中文图书分类号：TP***

密　　级：公开

UDC：*****

学　校　代　码：10005

工程硕士学位论文

M.E. DISSERTATION

论　文　题　目：基于神经网络的空气质量预测

论　文　作　者：闫硕

领　　域：软件工程

指　导　教　师：刘博

论文 提交 日期: 2019 年 5 月

UDC： ****　　学校代码：10005

中文图书分类号：TP ***　　学　　号：G20*****

密　　级：公开

北京工业大学硕士专业学位论文

（全日制）

题　　目：基于神经网络的空气质量预测

英文题目：Air Quality Prediction Based on Neural Networks

论 文 作 者： 闫硕

领　　域： 软件工程

研 究 方 向： 机器学习

申 请 学 位： 工程硕士专业学位

指 导 教 师： 刘博　企业导师姓名

所 在 单 位： 软件学院

答 辩 日 期： 2019 年 5 月 23 日

授予学位单位： 北京工业大学

独 创 性 声 明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，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，也不包含为获得北京工业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了谢意。

签　　名：_____

日　　期：年*月*日

关于论文使用授权的说明

本人完全了解北京工业大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

（保密的论文在解密后应遵守此规定）

签　　名：_____　　日　　期：　　年*月*日

导师签名：_____　　日　　期：　　年*月*日

摘 要

改革开放以来，国家集中精力进行经济建设，工业化进程急速加快，但是，与此同时带来了十分严重的环境污染。其中空气污染是近些年来人们最为关注的，尤其是近几年的雾霾十分严重。由于雾霾对人们的呼吸系统产生很大伤害，但是以人们现在的科技水平还无法根治空气污染，所以对空气质量进行预测变得非常重要，尤其是对空气质量的细粒度预测。本文首先采用了训练速度快，泛化性能好的 ELM(Extreme Learning Machine，极限学习机)作为基础进行研究。但是 ELM 对于函数的拟合能力比深度学习差很多。所以为了进一步提升预测精度，很多学者使用 CNN，RNN 等深度学习模型对空气质量进行预测，但是由于这些模型无法将输入和输出同时作为序列处理，所以精度较差。为了充分利用序列信息，本文基于 Seq2Seq(Sequence-to-Sequence，序列到序列)模型对空气质量预测进行了研究。

(1) 为了让 ELM 进行细粒度的预测，本文在其基础上本文提出了 GBELM (Gradient Boosting Extreme Learning Machine, 梯度提升极限学习机)。GBELM 主要针对其激活函数和随机确定输入层和隐藏层权值所带来的不稳定进行了改进。对于激活函数，采用 ReLU (Rectified Linear Unit，修正线性单元)和 softplus 进行代替，同时还试验了 RBF (Radius Basis Function)核函数对其的影响。对于不稳定的问题，采用梯度提升算法对其集成，稳定其输出并提高预测精度。实验表明，softplus 和 ReLU 都可以为 ELM 带来明显的提升，但是 RBF 函数无法使得 ELM 对空气质量的预测精度有很大的提升，同时应用梯度提升算法使得 ELM 的性能获得了进一步的提升。

(2)本文针对 Seq2Seq 训练速度慢和误差积累的问题提出了 n-step AAQP 模型。对于训练速度慢的问题，使用全连接层和位置编码代替了 RNN 作为编码器。原始的解码器采取递归预测的方式获取结果，会导致误差积累较大，为了减少其误差积累，解码过程采用 n-step 递归预测。实验证明，用全连接层和位置编码代替 RNN 后，预测精度能够接近甚至超过 Seq2Seq。另外，n-step 递归预测减少了解码时间步，有效的减弱了误差积累，同时大幅提升了训练速度。最后，通过对两个模型的对比，改进后的 Seq2Seq 的预测精度更高，而且对于突然的变化，GBELM 不能将其很好的拟合出来，但是改进后的 Seq2Seq 能够更好地拟合出空气质量的突发变化。

关键词：空气质量预测；极限学习机；梯度提升； 序列到序列； 注意力机制

Abstract

Last decades, China has put its focus on economic development, which expedites industrialization progress, but, it brought severe environmental pollution. Among these pollutions, air pollution has drawn much attention of people, especially the haze. Although haze is deleterious to human's respirational system, it is hardly possible to uproot air pollution given the current technology of human, so the most efficient way to prevent people from damage of air pollution is prediction especially fine-grained prediction First, we try to predict air pollution based on ELM (Extreme Learning Machine), which has high training speed and good generalization performance. However, ELM cannot approximate complex function, more and more researchers began using deep learning models, such as CNN, RNN, to predict air pollution. Whereas, these models cannot treat input sequence and output sequence as sequence simultaneously, so we carried out research based on Seq2Seq(Sequence-to-Sequence) in order to exploit the sequential information.

(1) To adapt ELM to fine-grained prediction, we proposed GBELM (Gradient Boosting Extreme Learning Machine). Its activation function was changed to ReLU(Rectified Linear Unit) and softplus and we also applied RBF(Radius Basis Function) kernel to it in order to improve its performance. Furthermore, to overcome its instability caused by determining the weight between input layer and hidden layer randomly, gradient boosting was used to integrate ELMs. The experiments turned out that ReLU and softplus could improve the performance of ELM obviously while RBF only improved little performance of ELM. With the help of gradient boosting, the performance of ELM was improved further.

