Modeling Coverage for Non-Autoregressive Neural Machine Translation

Yong Shan, Yang Feng*, Chenze Shao
Key Laboratory of Intelligent Information Processing,
Institute of Computing Technology, Chinese Academy of Sciences (ICT/CAS), Beijing, China
University of Chinese Academy of Sciences, Beijing, China
shanyong18s@ict.ac.cn, fengyang@ict.ac.cn, shaochenze18z@ict.ac.cn

Abstract—Non-Autoregressive Neural Machine Translation (NAT) has achieved significant inference speedup by generating all tokens simultaneously. Despite its high efficiency, NAT usually suffers from two kinds of translation errors: over-translation (e.g., repeated tokens) and under-translation (e.g., missing translations), which eventually limits the translation quality. In this paper, we argue that these issues of NAT can be addressed through coverage modeling, which has been proved to be useful in autoregressive decoding. We propose a novel Coverage-NAT to model the coverage information directly by a token-level coverage iterative refinement mechanism and a sentence-level coverage agreement, which can remind the model if a source token has been translated or not and improve the semantics consistency between the translation and the source, respectively. Experimental results on WMT14 En-De and WMT16 En-Ro translation tasks show that our method can alleviate those errors and achieve strong improvements over the baseline system.

I. INTRODUCTION

Neural Machine Translation (NMT) has achieved impressive performance over the recent years [1]–[5]. NMT models are typically built on the encoder-decoder framework where the encoder encodes the source sentence into distributed representations, and the decoder generates the target sentence from the representations in an autoregressive manner: the decoding at step $t$ is conditioned on the predictions of past steps. The autoregressive translation manner (AT) leads to high inference latency and restricts NMT applications.

Recent studies try to address this issue by replacing AT with non-autoregressive translation (NAT) which generates all tokens simultaneously [6]. However, the lack of sequential dependency between target tokens in NAT makes it difficult for the model to capture the highly multimodal distribution of the target translation. This phenomenon is termed as the multimodality problem [6], which usually causes two kinds of translation errors during inference: over-translation where the same token is generated repeatedly at consecutive time steps, and under-translation where the semantics of several source tokens are not fully translated [7]. Table I shows a German-English example, where “technical” and “has” are translated repeatedly while “also” is not translated at all.

Actually, there are over-translation and under-translation errors in AT models, too. They are usually addressed through explicitly modeling the coverage information step by step [8].

* Yang Feng is the corresponding author.

| Table I |
|---------|
| **ERRORS IN NAT** |
| **“Src”** | **“Ref”** |
| Das technische Personal hat mir ebenfalls viel gegeben. | The technical staff has also brought me a lot. |

In this paper, we propose a novel Coverage-NAT to address this challenge by directly modeling coverage information in NAT. Our model works at both token level and sentence level. At token level, we propose a token-level coverage iterative refinement mechanism (TCIR), which models the token-level coverage information with an iterative coverage layer and refines the source exploitation iteratively. Specifically, we replace the top layer of the decoder with the iterative coverage layer and compute the coverage vector in parallel. In each iteration, the coverage vector tracks the attention history of the previous iteration and then adjusts the attention of the current iteration, which encourages the model to concentrate more on the untranslated parts and hence reduces over-translation and under-translation errors. In contrast to the step-wise manner [8], TCIR models the token-level coverage information in an iteration-wise manner, which does not hurt the non-autoregressive generation. At sentence level, we propose a sentence-level coverage agreement (SCA) as a regularization term. Inspired by the intuition that a good translation should “cover” the source semantics at sentence level, SCA firstly computes the sentence representations of the translation and the source sentence, and then constrains the semantics agreement between them. Thus, our model can improve the sentence-level coverage of translation, too.

