Unlocking the Power of GANs in Non-Autoregressive Text Generation under Weak Conditions

Anonymous authors
Paper under double-blind review

Abstract

Non-autoregressive (NAR) models once received great attention from the community, but obtain much less attention in the quest for general artificial intelligence. Our analyses reveal that the convergence problem in existing NAR models trained under Maximum Likelihood Estimation (MLE) is more severe in tasks where input does not provide the definite semantic meaning of the output. These input conditions, which we denote as weak conditions, cover most "creative" tasks, so existing NAR models struggle to obtain satisfactory performance in these tasks and are only developed in limited scenarios. This causes existing NAR models to struggle to keep pace with the rapidly evolving demands of diverse and challenging tasks. Different with MLE, which is incompatible with NAR models, Generative Adversarial Networks (GANs) are more suitable for NAR models in terms of theoretical convergence and inference manners. We thus propose an Adversarial Non-autoregressive Transformer (ANT) based on GANs for weak condition tasks. ANT supports two features: 1) Position-Aware Self-Modulation to provide more effective input signals, and 2) Dependency Feed Forward Network to strengthen its capacity in dependency modeling. The experimental results demonstrate that ANT achieves comparable performance with mainstream models in much higher efficiency and has great potential in various applications like latent interpolation and semi-supervised learning.

1 Introduction

Non-autoregressive (NAR) models, known for their lower decoding latency compared to autoregressive (AR) models, once attracted significant attention from the research community (Huang et al., 2022a). Nevertheless, these types of models have diminished in prominence during the era of Large Language Models (LLMs), even though they hold potential for addressing the challenge of high decoding latency in LLMs. The main obstacle comes from the convergence problem in existing NAR models. These models are trained on Maximum Likelihood Estimation (MLE), while MLE-based NAR models remain non-negative low bounds between the learned distributions and target distributions (Huang et al., 2022a). In other words, the learned distributions cannot be identical to the target distributions unless all the words in a sentence are independent of each other, a condition that does not reflect real-world situations. It finally leads to the multi-modality problem (Gu et al., 2018), in which these models tend to mix words from different candidates and obtain ungrammatical results.

This inherent problem directly limits the extension of NAR models to various tasks. Existing NAR models are mainly developed in several specific tasks like machine translation (Gu et al., 2018) and text summarization (Liu et al., 2022). The input in these tasks provides and constrains definite semantic meaning of the output, with one or several output candidates representing this same meaning. We refer to the condition of such tasks as a **strong condition**. An example of a classical task with strong condition, machine translation, is shown in Figure 1 (a). These tasks always have clear corresponding relations between input and output, and only require several, or even one acceptable results. Existing NAR models can incorporate element level (e.g., words) mapping relations (Gu et al., 2018) or simplify target distributions (Kim & Rush, 2016) to hide the inherent problem and obtain satisfactory performance in tasks with strong conditions (Xiao et al., 2022).

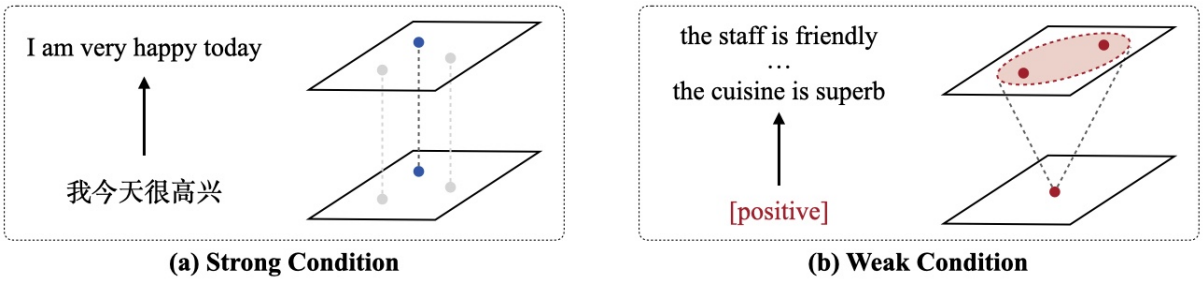

Figure 1: Comparisons between Strong Conditions and Weak Conditions. (a) Translating a sentence. (b) Generating comments based on an emotion label.

However, there is another kind of tasks, where the input does not constrain the semantic meaning of the output, which is denoted as tasks with **weak condition**. These tasks include most "creative" tasks like sentence completion (Brown et al., 2020), story generation (Fan et al., 2018; Guan et al., 2021), poetry creation (Ormazabal et al., 2022). As shown in Figure 1 (b), an input condition in these tasks describe a region in the output space, and all the samples in this region are output candidates. Different with strong condition tasks, tasks with weak conditions do not have strong corresponding relations between input and output, and they require models to learn the original target distributions instead of simplified ones. The multi-modality problem inherent in existing MLE-based NAR models can no longer be hidden. Even worse, there are always more candidates in weak condition tasks, so the problem will be more severe, and models will mix more unrelated words to obtain meaningless samples. Existing NAR models thus struggle to achieve satisfactory performance in tasks with weak conditions, and their development has stagnated in strong condition tasks. The limited application scenarios of existing NAR models have led to a decline in attention towards them in the quest for general artificial intelligence.

The incapacity of MLE-based NAR models in handling tasks with weak conditions prompt us to pay attention to another family of generative models: Generative Adversarial Networks (GANs) (Goodfellow et al., 2014). The synthetic distributions of GANs can theoretically converge to the real distributions, so their convergence can be guaranteed even with weak conditions. Besides, it can obtain high quality samples in a single forward pass, which exactly meets the needs of NAR models. Instead of adopting unstable *REINFORCE* (Williams, 1992) or biased *continuous relaxations* (Jang et al., 2017) to process the non-differentiable sampling operation in language GANs, we follow the research line of representation modeling methods (Ren & Li, 2024) and propose an **Adversarial Non-autoregressive Transformer (ANT)** for weak condition tasks.

There are two features in ANT: Position-Aware Self-modulation for obtaining clear signals for the model to obtain different words in a sentence; and Dependency Feed Forward network (Dependency FFN) for helping the model to capture more accurate word dependencies in the unstable training of GANs. The experimental results demonstrate that ANT narrows the performance gap between AR and NAR models in tasks with weak conditions, while maintaining high efficiency during inference. The contributions of this paper can be summarized as follows:

- We reveal the incapacity of exiting (MLE-based) NAR models in "creative" tasks by analyzing their limitations in weak conditions from both empirical and theoretical aspects. Furthermore, we highlight that GANs denote a more reasonable approach for constructing NAR models, as their convergence is guaranteed and their highly efficient inference capabilities are well-suited for NAR models.

