

电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

# 专业学位硕士学位论文

MASTER THESIS FOR PROFESSIONAL DEGREE



论文题目 基于深度学习的台风路径预测多模型算法研究

专业学位类别 工 程 硕 士

学 号 201722061021

作 者 姓 名 王瀚

指 导 教 师 侯孟书 教授

分类号 \_\_\_\_\_ 密级 \_\_\_\_\_

UDC <sup>注1</sup> \_\_\_\_\_

# 学 位 论 文

基于深度学习的台风路径预测多模型算法研究

(题名和副题名)

王瀚

(作者姓名)

指导教师

侯孟书

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 专业学位类别 工程硕士

工程领域名称 计算机技术

提交论文日期 2020.3.25 论文答辩日期 2020.5.19

学位授予单位和日期 电子科技大学 2020 年 6 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 Multi-Model Algorithm for Typhoon Prediction based on Deep Learning**

A Master Thesis Submitted to  
University of Electronic Science and Technology of China

Discipline: **Master of Engineering**

Author: **Han Wang**


Supervisor: **Mengshu Hou**

School: **School of Computer Science & 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：



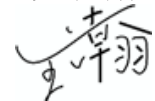
日期： 2020 年 6 月 1 日

## 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：



导师签名：



日期： 2020 年 6 月 1 日

## 摘 要

过去几十年,台风频频袭击我国,对沿海地区人民的生命和财产带来了不可估量的损失。如何准确预测台风路径,减少沿海地区人民损失,已成为当下热门研究课题。由于影响台风轨迹走向的因素众多,特征提取难度大,传统方法需要结合众多相关领域的先验知识,不仅耗时耗力,而且预测精度不高。近年来,随着深度学习技术的在众多领域取得突破,越来越多研究人员开始尝试将深度学习技术引入台风轨迹预测,取得了许多不错的成果。基于此,本文通过深度学习的方法进行台风轨迹预测,并通过多模型融合技术,对序列数据和图像数据预测结果进行融合,旨在提高现下方法预测精度,具体研究成果如下:

1) 针对台风轨迹序列数据,使用基于注意力机制的 Seq2seq 模型,提高预测精度。首先针对卡尔曼滤波算法结果受异常观测值影响大的缺陷,创新性地提出了改进的卡尔曼滤波算法,有效剔除野值,提升轨迹质量;接着针对轨迹采样过于频繁的问题,使用最小扇形简化算法,提升训练速度;最后建立深度学习模型进行台风轨迹预测,使用编码器和解码器解决序列输入输出不等长的问题,使用注意力机制优化模型的时序依赖,只关注关键的部分时序,剔除掉无关紧要部分。实验结果表明该方法能很好的利用数据特征和时序性,预测轨迹的走向。

2) 针对台风卫星图像数据,创新性地提出基于时序的 GAN 模型,提升预测精度。首先使用局部均值算法对图像数据进行预处理,以便更好关注图像局部特征;接着建立深度学习模型进行台风轨迹预测。传统的 CNN 网络生成图像是各种可能情况的平均值,生成效果不理想。而 GAN 模型是直接生成数据的分布,生成器负责从噪声数据中生成新的图像,判别器负责识别生成的图像“真伪”,二者动态博弈,直至纳什均衡。在 GAN 模型基础上,引入时序性,将前序时刻的图像作为输入,预测后序时刻的台风轨迹,能较好的提升现有方法在台风图像数据上的预测精度。实验结果表明该方法能很好学习轨迹的形状,应对突发情况。

3) 创新性地将序列数据和图像数据的预测结果进行多模型融合,进一步提升预测精度。不同算法对于不同的数据集表现不同,每个算法都有自己擅长的领域,进行多模型融合可以更好的“博采众长”。使用遗传算法 GASEN 进行多模型融合,通过遗传算法进化预测结果之间的权重,针对不同模型的优点对结果的融合进行加权平均。实验结果表明,融合后预测精度高于现有方法。

**关键词:** 台风轨迹预测, Seq2seq 模型, GAN 模型, 多模型融合

## ABSTRACT

In the past few decades, typhoons have hit our country frequently, causing immeasurable losses to people's lives and property in coastal areas. How to accurately predict the typhoon path and reduce the loss of people in coastal areas has become a hot research topic now. Due to the many factors affecting typhoon trajectories and the difficulty of feature extraction, traditional methods require more prior knowledge in related fields, which is not only time-consuming and labor-intensive, but also has low prediction accuracy. In recent years, with the development of deep learning technology, more and more researchers have begun to introduce deep learning technology into typhoon trajectory prediction, and have also achieved many good results. Based on this, this article uses deep learning to predict the typhoon trajectory, and uses multiple model fusion technology to fuse the prediction results of sequence data and image data to improve the prediction accuracy of the current method:

1) For the typhoon trajectory sequence data, the Seq2seq model based on the attention mechanism is used to improve the prediction accuracy. First of all, for the defect that the results of the Kalman filter algorithm are greatly affected by abnormal observations, an improved Kalman filter algorithm is proposed to effectively remove outliers and improve the quality of trajectory data. Then, for the problem of trajectory sampling too frequently, the minimum fan-shaped simplified algorithm is used. Simplify the trajectory and improve the training speed. Finally, build a model to predict the typhoon trajectory, use encoders and decoders to solve the problem of unequal sequence input and output, use attention mechanism to optimize the model's timing dependency, and improve the prediction accuracy of existing methods. The experimental results show that the method can make good use of data features and time series to predict the direction of the trajectory.

2) Aiming at the typhoon satellite image data, a GAN model based on time series is proposed to improve the prediction accuracy. First use digital coding algorithms to preprocess image data in order to better focus on local features of the image; then the temporality is introduced into the GAN model and applied to the prediction of typhoon trajectories. In traditional CNN networks, the image generation is just the average of various possible situations, and the generation accuracy is not high. The GAN directly

generates the distribution of data. The generator is responsible for generating new images from the noise, and the discriminator is responsible for identifying the "authenticity" of the generated images. The two form a dynamic game until the Nash equilibrium. As the input of the GAN, the pictures at the previous sequence time can predict the typhoon trajectory at the later sequence time, which can effectively improve the prediction accuracy of the existing method on the typhoon image data. Experimental results show that the method can learn the shape of the trajectory well and cope with unexpected situations.

3) Multi-model fusion of the prediction results of sequence data and the prediction results of image data is innovatively used to further improve the prediction accuracy. Different algorithms perform differently on different datasets, and each algorithm has its own area of expertise. Multi-model fusion can better "boiler people." The genetic algorithm GASEN is used for multi-model fusion. The weights between the results of different models are weighted and averaged based on the weights between the evolutionary prediction results of the genetic algorithm. Experimental results show that the prediction accuracy after fusion is higher than all existing methods.

**Keywords:** Typhoon trajectory prediction, Seq2seq model, GAN model, Multi-model

# 目 录

<b>第一章 绪 论</b> .....	1
1.1 研究背景及意义.....	1
1.2 国内外研究现状.....	2
1.2.1 深度学习技术的发展现状.....	3
1.2.2 台风轨迹预测对的发展现状.....	4
1.3 本文的研究内容和结构安排.....	5
1.4 论文结构安排.....	6
<b>第二章 理论基础及相关技术</b> .....	7
2.1 RNN 模型及其演化.....	7
2.1.1 RNN 模型.....	7
2.1.2 LSTM 模型.....	9
2.1.3 Seq2seq 模型.....	11
2.2 GAN 模型及其演化.....	13
2.2.1 基础 GAN 模型.....	13
2.2.2 DCGAN 模型.....	15
2.3 Pytorch 框架.....	16
2.4 本章小结.....	18
<b>第三章 基于序列数据的台风轨迹预测</b> .....	19
3.1 数据预处理.....	19
3.1.1 轨迹数据滤波.....	19
3.1.2 轨迹数据简化.....	22
3.2 方案设计.....	25
3.2.1 台风轨迹数据分析.....	25
3.2.2 激活函数和优化策略.....	29
3.2.3 模型结构设计.....	33
3.3 实验结果.....	35
3.3.1 台风轨迹清洗.....	35
3.3.2 台风轨迹预测.....	36
3.4 本章小结.....	39
<b>第四章 基于图像数据的台风轨迹预测</b> .....	41



4.1 数据预处理.....	41
4.1.1 图像格式转化 .....	41
4.1.2 图像数据裁剪 .....	44
4.2 方案设计.....	46
4.2.1 全卷积网络 .....	46
4.2.2 模型架构设计 .....	48
4.2.3 多模型融合 .....	51
4.3 实验结果.....	53
4.3.1 台风轨迹预测 .....	54
4.3.2 多模型融合 .....	57
4.4 本章小结.....	59
<b>第五章 总结与展望.....</b>	<b>60</b>
5.1 全文总结 .....	60
5.2 后续工作展望.....	61
<b>致 谢.....</b>	<b>62</b>
<b>参考文献.....</b>	<b>63</b>
<b>攻读硕士学位期间取得的成果 .....</b>	<b>66</b>

## 第一章 绪 论

过去几十年中,台风对沿海地区人民的生命和财产安全造成了巨大的损失。如何有效预测台风路径,更好地为防灾部门提供帮助,成为当下一个重要的研究课题。但影响台风轨迹的因素众多,特征选取难度大,传统的预测方法不仅耗时耗力,而且精度不高,对有关部门的帮助十分有限。近年来随着互联网数据量的飞速增加和计算机硬件的飞速发展,深度学习在各个领域备受关注,越来越多学者开始将深度学习方法引入台风轨迹预测,以提升传统方法的预测精度。本章首先分析台风轨迹预测的研究背景和研究意义,然后分析国内外研究现状和已有成果,最后介绍本文的组织结构。

### 1.1 研究背景及意义

台风是一种灾害性的热带天气系统,对沿海地区人民的生命安全和区域经济的发展构成了严重的威胁。准确及时的台风预报,可以为防灾部门提供有效的信息支持,大大减少生命和财产的损失。早期对台风轨迹的预测主要是通过结合热力学和动力学知识<sup>[1]</sup>以及分析沿海地区复杂的地形和海岸线特征,建立数据分析模型,进行台风轨迹预测。但受限于对台风先验知识的积累不足以及准确选择预测所需特征的选取难度较大,此种方法不仅耗时耗力,而且预测的精度不高,对于台风防灾的帮助十分有限。近年来,随着互联网时代数据的爆发式的增长以及以 Intel、Nvidia 为首的公司促进计算机硬件的飞速发展,深度神经网络被用于日常生活的方方面面同时也得到研究人员的高度重视,人类在自然语言处理<sup>[2]</sup>、计算机视觉<sup>[3]</sup>、自动驾驶<sup>[4]</sup>等领域取得了极大的进步。基于此,有越来越多的人开始思考,是否可以借鉴在大数据背景下深度学习技术在其他领域成功的经验,应用于台风轨迹预测,提升传统方法的预测精度,更好地帮助防灾部门,减少损失。

传统的机器学习一般由数据预处理、特征提取和分类器选择三个流程构成。其中特征提取最为重要,特征选择的质量往往能够对于算法的效果起着决定性的作用。传统机器通过人手动进行特征的选取和特征工程的构建,对于数据结构比较复杂的问题很难通用。深度学习<sup>[5]</sup>是机器学习的一个重要分支,不同于传统机器学习,深度学习是一种面向数据的学习算法。深度学习的不需要领域专家进行特征筛选和特征工程,而是依靠模型自动从数据中学习特征。相比较传统的机器学习,深度学习方法有更强的学习能力,覆盖的范围更广,可移植性也更强,但同时对历史数据和算力的需求也更高。大量的事实证明,在海量数据支撑的场景下,深度学习模

型的效果远好于传统的机器学习方法。从中可以看到,无论是在学术界前沿的研究成果,还是工业界落地一些造福于民的落地应用,绝大多数都是基于深度学习的。首先在 web2.0 时代,数据发生了爆发式的增长,通过卫星采集了大量的台风数据集,最近几年,相关机构工作人员对台风轨迹数据集处理做了大量的工作,为研究者提供了许多高质量的台风轨迹数据集;其次是当下硬件技术的高速发展,硬件成本显著降低,计算机算力的大大提高,大大缩短了模型训练所需的时间;最后得益于 Google、Facebook 等世界上优秀的科技巨头公司研发出了诸如 Tensorflow<sup>[6]</sup>、Pytorch<sup>[7]</sup>等方便、易用的深度学习框架,降低了科研实验的成本。上述三点,为使用深度学习的方法提高台风轨迹预测的精度提供了可能。

同时也应该清楚的认识到的,不同于当下成熟的深度学习落地应用,台风轨迹数据集具有其特殊性。一方面,台风轨迹数据并不是一种固定长度的向量,而是一种长度不确定的时间序列数据<sup>[8]</sup>。这就使得一些目前在图像领域耳熟能详的深度学习模型,诸如:卷积神经网络<sup>[9]</sup> (Convolutional Neural Networks, CNN)、残差网络<sup>[10]</sup> (Residual Network, ResNet)、fast R-CNN<sup>[11]</sup>。等并不直接适用。另一方面,由于影响台风轨迹预测准确性的因为较多,对于过去的状态依赖过长,很容易在神经网络反向传播的过程中造成梯度消失和梯度爆炸的问题,一些跟时间序列相关预测相关的神经网络,诸如:循环神经网络<sup>[12]</sup> (Recurrent Neural Network, RNN)、长短期记忆网络<sup>[13]</sup> (Long Short-Term Memory, LSTM)、GRU<sup>[14]</sup>等,也没法取得理想效果。

综上所述,在现有的研究成果中,缺少一种能够完全适用台风轨迹预测的深度学习方法。幸运的是,近几年对抗生成网络<sup>[15]</sup> (Generative Adversarial Networks, GAN) 技术在图像生成领域取得了巨大的突破,生成的图像和原始图像能够达到高度一致,这为生成台风轨迹的热力图提供了可能。另一方面,近年来随着注意力机制<sup>[16]</sup>的 Seq2Seq 模型<sup>[17]</sup>在自然语言处理取得的巨大成功,时间序列预测问题中存在的诸如梯度爆炸、梯度消失的问题得到了较好的解决。通过将 Seq2seq 技术和 GAN 技术结合起来,基于多模型对台风轨迹数据进行预测,能很大程度上提升预测精度,更好的帮助防灾部门,减少沿海地区人民的生命和财产损失。

## 1.2 国内外研究现状

如前文所述,提升预测台风轨迹的精度,减少人民生命和财产损失,无论在学术界还是在工业界都是非常重要的研究课题,同时也有许多难题待解决。想要解决该领域所面临的难题,就必须更全面的了解当下国内外研究进展。本小节,首先介绍深度学习技术的发展和研究现状,接下来,将详细介绍国内外台风轨迹预测,特别是基于国内外深度学习的台风轨迹预测的发展和研究现状。

### 1.2.1 深度学习技术的发展现状

深度学习是人工智能目前在各领域应用最成功的技术。作为机器学习的分支，深度学习是一种非常重要的工具。通过学习人脑的神经系统，建立的模型与人类一样智能。深度学习起源于上世纪 60 年代，起初被称为“感知器”模型，该模型具有 2 层神经元。“感知器”是一种线性分类器模型，研究者使用它对数据集进行二分类，并自动更新训练集中的参数和权值，这项研究开启了深度学习研究的先河。然而无法应对“感知器”对于非线性分类问题，而这部分问题又在生活占据绝大多数，这一致命缺陷使得深度学习的不被其他的学派认可，神经网络第一次陷入发展的瓶颈期。上世纪 80 年代，重要的反向传播算法——反向传播算法<sup>[18]</sup> (back propogation, BP 算法) 被研究者们提出，在之前的研究中，神经网络模型只能进行正向传播，而这次因为反向传播的引入研究取得了重大的突破。反向传播可以通过迭代不断更新整个模型神经元的权值，直到输出的结果足够完美可以去拟合期望的结果，或迭代的次数达到训练前设定的要求。BP 算法的出现让深度学习有了解决非线性的分类的能力，学者们也再次开始关注神经。但是由于神经网络参数非常玄学且具有很差的可解释性，同时期的硬件水平也只能支持浅层的神经网络进行学习，深度学习在当时未能被人们所认可，再加上那时传统的机器学习算法可谓百花齐放，在许多分类、回归问题上取得了较好的效果的同时对于硬件的需要也远低于深度学习，深度学习再次陷入了低潮期。近年来，伴随着互联网行业数据量的暴增，以及 Intel、Nvidia 等公司在计算机硬件提升上所作的努力，计算机算力的大大提高，深度学习再次引起人们的关注。随着 2012 年 AlexNet 网络<sup>[19]</sup>在 ImageNet<sup>[20]</sup>挑战赛以压倒性的优势夺得冠军，ReLU<sup>[21]</sup>函数提出使得困扰图像领域多年的梯度消失问题从根本上得到了解决。同时，第一次使用 Nvidia 公司的 GPU 极大加速了浮点运算和矩阵运算，深度学习再次得到了学术界和工业界广泛关注，并开始进行大量投资和研究。2016 年谷歌公司第一次将深度学习用于下围棋，其开发的 AlphaGo<sup>[22]</sup>在围棋上的水平已经超越了世界冠军李世石，深度学习的发展更是一时间被推上风口浪尖。

比起传统的机器学习，深度学习具有很强的可移植性。近年来，学术界和工业界对于深度学习的研究飞速发展，新算法、新模型的研发和落地需求不断加深。诸如 Tensorflow、Pytorch 等优质的深度学习框架应运而生，在为专业的研究人员进行实验带来方便的同时，也使得深度学习技术得到了极大的普及。可以说当下深度学习技术应用不仅仅只限于计算机科研人员，也被应用于医疗、生物、金融、教育等各行各业，极大促进了社会的进步。

### 1.2.2 台风轨迹预测对的发展现状

台风轨预测是一个非常重要,而又充满挑战的研究课题,在国内外无论是学术界还是工业界,都引起相关人员的高度关注。早期的台风轨迹预测,主要依靠热力学和空气动力学的专家对台风背景场进行分析,结合勘探领域专家对沿海地区复杂的海岸线以及内陆地形等影响因素的分析,建立台风领域特有的经验法则,以预测台风轨迹,这样的方法不仅耗时耗力,而且往往事倍功半,预测的准确度和实时性都难以跟上。从上世纪末到本世纪初,随着机器学习的快速发展,优秀的机器学习算法诸如支持向量机<sup>[23]</sup> (Support Vector Machine, SVM)、AdaBoost<sup>[24]</sup>、随机森林算法<sup>[25]</sup> (Random Forest, RF) 等百花齐放,许多数据挖掘领域的学术竞赛和科研成果开始涌现,人们开始将人工智能应用于日常生活的方方面面。这一时期,许多研究人员也开始尝试使用机器学习的方法进行台风轨迹预测。其中以基于 RF 的人机结合台风预测系统和基于 GRAPS 的数值预报系统<sup>[26]</sup>较为突出。虽然相较于早期根据经验法则进行的台风轨迹预测,已经有了较大的进步,但是受限于特征选取难度大以及分类器可移植差,预测结果的精度提升十分有限,同时也很大程度上需要人工的参与,很难做到完全的自动化操作。

近年来,在一些数据结构非常复杂的领域,深度学习有了广泛的应用,有越来越多的研究人员开始尝试用深度学习的方法来提升台风轨迹的预测精度。相比较于经典的机器学习算法,深度学习不需要每进行一个新课题,都请领域专家来手动设计特征,而是可以通过算法从数据集中自动学习出特征。在数据量爆发式增长的今天,越来越多的研究人员通过先进的技术手段为采集、整理了很多高质量的数据集,同时伴随着 GPU 并行编程技术<sup>[27]</sup>的发展和成熟,研究人员可以用更加简洁、高效的手段进行深度学习的实验,缩短训练所需时长,这一切都为使用深度学习方法进行台风轨迹预测提供了可能可以说使用深度学习方法进行台风轨迹的预测,不仅仅能大大减少研究人员积累相关领域先验知识,提取特征,进行特征工程所需的时间,实现操作的完全自动化,更可以说是未来大数据时代提高台风轨迹预测精度的大势所趋。

目前,国内外研究人员运用深度学习的方法进行台风轨迹的预测也取得了一些不错的成果。在 2009 年, Rita Kovordanyi 等人首次使用深度学习中的神经网络<sup>[28]</sup> (Artificial Neural Network, ANN) 对台风的卫星图像数据进行预测,算是开启了深度学习进行台风轨迹数据预测的先河。而在 2014 年,随着 LSTM 网络在自然语言领域取得了突破, GAO Song 等人开始将 LSTM 网络运用于台风轨迹预测,首次将时间序列引入神经网络台风轨迹预测,较大程度上提高了预测的精度。2017 年,随着 CNN 网络在图像挑战赛 ImageNet 上披荆斩棘,一时间计算机视觉

领域的 paper, 创业公司如雨后春笋般拔地而起, CNN 网络开始被应用于台风轨迹的预测, 但是由于对于时间序列的预测表现并不好, 发展陷入停滞。而在 2018 年的 NeurIPS 上, 韩国的 Seongchan Kim 等人结合 LSTM 的时序性和 CNN 的图像识别能力, 将时空卷积网络<sup>[29]</sup> (ConvolutionalLSTMNetwork, ConvLSTM) 应用于台风轨迹预测, 取得了非常突出的效果, 标志着使用深度学习方法进行台风轨迹预测的发展进入黄金期。

### 1.3 本文的研究内容和结构安排

本文来源于当下热门研究课题, 本文通过深度学习的方法创新性的结合序列数据和图像数据进行台风轨迹预测, 以提升现下方法预测精度。基于注意力机制的 Seq2seq 模型在自然语言处理领域取得了巨大的成功, 相比较 LSTM 网络, 增加了编码器和解码器的过程, 一方面可以解决输入输出不等长的问题, 另一方面通过注意力机制能更好的关注前后状态之间的依赖关系, 提高台风轨迹预测精度。而 GAN 技术被广泛用于图像的生成, 主要是由判别网络和生成网络共同构成的模型, 生成网络负责从无序的噪声种生成图片, 判别网络负责判断生成图片与真实图片的差距, 二者形成一个博弈过程, 最终达到纳什均衡。通过 GAN 技术, 生成台风轨迹热力图。通过结合基于注意力机制的 Seq2seq 对时序数据预测的结果和 GAN 技术生成的热力图上的结果, 形成最终对台风轨迹的预测模型, 提升预测的精度。

本文的研究内容主要包括:

(1) 在深入了解相关深度学习理论和现有台风轨迹预测方法的基础上, 选取基于注意力机制的 Seq2seq 模型对台风轨迹数据进行预测, 同时运用 GAN 技术生成台风轨迹热力图, 辅以提高预测精度。

(2) 通过分析 LSTM 网络在处理不定长的时间序列数据方面的优势, 进一步研究 Seq2seq 模型的编码器和解码器机制, 同时深入分析注意力机制的优势, 并采用该方法进行台风轨迹预测, 提升精度。

