Semi-Autoregressive Neural Machine Translation

Chunqi Wang* **Ji Zhang** **Haiqing Chen**
Alibaba Group
{shiyuan.wcq, zj122146, haiqing.chenhq}@alibaba-inc.com

Abstract

Existing approaches to neural machine translation are typically autoregressive models. While these models attain state-of-the-art translation quality, they are suffering from low parallelizability and thus slow at decoding long sequences. In this paper, we propose a novel model for fast sequence generation — the semi-autoregressive Transformer (SAT). The SAT keeps the autoregressive property in global but relieves in local and thus are able to produce multiple successive words in parallel at each time step. Experiments conducted on English-German and Chinese-English translation tasks show that the SAT achieves a good balance between translation quality and decoding speed. On WMT'14 English-German translation, the SAT achieves 5.58× speedup while maintaining 88% translation quality, significantly better than the previous non-autoregressive methods. When produces two words at each time step, the SAT is almost lossless (only 1% degeneration in BLEU score).

1 Introduction

Neural networks have been successfully applied to a variety of tasks, including machine translation. The encoder-decoder architecture is the central idea of neural machine translation (NMT). The encoder first encodes a source-side sentence $\mathbf{x} = x_1 \dots x_m$ into hidden states and then the decoder generates the target-side sentence $\mathbf{y} = y_1 \dots y_n$ from the hidden states according to an autoregressive model

$$p(y_t|y_1 \dots y_{t-1}, \mathbf{x})$$

Recurrent neural networks (RNNs) are inherently good at processing sequential data. Sutskever et al. (2014); Cho et al. (2014) successfully applied RNNs to machine translation. Bahdanau et al. (2014) introduced attention mechanism into the encoder-decoder architecture and greatly improved NMT. GNMT (Wu et al., 2016) further improved NMT by a bunch of tricks including residual connection and reinforcement learning.

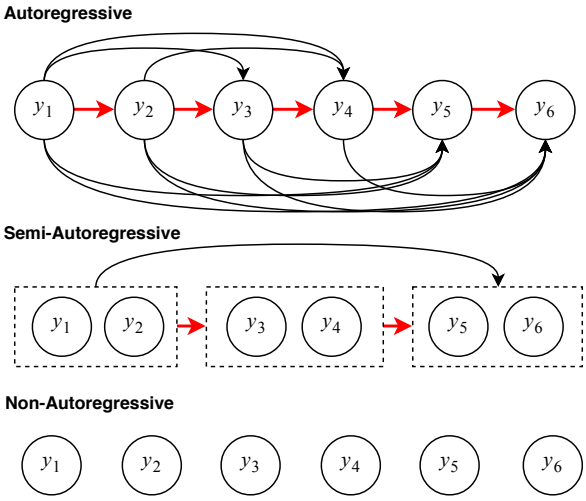

Figure 1: The different levels of autoregressive properties. Lines with arrow indicate dependencies. We mark the longest dependency path with bold red lines. The length of the longest dependency path decreases as we relieve the autoregressive property. An extreme case is *non-autoregressive*, where there is no dependency at all.

The sequential property of RNNs leads to its wide application in language processing. However, the property also hinders its parallelizability thus RNNs are slow to execute on modern hardware optimized for parallel execution. As a result, a number of more parallelizable sequence models were proposed such as ConvS2S (Gehring et al., 2017) and the Transformer (Vaswani et al., 2017). These models avoid the dependencies between dif-

*Part of this work was done when the author was at Institute of Automation, Chinese Academy of Sciences.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 479–488
Brussels, Belgium, October 31 - November 4, 2018.

ferent positions in each layer thus can be trained much faster than RNN based models. When inference, however, these models are still slow because of the autoregressive property.

A recent work (Gu et al., 2017) proposed a non-autoregressive NMT model that generates all target-side words in parallel. While the parallelizability is greatly improved, the translation quality encounter much decrease. In this paper, we propose the semi-autoregressive Transformer (SAT) for faster sequence generation. Unlike Gu et al. (2017), the SAT is semi-autoregressive, which means it keeps the autoregressive property in global but relieves in local. As the result, the SAT can produce multiple successive words in parallel at each time step. Figure 1 gives an illustration of the different levels of autoregressive properties.