- Based on GANs, we propose an Adversarial Non-autoregressive Transformer (ANT). ANT supports two features: 1) Position-Aware Self-modulation which provide more effective input signals to assist the model to obtain diverse words in a sentence; and 2) Dependency Feed Forward Network (Dependency FFN) which further improves model performance by enhancing its capacity in dependency modeling.

- We compare the performance of ANT with existing models in tasks with weak conditions. The experimental results demonstrate that ANT can get comparable performance as other models with much lower decoding latency. We further show the great potential of ANT in various applications like latent interpolation and semi-supervised learning. To the best of our knowledge, it is the first work demonstrating the effectiveness of GANs in building NAR models.

The rest of this paper is organized as follows: Section 2 introduces the background and limitations of existing NAR models in weak condition tasks. Section 3 presents the details of ANT. Section 4 describes the experimental settings and results. Finally, we draw out conclusion in Section 5.

2 Background

2.1 Limitations of MLE in Weak Condition Tasks

Most existing NAR models are based on MLE and have low decoding latency, but always sacrifice sample quality. They tend to mix words from different candidates due to the lack of word dependencies. This problem, which is knows as the multi-modality problem, will be augmented in weak condition tasks. Weak conditions describe a region in the semantic space of output which means there are more diverse candidates and the models will mix more unrelated words together in a sentence. This problem can also be illustrated from a theoretical perspective.

2.1.1 Theoretical Analysis

Given a MLE-based NAR model $P_\theta(Y|X)$, Huang et al. (2022a) reveal a non-negative lower bound between the leaned distribution and the real distribution. More specifically, $min_\theta KL(P_{data}(Y|X)||P_\theta(Y|X)) \geq C$, where $C = \sum_{i=1}^{l} H_{data}(y_i|X) - H_{data}(Y|X)$. C is also known as conditional total correlation. Based on it, we further analyze the difference of the conditional total correlation when the scenario is changed from strong conditions to weak conditions. We analyze this problem by considering the transformations from two different input $\mathbb{A}$ and $\mathbb{B}$ to output $\mathbb{T}$. Due to the complexity of real data, our analysis is based on the simplifications in **Assumption 1**.

Assumption 1. *Each target sequence $Y \in \mathbb{T}$ is consisted of l elements. There are no identical elements in the candidates. Each input $X_A \in \mathbb{A}$ has n target sequences as candidates and each input $X_B \in \mathbb{B}$ has m target candidates. Each Y can only by mapped by a specific input. All the input and target sequences follow uniform distributions.*

Based on **Assumption 1**, we obtain **Theorem 1** to describe the difference of the conditional total correlation between two different input $\mathbb{A}$ and $\mathbb{B}$.

Theorem 1. *The difference between the conditional total correlation of input $\mathbb{A}$ and $\mathbb{B}$ has the following relation: $C_A - C_B = (l-1) \cdot log(n/m)$.*

The proof can be found in Appendix A. **Theorem 1** reveals the relations between the difference of the conditional total correlation, sequence length l, and candidate numbers n and m. We further obtain a mark on **Theorem 1**:

Mark 1. *If $n > m$, then $C_A > C_B$. Higher l and n lead to higher difference between C_A and C_B.*

Comparing to the strong conditions, input in weak conditions always corresponds to more possible candidates, so its conditional total correlation will be higher. The larger gap between the learned distributions and the real distributions will lead MLE-based NAR models to generate more ungrammatical results in weak conditions.

2.1.2 Existing NAR models in Tasks With Weak Conditions

Many techniques in existing NAR models can be unified in a framework (Huang et al., 2022a): enhancing inputs (Ghazvininejad et al., 2019) or modifying targets (Gu et al., 2018; Du et al., 2021). Among them, there are two most popular techniques: 1) simplifying output with knowledge distillation (Kim & Rush, 2016); 2) enhancing input based on multiple iterations (Ghazvininejad et al., 2019). Different with tasks in strong conditions only requiring several acceptable results, weak condition tasks require models to obtain diverse results. This process describes a region in the output space. Using techniques like knowledge distillation to obtain a simplified ones is no longer acceptable. Besides, models like conditional masked language model (CMLM) (Ghazvininejad et al., 2019) and diffusion-lm (Li et al., 2022) uses multiple inference steps to improve performance, but their efficiency is sacrificed. The inapplicability of these two techniques renders the vast majority of NAR models unsuitable for tasks with weak conditions, since most of them are deeply bound to at least one of them.

In addition, researchers attempt to address the multi-modality problem by proposing alternative modeling methods that do not rely on the aforementioned techniques. DA-Transformer (Huang et al., 2022b) is a classical one. However, it is still based on MLE and its convergence has not yet been theoretically proved. When we migrate it to weak condition tasks, it obtains meaningless samples and fail to converge. It again demonstrates the high reliance of exiting NAR models to strong conditions, and the challenges of weak condition tasks to exiting MLE-based NAR models.

2.2 Language GANs

Different with the incapacity of MLE in building NAR models for weak condition tasks, GANs (Goodfellow et al., 2014) indicate a more promising training method. Their global optimal is achieved if and only if the learned distribution is exactly the same as the target one (Goodfellow et al., 2014). Unlike other popular generative models (e.g., autoregressive models (Yu et al., 2022) and diffusion models (Ho et al., 2020)), GANs can produce high quality samples in a single forward pass, which exactly meets the needs of building fully NAR models.

However, most of exiting language GANs are based on AR structures (Yu et al., 2017; Che et al., 2017; Lin et al., 2017; Fedus et al., 2018), and lost the high efficiency nature of GANs. To tackle the non-differentiable sampling operation, they adopt *REINFORCE* (Williams, 1992) or *continuous relaxations* (Jang et al., 2017). REINFORCE methods use the discriminator to calculate rewards and guide the generator to updates its parameters based on the reward. However, this methods have been proven to be unstable and suffer from high variance (Lin et al., 2020). This would exacerbate the inherent instability of the GAN training process. Continuous relaxations methods, on the other hand, use a continuous distribution to approximate the discrete one. However, it is also proven that these methods are biased estimators (Lin et al., 2020). Both of them are not suitable for building GAN-based NAR models

Our work thus follows another research line, namely representation modeling method (Ren & Li, 2024). This method firstly maps words into representations and trains a generator to model these representations. It avoids the non-differentiable sampling operation during training process, so the gradients from discriminators can pass through to generators directly. This method is free from the limitations in *REINFORCE* or *continuous relaxations* and is demonstrated to be an effective way to train language GANs (Ren & Li, 2024).