(2) Seq2Seq was used to predict air quality, but it had a very low training speed and the problem of error accumulation caused by recurrent prediction. To alleviate these two problems, we proposed n-step AAQP. In, n-step AAQP, a fully connected layer with position embedding replaced the RNN of encoder. In addition, n-step recurrent prediction was applied to decoder. Our experiments proved that n-step AAQP could achieve better result than original Seq2Seq and accelerate the training process. The n-step recurrent prediction was useful to allay the error accumulation when the n was chosen appropriately. Finally, by comparing GBELM and n-step AAQP, we found n-step AAQP was more accurate on air quality prediction. Moreover, when facing sudden change in air quality, n-step AAQP had a better performance.

Keywords：air quality prediction；extreme learning machine；Seq2Seq；attention

目 录

第1章 绪论

1.1 背景

随着人类工业化的不断深入，人类越来越依赖煤，石油等化石燃料。因为化石燃料具有储量大，易于利用的特点，被广泛应用于人们的社会生产中。虽然化石燃料为人类社会的发展提供了巨大的能量，但是同时也造成了十分严重的环境污染。化石燃料的燃烧会释放二氧化碳，一氧化碳，一氧化氮，二氧化硫等污染物。这些污染物会造成温室效应，酸雨等现象威胁地球上生物的生命安全。除了气体外，燃烧化石燃料还会释放微小的颗粒物，如 PM2.5 或者 PM10，这些颗粒物聚集起来会形成巨大的雾霾。这样的雾霾对人们的身体健康，尤其是呼吸系统是个十分严重的威胁，甚至危及生命。几十年前，著名的伦敦大雾使得所有伦敦市民的生命受到威胁。而在过去几十年，发达国家逐渐减少了污染的排放，但是像中国这样的发展中国家一直受到雾霾问题的困扰。在中国的北京，人口密度大，而其 PM2.5 的浓度峰值可能会达到 $1000\mu g/m^3$，因此对人们的健康伤害很大。所以如果人们无法及时采取措施，将会受到巨大的伤害。除了中国以外，印度，蒙古，甚至非洲国家也深受雾霾的困扰。可是以人类现在的科技水平无法从根本上解决雾霾的问题，因此能够准确预测雾霾的出现变得尤为重要。只要可以提前给出人们预警，人们就可以提前采取措施从而减少雾霾对自身健康的损伤。目前被人们广泛使用的空气质量预报服务，如电视台空气质量播报，手机 APP 多是对未来一天空气污染的最高值进行预报，但是这种预报难以满足人们的需求。由于空气质量在一天之内就有可能会产生很大的变化，如果只对未来一天的最高值进行预报，人们无法很好地安排自己的生活。例如第二天的空气质量只有下午是重度污染，而早上的空气质量是很好的，如果是粗粒度预报则会导致人们一整天无法安排户外活动，但是细粒度预报可以让人们在上午的时候安排一些户外活动。因此细粒度的预报更有助于人们对生产生活的安排，所以更多的研究人员也着手于对空气质量的细粒度预报。

1.2 国内外研究现状

人类处于大数据时代，机器学习成为对大数据进行挖掘的重要手段。随着气象数据和空气质量数据的积累，利用机器学习进行空气质量预测变得越来越流行，因为它容易实现而且精度较高。线性模型是机器学习中的基础模型，而现在很少有学者使用这个模型进行空气质量预测。T.S. Rajput 等[1]使用 MLR(Multiple

Linear Regression 多元线性回归)对印度的空气质量做了预测。但是，在实践中，污染物和其影响因子之间有着复杂的非线性关系，因此线性模型往往难以达到很高的预测精度。K.P. Singh 等[2]对比了线性模型和非线性模型之后发现，非线性模型能够有效的捕捉空气质量数据中的非线性关系。因此像 ANN(Artificial Neural Network，人工神经网络)[3]或者 SVM(Support Vector Machine，支持向量机)[4]这样的非线性模型更加适合空气质量预测。Sechan Park 等[5]利用 ANN 对首尔地区的空气质量进行了预测。Yun Bai 等[6]利用小波分解和 ANN 对中国重庆地区的空气质量进行了预测。A. Azid 等[7]将 ANN 和 PCA(Principal Component Analysis，主成分分析)结合对马来西亚的空气质量进行了预测。S. De Vito 等[8]通过一种动态方法提升了 ANN 的性能。Z. Kang 等[9]在进行预测时使用一般模拟退火算法求解 ANN 的参数。C. Paoli 等[10]使用 ANN 对科西嘉岛的臭氧进行了预测。Mahajan 等[11]使用基于地理距离的聚类方式提升了 ANN 的预测性能并对台湾 4 个城市的空气质量进行了预测。M. Asghari 等[12]采用基因算法加速对 ANN 的训练，并建立了逐日预测模型。SVM 是一种拥有比 ANN 更强泛化性能的模型，因此也有很多学者在空气质量预测的研究中使用这个模型。A.S. Sánchez 等[13]对使用不同核函数的 SVM 和 ANN 进行了对比发现 SVM 的性能更加具有优势。P. J. G. Nieto 等[14]利用粒子群 SVM 预测了北西班牙地区的空气质量。K. Gu 等[15]利用递归预测的方式使得 SVM 可以捕捉序列信息。还有一种 ANN 被称为 ELM(极限学习机，Extreme Learning Machine)，ELM[16]是由黄广斌提出的。相比起 ANN，ELM 的训练速度更快而且不会陷入局部极值，所以其泛化能力很好，而且在使用核技巧之后有 SVM 一样的泛化能力。也有一些学者将 ELM 应用于空气质量的预测。J. Zhang 等[17]对比了 ANN，ELM 和 MLR 对香港的空气质量进行了预测，并发现 ELM 的预测效果更好。Deyun Wang 等[18]利用 Differential evolution 算法降低极限学习机输入层和隐藏层之间的参数的随机性，并提出一种预测框架对北京和上海的空气质量进行预测。Chi-Man Vong 等[19]为了解决数据不均衡问题，使用元认知在线 ELM 对空气质量进行了预测。综合来看，以上方法虽然可以完成对空气质量的预测任务，但是这些大多无法完成细粒度的预测，多是逐天的预测或者只对未来一个时刻进行预测。由于传统的机器学习方法对拟合非线性函数的能力有限，所以很难直接对未来一段时间的空气质量做细粒度预测。K. Gu 等[15]虽然建立了对未来 12 小时逐小时的预测模型，但是其采用递归预测，本质训练的模型也只是对未来 1 小时的预测模型。想要建立更加精确的预测模型需要提升模型本身的预测能力。