Experiments conducted on English-German and Chinese-English translation show that compared with non-autoregressive methods, the SAT achieves a better balance between translation quality and decoding speed. On WMT'14 English-German translation, the proposed SAT is 5.58× faster than the Transformer while maintaining 88% of translation quality. Besides, when producing two words at each time step, the SAT is almost lossless.

It is worth noting that although we apply the SAT to machine translation, it is not designed specifically for translation as Gu et al. (2017); Lee et al. (2018). The SAT can also be applied to any other sequence generation task, such as summary generation and image caption generation.

2 Related Work

Almost all state-of-the-art NMT models are autoregressive (Sutskever et al., 2014; Bahdanau et al., 2014; Wu et al., 2016; Gehring et al., 2017; Vaswani et al., 2017), meaning that the model generates words one by one and is not friendly to modern hardware optimized for parallel execution. A recent work (Gu et al., 2017) attempts to accelerate generation by introducing a non-autoregressive model. Based on the Transformer (Vaswani et al., 2017), they made lots of modifications. The most significant modification is that they avoid feeding the previously generated target words to the decoder, but instead feeding the source words, to predict the next target word. They also introduced a set of latent variables to model the *fertilities* of source words to tackle the multimodality problem in translation. Lee et al. (2018) proposed another non-autoregressive sequence model based on iterative refinement. The model can be viewed as both a latent variable model and a conditional denoising autoencoder. They also proposed a learning algorithm that is hybrid of lower-bound maximization and reconstruction error minimization.

The most relevant to our proposed semi-autoregressive model is (Kaiser et al., 2018). They first autoencode the target sequence into a shorter sequence of discrete latent variables, which at inference time is generated autoregressively, and finally decode the output sequence from this shorter latent sequence in parallel. What we have in common with their idea is that we have not entirely abandoned autoregressive, but rather shortened the autoregressive path.

A related study on realistic speech synthesis is the parallel WaveNet (Oord et al., 2017). The paper introduced *probability density distillation*, a new method for training a parallel feed-forward network from a trained WaveNet (Van Den Oord et al., 2016) with no significant difference in quality.

There are also some work share a somehow simillar idea with our work: character-level NMT (Chung et al., 2016; Lee et al., 2016) and chunk-based NMT (Zhou et al., 2017; Ishiwatari et al., 2017). Unlike the SAT, these models are not able to produce multiple tokens (characters or words) each time step. Oda et al. (2017) proposed a bit-level decoder, where a word is represented by a binary code and each bit of the code can be predicted in parallel.

3 The Transformer

Since our proposed model is built upon the Transformer (Vaswani et al., 2017), we will briefly introduce the Transformer. The Transformer uses an encoder-decoder architecture. We describe the encoder and decoder below.

3.1 The Encoder

From the source tokens, learned embeddings of dimension d_{model} are generated which are then modified by an additive positional encoding. The positional encoding is necessary since the network does not leverage the order of the sequence by recurrence or convolution. The authors use additive

encoding which is defined as:

$$PE(pos, 2i) = sin(pos/10000^{2i/d_{model}})$$
$$PE(pos, 2i+1) = cos(pos/10000^{2i/d_{model}})$$

where pos is the position of a word in the sentence and i is the dimension. The authors chose this function because they hypothesized it would allow the model to learn to attend by relative positions easily. The encoded word embeddings are then used as input to the encoder which consists of N blocks each containing two layers: (1) a multi-head attention layer, and (2) a position-wise feed-forward layer.

Multi-head attention builds upon scaled dot-product attention, which operates on a query Q, key K and value V:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

where d_k is the dimension of the key. The authors scale the dot product by $1/\sqrt{d_k}$ to avoid the inputs to softmax function growing too large in magnitude. Multi-head attention computes h different queries, keys and values with h linear projections, computes scaled dot-product attention for each query, key and value, concatenates the results, and projects the concatenation with another linear projection:

$$H_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$
$$MultiHead(Q, K, V) = Concat(H_1, \dots H_h)$$

in which $W_i^Q, W_i^K \in \mathbb{R}^{d_{model} \times d_k}$ and $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$. The attention mechanism in the encoder performs attention over itself ($Q = K = V$), so it is also called self-attention.