NAGAN (Huang et al., 2021), which also adopts GANs to build NAR models, is the most similar work to ours, yet our work is different in three aspects: 1) we give in-depth analyses about the limitations of MLE-based NAR models from the perspective of weak conditions; 2) the performance of their model is significantly limited by the biased *straight-through estimator* (Bengio et al., 2013), whereas our model is free from this problem; 3) our proposed facilities: Position-Aware Self-Modulation and Dependency Feed Forward Network can further boost model performance and they are not explored in previous work.

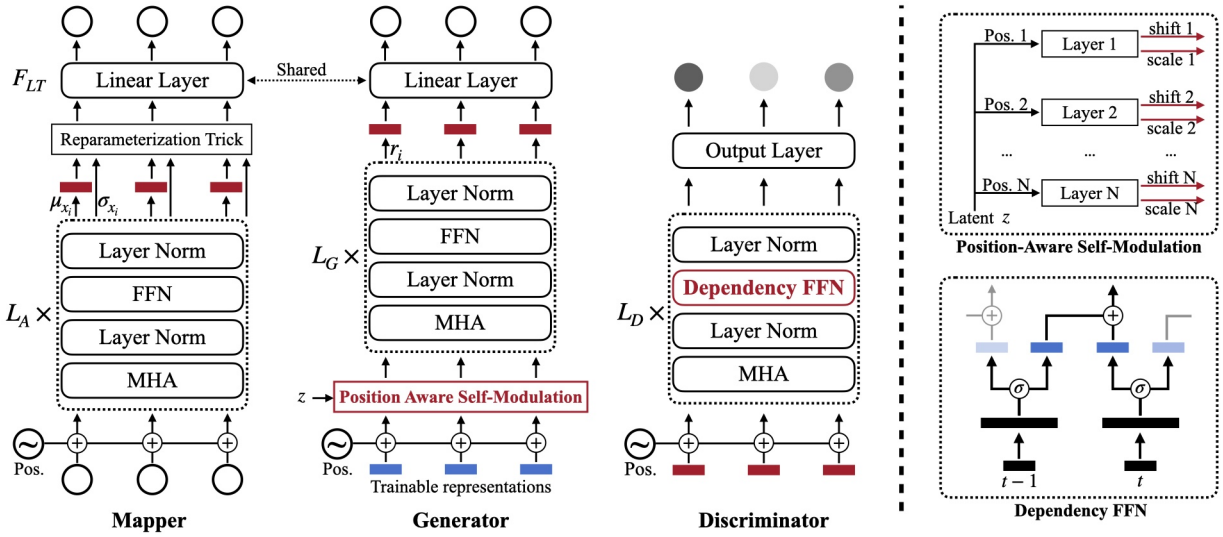

Figure 2: Structure of Adversarial Non-autoregressive Transformer (ANT)

3 Model

3.1 Model Structure

In this paper, we propose an Adversarial Non-autoregressive Transformer (ANT) which generates text in a fully NAR manner. ANT is based on the representation modeling framework (Ren & Li, 2024). As shown in Figure 2, there are three parts in ANT: Mapper, Discriminator and Generator. The mapper converts words into representations, and the generator tries to recover these representations. The discriminator needs to identify whether input representations are from the mapper or the generator. We adopt Transformer (Vaswani et al., 2017) as the backbones of all the three parts to support highly parallel computation.

$$\begin{aligned} h_0 &= h'_0 + E_{pos} \\ h'_i &= LN(MSA(h_{i-1}) + h_{i-1}) \\ h_i &= LN(FFN(h'_i) + h'_i) \end{aligned} \tag{1}$$

An input is firstly added with a positional encoding and fed into encoder layers. Each encoder layer has a multi-head attention (MHA) module and feed forward network (FFN) module. A layer normalization is added after each module.

Based on Transformer, the mapper is trained to reconstruct words based on the masked input, which is the same as the training process of BERT (Devlin et al., 2019). Following the previous work which adopts representation modeling methods to train language GANs (Ren & Li, 2024), we use the loss function of variational autoencoder (VAE) (Kingma & Welling, 2014) to train the mapper:

$$L_M = -\mathbb{E}_{z'_i \sim q(z'_i|x_i)}(logp(x_i|z'_i)) + KL(q(z'_i|x_i)||p(z'_i)) \tag{2}$$

where x_i is the i-th word in the sentence, z'_i is obtained by using reparameterization trick: $z'_i = \mu_{x_i} + \sigma_{x_i} \cdot \mathcal{N}(0,1)$, and z'_i is transformed back into words with a linear transformation layer F_{LT}. Different from cross entropy which maps words into specific points in the representation space, this method describes a region for each word, so representations slightly away from their central points μ_{x_i} can still be transformed into correct words (Ren & Li, 2024).

A non-autoregressive generator cannot input previously generated words, so trainable representations are adopted as input. The generator then gives output representations r_i in different positions and uses the same linear transformation layer F_{LT} in the mapper to transform these representations back into words.

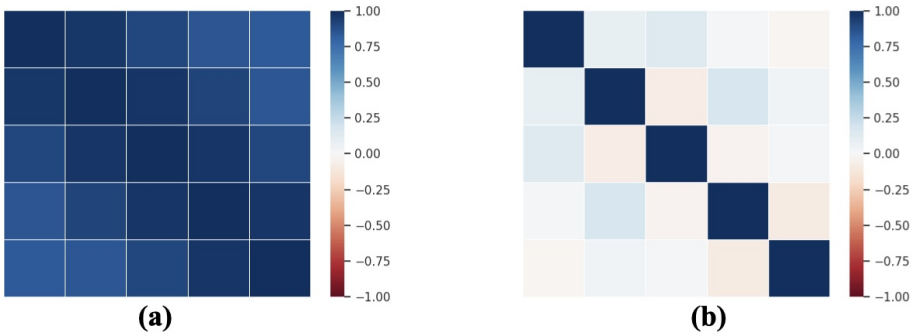

Figure 3: Cosine similarity of the output from (a) Self-Modulation; and (b) Position-Aware Self-Modulation.