(3) 通过分析 GAN 及 GAN 家族的各种变体在图像生成方面的杰出贡献, 探究生成网络和判别网络的原理和理论基础。根据历史的台风轨迹热力图, 运用 GAN 生成当前的台风轨迹热力图, 并在计算出在图上的轨迹坐标, 形成台风轨迹的预测结果。

(4) 采用多模型融合的方法, 通过基于注意力机制的 Seq2seq 模型进行台风轨迹预测, 并用 GAN 生成的热力图对预测结果进行修正, 提升预测精度。将实验成果与已有的研究成果进行对比, 检验该模型是否在已有成果的基础上有所提升。

## 1.4 论文结构安排

本篇论文共由五个章节组成，各个章节的安排如下：

第一章为绪论部分，主要介绍本文的研究背景与研究意义，主要针对使用传统机器学习算法进行台风轨迹预测特征提取难、预测精度差的问题，分析使用深度学习方法进行台风轨迹预测的意义，并对当下国内外研究现状进行阐述。

第二章为理论基础及相关技术的论述，介绍了相关方法的可行性，介绍了基于注意力机制的 Seq2seq 模型，以及用于图像生成的 GAN 模型的理论基础，最后介绍 Pytorch 框架，整个实验是基于 Pytorch 框架进行开发的。

第三章详细阐述了如何基于序列数据进行台风轨迹预测。首先会介绍如何对台风轨迹数据进行清洗，接着介绍了使用基于注意力机制的 Seq2seq 建模，进行台风轨迹预测，最后进行相关实验，并对实验结果加以说明。

第四章讲述了如何基于图像数据进行台风轨迹预测。首先会对图像数据进行预处理，接着详细介绍如何基于时序 GAN 模型建模，进行台风轨迹预测；然后将第三章的预测结果和本章预测结果进行模型融合，进一步提升预测精度。最后通过实验，证明方法的可行性，并对结果进行解释。

第五章为全文工作总结与后续研究内容展望，反思了本文工作的优点和不足，进一步指出了后续有价值的研究工作和方向。

## 第二章 理论基础及相关技术

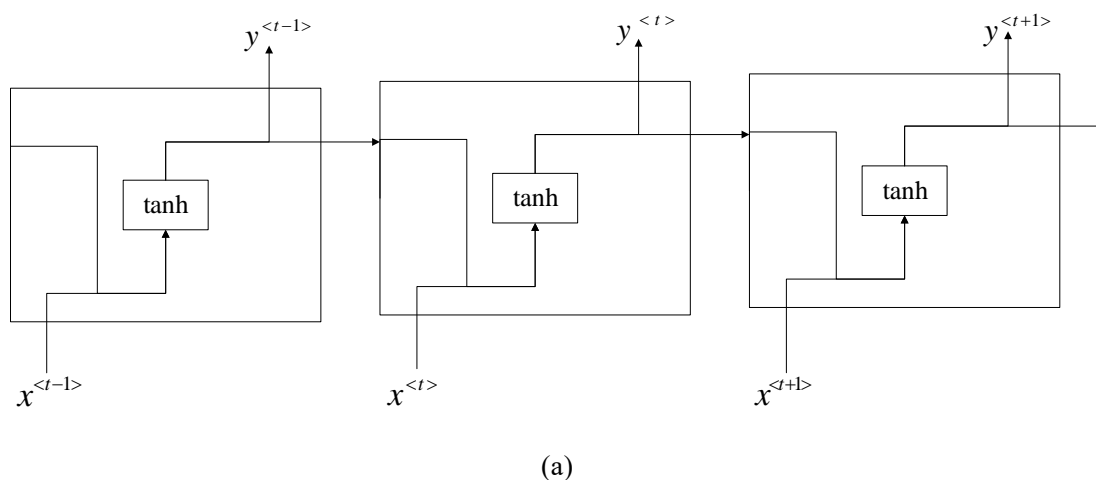
本章主要介绍本文涉及到的理论基础和相关技术。首先介绍 RNN 模型及其演化过程，主要介绍从 RNN 模型演化到 LSTM 模型，再演化到 Seq2seq 的过程；接着介绍 GAN 模型及其演化过程，主要介绍从基础 GAN 演化到 DCGAN 的过程；然后介绍本次实验所用到的深度学习框架 Pytorch，并与其他基于静态图的深度学习框架进行对比，突出动态图机制在深度学习方面的优势；最后对本章主要内容进行回顾。

### 2.1 RNN 模型及其演化

这一部分主要介绍 Seq2seq 模型的演化历程及其理论基础。

#### 2.1.1 RNN 模型

近年来，深度学习技术被应用于日常生活的方方面面，是实现人工智能技术落地的重要手段之一。其中以前馈神经网络（Feedforward Neural Network, FNN）贡献最为突出，然而随着人们对深度学习研究的深入，渐渐发现有些场景下 FNN 并不能很好的满足需求。比如在做机器翻译的时候，很多词汇在场景 A 下和场景 B 下所代表的意思并不完全相同，这时候需要联系上下文的内容才能对此较好的对词汇进行准确的翻译。而 FNN 的性质和网络结构决定了它并不适用，此时就需要研究一种新的网络结构来解决此类问题。而 RNN 就能比较好的满足需求，不同于 FNN, RNN 网络具有“记忆效应”，在计算当前状态时，能很好的依赖前序状态，因此非常适合解决时间序列相关的问题。RNN 的结构如图 2-1 所示：





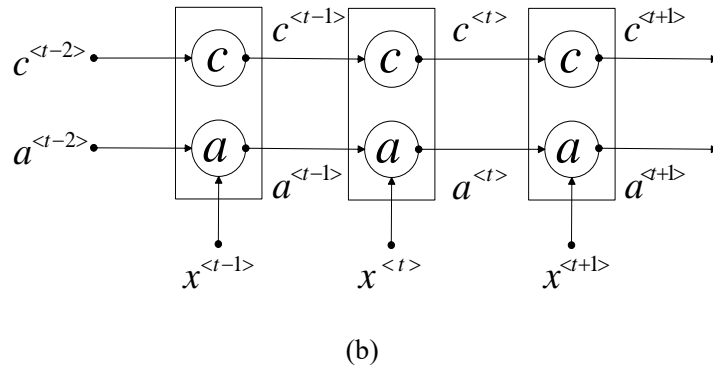


图 2-1 RNN 模型图

上图为 RNN 模型图，其中(a)为整体结构图，(b)为展开图。RNN 由输入层、隐藏层和输出层组成。不同于 FNN，在 RNN 中每次隐藏层产生输出的时候，这个输出会被存储到一个“记忆”里面去，当有新的输入到来时，新的状态不仅要考虑当前输入的值，还需要考虑上一个阶段存储于“记忆”里面的值，二者共同作用产生新的状态。与 FNN 相比，RNN 的具有更好的“记忆性”，能结合之前的状态做出决策。

设  $x$  代表输入层的值， $s$  代表隐藏层的值， $U$  代表输入层到隐藏层的权重矩阵， $o$  代表输出层的值， $V$  是隐藏层到输出层的权重矩阵，循环神经网络的隐藏层的值  $s$  不仅仅取决于当前这次的输入  $x$ ，还取决于上一次隐藏层的值  $s$ 。权重矩阵  $W$  就是隐藏层上一次的值作为这一次的输入的权重。

在当前时刻  $t$ ，输出层  $o$  的计算公式如式 (2-1) 所示：

$$O_t = g(V \cdot S_t) \quad (2-1)$$

在当前时刻  $t$ ，隐藏层  $s$  的计算公式如式 (2-2) 所示：

$$S_t = f(U \cdot X_t + W \cdot S_{t-1}) \quad (2-2)$$

反向传播算法顾名思义是从后往前传播的，从最后开始，经过每个隐藏层的输出、权重和输入倒着移动，将误差按照一定的比例分配给每个权重。RNN 为反向传播算法引入了时序性，是一种时序的反向传播算法，称为 BPTT<sup>[30]</sup>。BPTT 是一种特殊的 BP 算法，从前文分析中可以看出 RNN 是一种与时间序列相关的网络，所以反向传播也会带有时序性，故称为基于时序的反向传播算法。BPTT 算法的基本思想还是 BP 算法那一套，通过优化参数的负梯度方向，不断寻找最优解。在反向传播过程中，有  $V$ 、 $W$ 、 $U$  三个参数需要优化，需要的公式如下：

在当前时刻  $t$ ，对  $V$  求偏导的计算公式如式 (2-3) 所示：

$$\frac{\partial L}{\partial V} = \sum_{t=1}^n \frac{\partial L^{(t)}}{\partial o^{(t)}} \cdot \frac{\partial o^{(t)}}{\partial V} \quad (2-3)$$

在当前时刻  $t$ ，对  $W$  求偏导的计算公式如式 (2-4) 所示：

$$\frac{\partial L^{(t)}}{\partial W} = \sum_{k=0}^t \frac{\partial L^{(t)}}{\partial o^{(t)}} \frac{\partial o^{(t)}}{\partial h^{(t)}} \left( \prod_{j=k+1}^t \frac{\partial h^{(j)}}{\partial h^{(j-1)}} \right) \frac{\partial h^{(k)}}{\partial W} \quad (2-4)$$

在当前时刻  $t$ ，对  $U$  求偏导的计算公式如式 (2-5) 所示：

$$\frac{\partial L^{(t)}}{\partial U} = \sum_{k=0}^t \frac{\partial L^{(t)}}{\partial o^{(t)}} \frac{\partial o^{(t)}}{\partial h^{(t)}} \left( \prod_{j=k+1}^t \frac{\partial h^{(j)}}{\partial h^{(j-1)}} \right) \frac{\partial h^{(k)}}{\partial U} \quad (2-5)$$

从上面的求导公式可以看出，累乘会导致激活函数导数的累乘，进而会导致“梯度消失”和“梯度爆炸”的现象。特别是如果选取 **sigmoid** 作为激活函数的话，由于 **sigmoid** 函数把区间映射到  $[0,1]$ ，则问题转化为很多  $[0,1]$  之间数的连乘。由于 RNN 求导是累乘，大量  $[0,1]$  之间的数累乘结果会无限趋近于 0。如前文所述，因为 RNN 天生的缺陷，无论如何选择激活函数，都不可避免会地在反向传播过程中出现“梯度消失”和“梯度爆炸”，基于此，需要对 RNN 网络做一些改良。

### 2.1.2 LSTM 模型

如前文所述，在理论上，RNN 的网络结构非常适用于时间序列相关的问题，但是在实际操作中，RNN 在使用 BPTT 算法进行训练的时，由于累乘会导致激活函数的导数出现“梯度消失”和“梯度爆炸”的问题，因此需要 RNN 此结构做出一些改变来有效缓解此类问题。基于此，深度学习的研究人员对 **Simple RNN** 模型做出一些改进，提出了 **LSTM** 模型。**LSTM** 的模型图如图 2-2 所示：

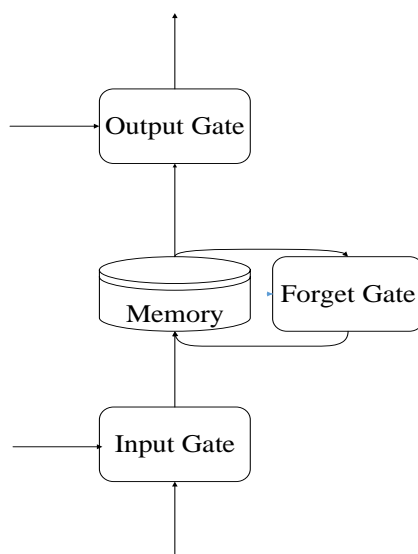


图 2-2 LSTM 模型

如果可以通过某种方法，控制前序状态中信息的流过量，使得每次只留下有用的信息，过滤那些无用的。就能有效缓解前序状态的长期依赖，从而较好的利用先前的状态，而 LSTM 就是这样的一种网络结构。在 RNN 中的“记忆”是比较简单的，可以方便随心所欲把信息存进去和取出来。而 LSTM 网络中，存在三个门来控制信息的流入和流出。当信息需要被写入“记忆”时，必须要经过输入门，只有当输入门打开时，才允许把信息写入“记忆”。与之对应的，在信息输出的地方就有一个输出门，控制“记忆”里面信息的流出，只有当输出门打开时，外界才可以把“记忆”里面的值读出来。而在“记忆”上存在一个遗忘门，这个门的作用是控制对过去存在 memory 里面的值记忆与否的，只有当遗忘门打开时，代表对过去的状态有记忆。LSTM 网络通过这三个门来控制信息对于过去信息的流过，而这三个门的开关状态是由神经网络自己学习而得到，因此说对比 RNN 模型，LSTM 网络的依赖不再是从头到尾的累乘，而是通过学习决定前序状态的依赖，这样就能较好地过滤掉一些无用状态，从而有效缓解“梯度消失”和“梯度爆炸”。下面将展示 LSTM 的内部架构图，如图 2-3 所示：

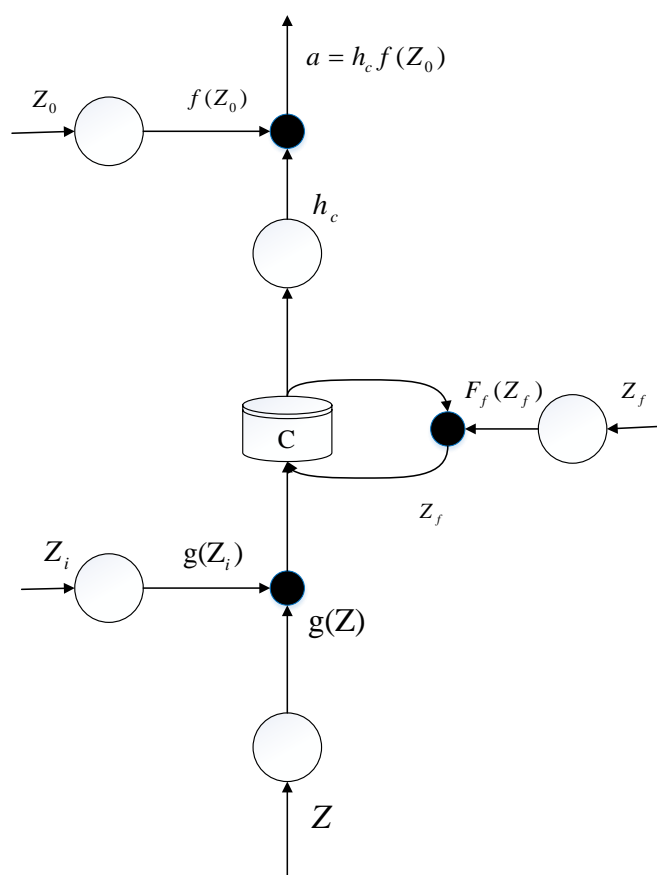


图 2-3 LSTM 内部结构图

不同于 RNN 中选择  $\tanh$  作为激活函数，LSTM 中的三个门选择  $\text{sigmoid}$  作为激活函数。之所以选择  $\text{sigmoid}$  函数，是因为它的值是介于 0 到 1 之间，这个 0 到 1 之间的值表示每一个门的开关状态。可以看出 LSTM 是 RNN 的一个优秀的演变，很好的缓解了 RNN 中“梯度消失”和“梯度爆炸”的问题。但随之而来的也引入了更多的参数，LSTM 具有 4 个输入和一个输出，这也就是说 LSTM 的参数量约为普通神经网络的 4 倍，这会增加训练的复杂度。同时观察 LSTM 的结构并不是万能，因为 LSTM 的结构中要求输入和输出是等长的，如果其中一个是变长的，则 LSTM 并不适用，基于此，需要对 LSTM 网络模型做进一步的扩展。

### 2.1.3 Seq2seq 模型

在机器翻译，语音识别，时间序列预测等问题的时候常常会遇到这样一种场景。当输入和输出都是序列时，也就是人们常说的多对多的模型时，LSTM 就不再适用。通过分析 LSTM 网络的输入和输出结构可以发现，如果输出的序列长度比输入短时，可以使用 LSTM 进行预测。但当输出的长度比输入长时，LSTM 模型就不再适用，此时需要对 LSTM 模型做进一步的改良，于是就有了 Seq2seq 模型。其内部结构如图 2-4 所示：

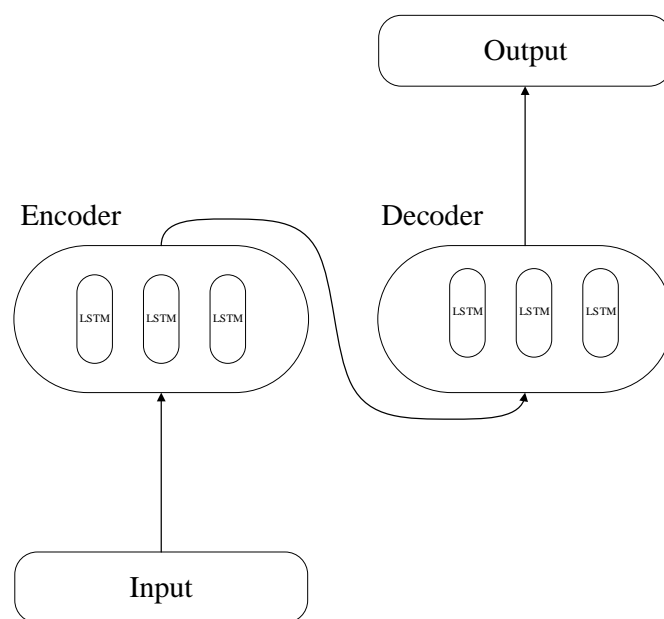


图 2-4 Seq2seq 模型

Seq2seq 模型的核心就是编码器和解码器。编码器编码的过程是将输入数据通过一系列非线性变换映射成一个定长的向量，被称为的中间向量，作为中间转换的桥梁。而解码器的过程就是根据编码器输出的结构和之前生成的历史信息，来输出

最终的结果。编码器和解码器模型，对任何端到端的深度学习任务都是适用的。随着深度学习技术的发展，人们感觉到单模型进行深度学习不再能满足日常的需求，于是人们开始研究 Multimodal Learning<sup>[31]</sup>。起初人们考虑把 CNN 网络和 LSTM 网络结合起来做看图说话，后来人们把 LSTM 网络和 LSTM 网络结合起来做机器翻译于是就诞生了 Seq2seq 模型。最初 Seq2seq 模型被应用于自然语言处理，现在渐渐被扩展到语音识别、时间序列分析等领域。

但是如前文所述，编码器和解码器模型虽然在众多领域取得了重大的成功，也不可避免的存在一些缺陷。首先解码器的输入全部来自于编码器产生的中间向量，可以说整个模型的效果的瓶颈很大程度由中间向量决定，如果前面编码器过程训练的效果不佳，那整个模型都会受到很大的影响。还有就是 RNN 家族都会存在的通病，“梯度消失”和“梯度爆炸”。因为编码器和解码器是由很多 LSTM 网络的组成的，如果关注编码器的所有过程，生成中间向量再去训练对应的解码器，就会产生更长的依赖，给神经网络训练造成了很大的难度。基于此需要对编码器和解码器模型做出一些改进，于是将注意力机制引入了 Seq2seq 模型。其架构图如图 2-5 所示：

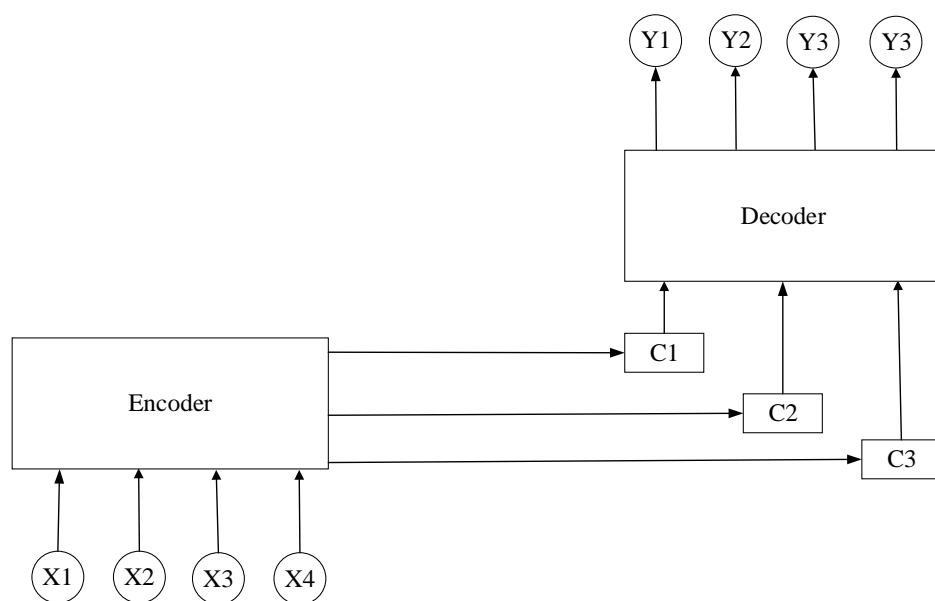


图 2-5 注意力机制

对于许多时间序列相关的问题，很多时候并不需要依赖所有的前序状态，而只关注其中的某一关键的部分，剔除掉一些无关紧要部分。如果的解码器的数据不仅仅只依赖于编码器生成的中间向量，而是与编码器中每个前序状态做一次点乘，然后过一个 Normalization，然后计算出每一个的概率，将概率和隐藏层状态相乘累加得到加权平均。如此一来，可以学习出编码器和解码器间权重的关系，从更好的分配注意力权重，与编码器输出状态关联性比较大的输入状态，会分配比较大的权重。

一方面，注意力机制的引入，打破了只能利用编码器生成的中间向量单一向量结果的限制，从而使模型可以集中关注下一个状态关联性比较紧密的输入信息上，提升模型对于时序问题的表现。另一方面，注意力机制可以更好的可视化权重向量的变化情况，方便学者查看当前状态是跟之前哪些时序状态有关联，有助于更好的理解模型工作机制，提升深度学习的可解释性。

## 2.2 GAN 模型及其演化

这一部分主要介绍 GAN 及 GAN 家族的理论基础。

### 2.2.1 基础 GAN 模型

GAN 最早在 2014 年由 Ian Goodfellow 的论文《Generative Adversarial Networks》提出，是近几年来深度学习的热门方向。GAN 的提出打开了深度学习的新世界，被誉为近年来最耀眼的新思想之一，GAN 的提出不仅催生了很多优质的深度学习论文，更带来了许多层出不穷的实际应用。生成模型在深度学习领域具有非常广泛的应用，不仅可以用于测试模型的高维概率分布的表达能力，而且可以用于多模输出问题，以及最常见的产生“真实”数据的问题。GAN 的出现提供了一种全新的生成模型的方式。GAN 模型如图 2-6 所示：

