Non-autoregressive Translation with Dependency-Aware Decoder

Jiaao Zhan\textsuperscript{1}, Qian Chen\textsuperscript{2}, Boxing Chen\textsuperscript{2}, Wen Wang\textsuperscript{2}, Yu Bai\textsuperscript{1}, Yang Gao\textsuperscript{1}\textsuperscript{†}

\textsuperscript{1}School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
\textsuperscript{2}Machine Intelligence Technology Lab, Alibaba DAMO Academy

\{jiaao_zhan, yubai, gyang\}@bit.edu.cn
\{tanqing.cq, Boxing.cbx, w.wang\}@alibaba-inc.com

Abstract

Non-autoregressive translation (NAT) models suffer from inferior translation quality due to removal of dependency on previous target tokens from inputs to the decoder. In this paper, we propose a novel and general approach to enhance the target dependency within the NAT decoder from two perspectives: \textit{decoder input} and \textit{decoder self-attention}. First, we transform the initial decoder input from the source language space to the target language space through a novel attentive transformation process. The transformation reassembles the decoder input based on target token embeddings and conditions the final output on the target-side information. Second, before NAT training, we introduce an effective forward-backward pre-training phase, implemented with different triangle attention masks. This pre-training phase enables the model to gradually learn bidirectional dependencies for the final NAT decoding process. Experimental results demonstrate that the proposed approaches consistently improve highly competitive NAT models on four WMT translation directions by up to 1.88 BLEU score, while overall maintaining inference latency comparable to other fully NAT models.\textsuperscript{1}

1 Introduction

Neural machine translation (NMT) systems based on autoregressive models achieve state-of-the-art (SOTA) performance, where the Transformer (Vaswani et al., 2017) encoder-decoder framework is the prevalent architecture. Autoregressive translation (AT) systems generate target tokens sequentially, i.e., each generation step depends on the previously generated tokens, resulting in high inference latency. Non-autoregressive translation (NAT) (Gu et al., 2018) models are proposed to speed up the inference of NMT, by generating all the target tokens independently and simultaneously. However, this independence assumption prevents NAT models to properly learn the target-side dependency in real data distribution. Therefore, while NAT models significantly reduce the inference latency, they suffer from accuracy degradation compared to AT models.

The mainstream NAT models fall into two categories, iterative NAT models and fully NAT models. Iterative NAT models (Gu et al., 2019; Ghazvininejad et al., 2019; Lee et al., 2018) employ an iterative procedure to refine translations, achieving improved translation accuracy at the expense of the decoding speed. In fact, iterative NAT models have similar latency and translation accuracy with AT models with deep encoder and shallow decoder (Kasai et al., 2020b). In contrast, fully NAT models maintain the latency advantage by making parallel predictions with a single decoding round, but produce outputs with poor translation accuracy. Due to their high efficiency, there has been increased research effort to improve the translation accuracy of fully NAT models while maintaining their latency advantage (Guo et al., 2019, 2020a,b; Saharia et al., 2020; Ma et al., 2019; Ran et al., 2020). Recently, Gu and Kong (2021) achieves competitive results for fully NAT models on three translation benchmarks. They argue that the key to successfully training a fully NAT model is to perform dependency reduction in the learning space of output tokens.

While easing the difficulty of generating output tokens may be practical, it limits the performance upper bound of fully NAT models. Models trained with these methods would struggle for generating complex sentences. Hence, different from Gu and Kong (2021) to perform dependency

\textsuperscript{1}We have released our code at \url{https://github.com/zja-nlp/NAT_with_DAD}.  
\textsuperscript{†}Corresponding Author

\textsuperscript{*}Contribution during internship at Machine Intelligence Technology Lab, Alibaba DAMO Academy.
Figure 1: The illustration of our proposed approach DAD. $S$ denotes the source text (in German). $P_{FB}$ denotes the prediction from the NAT model with both forward and backward dependency while $P_F$ denotes the prediction of the NAT model with only forward dependency. $\text{spaces}_S$ and $\text{spaces}_T$ denote the source representation space and the target representation space, respectively.

With the reduction in the learning space of output tokens, we focus on enhancing the learning capacity of fully NAT models by helping the model learn these complex dependencies among target tokens. Dependencies generally include forward dependencies and backward dependencies. Previous works explore bidirectional decoding to improve modeling of both forward and backward dependencies in phrase-based statistical machine translation (Finch and Sumita, 2009) and RNN-based machine translation (Zhang et al., 2018). We find that the less-investigated backward dependencies are critical for target generation in NAT. As shown in Figure 1, “woher” means both “where” and “how” in German. Without considering the backward dependency, the model would be confused of the selection of these two possible translations, leading to a multi-modality error (Ran et al., 2020). Multimodality is the main problem that NAT models suffer from, i.e., the target tokens may be generated based on different possible translations, often causing repeating or missing tokens. Hence, we propose approaches for fully NAT models to model both forward and backward target dependencies. On the other hand, decoder input is a crucial factor in target dependency modeling since the final output is directly conditioned on them. We hypothesize that accurate modeling of target dependencies may require the decoder input to be in the same representation space of the target output. Prior NAT models (Gu et al., 2018; Wang et al., 2019) mostly use a copy of source text embedding as decoder input, i.e., $z$ in Figure 2, which is target-independent. Various methods have been proposed to make the NAT decoder input similar to the target space. Qian et al. (2021), Guo et al. (2020a) and Guo et al. (2020b) choose a number of positions in $z$ and substitute them with the corresponding target embedding. Guo et al. (2019) uses linear mapping to make $z$ closer to target embedding. Unfortunately, these methods cannot guarantee to have the decoder input in the exact target space and still lead to a difference from real target distribution.

