Enriching Non-Autoregressive Transformer with Syntactic and Semantic Structures for Neural Machine Translation

Ye Liu[1], **Yao Wan**[2], **Jian-Guo Zhang**[1], **Wenting Zhao**[1], **Philip S. Yu**[1]
[1]Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA
[2]School of Computer Sci. & Tech., Huazhong University of Science and Technology, Wuhan, China
{yliu279, jzhan51, wzhao41, psyu}@uic.edu, wanyao@hust.edu.cn

Abstract

The non-autoregressive models have boosted the efficiency of neural machine translation through parallelized decoding at the cost of effectiveness, when comparing with the autoregressive counterparts. In this paper, we claim that the syntactic and semantic structures among natural language are critical for non-autoregressive machine translation and can further improve the performance. However, these structures are rarely considered in existing non-autoregressive models. Inspired by this intuition, we propose to incorporate the explicit syntactic and semantic structures of languages into a non-autoregressive Transformer, for the task of neural machine translation. Moreover, we also consider the intermediate latent alignment within target sentences to better learn the long-term token dependencies. Experimental results on two real-world datasets (i.e., WMT14 En-De and WMT16 En-Ro) show that our model achieves a significantly faster speed, as well as keeps the translation quality when compared with several state-of-the-art non-autoregressive models.

1 Introduction

Recently, non-autoregressive models (Gu et al., 2018), which aim to enable the parallel generation of output tokens without sacrificing translation quality, have attracted much attention. Although the non-autoregressive models have considerably sped up the inference process for real-time neural machine translation (NMT) (Gu et al., 2018), their performance is considerably worse than that of autoregressive counterparts. Most previous works attribute the poor performance to the inevitable conditional independence issue when predicting target tokens, and many variants have been proposed to solve it. For example, several techniques in non-autoregressive models are investigated to mitigate the trade-off between speedup and performance, including iterative refinement (Lee et al., 2018), insertion-based models (Chan et al., 2019; Stern et al., 2019), latent variable based models (Kaiser et al., 2018; Shu et al., 2020), CTC models (Libovický and Helcl, 2018; Saharia et al., 2020), alternative loss function based models (Wei et al., 2019; Wang et al., 2019; Shao et al., 2020), and masked language models (Ghazvininejad et al., 2019, 2020).

Although these works have tried to narrow the performance gap between autoregressive and non-autoregressive models, and have achieved some improvements on machine translation, the non-autoregressive models still suffer from syntactic and semantic limitations. That is, the translations of non-autoregressive models tend to contain incoherent phrases (e.g., repetitive words), and some informative tokens on the source side are absent. It is because in non-autoregressive models, each token in the target sentence is generated independently. Consequently, it will cause the multimodality issue, i.e., the non-autoregressive models cannot model the multimodal distribution of target sequences properly (Gu et al., 2018).

One key observation to mitigate the syntactic and semantic error is that source and target translated sentences follow the same structure, which can be reflected from Part-Of-Speech (POS) tags and Named Entity Recognition (NER) labels. Briefly, POS, which aims to assign labels to words to indicate their categories by considering the long-distance structure of sentences, can help the model learn the syntactic structure to avoid generating the repetitive words. Likewise, NER, which discovers the proper nouns and verbs of sentences, naturally helps the model recognize some meaningful semantic tokens that may improve translation quality. This observation motivates us to leverage the syntactic as well as semantic structures of natural language to improve the performance of non-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1235–1244
April 19 - 23, 2021.

Table 1: A motivating example on WMT14 En→De dataset. English with POS|NER and its corresponding German translation with POS|NER. The Blue labels show the same tags, while the Red labels show the different tags in two languages.

EN:	A	republican	strategy	to	counter	the	rel-election	of	Obama	.
EN POS:	DET	ADJ	NOUN	PART	VERB	DET	NOUN	ADP	PROPN	PUNCT
EN NER:	O	B_NORP	O	O	O	O	O	O	B_PERSON	O
DE:	Eine	republikanische	strategie	gegen	die	wiederwahl	Obama	.		
DE POS:	DET	ADJ	NOUN	ADP	DET	NOUN	PROPN	PUNCT		
EN NER:	O	B_NORP	O	O	O	O	B_PERSON	O		

autoregressive NMT. We present a motivating example in Table 1 to better illustrate our idea. From this table, we can find that although the words are altered dramatically from the English sentence to its German translation, the corresponding POS and NER tags still remain similar. For example, most POS tags are identical and follow the same pattern, except that PART, VERB, and ADP in the English do not match the German ADP, while the NER tags are exactly the same in both sentences.

