Contrastive Conditional Masked Language Model for Non-autoregressive Neural Machine Translation

Anonymous ACL submission

Abstract

Inspired by the success of contrastive learning in natural language processing, we incorporate contrastive learning into the conditional masked language model which is extensively used in non-autoregressive neural machine translation (NAT) that we term Contrastive Conditional Masked Language Model (CCMLM). CCMLM optimizes the similarity of several different representations of the same token in the same sentence, resulting in a richer and more robust representation. We propose two methods to obtain various representations: Contrastive Common Mask and Contrastive Dropout. Positive pairs are various different representations of the same token, while negative pairs are representations of different tokens. In the feature space, the model with contrastive loss pulls positive pairs together and pushes negative pairs away. We conduct extensive experiments on four translation directions with different data sizes. The results demonstrate that CCMLM showed a consistent and significant improvement with margins ranging from 0.80-1.04 BLEU and is state-of-the-art on WMT'16 Ro-En (34.18 BLEU).

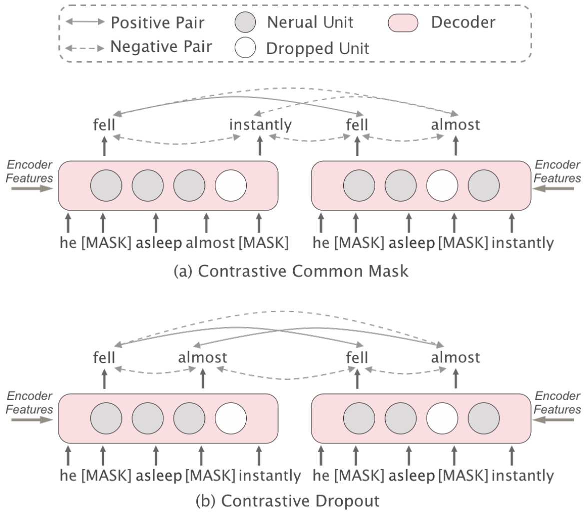

Figure 1: Methods to construct positive pairs and negative pairs. (a) Contrastive Common Mask. (b) Contrastive Dropout.

1 Introduction

Neural machine translation has developed rapidly with the development of deep learning. The traditional neural machine translation models (Sutskever et al., 2014; Bahdanau et al., 2015; Wu et al., 2016; Vaswani et al., 2017) are autoregressive (AT), which means that they predict target tokens one by one based on source tokens and previously predicted tokens. This dependence leads to the limitation of translation speed, and the time required for translation is directly proportional to the sentence length.

Recently, non-autoregressive machine translation (NAT) becomes a research hotspot. The non-autoregressive generation mode eliminates token dependency in the target sentence and generates all tokens in parallel, considerably improving translation speed. However, the increase in speed is accompanied with a decrease in translation quality. Many iterative models have been developed to make a trade-off between translation speed and quality. The iterative model improves translation quality by continually and iteratively optimizing the generated target sentence. The iterative model is usually to predict the masked token in the target sentence, such as BERT (Devlin et al., 2019).

The masked tokens are usually chosen at random. A sentence can be masked in a variety of ways. In different masked sequences of the same sentence, the representation of the same masked token should be similar because they are from the same token and have the same semantics in a similar context (the same source sentence and the different masked results of the same target sentence). We think about how to make these different representations of the same token more similar. Inspired by the successful use of contrastive learning in NLP pre-trained models (e.g., Gao et al., 2021), We explore combining contrastive learning and the conditional masked language model, treating different representations of the same masked token as positive pairs and representations of different tokens as negative pairs. We pull in positive pairs and push out negative pairs using contrastive learning.

As illustrated in Figure 1, we propose two strategies for constructing positive pairs in this paper. Contrastive Common Mask is a method that utilizes representations of the same token in different masked sequences of the same sentence. As shown in Figure 1(a), *"fell"* is masked both in *"he [mask] asleep almost [mask]"* and *"he [mask] asleep [mask] instantly"*, which are different randomly masked results of *"he fell asleep almost instantly"*. The other is inspired by Gao et al. (2021), where we feed the same input to the decoder twice and get two different representations due to the dropout setting, which we call Contrastive Dropout. The two representations of the same token should be similar, as shown in Figure 1(b).

We use the constructed positive and negative pairs to calculate the contrastive loss and jointly optimize it with the cross-entropy loss. We verify the effectiveness of our model in four translation directions of two standard datasets with varying data sizes. Experiments show that our model beats CMLM with 0.80-1.04 BLEU margins at the same translation speed. It also outperforms other CMLM-based models and beats the state-of-the-art NAT model on WMT'16 Ro-En (34.18 BLEU). We will make our code publicly available.

The main contributions of this work can be concluded as follows:

- To the best of our knowledge, our work is the first effort to combine token-level contrastive learning and the conditional masked language model.
- We propose two methods to construct positive pairs for the contrastive conditional masked language model: Contrastive Common Mask and Contrastive Dropout.
- Our model CCMLM achieves a consistent and significant improvement with margins ranging from 0.80-1.04 BLEU in four translation directions and is state-of-the-art on WMT'16 Ro-En (34.18 BLEU).

