameterize them with the multi-head attention-based encoder or decoder, similar to Transformer (Vaswani et al., 2017). Their functions can be formalized as:

$$\begin{aligned}(e_1, e_2, \cdots, e_n) &\leftarrow \mathbb{F}_{\text{ENC}}(x_1, x_2, \cdots, x_n),\\ (h_1, h_2, \cdots, h_m) &\leftarrow \text{softcopy}(e_{1:n}),\\ p_\theta(\boldsymbol{z}|X) &\leftarrow \mathbb{F}_{\text{LP}}(h_{1:m}, e_{1:n}),\\ p_\theta(Y|\boldsymbol{z}, X) &\leftarrow \mathbb{F}_{\text{DEC}}(z_{1:m}, h_{1:m}, e_{1:n}),\end{aligned}$$

where we use an extra module $\mathbb{F}_{\text{LEN}}$ to predict the target length m and initialize the decoder inputs $H = (h_1, h_2, \cdots, h_m)$ with the softcopy (Wei et al., 2019) mechanism.

3.2 Glancing at Discrete Latent Variables for Parallel Sequence Decoding

The small number ($K < 128$) of discrete latent variables can capture high-level categorical information of the target words, supporting better learning design for parallel sequence decoding.

Our first insight is that we can learn to non-autoregressively predict the discretized latent variables directly without the help of distillation. Specifically, we parameterize the $\mathbb{F}_{\text{LP}}$ in a non-autoregressive fashion and use a glancing training technique (GLT, Qian et al., 2021a) for optimizing it, as shown in Figure 2a:

$$\mathcal{L}_{\text{LP}}^{\text{GLT}} = -\log p_\theta(\overline{\boldsymbol{z}_{\text{obs}}}|\boldsymbol{z}_{\text{obs}}, X) \tag{9}$$

where $\boldsymbol{z}_{\text{obs}}$ is uniformly sampled from $\boldsymbol{z}$, refer to Qian et al. (2021a). We provide more training details of *latent*-GLAT in Appendix B.

Our next insight is modeling the sentence based on the sampled latent variables $\boldsymbol{z}_{\text{obs}}$ rather than $\boldsymbol{z}$, namely, glancing at $\boldsymbol{z}_{\text{obs}}$ for optimizing $\mathbb{F}_{\text{DEC}}$:

$$\mathcal{L}_{\text{WP}} = -\log p_\theta(Y|\boldsymbol{z}_{\text{obs}}, X). \tag{10}$$

We find Eqn. (10) works robustly in experiments and analyze it in Section (§ 4.3).

As shown in Figure 2b, we eventually employ words to invoke glancing training for minimizing $\mathcal{L}_{\text{WP}}$, namely we optimize the $\mathbb{F}_{\text{DEC}}$ by minimizing

$$\mathcal{L}_{\text{WP}}^{\text{GLT}} = -\log p_\theta(\overline{Y_{\text{obs}}}|\boldsymbol{z}_{\text{obs}}, Y_{\text{obs}}, X), \tag{11}$$

where Y_{obs} and $\boldsymbol{z}_{\text{obs}}$ are the sampled target tokens and discrete latent variables.

Overall Training Loss. Our full-fledged loss includes latent variable prediction, sentence reconstruction, and length prediction losses:

$$\mathcal{L} = \mathcal{L}_{\text{WP}}^{\text{GLT}} + \mathcal{L}_{\text{LP}}^{\text{GLT}} + \alpha\mathcal{L}_{\text{LEN}}, \tag{12}$$

where $\alpha = 0.1$ are the hyperparameters to adjust the importance of length prediction loss $\mathcal{L}_{\text{LEN}}$.

3.3 Inference

In inference phase, *latent*-GLAT predicts the target length, latent variables, and sentence in turn.

For the target length, *latent*-GLAT first predicts the target length m with the length predictor $\mathbb{F}_{\text{LEN}}$. To avoid the length prediction errors during inference, *latent*-GLAT expands the length m to a ranges (we use $[m-3, \cdots, m+2]$, total six candidates in our experiments).

