Appendix

A Related Works

A.1 AutoRegressive Translation

Given a sentence $x = (x_1, \ldots, x_{T_x})$ from the source language, the straight-forward way for translation is to generate the words in the target language $y = (y_1, \ldots, y_{T_y})$ one by one from left to right. This is also known as the autoregressive factorization in which the joint probability is decomposed into a chain of conditional probabilities, as in the Eqn. (1). Deep neural networks are widely used to model such conditional probabilities based on the encoder-decoder framework. The encoder takes the source tokens $(x_1, \ldots, x_{T_x})$ as input and encodes x into a set of context states $c = (c_1, \ldots, c_{T_x})$. The decoder takes c and subsequence $y_{<t}$ as input and estimates $P(y_t|y_{<t}, c)$ according to some parametric function.

There are many design choices in the encoder-decoder framework based on different types of layers, e.g., recurrent neural network(RNN)-based (Bahdanau et al., 2014), convolution neural network(CNN)-based (Gehring et al., 2017) and recent self-attention based (Vaswani et al., 2017) approaches. We show a self-attention based network (Transformer) in the left part of Figure 3. While the ART models have achieved great success in terms of translation quality, the time consumption during inference is still far away from satisfactory. During training, the ground truth pair (x, y) is exposed to the model, and thus the prediction at different positions can be estimated in parallel based on CNN or self-attention networks. However, during inference, given a source sentence x, the decoder has to generate tokens sequentially, as the decoder inputs $y_{<t}$ must be inferred on the fly. Such autoregressive behavior becomes the bottleneck of the computational time (Wu et al., 2016).

A.2 Non-AutoRegressive Translation

In order to speed up the inference process, a line of works begin to develop non-autoregressive translation models. These models follow the encoder-decoder framework and inherit the encoder structure from the autoregressive models. After generating the context states c by the encoder, a separate module will be used to predict the target sentence length T_y and decoder inputs $z = (z_1, \ldots, z_{T_y})$ by a parametric function: $(T_y, z) \sim f_z(x, c; \theta)$, which is either deterministic or stochastic. The decoder will then predict y based on following probabilistic decomposition

$$P(y|x, T_y, z) = \Pi_{t=1}^{T_y} P(y_t|z, c). \quad (4)$$

Different configurations of T_y and z enable the decoder to produce different target sentence y given the same input sentence x, which increases the output diversity of the translation models.

Previous works mainly pay attention to different design choices of f_z. Gu et al. (2017) introduce *fertilities*, corresponding to the number of target tokens occupied by each of the source tokens, and use a non-uniform copy of encoder inputs as z according to the fertility of each input token. The prediction of fertilities is done by a separated neural network-based module. Lee et al. (2018) define z by a sequence of generated target sentences $y^{(0)}, \ldots, y^{(L)}$, where each $y^{(i)}$ is a refinement of $y^{(i-1)}$. Kaiser et al. (2018) use a sequence of autoregressively generated discrete latent variables as inputs of the decoder.

While the expressiveness of z improved by different kinds of design choices, the computational overhead of z will hurt the inference speed of the NART models. Comparing to the more than 15× speed up in Gu et al. (2017), which uses a relatively simpler design choice of z, the speedup of Kaiser et al. (2018) is reduced to about 5×, and the speedup of Lee et al. (2018) is reduced to about 2×. This contradicts with the design goal of the NART models: to parallelize and speed up neural machine translation models.

A.3 Knowledge Distillation and Hint-Based Training

Knowledge Distillation (KD) was first proposed by Hinton et al. (2015), which trains a small *student network* from a large (possibly ensemble) *teacher network*. The training objective of the student network contains two parts. The first part is the standard classification loss, e.g, the cross entropy loss defined on the student network and the training data. The second part is defined between the output distributions of the student network and the teacher network, e.g, using KL-divergence . Kim and Rush (2016) introduces the KD framework to neural machine translation models. They replace the ground truth target sentence by the generated sentence from a well-trained teacher model. Sentence-level KD is also proved helpful

for non-autoregressive translation in multiple previous works (Gu et al., 2017; Lee et al., 2018).

