Non-Autoregressive Translation with Layer-Wise Prediction and Deep Supervision

Chenyang Huang[*1,2], **Hao Zhou**[†2], **Osmar R. Zaïane**[1], **Lili Mou**[†1], **Lei Li**[‡3]

[1]Department of Computing Science,
Alberta Machine Intelligence Institute (Amii), University of Alberta
[2]ByteDance AI Lab   [3]University of California, Santa Barbara
{chenyangh,zaiane}@ualberta.ca   zhouhao.nlp@bytedance.com
doublepower.mou@gmail.com   lilei@cs.ucsb.edu

Abstract

How do we perform efficient inference while retaining high translation quality? Existing neural machine translation models, such as Transformer, achieve high performance, but they decode words one by one, which is inefficient. Recent non-autoregressive translation models speed up the inference, but their quality is still inferior. In this work, we propose DSLP, a highly efficient and high-performance model for machine translation. The key insight is to train a non-autoregressive Transformer with **D**eep **S**upervision and feed additional **L**ayer-wise **P**redictions. We conducted extensive experiments on four translation tasks (both directions of WMT'14 EN–DE and WMT'16 EN–RO). Results show that our approach consistently improves the BLEU scores compared with respective base models. Specifically, our best variant outperforms the autoregressive model on three translation tasks, while being 14.8 times more efficient in inference.[1]

Introduction

Most state-of-the-art neural machine translation (NMT) systems are autoregressive, where they predict one word after another to generate the target sequence (Bahdanau, Cho, and Bengio 2015; Vaswani et al. 2017). However, autoregressive NMT is slow in inference, which sometimes does not meet the efficiency requirement from the industry.

Gu et al. (2018) propose non-autoregressive neural machine translation (NAT), which is 15.6 times faster than its autoregressive counterpart. While NAT achieves fast inference by generating target tokens in parallel, it assumes that the generated words are conditionally independent given the input. Such an independence assumption, however, weakens the power of sequence modeling, and results in worse performance than autoregressive translation.

In recent studies, researchers propose partially (non-)autoregressive models by progressively generating several words at a time (Ghazvininejad et al. 2019) or iteratively editing with the Levenshtein Transformer (Gu,
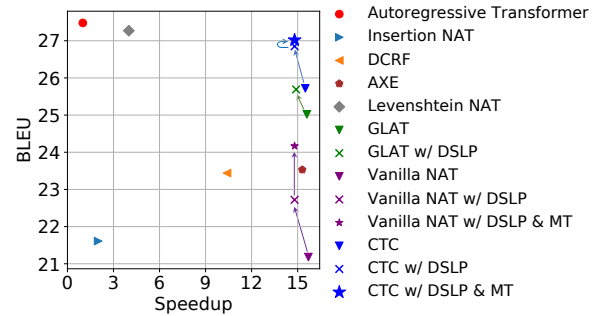
---

[1]Our code, training/evaluation scripts, and output are available at https://github.com/MANGA-UOFA/DSLP.git.

Figure 1: Quality–efficiency trade-off for state-of-the-art NAT models and our DSLP on the WMT'14 EN–DE dataset. A cross "×" represents our DSLP variants, and a star "⋆" refers a DSLP model that is enhanced with our mixed training (MT) technique. Its base model is shown in the same color, and the correspondence is represented by an arrow.

Wang, and Zhao 2019). However, their inference is significantly slower than a full NAT model. As shown by Kasai et al. (2021), a comparable-sized autoregressive model with a shallow decoder is able to outperform these approaches with similar latency. Therefore, these approaches do not achieve a desired quality–efficiency trade-off.

In this work, we propose a simple yet effective approach to non-autoregressive generation with a Deeply Supervised, Layer-wise Prediction-aware (DSLP) Transformer. It is noticed that a traditional NAT Transformer (Vaswani et al. 2017) only makes predictions at the last layer, where all the words are generated in parallel. Thus, the prediction of a word is unaware of other time steps, which typically leads to inconsistent sentences. For example, the English phrase *thank you* has two translations in German: *danke schön* and *vielen dank*. Instead of generating either of the phrases, a non-autoregressive model may predict *danke dank*, which is absurd. Therefore, we propose a layer-wise prediction-aware Transformer that predicts the output at every decoding layer. The prediction is fed to the next decoding layer for further processing. Hence, our decoder model is aware of different steps' predictions and can calibrate the NAT output through its decoding layers. We further introduce deep supervision for training our prediction-aware decoder. This is essential

to our model, because otherwise the intermediate predictions are not grounded to the target output and the calibration would be less meaningful. We also propose to mix the intermediate predictions with groundtruth tokens during training, to further enhance meaningful calibration.

Our DSLP is a generic framework that can be combined with different base NAT models. In our experiments, we evaluated DSLP on WMT'14 English–German and WMT'16 English–Romanian pairs, and considered the translation of both directions. Results show that our DSLP consistently improves the NAT performance for every translation language pair and every base model, including vanilla NAT, CMLM, GLAT, and CTC (see experiments section for details), demonstrating the generality of our approach. In addition, we show that mixing the layer-wise predictions with groundtruth during training further improves the performance. Remarkably, our best model, CTC with DSLP and mixed training, achieves better performance compared with its autoregressive teacher model on three of the four datasets with a 14.8x speedup. Figure 1 positions DSLP in prior work in terms of quality–efficiency trade-off.