In this work, we design a novel Dependency-Aware Decoder (DAD) for fully NAT models so that the model can learn complex target-side dependencies. Firstly, we enhance the target dependency within the NAT decoder from the perspective of decoder input. The initial decoder input (see $z$ in Figure 1) is obtained by copying the source text embedding with uniform copy (Gu et al., 2018) or soft copy (Wei et al., 2019), which is target-independent. Various methods have been proposed to make the decoder input similar to the target space. Qian et al. (2021), Guo et al. (2020a) and Guo et al. (2020b) choose a number of positions in $z$ and substitute them with the corresponding target embedding. Guo et al. (2019) uses linear mapping to make $z$ closer to target embedding. Unfortunately, these methods cannot guarantee to have the decoder input in the exact target space and still lead to a difference from real target distribution.
or future information, learning the forward or backward dependencies respectively. We denote this method of forward-backward dependency modeling by FBD. We also investigate different curricula for optimizing initialization for fully NAT models (Section 4.3).

Our contributions can be summarized as follows: 1) we propose a novel dependency-aware decoder (DAD) for non-autoregressive translation models to model the target-side dependencies. The proposed approach is model-agnostic and can be applied to other NAT models. In this design, we propose a novel decoder input transformation approach and a novel approach for incorporating backward dependency modeling into the decoder of NAT models, through which the target-side dependencies can be better modeled. 2) Experimental results on multiple machine translation benchmarks demonstrate that our proposed approach significantly improves the representative vanilla NAT model (Gu et al., 2018) and the highly competitive fully NAT model GLAT (Qian et al., 2021) and DSLP (Huang et al., 2021), by up to +1.88 BLEU score on GLAT and +2.2 BLEU score on vanilla NAT, while adding only minor latency to these models and overall maintaining comparable inference latency to other fully NAT models.

2 Related Work

Due to their high efficiency, non-autoregressive models have been applied on a wide variety of tasks in addition to machine translation, including image captioning (Gao et al., 2019), text-to-speech synthesis (Oord et al., 2018), automatic speech recognition (Chen et al., 2019), etc. In this paper, we mainly focus on non-autoregressive translation, which has become a trending research topic recently (Gu et al., 2018; Wang et al., 2019; Qian et al., 2021; Gu and Kong, 2021; Huang et al., 2021; Savinov et al., 2021; Xie et al., 2021; Shao et al., 2021; Ding et al., 2021a,b; Bao et al., 2021). Although NAT models have inference efficiency advantage over autoregressive translation (AT) models, they suffer from poorer translation quality compared to AT. Among iterative NAT models, Lee et al. (2018) iteratively refines the target sentence. Ghazvininejad et al. (2019) iteratively masks words with the least probabilities and re-predicts them. Stern et al. (2019) and Gu et al. (2019) leverage edit-based methods to iteratively modify the decoder output. Ghazvininejad et al. (2020b) uses a semi-autoregressive training to improve the mask-predict decoding. Kasai et al. (2020a) iteratively refines every token in parallel and reduces the number of required iterations. These methods improve the translation accuracy at the expense of speed-up. Thus, the concern is how to build a fully NAT model that translates accurately.

To address the multi-modality problem (Section 1) for fully NAT models, Gu et al. (2018) uses knowledge distillation (Kim and Rush, 2016) to reduce the dataset complexity. Guo et al. (2019) proposes to enhance decoder input with phrase-table lookup and embedding mapping. Libovický and Helcl (2018) and Saharia et al. (2020) use connectionist temporal classification (CTC) (Graves et al., 2006) to conduct the latent alignment for NAT models. Sun et al. (2019) utilizes conditional random fields to model target-side positional contexts. Kaiser et al. (2018), Ma et al. (2019) and Shu et al. (2020) incorporate latent variables to guide generation, similar to standard VAEs (Kingma and Welling, 2013). Wei et al. (2019) and Guo et al. (2020a) use autoregressive models to guide the training of NAT models. Gu et al. (2020c) introduces pretrained language models to initialize the decoders of NAT models. Qian et al. (2021) proposes the Glancing Transformer (GLAT), which dynamically selects some positions based on the performance of the model and replaces them with corresponding target embeddings. While GLAT alleviates multi-modality by explicitly introducing target information, it causes mismatch between training and inference. Huang et al. (2021) introduces deep supervision and additional layer-wise prediction (DSLP) for each decoder layer, setting new SOTA for fully NAT models.

