Maximum Proxy-Likelihood Estimation for Non-autoregressive Machine Translation

Anonymous ACL submission

Abstract

Maximum Likelihood Estimation (MLE) is commonly used in machine translation, where models with higher likelihood are assumed to perform better in translation. However, this assumption does not hold in the non-autoregressive Transformers (NATs), a new family of translation models. In this paper, we present both theoretical and empirical analysis on why simply maximizing the likelihood does not produce a good NAT model. Based on the theoretical analysis, we propose Maximum Proxy-Likelihood Estimation (MPLE), a novel method to address the training issue in MLE. Additionally, MPLE provides a novel perspective to understand existing success in training NATs, namely much previous work can be regarded as implicitly optimizing our objective.

1 Introduction

Maximum Likelihood Estimation (MLE) is a widely-used method in machine translation models. The objective of MLE is to maximize the likelihood $P(Y|X)$, where X and Y are input and output sentences respectively. Recently, Non-Autoregressive Transformer (NAT, Gu et al., 2018) has received growing attention due to its efficiency of parallel decoding. MLE are also adopted to train the NAT model, but the MLE-based NATs suffer from poor translation quality compared with the classical autoregressive translation (AT) models.

To remedy the performance gap between ATs and NATs, many training methods have been proposed. For example, knowledge distillation (KD) (Gu et al., 2018) supervises NATs with sentences distilled from an AT teacher model. GLAT (Qian et al., 2021) helps the training by sampling some target tokens as the decoder input. These methods only change the training objectives without modifying the model structure, but also demonstrate significant improvements in translation quality.

However, there is major departure from the nature of MLE in these methods, where a NAT with low likelihood can even perform better in translation. As shown in Fig.1, we finetune two NATs with different objectives from an initial checkpoint and track the changes in the log-likelihood and the BLEU score. The optimal training directions under the two metrics are inconsistent, and the MLE training just misleads the NAT towards a sub-optimal point with a low BLEU score despite the high log-likelihood. The inconsistency between the likelihood and the generation quality has been found in AT models (Ranzato et al., 2016), but the problem in NATs can be much more severe (e.g., generating unreadable and repetitive outputs, Gu et al., 2018) and lacks theoretical investigation for the causes.

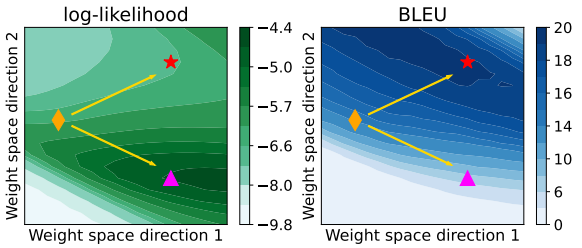

Figure 1: The log-likelihood and the BLEU score on a 2D section in NAT's weight space. GLAT+KD (★) and MLE (▲) indicate the models trained via different objectives starting from an initial checkpoint (♦) for 10k steps. Each point of the contour map is linearly interpolated from the three checkpoints. All values are evaluated on the validation set of WMT14 En-De.

In this paper, we argue that the MLE objective can severely mislead NATs' training due to the information loss in dependencies according to our theoretical analysis. Specifically, we show that the MLE training prevents NATs from learning the dependencies between target tokens, where the lost information can be measured by a property of the data distribution, namely, the *total correlation*.

To address the above issue, we propose a novel method, *Maximum Proxy-Likelihood Estimation* (MPLE), and show that training NATs with MPLE leads to less information loss theoretically and empirically. Intuitively, MPLE maximizes the likeli-

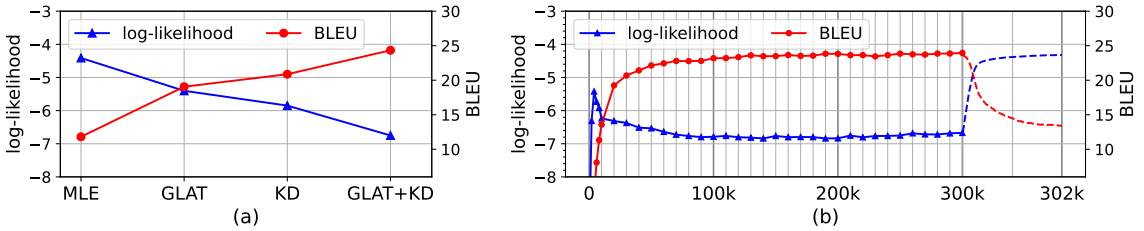

Figure 2: The log-likelihood and BLEU of NATs on the original validation set of WMT14 En-De, where higher log-likelihood does not lead to better translation performance. (a) NATs with different objectives. (b) Curves of a NAT trained with GLAT+KD (before 300k steps) and then finetuned with MLE (after 300k steps).

hood on a proxy distribution Q, which is carefully designed to reduce the total correlation, thereby alleviate the information loss in capturing the dependencies. Additionally, we show that many existing methods implicitly optimize MPLE's objective by using heuristic rules to obtain proxy variables, which provides a general perspective and sheds a light on deeper understanding of these methods, including KD (Gu et al., 2018), AXE (Ghazvininejad et al., 2020), OaXE (Du et al., 2021), CMLM (Ghazvininejad et al., 2019), and GLAT (Qian et al., 2021). Finally, we derive a new method named Dynamic KD, which optimizes MPLE's objective explicitly instead of using heuristic rules. Experiments show that our objective is highly correlated with the translation performance, and the proposed method achieves substantial improvements over the baselines. Our contributions are as follows:

- We present empirical and theoretical analysis showing that MLE misleads the NAT training due to the information loss in dependencies, which can be measured by the *total correlation*.
- We propose a novel method, Maximum Proxy-Likelihood Estimation (MPLE), which effectively reduces the total correlation theoretically and provides a novel perspective to understand the success of many existing methods.
- Based on MPLE, we propose a new method named Dynamic KD, which achieves substantial improvements over the baselines.

2 MLE Training Misleads NAT Models

MLE is in fact finding a model with the closest distribution to the data distribution in terms of KL divergence (Akaike, 1998). Given a source sentence $X = [x_1, x_2, \cdots, x_N]$ and a target sentence $Y = [y_1, y_2, \cdots, y_M]$, MLE training minimizes

$$\begin{aligned}\mathcal{L}_{\text{MLE}} &= \mathcal{D}_{\text{KL}}\left[P_{\text{data}}(Y|X)||P_\theta(Y|X)\right] \\ &= -H_{\text{data}}(Y|X) - \mathbb{E}_{P_{\text{data}}(Y|X)}\left[\log P_\theta(Y|X)\right],\end{aligned} \tag{1}$$

where H_{data} is a constant representing the dataset's Shannon Entropy, and the second term is the log-likelihood (LL). For AT models, LL is defined as

$$\log P^{\text{AT}}_\theta(Y|X) = \sum_{i=1}^{M} \log P^{\text{AT}}_\theta(y_i|y_{<i}, X), \tag{2}$$

where y_i is predicted based on the prefix $y_{<i}$.

The vanilla NAT makes a *conditional independent assumption* where each token is independent of each other when X is given. Formally, we have

$$\log P^{\text{NAT}}_\theta(Y|X) = \sum_{i=1}^{M} \log P^{\text{NAT}}_\theta(y_i|X). \tag{3}$$

In AT training, we usually assume that training the model towards a higher LL can lead to better translation quality. However, in this section, we show that the assumption does not apply to NAT models empirically and theoretically.

2.1 Practically Higher Likelihood does not Lead to Better Translation Performance

By comparing NATs trained with different methods, we find that a higher likelihood in NATs does not lead to better translation performance. Specifically, we adopt two SoTA methods, KD (Gu et al., 2018) and GLAT (Qian et al., 2021), and evaluate BLEU and LL defined on the validation set of WMT14 En-De. More settings are presented in Sec.4.

Fig.2(a) shows that the NAT model with higher LL even performs worse in BLEU. Fig.2(b) illustrates the training curves of GLAT+KD, where LL quickly drops after 4k steps despite the improvement on BLEU. At 300k steps, we finetune the model with MLE and find that the translation quality drops quickly regardless of LL's improvement.

All these results indicate a serious problem that the MLE training does not help but even misleads the NAT models. We will show that the problem is from the MLE objective when applied to NATs.