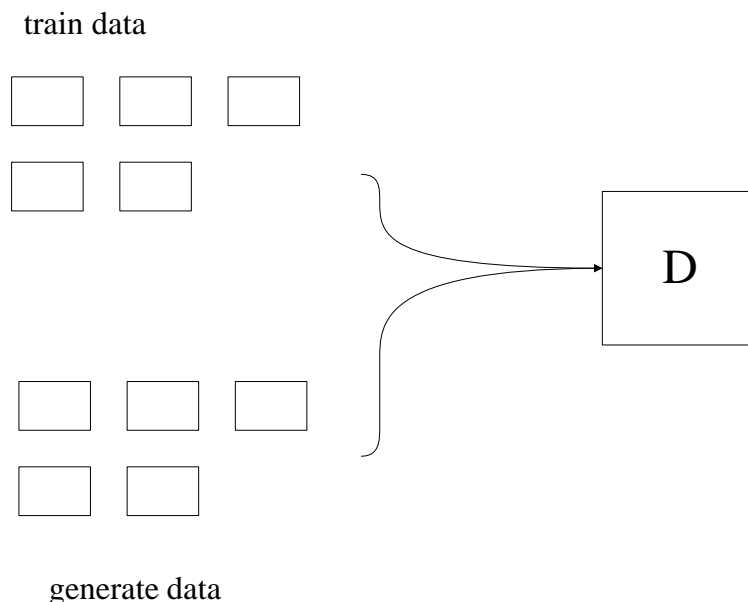


图 2-6 GAN 模型

GAN 的原理不同于传统的深度学习模型，GAN 由生成器和判别器两个网络构成。生成器主要用于通过随机噪声生成图片，而判别器是一个判别网络，判别一张

图片是不是“真实的”。在训练过程中，生成器的目标就是尽量生成真实的图片去欺骗判别网络判别器。而判别器的目标“挑刺”，尽量区分该图片是生成器生成的还是真实的。生成器网络和判别器网络构成了对立，直到最终生成的网络能完全欺骗判别的网络。在 GAN 以前，主流的生成式深度学习模型为 VAE<sup>[32]</sup>，其目的是希望生成的物体和输入越像越好，此时需要判断生成物体与原物体的相似度。深度学习中 model 的相似度都是通过 loss 函数来进行衡量，而 VAE 中采用的是 MSE<sup>[33]</sup> 函数。但是 loss 小并不一定真的代表二者相似。就拿图像生成来说，如果已知当前时刻的图像，预测下一时刻的图像，传统的生成模型是预测下一状态的所有可能进行然后取均值，这样很可能生成一个“四不像”的东西，而 GAN 的发展为这一切带来了可能。GAN 的博弈过程公式如式（2-6）所示：

$$\min_G \max_D (D, G) = E_{x \sim p_{data}(x)} [\log D(x) + E_{z \sim p_z(z)} \log(1 - D(z))] \quad (2-6)$$

从上式中可以看出，理论上说如果生成器和判别器复杂度足够，那么 GAN 就能生成数据的分布。生成器的目标就是通过不断优化使得生成的数据尽可能接近真实数据。从理论上讲，生成器生成函数的概率分布，要求生成的对象与真实的有相同的概率密度函数。真正在指导生成器进行训练的是隐式定义的分布和数据集真实分布的 JS 散度。当两个分布完全重合时，生成器的目标函数值达到最小  $-\lg 4$ ，且此时的判别器也为最优。理论上说，精度足够，模型一定能收敛到最优解。而判别器的功能类似于逻辑回归，对于一个样本  $x$ ，判别器  $D(x)$  将给出该样本来源于训练集的概率，所以一个“完美”的判别器应该对来自于训练集的样本输出概率 1，对来自于生成器的生成样本输出概率 0；对于生成器  $G$ ，它试图捕捉到训练集的本质模式，生成样本并将样本送至判别器  $D$ ，最好能使该样本“欺骗”判别器，使判别器误认为该样本来源于训练集而输出概率 1。GAN 是一种如此神奇的模型，同时如论文中提到那样 GAN 也可以使用像普通深度学习模型那样使用梯度下降算法进行训练

第一步，需要训练判别器  $D$ ，学习  $D$  的过程是计算 JS 散度的过程，需要最大化价值函数，希望  $V(G, D)$  越大越好，所以此时需要加上梯度，如式（2-7）所示：

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m [\log D(x^{(i)}) + \log(1 - D(G(Z^{(i)})))] \quad (2-7)$$

第二步，训练生成器  $G$ ， $G$  的学习过程恰恰相反，是希望  $V(G, D)$  越小越好，所以是减去梯度，如式（2-8）所示：

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log(1 - D(G(Z^{(i)}))) \quad (2-8)$$

整个过程交替进行，如此一来可以使用梯度下降算法像普通神经网络那样对 GAN 进行训练。GAN 从本质上来说和 CNN 等神经网络是不一样的，GAN 的训练过程需要依次迭代 G 和 D 网络，这样一类就会导致一些问题。如果判别器 D 网络学习效果欠佳，则在下一步中生成器 G 网络就得不到正确的反馈，就无法稳定的学习。而如果判别式 D 的学习效果过好，则 loss 下降就会非常快，G 网络就无法进行学习。因此对于 GAN 的训练来说，需要生成器和判别器达到纳什均衡，但是因为判别器和生成器是分开训练的，纳什平衡很能达到。这就导致了早期 GAN 非常难以训练。基于此，需要对 GAN 做一些改进，于是就衍生出了 GAN 的家族。

### 2.2.2 DCGAN 模型

如前文所示，GAN 作为一种生成式模型在图像领域取得了前所未有的成功，但是由于训练困难，需要对 GAN 做一些优化。由此会衍生出 GAN 的家族。其中 DCGAN<sup>[34]</sup>发展最为突出，其模型如图 2-7 所示：

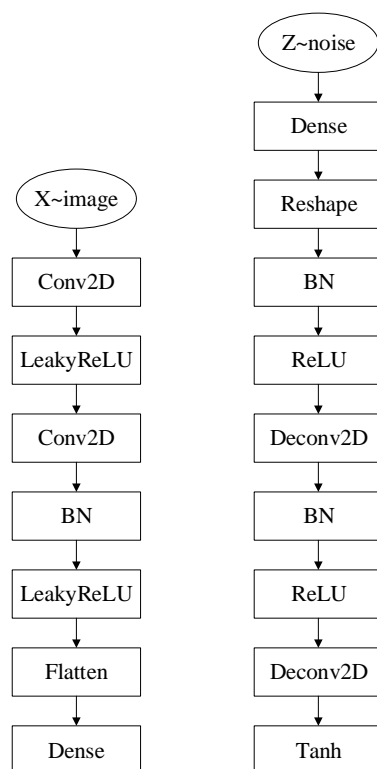


图 2-7 DCGAN 模型

2015 年，Alec Radford 等人发明了 DCGAN 模型，提出了一种特殊的生成器和判别器架构，能极大程度上稳定 GAN 的训练。DCGAN 可以看成将 GAN 扩展到 CNN 领域，其主要改进有以下几点：

- (1) 生成器和判别器均不采用 pooling，而采用带步长的卷积层；其中判别



器采用普通卷积 (Conv2D), 而生成器采用反卷积 (DeConv2D)

- (2) 生成器和判别器上均使用 Batch Normalization, 一方面可以使得梯度传播到每一层, 防止生成器把所有样本都收敛到同一点, 另一方面可以有效缓解样本的震荡和模型的不稳定
- (3) 移除了全连接层, 增加了模型的稳定性
- (4) 在生成器除输出层外的所有层上使用 ReLU 激活函数, 而输出层使用 Tanh 激活函数
- (5) 判别器的所有层上使用 LeakyReLU 激活函数

而后随着 ResNet 网络在的 ImageNet 挑战赛上取得举世瞩目的成绩, 人们开始考虑将 ResNet 网络引入 DCGAN 模型, 在算力足够的情况下, 就可以放心大胆地把网络做得更深。因为 ResNet 的机制可以保证在模型层数更多时, 效果不会比浅层模型更差, 利用卷积网络对于边缘特征强大的提取和识别能力进一步提高生成网络的学习效果。DCGAN 的提出标志着 GAN 的研究开始进行高峰期, 越来越多的产品都是基于 DCGAN 或者 DCGAN 的变种衍生出来的。

## 2.3 Pytorch 框架

任何一项技术的快速发展都不可避免的要向普及性和易用性发展, 深度学习技术也是如此。得益于计算机算力的高速发展以及 GPU 并行化的技术的推广, 使得利用深度学习训练更深更大的模型变成了可能。为了能将深度学习技术运用于更多领域许许多多的研究人员为大家开发了很多好用的深度学习框架, 使得深度学习的实验变得简单而高效。以 Caffe 为主的第一代深度学习框架秉承 a 序列 of layers 的理念, 但是因为 layer 的粒度太粗, 缺少计算图的概念, 随着深度学习的发展, 对很多复杂的模型很难保证框架的易用性。然后以 Tensorflow、MxNet、Caffe2 为主的第二代深度学习框架被提出, 这代框架在上一代的基础上做了较多的改良, 以 a graph of operators 为理念, 引入了静态图机制。其特点是需要优先把深度神经网络经过拓扑构造成一个 DAG。相比较于上一代模型已经有了很大的改进, 但是由于对于中间过程无法像 python 一样进行调试, 为调参带来了很大的工作量。基于此, 人们又引入了第三代深度学习框架, 以 Pytorch 为代表的深度学习框架, 引入了 a dynamic graph of operators, 使得变成人员能够像操纵 python 语言那样去操纵神经网络框架, 为开发和调参带来了极大的便捷, 正是由于第三代深度学习框架的开发, 使得深度学习技术得到了极大的普及, 而 Pytorch 在学术界和工业界也受到了更大的重视, 近年来, 更有赶超 Tensorflow 的趋势。

Pytorch 作为现今增长最快的深度学习框架, 主要具有以下优势:

### （1）简洁

Pytorch 框架的设计条理清晰，从上到下由 `tensor`、`autograd`、`nn.Module` 三层构成，层与层之间相互作用，可以同时进行修改和操作。而从源码的可读性来说，Pytorch 的源码不到 Tensorflow 的十分之一，非常方便开发人员阅读源码，并作进一步的扩展。

### （2）速度

Pytorch 以提倡向 python 那样面向对象编程，这为模型的搭建提供更大的灵活性，但却又不以牺牲运行速度为代价，在多项性能评测中，Pytorch 运行效率并不比 Tensorflow 差。另一方面，Pytorch 便捷简单的接口，以及和 python 的无缝对接，为开发人员带来了良好的编程体验。

### （3）活跃的社区

Pytorch 提供完整的接口文档与使用教程。相比于 Tensorflow 繁杂的文档，Pytorch 的开发文档更加简洁，同时 Facebook 研究院的大力支持，确保了 Pytorch 持续的开发与优化。Pytorch 中的核心机制如图 2-8 所示：

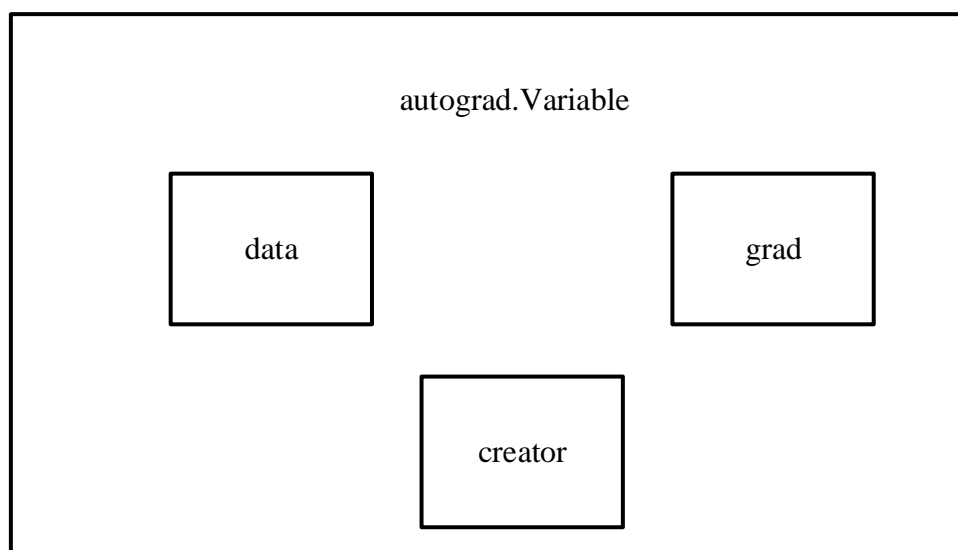


图 2-8 Pytorch 中的 autograd 图

Pytorch 的核心就是动态图机制，支持动态建立神经网络。不同于 Tensorflow 等静态图框架，用户在构建模型时不必一开始就定义整个神经网络，然后再触发 `run` 操作输出结果，重复进行迭代和训练，而是可以实时观测到训练的中间过程。Autograd 是动态图机制的核心。Autograd 由 `data`，`grad`，`creator` 三部分组成，共同维护一张无环图的张量并同时执行所有运算，输入向量由图的叶子结点进行维护，输出向量由图的根节点维护。在有向无环图中遵循链式求导法则计算梯度。

## 2.4 本章小结

本章主要介绍了基于深度学习的台风轨迹预测所用到的算法理论基础和相关技术。首先介绍了从 RNN 进化到 LSTM 模型再进一步进化到的 Seq2seq 模型的演化历程及其基础知识。然后介绍了 GAN 的基本模型和近年来在各领域取得成功的 DCGAN 模型。最后介绍了本次实验所用到的深度学习框架 Pytorch 以及其优异的动态图机制。本章内容即是对前一章节内容的扩充,同时也为后续章节内容提供了良好的理论支撑。

## 第三章 基于序列数据的台风轨迹预测

本章主要通过对序列数据建模进行台风轨迹预测。首先,针对卡尔曼算法误差受观测值影响大的缺陷,提出改进的卡尔曼滤波算法对轨迹数据进行清洗,剔除野值,提高数据质量;接着使用最小扇形简化算法简化台风轨迹,提升训练速度。然后将基于注意力机制的 Seq2seq 模型应用于台风轨迹预测,提升现有方法的预测精度。最后对研究内容进行实验,并对实验结果进行说明,证明方法的可行性。

### 3.1 数据预处理

台风轨迹数据大多是由卫星进行采集,且是由点构成的时间序列,由于受到采样频率和数据质量等因素的限制,轨迹采样得到的数据质量也不一,点数据包含的信息可能存在误差、错误和缺失。因此,在实际应用中,有必要对台风轨迹数据进行预处理。

#### 3.1.1 轨迹数据滤波

由于台风轨迹主要是由卫星采集到的,采集过程中可能会存在一些严重偏离实际状态的数据。如果观测值和真实值的偏差超过了该观测源最大的测量误差,那么该观测值就是野值。在实际情况中,这些野值甚至是随机分布的,与被观测目标的台风轨迹没有任何关联。野值的存在会使得台风轨迹的特征发生重大变化,对台风轨迹预测结果精度造成严重的影响。因此,在预处理中准确的检测并剔除这些野值是有必要的,基于此,需要对台风轨迹数据集进行滤波,剔除野值点,提高训练数据质量。

主流的轨迹滤波算法主要有中值、均值滤波算法和卡尔曼滤波算法,分别具有以下特点:

(1) 中值、均值滤波算法<sup>[35]</sup>:

依次检查输入信号的每一个值,用它和它的邻居的中值或均值来代替它自己。因此该算法是对观测窗口进行简单高效的算术运算,因此可能会存在一定的延迟。同时中值滤波算法可能会损失较多信息,均值滤波算法会被严重偏离正常值的数据影响,因此中值滤波算法的效果非常依赖于调参,且参数不具有普适性,对于不同数据需要不断调参。

(2) 卡尔曼滤波算法<sup>[36]</sup>

卡尔曼滤波算法是一种通过递归方法实现的高效滤波算法,主要是为了解决

从含有噪声的测量数据中，实时预估系统的状态。不同于一般滤波算法，卡尔曼滤波除了使用当前时刻的观测值，还对移动目标建立模型，对其位置进行预测，而且结合观测值和预测值得到目标的位置的最优估计。因此该算法不但具有较低的时间复杂度和空间复杂度而且能从存在噪声的观测数据对轨迹数据作出最优估计，得到更加准确的结果。

中值滤波和均值滤波简单高效，但是中值滤波可能会损失较多信息，均值滤波会被严重偏离正常值的数据影响。卡尔曼滤波通过建立模型，对移动目标的位置进行预测，并结合存在噪音的观测值，得出接近目标真实位置的最优估计。根据上面的分析，卡尔曼算法更适合对台风轨迹数据进行滤波，并对基础的卡尔曼算法做出改进，下面将重点进行介绍。

卡尔曼滤波器的滤波分为预测和更新两个阶段。预测时，需要基于上一时刻的估计值，同时结合当前时刻的观测值，进行估计。更新时，需对观测值进行优化，用当前时刻的观测值修正预测阶段的结果，获得更加精确的结果。在卡尔曼滤波中存在两个表示状态的变量， $X(k/k-1)$  是在时刻  $k$  的系  $k$  统状态的估计， $P(k/k-1)$  是估计的误差协方差矩阵，代表估计值的精确程度。

设  $F(k)$  是状态转移矩阵  $B(k)$  和  $u(k)$  为控制矩阵和控制向量， $Q(k)$  代表了外部不确定因素的影响，于是预测阶段协方差矩阵如式 (3-1) 所示：

$$\begin{aligned} X_{k|k-1} &= F_k X_{k-1|k-1} + B_k U_k \\ P_{k|k-1} &= F_k P_{k-1|k-1} + Q_k \end{aligned} \quad (3-1)$$

同时在更新阶段假设  $z(k)$  是观测向量， $R(k)$  是测量误差矩阵，代表测量误差，则更新阶段的状态矩阵如式 (3-2) 所示：

$$\begin{aligned} K' &= P_k H_k^T (H_k P_k H_k^T + P_k)^{-1} \\ P'_k &= P_k - K' H_k P_k \\ X_k &= X_k + K' (\bar{Z}_k - H_k \hat{X}_k) \end{aligned} \quad (3-2)$$

由卡尔曼滤波方法可知，若预测值  $x(k/k-1)$  准确，那么预测值和观测值的误差为观测误差。但是在现实中观测值经常会出现异常，此时观测误差会变得很大。出现这种情况不是预测值不准确导致的，而是观测值发生异常导致的，也是滤波结果误差大甚至说结果难以收敛的主要原因。需要对卡尔曼滤波进行改进从而达到剔除野值的目的。卡尔曼滤波根据轨迹的历史位置信息对当前时刻的位置进行预测，如果当前的观测值是野值，那么观测值会和预测结果存在较大的差值，如果该差值超过了测量误差和预测误差的总和，那么就有理由认为这一观测值是野值，将其剔除，并直接使用预测值作为该目标当前时刻位置的最优估计，否则认为

该测量值正常，予以保留。基于此，可以对卡尔曼算法进行改进，在进行更新步骤之前，判断观测值是否严重偏离预测值，若是，则视为野值，进行剔除，并用预测值代替。改进卡尔曼算法对台风轨迹滤波的流程如图 3-1 所示：

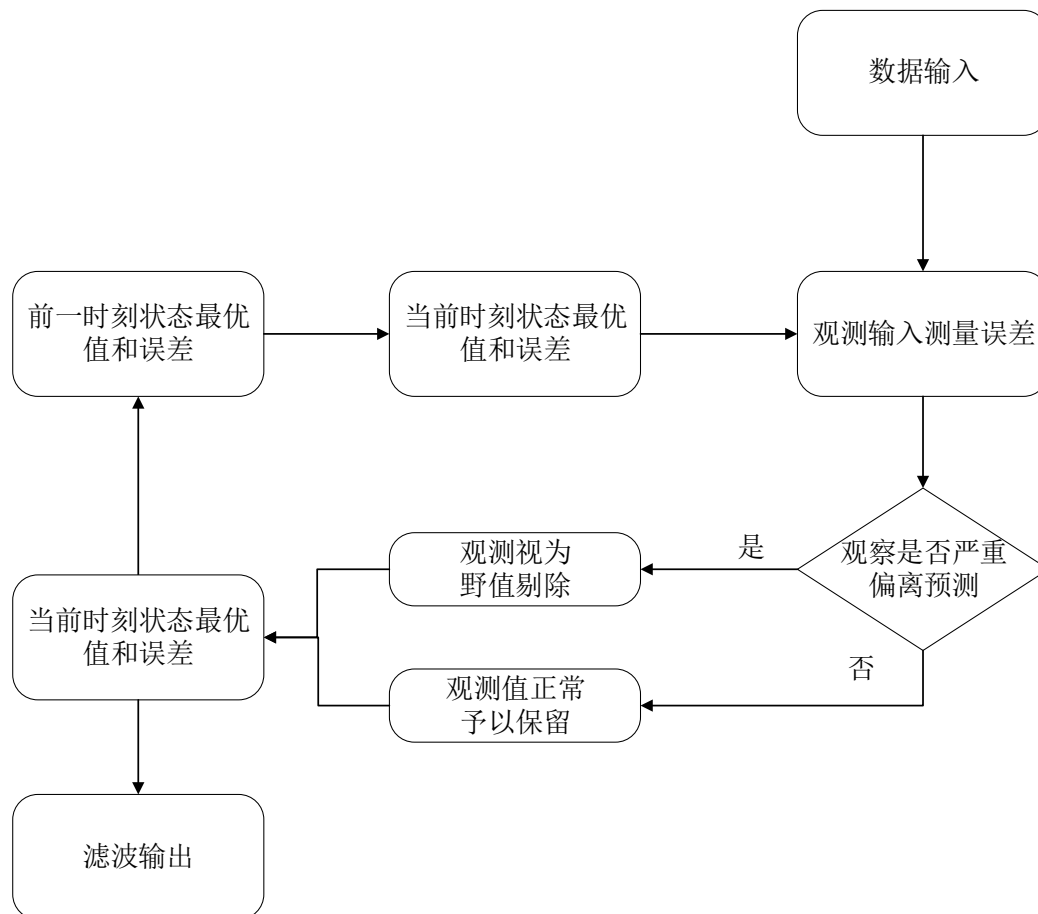


图 3-1 改进卡尔曼滤波流程图

需要结合目标的历史状态和当前时刻的状态对目标进行预测。相比较传统的卡尔曼滤波算法，改进后的卡尔曼滤波算法可以更好地发现轨迹中的野值，并进行剔除，更加有助于提高台风轨迹训练数据集的质量。在线性系统种，噪声项和状态都服从高斯分布。如此一来，需要估计的只有协方差矩阵和均值，就可以表示状态，由于只对移动目标进行观测，不会对其进行控制，所以可以忽略控制向量和控制矩阵。同时，由实际数据可知，移动目标的位置通常是逐渐变化的，所以在较短的时间间隔内，认为移动目标做匀速直线运动。同时，数据集中影响台风的因素主要有：台风强度标记(I)，纬度(LAT)，经度(LON)，中心最低气压(PRES)，中心最大风速(WND)，基于此，可以得到改进后的卡尔曼滤波算法的伪代码如表 3-1 所示：