In this paper, we propose an end-to-end Syntactic and semantic structure-aware Non-Autoregressive Transformer model (**SNAT**) for NMT. We take the structure labels and words as inputs of the model. With the guidance of extra sentence structural information, the model greatly mitigates the multimodality issue's negative impact. The core contributions of this paper can be summarized as that we propose 1) a syntax and semantic structure-aware Transformer which takes sequential texts and the structural labels as input and generates words conditioned on the predicted structural labels, and 2) an intermediate alignment regularization which aligns the intermediate decoder layer with the target to capture coarse target information. We conduct experiments on four benchmark tasks over two datasets, including WMT14 En→De and WMT16 En→Ro. Experimental results indicate that our proposed method achieves competitive results compared with existing state-of-the-art non-autoregressive and autoregressive neural machine translation models, as well as significantly reduces the decoding time.

2 Background

Regardless of its convenience and effectiveness, the autoregressive decoding methods suffer two major drawbacks. One is that they cannot generate multiple tokens simultaneously, leading to inefficient use of parallel hardware such as GPUs. The other is that beam search has been found to output low-quality translation with large beam size and deteriorates when applied to larger search spaces. However, non-autoregressive transformer (NAT) could potentially address these issues. Particularly, they aim at speeding up decoding through removing the sequential dependencies within the target sentence and generating multiple target tokens in one pass, as indicated by the following equation:

$$P_{\mathbf{NAT}}(\mathbf{y}|\mathbf{x};\phi) = \prod_{t=1}^{m} p\left(y_t|\hat{\mathbf{x}},\mathbf{x};\phi\right), \quad (1)$$

where $\hat{\mathbf{x}} = \{\hat{x}_1,\ldots,\hat{x}_m\}$ is the copied source sentence. Since the conditional dependencies within the target sentence (y_t depends on $y_{<t}$) are removed from the decoder input, the decoder is unable to leverage the inherent sentence structure for prediction. Hence the decoder is supposed to figure out such target-side information by itself given the source-side information during training. This is a much more challenging task compared to the autoregressive counterparts. From our investigation, we find the NAT models fail to handle the target sentence generation well. It usually generates repetitive and semantically incoherent sentences with missing words. Therefore, strong conditional signals should be introduced as the decoder input to help the model better learn internal dependencies within a sentence.

3 Methodology

In this section, we present our model **SNAT** to incorporate the syntactic and semantic structure information into a NAT model as well as an intermediate latent space alignment within the target. Figure 1 gives an overview of the network structure of our proposed **SNAT**. In **SNAT**, the input sequence is segmented into sub-words by byte-pair

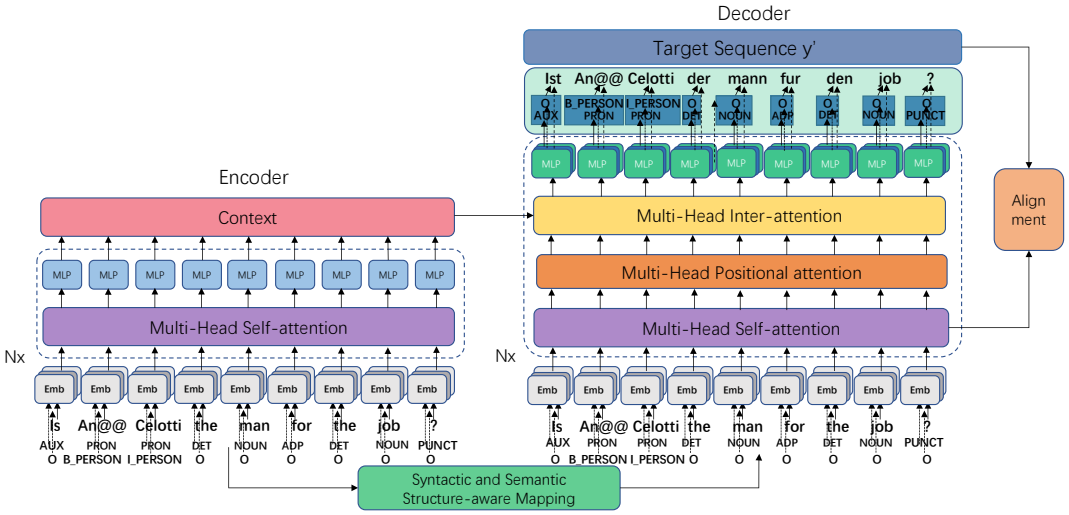

Figure 1: An overview of the proposed **SNAT** for neural machine translation.

tokenizer (Sennrich et al., 2016). In parallel, words in the input sequence are passed to POS and NER annotators to extract explicit syntactic and semantic structures, and the corresponding embeddings are aggregated by a linear layer to form the final syntax and semantic structure-aware embedding. The **SNAT** model copies the structured encoder input as the decoder input and generates the translated sentences and labels.

