电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

硕士学位论文

MASTER THESIS

论文题目 基于生成对抗网络的中文文本生成

学科专业	通信与信息系统
学　　号	201821010601
作者姓名	段明君
指导教师	李玉柏 教授

分类号 _____ 密级 _____

UDC [注1] _____

学 位 论 文

基于生成对抗网络的中文文本生成

（题名和副题名）

段明君

（作者姓名）

指导教师 李玉柏 教授

电子科技大学 成都

（姓名、职称、单位名称）

申请学位级别 硕士 学科专业 通信与信息系统

提交论文日期 2021.03.31 论文答辩日期 2021.06.02

学位授予单位和日期 电子科技大学 2021 年 06 月

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

Research on Chinese Text Generation based on Generative Adversarial Networks

A Masteral Dissertation Submitted to

University of Electronic Science and Technology of China

Discipline: Communication and Information System

Author: Mingjun Duan

Supervisor: Prof. Yubai Li

School: School of Information and Communication Engineering

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名：段明君 日期：2021年6月24日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名：段明君 导师签名：李玉柏

日期：2021年6月24日

摘　要

深度学习技术在近几十年来得到高速发展，这一技术随即被应用在越来越多的领域中，自然语言处理也有诸多研究方向可以使用深度学习，文本生成就是重要方向之一。文本生成是一项基础性研究，能够在许多实际场景落地，例如提取文本摘要、文本风格转换、文本自动纠错等。

生成对抗网络是深度学习中一个备受关注的框架模型，现有的文本序列模型都具有数据离散的特点，将生成对抗网络直接应用于文本生成时，就会面临训练过程中无法通过反向传播完成参数更新的问题。另外，由于生成对抗网络的训练方式是把一个噪声分布映射在先验真实文本分布，但目前文本生成任务一般是字符级生成，这样的方式很容易出现生成文本重复性过高甚至模式崩溃的情况。最后，由于原始生成对抗网络中生成器的限制，训练过程中对于文本特征的提取能力有限，生成长文本的质量普遍较低。针对上述问题，本文提出了基于生成对抗网络的文本生成算法模型，主要工作内容如下：

（1）本文基于序列生成对抗网络与自注意力机制的思想，提出了一种无监督的文本生成算法，生成网络融入了自注意力机制，在原始的 Transformer 模型上增加了使用高斯偏差的局部建模，改善了原模型不可并行化的问题，同时提升了捕获长距离文本特征的能力。同时目标函数使用惩罚最小化的机制，引入一种新方式来衡量 Wasserstein 距离，保证梯度不会消失，并且模式崩溃问题也得到改善。本文将新的模型在商品描述数据集上进行实验，判断在长文本方面的文本生成效果，实验结果证明，本文提出的生成模型与对比模型相比，生成的文本质量更高，文本多样性也更丰富。

（2）将本文提出的文本生成模型应用于诗歌生成领域，以证明模型的普适性。针对诗歌特有的固定字数、尾词押韵与韵律声调规则，本文在上述工作的基础上，引入了拼音汉字对照表，并在算法中增加了尾词押韵判断部分，使得不仅能够保证生成诗歌的字数符合规则，格律音调上也能够最大程度符合诗歌规律。本文在唐代绝句诗词数据集上进行对比实验，与基线模型相比，本文提出的算法模型在综合评价尤其是押韵部分，取得了更高的分数。实验数据充分验证本研究提出的模型在文本生成领域的价值，也为未来工作的研究方向提供参考。

关键词：文本生成，生成对抗网络，强化学习，自注意力机制

ABSTRACT

Deep learning technology has developed rapidly in recent decades, and this technology has been applied in more and more fields. There are also many research directions in natural language processing that can use deep learning, and text generation is one of the important directions. Text generation is a basic research that can be applied to many real scenes, such as text abstract extraction, text style conversion, and automatic text error correction.

Generative adversarial network is a framework model that has received much attention in deep learning. Existing text sequence models all have the characteristics of data discrete. When GAN is directly applied to text generation, it will face the problem that the parameter update cannot be completed through backpropagation during the training process. In addition, the training method of GAN is to map a noise distribution to the prior real text distribution. However, the current text generation task is generally character-level generation, and this method is prone to excessively high repetitiveness of the generated text and even mode collapse. Finally, due to the limitation of the generator in the original GAN, the ability to extract text features during the training process is limited, and the quality of the generated long text is generally low. In response to the above problems, this paper proposes a text generation model based on GAN. The main work content is as follows:

(1) Based on the idea of seqGAN and self-attention, this paper proposes an unsupervised text generation algorithm. The generative network incorporates self-attention, and adds local modeling using Gaussian deviation to the original Transformer model. This method improves the problem that the original model cannot be parallelized, and at the same time improves the ability to capture long-distance text features. At the same time, the objective function uses the mechanism of minimizing the penalty and introduces a new way to measure the Wasserstein distance. This ensures that the gradient will not disappear, and the mode collapse problem is also improved. In this paper, the new model is tested on the product description data set to judge the effect of text generation in long text. The experimental results prove that compared with the comparative model, the generated text quality of the generated model proposed in this paper is higher, and the text diversity is also richer.

(2) The text generation model proposed in this paper is applied to the field of poetry generation to prove the universality of the model. Aiming at the unique fixed word count and rhythm rules of poetry, based on the above work, this paper introduces the pinyin Chinese character comparison table, and adds the rhyming judgment part of the tail word to the algorithm. This method makes it possible not only to ensure that the number of words in the generated poetry conforms to the rules, but also that the metrical tones can also conform to the rules. This paper conducts comparative experiments on the Tang Dynasty quatrain poetry data set. Compared with the baseline model, the algorithm model proposed in this paper has achieved higher scores in the comprehensive evaluation, especially the rhyme part. The experimental data fully proves the value of the model proposed in this study in the field of text generation, and also provides a reference for the research direction of future work.

Keywords: text generation, generative adversarial network, reinforcement learning, self-attention

目 录

缩略词表

英文简写	英文全称	中文全称
NLP	Natural Language Processing	自然语言处理
NLG	Natural Language Generation	自然语言生成
GAN	Generative Adversarial Networks	生成对抗网络
SeqGAN	Sequence Generative Adversarial Networks	序列生成对抗网络
LSTM	Long Short-Term Memory	长短时记忆模型
AE	Autoencoder	自编码器
DAE	Denoising Auto-Encoder	降噪自编码器
MaliGAN	Maximum-Likelihood Augmented Discrete GAN	极大似然增强离散数据生成对抗网络
RNN	Recurrent Neural Network	循环神经网络
CNN	Convolution Neural Network	卷积神经网络
FFNNLM	Feedforward Neural Network Language Model	前馈神经网络语言模型
RNNLM	Recurrent Neural Network Language Model	基于循环神经网络的语言模型
GRU	Gated Recurrent Unit	门控循环单元
MCTS	Monte Carlo Tree Search	蒙特卡洛树搜索
PB-ESGAN	Penalty-based Enhanced Sequence Generative Adversarial Net	基于惩罚最小化强化序列生成对抗网络算法
BLEU	Bilingual Evaluation Understudy	双语互译质量辅助工具
RNNPG	RNN-based Poetry Generation	基于循环神经网络的生成算法模型

第一章 绪论

1.1 研究背景及意义

近些年来，人工智能领域迎来高速发展时期，在越来越多的领域发挥作用，取代人力消耗，用机器完成更多样化的工作。自然语言处理（Natural Language Processing, NLP）是格外受关注的方向之一。自然语言在人类日常行为交流中是最常见通用的方式，因此许多研究致力于生成人类能够理解的自然语言文本内容，这也是人工智能系统的交互模块中极其重要的构成部分。除此之外，机器翻译[1]、图像描述[2]、文本摘要提取[3]、对话系统[4]等应用的实现都依赖于文本生成技术，自然语言生成（Natural Language Generation, NLG）也因此成为自然语言处理里备受关注的方向，同样，完成高质量的文本生成也能够大程度推进自然语言处理领域的研究进展。

最初的文本生成，依赖于规则模板的形式，意即人工制定语言规则，比如词性标注、词法分析等方式，实现文本生成。但是类似的方法仅仅适用于数据量较小且规则较为简单的情况，可扩展性和通用性都欠佳。随着时代发展，数据量与数据规则都逐渐丰富，显然这种依赖于模板的方式已经不再适用。

深度学习技术快速发展，也在诸多领域卓有成效，自然也有许多研究人员尝试着将其应用在文本生成领域，过去曾被多次应用过的模型包括了前馈神经网络[5]和循环神经网络[6]等等，这些模型基本上都是建立在极大似然估计的基础上，在整个训练过程中通过反向传播完成参数的更新，并且得到与训练集相似度较高的结果。但是这些基础的循环神经网络一般都会陷入同样的困境，即对于训练数据的分布存在过度依赖，形成过拟合，导致生成的文本质量一般，多样性也不高。

针对上述问题，科研人员在基础的循环神经网络上不断提出新的解决方案。比如 Ranzato 等人提出在训练过程中引入强化学习的思想[7]，而 Kiros 等人尝试将编解码形式用于将输入的句子进行重建，编解码器都采用循环神经网络，也就是跳读模型（skip-through）[8]。这些模型在机器翻译等领域获得成功，但是在文本生成方面的表现仍不足以令人满意。其根本原因是这样的模型更关注每一句生成的内容，但在句与句之间的语义连贯上缺少关注。同时，将循环神经网络应用于文本生成还存在另一个困境，即随着生成句子长度的增长，错误率会逐渐增加，这对于长文本的生成是一个挑战。

在这样的情况下，生成对抗网络（Generative Adversarial Networks, GAN）逐渐被更多研究人员关注和使用 [9]。这一模型框架在 2014 年由 Goodfellow 等人提出，自出现起就因其在数据生成方面取得的巨大突破备受关注。生成对抗网络的核心

思想在于博弈，整个网络包括生成器与判别器两部分，生成器持续学习，优化生成内容，以生成与现实世界的真实数据更接近的新样本，而判别器则要避免被两种数据混淆，致力于每一轮训练都能够判断哪些是真实数据，哪些是生成数据。将生成网络和判别网络进行联合训练，在经过多回合训练之后，生成器趋于生成足以混淆判别器的新样本。

在图像等连续数据类型的生成上，生成对抗网络展现了强大的功能。但语言模型的特殊之处在于这是一种典型的离散数据空间，梯度更新无法通过判别器的反向传播传递回生成器，这使生成对抗网络这一基于梯度的模型能力大大受限。但这一问题仍然有解决方法，主要有以下两类：第一类是在最原始的GAN上进行改进，例如 Gumbel-softmax GAN[10]和 TextGAN[11]这两种典型代表，就是分别引入了一个 Gumbel-softmax 技巧和一个软 argmax 算子，借此实现在文本数据中提供连续近似的离散分布，这就使整个模型仍然是端到端可微，提升了训练的稳定性。第二类方法引入强化学习里策略梯度[12]的思想，把整个文本生成的过程视作连续决策的过程，也就是利用策略梯度算法实现生成器梯度的估计，这种方法最典型的模型代表是 Lantao Yu 等人提出的序列生成对抗网络（Sequence Generative Adversarial Networks, SeqGAN）[13]。

然而基于序列生成对抗网络的文本生成仍然面临着诸多挑战，例如其使用 LSTM 作为生成器，限制了算法的并行化；同时，因损失函数汇总“奖励”设置的限制，易出现模式崩溃的问题。

综合以上背景，本文的研究目的是建立一个基于序列生成对抗网络的文本生成模型，实现基于生成对抗网络算法的文本生成模型改进。这一模型将结合自注意力机制和基于惩罚的目标函数，利用自注意力机制解决无法并行化和文本特征提取能力有限的问题，借助新的目标函数改善模式崩溃问题。从而达到提升生成文本质量、丰富生成文本多样化的目标。

1.2 国内外研究现状

生成对抗网络模型的本质是找到两个不同的分布之间的映射，一般来说两个分布是指真实的数据样本和先验噪声。作为新兴的生成模型训练网络，在连续数据生成问题上，生成对抗网络已经有了极大进展，但是面对离散序列的生成时，仍然有着明显局限。除了离散输出很难进行梯度更新之外，还有一个重要原因是判别器做出判断需要等待整个序列全部生成之后，面对没有全部生成的内容，原始的判别器很难找到目前与未来评估分数的平衡点。

下文将通过基于传统神经网络的文本生成、基于生成对抗网络的文本生成、以

及生成对抗网络引入强化学习的文本生成几个方面，介绍国内外的研究现状。

1.2.1 基于传统神经网络的文本生成

循环神经网络长期以来都被视为处理序列问题的有效工具，并且也已经有大量研究人员把这一模型应用在构造语言模型上[6]。由于前文中提到循环神经网络存在的对序列依赖问题无法有效解决，诸多变体也应运而生。Schmidhuber 等人提出的长短时记忆模型（Long Short-Term Memory, LSTM）就是其中的一种[14]，LSTM 的关键思想是在原始的循环神经网络上加入了三个门，用来控制传递进下个单元的向量。Wen[15]等人就在 LSTM 的基础上建立了对话系统，模拟了自然语言的文本风格。2014 年，Zhang[16]等人运用 RNN 进行中文诗词的生成尝试，但之后 Bengio[17]等人的研究表明，RNN 方式需要把每个目标单词的可能性都最大化才能完成，但在真实的文本生成过程中，目标单词是不可用的，模型自生成的单词会把目标单词替换掉，这就导致了整个模型在训练和生成的过程中，使用方法存在一定的差异。基于这一问题，他们提出了计划采样（Scheduled Sampling），用这一方式尝试改变训练的过程。但 Huszar[18]等人发现，计划采样也无法彻底消除掉训练阶段与生成阶段的差异。

自编码器（Autoencoder, AE）这一算法模型也在文本生成上备受关注。自编码器和生成对抗网络一样，都属于隐含变量的模型，这一算法的核心思想是把文本向量用一个隐变量来表示，计划构建一个模型，使得目标数据 X 能够利用隐变量 Z 生成得到。自编码器这一类算法模型中，较为出名的是 Kingma[19]等人提出的变分自编码器（VAE），这一算法假设前提是，每一个输入的真实文本样本，都有一个唯一对应的后验分布，并且这一后验分布是正态分布模型，同时利用神经网络来完成这个正态分布均值与方差的拟合。另外，为了防止生成过程中噪声为零，特别地将这一正态分布向着标准正态分布靠近，即均值是 0、方差是 1 的正态分布，这样可以确保模型具有生成能力。Bowman[20]等人的研究则是把 VAE 模型使用在语言模型中，并且借此得到了相对平滑的隐向量空间，在这一空间之中，可以在两个不同的向量中插值，这样就可以让已经生成的文本有更加平滑的改变。不过他们同样提到，VAE 可能会带来后验坍缩的后果，也就是解码器里的损失函数中的 KL 项趋近于零值[21]。为了改善这种后验坍缩的情况，Semeniuta[22]等人发现了新的混合结构，这一结构把正向卷积和反向卷积分别跟递归语言模型进行结合，使其有更短的运行时间和更好的收敛性。

文本生成领域中，另一个主流的研究方向是深度生成模型。Salakhutdinov[23]等人提出，深度生成模型的特征表达能力更好，这一模型的实现形式也非常多种多样。

2006 年，Hinton[24]等人发现为了优化深度信念网络（Deep Belief Nets, DBN）的模型训练效果，可以引入对比散度算法。2013 年，Bengio[25]等人的研究提出把有监督的训练方式与网络结构进行结合，实现无监督的数据分布拟合，这被称为降噪自编码器（Denoising Auto-Encoder, DAE）。但是，Kong[26]等人提出，只有把训练数据取上限值，之后判断其被输出的概率最小值，才能够将编解码器的结构应用在生成网络中，而实际场景中，计算复杂数据的分布概率是非常难以操作的。

1.2.2 基于生成对抗网络的文本生成技术

前文中所提到的有监督生成模型都面临一个同样的问题，那就是很难确定一个合适的标准来评价生成网络的生成内容，而对抗机制这一崭新的形式恰恰可以克服这个困境。生成对抗网络框架在 2014 年由 Goodfellow 等人提出，这种模型与以往传统的编解码器模型的训练方法相去甚远。自编码器主要构成部分是编码器与解码器，但生成对抗网络的主要构成部分则是生成网络和判别网络，在这之中，生成网络的作用更接近于自编码器里的解码器，目的都是把隐变量 Z 当做输入，实现生成输出文本。

在生成对抗网络中，判别器需要在诸多信息中将真实数据和生成的数据进行分辨，分辨的精确程度需要通过训练来提高。而生成器的训练过程则致力于输出更加逼近真实数据的内容，从而迷惑判别器的判断。整个过程导致一个博弈情况的出现，在训练最终，会进入纳什均衡。

生成对抗网络在计算机视觉领域已取得了显著成效，例如使用 GAN 模型进行较大且逼近真实世界图像的生成就早已顺利实现，这一成果由 Denton[27]等人完成。然而将生成对抗网络应用在文本生成领域却遇到了阻碍，主要有两方面的因素。

第一，生成对抗网络一般的应用都是在连续值数据中，但是文本数据是典型的离散数据，由连续数据直接生成离散符号的序列十分困难。Huszar[18]等人提出这种困难的原因是生成网络部分想要进行参数调节必须借助其他模块。针对连续数据的任务，整个训练过程中，生成器最初会进行随机采样，并且慢慢向着参数控制采样的形式过渡。

模型的全部损失需要通过生成器参数和判别器参数共同进行计算，而生成器参数的小幅度调整就依赖于损失的梯度，这些调整保证了生成器会逐渐输出更逼真的数据。然而由于生成器参数的变动一般幅度都很小，因此这些情况都建立在输入数据是连续值的基础上，否则就无法采样得到新的离散标签。也就是说，在离散空间里采样得到的序列结果是不会发生改变的，这直接导致判别器给出的结论不再能够给出生成器下一步的训练方向。

第二，生成一个序列的时候，生成对抗网络的判别器只能对整个序列进行评估或损失的计算，但是在序列生成的过程中，也就是一个仅有部分生成的序列时，判别器无法对此进行指导。这些问题导致生成对抗网络应用在文本生成领域十分困难。

Kusner 等人在生成对抗网络中引入了 Gumbel-softmax 分布，提出了 GSGAN，利用一个符合 Gumbel 分布[28]的向量结合原先的输出向量，并且将一个逐渐变小的向量用作除数，这样就可以用可微的 softmax 函数替代掉原先采样且不可微的 argmax 函数。Zhang 等人则提出了 TextGAN 这一文本生成模型[29]，这一模型的生成器采用了 LSTM，而判别器采用卷积网络。并且在训练生成器的过程中，并不采用生成对抗网络的标准目标，而是匹配了特征分布，用协方差矩阵代表句子的向量，利用协方差矩阵的差异化度量实现真实句子与合成句子的特征分布匹配，这很大程度上缓解了模式崩溃的问题，在定量评价的角度性能比较好，可以生成十分逼真的句子。

针对生成对抗网络无法生成离散数据这一问题，有许多研究人员对原始模型进行了改良，避开生成器对离散数据进行采样的部分。Che[30]等人的研究设计了一套新的训练技术，对生成数据分布和真实数据分布的差距进行直接计算，称之为极大似然增强离散数据生成对抗网络（Maximum-Likelihood Augmented Discrete GAN, MaliGAN）。另外，为了缓解常见的梯度消失的情况，Lin[31]等人提出了排序生成对抗网络（RankGAN），用这一模型相比其他模型，能够让判别网络给出更连续的反馈指标，主要的方式是替换掉了二值分类判别器，采用了基于余弦相似度的排序模型。

1.2.3 生成对抗网络引入强化学习的文本生成

原始的生成对抗网络无法生成质量高的文本内容，重要问题之一是判别器提供的二值反馈信息量不足，这就导致生成器需要用更多轮的样本生成、更长的训练时间，去对生成器进行改进，也容易带来模式崩溃的问题。上文中提到的特征匹配机制，就是对这种问题的尝试，但这一方式仍然只能发生在整个文本样本全部生成之后，在训练过程中带来的改善十分有限。

Bachman 和 Precup[32]等人则尝试把序列数据的生成问题转化为一个连续的决策过程，这就意味着可以通过引入强化学习的思想来解决序列生成的问题。Sutton[33]等人设计的策略梯度算法就是针对这一模型进行训练的方法，当存在一隐性的回报函数对策略学习进行指导时，就可以实现生成器参数的优化。另外，Sutskever 与 Vinyals 等人使用序列到序列模型（Seq to Seq）以及强化学习技术，搭