The discriminator adopts the output representations from the mapper and the generator (μ_{x_i} and r_i) as input. Different from image GANs whose discriminators give a single scaler output for an image, our discriminator gives output for each representation. During training, the mapper will be trained first, and its parameters are fixed during the training of the discriminator and the generator. The representations given by the generator need not be transformed into words in training process, so the gradients from the discriminator can directly pass through to the generator.

Causal masks are adopted in both the discriminator and the generator to break the possible symmetry in the input. We use Wasserstein distance (Arjovsky et al., 2017) as the training objective and adopt Lipschitz penalty (Petzka et al., 2018) to regularize the discriminator. However, there is still a gap between our basic model and existing autoregressive models and we further propose Position-Aware Self-Modulation and Dependency Feed Forward Network (Dependency FFN) to improve model performance.

3.2 Position-Aware Self-Modulation

An effective sampling method plays a key role in the success of GANs. Transformer based image GANs (Lee et al., 2021) generate different samples by adopting self-modulation (Chen et al., 2019) to incorporate latent variables. Self-modulation assigns the same shift and scale factors to the normalized results in different positions, which leads the representations in various positions to be highly similar even with positional encodings (as shown in Figure 3 (a)). However, the output of the generator (i.e., word representations in different positions) are of high diversities. Similar input representations cannot describe the diversity among different words and thus leading to inaccurate sentence generation.

To tackle this problem, we propose **Position-Aware Self-Modulation**. As shown in the top-right corner of Figure 2, this method adopts different mapping layers for the calculations in different positions so as to gain diverse results. In practice, a parallel implementation is adopted to improve the computation efficiency, which is:

$$\begin{pmatrix} \mathbf{h}'_1 \\ \mathbf{h}'_2 \\ \vdots \\ \mathbf{h}'_N \end{pmatrix} = MLP(z) \qquad (3)$$
$$\mathbf{h}_i = \gamma(\mathbf{h}'_i) \circ LN(\mathbf{x}_i) + \beta(\mathbf{h}'_i)$$

where z is the latent variable, $\mathbf{h}'_i$ is the hidden representation in the i-th position, $MLP(\cdot)$ is a non-linear transformation whose activation function is GELU (Hendrycks & Gimpel, 2016), $LN(\cdot)$ is the layer normalization, N is the length of the sentence, and $\gamma(\cdot)$ and $\beta(\cdot)$ are linear transformations. In Position-Aware Self-Modulation, representations in different positions are calculated based on unique parameters and have clear differences (as shown in Figure 3 (b)), so as to provide more effective signals to obtain target sentences.

Table 1: FED and I. BLEU on the COCO Dataset and EMNLP Dataset.

Model	DI	COCO		EMNLP	
		FED ↓	**I. BLEU** ↑	**FED** ↓	**I. BLEU** ↑
Training Data	-	0.007	35.36	0.010	20.62
Transformer	O(N)	**0.008**	**34.28**	**0.014**	**19.50**
RelGAN	O(N)	0.062	29.53	0.136	14.74
ScratchGAN	O(N)	0.014	30.76	0.018	17.19
InitialGAN	O(N)	0.013	33.06	0.025	17.74
CMLM	O(k)	0.016	27.65	0.062	**16.67**
NAT	O(1)	0.024	26.41	0.111	11.38
NAGAN	O(1)	0.084	24.98	0.748	2.01
ANT	O(1)	**0.013**	**31.12**	**0.026**	15.51

3.3 Dependency Feed Forward Network

Transformer (Vaswani et al., 2017) builds word dependencies by dynamically assigning weights in the attention mechanism. This process, however, is unstable under the training of GANs. It will lead the models to lose word dependencies, and finally result in ungrammatical sentences. We tackle this problem by proposing Dependency Feed Forward Network (Dependency FFN) to strengthen the FFN module with the capacity of dependency modeling. The structure of Dependency FFN is shown in the bottom-right corner of Figure 2, and calculated as follows:

$$\begin{aligned} \mathbf{s}_t &= \sigma(\mathbf{x}_t W_s + b_s) \\ \mathbf{o}_t &= \mathbf{s}_{t-1} W_a + \mathbf{s}_t W_b + b_o \end{aligned} \tag{4}$$

where $\sigma(\cdot)$ is an activation function which is GELU in this work. With causal masks, $\mathbf{s}_{t-1}$ and $\mathbf{s}_t$ contain the information of first $(t-1)$ and t words, respectively. Using the sum of these two variables can help the model to explicitly build stable dependencies between the t-th word and previous $(t-1)$ words in the fragile training process of GANs.

3.4 Extension to Conditional Generation

Besides unconditional generation, conditional generation is frequently employed in a variety of tasks. We thus also extend ANT to conditional generation. Given a condition representation c, the generator can consider it by shifting the original latent variable z. We find that using trainable factors to assign weights to z and c can slightly improve the performance. Thus, we incorporate the condition as follows:

$$\hat{z} = \alpha_1 \circ z + \alpha_2 \circ c \tag{5}$$

where α_1 and α_2 are two trainable variables. Then, $\hat{z}$ is fed into Position-Aware Self-Modulation, so the generator can consider the condition representation.

For the discriminator, we use the sum of word representations x_t^d and conditional representations c as the input: $\hat{x}_t^d = x_t^d + c$. Then, $\hat{x}_t^d$ is fed into the remaining modules of the discriminator.