2 Preliminaries

Non-Autoregressive Machine Translation The machine translation task is defined as generating a target sentence $\boldsymbol{Y} = \{y_1, \ldots, y_{T_y}\}$ under the condition of a given source sentence $\boldsymbol{X} = \{x_1, \ldots, x_{T_x}\}$. Most models factorize the conditional probability $P_\theta(\boldsymbol{Y} \mid \boldsymbol{X})$ by:

$$P_\theta(\boldsymbol{Y} \mid \boldsymbol{X}) = \prod_{t=1}^{T_y} P\left(y_t \mid \boldsymbol{Y_{<t}}, \boldsymbol{X}; \theta\right), \tag{118}$$

where $\boldsymbol{Y_{<t}}$ denotes the target tokens generated before timstep t, T_y denotes the target sentence length and θ denotes the model parameters. This autoregressive mode makes the decoding process time-consuming, because the target tokens are generated step by step.

Non-autoregressive models break the conditional dependency between target tokens and generate all target tokens in parallel. The conditional probability $P_\theta(\boldsymbol{Y} \mid \boldsymbol{X})$ is factorized as:

$$P_\theta(\boldsymbol{Y} \mid \boldsymbol{X}) = \prod_{t=1}^{T_y} P\left(y_t \mid \boldsymbol{X}; \theta\right).$$

Although the assumption of conditional independence improves the translation speed, it also impairs the model performance.

The Conditional Masked Language Model Ghazvininejad et al. (2019) introduced the conditional masked language model (CMLM), which takes the masked language model as training objective (Devlin et al., 2019) and generate the target sentence through iterative refinement. The objective function allows the model to learn to predict any arbitrary subset of the target sentence in parallel:

$$P_\theta(\boldsymbol{Y_{ms}} \mid \boldsymbol{X}, \boldsymbol{Y_{obs}}) = \prod_{t=1}^{T_{Y_{ms}}} P\left(y_t \mid \boldsymbol{X}, \boldsymbol{Y_{obs}}; \theta\right),$$

where $\boldsymbol{Y_{ms}}$ is a set of target tokens randomly replaced by the special token `[mask]`, and $\boldsymbol{Y_{obs}}$ is the set of reserved target tokens.

Contrastive Learning Contrastive learning algorithms compare positive and negative pairs to learn representations, and they have achieved remarkable success in computer vision, natural language processing, recommendation systems, and

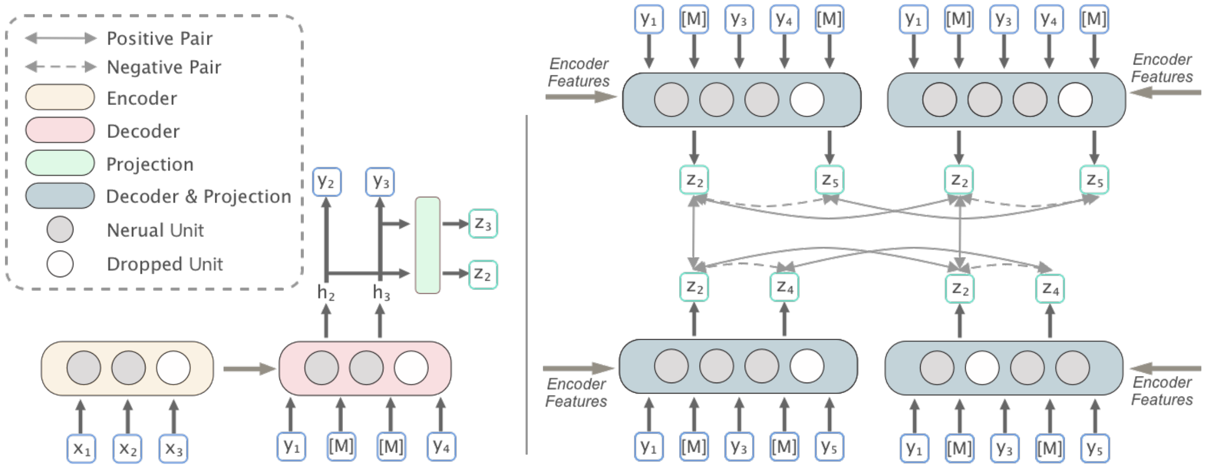

Figure 2: The overall framework of our fully CCMLM model. **[M]** is the special token `[mask]`. Left figure: the model structure. Right figure: the combination of Contrastive Common Mask and Contrastive Dropout. For different masked results of the same sentence, it is Contrastive Common Mask when combined horizontally, and Contrastive Dropout when combined vertically.

other fields. It pulls positive pairs together and pushes negative pairs apart in the feature space. For positive and negative pairs, different algorithms and applications use different selection strategies.