However, knowledge distillation only uses the outputs of the teacher model, but ignores the rich hidden information inside a teacher model. Romero et al. (2014) introduced *hint-based training* to leverage the intermediate representations learned by the teacher model as hints to improve the training process and final performance of the student model. Hu et al. (2018) used the attention weights as hints to train a small student network for reading comprehension.

B Network Architecture

Encoder and decoder Same as the ART model, the encoder of the NART model takes the embeddings of source tokens as inputs,[3] and generates a set of context vectors. As discussed in the main paper, the NART model needs to predict z given length T_y and source sentence x. We use a simple and efficient method to predict $z = (z_1, \ldots, z_{T_y})$. Given source sentence $x = (x_1, \ldots, x_{T_x})$ and target length T_y, we denote $e(x_i)$ as the embedding of x_i. We linearly combine the embeddings of all the source tokens to generate z as follows:

$$z_j = \sum_i w_{ij} \cdot e(x_i), \tag{5}$$

where w_{ij} is the normalized weight that controls the contribution of $e(x_i)$ to z_j according to

$$w_{ij} \propto \exp\left(-(j - j'(i))^2/\tau\right), \tag{6}$$

where $j = 1, \ldots, T_y$ and $j'(i) = (T_y/T_x) \cdot i$. τ is a hyperparameter to control the "sharpness" of the weight distribution. We use $f_z(x, T_y, \tau)$ for this weighted average function to be consistent as in the non-autoregressive decomposition.

Three types of multi-head attention The ART and NART models share two types of multi-head attentions: multi-head *self* attention and multi-head *encoder-to-decoder* attention. The NART model specifically uses multi-head *positional* attention to model local word orders within the sentence (Vaswani et al., 2017; Gu et al., 2017). A general attention mechanism can be formulated as querying a dictionary with key-value pairs (Vaswani et al., 2017), e.g.,

$$\begin{aligned} &\text{Attention}(Q, K, V) \\ &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_{model}}}\right) \cdot V, \end{aligned} \tag{7}$$

where d_{model} is the dimension of hidden representations and Q (Query), K (Key), V (Value) differ among three types of attentions. For self attention, Q, K and V are hidden representations of the previous layer. For encoder-to-decoder attention, Q is hidden representations of the previous layer, whereas K and V are context vectors from the encoder. For positional attention, positional embeddings are used as Q and K, and hidden representations of the previous layer are used as V. The multi-head variant of attention allows the model to jointly attend to information from different representation subspaces, and is defined as

$$\begin{aligned} &\text{Multi-head}(Q, K, V) \\ &= \text{Concat}(\text{head}_1, \cdots, \text{head}_H)W^O, \end{aligned} \tag{8}$$

$$\begin{aligned} &\text{head}_h \\ &= \text{Attention}(QW_h^Q, KW_h^K, VW_h^V), \end{aligned} \tag{9}$$

where $W_h^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_h^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_h^V \in \mathbb{R}^{d_{model} \times d_k}$, and $W_h^O \in \mathbb{R}^{d_{model} \times Hd_v}$ are project parameter matrices, H is the number of heads, and d_k and d_v are the numbers of dimensions.

In addition to multi-head attentions, the encoder and decoder also contain fully connected feed-forward network (FFN) layers with ReLU activations, which are applied to each position separately and identically. Compositions of self attention, encoder-to-decoder attention, positional attention, and position-wise feed-forward network are stacked to form the encoder and decoder of the ART model and the NART model, with residual connections (He et al., 2016) and layer normalization (Ba et al., 2016).

C Extra Experimental Settings

Dataset specifications The split of the training/validation/test sets of the IWSLT14 dataset[4] contain about 153K/7K/7K sentence pairs, respectively. The training set of the WMT14 dataset[5]

[3] Following (Vaswani et al., 2017; Gu et al., 2017) we also use *positional embedding* to model relative correlation between positions and add it to word embedding in both source and target sides. The positional embedding is represented by a sinusoidal function of different frequencies to encode different positions. Specifically, the positional encoding e_{pos} is computed as $e_{pos}(j, k) = \sin(j/10000^{k/d})$ (for even k) or $\cos(j/10000^{k/d})$ (for odd k), where j is the position index and k is the dimension index of the embedding vector.