One of the most important properties of **SNAT** is that it naturally introduces syntactic and semantic information when taking the structure-aware information as inputs and generating both words and labels. More precisely, given a source sentence $\mathbf{x}$, as well as its label sequence $\mathbf{L_x}$, the conditional probability of a target translation $\mathbf{y}$ and its label sequence $\mathbf{L_y}$ is:

$$P_{\textbf{SNAT}}(\mathbf{y}, \mathbf{L_y}|\mathbf{x}, \mathbf{L_x}; \varphi) = \prod_{t=1}^{m} p\left(y_t, L_{y_t}|\hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}, \mathbf{x}, \mathbf{L_x}; \varphi\right), \quad (2)$$

where $\mathbf{x}$ and $\mathbf{L_x}$ are first fed into the encoder of **SNAT** model. $\hat{\mathbf{x}}$ and $\hat{\mathbf{L}}_\mathbf{x}$ with length m are syntactic and semantic structure-aware copying of word and label from encoder inputs, respectively. We show the details in the following sections.

3.1 Syntactic and Semantic Labeling

We use POS and NER to introduce the syntactic and semantic information existing in natural language, respectively. During the data pre-processing, each sentence is annotated into a semantic sequence using an open-source pre-trained semantic annotator. In particular, we take the Treebank style (Marcus et al., 1999) for POS and PropBank style (Palmer et al., 2005) for NER to annotate every token of input sequence with semantic labels. Given a specific sentence, there would be predicate-argument structures. Since the input sequence is segmented into subword units using byte-pair encoding (Sennrich et al., 2016), we assign the same label to all subwords tokenized from the same word. As shown in Figure 1, the word "*Ancelotti*" is tokenized as "*An@@*" and "*Celotti*". The corresponding POS tags are PRON and PRON while the corresponding NER tags are B_PERSON and I_PERSON. For the text "*Is An@@ Celotti the man for the job ?*", the corresponding POS tag set is {AUX, PRON, PRON, DET, NOUN, ADP, DET, NOUN, PUNCT} and the NER tag set is {O, B_PERSON, I_PERSON, O, O, O, O, O, O}. The data flow of the proposed model is also shown in Figure 1.

3.2 Encoder

Following Transformer (Vaswani et al., 2017), we use a stack of 6 identical Transformer blocks as encoder. In addition to the word embedding and position embedding in the traditional Transformer, we add structure-aware label embedding. The input to the encoder block is the addition of the normalized word, labels (NER and POS) and position embedding, which is represented as $\mathbf{H}^0 = [\mathbf{h}_1^0, \ldots, \mathbf{h}_n^0]$.

The input representation $\mathbf{H}^0 = [\mathbf{h}_1^0, \ldots, \mathbf{h}_n^0]$ is encoded into contextual layer representations through the Transformer blocks. For each layer, the layer representation $\mathbf{H}^l = [\mathbf{h}_1^l, \ldots, \mathbf{h}_n^l]$ is computed by the l-th layer Transformer block $\mathbf{H}^l = \text{Transformer}_l(\mathbf{H}^{l-1})$, $l \in \{1, 2, \ldots, 6\}$. In each

Transformer block, multiple self-attention heads are used to aggregate the output vectors of the previous layer. A general attention mechanism can be formulated as the weighted sum of the value vector $\mathbf{V}$ using the query vector $\mathbf{Q}$ and the key vector $\mathbf{K}$:

$$\mathrm{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{\text{model}}}}\right) \cdot \mathbf{V}, \quad (3)$$

where d_{model} represents the dimension of hidden representations. For self-attention, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are mappings of previous hidden representation by different linear functions, i.e., $\mathbf{Q} = \mathbf{H}^{l-1}\mathbf{W}_Q^l$, $\mathbf{K} = \mathbf{H}^{l-1}\mathbf{W}_K^l$, and $\mathbf{V} = \mathbf{H}^{l-1}\mathbf{W}_V^l$, respectively. At last, the encoder produces a final contextual representation $\mathbf{H}^6 = [\mathbf{h}_1^6, \ldots, \mathbf{h}_n^6]$, which is obtained from the last Transformer block.

3.3 Decoder

The decoder also consists of 6 identical Transformer blocks, but with several key differences from the encoder. More concretely, we denote the contextual representations in the i-th decoder layer is $\mathbf{Z}^i (1 \leq i \leq 6)$. The input to the decoder block as $\mathbf{Z}^0 = [\mathbf{z}_1^0, \ldots, \mathbf{z}_m^0]$, which is produced by the addition of the word, labels (NER and POS) copying from encoder input and positional embedding.

For the target side input $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}]$, most previous works simply copied partial source sentence with the length ratio $\frac{n}{m}$ where n refers to the source length and m is the target length as the decoder input. More concretely, the decoder input y_i at the i-th position is simply a copy of the $\lfloor \frac{n}{m} \times i \rfloor$th contextual representation, i.e., $x_{\lfloor \frac{n}{m} \times i \rfloor}$ from the encoder. From our investigation, in most cases, the gap between source length and target length is relatively small (e.g. 2). Therefore, it deletes or duplicates the copy of the last a few tokens of the source. If the last token is meaningful, the deletion will neglect important information. Otherwise, if the last token is trivial, multiple copies will add noise to the model.