We assume that there is a mini-batch of $2N$ examples. For example i, there is a positive pair $(i, j(i))$, and the other $2(N-1)$ examples are treated as negative examples of i. The training objective for example i is:

$$\ell_i = -\log \frac{\exp\left(\mathrm{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_{j(i)}\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp\left(\mathrm{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/\tau\right)},$$

where z denotes the example feature, τ is a temperature hyperparameter and sim is the similarity function (e.g. the cosine similarity: $\mathrm{sim}(\boldsymbol{z_i}, \boldsymbol{z_{j(i)}}) = \boldsymbol{z_i}^\top \boldsymbol{z_{j(i)}} / \|\boldsymbol{z_i}\| \|\boldsymbol{z_{j(i)}}\|$).

3 Methodology

In this chapter, we incorporate contrastive learning into NAT. We begin by introducing the structure of our model CCMLM, followed by two positive pair construction methods for contrastive learning, and lastly, the training objective combined with the contrastive loss. Figure 2 shows the overall framework.

3.1 Model

We use the standard CMLM as our base model. The encoder is a standard transformer encoder, and the decoder is a transformer decoder without the causal mask. As the token representation, we utilize the output of the last layer of the decoder, which is denoted as $\boldsymbol{h}$. A projection head f_{proj} maps the representation $\boldsymbol{h}$ into a vector representation $\boldsymbol{z}$ that is more suitable for the contrastive loss. Such a projection head has been shown to be important in improving the representation quality of the layer before it (Chen et al., 2020). This projection head is implemented as a multi-layer perceptron with a single hidden layer. We formulate the process of obtaining $\boldsymbol{z}$ as follows:

$$\begin{aligned} \boldsymbol{h} &= f_{\mathrm{CMLM}}\left(\boldsymbol{Y_{obs}}, \boldsymbol{X}; \theta\right), \\ \boldsymbol{z} &= f_{\mathrm{proj}}\left(\boldsymbol{h}\right). \end{aligned}$$

3.2 Contrastive Learning

Positive pairs are different representations of the same token in the same sentence, while negative pairs are representations of other tokens in the same mini-batch. For the acquisition of different representations of the same token, we adopt two methods. One is to randomly mask the same sentence twice in a row, and the tokens that are masked twice constitute a positive pair, which we call Contrastive Common Mask. The other is inspired by Gao et al. (2021) and simply feeds the same input to the decoder twice. We can obtain two different representations of the same token as positive pairs by applying the standard dropout twice, which we call Contrastive Dropout.

Contrastive Common Mask During training, the model randomly masks some of the tokens from the target sentence. We perform this process on the same target sentence twice and get two sets of results, $\{\boldsymbol{Y_{obs_1}}, \boldsymbol{Y_{ms_1}}\}$ and $\{\boldsymbol{Y_{obs_2}}, \boldsymbol{Y_{ms_2}}\}$. And we get $\boldsymbol{z}^{(m_1)}$ and $\boldsymbol{z}^{(m_2)}$ as follows using different decoder inputs:

$$
\begin{aligned}
\boldsymbol{h}^{(m_1)} &= f_{\text{CMLM}}\left(\boldsymbol{Y_{obs_1}}, \boldsymbol{X}; \theta\right), \\
\boldsymbol{z}^{(m_1)} &= f_{\text{pro}}\left(\boldsymbol{h}^{(m_1)}\right),
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{h}^{(m_2)} &= f_{\text{CMLM}}\left(\boldsymbol{Y_{obs_2}}, \boldsymbol{X}; \theta\right), \\
\boldsymbol{z}^{(m_2)} &= f_{\text{pro}}\left(\boldsymbol{h}^{(m_2)}\right).
\end{aligned}
$$

Contrastive Dropout There are dropout modules in the fully-connected layers and multi-head attention layers. Due to their randomness, we will get different features if we feed the same input sentence into the model multiple times. Similarly, with the same decoder input and different dropout parameters, we get $\boldsymbol{z}^{(d_1)}$ and $\boldsymbol{z}^{(d_2)}$ as follows :

$$
\begin{aligned}
\boldsymbol{h}^{(d_1)} &= f_{\text{CMLM}}\left(\boldsymbol{Y_{obs}}, \boldsymbol{X}; \theta, \theta_{drop_1}\right), \\
\boldsymbol{z}^{(d_1)} &= f_{\text{pro}}\left(\boldsymbol{h}^{(d_1)}\right),
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{h}^{(d_2)} &= f_{\text{CMLM}}\left(\boldsymbol{Y_{obs}}, \boldsymbol{X}; \theta, \theta_{drop_2}\right), \\
\boldsymbol{z}^{(d_2)} &= f_{\text{pro}}\left(\boldsymbol{h}^{(d_2)}\right).
\end{aligned}
$$

where θ_{drop_2} and θ_{drop_2} denote different dropout masks.