最近几年，深度学习在图像，自然语言处理，音频的处理上展现了卓越的性能，其本质是拥有更多隐藏层的 ANN。深度学习模型相比传统的机器学习明有

着更强大的非线性拟合能力，因此有很多学者将其应用到空气质量预测中以完成更加复杂的任务。深度学习的使用让学者们可以集成多个监测站的空气质量数据，同时能够对未来一段时间空气质量做细粒度的预测。X. Li 等[20]使用 SAE (Stacked Auto-Encoder，栈式自编码器) 从 12 个监测站提取信息，最后将提取到的信息输入到线性回归中得到预测结果。空气质量数据作为时间序列数据，如果采用 SAE 对其处理会损失很多序列信息。而另外一种深度学习模型 RNN(Recurrent Neural Network，循环神经网络)[21] 在处理序列数据上比 SAE，SVM 等更胜一筹。B. T. Ong 等[22]继续使用 SAE 但是使用了 RNN 来替换线性回归给出了未来 12 个小时的空气质量。RNN 的问题在于难以训练[23]，因为在训练过程中常常出现梯度消失或者梯度爆炸的现象，因此很多研究者会使用门控神经网络代替 RNN。LSTM (Long Short-Term Memory，长短期记忆)[24] 就是最常用的一种门控结构。它可以代替 RNN 的基本结构并克服梯度消失和梯度爆炸的问题。V. Chaudhary 等[25]和 E. Pardo 等[26]使用简单的 LSTM 分别对未来 12 小时和 24 小时的空气质量做了预测。X. Li 等[27]除了使用天气和污染物数据还加入了对时间索引的编码作为 LSTM 的输入，并以全连接层输出结果。J. Zhao 等[28]则利用了目标监测站附近几个监测站的数据并利用 LSTM 构建模型。J. Wang 等[29]也使用了 LSTM 但是它采用格兰杰因果分析选择最相关的监测站。Y. Zhou 等[30]基于 LSTM 建立了可以对多个监测站进行预测的模型。M. Kim 等[31]和 İ. KÖK 等[32]发现 RNN 和 LSTM 可以比 ANN，SVM 等取得更好的结果。GRU (Gated Recurrent Unit)[33] 是一种 LSTM 的简化版本，但是它往往能够取得比 LSTM 更好的结果。V. Athira 等[34]对比了原始 RNN，LSTM 和 GRU 在空气质量预测的性能后，发现 GRU 的预测精度最高。B. Wang 等[35]给 LSTM 和 GRU 加入残差连接后对比这两个模型的预测效果，实验结果显示 GRU 的精度更高。X. Sun 等[36]放弃了使用 RNN 处理输入序列转而使用卷积函数预处理数据，然后使用 GRU 对空气质量进行了预测，同时实验结果还显示 GRU 能够达到比 LSTM，ANN，SVM，随机森林和多元线性回归更高的精度。S. Du 等[37]和 C. Huang 等[38]在将数据输入 LSTM 之前，使用 CNN(Convolutional Neural Network，卷积神经网络)[39]而不是卷积函数处理输入数据。比起固定卷积函数，CNN 可以根据任务学习出卷积函数。Chao Zhang 等[40]预测空气质量时未采用空气质量监测站所记录的数据，而是直接用 CNN 通过照片对空气质量进行预测。Soh 等[41]使用 CNN 提取诸如山川之类的地理信息，然后使用 LSTM 和 ANN 提取目标监测站和与其最相关监测站的信息。最后将这些提取到的信息合并到一起获取最终的预测结果。Z. Pan 等[42]建立了一个具有时序模块，空间模块和推理模块的模型。推理模块从空间模块中得到时序模块所需的模型参数。时序模块可以是 CNN，RNN 或者