2.2 Training NAT with MLE is Bounded by *Conditional Total Correlation*

The KL divergence in Eq.1 can reach zero for ATs when $P_\theta(y_i|y_{<i}, X) = P_{\text{data}}(y_i|y_{<i}, X)$. However,

	C	ΔBLEU	BLEU_{AT}	BLEU_{NAT}
WMT14 En-De	2.50	15.32	27.11	11.79
WMT16 EN-RO	2.20	9.98	33.70	23.72
Denoise(0.5,0.1)	1.51	5.66	20.97	15.31
Denoise(0.5,0.0)	0.92	0.35	26.96	26.61

Table 1: Estimated total correlation C and the gap of BLEU between AT and NAT on various datasets. [1] $\Delta\text{BLEU} = \text{BLEU}_{\text{AT}} - \text{BLEU}_{\text{NAT}}$. We adopt WMT14 En-De, WMT16 EN-RO, and two synthetic datasets. Denoise(0.5, 0.1) indicates the source sentence is a corrupted version of the target sentence with 50% tokens replaced by random words and 10% tokens dropped.

a NAT model can hardly fit the data distribution, where we show that the minimum of the KL divergence is bounded by an non-negative constant.

Theorem 1. *For a NAT model $P_\theta(Y|X)$, we have $\min_\theta \mathcal{D}_{KL}[P_{data}(Y|X)||P_\theta(Y|X)] \geq C$, where $C = \sum_{i=1}^{M} H_{data}(y_i|X) - H_{data}(Y|X)$, and $H_{data}(\cdot|X)$ is the Shannon Entropy.*

Proof.

$$\begin{aligned}
&\mathcal{D}_{\text{KL}}[P_{\text{data}}(Y|X)||P_\theta(Y|X)] \\
&= -H_{\text{data}}(Y|X) - \mathbb{E}_{P_{\text{data}}(Y|X)}\left[\sum_{i=1}^{M} \log P_\theta(y_i|X)\right] \\
&\qquad \textit{(Conditional Independent Assumption of Eq.3)} \\
&= -H_{\text{data}}(Y|X) - \sum_{i=1}^{M} \mathbb{E}_{P_{\text{data}}(y_i|X)}\left[\log P_\theta(y_i|X)\right] \\
&\geq -H_{\text{data}}(Y|X) + \sum_{i=1}^{M} H_{\text{data}}(y_i|X) \quad \textit{(Gibbs' Inequality)}
\end{aligned}$$

Note that C is a non-negative constant called *conditional total correlation* (Watanabe, 1960) or *multi-information* (Studený and Vejnarová, 1998), which measures the information of dependencies between the target tokens when X is known. Theorem 1 says that MLE-based NATs cannot capture the dependencies between target tokens, leading to low consistency in generated tokens. The total correlation C is a property of the data distribution, where a large C indicate the large amount of information dropped by the NAT model and thus severely damage the translation performance.

To better understand the severity of the problem, we estimate the total correlation C and compare the generation performance of AT and NAT models on different datasets. We utilize two benchmarks and further construct two synthetic datasets, whose target sentences are English corpus in WMT14 En-De, and the source sentences are modified from the targets by word replacement and dropping.

The results are shown in Table 1. Larger C indicates stronger dependencies between target tokens, leading to more serious performance gap between NAT and AT models. When C becomes smaller, the gap can be quickly narrowed, which manifests that the large total correlation is the main obstacle of MLE-based NATs in machine translation.

3 Training NAT Models by Maximum Proxy-Likelihood Estimation

Sec.2 describes that the conditional total correlation prevents NAT from well training with MLE. To address the problem, we introduce a novel objective, Proxy-Likelihood (PL), and propose to maximize PL instead of vanilla MLE. We call it Maximum Proxy-Likelihood Estimation (MPLE). Specifically, PL is the likelihood defined on a proxy distribution Q, where Q is carefully designed (and adjusted) to reach a lower total correlation, and thus reduce the dependencies loss in NATs. Intuitively, MPLE's objective can be expressed as follows:

$$\mathcal{L} = \mathcal{D}_{\text{KL}}(Q||P_\theta) + \mathcal{R}(Q, P_{\text{data}}). \tag{4}$$

The first term is similar to the MLE objective, which trains the model towards the proxy distribution Q instead of P_{data}. The second term is a regularizer controlling the gap between Q and P_{data}.

Equation 4 is just an intuitive introduction to our objective. In the following sections, we will describe it in details. Sec.3.1 will present our design philosophy about the proxy distribution Q and describe why it works for NAT training. Sec.3.2 will formalize how to derive the full MPLE objective from the original MLE one. Then in Sec.3.3, we will theoretically verify that MPLE indeed leads to a lower total correlation. Finally, we will show that many current progresses in NAT training can be understood in the MPLE framework (Sec.3.4), and propose a new knowledge distillation approach for NAT based on the MPLE framework (Sec.3.5).

3.1 Proxy Distribution $Q(T|Z, X)$

In this section, we specify the proposed proxy distribution $Q(T|Z, X)$, which is used to train the NAT decoder $P_\theta(T|Z, X)$ in Fig.3 (Left). Compared with $P_{\text{data}}(Y|X)$, the proxy distribution $Q(T|Z, X)$ reduces the conditional total correlation by introducing two proxy variables, namely the proxy input Z and the proxy target T.

Proxy Target T The first way to reduce the conditional total correlation is by replacing the original

[1] We use *V-entropy* (Xu et al., 2020) instead of the Shannon entropy because the latter is intractable due to the unknown data distribution. Specifically, the estimation of C is based on $P_{\text{data}}(y_i|X)$ and $P_{\text{data}}(y_i|y_{<i}, X)$, which are separately approximated by a NAT and an AT model.

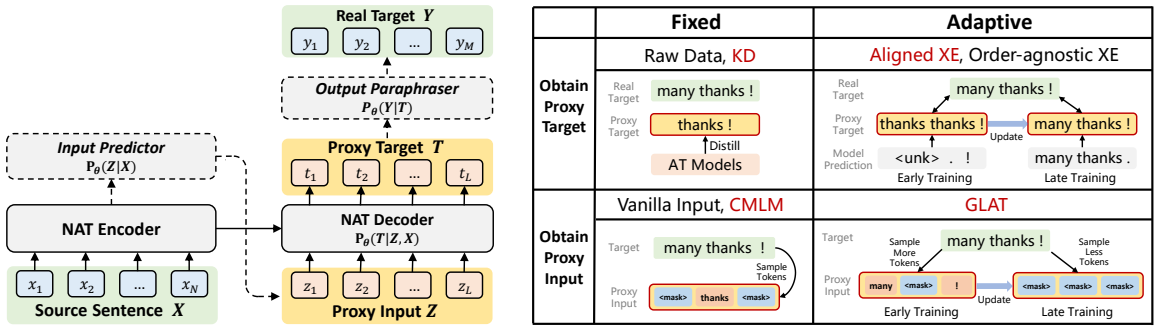

Figure 3: **Left:** The latent variable model used in the derivation of MPLE. The source sentence X and the real target Y are observable, whereas the proxy input Z and the proxy target T are latent. **Right:** Existing methods as instantiations of MPLE, which obtain either proxy target or proxy input. Methods in red are illustrated as examples. Raw Data, KD (Gu et al., 2018), Vanilla Input (Gu et al., 2018), and CMLM (Ghazvininejad et al., 2019) use **fixed** proxy variables.[2] AXE (Ghazvininejad et al., 2020), OaXE (Du et al., 2021), and GLAT (Qian et al., 2021) use **adaptive** proxy variables that are updated through the training process.

target with a simpler sentence with fewer dependencies. Specifically, we define the proxy target as a sentence $T = [t_1, t_2, \cdots, t_L]$, with the same meaning as the target Y but different lexical forms.

Intuitively, human-written translations are diverse in word usages and phrases' order (Ott et al., 2018). A model is required to capture the dependencies to form plausible sentences. In contrast, the proxy targets are expected to be easily predictable, e.g., aligned well with the source sentences.

Proxy Input Z Another way to reduce the conditional total correlation is by introducing some hints of the target into the input. Specifically, we define the proxy input as a sequence $Z = [z_1, z_2, \cdots, z_L]$. We use Z as the decoder's input and keep the encoder's input unchanged as X.

Intuitively, some hints of the target can make the prediction easier and no longer require the dependencies between generated tokens. For example, if we sample some target tokens as the proxy input, the target sentence may be easily reconstructed without knowing the tokens not sampled. However, introducing too many hints may bring large gaps between training and inference, where we actually update Z (as well as T) in MPLE.