We evaluate our model on four translation tasks (WMT14
Experimental results show that our method outperforms the baseline system by up to +2.75 BLEU with competitive speedup (about $10 \times$), and further achieves near-Transformer performance on WMT16 dataset but $5.22 \times$ faster with the rescoring technique. The main contributions of our method are as follows:

- To alleviate the over-translation and under-translation errors, we introduce the coverage modeling into NAT.
- We propose to model the coverage information at both token level and sentence level through iterative refinement and semantics agreement respectively, both of which are effective and achieve strong improvements.

## II. BACKGROUND

Recently, neural machine translation has achieved impressive improvements with the autoregressive decoding. Despite its high performance, AT models suffer from long inference latency. NAT [6] is proposed to address this problem through parallelizing the decoding process.

The NAT architecture can be formulated into an encoder-decoder framework [2]. The same as AT models, the NAT encoder takes the embedding of source tokens as inputs and outputs the contextual source representations $E_{enc}$, which are further used to compute the target length $T$ and inter-attention at decoder side.

### Multi-Head Attention: The attention mechanism used in Transformer and NAT are multi-head attention (Attn), which can be generally formulated as the weighted sum of value vectors $V$ using query vectors $Q$ and key vectors $K$:

$$
A = \text{softmax}(\frac{QK^T}{\sqrt{d_{\text{model}}}})
$$

$$
\text{Attn}(Q, K, V) = AV
$$

where $d_{\text{model}}$ represents the dimension of hidden states. There are 3 kinds of multi-head attention in NAT: multi-head self attention, multi-head inter-attention and multi-head positional attention. Refer to [6] for details.

### Encoder: A stack of $L$ identical layers. Each layer consists of a multi-head self attention and a position-wise feed-forward network (FFN). FFN consists of two linear transformations with a ReLU activation [5].

$$
\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2
$$

Note that we omit layer normalization, dropout and residual connections for simplicity. Besides, a length predictor is used to predict the target length.

### Decoder: A stack of $L$ identical layers. Each layer consists of a multi-head self attention, a multi-head inter-attention, a multi-head positional attention, and a position-wise feed-forward network.

Given the input sentence in the source language $X = \{x_1, x_2, \cdots, x_n\}$, NAT models generate a sentence in the target language $Y = \{y_1, y_2, \cdots, y_T\}$, where $n$ and $T$ are the lengths of the source and target sentence, respectively. The translation probability is defined as:

$$
P(Y|X, \theta) = \prod_{t=1}^{T} p(y_t|X, \theta)
$$

The cross-entropy loss is applied to minimize the negative log-likelihood as:

$$
\mathcal{L}_{\text{mle}}(\theta) = -\sum_{t=1}^{T} \log(p(y_t|X, \theta))
$$

During training, the target length $T$ is usually set as the reference length. During inference, $T$ is determined by the target length predictor, and then the translation is obtained by taking the token with the maximum likelihood at each step:

$$
\hat{y}_t = \arg \max_{y_t} p(y_t|X, \theta)
$$

## III. COVERAGE-NAT

In this section, we describe the details of Coverage-NAT. We first briefly introduce the NAT architecture in Section III-A

Then, we illustrate the proposed token-level coverage iterative refinement (TCIR) and sentence-level coverage agreement (SCA) in Section III-B and III-C respectively. Figure II shows the architecture of Coverage-NAT.

### A. NAT Architecture

The NAT architecture we used is similar to [6] with the following differences in the decoder: (1) Instead of copying source embedding according to fertility or uniform mapping, we feed a sequence of “(unk)” tokens to the NAT decoder directly. (2) We remove the positional attention in decoder layers. This design simplifies the decoder but works well in our experiments.

In Coverage-NAT, the bottom $L - 1$ layers of the decoder are identical. Specifically, we design a novel iterative coverage layer as the top layer, which refines the source exploitations by iteratively modeling token-level coverage.

### B. Token-Level Coverage Iterative Refinement

The token-level coverage information has been applied to phrase-based SMT [12] and AT [8], where the decoder maintains a coverage vector to indicate whether a source token is translated or not and then pays more attention to the untranslated parts. Benefit from the autoregressive nature, AT models can easily model the coverage information step by step. Generally, it is a step-wise refinement of source exploitation with the assistance of coverage vector. However, it encounters difficulties in non-autoregressive decoding. Since all target tokens are generated simultaneously, step $t$ cannot know if a source token has been translated by other steps during decoding.