The second component in each encoder block is a position-wise feed-forward layer defined as:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2$$

where $W_1 \in \mathbb{R}^{d_{model} \times d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff} \times d_{model}}$, $b_1 \in \mathbb{R}^{d_{ff}}$, $b_2 \in \mathbb{R}^{d_{model}}$.

For more stable and faster convergence, residual connection (He et al., 2016) is applied to each layer, followed by layer normalization (Ba et al., 2016). For regularization, dropout (Srivastava et al., 2014) are applied before residual connections.

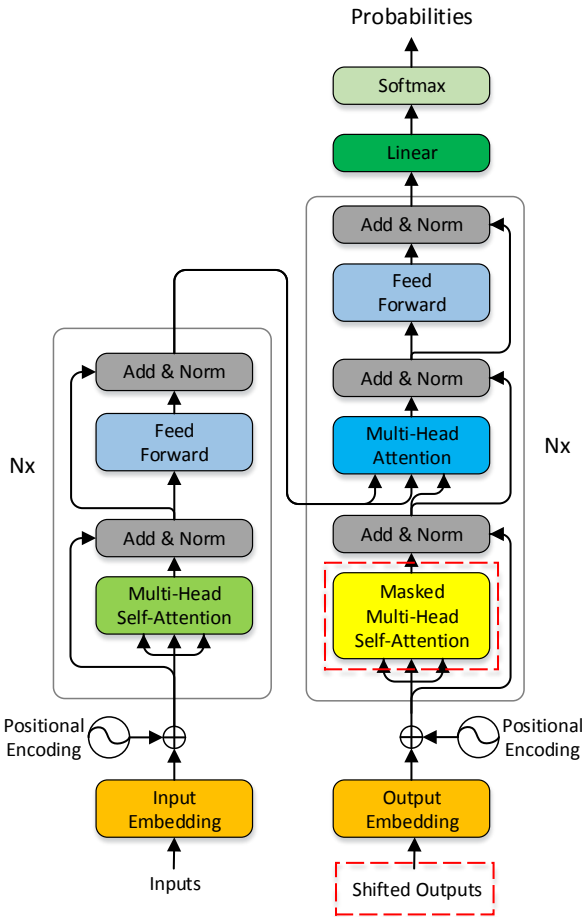

Figure 2: The architecture of the Transformer, also of the SAT, where the red dashed boxes point out the different parts of these two models.

3.2 The Decoder

The decoder is similar with the encoder and is also composed by N blocks. In addition to the two layers in each encoder block, the decoder inserts a third layer, which performs multi-head attention over the output of the encoder.

It is worth noting that, different from the encoder, the self-attention layer in the decoder must be masked with a causal mask, which is a lower triangular matrix, to ensure that the prediction for position i can depend only on the known outputs at positions less than i during training.

4 The Semi-Autoregressive Transformer

We propose a novel NMT model—the Semi-Autoregressive Transformer (SAT)—that can produce multiple successive words in parallel. As shown in Figure 2, the architecture of the SAT is almost the same as the Transformer, except some modifications in the decoder.

4.1 Group-Level Chain Rule

Standard NMT models usually factorize the joint probability of a word sequence $y_1 \dots y_n$ according to the word-level chain rule

$$p(y_1 \dots y_n|\mathbf{x}) = \prod_{t=1}^{n} p(y_t|y_1 \dots y_{t-1}, \mathbf{x})$$

resulting in decoding each word depending on all previous decoding results, thus hindering the parallelizability. In the SAT, we extend the standard word-level chain rule to the group-level chain rule.