Related Work

Non-autoregressive neural machine translation is attracting an increasing attention in the community (Sun et al. 2019; Gu, Wang, and Zhao 2019; Saharia et al. 2020; Qian et al. 2021; Hao et al. 2021), because it is more efficient in inference than an autoregressive model, and has high potentials in industrial applications. Compared with autoregressive generation, NAT faces two major challenges: 1) determining the output length, and 2) handling multi-modality, i.e., generating a coherent text when multiple translations are plausible.

Gu et al. (2018) propose a token-fertility approach that predicts the repetition of source tokens to obtain the target length. Lee, Mansimov, and Cho (2018) predict the length difference between the source and target sequences.

The Insertion Transformer (Stern et al. 2019) dynamically expands the canvas size (generation slots) through multiple rounds of inference. The Levenshtein Transformer (Gu, Wang, and Zhao 2019) generates a sentence by explicitly predicting edit actions, such as insertion and deletion. Alternatively, an empty token can be introduced into the vocabulary for generating a sentence of appropriate length (Graves et al. 2006; Saharia et al. 2020).

To address the second challenge, Gu et al. (2018) propose a knowledge distillation (KD) approach that first trains an autoregressive translation (AT) model, and then trains the NAT model by AT's output instead of the groundtruth. Zhou, Gu, and Neubig (2020) show that AT's output has fewer modes than the original corpus, and thus is easier for training NAT models.

Partially (non-)autoregressive generation is also widely used to mitigate the multi-modality issue, such as insertion-based methods (Stern et al. 2019), and the Levenshtein Transformer (Gu, Wang, and Zhao 2019). Ghazvininejad et al. (2019) propose to pretrain a conditional masked language model (CMLM) for NAT, where they progressively generate several new words conditioned on previous ones. Wang, Zhang, and Chen (2018) model the dependencies among the words in a chunk. Sun et al. (2019) propose to apply the linear-chain conditional random field on top of NAT predictions to capture token dependency. Their method is also partially autoregressive as beam search is required during inference.

Recently, Qian et al. (2021) propose the Glancing Transformer (GLAT), which trains NAT predictions in a curriculum learning. Specifically, their method is built upon CMLM, and the amount of masked out groundtruth is dependent on the model's current performance. Thus, the model learns easy tokens first, and gradually moves on to hard ones. This also alleviates the multi-modal problem.

A similar study to ours is Lee, Mansimov, and Cho (2018). They train two decoders: one is for the initial generation, and the other is applied iteratively for refinement. They also propose a deterministic variant of token fertility.

Different from previous work, we perform layer-wise prediction within a decoder. This is an insightful contribution, being the key to efficient inference, as previous evidence shows that decoder-level iterative refinement does not have an advantage over a carefully designed autoregressive model (Kasai et al. 2021). We also show that our layer-wise prediction awareness is agnostic to base NAT models, being a generic framework.

Our work requires deep supervision on the Transformer decoder. This is similar to the treatment in the depth-adaptive Transformer (DAT, Elbayad et al. 2020), where they apply deep supervision for an autoregressive model to allow "early exit," i.e., using a subset of decoding layers. Our NAT work is apparently different from DAT.

Discussion on autoregressiveness. At first glance, it is doubtful whether our DSLP is non-autoregressive, as we have layer-wise prediction. In statistics, *autoregression* means using past predictions for future ones, typically in a time series (Akaike 1969). Our DSLP differs from partially autoregressive models (Stern et al. 2019; Gu, Wang, and Zhao 2019; Ghazvininejad et al. 2019). We predict the output words simultaneously in the last layer, whereas its layer-wise prediction is an internal mechanism, similar to Transformer hidden states. Thus, DSLP should be considered as a full NAT model from the perspective of statistics.

More importantly, the motivation of NAT research is the inference efficiency for industrial applications. Being non-autoregressiveness itself is not the research goal. As demonstrated in experiments, our work achieves satisfactory quality–efficiency trade-off, as it does not cause much extra computational cost.

Methodology

In this section, we describe our model in detail. We first introduce a generic Transformer-based NAT model. Then, we present our layer-wise prediction-aware approach and the deeply supervised training. We also present an improved training scheme that takes groundtruth tokens as input and mixes them the layer-wise predictions.

Non-Autoregressive Transformer

Recent advances in neural machine translation are built upon the Transformer architecture (Vaswani et al. 2017). It is an
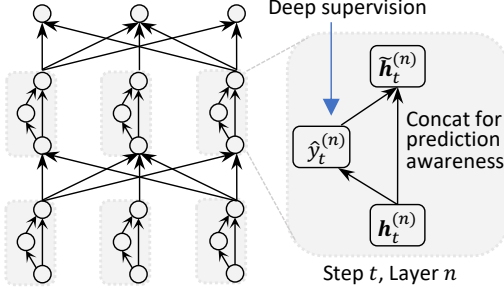
Figure 2: Overview of our Deeply Supervised Layer-wise Prediction-aware (DSLP) Transformer.

encoder–decoder neural network, where the encoder is responsible for representing the source sentences, and the decoder is used to generate the target translation.