Instead, we propose a syntactic and semantic structure-aware mapping method considering the POS and NER labels to construct the decoder inputs. Our model first picks out the informative words with NOUN and VERB POS tags, and the ones recognized as entities by the NER module. If the source length is longer than the target length, we retain all informative words, and randomly delete the rest of the words. On the other hand, if the source length is shorter than the target, we retain all words and randomly duplicate the informative words. The corresponding label of a word is also deleted or preserved. Moreover, by copying the similar structural words from the source, it can provide more information to the target input than just copying the source token, which is greatly different from the target token. The POS and NER labels of those structure-aware copied words from the source sentence are also copied as the decoder input. So by using the structure-aware mapping, we can assign $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}]$ as decoder input.

For positional attention which aims to learn the local word orders within the sentence (Gu et al., 2018), we set positional embedding (Vaswani et al., 2017) as both $\mathbf{Q}$ and $\mathbf{K}$, and hidden representations of the previous layer as $\mathbf{V}$.

For inter-attention, $\mathbf{Q}$ refers to hidden representations of the previous layer, whereas $\mathbf{K}$ and $\mathbf{V}$ are contextual vectors $\mathbf{H}^6$ from the encoder. We modify the attention mask so that it does not mask out the future tokens, and every token is dependent on both its preceding and succeeding tokens in every layer. Therefore, the generation of each token can use bi-directional attention. The position-wise Feed-Forward Network (FFN) is applied after multi-head attention in both encoder and decoder. It consists of two fully-connected layers and a layer normalization (Ba et al., 2016). The FFN takes $\mathbf{Z}^6$ as input and calculates the final representation $\mathbf{Z}^f$, which is used to predict the whole target sentence and label:

$$p\left(\mathbf{y} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}, \mathbf{x}, \mathbf{L}_\mathbf{x}\right) = f\left(\mathbf{Z}^f \mathbf{W}_w^\top + \mathbf{b}_w\right), \quad (4)$$

$$q\left(\mathbf{L}_\mathbf{y} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}, \mathbf{x}, \mathbf{L}_\mathbf{x}\right) = f\left(\mathbf{Z}^f \mathbf{W}_l^\top + \mathbf{b}_l\right), \quad (5)$$

where f is a GeLU activation function (Hendrycks and Gimpel, 2016). $\mathbf{W}_w$ and $\mathbf{W}_l$ are the token embedding and structural label embedding in the input representation, respectively. We use different FFNs for POS and NER labels. To avoid redundancy, we just use $q\left(\mathbf{L}_\mathbf{y} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}, \mathbf{x}, \mathbf{L}_\mathbf{x}\right)$ to represent the two predicted label likelihood in general.

3.4 Training

We use $(\mathbf{x}, \mathbf{L}_\mathbf{x}, \mathbf{y}^*, \mathbf{L}_\mathbf{y}^*)$ to denote a training instance. To introduce the label information, our proposed **SNAT** contains a discrete sequential latent variable $L_{y_{1:m}}$ with conditional posterior distribution $p(L_{y_{1:m}} | \hat{\mathbf{x}}, \hat{\mathbf{L}}_\mathbf{x}, \mathbf{x}, \mathbf{L}_\mathbf{x}; \varphi)$. It can be approximated using a proposal distribution $q(\mathbf{L}_\mathbf{y} \mid$

$\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}})$. The approximation also provides a variational bound for the maximum log-likelihood:

$$\begin{aligned}
&\log P_{\textbf{SNAT}} = \log \sum_{t=1}^{m} q\left(L_{y_t} | \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right) \\
&\times p\left(y_t | L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right) \\
&\geq \mathop{\mathbb{E}}_{L_{y_{1:m}} \sim q} \Big\{ \underbrace{\sum_{t=1}^{m} \log q\left(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right)}_{\text{Label likelihood}} \\
&+ \underbrace{\sum_{t=1}^{m} \log p\left(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right)}_{\text{Structure-aware word likelihood}} \Big\} + \mathcal{H}(q).
\end{aligned} \tag{6}$$

Note that, the resulting likelihood function, consisting of the two bracketed terms in Eq. (6), allows us to train the entire model in a supervised fashion. The inferred label can be utilized to train the label predicting model q and simultaneously supervise the structure-aware word model p. The label loss can be calculated by the cross-entropy $\mathcal{H}$ of $L^*_{y_t}$ and Eq. (5):

$$\mathcal{L}_{\text{label}} = \sum_{t=1}^{m} \mathcal{H}\left(L^*_{y_t}, \ q(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}})\right), \tag{7}$$