Then, *latent*-GLAT predicts the latent variables $\hat{\boldsymbol{z}}$ with $\arg\max_{\boldsymbol{z}} p_\theta(\boldsymbol{z}|X)$ and sentence $\hat{Y}$ with $\arg\max_Y p_\theta(Y|\hat{\boldsymbol{z}}, X)$ for each candidate.

Similar to Ma et al. (2019), *latent*-GLAT also ranks the candidates by itself (*self-reranking*) and chooses the highest score output with:

$$\hat{Y} = \arg\max_Y p_\theta(Y|\hat{\boldsymbol{z}}, X) \cdot \gamma^{|Y|} \tag{13}$$

where γ is the length penalty ratio to avoid the length bias, and $|Y|$ denotes the length of Y.

4 Experiments

We conduct experiments on several generation tasks, including machine translation, paraphrase generation, and dialog generation.

4.1 Experimental Setup

Dataset. We chose the most popular benchmarks for each task:

- **Machine Translation (MT)**: We follow previous practices in NAT models and use the WMT14 English (EN) ↔ German (DE) corpus (4.5M sentence pairs) and the IWSLT14 German (DE) → English (EN) corpus (160K sentence pairs) to validate our proposed model. We obtain the datasets following the instruction open-sourced in `fairseq`[4]. In detail, we first tokenize the datasets with `Moses` script. Then, we use 37,000 and 10,000 operations to split the words into byte-pair encodings (BPE, Sennrich et al., 2016) in WMT14 and IWSLT14 datasets, respectively. We also share subword embeddings between the source and target language for each dataset.
- **Paraphrase Generation (PG)**: We use the Quora[5] dataset to evaluate the paraphrase generation task. The Quora dataset contains around 135K labeled paraphrases pairs. Following the standard dataset split, we sample 100K sentence pairs from the labeled paraphrases as training data and hold out 30K pairs for testing, the remaining about 5K pairs for validation. Like the MT tasks, we tokenize the corpus with Moses scripts and split the words into BPE units with total 32K operations.
- **Dialog Generation (DG)**: We conduct the dialog generation experiments on the DailyDialog dataset (Li et al., 2017). We obtain the processed DailyDialog dataset from Bao et al. (2020)[6]. The training set contains 87,170 sentence pairs (11,118 dialogues). The validation and testing set in the dataset contain 8069 pairs (1000 dialogues) and 7740 pairs (1000 dialogues), respectively.

Note that these tasks emphasize different aspects. The task of MT aims to transfer bilingual sentences with semantically invariant conditions. The PG task differs from machine translation and works on mode transformation in the same language, whose goal is to synthesize a sentence different from the original input but conveys the same meaning. The DG task is most challenging due to the complex generation goal.

Implementations. We compare *latent*-GLAT with Transformer (Vaswani et al., 2017), NAT (Gu et al., 2018), and GLAT (Qian et al., 2021a) models. We implement them based on the open-source framework `fairseq` (Ott et al., 2019).

For machine translation tasks, we use the base setting ($d_{\text{model}} = 512$, $d_{\text{hidden}} = 2048$, dropout $= 0.1$, $n_{\text{head}} = 8$, and $n_{\text{layer}} = 6$) of Transformer (Vaswani et al., 2017) for WMT14 dataset and a smaller setting ($d_{\text{model}} = 512$, $d_{\text{hidden}} = 1024$, dropout $= 0.3$, $n_{\text{head}} = 4$, and $n_{\text{layer}} = 6$) for IWSLT14 dataset. The number of layers in *latent*-GLAT decoder and latent predictor are both set to 4 in experiments. We use inverse square root learning rate scheduling for WMT14 and a linear annealing learning rate from 3.0×10^{-4} to 1.0×10^{-5} in 250K steps for IWSLT14. The models are optimized with Adam (Kingma and Ba, 2015) optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) in 300K steps for WMT14 and 250K steps for IWSLT14. As for the ratio τ that used in glancing sampling, we linear anneal the ratio from 0.5 to 0.3 in whole training steps. The mini-batch in each step consists of 2K tokens for IWSLT14 and 64K tokens for WMT14.