Formally, let the source text be $\mathbf{x} = (\mathrm{x}_1, \cdots, \mathrm{x}_{T_x})$, where T_x is the length. The encoder first takes $\mathbf{x}$ as input and maps these discrete words[2] into vector representations by an embedding function, denoted by $\boldsymbol{e}_t^{(0)} = \mathrm{emb}(\mathrm{x}_t)$. Each Transformer layer performs multi-head attentions with a feed-forward layer to compute deep contextual representation, denoted by

$$\boldsymbol{e}_{1:T_x}^{(n)} = \mathrm{Layer}_{\mathrm{enc}}^{(n)}(\boldsymbol{e}_{1:T_x}^{(n-1)}) \qquad (1)$$

where $\mathrm{Layer}_{\mathrm{enc}}^{(n)}$ is the nth encoder layer. Such processing is repeated for every layer, and we take the last layer as the encoder's hidden representation, denoted by $\mathbf{E}$.

For non-autoregressive decoding, the first step is to determine the length T_y of the output sentence. This can be handled in various ways. For example, Gu et al. (2018) train a classifier to predict the length during inference, and in training, the length is given by the groundtruth. In latent alignment models (Saharia et al. 2020), an empty prediction is allowed, and T_y is typically set as kT_x for a constant k. Our DSLP approach can be built upon multiple base models, and the output length is handled accordingly.

For the decoder input $\boldsymbol{h}_t^{(0)}$, we follow Saharia et al. (2020) and feed the embedding of a special token s for every step. In our preliminary experiments, this treatment is on par with copying encoder representations as the decoder input (Gu et al. 2018; Wei et al. 2019).

The decoder processes the information in a similar way to the encoder, except that it has an additional encoder–decoder attention to obtain input information. Thus, a decoding layer can be represented by

$$\boldsymbol{h}_{1:T_y}^{(n)} = \mathrm{Layer}_{\mathrm{dec}}^{(n)}(\boldsymbol{h}_{1:T_y}^{(n-1)}, \mathbf{E}) \qquad (2)$$

where $\mathrm{Layer}_{\mathrm{dec}}^{(n)}$ is the nth decoding layer.

Finally, NAT uses a softmax layer to predict the target words based on the decoder's last hidden states, given by

$$p(\mathrm{y}_t|\boldsymbol{h}_t^{(N)}) = \mathrm{softmax}(\mathbf{W}\boldsymbol{h}_t^{(N)}) \qquad (3)$$

---

[2]We adopt BPE segmentation, and strictly speaking, they should be tokens. For clarity, we use *word* and *token* interchangeably in the paper.

where N is the number of decoding layers. In this way, the output words $\mathrm{y}_1, \cdots, \mathrm{y}_{T_y}$ are predicted simultaneously in a non-autoregressive fashion.

The training objective is to maximize the log-likelihood

$$\mathcal{L} = \sum_{t=1}^{T_y} \log p(\mathrm{y}_t|\boldsymbol{h}_t^{(N)}) \qquad (4)$$

Layer-Wise Prediction-Awareness

As we observe from (3), a conventional NAT model predicts a word y_t without being aware of the other words $\mathrm{y}_1, \cdots, \mathrm{y}_{t-1}, \mathrm{y}_{t+1}, \cdots, \mathrm{y}_{T_y}$. This is undesired when the target distribution is "multi-modal," i.e., multiple outputs are appropriate given an input.

To this end, we propose a layer-wise prediction-aware decoder that predict the output sequence at every layer. Such prediction serves as a tentative translation, and is fed to the next layer for further processing.

Consider the tth step of the nth decoding layer. We perform linear transformation on the conventional Transformer hidden state $\boldsymbol{h}_t^{(n)}$, and use softmax to predict the probability of words:

$$p(\mathrm{y}_t^{(n)}|\boldsymbol{h}_t^{(n)}) = \mathrm{softmax}(\mathbf{W}\boldsymbol{h}_t^{(n)}) \qquad (5)$$

Then, we obtain the most probable word by

$$\widehat{\mathrm{y}}_t^{(n)} = \mathrm{argmax}\, p(\mathrm{y}_t^{(n)}|\boldsymbol{h}_t^{(n)}) \qquad (6)$$

Its embedding is concatenated with $\boldsymbol{h}_t^{(n)}$ and further processed by a linear layer

$$\widetilde{\boldsymbol{h}}_t^{(n)} = \mathbf{W}_c[\boldsymbol{h}_t^{(n)}; \mathrm{emb}(\widehat{\mathrm{y}}_t^{(n)})] \qquad (7)$$

where $\mathbf{W}_c$ is a weight matrix. $\widetilde{\boldsymbol{h}}_t^{(n)}$ is an updated prediction-aware hidden state, fed to the next layer. Notice that the last layer does not require such an update, and the final output is directly given by (5).

In this way, the prediction at the $(n+1)$th layer is aware of all the predictions at the nth layer, due to the Transformer attention. This serves as a calibration mechanism that can revise the tentative generation through multiple decoding layers.

Deeply Supervised Training

We further propose to train our layer-wise prediction-aware Transformer with deep supervision.

Typically, NAT models are trained with the supervision at the last layer. With such training, however, our intermediate predictions are not grounded to the desired output sentence, and such prediction-awareness becomes less meaningful.