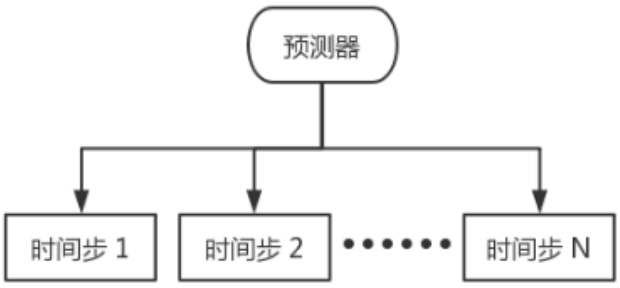
ANN 并且负责给出最后的预测结果。以上的研究中，研究者不是利用全连接网络预测结果就是使用 CNN 提取序列信息，但是空气质量预测中的输入和输出都是序列。虽然 CNN 可以提取序列信息，但是只能提取固定长度的序列信息。所以为了更加有效的提取序列信息，Seq2Seq(Sequence-to-Sequence，序列到序列)模型是更好的选择。Seq2Seq 同时使用 RNN 处理输入数据和输出数据，因此可以充分利用输入序列和输出序列的序列信息。V. Reddy 等[43]使用原始的 Seq2Seq 对北京的空气质量进行了预测，并证明了 Seq2Seq 在空气质量预测上的的效果很出色。在 Seq2Seq 中，编码器最后一个时间步的输出结果往往做为上下文向量传给解码器解码，但是 RNN 的容量有限，而且会遗忘，因此只将编码器最后一个时间步的输出结果作为上下文向量会损失很多信息。T.C. Bui 等[44]则利用编码器所有输出结果在时间上的均值作为上下文向量来解决这个问题。然而不同时间步的编码器输出对解码结果的影响不同，所以简单使用均值是不合理的。一个更好的解决方案是 AM (Attention Mechanism，注意力机制)[45]。AM 可以给不同时间步的编码器输出不同的权值，因此更加合理。H. Wang 等[46]使用 CNN 提取不同监测站之间的关系，然后使用应用了简化版 AM 的 Seq2Seq 得到最终的预测结果。Y. Liang 等[47]使用了全局 AM 为不同监测站分配不同的权值，同时使用局部 AM 对编码器输出进行加权。W. Cheng 等[48]采用 RNN 提取输入数据的信息并用 AM 为其他监测站加权最后利用全连接层得到结果。

1.3 研究内容

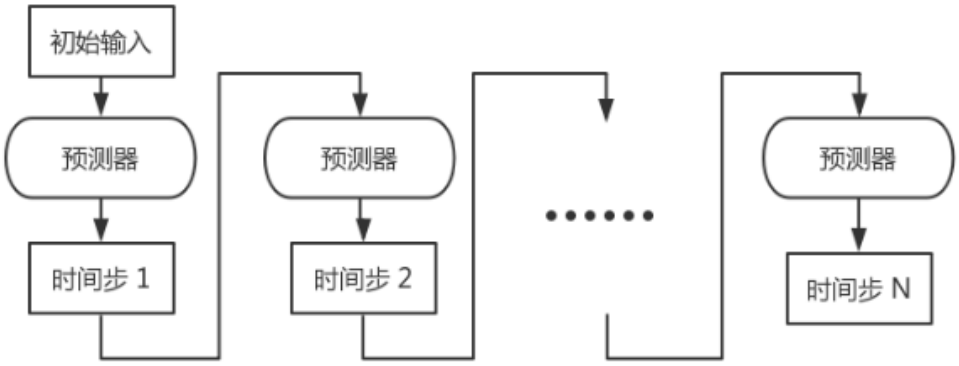
本文对空气质量的预测分为两个部分，一部分是使用浅层神经网络对空气质量预测进行研究，另一部分是使用深层神经网络，即深度学习对空气质量预测进行研究。

第一部分以 ELM 为基础对空气质量预测进行研究。虽然 ELM 在空气质量预测中的应用较少，但是其训练速度快，泛化能力强决定了它是较理想的预测方法。然而在空气质量预测中学者们主要针对其随机性的问题进行改进，对其本身的预测精度没有较明显的提升。因此，想要使得 ELM 完成对空气质量的细粒度的预测还需要对其预测的能力进行提升。首先需要对其激活函数进行改进，目前 ELM 的训练中多以 sigmoid 作为激活函数，虽然可以的不错的预测结果，但是在当前机器学习领域中 sigmoid 主要用作二分类时输出层的激活函数，而不再用作隐藏层的激活函数。目前应用较多的激活函数为 ReLU(Rectified Linear Unit，线性修正单元)[49]和 softplus[49]这样的激活函数，因为 ReLU 的使用使得在训练深层神经网络时不必再进行预训练，同时其本身自带稀疏度限制，可以提高泛化性能。