We first divide the word sequence $y_1 \dots y_n$ into consecutive groups

$$G_1, G_2, \dots, G_{[(n-1)/K]+1} =$$
$$y_1 \dots y_K, y_{K+1} \dots y_{2K}, \dots, y_{[(n-1)/K]\times K+1} \dots y_n$$

where $[\cdot]$ denotes floor operation, K is the group size, and also the indicator of parallelizability. The larger the K, the higher the parallelizability. Except for the last group, all groups must contain K words. Then comes the group-level chain rule

$$p(y_1 \dots y_n|\mathbf{x}) = \prod_{t=1}^{[(n-1)/K]+1} p(G_t|G_1 \dots G_{t-1}, \mathbf{x})$$

This group-level chain rule avoids the dependencies between consecutive words if they are in the same group. With group-level chain rule, the model no longer produce words one by one as the Transformer, but rather group by group. In next subsections, we will show how to implement the model in detail.

4.2 Long-Distance Prediction

In autoregressive models, to predict y_t, the model should be fed with the previous word y_{t-1}. We refer it as *short-distance prediction*. In the SAT, however, we feed y_{t-K} to predict y_t, to which we refer as *long-distance prediction*. At the beginning of decoding, we feed the model with K special symbols <s> to predict $y_1 \dots y_K$ in parallel. Then $y_1 \dots y_K$ are fed to the model to predict $y_{K+1} \dots y_{2K}$ in parallel. This process will continue until a terminator </s> is generated. Figure 3 gives illustrations for both short and long-distance prediction.

4.3 Relaxed Causal Mask

In the Transformer decoder, the causal mask is a lower triangular matrix, which strictly prevents earlier decoding steps from peeping information from later steps. We denote it as *strict causal mask*. However, in the SAT decoder, strict causal mask is not a good choice. As described in the previous subsection, in long-distance prediction, the model predicts y_{K+1} by feeding with y_1. With strict causal mask, the model can only access to y_1 when predict y_{K+1}, which is not reasonable since $y_1 \dots y_K$ are already produced. It is better to allow the model to access to $y_1 \dots y_K$ rather than only y_1 when predict y_{K+1}.

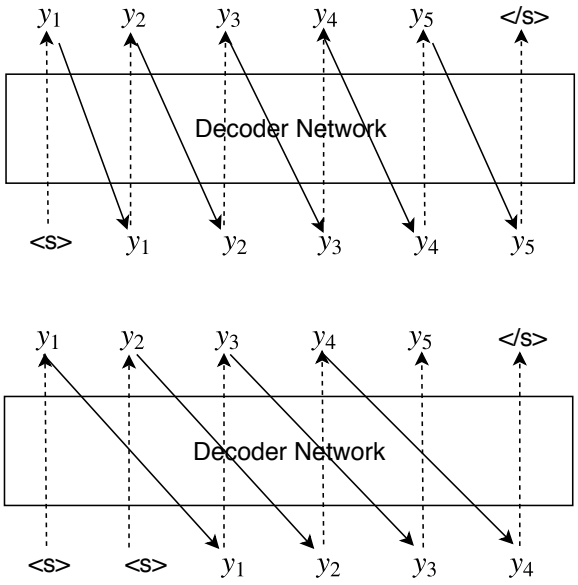

Figure 3: Short-distance prediction (top) and long-distance prediction (bottom).

Therefore, we use a coarse-grained lower triangular matrix as the causal mask that allows peeping later information in the same group. We refer to it as *relaxed causal mask*. Given the target length n and the group size K, relaxed causal mask $M \in \mathbb{R}^{n\times n}$ and its elements are defined below:

$$M[i][j] = \begin{cases} 1 & \text{if } j < ([(i-1)/K]+1) \times K \\ 0 & \text{other} \end{cases}$$

For a more intuitive understanding, Figure 4 gives a comparison between strict and relaxed causal mask.

4.4 The SAT

Using group-level chain rule instead of word-level chain rule, long-distance prediction instead of short-distance prediction, and relaxed causal

$$\begin{bmatrix} 1 & \mathbf{0} & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & \mathbf{0} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & \mathbf{0} \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & \mathbf{1} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & \mathbf{1} \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figure 4: Strict causal mask (left) and relaxed causal mask (right) when the target length $n = 6$ and the group size $K = 2$. We mark their differences in bold.