Since the scale of the Quora and DailyDialog datasets are close to the IWSLT14, we keep the same setting to the IWSLT14, such as the Adam, learning rate (linear annealing from 3.0×10^{-4} to 1.0×10^{-5}), and batch size (2K tokens).

Evaluation. To validate the effectiveness of our proposed method, we evaluate it in terms of quality and efficiency. We use tokenized and cased BLEU scores (Papineni et al., 2002)[7] to evaluate the generation quality of MT and PG tasks. For dialog generation, we also include BLEU-1 and BLEU-2 scores for analysis. Following the common practices (Gu et al., 2018; Qian et al., 2021a), we measure the decoding latency of each model by decoding sentence by sentence and compute the speedup compared with the autoregressive Transformer (AT) model to reflect its decoding efficiency. We highlight the **best NAT** result.

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

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

[6] https://github.com/gmftbyGMFTBY/MultiTurnDialogZoo

[7] We evaluate BLEU using `fairseq_score` script.

Models	WMT14		IWSLT14	Quora	DailyDialog			Latency$^{\downarrow}$	Speedups$^{\uparrow}$
	EN→DE	DE→EN	DE→EN		BLEU-1	BLEU-2	BLEU		
Transformer (AT)	27.17	31.53	34.29	27.97	31.40	10.70	5.05	512.3 ms	1.00 ×
NAT	10.78	15.19	17.77	24.65	**41.50**	1.40	0.01	**33.5** ms	**15.29** ×
GLAT	16.71	24.78	29.07	27.01	39.50	26.20	26.13	**33.5** ms	**15.29** ×
latent-GLAT	**24.71**	**29.16**	**32.31**	**29.11**	41.00	**28.30**	**27.50**	45.3 ms	11.31 ×

Table 1: **Main results of different models on the test set of each dataset.** We measure the decoding latency and speedups on the WMT14 EN→DE test set.

4.2 Main Results

We can see from Table 1 that our *latent*-GLAT almost outperforms all the NAT baselines (NAT and GLAT) in generation quality on all tasks while keeping a competitive decoding speedup to the autoregressive counterpart.

Machine Translation. As seen, without the help of an AT model for training, the vanilla NAT and advanced GLAT model only obtain inferior generation quality. In contrast, *latent*-GLAT achieves competitive generation quality in machine translation tasks, indicating that the introduced latent variables effectively reduce the multi-modality issue and support glancing training well. It narrows the performance gap between non-autoregressive decoding and autoregressive decoding from 11.46 (GLAT vs. AT) to 2.34 (*latent*-GLAT vs. AT) BLEU points on WMT14 EN→DE task while keeping a high-speed decoding efficiency.

Paraphrasing. Unlike the translation task, the performance gap between non-autoregressive and autoregressive decoding on the paraphrase generation task is minor (NAT vs. AT, -3.32 BLEU points, GLAT vs. AT, -0.96 BLEU points). Nevertheless, introducing discrete latent variables still is helpful to obtain a better performance. *latent*-GLAT realizes a non-autoregressive model with better performance than the autoregressive model on Quora (*latent*-GLAT vs. AT, $+1.14$ points).

Dialog Generation. We can see a different trend on the DailyDialog dataset — an AT model performs poorly than NAT models. Both GLAT and *latent*-GLAT outperform the AT model in BLEU-1, BLEU-2, and BLEU scores, indicating that these models recall more reference tokens and organize the tokens well.

We conjecture that the weak and indirect association between the inputs and outputs of the dialogue results in this unusual phenomenon. Specifically, the weak connection may encourage the AT model to predict the tokens by paying more attention to their history outputs, which degenerate to a target-side language model. In contrast, the NAT models do not have this fast track, pushing them to pay more attention to the inputs and recall more target tokens. We further find that there are so-called *safe response* (Li et al., 2016) in AT's outputs, which verify our conjecture.