3 Proposed Methods

3.1 Problem Formulation

Neural machine translation can be formulated as a sequence-to-sequence generation problem. Given a sequence \( X = \{x_1, x_2, ..., x_N\} \) in the source language, a sequence \( Y = \{y_1, y_2, ..., y_T\} \) in the target language is generated following the conditional probability \( P(Y|X) \). Non-autoregressive translation (NAT) models are proposed to speedup generation by decoding all the target tokens in parallel, using conditional independent factorization as:

\[
P_{NAT}(Y|X) = P_L(T|x_{1:N}) \cdot \prod_{t=1}^{T} P(y_t|x_{1:N}, z) \quad (1)
\]
where the target sequence length $T$ is modeled by the conditional distribution $P_{T_\text{t}}$, and dependence on previous target tokens is removed. Compared to AT models, NAT models speed up inference significantly at the expense of translation quality, because the conditional independence assumption in Eqn.1 enables parallel processing but lacks explicit modeling of dependency between target tokens. To enhance target dependency modeling, we propose two innovations of transforming the decoder input into the target space and introducing the backward dependency modeling into the training process. We elaborate the two proposed methods in the next two sections.

### 3.2 Decoder Input Transformation for Target Dependency Modeling

We propose to directly select relevant representations from target embedding to form a new decoder input $z'$. Given the initial decoder input $z$ as a copy of source text embedding, $z$ is used as the query and the selection is realized by a learnable attention module. The learnable parameters bridge the gap between training and inference while the selection guarantees consistency between the decoder input space and the target embedding space. This way, the decoder input is closer to target-side embedding and more conducive to modeling target dependencies for NAT models. Additionally, to save computational cost, we propose a target-side embedding compression approach to reduce the size of the selection candidates. We hypothesize that this compression may also alleviate over-fitting on small datasets. We observe improved results on some small datasets from target-side embedding compression and detailed discussions are presented in Appendix C.1.

**Decoder Input Transformation** To transform $z$ into the target space, we apply the attention mechanism between $z$ and the output embedding matrix $Emb \in \mathbb{R}^{d \times v}$, where $d$ and $v$ refer to the size of hidden states and the target vocabulary, respectively. Since the target vocabulary are usually shared with the source vocabulary, first, we conduct a filtering process to remove the influence from source-side tokens. We select all the tokens appearing in target samples of the training set to build a dictionary that contains only target-side tokens. We then use this dictionary to filter $Emb$ and obtain the target space $Emb' \in \mathbb{R}^{d \times v'}$, where $v'$ denotes the size of filtered vocabulary. This filtering process guarantees that $Emb'$ is strictly from the target representation space. The attention process starts with a linear transformation:

$$z' = W_q \cdot z$$  \hspace{1cm} (2)

Then, the dot-product attention is performed on the result $z'$ (as query) and $Emb'$ (as key and value).

$$Sim = \text{softmax}(z' \cdot Emb')$$  \hspace{1cm} (3)

$Sim$ represents similarity between each $z'_l$ and each embedding in the target dictionary. Finally, we compute a weighted sum $z'$ of target embedding based on their similarity values:

$$z' = Sim \cdot Emb'^T$$  \hspace{1cm} (4)

Since $z'$ is a linear combination of $Emb'$ which is strictly in the target representation space, $z'$ is also strictly in the target space, hence using $z'$ as the decoder input provides a more solid basis for the target dependency modeling.

**Target-side Embedding Compression** Applying the attention mechanism to the whole target embedding is costly. Hence we propose to compress the large target embedding space to reduce the computational cost.

For target-side embedding compression, we process $Emb'$ through a linear layer to obtain a new target embedding $Emb^* \in \mathbb{R}^{d \times v'^*}$:

$$Emb^* = (W_c \cdot Emb'^T)^T$$  \hspace{1cm} (5)

where $W_c \in \mathbb{R}^{v' \times v'}$ is trainable and $v'^*$ can be set manually. The result $Emb^*$ is the combination of the original $Emb'$, which means it is still in the target space. Meanwhile, since we can manually set $v'^*$ as a relative small number (e.g., 1000, 2000, etc.), the computational cost of the attention mechanism can be greatly reduced.

### 3.3 Target Dependency Modeling with Attention Mask

In the decoder of AT models, each token can only attend to the previous tokens, which facilitates modeling forward dependency from left to right between target tokens. In the decoder of NAT models, each token can attend to the tokens in all positions and all the tokens are generated in parallel. As discussed in Section 3.1, the parallelism characteristics of NAT models is based on the independence assumption in Eqn. 1, which does not apply to real data distribution for machine translation. It leads
Figure 2: The proposed decoder input transformation from $z$ to $z'$. $z \in \mathbb{R}^{T \times d}$ denotes the initial decoder input copied from the source text embedding $x_{emb}$, where $T$ and $d$ denote the length of the target text $y$ and the size of hidden states, respectively. $Emb \in \mathbb{R}^{d \times v}$ denotes the output embedding matrix in decoder, where $v$ denotes the size of target vocabulary.