表 3-1 改进卡尔曼滤波算法

改进卡尔曼滤波算法
输入: points[M]轨迹点数据 初始化: Let's $X = \text{matrix}[I, \text{Lat}, \text{Lon}, \text{Pres}, \text{Wnd}]$ F: 状态 1 转移矩阵 R: 测量误差的协方差矩阵 for $i = 0 \dots M-1$ do: set $X_{k k-1} = F_k X_{k-1 k-1} + B_k u_k$ set $P_{k k-1} = F_k P_{k-1 k-1} F_k^T + Q_k$ 更新状态: set $y_k = z_k - H_k X_{k k-1}$ set $S_k = H_k P_{k k-1} H_k^T + R_k$ set $K_k = P_{k k-1} H_k^T S_k^{-1}$ 然后更新滤波器变量 X,P: Set $X_{k k} = X_{k k-1} + K_k y_k$ Set $P_{k k} = (I - K_k H_k) P_{k k-1}$ return X

相比较于朴素的卡尔曼算法,改进的卡尔曼算法出来对下一状态的值的预估之外,更加入了对于野值的剔除。朴素的卡尔曼算法对于滤波的过程非常依赖于调参,而改进的卡尔曼算法能减少调参的工作量,相比较朴素卡尔曼算法,具有更好的泛化能力。

### 3.1.2 轨迹数据简化

由于台风处于高速运动,卫星的采样一般以秒级响应 但是计算机的存储能力和计算能力是有限的,对于台风轨迹数据的研究并不需要秒级的响应,因此通常需要对轨迹数据进行压缩。目前比较常用的道格拉斯-普克算法<sup>[37]</sup>,主要是利用轨迹点到直线距离的大小决定是否保留轨迹上的点。该算法在处理轨迹数据点密集且距离较长的时候算法的效率较低,需要不断的迭代搜索,除此之外该算法在处理变化幅度较大的轨迹的时候,例如当轨迹中出现较多的转角,可能导致许多关键角度的信息丢失。为了解决这样的问题,受上面算法的启发创新性的提出基于最小边界扇形的轨迹简化算法。其流程如图 3-2 所示:

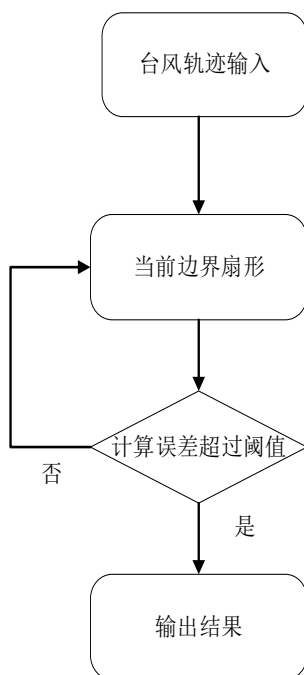


图 3-2 最小扇形简化算法流程图

基于最小边界扇形的轨迹简化算法,能够在可控的误差范围内高效的简化轨迹并且最大限度的保留原始轨迹特征信息。基本思想是用标准图形(扇形)逐段的匹配整个台风的轨迹。最小扇形的构造方法根据选择顶点的方式不同分为以下两种:

(1) 向后构造最小边界扇区,以当前时间的位置点为顶点。扇形顶点的选择随着位置信息的更新而连续变化。

(2) 简化并更新前一时刻的下一个位置点或感知定位开始时的初始位置点,以构造顶点的最小边界扇区。

基于上述分析,选择第一种方法来构造扇形,因为第一种方法性能更好,因此简化后轨迹信息保留不比第二种差。完成了最小边界扇形的构造以后,需要一定的选择策略来近似的简化扇形内的航迹点。选取扇形中顶角平分线上的点作为对应点,同时保证对应点和初始点离顶点具有相同的距离。边界扇形构造的过程需要同时控制误差,若超过当前时刻的误差允许的范围,就需要选择新的位置信息,然后以该点开始重新构造新的边界扇形。可将误差定义为图中的  $d$ 。易知,对于给点的一系列点,最小扇形是唯一确定的, $d$ 也是确定的,任意一个原始点与被简化后的点间的距离(误差)都不会超过  $d$ 。

$d$  的值与该边界扇形的中心角大小和扇形的半径有关,假设  $l$  为扇形的半径长度, $\theta$  为扇形的中心角,在等腰三角形中  $d$  的求法如式 (3-3) 所示:

$$d^2 = 2l^2 - 2l^2 \cos\left(\frac{\theta}{2}\right) \quad (3-3)$$



假设误差阈值为  $error$ ，则可以得出  $error$  的取值范围公式如式 (3-4) 所示：

$$2l^2 - 2l^2 \cos\left(\frac{\theta}{2}\right) \leq error^2 \quad (3-4)$$

在设定了误差可接受的范围后，如果当前时刻位置信息发生了改变，判断新产生的边界扇形是否满足式 3-4，如果不满足，就需要更新移动目标的位置并构造新的扇形。该算法能够有效的解决复杂轨迹的压缩问题，与传统的压缩算法不同的是，该算法可以很有效的解决传统算法无法有效压缩复杂轨迹，摒弃传统方法中基于距离的评判而使用基于角度的评判标准，并且不仅考虑位置信息的影响还将轨迹的时序考虑在内从而能够在局部信息中对轨迹进行局部压缩从而极大限度的保留了轨迹原始飞行特征。则算法的伪代码如表 3-2 所示：

表 3-2 最小扇形简化算法

最小扇形简化算法
输入: $points[M]$ 轨迹数据点 初始化: $error=10$ 设置 $d$ 的阈值 $segments[]$ 分段的轨迹 $start=0$ 起始位置 $end=0$ 结束位置 $d=-1$ 设置最小阈值 for $i = 0 \dots M-1$ do : 对当前起点和结束点更新最小扇形 if $end-start \neq 0$ : 有多个点 $d = update(start, end)$ 更新 $d$ if $d > error$ : $mid = getMid(start, end)$ 获取线段中点 $segments.append(start, mid)$ $start = end$ 重新设置起点 $end++$ return $segments$

从上表可以看出，简化后对于一条轨迹来说，只需要按照轨迹的时序从开始节点往后遍历，对于每个新的点来说不断的更新当前扇形范围，如果超过阈值则开始新一轮的更新。算法的时间复杂度为  $O(N)$ ，相比较于传统的算法来说，该算法时间

效率更高，能更好的应对轨迹形状复杂且轨迹点数目巨大的情况。

## 3.2 方案设计

在使用深度学习模型对台风轨迹预测之前，需要对台风轨迹数据进行预处理。在 3.1 节中，对台风轨迹数据进行了清洗，一方面，通过改进的卡尔曼滤波算法剔除了野值，提升了轨迹质量；另一方面，通过最小扇形简化算法简化了轨迹，提升了训练的效率。有了前面工作的铺垫，本节将会重点介绍如何用基于注意力机制的 Seq2seq 模型进行台风轨迹预测。首先，会对训练数据集进行分析，确定特征的选取。然后会讨论激活函数的选取以及使用何种优化策略来更新梯度。最后，详细介绍模型的网络结构设计。

### 3.2.1 台风轨迹数据分析

本次课题研究所采用的数据集为中国气象局热带气旋资料中心提供，是 1995-2015 年真实的台风路径（经纬度序列）以及对应时刻的静止卫星图像。本章主要用到轨迹数据，而图像数据会在下一章节中用到。受限于计算机的存储能力和计算能力，从数据集中选取 4 天的数据进行本课题的研究。其中前 3 天的数据用于训练，最后一天的数据用于验证。影响台风轨迹走向的特征很多，从中选取 7 个最重要的特征进行实验，分别是：

(1) ID：台风轨迹的唯一标识，为 int 类型，同一条轨迹上不同时间点采集到不同轨迹点的 ID 都是相同的，一般来说，在处理轨迹问题时都是将 ID 相同的点归为一类

(2) TIME：台风轨迹数据时间戳，由于选取的是连续 4 天的台风轨迹数据，所以使用相对时间进行记录，并且按小时进行划分，分为 000-090 小时，以 6 小时为间隔，前 3 天数据（000-066 小时）作为训练数据，第 4 天数据（072-090）用于对模型的验证。有了 ID 之后，在对相同 ID 的数据时间戳做进一步排序，这样数据就有了时间性

(3) I：台风强度标记，按照国家标准（GB/T 19201-2006），为 int 类型，这个值越大，说明台风强度越强。

(4) LAT：台风轨迹数据纬度坐标，为 float 类型数据，精确到（0.1°N），反应当前时间点下的位置坐标，在某个时间点的下坐标由经纬度唯一决定。

(5) LON：台风轨迹数据经度坐标，为 float 类型数据，精确到（0.1°E），反应当前时间点下的位置坐标，在某个时间点的下坐标由经纬度唯一决定。

(6) PRES：台风中心最低气压，为 int 类型，单位为（hPa），会对台风路径

的走势和偏移造成影响。

(7) WND: 台风轨迹近中心的最大风速, 为 `int` 类型, 单位为 (`MSW, m/s`), 会对台风路径的走势和偏移造成影响。

从数据集中可以看出, 一条轨迹由若干个点构成, 这些样本点来自卫星在不同时间点对轨迹数据进行的采样。如此一来, 轨迹数据可以看成基于时间序列的数据集, 可以把台风轨迹预测问题转化为时间序列分析的问题进行处理。时序数据如图 3-3 所示:

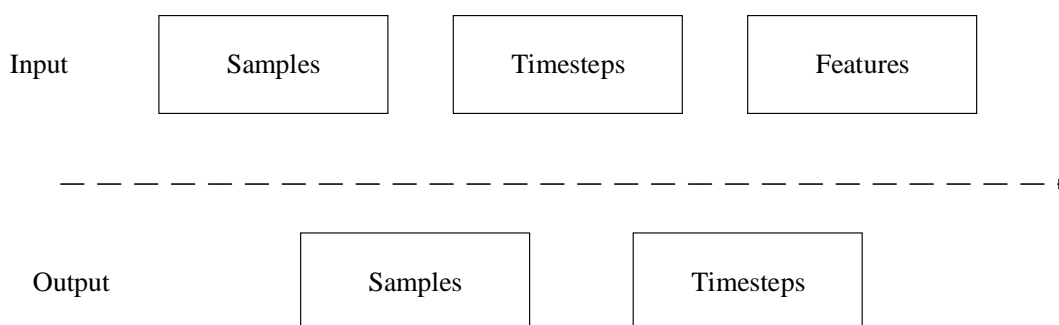


图 3-3 时间序列数据图

时间序列数据不同于普通的数据, 他并不是由等长的向量构成的, 而是由不等长的特征点组成的, 因此是一种变长的数据。时间序列数据的输入数据是以三维的数据, 而不像普通的数据是二维的。第一维是 `Samples`, 代表该数据属于哪个样本, 因此同一样本的数据应该将其归类到一起。第二维是 `Timesteps`, 即代表数据的时间戳, 也就是采样的时刻, 是个瞬时时间。而第三维是 `Features`, 代表的是特征, 也即是模型中用到的能对结果产生影响的特征, 可以说从输入上来说, 时间序列数据比普通数据要多出一维时间戳。而从输出来说, 时间序列数据至少得有两维, 分别是 `Samples` 和 `Timesteps`, 这是因为不同于普通的数据, 时间序列分析特别是对于时间序列的预测, 往往会有不止一个的输出, 且这些输出会按照时间序列从小到大进行排列, 并依次从模型中输出出来。基于此, 可以通过如下流程来转化轨迹数据, 使其能满足后续模型训练的需要。其处理流程如图 3-4 所示:

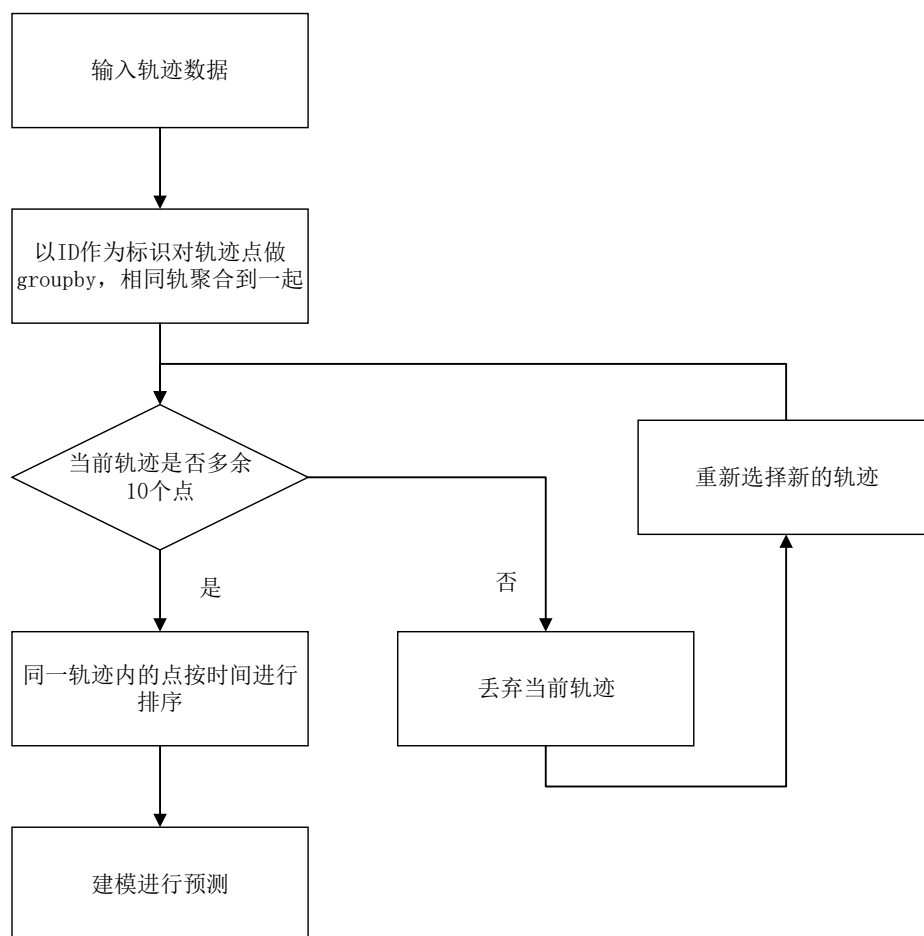


图 3-4 台风轨迹数据处理流程图

因为台风轨迹数据是时序数据，同时后续建立的神经网络结构是适用于时序数据的预测，所以需要保证读入数据的时序性。由于 ID 是轨迹之间的唯一的标识符，因此同一条轨迹上的所有点，具有相同的 ID，于是可以用 Python 当中的科学计算包 Pandas 读入数据，并以 ID 为关键字做 Groupby，这样就能保证所有 ID 相同的轨迹点被聚合成一个“超级结点”。接着，由于深度学习是需要大量的数据为基础的，如果台风轨迹中的轨迹点过少，很难从历史数据中学出规律，于是剔除那些轨迹点过少的轨迹。最后为了保证训练数据的时序性，需要对“超级结点”内的数据按照时间序进行排序。这样就能保证在每一条轨迹都可以独立进行训练，同一轨迹中所有轨迹点都具有时间序列性。

前文已经叙述了影响台风轨迹的 7 个特征，其中 ID 是唯一标识符，TIME 是时间戳，而 LAT, LON 代表经度和纬度，这 4 个特征都具有很有的时序性，通过这 4 个特征可以很好地去描述台风轨迹的形状，方便后续模型的学习。但台风轨迹预测不同于简单的做时间序列预测，还会受到其他特征的影响，因此有必要对台风强度(I)，台风中心最低气压(PRES)，近中心的最大风速(WND)3 个特征做一步

分析，看其对结果的影响。其特征的分布图如图 3-5 所示：

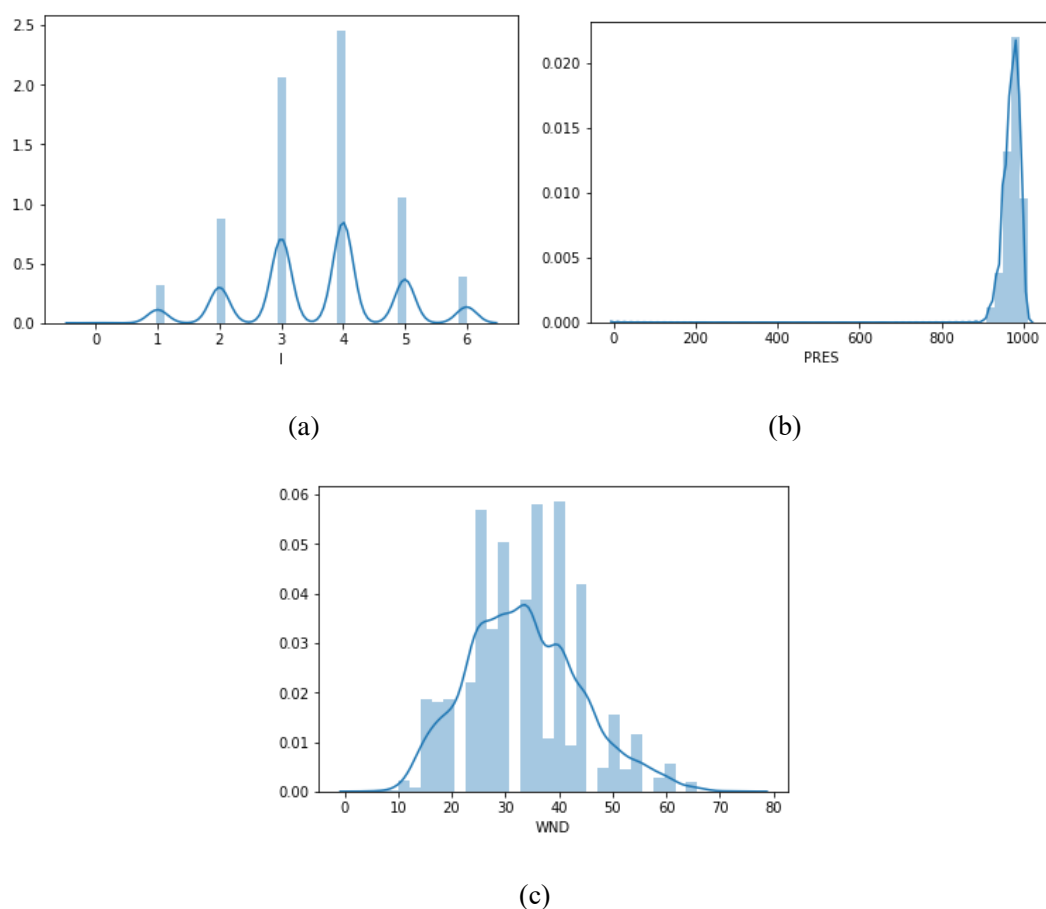


图 3-5 特征分布图

上图为特征分布图，(a)为强度  $I$ ，(b)为台风最低气压  $PRES$ ，(c)为最大风速  $WIND$ 。通过分析，可以发现台风强度主要分布在 2-5 级之间，而 1 级和 6 级很少，说明在 2-5 级之间的台风强度对路径影响较大。而中心最低气压几乎都分布在 900-1000 之间，其余数值基本为 0，说明只有中心气压在这个范围内，才会对轨迹的偏移产生较大影响，不在这个范围内的可以忽略。近中心最大风速的分布就比较分散，大概在 20—50 之间属于，其他范围的数据较少。基于上面的分析，可以得出特征的大致范围。然后进一步观察特征和数据范围之间的联系，如果特征不在众数范围内，且特征对轨迹最终的走势影响较大，则说明这个重要特征，在后面构造模型的时候应该增加这个特征的权重。反之，则说明此特征对于提升效果不起决定性作用，可以减小权重，甚至忽略这个特征。然后通过分析特征之间的关联性，可以发现除了经度与纬度以外，其余 3 个特征之间并无耦合关系，而是独立作用与数据集，因此，在后续建模时，应该把 3 个特征都加入，同时做好归一化，保证不会因为单位量纲的不同对结果产生较大的影响。

### 3.2.2 激活函数和优化策略

在 2.1 节中, 已经介绍过 Seq2seq 模型的演化过程及其基础知识, 但对于使用深度学习模型进行台风轨迹识别来说, 单单了解这些基础知识是不够的。要进一步去剖析模型内部结构和机制, 熟悉其参数更新的过程。这就需要了解模型的激活函数选取以及参数的更新策略, 而本小节, 就基于这两个问题展开深入的研究。

如前文所述, 深度学习是一种表征学习, 引入激活函数式为了让神经网络有更好的表达能力, 激活函数层是神经网络的非线性变化层, 用于对神经网络进行非线性变化, 正是由于有了激活函数层的存在, 才能使得是模型可以做得足够深。试想一下, 如果不对深度学习做非线性变化, 那所有变化都是线性的, 再复杂的线性变化都是可以等效, 无论模型做得再深, 跟只做一层的模型是没有任何分别的。但是这种非线性设计不可避免的所带来的一系列副作用, 研究人员不得不设计更多的激活函数来约束非线性的合理范围。主流的激活函数如图 3-6 所示:

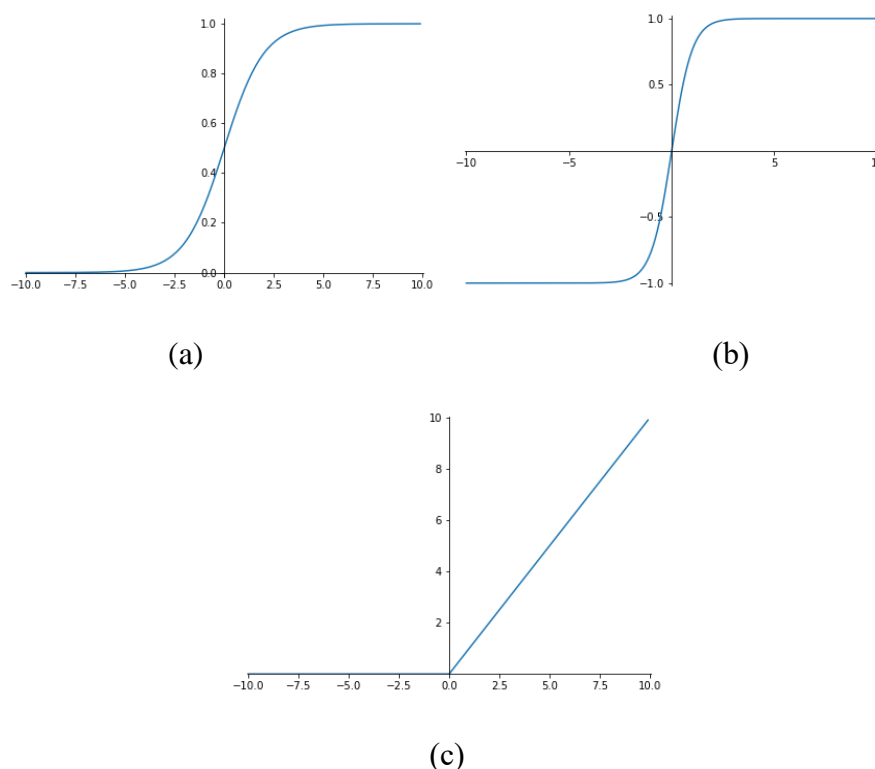


图 3-6 激活函数图

如上图所示, (a)为 Sigmoid 激活函数, (b)为 tanh 激活函数, (c)为 ReLU 激活函数。Sigmoid 一般来说被用作神经网络的阈值函数, 将变量映射到 (0, 1) 之间, 但是受限于梯度下降算法, Sigmoid 函数在后期会出现梯度为 0 的一正一负两块“死区”。tanh 激活函数, 将期望均值平移到 0 这一理想状态, 是 Sigmoid 的改进,