If we combine these two construction methods, we get four sets of features, $\boldsymbol{z}^{(m_1,d_1)}$, $\boldsymbol{z}^{(m_1,d_2)}$, $\boldsymbol{z}^{(m_2,d_1)}$ and $\boldsymbol{z}^{(m_2,d_2)}$.

Contrastive Loss Now that we have different representations of the same token in the same sentence, we use it to calculate the loss of contrastive learning. Let $\boldsymbol{Y_1}$ and $\boldsymbol{Y_2}$ represent two types of randomly masked tokens for the same sentence, which may or may not be the same, $\boldsymbol{z_1}$ and $\boldsymbol{z_1}$ denote the corresponding features. Let $N = |\boldsymbol{Y_1} \cap \boldsymbol{Y_2}|$ denote the number of common masked tokens. We select the representations of common masked tokens from $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ to form $\boldsymbol{Z}$, where $|\boldsymbol{Z}| = 2N$. Let $i, k \in I \equiv \{1 \ldots 2N\}$ be the index of one representation of an arbitrary token, $j(i) \in I$ be index of the other representation for the same token. Then the contrastive loss is given by:

$$
\begin{aligned}
\mathcal{L}_{con} &= \sum_{i \in I} \mathcal{L}_i \\
&= -\sum_{i \in I} \log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_{j(i)}\right)/\tau\right)}{\sum_{k \neq i} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/\tau\right)}.
\end{aligned}
$$

As shown above, for both $\boldsymbol{Y_{ms_1}}$ and $\boldsymbol{Y_{ms_2}}$, we get two representations for contrastive learning, $\boldsymbol{z}^{(m_1,d_1)}$, $\boldsymbol{z}^{(m_1,d_2)}$ and $\boldsymbol{z}^{(m_2,d_1)}$, $\boldsymbol{z}^{(m_2,d_2)}$, respectively. Different representation combinations are used to calculate the different losses of contrastive learning. For the Contrastive Common Mask, we get two losses:

$$
\begin{aligned}
\mathcal{L}_m^1 &= \mathcal{L}_{con}(\boldsymbol{z}^{(m_1,d_1)}, \boldsymbol{z}^{(m_2,d_1)}), \\
\mathcal{L}_m^2 &= \mathcal{L}_{con}(\boldsymbol{z}^{(m_1,d_2)}, \boldsymbol{z}^{(m_2,d_2)}).
\end{aligned} \tag{1}
$$

For the Contrastive Dropout, we can also get two losses:

$$
\begin{aligned}
\mathcal{L}_d^1 &= \mathcal{L}_{con}(\boldsymbol{z}^{(m_1,d_1)}, \boldsymbol{z}^{(m_1,d_2)}), \\
\mathcal{L}_d^2 &= \mathcal{L}_{con}(\boldsymbol{z}^{(m_2,d_1)}, \boldsymbol{z}^{(m_2,d_2)}).
\end{aligned} \tag{2}
$$

3.3 Training Losses

Masked Language Model CMLM-based models are optimized by cross-entropy loss over every masked token in target sentence. We calculate losses for both $\{\boldsymbol{Y_{obs_1}}, \boldsymbol{Y_{ms_1}}\}$ and $\{\boldsymbol{Y_{obs_2}}, \boldsymbol{Y_{ms_2}}\}$ by:

$$
\begin{aligned}
\mathcal{L}_{ce}^1 &= -\sum_{t=1}^{T_{y_{mask_1}}} \log P\left(y_t \mid \boldsymbol{X}, \boldsymbol{Y_{obs_1}}; \theta\right), \\
\mathcal{L}_{ce}^2 &= -\sum_{t=1}^{T_{y_{mask_2}}} \log P\left(y_t \mid \boldsymbol{X}, \boldsymbol{Y_{obs_2}}; \theta\right).
\end{aligned} \tag{3}
$$

Length Predict The length of the target sentence must be known in advance for CMLM-based models to predict the entire sentence in parallel. Also, we follow Ghazvininejad et al. (2019) and add a special token [LENGTH] to the encoder. The model uses the decoder output of [LENGTH] to predict the length of the target sentence. The length loss is:

$$
\mathcal{L}_{len} = -\sum_{i}^{L_{\max}} P(i = T_y) \log P(T_y \mid X), \tag{4}
$$

where $L_{\max}$ represents the maximum length of the target sentence.

Training Objective During the training of CCMLM, the model can be optimized by jointly minimizing the contrastive loss and translation loss. As the training objective, we add up the above-mentioned losses, two cross-entropy losses for translation as (3), four contrastive losses for optimizing feature space as (1) and (2), and one length loss for predicting target length as (4):

$$\mathcal{L} = \frac{1}{2}\left(\mathcal{L}_{ce}^{1} + \mathcal{L}_{ce}^{2}\right) + \mathcal{L}_{len} \\ + \frac{\alpha}{4}\left(\mathcal{L}_{m}^{1} + \mathcal{L}_{m}^{2} + \mathcal{L}_{d}^{1} + \mathcal{L}_{d}^{2}\right).$$

where α is a hyper-parameter to control the intensity of contrastive losses.