3.2 Derivation of MPLE

Given the proxy distribution $Q(T|Z, X)$, now the unresolved problem lies in that how can we achieve the proxy-likelihood maximizing when we only have annotated data $\langle X, Y \rangle$. A straightforward solution is to regard T and Z as latent variables, so we build a latent variable model that connects T, Z and X, Y, as shown in Fig.3 (Left). Formally,

$$P_\theta(Y|X) = \sum_Z \sum_T P_\theta(Y|T) P_\theta(T|Z, X) P_\theta(Z|X),$$

where $P_\theta(T|Z, X)$ is the NAT decoder, and the other two modules bridge the proxy variables with X, Y. Then we derive the objective of MPLE from the log-likelihood on $P_{\text{data}}(Y|X)$:

$$\begin{aligned}
& -\mathbb{E}_{P_{\text{data}}(Y|X)} \log P_\theta(Y|X) \\
&= -\mathbb{E}_{P_{\text{data}}(Y|X)} \log \left[\mathbb{E}_{Q(T,Z|X)} \frac{P_\theta(Y, T, Z|X)}{Q(T, Z|X)} \right] \\
&\le -\mathbb{E}_{P_{\text{data}}(Y|X)} \mathbb{E}_{Q(T,Z|X)} \left[\log \frac{P_\theta(Y, T, Z|X)}{Q(T, Z|X)} \right] \quad (5) \\
&= -\mathbb{E}_{P_{\text{data}}(Y|X)} \mathbb{E}_{Q(T,Z|X)} \Big[\log P_\theta(Y|T) + \\
&\qquad \log \frac{P_\theta(T|Z, X)}{Q(T|Z, X)} + \log \frac{P_\theta(Z|X)}{Q(Z|X)} \Big] \quad (6)
\end{aligned}$$

In Eq.5, we apply *variational principle* (Fox and Roberts, 2012) by introducing $Q(T, Z|X)$, which specifies how we obtain proxy variables and can be decomposed into the proxy distribution $Q(T|Z, X)$ and another distribution $Q(Z|X)$.

Eq.6 is our new objective $\mathcal{L}_{\text{MPLE}}$, which can be simplified and recovers our intuition in Eq.4:

$$\mathcal{L}_{\text{MPLE}} = \underbrace{\mathcal{L}_{\text{NAT}}}_{\mathcal{D}_{\text{KL}}(Q||P_\theta)} + \underbrace{\mathcal{L}_{\text{target}} + \mathcal{L}_{\text{input}}}_{\mathcal{R}(Q, P_{\text{data}})}, \quad (7)$$

$$\mathcal{L}_{\text{NAT}} = \mathbb{E}_{Q(Z|X)} \mathcal{D}_{\text{KL}} \left[Q(T|Z, X) || P_\theta(T|Z, X) \right], \quad (8)$$

$$\mathcal{L}_{\text{target}} = \mathbb{E}_{P_{\text{data}}(Y|X)} \mathbb{E}_{Q(T|X)} \left[-\log P_\theta(Y|T) \right], \quad (9)$$

$$\mathcal{L}_{\text{input}} = \mathcal{D}_{\text{KL}} \left[Q(Z|X) || P_\theta(Z|X) \right]. \quad (10)$$

In Eq.7, $\mathcal{L}_{\text{NAT}}$ supervises the decoder $P_\theta(T|Z, X)$ to maximize the proxy-likelihood. $\mathcal{L}_{\text{target}}$ and $\mathcal{L}_{\text{input}}$ measure the cost in bridging T, Z with X, Y, which act as regularizers to avoid large gaps between the proxy and original variables.

Optimization MPLE training needs to: (1) find optimal proxy variables Z and T and (2) optimize the model parameter θ. We utilize *Expectation*

[2] We only discuss the non-iterative version of CMLM, following Qian et al. (2021); Du et al. (2021).

Algorithm 1 EM Training for MPLE

1: **for** X sampled from P_{data} **do**
2: **E-step:** Update Z and T by heuristic rules (Sec.3.4), which implicitly balances $\mathcal{L}_{\text{NAT}}$ and regularizers.
3: **M-step:** Given Z and T obtained in the E-step,
4: Calculate $\mathcal{L}_{\text{NAT}}$ according to Eq.8.
5: Calculate $\mathcal{L}_{\text{input}}$ according to Eq.9.
6: Calculate $\mathcal{L}_{\text{target}}$ according to Eq.10.
7: Update θ by minimizing
$\mathcal{L}_{\text{MPLE}} = \mathcal{L}_{\text{NAT}} + \mathcal{L}_{\text{target}} + \mathcal{L}_{\text{input}}$.
8: **end for**

Maximization to update the proxy variables and optimize θ alternatively, as shown in Algo.1.

In **E-step**, we update Z and T to reduce $\mathcal{L}_{\text{MPLE}}$, which aims to find good proxy variables to balance $\mathcal{L}_{\text{NAT}}$ and the regularizers.[3] However, finding optimal proxy variables is non-trivial, where we utilize some heuristic rules to obtain Z and T. Rule examples are shown in Fig.3 (Right). Some rules adaptively choose proxy variables according to the NAT performance, so Z and T are updated through the training. We will introduce the rules in Sec.3.4.

In **M-step**, all Q's entropies can be regarded as constants and ignored, where the three losses can be easily calculated based on the proxy variables previously obtained in E-step.

3.3 MPLE breaks the Bounds of *Conditional Total Correlation*

Theorem 1 cannot apply to an MPLE-based NAT because our latent variable model does not directly predict each token of Y based on X (i.e., not satisfying Eq.3). However, since the NAT decoder still uses a *conditional independent assumption* similar to Eq.3, the proxy distribution provides a new bound C' satisfying $\mathcal{L}_{\text{NAT}} \geq C'$, where

$$C' = \mathbb{E}_{Q(Z|X)}\left[\sum_{i=1}^{L} H_Q(t_i|Z,X) - H_Q(T|Z,X)\right].$$

In E-step, MPLE finds proxy variables that minimize $\mathcal{L}_{\text{NAT}}$ and the regularizers, which equivalently reduces the upper bound of C' for the proxy distribution. In M-step, the NAT decoder is trained towards the proxy distribution with a reduced total correlation. Therefore, the EM training can effectively reduce information loss in capturing the dependencies (also verified empirically in Sec.4.1), which improves the NAT's performance.

3.4 Understanding Existing Methods in the Framework of MPLE

Many existing training methods for NATs can be regarded as instantiations of MPLE, where they use heuristic rules to obtain proxy variables, as shown in Fig.3 (Right). MPLE provides a unified perspective to explain the success of these methods: their heuristic rules actually find good proxy variables that implicitly balance $\mathcal{L}_{\text{NAT}}$ and the regularizers, and the methods with a smaller $\mathcal{L}_{\text{MPLE}}$ have a better translation performance (verified in Sec.4). In this section, we briefly introduce these methods and leave the details in Appendix D.

Obtaining Proxy Target Raw Data and KD use fixed proxy targets that are not updated during the training. Specifically, Raw Data uses the original target sentence as the proxy target. KD first trains an autoregressive model P_{AR} on the raw data and then generates the proxy target by beam search.

AXE and OaXE adaptively obtain the proxy target according to the decoder's performance. Specifically, they first pick a reference R from the raw or KD data, and then find $T \in \mathcal{S}(R)$ that minimizes $\mathcal{L}_{\text{NAT}}$, where $\mathcal{S}(R)$ indicates all subsequences or permutations of R. They also use tricks to avoid a large gap between T and Y (detailed in Appendix D), which implicitly balance $\mathcal{L}_{\text{NAT}}$ and $\mathcal{L}_{\text{target}}$.

Obtaining Proxy Input Vanilla Input obtains proxy inputs with a deterministic function, such as copying source embeddings (Gu et al., 2018).

CMLM and GLAT construct the proxy input by *glancing* at target tokens, i.e., sampling several target tokens to replace the elements in the vanilla input. Specifically, the sampling process has three steps: (a) Sample $l \in [1, L]$ as the number of glanced tokens. (b) Determine which tokens are glanced by sampling a mask sequence M where only l elements are ones. (c) $Z = M \odot emb(T) + (1-M) \odot Z^*$, where $\odot$ is element-wise multiplication, $emb(T)$ is T's embedding, and Z^* is copied from X's embeddings.

The differences between CMLM and GLAT mainly lie in step (a). CMLM use a predefined distribution for l. GLAT uses an adaptive sampling strategy, which determines l based on the NAT decoder's prediction (detailed in Appendix D).

Implementation of Regularizer To measure the gap between the proxy and original variables in these methods, we define $\mathcal{L}_{\text{target}}$ and $\mathcal{L}_{\text{input}}$ and specify the implementations of the input predictor and output paraphraser.