Models	WMT14		IWSLT14	Speedups$^{\uparrow}$
	EN→DE	DE→EN	DE→EN	
CMLM_1	*10.88	-	-	-
CMLM_4	*22.06	-	-	†9.79 ×
CMLM_{10}	*24.65	-	-	†3.77 ×
$\text{LevT}_{2.05}$	24.43	-	-	2.93 ×
LV-NAR	11.80	-	-	**22.30** ×
SynST	20.74	25.50	23.82	4.86 ×
Flowseq	20.85	25.40	-	‡1.10 ×
CNAT	21.30	25.73	29.81	10.37 ×
AT	27.17	31.53	34.29	1.00 ×
NAT	10.78	15.19	17.77	15.29 ×
GLAT	16.71	24.78	29.07	15.29 ×
latent-GLAT	**24.71**	**29.16**	**32.31**	11.31 ×

Table 2: **BLEU scores and speedups of different models trained with raw datasets on machine translation tasks.** We quote some results from *Ma et al. (2019), †Guo et al. (2020b), ‡Qian et al. (2021a), and the original paper. CMLM_n and LevT_n: using n iterations during inference. −: no corresponding results.

More Comparisons. we further compare the advanced NAT models that builds upon latent variables or iterative refinement in machine translation tasks:

- NATs w/ latent variables: LV-NAR (Shu et al., 2019), SynST (Akoury et al., 2019), Flowseq (Ma et al., 2019), and CNAT (Bao et al., 2021).
- Iterative NATs: CMLM (Ghazvininejad et al., 2019) and LevT (Gu et al., 2019).

Table 2 shows that introducing latent variables

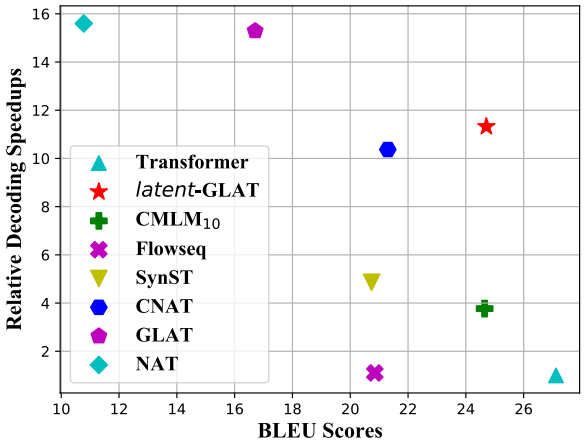


Figure 3: **BLEU scores and their relative decoding speedups of different models on WMT14 EN→DE test set.** Note that we evaluate the speedups with a single GTX 1080-Ti GPU and include the results with the same evaluating hardware for fair comparisons.

(LV-NAR, Flowseq, and CNAT) or decoding with multiple iterations (CMLM and LevT) both improve non-autoregressive decoding in translation quality. However, iterative refinements or deep transformations always sacrifice decoding efficiency. In contrast, the proposed *latent*-GLAT outperforms all NAT models with a relatively low cost, keeping a competitive speedup over autoregressive Transformer (AT). Specifically, *latent*-GLAT with one-pass decoding narrows the performance gap to the AT from 5.87 BLEU points to 2.34 BLEU points on the WMT14 EN→DE test set.

Decoding efficiency. We can see there is a trade-off between the translation quality and decoding efficiency in Table 2. We thus present the scatter plot of different models in Figure 3, showing the trend of translation quality and decoding efficiency.

As seen, *latent*-GLAT is located on the top-right of the baselines. It outperforms the baselines in the BLEU score if decoding speedup is fixed and in decoding speedup if the BLEU score is fixed.

4.3 Analysis

We now turn to verify our intuition that *latent*-GLAT can alleviate the multi-modality problem.

***latent*-GLAT largely alleviates the sentence-level multi-modal problem.** Previous researches (Gu et al., 2018; Ma et al., 2019; Qian et al., 2021a; Bao et al., 2021) always utilize a Transformer model as a teacher for training NAT models, namely sequence-level knowledge distillation (Kim and Rush, 2016), which can directly reduces the sentence-level multi-modal phenomenon in datasets. Therefore, we use the average gains from the knowledge distillation to reflect the ability of the NAT models to overcome this issue.