但是依然无可避免的出现“死区”。ReLU 激活函数,彻底消灭了正半轴上的“死区”,而且 ReLU 函数的函数结构非常简单,但是期望均值偏离 0 更远了,同时对于负半轴来说“死区”更为严重了,直接到了 0。ReLU 作为现今当下运用最火的激活函数,在图像领域取得了巨大的成功,ReLU 有效的缓解了梯度弥散,使得网络深度的天花板第一次被打破。借鉴于 ReLU 函数在卷积类的神经网络上取得的重大突破,人们开始考虑把 ReLU 函数引入循环神经网络,但是不曾想到却适得其反,并没有起到正向的作用。最初,ReLU 函数发明的初衷就是为了克服 tanh 函数在 RNN 中的“梯度消失”现象,但是由于 Relu 的导数不是 0 就是 1,恒为 1 的导数相乘又会引入新的问题那就是“梯度爆炸”。ReLU 在 RNN 中会导致非常大的输出值,由于每个神经元的输出都会作为输入传给下一个时刻本神经元,这就导致会对系数矩阵执行累乘,如果其中有一个特征值大于 1,若干次累乘之后数值就会很庞大,在前向传播时候导致溢出,后向传播导致梯度爆炸。而在 CNN 中,不存在系数矩阵的累乘,每一层的系数都是独立的,互不相同,有大于 1 的,有小于 1 的,所以在很大程度上可以抵消,而不至于让最终的数值很大。因此 ReLU 函数并不能代替 tanh 函数,如果非要使用 ReLU 函数,必须设置梯度截断,但这是一种更加耗时耗力,且无法保证结果更优的方法。而 Sigmoid 函数的导数范围是 $(0,0.25]$ , tanh 函数的导数是 $(0,1]$ 。由于 RNN 中在反向传播过程中会执行很多累乘,小于 1 的数大量累乘会导致梯度越来越接近于 0,出现“梯度消失”现象。而 tanh 梯度更大,收敛速度更快,因此出现“梯度消失”的情况要优于 Sigmoid。同时 Sigmoid 关于不是零对称的,这在训练过程中会导致偏移现象,而 tanh 函数是关于原点对称的数据输入,所以训练效果更好。因此 Sigmoid 也无法代替 tanh 成为激活函数。综上所述,对于 RNN 类的模型,应该选择 tanh 作为激活函数,如前文所述,Seq2seq 模型实质上是由编码器和解码器两个阶段组成的,而编码器和解码器的过程都是由 RNN 组成的网络,因此,Seq2seq 模型的激活函数的选择应该遵循 RNN 中激活函数的选择,选用 tanh 作为激活函数。

在探究完激活函数之后,紧接着研究一下模型的梯度下降算法,几乎所有的深度学习模型,都是采用梯度下降算法来进行优化训练的。一般来说,梯度下降算法主要分为批量梯度下降算法,小批量梯度下降算法和随机梯度下降算法三类。批量梯度下降算法,一次将整个训练集读入内存,由于数据集过大,模型的收敛速度普遍比较慢。随机梯度下降算法却走截然不同的路线,该法每次会从训练集中随机选出一个训练样本进行梯度下降,同时完成一次模型参数更新。小批量梯度下降算法是二者的结合方案,选取训练集中的少量样本进行计算,一般设置不同大小的 batch,保证训练的稳定性。一般来说,对于时间序列模型,采用批量梯度下降算法肯定是

不可取的。一方面是显存比较昂贵，一般显存的大小不够；另一方面是时间序列数据是一种不等长的数据，如果读入的 `batch_size` 过大的化，需要想办法将其变成等长的，否则没法以 `batch` 进行训练，如果整个数据集一起变成等长的话，势必会造成训练数据过于稀疏。对存储造成极大的浪费。因此批梯度下降算法并不合适。SGD 伪代码如表 3-3 所示：

表 3-3 随机梯度下降算法

随机梯度下降算法
初始化： Learning rate schedule $\epsilon_1, \epsilon_2, \dots$ Parameter $\theta$ $K \leftarrow 1$ While stopping criterion not met do: Sample a minibatch for $m$ examples from training set $\{X_1, X_2, \dots\}$ with corresponding target $y^{(i)}$ Compute Gradient $\hat{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(X^{(i)}; \theta), y^{(i)})$ Apply update $\theta \leftarrow \theta - \epsilon_k g$ $K \leftarrow K + 1$ End while

首先，对参数设置初始值，并初始化不同的学习率。紧接着，会随机采样，然后根据和目标的差距，进行梯度更新，不断使得参数往目标的方向进行调节，如果当前学习率收敛了，会进一步选取更小的学习率进行学习，直到最小的学习率也可以收敛时，则认为模型收敛，整个训练过程完成。但是在实际训练中，却发现朴素的 SGD 存在一些问题，梯度下降算法的期望情况是希望优化算法能够收敛速度快，并且想找到全局最优。对于凸函数来说，其仅有一个极值点，就是全局最优点，此时采用梯度下降算法是可以收敛到最优点的，因为沿着下坡的道路走就可以了。但是现在的深度学习模型一般都是非线性结构，所以一般是会面临非凸函数问题，就会存在很多局部最优点，一旦梯度下降算法跳进局部陷阱，就很会难走出来，梯度下降算法此时也就变得不再可靠。基于此，需要使用一种朴素 SGD 的优化，Adam 算法<sup>[38]</sup>。Adam 计算梯度的一阶矩估计和二阶矩估计，用不同的学习率对应不同的参数，在每次迭代的时候不仅仅可以自适应地调整每个参数的学习率，而且在每次迭代的时候会引入一个动量，给模型增加动能，防止模型收敛于局部最优解。Adam 算法伪代码如表 3-4 所示：



表 3-4 Adam 算法

Adam 算法
初始化: 初始化时间步 $t=0$ 初始参数 $\theta$ 步长 $\epsilon$ 一阶和二阶矩变量 $s=0, r=0$ 矩估计的指数衰减率 $\rho_1, \rho_2$ 用于数值稳定的小常数 $\delta$ While 没有达到停止准则: 从训练集中采 $m$ 个样本的小批量 $\{X^1, \dots, X^m\}$ , 对应目标为 $y^{(i)}$ 计算梯度 $\hat{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(X^{(i)}; \theta), y^{(i)})$ $t \leftarrow t + 1$ 更新有偏一阶矩估计: $s \leftarrow \rho_1 s + (1 - \rho_1) g$ 更新有偏二阶矩估计: $r \leftarrow \rho_2 r + (1 - \rho_2) g \odot g$ 修正一阶矩的偏差: $\hat{s} \leftarrow \frac{s}{1 - \rho_1}$ 修正二阶矩的偏差: $\hat{r} \leftarrow \frac{r}{1 - \rho_2}$ 计算更新: $\Delta \theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{r} + \delta}}$ 应用更新: $\theta \leftarrow \theta + \Delta \theta$ End while

Adam 算法结合了 Momentum 算法和 RMSProp 算法。用基础 SGD 在训练过程中 loss 会陷入局部最优最优的情况，导致 loss 就不再更新，同时，改用了其他自适应学习率的方法也无法解决这类问题。因为需要一个动量，在梯度下降过程中帮助打破枷锁，突破模型的局部最优值，去寻求全部最优值。换用 Adam 算法之后，有了动量的助推，模型的 Loss 不再陷入局部最优解，开始向全部最优方向收敛，模型的参数也重新开始更新，训练的准确率也得到了极大的改善。tanh 函数有效解决“梯度消失”的问题，而“梯度爆炸”问题，却迟迟没有得到解决，因此引入了梯度截断技术。在梯度下降的过程中，有些点会出现梯度突然增大，用非常大的梯度进行参数更新，为了避免这种情况，当梯度大于一定阈值时，就对梯度进行截断，称为梯度截断。在梯度更新过程中，可以把梯度的模限定在一个区间内，当梯度的模小于或者大于这个区间时就进行截断。最直接的做法通过某个阈值来控制系数的大小，若系数小于某个阈值便将该系数设置为 0。通过在 RNN 类的网络加入“梯度截断”，能比较好的解决“梯度爆炸”，使得模型能更快的收敛。

### 3.2.3 模型结构设计

在前文中，已经就深度学习中两个重要问题进行了深入的研究，一方面，分析了训练的台风轨迹数据集，了解了数据集中各项特征之间的关联特性；另一方面，分析了模型的激活函数以及梯度更新策略。接下来，在这一章节当中，将进一步研究深度学习模型结构的设计。

在前文中已知，特征除了经纬度，还有强度，中心最低气压和最大风速，将其编码成为一个向量，作为  $X$ ，当作特征。同时，为了加快训练的速度，必须以 `batch` 来进行训练，而不是一条数据一条数据的训练，所以需要对其进行变化，使得不等长的时序数据变成等长的数据，而添加的部分补上一个空值，这样不会对结果产生影响。如前文所述，本次实验采用的 Pytorch 深度学习框架，在 Pytorch 框架中，一切对象都是张量，而现行当下流行的 GPU 并行技术更多的是对矩阵运算和浮点数运算进行加速，基于此，需要把  $X$  和  $Y$  都转化成张量，然后以张量的形式输入到定义好的网络当中。接着会设置轮数，并在每一轮中使用 Adam 算法更新参数和梯度，直至训练完成。在前文中也分析过，由于数据是时序数据，使用前馈神经网络并不能很好的解决，所以使用基于注意力机制的 Seq2seq 模型进行台风轨迹预测，其模型的架构图如图 3-7 所示：

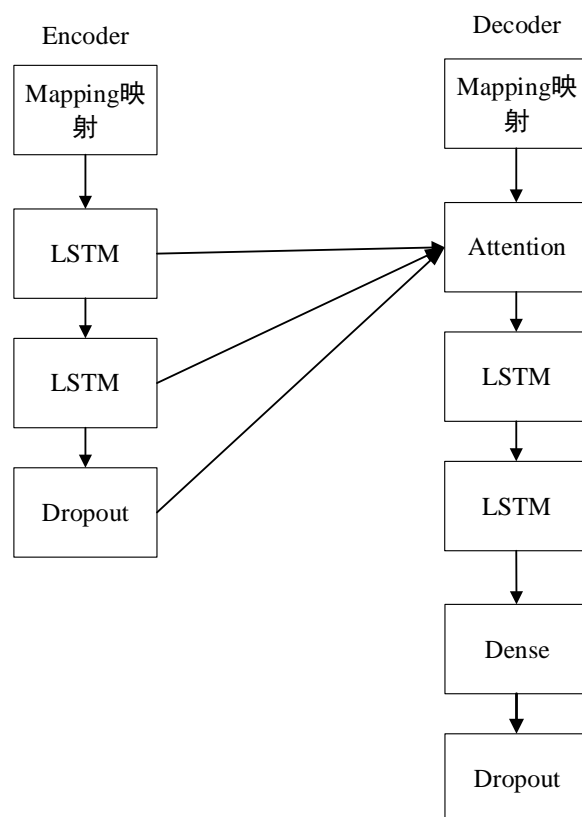


图 3-7 模型网络架构图

本次实验采用基于注意力的 Seq2seq 模型，模型一共分为 2 个部分，一部分是编码器，一部分是解码器。在编码器部分用于读取和编码输入序列，而解码器部分负责读取编码的输入序列并对输出序列中的每个元素进行预测。一般来说，使用编码器学习出前半段序列的 **pattern**，然后用解码器的输出作为对后半段数据的预测。因此在编码器阶段，将时序数据编码成一个固定长度的向量，而解码器负责解码固定长度的向量并输出预测序列。所以对于回归类的问题，一般所用到模型结构都是一个编码器网络，后面接一个解码器网络，最后还会接一个全连接网络，对台风轨迹进行结果的预测。模型各部分的参数如表 3-5 所示：

表 3-5 模型网络架构

编码器 Encoder	
LSTM	100, 100
LSTM	100, 100
Dropout	0.2
注意力机制 Attention	
Linear	200, 100
解码器 Decoder	
LSTM	100, 100
LSTM	100, 100
Dense	100, 1
Dropout	0.2

编码器实质是 LSTM 网络的组合，台风轨迹数据首先会被映射成合适网络的张量，然后定义 2 层的 LSTM 网络模型，大小均为  $100 \times 100$ ，对数据进行编码。LSTM 网络在训练过程中比较容易过拟合，引入 Dropout 技术使得其中一些神经元在一定概率下不被激活，dropout 的值设置为 0.2，完成编码器过程。注意力机制是一个线性回归模型，因为注意力机制需要编码器中的每个 LSTM 网络和解码器中的每个 LSTM 相互作用。编码器的实质也是 LSTM 的组合，解码器阶段是 2 层的 LSTM，同时注意力以后的结果又要跟解码器的过程相接，所以注意力阶段需要一个  $200 \times 100$  的 Linear 模型。解码器阶段就是对注意力以后的结果做解码，所以解码器是一个跟编码器相同的 2 层  $100 \times 100$  的 LSTM 结构，另外由于最终需要对结果进行预测，所以是再接一个全连接层，大小为  $100 \times 1$ ，最后引入 Dropout 对在一定概率下部分神经元失活，防止发生过拟合。

### 3.3 实验结果

在本节中，主要就前面的研究内容进行实验结果的展示，并对实验的结果做出相应的分析。首先展示数据预处理部分，在数据预处理部分中，首先展示对轨迹数据的滤波结果，并对结果做出分析；然后会展示轨迹的简化，并对简化后的轨迹加以说明。在分析完预处理部分的实验结果之后，会进一步展示台风轨迹预测的实验结果，这也是本章的重点的内容。首先会展示训练过程中的 Loss 函数，保证训练过程的正确性，接着，通过饼状图直观展示基于注意力机制的 Seq2seq 模型预测结果分别和基于传统机器学习的 Xgboost 和基于深度学习的 LSTM 模型预测结果进行比较，并进行实验结果分析。最后还会用基于注意力机制的 Seq2seq 模型预测未来 6h, 12h, 18h, 24h 的轨迹情况，并画出折线图分析其准确率。

#### 3.3.1 台风轨迹清洗

使用改进卡尔曼算法对台风轨迹进行滤波效果如图 3-8 所示：

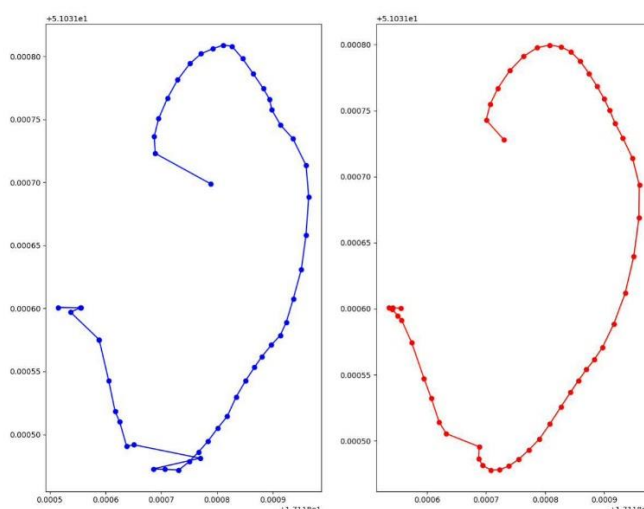


图 3-8 台风轨迹滤波效果图

图中左边蓝色为原始轨迹，右边红色为经过滤波后的轨迹。从图中可以看出，使用改进的卡尔曼算法滤波之后，台风轨迹相比较原先更加平滑，剔除了一些由于采样误差产生的野值点，整个轨迹的时序状态也更加趋于稳定。在结合了影响台风轨迹走向的特征之后，改进卡尔曼滤波算法极大提高了训练数据集的质量，减少了采样误差对后续结果产生的影响，为后续的使用深度学习模型打下了坚实的基础。

使用最小扇形简化算法简化轨迹后的结果如图 3-9 所示：

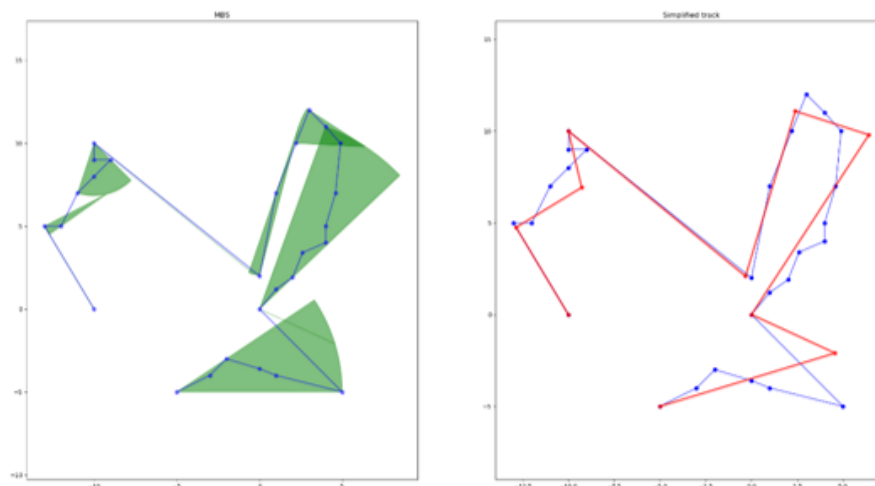


图 3-9 台风轨迹简化效果图

图中左边为原始台风轨迹图，右边蓝色曲线为原始轨迹，红色曲线为简化后的台风轨迹。从图中可以看出，经过简化之后的轨迹相比较原始轨迹并没有过多的信息丢失，证明台风轨迹信息丢失很少。由于在简化过程中，利用了所有的特征，而不仅仅是只使用经纬度，因此不会对后续使用深度学习模型对台风轨迹精度上产生较大影响，所以说明简化算法是行之有效的。

### 3.3.2 台风轨迹预测

在前文中，已经就台风轨迹滤波和台风轨迹简化进行了实验，同时实验结果也证明了之前提出的方法是正确的。接下来，将对台风轨迹预测算法进行测试因为本次实验中有大量的矩阵运算和浮点运算，所以本次实验需要用到 GPU 进行加速，整个实验在云服务器上进行，实验的硬件参数如表 3-6 所示：

表 3-6 实验的硬件参数

内存	CPU	GPU	显存
16GB	Intel core i7-8700	NVIDIA RTX 2070	8GB

下面将对序列数据建立基于注意力机制的 Seq2seq 模型，训练过程中的 Loss 函数如图 3-10 所示：



图 3-10 训练 Loss 图

因为需要使用 Seq2seq 模型对台风轨迹进行预测,故这里使用 L2 的 Loss 函数来对结果误差进行衡量。从图中可以发现, Loss 起初下降很快,然后逐渐趋于平缓,最后几乎变为直线。这表明模型在学习的过程中不断收敛,而后期随着距离目标值越来越近,需要调小学习率,进行新的迭代,Adam 算法的使用保证模型在梯度下降过程中,不会陷入局部最优解,说明训练过程是行之有效的,而且模型在测试集中也比较良好,说明训练过程没有出现明显的过拟合。下面将会对三种方法预测结果做对比,实验结果如图 3-11 所示:

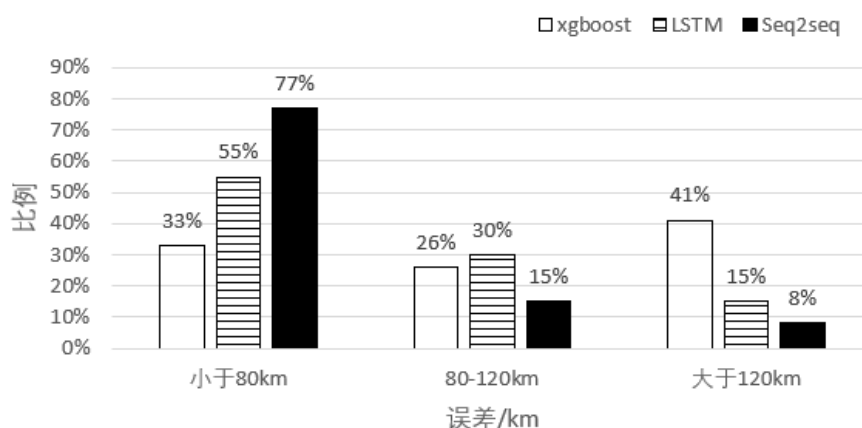


图 3-11 预测结果对比图

通过与测试集中台风坐标地理距离,将误差分为3个档次,第一档次为80km以内的,第二档次为80-120km,第三档次为大于120km的。在本实验中,分别使用了传统机器学习算法,LSTM模型和Seq2seq模型对序列数据进行台风轨迹预测。首先在传统机器学习算法中,采用xgboost<sup>[39]</sup>算法。可以看到在传统机器学习算法的预测中,小于80km的大概占33%,80-120km的大概占26%,而大于120km的,大概占41%。接着是对于LSTM模型,在LSTM中,小于80km的占55%,80-120km的占30%,而120km以上的占15%。最后是基于注意力机制的Seq2seq模型,其中小于80km的占77%,80-120km的占15%,而120km的约占8%。从实验中可以看出,Seq2seq模型相比较于传统的机器学习算法,在预测的准确度上面有了质的提升。而相比于同样LSTM结构,因为Seq2seq模型引入了注意力机制,使得在预测的过程中更关注于与当前状态相关的其他状态,而忽略掉去当前状态无关的状态,因此在小于80km范围内,远好于LSTM模型。上述结果证明了基于注意力机制的Seq2seq模型比起当下现有的方法,在预测精度上,确实是有了比较大的提升。下面将会对Seq2seq模型预测结果做进一步的测试,通过选取未来不同时间进行预测,比较预测精度,实验结果如图3-12所示:

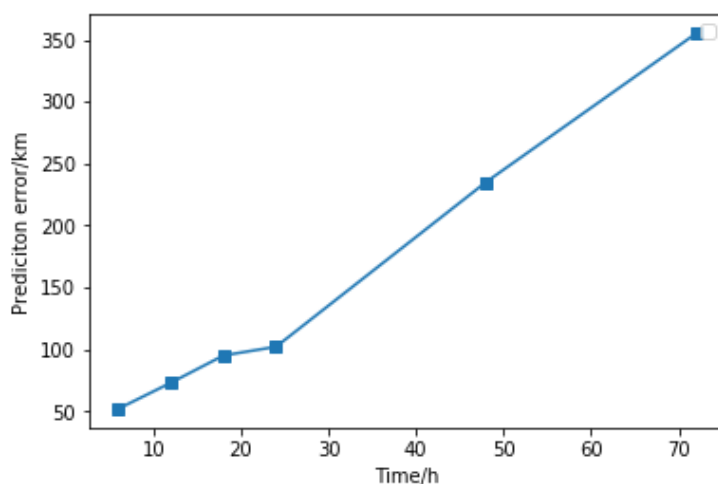


图 3-12 Seq2seq 误差实验

在本次实验中分别测试了6h, 12h, 18h, 24h, 48h和72h的预测结果,结果表明,随着时间的增加,预测精度明显降低,在6h, 12h, 18h内的预测结果相对准确。同时将在24h内的预测结果跟领域专家预测结果相比较为接近,而48h, 72h的结果跟气象专家预测结果相比却相差较大,实验说明,Seq2seq模型对于24h内的台风轨迹预测结果是可靠的,有助于帮助相关机构预测台风轨迹。最后在进一步对24h内的数据,探究模型的预测精度与数据集比例大小的关系。实验结果如图