Specifically, we apply maximum likelihood estimation to the prediction of every decoding layer, given by

$$\mathcal{L}_D = \sum_{n=1}^{N} \sum_{t=1}^{T_y} \log p(\mathrm{y}_t^{(n)}|\widetilde{\boldsymbol{h}}_t^{(n)}) \qquad (8)$$

This layer-wise deep supervision ensures that our DSLP predicts the target output (although imperfect) at every decoding layer. By feeding back such prediction, the model is able to calibration the words of different time steps, alleviating the weakness of NAT models. Figure 2 presents an overview of our DSLP.

Mixing Predictions and Groundtruth in Training

During training, the layer-wise predictions can still be of low quality despite deep supervision, in which case the next layers may not be well trained. As a remedy, we propose to partially feed the groundtruth to intermediate layers.

As opposed to (7), we compute the hidden states of the next layer as,

$$\widetilde{\boldsymbol{h}}_t^{(n)} = \mathbf{W}_c[\boldsymbol{h}_t^{(n)}; \mathrm{emb}(\overline{\mathrm{y}}_t^{(n)})] \tag{9}$$

where $\overline{\mathrm{y}}_t^{(n)} = s_t \mathrm{y}_t + (1 - s_t)\widehat{\mathrm{y}}_t^{(n)}$, and $s_t \sim \mathrm{Bernoulli}(\lambda)$. λ is a hyperparameter controlling the amount of mixed groundtruth tokens, referred to as the mixing ratio. Note that s_t is not dependent on the layer number n. Otherwise, we may feed the model with all the groundtruth tokens during training and having low performance.

Although our DSLP is generic to non-autoregressive models, the proposed mix training scheme needs to be adapted to the base NAT model accordingly.

For the vanilla NAT (Gu et al. 2018), we mask out the observed groundtruth tokens in the loss, which prevents the model from simply copying tokens. This is similar to masked language models (Devlin et al. 2019; Ghazvininejad et al. 2019).[3]

For the connectionist temporal classification (CTC, Graves et al. 2006) model, the predictions are usually longer than the groundtruth. This is because CTC allows predicting extra tokens (including repeated tokens and empty tokens) to generate sentences of different lengths, and the probability of generating the groundtruth sequence $\mathrm{y}_{1:T_y}$ is computed by marginalizing all valid alignments.[4] The length discrepancy prevents us from directly replacing a predicted token with a groundtruth token. Alternatively, we regard the best alignment given by the model as the pseudo-groundtruth. This is similar to how Gu and Kong (2021) combine CTC with the Glancing Transformer (GLAT, Qian et al. 2021).

The mixed training does not improve the models that have already used groundtruth as input during training, such as the conditional masked language model (CMLM, Ghazvininejad et al. 2019) and GLAT. This is because further feeding groundtruth tokens interferes with their training schemes.

Our mixed training is also similar to the scheduled sampling (Bengio et al. 2015), which is proposed to improve the training of autoregressive models.

Inference

The inference of our DSLP resembles its training process: the predicted output of every layer is fed to the next layer,

---

[3]Empirically, we find that masking improves the BLEU score by 0.2 on WMT 14' EN→DE. However, it is not our main focus, and our mixed training with masking improves the vanilla NAT-based DSLP model by 1.4 BLEU (see Table 1).

[4]An alignment is an expanded sequence that can be reduced to the groundtruth sequence. For example, "␣␣aabb␣c" is a valid alignment to the sequence "abc", where "␣" represents an empty token. The marginalization of all valid alignments is computed efficiently via dynamic programming. More details can be found in Section 3 of Graves et al. (2006).

and we take the last layer as the final output. It should be pointed out that our layer-wise prediction does not introduce much extra computational cost. We will show that, despite the high translation quality of DSLP, the time difference between DSLP and a vanilla NAT model is negligible.

Experiments

Experimental Setup

Datasets. We evaluated our models on benchmark translation datasets: WMT'14 English–German (4.0M sentence pairs) and WMT'16 English–Romanian (610K pairs). For fair comparison, we obtained the preprocessed corpus (tokenization and vocabulary) released by previous work: Zhou, Gu, and Neubig (2020) for WMT'14 EN–DE, and Lee, Mansimov, and Cho (2018) for WMT'16 EN–RO. We consider both translation directions, and in total, we have 4 translation tasks.

Hyperparameters. We mostly followed the standard hyperparameters used in NAT research. We used the Transformer-base (Vaswani et al. 2017) configuration. To train the models, we used a batch size of 128K tokens for EN–DE and 32K tokens for EN–RO, with a maximum 300K updates. For regularization, we set the dropout rate to 0.1 for EN–DE and 0.3 for EN–RO. For the mixed training, we used a fixed mixing ratio λ and set it to 0.3. More details can be found in our GitHub repository (Footnote 1).

Knowledge Distillation. It is a common technique to train NAT by an autoregressive model's output. Previous evidence shows that this largely improves NAT performance (Gu et al. 2018; Lee, Mansimov, and Cho 2018; Stern et al. 2019). In our study, we also adopted knowledge distillation from an autoregressive Transformer-base model for all baselines and our DSLP variants.

Evaluation Metrics. For translation quality, we computed BLEU (Papineni et al. 2002) scores over tokenized sentences. To measure inference latency, we used a single Nvidia V100 GPU and performed inference with one sentence at a time. This mimics a deployed NMT system in the industry. Our models were implemented and evaluated with the open-source toolkit `FairSeq` (Ott et al. 2019).