Methods	WMT14		IWSLT14	Avg $\Delta^{\downarrow}$
	EN→DE	DE→EN	DE→EN	
NAT	10.78	15.19	17.77	+6.58
w/ KD	17.69	22.02	23.78	
GLAT	16.71	24.78	29.07	+5.19
w/ KD	25.21	29.84	31.07	
Flowseq	20.85	25.40	24.75	+2.87
w/ KD	23.72	28.39	27.55	
CNAT	21.30	25.73	29.81	+3.08
w/ KD	25.56	29.36	31.15	
latent-GLAT	24.71	29.16	32.31	**+0.95**
w/ KD	**26.64**	**29.93**	**32.47**	

Table 3: **BLEU scores of NAT models trained with (or without) knowledge distillation (KD) on translation tasks.**

Datasets	Configuration (d)	$C_{\text{TOK}}(d)$	$C_{\text{SEN}}(d)$
WMT14	Inputs ↔ Raw outputs	2.19	3.03
	Inputs ↔ AT outputs	1.38	2.13
	Inputs ↔ $\boldsymbol{z}$	**1.01**	**1.35**
Quora	Inputs ↔ Raw outputs	0.86	1.48
DailyDialog	Inputs ↔ Raw outputs	1.19	4.23

Table 4: **Token-level or sentence-level complexity of different text generation datasets.** The higher $C_{\text{TOK}}(d)$ or $C_{\text{SEN}}(d)$, the more complex.

As seen in Table 3, the pure NAT models heavily rely on knowledge distillation. By introducing the target information with the latent variables (Flowseq and CNAT) or sampled tokens (GLAT), the NAT models improve its' ability to overcome the multi-modality issue. Our proposed *latent*-GLAT well combines the above two techniques. It obtains only 0.95 BLEU points average gains and validates our motivation.

Discrete latent variables have fewer modes than raw sentences. To validate our intuition that the introduced latent variables are easier to predict than tokens, we refer to Zhou et al. (2020) to compute the complexity metrics on each dataset according to alignment relations. Specifically, we use the *fast_align*[8] toolkit to align source input X and target outputs Y or discretized latent variable se-

[8] https://github.com/clab/fast_align

L#	Introduce z	Glancing Training		BLEU $(\Delta)^{\uparrow}$
		with z	with Y	
1				12.60
2	✓			13.43 (+0.83)
3			✓	17.11 (+4.51)
4	✓		✓	18.88 (+6.20)
5	✓	✓		22.35 (+9.75)
6	✓	✓	✓	**23.64** (+11.04)

Table 5: **BLEU scores of different *latent*-GLAT configurations on the WMT14 EN→DE valid set.**

K	8	16	32	64	128	256
BLEU (%)	20.80	22.16	22.61	**23.64**	23.26	21.94
ACC$_z$ (%)	61.20	53.10	43.57	39.24	36.39	33.84

Table 6: **Performances of *latent*-GLAT with different K on the WMT14 EN→DE valid set.** We compute the accuracy (ACC$_z$) of latent prediction by taking the discretized latent variables as reference.

quences z. Then, we compute the token-level complexity $C_{\text{TOK}}(d)$ and the sentence-level complexity $C_{\text{SEN}}(d)$ according to Zhou et al. (2020). These metrics can trivially understand as the number of valid candidates for each input.

As shown in Table 4, the latent variables have the lowest complexity in both token-level complexity and sentence-level complexity. In other words, predicting the latent variable sequences is effortless than predicting others, which is consistent with our intuition. Although we obtain a lower complexity dataset by filtering the datasets with an autoregressive model (AT outputs versus Raw outputs), they may introduce model error and need extra training for AT model. In contrast, the discrete latent variables are simple and informative enough to serve as a springboard for modeling target sentences.

Glancing with latent variables improves the performance with a large margin. We can see in Table 5 that introducing latent variables both obtain performance gains to their counterpart (L#2 vs. L#1, +0.83 points, and L#4 vs. L#3, +1.69 points). As expected, the gains are largely improved while adopting the glancing training with discrete latent variables (L#5 vs. L#1, +9.75 points), which already outperforms glancing training with the reference token (L#5 vs. L#4, +3.55 points). Finally, we jointly perform glancing training with the reference tokens and discrete latent variables, achieving the best result (L#6 vs. L#1, +11.04 points).