The structure-aware word likelihood is conditioned on the generation result of the label. Since the Eq. (4) does not depend on the predicted label, we propose to bring the structure-aware word mask $\mathbf{M}_{wl} \in \mathbb{R}^{|V_{word}| \times |V_{label}|}$, where $|V_{word}|$ and $|V_{label}|$ are vocabulary sizes of word and label, respectively. The mask $\mathbf{M}_{w_l}$ is defined as follows:

$$\mathbf{M}_{w_l}(i,j) = \begin{cases} 1, & \mathcal{A}(y_i) = label_j, \\ \epsilon, & \mathcal{A}(y_i) \neq label_j, \end{cases} \tag{8}$$

which can be obtained at the preprocessing stage, and $\mathcal{A}$ denotes the open-source pre-trained POS or NER annotator mentioned above. It aims to penalize the case when the word y_i does not belong to the label $label_j$ with ϵ, which is a small number defined within the range of $(0, 1)$ and will be tuned in our experiments. For example, the word "great" does not belong to VERB. The structure-aware word likelihood can be reformulated as:

$$\begin{aligned}
&p(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi) = p(y_t \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}) \\
&\times \mathbf{M}_{w_l} \times q(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}).
\end{aligned} \tag{9}$$

Consequently, the structure-aware word loss $\mathcal{L}_{\text{word}}$ is defined as the cross-entropy between true $p^{'}(y^*_t | L^*_{y_t})$ and predicted $p(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi)$, where $p^{'}(y^*_t | L^*_{y_t}) \in \mathbb{R}^{|V_{word}| \times |V_{label}|}$ is a matrix where only item at the index of $(y^*_t, L^*_{y_t})$ equals to 1, otherwise equals to 0. We reshape $p^{'}(y^*_t | L^*_{y_t})$ and $p(y_t | L_{y_t})$ to vectors when calculating the loss.

Intermediate Alignment Regularization One main problem of NAT is that the decoder generation process does not depend on the previously generated tokens. Based on the bidirectional nature of **SNAT** decoder, the token can depend on every token of the decoder input. However, since the input of decoder $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}]$ is the duplicate of encoder input $[\mathbf{x}, \mathbf{L}_{\mathbf{x}}]$, the generation depends on the encoder tokens rather than the target $\mathbf{y}^*$.

To solve this problem, we align the output of the intermediate layer of the decoder with the target. The alignment makes the generation of following layers dependent on the coarse target-side information instead of the mere encoder input. This alignment idea is inspired by (Guo et al., 2019), which directly feeds target-side tokens as inputs of the decoder by linearly mapping the source token embeddings to target embeddings. However, using one FFN layer to map different languages to the same space can hardly provide promising results. Thus, instead of aligning the input of decoder with the target, we use the intermediate layer of decoder to align with the target. In this case, our model avoids adding additional training parameters and manages to train the alignment together with **SNAT** model in an end-to-end fashion. Formally, we define the intermediate alignment regularization as cross-entropy loss between the predicted word and the true word:

$$\mathcal{L}_{\text{reg}} = \sum_{t=1}^{m} \mathcal{H}\left(y^*_t, \ \text{FFN}(\mathbf{Z}^{md}_t)\right), \tag{10}$$

where $\mathbf{Z}^{md}$ $(1 < md < 6)$ represents the output of each intermediate layer. Consequently, the final loss of **SNAT** can be represented with the coefficient λ as:

$$\mathcal{L}_{\textbf{SNAT}} = \mathcal{L}_{\text{word}} + \mathcal{L}_{\text{label}} + \lambda \mathcal{L}_{\text{reg}}. \tag{11}$$

4 Experiment

In this section, we conduct experiments to evaluate the effectiveness and efficiency of our proposed model, with comprehensive analysis.

Table 2: Performance of BLEU score on WMT14 En↔De and WMT16 En↔Ro tasks. "-" denotes that the results are not reported. LSTM-based results are from (Wu et al., 2016); CNN-based results are from (Gehring et al., 2017). †The Transformer (Vaswani et al., 2017) results are based on our own reproduction.