建了一个完善的机器翻译系统[33]，也是对这种训练方法的实践。

而将生成对抗网络与强化学习结合的方法中，引起较多关注的是 2016 年 Yu[13]等人提出的序列生成对抗网络（Sequence GAN, SeqGAN）。这一模型把生成器视作序列决策的过程，整个过程里包括代理（agent）、状态（state）、行动（action）和奖励（reward）几个概念，简单来说，就是把生成器当成代理，当前已经产生的文本看做状态，而行动意味着下一步需要生成的单词，最后奖励代表了判别器得到的评价分数，这样就使得生成器的目标转化为价值最大化，也就是从初始状态起，累积的奖励值最大化。另外，由于存在判别器无法判断生成过程中的句子序列，Yu 等人采用了 roll-out 策略，这种策略基于 Monte Carlo 搜索，可以把句子余下的单词模拟出来。

在此基础上，由于这一模型仍然存在可能出现梯度消失的问题，Wang[35]等人提出的 SentiGAN 进行了进一步的改良。SentiGAN 模型用基于惩罚的函数，替换掉了 SeqGAN 中生成器需要获得的基于奖励的目标，借此强制生成器产生规定情感标签的不同文本示例内容。

1.2.4 目前存在的问题和挑战

生成对抗网络跟强化学习的结合，给文本生成方向带来了崭新的发展契机，诸多研究人员提出了许多基于这一思想的文本生成的算法模型，上文中提到的序列生成对抗网络就是一个典型例子。然而这一思路下的文本生成依然面临着许多挑战。

首先，现有的模型成果都限于短文本的生成，即生成的文本单句内容一般短于 20 个字，长文本生成这一极具挑战的领域目前仍没有得到较大突破。这是因为文本样本序列的每个字和词之间，都存在着前后依赖的关系，但现有模型的生成器基本都存在文本特征的提取能力较为欠缺的问题，这就导致模型无法有效地进行语义和语法结构有关信息的学习，从而使得生成的长文本内容质量较差。然而长文本在许多实际场景中都十分必要，例如电商的商品描述、电影的影评，都是典型的长文本场景，长文本生成的研究有非常大的实际应用价值。

另外，以生成对抗网络为基础的所有生成算法均要解决模式崩溃这一困境，这是算法本身决定的。出现模式崩溃的生成模型，会不断生成重复的“好”样本，导致文本生成的多样性降低。这一问题目前尚无从模型层面直接解决的方案，依然需要借助研究人员对参数进行细致地调整，才能缓解模式崩溃问题的出现。模式崩溃对文本质量的影响极大，目前却又没有从根本上解决问题的完美方案，因而如何利用模型设计解决模式崩溃，使得算法能够生成多样性更好的文本，也是目前迫切需

要解决的问题。

最后，由于强化学习和蒙特卡洛树搜索本身都是计算量相当大的模型，将二者进行结合，再应用于文本生成里，计算量更是大幅增加。实际运算过程中，要经过数个循环神经网络的计算，才能组合成一个单轮的强化学习，并且随着目标生成文本的长度增长，这一计算量还会出现指数级的增加。尽可能减小计算量，能够提升运算效率，也会降低对研究设备的要求，是文本生成研究中的迫切需要。

1.3 本文的主要研究内容

对于上文中提及的现有研究仍然存在的不足，本研究对生成对抗网络和强化学习结合的文本生成模型进行了改良，主要研究在序列生成对抗网络的基础上，长文本生成的算法方案，同时尝试解决模式崩溃的问题。本研究主要进行下述几个方面的研究：第一，针对目前生成器提取文本特征能力欠缺的问题，提出生成高质量长文本的解决方案，使上下文信息能够被有效利用，提升捕获长短距离依存关系的能力；第二，通过替换损失函数的方式，提出从模型层面解决模式崩溃问题的解决方案，丰富文本生成内容；第三，将算法模型复用，在具有不同特征的文本数据集上对模型的文本生成效果进行检验，提升模型的普适性。

在完成算法模型的搭建后，本研究选择采用电商中的商品描述数据集进行长文本生成的实验，商品描述是典型的长文本实际场景，利于直观地看出模型在长文本方向的生成效果。同时，本研究为了展示模型的普适性，选择在诗歌这一任务上对模型进行生成测试。诗歌带有特殊韵律、特定格式等特征，通过对模型进行适当结构调整，使其适用于这一特殊文本类型。

1.4 本文的结构安排

本文共包含五个章节：

第一章：绪论。本章介绍了自然语言生成的研究背景，阐述了本文的研究目标和研究意义，并根据与本研究的相关性，介绍了基于传统神经网络、基于生成对抗网络、以及引入了强化学习的国内外在文本生成领域的研究。同时简单介绍了论文的研究内容与组织结构。

第二章：文本生成相关技术和理论介绍。本章详细地介绍了本研究相关的关键技术和理论基础，主要包括了经典语言模型、一些语言表征方式和生成对抗网络训练框架，旨在给后文中提出的生成模型提供必备的理论说明，让本文读者对于理论与算法模型有预备知识。

第三章：基于惩罚最小化强化序列生成对抗网络的文本生成模型。本章针对长

文本生成的任务提出了基于生成对抗网络，结合自注意力机制与惩罚目标函数的文本生成算法模型，并详细阐述生成器、判别器与训练过程。最后使用相关文本数据集，利用实验验证模型在长文本生成方面的可靠性。

第四章：生成对抗网络文本生成模型用于诗歌生成。本章首先对诗歌生成进行了介绍，借此确定了诗歌生成的评估标准。之后对上一章所述模型进行进一步优化，使其更加适用于诗歌文本的独有特征。最后用绝句数据集进行实验验证，确保本章提出的改良模型在诗歌生成中仍有较好的生成效果。

第五章：全文总结与展望。总结本文所做的工作与贡献，并提出对下一步工作可能发展方向的展望。

第二章 文本生成相关技术和理论介绍

文本生成是自然语言处理中非常核心的发展方向之一，在实际场景中的应用也相当多样。文本生成这一概念包含的范围非常广，不仅限于常见的文本生成文本，如图像生成本文、数据生成文本等，也都属于文本生成的范畴。

一个理想的文本生成框架需要许多技术支撑，最常见的文本生成文本中，也涉及非常多的技术方案，完整的文本生成包括文本理解和文本生成两部分，其中又可以细分为语料预处理、特征提取、模型训练等步骤。

随着文本生成领域不断发展，每个步骤可以用到的新算法新技术都层出不穷，本章简要介绍这些已经被广泛使用的技术，包括语言表示方法、经典的语言模型、文本数据处理技术与生成对抗框架训练等，给后文中提出的生成模型提供必备的理论说明。

2.1 语言表示方法

在自然语言处理的任务里，都会涉及对语料进行预处理的步骤，简单来说就是要把文本的内容转变形式，让计算机能够理解和处理。与图像数据不同，文本内容无法用类似像素的形式展示，因此必须进行预处理，以便用向量代替文本。并且要求转变形式后的表向量能和原文本内容实现一一对应，更高的要求是希望转变后仍然能展现文本独特的信息内容，如语言结构或语义等。

现有的语言表示方式主要有两类，过去常用的是独热表示（One-Hot Representation），目前较为常见的是分布式表示（Distributed Representation）。

2.1.1 词语独热表示

独热编码是一种十分直观又简单的方法，也被叫做一位有效编码，是最常用的方法之一。独热编码的表示方式采用了二进制的思想，所有的独热向量都是一维的二进制向量，并且向量长度就代表了整个词汇表里单词的个数，如向量长度为 N，则意味着词表大小为 N。

具体编码方法是，对词语进行编号，词汇表里的每个词语都有其一一对应的序号 i，而利用长度为 N 的 one-hot 向量表示这个单词时，将序号 i 对应的位置设置为 1，其他位置均设置为 0，此时就得到一个与该词语对应的标准独热向量。

这种方式非常简洁清晰，但是无法表现出不同单词之间的关系，例如某两个词语含义接近、某两个词语是常见组合等，因此很难确定上下文之间的关联。另外目

前常用的词语量级巨大，已有十万多，用独热表示会带来向量维度过大的问题，同样数据也是极稀疏的，都会造成非常大的存储消耗。因此，目前已经越来越少采用独热表示，取而代之的是基于分布式表示的方法。

2.1.2 词语分布式表示

词语用分布式表示的过程称为词嵌入（Word Embedding），这种表示方式把词用一个固定长度的稠密连续向量来展示，称为词向量。其理论基础是由 Harris 提出的分布假说（Distributional Hypothesis），这一假说的核心思想是，当两个词语的上下文内容十分相似时，这两个词语的语义也是相似的。

用分布式表示的优点是显而易见的，单个的词向量里，每个维度都变得更有意义，有其对应的含义，这就使得词向量里带有的信息内容更加多和详细。不同的词彼此间，可以用词向量来展示距离的概念。获得词向量则是用建立对应语言模型（Language Model）的方式，所有的建模方法都来自于同样的分布假说核心思想，即每一个词语的含义都可以利用上下文词语进行表示。一般分成两个步骤进行：第一是确定对上下文进行描述的方法，第二是选择一种模型，用来描述目标词语和上下文词语间的关系。常用的分布式表示方法有下面几个例子：

CBOW 模型[36]。又称为连续词袋模型，具体方法是要选定目标词，之后用这个目标词的上下文词语来对这个词进行预测，其中上下文单词的数量是指定的。CBOW 模型由输入层、投影层与输出层构成。输入层传递进投影层的内容是 one-hot 编码完成的向量，这些向量包含了输入的全部上下文词语。投影层则是把所有词向量嵌入矩阵 E 中，E 是一个 $m \times |V|$ 的矩阵，其中 $|V|$ 是词典所有单词的数量，m 是自定义的词向量的维度。把所有上下文的独热向量在矩阵 E 中进行投影，再进行平均运算，可以得到每个词的概率，最后把目标词语的对数似然值最大化，或是把负对数似然值最小化，通过这种方式进行模型训练。

Skip-gram 模型[37]。这种模型和上述的 CBOW 思想类似，同样是利用上下文词语进行预测，区别在于其中指定的目标词是固定范围内随机的，即在上下文词语的正负 n 个词距内随机抽取得到目标词。另外 Skip-gram 中不设置隐藏层，仍然把上下文词语进行独热编码，之后通过矩阵 E 投影得到词向量，最后把所有词向量一起输入 softmax 层，从而得到每一个单词的概率，通过这样的方法同样可以对词嵌入矩阵进行模型训练。

Word2vec 模型。这种模型可以视为在 CBOW 和 Skip-gram 的基础上进行发展，补充了霍夫曼树的核心思想，把整个输出层设置成一棵霍夫曼树，树种的叶节点是词汇表里的全部词汇，隐藏层则用非叶节点来充当。其中霍夫曼树一般都需要研究

人员进行人为的构建，构建原则依据了不同单词的使用频率，把频率较低的放在树底部，频率较高的就放在树顶部。每次需要输出词语时，就从树根出发，这样使用频率较高的词语会有更大概率被输出。训练过程中，模型只需要训练根节点到输出的叶节点这一路径上所有的节点。

Glove 模型。这种模型是建立在矩阵的基础上的一种分布式表示。一般而言，矩阵的行用每个单个单词来构建，而矩阵的列则是其上下文中的某个词，所以这一矩阵又可以称为“词-词”矩阵。把这一矩阵设为 X，内部所有元素均初始化为 0，针对某个句子，把第 i 个词语设置成中心词，并且对窗口长度 w 进行自定义，如果第 j 个词语是窗口包含的词，那么进行 $X_{ij}+1$的操作，直到把整个句子遍历完。之后把下面的公式最小化：

$$J=\sum_{i=1}^{N}\sum_{j=1}^{N}f(X_{ij})(p_i^T q_j+b_i+b_j-\log(X_{ij}))^2 \tag{2-1}$$

公式里 N 代表词典里的单词总数，p_i和q_j则分别代表了第 i 个和第 j 个词的词向量，另外b_i和b_j是预置的偏置项。Glove 模型没有使用神经网络的思想，且这种模型比 CBOW 和 Skip-gram 更复杂，所以研究人员更倾向于使用上面两种模型。

2.2 经典语言模型

要进行模型训练，需要刻画完整的语言特征或者某个特定场景中的语言体系，通过这样的语言建模可以得到语言模型。语言模型简单来说就是可以通过对某一段语句在目前存在的语言体系下出现的概率进行计算，来判断某个句子的合理性，这也是文本生成里对文本质量进行评估的基础。

以往常见的语言模型是基于规则的，包含了内容选择与表层生成两部分。但是一般需要人工对规则进行制定，并且生成的文本内容相对生硬，因而基于数据驱动的方法逐渐取代基于规则的成为主流。一般来说基于数据驱动的模型都利用了概率模型，把新生成的词语跟前文的词语建立联系，之后对词典里所有的词语进行概率计算，最后输出概率最高的那个词，本质上也是用上下文的信息去对生成词进行推测。

目前多用的语言模型有 N-gram 语言模型和基于神经网络的语言模型，另外还有些其他的模型也会在下文介绍。

2.2.1 N-gram 语言模型

N-gram 语言模型是一个典型的概率模型，模型中不考虑生成的语句是否能与人类的语言结构和语义匹配，而是直接给每个长度是 L 的句子输出一个概率值。

在 N-gram 的算法里，某个语句中第 n 个词语的生成会由前 n-1 个词语决定，通过前 n-1 个 item 对第 n 个 item 进行预测。而某个完整语句生成的几率则可以用每个词语出现的几率相乘进行表示，设每个句子的最小单位是词，并且用 w_i 来表示，w_i 表示第 i 个词，则长度为 L 的句子 $W = w_1 w_2 \cdots w_L$，这样生成该句子的概率模型可以表示为：

$$\begin{aligned} P(W) &= P(w_1, w_2, \cdots, w_L) \\ &= P(w_1) \bullet P(w_2 \mid w_1) \bullet P(w_3 \mid w_1 w_2) \bullet \cdots \bullet P(w_L \mid w_1 w_2 \cdots w_{L-1}) \end{aligned} \tag{2-2}$$

上述公式代表最新生成词语的概率和之前所有生成的词语有关，但这一条件概率难于计算，尤其是最后一项的参数十分巨大，很难进行模型训练，因此作出进一步的假设，把 n 的值取为 2 或是 3，令每个新生成词的概率只和前面的 1 或者 2 个词有关，这样的模型也可以称作二元和三元模型。建立在这样假设下的语言模型就可以称之为 N-gram 模型，最易计算的二元模型也称作 Bi-gram 模型，可以把句子概率的公式简化为：

$$P(W) = P(w_1) \bullet P(w_2 \mid w_1) \bullet P(w_3 \mid w_2) \bullet \cdots \bullet P(w_L \mid w_{L-1}) \tag{2-3}$$

另外，为了使公式更加完备，一般会人工设置一个 begin 词，作为 $P(w_1)$ 的条件概率，设置方法如下：

$$P(w_1) = P(w_1 \mid begin) \tag{2-4}$$

不难看出，所有句子被生成的概率总和应该为 1，这也就意味着，当一个句子被生成的概率越大，这个句子的合理性就越强。

不过 N-gram 语言模型也存在一些问题，例如当 n 值比较小的情况下，模型很难基于前 n-1 个词准确地判断出目前最合理的词语；而当 n 值取值过大时，虽然可以得到更多信息，但是数据也会相应变得更加稀疏，很容易看出，$P(w_i \mid w_{i-1})$ 的值会比 $P(w_i \mid w_{i-1}, w_{i-2}, w_{i-3})$ 更大，这也就导致在完整数据集里，长序列出现的频率会更低。

另外由于语料规模总是有限的，所以也会导致一些某些本身是真实存在并且合理的句子，在现有训练集里不存在，这就出现了在现有语料库里，这个 N 元词组出现的频率是 0，也称作 0 概率问题，这是非常不符合实际情况的。这一问题一般会用平滑方式解决，也就是根据训练集里已经存在的句子对集里没有出现的句子概率进行预估，或是把训练集里其他句子的频率分出一些给没有出现的句子。主流的平滑方法包括加性平滑、拉普拉斯平滑和古德图灵平滑，这之中拉普拉斯平滑是将所有有概率出现的句子频率都加 1，这样就使得原本不可能出现、概率等于 0 的事件被改变，同时对于高频的句子影响不大，非常有效且简单。

总的来说，N-gram 模型计算简单，概念清晰，经常被用来对句子的合理性与有效性进行判断，许多文本质量的评估都用到了这一思想。

2.2.2 前馈神经网络语言模型

前文中已经提到，N-gram 语言模型仍然存在一些缺陷。在 2003 年，Bengio 等人提出了前馈神经网络语言模型（Feedforward Neural Network Language Model, FFNNLM）[38]，这一模型包含了三层神经网络，是首个基于神经网络的模型，也是第一次在文本生成中引入了词向量的概念，这使得词与词之间建立起联系，能够更好地展示词之间的相似性与前后文关联。

前馈神经网络语言模型一样建立在 n-1 阶马尔可夫假设上，主要思想依然是通过目标词的前 n-1 个词对目标词的出现概率进行计算。完整的模型可以分成两个阶段，特征映射和联合概率计算。第一个特征映射模块中构建了一个词嵌入的矩阵 E，大小为$k \times |V|$，其中 V 代表词汇表，k 代表自定义的词向量的长度。令w_i是当前的目标词，需要进行计算输出，则把前期已经输入的 n-1 个词进行处理。首先对每个词进行独热编码，之后将词向量输入进嵌入层，嵌入层中每个词向量都嵌入矩阵 E，得到投影向量，此时每个独热编码向量都有了一一对应的用分布式表示的向量。之后把所有获得的分布式词向量进行拼接，产生矩阵e，并假设$E(w_{i-t})$表示第 t 个词语，则有：

$$e = \left[E(w_{i-n+1}); E(w_{i-n+2}); \cdots; E(w_{i-1})\right] \tag{2-5}$$

把拼接得到的词向量矩阵e输入进隐藏层，由$Hx + d$运算得到，式中 d 代表偏置项。经过矩阵变换以后再通过 tanh 激活函数完成非线性化，最后进入输出层。此层中可以计算出目标词在词汇表汇总每一个有可能的词语的概率，方法是把非线性化后的矩阵输入 softmax 层，通过式 2-6 计算得出每个词的概率：

$$y = \text{soft}\max(U \cdot \tanh(W \cdot e + b)) \tag{2-6}$$

公式中 U 与 W 都表示权重矩阵，b 是偏置项，这些参数均需要在训练中进行更新。

另外前文中提及的 tanh 激活函数，在模型中一般被用于非线性化处理，模型必须使用过激活函数才能很好地拟合异或问题，常见的激活函数包括 tanh、relu、sigmoid、leaky relu 等。激活函数的设计初衷是仿真生物的神经元，即神经元需要得到一定刺激才会出现明显的反应；另外激活函数也可以维持模型处在非线性环境，把输出值域控制在一个期待的范围中。

上文中求得每个词的概率之后要计算联合概率，神经网络中，可以通过训练很

好地得到从一个空间的分布到另一个空间的分布的映射函数，所以 Bengio 把条件概率视作一个映射，即从矩阵 e 的分布到词汇表里所有词概率的映射，因此可以知道第 i 个词语的概率公式为：

$$f(w_1, w_2, w_3, \cdots, w_{i-1}) = P(w_i \mid w_1, \cdots, w_{i-1}) \tag{2-7}$$

这样的算法获得的语言模型与传统的 N-gram 相比，很自然地维持了平滑，不再需要针对零概率的问题另外设计其他平滑算法。前馈神经网络语言模型还存在着许多缺点，比如有过多的训练参数存在，且允许输入的单词量是固定数值，这就导致生成能力被大幅限制，但词向量矩阵是首个使用分布式表达词向量的模型，这种模型也是首次在语言模型中使用了神经网络，对于后面的研究仍然具有极强的指导价值。

2.2.3 其他语言模型

除了上文提到的两种比较主流的语言模型以外，也有一些其他的语言模型。例如上文中 Bengio 在前馈神经网络之外，也提出循环神经网络能够设计成新语言模型的思路，循环神经网络的优势十分明显，其输入内容不仅能与当前时刻的词向量相关，还可以结合上一个时刻的隐藏状态，输出内容则是当前时刻词语的概率分布。下文将详细介绍，循环神经网络的特性会使得文本序列的长度依赖这一问题得到一定缓解。另外循环神经网络的另一特性是每个节点的参数实现共享，这就导致训练参数比较少，能得到更好的实验结果。

Mikolov 等人在 2010 年实现了这个构想，他们提出了基于循环神经网络的语言模型（Recurrent Neural Network Language Model, RNNLM）[39]。这个模型相比神经概率的语言模型，在后者基础上加了一个具有记忆能力的新单元，用来对前面时刻词语的信息进行记录。另外和循环神经网络的特性相同，这一模型把上一步里的隐藏层跟当前步里的目标词叠加，作为输入项。

在 2013 年，Mikolov 等人又提出了新的网络结构 word2vec，这一结构包括了两种把词向量嵌入矩阵的方式，分别是 CBOW 和 Skip-gram。与 FFNNLM 不同之处在于，FFNNLM 的核心目的在于把各个单词的概率求出来，生成词向量并不是主要目标，而 word2vec 的主要目的就是要训练词表征。所以 word2vec 在模型上也有许多改变，例如隐藏层被去掉，同时引入了负采样与分层 softmax 的技术。

