REPRESENTATION MODELING BASED LANGUAGE GANS: FROM AUTOREGRESSIVE MODELS TO NON-AUTOREGRESSIVE MODELS

DA REN

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University
Department of Computing

Representation modeling based language GANs: from autoregressive models to non-autoregressive models

Da Ren

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
April 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Da Ren

Abstract

Training autoregressive models based on maximum likelihood estimation (MLE) has become a mainstream method in text generation. However, this method has two inherent limitations. First, the discrepancy between training and inference causes the exposure bias problem. Secondly, these models are based on autoregressive structures which have high latency during inference. They are thus inappropriate in scenarios requiring low latency. Instead, Generative Adversarial Networks (GANs) are free from the exposure bias problem and have the potential to construct non-autoregressive (NAR) models. However, GANs have their own limitations in text generation.

First, how to make use of the signals from discriminators to update generators. In text generation, tokens are always sampled from probability distributions while the sampling operation prevents gradients from being passed to generators. Existing methods, which model output probabilities, are either high variance or biased estimators. Instead, we first transform words into representations, and then train the generator to recover these representations. We denote these methods as representation modeling methods. We adopt dropout sampling and fully normalized LSTM to provide a more effective sampling method and keep healthier gradients. Our proposed model outperforms MLE-based models and existing GAN-based models in various evaluations metrics.

Nevertheless, most of existing language GANs are based on autoregressive structures which have high latency. We thus build GAN-based NAR models to obtain the

results more efficiently. We divide text generation tasks from two different categories: incomplete information scenarios and complete information scenarios.

For the incomplete information scenarios, whose target contains more information than the input, the multi-modality problem in MLE-based NAR models will be further augmented. In this scenario, each input has lots of diverse candidates which will be more easily to be mixed. Language GANs tend to generate ungrammatical sentences after adopting NAR structures. The input representations obtained by existing methods are similar between different positions. Besides, Transformer builds word dependencies only based on the attention mechanism, while this process becomes unstable during the training of GANs. We tackle these problems by proposing two facilities: 1) Position-Aware Self-Modulation to provide more effective input signals, and 2) Dependency Feed Forward Network to strengthen the feed forward network layer with the capacity of dependency modeling. The experimental results demonstrate that our proposed model can obtain comparable performance as existing mainstream models with much fewer decoding iterations.

For the complete information scenarios, whose input has complete information of the output, the complicated mapping relations will cause greater errors in the learned marginal distributions of MLE-based NAR models and thus exacerbate their multi-modality problem. Even our previously proposed GAN-based NAR model also fails to obtain satisfied performance due to the incapacity of modeling the complicated relations. To tackle this problem, we first revise the discriminator structure to make use of unpaired samples. Then, we integrate the reconstruction procedure to better utilize paired samples. We test the performance of our proposed model in image captioning, and our model achieves a new state-of-the-art for fully NAR models on the MSCOCO dataset with much higher speedup and lower parameter number.

Publications Arising from the Thesis

1. Da Ren, and Qing Li, "InitialGAN: A Language GAN With Completely Random Initialization", in *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* (2023).

2. Da Ren, Yi Cai, and Qing Li, "Unlocking the Power of GANs in Non-Autoregressive Text Generation", manuscript submitted to *The Thirteenth International Conference on Learning Representations (ICLR 2025), 2025.*

3. Da Ren, and Qing Li, "Releasing the Capacity of GANs in Non-Autoregressive Image Captioning", in *The 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 2024.

Acknowledgments

I would like to acknowledge a number of people who provides various help for my study.

First of all, I would like to express gratitude towards Prof. Qing Li. He provides fully support to my research. He builds a free and open academic research environment, so we can study our interested research topic. Besides, he also provides us sufficient hardware resources. During my study, he gives me detailed editing suggestions, so I can learn how to revise the paper through this process. When my papers are rejected by conferences or journals, he gives me confident to continuously revise the paper and submit it to top-tier conferences or journals. This process is important for me to learn how to treat the comments and decisions from reviewers correctly.

Then, I am also grateful to Prof. Yi Cai. He provides considerate help in my study and life. When I meet difficulties in my study, his valuable suggestions and guidance are important for me to continue revising and improving my paper. Furthermore, he also helps me to find my shortcomings and helps me overcome them.

Besides, I would like to acknowledge Mr. Hongzhi Zhao. His helpful discussion is significant for me to complete the theoretical proof. Furthermore, he also provides valuable guidance in my math study.

Last but not least, I would like to express gratitude towards all our group members and the supporting staffs of our department. They provide various helps and supports

for my study.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Deep Generative Models

Generative models, which can learn data distributions, play a crucial role in machine learning. Building high quality generative models is a long-standing goal and their wide range of applications in real scenarios have gathered increasing interest from researchers. More specifically, their applications can be summarized as follows [33].

First, adopting generative models for training and sampling provides an effective method to evaluate the capacity of representing and manipulating high-dimensional probability distributions. This capacity is important in various fields like mathematics and engineering.

Secondly, generative models can be incorporated into reinforcement learning. For example, it can be adopted to predict possible futures of the environment so to assist the planning in reinforcement learning. Besides, generative models can also be used to simulate environment, so we can avoid physical damage from agents' possible mistakes.

Besides, generative models can be trained to predict missing data. The complicated

real scenarios may bring various data problems, while one of the most popular problems is data loss. Generative models, which can obtain new data from the original distributions, can assist to tackle this problem. An important application is semi-supervised learning in which generative models can either provide more synthetic training data or be incorporated into training processes directly.

In addition, generative models can be used to learn multi-modal outputs. In many tasks, each input corresponds to many possible outputs, while traditional methods may not be able to learn all these candidates. Generative models can be trained with these one-to-many data, and provide methods to obtain these various outputs.

Lastly, many tasks require model to obtain realistic samples from certain distributions. In computer vision, single image super-resolution and image-to-image translation have a wide range of applications. In natural language processing, machine translation is a classical and important task. Besides, there are also a number of cross modal tasks like text-to-image generation, and image captioning.

Recent generative models like ChatGPT[1] and Sora[2] have brought great impact to the society. It demonstrates the significance of studying and developing generative models.

The rapid development of deep neural networks over the past few decades has assisted the emergence of deep generative models. Models like Variational Autoencoder (VAE) [67], Generative Adversarial Networks (GANs) [34] and diffusion models [45] have been widely adopted in various tasks. Although they have various training methods, they all try to obtain a deep learning model, with parameter θ, which can learn the data distributions p_{data}:

$$p_\theta(x) = p_{data}(x) \tag{1.1}$$

After obtaining p_θ, a generative model can obtain new data by sampling from the

[1]https://chat.openai.com

[2]https://openai.com/research/video-generation-models-as-world-simulators

learned distribution:

$$x \sim p_\theta(x) \tag{1.2}$$

In practice, we usually need to obtain new data based on conditions, in which a model learns conditional distribution as follows:

$$p_\theta(x|c) = p_{data}(x|c) \tag{1.3}$$

Similarly, we can sample new data based on the learned conditional distribution:

$$x \sim p_\theta(x|c) \tag{1.4}$$

The flexibility of deep neural networks enables deep generative models to be trained on various structures, so researchers can choose appropriate ones for different tasks.

1.2 Text Generative Models

Although there are various generative models, adopting autoregressive structures with Maximum Likelihood Examination (MLE) is the most popular one in text generation. Given a sequence $(x_1, x_2, ..., x_T)$, autoregressive models learn its probability by calculating their product:

$$p_\theta(x_1, x_2, ..., x_T) = \prod_{t=1}^{T} p_\theta(x_t|x_0, x_1, ..., x_{t-1}) \tag{1.5}$$

This model can be incorporated with various neural networks. The most popular one in the early stage is Sequence-to-Sequence (Seq2Seq) model [119]. It uses Long Short-Term Memory (LSTM) [46] to construct an encoder to encode input into a hidden vector, and then uses another LSTM, which is denoted as decoder, to decode the target sequence. This structure allows models to process the data whose input

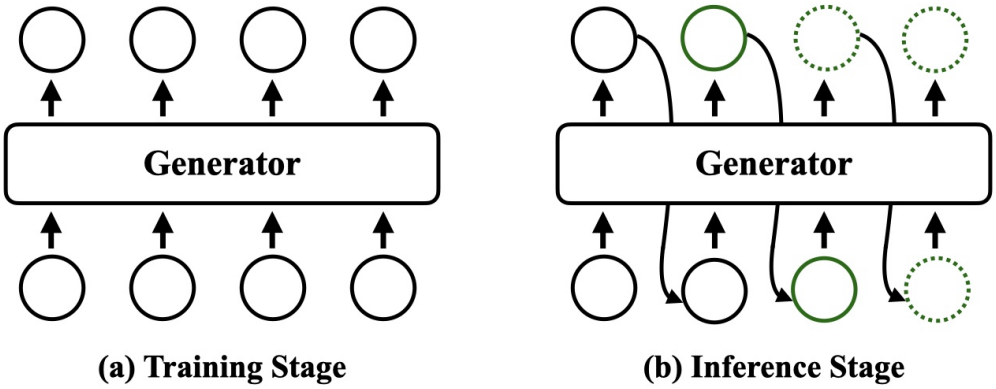

Figure 1.1: Training and Inference Stage of Autoregressive Models.

and output are in arbitrary length. Seq2Seq quickly becomes a popular model in text generation tasks like machine translation.

However, LSTM can not fully make use of GPU hardware by processing input in parallel, so Gehring et al. [30] propose a Convolutional Sequence to Sequence model (ConvS2S) by replacing LSTM with Convolutional Neural Networks (CNNs) [55, 68, 72]. In addition, Vaswani et al. [120] propose Transformer only based on the attention mechanism, which is originally proposed to enhance Seq2Seq model [7]. The attention mechanism supports highly parallel computation and enables models to consider tokens regardless of distance. Its remarkable performance makes it quickly become mainstream structures in text generation. Nevertheless, training models based on autoregressive structures with MLE has its own limitations.

First, they use ground truth as input during training stage while read previously generated tokens during inference. These two stages are shown in Figure 1.1. During inference, if the model generates a wrong token (the green circle in Figure 1.1 (b)), this token will be the fed into the model and the model will be in the state space it has never met during training. The quality of generated sentences will thus decrease sharply. This problem is known as the exposure bias problem [9].

In addition, these models generate tokens one-by-one, so they have high decoding

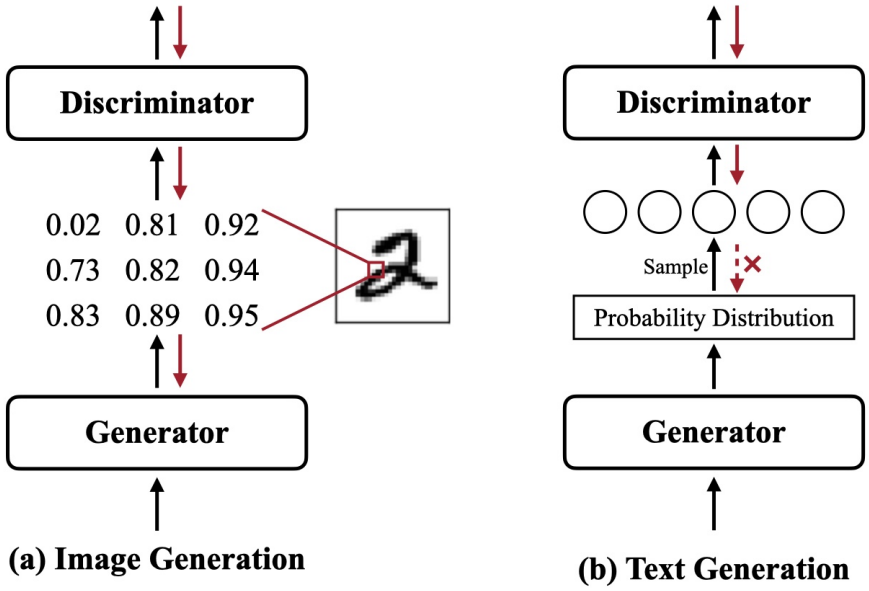

Figure 1.2: Differences of GANs in Image Generation and Text Generation.

latency during inference, and are not suitable for scenarios requiring low latency.

1.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [34] have the potential to tackle the two problems above. There are two models in GANs: Discriminator and Generator. The discriminator D is trained to identify whether the input is synthetic or not, while the generator G tries to generate realistic samples. More specifically, these two models play a minimax game with a value function $V(D,G)$[3]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}}[logD(x)] + \mathbb{E}_{z\sim p_z}[log(1-D(G(z)))] \tag{1.6}$$

Goodfellow et al. [34] have proved that GANs can achieve the global optimality if and only if the learned distribution is exactly same with the data distribution. During

[3]There are various training objectives for GANs. We use the initial one in the original GAN paper [34] to illustrate its main idea.

training, GANs do not rely on ground truth as input, so they provide a consistent manner for both training and inference procedures and are free from the exposure bias problem. Furthermore, their convergence does not rely on model structures, so they provide a method to build non-autoregressive (NAR) models which can have lower latency by obtaining results in parallel.

However, GANs have their own limitations when adapting to text generation. There are fundamental differences when applying GANs in image generation and text generation.

In image generation, the generator obtains real value data and they are fed into the discriminator directly. The gradients from the discriminator can pass through to the generator directly (as shown Figure 1.2 (a)). Thus, the parameters in the generator can be updated based on the gradients. In text generation, however, the generator usually obtains probability distributions first, specific tokens are then sampled from the distributions. This sampling operation is non-differentiable. It stops the gradients from being passed to the generator. This process is shown in Figure 1.2 (b).

Most of existing language GANs adopt *REINFORCE* or *continuous relaxations* to tackle this problem. *REINFORCE* adopts the output of the discriminator as reward and update the generator based on it. *Continuous relaxations* (e.g., Gumbel-softmax [54]) obtains continuous distributions which can be used as relaxations of discrete distributions. However, *REINFORCE* is high variance, and *continuous relaxations* like Gumbel-softmax are biased [18]. Thus, these models highly rely on pre-training techniques and their performance is also limited by their inherent problems.

1.4 Contribution

In this thesis, we first explore how to adopt GANs in text generation. Instead of adopting *REINFORCE* or *continuous relaxations*, we transform words into representations and train the generator to obtain these representations. The representations are then directly fed into the discriminator, so as to avoid the non-differentiable sampling operation. We denote this method as **Representation Modeling Method**. Although this method can allow the gradients from the discriminator to pass through to the generator, their performance is still limited by two problems: 1) invalid sampling methods; and 2) unhealthy gradients. We tackle these two problems by presenting two techniques: dropout sampling and fully normalized LSTM. Armed with these two techniques, our model outperforms MLE-based models and existing GAN-based models in both existing evaluations metrics and our newly proposed metrics without any pre-training techniques. These experimental results demonstrates the effectiveness of building language GANs based on representation modeling methods.

However, most of existing language GANs adopt autoregressive structures, and they do not support parallel computation in both the training and inference stage. It leads to high latency and brings difficulties to adopt it on larger and more complicated datasets. Thus, we further study how to extend representation modeling methods to NAR models, which significantly decreases latency by obtaining all the results in parallel. To conduct complete exploration about GAN-based NAR models, we divide existing text generation tasks into two categories: 1) Incomplete information scenario (IIS), where the target output has more information than the input; and 2) Complete information scenario (CIS), where the input maintains complete information of the target output;. We study GAN-based NAR models in both these two scenarios.

For the IIS, the inherent multi-modality problem in existing MLE-based NAR models will be further augmented because of the increase of candidate numbers and diversity. In this scenario, each input needs to complement additional information to obtain

output, and different information will lead to completely different results. It thus has much more candidates which will be more easily to be mixed. Although the global optimality of GANs can be guaranteed regardless of model structures and representation modeling methods are also demonstrated to be effective methods in our previous work, they tend to generate ungrammatical sentences after adopting NAR structures. Our analyses reveal that the input representations obtained by existing methods are similar between different positions, and it is difficult to generate the diverse target output based on this similar input. Besides, Transformer, the widely used backbone in NAR models, builds word dependencies only based on the attention mechanism, while this dynamic process becomes unstable during the fragile training of GANs. We tackle these problems by proposing two facilities: 1) Position-Aware Self-Modulation to provide more effective input signals, and 2) Dependency Feed Forward Network to strengthen the feed forward network layer with the capacity of dependency modeling. We test the performance of the proposed model in incomplete information scenarios. The experimental results demonstrate that our proposed model can obtain comparable performance as existing mainstream models with much fewer decoding iterations.

For the CIS, we investigate how to apply the model in complete information scenarios with complicated mapping relations. Complete information scenarios, whose input has complete information of the output, often have less candidates than incomplete information scenarios, but they often have complicated mapping relations between input and output. We study these scenarios based on a classical but challenging task: image captioning. We find that the complicated relations will cause greater errors in the learned marginal distributions of MLE-based NAR models and thus exacerbate their multi-modality problem. Even our previously proposed GAN-based NAR model, which is free from the multi-modality problem, also fails to generate high quality samples that are consistent with input condition. The main difficulties come from the incapacity of modeling the complicated mapping relations. To tackle this

problem, we first revise the discriminator structure to be compatible with contrastive learning. It can help the model to effectively make use of unpaired samples. Then, we integrate the reconstruction procedure into the training process to better make use of paired samples. By further adopting other effective techniques and our proposed lightweight structure, our model achieves a new state-of-the-art for fully NAR models on the challenging MSCOCO dataset with much higher speedup and lower parameter number.

1.5 Thesis Overview

In section 2, we first give a comprehensive introduction about language GANs and existing MLE-based NAR models. Then, in section 3, we introduce representation modeling methods which allow the generator to update the parameters based on the gradients from the discriminator directly. After that, we illustrate how to build GAN-based NAR models in incomplete information scenarios and complete information scenarios in section 4 and section 5, respectively. Finally, we draw our conclusion and discuss future directions in section 6.

Chapter 2

Background

2.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [34] are popular in image generation. There are two models in GANs: Discriminator and Generator. The Discriminator is trained to identify generated samples from true samples, while generators try to generate indistinguishable samples. The high quality samples obtained by GANs prompt researchers to adopt them in various tasks [10, 58, 140]. However, the training of GANs is finding a saddle point in the optimization space. It leads the training process to be extremely difficult. Researchers try to tackle this problem from different perspectives [99, 118, 127].

Model structure is an important perspective to improve the performance of GANs. Radford et al. [103] propose to use CNNs and construct their models. They denote these models as deep convolutional generative adversarial networks (DCGANs) [103], which becomes a classical structure in GANs. Besides, ResNet [41] is also another widely-used choice for researchers [93]. Recently, the success of vision transformer [21] inspires researchers to apply Transformer [120] in GANs [56, 73].

The training objective is another key in stabilizing and improving the performance of GANs [97, 126]. Mao et al. [91] consider that the original loss function will have the problem of gradient vanishment when fake samples are too far away from decision boundaries. Thus, they propose to use least squares loss to move fake samples towards the decision boundaries [91]. Hinge loss is another popular loss function [79]. Besides, Arjovsky et al. [5] find JS divergence is not sensible loss functions when learning distributions supported by low dimensional manifolds. Thus, they propose to use Earth Mover (EM) distance which is also known as Wasserstein distance as training objective. Considering the original form of Earth Mover (EM) distance is highly intractable, Arjovsky et al. [5] use Kantorovich-Rubinstein duality [122] as the objective.

The duality form requires the model following 1-Lipschitz constraint. Arjovsky et al. [5] keep this constraint by clipping the weights in a fixed region. Gulrajani et al. [38] consider that weight clipping may lead discriminators to become a too simple function. Thus, they propose to use gradient penalty to control the overall gradients of discriminators to be close to 1. This idea also inspires a number of penalty methods [101, 141]. Another method to keep 1-Lipschitz constraint is spectral normalization [93]. Additionally, researchers find that keeping 1-Lipschitz constraint can also improve the performance GANs in other objectives, so spectral normalization is always adopted even when the training objectives are not Wasserstein distance [73].

Besides, researchers also propose a number of training techniques to improve the performance of GANs, which include unrolled strategy [92], Exponential Moving Average (EMA) [132], top-k training [116] and the two time-scale update rule [43].

These techniques enable GANs to generate high quality images. Recently, autoregressive models [28, 133, 137] and diffusion models [76, 94, 104, 110, 111] show their effectiveness in image generation. Even so, Recent studies show that GANs are able to obtain comparable performance in much lower latency [57, 113].

2.2 Language GANs

The success of GANs in image generation prompts researchers to build GAN-based models for text generation. However, the non-differentiable sampling process leads the gradients from discriminators can not pass through to generators. Researchers propose a number of methods to tackle this problem and most of them can be divided into two categories: *REINFORCE* methods [13, 63, 134] and *continuous relaxation* methods [14, 70, 139].

Yu et al. [134] first adopts *REINFORCE* in GANs in unconditional text generation. As an early attempt, Yu et al. [134] focus on two problems: 1) Classical GANs can not generate discrete output. 2) The discriminator can return the reward only when the whole sentence is generated. They use *REINFORCE* methods to tackle the first problem. In their model, generators do not need the gradients from discriminators any more. Instead, they regard the outputs of discriminators as rewards and use these rewards to update generators. However, discriminators in traditional GANs structure can only provide rewards for each action (sampling word). They further adopt Monte Carlo search with a roll-out policy to calculate rewards for each action. Li et al. [75] also use similar methods to improve the performance in conversation generation.