Model	Complexity	Acceleration
Transformer	$N(a+b)$	1
SAT (beam search)	$\frac{N}{K}a + Nb$	$K(\frac{a+b}{a+Kb})$
SAT (greedy search)	$\frac{N}{K}(a+b)$	K

Table 1: Theoretical complexity and acceleration of the SAT. a denotes the time consumed on the decoder network (calculating a distribution over the target vocabulary) each time step and b denotes the time consumed on search (searching for top scores, expanding nodes and pruning). In practice, a is usually much larger than b since the network is deep.

mask instead of strict causal mask, we successfully extended the Transformer to the SAT. The Transformer can be viewed as a special case of the SAT, when the group size $K = 1$. The non-autoregressive Transformer (NAT) described in Gu et al. (2017) can also be viewed as a special case of the SAT, when the group size K is not less than maximum target length.

Table 1 gives the theoretical complexity and acceleration of the model. We list two search strategies separately: beam search and greedy search. Beam search is the most prevailing search strategy. However, it requires the decoder states to be updated once every word is generated, thus hinders the decoding parallelizability. When decode with greedy search, there is no such concern, therefore the parallelizability of the SAT can be maximized.

5 Experiments

We evaluate the proposed SAT on English-German and Chinese-English translation tasks.

5.1 Experimental Settings

Datasets For English-German translation, we choose the corpora provided by WMT 2014 (Bojar et al., 2014). We use the newstest2013 dataset for development, and the newstest2014 dataset for test. For Chinese-English translation, the corpora we use is extracted from LDC[1]. We chose the NIST02 dataset for development, and the NIST03, NIST04 and NIST05 datasets for test. For English and German, we tokenized and segmented them into subword symbols using byte-pair encoding (BPE) (Sennrich et al., 2015) to restrict the vocabulary size. As for Chinese, we segmented sentences into characters. For English-German translation, we use a shared source and target vocabulary. Table 2 summaries the two corpora.

	Sentence Number	Vocab Size	
		Source	Target
EN-DE	4.5M	36K	36K
ZH-EN	1.8M	9K	34K

Table 2: Summary of the two corpora.

Baseline We use the base Transformer model described in Vaswani et al. (2017) as the baseline, where $d_{model} = 512$ and $N = 6$. In addition, for comparison, we also prepared a lighter Transformer model, in which two encoder/decoder blocks are used ($N = 2$), and other hyper-parameters remain the same.

Hyperparameters Unless otherwise specified, all hyperparameters are inherited from the base Transformer model. We try three different settings of the group size K: $K = 2$, $K = 4$, and $K = 6$. For English-German translation, we share the same weight matrix between the source and target embedding layers and the pre-softmax linear layer. For Chinese-English translation, we only share weights of the target embedding layer and the pre-softmax linear layer.

Search Strategies We use two search strategies: beam search and greedy search. As mentioned in Section 4.4, these two strategies lead to different parallelizability. When beam size is set to 1, greedy search is used, otherwise, beam search is used.

Knowledge Distillation Knowledge distillation (Hinton et al., 2015; Kim and Rush, 2016) describes a class of methods for training a smaller *student* network to perform better by learning from a larger *teacher* network. For NMT, Kim and Rush (2016) proposed a sequence-level knowledge distillation method. In this work, we apply this method to train the SAT using a pre-trained

[1]The corpora include LDC2002E18, LDC2003E14, LDC2004T08 and LDC2005T0.

Model	Beam Size	BLEU	Degeneration	Lattency	Speedup
Transformer	4	27.11	0%	346ms	1.00×
	1	26.01	4%	283ms	1.22×
Transformer, N=2	4	24.30	10%	163ms	2.12×
	1	23.37	14%	113ms	3.06×
NAT (Gu et al., 2017)	-	17.69	25%	39ms	15.6×
NAT (rescroing 10)	-	18.66	20%	79ms	7.68×
NAT (rescroing 100)	-	19.17	18%	257ms	2.36×
LT (Kaiser et al., 2018)	-	19.80	27%	105ms	-
LT (rescoring 10)	-	21.00	23%	-	-
LT (rescoring 100)	-	22.50	18%	-	-
IRNAT (Lee et al., 2018)	-	18.91	22%	-	1.98×
This Work					
SAT, K=2	4	26.90	1%	229ms	1.51×
	1	26.09	4%	167ms	2.07×
SAT, K=4	4	25.71	5%	149ms	2.32×
	1	24.67	9%	91ms	3.80×
SAT, K=6	4	24.83	8%	116ms	2.98×
	1	23.93	12%	62ms	5.58×