	En→De	De→En	En→Ro	Ro→En		
Autoregressive Models					**Latency**	**Speedup**
LSTM Seq2Seq (Bahdanau et al., 2017)	24.60	-	-	-	-	-
Conv S2S (Gehring et al., 2017)	26.43	-	30.02	-	-	-
Transformer† (Vaswani et al., 2017)	27.48	31.29	34.36	33.82	642ms	1.00X
Non-autoregressive Models					**Latency**	**Speedup**
NAT (Gu et al., 2018)	17.69	20.62	29.79	-	39ms	15.6X
NAT, rescoring 10 (Gu et al., 2018)	18.66	22.41	-	-	79ms	7.68X
NAT, rescoring 100 (Gu et al., 2018)	19.17	23.20	-	-	257ms	2.36X
iNAT (Lee et al., 2018)	21.54	25.43	29.32	-	-	5.78X
Hint-NAT (Li et al., 2020)	21.11	25.24	-	-	26ms	23.36X
FlowSeq-base (Ma et al., 2019)	21.45	26.16	-	29.34	-	-
ENAT-P (Guo et al., 2019)	20.26	23.23	29.85	-	25ms	24.3X
ENAT-P, rescoring 9	23.22	26.67	34.04	-	50ms	12.1X
ENAT-E	20.65	23.02	30.08	-	24ms	25.3X
ENAT-E, rescoring 19	24.28	26.10	34.51	-	49ms	12.4X
DCRF-NAT (Sun et al., 2019)	23.44	27.22	-	-	37ms	16.4X
DCRF-NAT, rescoring 9	26.07	29.68	-	-	63ms	6.1X
DCRF-NAT, rescoring 19	26.80	30.04	-	-	88ms	4.4X
NAR-MT(rescoring 11) (Zhou and Keung, 2020)	23.57	29.01	31.21	32.06	-	-
NAR-MT(rescoring 11) + monolingual	25.53	29.96	31.91	33.46	-	-
AXE CMLM (Ghazvininejad et al., 2020)	23.53	27.90	30.75	31.54	-	-
SNAT	24.64	28.42	32.87	32.21	26.88ms	22.6X
SNAT, rescoring 9	26.87	30.12	34.93	33.11	54.63ms	11.1X
SNAT, rescoring 19	**27.50**	**30.82**	**35.19**	**33.98**	65.62ms	9.3X

4.1 Experimental Setup

Data We evaluate **SNAT** performance on both the WMT14 En-De (around 4.5M sentence pairs) and the WMT16 En-Ro (around 610k sentence pairs) parallel corpora. For the parallel data, we use the processed data from (Ghazvininejad et al., 2019) to be consistent with previous publications. The dataset is processed with Moses script (Hoang and Koehn, 2008), and the words are segmented into subword units using byte-pair encoding (Sennrich et al., 2016). The WMT14 En-De task uses newstest-2013 and newstest-2014 as development and test sets, and WMT16 En-Ro task uses newsdev-2016 and newstest-2016 as development and test sets. We report all results on test sets. The vocabulary is shared between source and target languages and has ∼36k units and ∼34k units in WMT14 En-De and WMT16 En-Ro, respectively.

Model Configuration Our implementation is based on the PyTorch sequence modeling toolkit Fairseq.[1] We follow the weights initialization scheme from BERT and follow the settings of the base Transformer configuration in (Vaswani et al., 2017) for all the models: 6 layers per stack, 8 attention heads per layer, 512 model dimensions and 2,048 hidden dimensions. The dimension of POS and NER embedding is set to 512 which is the same as the word embedding dimension. The autoregressive and non-autoregressive model have the same encoder-decoder structure, except for the decoder attention mask and the decoding input for the non-autoregressive model as described in Sec. 3. We try different values for the label mismatch penalty ϵ from $\{0.01, 0.1, 0.5\}$ and find that 0.1 gives the best performance. The coefficient λ is tested with different values from $\{0.25, 0.5, 0.75, 1\}$, and $\lambda = 0.75$ outperforms other settings. We set the initial learning rate as values from {8e-6, 1e-5, 2e-5, 3e-5}, with a warm-up rate of 0.1 and L2 weight decay of 0.01. Sentences are tokenized and the maximum number of tokens in each step is set to 8,000. The maximum iteration step is set to 30,000, and we train the model with early stopping.

Baselines We choose the following models as baselines: **NAT** is a vanilla non-autoregressive Transformer model for NMT which is first introduced in (Gu et al., 2018). **iNAT** (Lee et al., 2018) extends the vanilla NAT model by iteratively read-

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

ing and refining the translation. The number of iterations is set to 10 for decoding. **Hint-NAT** (Li et al., 2020) utilizes the intermediate hidden states from an autoregressive teacher to improve the NAT model. **FlowSeq** (Ma et al., 2019) adopts normalizing flows (Kingma and Dhariwal, 2018) as latent variables for generation. **ENAT** (Guo et al., 2019) proposes two ways to enhance the decoder inputs to improve NAT models. The first one leverages a phrase table to translate source tokens to target tokens **ENAT-P**. The second one transforms source-side word embedding into target-side word embeddings **ENAT-E**. **DCRF-NAT** (Sun et al., 2019) designs an approximation of CRF for NAT models and further uses a dynamic transition technique to model positional context in the CRF. **NAR-MT** (Zhou and Keung, 2020) uses a large number of source texts from monolingual corpora to generate additional teacher outputs for NAR-MT training. **AXE CMLM** (Ghazvininejad et al., 2020) trains the conditional masked language models using a differentiable dynamic program to assign loss based on the best possible monotonic alignment between target tokens and model predictions.