The key to solving this problem of NAT lies in: (1) How to compute the coverage vector under the non-autoregressive setting? (2) How to use the coverage vector to refine the source exploitation under the non-autoregressive setting? To address
it, we propose a novel token-level coverage iterative refinement mechanism. Our basic idea is to replace the top layer of the decoder with an iterative coverage layer which models the token-level coverage information and then refines the source exploitation in an iteration-wise manner. The iterative coverage layer also contains a multi-head self attention, a multi-head inter-attention and an FFN.

For question (1), we maintain an attention precedence which means the former steps have priorities to attend to source tokens and later steps shouldn’t scramble for those already “covered” parts. We manually calculate the coverage vector $C^k$ on the basis of the inter-attention weights of the previous iteration. At step $t$, we compute the accumulation of past steps’ attention on $x_i$ as $C_{t,i}^k$ to represent the coverage of $x_i$:

$$C_{t,i}^k = \min\left(\sum_{t'=0}^{t-1} A_{t',i}^{k-1}, 1\right) \quad (6)$$

where $k$ is the iteration number, $A_{t',i}^{k-1}$ is the inter-attention weights from step $t'$ to $x_i$ in the previous iteration. We use $\min$ function to ensure all coverage do not to exceed 1.

For question (2), we use $C^k$ as a bias of inter-attention so that step $t$ will pay more attention to source tokens untranslated by past steps in the previous iteration. A full iteration can be given by:

$$\hat{H}^k = \text{Attn}(H^{k-1}, H^{k-1}, H^{k-1}) \quad (7)$$

$$B^k = (1 - C^k) \quad (8)$$

$$A^k = \text{softmax}\left(\frac{\hat{H}^k E_{\text{enc}}^T}{\sqrt{d_{\text{model}}} + \lambda \times B^k}\right) \quad (9)$$

$$H^k = \text{FFN}(A^k E_{\text{enc}}) \quad (10)$$

where $H^k$ is the hidden states of iteration $k$, $\lambda \in \mathbb{R}^1$ is trainable and is initialized to 1, $\times$ means the scalar multiplication. In Equation (7) we compute the self attention and store the output as $\hat{H}^k$. In Equation (8-9) we firstly calculate the subtraction between 1 and $C^k$ to make the source tokens with a lower coverage obtain a higher bias, and then integrate the bias $B^k$ into the inter-attention weights by a trainable scaling factor $\lambda$. Finally, we output the hidden states of iteration $k$ in Equation (10). In this way, each step is discouraged to consider source tokens heavily attended by past steps in the previous iteration, and pushed to concentrate on source tokens less attended. The source exploitation is refined iteratively with token-level coverage, too.

We use the hidden states and the inter-attention weights of $(L-1)$th layer as $H^0$ and $A^0$, respectively. After $K$ iterations, we use $H^K$ as the output of the decoder.

### C. Sentence-Level Coverage Agreement

In AT models, the sentence-level agreement between the source side and the target side has been exploited by Yang...
et al. [13], which constrains the agreement between source sentence and reference sentence. However, it can only enhance the representations of the source side and target side instead of improving the semantics consistency between the source sentence and the translation.

Different from them, we consider the sentence-level agreement from a coverage perspective. Intuitively, a translation “covers” source semantics at sentence level if the representation of translation is as close as possible to the source sentence. Thus, we propose to constrain the sentence-level agreement between the source sentence and the translation. For the source sentence, we compute the sentence representation by considering the whole sentence as a bag-of-words. Specifically, we firstly map the source embeddings to target latent space by a linear transformation, and then use a mean pooling to obtain the sentence representation $E_{src}$:

$$E_{src} = \text{Mean}(\text{ReLU}(E_{src}W_e)) \quad (11)$$

where $E_{src} = \{E_{src,1}, \cdots, E_{src,n}\}$ is the word embeddings of source sentence, $W_e \in \mathbb{R}^{d_{model} \times d_{model}}$ represents the parameters of linear transformation and ReLU is the activation function. For the translation, we need to calculate the expectation of hypothesis at each position, which takes all possible translation results into consideration. Although it is inconvenient for AT to calculate the expectation by multinomial sampling [14], we can do it easily in NAT due to the token-independence assumption:

$$E_{hyp,t} = p_tW_e \quad (12)$$

where $p_t \in \mathbb{R}^{d_{vocab}}$ is the probability vector of hypothesis at step $t$, $W_e \in \mathbb{R}^{d_{vocab} \times d_{model}}$ is the weights of target-side word embedding. Then, we calculate the sentence representation of translation by mean pooling as follows:

$$E_{hyp} = \text{Mean}(\{E_{hyp,1}, \cdots, E_{hyp,T}\}) \quad (13)$$

Finally, we compute the L2 distance between the source sentence and the translation as the sentence-level coverage agreement. Note that we use a scaling factor $\frac{1}{\sqrt{d_{model}}}$ to stabilize the training:

$$\mathcal{L}_{coverage} = \frac{L^2(E_{src}, E_{hyp})}{\sqrt{d_{model}}} \quad (14)$$

D. Two-Phase Optimization

In our approach, we firstly pre-train the proposed model to convergence with basic loss functions, and then fine-tune the proposed model with the SCA and basic loss functions.

Pre-training During the pre-training phase, we optimize the basic loss functions for NAT:

$$\mathcal{L} = \mathcal{L}_{mle} + \alpha \mathcal{L}_{length} \quad (15)$$

where $\mathcal{L}_{length}$ is the loss of target length predictor [6]. $\alpha$ is a hyper-parameter to control the weight of $\mathcal{L}_{length}$.

Fine-tuning During the fine-tuning phase, we optimize the SCA and basic loss functions jointly:

$$\mathcal{L} = \mathcal{L}_{mle} + \alpha \mathcal{L}_{length} + \beta \mathcal{L}_{coverage} \quad (16)$$

where $\beta$ is a hyper-parameter to control the weight of $\mathcal{L}_{coverage}$. We have tried to train the model with all loss functions jointly from scratch. The result is worse than fine-tuning. We think that this is because the SCA will interfere the optimization of translation when the training begins.

IV. EXPERIMENTS SETUP

Datasets We evaluate the effectiveness of our proposed method on WMT14 En→De (4.5M pairs) and WMT16 En→Ro (610k pairs) datasets. For WMT14 En→De, we employ newstest2013 and newstest2014 as development and test sets. For WMT16 En→Ro, we employ newstest2016 and newstest2016 as development and test sets. We tokenize all sentences into subword units [18] with 32k merge operations. We share the vocabulary between the source and target sides.

Metrics We evaluate the model performance with case-sensitive BLEU [19] for the En→De dataset and case-insensitive BLEU for the En→Ro dataset. Latency is computed as the average of per sentence decoding time (ms) on the full test set of WMT14 En→De without mini-batching. We test latency on 1 Nvidia RTX 2080Ti GPU.

Baselines We take the Transformer model [5] as our AT baseline as well as the teacher model. We choose the following models as our NAT baselines: (1) NAT-FT which equips vanilla NAT with fertility fine-tuning [6]. (2) NAT-Base which is the NAT implementation of Fairseq [4] which removes the multi-head positional attention and feeds the decoder with a sequence of ‘‘(unk)’’ instead of copying the source embedding. (3) NAT-IR which iteratively refines the translation for multiple times [15]. (4) Reinforce-NAT which introduces a sequence-level objective using reinforcement learning [10]. (5) NAT-REG which proposes two regularization terms to alleviate translation errors [7]. (6) BoN-NAT which minimizes the bag-of-ngrams differences between the translation and the reference [11]. (7) Hint-NAT which leverages the hints from AT models to enhance NAT [16]. (8) Flowseq-large which integrates generative flow into NAT [17].