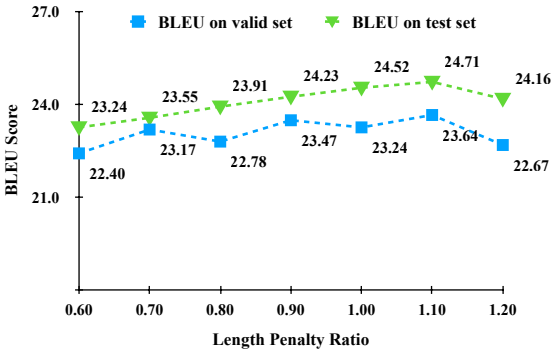


Figure 4: **BLEU scores of *latent*-GLAT using different length penalty ratios on the WMT14 EN→DE valid set.** We search the length penalty ratio γ for *latent*-GLAT while fixing the $K = 64$.

Effects of K and γ. As shown in Figure 4 and Table 6, we search the hyper-parameter of *latent*-GLAT that the number of discrete latent variables and the length penalty ratio γ according to the validation performance. We notice that using more latent codes causes performance degradation during inference, in which the latent variables may degenerate to tokens and contains more prediction error during inference. The *latent*-GLAT implemented with 64 latent variables and $\gamma = 1.1$ obtains the best result on WMT14 EN→DE valid set.

5 Related Work

Gu et al. (2018) first propose a non-autoregressive Transformer (NAT) model for neural machine translation (NMT) and begin to explore parallel decoding. It abandons explicitly modeling word interdependencies to decode the tokens in parallel, significantly improving the inference speed. However, its translation quality is inferior to the Transformer (Vaswani et al., 2017).

To alleviate this performance degradation, many researchers work to enhance word dependency modeling, including imitation learning (Wei et al., 2019; Li et al., 2019), curriculum learning (Guo et al., 2020a; Liu et al., 2020), iterative refinements (Lee et al., 2018; Ghazvininejad et al., 2019; Gu et al., 2019; Guo et al., 2020b; Huang et al., 2022), and a simplified autoregressive process (Sun et al., 2019). The most representative method is the glancing transformer model (Qian et al., 2021a), which adaptively and progressively samples partial tokens as inputs and predicts the remaining tokens, effectively establishing the dependencies between the sampled tokens and the remaining tokens. However, these models still rely on a teacher

for training, which cannot directly learn the raw dataset that contains one-to-many multi-modality phenomenon.

Introducing latent variables (Bao et al., 2019, 2021) to organize the target sentence is also a helpful route. Among them, our method is close to Kaiser et al. (2018); Shu et al. (2019); Ma et al. (2019); Akoury et al. (2019); Bao et al. (2021). These methods decompose the latent variables (hints) from the target sentence and divide the origin goal into two parts: modeling latent variables and modeling the target sentences based on latent variables. It implicitly overcomes the multi-modality phenomenon of target sentences because the latent variables can largely determine the mode of the sentence. However, these methods always model the latent variables with an autoregressive predictor, which naturally sacrifices the decoding efficiency.

Unlike them, our approach models the discrete latent variables in a non-autoregressive fashion and extends glancing training with the discrete latent variables. As a result, *latent*-GLAT accomplishes a competitive performance both in decoding efficiency and quality.

6 Conclusion

We propose *latent*-GLAT, which can be directly trained without the help of knowledge distillation. Specifically, we employ discrete latent variables to capture the word categorical information and divide the original goal into the latent variables modeling and word prediction tasks. Then, we learn each task with the glancing training and encourage the model to build dependencies on the latent variables, which have fewer modes than the words and are also informative for modeling the target sentences. Experiments results on machine translation, paraphrase generation, and dialogue generation tasks validate the effectiveness of our *latent*-GLAT.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang is the corresponding author. This work is supported by National Science Foundation of China (No. U1836221, 6217020152).