Figure 3: The proposed forward-backward dependency modeling with triangular attention masks. The red dashed lines indicate the attention masks. For simplicity, we only draw masks for the first three tokens. We use different colors to highlight the difference of inputs and targets in each phase.

to lack of target-side dependencies in NAT models and in turn worse performance compared to AT models.

Some previous methods (Guo et al., 2020a) utilize forward dependency in AT models to help NAT models explore target-side dependency. However, for NAT models, only modeling forward dependency is not sufficient (Finch and Sumita, 2009; Zhang et al., 2018), as also shown in the example in Figure 1. Consequently, we propose to introduce backward dependency modeling into NAT models. As shown in Figure 3, we explore curriculum learning to enhance the target-side dependency modeling of NAT models. In the first two phases, we model forward dependency and reverse dependency sequentially, and then transition to NAT training. The first two phases serve as pre-training for dependency modeling, through which we can initialize NAT models with better dependencies. Furthermore, we investigate efficacy of different curricula and find that if we add another phase of forward dependency modeling after the second phase of backward dependency modeling, further improvement can be achieved. Detailed discussions are presented in Section 4.3.

4 Experiments

4.1 Experimental Setup

Datasets We compare with previous works on two most popular machine translation benchmarks: WMT14 EN$\leftrightarrow$DE (4.5M pairs), WMT16 EN$\leftrightarrow$RO (610K pairs). Also, we use IWSLT16 DE-EN (196K pairs), IWSLT14 DE-EN (153K pairs), and
SP EN-JA\(^2\) (50K pairs) for further ablation analysis. For WMT16 EN-RO and IWSLT16 DE-EN, we use the processed data provided in (Lee et al., 2018). For WMT14 En-De, we apply the same preprocessing steps and learn subwords as in (Gu and Kong, 2021). For IWSLT14 DE-EN, we follow steps in (Guo et al., 2019). For SP EN-JA, we use sentencepiece\(^3\) to tokenize text into subword units following (Chousa et al., 2019). We share the source and target vocabulary and embeddings in each language pair except EN-JA. Following previous efforts (Gu et al., 2018; Lee et al., 2018; Wang et al., 2019; Qian et al., 2021), we use sequence-level knowledge distillation for all datasets except EN-JA. The NAT models in our experiments are trained on distilled data generated from pre-trained AT models following (Qian et al., 2021).

**Baselines and Setup** We apply our methods to vanilla NAT (Gu et al., 2018), GLAT (Qian et al., 2021), and DSLP (Huang et al., 2021), because GLAT and DSLP are both highly competitive fully NAT models while vanilla NAT is highly representative. As for the setting of the models, following Qian et al. (2021), we use base-Transformer (\(d_{model} = 512, n_{head} = 8, n_{layer} = 6\)) for the WMT datasets and small-Transformer (\(d_{model} = 256, n_{head} = 4, n_{layer} = 5\)) for the IWSLT and SP EN-JA datasets.

**Training** The training settings of vanilla NAT, GLAT, and DSLP are consistent with the original papers (Gu et al., 2018; Qian et al., 2021; Huang et al., 2021), which we present in Appendix A.1. For experiments using our method FBD (see Figure 3), we train the same number of steps at each phase, with 30K steps per phase for the WMT datasets and 10K steps per phase for the IWSLT datasets and SP EN-JA.

**Evaluation** We use BLEU (Papineni et al., 2002) to evaluate the translation quality for all models. To measure the inference latency, following (Gu and Kong, 2021), we measure the speed of each model by the wall-clock time for translating the whole WMT14 EN-DE test set on one GPU, and report the averaged time per sentence. We also use Distinct-N (Li et al., 2016), a measure of diversity, to measure repetition on the IWSLT16 DE-EN validation set.

### 4.2 Results

The main results on the WMT benchmarks are presented in Table 1. We report the performance from applying our method DAD to two representative NAT models and compare with AT and existing NAT models. Detailed results of the repetition evaluation are shown in Appendix B.

**Quality and Latency** We apply our proposed method (DAD), including IT and FBD, to vanilla NAT (Gu et al., 2018) and GLAT (Qian et al., 2021) on the WMT benchmarks and to vanilla NAT, GLAT, and DSLP (Huang et al., 2021) on the IWSLT benchmarks. For a fair comparison, we reproduce results from these models based on their open-source code and the results are denoted by \(^\dagger\) in Table 1. Our replication on GLAT performs consistently with the original paper while our replication of vanilla NAT outperforms the implementation of Gu et al. (2018), but matches the replication in Qian et al. (2021). Note that our replication of vanilla NAT performs slightly worse compared to Huang et al. (2021), probably due to the fact that Huang et al. (2021) uses different hyperparameters such as a larger batch size. As shown in Table 1, the proposed method DAD achieves steady and significant improvement on translation quality for vanilla NAT and GLAT on each benchmark, up to +1.88 BLEU score on GLAT and up to +2.2 BLEU score on vanilla NAT, effectively closing the gap on translation accuracy between the two baseline models and AT models/iterative NAT models. Nevertheless, our method still improves it by up to 0.51 BLEU score.