4 Experiments

4.1 Experimental Settings

Dataset We evaluate our models on four directions from two standard datasets with different training data sizes widely used in previous NAT studies: WMT'16 En-Ro (610K sentence pairs), WMT'14 En-De (4.5M sentence pairs). All datasets are tokenized into subword units by joint BPE (Sennrich et al., 2016). We use the same preprocessed data as Kasai et al. (2020) for a fair comparions with other models (WMT'16 En-Ro: Lee et al. (2018); WMT'14 En-De: Vaswani et al. (2017)). We evaluate performance with BLEU (Papineni et al., 2002) for all language pairs.

Sequence-Level Knowledge Distillation We use sequence-level knowledge distillation (Kim and Rush, 2016) as previous works on non-autoregressive translation (e.g., Gu et al., 2018; Ghazvininejad et al., 2019). Since the performance of the AT teacher will affect the final performance of the NAT student model (Wang et al., 2019), we used the distillation data provided by Kasai et al. (2020). They are produced by standard left-to-right transformer models (transformer large for En-De, transformer base for En-Ro) for a fair comparison.

Hyperparameters We follow the hyperparameters for a transformer base (Vaswani et al., 2017; Ghazvininejad et al., 2019; Kasai et al., 2020). The projection head is implemented as a multi-layer perceptron with a single hidden layer of size 256 and output vector of size 64. Please see Appendix A for details of other hyperparameters. Our code is based on `CMLM`[1] and `DisCo`[2].

Baselines We adopt Transformer (AT) and existing NAT models for comparison. Table 1 for more details. NAT models can be divided into fully NAT models and iterative NAT models. See Iterative NAT models with enough number of iterations generally outperform fully NAT models. Noisy parallel decoding (NPD) is an important technique for fully NAT to improve the performance of the model, which requires an additional AT model for re-ranking. The models trained with CTC loss are usually better than the models trained with cross-entropy loss because of its inherent de-duplication mechanism. The current state-of-the-art model is the Imputer, which combines the CTC and the masked language model.

4.2 Overall Results

Table 1 shows the main results on WMT'14 En-De and WMT'16 En-Ro test sets. Compared to existing NAT models, except for Imputer, our model significantly and consistently improves the quality of translation across four translation directions. Furthermore, our model outperforms the Imputer on the WMT'16 Ro-En and is state-of-the-art (34.18 BLEU).

Our model outperforms standard CMLM with margins from 0.80 to 1.04 BLEU points, demonstrating the usefulness of our methods. It is also significantly superior to other CMLM-based models, such as SMART, CMLM+LFR, CMLM+PMG, and MvCR. It is worth noting that the contrastive module is only used in the training process and is discarded during inference. Therefore the translation latency is not increased.

4.3 Analysis

Comparison of Different Iterations Iterative NAT can effectively improve model performance by increasing the number of iterations. Naturally, the larger the number of iterations is, the slower the translation speed is. Therefore we need strike a balance between translation speed and model performance. One, four, and ten iterations are widely employed for CMLM-based models. We compare the model performance of CMLM and CCMLM in the four translation directions in the Table 2. As we can see, CCMLM constantly beats CMLM in every iteration step and task, and the fewer the iterations, the more significant the improvement. Furthermore, the CCMLM performance with four iterations outperforms the CMLM performance with ten iterations, which the other previous CMLM-based models do not achieve.

Repeated Translation In NAT, a major issue is repeated translation, which means that illogical consecutive repeated tokens frequently exist in

[1] https://github.com/facebookresearch/Mask-Predict
[2] https://github.com/facebookresearch/DisCo

		Mdels	Iter.	En-De	De-En	En-Ro	Ro-En
AT		Transformer	T	27.38	31.78	34.16	34.46
Fully NAT	w/ NPD	NAT-FT (m=100) (Gu et al., 2018)		19.17	23.20	29.79	31.44
		imit-NAT (m=7)(Wei et al., 2019)		24.15	27.28	31.45	31.81
		NAT-HINT (m=9) (Li et al., 2019)		25.20	29.52	-	-
		Flowseq (m=30) (Ma et al., 2019)		25.31	30.68	32.20	32.84
		NAT-DCRF (m=9) (Sun et al., 2019)		26.07	29.68	-	-
		GLAT (m=7) (Qian et al., 2021)		26.55	31.02	32.87	33.51
		AXE (Ghazvininejad et al., 2020a)		23.53	27.90	30.75	31.54
		OAXE (Du et al., 2021)		26.10	30.20	32.40	33.30
	w/ CTC	NAT-CTC (Saharia et al., 2020)		25.70	28.10	32.20	31.60
		Imputer (Saharia et al., 2020)		25.80	28.40	32.30	31.70
		GLAT (Qian et al., 2021)		26.39	29.54	32.79	33.84
		Tricks (Gu and Kong, 2021)		27.49	31.10	33.79	33.87
Iterative NAT	w/ CTC	Imputer (Saharia et al., 2020)	8	**28.20**	**31.80**	**34.40**	**34.10**
		CMLM (Ghazvininejad et al., 2019)	10	27.03	30.53	33.08	33.31
		SMART (Ghazvininejad et al., 2020b)	10	27.65	31.27	-	-
		ENGINE (Tu et al., 2020)	10	-	-	-	34.04
		DisCo (Kasai et al., 2020)	Adv.	27.34	31.31	33.22	33.25
		MvCR (Xie et al., 2021)	10	27.39	31.18	33.38	33.56
		CMLM+PMG (Ding et al., 2021a)	10	27.60	-	-	33.80
		CMLM+LFR (Ding et al., 2021b)	10	27.80	-	-	33.90
	Ours	CCMLM	10	**27.93**	**31.57**	**33.88**	**34.18**