References

Nader Akoury, Kalpesh Krishna, and Mohit Iyyer. 2019. Syntactically supervised transformers for faster neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1269–1281, Florence, Italy. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, Online. Association for Computational Linguistics.

Yu Bao, Shujian Huang, Tong Xiao, Dongqi Wang, Xinyu Dai, and Jiajun Chen. 2021. Non-autoregressive translation by learning target categorical codes. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5749–5759, Online. Association for Computational Linguistics.

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

Nanxin Chen, Shinji Watanabe, Jesús Villalba, and Najim Dehak. 2019. Listen and fill in the missing letters: Non-autoregressive transformer for speech recognition. *arXiv preprint arXiv:1911.04908*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. 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)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

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. 2020a. 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.

Junliang Guo, Linli Xu, and Enhong Chen. 2020b. Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 376–385, Online. Association for Computational Linguistics.

Richard W Hamming. 1950. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160.

Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. 2022. Non-autoregressive translation with layer-wise prediction and deep supervision. In *AAAI*.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2395–2404. PMLR.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

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*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Zhuohan Li, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Hint-based training for non-autoregressive translation. In *NeuralIPS (to appear)*.

Jinglin Liu, Yi Ren, Chen Zhang Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. Task-level curriculum learning for non-autoregressive neural machine translation. *AAAI*.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard 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)*, pages 4282–4292, Hong Kong, China. Association for Computational Linguistics.

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 (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. 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*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Kainan Peng, Wei Ping, Zhao Song, and Kexin Zhao. 2020. Non-autoregressive neural text-to-speech. 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 7586–7598. PMLR.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021a. Glancing transformer for non-autoregressive neural machine translation. In *ACL*.

Lihua Qian, Yi Zhou, Zaixiang Zheng, Yaoming Zhu, Zehui Lin, Jiangtao Feng, Shanbo Cheng, Lei Li, Mingxuan Wang, and Hao Zhou.

2021b. The volctrans glat system: Non-autoregressive translation meets wmt21. *arXiv preprint arXiv:2109.11247*.

Aurko Roy, Ashish Vaswani, Niki Parmar, and Arvind Neelakantan. 2018. Towards a better understanding of vector quantized autoencoders. *arXiv*.

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 (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2019. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. *arXiv preprint arXiv:1908.07181*.

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.

Zhiqing Sun and Yiming Yang. 2020. An em approach to non-autoregressive conditional sequence generation. In *International Conference on Machine Learning*, pages 9249–9258. PMLR.

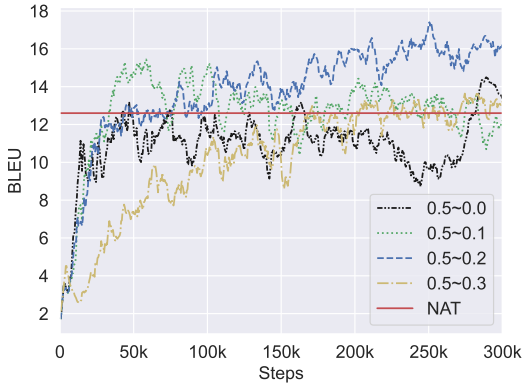
Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017,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

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 Annual Meeting of the Association for Computational Linguistics*, pages 1304–1312, Florence, Italy. Association for Computational Linguistics.

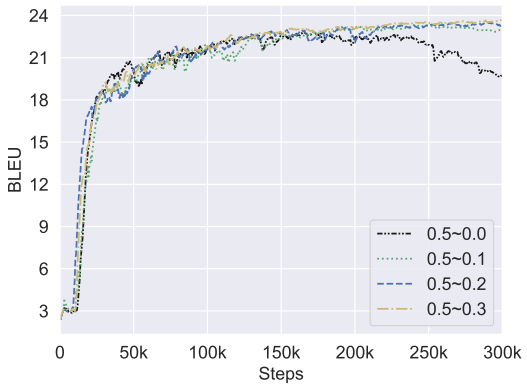
Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Details of GLAT

According to the performance shown in Figure 5a, we can see a GLAT model will degenerate to a NAT model while using a small sampling ratio. In such a case, introducing an autoregressive Transformer as a teacher for training the GLAT model alleviates this issue (Figure 5b), indicating that the GLAT model still needs the help of knowledge distillation for alleviating multi-modality problems.