Base Models. Our DSLP can be applied to various base NAT models. To evaluate its generality, we consider the following models: 1) Vanilla NAT (Gu et al. 2018), which is the foundation of NAT models. 2) Partially non-autoregressive generation based on a conditional masked language model (CMLM, Ghazvininejad et al. 2019), which progressively generates several words of a sentence. CMLM_k refers to k iterations. 3) Glancing Transformer (GLAT, Qian et al. 2021), which progressively trains NAT word predictors in a curriculum learning fashion. 4) Connectionist temporal classification (CTC, Graves et al. 2006), which allows empty tokens and performs dynamic programming for marginalization of latent alignment. With these base models, we would be able to evaluate DSLP with a wide range of prediction schemas and training objectives.

Row#	Model	WMT'14		WMT'16		Latency (ms)	Speedup
		EN–DE	DE–EN	EN–RO	RO–EN		
1	Transformer (teacher)	27.48	31.21	33.70	34.05	326.80	1×
2	CMLM$_1$	19.91	22.69	27.60	29.00	20.91	15.6×
3	w/ DSLP	**21.76**	**25.30**	**30.29**	**30.89**	21.76	15.0×
4	GLAT	25.02	29.63	31.33	32.43	20.91	15.6×
5	w/ DSLP	**25.69**	**29.90**	**32.36**	**33.06**	21.73	14.9×
6	Vanilla NAT	21.18	24.93	29.15	29.69	20.86	15.7×
7	w/ DSLP	22.72	25.83	30.48	31.46	22.12	14.8×
8	w/ DSLP & Mixed Training	**24.17**	**28.63**	**31.49**	**32.64**	22.12	14.8×
9	CTC	25.72	29.89	32.89	33.79	21.04	15.5×
10	w/ DSLP	26.85	31.16	33.85	34.24	22.06	14.8×
11	w/ DSLP & Mixed Training	**27.02**	**31.61**	**34.17**	**34.60**	22.06	14.8×
	Average Improvement	1.34	1.38	1.58	1.28	–	–

Table 1: Applying DSLP to different base NAT models, which shows the generality of our approach. Translation quality is evaluated in BLEU. Latency is the processing time (in milliseconds) of a single sentence. Speedup is relative to an autoregressive model. All results are based on our implementation. CMLM$_k$ refers to k iterations of progressive generation. Here, we consider $k = 1$, as more iterations make CMLM closer to autoregressive models.

Category	Row#	Model	WMT'14		WMT'16		Speedup
			EN–DE	DE–EN	EN–RO	RO–EN	
Autoregressive	1	Transformer	27.48	31.21	33.70	34.05	1×
Iterative	2	Iterative Refinement (Lee, Mansimov, and Cho 2018)	21.61	25.48	29.32	30.19	2.0×
	3	Blockwise (Stern, Shazeer, and Uszkoreit 2018)	27.40	–	–	–	3.0×
	4	Insertion Transformer (Stern et al. 2019)	27.41	–	–	–	4.8×
	5	Levenshtein Transformer (Gu, Wang, and Zhao 2019)	27.27	–	–	–	4.0×
	6	CMLM$_{10}$ (Ghazvininejad et al. 2019)	27.03	30.53	33.08	33.08	2.6×[‡]
	7	Imputer (Saharia et al. 2020)	**28.2**	**31.8**	**34.4**	34.1	3.9×
	8	DisCO (Kasai et al. 2021)	27.34	31.31	33.22	33.25	3.5×[†]
Non-iterative	9	Vanilla NAT (Gu et al. 2018)	17.69	21.47	27.29	29.06	15.6×
	10	DCRF (Sun et al. 2019)	23.44	27.22	–	–	10.4×
	11	CMLM$_1$ (Ghazvininejad et al. 2019)	18.05	21.83	27.32	28.20	15.6×[‡]
	12	AXE (Ghazvininejad et al. 2020)	23.53	27.90	30.75	31.54	15.3×
	13	CTC (Saharia et al. 2020)	25.7	28.1	32.2	31.6	18.6×
	14	CNAT (Bao et al. 2021)	25.67	29.36	–	–	10.4×
	15	REDER (Zheng et al. 2021)	26.70	30.68	33.10	33.23	15.5×
	16	GLAT (Qian et al. 2021)	25.21	29.84	31.19	32.04	15.3×
	17	CTC + GLAT (Qian et al. 2021)	26.39	29.54	32.79	33.84	14.6×
	18	CTC + VAE (Gu and Kong 2021)	27.49	31.10	33.79	33.87	16.5×
	19	CTC + GLAT (Gu and Kong 2021)	27.20	31.39	33.71	34.16	16.8×
Ours	20	CTC w/ DSLP & Mixed Training	27.02	31.61	34.17	**34.60**	14.8×

Table 2: Comparing our model with state-of-the-art NAT models. Results of prior work are quoted from respective papers. [†]Estimated from the plot in the previous paper. [‡]Given by our own implementation, as it is not available in the original paper.

Main Results

Generality of DSLP. Table 1 shows the performance of our DSLP compared with base models: Vanilla NAT, CMLM, GLAT, and CTC. Comparing with results published in previous papers (quoted in Table 2), our replications mostly match the previous work, except that our vanilla NAT outperforms the implementation of Gu et al. (2018). It is probably due to the engineering efforts by the community for NAT in the past years. The results on base models implicate that our implementation is fair for the study on DSLP.