Model Configurations We implement our approach on NAT-Base. We closely follow the settings of [6], [15]. For all experiments, we use the base configurations ($d_{model}=512$, $d_{hidden}=2048$, $n_{layer}=6$, $n_{head}=8$). In the main experiments, we set the number of iterations during both training ($K_{train}$) and decoding ($K_{dec}$) to 5, $\alpha$ to 0.1 and $\beta$ to 0.5.

Training We train the AT teacher with the original corpus, construct the distillation corpus with the sequence-level knowledge distillation [20], and train NAT models with the distillation corpus. We choose Adam [21] as the optimizer. Each mini-batch contains 64k tokens. During pre-training, we stop training when the number of training steps exceeds 400k, 180k for WMT14 En→De, WMT16 En→Ro, respectively. The learning rate warms up to 5e-4 in the first 10K steps and then decays under the inverse square-root schedule. During fine-tuning, we fix the learning rate to 1e-5 and stop training when the number of training steps exceeds 1k. We run all
Jeffreez said on Facebook that the impact of each method is near the performance of Transformer but 5.22 BLEU score with competitive speedup over Transformer (10.04×)

As Figure 2 shows, we use Coverage-NAT (w/o SCA, $K_{train} = 5$) to decode the WMT14 En→De development set with different iterations during decoding ($K_{dec}$). As $K_{dec}$ grows, the BLEU increases and gets the maximum value near $K_{train}$ while the latency increases continuously. The reason might be that it is harder for NAT to align to long source sentences which are more frequent in WMT16 dataset, and Coverage-NAT exactly models the coverage information. Because of the trade-off between performance and latency, we set $K_{train} = 5$ in our main experiments for an equilibrium although $K_{train}$ and $K_{dec}$ are the same. With the increase of $K_{train}$, the BLEU firstly drops slightly then increases continuously. It demonstrates that the TCIR needs enough iterations to learn effective source exploitation with the help of token-level coverage information. Because of the trade-off between performance and latency, we set $K_{train} = 5$ in our main experiments for an equilibrium although $K_{train} = 15$ brings more gains. Second, we evaluate the effects of different SCA weights when we set $\beta$ to 0, 1 and 0.5, the BLEU increases by +0.63, +1.07 and +1.20, respectively. It proves that a moderate $\beta$ is important for the effects of SCA.

C. Effects of Iterations During Decoding

As Table III shows, we conduct the ablation study on WMT16 En→Ro development set to show the effectiveness of each module in Coverage-NAT. First, we evaluate the effects of different iterations during training ($K_{train}$). In ablation study, $K_{train}$ and $K_{dec}$ are the same. With the increase of $K_{train}$, the BLEU firstly drops slightly then increases continuously. It demonstrates that the TCIR needs enough iterations to learn effective source exploitation with the help of token-level coverage information. Because of the trade-off between performance and latency, we set $K_{train} = 5$ in our main experiments for an equilibrium although $K_{train} = 15$ brings more gains. Second, we evaluate the effects of different SCA weights when we set $\beta$ to 0, 1 and 0.5, the BLEU increases by +0.63, +1.07 and +1.20, respectively. It proves that a moderate $\beta$ is important for the effects of SCA.

| Model              | $K_{train}$ | $\beta$ | BLEU (%) | Latency (ms) |
|--------------------|-------------|---------|----------|--------------|
| NAT-Base           | 1           | /       | 28.77 (-0.19) | 20.82       |
|                    | 3           | /       | 29.20 (+0.24) | 24.84       |
|                    | 5           | /       | 29.33 (+0.37) | 28.34       |
| Coverage-NAT       | 10          | /       | 29.45 (+0.49) | 40.80       |
|                    | 15          | /       | 29.61 (+0.65) | 48.90       |
|                    | 5           | 0       | 29.59 (+0.63) | 28.34       |
|                    | 5           | 1       | 30.03 (+1.07) | 28.34       |
|                    | 5           | 0.5     | 30.16 (+1.20) | 28.34       |
that the performance doesn’t increase continuously may be that our model are trained to provide the best coverage at $K_{\text{train}}$, which indicates our model can work well with less iterations and lower latency.