而 softplus 则是 ReLU 的改进版，因为 ReLU 会强制将所有输入的负值结果归 0，因此会造成死神经元的现象，所以 softplus 不再将这些值强制归零，而是使其非常接近 0，从而避免死神经元的问题。所以使用 ReLU 和 softplus 作为 ELM 的激活函数可以进一步提升其预测精度。另外还可以利用核函数最 ELM 的泛化性能做进一步的提升。其中 RBF(Radius Basis Function，径向基函数)核函数在 SVM 中被广泛应用，这个函数可以将低维空间中的数据映射到高维空间，从而使得模型更容易对数据进行拟合。应用 RBF 和函数的 ELM(RBF)[50]能够获得像 SVM 一样的泛化能力。ELM 的另外一个问题是其输入层和隐藏层之间的权值是随机确定的，所以其预测结果的方差较大。所以实验中需要进行大量的实验以得到比较理想的模型。既然得到的一组好的参数需要做对 ELM 进行多次训练，不如直接将多次的预测结果综合起来，从而起到降低 ELM 预测结果的方差的目的，并在此基础之上提升预测精度。综合预测结果最理想的方法是集成方法。在众多集成算法中，梯度提升(Gradient Boosting)[51] 算法在很多数据挖掘竞赛中往往可以取得最佳的成绩，因此是集成算法中较为优秀的，所以将其应用于 ELM 并对空气质量数据进行挖掘，以此提升 ELM 的预测性能。整个方法被称为 GBELM(Gradient Boosting Extreme Learning Machine，梯度提升极限学习机)。



(a)



(b)

图 1-1　直接预测(a)和递归预测(b). 直接预测同时给出 N 个小时的预测结果。递归预测将上一时刻的预测结果作为下一时刻的输入。初始输入可以是真实值。

Figure 1-1 Direct prediction (a) and recurrent prediction (b). Direct prediction give the N hours prediction simultaneously. Recurrent prediction takes the output of last step as input. The initial input can be the real value of last step.

第二部分则是以 Seq2Seq 为基础的深度学习对空气质量预测进行研究。虽然 ELM 是比较出色的机器学习模型，即便进行了改进，但是其预测能力还是无法获得质的飞跃。因此为了进一步获得更加精确的预测结果，深度学习方法便成为新的选择。虽然深度学习作为神经网络的一种容易过拟合，但是近几年深度学习领域已经发展出了一些有效的防止过拟合的技术。在众多深度学习模型中，Seq2Seq 是最符合空气质量预测的要求的，因为它是将一个序列转化为另一个序列的过程。空气质量预测是将历史数据，如污染物，天气等随时间变化的序列作为输入，再获得未来一段时间的污染物序列，因此其本身也是一个序列转化到另一个序列的过程，所以非常适合使用 Seq2Seq 对空气质量进行预测。原始的 Seq2Seq 往往预测能力较差，所以会引入 AM 来增强其预测能力。可是即便有了 AM，Seq2Seq 还是有两个问题。第一个问题是误差的积累。实际上，Seq2Seq 使用的是递归预测，也就是将上个时间步的预测结果作为当前时间步的输入。由于每一个时间步的预测都存在误差，所以递归预测会在每次预测之后积累误差。随着时间的推移，误差积累会逐渐增大导致后面时间步的预测结果变差，同时因为 Seq2Seq 的优化是基于所有输出结果的，所以这会对整体的预测产生不利的影响。相对的通过全连接层给出预测结果的方法采用的是直接预测。直接预测将未来一段时间的预测结果同时给出，如 ELM，ANN 等都使用这个策略，因此不存在误差积累的问题但同时也将输出序列的序列信息丢失。图 1-1 是直接预测和递归预测的示意图。另外一个问题则是训练速度慢。Seq2Seq 模型由于采用 RNN 作为编码器和解码器，在执行时只能逐步计算结果无法并行计算从而同时得出每个时间步的输出，所以训练速度受到很大影响。在空气质量预测中，往往对每个监测站建立一个模型，如果监测站较多则需要耗费大量时间训练。一旦模型衰退严重，需要重新训练则又会消耗大量的时间。因此提升训练速度是十分必要的。为了解决这两个问题，本文使用 n-step 预测，这是一种介于递归预测和直接预测的方法。在使用 n-step 预测时，解码器的每个时间步给出几步的直接预测。例如，在空气质量预测中，如果 n=3 则解码器的每个时间步给出未来 3 小时的预测结果。因此当 n=1 时则完全等同于递归预测，而如果 n 等于目标序列的长度时则等同于直接预测。因此减少了时间步所以误差的积累减小了，同时也减小了训练时间。图 1-2 是 n-step 预测的示意图。除了 n-step 预测之外，本文将 RNN 编码器替换为全连接编码器，这样就使得编码器不必逐步执行而是可以并行直接同时得到输