Table 3: Results on English-German translation. Latency is calculated on a single NVIDIA TITAN Xp without batching. For comparison, we also list results reported by Gu et al. (2017); Kaiser et al. (2018); Lee et al. (2018). Note that Gu et al. (2017); Lee et al. (2018) used PyTorch as their platform, but we and Kaiser et al. (2018) used TensorFlow. Even on the same platform, implementation and hardware may not exactly be the same. Therefore, it is not fair to directly compare BLEU and latency. A fairer way is to compare performance degradation and speedup, which are calculated based on their own baseline.

autoregressive Transformer network. This method consists of three steps: (1) train an autoregressive Transformer network (the *teacher*), (2) run beam search over the training set with this model and (3) train the SAT (the *student*) on this new created corpus.

Initialization Since the SAT and the Transformer have only slight differences in their architecture (see Figure 2), in order to accelerate convergence, we use a pre-trained Transformer model to initialize some parameters in the SAT. These parameters include all parameters in the encoder, source and target word embeddings, and pre-softmax weights. Other parameters are initialized randomly. In addition to accelerating convergence, we find this method also slightly improves the translation quality.

Training Same as Vaswani et al. (2017), we train the SAT by minimize cross-entropy with label smoothing. The optimizer we use is Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\varepsilon = 10^{-9}$. We change the learning rate during training using the learning rate funtion described in Vaswani et al. (2017). All models are trained for 10K steps on 8 NVIDIA TITAN Xp with each minibatch consisting of about 30k tokens. For evaluation, we average last five checkpoints saved with an interval of 1000 training steps.

Evaluation Metrics We evaluate the translation quality of the model using BLEU score (Papineni et al., 2002).

Implementation We implement the proposed SAT with *TensorFlow* (Abadi et al., 2016). The code and resources needed for reproducing the results are released at `https://github.com/chqiwang/sa-nmt`.

5.2 Results on English-German

Table 3 summaries results of English-German translation. According to the results, the translation quality of the SAT gradually decreases as K increases, which is consistent with intuition. When K = 2, the SAT decodes 1.51× faster than the Transformer and is almost lossless in translation quality (only drops 0.21 BLEU score). With K = 6, the SAT can achieve 2.98× speedup while the performance degeneration is only 8%.

When using greedy search, the acceleration becomes much more significant. When K = 6, the decoding speed of the SAT can reach about 5.58× of the Transformer while maintaining 88%

Model	b=1	b=16	b=32	b=64
Transformer	346ms	58ms	53ms	56ms
SAT, K=2	229ms	38ms	32ms	32ms
SAT, K=4	149ms	24ms	21ms	20ms
SAT, K=6	116ms	20ms	17ms	16ms

Table 4: Time needed to decode one sentence under various batch size settings. A single NVIDIA TIAN Xp is used in this test.

Model	K=1	K=2	K=4	K=6
Latency	1384ms	607ms	502ms	372ms

Table 5: Time needed to decode one sentence on CPU device. Sentences are decoded one by one without batching. K=1 denotes the Transformer.

of translation quality. Comparing with Gu et al. (2017); Kaiser et al. (2018); Lee et al. (2018), the SAT achieves a better balance between translation quality and decoding speed. Compared to the lighter Transformer ($N = 2$), with $K = 4$, the SAT achieves a higher speedup with significantly better translation quality.

In a real production environment, it is often not to decode sentences one by one, but batch by batch. To investigate whether the SAT can accelerate decoding when decoding in batches, we test the decoding latency under different batch size settings. As shown in Table 4, the SAT significantly accelerates decoding even with a large batch size.

It is also good to know if the SAT can still accelerate decoding on CPU device that does not support parallel execution as well as GPU. Results in Table 5 show that even on CPU device, the SAT can still accelerate decoding significantly.

5.3 Results on Chinese-English

Table 6 summaries results on Chinese-English translation. With $K = 2$, the SAT decodes 1.69× while maintaining 97% of the translation quality. In an extreme setting where $K = 6$ and beam size = 1, the SAT can achieve 6.41× speedup while maintaining 83% of the translation quality.