2.3 文本数据处理技术

在自然语言处理的领域里，不管应用场景为何，最主要的任务都是必须先对输

入进来的文本数据集进行处理，即需要利用词嵌入矩阵把文本内容转化成矩阵，在此之后对这一矩阵进行后续操作，例如进行特征提取。常见的特征提取算法模型是循环神经网络（Recurrent Neural Network, RNN）[40]与卷积神经网络（Convolution Neural Network, CNN））等。

2.3.1 循环神经网络

循环神经网络最核心的能力就是对序列数据进行处理。一般序列数据中，某一部分的数据和其前后部分的数据都会有一定关联性，文本数据也满足这一特征，当前的语句中某一个词语和其前后文都会有一定关联，这也是文本中语义理解、语言结构的体现[41]。

与序列数据匹配的是，循环神经网络可以实现结合之前信息进行时刻输出，所以循环神经网络十分适合进行构建语言模型。这一模型在自然语言生成这一领域的作用主要有两方面，一是对生成文本的准确性进行评估，因为循环神经网络可以对某个生成文本序列在真实世界的文本数据中出现的可能性进行计算；二是可以直接用于生成文本序列，正因此这一模型可以被用作生成模型。

循环神经网络的每一个单元有部分地方和前馈神经网络的最小单元是一致的，输入内容依然一个词向量，且这个词向量已通过词嵌入矩阵映射。不同之处在于，循环神经网络增加了一个隐藏向量，这一向量带有自我循环链接，下一时刻的隐藏向量 a_t 会与前面时刻的 a_{t-1} 有所关联。这就使得循环神经网络可以存储短期记忆，因此其处理文本数据时会比前馈神经网络表现得更好。图 2-1 是最普通的循环神经网络结构图：

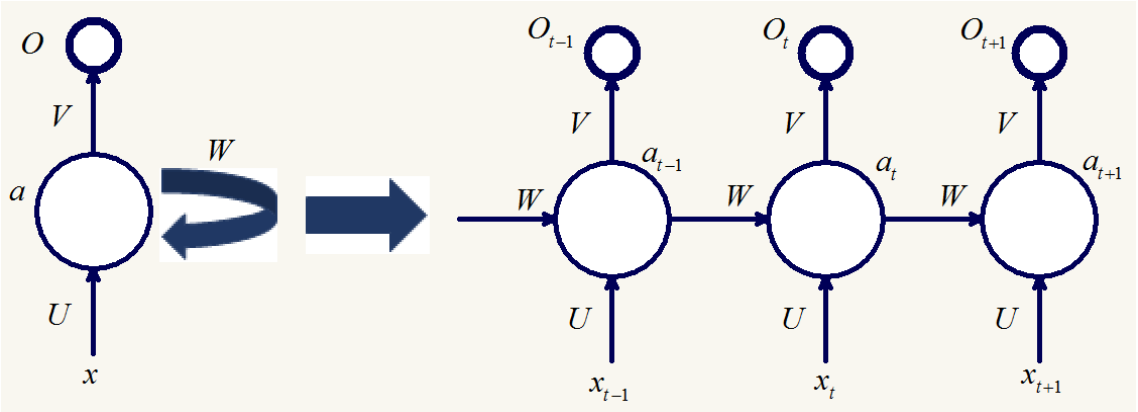

图 2-1 循环神经网络结构框图

由图 2-1 可以看出，依据循环神经网络中单个隐藏向量可以循环连接的特点，可以把它展开并视为一个有 L 层隐藏层的深度模型，其中 L 代表输入进来的文本长度。此外，循环神经网络与全连接的神经网络最突出的不同之处在于，其每一个

时刻的单元权重都可以共享，这就意味着其每个步骤都在重复相同的工作，所以使用循环神经网络就可以大幅减少需要用到的训练参数。令一个输入句子长度为 T，其序列为 $X_{1:T}=(x_1,x_2,\cdots,x_T)$，那么每个时刻 t，都将当前的词向量 x_t 输入循环神经网络单元里。其中词向量可以用独热编码生成，也可以用词嵌入矩阵映射生成。则通过下方公式，可以计算求得隐藏向量 a_t：

$$a_t=g_1(w_a^a\cdot a_{t-1}+w_x^a\cdot x_t+b_a) \tag{2-8}$$

其中 w_x^a 代表输入向量至隐藏向量的变换矩阵，w_a^a 代表隐藏向量至隐藏向量的变换矩阵，g_1 代表激活函数，常见的是上文介绍过的非线性函数，g_1 一般使用的是 tanh 或 Relu 函数。

在每一个时刻，计算得到隐藏向量 a_t，一来可以直接当做下一时刻的输入信息，二来可以经过计算再作为现在时刻的输出内容，现在时刻输出的计算公式为：

$$a_t=g_2(w_a^a a_t+b_o) \tag{2-9}$$

式中 g_2 可以代表另外一个神经网络，也可以代表另一个激活函数，用来把变换得到的隐藏向量直接映射进目标空间里。

将循环神经网络应用在文本生成的过程里，可以在上一时刻的词确定之后，求得当前输出每个词语的概率，这使得循环神经网络可以度量每个生成句子的准确性，并且对输出概率的分布完成采样生成文本，损失函数使用极大似然估计。相比于前馈神经网络同层中各个神经元毫无关联，循环神经网络中则利用之前时刻的输入对当前的输出产生了影响[42]。

在实际情况中，由于损失函数使用的是极大似然估计，则训练的过程中会产生梯度，神经元的参数调整也会跟随梯度的变化而变化。那么，在反向传播过程中，当前神经元跟随梯度调整的情况也会受到前期神经元的调整影响，当前期神经元的调整幅度非常小的情况下，损失函数对于当前神经元调整的幅度会变得更小，直接导致指数级的缩减，以至于后期的神经元不再有可以学习的方向，这就是循环神经网络非常显著的一个缺点：梯度消失。

为了改善梯度消失这一情况，有两种优化方式相继被提出，二者分别是 LSTM（Long Short Term Memory）和 GRU（Gated Recurrent Unit），以实现序列长期的记忆。这两种模型的核心思想都是通过计算再隐藏向量 a 里完成一些信息的删除或者增加。

LSTM 在每个神经元内部另外增加了一个用来记录当下神经元状态的向量，与原来已有的隐藏向量 a 一样，这个状态变量会跟随神经元递归循环连接，同时在推导的过程里，会有线性运算的过程。然而真正能够对隐藏向量 a 进行添加或删除

操作的是一种叫做“门”的结构，LSTM 的三个连续神经元的常见结构如图：

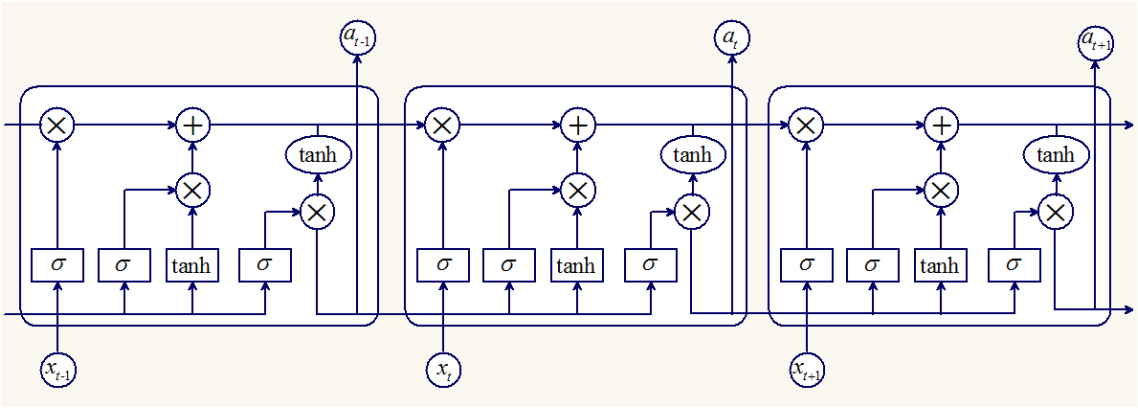

图 2-2 LSTM 神经元结构框图

门是 LSTM 的典型组成结构，每个神经元中都有三个门，依次是输入门、遗忘门和输出门，都用 0 到 1 之间的数表示。用 I_t 、 I_t、F_t、O_t 、O_t 来表示神经元在当前时刻的三个门，分别进行介绍。首先是输入门，意味着会允许多少新信息进入当前神经元的计算过程里，其计算公式如下：

$$I_t = \sigma(W_i \cdot x_t + U_i \cdot a_{t-1} + b_i) \tag{2-10}$$

其次是遗忘门，遗忘门控制了要把多少前期的信息进行丢弃，它的值在 0 到 1 的范围内，随着值增大代表保留量需要增加。另外这个门也会获取之前时刻中的当前输入和隐藏向量。遗忘门的计算公式如下：

$$F_t = \sigma(W_f \cdot x_t + U_f \cdot a_{t-1} + b_f) \tag{2-11}$$

进入最后一个输出门之前，还需要一个步骤，就是计算出目前所有备选的神经元的状态，备选代表着并不是所有信息都会转化成最终真正的神经元状态，要取决于之前的输入门才能判断，此处的激活函数大多会使用 tanh 函数，它的计算公式如下：

$$\tilde{C}_t = \tanh(W_c \cdot x_t + U_c \cdot a_{t-1} + b_c) \tag{2-12}$$

输出门会控制完成计算之后的当前神经元状态，要有多少信息会被保存以及作为隐藏信息输入进下一个神经元参与计算，整个流程计算公式为：

$$\begin{aligned} O_t &= \sigma(W_f \cdot x_t + U_f \cdot a_{t-1} + b_f) \\ C^t &= I_t \cdot \tilde{C}_t + F_t \cdot C^{t-1} \\ a^t &= O_t \cdot C^t \end{aligned} \tag{2-13}$$

从上述公式可以知道，LSTM 的每个神经元里，都会经过非常大量的门相关计

算，如果输入的序列较长，训练时间会很慢。所以 LSTM 也衍生出了许多变形，其中最受关注的就是 2014 年提出的 GRU。GRU 在 LSTM 的基础上，不仅同样可以满足长期记忆，还大大降低了“门”的数量，这就大幅减少了需要训练的参数[43]。同时 GRU 里神经元的状态本身就等同于隐藏向量，可以减少删除相关信息的操作。

2.3.2 卷积神经网络

卷积神经网络也是特征提取算法的一类，这一人工神经网络最初被提出是希望能够对生物视觉系统实现模拟，属于特殊前馈神经网络，主要的应用场景在计算机视觉领域，包括图像识别、图像生成和图像检索等。

1998 年，Yann LeCun 等人首次设计在数字手写体的识别中应用 CNN 算法[44]，并且识别正确率超过 99%。在这以后，有更多研究人员开始将 CNN 应用于图像分类中，其中比较著名的网络有 LeNet 和 Alex-Net[45]。CNN 在计算机视觉领域的突出效果使得诸多学者开始尝试将其应用在自然语言处理领域中。Yoon Kim 等人在 2014 年将 CNN 应用在文本分类任务中[46]，整个模型包含了卷积层、池化层与全连接层。

卷积神经网络的步骤里包含了卷积操作，这也是它与其他神经网络相比最特殊的部分。卷积操作一般都在卷积层中被应用，这一操作可以简单描述为，需要被卷积的矩阵通过卷积核来按照固定的步长进行滑动，把矩阵里跟卷积核重叠的元素和卷积核里一一对应的部分进行运算，得到的数值就构成卷积以后新矩阵的元素，其中卷积核也是矩阵，且一般维度会小于等于被卷积的矩阵。可以把卷积核比喻成眼睛或是聚焦的窗口，在卷积的过程里可以对当前聚焦的位置进行特征提取，要提取不同位置的特征，可以通过滑动卷积核窗口来实现，最后把得到的特征合并成一个新矩阵。

令一个句子 $s=\{w_1,w_2,\cdots,w_L\}$，这个句子在通过词嵌入矩阵投影后，得到一个表示为 $W\in R^{L\times d}$，其中 d 是词向量维度，那么这时候的卷积核由于某个维度与词向量的维度相同，以及可以沿着另一个维度滑动，则可以表示为 $K=R^{k\times d}$，这个操作可以视作对前后相邻的 k 个词语提取特征，抽象计算的公式如下：

$$V=s\otimes K$$
$$V_i=sum(W_{i:i+k-1}\odot K) \tag{2-14}$$

上式中，符号 $\odot$ 代表矩阵中的对应元素相乘，sum 运算代表了矩阵里元素求和，V_i 表示了卷积操作以后 V 的第 i 个元素的值。一般来说，卷积算法模型都会选择许多大小一致的卷积核，完成多次卷积操作，来提取出不同特征。卷积层与全连接层相比，一个显著优势是卷积核的参数可以共享，同时支持并行运算，所以与

RNN 网络或全连接网络比起来，有更快的训练速度。

卷积层之后的一般是池化层，可以分成平均池化层与最大池化层，二者分别的作用是获取在一定区域里数值的平均值与最大值。处理文本数据时，一般会选择最大池化层，以便得到卷积操作之后的特征向量里最为突出的特征值，计算过程可以抽象成：

$$V_{pool} = Max\_Pool(V) \tag{2-15}$$

选用最大池化层一来可以排除掉较弱特征的干扰，二来可以缩小矩阵的大小，另外还能够实现避免过拟合。

这些年来，研究人员使用卷积神经网络组成了不少有亮眼表现的算法框架，比如 VGG[46]、DesNet[47]等等，都在自然语言处理领域里有了大范围应用。

2.4 生成对抗框架训练

生成对抗网络在计算机视觉的领域取得了卓越表现，这使得越来越多研究人员尝试将其应用在自然语言处理里，本章将介绍常见的生成对抗框架，与自然语言生成里常用到的改进模型。

2.4.1 标准生成对抗网络

生成对抗网络的本质其实是一个框架，而非模型，这个框架中包括了生成器和判别器两个结构，而这两个结构中具体使用什么网络则需要另外制定。一般我们会用 $G(\bullet)$ 表示生成器结构，用 $D(\bullet)$ 表示判别器结构。 $G(\bullet)$ 的输出是其生成的图像或文字向量，这些会变成 $D(\bullet)$ 的输入，通过这样的方式将生成器与判别器连接起来。整个训练过程里，生成网络的目标是生成足够逼近真实样本的数据，然后把这些可以以假乱真的数据和真实数据一起输入判别器，以达到判别器无法分辨的效果。

但如果需要生成网络能够生成更接近现实世界数据的样本，同样需要一个有强判别能力的判别网络来支持，判别网络的目标是要准确地区分生成数据与真实数据。所以，一般判别器网路都会采用二值分类器，判别器的反馈可以转而激励生成器，整个生成对抗网络的一般框架如图 2-3 所示。

生成对抗网络的核心思想是二人零和，这一概念来自于博弈论，零和代表着彼此博弈的双方获得利益总和是一个固定的值，这就意味着，当需要对博弈中某一方进行训练时，要将利益尽量向那一边靠拢。在以生成对抗网络为基础的文本生成算法模型里，目标是让生成器尽可能地生成质量高的文本，因此对于判别器而言，目标就是达到让判别器无法区别真实文本与生成器生成的“假”文本。简单来说，当

判别器面对任意文本，判断是否是真实文本的概率都逼近 0.5 时，就意味着生成器的训练十分成功。

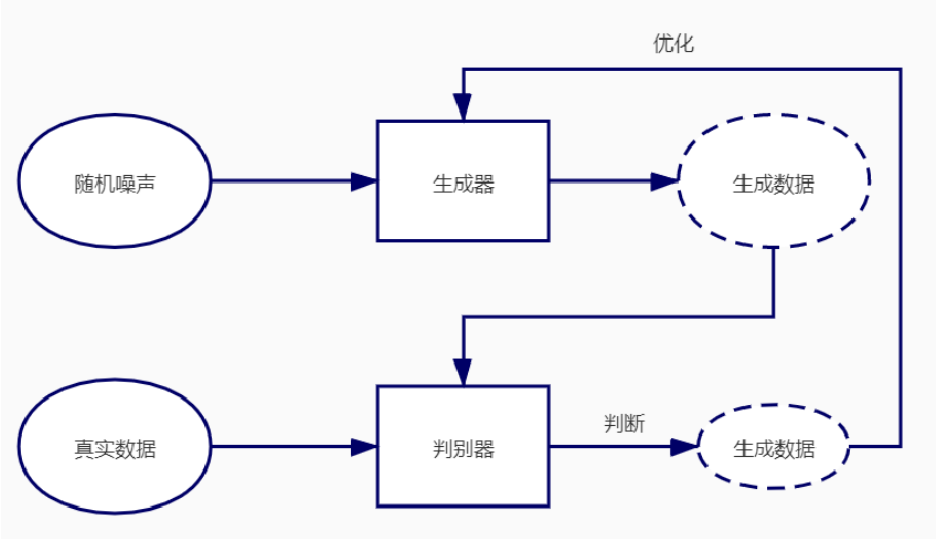

图 2-3 生成对抗网络框架

将输入生成器网络的先验随机噪声用 z 表示，p_z 代表噪声的先验分布，这一分布一般符合正态分布或符合均值分布。此时可以用 $G(z;\theta_g)$ 表示生成器，其中 θ_g 代表生成器的参数，此处假设生成器生成数据最终的分布满足 p_g，真实数据 x 的分布则符合 p_{data}，那么生成器训练的目标就是让 p_g 尽可能地拟合 p_{data}。所以生成对抗网络的训练本质上是在训练生成网络模型，这个模型会把先验噪声的分布映射到真实数据的分布。另外可以用 $D(x;\theta_d)$ 表示判别器，其中 θ_d 代表了判别器的参数，这一参数的真实作用在于用 JS 散度来度量生成数据分布与现实世界数据分布彼此间的距离，所以生成对抗网络的训练目标可以用下面公式表示：

$$\min_G \max_D (D,G) \tag{2-16}$$

式中 $V(D,G)=E_{x\sim p_{data}}\left[\log(D(x))\right]+E_{x\sim p_x}\left[\log(1-D(G(z)))\right]$ 代表判别器判断的两分布之间的距离，判别器的训练目标是让 $V(D,G)$ 的值尽量大，所以当数据的分布满足了连续概率分布的时候，通过概率密度函数与期望的转换可以得到下式：

$$V(D,G)=\int_x p_{data}(x)\cdot\log(D(x))+p_g(x)\cdot\log(1-D(x))dx \tag{2-17}$$

那么固定生成器的参数指标以后，再对 $V(D,G)$ 取极值，就可以得到判别器的解：

$$D_G^*(x)=\frac{p_{data}(x)}{p_{data}(x)+p_g(x)} \tag{2-18}$$

由于针对所有不全为 0 的实数 a 和实数 b，类似 $y=a\log(y)+b\log(1-y)$ 形式的

函数，一般在介于 0-1 之间的数 $\frac{a}{a+b}$ 上能取得极大值，对判别器的训练过程可以视作对条件概率 $P(Y=y|x)$ 进行对数似然估计的估算值。那么有：

$$\max_{D} V(G,D) = V(G,D_G^*)$$

$$V(G,D_G^*) = E_{x\sim p_{data}}\left[\log D_G^*(x)\right] + E_{z\sim p_z}\left[\log(1-D_G^*(x))\right]$$

$$V(G,D_G^*) = E_{x\sim p_{data}}\left[\log\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + E_{z\sim p_z}\left[\log\frac{p_g(x)}{p_{data}(x)+p_g(x)}\right] \tag{2-19}$$

此处由于目标是让 p_g 的分布和真实数据的分布 p_{data} 相同，那么有 $D_G^*(x)=0.5$，则上述公式可以转化得到：

$$V(G,D_G^*) = -2\log(2) + KL(p_{data}\,\|\,\frac{p_{data}(x)+p_g(x)}{2}) + KL(p_g\,\|\,\frac{p_{data}(x)+p_g(x)}{2})$$

$$V(G,D_G^*) = -2\log(2) + 2\cdot JSD(p_{data}\,\|\,p_g) \tag{2-20}$$

由上述公式可以看出，在 JSD 函数的值达到 0 时，两个分布达到完全相同，意味着生成器的训练此时已经完成了；而当 JSD 函数的值为 log2 时，两个分布达到完全不同。

在单一的循环神经网络和前馈神经网络里，一般要确定一个损失函数，用于对生成数据和现实世界数据的分布距离进行度量，这个损失函数还需要能够很好地传递梯度。在生成对抗网络中，判别器就扮演了损失函数的角色，不仅可以通过二值反馈产生的梯度传递协助生成器完成参数更新，也可以利用生成数据与真实数据达到有监督训练，借此完成自我更新。但是传统的生成对抗网络建立在 JS 散度的基础上，当真实数据和生成数据的分布不同时，JS 散度能取到的最大值只能达到 log2，但一般情况下判别器的训练会好过生成器，这时就会导致生成对抗网络的梯度消失，即判别器只能反向传播给生成器非常小的梯度，以至于生成器无法得到充分有效的训练。