出结果。这个方法被称作 n-step AAQP(Attention Based Air Quality Predictor，基于注意力机制的空气质量预测器)。

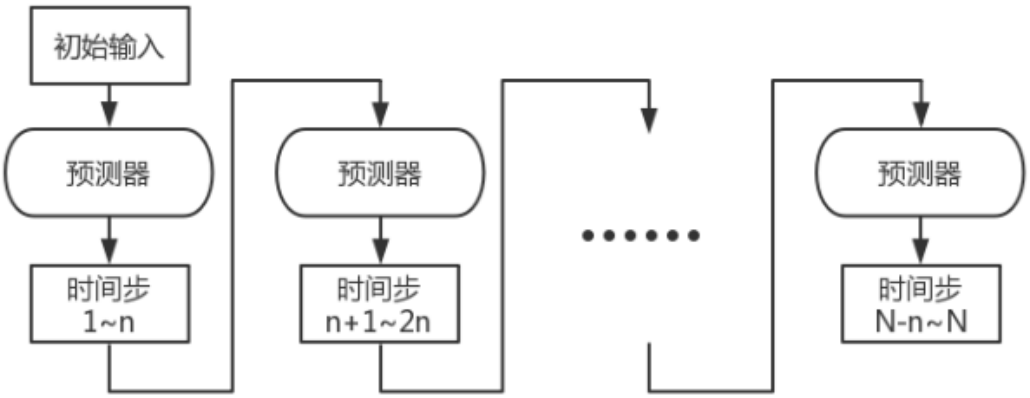


图 1-2　N-step 递归预测。预测器将前 n 步结果作为下一时间步的输入。

Figure 1-2 The n-step recurrent prediction. The predictor takes the last n-step as input and output n-step result.

综上所述本文的主要贡献如下：

- 通过修改 ELM 中激活函数或使用核函数的来达到增加预测精度的目的。
- 使用梯度提升算法对 ELM 进行集成，提高算法出结果的稳定性，并进一步提高预测精度。
- 使用全连接层和位置编码对 Seq2Seq 的 RNN 编码器进行替换，从而提高训练速度。
- 使用 n-step 递归预测，减少解码时间步，进而减少误差积累提升预测精度，并进一步提高训练速度。

1.4 文章组织结构

第一章为绪论，先对课题的研究背景和意义进行简介，阐明了当前空气质量预测的发展方向，之后对目前国内外研究现状进行了综述。最后对当前研究中存在的问题以及本文提出的解决方案进行阐述。

第二章是基于神经网络的空气质量预测方法。本章主要对应用于空气质量预测的主要神经网络方法进行介绍，包括 ANN，RNN，LSTM，GRU 和 Seq2Seq。

第三章是基于集成极限学习机的空气质量预测。本章首先阐述了梯度提升极限学习机的算法流程，之后对其预测精度和训练时间进行了讨论。

第四章是基于序列到序列模型的空气质量预测。本章首先对 n-step AAQP 的

算法流程进行描述，然后讨论了其在空气质量预测中的预测效果。

第五章对 AAQP 和 GBELM 的预测效果进行了对比，还有两种方法的建模时间预测时间的对比，以及最后可视化结果的对比。

第2章 相关理论技术

本章主要介绍广泛应用于空气质量预测的人工神经网络技术。2.1 节介绍了经典的 3 层 ANN，包含一个输入层，一个隐藏层和一个输出层。但是 ANN 容易陷入局部极值，泛化性能较差，因此本文使用在 ELM 的基础进行研究。2.2 节介绍了 ELM 的相关理论。ELM 有着更好的泛化性能，但是其难以向深层扩展。所以为了获取更高的预测精度，本文又基于深度学习对空气质量预测进行了研究。2.3 节介绍了 RNN 的基本理论。RNN 是用来处理时间序列数据的，所以十分适合用于空气质量预测，但是其由于 RNN 本身存在严重的梯度爆炸和梯度消失的问题，所以很少使用。2.4 节介绍了 LSTM，其有效的减少了 RNN 的梯度爆炸和梯度消失问题，因此常常被应用于实际问题中。2.5 节介绍了 GRU，一种 LSTM 的改进版本，但是其结构更加简单，而性能往往更加强大。然而 RNN，LSTM 和 GRU 都无法将输入和输出同时作为序列处理，但是空气质量的预测需要输入和输出都是序列，因此本文基于 Seq2Seq 模型进行研究。2.6 节介绍了 Seq2Seq 模型。它可以将一个序列编码为一个定长向量，然后经由解码器将这个定长向量解码为输出序列，所以十分适合空气质量预测的场景。

为了更好地说明本文设计的方法，设数据集 D={$(\boldsymbol{X},\boldsymbol{Y})$}，其中$\boldsymbol{X}$为输入数据，对于 ANN 和 ELM 而言，$\boldsymbol{X}\epsilon R^{M\times S*N}$，$M$为样本数量，$S$为输入序列的长度，$N$为特征数量。同时$\boldsymbol{Y}\epsilon R^{M\times T}$，其中$T$表示输出序列的序列长度。而对于 Seq2Seq 而言，$\boldsymbol{X}\epsilon R^{M\times S\times N}$, $\boldsymbol{Y}\epsilon R^{M\times T\times Q}$，其中$Q$为预测序列的特征数量。

2.1 人工神经网络

ANN 的灵感来自于动物神经网络，通过在计算机上模拟动物神经网络的工作方式构建的虚拟神经网络。神经网络被广泛用于对于文本，声音，图像等信息的挖掘中。ANN 由多层的神经元构成，相邻层的神经元相互连接，其示意图为图 2-1。一般的全连接神经网络多采用三层结构，即输入层，隐藏层和输出层，因为增加层数会导致梯度消失问题。数据由输入层输入，经由隐藏层做非线性变换，从输出层的到预测结果。