For $\mathcal{L}_{\text{target}}$, we define the output paraphraser as a simple non-trainable distribution related to the similarity between Y and T:

$$P_\theta(Y|T) = exp(\beta S(Y,T))/\zeta, \qquad (11)$$

[3] Formally, we adjust $Q(T,Z|X)$ to minimize $\mathcal{L}_{\text{MPLE}}$, which updates T, Z for a given X.

where β is a hyper-parameter, $S(Y,T)$ is the sentence BLEU, and $\zeta = \sum_Y exp(\beta S(Y,T))$. However, the normalization term ζ is intractable, so we drop ζ and use $\hat{\mathcal{L}}_{\text{target}}$ in our experiments:

$$\hat{\mathcal{L}}_{\text{target}} = E_{P_{\text{data}}(Y|X)} E_{Q(T|X)} \left[-\beta S(Y,T)\right]. \quad (12)$$

Intuitively, Eq.12 measures the gap between proxy and real targets by the average BLEU score.

For $\mathcal{L}_{\text{input}}$, Vanilla Input always has $\mathcal{L}_{\text{input}} = 0$ because copying source embeddings are fully predictable. However, a trainable input predictor is required in GLAT and CMLM. Specifically, we define the input predictor as: (1) Do a binary classification at each position to determine whether z_i is a glanced token or the vanilla input. (2) Predict t_i from the vocabulary if z_i is a glanced token. Formally, $P_\theta(z_i = emb(t_i)|X) = P_\theta(z_i \text{ is glanced}|X) P_\theta(t_i|X)$, and $\mathcal{L}_{\text{input}}$ can be calculated according Eq.9.

3.5 A New Method: Dynamic KD

Existing methods heuristically obtain proxy variables and optimize $\mathcal{L}_{\text{MPLE}}$ implicitly. We propose a simple method named dynamic KD to obtain T, which explicitly balances $\mathcal{L}_{\text{NAT}}$ and $\mathcal{L}_{\text{target}}$.

For a source sentence X, we define a target candidate set Γ, which contains the raw data and distilled data from AT teachers of different sizes, i.e., Transformer-*tiny/small/base/big*. Then we choose a best target $T \in \Gamma$ that minimizes $\mathcal{L}_{\text{NAT}} + \mathcal{L}_{\text{target}}$. Note that Eq.12 is intractable due to the sampling from P_{data}, so we use pairwise BLEU between candidates in Γ instead. See Appendix F for details.

Previous work (Zhou et al., 2020) finds that the KD data from a larger AT teacher is closer to the real data but more difficult to predict, where they suggest choosing the teacher size according to NAT's capacity. Our method dynamically selects the best proxy target from multiple KD candidates, which achieves substantial improvement over NATs that utilizes any single KD data.

4 Experiments

Dataset We conduct experiments on machine translation benchmarks, WMT14 En-De (4.5M) and WMT17 Zh-En (20M). We follow Zhou et al. (2020); Kasai et al. (2020) for preprocessing.

Knowlegde Distillation We use Transformer-*base* with the same settings in Vaswani et al. (2017) and generate the distilled data with beam size 5. All models are based on KD unless otherwise specified.

Implementation Details We implement Raw Data, KD, AXE, OaXE for obtaining proxy targets, and Vanilla Input, CMLM, GLAT for obtaining proxy inputs based on Fairseq (Ott et al., 2019). We generally follow the hyper-parameters in Qian et al. (2021). For fair comparisons, we only modifies the proxy variables across different methods, which may be different from their original implementations. For example, we do not use iterative refinement for CMLM, or combine OaXE with CMLM. Unless otherwise specified, we do not utilize reranking methods or other decoding tricks. More details are in Appendix E.

Decoding Strategies We utilize two decoding strategies. **Greedy Decoding**: The input predictor, the NAT decoder, and the output paraphraser take the most possible choices in each generation step. We use Greedy Decoding for all models unless otherwise specified. **Random Glance**: For the input predictor of CMLM/GLAT, we first randomly determine whether z_i is glanced according to $P_\theta(z_i \text{ is glanced}|X)$, and then choose the most likely tokens for the glanced tokens according to $P_\theta(t_i|X)$. The other steps remain the same as Greedy Decoding.

Metrics The translation performance is evaluated based on tokenized BLEU (Papineni et al., 2002). $\mathcal{L}_{\text{input}}$ and $\mathcal{L}_{\text{NAT}}$ are averaged per token on validation set. $\hat{\mathcal{L}}_{\text{target}}$ in Eq.12 needs multiple real targets Y from P_{data}, so we utilize multi-reference sets (Ott et al., 2018; Hassan et al., 2018), where each source has 10(2) extra human-annotated references for En-De(Zh-En). We use $\beta = 0.2$ for En-De, $\beta = 0.25$ for Zh-En. $\hat{\mathcal{L}}_{\text{MPLE}} := \mathcal{L}_{\text{NAT}} + \mathcal{L}_{\text{input}} + \hat{\mathcal{L}}_{\text{target}}$. We measure the speedup by the average decoding latency on WMT14 En-De with batch size 1.

4.1 Empirical Verification of MPLE's *Conditional Total Correlation*

We compare $\mathcal{L}_{\text{NAT}}$ of models with MPLE against the dataset's total correlation in Fig.4. The models with appropriate proxy variables can achieve lower $\mathcal{L}_{\text{NAT}}$ than the total correlation, indicating that MPLE breaks the bound of the conditional total correlation and reduces the information loss in capturing the dependencies between target tokens.

However, lower $\mathcal{L}_{\text{NAT}}$ does not promise higher BLEU because we do not control the regularizer. In next sections, we will analyze how different methods affect the translation performance and balance the MLE objective and the regularizer.

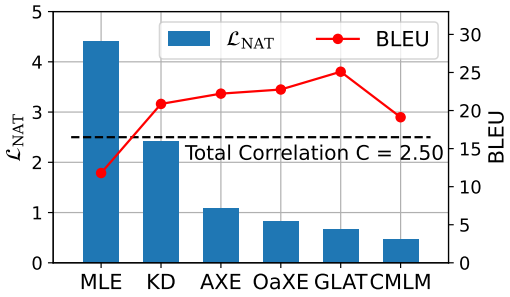

Figure 4: $\mathcal{L}_{\text{NAT}}$ and BLEU of different models on WMT14 En-De. The right five methods are instantiations of MPLE, which achieve lower $\mathcal{L}_{\text{NAT}}$ than the dataset's total correlation C. C is the KL lower-bound of MLE-based NATs given in Theorem 1.

Models	$\mathcal{L}_{\text{NAT}}$	$\hat{\mathcal{L}}_{\text{target}}$	$\hat{\mathcal{L}}_{\text{MPLE}}$	BLEU
MLE (Raw Data)	4.41	-6.42	-2.01	11.79
KD	2.42	**-7.08**	-4.66	20.87
+ AXE(τ=1)	**0.78**	-5.13	-4.35	18.56
+ AXE(τ=5)	1.09	-6.34	-5.25	22.22
+ AXE(τ=10)	1.25	-6.50	-5.26	22.35
+ OaXE(10k)	1.03	-4.41	-3.38	15.00
+ OaXE(50k)	0.79	-5.84	-5.06	21.37
+ OaXE(300k)	0.83	-6.28	**-5.44**	**22.76**

Table 2: Comparison of methods that obtain proxy targets on WMT14 En-De. All methods use Vanilla Input and $\mathcal{L}_{\text{input}}$=0. $\hat{\mathcal{L}}_{\text{MPLE}}$ and BLEU are strongly correlated (Pearson's $|r|$=0.99). AXE's τ and OaXE's numbers indicate the skip penalty and the pre-training steps, which are hyper-parameters in choosing proxy targets. Results on WMT17 Zh-En are in Appendix A.

4.2 Effects of Proxy Target

In this section, we compare different methods obtaining proxy targets while keeping Z as Vanilla Input. The results are shown in Table 2.

Strong Correlation. $\hat{\mathcal{L}}_{\text{MPLE}}$ is strongly correlated with BLEU, where $\mathcal{L}_{\text{NAT}}$ and $\hat{\mathcal{L}}_{\text{target}}$ are both important. For example, AXE(τ=1) achieves low $\mathcal{L}_{\text{NAT}}$ with high $\hat{\mathcal{L}}_{\text{target}}$, indicating that T is easy to predict but heavily distorted from the real target. On the contrary, KD's proxy target is less distorted but hard to predict. OaXE(300k) balances the two losses well and thus achieves the best BLEU.