Table 1: Performance (BLEU) comparison between our proposed model CCMLM and existing models. **Iter.** denotes the number of iterations, **Adv.** means adaptive and m is the number of re-ranking candidates.

Model		En-De	De-En	En-Ro	Ro-En
CMLM	1	18.05	21.83	27.32	28.20
	4	25.94	29.90	32.53	33.23
	10	27.03	30.53	33.08	33.31
CCMLM	1	20.19	25.02	30.90	31.77
	4	27.28	31.18	33.45	33.83
	10	27.93	31.57	33.88	34.18

Table 2: Performance (BLEU) comparison between CCMLM and CMLM with different iterations.

Model		1	4	10
CMLM	Short	0.84	0.09	0.04
	Long	8.10	0.79	0.27
	All	4.60	0.45	0.16
CCMLM	Short	0.39	0.06	0.02
	Long	4.01	0.41	0.18
	All	2.29	0.25	0.10

Table 3: The average number of consecutive repeated tokens per sentence with different iterations on the WMT'16 En-Ro test set.

translated sentences. This is especially noticeable in long sentences. We calculate the average number of consecutive repeated tokens per sentence on the WMT'16 En-Ro test set. Table 3 shows the results. According on whether the sentence length is fewer than 25, all samples are divided into Short and Long groups. It can be seen that after the addition of the contrastive module, the number of consecutive repeated tokens is significantly reduced.

Different Source Length We divide the samples into different length buckets based on the source sentence length to assess the model ability to translate sentences of various lengths. Figure 3 shows the results on the test set of WMT'16 En-Ro with one iteration. As the length of the source sentence increases, the performance of CMLM drops

Models	Iter.	En-De	De-En	En-Ro	Ro-En
CMLM	10	27.03	30.53	33.08	33.31
+ Common Mask	1	19.71	24.29	30.16	31.69
	4	27.05	30.86	33.31	34.05
	10	27.76(+0.73)	31.52(+0.99)	33.63(+0.55)	**34.32(+1.01)**
+ Dropout	1	18.68	24.00	29.93	30.81
	4	26.61	30.61	33.14	33.33
	10	27.18(+0.15)	31.14(+0.61)	33.41(+0.33)	33.59(+0.28)
CCMLM	10	**27.93(+0.90)**	**31.57(+1.04)**	**33.88(+0.80)**	34.18(+0.87)

Table 4: Ablation experiments on two methods of constructing positive pairs.

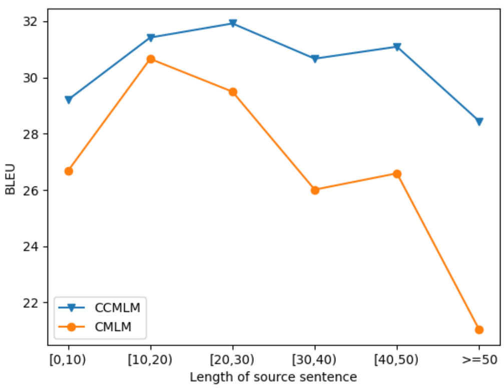

Figure 3: The BLEU points on the test set of WMT'16 En-Ro over sentences in different length buckets.

quickly, whereas the performance of our model CCMLM decrease is is noticeably slower. The longer the source sentences are, the more considerable the margin between CCMLM and CMLM is.

Complementary to Related Work In the course of our work, we discovered MvCR (Xie et al., 2021), which is relevant to our work. MvCR introduces Shared Mask Consistency and Model Consistency through bidirectional Kullback-Leibler (KL) divergence. Shared Mask Consistency is similar to the idea of Contrastive Common Mask proposed by us. The difference is that we use the last layer of Decoder and the method of contrastive learning, while they use the predicted distributions and the method of consistency regularization. And we do not use the features of an online model and an average model for contrastive learning, while they do not use the consistency between different dropout parameters.

Contrastive Layer	En-Ro
6	**33.88**
5	33.64
4	33.51
6+5 w/shared-head	33.59
6+5 w/different-heads	33.34
word embed	33.65

Table 5: Performances on WMT'16 En-Ro with different contrastive layers.