We apply DSLP to all baseline models. As seen, DSLP consistently achieves higher performance for every base model and every translation task. The average improvement is more than 1 BLEU point, which is a considerable improvement over strong baselines for machine translation.

For the training of vanilla NAT- and CTC-based DSLP, we mix the layer-wise predictions with groundtruth tokens.

We do not apply the mixed training on CMLM and GLAT, as they have already used groundtruth tokens in training, and further feeding groundtruth causes potential conflicts.

We show the improvement of the mixed training in Table 1 (Lines 8 & 11). We observe mixed training consistently improves the performance, with an average of 0.97 BLEU improvement compared with respective DSLP variants.

When combined with the weak vanilla NAT model, DSLP with mixed training outperforms the strong baseline model, the Glancing Transformer (Qian et al. 2021), on EN→RO, and RO→EN datasets. When combining with the strong CTC model, the mixed training scheme further improves the performance consistently, even though the CTC with DSLP model has already achieved superb results.

Comparing with the State of the Art. We compare our best variant (CTC w/ DSLP & Mixed Training) with previous state-of-the-art NAT models in Table 2. Profoundly, our
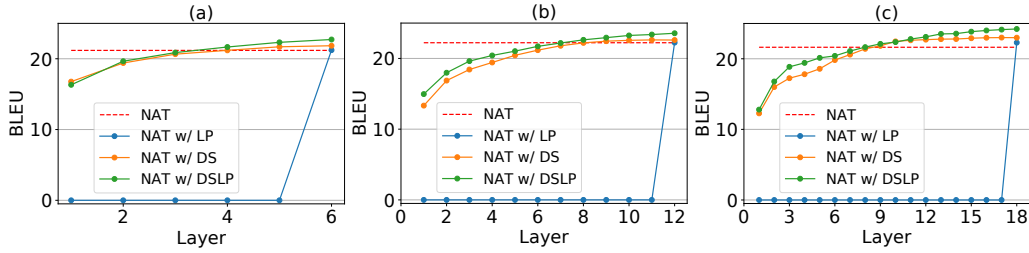
Figure 3: Layer-wise performance of a fixed, trained vanilla NAT model. Decoding layers: 6 for (a), 12 for (b), and 18 for (c).

	6	12	18
Vanilla NAT	21.18	22.22	21.63
w/ LP	21.22(+0.04)	22.24(+0.02)	22.29(+0.66)
w/ DS	21.84(+0.66)	22.66(+0.38)	22.99(+1.36)
w/ DSLP	**22.72**(+1.54)	**23.56**(+1.34)	**24.22**(+2.59)

Table 3: Ablation study on layer-wise prediction (LP) and deep supervision (DS). We tested models of 6 layers, 12 layers, and 18 layers. The number in bracket shows the improvement over vanilla NAT.

approach achieves close results to the autoregressive (AT) teacher model on the WMT 14' EN→DE dataset, and even outperforms AT on other three translation datasets.

Compared with iterative methods (Line 2–8, Table 2), our model produces very competitive translation quality while being approximately 4 times faster in inference. Compared with non-iterative methods (Line 9–19, Table 2), our CTC w/ DSLP & Mixed Training outperforms all existing systems on three datasets (DE→EN, EN→RO, and RO→EN), while only costing extra 4–6% latency.

Analysis

In this part, we present in-depth analysis on DSLP. Our development was mainly conducted on vanilla NAT because we would like to rule out complications brought by various NAT models. It is also more efficient (and greener). Therefore, the base model is vanilla NAT for this subsection (unless otherwise stated).

Ablation Study on DSLP. Table 3 presents an ablation study on the layer-wise prediction (LP) and deep supervision (DS). We observe that LP or DS alone shows little improvement over the vanilla NAT. This is understandable since LP alone does not guarantee that intermediate predictions are grounded to target, and such awareness is mostly ineffective. DS alone does not have a calibration mechanism of different time steps. It nevertheless improves the performance to a small extent when the model is deep. This is because deep Transformers are difficult to train (Huang et al. 2020), as the vanilla NAT's performance drops by 0.59 from 12 layers to 18 layers; DS may help training a deep structure.

Our DSLP significantly outperforms LP and DS on all the settings. Compared with DS, our full DSLP shows more improvement when the network is deep, as more layers allow larger calibration capacity of DSLP.

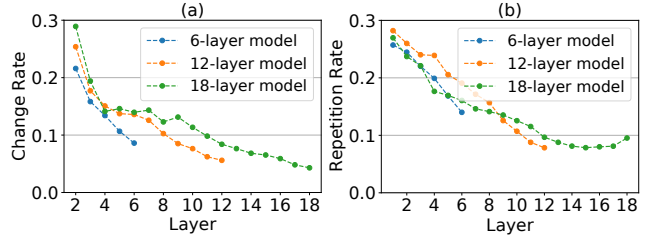**Layer-Wise Performance.** Since our model makes inter-

Figure 4: (a) Change rate for vanilla NAT w/ DSLP. The first layer is excluded as it does not perform editing. (b) Word repetition rate in each layer of vanilla NAT w/ DSLP.

mediate predictions, we are curious how the performance evolves through the decoding layers. We took a fixed, trained model, and tested the BLEU score of every layer. In Figure 3, we show the performance dynamics of a 6-, 12-, and 18-decoding-layer models.

With only LP, the model achieves zero BLEU for all intermediate layers. This verifies that LP alone does not provide meaningful predictions at intermediate layers, leading to lower performance than our full DSLP.