β in Eq.12 will affect the scale of $\hat{\mathcal{L}}_{\text{target}}$, where we choose $\beta = 0.2$ by maximizing the correlation on the validation set. However, the choose of β is not sensitive that $|r| \geq 0.8$ for all $\beta \in [0.1, 0.5]$.

Secret Advantage of KD. Previous work (Gu et al., 2018) has shown that KD can simplify the training data and thus reduce $\mathcal{L}_{\text{NAT}}$. However, our results reveal a secret advantage that KD achieves the lowest $\hat{\mathcal{L}}_{\text{target}}$, which means that the KD data are more similar with the multiple references rather than the raw data. This result can be explained by the diversity of human annotations as shown in Fig.5. Although the KD data may not belong to any modes of the data distribution, it still has higher similarity on average.

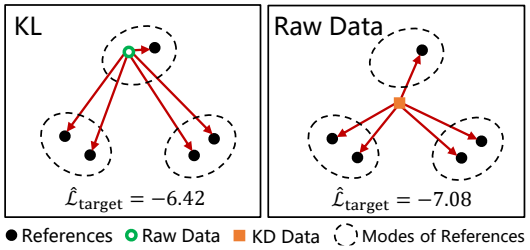

Figure 5: The similarity between proxy targets and references in two methods. KD data achieve higher similarity with references on average than the raw data.

Models	$\mathcal{L}_{\text{input}}$	$\mathcal{L}_{\text{NAT}}$	$\hat{\mathcal{L}}_{\text{MPLE}}$	Random	Greedy
Vanilla Input	**0**	2.42	-4.66	20.87	20.87
CMLM	0.99	**0.46**	-5.63	23.48	19.39
+ Fixed	0.48	0.55	**-6.05**	24.28	19.35
GLAT	0.45	0.66	-5.96	23.98	25.12
+ Levenshtein	0.41	0.73	-5.94	24.03	24.84
+ P_{ref}	0.25	1.24	-5.59	22.98	24.22
+ $1 - P_{\text{ref}}$	0.57	0.50	-6.01	**24.35**	**25.19**

Table 3: Comparison of methods that obtain proxy inputs on WMT14 En-De. All use KD and $\hat{\mathcal{L}}_{\text{target}} = -7.08$. Random and Greedy indicate the BLEU score with Random Glance and Greedy Decoding. $\hat{\mathcal{L}}_{\text{MPLE}}$ is strongly correlated with Random BLEU ($|r|$=0.99) but less correlated with Greedy BLEU ($|r|$=0.37). Results on WMT17 Zh-En are in Appendix A.

Hyper-parameters and Trade-off. AXE and OaXE utilize tricks to avoid large gap between the proxy target and the real target. For example, AXE tunes the skip penalty τ, and OaXE tunes the pre-training step.[4] MPLE provides a quantifiable objective to balance the KL divergence $\mathcal{L}_{\text{NAT}}$ and the regularizer $\hat{\mathcal{L}}_{\text{target}}$, which improves the interpretability for hyper-parameter selection.

4.3 Effects of Proxy Input

We compare methods that obtain proxy inputs including several variants of CMLM and GLAT, which are also used in Qian et al. (2021). CMLM + Fixed uses $l = 0.2L$ instead of random sampling. GLAT + Levenshtein uses Levenshtein Distance to determine the number of glanced tokens. P_{ref} and $1 - P_{\text{ref}}$ choose the glanced tokens according to the difficulties in predicting them, where $P(z_i \text{ is glanced})$ is proportional to the prediction probability $P_\theta(t_i|Z^*, X)$ or $1 - P_\theta(t_i|Z^*, X)$. The results are shown in Table 3.

[4] See Appendix D.3, D.4 for details of the tricks.

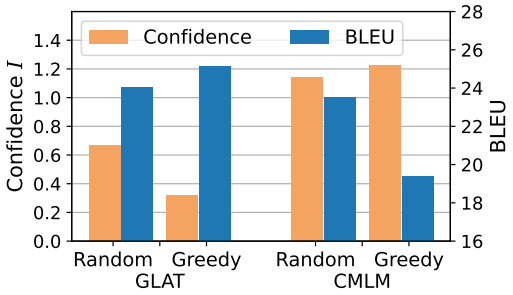

Figure 6: The BLEU score and decoding confidence $I = -\log P_\theta(Z|X) - \log P_\theta(T|Z,X)$ of GLAT and CMLM with two decoding strategies. Smaller I indicates that the model is more confident in generation.

Strong Correlation with Random BLEU. Our objective is strongly correlated with BLEU when using Random Glance, where $\mathcal{L}_{\text{input}}$ and $\mathcal{L}_{\text{NAT}}$ should be balanced to achieve the best performance. For example, Vanilla NAT does not introduce hints in the proxy input, leading to large $\mathcal{L}_{\text{NAT}}$. CMLM introduces too much hints in training, making it hard to predict the proxy input in inference.

However, $\hat{\mathcal{L}}_{\text{MPLE}}$ is less correlated with BLEU of Greedy Decoding, where CMLM may not prefer the decoding strategy as discussed below.

Potentials for Decoding Strategies. Previous works (Qian et al., 2021) showed that CMLM performs poorly without iterative refinement, but we find that it can be improved by changing the decoding strategy. For example, CMLM + Fixed outperforms GLAT when using Random Glance, showing the potentials for the decoding strategies.

We find that CMLM and GLAT prefer different decoding strategies, which can be partially explained by the decoding confidence $P_\theta(Z|X)$ and $P_\theta(T|Z,X)$. As shown in Fig.6, GLAT is more confident with Greedy Decoding than Random Glance, whereas CMLM is the opposite.

4.4 Results of Dynamic KD

We combine Dynamic KD with Vanilla Input and GLAT, and compare our methods against AT and SoTA NATs. As shown in Table 4, Dynamic KD achieves substantial improvement over the best result on any single KD data and further help GLAT achieves competitive translation quality with AT Transformer with remarkable speedup.

This experiment suggests that our objective effectively guides the design of new training methods. Explicitly optimizing $\mathcal{L}_{\text{MPLE}}$ provides a promising way to find better proxy distributions, which outperforms existing heuristic methods.

	Models	En-De	Zh-En	Speed Up
AT	Transformer	27.11	23.89	1.0x
NAT	MLE	11.79	8.69	15.3x
	GLAT (NPD=7)†	26.55	/	7.9x
	OaXE (LPD=5)†	26.1	22.1	14.2x
Ours	Vanilla ‡	21.33	17.48	15.3x
	+ Dynamic KD	22.82	18.29	15.3x
	+ LPD=3 §	24.83	19.97	14.6x
	GLAT ‡	25.33	22.51	15.3x
	+ Dynamic KD	25.88	23.07	15.3x
	+ LPD=3 §	**26.89**	**24.42**	14.6x

Table 4: Comparing Dynamic KD against AT and SoTA NATs. NPD (Gu et al., 2018) and LPD (Wei et al., 2019a) indicate reranking methods with the number of candidates. †: Reported by Qian et al. (2021) and Du et al. (2021). ‡: The best results on single KD data. §: Some decoding tricks are applied following baselines. See Appendix B for details and more results.

5 Related Work

NATs are proposed to accelerate the decoding speed but suffers from poor translation quality. Many works are devoted in solving the problem, where some of them have been covered in our objective as shown in Fig 3. The other methods mainly includes (1) utilizing objectives not based on MLE (Libovický and Helcl, 2018; Wei et al., 2019a; Saharia et al., 2020); (2) iteratively refining the generated outputs for better quality (Lee et al., 2018; Gu et al., 2019; Kasai et al., 2020). Although the iterative approaches usually lead to better quality, Kasai et al. (2021) point out that the iterative approaches are much slower than non-iterative ones and may not have advantages against AT models.

The gap between likelihood and generation performance has been spotted in ATs, where previous works utilize reinforcement learning (Ranzato et al., 2016; Bahdanau et al., 2017) or adversarial training (Yu et al., 2017) to alleviate the problem. Lee et al. (2020) conduct an empirical study to quantify this gap in both ATs and NATs, but they focus on comparing latent variable models with different priors. To our knowledge, we are the first to point out that NATs suffer from the misleading MLE objective and propose a method to allieviate the problem.

6 Conclusion

In this paper, we analyze why MLE misleads NATs' training and find that NATs cannot capture the dependencies between target tokens. Based on the analysis, we propose a new method, MPLE, to alleviate the problem. Our solution provides a general perspective for many SoTA methods and can inspire new methods for better NATs.