We also demonstrate the effectiveness of our proposed method DAD on the latest SOTA model for fully NAT models, which is the best-performing configuration in Huang et al. (2021), i.e., “DSLP w/ CTC + Mixed Training” in Table 1. For simplicity, we denote this model by DSLP. Due to limited computing resources, we only compare with DSLP w/o and w/ DAD on the relatively small IWSLT16 DE-EN task. As shown in Table 2, our method DAD (i.e., IT+FBD) also improves DSLP by +0.82 BLEU score and establishes new SOTA results on this task.

As to the inference latency, compared to the AT baseline model Transformer-base, vanilla NAT w/ DAD and GLAT w/ DAD achieve 15.4× and
We analyze their respective effect on the IWSLT16 with other fully NAT models.

### Table 1: Performance comparison between our proposed method DAD and existing methods.

| Models | Iter. | Speed-up | WMT'14 EN-DE | WMT'16 EN-RO | RO-EN |
|--------|-------|-----------|---------------|---------------|-------|
| AT     | Transformer base (teacher) + KD | N | 1.0× | 27.48 | 31.39 | 33.70 | 34.05 |
|        | SNAT (Lee et al., 2018) | 10 | 1.5× | 21.61 | 25.48 | 29.32 | 30.19 |
|        | CMLM (Ghazvininejad et al., 2019)* | 10 | 1.7× | 27.03 | 30.53 | 33.08 | 33.31 |
|        | LevT (Gu et al., 2019) Adv | 4.0× | 27.27 | - | - | - | - |
|        | JM-NAT (Guo et al., 2020b) | 10 | 1.8× | 27.69 | 32.24 | 33.52 | 33.72 |
|        | DiCO (Kashi et al., 2020a)* Adv | 3.5× | 27.34 | 31.31 | 33.22 | 33.25 |
|        | Multi-Task NAT (Hao et al., 2021) | 10 | 1.7× | 27.98 | 31.27 | 33.80 | 33.60 |
|        | RewrittenAT (Geng et al., 2021) Adv | 8 | 3.9× | 27.83 | 31.52 | 33.63 | 34.09 |
|        | Imputer (Saharia et al., 2020)* | 28.20 | 31.80 | 34.40 | 34.10 |
| Iterative NAT | Vanilla NAT (Gu et al., 2018) | 1 | 15.6× | 17.69 | 21.47 | 27.29 | 29.06 |
|        | CTC (Libovick`y and Helcl, 2018) | - | 16.5× | 18.64 | 19.54 | 24.67 |
|        | FCL-DT (Guo et al., 2020a) | - | 20.23 | 23.16 | - | - |
|        | FCL (Guo et al., 2020a) | - | 21.70 | 25.32 | - | - |
|        | DCRF (Sun et al., 2019) | 1 | 10.4× | 23.44 | 27.22 | - | - |
|        | Flowseq (Ma et al., 2019) | 1 | 1.1× | 23.72 | 28.39 | 29.73 | 30.72 |
|        | ReorderNAT (Ren et al., 2020) | 1 | 16.1× | 22.79 | 27.28 | 29.30 | 29.50 |
|        | AXE (Ghazvininejad et al., 2020a)* | 1 | 15.3× | 23.53 | 27.90 | 30.75 | 31.54 |
|        | EM+ODD (Sun and Yang, 2020) | 1 | 16.4× | 24.54 | 27.93 | - | - |
|        | SNAT (Lu et al., 2021) | 1 | 22.6× | 24.64 | 28.42 | 32.87 | 32.21 |
|        | Imputer (Saharia et al., 2020)* | 1 | 18.6× | 25.80 | 28.40 | 32.30 | 31.70 |
|        | AlignNART (Song et al., 2021) | 1 | 13.4× | 26.40 | 30.40 | 32.50 | 33.10 |
|        | OAHE (Du et al., 2021) | 1 | 15.3× | 26.10 | 30.20 | 32.40 | 33.30 |
|        | F-CTC (Gu and Kong, 2021) | 1 | 16.8× | 26.51 | 30.46 | 33.41 | 34.07 |
| Fully NAT | Vanilla NAT (Huang et al., 2021) w/ DSLP | 1 | 15.7× | 21.18 | 24.93 | 29.15 | 29.69 |
|        | Vanilla NAT (Qian et al., 2021) w/ DAD (ours) | 1 | 15.6× | 20.36 | 24.81 | 28.47 | 29.43 |
|        | GLAT (Qian et al., 2021)† w/ DSLP | 1 | 15.4× | 23.15 | 26.59 | 30.78 | 31.89 |
|        | GLAT (Qian et al., 2021)† w/ DSLP w/ CTC + Mixed Training (Huang et al., 2021) w/ DAD (ours) | 1 | 14.8× | 27.02 | 31.61 | 34.17 | 34.60 |
|        | DAD (ours) | 1 | 14.7× | 27.51 | 31.96 | 34.68 | 34.98 |