D. Analysis of Repeated Tokens

We study the ratio of repeated tokens during post-process to see how our methods improve the translation quality. We split the WMT14 En→De development set equally into the short part and the long part according to the sentence length. Then, we translate them using NAT models and score the ratio of repeated tokens during post-process. As shown in Table IV, NAT-base suffers from over-translation errors especially on the long sentences, which is in line with [11]. As the sentence length grows, it becomes harder for NAT to align to source, which leads to more translation errors and eventually damages the translation quality. However, the ratio of repeated tokens reduces obviously in Coverage-NAT by 2.16% and 4.53% on short and long sentences, which proves that our model achieves higher improvements as the sentence length becomes longer. Moreover, we remove the SCA from Coverage-NAT and compare it with NAT-Base.

![Fig. 2. The BLEU and the latency with different iterations during decoding ($K_{\text{dec}}$) on WMT14 En→De development set. The results are from Coverage-NAT (w/o SCA, $K_{\text{train}} = 5$). It shows that our model reaches the maximum BLEU when $K_{\text{dec}} = K_{\text{train}}$, and achieves almost the maximum improvement when $K_{\text{dec}} = 4$ which indicates our model can work well with less iterations and lower latency.](image2)

![Fig. 3. The BLEU with different source lengths on the WMT14 En→De development set. Compared to NAT-Base, Coverage-NAT is closer to Transformer with a slighter decline in “[10,50)” and a greater increase in “[50,80)”.](image3)

**TABLE IV**
The ratio of repeated tokens (%) for translations on the WMT14 En→De development set. It proves that over-translation errors grow as the sentence becomes longer and the SCA achieves higher improvements than the TCIR.

| Model             | Short     | Long      | All       |
|-------------------|-----------|-----------|-----------|
| NAT-Base          | 5.47      | 12.05     | 10.23     |
| Coverage-NAT      | 3.31 (-2.16) | 7.52 (-4.53) | 6.36 (-3.87) |
| – SCA             | 4.82 (-0.65) | 10.21 (-1.84) | 8.72 (-1.51) |

It leads to 0.65% and 1.84% decreases of repeated tokens on short and long sentences, respectively. The results prove that both token-level and sentence-level coverage can reduce over-translation errors and sentence-level coverage contributes more than token-level coverage.

E. Subjective Evaluation of Under-Translation

We conduct a subjective evaluation to validate our improvements to under-translation errors. For each sentence, two human evaluators are asked to select the one with the least under-translation errors from translations of three anonymous systems. We evaluate a total of 300 sentences sampled randomly from the WMT14 De→En test set. Then, we calculate the under-translation metric which is the frequency that each system is selected. The higher the under-translation metric, the less the under-translation errors. Table V shows the results. Note that there are some tied selections so that the sum of three scores does not equal to 1. It proves that Coverage-NAT produces less under-translation errors than NAT-Base, and both word-level and sentence-level coverage modeling are effective in reducing them.

F. Effects of Source Length

To evaluate the effects of sentence lengths, we split the WMT14 En→De development set into different buckets according to the source length and compare the BLEU performance of our model with other baselines. Figure 3 shows the results. As the sentence length grows, the performance of NAT-Base firstly drops continuously, then has a slight rise.
and finally declines heavily on very long sentences. However, Transformer can keep a high performance until the sentence length becomes very long, which indicates that translating very long sentences is extremely difficult. Compared to NAT-Base, the result of Coverage-NAT is closer to Transformer, which has a more slight decline in “[10,50)” and a greater increase in “[50,80)”. The result of Coverage-NAT (w/o SCA) lies between Coverage-NAT and NAT-Base. It demonstrates that our model generates better translations as the sentence becomes longer and both token-level and sentence-level coverage modeling are effective to improve the translation quality, in line with Table [IV].