2.4.2 WGAN 和它的改进版本

由于上文中提到的生成对抗网络存在的问题，Arjovsky 等人提出了 Wasserstein GAN[48]，这个生成对抗网络的变形使用 Wasserstein 距离来计算两个分布的距离，取代了之前的 JS 散度。Wasserstein 距离也叫做推土距离（Earth-Mover`s Distance, EMD），举个例子，设两个特征集合分别为 p 和 q，其中 p_i 代表第 i 个特征，p 中

的每个特征都有一一对应的权重值w_i。同样的，在集合 q 里第 j 个特征表示为q_j，与之对应的权重是w_j。此处还会设一个代价矩阵 C，用C_{ij}代表由p_i转换成q_j的代价。此时的目标就是用最小的代价来把 p 中所有特征转换成 q，进而获得全局最小化的代价函数，可以用下述公式表示：

$$Work(P,Q,T)=\sum_{i=1}^{m}\sum_{j=1}^{n}C_{ij}T_{ij} \tag{2-21}$$

式中 T 代表传输矩阵，T_{ij}表示从 p 集合中的第 i 位向 q 集合中的第 j 位的传输量。Wasserstein 距离的定义由这一概念处理而来，具体公式为：

$$W_C\left[p_1(x),p_2(x)\right]=\inf_{\gamma\in\prod(x\sim p_1,y\sim p_2)}E_{(x,y)\sim y}\left[C(x,y)\right] \tag{2-22}$$

式中$C(x,y):\chi\times\chi\to R^n$，$\chi$是包含了 x 和 y 的集合。上式的计算量相当巨大，当使用的代价函数是$\|x-y\|$时，WGAN 可以把原先的 Wasserstein 距离调整成对偶形式表示，且对偶形式与原形式是等价的，公式如下：

$$W\left(p_1(x),p_2(x)\right)=\sup_{\|f\|_L<1}E_{x\sim p_1(x)}\left[f(x)\right]-E_{x\sim p_2(x)}\left[f(x)\right] \tag{2-23}$$

公式中$\|f\|_L$表示 Lipschitz 范数，由于神经距离 f 被推土距离所约束，这种时候就要求必须符合 1-Lipschitz 的约束，而 K-Lipschitz 约束可以表示如下$\|f(x_1,\theta)-f(x_2,\theta)\|\le K\|x_1-x_2\|$。

通过公式能够看出，设置 Lipschitz 是为了保证函数 f 的偏差固定在一定范围内，即便两个对象的波动较小，f 的偏差也不会太大，损失函数不会再训练过程中变成负无穷大。只要满足了 1-Lipschitz 的约束，训练过程中就能确保稳定性。Arjovsky 等人用构建神经网络来表现函数 f，以求解函数 f 的参数。

在模型总架构中，f 可以视为特殊的判别器，输入真实样本的时候，f 的值会更大，输入生成样本的时候，期待得到的结果更小。生成器的训练目的是把生成分布与真实分布的计算距离缩小，与判别器刚好相反。这样整个模型就变成了一个基于 Wasserstein 距离的最小最大问题，用公式表示为：

$$\arg\min_{G}\arg\max_{f,\|f\|_L<1}\frac{1}{N}\sum_{i=1}^{N}\left[f(y_i,\theta_f)-f(G(z_i),\theta_f)\right] \tag{2-24}$$

式中 N 代表同一个批次里样本的总量，y_i代表真实的样本数据，$G(z_i)$代表生成的样本数据。生成器的训练可以通过把两个分布的距离$W\left(p_1(x),p_2(x)\right)$最小化来获得。

上文中已经提到，WGAN 采用权重剪裁，以保证函数 f 可以满足 Lipschitz 的约束。方法是设一个实数 c，其值大于 0，如果权重 w 的绝对值超过了 c，就要对

w 进行裁剪，变成 c 或-c。这是一种很简单的方式，但同样非常机械，只能限制网络 f 的波动不会超过某个线性函数。

后续对于 WGAN 的改进很多都集中在改进约束神经网络 f 的方式，有一种常见的方式是谱归一化，相比于权重裁剪来说，谱归一化会在每一层都进行裁剪操作，而非所有梯度完成以后再裁剪。谱归一化的理论依据更加完善，但约束的方式和权重裁剪一样机械。

Gulrajani 等人提出了梯度裁剪的方案，这一方案选择直接把函数 f 生成的梯度产生一个惩罚项，再加入 f 的训练函数里，这一惩罚项的公式表示如式 2-25：

$$\lambda E_{x\sim\gamma(x)}\left[(\left\|\frac{\partial f(x,\theta_D)}{\partial x}\right\|-1)^2\right] \tag{2-25}$$

式中 λ 代表惩罚的幅度，由于 Lipschitz 的约束里 K 值为 1，因此式中减 1。$\gamma(x)$ 代表了一个过渡空间，介于真实样本与生成样本之间。Wu 等人提出了 WGAN-div，这种模型里引入了一个新散度，称为 Wasserstein 散度。

这种散度在训练过程中，先把基于 Wasserstein 散度上真实数据与生成数据的分布距离最大化，以获得函数 f，之后同样用最小最大问题公式对生成器进行训练，因为约束问题不再考虑，所以这种方式的稳定性比上述几种方式更好。

WGAN 的优势非常明显，它的特性决定了它能够很好地处理离散数据，适合应用在文本领域。另外 Wasserstein 距离缓解了标准生成对抗网络的训练问题，基本的生成对抗网络需要对生成网络 G 与判别网络 D 的训练速度进行调控，否则判别网络的训练程度可能远高于生成网络，就会导致生成的梯度为 0，以至于生成器失去训练方向等等。而 WGAN 不用考虑这种训练的差距，判别器的训练程度更高，获得的 Wasserstein 距离会更可靠，生成器的训练也会更好。

尽管 WGAN 具有很多优势，但其作为生成对抗网络的本质问题并没有解决，即依然没有解决采样过程不可导的问题，应用于文本生成时，梯度更新不能实现回传。因此 Kusner 等人提出了 GSGAN 这一新模型，通过 Gumbel-softmax（GS）分布来解决上述问题。这一模型可以利用 GS 分布无限接近于原始分布，这就保证了生成器与判别器都可以完成基于梯度的参数更新，能够实现文本序列生成。但由于 GSGAN 对于生成文本和真实文本的判别仍然使用 JS 散度，所以模型难以训练和模式崩溃的问题依然没有得到解决。

2.5 本章小结

本章主要介绍了与文本生成相关的理论基础与关键技术，首先是主流的词向

量训练方式，简单介绍了自然语言处理里的经典语言模型。在这一基础上，详细介绍了多用于文本序列数据处理的循环神经网络与卷积神经网络结构。之后介绍了生成对抗网络的优势，以及传统的生成对抗网络应用在语言模型中时，会面临的梯度消失与模式崩溃等问题。为了解决这些问题，WGAN 在此基础上进行了改进，本章对于 Wasserstein 距离等改进方案做了详细描述。最后简略介绍了 WGAN 之后出现的优化模型，并简要分析其优劣势。

基于这些理论知识，本文针对文本生成这一领域提出了一种新的生成模型，来改进目前尚存的问题，以生成质量更高的文本。

第三章 基于惩罚最小化强化序列 GAN 的文本生成模型

第二章中介绍了与文本生成有关的各类技术，其中卷积神经网络和循环神经网络都可以在处理文本数据时使用，但是由于其概率模型基于极大似然估计，曝光偏差与过拟合的问题都很难避免，因此研究人员在文本生成领域引入了生成对抗网络。

将生成对抗网络直接在文本生成中使用并不可行，因为文本生成算法在输出之前都需要对输出向量进行采样提取，得到最大概率的序号，再利用词嵌入矩阵转化为词向量，这一过程不可微，与生成对抗网络的基本要求不符。同时，原始的生成对抗网络在输入数据离散的情况下，无法正常传递梯度，导致生成器失去更新方向。序列生成对抗网络通过引入强化学习方法，克服了数据离散的难题，使得生成对抗网络框架可以用在文本生成中，但也同样面临模式崩溃与提取文本特征能力欠缺的问题。

针对上述问题，本文提出了新的生成对抗网络文本生成算法，引入了结合高斯偏差的自注意力机制，将其作为生成网络，提高捕获文本特征的能力。另外目标函数使用基于惩罚的机制，改善了模式崩溃问题。这些改进使得生成的文本在多样性与质量上都有所提升。

3.1 序列生成对抗网络理论

序列生成对抗网络（Sequence Generative Adversarial Nets, SeqGAN）源自于生成对抗网络，是它的一种结构变体，由 Lantao Yu 等人提出。这一模型仍然采用了生成对抗网络中对抗的核心思想，改变之处是在原模型的基础上增加了强化学习的理念。

前文中已经提过，生成对抗网络使用的是经典的无监督学习方式，意即不会对数据实现进行建模，机器需要在训练过程中自主摸索数据所含的规律。这一模式比起传统的有监督学习方式而言，更能够凸显机器学习的本质，训练其“智能化”。并且，生成对抗网络已经在计算机视觉的许多研究方向中有过成功的落地实现，如果能将这一算法模型应用在自然语言处理领域，势必能推动整个领域的发展。

尽管在诸多领域得到了显著成果，但在自然语言处理领域里，想要落地使用生成对抗网络仍然有很多困难。这是因为整个 GAN 模型主体分为两个部分，分别是生成器和判别器。想要构建生成网络与判别网络，需要有一个可微分的函数，这是因为生成对抗网络的参数更新基于连续空间，也就是说使用这一网络的前提条件

是数据连续。然而语言的数据是离散的，文本与图片数据不同，并不是连续可微分的，因此当试图将生成对抗网络应用在自然语言处理的问题解决中时，并没有取得较好的效果。

有许多研究尝试着解决这一问题，其中包括了 Gumbel-softmax 分布等。但引入强化学习的策略梯度概念，是这一问题更加常见的解决方案，这其中最为经典的一个模型就是 SeqGAN。

序列生成对抗网络这一算法模型的核心理念，是把生成对抗网络与强化学习相结合，将一个完整的生成对抗网络视作强化学习的系统，模型里参数的更新则借助于策略梯度算法来实现。同时，也借鉴了蒙特卡洛树搜索（Monte Carlo Tree Search, MCTS）的概念，借此来实现对任何某个时间点的不完整序列完成实时评估。当然序列生成对抗网络的算法本身仍然保持了生成对抗网络算法模型中生成器和判别器的架构，算法框架如图 3-1：

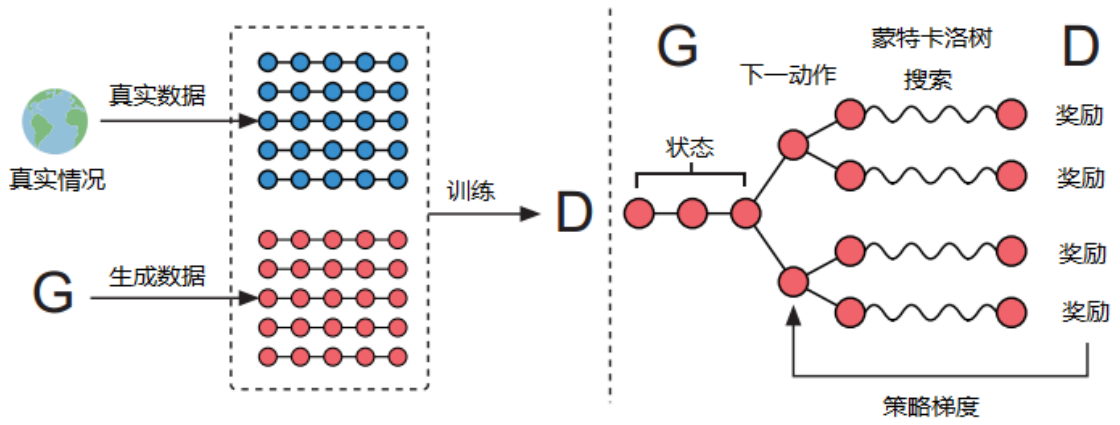

图 3-1 seqGAN 算法框架图

算法的详细过程可以描述如下：首先给定一个真实序列的数据集，并用这个数据集训练一个参数为θ的生成器G_θ，这个生成器用于生成一个序列：

$$y_{1:T} = (y_1, y_2, \cdots y_t, \cdots\cdots y_T), y_T \in Y \tag{3-1}$$

其中 Y 代表用来生成序列单元的词典。结合强化学习的概念可以解释为，在某一时刻 t，现在已经生成的序列$(y_1, y_2, \cdots y_{t-1})$就是状态，而下一个计划输出的单词$y_t$动作。另外，也要训练一个参数是$\phi$的判别器$D_\phi$，其目的是判断生成得到的序列是否来自于真实数据集，从而引导生成器G_θ的改进优化，其中用$D_\phi(y_{1:T})$表示生成的序列$y_{1:T}$是真实数据的概率。

在序列生成对抗网络的算法模型中，生成器选择的模型架构是长短期记忆网络模型 LSTM。针对输入的文本序列，首先要把文本的表达形式转变为词向量，之

后再继续输入到网络的每一个单元中，最后与全连接隐藏层结合，输出获得下一个生成的单词。

判别器采用的模型是卷积神经网络（Convolutional Neural Networks, CNN），针对整个序列而非未完成序列进行判别分类。算法思路如下：将原数据集文本序列表示为$x_1, x_2, \cdots, x_T$，将目标生成文本序列表示为$y_1, y_2, \cdots, y_T$，之后分别构建源数据矩阵与目标数据矩阵如下：

$$X_{1:T} = x_1; x_2; \cdots; x_T \tag{3-2}$$

与：

$$Y_{1:T} = y_1; y_2; \cdots; y_T \tag{3-3}$$

其中$x_t, y_t \in R^k$为 k 维词嵌入，分号是拼接算子。对于源矩阵$X_{1:T}$，内核$\omega_j \in R^{l \times k}$对尺寸为 1 单词的窗口进行卷积运算，并生成一系列特征映射：

$$c_{ij} = \rho(\omega_j \otimes X_{i:i+l-1} + b) \tag{3-4}$$

其中⊗运算符是所有元素产量相加所得的和，b 代表了偏差项，ρ代表了非线性激活函数，在本文后面的研究里被实现为 ReLu。为了获得有关内核ω_j的最终功能，在功能图上利用了最大时间池化操作：

$$\tilde{c}_j = \max\left\{c_{j1}, \cdots, c_{jT-l+1}\right\} \tag{3-5}$$

本研究使用具有不同窗口大小的多种内核来提取不同的特征，然后将它们组合起来以形成原数据集句子表示形式c_x。相同地，可以从目标矩阵$Y_{1:T}$中提取目标生成句子表示c_y。最后，在给定源数据库文本语句的情况下，目标文本语句是真实的概率可以计算为：

$$p = \sigma(V\left[c_x; c_y\right]) \tag{3-6}$$

其中 V 是将c_x和c_y的并置转换为二维嵌入的变换矩阵，而σ是对数函数。

这一计算得到的概率值会作为算法中的奖励值，并且使用前文中提到的策略梯度算法（Policy Gradient）把这一概率值回传到生成器，借此来对生成器的参数实现更新。通过这个方法，上文所说的文本这一类离散数据无法使用判别器的反向传播来实现参数更新的问题就得以解决。在生成器生成的序列不是完整的序列时，可以把生成器视为策略，在某一时刻t，生成器可以表示为$G_\theta(y_t \mid y_{1:t-1})$，而策略的目标是使从初始状态$s_0$（之前的$t-1$时刻生成的序列）到生成了完整序列时的奖励是最大的。在t与T时刻间的序列，要用蒙特卡洛树搜索来实现，采样的策略与G_θ一致，将其设为G_ϕ：

$$\left\{y_{1:T}^{1}, y_{1:T}^{2}, \ldots, y_{1:T}^{N}\right\} = MC^{G_\phi}(y_{1:t-1}; N) \tag{3-7}$$

在此之中，$\left\{y_{1:T}^{1}, y_{1:T}^{2}, \ldots, y_{1:T}^{N}\right\}$ 代表了搜索过程中生成的 N 种有可能的序列，对所有序列的判别概率再取均值，就可以求得整个序列的价值函数：

$$Q_{D_\phi}^{G_\theta}(y_t = y_T, s = y_{1:T-1}) = \begin{cases} D_\phi(y_{1:T}), & t = T \\ \dfrac{1}{N}\sum_{n-1}^{N} D_\phi(y_{1:T}^{n}), y_{1:T}^{n} \in MC^{G_\phi}(y_{1:t-1}; N) & t < T \end{cases} \tag{3-8}$$

上述即为序列生成对抗网络模型中生成器的原理，而判别器的原理是采用了类似逻辑回归来计算出损失的对数函数，并令其得到最小：

$$\min_{\phi} -E_{y\sim pdata(x)}\left[\log D_\phi(Y)\right] - E_{y\sim G_\theta}\left[\log(1 - D_\phi(Y))\right] \tag{3-9}$$

简单概括来说，判别网络 D_ϕ 借助用现实世界数据与生成网络 G_θ 输出的生成数据获得，生成网络 G_θ 借助策略梯度（policy gradient）和经过判别网络 D_ϕ 取得的蒙特卡洛搜索奖励（MC search reward）来实现更新。其中需要最大程度地混淆判别器 D_ϕ，这就是奖励的计算依据，也就是尽力地让 D_ϕ 以为 G_θ 输出的数据是真实的数据。

算法中使用到的策略梯度，其基本思想就是将反馈参数设定为 reward 值，把得到 reward 较大的行为出现的几率提高，把得到 reward 较小的行为出现的几率降低，因此求得 reward 就可以进行梯度训练，从而更新参数。在 seqGAN 算法中，由于用到策略梯度算法，因此在生成器每产生一个新参数时，就可以得到一个反馈的 reward，之后便可以更新生成器的参数，相比起基础的 GAN 需要借助判别网络的反向传播来实现参数的更新，seqGAN 摆脱了对判别器的依赖；而蒙特卡洛搜索的运用也保证算法可以立刻评估当前单词的好坏，而不必等到所有序列生成后才可评价。正因此，seqGAN 才解决了上文中提到的原始 GAN 应用于文本生成时会遇到的问题，可以生成高质量的文本内容。

seqGAN 相较于 GAN 的原始算法模型而言，在文本生成领域已经取得了更好的效果，但 seqGAN 仍有其缺陷，seqGAN 模型的生成器采用了 LSTM 模型，这种架构限制了算法的并行化。本研究的优化方向之一也正是集中于这一方面的问题，基于自注意力的 Transformer 可以改善这一点。

3.2 基于自注意力机制的文本依赖关系建模方法

3.2.1 自注意力机制理论

自注意力是一种关注机制，能够简要概括为将查询（query）和键（key）都以

向量的形式进行归一化处理，之后将输出计算为值（value）的加权综合。最近的一些研究证据表明，自注意力机制在自然语言处理中的各种任务上表现良好，如机器翻译、文本分类等。

Transformer 是一种以编码器-解码器形式完成的算法模型[49]，如果要从本质上深入理解注意力机制的概念，需要将其从 Encoder-Decoder 的框架里剥落出来。首先将一系列<Key, Value>的数据对构成的内容视为数据源（Source），上文中提到的查询（query）、键（key）和值（value）是键值查询的三个基本元素。完整的计算注意力数值的过程可以总结为：首先求出所有查询和每一个键的相关性，从而求得各个键对照的值的权重系数，之后还要把求得的权重值与对应的键值进行加权求和。因此，注意力机制本质的思想可以总结为一个查询到一个键-值对的映射。

在编解码器模型中，主要是将上述的注意力机制应用于目标文本与源文本之间。而自注意力机制（Self-Attention）与此不同，其主要存在于源文本序列之间，或是目标文本序列内部，作用是抽取同一个文本数据内部间距比较远的字词之间的联系。对比来说，在传统的循环神经网络中，如果文本序列的长度不断增加，训练网络在学习到靠后部分的字词时，靠前部分的字词相关的记忆信息就会越来越少，并且一般伴随着计算时间的增长，这就是循环神经网络的不足之处。如果在序列的内部加入了自注意力机制，借助其在序列内部构建的长距离依赖关系，就可以很直接地计算得到文本中任意两个字词向量之间彼此相互依赖的特征。

Transformer 模型就是建立在自注意力基础之上，其编码器是由六个相同层组成的，每一层包括了一个自注意力和一个前馈神经网络。译码器也同样包括六个相同层，每层中除了与编码器中完全一样的两个子层以外，解码器还插入了第三个子层，称为编解码器注意力。图 3-2 是为 Transformer 的结构图。