Table 1: Performance comparison between our proposed method DAD and existing methods. All results reported standalone are without re-scoring. Iter. denotes the number of iterations at the inference time. Adv. denotes adaptive. * denotes models trained with distillation from a big Transformer. † means the results are obtained by our implementation. The speed-up is measured on the WMT14 EN-DE test set.

15.1× speed-up, respectively, which is comparable with other fully NAT models.

**Raw Data** To evaluate how much the proposed approach DAD can reduce dependence of training NAT models on AT models, we compare the performance of GLAT w/ and w/o DAD on raw data. The raw data refers to the original IWSLT16 DE-EN dataset without knowledge distillation (Section 4.1). The results are shown in Table 2. With our method DAD, GLAT is significantly improved on raw data with +1.57 BLEU score gain, demonstrating the robustness of our method. It can be concluded that our method effectively enhances the dependency modeling ability of NAT models and make them more independent of AT models.

### 4.3 Ablation Study

**Impact of IT and FBD** Our method DAD includes decoder input transformation (IT) and forward-backward dependency modeling (FBD). We analyze their respective effect on the IWSLT16 DE-EN validation set and the test sets of IWSLT14 DE-EN and SP EN-JA and report results in Table 2. Applying IT without compression (Sec-
Figure 4: Comparison on the similarity of the target embedding with the original decoder input and transformed decoder input. Experiments are conducted on the validation set of IWSLT16 DE-EN.

| Models   | BLEU   | Models   | BLEU   |
|----------|--------|----------|--------|
| NAT      | 29.61  | BF-NAT   | 27.83  |
| F-NAT    | 30.04  | FB-NAT   | 30.87  |
| B-NAT    | 27.05  | FBF-NAT  | 31.19  |

| SP EN-JA |
|----------|
| NAT      | 27.67  |
| F-NAT    | 28.15  |
| B-NAT    | 25.83  |

Table 3: Performances of different curricula of dependency modeling. \textbf{F} and \textbf{B} denote forward dependency and backward dependency, respectively.

Impact of Different Dependency Curricula

For the forward-backward dependency modeling, we investigate a variety of curricula based on GLAT (Qian et al., 2021) on the IWSLT16 DE-EN task to determine the order of the two modeling phases. The results are shown in Table 3. Compared with the models with backward dependency modeling as the first phase (see B-NAT and BF-NAT in Table 3), the models with forward dependency modeling as the first phase perform better (see F-NAT, FB-NAT, and FBF-NAT). It seems that the forward dependency modeling achieves a good initialization for the following model training, while the backward dependency modeling cannot. Considering the nature of languages, learning the forward dependency first is easier for the model on the language generation task. We hypothesize that the backward dependency needs to be modeled in the second phase, based on some forward dependency knowledge. We observe the best combination as FBF-NAT, i.e., first learn forward dependency and next model backward dependency, then add another round of forward dependency modeling before NAT training. We speculate that NAT training benefits from the second forward dependency modeling because the order of left-to-right is more consistent with the characteristics of natural languages as discussed above, smoothing the transition to the NAT training phase. As shown in Table 3, we observe the same trend of curriculum on SP EN-JA dataset as on IWSLT16 DE-EN, with FBF-NAT performing best, although the target language Japanese is almost fully left-branching.

4.4 Visualization of Self-Attention Distribution

For a deeper and more intuitive exploration of dependency modeling in NAT models, we present visualization of the self-attention distribution of different NAT models in Figure 5. All the models are based on GLAT and the meanings of model names are the same as in Table 3. In the decoder of NAT model (see Figure 5a), self-attention distribution of each position is scattered in adjacent positions, indicating that NAT model lacks dependence and has much confusion during decoding, which causes local repetitions and multi-modal errors. In F-NAT and B-NAT models, significant forward and backward dependencies can be observed in Figure 5b and Figure 5c, respectively, indicating that these
**Table 4: Case studies of our proposed method FBD on the IWSLT16 DE-EN task.** The row for the bold-faced FB-NAT shows predictions from our method.

| Source | Target | F-NAT | FB-NAT |
|--------|--------|-------|--------|
| Wir haben es weltweit in 300 Gemeinden gemacht. | Einige Leute wollten ihn einfach König nennen. | Some people just wanted to call him King. | Where am I from? Who am I? |
| **F-NAT** | We did it in 300 communities. | Some people just wanted to call him king. | How do I come from? Who am I? |
| **FB-NAT** | We’ve done it in 300 communities around the world. | Some people just wanted to call him king. | Where do I come from? Who am I? |

Figure 5: Visualization of the decoder self-attention distribution in different NAT models. The meanings of these model names are the same as in Table 3.