4 Experiment

4.1 Experiment Setup

The experiment focuses on tasks with weak conditions and covers both unconditional generation and conditional generation to evaluate model performance comprehensively. For the unconditional generation, the task is to generate sentences whose distribution can be as close as to the target sets. We follow the settings of previous work (de Masson d'Autume et al., 2019; Ren & Li, 2024) and use sentences from two datasets:

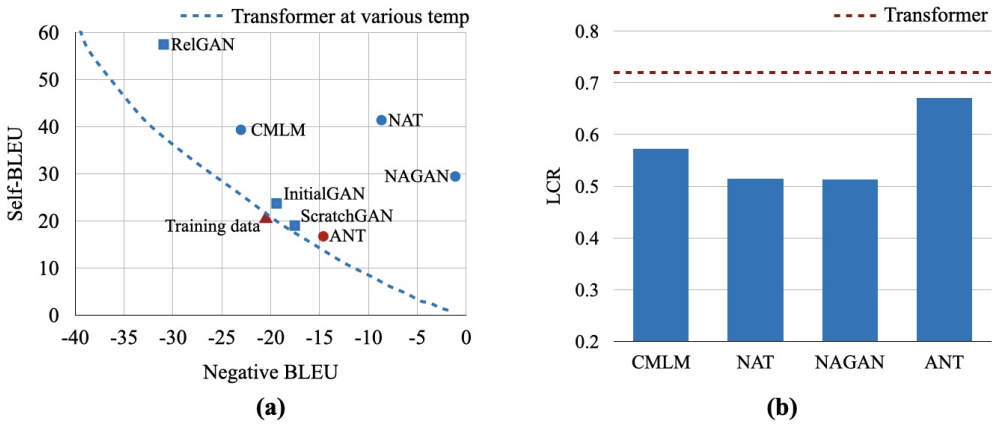

Figure 4: Additional Experimental Results: (a) Negative BLEU and Self-BLEU curve on the EMNLP dataset. Lower is better; (b) Least Coverage Rate (LCR) on the COCO dataset. Higher is better.

the COCO Image Caption Dataset (Lin et al., 2014)[1] and the EMNLP 2017 News Dataset[2]. The size of training sets of the COCO dataset and the EMNLP dataset are set to be 50,000 and 200,000, respectively. The COCO dataset can support evaluations in short sentence generation, while the EMNLP dataset focuses on long sentence generation. For the conditional generation, we randomly select 100,000 sentences from the Yelp Dataset[3] as training data and use emotion labels (positive or negative) as conditions.

4.2 Evaluation Metrics

The evaluation is conducted at both embedding level and token level. In embedding level, we use Universal Sentence Encoder[4] (Cer et al., 2018) to transform sentences into embeddings. Then, we calculate both **Fréchet Embedding Distance (FED)** (de Masson d'Autume et al., 2019) and **Least Coverage Rate (LCR)** (Ren & Li, 2024) to evaluate the overall similarity and the fine-grained similarity of two distributions, respectively.

In token level, we use **Inverse-BLEU (I. BLEU)** to evaluate model performance in terms of quality and diversity together. Besides, we also draw a curve of **BLEU** (Papineni et al., 2002) and **Self-BLEU** (Zhu et al., 2018) by tuning the temperature of the model (Caccia et al., 2020). In the case of conditional generation, **Accuracy (Acc.)** is also employed to assess whether the models produce sentences that align with the input labels.

4.3 Compared Model

An important compared model is Transformer, which adopts AR structures and is trained on MLE. It is the mainstream model in various text generation tasks. Besides, a number of AR language GANs are also compared: SeqGAN (Yu et al., 2017), RankGAN (Lin et al., 2017), MaliGAN (Che et al., 2017), LeakGAN (Guo et al., 2018), and ScratchGAN (de Masson d'Autume et al., 2019), which are based on *REINFORCE*; RelGAN (Nie et al., 2019), which uses *Gumbel-softmax* to obtain gradients; InitialGAN (Ren & Li, 2024), which does not use the above two method and adopts representation modeling. All the models mentioned-above are AR models whose Decoding Iteration (DI) is $O(N)$ (N is the sequence length).

For NAR models, we compare with another GAN-based model: NAGAN (Huang et al., 2021). Furthermore, two classical and popular MLE-based NAR structures are compared in our experiments: Non-autoregressive Transformer (NAT) (Gu et al., 2018) and conditional masked language model (CMLM) (Ghazvininejad et al.,

[1] https://cocodataset.org
[2] http://www.statmt.org/wmt17/
[3] https://www.yelp.com/dataset
[4] https://tfhub.dev/google/universal-sentence-encoder/4

Table 2: FED, I. BLEU and Acc. on the Yelp dataset

Model	DI	FED	I. BLEU	Acc.
Training Data	-	0.008	24.18	92.47%
Transformer	O(N)	0.011	23.04	91.73%
CMLM	O(k)	**0.015**	18.35	87.85%
NAT	O(1)	0.032	11.81	83.54%
ANT	O(1)	0.018	**19.08**	**88.35%**

2019). More illustrations about experiment details can be found in Appendix A. We will release our code to the public in the future.

4.4 Implementation Details

The layer numbers of the mapper, generator and discriminator are all set to be 4. Their input dimension is 256, and the hidden dimension of FFN / Dependency FFN is 1,024. The head number is set to be 8. We use AdamW (Loshchilov & Hutter, 2019) as the optimizer of the mapper and the weight decay is set to be 1e-5; its learning rate is 0.0001. During the adversarial training, AdamW (Loshchilov & Hutter, 2019) is used as the optimizer of the discriminator whose weight decay is set to be 0.0001; its learning rate is 0.0002 for the COCO and Yelp dataset, and 0.00015 for the EMNLP dataset. We choose Adam (Kingma & Ba, 2015) as the optimizer of the generator and its learning rate is 0.0001. The β_1 and β_2 in the optimizers of the discriminator and the generator are set to be 0.5 and 0.9, respectively. The maximum training epoch is set to be 4,500. We implement our model based on Tensorflow[5] (Abadi et al., 2015) and the model is trained on NVIDIA GeForce RTX 3090.