5.4 Analysis

Effects of Knowledge Distillation As shown in Figure 5, sequence-level knowledge distillation is very effective for training the SAT. For larger K, the effect is more significant. This phenomenon is echoing with observations by Gu et al. (2017); Oord et al. (2017); Lee et al. (2018). In addition,

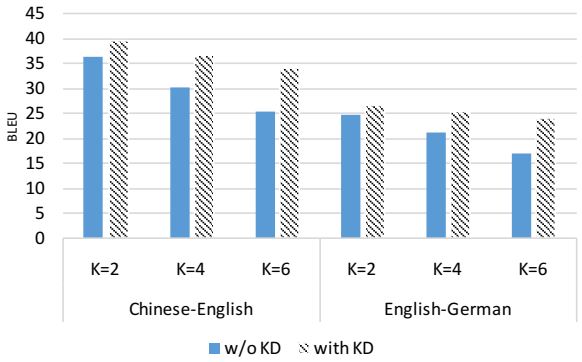

Figure 5: Performance of the SAT with and without sequence-level knowledge distillation.

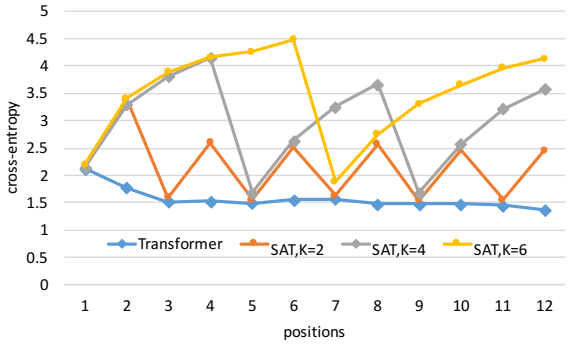

Figure 6: Position-wise cross-entropy for various models on English-German translation.

we tried word-level knowledge distillation (Kim and Rush, 2016) but only a slight improvement was observed.

Position-Wise Cross-Entropy In Figure 6, we plot position-wise cross-entropy for various models. To compare with the baseline model, the results in the figure are from models trained on the original corpora, i.e., without knowledge distillation. As shown in the figure, position-wise cross-entropy has an apparent periodicity with a period of K. For positions in the same group, the position-wise cross-entropy increase monotonously, which indicates that the long-distance dependencies are always more difficult to model than short ones. It suggests the key to further improve the SAT is to improve the ability of modeling long-distance dependencies.

Case Study Table 7 lists three sample Chinese-English translations from the development set. As shown in the table, even when producing $K = 6$ words at each time step, the model can still gen-

Model	Beam Size	BLEU				Degeneration	Lattency	Speedup
		NIST03	NIST04	NIST05	Averaged			
Transformer	4	40.74	40.54	40.48	40.59	0%	410ms	1.00×
	1	39.56	39.72	39.61	39.63	2%	302ms	1.36×
Transformer, N=2	4	37.30	38.55	36.87	37.57	7%	169ms	2.43×
	1	36.26	37.19	35.50	36.32	11%	117ms	3.50×
This Work								
SAT, K=2	4	39.13	40.04	39.55	39.57	3%	243ms	1.69×
	1	37.94	38.73	38.43	38.37	5%	176ms	2.33×
SAT, K=4	4	37.08	38.06	37.12	37.42	8%	152ms	2.70×
	1	35.77	36.43	35.04	35.75	12%	94ms	4.36×
SAT, K=6	4	34.61	36.29	35.06	35.32	13%	129ms	3.18×
	1	33.44	34.54	33.28	33.75	17%	64ms	6.41×

Table 6: Results on Chinese-English translation. Latency is calculated on NIST02.