4.5 Case Study

We conduct case studies on the IWSLT16 DE-EN task to intuitively compare the performance of our methods and baselines. Table 4 shows example translations of “FB-NAT” as GLAT w/ the proposed FBD, and “F-NAT” as GLAT with only forward dependency modeling. We can see that some typical errors in the predictions of the F-NAT model can be corrected by introducing backward dependency through our method. For the instance of multi-modality error, “woher” means both “where” and “how” in German. F-NAT incorrectly translates “Woher” into “How”; whereas With the backward dependency, FB-NAT translates “Woher” into “Where” depending on the future information “come from”. As to the under-translation error, FB-NAT translates “weltweit” to “around the world” (see the red part in Table 4), while F-NAT does not. Regarding the over-translation error, F-NAT improperly generates repetition of “him” (see the blue part in Table 4), while FB-NAT does not. More case studies are presented in Appendix D.

5 Conclusion

We propose to enhance target dependency modeling within the decoder of NAT models from two perspectives: decoder input transformation and backward dependency modeling. Experiments on five machine translation benchmarks demonstrate that the proposed methods can effectively improve the translation accuracy of highly competitive fully NAT models with little effect on latency. In future work, we will investigate the efficacy of the proposed methods on iterative NAT models and whether our methods can benefit from re-ranking approaches such as NPD (Gu et al., 2018).

**References**

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.

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.

Katsuki Chousa, Katsuhito Sudoh, and Satoshi Nakamura. 2019. Simultaneous neural machine translation using connectionist temporal classification. arXiv preprint arXiv:1911.11933.

Liang Ding, Longyue Wang, Xuebo Liu, Derek F Wong, Dacheng Tao, and Zhaopeng Tu. 2021a.
Progressive multi-granularity training for non-autoregressive translation. "arXiv preprint arXiv:2106.05546."

Liang Ding, Longyue Wang, Xuebo Liu, Derek F Wong, Dacheng Tao, and Zhaopeng Tu. 2021b. Rejuvenating low-frequency words: Making the most of parallel data in non-autoregressive translation. "arXiv preprint arXiv:2106.00903."

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. "arXiv preprint arXiv:2106.05093."

Andrew M. Finch and Eiichiro Sumita. 2009. Bidirectional phrase-based statistical machine translation. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL, pages 1124–1132. ACL.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 3723–3730.

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.

Junliang Guo, Zhirui Zhang, Linli Xu, Hao-Ran Wei, Boxing Chen, and Enhong Chen. 2020c. Incorporating bert into parallel sequence decoding with adapters. In NeurIPS.

Yongchang Hao, Shilin He, Wenxiang Jiao, Zhaopeng Tu, Michael Lyu, and Xing Wang. 2021. Multi-task learning with shared encoder for non-autoregressive machine translation. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3989–3996.

Chenyang Huang, Hao Zhou, Osmar R Zaíane, Lili Mou, and Lei Li. 2021. Non-autoregressive translation with layer-wise prediction and deep supervision. "arXiv preprint arXiv:2110.07515."

Lukasz Kaiser, Szymon Bengio, Arunko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In International Conference on Machine Learning, pages 2390–2399. PMLR.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020a. Non-autoregressive machine translation with disentangled context transformer. In International Conference on Machine Learning, pages 5144–5155. PMLR.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2020b. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In International Conference on Learning Representations.
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.

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

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *HLT-NAACL*.

Jindrich Libovicky 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*, pages 3016–3021.

Ye Liu, Yao Wan, Jianguo Zhang, Wenting Zhao, and S Yu Philip. 2021. Enriching non-autoregressive transformer with syntactic and semantic structures for neural machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1235–1244.

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.

Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. 2018. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR.

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.

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 (Volume 1: Long Papers)*, pages 1993–2003, Online. Association for Computational Linguistics.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2020. Learning to recover from multi-modality errors for non-autoregressive neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3059–3069.

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)*, pages 1098–1108.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. 2021. Step-unrolled denoising autoencoders for text generation. *CoRR*, abs/2112.06749.

Chenze Shao, Yang Feng, Jinchao Zhang, Fandong Meng, and Jie Zhou. 2021. Sequence-level training for non-autoregressive neural machine translation. *arXiv preprint arXiv:2106.08122*.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8846–8853.

Jongyoon Song, Sungwon Kim, and Sungroh Yoon. 2021. 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.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985. PMLR.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhi-Hong Deng. 2019. Fast structured decoding for sequence models. In *NeurIPS*.