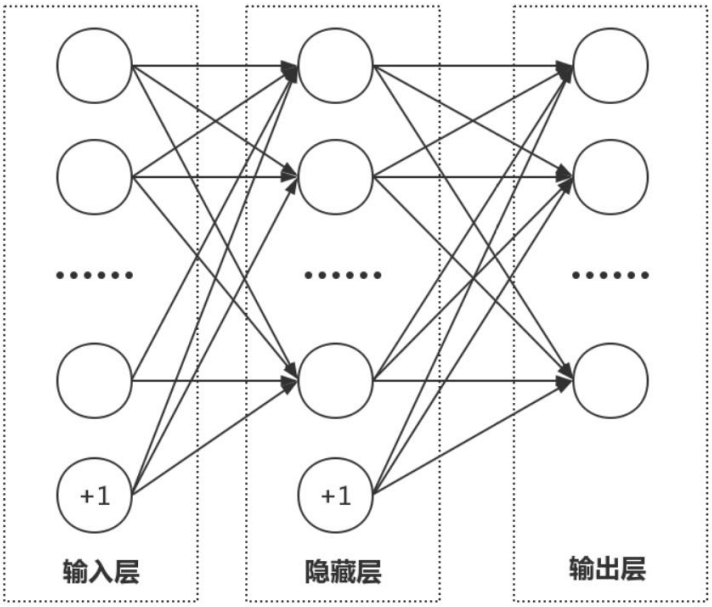


图 2-1　ANN 的示意图。

Figure 2-1 Illustration of ANN.

输入层神经元由特征数量决定，因此是N个；而输出层神经元由输出数据决定，因此是T个；隐藏层神经元属于超参数，所以需要手动确定。不同层之间的连接称为权重，其中输入层和隐藏层的权重用矩阵$\boldsymbol{W}_{ih} \in \mathbb{R}^{N\times L}$表示，隐藏层和输出层的权重用矩阵$\boldsymbol{W}_{ho} \in \mathbb{R}^{L\times T}$，其中$L$表示隐藏层神经元的数量。图中“+1”表示偏置单元。和权重相似，输入层和隐藏层的编制表示为$\boldsymbol{b}_{ih} \in \mathbb{R}^{L}$，隐藏层和输出层的偏置表示为$\boldsymbol{b}_{ho} \in \mathbb{R}^{T}$。因此神经网络的输出结果可以用如下公式计算：

$$\boldsymbol{h} = \mathrm{g}(\boldsymbol{X} * \boldsymbol{W}_{ih}) + \boldsymbol{b}_{ih} \tag{2-1}$$

$$\boldsymbol{P} = \mathrm{g}(\boldsymbol{h} * \boldsymbol{W}_{ho}) + \boldsymbol{b}_{ho} \tag{2-2}$$

其中$\boldsymbol{P}$是神经网络的输出结果，$\boldsymbol{h}$是隐藏层神经元的激活值，$\mathrm{g}(\cdot)$是激活函数，一般采用 sigmoid, tanh, relu 等。为了使神经网络可以得到正确的输出结果，需要对神经网络进行训练。首先需要定义损失函数，由于本文对空气质量进行预测所以采用 MSE(Mean Squared Error, 均方误差) 作为损失函数，其定义如下：

$$loss = \frac{1}{2m}||\boldsymbol{P} - \boldsymbol{Y}||_2^2 \tag{2-3}$$

最后通过梯度下降法不断调整模型中的权重和偏置直到损失函数最小化就可以完成对神经网络的训练。而实际使用中一般采用小批量梯度下降，即每次使用一个小批量数据对模型参数进行更新，m表示批量大小。小批量梯度下降不仅节省

内存，其收敛速度往往也更快。对所有模型中的所有权重$\boldsymbol{W}$和偏置$\boldsymbol{b}$，使用一下公式更新参数：

$$\boldsymbol{W} = \boldsymbol{W} - \boldsymbol{\alpha}(\frac{\mathbf{1}}{\boldsymbol{m}}\Delta\boldsymbol{W}) \tag{2-4}$$

$$\boldsymbol{b} = \boldsymbol{b} - \boldsymbol{\alpha}(\frac{\mathbf{1}}{\boldsymbol{m}}\Delta\boldsymbol{b}) \tag{2-5}$$

其中$\boldsymbol{\alpha}$是学习速率，通过手动设定。$\Delta\boldsymbol{W}$和$\Delta\boldsymbol{b}$为所有样本下损失函数对权重和偏置的偏导数之和。偏导数一般可以通过自动微分工具求得。

2.2 极限学习机

ELM(Extreme Learning Machine，极限学习机)和 ANN 有着同样的三层结构，但是与 ANN 不同的是它不使用反向传播算法和梯度下降算法进行训练。基于反向传播和梯度下降的 ANN 有四个主要问题。首先梯度下降算法中学习速率的选择需要很多次试验，过小的学习速率会导致训练速度慢而过大的学习速率会使得 ANN 无法收敛。第二，ANN 容易陷入局部极值，因为 ANN 的损失函数有多个极值点。第三，ANN 容易过拟合，这会导致 ANN 的泛化性能很差，因此为了更好的训练 ANN 需要通过适当的方式停止训练。最后，由于基于梯度下降和反向传播算法需要大量的迭代，因此需要耗费大量的时间。ELM 为了克服以上的缺点没有使用反向传播和梯度下降算法，而是采用了其他方式。首先一个 ELM 的输出结果可由如下公式获得：

$$\boldsymbol{P} = \mathrm{g}(\boldsymbol{W} * \boldsymbol{X} + \boldsymbol{b})\boldsymbol{\beta} \tag{2-6}$$