4.2 Training and Inference Details

To obtain the part-of-speech and named entity labels, we use industrial-strength spaCy[2] to acquire the label for English, German, and Romanian input. In our implementation, there are 17 labels for POS in total, i.e., ADJ (adjective), ADV (adverb), ADP (ad-position), AUX (auxiliary), CCONJ (coordinating conjunction), DET (determiner), INTJ (interjection), NOUN (noun), NUM (numeral), PART (particle), PRON (pronoun), PROPN (proper noun), PUNCT (punctuation), SCONJ (subordinating conjunction), SYM (symbol), VERB (verb), and X (other). The NER task is trained on OntoNotes v5.0 benchmark dataset (Pradhan et al., 2013) using formatted BIO labels and defines 18 entity types: CARDINAL, DATE, EVENT, FAC, GPE, LANGUAGE, LAW, LOC, MONEY, NORP, ORDINAL, ORG, PERCENT, PERSON, PRODUCT, QUANTITY, TIME, and WORK_OF_ART.

Knowledge Distillation Similar to previous works on non-autoregressive translation (Gu et al., 2018; Shu et al., 2020; Ghazvininejad et al., 2019), we use sequence-level knowledge distillation by training **SNAT** on translations generated by a standard left-to-right Transformer model (i.e., Transformer-large for WMT14 EN→DE, and Transformer-base for WMT16 EN→RO). Specifically, we use scaling NMT (Ott et al., 2018) as the teacher model. We report the performance of standard autoregressive Transformer trained on distilled data for WMT14 EN→DE and WMT16 EN→RO. We average the last 5 checkpoints to obtain the final model. We train the model with cross-entropy loss and label smoothing ($\epsilon = 0.1$).

Inference During training, we do not need to predict the target length m since the target sentence is given. During inference, we use a simple method to select the target length for **SNAT** (Wang et al., 2019; Li et al., 2020). First, we set the target length to $m' = n + C$, where n is the length of the source sentence and C is a constant bias term estimated from the overall length statistics of the training data. Then, we create a list of candidate target lengths with a range of $[m' - B, m' + B]$ where B is the half-width of the interval. Finally, the model picks the best one from the generated $2B + 1$ candidate sentences. In our experiments, we set the constant bias term C to 2 for WMT 14 EN→DE, -2 for WMT 14 DE→EN, 3 for WMT 16 EN→RO, and -3 for WMT 14 RO→EN according to the average lengths of different languages in the training sets. We set B to 4 or 9, and obtain corresponding 9 or 19 candidate translations for each sentence. Then we employ an autoregressive teacher model to rescore these candidates.

4.3 Results and Analysis

Experimental results are shown in Table 2. We first compare the proposed method against autoregressive counterparts in terms of translation quality, which is measured by BLEU (Papineni et al., 2002). For all our tasks, we obtain results comparable with the Transformer, the state-of-the-art autoregressive model. Our best model achieves 27.50 (+0.02 gain over Transformer), 30.82 (-0.46 gap with Transformer), 35.19 (+0.82 gain), and 33.98 (+0.16 gain) BLEU score on WMT14 En↔De and WMT16 EN↔Ro, respectively. More importantly, our **SNAT** decodes much faster than the Transformer, which is a big improvement regarding the speed-accuracy trade-off in AT and NAT models.

Comparing our models with other NAT models, we observe that the best **SNAT** model achieves a significant performance boost over NAT, iNAT, Hint-NAT, FlowSeq, ENAT, NAR-MT and AXE CMLM by +8.33, +5.96, +6.39, +6.05, +3.22, 3.93

[2] https://spacy.io/usage/models

and +3.97 in BLEU on WMT14 En→De, respectively. This indicates that the incorporation of the syntactic and semantic structure greatly helps reduce the impact of the multimodality problem and thus narrows the performance gap between Autoregressive Transformer (AT) and Non-Autoregressive Transformer (NAT) models. In addition, we see a +0.69, +0.78, +0.68, and 0.52 gain of BLEU score over the best baselines on WMT14 En→De, WMT14 De→En, WMT16 En→Ro and WMT16 Ro→En, respectively.

From the result of our methods at the last group in Table 2, we find that the rescoring technique substantially assists in improving the performance, and dynamic decoding significantly reduces the time spent on rescoring while further accelerating the decoding process. On En→De, rescoring 9 candidates leads to a gain of +2.23 BLEU, and rescoring 19 candidates gives a +2.86 BLEU score increment.

Decoding Speed Following previous works (Gu et al., 2018; Lee et al., 2018; Guo et al., 2019), we evaluate the average per-sentence decoding latency on WMT14 En→De test sets with batch size being 1, under an environment of NVIDIA Titan RTX GPU for the Transformer model and the NAT models to measure the speedup. The latencies are obtained by taking an average of five runs. More clearly, We reproduce the Transformer on our machine. We copy the runtime of other models but the speedup ratio is between the runtime of their implemented Transformer and their proposed model. We think it's reasonable to compare the speedup ratio because it is independent of the influence caused by different implementation software or machines. And to clarify, the latency does not include preprocessing of tagging, because it's a very fast process as executing around 7000 sentences in one second.