G. Case Study

We present several translation examples on the WMT14 De→En test set in Table [VI] including the source, the reference and the translations given by Transformer, NAT-Base, Coverage-NAT (w/o SCA) and Coverage-NAT. As it shows, NAT-Base suffers severe over-translation (e.g. “other other”) and under-translation (e.g. “important”) errors. However, Our Coverage-NAT can alleviate these errors and generate higher quality translations. The differences between Coverage-NAT (w/o SCA) and Coverage-NAT further prove the effectiveness of sentence-level coverage.

| Source | Edis erklärte, Coulsons Aktivitäten bei dieser Story folgten demselben Muster wie bei anderen wichtigen Persönlichkeiten, etwa dem früheren Innenminister David Blunkett. |
| Reference | Mr Edis said Coulson’s involvement in the story followed the same pattern as with other important men, such as former home secretary David Blunkett. |
| Transformer | Edis explained that Coulson’s activities in this story followed the same pattern as those of other important personalities, such as former Interior Minister David Blunkett. |
| NAT-Base | Edis explained that Coulson’s activities in this story followed the same pattern as with other important personalities, such as former Interior Minister David Blunkett. |
| Coverage-NAT (w/o SCA) | Edis explained that Coulson’s activities in this story followed the same pattern as other other (important) person personalities such as former Interior Minister David Blunkett. |
| Coverage-NAT | Edis explained that Coulson’s activities in this story followed the same pattern as with other important personalities, such as former Interior Minister David Blunkett. |

VI. RELATED WORK

Gu et al. [6] introduce non-autoregressive neural machine translation to accelerate inference. Recent studies have tried to bridge the gap between non-autoregressive models and autoregressive models. Lee et al. [15] refine the translation iteratively where the outputs of decoder are fed back as inputs in the next iteration. Ghazvininejad et al. [22] propose a mask-predict method to generate translations in several decoding iterations. [23], [24] are based on iterative refinements, too. These methods refine the translation by decoding iteratively while our TCIR refines the source exploitation by iteratively modeling the coverage with an individual layer. Besides, Libovický and Helcl [25] introduce a connectionist temporal classification loss to guide the NAT model. Shao et al. [10], [11] train NAT with sequence-level objectives and bag-of-ngrams loss. Wang et al. [7] propose NAT-REG with two auxiliary regularization terms to alleviate over-translation and under-translation errors. Guo et al. [26] enhance the input of the decoder with target-side information. Some works introduce latent variables to guide generation [17], [23], [27], [28]. Ran et al. [29] and Bao et al. [30] consider the reordering information in decoding. Li et al. [16] use the AT to guide the NAT training. Wang et al. [31] and Ran et al. [32] further propose semi-autoregressive methods.

To alleviate the over-translation and under-translation errors, coverage modeling is commonly used in SMT [12] and NMT [8], [9], [33]–[35]. Tu et al. [8] and Mi et al. [9] model the coverage in a step-wise manner and employ it as additional source features, which can not be applied to NAT directly. However, our TCIR models the coverage in an iteration-wise manner and employs it as attention biases, which does not hurt the non-autoregressive generation. Li et al. [33] introduce a coverage-based feature into NMT. Zheng et al. [34], [35] explicitly model the translated and untranslated contents.

Agreement-based learning in NMT has been explored in [36]–[39]. Yang et al. [13] propose a sentence-level agreement between source and reference while our SCA considers the sentence-level agreement between source and translation. Besides, Tu et al. [40] propose to reconstruct source from translation, which improves the translation adequacy but can not model the sentence-level coverage directly.

VII. CONCLUSION

In this paper, we propose a novel Coverage-NAT which explicitly models the coverage information at token level and sentence level to alleviate the over-translation and under-translation errors in NAT. Experimental results show that our model can achieve strong BLEU improvements with competitive speedup over the baseline system. The analysis further prove that it can effectively reduce translation errors.