4.4 Ablation Study

Common Mask vs. Dropout As shown in Table 4, we test the individual contributions of the two contrastive methods in the four translation directions. It can be seen that when Contrastive Common Mask and Contrastive Dropout are used alone, the performance of the model has also been improved to varying degrees compared with the baseline CMLM. In the WMT'16 Ro-En task, CMLM with Contrastive Common Mask is state-of-the-art (34.32 BLEU). Furthermore, the improvement of Contrastive Common Mask is more significant than that of Contrastive Dropout. On the one hand, we think that the decoder input context of Contrastive Common Mask is different, allowing the model to explicitly capture the similarity of generated features in different contexts and making features richer and more robust, whereas dropout is only implicitly optimized by the parameters of the model which is a little weaker. On the other hand, Contrastive Common Mask also needs to feed the sample to the model twice, which means that part of Contrastive Dropout is included in Contrastive Common Mask. When we combine the two methods, except in the WMT'16 Ro-En task, the model

α	0.3	0.5	1.0	2.0
En-Ro	33.41	33.54	**33.88**	33.81

Table 6: Performances on WMT16'En-Ro with different contrastive loss weights α.

Dropout	0.1	0.2	0.3	0.4	0.5
En-Ro	33.19	33.69	**33.88**	33.79	33.41

Table 7: Performances on WMT16'En-Ro with different dropout rates.

performance has been improved again.

Contrastive Layer For contrastive learning, we can obtain various representations from different layers of the Decoder. The impact of different layer representations is discussed here. First, we choose the output of the Decoder's fourth, fifth, and sixth layers independently. Second, we combine the contrastive losses of the fifth and the sixth layers together. The projection heads for these two layers can be same or different. Finally, we also compare the word embedding output of the Decoder. Table 5 shows the result. Using representations of the sixth layer alone has the best performance, followed by word embedding. The shallower the representation used, the worse the performance is. Combining the contrastive losses for different layers do not helpful, whether using the same head or different heads.

Effect of α α controls the intensity of contrastive losses. To further understand the role of contrastive losses, we try out different values in Table 6 and observe that all the variants outperform the baseline CMLM. The best choice of contrastive losses weight is $\alpha = 1.0$.

Dropout Probability Since we use dropout explicitly and implicitly in Contrastive Dropout and Contrastive Common Mask, respectively, we conduct ablation experiments on WMT'16 En-Ro with different dropout rates in $\{0.1, 0{,}2, 0.3, 0.4, 0.5\}$. As Table 7 shows, dropout rates that are too high or too low hurt the performance of the model. The best choice of dropout rate is 0.3.

5 Related Work

In order to speed up the translation process, Gu et al. (2018) introduced non-autoregressive translation. We divide NAT models into three types according to the training loss. The first is the conditional independent language model, which include: enhancing the decoder input (Guo et al., 2019; Bao et al., 2019; Ran et al., 2019), enhancing the decoder output (Wang et al., 2019; Sun et al., 2019), learning or transforming from autoregressive model (Li et al., 2019; Guo et al., 2020a; Sun and Yang, 2020; Tu et al., 2020; Liu et al., 2020), latent variable-based model (Lee et al., 2018, 2020; Shu et al., 2020). The second is the conditional masked language model, inculde: strong baseline model CMLM (Ghazvininejad et al., 2019), disentangled context transformer (Ding et al., 2020), jointly masked sequence-to-sequence model (Guo et al., 2020b), semi-autoregressive training (Ghazvininejad et al., 2020b), increasing the mask ratio gradually (Qian et al., 2021), learning autoregressive model (Tu et al., 2020), progressive multi-granularity training (Ding et al., 2021a), using the bidirection distillation data (Ding et al., 2021b), improving the alignment of cross entropy (Ghazvininejad et al., 2020a; Du et al., 2021). The last is the CTC model, which includes CTC (Libovický and Helcl, 2018) and Imputer (Saharia et al., 2020) which combines the CTC and the masked language model. Other excellent approaches include: flow-based generative model (Ma et al., 2019), adding a lite autoregressive module (Kong et al., 2020), training with monolingual data (Zhou and Keung, 2020), incorporating the pre-trained model (Guo et al., 2020c), and tricks of the trade (Gu and Kong, 2021).

6 Conclusion

In this work, we propose CCMLM, which is the first effort to combine token-level contrastive learning and the conditional masked language model. CCMLM optimizes the similarity of different representations of the same token in the same sentence by contrastive learning. We propose Contrastive Common Mask and Contrastive Dropout to construct positive pairs, using different random masks and dropout masks, respectively. Our model achieves consistent and significant improvement in the four translation tasks and is state-of-the-art on WMT'16 Ro-En. The lightweight contrastive module is removed during inference, so it does not affect the translation speed.