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, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
A Experimental Details

A.1 Experimental Setup

Using Nvidia V100 GPUs, we train models with batches of 64K tokens for WMT datasets, and 8K tokens for IWSLT and SP EN-JA datasets. For three baseline models (Gu et al., 2018; Qian et al., 2021; Huang et al., 2021), we keep the same setup with them. For GLAT, we use Adam optimizer (Kingma and Ba, 2015) with $\beta = (0.9, 0.999)$ and set the dropout rate to 0.1. For Vanilla NAT and DSLP, we use Adam optimizer (Kingma and Ba, 2015) with $\beta = (0.9, 0.98)$. For WMT datasets, the learning rate warms up to $5e^{-4}$ in 4k steps and gradually decays according to inverse square root schedule n Vaswani et al. (2017). As for IWSLT and SP EN-JA datasets, we adopt linear annealing (from $3e^{-4}$ to $1e^{-5}$ ) as in Lee et al. (2018). We choose the model with the best performance on the validation set as the final model.

B Evaluation of Token Repetition

The phenomenon of generating repetitive words in translation is a major drawback of NAT models. We hypothesize that enhanced target-side dependency modeling may enforce linguistic constraints and reduce repetition. We use Distinct-N (Li et al., 2016), the ratio of distinct n-gram over the whole validation set, to evaluate repetition on the IWSLT16 DE-EN validation set. Table 5 presents the Distinct-N scores of sentences generated by vanilla NAT and GLAT w/o and w/ our method DAD. A higher score indicates less repetition. The results show that our method effectively reduces repetition. We hypothesize that reduction in repetition is due to the deeper dependencies modeled by DAD. Since our method improve the ability of the decoders to model target-side dependencies, the repetition problem can be largely alleviated.

| Models  | Distinct-N 1 | Distinct-N 2 | Distinct-N 3 |
|---------|--------------|--------------|--------------|
| Vanilla NAT | 0.8879 | 0.9829 | 0.9916 |
| w/ DAD   | **0.9059** | **0.9897** | **0.9964** |
| GLAT     | 0.8942 | 0.9912 | 0.9923 |
| w/ DAD   | **0.9298** | **0.9943** | **0.9958** |

Table 5: Evaluation of token repetition on IWSLT16 DE-EN.
C More Ablation Study

C.1 Effectiveness of the Compression of Target Space

As discussed in Section 3.2, considering the computational cost, we propose to adopt a linear compression module to reduce the selection candidates of the target embedding. We use the dichotomy to determine the compression dimension interval [1000, 2000], then we experiment with per 200 dimensions in this interval. We present the corresponding results on IWSLT16 DE-EN dataset in Table 6. With compression, the performance can be improved by up to 0.78 BLEU score. We also experiment with bigger model on WMT16 EN-RO, but the compression has no effect. We assume that for relatively small model and data, the compression of the target space can help filter some redundant target information, play a refining effect, and improve the performance of the model.

| Compressed Dimension | BLEU  |
|----------------------|-------|
| w/o                  | 29.61 |
| 1000                 | 29.45 |
| 1200                 | 29.56 |
| 1400                 | 29.77 |
| 1600                 | 29.85 |
| 1800                 | 30.39 |
| 2000                 | 29.14 |

Table 6: Performances of GLAT+DAD on IWSLT16 DE-EN dataset with compression of target space.

D More Case Study

Analysis on method IT NAT models generally suffer from three main problems, over translation (repetition), under translation (missing information), and multi-modality (incorrect translations caused by polysemy). As shown in Table 7, vanilla NAT and GLAT tend to generate repetitive tokens which are highlighted in red. Additionally, vanilla NAT omit the translation of “schließlich”. By applying our method IT, the decoder input is closer to the target space and the model have a better perception for target-side information, so that the repetition and under-translation problems can be effectively alleviated. As for multi-modality error, see the first example in Table 7, “Drucks” means both “printing” and “pressure” in German. GLAT mistakenly translates “Drucks” into “printing”, but our method can help the model correctly translate it into “pressure”. Particularly, in the second example, “Bauplan” means “blueprint” in German. Although both baseline models generate the correct words, they generate the redundant word “plan” which is also a subword of “Bauplan”. It illustrates that the baseline approaches may confuse the source space and target space during generation, but our method avoids this.
| Case #1 | Case #2 |
|---------|---------|
| **Source** | obwohl sie erwischt wurden, wurden sie schließlich freigelassen aufgrund immensen internationalen Drucks. | das ist ein Bauplan für Länder wie China und den Iran. |
| **Target** | even though they were caught, they were eventually released after heavy international pressure. | this is a blueprint for countries like China and Iran. |
| Vanilla NAT | although they were caught, they were released released because because of huge drug. | this is a blueprint plan for countries like China and Iran. |
| Vanilla NAT w/ IT | although they were caught, they were finally released because huge international pressure. | this is a blueprint for countries like China and Iran. |
| GLAT | although they were caught, they finally were released because of international printing. | this is a blueprint plan for countries like China and Iran. |
| GLAT w/ IT | although they were caught, they were finally released after huge international pressure. | this is a blueprint for countries like China and Iran. |

Table 7: Case studies of our method IT on IWSLT16 DE-EN task. The bold part is our method.