(a) GLAT w/ raw data.



(b) GLAT w/ distillation.

Figure 5: BLEU score and training steps of GLAT trained with different glancing strategy (start → end ratio).

B Model Details of *latent*-GLAT

Decoder Inputs. Following the most common practices in NAT models (Wei et al., 2019; Li et al., 2019), we use *Softcopy* mechanism for initializing the decoder inputs $\boldsymbol{H} = (h_1, h_2, \cdots, h_m)$:

$$h_i = \sum_{i}^{n} \alpha_{ij} \cdot e_i, \qquad \alpha_{ij} \propto \exp\left[-(i - j \cdot \frac{n}{m})^2\right], \tag{14}$$

where $\boldsymbol{E} = (e_1, e_2, \cdots, e_n)$ is the encoded representation of $X = (x_1, x_2, \cdots, x_n)$, n and m are the length of source and target sentences, respectively.

Training the Latent Predictor by glancing sampling discrete latent variables. With the decoder input $H = h_{1:m}$ and the discretized latent variable sequence $\boldsymbol{z} = z_{1:m}$, we adopt the glancing sampling technique for training the latent predictor in the following steps:

- **Predicting $\hat{\boldsymbol{z}}$**: *latent*-GLAT predicts the latent variable sequence with its latent predictor: $\hat{\boldsymbol{z}} \leftarrow \mathbb{F}_{\text{LP}}(h_{1:m}, e_{1:n})$.
- **Determining sample number $N_{\mathbf{z}}$**: Given $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$, we compute the sampling number as:

$$N_{\text{z}} = \tau \cdot \text{Hamming}(\boldsymbol{z}, \hat{\boldsymbol{z}}) \qquad (15)$$

  where τ is the sampling ratio decreasing in the training steps, and we use Hamming distance (Hamming, 1950) for measuring the prediction quality.
- **Sampling observed latent variables $\boldsymbol{z}_{\text{obs}}$**: Given discretized latent variable sequence $\boldsymbol{z}$ and sample number N_{z}, we obtain $\boldsymbol{z}_{\text{obs}}$ by random selecting N_{z} elements from $\boldsymbol{z}$.
- **Re-constructing inputs H_{LP}**: We construct H_{LP} by position-wise replacing the decoder input $h_{1:m}$ with $\boldsymbol{z}_{\text{obs}}$.
- **Updating Latent Predictor**: With the H_{LP} as inputs, we train the latent predictor to predict the unobserved references $\overline{\boldsymbol{z}_{\text{obs}}}$.

Training the Mix. Decoder with sampled discrete latent variables. Training of Mix. Decoder is largely follow the Qian et al. (2021a), except using extra latent variables as inputs. With the input $H = h_{1:m}$, the reference sentence Y, and the sampled latent variables $\boldsymbol{z}_{\text{obs}}$, we train Mix. Decoder in the following steps:

- **Predicting $\hat{Y}$**: *latent*-GLAT predicts the target sentences: $\hat{Y} \leftarrow \mathbb{F}_{\text{DEC}}(\boldsymbol{z}_{\text{obs}}, h_{1:m}, e_{1:n})$.
- **Determining sample number N_{y}**: Given Y and $\hat{Y}$, we compute the sampling number $N_{\text{y}} = \tau \cdot \text{Hamming}(Y, \hat{Y})$.
- **Sampling target tokens Y_{obs}**: We obtain the glancing reference Y_{obs} by random selecting N_{y} tokens from reference sequence Y.
- **Re-constructing inputs H_{DEC}**: H_{DEC} is constructed by position-wise replacing the decoder input H with embedding of Y_{obs}.
- **Updating Mix. Decoder**: We then train the Mix. Decoder to predict the unobserved references $\overline{Y_{\text{obs}}}$, with the H_{DEC} and $\boldsymbol{z}_{\text{obs}}$ as inputs.