On the other hand, DS is able to achieve reasonable BLEU scores at every decoding layer. However, their BLEU scores are consistently lower than those of DSLP.

Notably, DS shows no improvement (or sometimes even decreases) at the last few layers. By contrast, the performance of DSLP improves steadily by the calibration over multiple layers.

Change Rate. Since our DSLP performs layer-wise calibration, we investigate how much word changing is performed in every layer. Figure 4a shows the percentage of changed words, where we also consider three variants: 6-layer, 12-layer, and 18-layer models.

We observe that, in each model, early layers perform more changes, and the change rate decreases gradually. This is reasonable because the last few layers have produced high-quality translations, requiring less calibration.

Word Repetition. NAT models often generate repetitive tokens, because of the weakness in dealing with multimodality (Gu et al. 2018), as expressions with slight different wordings can be thought of as a mode of the target distribution. Following Ghazvininejad et al. (2019) and Chan et al. (2020), we measure the fraction of repetitive tokens in each layer, shown in Figure 4b.

As seen, the repetition rate is high at early layers. How-

	Source: Würde ich Gleichgesinnte erst nach Einbruch der Dunkelheit treffen können?

Example 1

Source: Würde ich Gleichgesinnte erst nach Einbruch der Dunkelheit treffen können?
Reference: Would I be forced to meet like-minded people only after dark?
Generation:

Step:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Layer 1:	Would	I	meet	meet	meet	minded	minded	minded	after	dark-	dark-	dark-	collapsed	?
Layer 2:	Would	I	only	meet	like	minded	minded	minded	after	dark-	ness	dark-	collapsed	?
Layer 3:	Would	I	meet	meet	like	@-@	minded	people	after	dark-	ness	dark-	collapsed	?
Layer 4:	Would	I	meet	meet	like	@-@	minded	people	after	dark-	dark-	ness	collapsed	?
Layer 5:	Would	I	only	meet	like	@-@	minded	people	after	dark-	ness	ness	collapsed	?
Layer 6:	Would	I	only	meet	like	@-@	minded	people	after	dark-	dark-	ness	collapsed	?

Example 2

Source: Konflikte sind unvermeidbar, sie müssen aber im Streben nach einem gemeinsamen Weg überwunden werden.
Reference: Conflict is inevitable, but must be overcome by the desire to walk together.
Generation:

Step:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Layer 1:	Conf-	lic-	ts	are	inevitable	but	they	must	be	overcome	the	pursuit	for	a	common	path	.
Layer 2:	Conf-	lic-	ts	inevitable	,	but	they	must	be	overcome	the	pursuit	of	a	common	path	.
Layer 3:	Conf-	lic-	ts	inevitable	inevitable	but	they	they	must	overcome	in	pursuit	for	a	common	path	.
Layer 4:	Conf-	lic-	ts	inevitable	inevitable	but	but	they	must	overcome	overcome	pursuit	for	a	common	path	.
Layer 5:	Conf-	lic-	ts	are	inevitable	,	but	they	must	be	the	pursuit	for	a	common	path	.
Layer 6:	Conf-	lic-	ts	are	inevitable	,	but	they	must	be	the	pursuit	for	a	common	path	.

Figure 5: Layer-wise predictions of our DSLP w/ vanilla NAT with examples in the test set of WMT'14 DE–EN translation. In our model, we use BPE segmentation, where prefixes are marked by - for the convenience of reading. A hyphen token is represented by "@-@."
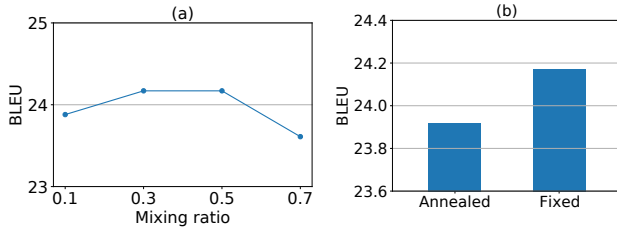

Figure 6: (a) Performance with different mixing ratios. (b) Impact of annealing the mixing ratio to zero during training.

ever, it decreases drastically along the decoding layers. In general, the repetition rate is less than 15% at the final layer for all three models. We also observe that a deeper model (12- or 18-layer) has fewer repetitions than a shallow model (6-layer). This further confirms that our layer-wise prediction is able to perform meaningful calibration and address the weakness of NAT models.

Mixed Training. We tune the mixing ratio λ and show the results in Figure 6a. We find that a mixing ratio around 0.3 performs well, but the performance decreases significantly when the ratio is too high. This is because the intermediate predictions are important for the prediction awareness of follow-up layers; therefore the layer-wise predictions can not be fully replaced with groundtruth tokens.

To better understand how the mixed training scheme works, we let the ratio of mixed groundtruth tokens anneal to zero throughout the training. In our experiments, we observe that with the annealed mixing ratio, the training eventually degenerates to that of the standard DSLP model, which results in worse final performance (see Figure 6b). This phenomenon suggests that the intermediate predictions are not perfect during the entire training; constantly mixing the groundtruth provides correction to layer-wise predictions, which results in a better training. Based on our analysis, we set the mixing ratio as 0.3 during the entire training.

Case Study. In Figure 5, we demonstrate a case study on the test set of WMT'14 DE–EN dataset with our DSLP.