To further enhance model performance, researchers propose various modeling methods. In original GANs, the discriminator is trained to finish a binary classification task [34]. Lin et al. [82] consider that it significantly limits the diversity and richness inside sentences. Thus, they propose to use rankers to replace discriminators to calculate a relative rank among the sequences when given a reference. Besides, Fedus et al. [24] focus on the training instability and mode dropping problem. To tackle these problems, They propose to train the generator on a text in-filling task.

When using *REINFORCE* methods, rewards given by discriminators may be extremely unstable. To stabilize the reward value, Che et al. [13] propose a normalized maximum likelihood optimization target. Besides, they further adopt importance

sampling and several variance reduction techniques to stabilize training process. Ke et al. [63] make an another attempt to stabilize training process. Inspired by the work of Norouzi et al. [96], they propose to sample data from a distribution near the real distribution instead of using distributions given by generators directly.

For the *continuous relaxation* methods, the most widely-used one is incorporating Gumbel-softmax into GANs [70]. With the help of Gumbel-softmax, generators can get a distribution that only the value in one dimension is close to 1 and the values in the other dimensions are all close to 0. Thus, these results can be fed into discriminator directly and generators can update the parameters based on the gradients from discriminator.

Researchers also propose a number of variants to improve the performance, like using feature matching scheme [139] and Feature-Mover's (FM) Distance [14]. Nie et al. [95] further improve the performance by making use of a relational memory based generator [112] and multiple embedded representations.

However, these methods all heavily rely on MLE pre-training. Furthermore, Cacciaet al. [11] find that when evaluating language GANs by considering fluency and diversity together, existing models can not outperform the classical MLE methods. Researchers begin to build models based on large scale pre-training models directly [114]. The dependency on pre-training implies the internal limitations in current models. d'Autume et al. [18] first attempt to train GANs in text generation without pre-training generators directly. Their method obtains comparable performance as MLE. Lin et al. [81] further adopt the first-order Taylor expansion into models to try to reduce the batch size. However, both of these two models are built on pre-training component, which means their generators are dependent on pre-training embeddings. In other words, existing GAN-based text generative models rely on either MLE pre-training or pre-trained embedding.

In addition to *REINFORCE* methods and *continuous relaxation* methods, researchers

also give attempt to modelling word representations directly. In this method, generators obtain word representation instead of probability distributions. However, these methods have large gaps comparing with traditional MLE methods [69], or even are listed as failure cases in some work [18]. The limited performance of this method makes it less popular comparing with the other two methods.

2.3 Non-Autoregressive Models

NAR models are first proposed in machine translation [37]. Instead of obtaining tokens one-by-one, they support parallel decoding so have much lower decoding latency comparing with AR models [128]. Different with other generative models which are first raised in unconditional generation and then extended to conditional scenarios. NAR models are mainly discussed in specific tasks like machine translation [32, 62, 102]. Thus, these models were lack of complete theoretical analyses in the early development stage and their development can not completely cover all important scenarios.

Machine translation is one the most popular tasks to study NAR models. Xiao et al. [128] analyze existing work from five aspects:

- Data Manipulation. The lost of word dependencies lead most of NAR models to be incapable in modeling complicated data distributions. Thus, researchers often simplify the distributions by adopting knowledge distillation [44, 65].
- Modeling. Different modeling methods are also proposed to enhance model performance. For example, using iteration-based methods to construct the model, which improve output quality iteratively [32]. Besides, researchers also explore the method of using latent variables to learn target dependencies or integrating more information in the input or hidden states to improve model performance [2, 8, 88].

- Criterion. A number of researchers argue that the vanilla MLE is not suitable for the training of NAR models, so they propose to use different loss functions like Connectionist Temporal Classification (CTC) loss [36], N-gram level loss [115], Aligned Cross Entropy (AXE) loss [31] and Order-Agnostic Cross Entropy (OAXE) loss [22].

- Decoding. Decoding strategy is another perspective to improve model performance. In addition to decode sentences iteratively [32], researchers also propose a semi-autoregressive (SAR) structure whose decoding stage contains both AR and NAR processes [123].

- Pre-trained Model. Recent research also adopts pre-trained techniques to further boost the performance of NAR models [78, 84, 124].

Recently, Huang et al. [49] provide a unified perspective to analyze existing models. They first reveal that optimizing the NAR models with MLE remains a non-negative lower bound between learned distributions and real distributions, which is:

$$D_{KL}[P_{data}(Y|X)||p_\theta(Y|X)] \geq \underbrace{-H_{data}(Y|X) + \sum_{i=1}^{T} H_{data}(y_i|X)}_{C} \tag{2.1}$$

where C is a non-negative constant called conditional total correlation. It quantifies the dependency among a set of variables.

Huang et al. [49] find that existing mainstream techniques improve the performance of NAR models by decreasing the conditional total correlation. They divide existing methods into two categories: modifying targets and enhancing inputs. For the methods of modifying targets, knowledge distillation uses well-trained AR model to generate pseudo targets to train NAR models, while methods like AXE and OAXE change the targets adaptively. For the methods of enhancing inputs, CMLM is a fixed method in enhancing inputs while GLAT is an adaptive methods. All these methods

try to maintain a one-to-one mapping relation so to intuitively reduce the conditional total correlation.

However, this limitation inherently exists in MLE-based NAR models. Existing methods can only relieve it instead of completely tackling it. Comparing with the rapid development of NAR models in machine translation, their development in some tasks are relatively slow. Image captioning is one of them.

Early study of NAR models in image captioning adopts iterative-based methods to accelerate inference [25, 29]. However, these methods are trained on cross-entropy and are not able to keep sentence-level consistency. To maintain sentence-level consistency, Guo et al. [40] integrate the counterfactuals-critical multi-agent learning into the training objective. Recently, researchers further enhance model performance by making use of various structures. Semi-autoregressive (SAR) structures [26, 131] are one of the methods. These methods have both autoregressive and non-autoregressive generation processes and thus require multiple steps to obtain the results. In addition, Luo et al. [87] incorporate diffusion models and obtain image captions in an iterative manner. Although these methods obtain better performance than fully NAR models, their inference latency is also higher.

Chapter 3

Representation Modeling Based Language GANs

3.1 Introduction

Text generative models are stepping stones for various natural language processing tasks [89, 135]. Implementing Maximum Likelihood Estimation (MLE) with autoregressive structure has gained great success [30, 119, 120]. This method uses ground truth as input during training, but reads previously generated tokens during inference. The discrepancy between training and inference, however, causes the exposure bias problem [9, 35, 138]. This problem affects the quality of generated sentences and grows the needs of exploring other alternatives in text generation. Generative Adversarial Networks (GANs) [34] are central in many image generation success stories [10, 60, 61]. GANs can tackle the exposure bias problem by providing a consistent generation manner in training and inference.

However, the non-differentiable sampling operations in text generators stop gradients from passing through to generators, which limit the direct applications of GANs in text generation [134]. Currently, many researchers tackle this problem by *RE-*

INFORCE [125] or *continuous relaxations* [54, 90]. *REINFORCE* is an unbiased but high variance estimator [81], whereas *continuous relaxations* are low variance but biased estimators [18]. The inherent limitations of these two methods not only constraint the performance, but also lead the fragile training of GANs to be more unstable. Existing language GANs thus rely on either MLE pre-training or pre-trained embedding to be comparable with MLE [81, 18, 114].

Methods based on *REINFORCE* or *continuous relaxations* explicitly model word probabilities at each timestep, so we denote them as **Probability Modeling Methods (PMMs)**. Another type of methods is to first transform words into representations, and then train generators to model these representations. We denote these methods as **Representation Modeling Methods (RMMs)**. Research on RMMs is extremely limited, due to the unsatisfactory performance in previous attempts [18, 69]. However, such methods should be a promising research line, since they contain neither non-differentiable operations nor biased estimators. The discrepancy between theoretical feasibility and unexpected poor performance prompts us to conduct an in-depth analysis of its reasons, based on which two problems are found as responsible for the poor performance of RMMs.

The first one is called "invalid sampling" problem. RMMs do not have word probabilities that can be sampled. Injecting random noise into generators is also demonstrated as ineffective in autoregressive structures [105, 143]. Generators with an invalid sampling method will generate samples in high similarities, and leads to the mode collapse problem [5, 43]. Another problem is unhealthy gradients. RMMs update generators based on gradients from discriminators. Compared with other sequence models [100], more layers are stacked to build the discriminator and the generator, so RMMs place higher demands on healthy gradients. Gradient vanishment is more severe in LSTM [46], for the output gate there further narrows down the gradients from other layers. Unhealthy gradients will directly influence the performance of generators.

To tackle the first problem, we introduce a simple but effective sampling method: **dropout sampling**. Unlike injecting random noise, it provides a non-negligible random factor by masking a certain number of dimensions in input. This method improves both diversity and quality of generated samples via relieving the mode collapse problem.

We solve the second problem by proposing a new variant of LSTM: **fully normalized LSTM**. Our theoretical analyses show that incorporating layer normalization [6] in the calculation of hidden states can relieve gradient vanishment by providing an additional augmentation term in its derivative. This operation, however, is omitted in the existing combination of layer normalization and LSTM [6]. Fully normalized LSTM makes up this shortcoming by simultaneously obtaining strong sequence modeling capabilities and healthier gradients.

Theoretically, language GANs can get satisfactory performance without any pre-training techniques (MLE pre-training or pre-trained embedding).We present **InitialGAN** to echo this significant goal in text generation. The contributions of this work can be summarized as follows:

- We provide in-depth analysis and offer effective solutions to the two main limitations of representation modeling methods. For the invalid sampling problem, we introduce dropout sampling which is a simple but effective sampling method to improve both the quality and diversity of generated samples. For the unhealthy gradient problem, we propose a fully normalized LSTM which can relieve gradient vanishment by making use of layer normalization to provide an augmentation term.

- We put forward InitialGAN as a representation modeling based language GAN which is characterized by having all the parameters to be initialized randomly. In particular, InitialGAN has three models: mapper, generator and discriminator. The mapper transforms words into representations, and the generator tries

to model word representations; the discriminator uses these representations as input and identify whether the representations are from the mapper or the generator. Different from existing language GANs which are based on pre-training techniques, all the parameters in InitialGAN are initialized randomly.

- Observing that the existing embedding level metric is not sensitive to the change of sample quality, we propose a new metric: Least Coverage Rate, which can better identify the differences among different models. The experimental results show that InitialGAN outperforms both MLE and other compared models. To the best of our knowledge, it is the first time a language GAN can outperform MLE without using any pre-training techniques. It also demonstrates that RMMs denote a promising research line for language GANs.

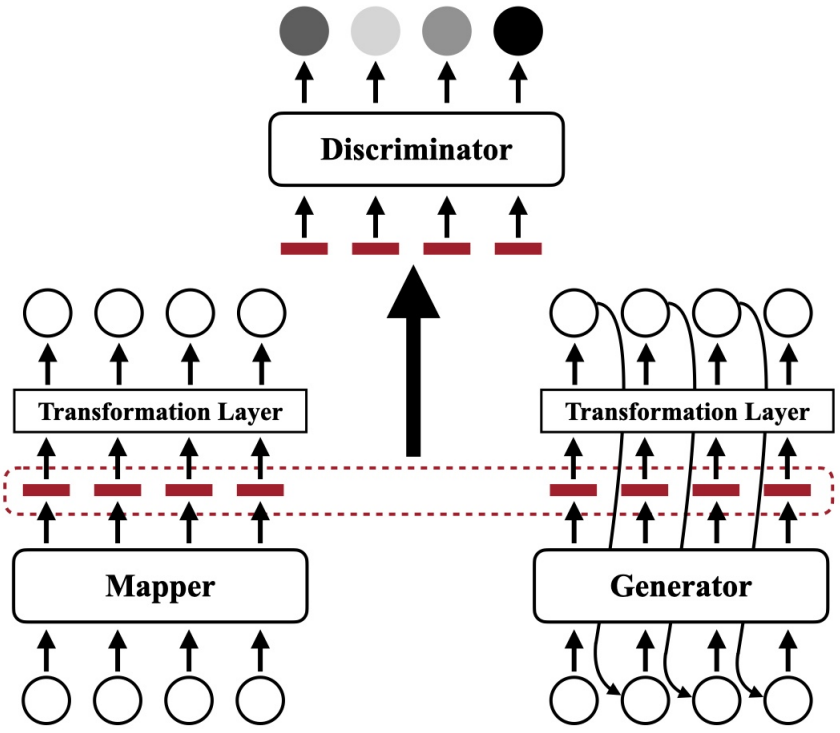

Figure 3.1: Structure of InitialGAN.

3.2 Model

In this section, we firstly introduce the structure of InitialGAN. Next, we present dropout sampling and fully normalized LSTM to tackle invalid sampling method and unhealthy gradients, respectively. After that, we introduce the training objectives of InitialGAN.

3.2.1 Model Structure

The structure of InitialGAN is shown in Figure 3.1. There are three models in InitialGAN: mapper, generator and discriminator. The mapper is based on the encoder in Transformer [120]. It needs to map words to representations. For the discriminator, we train it to identify whether a specific representation in the t-th timestep is from the mapper or the generator based on the previous $(t-1)$ representations:

$$\mathbf{c}_t = D(\mathbf{r}_t|\mathbf{r}_{t-1}, ..., \mathbf{r}_1)$$

where $\mathbf{r}_t$ is the t-th representation from the mapper or the generator. For the generator, we use latent input $\mathbf{z}_t$ to generate representations and their corresponding words:

$$\mathbf{r}_t^{(g)} = G(\mathbf{z}_t|\mathbf{z}_{t-1}, ..., \mathbf{z}_0) \tag{3.1}$$

$$\hat{x}_t = F_{LT}(\mathbf{r}_t^{(g)}) \tag{3.2}$$

where $\mathbf{r}_t^{(g)}$ is the t-th representation and $F_{LT}(\cdot)$ is the transformation layer from the mapper. Its parameters are fixed during the training of the generator. In the following, we elaborate the details about the two important techniques in InitialGAN: Dropout Sampling and Fully Normalized LSTM.

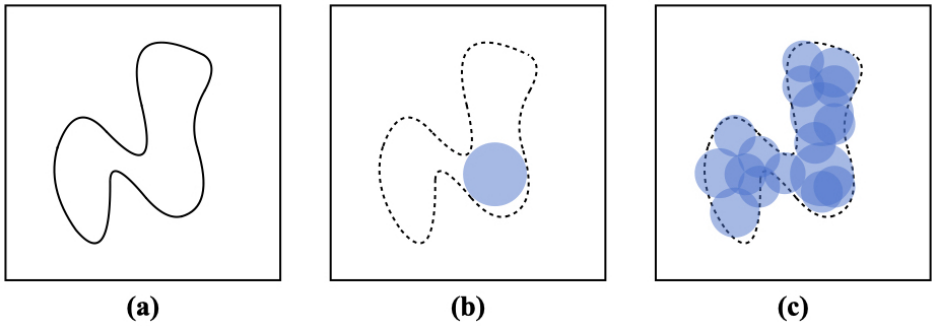

Figure 3.2: Effects of Dropout Sampling. (a) The real distribution. (b) The distribution learned by a sub-model (blue area). (c) The distribution learned by the complete model.

3.2.2 Dropout Sampling

An effective sampling method plays a crucial role in training GANs. Once we choose to model representations, we can no longer sample words from word probabilities directly. Although we can sample results by feeding random noise into generators as a part of input, previous work [105, 143] shows that generators with autoregressive structures tend to ignore those additional input. As a result, these generators will suffer from the mode collapse problem [5, 43] and give samples in high similarities. In this time, the discriminator will prompt generated samples to rotate between the different modes, so the generator can only learn a small subset of the real distributions instead of the whole one.

Ever since Dropout [117] was introduced in 2014, it has been widely used in training neural networks. Previous work [53] also adopts dropout as random noises in image GANs. Using dropout in both training and inference as a sampling method is extremely suitable in representation modeling methods because it provides a non-negligible random factor. During training, the hidden representations in the generator will be randomly masked. Under the guidance of the discriminator, the generator will learn to use different combinations of the values in hidden representations to obtain

results following real distributions. During inference, the generator has consistent masking strategy to ensure the quality of generated results.

However, the distribution provided by dropout sampling is decided by the input which is always in the form of trainable embeddings. It leads the distribution to keep changing during training and may cause the training process to be unstable. Thus, we concatenate the input with random noise to increase the robustness of the model. The complete method is:

$$\mathbf{z}_t = Dropout(E(\hat{x}_{t-1}) \oplus \epsilon, \rho)$$

where $\mathbf{z}_t$ is the t-th latent variable, $\hat{x}_{t-1}$ is the word generated in the last timestep, $E(\cdot)$ is a function to transform words into embeddings, ϵ is random noise sampled from a pre-defined distribution and ρ is the dropout rate. Dropout can be viewed as the selection of sub-models. Given a real distribution in Figure 3.2 (a), even though each sub-model may still suffer from mode collapse as shown in Figure 3.2 (b), different sub-models can cover different modes. Hence the distribution given by the complete model is closer to the real data distribution (as shown in Figure 3.2 (c)).

The dropout operation will mask a certain number of dimensions of hidden vectors. Dropout sampling will thus slow down the convergence of generators, since the parameters are updated less frequently. To speed up the training process, we propose to use imbalanced batch size. Suppose bs_d is the batch size of discriminator's training, setting the batch size of the generator's training as $bs_g = bs_d/(1-\rho)$ can have more samples for the generator to update and thus bridge the gap in update frequency.

3.2.3 Fully Normalized LSTM

When building representation modeling based language GANs, generators update their parameters based on the gradients from discriminators. Compared with other sequence models [100], representation modeling methods place higher demands on

healthy gradients, since they need to stack more layers to build discriminators and generators.

In a language GAN, a generator makes predictions based on previous output in both training and inference. It limits the use of Transformer [120] whose computational speed is extremely slow without parallel computation. When generating the t-th word, Transformer needs to calculate attention weights for the previous $t-1$ words, and it has relatively high complexity. Consequently, LSTM [46] is more popular in language GANs [18]. To obtain the output, it only needs to consider the hidden state from the last timestep and the current input, so it has a constant fast computational speed.

When using LSTM to build a representation modeling method, however, we need to care about possible gradient vanishment among both different timesteps and different layers. This time, the gated mechanism in LSTM cuts both ways. Although the grated mechanism can relieve gradient vanishment among different timesteps, the hidden states of LSTM needs to multiply with the results from the output gate, whose values are between $(0, 1)$. Thus, the gradients from the previous layers will be inevitably narrowed.

The problem gets worse when we stack several LSTM layers to build the generator and the discriminator. Unhealthy gradients will exist throughout the whole training process and affect model performance directly. Thus, we need to find a method to relieve the gradient vanishment problem.

Layer normalization [6] is a widely used technique in neural networks [120]. Previous work [129] shows that layer normalization helps stabilize training by reducing the variance of gradients. We further find that layer normalization has potential to relieve gradient vanishment. A common understanding is that it can shift and scale input into a more reasonable interval to avoid the interval whose gradients are small. However, layer normalization does more than that. According to our analyses, it provides an

addition term to augment gradients when the deviation of the normalized term is smaller than 1, as stated in Theorem 1 below.

Theorem 1 *Suppose $\mathbf{y}_{l+1} = F_l(\mathbf{y}_l)$ is the l-th layer in a model, $\mathbf{y}_l$ is the input of the l-th layer and also the output from the $(l-1)$-th layer. Adopting layer normalization in the input (i.e., $\mathbf{y}_{l+1} = F_l(LN(\mathbf{y}_l))$) provides an addition term when calculating the partial derivative of $\mathbf{y}_{l+1}$ with respect to $\mathbf{y}_l$. This term can augment the gradients when the deviation of $\mathbf{y}_l$ is smaller than 1.*

The proof of Theorem 1 can be found in the Appendix A.1. In LSTM, the hidden state can be regarded as the $\mathbf{y}_l$ in Theorem 1. It is the element-wise product of the output gate (which is in $(0, 1)$) and cell state (which is in $(-1, 1)$). Its deviation must be smaller than 1, which can meet the conditions of Theorem 1. Thus, we have the following:

Corollary 1 *Adopting layer normalization when calculating the hidden state in LSTM provides a scaler factor to augment the gradients, thereby mitigating the gradient vanishment between different layers in LSTM.*

Based on our analyses, we propose a fully normalized LSTM as follows:

$$\begin{aligned}
\begin{pmatrix} \mathbf{f}_t \\ \mathbf{i}_t \\ \mathbf{o}_t \\ \hat{\mathbf{c}}_t \end{pmatrix} &= LN(\mathbf{W}_h \mathbf{h}_{t-1}) + LN(\mathbf{W}_x \mathbf{x}_t) + \mathbf{b} \\
\mathbf{c}_t &= LN(F_s(\mathbf{f}_t) \circ \mathbf{c}_{t-1} + F_s(\mathbf{i}_t) \circ F_h(\hat{\mathbf{c}}_t)) \\
\mathbf{h}_t &= LN(F_s(\mathbf{o}_t) \circ F_h(\mathbf{c}_t))
\end{aligned} \tag{3.3}$$

where $\circ$ is the element-wise product, $LN(\cdot)$ is layer normalization, $\mathbf{b}$ is a trainable vector, $F_s(\cdot)$ is sigmoid function and $F_h(\cdot)$ is tanh. The fully normalized LSTM is based on LayerNorm LSTM [6] whose effectiveness has been shown by various tasks [18, 47].