3-13 所示:

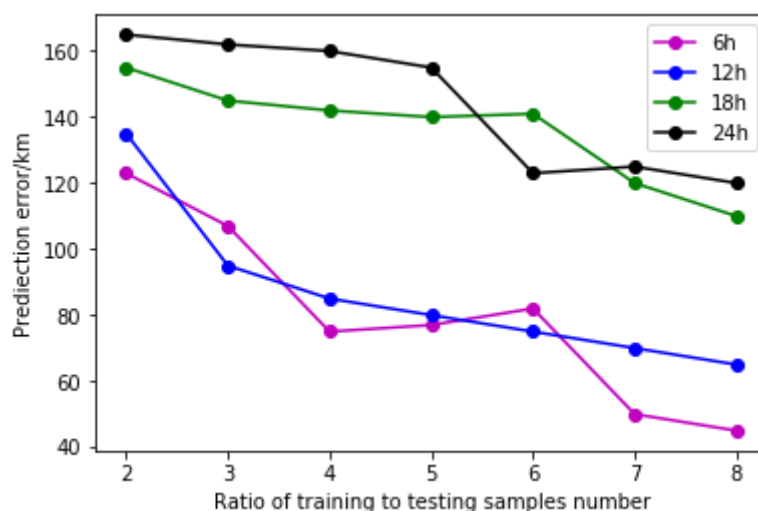


图 3-13 精度与数据集比例

上图中对 24h 内的数据进行进一步的实验, 通过设置训练集和测试集的比例分别为 2:1,3:1,4:1,5:1,6:1,7:1,8:1, 分别对 6h, 12h, 18h, 24h 的数据进行进一步实验, 通过实验发现预测的精度随着训练集和测试集的比例增大逐渐提升, 这与深度学习需要大量的训练数据做为支撑的理念不谋而合, 说明了在数据量足够大的情况下, Seq2seq 模型预测精度远好于传统机器学习算法。

最后需要进一步对于预测精度不足的地方进行分析。在实验中, 以相对误差来衡量预测台风轨迹与真实台风轨迹之间的差距, 但是有许多影响因素确实不可避免的。首先, 台风有很多路线是会经过海洋, 而不是走在陆地上, 相比于陆地, 海洋更加开阔和不可控, 很难通过时序模型来判断海洋对于风向的影响。其次, 台风的方向如果发生突变, 会对结果产生很大的影响, 因为在预测过程中, 没有很好的前序状态可以依赖。最后台风在登陆前, 会产生很大的相对误差, 导致结果不可控。基于上述原因, Seq2seq 模型在所预测的台风轨迹和实际的台风轨迹之间依然会存在一定的误差。

### 3.4 本章小结

在本章中主要使用序列数据对台风轨迹进行预测。首先, 讨论了如何对数据集进行了清洗。提出了改进的卡尔曼算法对数据集进行滤波, 去掉轨迹中的野值点, 提高了数据质量。然后基于最小扇形简化算法对台风轨迹进行了简化, 减少了训练过程中的计算量。接着讨论如何基于注意力的 Seq2seq 模型对台风轨迹进行预测。首先, 对清洗后的数据集进行了分析, 并探究了特征之间的关系, 选取了有用的特



征。接着，分析了为什么使用  $\tanh$  作为激活函数，以及为什么使用 Adam 进行梯度下降。然后讨论了模型的网络架构设计。最后对本章提出的方法进行了实验，首先展示了轨迹滤波和简化后的结果，接着可视化了训练过程，展示了训练过程中 Loss 的变化，并当下比较流行机器学习模型 Xgboost 及在时序数据处理上运用比较多的深度学习模型 LSTM 模型预测的结果进行了对比，验证了 Seq2seq 模型相较于现下的方法，在精度上确实比了较大的提升。最后，结合台风产生的背景和台风入侵过程中的路径，分析了预测值和真实值之间存在误差的原因。

## 第四章 基于图像数据的台风轨迹预测

第三章中，主要针对序列数据，使用基于注意力机制的 Seq2seq 模型提升了预测精度。而训练数据集中，除了序列数据，还有图像数据，本章将利用图像数据进一步提升预测精度。首先，对图像数据进行预处理，一方面压缩图像节省训练内存和显存；另一方面，为了从图像中更好的提取特征。接着，提出基于时序性的 GAN 模型进行台风轨迹预测，提高现有方法预测精度；然后通过遗传算法 GASEN 创新性地将本章结果与第三章序列数据的预测结果进行多模型融合，进一步提高预测精度。最后对研究内容进行实验，并对实验结果进行说明，证明方法的可行性。

### 4.1 数据预处理

本次课题研究所采用的数据集为中国气象局热带气旋资料中心提供，是 1995-2015 年真实的台风路径（经纬度序列）以及对应时刻的静止卫星图像。在上一章节中，主要用到了序列数据，在本节中，将使用卫星图像数据进一步进行台风轨迹预测。卫星图像数据均为灰度图像，采用 pgm 格式存储，像素均为 250\*238，受限于计算机的内存和显存，同时也为了能更好的为后续的使用 GAN 模型进行训练提供保障，需要先对图像数据进行预处理。

#### 4.1.1 图像格式转化

受限于计算机的存储能力和计算能力，在本次课题中，选取 4 天的台风轨迹卫星图像作为训练数据，按照相对时间进行记录，并且按小时进行划分，分为 000-090 小时，以 6 小时为间隔，其中前 3 天数据（000-066 小时）作为训练数据，第 4 天数据（072-090）用于对模型的验证。卫星图像数据保证和之间的轨迹数据一一对应。图像数据全部使用 pgm 格式，左上角经纬度为 E70°, N70°，图像空间分辨率为 0.05°，像素灰度范围为 [0, 255]。

pgm 是灰度图像格式中一种最简单的格式标准，是一种便携式灰度图像格式，pgm 格式的存放方式非常简单，不对数据进行压缩，直接存放。pgm 由文件头部和数据部分组成，头部如下所图 4-1 所示：

格式	图像宽度	图像高度	灰度最大值

图 4-1 pgm 头部格式图

第一部分是格式类型，大小为 2 字节，pgm 格式的文件一般分为 2 种，P2 和 P5。P2 使用字符串来表示像素；P5 格式用二进制来表示像素。接着是图像的宽度和图像的高度，这部分为 4 字节，灰度图仅有 2 通道构成，一般来说由一张图片的大小由宽度  $\times$  高度来表示。最后是灰度最大值，对于灰度图来说，最大值一般为 255，所以一般是 3 字节。数据部分为像素值，灰度图中为灰度值，按照图像从上到下，从左到右进行存储。图像中像素灰度值的表示 P2 格式和 P5 格式有所不同。P2 格式是用字符串表示，所以 P2 格式可以直接用记事本打开，而 P5 格式是二进制，记事本打开就是乱码。

如前文所述，pgm 是一种没有压缩的格式，这就导致图片的大小会非常大。图像数据不同于序列数据，图像数据是一种天生等长的数据，这就为 GPU 并行技术来加速训练提供了便捷。如果图像过大，不仅仅会给内存和显卡造成了巨大的压力，而且也会导致并行化程度不够，失去了 GPU 加速的意义。基于此，需要对图像数据进行压缩。在综合比较各种格式之后，决定使用 jpg 格式对图像进行存储。

一方面，需要对图像数据进行一定的压缩，另一方面，希望压缩后的图像尽量无损，为了要满足无损的压缩，必须对图像进行编码。编码算法必须满足以下 3 点：

- (1) 编码后的编码必须可以准确解码，编码必须和源码一一对应
- (2) 编码和解码之间要容易转化，可以很容易定位信息末尾，支持直接对编码进行解码而不用知道完整的编码信息；
- (3) 编码必须是压缩的，否则失去了编码的意义

基于上述要求，选用算术编码<sup>[40]</sup>，其编码算法如表 4-1 所示：

表 4-1 算术编码编码器算法

算术编码编码器算法
初始化： $L = 0$ $H = 1$ While there are 输入 symbols: 获取一个 symbol $R = H - L$ $H = L + R * HR(\text{symbol})$ $L = L + R * LR(\text{symbol})$ End while

编码器时初始化间隔  $L=0$ ， $H=1$ ，每次操作，编码器将区间分为几个不同的子

区间,子区间的大小与新事件产生的概率成正比。当前状态下编码器选择的子区间应该对应下一个肯定会发生的事件,同时将其标记为新的当前区间,并进行新一轮的迭代,直至最后收敛,把当前区间的下边界作为给定事件序列的算术编码。

下面再来看看解码算法,对于算术编码,其解码算法如表 4-2 所示:

表 4-2 算术编码解码器算法

算术编码解码器算法
初始化: 获取解码字符 do: 查找范围跨越编码数字的符号 输出 symbol 范围是 symbol.L- symbol.H 从编码数字中减去 symbol.LowValue 将编码数字除以范围 until no more symbols

对于不同的信号源,对应不同的概率,会在 $[0,1]$ 范围内将其映射为对应的区间。然后对于一个输入的二进制序列,分别以上一次范围的十分之一和当前的下界作为当前的范围,如此下去,直到最后输入编码。初始时,如果消息的长度对于编码器和译码器已知,译码过程一定会收敛,不可能无限迭代下去。但在实际操作中,会为译码器添加一个专用的终止符号作为标识,当程序运行到终止符就停止译码过程。算术编码解决了哈夫曼编码不能解决的一些问题,在编码过程中,每个字符都会使用几个不同的二进制位区别表示。算术编码是通过二进制形式进行编码,消息长度通过 0 和 1 之间的间隔进行表示,消息越长,间隔越小,编码之间关系越紧密,就需要更多的二进制位。

如前文所述,可以借助 python 的 PIL 工具包,对 pgm 格式的原图像进行转化,将其转化为 jpg 格式。首先,遍历整个文件夹,每次查看当前被遍历到的文件是否为 pgm 格式的,如果是 pgm 格式的,调用 os.path.join 函数获取当前对象,存为 f。然后获取当前文件的名称, filename, 并对其做截断,将其后缀改为.jpg,最后通过调用 PIL 中的 image.open 函数和 image.save 函数将其保存为转化为 jpg 格式。如此一来就将 pgm 格式的文件转化为 jpg 格式,在不损坏图像质量的情况下,极大的降低了图像的存储空间。

### 4.1.2 图像数据裁剪

在上一节中，通过对图像进行算法编码，将 pgm 格式的灰度图转化为 jpg 格式，使得图像占用空间大大减小。但由于图像数据是每隔 6 小时做一次采样，轨迹数量众多，这就造成训练数据集的图片基数很大。同时由于提供的图片均为 250\*238 像素的，所以即使经过了格式转化，训练数据集依然非常大。而实际在训练过程中，不需要使用这么大像素的数据，因此需要对数据做进一步裁剪。数据分为了训练集和测试集，对于深度学习来说，训练集和测试集对数据的需求是不同的。本次课题中，将训练集中的图片裁剪成像素值为 32\*32 的图片，这是因为一方面如此可以在训练过程中节约内存和显存，另一方面，专注于图像的一小部分可以更好地了解图像的细节。而对于测试集中的数据，为了保证测试数据的准确性，应该尽量用完整的数据进行训练。基于此，为了满足对训练数据集的需要，需要对裁剪图像。

图像缩放主要通过减少像素和增加像素来改变图像的尺寸，图像裁剪最大的挑战就是在裁剪过程中最大程度的保证图像不被损坏。一般来说图像缩小是通过减少像素个数来实现的，因此要根据缩小的尺寸，从原图像中选择合适的像素点，使图像缩小后可以保持原有图像的特征。图像的缩小算法有很多，一般来说比较主流的是：等间隔采样和局部均值。

等间隔采样指的是在原图中每间隔一定的距离就会取一个像素点放到输出图像上。假设原图的像素为  $W \times H$ ，同时长度和高度的缩小比例分别为  $a$  和  $b$ ，则采样间隔就为  $W/a$  和  $H/b$ 。也就是说，在水平方向上每个  $W/a$  取一个像素点，在竖直方向上，每隔  $H/b$  取一个像素点。等间隔采样法虽然原理简单，但是在图像裁剪的过程中，会发生明显的损坏，原图像中未被选中的像素信息会在缩小后的图像中丢失。基于此，需要对上述方法做出改进。

局部均值法不同于等间隔采样，是等间隔采样法的改进。这种方法在图像缩小的过程中，不再依赖于等间隔进行采样。在求缩小图像的像素时，不是简单的对原图像中的某一个像素点采样点像素，而是按照比例对原图像做一定的分割，通过合适的分块策略将原图像分割成不同的子块，对于每一个子块，取子块内所有像素点的平均值作为缩小图像对应的像素点。如此一来，在图像缩小过程中，那些丢失的像素点不是简单的被忽略掉，而是被块内所有数据的平均值代替了，这样可以大大改进图像在裁剪过程中的信息丢失。压缩前后如图 4-2 所示：

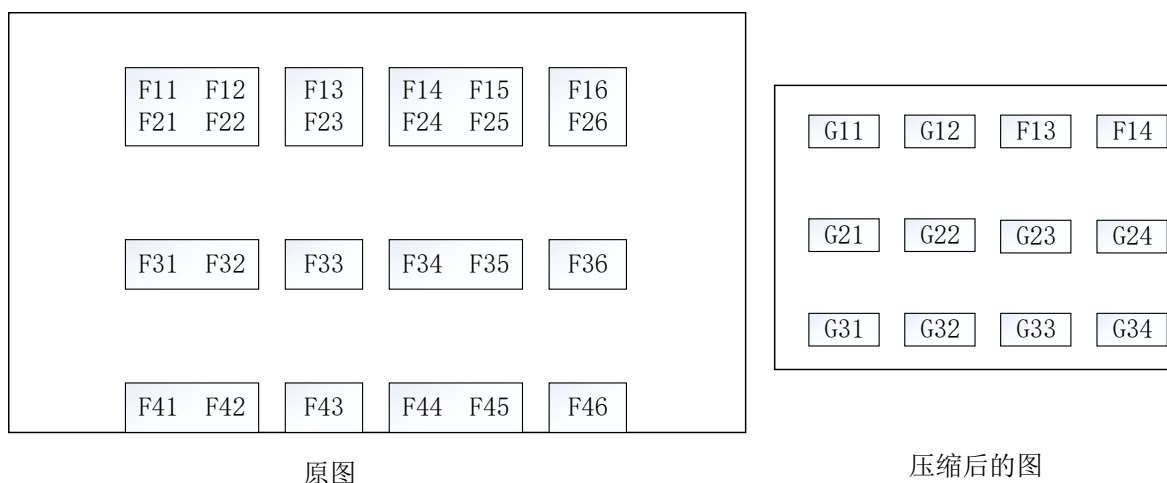


图 4-2 局部均值法

上图为局部均值法示意图，按照局部均值法，G11 是等效替代的 F11，F12，F21，F22 这一个块，所以有  $G11 = \frac{F11+F12+F21+F22}{4}$ ，同理 G12，G13，G14 也是类似的做法。总的来说，该算法的特点就是使用平均值来代替原来的像素点。经过这样的变化，一张高像素点的图，就能够有效裁剪为一张低像素点的图。局部均值法的伪代码如表 4-3 所示：

表 4-3 局部均值算法

局部均值算法
输入： Matrix img 原始图像 Point A 原始图像上的点坐标 Point B 缩小后图像上的点坐标 For i from A.x to B.x: Pix=img(i) For j from A.y to B.y: V[0] = V[0]+pix[j][0] V[1] = V [1]+pix[j][1] V[2] = V[2]+pix[j][2] count = (B.x-A.x+1)/(B.y-A.y+1) p[0] = V[0]/count p[1] = V[1]/count p[2] = V[2]/count return p

## 4.2 方案设计

上一节已经详细介绍了如何对卫星图像数据进行预处理。为了节约训练过程中的内存，同时也为了能够更好的将注意力集中在所关注的区域，分别对图像数据进行了格式的转化和裁剪。在本节中，将进一步探讨模型的设计。首先，会深入讨论全卷积网络，并比较他和普通卷积网络的不同，说明为什么 GAN 模型要使用全卷积网络，然后详细介绍本论文中用到的 GAN 模型架构，包括如何使用 GAN 生成台风轨迹图片，以及如何通过图片之间的轨迹去比较实际轨迹之间的误差。

### 4.2.1 全卷积网络

一般来说，对于深度学习的类的图像任务，都是会在 CNN 网络及其变形后面接上全连接层，利用卷积强大的边缘特征识别特性图像特征转化为一个张量。CNN 网络适用于图像级别的分类和回归任务，需要经过卷积层使用全连接层得到固定长度的特征向量，而在实际应用中很多问题仅仅使用图像级别的预测是远远不够的，需要使用像素级别的预测，这种场景就不再使用选择 CNN 网络了，而是选用全卷积网络<sup>[41]</sup>（Fully Convolutional Network, FCN）应对此类问题，与 CNN 网络不同，FCN 网络不提取图像的边缘特征，而是从抽象特征中恢复每个像素所属的类别，这可以进一步从图像分类扩展到像素分类。FCN 可以接受任意尺寸的输入图像，并通过卷积操作转换成一个热图，然后利用反卷积的方法对生成的热图进行上采样，恢复到和输入图像相同的像素。

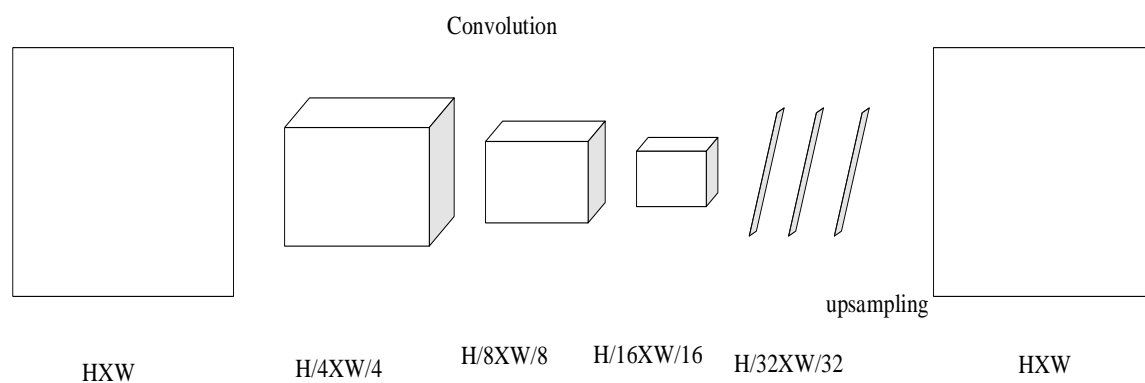


图 4-3 FCN 网络

可以说 FCN 网络和 CNN 网络最大的不同就在于把卷积之后全连接层换成了  $1 \times 1$  的卷积层，而这一改变却使得识别效果有了质的飞跃。CNN 只能通过识别的边缘特征来学习图像，如果要检测图像中的某些物体，或者进行图像的语义分割等问题却无能为力，但从热力图中提取台风轨迹的信息，却属于图像语义分割类的问

题,因此 FCN 的出现为这一切带来了可能。如图 4-3 所示,为 FCN 网络的模型架构图,不同于传统的 CNN 网络,FCN 网络将 CNN 网络最后的全连接层换成了  $1 \times 1$  的卷积层,用于提取特征,形成热点图。接着利用反卷积,将小尺寸的热点图上进行采样,得到原尺寸的语义分割图像。输入图像经过卷积和池化之后,得到的热点图的宽高相对原图缩小了数倍,如上图中变为原来的  $1/32$ 。为了得到与原图一样大小的输出,需要对其进行上采样。同时由于卷积网络具有对称性,卷积的导数的导数是卷积本身,基于卷积的这个性质,可以对其采用反卷积来恢复原来的图像。如此一来就可以从图像抽象的特征中恢复像素的信息,大大增加了网络的适用范围。而这一切都是 因为反卷积的灵活运用。反卷积示意图如图 4-4 所示:

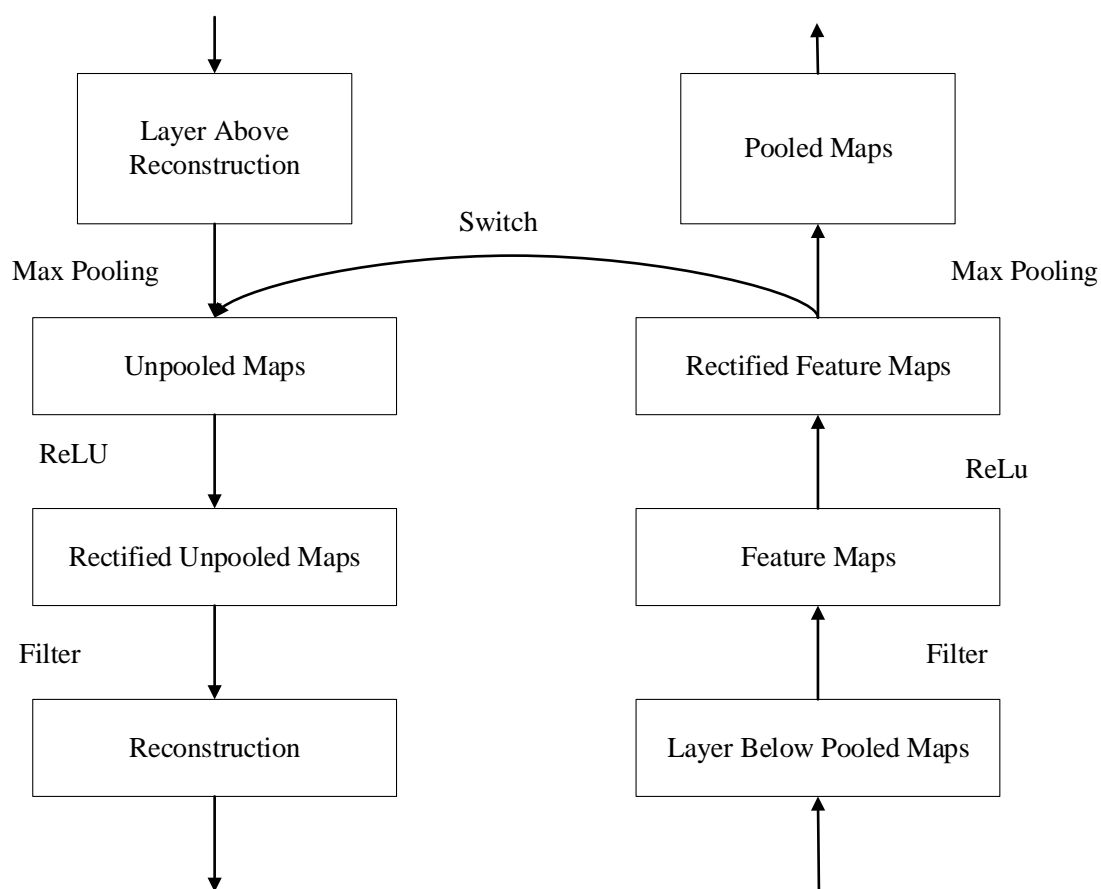


图 4-4 反卷积图

反卷积技术是上采样的核心,也正是因为反卷积技术的存在,使得图像可以进一步还原成原始的大小。反卷积过程其实就是对原图进行稀疏编码生成多个热力图,然后再对热力图进行稀疏编码生成下一层的热力图,以此内推。与普通的 CNN 中的方法相比,更好的保留了像素点之间的相关性,除了可以学习到边缘这种低级特征之外,还可以学习到角点,交叉等特征,对于图像的重建来说这些是非常好的