形式上，给定一个输入序列 $x=\{x_1,\cdots,x_I\}$，每个 l 层的隐藏状态都是通过关注第(l-1)层构造的。具体来说，第(l-1)层 $H^{l-1}\in R^{I\times d}$ 先变换为查询（$Q\in R^{I\times d}$）、键（$K\in R^{I\times d}$）与值（$V\in R^{I\times d}$）与三个独立的权值矩阵。第 l 层计算方式为：

$$H^l = ATT(Q,K)V \tag{3-10}$$

其中 ATT(·)是点乘注意力模型，定义如下：

$$ATT(Q,K) = softmax(energy) \tag{3-11}$$

$$energy = \frac{QK^T}{\sqrt{d}} \tag{3-12}$$

其中 $\sqrt{d}$ 是尺度因子，d 是层次状态的维数。

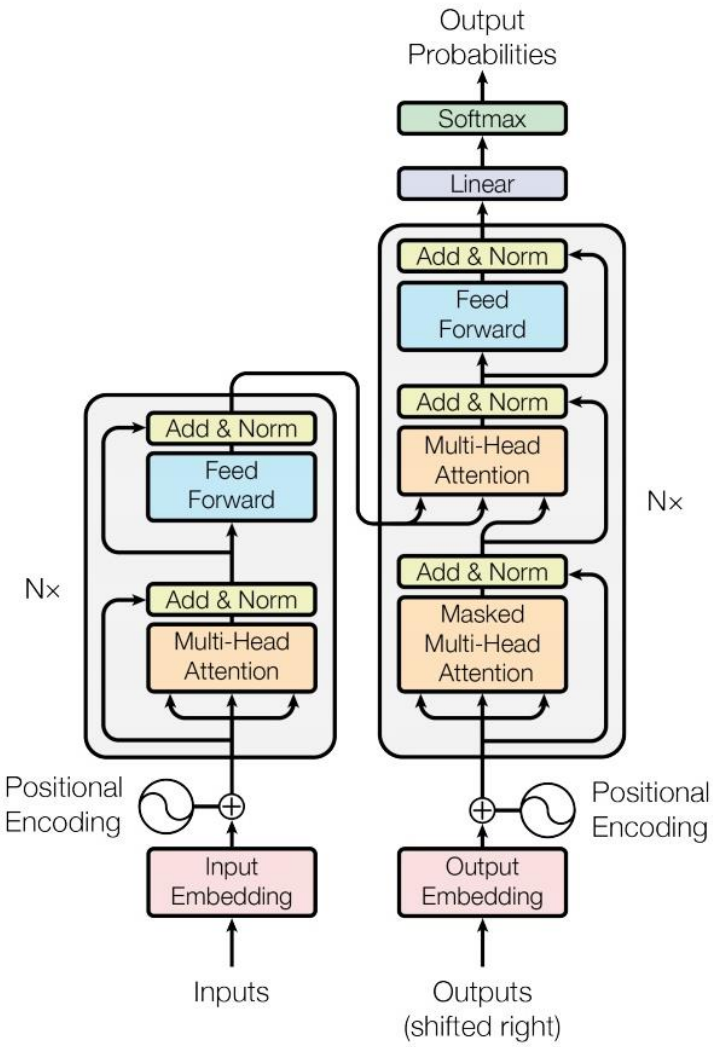

图 3-2 Transformer 结构图[49]

Transformer 摆脱了基于循环层或卷积层的架构，因此允许更多的并行化，大幅提升了训练速度。Transformer 的输出采用加权平均的方法，充分考虑了所有的信号，但也导致分散了注意的分布，从而忽略了相邻信号之间的关系。

在本研究的模型中，综合了上述提及的 seqGAN 与 Transformer 算法，并在这一基础上进行算法的优化和创新，进一步推进了 GAN 在 NLP 中的应用。Transformer 算法本身就可以视为一个文本生成结构，本研究将其复用在完整的生成对抗网络里，作为生成器存在。新模型中使用了 seqGAN 的概念，使其能够克服离散向量带来的参数更新问题。同时也在 Transformer 的算法基础上继续进行优化，通过对文本进行局部化建模，对 Transformer 的原始算法实现改进，从而增强捕获短程依赖项的能力。

3.2.2 基于高斯偏差的强化自注意力机制

在文本生成过程中，每个句子的生成过程可以视为依据生成器规定策略所采取的一系列操作，生成器定义策略，根据源语句生成目标语句。在本研究改进的算法模型中，将带有自注意力的编解码器结构作为生成器，并引入了以高斯偏差形式表示的局部建模。

通过引入局部建模的方式，来解决原始的 Transformer 算法模型存在的问题，即其训练过程中存在注意分布分散的情况，从而导致长距离的依赖关系被关注，但相邻的信号之间的彼此依赖关系被忽略。传统的自注意力模型优点非常明显，其可以直接关注全部输入元素，并借此实现获取长距离的依赖关系。不过这一操作过程是利用加权平均来完成的，这使得相邻元素彼此间的关系被忽视[50]。相邻元素之间的关系在人类语言中一般意味着短语，在文本生成里能够起到重要作用。可以借助给自注意力增加局部建模的方式，达到使学习局部上下文能力提升的目的。

把局部建模方案设计成可学习的高斯偏差 G，如式 3-13，其中 P_i 与 D_i 分别代表需要获得更多注意的局部范围的中心与尺寸：

$$G_{i,j} = -\frac{(j-P_i)^2}{2\sigma_i^2}, \quad \sigma_i = \frac{D_i}{2} \tag{3-13}$$

$$\left[\frac{P_i}{D_i}\right] = I \bullet sigmoid\left(\left[\frac{P_i}{z_i}\right]\right) \tag{3-14}$$

利用对应目标词 i 的查询（query）对 P_i 进行计算：

$$P_i = U_P^T \tanh(W_P Q_i) \tag{3-15}$$

选用 Query-Specific Window 方式计算窗口尺寸 D_i：

$$z_i = U_d^T \tanh(W_P Q_i) \tag{3-16}$$

最后，在原始注意力分布里引入高斯偏差，以实现修正，从而可以获得局部强化的权重分布为：

$$ATT(Q,K) = softmax(energy + G) \tag{3-17}$$

修正后的自注意力模型，不仅保留了基础模型学习长距离依存关系的能力，还提升了获取短距离依赖的能力。通过这样的方式，对传统的 Transformer 算法实现强化。

3.2.3 基于强化自注意力机制改进生成器网络

本研究的强化序列生成对抗网络中，将强化后的 Transformer 应用为生成器，

相当于在强化 Transformer 文本生成能力的基础上，加入了生成对抗训练，借此融合了序列生成对抗网络与强化 Transformer 的双重能力，实现了生成文本质量的提升。

本研究改进的算法模型采用的生成器仍然保持了 Transformer 中六层编码器-解码器的结构，每一个原始模型的编码器与解码器中，都包括了自注意力机制，而本研究即为在原有模型的自注意力机制中，引入了局部建模，构成新的生成模型，作为本研究训练网络中的生成器部分。完整的生成器网络结构图如图 3-3：

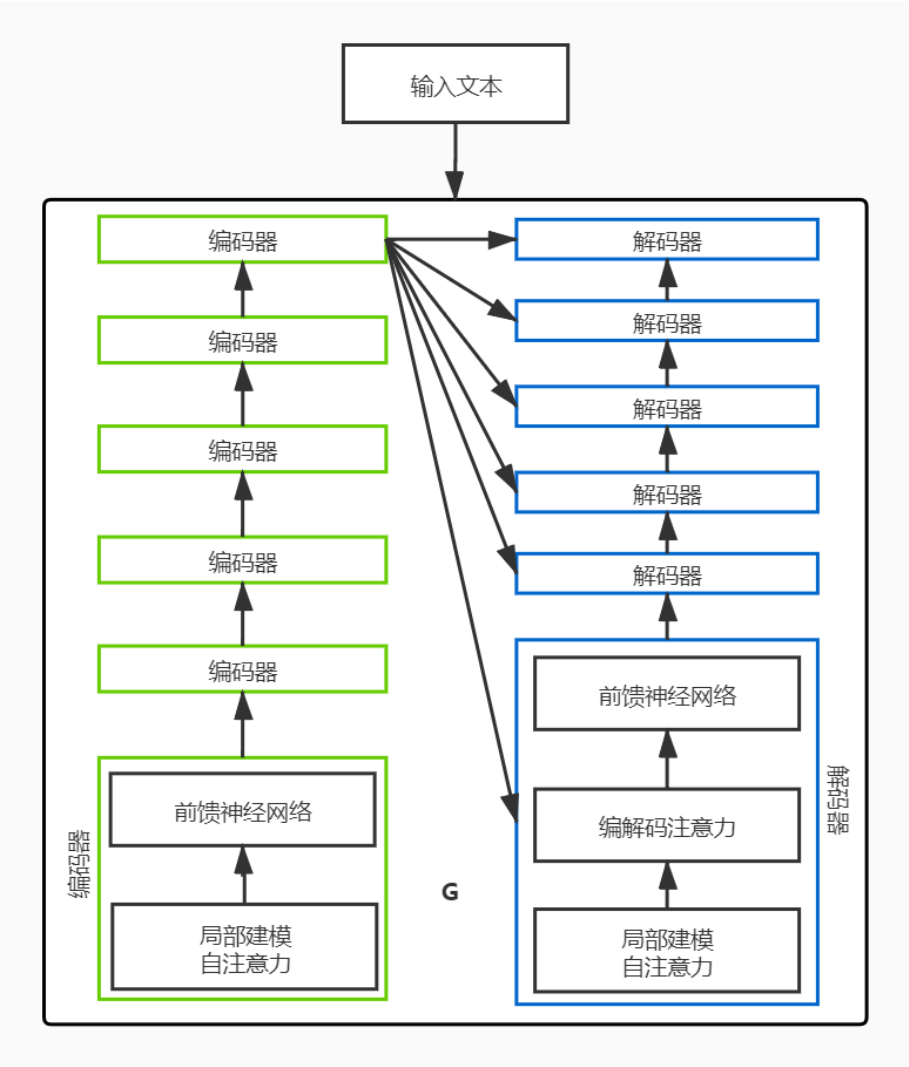

图 3-3 生成器网络结构图

3.3 基于惩罚的目标函数

除了上一节所讨论的文本特征提取能力有限之外，生成对抗网络的另一明显缺陷是模式崩溃，意即多次训练后的生成对抗网络会因“奖励”设置的限制，仅在少数几个模式附近生成文本，而忽略其他模式，使得产生的文本缺少多样性。本研

究用一种基于惩罚的函数取代了序列生成对抗网络的损失函数，它以最小化总体惩罚而不是最大化奖励作为目标。

通过以对损失函数进行更改的方式，去规避生成网络为了得到更高的奖励分值，而一直生成重复内容，迫使生成网络离开“安全”样本，进而生成多样性更丰富的内容。本研究最终选用了 SentiGAN 中目标函数的概念，具体结构如下：首先与序列生成对抗网络的思路一致，把文本生成转化成序列决策步骤，来解决梯度无法传递回生成模型的问题。设定在任一时间点t，都训练一个生成器G_i来生成一个序列：

$$X_{1:t}=\{X_1,X_2,\ldots,X_t\} \tag{3-18}$$

其中X_t代表给定的词典 C 中的一个向量。$G_i(X_{t+1}\mid S_t;\theta_g^i)$代表了基于之前生成的词语，$S_t=\{X_1,X_2,\ldots,X_t\}$来选择第$t+1$个词条的概率。基于上述讨论，定义本研究中基于惩罚的损失函数为：

$$L(X)=G_i(X_{t+1}\mid S_t;\theta_g^i)\bullet V_{D_i}^{G_i}(S_t,X_{t+1}) \tag{3-19}$$

其中，$V_{D_i}^{G_i}(S_t,X_{t+1})$是序列$X_{1:t+1}$的惩罚项，用判别器进行计算。最后，第$i$个生成器的目标是将整体的惩罚项最小化：

$$\begin{aligned}J_{G_i}(\theta_g^i)&=E_{X\sim P_{g_i}}\left[L(X)\right]\\&=\sum_{t=0}^{t=|X|-1}G_i(X_{t+1}\mid S_t;\theta_g^i)\bullet V_{D_i}^{G_i}(S_t,X_{t+1})\end{aligned} \tag{3-20}$$

同时，由于判别器只能判定一个完整的句子，因此本研究选择用蒙特卡洛搜索结合 roll-out 策略对剩下的$X-t$个未知的单词完成采样。综上所述，第i个生成器的惩罚函数可以表示如下：

$$V_{D_i}^{G_i}(S_{t-1},X_t)=\begin{cases}\dfrac{1}{N}\sum_{n-1}^{N}(1-D_i(X_{1:t}^n;\theta_d)) & t<|X|\\ 1-D_i(X_{1:t};\theta_d) & t=|X|\end{cases} \tag{3-21}$$

将三种模型的目标函数进行对比，分别是生成对抗网络、序列生成对抗网络以及本研究的算法模型：

$$J_G(X)=\begin{cases}E_{X\sim P_g[-\log(D(X;\theta_d))]} & GAN\\ E_{X\sim P_g\left[-\log(G(X|S;\theta_g)D(X;\theta_d))\right]} & SeqGAN\\ E_{X\sim P_g\left[G(X|S;\theta_g)V(X)\right]} & Our\ Model\end{cases} \tag{3-22}$$

目标函数主要有两方面改进。首先，本研究中选择用了基于惩罚的目标函数，这是一种衡量的 Wasserstein 距离方式，因此能够在训练过程中持续给出有意义的梯度，但其他两种损失函数是无法满足这个要求的。其次，本研究所改进的算法模型采用了惩罚而非奖励。本研究中基于惩罚的目标函数方程$G(X|S;\theta_g)V(X)$能够被视为在给基于奖励的目标函数$(-G(X|S;\theta_g)D(X;\theta_d))$加上$G(X|S;\theta_g)$，因此可以生成更有多样性的样本，而不是重复性的“好”样本。

3.4 模型整体框架

综合上文中的推导，对于文本生成的任务，本研究设计了一个新的基于惩罚最小化强化序列生成对抗网络算法（Penalty-based Enhanced Sequence Generative Adversarial Net, PB-ESGAN），它可以改进模型的特征提取与文本多样化。

这一算法基于序列生成对抗网络的概念，结合具有局部建模的自注意力，同时引入了最小惩罚的目标函数。PB-ESGAN 具有比原始模型更好的性能。原始模型中，生成器的文本特征提取能力不足，本研究用带有注意力机制的编解码器结构替代了序列生成对抗网络的生成器，使解码器更有效地利用上下文信息，极大增强了其捕获长短距离依存关系的能力。此外，PB-ESGAN 用最小惩罚机制替代原模型的损失函数，令其将最小化惩罚作为训练目标，以改善模式崩溃问题，丰富文本的生成内容。

图 3-4 描述了 PB-ESGAN 模型的完整体系结构，整个框架主要包括两个对立学习目标：生成器学习与判别器学习。生成器以真实文本数据集句子为基础，旨在生成与原数据集难以区分的句子，具体来说，它的目的是最小化我们提出的基于惩罚的目标。相反，判别器以真实文本数据集词句当作条件，目的是将机器生成的词句与原数据集的词语区分开来，可以将判别器视为一个动态目标，因为其与生成器同步更新。

3.5 实验设计和结果分析.

基于上述理论分析，得到了完整的算法模型搭建，实现了结合序列生成对抗网络与具有局部建模的自注意力，并引入了最小惩罚的目标函数的 PB-ESGAN 算法。本研究对这一算法模型设计了相关实验，以证明 PB-ESGAN 算法的优越性。并且由于该算法的基础是序列生成对抗网络，因此使用了同样以序列生成对抗网络为基础的一系列模型作为对比实验，通过恰当的评估指标来观察文本生成效果的区别。

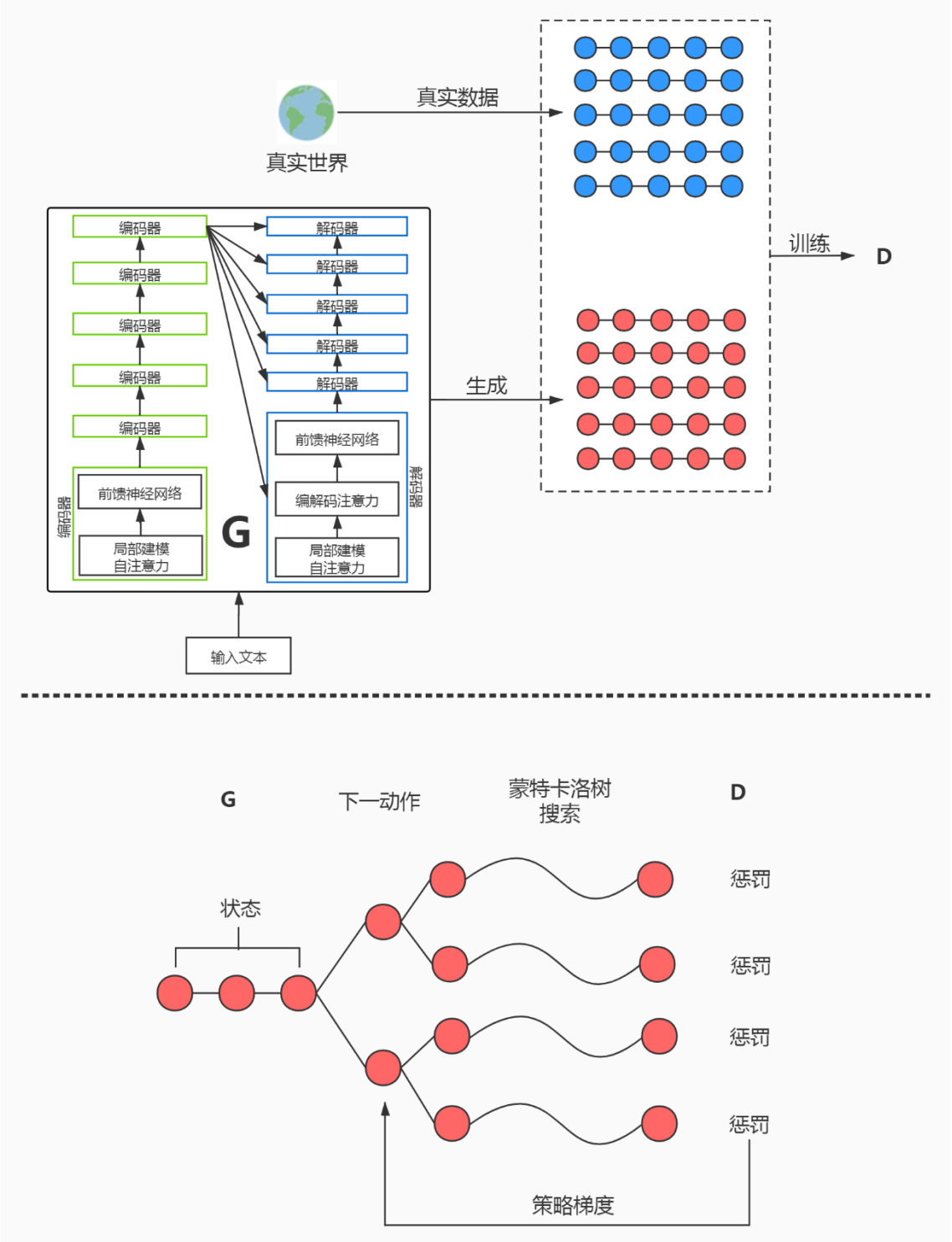

图 3-4 结合具有局部建模自注意力和基于最小惩罚的 PB-ESGAN 模型的整体架构

3.5.1 实验环境和实验数据

本研究采用的实验环境具体情况如下：软件使用的操作系统为 Windows，基于 PyTorch 的深度学习框架，使用 Python 作为编程语言。硬件使用的 CPU 是 amd R5

2600x，GPU 使用的是英伟达 RTX2070。

对于文本生成，本实验选用了阿里云中的商品描述文案数据集这一数据集。商品描述文案数据集的语料库来自中国最大的电商网站阿里巴巴，包括了商品标题与商品的描述文案。

对于数据集的预处理采取了中文语言实验中较多采用的预处理方法，首先对其进行中文分词。为了加快训练速度，在模型中训练时，我们在语料库中随机选取其中了 10 万条描述文案记录作为数据集。

3.5.2 参数设置

对于生成器，本实验参考 Transformer 的训练参数进行设置。按照 Transformer 原论文中的基本模型，将词嵌入维度设置为 512，丢失率设置为 0.1，头数设置为 8。编码器与解码器各具有六层堆栈，使用波束大小为 4 且长度损失为α=0.6 的波束搜索。所有模型都在 Pytorch 中实现。