The main difference is that the fully normalized LSTM adopts additional layer normalization when calculating the hidden state, so as to offset the influence from the output gate. Thus, it can obtain both strong sequence modeling capabilities and healthier gradients. In practice, we adopt fully normalized LSTM to construct both the discriminator and the generator to obtain healthier gradients.

3.2.4 Training Objective

The training objective of the mapper is based on the loss function of Variational Autoencoder (VAE) [67], so the mapper can map word x_i into a distribution $\mathcal{N}(\mu_{x_i}, \sigma^2_{x_i})$. The vector sampled from $\mathcal{N}(\mu_{x_i}, \sigma^2_{x_i})$ is transformed back into words with a linear transformation F_{LT}. The distribution $\mathcal{N}(\mu_{x_i}, \sigma^2_{x_i})$ describes a region in the space as representation rather than a point. It can increase the robustness during generation, since F_{LT} can map representations with minor errors into correct words.

However, the mapper trained via the VAE objective tends to assign large regions to high frequency words, while the regions of low frequency words are extremely small. It brings difficulties to generate low frequency words, since small errors may lead the representations to lie in the region of other words. Thus, we propose a new training objective to tackle this problem:

$$L_A = -\mathbb{E}_{\hat{\mathbf{z}} \sim q(\hat{\mathbf{z}}|x_i)}(logp(x_i|\hat{\mathbf{z}})) + KL(q(\hat{\mathbf{z}}|x_i)||p(\hat{\mathbf{z}})) + \lambda_a log(\sigma^2_{x_i}) \tag{3.4}$$

where $\sigma^2_{x_i}$ is the variance of word x_i and λ_a is a hyperparameter. The first two terms consist of the original objective in VAE. The last term provides a penalization to the variance, which controls the size of the representation region directly. This objective can help keep the regions of different words in similar size by giving more penalization to the variance of high frequency words. In practice, λ_a is set to be a small number and the original training objective of VAE can ensure that $\sigma^2_{x_i}$ is not too small.

We train the mapper based on the strategy of BERT [19]. More specifically, 15% of

Algorithm 1 Training of InitialGAN

Input: Initial parameters of the mapper, discriminator, and the generator. Batch size m and imbalanced batch size m'.

while the mapper has not converged **do**

for $i = 1 \rightarrow m$ **do**

Sample real data $x \sim \mathbb{P}_x$

$\mu_x, \sigma_x^2 \leftarrow M(x)$

$\hat{\epsilon} \sim \mathcal{N}(0, 1)$

$\hat{\mathbf{z}} \leftarrow \hat{\epsilon} \circ \sigma_x + \mu_x$

$L_A^i \leftarrow -\mathbb{E}_{\hat{\mathbf{z}} \sim q(\hat{\mathbf{z}}|x)}(logp(x|\hat{\mathbf{z}})) + KL(q(\hat{\mathbf{z}}|x)||p(\hat{\mathbf{z}})) + \lambda_a log(\sigma_x^2)$

$L_A \leftarrow L_A + L_A^i$

end for

Update the parameters of the mapper based on L_A

end while

while the generator has not converged **do**

for $i = 1 \rightarrow m$ **do**

Sample $x \sim \mathbb{P}_x$, $\mathbf{z} \sim \mathbb{P}_z$.

$\mu_x, \sigma_x^2 \leftarrow M(x)$

$L_D^i \leftarrow -D(\mu_x) + D(G(\mathbf{z}))$

$L_D \leftarrow L_D + L_D^i$

end for

Update the parameters of the discriminator based on L_D

for $i = 1 \rightarrow m'$ **do**

Sample latent variable $\mathbf{z}^{(i)} \sim \mathbb{P}_z$

$L_G^i = -D(G(\mathbf{z}^{(i)}))$

$L_G \leftarrow L_G + L_G^i$

end for

Update the parameters of the generator based on L_G

end while

words in training data will be randomly selected as the ones to be predicted. Among the replacing words, 80% of them are replaced with [MASK] token, 10% of them are replaced with random tokens, and 10% of them are unchanged. In the training objective of the mapper, we propose to add an additional term to penalize $\sigma^2_{x_i}$. Vanilla VAE also needs to calculate $\sigma^2_{x_i}$ to update the KL divergence term in the objective, so there is nearly no additional computational cost in the new training objective. After the training of the mapper is finished, all its parameters are fixed and the transformation layer which transforms representations back into words will be shared with the generator.

Both the discriminator and the generator are constructed based on the fully normalized LSTM. Dropout sampling is adopted in both training and inference stage of the generator. The generator uses the same linear transformation F_{LT} in the mapper to transform representations back into words. We adopt Wasserstein distance [5] as the training objective, and use Lipschitz penalty [101] to regularize the discriminator. The loss functions of the discriminator L_D and the generator L_G are:

$$L_D = -\mathbb{E}_{\mu_x \sim \mathbb{P}_d}[D(\mu_x)] + \mathbb{E}_{z \sim \mathbb{P}_z}[D(G(\mathbf{z}))] \tag{3.5}$$

$$L_G = -\mathbb{E}_{z \sim \mathbb{P}_z}[D(G(\mathbf{z}))] \tag{3.6}$$

where $\mathbf{z}$ is from dropout sampling, and μ_x is the word representation from the mapper. The full training process of InitialGAN is described in Algorithm 1.

3.3 Experiment

In this section, we introduce the evaluation metrics, datasets and compared models. After that, we provide the experimental results with our analyses. In the experiment, we first demonstrate the effectiveness of InitialGAN in both short sentence generation and long sentence generation. Then, we show the reliance of existing models on pre-trained techniques. Besides, we conduct ablation study to show the effectiveness

of dropout sampling and fully normalized LSTM. Furthermore, models are compared from the training stability and sentence length distribution. Last but not least, additional experimental results like the effectiveness of fully normalized LSTM to gradient are demonstrated to give more complete analyses to our proposed technique.

3.3.1 Evaluation Metrics

For the token level metrics, we use **BLEU** [98] to evaluate fluency and **Self-BLEU** [144] to evaluate diversity. In addition, **Inverse BLEU** is also a good choice for evaluating the overall performance in both fluency and diversity. Inverse BLEU uses sentences in test sets as inference and generated sentences as references. Sentences in test sets are fluency and diverse, so the generated sentences can get high Inverse BLEU only when they have good performance in terms of both aspects. When calculating token level metrics, all of them are calculated up to 5 grams and the size of each set is set to be 5,000.

For the embedding level metrics, we use **Fréchet Embedding Distance (FED)** [18] which is identical with Fréchet inception distance (FID) [43] except for the encoding model. Although it can evaluate the global similarity of two distributions, previous work [81, 18] shows that its values are extremely small, and it is not sensitive to the change of sample quality. When models get similar FED, it does not mean that they get close performance. To further identify the differences of compared models, we propose a new metric, **Least Coverage Rate (LCR)**, which is calculated as follows:

$$
\begin{aligned}
S_{ij} &= Sim(\mathbf{E}(\mathbf{x}_i^a), \mathbf{E}(\mathbf{x}_j^b)) \\
\mathbf{R}_a &= \frac{1}{n}\sum_{i=1}^{n}\delta(\sum_{j=1}^{m} S_{ij} \geq \tau) \\
\mathbf{R}_b &= \frac{1}{m}\sum_{j=1}^{m}\delta(\sum_{i=1}^{n} S_{ij} \geq \tau) \\
LCR(\mathbf{X}_a, \mathbf{X}_b) &= min(\mathbf{R}_a, \mathbf{R}_b)
\end{aligned}
\tag{3.7}
$$

where $\mathbf{x}_i^a$ and $\mathbf{x}_i^b$ are the i-th and j-th sentences from sentence sets $\mathbf{X}_a$ and $\mathbf{X}_b$, respectively. $\mathbf{E}(\cdot)$ is the model to transform sentences into embeddings, τ is a hyper-parameter, $Sim(\cdot)$ is a similarity function and $\delta(\cdot)$ is a function which returns 1 if input is higher than 0, and 0 for others. $\mathbf{R}_a$ and $\mathbf{R}_b$ are the coverage rates of $\mathbf{X}_a$ and $\mathbf{X}_b$, respectively. LCR has following features:

- It compares two distributions in a fine-grained level. Given two sets of sentences, LCR computes the similarity of every two sentences in the two sets to make sure whether specific modes are covered or not. Thus, it can be more sensitive to the change of sample quality.

- The minimum operation in LCR helps it be sensitive to two common problems in generative models: 1) generating samples out of the real distribution; 2) generating samples in high similarities. The coverage rates on test sets are aware of the mode collapse problem, while the coverage rates on inference sets can identify the generated samples out of the real distribution.

- It better makes use of sentence encoders. Most of sentence encoders are designed to compare sentence similarities with a pre-defined method (e.g., cosine similarity). FED only considers the mean and covariance of two sets, and does not make use of this feature directly.

- It is efficient in computation. Although it needs to compute the similarity of every two sentences in the two sets, it can be implemented in a high efficiency way. For example, if we use Universal Sentence Encoder to transform sentences into embeddings, we can easily implement the calculation by making use of matrix multiplication.

When adopting these two embedding level metrics, we select 10,000 sentences in each set and use Universal Sentence Encoder [12][1] to encode them into embeddings. For

[1] `https://tfhub.dev/google/universal-sentence-encoder/4`

the similarity function in LCR, we use cosine similarity suggested by the Universal Sentence Encoder [12].

3.3.2 Experiment Setup

We use two datasets in our experiment: COCO Image Caption Dataset [83] [2] and EMNLP 2017 News Dataset [3]. For the COCO Image Caption Dataset, we choose 50,000 sentences as training set. For the EMNLP 2017 News Dataset, we choose 200,000 sentences as training set. These sentences are used to prepare experiment for unconditional generation instead of their original tasks.

We compare the performance of InitialGAN with MLE and other language GANs. For *REINFORCE* methods, we choose SeqGAN [134], RankGAN [82], MaliGAN [13], LeakGAN [39] and ScratchGAN [18]. For *continuous relaxation* methods, we choose RelGAN [95]. All these methods except for ScratchGAN rely on MLE pre-training, while ScratchGAN is based on pre-trained embeddings. InitialGAN is the only language GAN whose parameters are initialized completely randomly.

The mapper is constructed based on the original Transformer structure. The word embeddings will be added with fixed positional encoding and then feed into a stack of Transformer blocks. Each block is consisted of a multi-head attention layer and feed forward network layer. Layer normalization is adopted after each layer. Both the generator and the discriminator are constructed based on the fully normalized LSTM.

The batch size of InitialGAN is set to be 128. The maximum training epoch of the mapper is 200, while the maximum adversarial training epoch is set to be 3,000 for the COCO dataset and 2,000 for the EMNLP dataset. The embedding size is 512. The feature size of the mapper is set to be 512, while the feature size of the discriminator

[2] `https://cocodataset.org`

[3] `http://www.statmt.org/wmt17/`

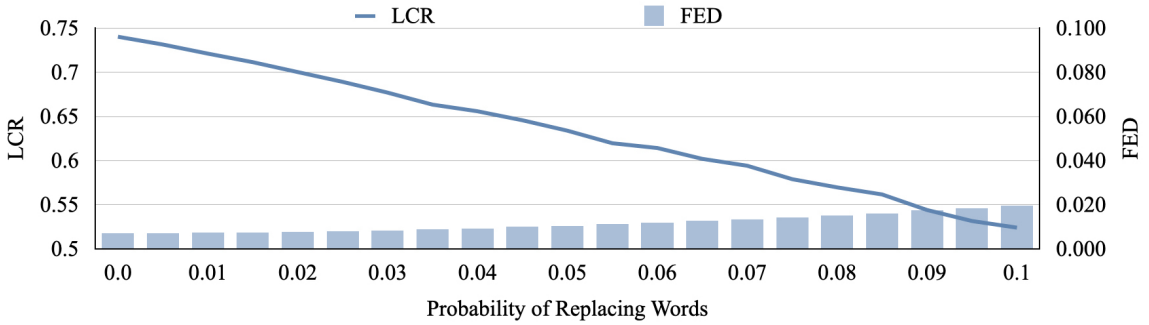

Figure 3.3: Changes of LCR and FED on Image COCO Caption Dataset.

and the generator is set to be 1024 and 512, respectively. The dimension of random noise is 128. We stack 4 transformer layers to build the mapper and its head number is set to be 8. Both the layer number of the discriminator and the generator is set to be 2. The dropout rate of the mapper is 0.5, and the dropout rate in dropout sampling is set to be 0.75. The learning rate of the mapper is 0.0001 and it is optimized with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay=0.00001). The learning rates of the discriminator are set to be 0.0004 for the COCO dataset and 0.0002 for the EMNLP dataset. Its optimizer is AdamW ($\beta_1 = 0.5$, $\beta_2 = 0.9$, weight decay=0.0001). The learning rate of the generator is set to be 0.0001 and its optimizer is Adam ($\beta_1 = 0.5$, $\beta_2 = 0.9$).

We save the model after every epoch, and select the best model based on FED on validation sets. We follow the settings of the previous work [144] to implement MLE except for the layer number and feature size which are set to be same with the generator in InitialGAN. It not only leads the model to be more comparable, but also improves its performance. We obtain the results of other language GANs by running the public code [4] [5] [6]. We implement InitialGAN based on Tensorflow [1]. It is trained on NVIDIA GeForce RTX 3090.

[4] `https://github.com/geek-ai/Texygen`

[5] `https://github.com/deepmind/deepmind-research/tree/master/scratchgan`

[6] `https://github.com/weilinie/RelGAN`

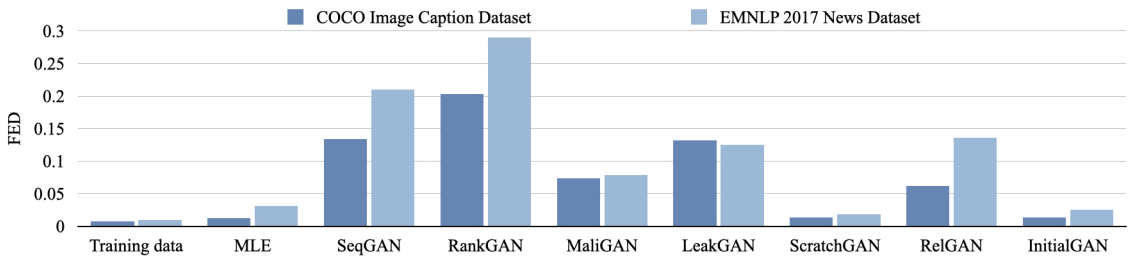

Figure 3.4: Evaluation Results in Fréchet Embedding Distance (FED) on COCO Caption Dataset and EMNLP 2017 News Dataset. Lower is Better.

3.3.3 Experimental Results

Comparisons between FED and LCR

We first conduct experiments to explore the effectiveness of Least Coverage Rate (LCR). We explore the sensitivity of LCR and FED to data changes by replacing words in sentences with random ones in a certain probability. The experimental results are shown in Figure 3.3.

Generally, the changes of FED are not obvious even when replacing 10% of words with random words. In the early stage, we nearly cannot observe any changes in FED. It shows that FED is not sensitive to the change of sample quality. Different with FED, LCR is a much more sensitive evaluation metric. Even minor changes can be reflected in LCR. LCR decreases from around 0.75 to less than 0.5 when 10% of words are replacing with random words. This result shows that LCR can be a good compliment when the compared model gets close performance in FED.

Performance of Different Models

Figure 3.4 reports FED of different models. On the COCO Image Caption Dataset, MLE, ScratchGAN and InitialGAN can significantly outperform other compared models; the differences among these three models are not clear. On the EMNLP

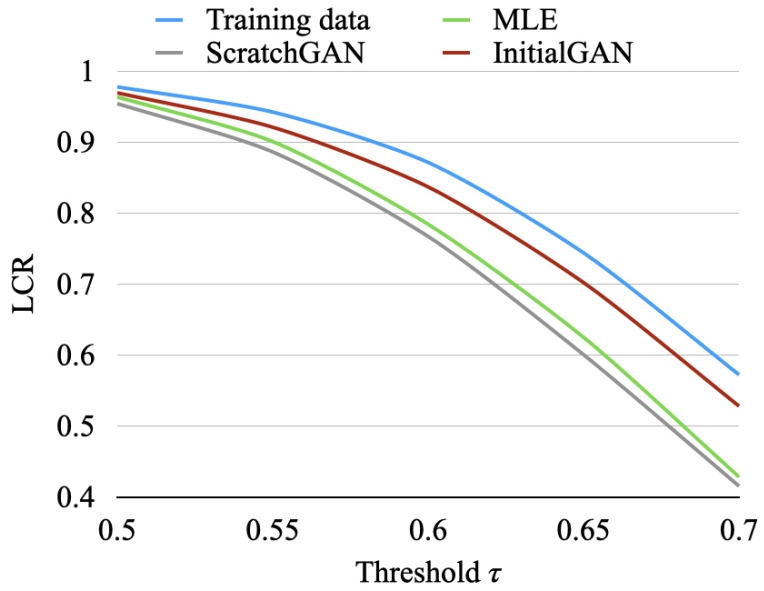

Figure 3.5: LCR with different τ on the COCO Dataset. Higher is Better.

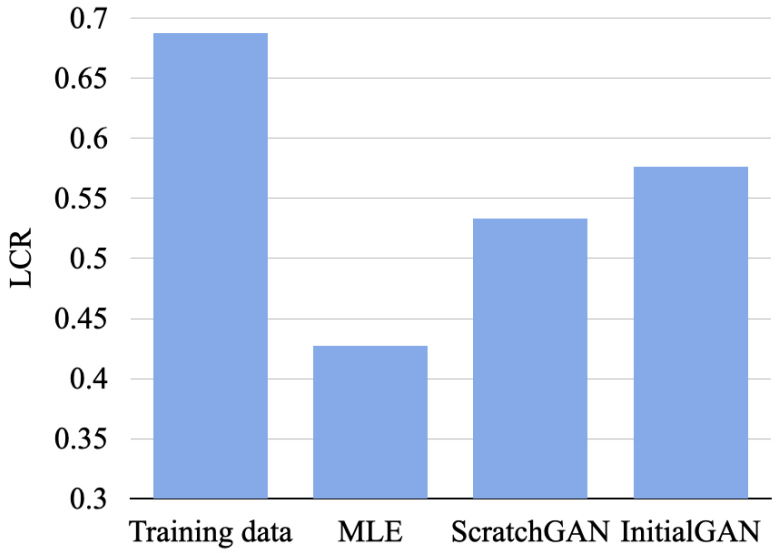

Figure 3.6: LCR on EMNLP 2017 News Dataset ($\tau = 0.45$). Higher is Better.

Table 3.1: Evaluation Results of Token Level Metrics on Image COCO Caption Dataset

Model	**BLEU**	**Self-BLEU**	**Inverse BLEU**
Training Data	34.99	34.80	35.36
MLE	32.59	37.15	32.03
SeqGAN	34.68	69.85	22.34
RankGAN	37.32	73.30	22.10
MaliGAN	26.49	53.47	25.95
LeakGAN	33.14	56.88	29.43
ScratchGAN	30.98	35.72	30.76
RelGAN	54.04	73.70	29.53
InitialGAN	34.87	39.06	**33.06**

Table 3.2: Evaluation Results of Token Level Metrics on EMNLP 2017 News Dataset

Model	**BLEU**	**Self-BLEU**	**Inverse BLEU**
Training Data	20.50	20.47	20.62
MLE	16.66	17.21	16.97
SeqGAN	9.01	27.89	9.90
RankGAN	10.35	56.77	10.37
MaliGAN	12.23	21.34	13.11
LeakGAN	27.61	50.55	11.59
ScratchGAN	17.54	19.04	17.19
RelGAN	30.95	57.48	14.74
InitialGAN	19.40	23.74	**17.74**

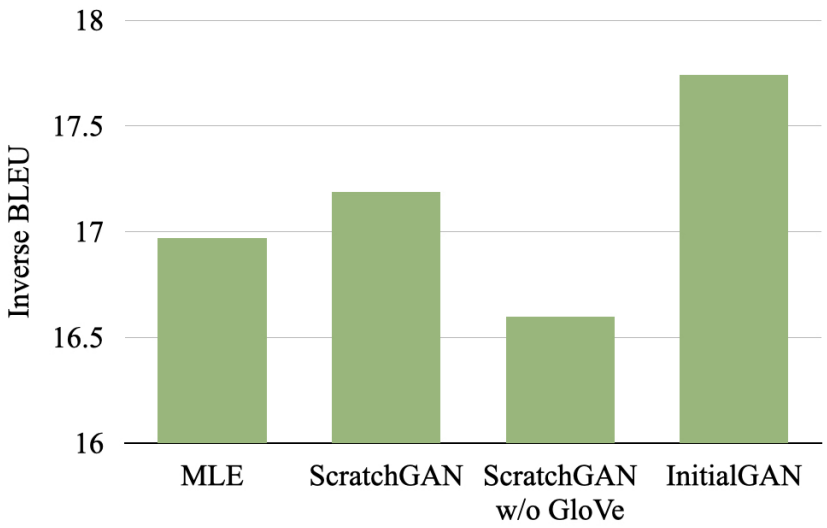

Figure 3.7: Ablation Study of ScratchGAN on EMNLP 2017 News Dataset.