References

Hirotogu Akaike. 1998. Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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

Charles W. Fox and Stephen J. Roberts. 2012. A tutorial on variational bayesian inference. *Artif. Intell. Rev.*, 38(2):85–95.

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

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

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.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *CoRR*, abs/1803.05567.

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

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

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

Jason Lee, Dustin Tran, Orhan Firat, and Kyunghyun Cho. 2020. On the discrepancy between density estimation and sequence generation. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP@EMNLP 2020, Online, November 20, 2020*, pages 84–94. Association for Computational Linguistics.

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

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. 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 3953–3962. PMLR.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics.

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

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

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

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

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

Harshil Shah and David Barber. 2018. Generative neural machine translation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1353–1362.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5719–5728. PMLR.

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 *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8846–8853. AAAI Press.

Milan Studený and Jirina Vejnarová. 1998. The multiinformation function as a tool for measuring stochastic dependence. In Michael I. Jordan, editor, *Learning in Graphical Models*, volume 89 of *NATO ASI Series*, pages 261–297. Springer Netherlands.

Jinsong Su, Shan Wu, Deyi Xiong, Yaojie Lu, Xianpei Han, and Biao Zhang. 2018. Variational recurrent neural machine translation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5488–5495. AAAI Press.

Zhiqing Sun and Yiming Yang. 2020. An EM approach to non-autoregressive conditional sequence generation. 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 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.

Michael Satosi Watanabe. 1960. Information theoretical analysis of multivariate correlation. *IBM J. Res. Dev.*, 4(1):66–82.

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

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

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press.

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.

Models	$\mathcal{L}_{\text{NAT}}$	$\hat{\mathcal{L}}_{\text{target}}$	$\hat{\mathcal{L}}_{\text{MPLE}}$	BLEU
Raw Data	4.43	**-6.25**	-1.82	8.69
KD	2.85	-5.72	-2.87	15.53
+ AXE(τ=1)	**1.02**	-2.95	-1.93	9.68
+ AXE(τ=5)	1.93	-5.00	-3.07	18.39
+ AXE(τ=10)	2.31	-5.20	-2.90	18.25
+ OaXE(10k)	1.46	-3.38	-1.92	12.31
+ OaXE(50k)	1.19	-4.50	-3.31	18.79
+ OaXE(300k)	1.15	-4.66	**-3.50**	**19.46**

Table 5: Comparison of methods that obtain proxy targets on WMT17 Zh-En. All methods use Vanilla Input and $\mathcal{L}_{\text{input}}$=0. $\hat{\mathcal{L}}_{\text{MPLE}}$ and BLEU are strongly correlated (Pearson's $|r|$=0.96). AXE's τ and OaXE's numbers indicate the skip penalty and the pre-training step, which are hyper-parameters in choosing proxy targets.

Models	$\mathcal{L}_{\text{input}}$	$\mathcal{L}_{\text{NAT}}$	$\hat{\mathcal{L}}_{\text{MPLE}}$	Random	Greedy
Vanilla Input	**0**	2.85	-2.87	15.53	15.53
CMLM	1.13	0.76	-3.83	19.74	14.12
+ Fixed	0.29	1.13	**-4.30**	**20.81**	14.49
GLAT	0.33	1.26	-4.13	20.73	22.51
+ Levenshtein	1.08	**0.44**	-4.19	20.71	21.70
+ P_{ref}	0.40	1.60	-3.73	18.98	21.22
+ $1 - P_{\text{ref}}$	0.73	0.81	-4.17	20.79	**21.51**

Table 6: Comparison of methods and variants that obtain proxy inputs on WMT17 Zh-En. All use KD and $\hat{\mathcal{L}}_{\text{target}}$=-5.72. Random and Greedy indicate the BLEU score in Random Glance and Greedy Decoding. $\hat{\mathcal{L}}_{\text{MPLE}}$ is strongly correlated with Random BLEU (Pearson's $|r|$=0.99) but less correlated with Greedy BLEU ($|r|$=0.35).

A Results on WMT17 Zh-En

We repeat the experiments in Sec.4.2 and Sec.4.3 on WMT17 Zh-En. As shown in Table 5 and Table 6, our objective is strongly correlated with the translation quality, which supports our claim well.

B More Results of Dynamic KD

Decoding Tricks In Table 4, we apply some decoding tricks for the final results:

- We use length parallel decoding (LPD, Wei et al., 2019b). We use a candidate set of 3. Since all candidates can be generated simultaneously, LPD is still fast in inference. It is worth noting that LPD is faster than NPD (Gu et al., 2018) since it does not need an external reranker.
- We use the de-duplication trick (Lee et al., 2018), i.e., removing the repeated tokens in generated sentences.
- We adjust the predicted length by a factor λ (Ghazvininejad et al., 2020). The factor is tuned on the validation set. We use $\lambda = 1$ (i.e., the predicted length is not changed) for WMT14 En-De, and $\lambda = 1.05$ for WMT17 Zh-En.

	Models	# Iters	En-De	Zh-En	Speed Up
AT	Transformer	L	27.11	23.89	1.0x
Iterative NATs	CMLM†	4	25.94	21.90	3.0x
		10	27.03	23.21	1.3x
	DisCo†	4	25.83	22.42	4.3x
		10	27.06	23.68	3.2x
		∼ 5 §	27.34	23.83	/
	Imputer†	1	25.8	/	14.9x
		2	27.5	/	7.5x
		8	**28.2**	/	2.7x
Ours	Vanilla ‡	1	24.83	19.97	14.6x
	GLAT ‡	1	26.89	**24.42**	14.6x

Table 7: Comparing Dynamic KD against Iterative NATs. †: Reported by Ghazvininejad et al. (2019), Kasai et al. (2020), and Saharia et al. (2020). Speed up of Imputer are re-evaluated in our implementation. §: Use adaptive iteration numbers. ‡: Use Dynamic KD and decoding tricks.

	WMT14 En-De		
	Vanilla	GLAT	AT Teacher
Tiny	18.74	20.37	20.46
Small	21.33	23.72	24.29
Base	20.98	25.33	27.11
Big	19.27	25.12	28.49
Raw	11.79	19.42	/
Dynamic KD	**22.82**	**26.89**	/
w/o regularizer	20.51	22.66	/
	WMT17 Zh-En		
	Vanilla	GLAT	AT Teacher
Tiny	15.98	19.87	19.38
Small	17.48	22.38	22.47
Base	15.53	22.51	23.89
Big	14.81	21.84	24.84
Raw	8.69	18.88	/
Dynamic KD	**18.29**	**23.07**	/
w/o regularizer	18.10	22.14	/

Table 8: Comparisons between Dynamic KD and single KD data. *w/o regularizer* indicates removing $\hat{\mathcal{L}}_{\text{target}}$ in choosing the proxy target (i.e., $T^* = \arg\min_T \mathcal{L}_{\text{NAT}}$). All NAT results do not use reranking or other decoding tricks. AT teachers' performances are presented as references.

Comparisons with Iterative NATs In Table 7, we compare Dynamic KD against iterative NATs. GLAT + Dynamic KD achieves good translation quality with remarkable speed up.

Ablation Study In Table 8, we present the results on each single KD data and performance of AT teachers as reference. Moreover, we ablate the regularizer ($\hat{\mathcal{L}}_{\text{target}}$) in Dynamic KD, which justifies the necessity of the regularizer.

C More Related Work

Our method is connected with Sun and Yang (2020), who also utilize the EM algorithm to improve NATs' performance. They alternatively train AT and NAT models to refine the training data, where each model is trained with the data distilled from the previous one. Different from their method, we utilize a proxy distribution to guide the training, which is more general and simpler than training multiple models iteratively.

Latent variable models are widely used in machine translation, which can capture complex dependencies (Su et al., 2018), model the diversity of outputs (Shen et al., 2019), enable the translation between unseen language pairs (Shah and Barber, 2018), and so on. Some NATs also utilize latent variable models, which mainly aim to alleviate the multi-modality problem (Gu et al., 2018) by making a sentence-level plan before the non-autoregressive generation (Gu et al., 2018; Lee et al., 2018; Shu et al., 2020). Unlike previous works, we introduce the latent variable model to construct a proxy distribution with reduced total correlation, which addresses the theoretical problem of MLE-based NATs.

D Details of Understanding Existing Methods in MPLE

In Sec.3.4, we briefly describe that how existing methods obtain proxy variables. In this section, we provide more details.