3.5.3 评估指标

本研究中采用 BLEU 评分作为评价指标来衡量生成文本和真实文本间的相似性。BLEU(Bilingual Evaluation Understudy)[51]，又叫双语互译质量辅助工具。最初设计用于自动评价机器翻译质量。关键是要比较机器生成的结果和人类提供的参考彼此间的相似度。计算 BLEU 指标，会用到机器翻译产生的文本（candidate docs）和翻译人员给出的参考翻译样本（reference docs），将二者进行对比。实际上，BLEU 这一指标能够判断机器翻译和参考翻译之间的相似程度，其值范围是 0-1，当值越接近 1，代表机器翻译的效果越好。

现在 BLEU 已经扩展应用于文本生成中，对生成文本和原数据集本文进行对比，也是文本生成领域对生成本文最通用的评价指标。BLEU 的主要思想如下：首先，在考察模型生成的文本的质量时，生成的文本与参考文本有越多的重复片段，则认为两者越相似。因此，可以使用 n-gram 精确度来计算两个文本间的相似度，即生成的文本中出现在参考文本中的 n-gram 的次数总和在生成文本中所有 n-gram 次数总和的占比，这个占比越高，则说明相似度越高。

BLEU 一般考虑 1-5 个 n-gram，评估长文本时会选择较大的 n 值，短文本则选取较小的 n 值。BLEU 的具体计算公式如下：

$$BLEU = BP \cdot \exp(\sum_{n=1}^{N} \omega_n \log p_n) \tag{3-23}$$

其中 N 代表了选取到的最大的 N-gram 匹配规则，p_n 则表示这一规则下对应的匹

配程度，其值域是[0,1]，而 ω_n 为不同 N 值匹配规则对应的权重。BP 代表着过短惩罚，作用是判断生成文本的长度并且在文本过短时给予对应的惩罚。一般 BP 的公式如式 3-24，公式里 c 是机器译文（生成文本）的长度，而 r 是人工译文（真实文本）的长度：

$$BP=\begin{cases}1 & \text{if}(c>r)\\ e^{(1-r/c)} & \text{if}(c\le r)\end{cases} \tag{3-24}$$

除了 BLEU-3 与 BLEU-4 以外，实验中还用到了 Self-BLEU，这个指标可以用来对生成的数据内容进行其多样性是否丰富的评估。其核心思想是判断一个句子与生成的集合中其他句子的相似程度，借用了 BLEU 算法判断语句是否相似的思路。以一个语句为生成内容，另一个语句为参考内容，能够求出每一条生成语句的 BLEU 分数，之后就可以把这样求得的 BLEU 分数判定为文档的自 BLEU。本文以 Self-BLEU-3 作为 Self-BLEU 值。Self-BLEU 分数越高，说明生成文本的重复程度越高，多样性越少，算法模型的模式崩溃越严重。

3.5.4 实验结果

本研究中，为了更好地展示 PB-ESGAN 算法模型生成文本的质量，一共实现了 4 个不同的算法模型进行文本生成，以进行效果对比。分别是序列生成对抗网络、序列生成对抗网络+Transformer、序列生成对抗网络+强化 Transformer（引入局部建模）以及 PB-ESGAN（序列生成对抗网络+强化 Transformer +基于惩罚的目标函数）。

此处对于 PB-ESGAN 的模型训练过程进行展示，截图如下：

首先训练生成器若干次，一般设置在 150-200 次：

```
Starting Generator MLE Training...
[MLE-GEN] epoch 0 : pre_loss = 6.5744, BLEU-[3] = [0.355], gen_NLL = 5.6299, self_bleu = [0.37],
[MLE-GEN] epoch 5 : pre_loss = 4.4034, BLEU-[3] = [0.551], gen_NLL = 4.2409, self_bleu = [0.547],
[MLE-GEN] epoch 10 : pre_loss = 4.0324, BLEU-[3] = [0.571], gen_NLL = 3.9632, self_bleu = [0.528],
[MLE-GEN] epoch 15 : pre_loss = 3.8781, BLEU-[3] = [0.558], gen_NLL = 3.8604, self_bleu = [0.546],
[MLE-GEN] epoch 20 : pre_loss = 3.7905, BLEU-[3] = [0.576], gen_NLL = 3.7986, self_bleu = [0.548],
[MLE-GEN] epoch 25 : pre_loss = 3.7335, BLEU-[3] = [0.583], gen_NLL = 3.7627, self_bleu = [0.523],
[MLE-GEN] epoch 30 : pre_loss = 3.6933, BLEU-[3] = [0.578], gen_NLL = 3.7344, self_bleu = [0.519],
[MLE-GEN] epoch 35 : pre_loss = 3.6627, BLEU-[3] = [0.573], gen_NLL = 3.7103, self_bleu = [0.529],
[MLE-GEN] epoch 40 : pre_loss = 3.6416, BLEU-[3] = [0.561], gen_NLL = 3.6817, self_bleu = [0.517],
[MLE-GEN] epoch 45 : pre_loss = 3.6176, BLEU-[3] = [0.579], gen_NLL = 3.6676, self_bleu = [0.53],
[MLE-GEN] epoch 50 : pre_loss = 3.6127, BLEU-[3] = [0.588], gen_NLL = 3.6540, self_bleu = [0.55],
[MLE-GEN] epoch 55 : pre_loss = 3.5945, BLEU-[3] = [0.578], gen_NLL = 3.6489, self_bleu = [0.506],
[MLE-GEN] epoch 60 : pre_loss = 3.5788, BLEU-[3] = [0.564], gen_NLL = 3.6379, self_bleu = [0.528],
[MLE-GEN] epoch 65 : pre_loss = 3.5688, BLEU-[3] = [0.561], gen_NLL = 3.6408, self_bleu = [0.535],
[MLE-GEN] epoch 70 : pre_loss = 3.5659, BLEU-[3] = [0.579], gen_NLL = 3.6280, self_bleu = [0.56],
[MLE-GEN] epoch 75 : pre_loss = 3.5557, BLEU-[3] = [0.585], gen_NLL = 3.6372, self_bleu = [0.518],
```

图 3-5 PB-ESGAN 训练生成器

接下来训练判别器数次，一般设置为 4-5 次：

```
Starting Discriminator Training...
[MLE-DIS] d_step 0: d_loss = 0.1723, train_acc = 0.9374,
[MLE-DIS] d_step 1: d_loss = 0.0162, train_acc = 0.9984,
[MLE-DIS] d_step 2: d_loss = 0.0029, train_acc = 0.9999,
[MLE-DIS] d_step 3: d_loss = 0.0018, train_acc = 0.9999,
[MLE-DIS] d_step 4: d_loss = 0.0022, train_acc = 0.9997,
```

图 3-6 PB-ESGAN 训练判别器

在判别器与生成器联合训练前，对生成器先进行初始化：

```
Starting Adversarial Training...
Initial generator: BLEU-[3] = [0.566], gen_NLL = 3.6246, self_bleu = [0.506],
```

图 3-7 生成器初始化

之后按照每轮训练一次生成网络和数次判别网络的顺序，训练数轮，训练的 epoch 越多，生成文本质量越高：

```
-----
ADV EPOCH 0
-----
[ADV-GEN]: g_loss = 692.2173, BLEU-[3] = [0.577], gen_NLL = 3.6376, self_bleu = [0.534],
[ADV-DIS] d_step 0: d_loss = 0.0041, train_acc = 0.9992,
[ADV-DIS] d_step 1: d_loss = 0.0028, train_acc = 0.9995,
[ADV-DIS] d_step 2: d_loss = 0.0033, train_acc = 0.9991,
[ADV-DIS] d_step 3: d_loss = 0.0021, train_acc = 0.9996,
-----
ADV EPOCH 1
-----
[ADV-GEN]: g_loss = 218.5261, BLEU-[3] = [0.592], gen_NLL = 3.6564, self_bleu = [0.545],
[ADV-DIS] d_step 0: d_loss = 0.0034, train_acc = 0.9994,
[ADV-DIS] d_step 1: d_loss = 0.0029, train_acc = 0.9994,
[ADV-DIS] d_step 2: d_loss = 0.0020, train_acc = 0.9995,
[ADV-DIS] d_step 3: d_loss = 0.0022, train_acc = 0.9993,
-----
ADV EPOCH 2
-----
[ADV-GEN]: g_loss = 78.1953, BLEU-[3] = [0.604], gen_NLL = 3.6766, self_bleu = [0.507],
[ADV-DIS] d_step 0: d_loss = 0.0027, train_acc = 0.9993,
[ADV-DIS] d_step 1: d_loss = 0.0027, train_acc = 0.9993,
[ADV-DIS] d_step 2: d_loss = 0.0019, train_acc = 0.9996,
[ADV-DIS] d_step 3: d_loss = 0.0018, train_acc = 0.9995,
```

图 3-8 PB-ESGAN 训练

算法训练结果首先用 BLEU-3 这一评估指标在过程中的变化进行展示，BLEU 指标值处于 0-1 之间，越接近 1 代表文本质量越高，对于四种算法的实现结果合

并截图展示如图 3-9。

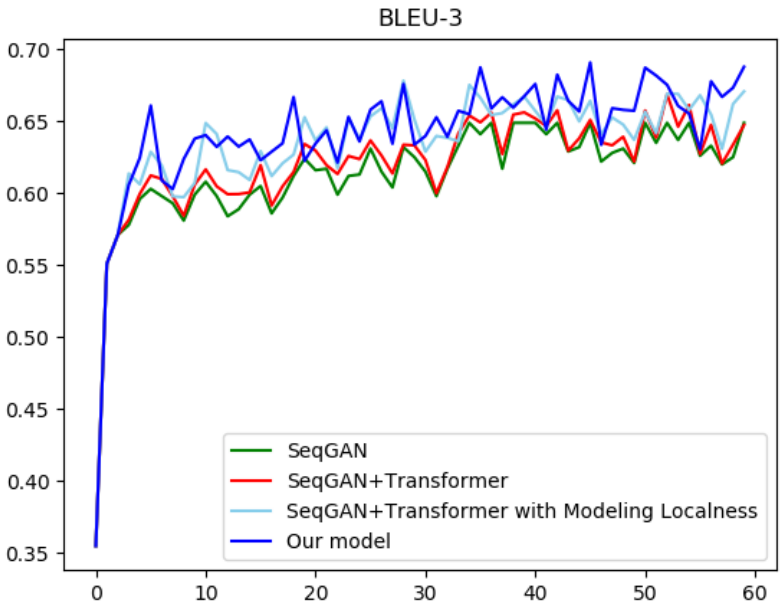

图 3-9 四种算法模型训练过程对比

图 3-9 中列出了四种算法模型在训练过程中，在训练到不同 epoch 时 BLEU-3 指标的变化趋势，并对四种算法进行了对比，可以看出在整个训练过程中，PB-ESGAN 算法模型的训练情况相对其他三种算法一直是更优的状态。下面列出四种算法在商品描述文案数据集上实现的 BLEU-3、BLEU-4 与 Self-BLEU 指标的具体数值统计，并对实验结果进行分析。

表 3-1 商品描述文案数据集上的实验结果

Models	Metrics		
	BLEU-3	BLEU-4	Self BLEU
seqGAN	0.632	0.489	0.685
seqGAN + Transformer	0.649	0.492	0.678
seqGAN + Transformer with Modeling Localness	0.663	0.509	0.656
Our Model	0.681	0.514	0.644

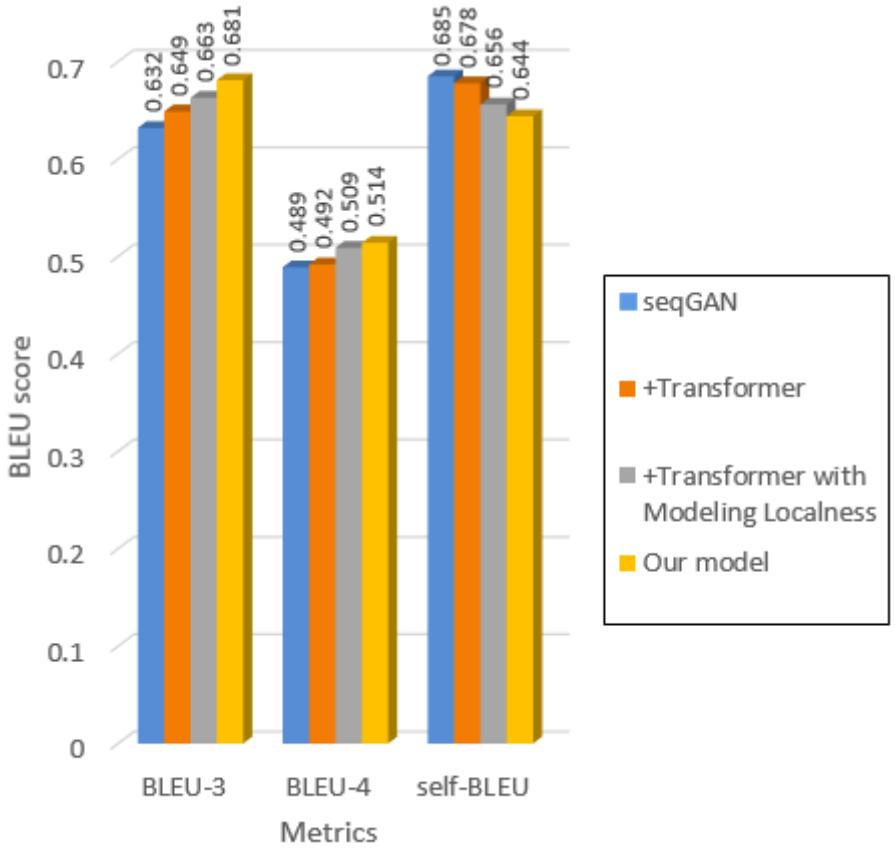

图 3-10 商品描述文案数据集上四种算法的 BLEU-3、BLEU-4 与 Self BLEU 值

通过表 3-1 中四种算法生成文本的 BLEU 评估指标分析不难看出，在充分利用文本信息的基础上，引入了带有局部建模的自注意力机制与最小惩罚目标函数，本研究的模型在数据集上表现出良好的性能。如图 3-10 所示，与原始序列生成对抗网络模型相比，在商品描述数据集中，新算法模型得到显著改进，提高了+4.9 个 BLEU 点。相比于引入 Modeling Localness with Transformer 的序列生成对抗网络，新算法仍然达到了+1.8 个 BLEU 点的提高。

同时可以看到，Self-BLEU 的评分显著降低。这一指标针对生成文本的重复程度进行评价，PB-ESGAN 的分数为 0.644，与传统的 seqGAN 相比，降低了-4.1 个 BLEU 点，这意味着 PB-ESGAN 模型生成的文本重复程度大幅降低，即文本内容的多样性更加丰富。说明通过算法结构的优化，确实实现了模型模式崩溃问题的改善。这表明本研究的模型可以应用于文本生成，并获得更好的性能。

在不同数据集上的实验都可以看出，Transformer 模型对于文本特征提取能力的显著改进，而带有局部建模的自注意力机制比 Transformer 更胜一筹。与基于原始损失函数的朴素序列生成对抗网络相比，PB-ESGAN 利用基于最小惩罚的目标函数生成具有较高 BLEU 点的句子。实际上，PB-ESGAN 融合了上述两种方法的

优势，达到了更好的文本生成效果。

为了更直观地展示文本生成的结果，表 3-2 对四种算法生成的文本内容进行样例列举。

表 3-2 四种算法生成的文本样例列举

模型	生成句子
seqGAN	1. 时尚有型，给你天然防水实木，清新有趣的图案，折叠棉麻面料打造，不易变形。 2. 美观与当储物空间，收到镜面工艺为小朋友使用更安全。佩戴牢固。人体工学设计一款冬季上穿臃肿的简约。
seqGAN + Transformer	1. 让你在跑步感受，呵护日常健康。更好的振膜，居家稳重慢慢声音，同时不失和谐。 2. 简约百搭款打造，舒适有型。简约百搭的款式，舒适的材料很很舒服。简洁的版型百搭实穿，复古修身版型。
seqGAN + Transformer with Modeling Localness	1. 这款冬季上毛衣帅气，宽松的版型，线条韩版新颖，穿脱方便。潮男成熟稳重厚实，穿上以后更加美观大方。 2. 色泽细腻，实木的胡桃木白橡木材质，带来出色的环保效果。采用实木的设计，造型别致。
Our Model	1. 腰间搭配腰带，可穿出青春潮流感觉。款式是透气且柔软舒适的搭配，超级柔软面料手感细腻挺括有质感。 2. 十分颜值在线，支持超级快速充电。超长续航容量极大，续航久不发烫。

表中随机截取各模型生成的商品描述文本，由于内容生成与选取都具有随机性，因此并不针对某一固定商品进行描述。通过文本样例可以看出，seqGAN 生成的文本含义并不通顺，部分语句不符合语法规范，另外不同短句的语义可能在描述不同的商品，即无法有效实现长文本之间的语义与逻辑连接。对比来看，本研究的模型生成的商品描述文案语义更连贯，前后文也更加符合语言逻辑，生成的长文本质量更高。

在模式崩溃方面，可以直观地看出，几个对比模型生成的文本中仍存在较多的

重复内容，即缺乏多样性，当重复内容过多甚至固定内容循环出现，就意味着模式崩溃。而本研究模型生成的重复文本大幅减少，说明引入惩罚最小化的目标函数有效缓解了模式崩溃的问题。

3.6 本章小结

本章提出了一种基于惩罚最小化强化序列生成对抗网络的文本生成模型，这一算法模型的改进主要表现在生成器与损失函数两个部分中。首先是生成器部分，原始的生成器是 LSTM 模型，这一结构限制了算法的并行化，同时文本特征的提取能力也不足，改进后的生成器结构为加入了具有局部建模的自注意力机制，利用自注意力机制本身的特性提升了算法的并行效率，并通过引入局部建模，极大提升了其捕获长短距离依存关系的能力。另一个优化点在于损失函数的改进，本模型使用的损失函数使用最小惩罚机制替代原模型的损失函数，最小惩罚机制的本质是引入了 Wasserstein 距离的衡量，产生有意义的梯度，从而缓解原本存在的模式崩溃问题。

本章还使用了序列生成对抗网络、序列生成对抗网络+Transformer、以及序列生成对抗网络+强化 Transformer（引入局部建模）这三种算法作为对比模型，在商品描述文本数据集上进行了对比实验。实验结果数据显示，本研究的模型 PB-ESGAN 优于当前实现文本生成的其他先进方法的实验结果，其生成的文本内容相比于其他对比模型而言，有更高的准确性与多样性。

本章的生成文本专注于长文本内容，所使用的文本生成算法模型也可以复用在其他类型的文本生成领域中，比如诗词文本等极富特色的文本类型，同样也可以使用该模型，这证明本章提出的这一模型具有较好的普适性。

第四章 生成对抗网络文本生成模型用于诗歌生成

上一章中提出了基于惩罚最小化强化序列生成对抗网络的文本生成模型，并将其在商品描述数据集上进行长文本生成实验，取得了较好效果。为了证明这一模型的普适性，应选择具有不同特点的文本数据集进行生成实验，并尝试在模型上进行适当算法调整，以得到更好的生成效果。

本章选择唐代绝句数据集进行实验，因绝句诗歌具有固定字数、特定格律等特点，并且是一种典型的短文本，与上一章中使用的商品描述数据集区分明显，能够更好证实算法模型的普适性。本章在上一章研究的基础上，增加了拼音汉字对照表，以判断字与字之间是否满足押韵条件，并在模型中引入了押韵判断模块，促使生成的诗句之间符合格律规则。实验证实本研究模型在诗歌文本中也能得到较好的生成效果。

4.1 诗歌生成介绍

诗歌，在中国千年古老璀璨的传统文化中，无疑是文学史上耀眼的明珠。在历史的长河里，无数朗朗上口的诗词没有失去其光辉，被一代又一代的人传诵至今。诗歌的耀眼和独特之处不仅仅是每一句简短的诗句之中都饱含情感、值得反复咀嚼，也更是因为每一句话都平仄有序、音调押韵，极讲究格律之美。

数千年来，中国古典诗歌的形式一直在不断地发展和演变，而每一种类型的诗歌都会遵从其独特的结构、音调与韵律规则。这也使得诗歌这一独特的文本类型始终被诸多研究人员关注。

表 4-1 五言绝句规则之一及其示例

塞下曲（唐·卢纶）	格律规则（仄起首句入韵式）
月黑雁飞高，	仄仄仄平平（韵），
单于夜遁逃。	平平仄仄平（韵）。
欲将轻骑逐，	平平平仄仄，
大雪满弓刀。	仄仄仄平平（韵）。

也正是因为这些独特的格律规则，使得普通人学习创作诗词有一定的门槛和难度。随着深度学习的高速发展，利用一些合适的算法模型，来严格依据诗歌的格律规则完成诗词的生成，也是一种向中国文化传承注入新鲜血液的方式，对于中华