特征。由此可见，反卷积并不是卷积的逆过程，相反，反卷积是一种特殊的卷积，或者说是一种温和的上采样的方法。

反卷积技术的提出，为上采样过程提供了极大的方便，但是如果直接从最后的热力图上采样到图片大小，精度上会过于粗糙。这是由于深层网络在学习比较深度的特征的同时，也会丢失很多空间位置信息。因此需要将其和比较浅的网络结合起来。因此使用 Skips 结构将最后一层的预测和更浅层的预测结合起来，这样可以在遵守全局预测的同时进行局部预测。如此一来，能从图像中较好的分割出需要部分的语义信息。

## 4.2.2 模型架构设计

在前面章节中，介绍了图像数据的预处理，在尽可能不丢失图像信息的情况下，压缩了图片的大小，保证了数据可以以 batch 加载到 GPU 的显存中，极大加速了模型的训练。而在上一节中详细研究了 FCN 网络，这对于本节的研究内容非常重要。因为在本次课题中，需要从热力图中生成台风轨迹并识别出来验证生成轨迹与实际的偏差，如果采用传统的 CNN 网络架构，只能对图像做一些边缘特征的识别，效果就会不甚理想。因此在 GAN 中使用 FCN 网络模型能够极大程度提高模型识别的准确率。有了前文的铺垫，在本节中，将会详细介绍模型的架构设计。

先来看看普通卷积网络进行时间序列相关的图像预测存在的问题。假设  $Y = \{Y^1, \dots, Y^m\}$  为预测的轨迹图像序列， $X = \{X^1, \dots, X^n\}$  为输入的图像的图像序列。同时定义  $X$  经过卷积网络得到的结果为  $G(X)$ ，则生成图像与真实图像之间的误差如式 (4-1) 所示：

$$L_p(X, Y) = \ell_p(G(X), Y) = \|G(X) - Y\|_p^p \quad (4-1)$$

一方面来说，卷积受到卷积核尺寸的限制，只能产生短程依赖。另一方面，对于图像之间距离的衡量，一般采用  $\ell_2$  很少采用  $\ell_1$ ，但是这样会产生模糊的预测结果，而且随着时间的推移，这种模糊会变得越来越严重。而采用  $\ell_1$  虽然会有所缓解，但是生成图像依然是下一状态所有各种可能情况的平均。基于这种情况，引入 GAN 模型来进行台风轨迹预测。假设对于输入序列  $X = \{X^1, \dots, X^m\}$  存在两种输出序列  $Y = \{Y^1, \dots, Y^n\}$  和  $Y' = \{Y'^1, \dots, Y'^n\}$ ，如果采用传统的卷积的话，得到的预测结果是  $Y_{ave} = (Y + Y')/2$ ，而预测结果为  $(X, Y_{ave})$ ，这显然不合理。但是在 GAN 中，除了生成网络以外，还引入了判别网络，可以对不合理的结果进行筛选。只有当的生成结果能“欺骗”过判别网络，才能说明当前预测状态是有效的，因此结果就会更加的准确，可信。基于此，构造了如下的 GAN 模型架构，来预测台风轨迹。模型架构如表 4-4 所示：

表 4-4 模型网络架构

生成器 model				
Generate	G1	G2	G3	G4
Feature maps	128,256,128	128,256,128	128,256,512,256,128	128,256,512,256,128
Kernel size	3,3,3,3	5,3,3,5	5,3,3,3,5	7,5,5,5,5,7
判别器 model				
Discriminate	D1	D2	D3	D4
Feature maps	64	64,128,128	128,256,256	128,256,512,128
Kernel size	3	3,3,3	5,5,5	7,7,5,5
Fully Connected				
	512,256	1024,512	1024,512	1024,512

基于时序的 GAN 的网络模型架构，由生成器模型，判别器模型和全连接层 3 个部分构成。为了预测未来台风轨迹的序列，需要依赖过去的状态，因此需要将当前轨迹的前序状态按照时间进行排序，然后一次输入网络模型。深度学习的生成器模型由 4 个 FCN 网络模型组成，G1, G2, G3, G4，并且采用多尺度架构，每个 FCN 的大小各不相同，每一个全卷积网络  $G_k \{k = 1, 2, 3, 4\}$  都会生成对台风轨迹的预测与真实的台风轨迹进行对比。同样，判别网络也是由 4 个 FCN 网络构成，这些全卷积网络 D1, D2, D3, D4 也采用多尺度架构，每个 FCN 的大小同样各不相同。同时每一个都可以对前面生成的不同尺度的图像进行判别。判别网络模型的最后加上全连接层，将含有预测的图像分类为 0，将真实的图像分类为 1。对于表 4-4 中给出的生成模型，使用了 ReLU 函数作为激活函数，消除了“死区”，最后接上 tanh 函数，保证输出在  $[-1, 1]$  之间，然后设置学习率从 0.04 开始到 0.005，每次训练生成一张图片把他前序状态排序一并输入，设置 batch 大小为 6。而对于判别器，使用 ReLU 作为激活函数，判别网络的学习率设置为 0.02。

基于时序的 GAN 模型架构如图 4-5 所示：

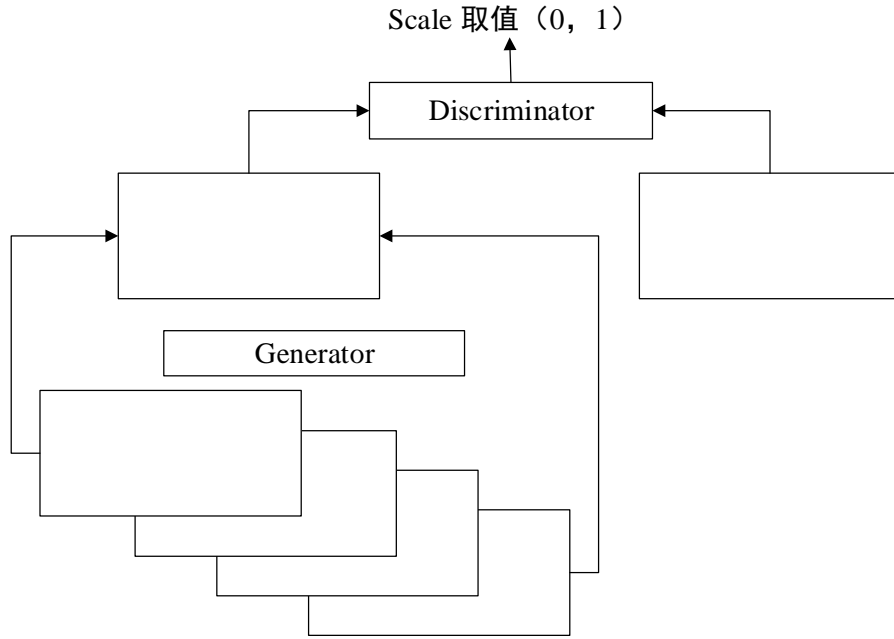


图 4-5 时序 GAN 模型图

在 GAN 中生成器和判别器的交替训练。生成器一定要做的就是使得生成的台风轨迹图像尽可能的逼真，最大程度的欺骗过判别器。如果只采用对抗损失函数可能会带来训练的不稳定性，所以在训练的时候还需要加入原先的损失函数，来权衡对抗损失项带来的清晰度预测和原先损失项带来的真实性预测。而判别器就需要尽可能去识别生成器生成的图片，不断找出他跟原图的不同，最后生成器和判别器二者进行不断的博弈，直至达到“纳什均衡”。

训练生成器的时候将判别器的权重固定住，采用 SGD 算法对生成器网络中的权重进行更新，其损失函数如式（4-2）所示：

$$L_{adv}^G(X, Y) = \sum_{k=1}^{N_{scales}} L_{bce}(D_k(X_k, G(X_k)), 1) \quad (4-2)$$

同理，在训练判别器的时候也需要将生成器的权重固定住，采用 SGD 算法对判别器网络中的权重进行更新，其损失函数如式（4-3）所示：

$$L_{adv}^D(X, Y) = \sum_{k=1}^{N_{scales}} L_{bce}(D_k(X_k, Y_k), 1) + L_{bce}(D_k(X_k, G_k(X)), 0) \quad (4-3)$$

$L_{bce}$ 指的的二进制交叉熵损失函数。其形式如式（4-4）所示：

$$L_{bce}(Y, \hat{Y}) = - \sum_i \hat{Y}_i \log(Y_i) + (1 - \hat{Y}_i) \log(1 - Y_i) \quad (4-4)$$

GAN 提供一种通过以无监督的方式提取台风的重要特征的方法，这保证判别

器网络模型不会被生成器模型迷惑。对于每一个卫星图像，将收到过去连续  $m$  个时刻的图像作为输入，并最终生成一个含有台风轨迹的热力图。其伪代码如表 4-5 所示：

表 4-5 时序 GAN 网络训练

时序 GAN 网络训练
初始化： Learning Rate $\rho_D$ 和 $\rho_G$ Weights $\lambda_{adv}$ 和 $\lambda_{lp}$ While not converged do: Update the 判别器 D: Get M data samples $(X, Y) = (X^1, Y^1) \dots (X^m, Y^m)$ Then $W_D = W_D - \rho_D \sum_{i=1}^M \frac{\alpha \mathcal{L}_{adv}^D(X^i, Y^i)}{aW_D}$ Update the 生成器 G: Get M new data samples $(X, Y) = (X^1, Y^1) \dots (X^m, Y^m)$ Then $W_G = W_G - \rho_G \sum_{i=1}^M (\lambda_{adv} \frac{\alpha \mathcal{L}_{adv}^G(X^i, Y^i)}{aW_G} + \lambda_{lp} \frac{\alpha \mathcal{L}_{lp}^G(X^i, Y^i)}{aW_G})$ End while

GAN 的训练不同于深度学习中的其他模型的训练，GAN 是由生成器和判别器共同构成的，所以这也就要求在训练的过程中需要对生成器和判别器分开训练，也就是说，在训练的过程中一般会产生 2 个不同的 Loss，这和其他深度学习模型的训练方法截然不同。在这里先训练判别器模型，判别器接受来自之前状态的卫星轨迹图像序列，并进行训练，以预测由生成器生成的图像的可能性。如此一来，只有最后一个状态是来自生成器生成的，而其余状态全是前面基于时间序列的状态，这样，对抗生成网络就可以充分利用时间信息，方便生成器学习并产生与其输入时间上一致的序列。对 GAN 模型引入了时序，有效的改善了卷积网络对于时间序列的图像数据预测的效果，也为进一步提高轨迹的预测精度提供了帮助。

### 4.2.3 多模型融合

如前文所述，分别采用了 2 种完全不同的方法对台风轨迹进行了预测。对于序列数据，使用了基于注意力机制的 Seq2seq 模型；而对于图像数据，使用了基于时序的 GAN 模型。但是台风轨迹在登陆过程中，除了可以将其当然时间序列模型学习轨迹的运动趋势外，同时还受许多诸如中心气压、中心风速等其他因素影响，而前面研究也表明，这些模型之间并没有什么关联性，而是独立作用于台风轨迹，

因此这些因素对台风轨迹的影响不可忽略；同时如果采用单一数据源训练模型也会存在一定的瓶颈，序列数据不够直观，同时在长期依赖过程中，真正能影响到最终结果的前序状态不好学习；而图像数据虽然比较直观，但是对于除经纬度以外的其他因素并不能较好的运用，决定轨迹走势的因素过去单一。基于此，希望能将预测结果进行模型融合，进一步提高预测的精度。

在深度学习中，对多个模型进行融合，来提升单模型的预测效果，是一种比较常用的方法，不同的子模型在不同的数据集上都有其擅长的部分，结合他们“擅长”的部分，可以得到一个在各个方面都很“擅长”的模型。常规来说，模型的融合方法主要分为以下 2 种：

#### (1) Bagging<sup>[42]</sup>

这种方法实质就是模型的再采样，即在每个样本上训练出来的模型取平均，从而降低模型的方差。首先从原始样本集中抽取训练数据集，每轮从原始样本集中使用自荐的方法从中抽取  $n$  个训练样本。具体做法是在含有  $n$  个样本的数据集中，每次随机挑选一个样本，将其作为训练样本，再将此样本放回到数据集中，这样有放回地抽样  $n$  次，生成一个与原数据集大小相同的数据集，这个新数据集就是训练集。如此共进行  $k$  轮抽取，共生成  $k$  个训练集。接着对于上面生成的  $K$  个训练集，每次使用一个训练集，使用  $k$  种不同的算法，得到  $k$  个不同的模型。因为进行的是回归问题，最后对  $k$  个模型预测的结果进行平均。

#### (2) Boosting<sup>[43]</sup>

这种方法是一种迭代的方法，迭代算法一般是根据上一轮预测的结果，对样本进行加权，减少误差。首先在初始化的时候，会给每一个数据集赋予一个权重  $1/n$ ，在未进行学习之前，假设所有的权重都是一样的。接着会对数据集进行  $k$  轮训练，在每次训练中，对准确率比较低的部分赋予更高的权重，而对于准确率比较高的部分，赋予很少的权重。最后因为是回归问题，对所有的结果进行加权平均。

综合比较上述两种方法，Bagging 要求使用在同一个数据集内进行采样，而且对结果使用简单的平均。但是通过实验结果可以发现，基于注意力机制的 Seq2seq 对于台风轨迹预测结果与预期结果在 80km 以内的百分比更大，因为在预测过程中，序列数据所使用的特征也更多；而基于时序的 GAN 模型，更适用于用来对结果进行修正，而不是直接做简单的平均。所以更适用于使用 Boosting 的方法。权重的分配问题上，采用一种叫做 GASEN<sup>[44]</sup>的遗传算法对权重进行进化，最终得到权值化的神经网络。算法的伪代码如表 4-6 所示：

表 4-6 GASEN 算法

GASEN 算法
输入: training set $S$ learner $L$ trials $T$ threshold $\lambda$ for $t = 1$ to $T$ : set $S_t = \text{bootstrap samples from } S$ set $N_t = L(S_t)$ generate a population of weight Vector evolve the population where the fitness of a weight Vector $w$ is measured as $f(w) = 1/E_w^V$ $w^* = \text{the evolved best weight vector}$ 输出 the regression problem: $N^*(X) = \text{Ave } \sum_{w^*, t > \lambda} N_t(X)$

假设存在许多不同的分类器，需要合适的策略去选择每种分类器的结果所占的权重，如果平均去选取，通常得到的结果不尽理想；如果暴力去解决，这是一个 NPC 问题，在多项式时间内无法解决，时间复杂度太高；如果贪心去解决，常常不能得到最优解。这时候使用遗传算法去进化权重是比较好的一种选择。GASEN 就是为解决这类问题而提出的一种算法。GASEN 一种启发式的方法，初始时，假设为每一个神经网络分配一个权重，表示这个网络在整个模型当中的适合程度。然后使用遗传算法来进化这些权重，以便它们在某种程度上可以表征神经网络加入整体的适应性，最后选取阈值大于  $\lambda$  加入集合。GASEN 算法除了能够使用遗传算法为模型分配权重以外，还具有较好的泛化能力，能很好的提升融合后模型的精度。因而在本次课题中，选择 GASEN 算法来进行模型的融合。

### 4.3 实验结果

在本节中，会就前面本章研究内容进行实验结果的展示，并对实验结果做出相应的分析。首先，简单叙述一下本次实验所用到的实验环境。接着会展示基于时序 GAN 模型的台风轨迹预测，在本部分中，首先会展示训练过程中生成器和判别器的 Loss，保证训练过程的正确性；接着会展示 GAN 生成的台风轨迹图像，并与传统 CNN 网络生成的图像进行对比，通过将图像数据之间的差异转化为真实坐标的

差异，并通过一些误差衡量指标，说明使用的方法的比较传统的 CNN 在生成图像方面精度有较大的提升；最后，将对第三章中的模型和本章中的模型进行融合，并于单模型预测进行对比，说明使用的模型算法是切实有效的，并分析预测过程中误差存在的原因。

### 4.3.1 台风轨迹预测

在前文中，详细论述了如何基于时序 GAN 模型进行台风轨迹预测，本节需要对该方法进行实验，证明前文论述是正确的。因为在 GAN 模型训练中有大量的矩阵运算和浮点运算，可以用 GPU 进行加速，整个实验在云服务器上进行，实验的硬件参数如下所示：

表 4-7 实验的硬件参数

内存	CPU	GPU	显存
16GB	Intel core i7-8700	NVIDIA RTX 2070	8GB

因为 GAN 训练的过程是生成器和判别器分开训练，然后交替进行，因此在训练过程中会出现两个 Loss 函数，一个是生成器的，一个是判别器的。对于本实验中的 GAN 模型，其训练过程中的 Loss 函数如图 4-6 所示：

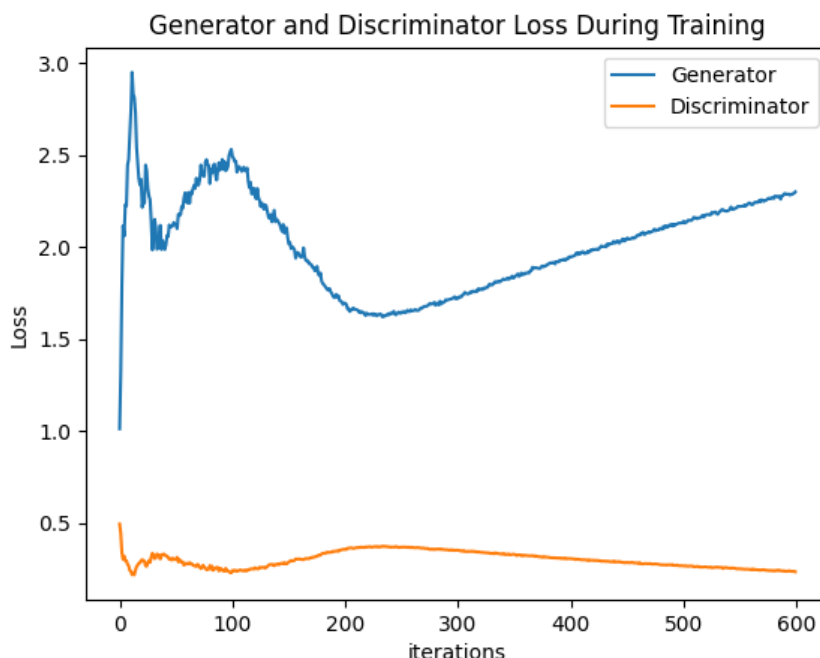


图 4-6 GAN 训练 Loss 函数图

图中所示为 GAN 训练过程的 Loss 函数图，蓝色部分为生成器，橙色部分为判

别器。生成器的 Loss 函数先增大再减小，判别器的 Loss 函数下降比较缓慢。这是由于生成器开始时生成的图片精度较低，判别器很容易分辨出来。随着训练过程的继续，生成器生成的图片精度越来越高，判别器就越来越难判别出来。在训练过程中会出现震荡，引入 BN 技术，防止生成器把所有样本都收敛到同一点。随着训练过程的进行，生成器生成的图像逐渐可以欺骗判别器，直至到达最终的“纳什均衡”。从实验结果也可以看出，前文所述方法是行之有效的。基于时序 GAN 模型，生成如图 4-7 所示的台风轨迹图：

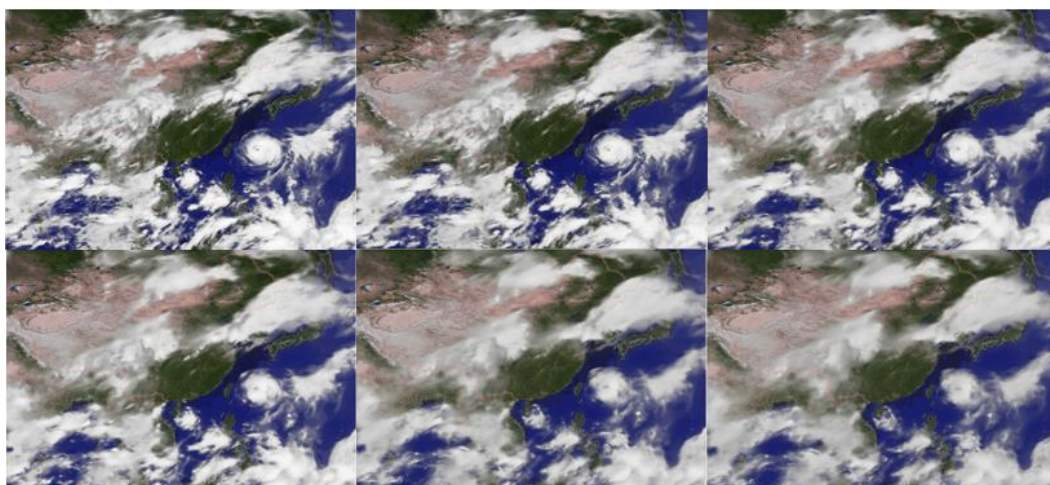


图 4-7 GAN 生成台风轨迹图

图中所示为时序 GAN 模型生成的连续 6 个时间点的台风轨迹云模型，按照从上到下，从左到右的方式进行排列。对于 GAN 模型的评估通常采用 Inception Score，能很好的反应生成图像的质量和多样性。但此处由于需跟前文的时序数据进行多模型融合，要将图像的预测结果转化为序列坐标形式，进行评估。通过式（4-5）计算预测误差：

$$E = 2R \arcsin \sqrt{\sin^2\left(\frac{x_{pred} - x_{real}}{2}\right) + \cos x_{pred} \cos x_{real} \sin^2\left(\frac{y_{pred} - y_{real}}{2}\right)} \quad (4-5)$$

有上述转化与衡量之后，将对基于 CNN 网络的预测结果和基于时序 GAN 模型预测结果进行对比。同第三章中实验相同，分别选取误差不超过 80km，误差处于 80-120km，误差大于 120km 三个维度进行度量，实验结果如图 4-8 所示：