Heuristic Rule and Variational Distribution In the derivation of MPLE (Eq.5), we introduce a variational distribution $Q(T, Z|X)$, which specifies how we sample proxy variables to train the NAT model. However, since the optimization of E-step is usually non-trivial, we instead use heuristic rules to obtain the proxy variables, where the obtained proxy variables in fact build the variational distribution $Q(T, Z|X)$ used in our objective. Specifically, for a given X, we select one rule to obtain T and another one to obtain Z, where we regard the (T, Z) pair as a sample from $Q(T, Z|X)$. Formally, we define the rules for proxy targets as $Q(T|X)$, and the rules for proxy inputs as $Q(Z|T, X)$. Then $Q(T, Z|X) := Q(T|X)Q(Z|T, X)$, which connects the heuristic rules with the variational distribution.[5] In later descriptions, we will formulate the heuristic rules as distributions over T or Z.

Equivalence of Objectives The original objectives in these methods are generally equivalent to minimizing $\mathcal{L}_{\text{NAT}}$ of Eq.8. For example, the origi-

[5] Do not mix it with the proxy distribution $Q(T|Z, X)$, which is from another decomposition of $Q(T, Z|X) = Q(T|Z, X)Q(Z|X)$. Generally, we only have one distribution $Q(T, Z|X)$, which is defined by the heuristic rules. All the other Qs are just decompositions of $Q(T, Z|X)$.

nal KD minimizes the negative log-likelihood on KD data, which is equivalent to reducing the KL divergence between $Q(T|Z,X)$ and $P_\theta(T|Z,X)$, where T is the KD target and Z is copied from source embeddings. In later descriptions, we will prove the equivalence of their original objectives and ours in complex methods, such as AXE.

D.1 Raw Data

Raw Data uses the original target sentence as the proxy target. Formally, we define $Q(T|X)$ as a one-point distribution that $Q(T = Y^*|X) = 1$, where Y^* is the original target in the dataset.

D.2 Knowledge Distillation (KD, Gu et al., 2018)

KD first trains an autoregressive model P_{AR} on the raw data, and then uses beam search to obtain $T^* = \arg\max_Y P_{AR}(Y|X)$. Formally, $Q(T|X)$ is defined as a one-point distribution at T^*.

D.3 Aligned Cross Entropy (AXE, Ghazvininejad et al., 2020)

$Q(T|X)$ is defined as a one-point distribution at T^*, where $T^* = \arg\min_{T\in\mathcal{S}(R)} \mathcal{L}_{\text{NAT}}$. The reference $R = [r_1, \cdots, r_L]$ is picked from Raw Data or KD. Any $T \in \mathcal{S}(R)$ is a subsequence of R with empty tokens ϵ inserted.[6] An example is shown in Fig.7.

Original Objective AXE introduces a monotonic alignment $\alpha = [\alpha_1, \cdots, \alpha_L]$, where the i-th token of the reference R is aligned to the α_i-th token of the NAT prediction. Formally, the AXE loss is defined as

$$\mathcal{L}_{\text{AXE}} = \min_{\alpha} \left[-\sum_{i=1}^{L} \log P_{\alpha_i}(r_i) - \sum_{k\notin\alpha} \log P_k(\epsilon) \right], \quad \text{s.t. } 1 \le \alpha_1 \le \cdots \le \alpha_L \le L.$$

The first term is the cross entropy between aligned targets and predictions, and the second term is a penalty for unaligned predictions.

In the AXE loss, a single prediction may be aligned to multiple target tokens. In their original paper, aligning the prediction to the first target token is called the "align" operation, and aligning the prediction to later tokens is called the "skip target" operation. However, one-to-many alignments will damage the performance, so they penalize the "skip target" operations with a factor δ. This trick is called the *skip penalty*.

[6] The NAT model may learn to predict empty tokens, which will be removed after generation.

Proof of Equivalence To connect their definition with ours, we convert the alignment to an adjacency list, as shown in Figure 7, where β_i is a list containing all aligned tokens for the i-th prediction. Specially, if the i-th prediction is not aligned, we set $\beta_i = [0]$ and $r_0 = \epsilon$. Then, $\mathcal{L}_{\text{AXE}}$ can be reformulated as

$$\min_{\beta} \left[-\sum_{i=1}^{L} \log P_i(r_{\beta_{i,1}}) - \delta \sum_{i=1}^{L} \sum_{j=2}^{|\beta_i|} \log P_i(r_{\beta_{i,j}}) \right],$$

where $\beta_{i,j}$ indicates the j-th element of β_i. The first term is the cross entropy between the prediction and a new target $T^* = [r_{\beta_{1,1}}, \cdots, r_{\beta_{L,1}}]$, and the second term is the penalty for "skipping target" operations.

When $\delta = 0$, the above formulation is equivalent to finding an optimal T^* to minimize $\mathcal{L}_{\text{NAT}}$. Since α is monotonic alignments, T^* is constrained and should be a subsequence of R with some empty tokens inserted, which recover our definition. When $\delta \neq 0$, the second term can be regarded as a regularizer to control the gap between proxy targets and real targets.

D.4 Order-agnostic Cross Entropy (OaXE, Du et al., 2021)

OaXE is similar to AXE despite the constraint $\mathcal{S}(R)$. Any $T \in \mathcal{S}(R)$ is a permutation of R. An example is shown in Fig.7.

Original Objective Different from AXE, OaXE's α is a non-monotonic alignment, and each predicted token can only be used once. The loss is defined as

$$\mathcal{L}_{\text{OaXE}} = \min_{\alpha\in Perm(L)} \left[-\sum_{i=1}^{L} \log P_{\alpha_i}(r_i) \right],$$

where $Perm(L)$ indicates the permutations of sequences containing 1 to L.

Proof of Equivalence Similar to the derivation for AXE, we can reformulate $\mathcal{L}_{\text{OaXE}}$ as

$$\min_{\beta} \left[-\sum_{i=1}^{L} \log P_i(r_{\beta_{i,1}}) \right].$$

The above formulation recovers our definition: It finds an optimal T^* to minimize $\mathcal{L}_{\text{NAT}}$, where T^* can be an arbitrary permutation of R.

However, without the monotonic constraints, T^* in OaXE may be heavily distorted from the real target Y. To alleviate the problem, OaXE first pretrains the NAT using the vanilla MLE and then finetunes it to minimize $\mathcal{L}_{\text{OaXE}}$. This trick is based on an intuition that the optimal T^* in a well-trained NAT will be close to the real target.

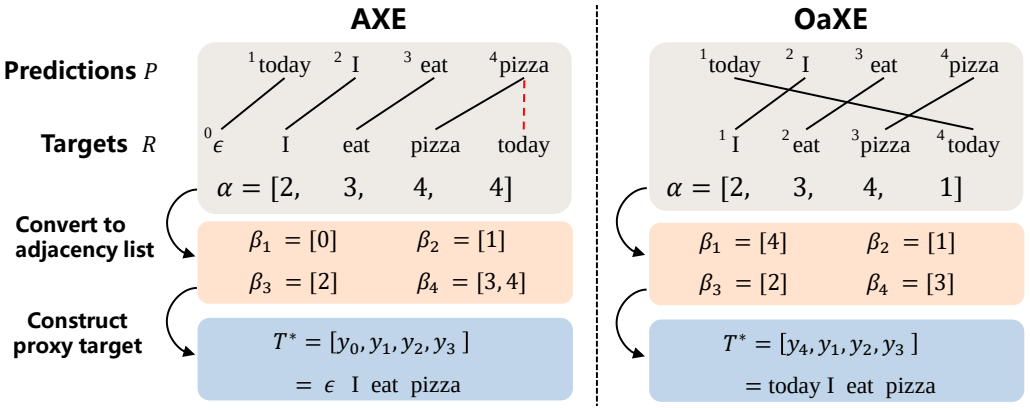

Figure 7: Examples of the alignments α, the adjacency list β, and the proxy target T^* in AXE and OaXE. Red dotted line in AXE indicates the *skip target* operation.

D.5 Vanilla Input

Many NATs predict Z by Uniform Copy (Gu et al., 2018) or attention (Qian et al., 2021). Formally, it can be formulated as a one-point distribution $Q(Z = Z^*|T, X) = 1$, where Z^* is obtained from a deterministic function $f(X)$.

D.6 VAE (Shu et al., 2020)

VAE uses two trainable networks, the prior and posterior networks, to model $P_\theta(Z|X)$ and $Q(Z|T, X)$, respectively. The posterior network $Q(Z|T, X)$ can be trained together with θ.