传统文化的弘扬和继承十分有利。

上个世纪 60 年代起，国外的一些研究人员就已经关注到机器学习实现诗歌生成这一领域，国内与传统诗歌自动生成有关的研究是从 20 世纪 90 年代才开始推动和发展。近些年，诗词自动生成这一领域也逐渐受到更高的关注。

例如，周昌乐等人将遗传算法应用在宋词的自动生成中，这一方法先随机地生成大量诗句信息，之后结合句法和语义加权定义一个评估函数，再利用制定好的评估函数给每一个诗句打分，把分数更高的诗句作为生成内容输出，这样持续进行每一步，最后可以生成一篇完整的宋词。这一方法产生的单句质量都比较高，但每一句诗彼此之间的连贯度和相关度较弱，同时生成效率也并不高。

另外，严睿等人在诗歌生成中应用了自动文摘的相关技术，首先利用摘要框架构建了一套诗词生成系统，之后根据输入的关键词，系统会在语料库中检索找出相关的词语，再根据语义、平仄、结构、押韵等条件对相关词完成聚类组合，最后利用约束迭代优化来修改词语，找出最优解形成完整的诗歌。这一方法在是个的语义连贯性的表现大幅提升，但对于评估函数的依赖非常重，因此也有其局限性。

还有将诗词生成问题视作机器翻译问题来解决的，周明等人采用了这样的方式。利用机器翻译模型，把前一句诗当做源语言，后一句生成的诗当做目标语言，依据之前的内容来翻译得到后面的内容，翻译的过程中也包括平仄和押韵这些约束条件。持续重复这一生成的过程，得到完整的诗歌。此外，作者还设定了一套人工评估的指标，分为押韵、流畅度、连贯性和语义四个维度，对生成的诗歌质量进行评定。这种方法的优势在于算法模型可以自主地学习语料库中的诗词规律，生成效率显著提高，但是生成的整首诗对于主旨的诠释并不够到位。

随着深度学习技术持续发展，也有越来越多相关的算法应用在诗歌生成领域中。其中一个比较典型的是基于循环神经网络的生成算法模型（RNN-based Poetry Generation, RNNPG），由 Zhang 等人提出。RNNPG 系统由 CSM、RCM 和 RGM 三个核心模块构成，其生成过程是首先依据用户给定的关键词，在固定语料库中搜索扩充意群，之后在一定的约束条件下产生诸多备选的诗句内容。接下来需要用一个固定的语言模型给每一句诗句评分，得分最高的诗句就是诗歌的第一句。之后生成每句诗的方法和机器翻译相似，通过之前的行来生成下一行，只不过将之前的所有行都统一作为输入。这种方式简化了诗歌生成的过程，也提高了多样性，但也存在生成过程中出现偏离主题的问题。

另一种典型的基于深度学习的诗歌生成模型是 2016 年 Wang 等人提出的 ANMT（Attention based Neural Machine Translation Network, ANMT）模型，这一模型建立在序列到序列（seq2seq）的框架上，加入了注意力机制。其编码器使用了双

向长短时记忆网络（Bi-directional Long Short Term Memory, BiLSTM），解码器用的是长短是记忆网络（LSTM）。完整的生成流程与 RNNPG 类似，以之前的语句作为输入，模型训练得到下一句，循环得到整首诗词。ANMT 相比于普通的 RNN 模型而言，可以引入更多的历史信息，梯度消失的问题也会明显减少，但仍然无法完全避免主题漂移的情况。

PPG（Planning based Poetry Generation）生成模型同样基于编解码器的结构，这一模型的创新之处在于引入了写作中常见的提纲技巧，规避了之前几种模型里会出现的主题漂移的问题。PPG 模型主要由两个部分构成，分别是规划模型与生成模型。规划模型的作用是制定规划信息，也就是写作的提纲，生成模型是结合得到的规划信息生成诗歌，直到得到整首诗。PPG 在诗歌生成中取得了不错的结果，不过也尤其局限，即规划信息对于生成的诗歌质量影响很大。

上述的几种算法模型都是诗歌生成领域中极受关注的里程碑式进展，可以看出诗歌生成这一方向在逐渐受到更多研究人员的关注，也有更多的科学家在尝试进一步推进这一领域的研究。

本研究选择诗词这一典型的文本类型作为算法模型的扩展研究主要出于以下三个原因：首先，机器生成诗歌显然降低了诗歌创作的难度，从而协助语言研究人员能够进一步地研究语言创作的可能性和多样性；其次，诗词有其独特的文本特征，包括固定的格律规则、短文本等等特征，与本文第三章研究的长文本有显著的区别，因此也更能够体现本研究的算法模型的普适性；最后，同样是因为诗词的格律规则极强，因此有格律分数这种特殊的评估指标，可以更直观地评判生成诗词的质量。本章即以中国古典诗歌中最为经典的绝句作为训练材料，采用第三章中的算法模型，再加以改进，进行诗歌自动生成方法的探索与尝试。

4.2 任务与训练目标

本章旨在证明本研究的算法模型 PB-ESGAN 的普适性，即能够在不同风格与特征类型的文本中都能够取得较好的生成效果，得到较好质量的生成文本。

本章选择诗歌中的绝句类型作为数据集，训练算法模型生成高质量的绝句内容。绝句作为诗歌的一种典型类型代表，具有短文本、每句字数固定、押韵、有特定的平仄规律等特点，与第三章中使用的商品描述长文本数据集有较大的区分，借此来展示模型的普适性更有说服力。同时，也会针对某些特点，如押韵等，对 PB-ESGAN 做出算法模型的调整，以便其更适用于诗歌类别的文本生成。

本章的模型训练目标与绝句这一文本类型的特点直接相关，绝句极富格律之美，格律也是评价一首绝句质量好坏的一个重要因素。绝句的格律规则列举包括以

下三个因素：结构（Structure）规则，即限制了诗句的行数和字数，每首诗有 4 行，每句为 5 个字或 7 个字；音调（Tone）规则，即每行诗的每一个字都需要符合不同绝句类型限定的几种平仄规则；韵律（Rhyme）规则，即在符合限定的音调规则的模式里，诗句的最后一个字需要满足押韵的条件。

本章的训练目标首先依据绝句的格律规则，旨在生成能够满足结构、音调与韵律规则的绝句诗句。同时，本章也以 BLEU-2 作为评估指标之一，按照上文中已经介绍过的 BLEU 公式，一般使用较大 n 值的 BLEU 评估长文本，短文本则适合用较小的 n 值进行评估。因此本章的目标即为，结合绝句特色对第三章算法模型进行适当调整，以便生成格律分数与 BLEU-2 两个评估指标上表现更好的绝句文本。

4.3 模型结构与训练方法

4.3.1 模型细节

本章采用的算法模型主体仍然为 PB-ESGAN，诗歌生成任务的目的就是为了证明本研究提出模型的普适性，因此依然以 PB-ESGAN 作为主要生成模型。同样，由于诗歌文本类型与普通长文本存在很大区别，也需要结合其特有的文本特征对算法模型进行适当地补充和调整。

前文已经提到，本章采用的绝句语料具有结构、音调和韵律的规则，其中结构和音调部分，更多地依赖于算法本身的自主学习来寻找规律。韵律的规则比较特殊，韵律就是押韵，而押韵本身是人为定义的一种概念，对算法而言不存在押韵与否的判断。

如果想要通过算法的改进提升押韵的指标，首先需要对数据集进行预处理，把每一个字和其对应的拼音与音调匹配，以便于通过拼音的韵母判断是否押韵。因此，本研究首先引入了 pypinyin 这一汉字拼音的对照表，完成汉字与拼音的对应匹配。其次，在原模型的基础上，加入了一个押韵判断的押韵字典，当每一句诗生成后，会自动把句末的字与上句诗的最后一个字提取判断押韵性，判断的方法如图 4-1。

简单来说，就是在生成的过程中，在第一个词的词表分增加一步操作，通过押韵字典的判断，在过程中把不押韵的词的概率调整为 0，借此增加押韵的诗句产生的比例。

这一步增加在 PB-ESGAN 的生成器之中，在生成器生成诗句的过程里增加押韵判断，促使其生成的诗句遵守韵律规则。

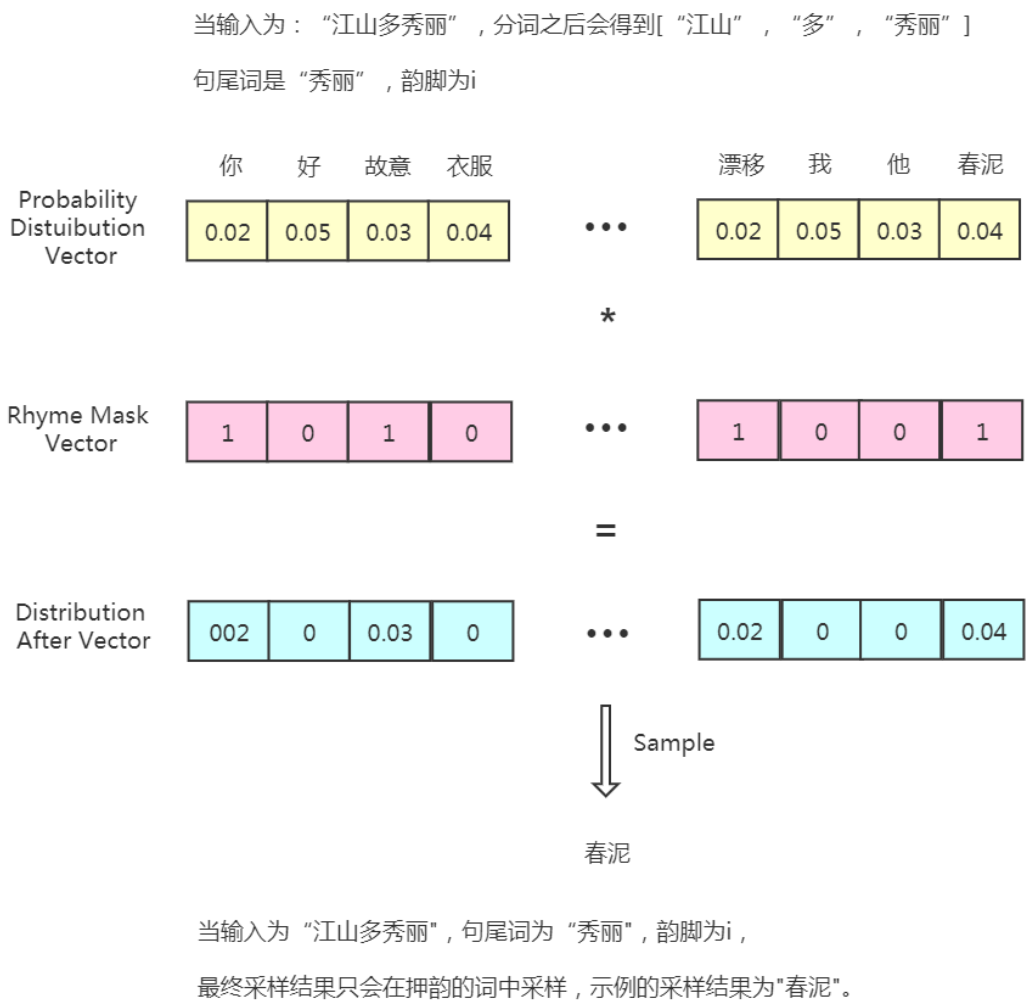

图 4-1 押韵判断采样过程

4.3.2 模型训练

应用于诗歌生成的整体算法模型仍然与第三章中模型基本一致，大框架基于生成对抗网络，包括了生成器与判别器两个部分。用引入了局部建模的自注意力模型作为生成器，同时在生成器中增加了押韵判定部分，判别器仍然保持使用 CNN 结构，并且加入了最小惩罚作为损失函数。

完整的训练过程与大部分生成对抗网络一样，一般分为三个部分，分别是预训练、生成器的训练、判别器的训练。

预训练是为了提高整个模型的训练速度，以及维持训练过程的稳定。因为在生成对抗网络中，如果生成器没有预先进行足够的训练，后期在和判别器共同对抗训练的过程中，生成器的训练会非常不稳定并速度极慢。出现这种情况的原因是，当生成器还没有得到很充分的训练就直接进入对抗训练时，生成器形成的文本语句

会呈现非常大的随机性，那么判别器就会十分容易判断出此时的文本是真实文本还是生成器生成的“虚假文本”，而这无法指导生成器向着更好的方向进行改进训练，导致整个训练过程无效化。因此，为了拥有更好的训练效果，本研究首先令生成器完成预训练过程。

之后的训练过程本质上是生成器与判别器交替进行的，其中生成网络需要最大程度生成更逼近的文本内容，以期能够骗过判别网络，让判别网络不能把两种文本区分开来。判别器的目标则是尽力地区别真实文本与生成的假文本。二者相互博弈、不断训练的过程，也是生成对抗网络优于其他基础算法模型的原因。

4.4 实验设计和结果分析

依据上述的算法模型，用 PB-ESGAN 为基础，增加了押韵判定的算法进行绝句生成的实验。同样，为了证明这一算法的有效性，使用序列生成对抗网络、序列生成对抗网络+Transformer、序列生成对抗网络+强化 Transformer（引入局部建模）三个基础算法模型作为对比。同时，也根据绝句文本的独特性质，采用了新的评估指标，更好地验证本研究算法的优越性。

4.4.1 实验环境和实验数据

本研究仍然沿用第三章中使用的实验环境，采用 Windows 作为操作系统，利用 PyTorch 的框架，并且编程语言选用 Python。硬件中 CPU 的型号为 amd R5 2600x，GPU 的型号为英伟达 RTX2070。

由于本章计划对绝句进行文本生成实验，因此选用了 Chinese-poetry 诗歌数据库，这一数据集中共包含 376241 首诗歌，本章仅选择其中的绝句语料。其中，唐代与唐代以后的绝句诗歌共有 77824 首，选择 2000 首作为测试集，其余内容作为模型的训练集。

由于绝句同为中文文本，同样需要对其进行中文分词作为数据集的预处理，以便进一步加快训练速度。

4.4.2 评估指标

本章中采取两个评估指标来判定生成诗歌的质量，第一个指标是 BLEU-2，第三章中已经详细介绍过 BLEU 指标的计算方式，本章选用 n 值为 2 的 BLEU。这是因为在 BLEU 的计算方法中，n 值意味着窗口的取值大小，一般生成文本越长，倾向于用越大的 n 值进行评估。而生成的绝句文本都具有短文本的特征，每行诗为 5 个字或 7 个字，此时就选用较小的 n 值 2 来进行评估。

另一个指标定义为格律分数，是针对绝句这一文本类型制定的评估指标，主要用于评估生成绝句格律的规范程度。前文中已经介绍过，绝句的格律规则包括三个：结构规则意味着绝句应符合每首 4 行、每行 5 个字或 7 个字；音调规则代表绝句每行诗的每一个字要满足一定的平仄规律；韵律规则则限定了诗句的最后一字需要在特定音调规则的模式里满足押韵要求。

首先，音调规则部分的平仄规律是绝句研究领域已经确定的固定类型，百度百科中有对所有类型的总结，本研究中的格律规则也是借此来进行限定。实验中将音调、韵律、结构分开各自求得一个分数值，并通过格律分数的计算公式进行综合计算，最后得到综合评分作为评估分数。格律分数的计算公式如下：

$$rules_{score} = \frac{1}{N}\sum_{i=1}^{N}(tone_i * \alpha + rhyme_i * (1-\alpha)) * structure_i \tag{4-1}$$

其中 N 值意味着算法模型生成的诗歌语料样本测试集的大小，α 是一个调节因子，作用是调整音调分数与韵律分数的权重大小，取值范围是[0,1]，本研究中将 α 取值为 0.5。$tone_i$ 代表算法模型生成的诗歌文本测试集中第 i 首诗歌的音调分数，分数的取值范围在[0,1]之间。$rhyme_i$ 表示算法模型生成的诗歌文本测试集里第 i 首诗的韵律分数，分数的取值范围也在[0,1]之间。$structure_i$ 代表算法模型生成的诗歌文本测试集里第 i 首诗的结构分数，取值只有 0 与 1 两个，如果完全符合固定限制的诗歌结构可得 1 分，否则就得 0 分。

4.4.3 实验结果

本章为了凸显 PB-ESGAN 在诗歌文本生成领域的优越性，仍然选用与第三章同样的三个算法模型共同进行实验，以展示生成效果的对比。三个对比模型同样采用改进递进的方式，在序列生成对抗网络的基础上，依次增加自注意力模型、强化自注意力模型与基于惩罚的目标函数。

此处按照上文的训练方法对 PB-ESGAN 模型进行训练，截图展示如下：
首先预训练生成器，仍然把训练设置在 150-200 次：

```
Starting Generator MLE Training...
[MLE-GEN] epoch 0 : pre_loss = 6.5744, BLEU-[2] = [0.108],
[MLE-GEN] epoch 5 : pre_loss = 4.4034, BLEU-[2] = [0.183],
[MLE-GEN] epoch 10 : pre_loss = 4.0324, BLEU-[2] = [0.211],
[MLE-GEN] epoch 15 : pre_loss = 3.8781, BLEU-[2] = [0.208],
[MLE-GEN] epoch 20 : pre_loss = 3.7905, BLEU-[2] = [0.216],
[MLE-GEN] epoch 25 : pre_loss = 3.7335, BLEU-[2] = [0.224],
[MLE-GEN] epoch 30 : pre_loss = 3.6933, BLEU-[2] = [0.218],
[MLE-GEN] epoch 35 : pre_loss = 3.6627, BLEU-[2] = [0.217],
[MLE-GEN] epoch 40 : pre_loss = 3.6416, BLEU-[2] = [0.209],
[MLE-GEN] epoch 45 : pre_loss = 3.6176, BLEU-[2] = [0.215],
[MLE-GEN] epoch 50 : pre_loss = 3.6127, BLEU-[2] = [0.223],
[MLE-GEN] epoch 55 : pre_loss = 3.5945, BLEU-[2] = [0.211],
[MLE-GEN] epoch 60 : pre_loss = 3.5788, BLEU-[2] = [0.206],
[MLE-GEN] epoch 65 : pre_loss = 3.5688, BLEU-[2] = [0.218],
[MLE-GEN] epoch 70 : pre_loss = 3.5659, BLEU-[2] = [0.221],
[MLE-GEN] epoch 75 : pre_loss = 3.5557, BLEU-[2] = [0.226],
```

图 4-2 PB-ESGAN 训练生成器

接下来按照训练方法，训练判别器数次，仍然设置为 4-5 次：

```
Starting Discriminator Training...
[MLE-DIS] d_step 0: d_loss = 0.1723, train_acc = 0.9298,
[MLE-DIS] d_step 1: d_loss = 0.0162, train_acc = 0.9986,
[MLE-DIS] d_step 2: d_loss = 0.0029, train_acc = 0.9999,
[MLE-DIS] d_step 3: d_loss = 0.0018, train_acc = 0.9997,
[MLE-DIS] d_step 4: d_loss = 0.0022, train_acc = 0.9998,
```

图 4-3 PB-ESGAN 训练判别器

在生成器和判别器进行联合训练前，再对生成器完成初始化：

```
Starting Adversarial Training...
Initial generator: BLEU-[2] = [0.218],
```

图 4-4 生成器初始化

完成初始化后，仍然按照训练方法，依据每轮分别训练数次判别器及一轮生成器的顺序，训练数论，在训练中整个生成对抗网络会不断优化，即训练的轮数越多，生成诗歌文本的质量会越高：

```
ADV EPOCH 0
-----
[ADV-GEN]: g_loss = 689.3364, BLEU-[2] = [0.213],
[ADV-DIS] d_step 0: d_loss = 0.0039, train_acc = 0.9994,
[ADV-DIS] d_step 1: d_loss = 0.0022, train_acc = 0.9997,
[ADV-DIS] d_step 2: d_loss = 0.0035, train_acc = 0.9992,
[ADV-DIS] d_step 3: d_loss = 0.0030, train_acc = 0.9995,
-----
ADV EPOCH 1
-----
[ADV-GEN]: g_loss = 209.4891, BLEU-[2] = [0.221],
[ADV-DIS] d_step 0: d_loss = 0.0032, train_acc = 0.9993,
[ADV-DIS] d_step 1: d_loss = 0.0037, train_acc = 0.9997,
[ADV-DIS] d_step 2: d_loss = 0.0019, train_acc = 0.9992,
[ADV-DIS] d_step 3: d_loss = 0.0020, train_acc = 0.9991,
-----
ADV EPOCH 2
-----
[ADV-GEN]: g_loss = 82.2016, BLEU-[2] = [0.230],
[ADV-DIS] d_step 0: d_loss = 0.0024, train_acc = 0.9998,
[ADV-DIS] d_step 1: d_loss = 0.0023, train_acc = 0.9994,
[ADV-DIS] d_step 2: d_loss = 0.0018, train_acc = 0.9996,
[ADV-DIS] d_step 3: d_loss = 0.0016, train_acc = 0.9997,
```