4.5 Experimental Result

4.5.1 Unconditional Generation

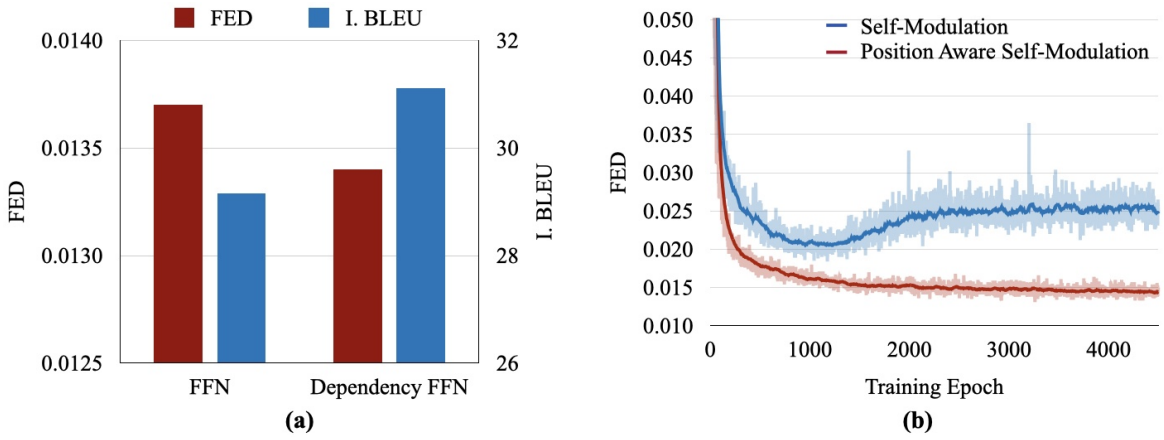

Figure 5: Ablation study of (a) Dependency Feed Forward Network (Dependency FFN); and (b) Position-Aware Self-Modulation.

The experimental results of the unconditional generation are shown in Table 1. "DI" indicates Decoding Iteration. For the COCO dataset, Transformer performs best among all compared models, and there are large gaps between MLE-based NAR models (NAT and CMLM) and AR models, which is consistent with our theoretical analyses. Weak condition input increases the difficulties of MLE training. However, ANT gets 0.013 in FED and 31.12 in Inverse BLEU, which are better than other NAR models and close to AR

[5] https://www.tensorflow.org

models like ScratchGAN and InitialGAN. It narrows the performance gap between NAR and AR models in tasks with weak conditions. Another GAN-based NAR model, NAGAN, is inferior to all the other models, which shows the limitations of the biased *straight-through estimator*.

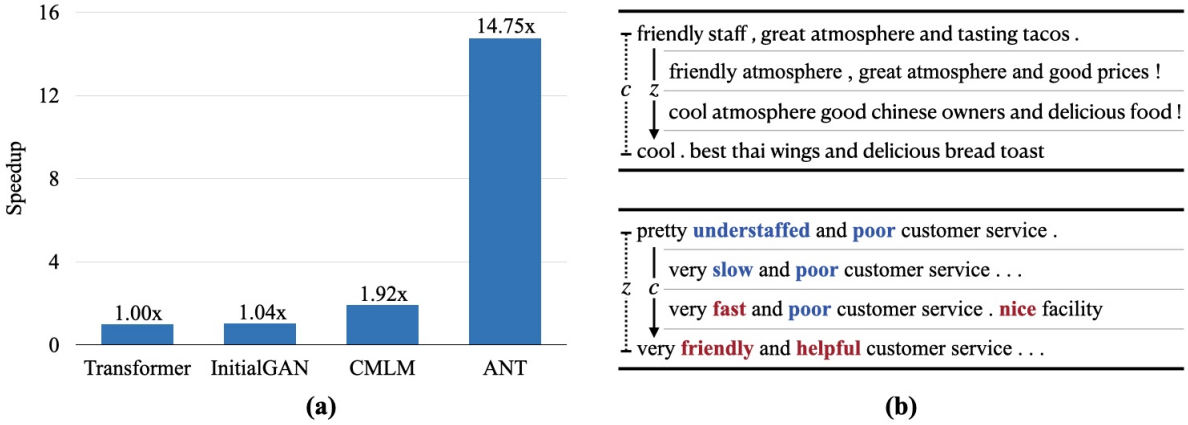

Figure 6: Speedup of Different Models.

For the EMNLP dataset, Transformer is still the best model. ANT outperforms other NAR models in FED, while CMLM can slightly outperform ANT in Inverse BLEU. The iterative decoding mechanism help CMLM to better process complicated datasets with higher decoding latency. To further discuss their performance in token level, we draw the curve of Self-BLEU and Negative BLEU by tuning the temperature in Transformer and show the results in Figure 4 (a). ANT is the only NAR model which can get comparable performance with AR models, while other NAR models (including CMLM) remain behind obviously. Specifically, NAGAN gets extremely low BLEU and Inverse BLEU, which reveals the difficulties of NAGAN to converge on complicated datasets. Furthermore, we compare Least Coverage Rate (LCR) of Transformer and other NAR models in Figure 4 (b). It shows that ANT is the only NAR model which can get close performance with Transformer.

4.5.2 Conditional Generation

The experimental results of conditional generation are shown in Table 2. Overall, Transformer gets the best performance in all the evaluation metrics with more decoding iterations. Among NAR models, ANT gets comparable performance with CMLM in FED, and achieves higher Inverse BLEU and Accuracy with lower decoding latency. NAT, which has the same decoding iterations as ANT, is inferior to other models. Fully NAR models trained with MLE rely on informative input representations, so they will meet additional difficulties when adapting to the weak condition tasks.

4.6 Discussion

In ANT, we adopt Dependency FFN to strengthen the dependency modeling capacity in the discriminator. We compare the performance between Dependency FFN and the original FFN on the COCO dataset and show the results in Figure 5 (a). ANT with Dependency FFN obtains lower FED and higher Inverse BLEU. It shows that Dependency FFN can help improve model performance by modeling more accurate word dependencies. Besides, we explore the effectiveness of Position-Aware Self-Modulation by comparing the training curves of FED on the COCO dataset. As shown in Figure 5 (b), ANT with Position-Aware Self-Modulation converges faster, and finally achieves better performance.

One advantage of ANT is that it only requires one decoding step and has high speedup. We compare the speedup of different models in Figure 6 (a). ANT is 14.75 times faster than Transformer. Even comparing with CMLM, it also has much lower decoding latency while obtaining comparable or even better performance. Besides, ANT has great potential in various applications. We explore the potential of ANT in different applications including semi-supervised learning and latent interpolation in the following.

Table 3: Effectiveness of ANT in Semi-supervised Learning (Num.: number of labeled data).

Method	Num.	P	R	F1
SL	500	91.28%	89.06%	90.15%
SSL		90.77%	92.15%	**91.46%**
SL	1000	92.42%	91.33%	91.87%
SSL		94.87%	92.39%	**93.62%**

Generative models can also boost the performance of classification models based on semi-supervised learning (SSL). We investigate the application of ANT in SSL by incorporating it into the training of a classification model. The classification model is trained to identify emotion labels of the sentences in the Yelp dataset. We prepare two training sets. One is composed of 500 labeled data and the other one consists of 1,000 labeled data. The results are shown in Table 3. The classification models trained in SSL consistently outperform the ones trained in supervised learning (SL). ANT can help the classification model capture more accurate data distribution so as to achieve better performance.