We can see from Table 2 that the best **SNAT** gets a 9.3 times decoding speedup than the Transformer, while achieving comparable or even better performance. Compared to other NAT models, we observe that the **SNAT** model is almost the fastest (only a little bit behind of ENAT and Hint-NAT) in terms of latency, and is surprisingly faster than DCRF-NAT with better performance.

4.4 Ablation Analysis

Effect of Syntactic and Semantic Structure Information We investigate the effect of using the syntactic and semantic tag on the model performance. Experimental results are shown in Table 3.

Table 3: The performance of different vision of SNAT models on WMT14 En→De development set. ✔ means selecting the label tag.

Model	POS tag	NER tag	BLEU
SNAT-V1	✔		24.21
SNAT-V2		✔	24.09
SNAT-V3			22.84

Table 4: The performance with respect to using different layer of intermediate interaction. Evaluated by the BLEU score on WMT14 En→De|WMT14 De→En.

Method	WMT14 En→ De	WMT14 De→ En
w/o	23.11	27.03
w/ $\mathbf{Z}^2$	24.32	28.21
w/ $\mathbf{Z}^3$	24.57	28.42

It demonstrates that incorporating POS information boosts the translating performance (+1.37 on WMT14 En→De) and NER information can also enhance the translating performance (+1.25 on WMT14 En→De). The POS label enriches the model with the syntactic structure, while the NER label supplements the semantic information to the model which are critical elements for **SNAT** model to exhibit better translation performance.

Effect of Intermediate Representation Alignment We conduct experiments for our **SNAT** model on WMT14 En→De with various alignments between decoder layers and target. As shown in Table 4, using the second layer $\mathbf{Z}^2$ in the decoder as intermediate alignment can gain +1.21 improvement, while using the third layer $\mathbf{Z}^3$ in the decoder as intermediate alignment can gain +1.46 improvement. This is in line with our expectation that aggregating layer-wise token information in intermediate layers can help improve the decoder's ability to capture token-token dependencies.

Effect of Sentence Length To evaluate different models on different sentence lengths, we conduct experiments on the WMT14 En→De development set and divide the sentence pairs into different length buckets according to the length of the reference sentences. As shown in Table 5, the column of 100 calculates the BLEU score of sentences that the length of the reference sentence is larger than 50 but smaller or equal to 100. We can see that the performance of vanilla NAT drops quickly as the sentence length increases from 10 to 50, while AT model and the proposed **SNAT** model have relatively stable performance over different sen-

Table 5: The performance with respect to different sentence lengths. Evaluated by the BLEU score on WMT14 En→De.

Model	10	20	30	50	100
AT	28.35	28.32	28.30	24.26	20.73
NAT	21.31	19.55	17.19	16.31	11.35
SNAT	28.67	28.50	27.33	25.41	17.69

tence lengths. This result confirms the power of the proposed model in modeling long-term token dependencies.

5 Conclusion

In this paper, we have proposed a novel syntactic and semantic structure-aware non-autoregressive Transformer model **SNAT** for NMT. The proposed model aims at reducing the computational cost in inference as well as keeping the quality of translation by incorporating both syntactic and semantic structures existing among natural languages into a non-autoregressive Transformer. In addition, we have also designed an intermediate latent alignment regularization within target sentences to better learn the long-term token dependencies. Comprehensive experiments and analysis on two real-world datasets (i.e., WMT14 En→De and WMT16 En→Ro) verify the efficiency and effectiveness of our proposed approach.

Acknowledgements

This work is supported in part by NSF under grants III-1763325, III-1909323, and SaTC-1930941.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, Dzmitry Bengio, Yoshua Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *5th International Conference on Learning Representations, ICLR 2017*.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. Kermit: Generative insertion-based modeling for sequences. *arXiv preprint arXiv:1906.01604*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1243–1252. PMLR.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *Proceedings of the International Conference on Machine Learning*, pages 9330–9338.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6114–6123.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

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

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Hieu Hoang and Philipp Koehn. 2008. Design of the moses decoder for statistical machine translation. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 58–65.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2390–2399, Stockholmsmässan, Stockholm Sweden. PMLR.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in neural information processing systems*, pages 10215–10224.

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

Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2020. Hint-based training for non-autoregressive machine translation. In *Proceedings of the International Conference on Learning Representations*.

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

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4282–4292, Hong Kong, China. Association for Computational Linguistics.

Mitchell P Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. *Linguistic Data Consortium, Philadelphia*, 14.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. 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*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. 2020. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 198–205.

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 *AAAI*, pages 8846–8853.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 5976–5985.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhihong Deng. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems*, pages 3011–3020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yiren Wang, Fei Tian, Di He, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2019. Non-autoregressive machine translation with auxiliary regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5377–5384.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1304–1312, Florence, Italy. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jiawei Zhou and Phillip Keung. 2020. Improving non-autoregressive neural machine translation with monolingual data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1893–1898, Online. Association for Computational Linguistics.