2017 News Dataset, ScratchGAN and InitialGAN slightly outperform MLE, though the gaps between these three models are still limited. FED can not effectively capture the change of data qualities.

To better evaluate the performance among these three models, we further compare their performance in LCR. The results are shown in Figures 3.5 and 3.6. We explore the effectiveness of the threshold τ in LCR on Image COCO Caption Dataset. The results are demonstrated in Figure 3.5. Although the values change significantly with different τ, the rankings of different models are kept when τ is in a reasonable interval. According to Figure 3.5, ScratchGAN is slightly inferior to MLE while InitialGAN can outperform both models. Figure 3.6 shows LCR on EMNLP 2017 News Dataset. InitialGAN gets the highest LCR among all three models. Unlike on Image COCO Caption Dataset, ScratchGAN outperforms MLE on this dataset. EMNLP 2017 News Dataset consists of long sentences and its distribution is more complicated. The exposure bias problem in MLE is more likely to happen, which leads to the poor performance on this dataset.

The evaluation results in token level evaluation metrics are shown in Tables 3.1

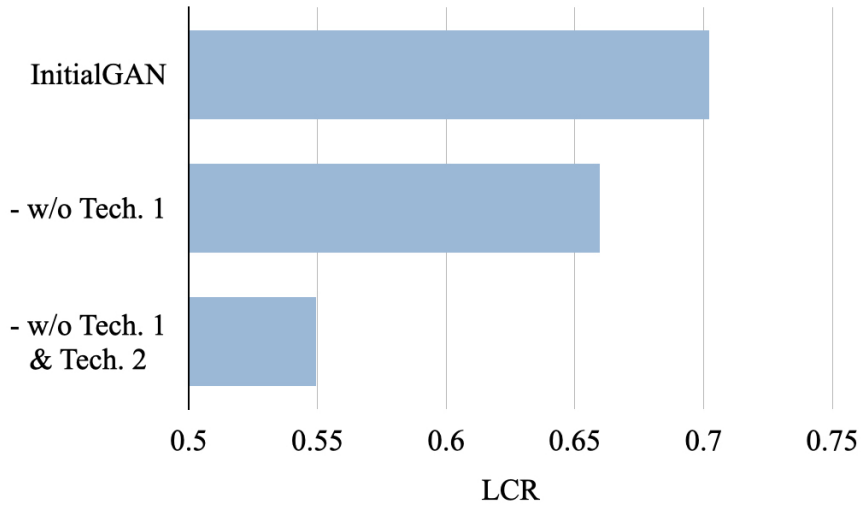

Figure 3.8: Ablation Study of InitialGAN on COCO Dataset. Tech. 1: Dropout Sampling. Tech. 2: Fully Normalized LSTM.

and 3.2. We first analyze the results on COCO Image Caption Dataset. Except for ScratchGAN and InitialGAN, most of language GANs tend to get very high Self-BLEU. Sentences generated by these models have high similarities, which indicates the mode collapse problem in these models. ScratchGAN can tackle this problem and get better result in Inverse BLEU. However, compared with MLE, it still has a gap even with the help of pre-trained embeddings. InitialGAN is the only language GAN which can outperform MLE when considering fluency and diversity together.

Similar results can be found on EMNLP 2017 News Dataset. The difference is that ScratchGAN can slightly outperform MLE in terms of Inverse BLEU. It is consistent with the results in LCR. When generating long sentences, MLE is more likely to meet exposure bias, so MLE gets lower BLEU, which means these sentences are lack of local consistency. Besides, the Self-BLEU shows that sentences generated by MLE are more diverse than training data. A number of generated sentences are out of the real distribution. It explains why MLE only get unsatisfactory performance on this dataset.

We further explore the performance of ScratchGAN without pre-trained embeddings,

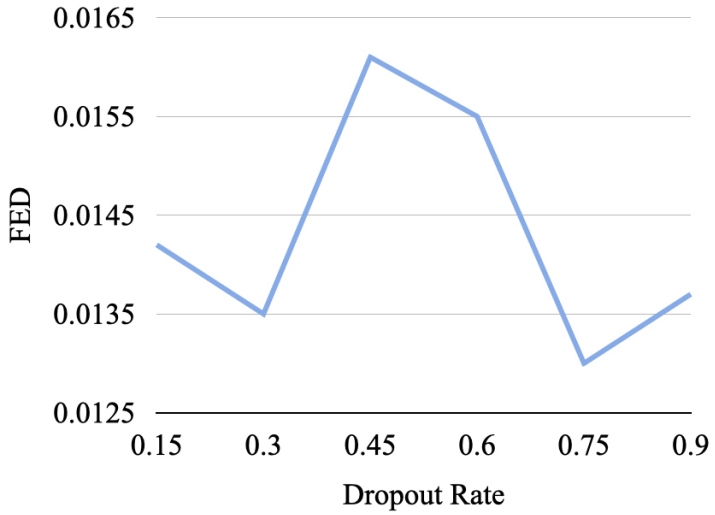

Figure 3.9: Dropout Sampling with different dropout rates.

and show the results in Figure 3.7. Once we remove the pre-trained embeddings, ScratchGAN can no longer outperform MLE, and the gap between ScratchGAN and InitialGAN becomes larger. It reflects the dependence of ScratchGAN on pre-trained embeddings.

Existing language GANs highly rely on pre-training techniques to be comparable to MLE, while InitialGAN is the only language GAN, which can get better performance without using any pre-training techniques. It demonstrates the effectiveness of RMMs against high variance *REINFORCE* or biased *continuous relaxation* methods.

Ablation Study

Figure 3.8 shows the experimental results of the ablation study about dropout sampling and fully normalized LSTM. Without dropout sampling, the LCR decreases a lot because of the mode collapse problem. In this time, the generated samples have extremely high similarities and its Self-BLEU is 71.88, which is much higher than the value of real data. The situation gets worse if we further remove the fully normalized LSTM. These results demonstrate the importance of dropout sampling and

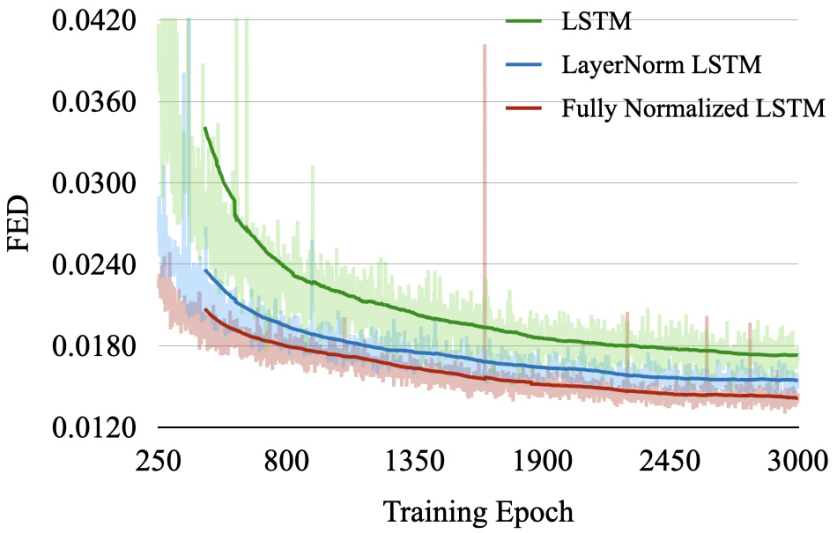

Figure 3.10: Comparisons of different LSTM.

fully normalized LSTM to the performance of InitialGAN. Additional experiments on COCO Image Caption Dataset are conducted to further explore these two proposed techniques. The results are shown in Figure 3.9 and Figure 3.10.

Figure 3.9 shows the influence of dropout rate to FED. The curve shows a rough symmetry. We suppose it comes from the symmetry of combinations. Given a d-dimension vector, the number of possible combinations of masking $\rho \cdot d$ dimensions is the same as the number of masking $(1-\rho) \cdot d$ dimensions (ρ is the dropout rate). Figure 3.10 shows the change of FED on the validation set during the training process. LayerNorm LSTM, the original combination of LSTM and layer normalization [6], can outperform LSTM, while our fully normalized LSTM can further speed up convergence and get better FED.

Additional Analysis

LCR uses the minimum values among two coverage rates as the final results, and analyzing these two coverage rates can also help us better understand the models. We train MLE and InitialGAN with different random seeds on COCO Image Cap-

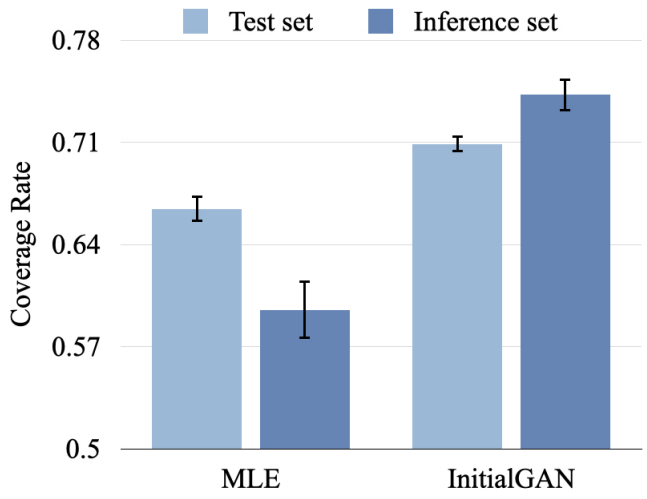

Figure 3.11: Coverage Rate of MLE and InitialGAN ($\tau = 0.65$).

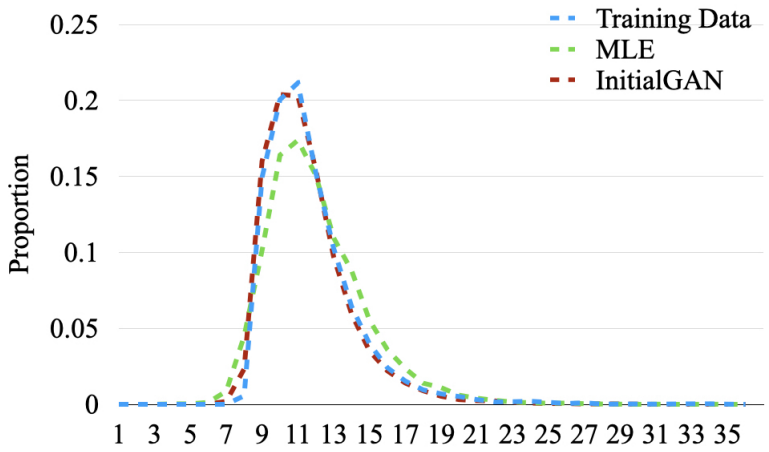

Figure 3.12: Sentence length distribution.

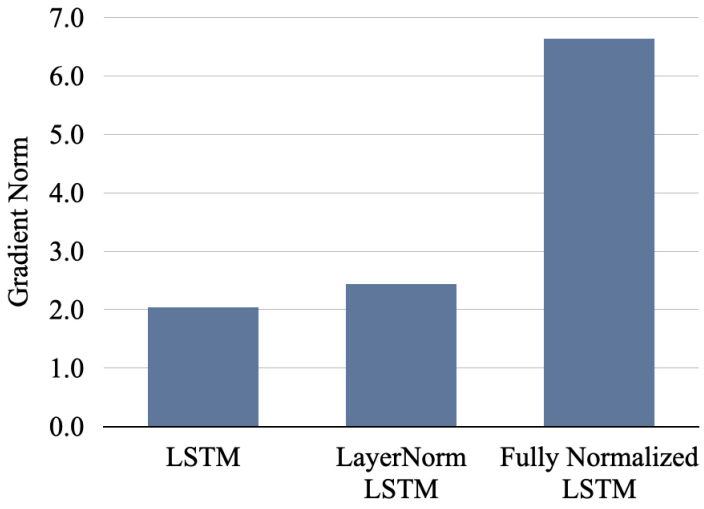

Figure 3.13: Gradient Norm.

tion Dataset, and show their average coverage rates with the standard deviations in Figure 3.11. InitialGAN gets higher coverage rates and its standard deviation is slightly smaller than that of MLE. It shows InitialGAN can get consistently better result with different random seeds. Besides, MLE gets a lower coverage rate in the inference set. We regard the exposure bias as the cause leading MLE to generate sentences out of the real distribution, so its coverage rate of inference set is lower. InitialGAN gets higher coverage rate on the inference set, so further work is needed to relieve the mode collapse problem. We also show the sentence length distributions in Figure 3.12. Compared with MLE, sentences generated by InitialGAN have a closer distribution to the training data.

According to our analyses, fully normalized LSTM can relieve the gradient vanishment by providing an augmentation term. We train models with different variants of LSTM, and show their average gradient norms of first 100 training batches in Figure 3.13. These norms are calculated based on the gradients of the input linear transformation matrix in the last layers of generators. LayerNorm LSTM can slightly augment the gradient norm, while fully normalized LSTM can obtain more obvious augmentation. The experimental results are consistent with our theoretical analyses.

Table 3.3: Number of Parameters and Computation Time of LayerNorm LSTM and Fully Normalized LSTM

Model	**Parameter number**	**Computation time**
LayerNorm LSTM	2,108,416	109.4ms
Fully Normalized LSTM	2,109,440	138.5ms

Besides, we also conduct experiments to compare the number of parameters and computation time of LayerNorm LSTM and Fully Normalized LSTM. We construct 1-layer model with these two structures and the hidden size is set to be 512. We feed a sequence of length 50 into the models 1,000 times and calculate the average running time. The results are shown in Table 3.3. The parameter numbers of Fully Normalized LSTM and LayerNorm LSTM are extremely close. The computation time of Fully Normalized LSTM is slightly higher than LayerNorm LSTM because of the additional layer normalization operation.

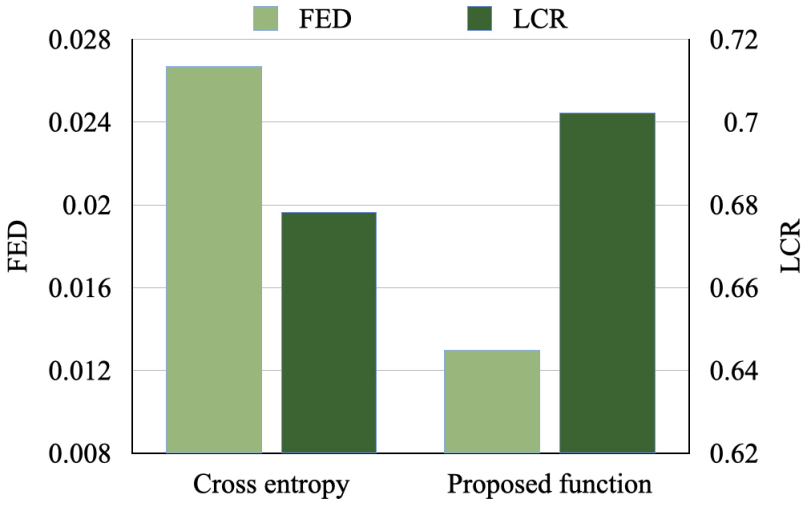

Figure 3.14: The performance of InitialGAN with mappers in different objectives.

In the mapper, we map words into distributions instead of specific embeddings. We show the FED and LCR ($\tau = 0.65$) of these two mapping methods on Image COCO Caption Dataset in Figure 3.14. The mapper trained by cross entropy, which maps

words into specific points, is inferior to the one trained by our proposed loss function in both FED and LCR. Cross entropy does not provide confirmed mapping relations when the generated representations are away from the real representations. Minor errors in the generated representations may lead them to be mapped into totally irrelevant words. It brings additional errors to training and increases instability in inference.

3.4 Summary

In this work, we conduct an in-depth study about constructing language GANs based on representation modeling. We analyze two main problems which limit the performance of representation modeling methods: invalid sampling and unhealthy gradients. To tackle the invalid sampling, we introduce dropout sampling, a simple but effective method. For the unhealthy gradients, we conduct thorough analyses of layer normalization and present the fully normalized LSTM. Armed with these two techniques, we propose InitialGAN which is composed of three models, i.e., mapper, generator and discriminator. Different from existing language GANs which are based on pre-training techniques, all the parameters in InitialGAN are initialized randomly. Besides, we find that FED is not sensitive to the change of sample quality, so we propose Least Coverage Rate (LCR) to better identify the differences among different models. We conduct experiments on two widely used datasets, and the experimental results show that InitialGAN can outperform both MLE and other compared models. To the best of our knowledge, it is the first time that a language GAN can outperform MLE without using any pre-training techniques. This work also demonstrates that RMMs denote a promising research line for language GANs.

Although language GANs can tackle the exposure bias problem, their training speed is one of the most notable limitations which must be tackled. Language GANs need to use previously generated tokens as input in both training and inference, so its training

speed can not be improved by parallel computation structures like Transformer. It limits the applications of current language GANs on large and complicated datasets. How to improve the training speed is a key problem that needs to be solved urgently.

Chapter 4

GAN-based NAR models for Incomplete Information Scenarios

4.1 Introduction

Existing language GANs adopt autoregressive (AR) structure, and have high latency during training and inference. It prompts us to build language GANs based on other structures, of which non-autoregressive (NAR) is a promising one. NAR models have lower decoding latency compared with AR models hence received growing attention from the research community [49]. NAR models are emerging in tasks like machine translation [37] and text summarization [85]. These tasks are based on the scenario where the input contains complete information of target sentences. However, another scenario in which the input does not possess complete target information, is seldomly explored in the study of NAR models. For convenience, we denote these two scenarios as the **Complete Information Scenario (CIS)** and **Incomplete Information Scenario (IIS)**, respectively.

The IIS covers both unconditional and conditional generation tasks, such as generating sentences based on given attributes [64], writing stories based on limited clues [23]

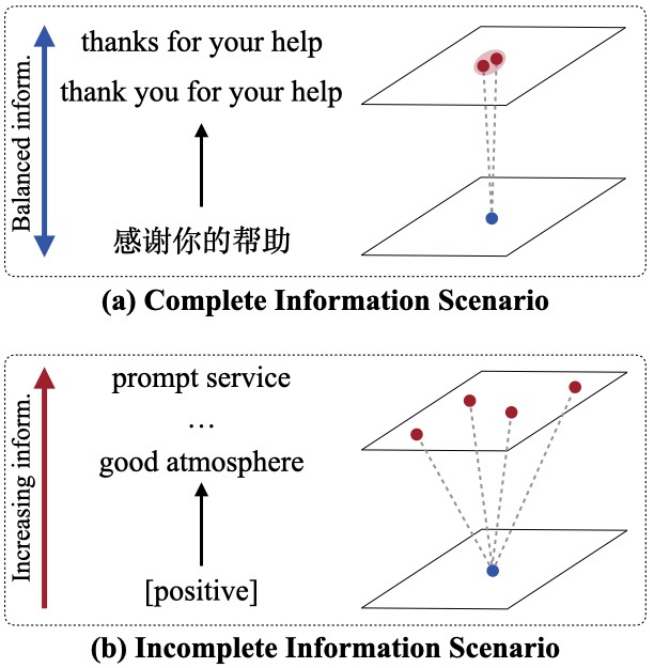

Figure 4.1: Comparisons between the CIS and IIS. (a) Translating a sentence. (b) Generating comments based on an emotion label.

and supporting semi-supervised learning [130]. Models for these tasks need to obtain long texts or even infer new data during training process. The decoding latency will be augmented, thereby significantly increasing the computation cost. Extending NAR models (which have lower decoding latency) to the IIS is therefore a promising line of further developments for these tasks.

To this end, we begin with a thorough analysis of existing NAR models which are trained on Maximum Likelihood Estimation (MLE). Due to the lost of word dependencies, these models tend to mix words in different candidates. This problem, which is known as the multi-modality problem [37], will be exacerbated in the IIS. As shown in Figure 4.1, in the CIS, the number of possible candidates and their diversity are significantly constrained by the input so as to maintain balanced input and output information. In the IIS, however, because of the incomplete input information, addition information is needed to obtain the output. Complementing different information will

lead to completely different results, so the candidate number and the output diversity will significantly increase. Models without word dependencies will easily mix words in different candidates and generate ungrammatical sentences.