Source	国 际 足 联 将 严 惩 足 球 场 上 的 欺 骗 行 为
Transformer	the international football federation will severely punish the fraud on the football field
SAT, k=2	fifa will severely punish the deception on the football field
SAT, k=4	fifa a will severely punish the fraud on the football court
SAT, k=6	fifa a will severely punish the fraud on the football football court
Reference	federation international football association to mete out severe punishment for fraud on the football field
Source	大 型 校 园 文 化 展 览 也 将 在 会 议 期 间 举 行 。
Transformer	the largescale exhibition of campus culture will also be held during the meeting .
SAT, k=2	the largescale cultural cultural exhibition on campus will also be held during the meeting .
SAT, k=4	the campus campus exhibition will also be held during the meeting .
SAT, k=6	a largescale campus culture exhibition will also be held on the sidelines of the meeting .
Reference	there will also be a large - scale campus culture show during the conference .
Source	这 是 小 泉 纯 一 郎 执 政 以 来 第 二 次 参 拜 靖 国 神 社 。
Transformer	this is the second time mr koizumi has visited the yasukuni shrine since he came to power .
SAT, k=2	this is the second time that mr koizumi has visited the yasukuni shrine since he took office .
SAT, k=4	this is the second time that koizumi has visited the yasukuni shrine since he came into power .
SAT, k=6	this is the second visit to the yasukuni shrine since mr koizumi came office power .
Reference	this is the second time that junichiro koizumi has paid a visit to the yasukuni shrine since he became prime minister .

Table 7: Three sample Chinese-English translations by the SAT and the Transformer. We mark repeated words or phrases by red font and underline.

erate fluent sentences. As reported by Gu et al. (2017), instances of repeated words or phrases are most prevalent in their non-autoregressive model. In the SAT, this is also the case. This suggests that we may be able to improve the translation quality of the SAT by reducing the similarity of the output distribution of adjacent positions.

6 Conclusion

In this work, we have introduced a novel model for faster sequence generation based on the Transformer (Vaswani et al., 2017), which we refer to as the semi-autoregressive Transformer (SAT). Combining the original Transformer with group-level chain rule, long-distance prediction and relaxed causal mask, the SAT can produce multiple consecutive words at each time step, thus speedup decoding significantly. We conducted experiments on English-German and Chinese-English translation. Compared with previously proposed non-autoregressive models (Gu et al., 2017; Lee et al., 2018; Kaiser et al., 2018), the SAT achieves a better balance between translation quality and decoding speed. On WMT'14 English-German translation, the SAT achieves 5.58× speedup while maintaining 88% translation quality, significantly bet-

ter than previous methods. When producing two words at each time step, the SAT is almost lossless (only 1% degeneration in BLEU score).

In the future, we plan to investigate better methods for training the SAT to further shrink the performance gap between the SAT and the Transformer. Specifically, we believe that the following two directions are worth study. First, use object function beyond maximum likelihood to improve the modeling of long-distance dependencies. Second, explore new method for knowledge distillation. We also plan to extend the SAT to allow the use of different group sizes K at different positions, instead of using a fixed value.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. We also thank Wenfu Wang, Hao Wang for helpful discussion and Linhao Dong, Jinghao Niu for their help in paper writting.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. *arXiv preprint arXiv:1603.06147*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Shonosuke Ishiwatari, Jingtao Yao, Shujie Liu, Mu Li, Ming Zhou, Naoki Yoshinaga, Masaru Kitsuregawa, and Weijia Jia. 2017. Chunk-based decoder for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1901–1912.

Łukasz Kaiser, Aurko Roy, Ashish Vaswani, Niki Pamar, Samy Bengio, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. *arXiv preprint arXiv:1803.03382*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. *international conference on learning representations*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *arXiv preprint arXiv:1802.06901*.

Yusuke Oda, Philip Arthur, Graham Neubig, Koichiro Yoshino, and Satoshi Nakamura. 2017. Neural machine translation via binary code prediction. *arXiv preprint arXiv:1704.06918*.

Aaron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C Cobo, Florian Stimberg, et al. 2017. Parallel wavenet: Fast high-fidelity speech synthesis. *arXiv preprint arXiv:1711.10433*.

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 on association for computational linguistics, pages 311–318. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

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

Hao Zhou, Zhaopeng Tu, Shujian Huang, Xiaohua Liu, Hang Li, and Jiajun Chen. 2017. Chunk-based bi-scale decoder for neural machine translation. *arXiv preprint arXiv:1705.01452*.