图 4-5 PB-ESGAN 训练

整个训练结束后，得到算法模型生成诗歌文本的 BLEU-2 指标值，同样，另外三种对比模型也都按照相同的训练方法完成训练。表 4-2 中列出四种算法在绝句诗歌语料数据集上训练实现的 BLEU-2 值：

表 4-2 绝句诗歌数据集的实验结果

模型	BLEU-2
seqGAN	0.189
seqGAN + Transformer	0.197
seqGAN + Transformer with Modeling Localness	0.229
Our Model	0.231

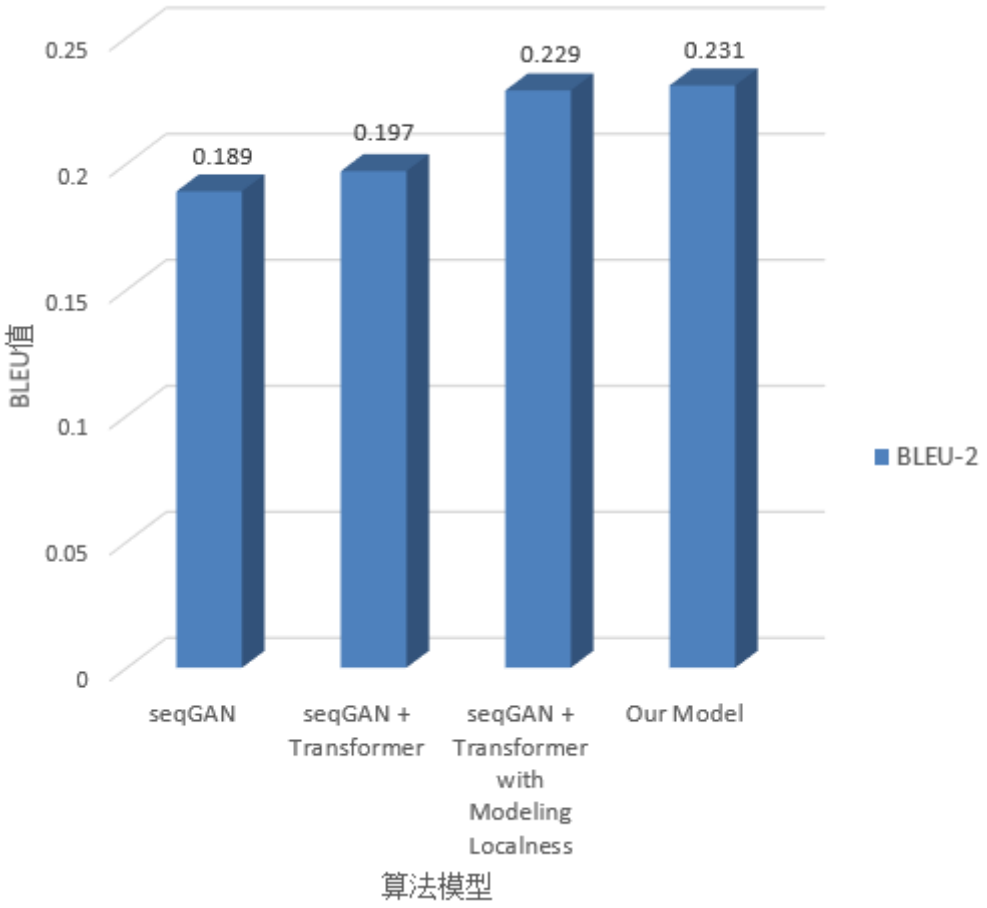

图 4-6 绝句诗歌数据集上四种算法的 BLEU-2 值

此外，表 4-3 中列出四种算法的格律分数对比，分别列出音调、韵律、结构值，直观地展示诗歌生成的评估效果：

表 4-3 绝句诗歌数据集的实验结果

模型	音调	韵律	结构	格律分数
seqGAN	0.824	0.207	0.991	0.522
seqGAN + Transformer	0.819	0.159	0.993	0.488
seqGAN + Transformer with Modeling Localness	0.827	0.281	0.981	0.556
Our Model	0.832	0.516	0.987	0.687

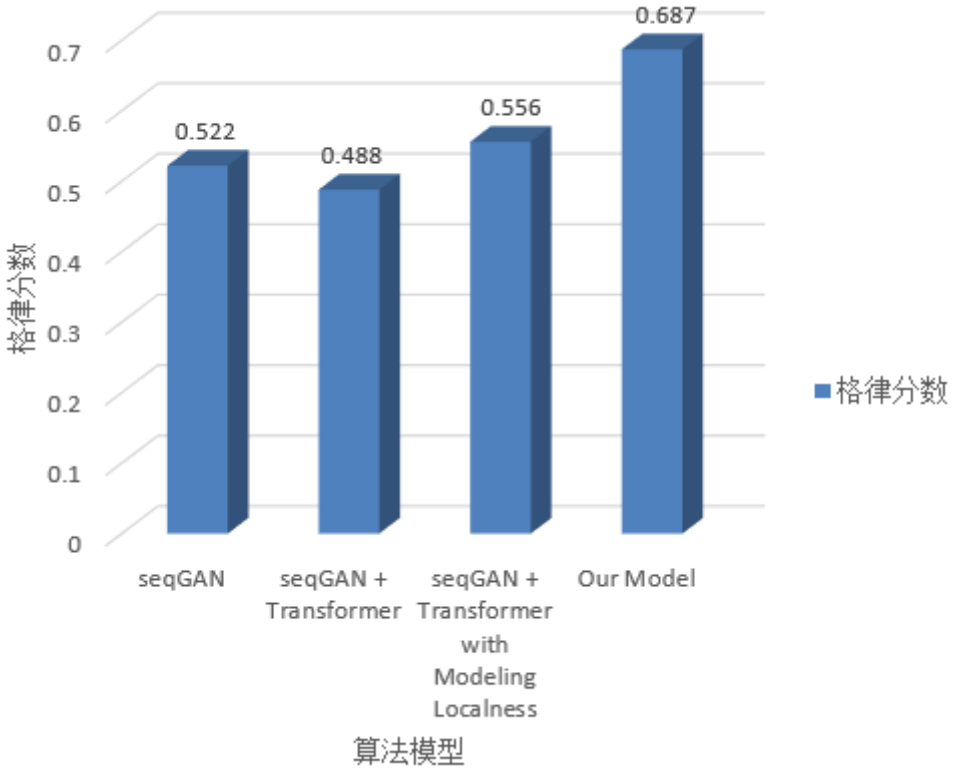

图 4-7 绝句诗歌数据集上四种算法的格律分数

首先，通过表 4-2 可以很直观地感受到，与 PB-ESGAN 应用在长文本中的优越效果类似，应用在短文本中也能够得到明显的 BLEU 指标提升。在序列生成对抗网络基础上增加自注意力模型，BLEU-2 的指标得到了+0.8 个 BLEU 点的提升，当将自注意力模型替换为引入局部建模的自注意力时，则在序列生成对抗网络的基础上能够提高+4.0 个 BLEU 点。加入基于惩罚的损失函数的提升效果也很明显，比起加入了带有局部建模的自注意力模型，BLEU-2 能够再提高+0.2 个点。

之后对格律分数这一指标进行分析，结构是最容易通过自主学习学到的特征，可以看到四种算法的结构分数都较高。另外音调分数中，本研究的模型也通过更强的自主学习能力训练得到更高的分数。更大的提升体现在韵律分数中，本研究在 PB-ESGAN 的基础上增加了押韵判断，这也直接导致本研究模型的韵律分数相比前几个模型有了十分显著的进步。综合三个指标得到的格律分数，也能够感受到本研究算法的优越性。

另外，表 4-4 列出了 PB-ESGAN 生成的诗歌样例，并与格律规则进行比对，以便更加直观地表现生成诗歌的效果。可以看出，在结构方面，生成诗歌完全符合五言规则；在押韵方面，生成诗句的末尾的押韵也符合格律规则的韵脚要求；而音调方面，则与几个典型格律规则中的仄起首句入韵式完全匹配。即生成的诗歌完全

符合传统诗歌的格律规则，证实了 PB-ESGAN 在文本生成方面的有效性。

表 4-4 PB-ESGAN 生成的诗歌样例与五言绝句规则之一

PB-ESGAN 生成的诗歌样例	格律规则（仄起首句入韵式）
夜气故人游， 春流雪影柔。 悠然何处所， 去处正留愁。	仄仄仄平平（韵）， 平平仄仄平（韵）。 平平平仄仄， 仄仄仄平平（韵）。

作为第三章长文本内容生成的补充，本章采用绝句短文本进行训练，实验结果证明，本研究的算法模型具有较好的普适性，成功地应用在绝句生成领域并取得了评估指标的提高，实现了更好的文本生成效果。

4.5 本章小结

本章的研究目的为证明 PB-ESGAN 的普适性，第三章将这一算法模型应用于长文本生成，并取得了较好的效果，本章选择绝句这一极富特色的文本进行普适性验证。绝句文本具有短文本及格律规则较强的特点，与长文本有较大的区分度，用于印证模型的普适性有较强的说服力。本章仍然沿用第三章的算法模型基础，并结合绝句押韵的特点，在原模型上增加了押韵判断模块，借此提高生成文本在押韵上的表现。

在实验中，本章选择 Chinese-poetry 诗歌数据库作为数据集，同样选择序列生成对抗网络、序列生成对抗网络+Transformer、以及序列生成对抗网络+强化 Transformer（引入局部建模）这三个模型进行对比实验。评估指标方面选择了更适合评价短文本的 BLEU-2 值，同时由于格律规则的存在，另外选择了格律分数作为另一个评估指标，以便更好地判断生成绝句的质量。

实验结果表明，本研究的 PB-ESGAN 算法在不同类型文本上都有较好的生成效果，有较强的普适性。在未来的计划中，尝试将模型落地在更多具体的应用场景中，并尝试进一步缩短模型训练学习的时间，都是重要的研究方向。

第五章 全文总结与展望

5.1 全文总结

本文对基于生成对抗网络的文本生成问题进行研究，分析并实现了部分前人的研究成果后，针对生成对抗网络框架存在的模式崩溃与提取文本特征的能力较弱的问题，提出了相关的改进方法，提出了基于惩罚最小化强化序列生成对抗网络的算法模型。并且将其应用在具有不同特征的文本数据集上，通过针对不同特征的优化确保本模型具有普适性。本文的主要贡献如下：

1. 针对原始序列生成对抗网络中生成器无法并行化的问题，提出将自注意力机制与生成对抗网络相结合，将应用了自注意力机制的 Transformer 模型应用于生成器；同时提出在 Transformer 中融合了高斯偏差的局部建模，用以提升其提取文本特征的能力。
2. 将基于惩罚的目标函数应用于序列生成对抗网络中，替换原有的奖励机制，新的目标函数可以视作一种衡量 Wasserstein 距离的方式，保证了有意义的梯度，不会出现梯度消失；另外使用损失项而非奖励项，改善了模式崩溃的问题，使得模型生成多样性更丰富的文本。
3. 针对长文本生成领域，本文选择了电商领域的文本数据集，之后根据训练需要对数据完成预处理，把文本提出的生成对抗网络模型与另外三个基线模型在电商数据集上进行实验对比，并依据文本生成的特有评价指标进行评估，比较得出结论。
4. 为了证实本文所提出模型的普适性，将其复用在诗歌生成领域，之后在模型中引入了汉字与拼音的对照表，并增加了押韵判断算法，以便生成质量更高、更符合韵律规则的诗歌文本；另外选择了绝句作为文本数据集，与基线模型的实验结果进行对比，得出结论。

5.2 后续工作展望

本文提供了基于序列生成对抗网络的长文本生成方案以及符合韵律规则的诗歌生成方案，在文本生成质量与模式崩溃问题的改善上做出一定贡献。基于上述的研究内容，对未来可以提出以下深入研究方向：

1. 在长文本生成质量的评估中，本文选用了文本生成常见的评价指标 BLEU，这一指标的本质是基于目标文本和参考文本的 n-gram 完成评分，

但对于长文本中特有的前后语义连贯与结构完整并没有给出评价。从实验结果的文本样例中可以看出，本模型生成的文本基本实现了语义的通顺，但目前还没有合理的公开评价指标进行分析与评估。因此，未来可以提出一种可靠的新评价指标，实现长文本领域评估指标的完善。

2. 生成对抗网络在应用于自然语言处理领域中时，会面临模型过分复杂，以至于需要实验所用硬件设备满足较高条件的问题，同时也会出现训练时间过长的情况，本研究也同样出现了此类问题。如何在算法模型层面直接作出改进，以改善模型的复杂度，提高模型训练效率，缩短训练时间，也是未来需要解决的问题。

文本生成这一领域面临的问题远不止上文中提及的内容，进一步提高文本质量等需求也等待着研究人员们继续解决。随着更多高质量的文本生成模型提出，未来的文本生成任务也一定会得到突破性的提升。

致　谢

研究生三年时光倏忽即逝，再有短短数月就要结束。在这样的时间点里，校园里错落有致的楼宇、郁郁葱葱的草木、擦肩而过的人群，都显得更加亲切可爱起来。接到电子科技大学录取通知书的日子仿佛就在昨天，转眼已过近一千个日夜。感谢母校给了我机会和平台，攻读硕士这三年，我开阔了眼界、有了自己的科研成果、交到了许多好朋友……最重要的是，我更加了解自己、接纳自己，对人生的规划也愈发清晰。在这毕业之际，向所有关心我、支持我的人由衷道一声感谢！

首先，我要由衷感谢我的研究生导师李玉柏教授。李老师不仅在科研道路上指引与帮助我，更是给了我充分的空间去寻找科研之外的人生志向。硕士期间我的诸多收获和成长，都离不开李老师的理解与支持。

另外，我要感谢一直陪伴我、鼓励我的父母。感谢他们足够开明，尊重我的每一个决定；也感谢他们的包容和鼓励，在我遇到困境时给予我最大的支持。他们永远是我成长路上最坚实的后盾。

我还要感谢三年认识的所有朋友同学，我们在三年时间里彼此陪伴，研一上课抢座位、研会办活动；研二搞科研发论文、投简历找工作；研三做毕设准备答辩，很快也会有许多旅行和聚会……三年中的大小事情，总有不同朋友的身影与我相伴，感谢结识的每一个好朋友，让这三年无比快乐和难忘。

最后，感谢每一位百忙之中抽出时间审阅本文的老师、学者和专家！

参考文献

[1] Kalchbrenner N, Blunsom P. Recurrent continuous translation models[C]. Proceedings of the 2013 conference on empirical methods in natural language processing. 2013: 1700-1709.

[2] Vinyals O, Toshev A, Bengio S, et al. Show and tell: A neural image caption generator[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 3156-3164.

[3] Rush A M, Chopra S, Weston J. A neural attention model for abstractive sentence summarization[J]. arXiv preprint arXiv:1509.00685, 2015.

[4] Serban I, Sordoni A, Bengio Y, et al. Building end-to-end dialogue systems using generative hierarchical neural network models[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2016, 30(1).

[5] Morin F, Bengio Y. Hierarchical probabilistic neural network language model[C]. Aistats. 2005, 5: 246-252.

[6] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]. Eleventh annual conference of the international speech communication association. 2010.

[7] Ranzato M A, Chopra S, Auli M, et al. Sequence level training with recurrent neural networks[J]. arXiv preprint arXiv:1511.06732, 2015.

[8] Naha S, Wang Y. Beyond verbs: Understanding actions in videos with text[C]. 2016 23rd International Conference on Pattern Recognition (ICPR). IEEE, 2016: 1833-1838.

[9] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks[J]. arXiv preprint arXiv:1406.2661, 2014.

[10] Kusner M J, Hernández-Lobato J M. Gans for sequences of discrete elements with the gumbel-softmax distribution[J]. arXiv preprint arXiv:1611.04051, 2016.

[11] Zhang Y, Gan Z, Carin L. Generating text via adversarial training[C]. Neural Information Processing Systems workshop on Adversarial Training. 2016, 21.

[12] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]. Neural Information Processing Systems. 1999, 99: 1057-1063.

[13] Lantao Yu, Weinan Zhang, Jun Wang, et al. Sequence generative adversarial nets with policy gradient. arxiv e-prints, page[J]. arXiv preprint arXiv:1609.05473, 2016.

[14] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[15] Salakhutdinov R. Learning deep generative models[J]. Annual Review of Statistics and Its Application, 2015, 2: 361-385.

[16] Zhang X, Lapata M. Chinese poetry generation with recurrent neural networks[C]. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014: 670-680.

[17] Bengio S, Vinyals O, Jaitly N, et al. Scheduled sampling for sequence prediction with recurrent neural networks[J]. arXiv preprint arXiv:1506.03099, 2015.

[18] Huszár F. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? [J]. arXiv preprint arXiv: 1511.05101, 2015.

[19] Kingma D P, Welling M. Auto-encoding variational bayes[J]. arXiv preprint arXiv:1312.6114, 2013.

[20] Bowman S R, Vilnis L, Vinyals O, et al. Generating sentences from a continuous space[J]. arXiv preprint arXiv:1511.06349, 2015.

[21] Touseef Iqbal,Shaima Qureshi. The Survey: Text Generation Models in Deep Learning.[J]. Journal of King Saud University - Computer and Information Sciences,2020.

[22] Semeniuta S, Severyn A, Barth E. A hybrid convolutional variational autoencoder for text generation[J]. arXiv preprint arXiv:1702.02390, 2017.

[23] Salakhutdinov R. Learning deep generative models[J]. Annual Review of Statistics and Its Application, 2015, 2: 361-385.

[24] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.

[25] Bengio Y, Yao L, Alain G, et al. Generalized denoising auto-encoders as generative models[J]. arXiv preprint arXiv:1305.6663, 2013.

[26] Kong T, Yao A, Chen Y, et al. Hypernet: Towards accurate region proposal generation and joint object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 845-853.

[27] Denton E, Chintala S, Szlam A, et al. Deep generative image models using a laplacian pyramid of adversarial networks[J]. arXiv preprint arXiv:1506.05751, 2015.

[28] Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax[J]. arXiv preprint arXiv:1611.01144, 2016.

[29] Zhang Y, Gan Z, Fan K, et al. Adversarial feature matching for text generation[C]. International Conference on Machine Learning. PMLR, 2017: 4006-4015.

[30] Che T, Li Y, Zhang R, et al. Maximum-likelihood augmented discrete generative adversarial

networks[J]. arXiv preprint arXiv:1702.07983, 2017.

[31] Lin K, Li D, He X, et al. Adversarial ranking for language generation[J]. arXiv preprint arXiv:1705.11001, 2017.

[32] Bachman P, Precup D. Data generation as sequential decision making[J]. arXiv preprint arXiv:1506.03504, 2015.

[33] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]. Neural Information Processing Systems. 1999, 99: 1057-1063.

[34] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[J]. arXiv preprint arXiv:1409.3215, 2014.

[35] Wang K, Wan X. SentiGAN: Generating Sentimental Texts via Mixture Adversarial Networks[C]. International Joint Conference on Artificial Intelligence. 2018: 4446-4452.

[36] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[J]. arXiv preprint arXiv:1310.4546, 2013.

[37] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[38] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. The journal of machine learning research, 2003, 3: 1137-1155.

[39] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]. Eleventh annual conference of the international speech communication association. 2010.

[40] Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning[C]. International Conference on Machine Learning. PMLR, 2017: 1243-1252.

[41] Zhou Shuohua. Research on the Application of Deep Learning in Text Generation[J]. Journal of Physics: Conference Series, 2020.

[42] Lai S, Xu L, Liu K, et al. Recurrent convolutional neural networks for text classification[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2015, 29(1).

[43] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[44] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks[C]. Advances in neural information processing systems 25 (2012): 1097-1105.

[45] Kim Y. Convolutional neural networks for sentence classification[J]. arXiv preprint arXiv:1408.5882, 2014.

[46] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image

recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[47] Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 4700-4708.

[48] Arjovsky, Martin, Soumith Chintala, and Léon Bottou. Wasserstein GAN [J]. arXiv preprint arXiv:1701.07875, 2017.

[49] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need [J]. arXiv preprint arXiv:1706.03762, 2017.

[50] Yang, Baosong, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. Modeling Localness for Self-Attention Networks [J]. arXiv preprint arXiv:1810.10182, 2018.

[51] Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation[C]. Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002: 311-318.

攻读硕士学位期间取得的成果

已发表论文：

[1] Mingjun Duan and Yubai Li, "Penalty-based Sequence Generative Adversarial Networks with Enhanced Transformer for Text Generation," 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 2020

参与项目：

[1] 四川省科技厅重点研发项目[2020YFG0142]：基于知识图谱的智能问答系统研究