Here, we present German-to-English translation for accessibility concerns.

As seen, the prediction by lower layers is of low quality, and calibration is indeed performed through the decoding layers. In Example 1, the words *meet*, *minded*, and *minded* are repeated several times in Layer 1. This is expected, as no calibration has been performed in this layer. Then, our layer-wise prediction-aware decoder gradually revises out most of the repetitions. In Example 2, the phrase *must be overcome the pursuit* is not fluent. The grammatical error is also corrected by the calibration process.

Admittedly, the output of our DSLP may not be prefect. The last layer in Example 1 still has one repetition of the word *dark-*, and in Example 2, the word *overcome* is mistakenly dropped. We conjecture that this is because the target length is incorrectly predicted. This can probably be addressed with more decoding layers to find a paraphrase, or using other decoding methods such as noisy parallel decoding (NPD, Gu et al. 2018). Like most translation models, our DSLP cannot perform non-literal translation well, and thus its output of Example 2 does not resemble the reference.

In general, our case study confirms that the proposed DSLP model indeed performs calibration by layer-wise prediction-awareness, and that the translation quality is generally improved layer-by-layer.

Conclusion

In this paper, we propose a deeply supervised, layer-wise prediction-aware Transformer (DSLP) for NAT. Our approach is generic and can be combined with different NAT models. We conducted experiments on four machine translation tasks with four base models (vanilla NAT, CMLM, GLAT, and CTC). The results show that our method consistently improves the translation quality with an average of more than 1 BLEU score, which is considered a large improvement over strong baselines. Our best variant achieves better BLEU scores than its autoregressive teacher model on three of four benchmark datasets, while accelerating the inference by 14.8x times.

Acknowledgments

References

Akaike, H. 1969. Fitting autoregressive models for prediction. *Annals of the Institute of Statistical Mathematics*, 21(1): 243–247.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Bao, Y.; Huang, S.; Xiao, T.; Wang, D.; Dai, X.; and Chen, J. 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*, 5749–5759.

Bengio, S.; Vinyals, O.; Jaitly, N.; and Shazeer, N. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, 1171–1179.

Chan, W.; Saharia, C.; Hinton, G.; Norouzi, M.; and Jaitly, N. 2020. Imputer: Sequence modelling via imputation and dynamic programming. In *Proceedings of the 37th International Conference on Machine Learning*, 1403–1413.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4171–4186.

Elbayad, M.; Gu, J.; Grave, E.; and Auli, M. 2020. Depth-adaptive Transformer. In *International Conference on Learning Representations*.

Ghazvininejad, M.; Karpukhin, V.; Zettlemoyer, L.; and Levy, O. 2020. Aligned cross entropy for non-autoregressive machine translation. In *Proceedings of the International Conference on Machine Learning*, 3515–3523.

Ghazvininejad, M.; Levy, O.; Liu, Y.; and Zettlemoyer, L. 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*, 6112–6121.

Graves, A.; Fernández, S.; Gomez, F.; and Schmidhuber, J. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, 369–376.

Gu, J.; Bradbury, J.; Xiong, C.; Li, V. O.; and Socher, R. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Gu, J.; and Kong, X. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 120–133.

Gu, J.; Wang, C.; and Zhao, J. 2019. Levenshtein Transformer. In *Advances in Neural Information Processing Systems*.

Hao, Y.; He, S.; Jiao, W.; Tu, Z.; Lyu, M.; and Wang, X. 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*.

Huang, X. S.; Perez, F.; Ba, J.; and Volkovs, M. 2020. Improving Transformer optimization through better initialization. In *Proceedings of the International Conference on Machine Learning*, 4475–4483.

Kasai, J.; Pappas, N.; Peng, H.; Cross, J.; and Smith, N. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *International Conference on Learning Representations*.

Lee, J.; Mansimov, E.; and Cho, K. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1173–1182.

Ott, M.; Edunov, S.; Baevski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; and Auli, M. 2019. FairSeq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 48–53.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318.

Qian, L.; Zhou, H.; Bao, Y.; Wang, M.; Qiu, L.; Zhang, W.; Yu, Y.; and Li, L. 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*, 1993–2003.

Saharia, C.; Chan, W.; Saxena, S.; and Norouzi, M. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 1098–1108.

Stern, M.; Chan, W.; Kiros, J.; and Uszkoreit, J. 2019. Insertion Transformer: Flexible sequence generation via insertion operations. In *Proceedings of the International Conference on Machine Learning*, 5976–5985.

Stern, M.; Shazeer, N.; and Uszkoreit, J. 2018. Block-wise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems*, 10107–10116.

Sun, Z.; Li, Z.; Wang, H.; He, D.; Lin, Z.; and Deng, Z. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems*, 3011–3020.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you Need. In *Advances in Neural Information Processing Systems*.

Wang, C.; Zhang, J.; and Chen, H. 2018. Semi-autoregressive neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 479–488.

Wei, B.; Wang, M.; Zhou, H.; Lin, J.; and Sun, X. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1304–1312.

Zheng, Z.; Zhou, H.; Huang, S.; Chen, J.; Xu, J.; and Li, L. 2021. Duplex sequence-to-sequence learning for reversible machine translation. In *Advances on Neural Information Processing Systems*.

Zhou, C.; Gu, J.; and Neubig, G. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.