In the future, we will focus on combining the idea with the CTC and the pre-trained masked language model.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yu Bao, Hao Zhou, Jiangtao Feng, Mingxuan Wang, Shujian Huang, Jiajun Chen, and Lei Li. 2019. Non-autoregressive transformer by position learning. *ArXiv preprint*, abs/1911.10677.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

PK Diederik and B Jimmy. 2014. Adam: A method for stochastic optimization. iclr. *ArXiv preprint*, abs/1412.6980.

Liang Ding, Longyue Wang, Xuebo Liu, Derek F. Wong, Dacheng Tao, and Zhaopeng Tu. 2021a. Progressive multi-granularity training for non-autoregressive translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2797–2803, Online. Association for Computational Linguistics.

Liang Ding, Longyue Wang, Xuebo Liu, Derek F. Wong, Dacheng Tao, and Zhaopeng Tu. 2021b. Rejuvenating low-frequency words: Making the most of parallel data in non-autoregressive translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3431–3441, Online. Association for Computational Linguistics.

Liang Ding, Longyue Wang, Di Wu, Dacheng Tao, and Zhaopeng Tu. 2020. Context-aware cross-attention for non-autoregressive translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4396–4402, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. *ArXiv preprint*, abs/2106.05093.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv preprint*, abs/2104.08821.

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

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, and Luke Zettlemoyer. 2020b. Semi-autoregressive training improves mask-predict decoding. *ArXiv preprint*, abs/2001.08785.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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

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 *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3723–3730. AAAI Press.

Junliang Guo, Xu Tan, Linli Xu, Tao Qin, Enhong Chen, and Tie-Yan Liu. 2020a. Fine-tuning by curriculum learning for non-autoregressive neural machine translation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7839–7846. AAAI Press.

Junliang Guo, Linli Xu, and Enhong Chen. 2020b. Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 376–385, Online. Association for Computational Linguistics.

Junliang Guo, Zhirui Zhang, Linli Xu, Hao-Ran Wei, Boxing Chen, and Enhong Chen. 2020c. Incorporating BERT into parallel sequence decoding with adapters. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Xiang Kong, Zhisong Zhang, and Eduard Hovy. 2020. Incorporating a local translation mechanism into non-autoregressive translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1067–1073, Online. Association for Computational Linguistics.

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.

Jason Lee, Raphael Shu, and Kyunghyun Cho. 2020. Iterative refinement in the continuous space for non-autoregressive neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1006–1015, Online. Association for Computational Linguistics.

Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Hint-based training for non-autoregressive machine translation. 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 5708–5713, Hong Kong, China. Association for Computational Linguistics.

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.

Jinglin Liu, Yi Ren, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. Task-level curriculum learning for non-autoregressive neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3861–3867. ijcai.org.

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.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2019. Guiding non-autoregressive neural machine translation decoding with reordering information. *ArXiv preprint*, abs/1911.02215.

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 (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

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.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhi-Hong Deng. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3011–3020.

Zhiqing Sun and Yiming Yang. 2020. An EM approach to non-autoregressive conditional sequence generation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9249–9258. PMLR.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Lifu Tu, Richard Yuanzhe Pang, Sam Wiseman, and Kevin Gimpel. 2020. ENGINE: Energy-based inference networks for non-autoregressive machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2826, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

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*, abs/1609.08144.

Pan Xie, Zexian Li, and Xiaohui Hu. 2021. Mvsr-nat: Multi-view subset regularization for non-autoregressive machine translation. *ArXiv preprint*, abs/2108.08447.

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.

A Hyperparameters

We follow the hyperparameters for a transformer base (Vaswani et al., 2017; Ghazvininejad et al., 2019; Kasai et al., 2020): 6 layers for the encoder and the decoder, 8 attention heads, 512 model dimensions, and 2048 hidden dimensions per layer. Set dropout rate to 0.3 for WMT'16 En-Ro and 0.2 for WMT'16 En-Ro. We sample weights from $\mathcal{N}(0, 0.02)$, initialize biases to zero and set layer normalization parameters to $\beta = 0$, $\gamma = 1$, following the weight initialization scheme from BERT (Devlin et al., 2019). We set weight decay to 0.01 and label smoothing to 0.1 for regularization. We train batches of approximately $2K \cdot 8$ (8 GPUs with 2K per GPU) tokens using Adam (Diederik and Jimmy, 2014) with $\beta = (0.9, 0.999)$ and $\epsilon = 10^{-6}$. We set update frequency to 4 which means accumulate gradients from 4 batches before each update (Ott et al., 2018), and enable mixed precision floating point arithmetic (Micikevicius et al., 2018). The learning rate warms up to $5 \cdot 10^{-4}$ for the first 10K steps, and the decays with the inverse square-root schedule. We train models for 300K steps on 8 NVIDIA TESLA V100 32G GUPs, and average the 10 best checkpoints as the final model. Following the previous works (Ghazvininejad et al., 2019; Kasai et al., 2020), we apply length beam with the size of 5.