The potential of ANT can be further explored in controllable text generation. There are two latent variables in ANT: z, which is sampled from a pre-defined distribution; and c, which is a condition representation. We fix one of them and gradually change the other one. The first case in Figure 6 (b) shows the samples given by tuning z, in which ANT transforms one sentence into another one, with the middle sentences kept understandable. The second case in Figure 6 (b) shows the samples given by changing c from the negative representation to the positive representation. ANT gradually transforms negative words into positive ones while keeping the main structure of the sentence. Such controllable generation is seldomly explored by NAR models, and it may inspire further ideas for related tasks.

5 Conclusion

In this work, we reveal the limited application scenarios of existing MLE-based NAR models due to their incapacity in tasks with weak conditions. We illustrate the reasons from both empirical and theoretical aspects. The inherent convergence problem in MLE-based NAR models make it incapable in weak condition tasks. Instead, GANs denote a more promising method. We thus propose an Adversarial Non-autoregressive Transformer (ANT) based on GANs. ANT supports two novel features: Position-Aware Self-Modulation and Dependency FFN. With the help of these two facilities, ANT narrows the performance gaps between NAR models and AR models in weak condition tasks with high efficiency. We also demonstrate the potential of ANT in both semi-supervised learning and controllable text generation.

Although ANT can significantly reduce the decoding latency, it still does not outperform Transformer. In the future, we will explore more techniques to further improve its performance. In addition, we will also scale up ANT and apply it to more complicated scenarios.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pp. 214–223, 2017.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. Language gans falling short. In *ICLR*, 2020.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal sentence encoder for english. In *EMNLP*, 2018. doi: 10.18653/v1/d18-2029.

Tong Che, Yanran Li, Ruixiang Zhang, R. Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. Maximum-likelihood augmented discrete generative adversarial networks. *CoRR*, abs/1702.07983, 2017.

Ting Chen, Mario Lucic, Neil Houlsby, and Sylvain Gelly. On self modulation for generative adversarial networks. In *ICLR*, 2019.

Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack W. Rae. Training language gans from scratch. In *NeurIPS*, pp. 4302–4313, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pp. 4171–4186, 2019. doi: 10.18653/v1/n19-1423.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. Order-agnostic cross entropy for non-autoregressive machine translation. In *ICML*, pp. 2849–2859, 2021.

Angela Fan, Mike Lewis, and Yann N. Dauphin. Hierarchical neural story generation. In *ACL*, pp. 889–898, 2018. doi: 10.18653/v1/P18-1082.

William Fedus, Ian J. Goodfellow, and Andrew M. Dai. Maskgan: Better text generation via filling in the _____. In *ICLR*, 2018.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP-IJCNLP*, pp. 6111–6120, 2019.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014.

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

Jian Guan, Zhexin Zhang, Zhuoer Feng, Zitao Liu, Wenbiao Ding, Xiaoxi Mao, Changjie Fan, and Minlie Huang. Openmeva: A benchmark for evaluating open-ended story generation metrics. In *ACL/IJCNLP*, pp. 6394–6407, 2021.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *AAAI*, pp. 5141–5148, 2018.

Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, pp. 6626–6637, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Fei Huang, Jian Guan, Pei Ke, Qihan Guo, Xiaoyan Zhu, and Minlie Huang. A text {gan} for language generation with non-autoregressive generator, 2021.

Fei Huang, Tianhua Tao, Hao Zhou, Lei Li, and Minlie Huang. On the learning of non-autoregressive transformers. In *ICML*, pp. 9356–9376, 2022a.

Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. Directed acyclic transformer for non-autoregressive machine translation. In *ICML*, pp. 9410–9428, 2022b.

Xiao Shi Huang, Felipe Perez, and Maksims Volkovs. Improving non-autoregressive translation models without distillation. In *ICLR*, 2022c.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *EMNLP*, pp. 1317–1327, 2016. doi: 10.18653/v1/d16-1139.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. *CoRR*, abs/2107.04589, 2021.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation. In *NeurIPS*, 2022.

Chun-Hsing Lin, Siang-Ruei Wu, Hung-yi Lee, and Yun-Nung Chen. Taylorgan: Neighbor-augmented policy update for sample-efficient natural language generation. *CoRR*, abs/2011.13527, 2020.

Kevin Lin, Dianqi Li, Xiaodong He, Ming-Ting Sun, and Zhengyou Zhang. Adversarial ranking for language generation. In *NeurIPS*, pp. 3155–3165, 2017.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, pp. 740–755, 2014.

Puyuan Liu, Chenyang Huang, and Lili Mou. Learning non-autoregressive models from search for unsupervised sentence summarization. In *ACL*, pp. 7916–7929, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Weili Nie, Nina Narodytska, and Ankit Patel. Relgan: Relational generative adversarial networks for text generation. In *ICLR*, 2019.

Aitor Ormazabal, Mikel Artetxe, Manex Agirrezabal, Aitor Soroa, and Eneko Agirre. Poelm: A meter- and rhyme-controllable language model for unsupervised poetry generation. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP*, pp. 3655–3670. Association for Computational Linguistics, 2022.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pp. 311–318, 2002.

Henning Petzka, Asja Fischer, and Denis Lukovnikov. On the regularization of wasserstein gans. In *ICLR*, 2018.

Da Ren and Qing Li. Initialgan: A language GAN with completely random initialization. *IEEE Trans. Neural Networks Learn. Syst.*, 35(12):18431–18444, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pp. 5998–6008, 2017.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992.

Yisheng Xiao, Lijun Wu, Junliang Guo, Juntao Li, Min Zhang, Tao Qin, and Tie-Yan Liu. A survey on non-autoregressive generation for neural machine translation and beyond. *CoRR*, abs/2204.09269, 2022.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *Trans. Mach. Learn. Res.*, 2022.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pp. 2852–2858, 2017.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *SIGIR*, pp. 1097–1100, 2018.

A Proof

A.1 Proof of Theorem 1.