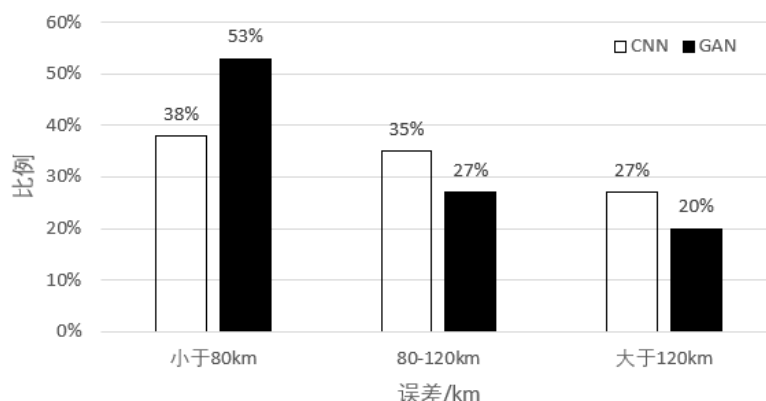


图 4-8 预测结果对比图

如上图所示，分别使用了基于时序的 CNN 模型和基于时序的 GAN 模型进行了台风轨迹预测。左边为 CNN 模型的预测结果，右边为 GAN 模型的预测结果。可以看出，基于 CNN 模型在 80km 以内的预测点，大概占 38%，在 80-120km 的，大概占 35%，而大于 120km 的大概占 27%。而基于时序的 GAN 模型在 80km 以内的占 53%，80-120km 占 27%，120km 以上的占 20%。从实验结果可以看出，在小于 80km 这个范围内，基于 GAN 模型的预测精度远高于 CNN 模型，这也进一步说明，基于时序 GAN 模型对台风轨迹预测效果远好于 CNN 模型，这与前文的结论完全吻合，说明前文的方法是行之有效的。

接下来将会针对基于时序的 GAN 模型，进一步探究输入图片的数量对结果的影响，实验结果如图 4-9 所示：

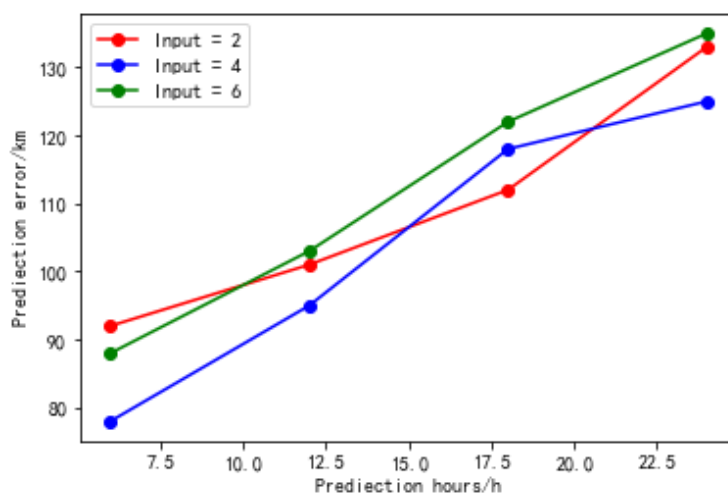


图 4-9 输入图片数量和预测精度关系图

如上图所示，分别设置输入图片数量为 2，4，6 同时对 6h，12h，18h，24h 这 4 个时间段进行预测，计算平均预测距离，红色曲线为依赖前序 2 个状态作为

输入，蓝色曲线为依赖前序 4 个状态作为输入，绿色状态为依赖前序 6 个状态作为输入。从图中可以看出，依赖前序 4 个状态作为输入建立时序 GAN 模型平均预测精度最高，因为在依赖 2 个状态时，所需要的信息尚不够完全，存在一些信息的丢失；而利用 6 个状态，对于前序依赖的信息过多，有些跟生成结果不想关的状态也被用于生成图像，导致结果的不准确。

上述实验，验证了基于时序 GAN 模型对于台风轨迹的预测的可行性和准确性。但是综合第三章的结果可以发现，基于注意力机制的 Seq2seq 模型预测精度高于 GAN 模型，这是由于在预测过程中时序数据可以更好的学习数据的走势，同时可以更好的利用除经纬度以外的其他特征，而 GAN 的预测只能学习数据的分布，对于除经纬度以外的其他特征其实是很难利用，所以使用图像数据进行预测的精度远不如序列数据。但是图像数据更为直观，同时可以更好的学习到轨迹数据的形状，这都是序列数据无法做到的。因此如果将本章的预测结果用于对时序预测精度的修正，与第三章的预测结果进行多模型的融合，能更好的提升台风轨迹的预测精度。

### 4.3.2 多模型融合

如前文所述，不同算法在不同的数据集上表现不同，对于台风轨迹预测问题也是一样。有的算法虽然在平均看来表现不是最好的，但却对于某些情况有更高的预测精度。基于此，本节将对第三章和第四章中的预测结果进行模型融合，博采众长，进一步提高预测精度。下面将对前文提到的三种模型融合方法 Bagging, Boosting 和 GASEN 融合效果进行对比，分别用三种方法对序列数据和图像数据预测结果进行多模型融合，并在取 6h, 12h, 18h 和 24h 这 4 这个时间点进行预测，计算预测的平均误差，结果如图 4-10 所示：

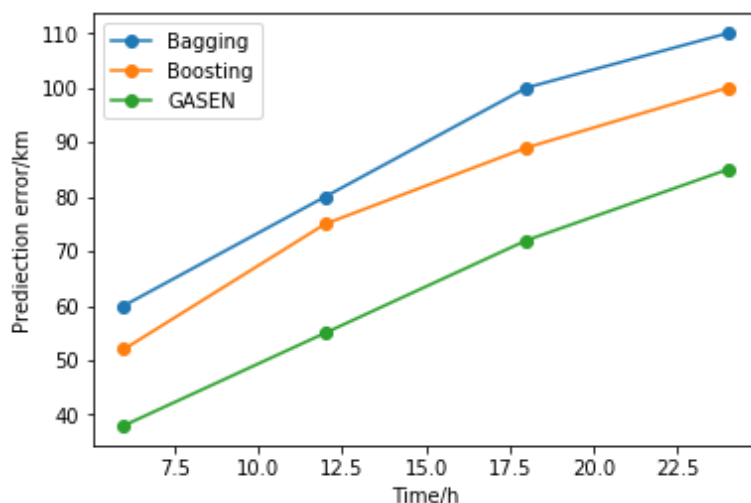


图 4-10 模型融合对比图

从图中明显可以看出, 基于 GASEN 的遗传算法预测结果远好于 Bagging 和朴素的 Boosting, 这是由于无论是 Bagging 还是 Boosting, 权重都只能通过取平均值或者是人手动取设置一个比例, 这样的预测的结果其实会偏向某个模型, 无法做到很好的集成性。而 GASEN 的权重是通过遗传算法进化出来的, 在每次预测的时候, 都会对于当前的情况, 选择最好的权重比例, 因此会有更好的效果。实验结果也证明了, 通过 GASEN 遗传算法进行模型融合后, 能够更好的同时利用图像数据和序列数据的优势, 提升预测精度, 这和前文结论保持一致。

在比较了三种模型融合方法之后, 需要对融合后的效果进一步做测试, 下面分别对 Seq2seq 模型, GAN 模型以及融合后的模型的预测结果进行测试, 取误差不超过 80km, 80-120km, 大于 120km 三个区间, 在测试集进行测试, 并误差范围进行统计, 实验结果如图 4-11 所示:

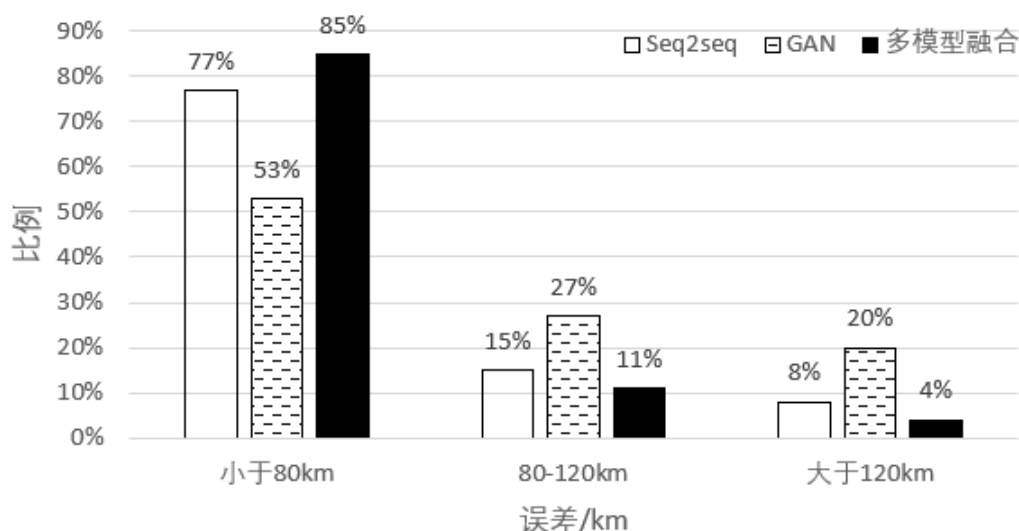


图 4-11 预测结果对比图

从图中可以看出, 进行模型融合之后, 误差小于 80km 的比例明显较单模型有较大的上升, 误差处于 80-120km 和大于 120km 的比例开始下降, 说明使用基于 GASEN 的遗传算法进行融合后比单模型预测精度有较大提升。这是由于单模型在预测过程中总会有些无法兼顾的地方。Seq2seq 模型能基于时间序列, 利用所有的特征, 学习轨迹的走向, 但是对于一些突发情况, 比如轨迹突然变向之类的不具有很好的泛化能力, 导致无法处理这类情况。而基于时序的 GAN 模型虽然能生成数据的分布, 学习轨迹图像的形狀, 却对于很多影响轨迹走向的特征无能为力。因此需要结合二者的优点进行模型预测, 进一步提高预测精度。上述实验也进一步证明了我们的方法是行之有效的。

## 4.4 本章小结

为了更好地利用训练集中的台风轨迹卫星图像数据，本章提出了一种基于时序的 GAN 模型来预测台风轨迹数据。首先，需要对图像数据进行预处理，使用数字编码算法将数据转化成 png，然后采用局部均值法对图像进行裁剪，减少在更可能减少图像信息的丢失的情况下裁剪图像。接下来，详细论述了方案的设计，首先引入了 FCN 网络，作为后面建模的基础；接着介绍了本课题模型使用的基于时序的 GAN 模型架构和网络参数的配置；然后对本章的实验结果和前一章的实验结果，采用 GASEN 的遗传算法进行了多模型融合，进一步提升了预测的精度。最后对前面研究内容进行实验。首先，对训练过程可视化，展示训练中的 Loss 函数和基于时序的 GAN 模型生成台风轨迹的结果；接着与传统的 CNN 网络预测结果进行对比，证明模型比较传统 CNN 网络预测结果更优；然后探究影响预测精度的因素；最后结合第三章的预测结果，进行多模型融合，验证融合后预测精度远高于单模型，并对实验结果加以说明。

## 第五章 总结与展望

### 5.1 全文总结

本文主要基于深度学习的方法，通过已有的历史数据对台风轨迹预测进行了研究。台风轨迹数据集主要分为时序数据和图像数据。对于序列类型的台风轨迹数据，首先提出了改进的卡尔曼算法对轨迹数据进行滤波，剔除野值；接着通过最小扇形简化算法简化台风轨迹；最后通过编码器和解码器的方法，建立 Seq2seq 模型对台风轨迹进行预测，并在模型中加入注意力机制，进一步提高预测精度。而对于卫星图像数据，提出一种基于时序的 GAN 模型，可以对带有时间序列的图像数据进行建模，并生成图像，相比较使用卷积网络，预测精度有了较大的提升。最后，使用 GASEN 遗传算法，对前文提出的两种算法进行多模型融合，博采众长，提升预测效果。本文的主要贡献如下：

（1）对于台风轨迹时序数据，本文使用基于注意力机制的 Seq2seq 模型用于台风轨迹预测，极大程度上提升了轨迹预测的精度。首先提出改进卡尔曼滤波算法保证了轨迹数据的质量，然后把问题转化为基于时间序列的预测进行建模。相比较于传统的机器学习方法，能自动提取轨迹数据特征，而不需要相关领域专家的去专门做特征工程；同时相比于 LSTM 网络，能处理输入与输出不等长的问题，同时注意力机制的引入能更好利用轨迹数据的前序状态。最终通过实验验证，本文所建立的模型用于台风轨迹预测精度比当下主流方法有较大的提高。

（2）对于台风轨迹卫星图像，本文提出基于时序的 GAN 模型用于台风轨迹预测，提升了预测的精度。一般来说，图像预测类的问题一般采用传统的卷积神经网络或者使用加入 LSTM 网络的时空卷积网络。但是卷积网络对于图像的预测仅仅是图像各种可能情况的平均，可能会生成一种“四不像”的物体。考虑到 GAN 过去在图像生成领域取得了巨大的成功，将生成器和判别器的模式来生成台风轨迹数据预测。在基础 GAN 模型的基础上加入了时序性，用于对台风轨迹图像的预测。经实验验证，本文建立的模型对于图像数据预测精度远高于当下主流的方法。

（3）最后通过遗传算法，创新性地对序列数据预测结果和图像数据预测结果进行多模型融合，进一步提升预测精度。因为对台风轨迹的属于回归类问题，需要对结果按照权重进行加权平均。基于此使用了一种 GASEN 的遗传算法，对权重进行进化，得出最终的结果，经实验验证，融合后当下所有算法预测效果更好。

## 5.2 后续工作展望

台风轨迹预测问题是当下研究的热点和难点,根据目前国内外的研究情况,仍然有许多可以继续开展的工作,在这里列举一二。

首先当今 AI 技术仍然面临两个挑战,一个是在大多数行业中存在数据孤岛,一个是数据的安全性和隐私性很难得到保障。近年来,联邦学习<sup>[45]</sup>技术的发展使得在保证数据隐私安全及合法合的基础上,实现共同建模,提升 AI 模型的效果变成了可能。横向联邦学习技术可实现样本的联合,而纵向联邦学习技术可实现特征的联合。在保证数据安全性的条件下,实现共同建模,共同提升模型效果。将联邦学习技术应用与台风轨迹预测,可在保证数据安全性和隐私性的条件下,最大限度的利用世界各地的台风轨迹数据联合建模,增加训练数据量,增加特征维度,提升台风轨迹预测的精度。

其次是在深度学习中,数据预处理与模型优化等问题依然需要具备专业知识的数据科学家来完成,而且耗时耗力。而近年来随着 Automl<sup>[46]</sup>技术的发展,在特征工程、模型构建、超参优化三个方面实现了自动化,并且提供了端到端的解决方案。将 Automl 技术应用于台风轨迹预测,可以节省科研人员的精力,降低实验成本和门槛。

最后,近年来随着半监督学习<sup>[47]</sup>和无监督学习<sup>[48]</sup>的发展,诸如 AlphaGo2.0 等新产品展现了并不亚于监督学习的效果。考虑到优质数据集打标过程艰难,耗时耗力,如果,能在不影响预测精度的情况下,采用半监督,甚至是无监督学习的方法进行台风轨迹的预测,可使得这些技术具有更好的普适性,同时能够更好的减低沿海地区人民的生命财产损失,更好的造福于人类。

## 致 谢

时光飞逝，转眼间，三年研究生生活即将结束。再回首，往事依旧，三年来的点点滴滴依然历历在目，清晰可见。行文至此，心中思绪万千，除了不舍，更多的是感谢。

首先，我要感谢我的研究生导师侯孟书教授。侯老师具有渊博的知识，严谨的治学态度，更重要的是可以站在更高角度剖析科学问题，认清问题的本质。在学术上，侯老师不仅仅为我指明方向，给我提供了很多宝贵的意见；更注重培养我刻苦自律的性格和勤于思考的习惯，深深影响着我的学习生活方式。生活中，侯老师常给我分享许多宝贵的人生经验，在黑暗中照亮我前行的道路，也让我的心智从幼稚走向成熟，努力去做一个真正对社会有用的人。能在人生最好的年纪，遇到侯老师并成为他的学生，是我的荣幸，也是我一生的幸运。

其次，我要感谢同教研室的詹思瑜副教授、廖建明教授和董浩老师在学习和生活中给我的帮助，你们严于律己的科研精神和一丝不苟的学术态度，是我一生学习的榜样。同时，我还要感谢我的女朋友曹婷婷同学多年来给予我的支持与理解，让我能更有勇气面对困难，勇往直前。

最后特别感谢我的父母，是你们不辞辛苦，为我创造良好的条件，让我可以无忧无虑安心科研学习。在我面对失败时，也是你们用切身经历给予我信心，让我相信事在人为，只要努力，就会有回报。感谢你们无私的付出！

向本论文的所有评阅老师们致以最真诚的祝福与衷心的感谢！

## 参考文献

- [1] Huang X Y, Jin L. An artificial intelligence prediction model based on principal component analysis for typhoon tracks[J]. Chinese J Atmospheric Sci, 2013, 37(5): 1154-1164.
- [2] Hassan H, Aue A, Chen C, et al. Achieving human parity on automatic chinese to english news translation[J]. arXiv preprint arXiv:1803.05567, 2018.
- [3] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]. International Conference on Neural Information Processing Systems. 2012.
- [4] Urtasun R, Lenz P, Geiger A. Are we ready for autonomous driving? The KITTI vision benchmark suite[C]. IEEE Conference on Computer Vision & Pattern Recognition. 2012.
- [5] Minar M R, Naher J. Recent Advances in Deep Learning: An Overview[J]. 2018.
- [6] Abadi M, Barham P, Chen J, et al. TensorFlow: a system for large-scale machine learning[J]. 2016.
- [7] Ketkar N. Introduction to pytorch[M]. Deep learning with python. Apress, Berkeley, CA, 2017: 195-208.
- [8] 贾澎涛, 何华灿, 刘丽, 等. 时间序列数据挖掘综述[J]. 计算机应用研究, 2007, 24(11): 15-18.
- [9] Zeiler M D, Fergus R. Visualizing and Understanding Convolutional Networks[J]. 2013.
- [10] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. IEEE conference on computer vision and pattern recognition. 2016: 770-778.
- [11] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]. Advances in neural information processing systems. 2015: 91-99.
- [12] Jeffrey L. Elman. Finding structure in time[J]. Cognitive Science, 14(2):179-211.
- [13] Staudemeyer R C, Morris E R. Understanding LSTM--a tutorial into Long Short-Term Memory Recurrent Neural Networks[J]. arXiv preprint arXiv:1909.09586, 2019.
- [14] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv:1406.1078, 2014.
- [15] Mirza M, Osindero S. Conditional generative adversarial nets[J]. arXiv preprint arXiv:1411.1784, 2014.
- [16] Mnih V, Heess N, Graves A. Recurrent models of visual attention[C]. 2014: 2204-2212.
- [17] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[C]. 2014: 3104-3112.



- [18] LeCun Y, Boser B, Denker J S, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4): 541-551.
- [19] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]. 2012: 1097-1105.
- [20] Rastegari M, Ordonez V, Redmon J, et al. Xnor-net: Imagenet classification using binary convolutional neural networks[C]. European conference on computer vision. Springer, Cham, 2016: 525-542.
- [21] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks[C]. The fourteenth international conference on artificial intelligence and statistics. 2011: 315-323.
- [22] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. nature, 2016, 529(7587): 484.
- [23] Cortes C, Vapnik V. Support vector machine[J]. Machine learning, 1995, 20(3): 273-297.
- [24] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting[C]. European conference on computational learning theory. Springer, Berlin, Heidelberg, 1995: 23-37.
- [25] Breiman L. Random forests leo breiman and adele cutler[J]. Random Forests-Classification Description, 2015.
- [26] Daosheng X, Zitong C. The influence of an improved cumulus parameterization scheme on typhoon forecast from GRAPES model[J]. Journal of Tropical Meteorology (in Chinese), 2014, 30(2): 210-218.
- [27] Jeon M, Venkataraman S, Qian J, et al. Multi-tenant GPU clusters for deep learning workloads: Analysis and implications[J]. Tech. Rep., 2018.
- [28] Gardner M W, Dorling S R. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences[J]. Atmospheric environment, 1998, 32(14-15): 2627-2636.
- [29] Chen X S Z, Yeung H W D Y, Woo W K W W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting[J]. arXiv preprint arXiv:1506.04214, 2015.
- [30] Chen K, Huo Q. Training deep bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2016, 24(7): 1185-1193.
- [31] Baltrušaitis T, Ahuja C, Morency L P. Multimodal machine learning: A survey and taxonomy[J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 41(2): 423-443.

- [32] Kingma D P, Welling M. Auto-encoding variational bayes[J]. arXiv preprint arXiv:1312.6114, 2013.
- [33] Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity[J]. IEEE transactions on image processing, 2004, 13(4): 600-612.
- [34] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[J]. arXiv preprint arXiv:1511.06434, 2015.
- [35] 张旭明, 徐滨士, 董世运. 用于图像处理的自适应中值滤波[D]. , 2005.
- [36] Kalman R E. A new approach to linear filtering and prediction problems[J]. 1960.
- [37] Saalfeld A. Topologically consistent line simplification with the Douglas-Peucker algorithm[J]. Cartography and Geographic Information Science, 1999, 26(1): 7-18.
- [38] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.
- [39] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System ACM SIGKDD International Conference on Knowledge Discovery and Data Mining[J]. ACM, 2016: 785-794.
- [40] 谢林, 虞露, 仇佩亮. 基于上下文的自适应二进制算术编码研究[D]. 2005.
- [41] Shelhamer E, Long J, Darrell T. Fully convolutional networks for semantic segmentation[J]. IEEE Annals of the History of Computing, 2017 (04): 640-651.
- [42] Wang B, Pineau J. Online bagging and boosting for imbalanced data streams[J]. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(12): 3353-3366.
- [43] Friedman J H. Stochastic gradient boosting[J]. Computational statistics & data analysis, 2002, 38(4): 367-378.
- [44] Wu Z, Chen Y. Genetic algorithm based selective neural network ensemble[C]. IJCAI-01: proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, Seattle, Washington. 2001.
- [45] Yang Q, Liu Y, Chen T, et al. Federated machine learning: Concept and applications[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2019, 10(2): 1-19.
- [46] Kaul A, Maheshwary S, Pudi V. Autolearn—Automated feature generation and selection[C]. 2017 IEEE International Conference on data mining (ICDM). IEEE, 2017: 217-226.
- [47] Li J, Monroe W, Ritter A, et al. Deep reinforcement learning for dialogue generation[J]. arXiv preprint arXiv:1606.01541, 2016.
- [48] Metz L, Maheswaranathan N, Cheung B, et al. Meta-learning update rules for unsupervised representation learning[J]. arXiv preprint arXiv:1804.00222, 2018.

## 攻读硕士学位期间取得的成果

- [1] 李明, 侯孟书, 詹思瑜, 董浩, 王瀚, 席瑞, 董林森. 一种基于数据并行策略的分布式深度学习方法及系统[P].中国, 发明专利, 201810662859.7, 2018 年 6 月 25 日