In contrast, the synthetic distributions of GANs can theoretically converge to the real distributions regardless of model structures, so they are free from the multi-modality problem. More importantly, it can obtain high quality samples in one single forward pass, which exactly meets the needs of NAR models. Instead of adopting unstable *REINFORCE* [125] or biased *continuous relaxations* [54] to process the non-differentiable sampling operation in language GANs, we follow the research line of representation modeling methods [108] and propose an **Adversarial Non-autoregressive Transformer (ANT)** for the IIS. There are two features in ANT: Position-Aware Self-modulation for obtaining more reasonable hidden representations; 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 gets comparable performance as existing AR models in the IIS but achieves lower decoding latency. The contributions of this work can be summarized as follows:

- We articulate the significance of the IIS in text generation, and compare the differences between the CIS and IIS. Moreover, we reveal the limitations of existing (MLE-based) NAR models in the IIS.
- Based on GANs, we propose an Adversarial Non-autoregressive Transformer (ANT), which supports two features, Position-Aware Self-modulation and Dependency FFN, to further improve its performance by providing more reasonable representations and enhance its capacity in dependency modeling.
- We compare the performance of ANT with existing models. The experimental results demonstrate that ANT can get comparable performance as other models with much lower decoding latency. We also show the great potential of ANT

in various applications like latent interpolation and data augmentation. To the best of our knowledge, it is the first work demonstrating the effectiveness of GANs in building NAR models.

4.2 Background

To narrow down the gaps between NAR models and AR models, researchers improve performance from different perspectives including simplifying data distributions [37], adopting new training objectives [22], and designing new modeling methods [32].

Among them, there are two most popular techniques: 1) simplifying output with knowledge distillation [65]; 2) enhancing input based on conditional masked language models [32]. Knowledge distillation simplifies the original one-to-many mapping relations into one-to-one relations, so to prevent models from mixing words in different candidates. However, the IIS requires the model to obtain a number of diverse results with same conditions, so it is necessary to directly model the original one-to-many relations. Conditional masked language models obtain results following an iteration manner, so it will inevitably have higher latency. Although these two techniques are so popular that most of exiting NAR models are deeply bound to at least one of them, they are not applicable methods if we want to build a fully NAR model in the IIS.

Recently, two NAR models, which do not rely on the two techniques above, are proposed: Diffusion-lm [77] and Directed Acyclic Transformer (DA-Transformer) [50]. Diffusion-lm draws the idea of diffusion models [45, 20] into text generation. It starts with Gaussian noise vectors and gradually denoises them into specific word representations. Although it is effective on various tasks, it inherits the high latency feature of diffusion models. During inference, it requires 200-2000 steps to obtain the results, which has much higher decoding latency than AR models [77].

DA-Transformer [50] remedies the multi-modality problem by assigning various can-

didates to different paths in a directed acyclic graph. Although it obtains remarkable performance in machine translation, its theoretical convergence has not been proved yet. In our experiments, it fails to converge when migrating to the IIS. The key to its successful training is the peak path distributions led by the limited number of candidates. This kind of path distributions can avoid the model changing the candidates of other paths when updating the sample in one path [50]. In the IIS, each input corresponds to a lot of diverse candidates, and each vertex thus has many possible next vertices. The model needs to assign comparable transition probabilities to all these vertices and finally leads to a flat path distribution. In this time, the candidates in other paths will be inevitably changed when updating the sample in one path, and the model will finally fail to learn different candidates. Therefore, although there are various NAR models in the CIS, they are not reasonable choices for building fully NAR models in the IIS.

4.3 Model

4.3.1 Model Structure

Based on the representation modeling framework [108], we propose an Adversarial Non-autoregressive Transformer (ANT) which generates text in a fully NAR manner. As shown in Figure 4.2, there are three parts in ANT: mapper, discriminator and generator. The mapper transforms words into representations, and the generator tries to recover these representations from input. The discriminator needs to identify whether input representations are from the mapper or the generator. We adopt Transformer [120] as the backbones of all the three parts to support highly parallel computation. 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.

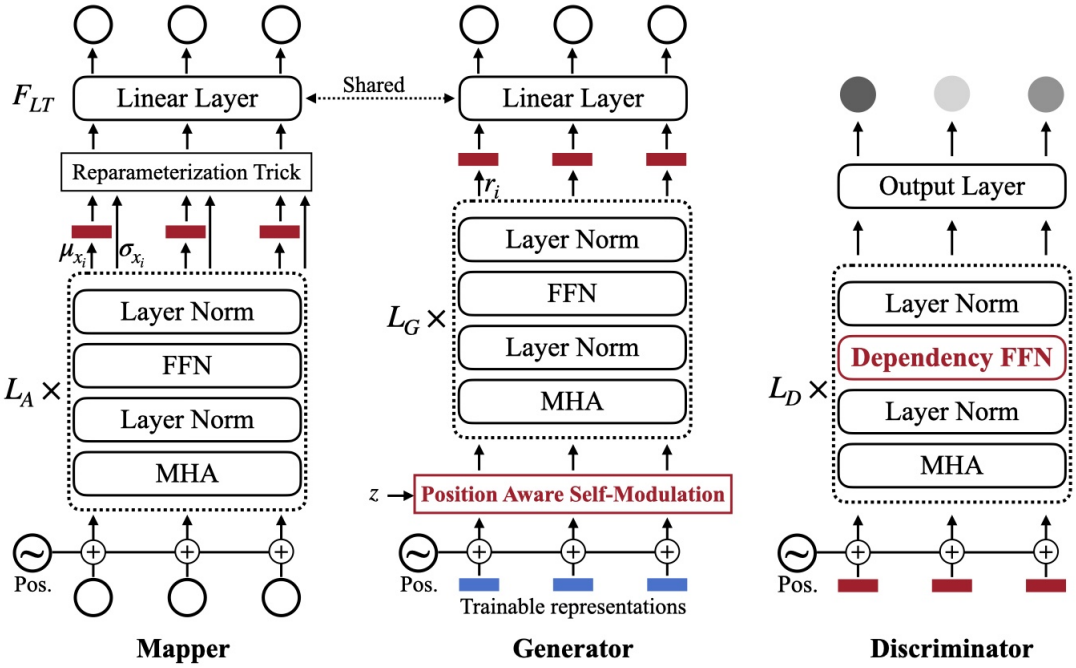

Figure 4.2: Structure of Adversarial Non-autoregressive Transformer (ANT)

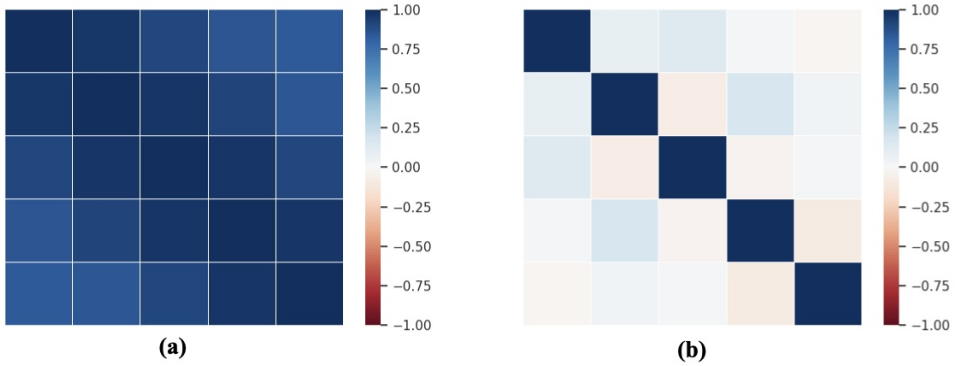

Figure 4.3: Cosine similarity of the output from (a) Self-Modulation; and (b) Position-Aware Self-Modulation.

The mapper is trained to reconstruct words based on the masked input, which is the same as the training process of BERT [19]. Following the previous work which adopts representation modeling methods to train language GANs [108], we use the loss function of variational autoencoder (VAE) [67] to train the mapper:

$$L_A = -\mathbb{E}_{z'_i \sim q(z'_i|x_i)}(logp(x_i|z'_i)) + KL(q(z'_i|x_i)||p(z'_i)) \tag{4.1}$$

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.

A non-autoregressive generator can not adopt previously generated words as the input, so we use trainable representations as input and incorporate latent variables into the representations. 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. The discriminator adopts the output representations from the mapper and the generator (μ_{x_i} and r_i) as input, and identifies whether they are synthetic or not. 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 [5] as the training objective and adopt Lipschitz penalty [101] to regularize the discriminator. More

specifically, the loss functions of the discriminator L_D and the generator L_G are:

$$L_D = -\mathbb{E}_{\mu_{x_i} \sim P_x}[D(\mu_{x_i})] + \mathbb{E}_{z \sim P_z}[D(G(z))] \tag{4.2}$$

$$L_G = -\mathbb{E}_{z \sim P_z}[D(G(z))] \tag{4.3}$$

where μ_{x_i} is obtained by the mapper and z is the latent variable sampled from a pre-defined distribution.

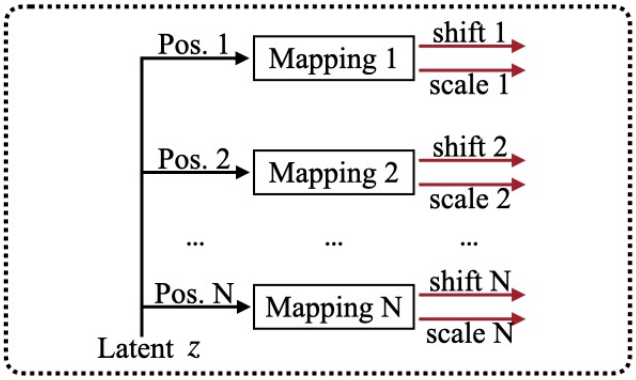

Figure 4.4: Position-Aware Self-Modulation

However, there is still a gap between our basic model and existing autoregressive models. To improve its performance, we propose Position-Aware Self-Modulation and Dependency Feed Forward Network (Dependency FFN).

4.3.2 Position-Aware Self-Modulation

An effective sampling method plays a key role in the success of GANs. Recent studies about Transformer in image GANs [73] adopt self-modulation [15] to sample data. 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 4.3 (a)). However, the output of the generator (i.e., word representations in different positions) are of high diversities. Similar input representations can not provide clear signals to the generator

to obtain those diverse output representations. It brings difficulties for the generator to converge and recover the data distribution.

To tackle this problem, we propose **Position-Aware Self-Modulation**. As shown in Figure 4.4, 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:

$$\begin{pmatrix} \mathbf{h}'_1 \\ \mathbf{h}'_2 \\ \vdots \\ \mathbf{h}'_N \end{pmatrix} = MLP(z)$$
$$\mathbf{h}_i = \gamma(\mathbf{h}'_i) \circ LN(\mathbf{x}_i) + \beta(\mathbf{h}'_i) \tag{4.4}$$

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 [42], $LN(\cdot)$ 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 4.3 (b)), so as to provide more effective signals to recover original sentences.

Our preliminary experiment shows that representations obtained by Position-Aware Self-Modulation are of high diversities, while equipping it in every layer will slow down the convergence. Thus, we adopt it only in the input and use dropout as an additional sampling method by using it in both training and inference stage [108]. It can improve model performance by injecting slight random noise and regularizing the model at the same time.

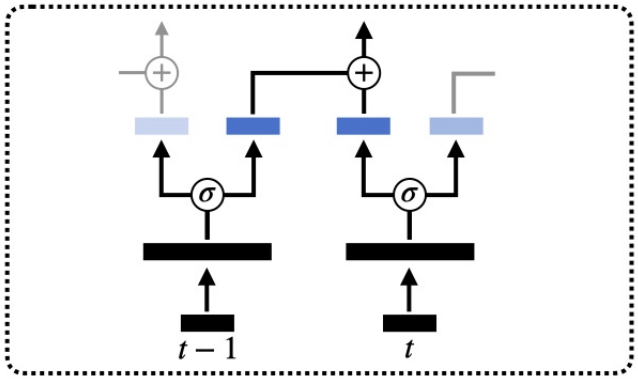

Figure 4.5: Dependency Feed Forward Network

4.3.3 Dependency Feed Forward Network

Transformer [120] builds word dependencies solely based on the attention mechanism by assigning weights dynamically. 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. To tackle this problem, we propose 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 Figure 4.5, and it is calculated as follows:

$$\begin{aligned} \mathbf{s}_t &= \sigma(\mathbf{x}_t W_s + b_s) \\ \mathbf{o}_t &= \mathbf{s}_{t-1} W_a + \mathbf{s}_t W_b + b_o \end{aligned} \tag{4.5}$$

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.

In our experiment, adopting Dependency FFN in the discriminator can significantly improve model performance, while its effectiveness to the generator is limited. A powerful discriminator can guide the generator to model word dependencies with the

Table 4.1: 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
V-CMLM	O(k)	0.016	27.65	0.062	**16.67**
V-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

original structure. We thus only adopt Dependency FFN in the discriminator.

4.3.4 Extension to Conditional Generation

IIS not only covers unconditional generation tasks, but also includes conditional generation tasks. In the following, we introduce how to 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:

$$\hat{z} = \alpha_1 \circ z + \alpha_2 \circ c \tag{4.6}$$

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

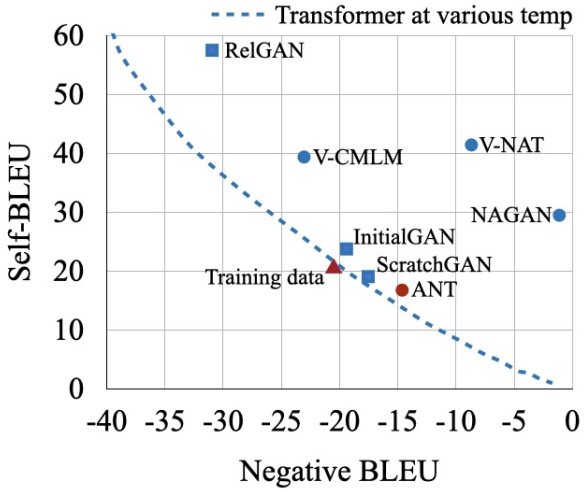

Figure 4.6: Model Performance at Various Temperature

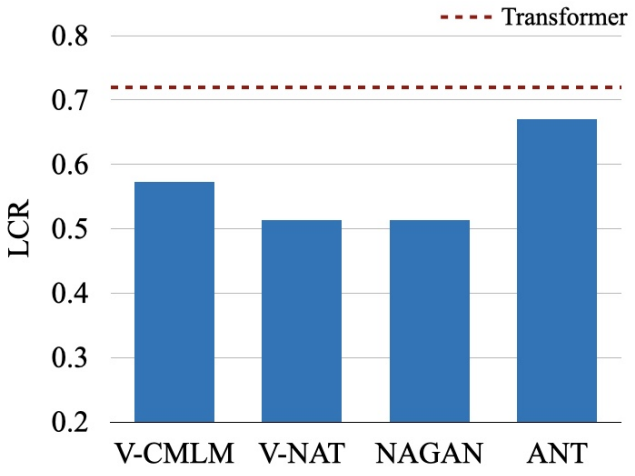

Figure 4.7: Least Coverage Rate

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.

4.4 Experiment

4.4.1 Experiment Setup

The experiment covers both unconditional generation and conditional generation to evaluate model performance under the IIS comprehensively. For the unconditional generation, we follow previous work [18, 108] and use sentences from two datasets: the COCO Image Caption Dataset [83][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. The training process of ANT is similar with the training algorithm of InitialGAN (as describe in Algorithm 1).

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 [86] 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 [86] 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 [66] as the optimizer

[1] `https://cocodataset.org`

[2] `http://www.statmt.org/wmt17/`

[3] `https://www.yelp.com/dataset`

Table 4.2: 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%
V-CMLM	O(k)	**0.015**	18.35	87.85%
V-NAT	O(1)	0.032	11.81	83.54%
ANT	O(1)	0.018	**19.08**	**88.35%**

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[4] [1] and the model is trained on NVIDIA GeForce RTX 3090.

4.4.2 Evaluation Metrics

The evaluation is conducted at both embedding level and token level. In embedding level, we use Universal Sentence Encoder[5] [12] to transform sentences into embeddings. Then, we calculate both **Fréchet Embedding Distance (FED)** [18] and **Least Coverage Rate (LCR)** [108] 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. The Inverse BLEU uses generated sentences as references and sentences in the test set as inferences. A model can get high Inverse BLEU only if it can obtain a good trade-off between quality and diversity [108]. We also draw a curve of **BLEU** [98] and **Self-BLEU** [144] by tuning the temperature

[4] `https://www.tensorflow.org`

[5] `https://tfhub.dev/google/universal-sentence-encoder/4`

of the model to evaluate the overall performance [11]. For conditional generation, we additionally use Universal Sentence Encoder to construct a classifier and use it to calculate **Accuracy (Acc.)**, so as to verify whether the models generate sentences with expected labels or not.

4.4.3 Compared Model

Transformer [120] is an important compared model in our experiment, since it is now the mainstream model in various text generation tasks. Considering ANT is a GAN-based model, we also choose several representative models in language GANs for comparisons: RelGAN [95], which uses *Gumbel-softmax* to obtain gradients; ScratchGAN [18], which is based on *REINFORCE*; InitialGAN [108], 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).

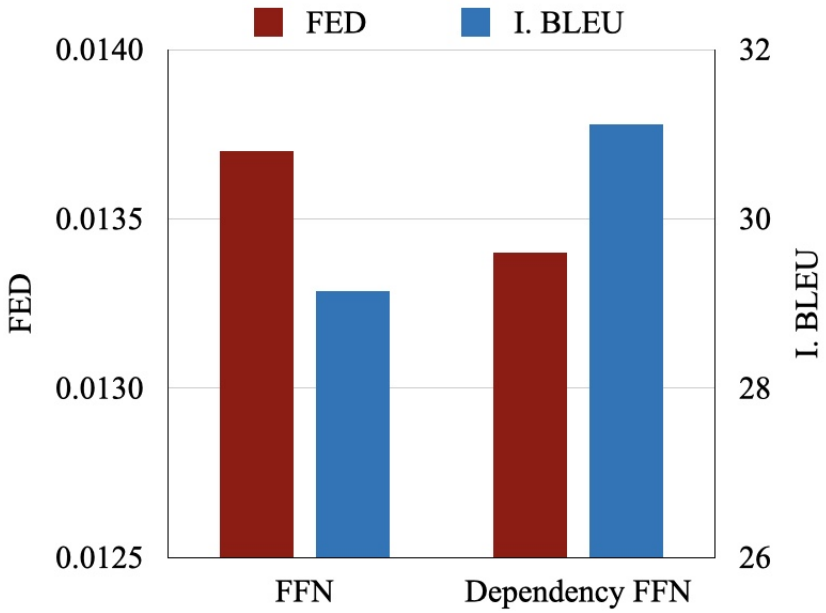

Figure 4.8: Ablation study of Dependency FFN

Figure 4.9: Ablation study of Position-Aware Self-Modulation.

For the NAR models, we compare with NAGAN [48], which is also a GAN-based NAR model. Although existing MLE-based NAR models can not be adopted to the IIS directly, we still want to compare the differences between MLE-based methods and GAN-based methods. Thus, we use the idea of VAE to get the hidden representations of sentences and adapt two existing popular NAR models into the IIS. One is based on Non-autoregressive Transformer (NAT) [37] which uses the hidden representations to generate sentences in a fully NAR manner; we denote this model as V-NAT. Another one in based on conditional masked language model (CMLM) [32] which generates sentences by iteratively masking and generating words; we denote this model as V-CMLM.

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. 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. The original structure of NAT and CMLM are kept as much as possible. For V-NAT the representations are fed into decoder as input (which is same with NAT). For V-CMLM, the representations are concatenated with the embeddings of input tokens (masked or unmasked words). The hyperparameters of V-NAT and V-CMLM are set as close as possible to those of ANT. The iteration number k of V-CMLM is set to be 10 as in previous work [32, 51].

4.4.4 Experimental Result

Unconditional Generation

The experimental results of the unconditional generation are shown in Table 4.1. "DI" indicates Decoding Iteration indicating the steps that require for generation. For the COCO dataset, Transformer gets the best performance in AR models, while ANT is the best one in NAR models. ANT gets 0.013 in FED, which is better than other NAR models and close to AR models like ScratchGAN and InitialGAN. Similar results can be found in Inverse BLEU. ANT gets 31.12 in I. BLEU and it is much better than other NAR models. There are large gaps between MLE-based NAR models (V-NAT and V-CMLM) and AR models, and it is consistent with our analyses. The input with incomplete information increases the difficulties of MLE training. NAGAN, another GAN-based NAR model, is inferior to all the other models. It shows the limitations of the biased *straight-through estimator*.

For the EMNLP dataset, Transformer is still the best model. ANT outperforms other NAR models in FED, while V-CMLM can slightly outperform ANT in Inverse BLEU. The iterative decoding mechanism helps V-CMLM to better process complicated datasets with higher decoding latency. To further discuss their performance in the token level, we draw the curve of Self-BLEU and Negative BLEU by tuning the temperature in Transformer and show the results in Figure 4.6. ANT is the only

NAR model which can get comparable performance with AR models, while other NAR models (including V-CMLM) remain behind obviously. Specifically, NAGAN gets extremely low BLEU, which indicates that NAGAN can not generate fluency sentences. It 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 4.7. ANT outperforms V-CMLM with lower decoding iterations, and it is the only NAR model which can get close performance with Transformer.