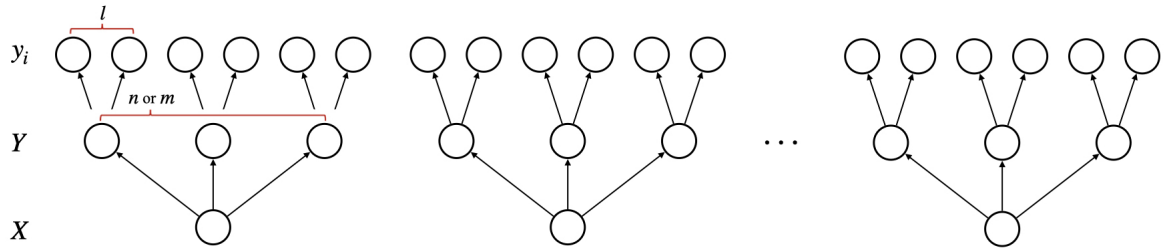

Figure 7: Mapping Relations Described in Assumption 1.

Assumption 1 describes a relation which is shown in Figure 7. Based on it, we can obtain the following conditional probabilities: $P_{data}(Y|X_A) = P_{data}(y_i|X_A) = 1/n$ and $P_{data}(Y|X_B) = P_{data}(y_i|X_B) = 1/m$. It should be noted that $P_{data}(y_i|X_A)$ and $P_{data}(y_i|X_B)$ are not related to the sequence length l in this scenario. Given a specific input, there are only n or m possible candidate elements at each position. Based on the conditional probabilities, we have:

$$\begin{aligned} C_A &= \sum_{i=1}^{l} H_{data}(y_i|X_A) - H_{data}(Y|X_A) \\ &= -l \cdot log(1/n) + log(1/n) \\ &= -(l-1) \cdot log(1/n) \end{aligned} \tag{6}$$

$$\begin{aligned} C_B &= \sum_{i=1}^{l} H_{data}(y_i|X_B) - H_{data}(Y|X_B) \\ &= -l \cdot log(1/m) + log(1/m) \\ &= -(l-1) \cdot log(1/m) \end{aligned} \tag{7}$$

Thus, we can calculate their difference as follows:

$$\begin{aligned} C_A - C_B &= (l-1) \cdot [-log(1/n) + log(1/m)] \\ &= (l-1) \cdot log(n/m) \end{aligned} \tag{8}$$

When $n > m$, $C_A - C_B$ will be positive (l is always higher than 1 in real tasks). It indicates the increase of the conditional total correlation, while higher values indicate higher difficulties in the optimization of MLE-based NAR models. In addition, our experimental results also illustrate the limitations of existing NAR models in tasks with weak conditions, which is consistent with our theoretical analyses.

B Experiment Details

B.1 Implementation of NAT and CMLM

Both NAT (Gu et al., 2018) and CMLM (Ghazvininejad et al., 2019) are designed for machine translation, so their original structures contains an encoder to encode input in source language. However, there may be no meaningful input in our experiment (e.g., unconditional generation), so this structure cannot be used in the experiment directly. We thus use the idea of VAE to obtain hidden representations, so they can be transferred to the tasks in our experiments.

More specifically, a Transformer-based encoder is adopted to encode the sentences into hidden representations during training. Then, these representations are fed into the decoder to reconstruct the input sentences. They use the training objective of VAE, so the representations can be close to the standard normal distribution. During inference, representations sampled from the standard normal distribution will be fed into the decoder, and the decoder will generate sentences based on the sampled representations. For NAT, the representations are fed into decoder as input. For CMLM, the representations are concatenated with the embeddings of input tokens (masked or unmasked words). The iteration number of CMLM is set to be 10 as in previous work (Ghazvininejad et al., 2019; Huang et al., 2022c).

B.2 Evaluation Metrics

Fréchet Embedding Distance (FED) (de Masson d'Autume et al., 2019) is same with the Fréchet Inception Distance (FID) (Heusel et al., 2017) except for the encoding model. The encoding model is changed to be fit into text generation. We adopts Universal Sentence Encoder following the settings of the previous work (de Masson d'Autume et al., 2019). It is calculated as follows:

$$FED = ||\mu_1 - \mu_2||_2^2 + Tr(c_1 + c_2 - 2(c_1 c_2)^{1/2}) \tag{9}$$

where μ_1 and μ_2 are the mean, and c_1 and c_2 are the covariance.

Least Coverage Rate (LCR) (Ren & Li, 2024) is proposed to be a compliment when the FED of two models are too close to each other, since LCR is more sensitive to the change of data quality (Ren & Li, 2024). Given two sets of sentence $X_i^a \in \mathbb{X}_a$ and $X_i^b \in \mathbb{X}_b$, LCR is calculated as follows:

$$\begin{aligned} S_{ij} &= Sim(\mathbf{Emb}(X_i^a), \mathbf{Emb}(X_j^b)) \\ R_a &= \frac{1}{|\mathbb{X}_a|} \sum_{i=1}^{|\mathbb{X}_a|} \delta(\sum_{j=1}^{|\mathbb{X}_b|} S_{ij} \geq \tau) \\ R_b &= \frac{1}{|\mathbb{X}_b|} \sum_{j=1}^{|\mathbb{X}_b|} \delta(\sum_{i=1}^{|\mathbb{X}_a|} S_{ij} \geq \tau) \\ LCR(\mathbb{X}_a, \mathbb{X}_b) &= min(R_a, R_b) \end{aligned} \tag{10}$$

where X_i^a and X_i^a are the i-th and j-th sentences from sentence sets $\mathbb{X}_a$ and $\mathbb{X}_b$), respectively. $\mathbf{Emb}(\cdot)$ is the model used to transform sentences into embeddings (which is Universal Sentence Encoder in this work), τ

is a hyperparameter, $Sim(\cdot)$ is cosine similarity and $\delta(\cdot)$ is a function which returns 1 if input is higher than 0, and 0 for others.

The idea of LCR is to identify whether a specific mode in one set is covered by the sentences in another set or not. Then, it uses the minimum coverage rates as the output, so LCR can be sensitive to two common problems in text generative models: 1) models tend to generate sentences which are out of the real distributions; and 2) the generated sentences are in high similarities.

All the token level metrics (i.e., **BLEU**, **Self-BLEU**, and **Inverse BLEU**) are calculated up to 5 grams following previous work (de Masson d'Autume et al., 2019; Ren & Li, 2024).