D.7 CMLM (Ghazvininejad et al., 2019) / GLAT (Qian et al., 2021)

As described in Sec.3.4, $Q(Z|X)$ is defined by three sampling steps.

In step (a), CMLM defines the number of glanced tokens $l = \lambda L$, where λ is uniformly sampled from 0 to 1. GLAT uses an adaptive sampling strategy, where $l = \lambda D(T, \hat{T})$. $\lambda \in [0, 1]$ is a hyper-parameter, $D(T, \hat{T})$ is the Hamming distance between the proxy target and the model prediction $\arg\max_T P_\theta(T|Z^*, X)$.

Original Objective In the original implementation, CMLM and GLAT use a masked language model objective, where the glanced tokens are not included in the loss $\mathcal{L}_{\text{NAT}}$. Formally,

$$\mathcal{L}_{\text{NAT}} = E_{Q(Z,T|X)}\left[-\sum_{i\notin\mathcal{G}} \log P_\theta(t_i|Z, X)\right],$$

where $\mathcal{G}$ is the set of the glanced token.

Proof of Equivalence The above objective does not violate Eq.8 if we add a *copy mechanism* in the NAT decoder. The decoder directly copies the glanced token as the prediction if available. As a result, for a glanced token t_i, $\log P_\theta(t_i|Z, X) = 0$ because the predictions for the glanced tokens are always correct. Therefore, the masked language model objective recovers the MLE objective towards $Q(T|Z, X)$:

$$\begin{aligned}\mathcal{L}_{\text{NAT}} &= E_{Q(Z,T|X)}\left[-\sum_{i\notin\mathcal{G}} \log P_\theta(t_i|Z, X)\right] + 0 \\ &= E_{Q(Z,T|X)}\Big[-\sum_{i\notin\mathcal{G}} \log P_\theta(t_i|Z, X) - \\ &\qquad \sum_{i\in\mathcal{G}} \log P_\theta(t_i|Z, X)\Big] \\ &= E_{Q(Z,T|X)}\left[-\log P_\theta(T|Z, X)\right]\end{aligned}$$

Note that the *copy mechanism* does not need any modifications to the network structures.

Implementation Details of Input Predictor In our implementation, the input predictor for CMLM and GLAT is composed of two modules: $P_\theta(z_i \text{ is glanced}|X)$ predicts whether z_i is from a glanced token t_i, and $P_\theta(t_i|X)$ predicts the target token t_i from the vocabulary. Formally,

$$P_\theta(z_i|X) = \begin{cases} 1 - P_\theta(z_i \text{ is glanced}|X), & \text{if } z_i = z_i^*; \\ P_\theta(z_i \text{ is glanced}|X) P_\theta(t_i|X), & \\ \qquad\qquad \text{if } z_i = emb(t_i); & \\ 0, \qquad\qquad \text{otherwise}, & \end{cases}$$

where z_i^* indicates the vanilla input at the i-th position.

Therefore, $\mathcal{L}_{\text{input}}$ can be formulated as

$$\mathcal{L}_{\text{input}} = \mathbb{E}_{Q(Z|X)}\left[-\sum_{i=1}^{L} \log P_\theta(z_i|X) + \log Q(Z|X)\right].$$

For the first module $P_\theta(z_i \text{ is glanced}|X)$, we reuse the Transformer encoder and the NAT decoder and further add a binary classification layer on top of the NAT decoder. For the second module $P_\theta(t_i|X)$, we use a pre-trained vanilla NAT and freeze its parameters during the training of CMLM

or GLAT. Moreover, its predictions can be computed offline to speed up the training.

For GLAT or CMLM in Greedy Decoding, Z always equals to the vanilla input Z^* because the vanilla input appears more frequently than any glanced tokens. Therefore, we do not need to train the input predictor in this case, which recovers their original implementations.

E Details of Experiment Settings

For WMT14 En-De, we follow Zhou et al. (2020) to use a joint BPE (Sennrich et al., 2016) with 32K merge operations, which leads to a vocabulary of 40k tokens. For WMT17 Zh-En, we follow Kasai et al. (2020) to use a BPE with 32K merge operations, which leads to vocabularies of 48k tokens in Chinese and 33k tokens in English.

Our NAT models generally follow the hyper-parameter of transformer-*base* (Vaswani et al., 2017). For regularization, we set dropout to 0.1, weight decay to 0.01, and label smoothing to 0.1. Except for OaXE, all models are trained for 300k updates with a batch of approximately 64k tokens. The learning rate warms up to $5 \cdot 10^{-4}$ within 10k steps and then decays with the inverse square-root schedule. For OaXE, we choose a pre-trained vanilla NAT and finetune the model for 100k steps with a fixed learning rate of 10^{-5}. We evaluate the BLEU scores on the validation set every epoch and average the best 5 checkpoints for the final model. All models are trained with mixed precision floating point arithmetic on 8 Nvidia V100-32G GPUs. It costs approximately 20 hours for a vanilla NAT and 30 hours for Dynamic KD + GLAT.

For fair comparisons in Table 2 and 3, we do not use any decoding tricks and only modify the methods for obtaining proxy variables. Taking OaXE as an example, our implementation differs from their original paper (Du et al., 2021) in: (1) Our OaXE is finetuned on a vanilla NAT, not a CMLM. (2) They use Transformer-*big* for KD whereas we use Transformer-*base*. (3) They use Length Parallel Decoding (Wei et al., 2019a) of beam 5 and the de-duplication trick (Lee et al., 2018) for decoding. We do not use any reranking methods. (4) We do not use the truncation trick because it is not compatible with $\mathcal{L}_{\text{NAT}}$.

Model	*tiny*	*small*	*base*	*big*
d_{model}	128	256	512	1024
d_{hidden}	512	1024	2048	4096
n_{layers}	3	3	6	6
n_{heads}	4	4	8	8
Dropout	0.1	0.1	0.3	0.3

Table 9: Hyper-parameters different AT teachers, which generate the target candidates in Dynamic KD.

F Implementation Details of Dynamic KD

For a given X, the target candidate set Γ contains Raw Data and four distilled sentences. The distilled sentences are generated with beam size 5 from four AT teachers, whose hyper-parameters in Table 9. For WMT14 En-De, we train the AT teachers for 100k updates with a batch of approximately 64k tokens. For WMT17 Zh-En, we raise the step to 300k to match the size of training data.

The dynamic KD chooses the proxy target by minimizing $\mathcal{L}_{\text{NAT}} + \hat{\mathcal{L}}_{\text{target}}$. However, $\hat{\mathcal{L}}_{\text{target}}$ needs samples from $P_{\text{data}}(Y|X)$ as defined in Eq.12, which is intractable on the training set. To tackle the issue, we utilize the pairwise BLEU between the candidates formulated as follows,

$$\hat{\mathcal{L}}_{\text{target}} \approx \mathbb{E}_{Q(T|X)} \sum_{i=1}^{5} \left[-\beta_i S(\Gamma_i, T)\right], \qquad (13)$$

where S is the sentence BLEU, Γ_i is the target distilled from the i-th teacher model, and β_i is hyper-parameters to bias the candidates from different models ($i = 5$ indicating Raw Data).

For the tuning of β_i, we introduce a multi-reference set (Ott et al., 2018; Hassan et al., 2018) to align $\hat{\mathcal{L}}_{\text{target}}$ defined in Eq.12 and its approximation in Eq.13. Specifically, we tune β_i to minimize the absolute differences between $\hat{\mathcal{L}}_{\text{target}}$ and its approximation on the candidates sentences. Note that tuning β_i only involves the BLEU score, which does not need to train a NAT model. We do a manual search from 0.1 to 0.5, where we finally choose $\beta = [0.14, 0.12, 0.26, 0.30, 0.46]$ for WMT14 En-De and $\beta = [0.09, 0.10, 0.13, 0.12, 0.20]$ for WMT17 Zh-En.

When combining GLAT with Dynamic KD, $\mathcal{L}_{\text{NAT}}$ may suffer from high variance because Z is sampled from $Q(Z|T)$ following the rule of GLAT. In our implementation, we utilize an approximation of $\mathcal{L}_{\text{NAT}}$ as

$$\hat{\mathcal{L}}_{\text{NAT}} = \mathbb{E}_{Q(T|X)} \left[-\log P_\theta(T|Z^*, X)\right],$$

where Z^* is the vanilla input. Then Dynamic KD chooses the proxy target according to $T^* = \arg\min_T \hat{\mathcal{L}}_{\text{NAT}} + \hat{\mathcal{L}}_{\text{target}}$.