Conditional Generation

The experimental results of conditional generation are shown in Table 4.2. Transformer gets the best performance in all the evaluation metrics with more decoding iterations. Among NAR models, ANT gets comparable performance with V-CMLM in FED, and achieves higher Inverse BLEU and Accuracy with lower decoding latency. V-NAT, which has the same decoding iterations as ANT, is inferior to other models. Given an emotion label, there are a large number of possible candidates, and it will augment the inherent multi-modality problem in MLE-based NAR models. Thus, fully NAR models trained under MLE will easily mix words in different candidates and finally obtains poor performance. For the accuracy, ANT gets 88.35% which is the highest one among all the NAR models. ANT can generate sentences that are consistent with the given labels. It reveals that ANT has great potential in both unconditional generation and conditional generation.

Ablation Study

We design two features for ANT: 1) Dependency FFN, which strengthens model capacity in dependency modeling; and 2) Position-Aware Self-Modulation, which provides clearer signals for the model to recover the original data. We also conduct experiments to demonstrate the effectiveness of these two modules.

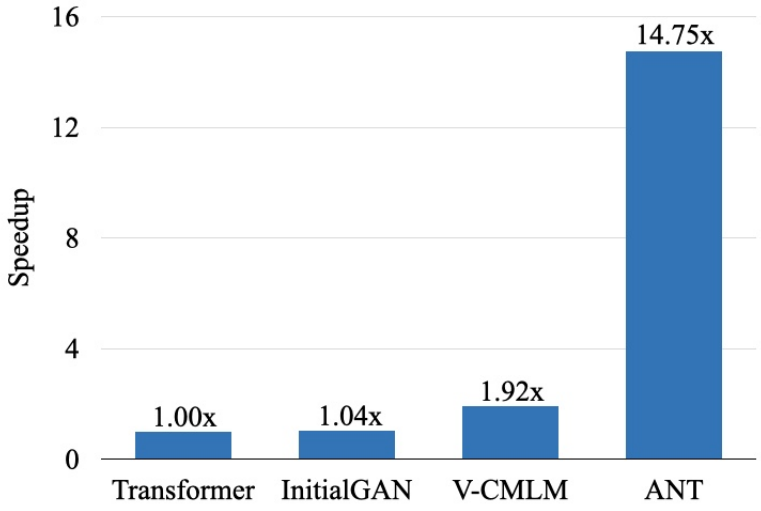

Figure 4.10: Speedup of Different Models.

Table 4.3: Effectiveness of ANT in Data Augmentation (Num.: number of labeled data).

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

For Dependency FFN, we compare the performance between Dependency FFN and the original FFN on the COCO dataset and show the results in Figure 4.8. ANT with Dependency FFN has lower FED and higher Inverse BLEU. It obtains better performance in both the token-level metrics and the embedding-level metrics. These results show that Dependency FFN can help improve model performance by modeling more accurate word dependencies.

For Position-Aware Self-Modulation, we compare the training curves with original Self-Modulation with FED. The results are shown in Figure 4.9. ANT with Position-

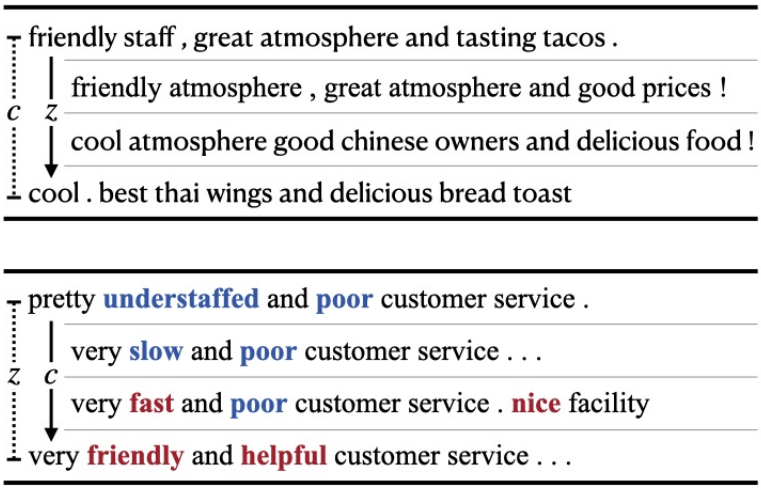

Figure 4.11: Case Study of Latent Interpolation

Aware Self-Modulation converges faster, and finally achieves better performance. For the Self-Modulation, it is rebounded around epoch 1000-1500. We assume that the discriminator has the overfitting problem and the performance of the generator thus decreases. Position-Aware Self-Modulation can provide clearer input signals for the model and further enhance model performance.

4.4.5 Discussion

ANT is a fully NAR model so it has much lower decoding latency. We compare the speedup of different models and show the results in Figure 4.10. ANT can generate sentences 14.75 times faster than Transformer. Even comparing with V-CMLM, it also has much lower decoding latency while obtaining comparable or even better performance.

Generative models can perform data augmentation to boost the performance of classification models. We investigate the application of ANT in data augmentation by incorporating it into the training of a classification model. The classification model is trained to identify emotion labels of 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 4.3. The classification models based on data augmentation consistently outperform the original ones. ANT can obtain data following same distributions as the original data so as to help the classification model capture more accurate data distribution and achieve better performance.

Besides, ANT enables latent interpolation just like image GANs. 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 upper part of Figure 4.11 shows the samples given by tuning z with fixed c, in which ANT transforms one sentence into another one, with the middle sentences kept understandable. The lower part of Figure 4.11 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 latent interpolation is seldomly explored by NAR models, and it may inspire further ideas for related tasks.

4.5 Summary

In this work, we firstly analyze the limitations of existing NAR models in the IIS. The features of the IIS will augment the inherent multi-modality problems of MLE-based NAR models and the lower bounds between their learned distributions and real distributions will be higher under the IIS. Instead, we find GANs denote a more promising method to train NAR models in the IIS. Thus, we 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 can get comparable performance as other AR models but with much lower decoding latency. Besides, we also demonstrate the great potential of ANT in various tasks like smooth latent interpolation and data augmentation.

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 the performance of ANT and extend it into other tasks.

Chapter 5

GAN-based NAR models for Complete Information Scenarios

5.1 Introduction

In addition to IIS, there is another kind of important scenario: CIS. The input in CIS contain complete information of the output. Although the candidates in CIS are less diverse, they may have complicated mapping relations. In this section, we study GAN-based NAR models in CIS based on a classical and challenge tasks: image captioning.

Comparing with the rapid development of NAR models in machine translation [128], its progress in image captioning is relative slow. Recent study directly enhances performance by significantly sacrificing decoding efficiency [26, 131]. Exiting work constructs NAR image captioning models based on Maximum Likelihood Estimation (MLE) [25, 29, 40], which meets obvious obstacles in their developments. First, MLE-based NAR models can learn the marginal distributions of different candidates, but lose word dependencies and remain non-negative lower bounds in the KL divergence between the learned distributions and real distributions [49]. Thus, these models

tend to generate ungrammatical sentences by mixing words in different candidates, which is known as the multi-modality problem [37]. Secondly, the difficulties in the alignment between images and text will cause greater errors in the learned marginal distributions, thereby exacerbating the multi-modality problem by mixing irrelevant candidates.

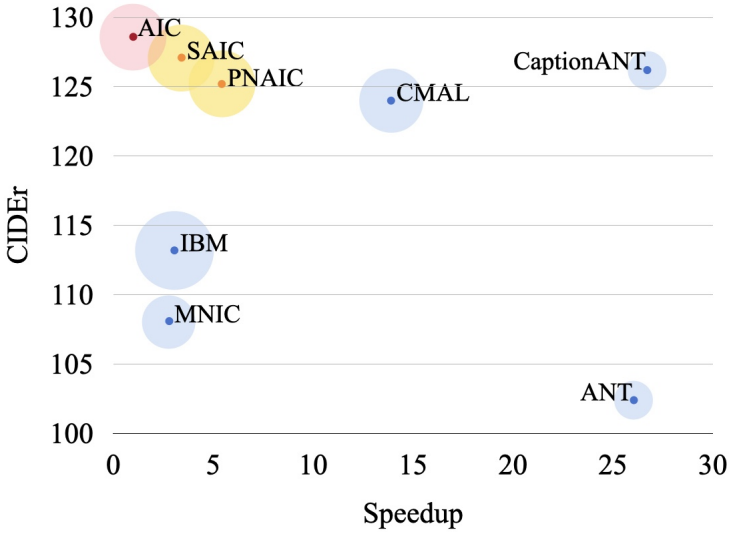

Figure 5.1: The Performance of Compared Models. The red, yellow and blue points indicate AR, SAR and NAR models, respectively. The area indicates the number of parameters.

Different with MLE, which is inherently incompatible with NAR models, Generative Adversarial Networks (GANs) [34] denote a more promising method. Their learned distributions can theoretically converge to the real distributions with one single forward pass [34]. It exactly fits the needs of NAR models. In text-to-image generation, GANs have been demonstrated to be an effective method. They can generate high quality images with much lower latency [113, 57]. However, their potentials in image-to-text generation have not been explored yet.

The main obstacle of adopting GANs in text generation comes from the non-differentiable sampling operation in the generator, which prevents the gradient of the discriminator

from being passed to the generator. Recently, a representation modeling method is introduced to tackle this problem by removing the sampling operation during training [108]. It is later extended to NAR models for incomplete information scenarios [107]. This model, which is denoted as Adversarial Non-autoregressive Transformer (ANT), obtains poor performance when transferred to image captioning (as shown in Figure 5.1). ANT is designed for the scenarios with relatively simple input conditions (e.g., class labels). In image captioning, however, the input images are highly diverse, and ANT becomes incapable of building complicated relations between images and text.

In this work, we release the capacity of GANs in image captioning by proposing an Adversarial Non-autoregressive Transformer for Image Captioning (CaptionANT). To enable the model to build more complicated relations, the discriminator structure in the previous work [107] is modified to be compatible with contrastive learning, so CaptionANT can better align images and text by effectively making use of unpaired samples. In addition, we integrate a reconstruction process to further boost model performance by better making use of paired samples. During the reconstruction process, the key challenge comes from the ambiguous reconstruction target led by the one-to-many mapping relations in image captioning. We tackle this problem by integrating part of target sentences into the input so as to have clearer reconstruction targets. By further combining with other effective techniques (like feature ensemble and the truncation trick) and our proposed lightweight structure, CaptionANT achieves new state-of-the-art performance for fully NAR models on the challenging MSCOCO dataset with much higher speedup and lower parameter number (as shown in Figure 5.1). The contributions of this work can be summarized as follows:

- Considering the limitations of MLE-based NAR image captioning models, we propose a GAN-based NAR model—CaptionANT. We redesign the model structure and incorporate contrastive learning in CaptionANT. It thus can effectively make use of unpaired samples to model complicated relations between images

and text. To the best of our knowledge, CaptionANT is the first GAN-based NAR model in image captioning.

- We further propose to incorporate a reconstruction process into the training stage of language GANs based on representation modeling methods. It can further improve model performance by better utilizing paired samples. For the ambiguous reconstruction targets led by the one-to-many mapping relations, we propose to integrate part of target information into the input so to have clear reconstruction targets.

- By further combining with other effective techniques (like feature ensemble and the truncation trick) and our proposed lightweight structure, CaptionANT achieves new state-of-the-art performance for fully NAR models with lower parameter number and faster speed.

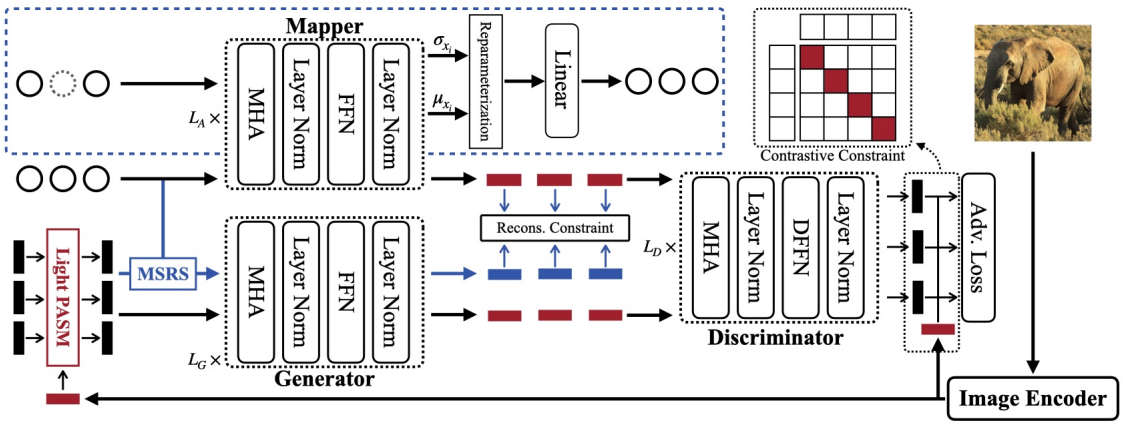

Figure 5.2: General Structure of CaptionANT.

5.2 Model

To allow the gradients from the discriminator to be passed to the generator directly, we adopt the representation modeling method [108], which can avoid the non-differentiable sampling operation during training. More specifically, we first adopt a

model, which is denoted as Mapper in this work, to map words into representations. Then, the generator is trained to recover these representations under the guidance of the discriminator. Both the representations from the mapper and the generator are fed into the discriminator as input, and the discriminator needs to identify whether the input is from the mapper or not.

The general structure of CaptionANT is shown in Figure 5.2. As described above, there are three different models in CaptionANT: Mapper, Discriminator and Generator. All these three models adopt Transformer [120] as backbones to support highly parallel computation.

5.2.1 Mapper

The mapper needs to map words into representations. It is trained before the generator and the discriminator. The training process of the mapper is described in the blue dashed box of Figure 5.2. A certain number of words in a sentence are randomly masked or replaced, and the mapper is trained to reconstruct the original input. We follow the settings in the previous work [107], and incorporate the idea of variational auto-encoder (VAE) [67] into the training objective. More specifically, after obtaining the mean μ_{x_i} and standard deviation σ_{x_i} for each word x_i, the mapper first adopt reparameterization trick to obtain hidden representations $\mathbf{z}'_i = \mu_{x_i} + \sigma_{x_i} \cdot \mathcal{N}(0, 1)$, and then use the following objective to train the model:

$$L_A = -\mathbb{E}_{\mathbf{z}'_i \sim q(\mathbf{z}'_i|x_i)}(logp(x_i|\mathbf{z}'_i)) + KL(q(\mathbf{z}'_i|x_i)||p(\mathbf{z}'_i)) \tag{5.1}$$

where $\mathbf{z}'_i$ is transformed into words by a linear layer F_{LT}. The vector μ_{x_i} will be regarded as the representation of x_i and fed into the discriminator.

This training objective provides a dense and continuous representation space, so representations slightly away from the central point can still be mapped into correct words [107].

5.2.2 Discriminator

Structure

The discriminator is consisted of a stack of Transformer blocks. Different with our previous work [107], we do not observe improvements from the look-ahead mask, so we remove it and the input in different positions can consider each other directly.

One key challenge in building the discriminator is how to incorporate conditions into the model. The previous work [107] feeds the condition representation as input. This structure is also adopted in other GAN-based text-to-image generation models [106, 136, 74]. However, it only considers one pair of mismatched samples at a time and can not effectively make use of them to build more accurate alignments between images and text.

To tackle this problem, we separate the condition representation from the input. Instead, we map input text to the same space as the condition representation, so it can measure the correlation between the two by calculating dot product. This modeling method can efficiently utilize unpaired samples through contrastive learning. The detailed calculation of this modeling methods is described as follows:

$$\begin{aligned}
\hat{\mathbf{h}}_i^{(l)} &= LN(MHA(\mathbf{h}_i^{(l-1)}) + \mathbf{h}_i^{(l-1)}) \\
\mathbf{h}_i^{(l)} &= LN(DFFN(\hat{\mathbf{h}}_i^{(l)}) + \hat{\mathbf{h}}_i^{(l)}) \\
\tilde{\mathbf{h}}_\mathbf{i} &= W_h \mathbf{h}_i^{(L_D)} + b_h \\
\mathbf{y}_i &= \tilde{\mathbf{h}}_\mathbf{i} \cdot \hat{\mathbf{c}}^\intercal
\end{aligned} \tag{5.2}$$

where $MHA(\cdot)$ is the multi-head attention mechanism (here is the self-attention, where query, key and value are the same), $\mathbf{h}_i^{(l)}$ and $\hat{\mathbf{h}}_i^{(l)}$ are the i-th hidden representations of the the l-th layer, $LN(\cdot)$ is layer normalization, $\hat{\mathbf{c}}$ is the normalized image representation ($\hat{\mathbf{c}} = \mathbf{c}/||\mathbf{c}||$, and $\mathbf{c}$ is the image representation provided by the image encoder) and L_D is the layer number. $DFFN(\cdot)$ is the dependency feed forward

network [107], which is calculated as follows:

$$\begin{aligned} \hat{\mathbf{g}}_i &= GELU(\hat{\mathbf{h}}_i W_g + b_g) \\ \mathbf{g}_i &= \hat{\mathbf{g}}_{i-1} W_l + \hat{\mathbf{g}}_i W_k + b_o \end{aligned} \tag{5.3}$$

where $\hat{\mathbf{h}}_i$ is the input of DFFN, and $\mathbf{g}_i$ is the output and will be incorporated into the calculation of Eq 5.2. DFFN directly models the relations between $\hat{\mathbf{g}}_{i-1}$ and $\hat{\mathbf{g}}_i$, so it can strengthen the dependency modeling capacity of the discriminator in the unstable training of GANs [107].

Training Objective

We follow the previous work [108], and adopt Wasserstein distance [5] as the training objective:

$$L_{AdvD} = -\mathbb{E}_{x \sim P_x}[D(M(x), \mathbf{c})] + \mathbb{E}_{\mathbf{z} \sim P_\mathbf{z}}[D(G(\mathbf{z}), \mathbf{c})] \tag{5.4}$$

where $M(\cdot)$ is the mapper, $\mathbf{c}$ is the condition representation obtained by the image encoder, $D(\cdot)$ and $G(\cdot)$ are the discriminator and the generator, respectively. We adopt Lipschitz penalty [101] to stabilize the training process.

Contrastive Constraint To fully make use of the advantages of our discriminator structure, we further integrate a contrastive constraint into the training objective to regularize the model by considering unpaired samples effectively. We first obtain the representation of the k-th sentence $\mathbf{H}_k$ by calculating the mean of $\tilde{\mathbf{h}}_\mathbf{i}$ in different timesteps. Then, the contrastive constraint is calculated as follows:

$$C_d = -\tau \frac{exp(\mathbf{H}_k \cdot \hat{\mathbf{c}}^\intercal / \tau)}{\sum_{j=1} exp(\mathbf{H}_j \cdot \hat{\mathbf{c}}^\intercal / \tau)} \tag{5.5}$$

where $\hat{\mathbf{c}}$ is the normalized condition representation. We obtain the negative samples from two different sources: 1) the real but mismatched sentences in the same batch; 2) the synthetic sentences given by the generator with the same batch of condition representations. The real but mismatched sentences can help the model quickly regularize

its representations in the early training, while the synthetic sentences can further boost model performance when the generator begins to generate real-like sentences.

Incorporating the contrastive constraint, the complete training objective of the discriminator is:

$$L_D = L_{AdvD} + \lambda_d \cdot C_d \tag{5.6}$$

where λ_d is a hyper-parameter which can adjust the importance of the contrastive constraint.

5.2.3 Generator

Structure

The generator is constructed based on Transformer [120]. The input is a trainable matrix. The vectors obtained by the final Transformer block will be the word representations after a linear transformation. During training, these representations (denoted as r_i) will be fed into the discriminator, and the discriminator will guide the generator to obtain the representations following same distributions with μ_{x_i} from the mapper. During inference, r_i will be transformed back into words with the same linear layer F_{LT} of the mapper.

Feature Ensemble An effective method to incorporate latent vectors plays a key role in the performance of the generator. Previous work [107, 73] calculates shift and scale vectors for the normalized input based on latent vectors. We further enhance model performance by adopting feature ensemble which can provide images features

from two representation spaces. It is described as follows:

$$\begin{pmatrix} \mathbf{s}'_1 \\ \mathbf{s}'_2 \\ \vdots \\ \mathbf{s}'_N \end{pmatrix} = \mathbf{F}^{\mathbf{1}}_{\mathbf{M}}(\mathbf{z}_1) + \mathbf{F}^{\mathbf{2}}_{\mathbf{M}}(\mathbf{z}_2) \tag{5.7}$$

$$\mathbf{s}_i = \gamma(\mathbf{s}'_i) \circ LN(X^g_i) + \beta(\mathbf{s}'_i)$$

where X^g_i is the trainable input matrix, $\gamma(\cdot)$ and $\beta(\cdot)$ are linear layers after GELU, and $\mathbf{s}_i$ will be fed into a set of Transformer blocks as input. $\mathbf{z}_1$ and $\mathbf{z}_2$ are the concatenations of random noises and image features extracted by two different models. For the random noise, we adopt the **truncation trick** [10] which samples the noise in a truncated distribution during inference.

The transformation modules $\mathbf{F}^{\mathbf{1}}_{\mathbf{M}}(\cdot)$ and $\mathbf{F}^{\mathbf{2}}_{\mathbf{M}}(\cdot)$ are in a same structure but with independent parameters. The design of the transformation modules will directly influence the performance, and a detailed discussion is conducted in the following.