[4] https://wit3.fbk.eu/

[5] http://www.statmt.org/wmt14/translation-task

Source:	ich weiß , dass wir es können , und soweit es mich betrifft ist das etwas , was die welt jetzt braucht .
Target:	i know that we can , and as far as i 'm concerned , that 's something the world needs right now .
ART:	i know that we can , and as far as i 'm concerned , that 's something that the world needs now .
NART w/o Hints:	i know that we can it , , as as as as it it it is , it 's something that the world needs now .
NART w/ Hints:	i know that we can do it and as as 's m concerned , that 's something that the world needs now .
Source:	jeden morgen fliegen sie 240 kilometer zur farm .
Target:	every morning , they fly 240 miles into the farm .
ART:	every morning , they fly 240 miles to the farm .
NART w/o Hints:	every morning , you fly 240 miles to every morning .
NART w/ Hints:	every morning , they fly 240 miles to the farm .
Source:	aber bei youtube werden mehr als 48 stunden video pro minute hochgeladen .
Target:	but there are over 48 hours of video uploaded to youtube every minute .
ART:	but on youtube , more than 48 hours of video are uploaded per minute .
NART w/o Hints:	but on youtube , uploaded than 48 hours hours of video per minute .
NART w/ Hints:	but on youtube , more than 48 hours video are uploaded per minute .

Table 3: Cases on IWSLT14 De-En.

contains 4.5M parallel sentence pairs. Newstest2014 is used as the test set, and Newstest2013 is used as the validation set. In both datasets, tokens are split into a 32000 word-piece dictionary (Wu et al., 2016) which is shared in source and target languages.

Model specifications For the WMT14 dataset, we use the default network architecture of the `base` Transformer model in Vaswani et al. (2017), which consists of a 6-layer encoder and 6-layer decoder. The size of hidden nodes and embeddings are set to 512. For the IWSLT14 dataset, we use a smaller architecture, which consists of a 5-layer encoder, and a 5-layer decoder. The size of hidden states and embeddings are set to 256 and the number of heads is set to 4.

Hyperparameter specifications Hyperparameters $(\tau, \gamma_{st}, \gamma_{tr}, \lambda, \mu)$ are determined to make the scales of three loss components similar after initialization. Specifically, we use $\tau = 0.3, \gamma_{st} = 0.1, \gamma_{tr} = 0.9$, $\lambda = 5.0, \mu = 1.0$ for IWSLT14 De-En, $\tau = 0.3, \gamma_{st} = 0.5, \gamma_{tr} = 0.9, \lambda = 5.0, \mu = 1.0$ for WMT14 De-En and WMT14 En-De.

BLEU scores We use the BLEU score (Papineni et al., 2002) as our evaluation measure. During inference, we set C to $2, -2, 2$ for WMT14 En-De, De-En and IWSLT14 De-En datasets respectively, according to the average lengths of different languages in the training sets. When using the teacher to rescore, we set $B = 4$ and thus have 9 candidates in total. We also evaluate the average per-sentence decoding latencies on one NVIDIA TITAN Xp GPU card by decoding on WMT14 En-De test sets with batch size 1 for our ART teacher model and NART models, and calculate the speedup based on them.

D Case Study

We provide some case studies for the NART models with and without hints in Table 3. From the first case, we can see that the model without hints translates the meaning of *"as far as I'm concerned"* to a set of meaningless tokens. In the second case, the model without hints omits the phrase *"the farm"* and replaces it with a repetitive phrase *"every morning"*. In the third case, the model without hints mistakenly puts the word *"uploaded"* to the beginning of the sentence, whereas our model correctly translates the source sentence. In all cases, hint-based training helps the NART model to generate better target sentences.