Light Position-Aware Self-Modulation Different methods adopt different transformation modules. Self-modulation [15] uses same layers at different positions, and the obtained representations at each position are thus too similar to recover the diverse word representations at different positions [107].Ren et al. [107] tackle this problem by proposing a Position-Aware Self-Modulation (PASM) which adopts unique layers at different positions to obtain diverse representations.

This method, however, has independent layers for each position. It causes a dramatic increase in the number of model parameters, which we find is not necessary. Instead, we propose and adopt a Light Position-Aware Self-Modulation (Light PASM). The

Table 5.1: Evaluation Results on the "Karpathy" Split of MSCOCO Dataset

Model	BLEU-1	BLEU-4	METEOR	ROUGE	SPICE	CIDEr	#Param.	Speedup
Autoregessive Models								
Up-Down [4]	79.8	36.3	27.7	56.9	21.4	120.1	-	-
M2-T [17]	80.8	39.1	29.2	58.6	22.6	131.2	-	-
$\mathcal{A}^2$-Transformer [27]	81.5	39.8	29.6	59.1	23.0	133.9	-	-
AIC (bw=1)	80.3	38.9	28.7	58.5	22.4	127.1	54.9M	1.22×
AIC (bw=3)	80.4	39.2	28.8	58.6	22.5	128.6		1.00×
Semi-Autoregessive Models								
PNAIC [26]	79.9	37.5	28.2	58.0	21.8	125.2	54.9M	5.43×
SAIC [131]	80.3	38.4	29.0	58.1	21.9	127.1		3.42×
Non-Autoregessive Models								
MNIC [29]	75.4	30.9	27.5	55.6	21.0	108.1	36.0M	2.80×
IBM [25]	77.2	36.6	27.8	56.2	20.9	113.2	77.0M	3.06×
NAIC [40]	80.3	37.3	28.1	58.0	21.8	124.0	50.1M	13.90×
CaptionANT	**80.8**	**38.0**	**28.7**	**58.7**	**22.5**	**126.2**	18.2M	**26.72**×

transformation module ($\mathbf{F_M^1}$ and $\mathbf{F_M^2}$ in Eq. 5.7) in our proposed model is:

$$\begin{pmatrix} \hat{\mathbf{s}}_1 \\ \hat{\mathbf{s}}_2 \\ \vdots \\ \hat{\mathbf{s}}_{\frac{N}{2}} \end{pmatrix} = \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_{\frac{N}{2}} \end{pmatrix} \cdot \hat{\mathbf{z}} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{\frac{N}{2}} \end{pmatrix} \tag{5.8}$$

$$\begin{pmatrix} \hat{\mathbf{s}}_{\frac{N}{2}+1} \\ \hat{\mathbf{s}}_{\frac{N}{2}+2} \\ \vdots \\ \hat{\mathbf{s}}_N \end{pmatrix} = W' \cdot \begin{pmatrix} \hat{\mathbf{s}}_1 \\ \hat{\mathbf{s}}_2 \\ \vdots \\ \hat{\mathbf{s}}_{\frac{N}{2}} \end{pmatrix} + b' \tag{5.9}$$

where $\hat{\mathbf{z}}$ is the $\mathbf{z}_1$ or $\mathbf{z}_2$ in Eq. 5.7. Light PASM first obtains the hidden representations of the previous half position with unique linear layers. Then, another linear layer is adopted to get the remaining half of the representations. This method can maintain the diversity of representations between different positions while significantly reducing the parameter number.

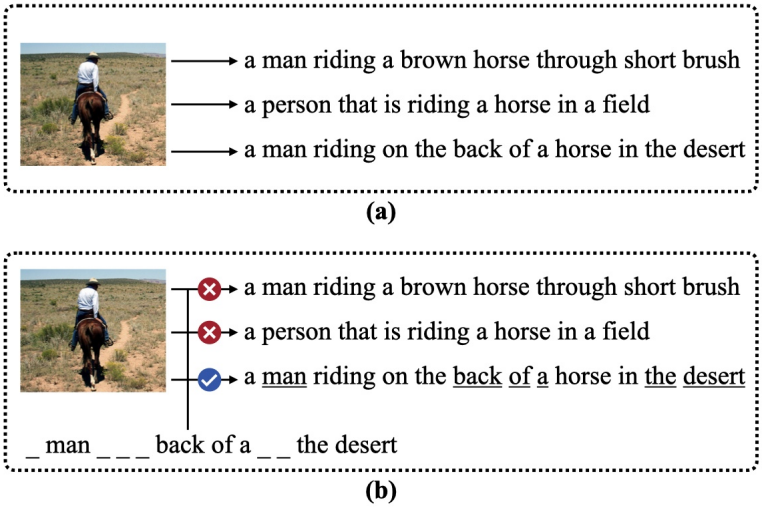

Figure 5.3: Effectiveness of Masked Sentence Representation Shift (MSRS).

Different with existing NAR image captioning models [40, 26] which first use an encoder to process image features and then generate sentences with a decoder, the generator in CaptionANT directly transforms image features into sentences, so it has a lighter and more efficient structure.

Training Objective

Corresponding to the discriminator, the adversarial training objective of the generator is:

$$L_{AdvG} = -\mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}}[D(G(\mathbf{z}), \mathbf{c})] \tag{5.10}$$

In addition, we also adopt the following constraints to boost its performance.

Contrastive Constraint Similar to the discriminator, we adopt a contrastive constraint to better align input images and output text.

$$C_g = -\tau \frac{exp(\mathbf{H}'_k \cdot \hat{\mathbf{c}}^{\intercal}/\tau)}{\sum_{j=1} exp(\mathbf{H}'_j \cdot \hat{\mathbf{c}}^{\intercal}/\tau)} \tag{5.11}$$

where $\mathbf{H}'_k$ is the mean of $\tilde{\mathbf{h}}_\mathbf{i}$ from the discriminator in different timesteps, and the negative samples are the captions generated based on the unpaired conditions in the same batch.

Reconstruction Constraint Reconstruction constraint has been adopted to stabilize the training and enhance model performance in image GANs [142]. It provides a more effective way to utilize paired samples. However, how to incorporate reconstruction constraint in language GANs, which are based on the representation modeling method, has not been explored yet. The key challenge is from the diverse words in the same positions among different candidates. We give an example in Figure 5.3 (a). If the model is trained to fit all candidates together, it will try being close to the diverse word representations in different candidates and finally degenerate to learn mean values instead of specific representations.

We tackle this problem by proposing a **Masked Sentence Representation Shift (MSRS)**. When calculating the reconstruction constraint term, the input representations $\mathbf{s}_i$ obtained by Eq. 5.7 are added with shift vectors as follows:

$$\begin{aligned}
\mathbf{e}_i &= Emb(x_i) + pos_i \\
\hat{\mathbf{e}}_i &= Mask(\mathbf{e}_i, \rho) \\
\dot{\mathbf{s}}_i &= \omega \circ MHA(\mathbf{s}_i, \hat{\mathbf{e}}_i, \hat{\mathbf{e}}_i) \\
\hat{\mathbf{s}}_i &= \mathbf{s}_i + \dot{\mathbf{s}}_i
\end{aligned} \tag{5.12}$$

where x_i is the i-th word of the sentence, $Emb(\cdot)$ is an embedding layer, pos_i is the positional encoding for the i-th position, ρ is the mask rate, $MHA(query, key, value)$ is the multi-head attention, ω is a trainable vector which can directly control the scale of $\dot{\mathbf{s}}_i$. This process is shown in the blue path of Figure 5.2. It should be noted that $\hat{\mathbf{s}}_i$ is only used when calculating the reconstruction constraint, and the generator still adopts $\mathbf{s}_i$ as input when calculating the adversarial loss and generating captions in inference stage.

Table 5.2: Evaluation Results on the Online MSCOCO Test Server

Model	BLEU-1		BLEU-2		BLEU-3		BLEU-4		METEOR		ROUGE-L		CIDEr	
	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40
Up-Down [4]	80.2	95.2	64.1	88.8	49.1	79.4	36.9	68.5	27.6	36.7	57.1	72.4	117.9	120.5
M2-T [17]	81.6	96.0	66.4	90.8	51.8	82.7	39.7	72.8	29.4	39.0	59.2	74.8	129.3	132.1
$\mathcal{A}^2$-Transformer [27]	82.2	96.4	67.0	91.5	52.4	83.6	40.2	73.8	29.7	39.3	59.5	75.0	132.4	134.7
NAIC [40]	79.8	94.3	63.8	87.2	48.8	77.2	36.8	66.1	27.9	36.4	57.6	72.0	119.3	121.2
CaptionANT	**80.3**	**94.7**	**64.5**	**88.2**	**49.4**	**78.5**	**37.1**	**67.3**	**28.4**	**37.3**	**58.2**	**73.0**	**120.9**	**124.7**

The effectiveness of the MSRS is described in Figure 5.3 (b). By providing shift vectors $\dot{\mathbf{s}}_i$, MSRS incorporates unmasked words into the input representations. This approach reduces the number of possible candidates and transforms the mapping relations from input to output to a roughly one-to-one relation. Thus, the model can learn to reconstruct specific word representations instead of ambiguous ones. The reconstruction constraint is:

$$C_r = ||\mu_{x_i} - r'_i||^2 + \lambda_s||\dot{\mathbf{s}}_i||^2 \tag{5.13}$$

where μ_{x_i} is the representation of the word x_i obtained by the mapper, r'_i is the i-th word representation given by the generator, and λ_s is a hyper-parameter. The norm of $\dot{\mathbf{s}}_i$ is also minimized, so the shifted representation $\hat{\mathbf{s}}_i$ can be as close to the original input representation $\mathbf{s}_i$ as possible.

With the constraints above, the complete training objective of the generator is:

$$L_G = L_{AdvG} + \lambda_g \cdot C_g + \lambda_r \cdot C_r \tag{5.14}$$

where λ_g and λ_r are both hyper-parameters which can control the effects from the constraints.

5.3 Experiment

5.3.1 Experiment Setup

The MSCOCO dataset [16] is one of most popular dataset in image captioning. We adopt the widely used "Karpathy" splits [59] to conduct experiments. It contains 113,287 images for the training set, 5,000 images for the validation set and the test set, respectively.

5.3.2 Evaluation Metric

We adopt standard evaluation metrics to compare the performance of different models comprehensively: BLEU [98], METEOR [71], ROUGE-L [80], SPICE [3], CIDEr [121]. Besides, we also show the parameter numbers of different models and the speedup value. The speedup value of CaptionANT is calculated based on the average latency of generating 10,000 sentences.

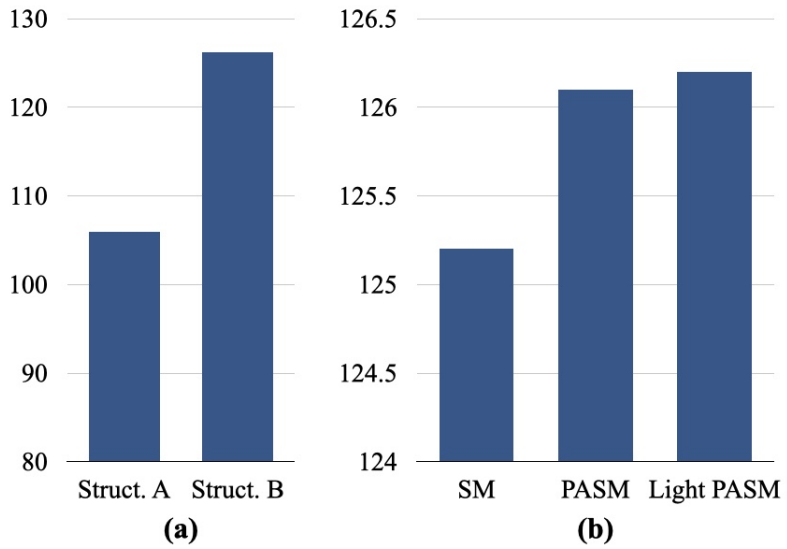

Figure 5.4: CIDEr Scores of Different Structures.

5.3.3 Implementation Details

The input size of the mapper and the generator is set to be 384, and the hidden size of the FFN is set to be 1,536, while the input size of the discriminator is 768 and the hidden size of DFFN is 3072. The head numbers are all set to be 8. They are all stacked with 4 blocks. We adopt AdamW as the optimizer of the mapper ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $weight\_decay = 1e-5$) and the discriminator ($\beta_1 = 0.5$, $\beta_2 = 0.9$, $weight\_decay = 1e-4$), and Adam ($\beta_1 = 0.5$, $\beta_2 = 0.9$) as the optimizer of the generator. The λ_d is Eq. 5.6, λ_g and λ_r in Eq. 5.14 are all set to be 1. The λ_s in Eq 5.13 is set to be 5.

For the discriminator, we use OpenCLIP ViT-G/14 [52] as the image encoder. For the generator, we additionally use the features from OpenCLIP ConvNext-XXLarge for the feature ensemble module. All the parameters of the image encoders are fixed during the training process. Knowledge distillation [65] is adopted as in previous work [40, 26]. The mapper is first trained and its parameters are fixed during the training of the discriminator and generator. Different with the previous work [40] which needs a careful adjustment of learning rates, our model can obtain remarkable performance with fixed ones. The learning rates of the mapper, generator and discriminator are set to be 1e-4, 1e-4 and 2e-4, respectively.

Our model is implemented based on Tensorflow[1] and trained on NVIDIA GeForce RTX 3090.

5.3.4 Experimental Result

Overall Performance We compare the performance of CaptionANT with both AR models [4, 17, 27], semi-autoregressive (SAR) models [26, 131] and NAR models [29, 25, 40]. Following previous work [40], we choose AIC as our AR baseline. AIC is

[1] `https://www.tensorflow.org`

Table 5.3: Ablation Study of CaptionANT.

	B1	B4	M	R	S	C
CaptionANT	**80.8**	**38.0**	**28.7**	**58.7**	**22.5**	**126.2**
- w/o T.	80.0	37.1	28.3	58.3	22.0	123.5
- w/o F.	79.9	36.4	28.1	57.9	22.0	121.4
- w/o R.	78.5	35.1	27.3	56.8	20.7	116.0
- w/o P. (ANT)	74.9	31.1	25.7	54.3	19.0	102.4

a Transformer based AR model which is first trained with cross entropy and then fine-tuned with SCST [109].

The evaluation results of the "Karpathy" split and the online server can be found in Table 5.1 and Table 5.2, respectively. CaptionANT obtains new state-of-the-art performance for fully NAR models. For the "Karpathy" split, it achieves 126.2 for CIDEr, which is 2.2 higher than the existing best fully NAR model, CMAL [40]. Besides, it is also the only fully NAR model which can outperform the reported results of PNAIC[2]. It obtains extremely close performance compared with AIC (bw=1), and even outperforms it on some metrics. It is the first time a fully NAR model can achieve such remarkable performance. More importantly, existing SAR and NAR models need more than 50M parameters to obtain close performance as the AR baseline, while CaptionANT obtains the remarkable performance with only 18.2M parameters. It is only 33.1% parameters of the models like AIC and PNAIC, and 36.3% parameters of CMAL. Different with SAR models, which improve model performance by sacrificing speedup, CaptionANT is 26.72× faster than AIC (bw=3). This speedup is much higher than other NAR models. These experimental results demonstrate that CaptionANT can achieve better performance with much fewer parameters and faster speed.

[2]SAR models can further improve performance by sacrificing speedup. More experimental results on these models can be found in their original papers.

Table 5.4: Effectiveness of the Contrastive Constraints.

	B1	B4	M	R	S	C
Only C_d	80.5	37.5	28.4	58.4	22.1	124.3
Only C_g	79.3	36.8	28.0	57.8	21.5	121.7
Both	**80.8**	**38.0**	**28.7**	**58.7**	**22.5**	**126.2**

Performance in Different Strucures We also explore the differences brought by different discriminator structures. We compare the performance between Struct. A: the structure which uses image representations as additional input of the discriminator as in the previous work [107], and Struct. B: the structure adopted in CaptionANT. The results can be found in Figure 5.4 (a). Compared to Struct. A, Struct. B can effectively make use of unpaired samples to regularize hidden representations. The discriminator thus can better align images and texts, and finally obtains better performance.

AIC: an old truck sitting in a field of flowers
Only C_d: an old truck parked in the grass in a field
Only C_g: an old truck parked in a field with a field
Both: an old truck is parked in a field of flowers

Ground Truth: a rusted out truck parked next to some yellow flowers

AIC: two bikes parked next to a blue bus
Only C_d: two blue bus with a bike on top of it
Only C_g: two blue buses with a bike on the top of it
Both: a bus with two bikes on the front of it

Ground Truth: a bus that has bikes mounted on the front of it

Figure 5.5: Examples of Generated Captions.

In addition, we replace the Light PASM in CaptionANT with Self-modulation (SM) and PASM. Their performance is shown in Figure 5.4 (b). Both PASM and Light PASM outperform Self-Modulation. It is consistent with the results in the previous

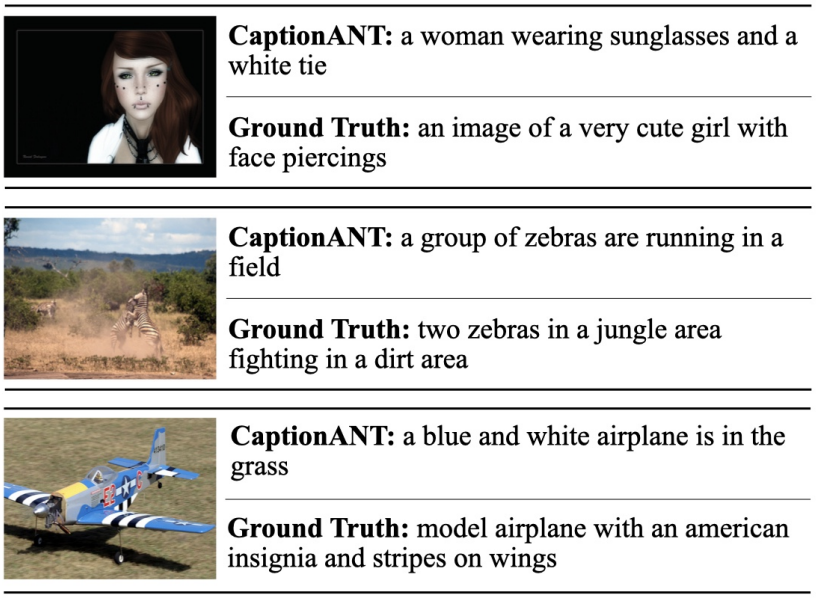

Figure 5.6: Failure Cases.

work [107]. PASM and Light PASM provide diverse input signals which can help models recover different word representations more effectively. The performance between PASM and Light PASM is extremely close, but the parameter number is significantly reduced after adopting Light PASM (the parameter number of the model with PASM is 27.0M while the number of adopting Light PASM is 18.2M). It demonstrates that Light PASM can make the model lighter while maintaining the original performance.

Ablation Study Furthermore, we explore the effectiveness of the adopted techniques and show the results in Table 5.3. The "T.", "F." and "R." indicate the truncation trick, feature ensemble and the reconstruction constraint, respectively. The "P." indicates the projection structure in the discriminator of CaptionANT. After further removing it, the settings will be similar to ANT [107]. The performance continuously decreases after removing these techniques, which demonstrates their effectiveness.

The contrastive constraints are adopted in the training objectives of the discriminator

and the generator. We also conduct experiments to explore its effectiveness and demonstrate the results in Table 5.4. Both C_d and C_g contribute to the improvement of model performance, while the contribution from C_d is more important. C_d can help the discriminator obtain more reasonable hidden representations, and identify irrelevant captions more accurately.

Case Study The effectiveness of the contrastive constraints can also be illustrated with the samples in Figure 5.5. In the first case, the model fails to capture the detail "flower" if one of the constraints is disabled, while the detail is captured accurately when using the two constraints together. In the second case, the models confuse the numbers of "bicycles" and the "bus" if the constraints are lost. With the two constraints, the model describes the numbers correctly.

To perform a complete analysis of CaptionANT, we also show failure cases in Figure 5.6. In the first case, CaptionANT meets an image in a less common style and uses unrelated words (like sunglasses, and white tie) to describe it. For the second case, although the style is a common one, the content that describes two fighting zebras is not frequent, and CaptionANT fails to describe this image accurately. For the third case, CaptionANT gives a general description, but misunderstands the relatively complicated details (black and white stripes). And it also fails to recognize that this plane is a model airplane. These cases demonstrate that the capacity of CaptionANT in processing less common image styles or content and identifying complicated details requires further enhancement.

5.4 Summary

In this work, we first analyze the limitations of existing MLE-based NAR models, whose inherent multi-modality problem will be exacerbated in image captioning. Although GANs have potential to tackle this problem, the existing GAN-based NAR

model fails to learn complicated relations between images and text, and thus obtains poor performance when transferred to image captioning.

To tackle this problem, we propose CaptionANT. CaptionANT is constructed based on GANs, so it is naturally free from the multi-modality problem. To model the complicated relations between various images and text, we first modify the discriminator structure to enable the use of contrastive learning. The model thus can effectively make use of unpaired samples. Then, we integrate a reconstruction process into the training to better utilize paired samples. By further combining with other effective techniques (like feature ensemble and the truncation trick) and our proposed lightweight structure, CaptionANT achieves new state-of-the-art performance for fully NAR models on the MSCOCO dataset with 36.3% parameters of the existing best fully NAR model and 26.72× speedup compared with the AR baseline.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we first analyze the limitations of existing MLE-based autoregressive models. Then, we explore how to construct language GANs based on representation modeling methods. After that, we adopt representation modeling methods to build NAR models. We demonstrate the effectiveness of GAN-based NAR models in both IIS and CIS.

More specifically, the contributions of this thesis can be summarized as follows:

- To avoid the non-differentiable sampling operation in language GANs, we adopt representation modeling methods to train the generator to obtain word representations instead of probability distributions. Thus, the generator can make use of gradients from the discriminator to update the generator. Even though, its performance is still limited by the invalid sampling problem and unhealthy gradients. We tackle these two problems by adopting dropout sampling and fully normalized LSTM. The experimental results demonstrate that our proposed model, InitialGAN, can outperform MLE without making use of any

pre-training techniques.

- We adopt representation modeling methods to construct GAN-based NAR models for IIS. We propose Position-Aware Self-modulation which can provide more effective signals to recover word representations, and Dependency Feed Forward network (Dependency FFN) which can help the model to capture more accurate word dependencies in the unstable training of GANs. The experimental results demonstrate that our proposed model, ANT, can obtain comparable performance as other models with much lower decoding latency.

- In order to enable GAN-based NAR models to model complicated mapping relations in the CIS, we revise the discriminator structure so it can effectively make use of unpaired samples with contrastive learning. Furthermore, reconstruction process is incorporated into the training procedure to better utilize paired samples. We test the performance of our proposed model on the MSCOCO dataset. The experimental results demonstrate that our proposed model achieves a new state-of-the-art for fully NAR models with lower parameter number and faster speed.

This thesis demonstrates the great potential of building NAR language GANs based on representation modeling methods in various tasks.

6.2 Future Work

There are several important directions need to be explored in the future.

First of all, it is important to scale up the models to apply it in more complicated scenarios. Building larger models can increase its capacity in learning more complicated data distributions. However, there may be various problems like gradient vanishment need to be tackled. How to improve model performance by scaling up the model size

is a significant problem need to be tackled to apply it in more complicated scenarios.

Secondly, a classical problem of GANs is its unstable training process. Although a number of techniques have been proposed to relieve this problem, it has not been completely tackled yet. How to further stabilize the training process of language GANs is an important problem need to be explored.

Last but not least, LLMs have obtained remarkable performance in various tasks. How to make use of LLMs to boost the performance of GAN-based NAR models is another research topic needs to be studied.

Appendix A

Theoretical Proof

A.1 Proof of Theorem 1

Suppose F is a non-linear transformation in a model. We investigate the effects of layer normalization by analyzing two different implementations:

$$\mathbf{y}_{l+1}^{(a)} = F_a(\mathbf{y}_l) \tag{A.1}$$

$$\mathbf{y}_{l+1}^{(b)} = F_b(LN(\mathbf{y}_l)) \tag{A.2}$$

where $LN(\cdot)$ is layer normalization. These two implementations are both based on $\mathbf{y}_l$, which is the output from the previous layer. The difference is that Eq. A.1 does not use layer normalization in the input, while Eq. A.1 uses it. The gradients of output to input are:

$$\frac{d\mathbf{y}_{l+1}^{(a)}}{d\mathbf{y}_l} = F_a'(\mathbf{y}_l) \tag{A.3}$$

$$\frac{d\mathbf{y}_{l+1}^{(b)}}{d\mathbf{y}_l} = F_b'(LN(\mathbf{y}_l)) \cdot LN'(\mathbf{y}_l) \tag{A.4}$$

There are two differences in Eq. A.3 and Eq. A.4. $\mathbf{y}_l$ is normalized when calculating $F_b'(\cdot)$. It can prevent input from lying in an interval whose gradients are extremely

small. Another difference is the additional term $LN'(y_l)$ in Eq. A.4. To simplify the notations, we use $\mathtt{x}$ to represent $\mathbf{y}_l$. Layer normalization is related to the mean and standard deviation among all the dimensions of $\mathtt{x}$. We start the analyses from the i-th dimension in $\mathtt{x}$, which is denoted as $\mathtt{x}_{[\mathtt{i}]}$.

$$\begin{aligned} LN'(\mathtt{x}_{[\mathtt{i}]}) &= [\frac{\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}}{\sigma_{\mathtt{x}}}]' \\ &= \frac{([\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}]' \cdot \sigma_{\mathtt{x}}) - ((\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}) \cdot \sigma'_{\mathtt{x}})}{\sigma^2_{\mathtt{x}}} \end{aligned} \tag{A.5}$$

The derivative of $[\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}]$ is:

$$[\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}]' = d(\mathtt{x}_{[\mathtt{i}]} - \frac{1}{H}\sum_{j=1}^{H}\mathtt{x}_{[\mathtt{j}]})/d\mathtt{x}_{[\mathtt{i}]} = 1 - \frac{1}{H} \tag{A.6}$$

where H is the dimension of $\mathtt{x}$. The derivative of $\sigma_{\mathtt{x}}$ is:

$$\begin{aligned} \sigma'_{\mathtt{x}} &= d\mathbb{E}(\mathtt{x}^2 - \mathbb{E}^2(\mathtt{x}))^{\frac{1}{2}}/d\mathtt{x}_{[\mathtt{i}]} \\ &= \frac{1}{2}\mathbb{E}[\mathtt{x}^2 - \mathbb{E}^2(\mathtt{x})]^{-\frac{1}{2}} \cdot [\mathbb{E}(\mathtt{x}^2) - \mathbb{E}^2(\mathtt{x})]' \\ &= \frac{1}{2\sigma_{\mathtt{x}}}(\frac{2\mathtt{x}_{[\mathtt{i}]}}{H} - \frac{2\mathbb{E}(\mathtt{x})}{H}) \\ &= \frac{\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}}}{H \cdot \sigma_{\mathtt{x}}} \end{aligned} \tag{A.7}$$

Considering Eq. A.5, A.6 and A.7, we have:

$$\begin{aligned} LN'(\mathtt{x}_{[\mathtt{i}]}) &= \frac{(1 - \frac{1}{H}) \cdot \sigma_{\mathtt{x}} - \frac{(\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}})^2}{H \cdot \sigma_{\mathtt{x}}}}{\sigma^2_{\mathtt{x}}} \\ &= \frac{-\frac{1}{H}\sigma^2_{\mathtt{x}} + \sigma^2_{\mathtt{x}} - \frac{(\mathtt{x}_{[\mathtt{i}]} - \mu_{\mathtt{x}})^2}{H}}{\sigma^3_{\mathtt{x}}} \\ &= \frac{-\frac{\sigma^2_{\mathtt{x}}}{H} + \frac{\sum_{j \neq i}(\mathtt{x}_{[\mathtt{j}]} - \mu_{\mathtt{x}})^2}{H}}{\sigma^3_{\mathtt{x}}} \\ &= \frac{-\frac{\sigma^2_{\mathtt{x}}}{H} + \frac{H-1}{H}\frac{\sum_{j \neq i}(\mathtt{x}_{[\mathtt{j}]} - \mu_{\mathtt{x}})^2}{H-1}}{\sigma^3_{\mathtt{x}}} \end{aligned} \tag{A.8}$$

When H is large enough, we can adopt *the law of large numbers* to obtain the following relation:

$$\frac{1}{H-1}\sum_{j \neq i}(\mathtt{x}_{[\mathtt{j}]} - \mu_{\mathtt{x}})^2 \approx \sigma^2_{\mathtt{x}} \tag{A.9}$$

Thus, Eq. A.8 can be further transformed as:

$$\begin{aligned} LN'(\mathtt{x}_{[\mathtt{i}]}) &\approx \frac{-\frac{\sigma_{\mathtt{x}}^2}{H} + \frac{H-1}{H}\sigma_{\mathtt{x}}^2}{\sigma_{\mathtt{x}}^3} \\ &= \frac{H-2}{H} \cdot \frac{1}{\sigma_{\mathtt{x}}} = (1 - \frac{2}{H})\frac{1}{\sigma_{\mathtt{x}}} \end{aligned} \tag{A.10}$$

We can regard $1 - \frac{2}{H} \approx 1$ when H is large enough, so we have:

$$LN'(\mathtt{x}_{[\mathtt{i}]}) \approx \frac{1}{\sigma_{\mathtt{x}}} \tag{A.11}$$

If the deviation of $\mathtt{x}$ is smaller than 1, this term will be a scaler factor larger than 1. In this time, it can help augment gradients and relieve gradient vanishment problem.

References

[1] 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.

[2] Nader Akoury, Kalpesh Krishna, and Mohit Iyyer. Syntactically supervised transformers for faster neural machine translation. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1269–1281. Association for Computational Linguistics, 2019.

[3] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. SPICE: semantic propositional image caption evaluation. In *ECCV*, pages 382–398, 2016.

[4] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, pages 6077–6086, 2018.

[5] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 2017.

[6] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

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

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

[9] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179, 2015.

[10] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on*

Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.

[11] Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. Language gans falling short. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[12] 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 Eduardo Blanco and Wei Lu, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 169–174. Association for Computational Linguistics, 2018.

[13] 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.

[14] Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. Adversarial text generation via feature-mover's distance. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4671–4682, 2018.

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

[16] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server. *CoRR*, abs/1504.00325, 2015.

[17] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. Meshed-memory transformer for image captioning. In *CVPR*, pages 10575–10584, 2020.

[18] Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack W. Rae. Training language gans from scratch. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *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 4302–4313, 2019.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[20] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 8780–8794, 2021.

[21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg

Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[22] Cunxiao Du, Zhaopeng Tu, and Jing Jiang. Order-agnostic cross entropy for non-autoregressive machine translation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2849–2859. PMLR, 2021.

[23] Angela Fan, Mike Lewis, and Yann N. Dauphin. Hierarchical neural story generation. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 889–898. Association for Computational Linguistics, 2018.

[24] William Fedus, Ian J. Goodfellow, and Andrew M. Dai. Maskgan: Better text generation via filling in the _____. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[25] Zhengcong Fei. Iterative back modification for faster image captioning. In Chang Wen Chen, Rita Cucchiara, Xian-Sheng Hua, Guo-Jun Qi, Elisa Ricci, Zhengyou Zhang, and Roger Zimmermann, editors, *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, pages 3182–3190. ACM, 2020.

[26] Zhengcong Fei. Partially non-autoregressive image captioning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The*

Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021, pages 1309–1316. AAAI Press, 2021.

[27] Zhengcong Fei. Attention-aligned transformer for image captioning. In *AAAI*, pages 607–615. AAAI Press, 2022.

[28] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XV*, volume 13675 of *Lecture Notes in Computer Science*, pages 89–106. Springer, 2022.

[29] Junlong Gao, Xi Meng, Shiqi Wang, Xia Li, Shanshe Wang, Siwei Ma, and Wen Gao. Masked non-autoregressive image captioning. *CoRR*, abs/1906.00717, 2019.

[30] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR, 2017.

[31] Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 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, 2020.

[32] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6111–6120. Association for Computational Linguistics, 2019.

[33] Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017.

[34] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[35] Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4601–4609, 2016.

[36] Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In William W. Cohen and Andrew W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, vol-

ume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM, 2006.

[37] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 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, 2018.

[38] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5767–5777, 2017.

[39] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5141–5148. AAAI Press, 2018.

[40] Longteng Guo, Jing Liu, Xinxin Zhu, Xingjian He, Jie Jiang, and Hanqing Lu. Non-autoregressive image captioning with counterfactuals-critical multi-agent learning. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 767–773. ijcai.org, 2020.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision*

and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, pages 770–778. IEEE Computer Society, 2016.

[42] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *CoRR*, abs/1606.08415, 2016.

[43] 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 Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6626–6637, 2017.

[44] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[45] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.

[47] Lu Hou, Jinhua Zhu, James T. Kwok, Fei Gao, Tao Qin, and Tie-Yan Liu. Normalization helps training of quantized LSTM. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *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 7344–7354, 2019.

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

[49] Fei Huang, Tianhua Tao, Hao Zhou, Lei Li, and Minlie Huang. On the learning of non-autoregressive transformers. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9356–9376. PMLR, 2022.

[50] Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. Directed acyclic transformer for non-autoregressive machine translation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9410–9428. PMLR, 2022.

[51] Xiao Shi Huang, Felipe Pérez, and Maksims Volkovs. Improving non-autoregressive translation models without distillation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[52] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021.

[53] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5967–5976. IEEE Computer Society, 2017.

[54] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[55] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pages 2146–2153. IEEE Computer Society, 2009.

[56] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two pure transformers can make one strong gan, and that can scale up. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 14745–14758, 2021.

[57] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 10124–10134. IEEE, 2023.

[58] Animesh Karnewar and Oliver Wang. MSG-GAN: multi-scale gradients for generative adversarial networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 7796–7805. Computer Vision Foundation / IEEE, 2020.

[59] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, pages 3128–3137, 2015.

[60] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 852–863, 2021.

[61] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 8107–8116. Computer Vision Foundation / IEEE, 2020.

[62] Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 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, 2020.

[63] Pei Ke, Fei Huang, Minlie Huang, and Xiaoyan Zhu. ARAML: A stable adversarial training framework for text generation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4270–4280. Association for Computational Linguistics, 2019.

[64] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858, 2019.

[65] Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1317–1327. The Association for Computational Linguistics, 2016.

[66] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[67] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012.

[69] Sachin Kumar and Yulia Tsvetkov. End-to-end differentiable {gan}s for text generation. In *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop*, 2020.

[70] Matt J. Kusner and José Miguel Hernández-Lobato. GANS for sequences of discrete elements with the gumbel-softmax distribution. *CoRR*, abs/1611.04051, 2016.

[71] Alon Lavie and Abhaya Agarwal. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. In *WMT@ACL*, pages 228–231, 2007.

[72] Honglak Lee, Roger B. Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 609–616. ACM, 2009.

[73] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[74] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip H. S. Torr. Controllable text-to-image generation. In *NeurIPS*, pages 2063–2073, 2019.

[75] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2157–2169. Association for Computational Linguistics, 2017.

[76] Muyang Li, Tianle Cai, Jiaxin Cao, Qinsheng Zhang, Han Cai, Junjie Bai, Yangqing Jia, Kai Li, and Song Han. Distrifusion: Distributed parallel inference for high-resolution diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 7183–7193. IEEE, 2024.

[77] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[78] Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. Hint-based training for non-autoregressive machine translation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5707–5712. Association for Computational Linguistics, 2019.

[79] Jae Hyun Lim and Jong Chul Ye. Geometric GAN. *CoRR*, abs/1705.02894, 2017.

[80] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[81] Chun-Hsing Lin, Siang-Ruei Wu, Hung-yi Lee, and Yun-Nung Chen. Taylorgan: Neighbor-augmented policy update towards sample-efficient natural language generation. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[82] Kevin Lin, Dianqi Li, Xiaodong He, Ming-Ting Sun, and Zhengyou Zhang. Adversarial ranking for language generation. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan,

and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3155–3165, 2017.

[83] 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 David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

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

[85] Puyuan Liu, Chenyang Huang, and Lili Mou. Learning non-autoregressive models from search for unsupervised sentence summarization. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7916–7929. Association for Computational Linguistics, 2022.

[86] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[87] Jianjie Luo, Yehao Li, Yingwei Pan, Ting Yao, Jianlin Feng, Hongyang Chao, and Tao Mei. Semantic-conditional diffusion networks for image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*

2023, Vancouver, BC, Canada, June 17-24, 2023, pages 23359–23368. IEEE, 2023.

[88] Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard H. Hovy. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4281–4291. Association for Computational Linguistics, 2019.

[89] Yun Ma and Qing Li. Exploring non-autoregressive text style transfer. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9267–9278. Association for Computational Linguistics, 2021.

[90] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[91] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2813–2821. IEEE Computer Society, 2017.

[92] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[93] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[94] Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. Diffeditor: Boosting accuracy and flexibility on diffusion-based image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 8488–8497. IEEE, 2024.

[95] Weili Nie, Nina Narodytska, and Ankit Patel. Relgan: Relational generative adversarial networks for text generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[96] Mohammad Norouzi, Samy Bengio, Zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. Reward augmented maximum likelihood for neural structured prediction. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1723–1731, 2016.

[97] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 271–279, 2016.

[98] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the*

40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA, pages 311–318. ACL, 2002.

[99] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[100] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018.

[101] Henning Petzka, Asja Fischer, and Denis Lukovnikov. On the regularization of wasserstein gans. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[102] Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. Glancing transformer for non-autoregressive neural machine translation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1993–2003. Association for Computational Linguistics, 2021.

[103] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[104] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125, 2022.

[105] Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. Preventing posterior collapse with delta-vaes. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[106] Scott E. Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *ICML*, pages 1060–1069, 2016.

[107] Da Ren, Yi Cai, and Qing Li. An adversarial non-autoregressive model for text generation with incomplete information. *CoRR*, abs/2305.03977, 2023.

[108] Da Ren and Qing Li. Initialgan: A language gan with completely random initialization. *CoRR*, abs/2208.02531, 2022.

[109] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1179–1195. IEEE Computer Society, 2017.

[110] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In

IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022, pages 10674–10685. IEEE, 2022.

[111] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L. Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[112] Adam Santoro, Ryan Faulkner, David Raposo, Jack W. Rae, Mike Chrzanowski, Theophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy P. Lillicrap. Relational recurrent neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7310–7321, 2018.

[113] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 30105–30118. PMLR, 2023.

[114] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. Coldgans: Taming language gans with cautious sampling

strategies. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[115] Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. Minimizing the bag-of-ngrams difference 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 198–205. AAAI Press, 2020.

[116] Samarth Sinha, Zhengli Zhao, Anirudh Goyal, Colin Raffel, and Augustus Odena. Top-k training of gans: Improving GAN performance by throwing away bad samples. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[117] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[118] Ruoyu Sun, Tiantian Fang, and Alexander G. Schwing. Towards a better global loss landscape of gans. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[119] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes,

Neil D. Lawrence, and Kilian Q. Weinberger, editors, *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, 2014.

[120] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *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, 2017.

[121] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, pages 4566–4575, 2015.

[122] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[123] Chunqi Wang, Ji Zhang, and Haiqing Chen. Semi-autoregressive neural machine translation. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 479–488. Association for Computational Linguistics, 2018.

[124] Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. Imitation learning for non-autoregressive neural machine translation. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1304–1312. Association for Computational Linguistics, 2019.

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

[126] Jiqing Wu, Zhiwu Huang, Janine Thoma, Dinesh Acharya, and Luc Van Gool. Wasserstein divergence for gans. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, volume 11209 of *Lecture Notes in Computer Science*, pages 673–688. Springer, 2018.

[127] Yan Wu, Jeff Donahue, David Balduzzi, Karen Simonyan, and Timothy P. Lillicrap. LOGAN: latent optimisation for generative adversarial networks. *CoRR*, abs/1912.00953, 2019.

[128] 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. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(10):11407–11427, 2023.

[129] Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *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 4383–4393, 2019.

[130] Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. Variational autoencoder for semi-supervised text classification. In Satinder Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3358–3364. AAAI Press, 2017.

[131] Xu Yan, Zhengcong Fei, Zekang Li, Shuhui Wang, Qingming Huang, and Qi Tian. Semi-autoregressive image captioning. In Heng Tao Shen, Yueting Zhuang, John R. Smith, Yang Yang, Pablo César, Florian Metze, and Balakrishnan Prabhakaran, editors, *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 2708–2716. ACM, 2021.

[132] Yasin Yazici, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, and Vijay Chandrasekhar. The unusual effectiveness of averaging in GAN training. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[133] 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, 2022.

[134] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In Satinder Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press, 2017.

[135] Fan Zhang, Mei Tu, and Jinyao Yan. Accelerating neural machine translation with partial word embedding compression. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14356–14364. AAAI Press, 2021.

[136] Han Zhang, Tao Xu, and Hongsheng Li. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, pages 5908–5916, 2017.

[137] Han Zhang, Weichong Yin, Yewei Fang, Lanxin Li, Boqiang Duan, Zhihua Wu, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vilg: Unified generative pre-training for bidirectional vision-language generation. *CoRR*, abs/2112.15283, 2021.

[138] Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. Bridging the gap between training and inference for neural machine translation. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4334–4343. Association for Computational Linguistics, 2019.

[139] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 4006–4015. PMLR, 2017.

[140] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient GAN training. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[141] Zhiming Zhou, Jiadong Liang, Yuxuan Song, Lantao Yu, Hongwei Wang, Weinan Zhang, Yong Yu, and Zhihua Zhang. Lipschitz generative adversarial

nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7584–7593. PMLR, 2019.

[142] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2242–2251, 2017.

[143] Qile Zhu, Wei Bi, Xiaojiang Liu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. A batch normalized inference network keeps the KL vanishing away. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2636–2649. Association for Computational Linguistics, 2020.

[144] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz, editors, *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM, 2018.