其中$\boldsymbol{W}$和$\boldsymbol{b}$输入层和隐藏层之间的权重和偏置，$\boldsymbol{\beta}$是隐藏层和输出层的连接权重，$\boldsymbol{O}$是输出。注意，ELM 中隐藏层和输出层之间不使用偏置。在 ELM 中，输入层和隐藏层之间的权重和偏置不通过梯度下降逐渐确定而是随机确定。当$\boldsymbol{W}$和$\boldsymbol{b}$被确定后，隐藏层的激活值便可以确定，所以整个 ELM 可以被简写为：

$$\boldsymbol{P} = \boldsymbol{H}\boldsymbol{\beta} \tag{2-7}$$

其中$\boldsymbol{H}$是隐藏层神经元的激活值。接下来需要确定的参数就只有隐藏层和输出层的连接权重$\boldsymbol{\beta}$了，因此这个问题可以被转化为如何求解$\boldsymbol{\beta}$。这时 ELM 的目的是使得输出$\boldsymbol{P}$尽可能地接近真实值$\boldsymbol{Y}$，所以实际上要求解的问题为：

$$\boldsymbol{Y} = \boldsymbol{H}\boldsymbol{\beta} \tag{2-8}$$

求解$\boldsymbol{\beta}$的过程可以看做是求解一个一般线性系统，所以$\boldsymbol{\beta}$的值可以由如下公式确定：

$$\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{Y} \tag{2-9}$$

其中$\boldsymbol{H}^{\dagger}$是$\boldsymbol{H}$的 Moore–Penrose 逆矩阵。若$\boldsymbol{H}^{\dagger}$是$\boldsymbol{H}$的 Moore–Penrose 逆矩阵则它满足如下条件：

$$\boldsymbol{H}\boldsymbol{H}^{\dagger}\boldsymbol{H} = \boldsymbol{H}, \boldsymbol{H}^{\dagger}\boldsymbol{H}\boldsymbol{H}^{\dagger} = \boldsymbol{H}^{\dagger}, (\boldsymbol{H}^{\dagger}\boldsymbol{H})^{T} = \boldsymbol{H}^{\dagger}\boldsymbol{H}, (\boldsymbol{H}\boldsymbol{H}^{\dagger})^{T} = \boldsymbol{H}\boldsymbol{H}^{\dagger} \tag{2-10}$$

这样求得的$\boldsymbol{\beta}$是唯一的因此不会陷入局部极值，而是得到唯一的全局极值，所以不存在泛化能力差的问题。在求解过程中不涉及学习速率的选取，而且只需要一次性的通过矩阵运算求得结果不需要进行迭代，因此训练速度非常快。

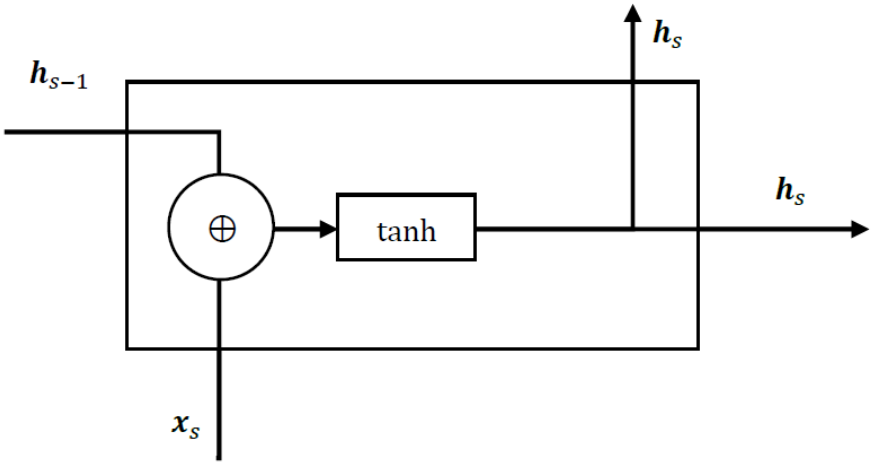


图 2-4　RNN 的单元结构。⊕符号表示矩阵的拼接。

Figure 2-4 The structure of vanilla RNN unit. The ⊕ denotes concatenation..

2.3 循环神经网络

深度学习是机器学习的一个分支，由 ANN 转化而来。在深度学习之前，神经网络大多如 2.1 中的 ANN 一样为 3 层结构，比较容易训练。万能逼近定理虽然指出只要 3 层的 ANN 有足够多的隐藏层神经元就可以逼近任意函数，可是在

实践中，如果隐藏层神经元的数量过多会导致过拟合等问题使得 ANN 难以训练。另外一个让 ANN 发挥逼近函数能力的方法是增加网络层数，但是简单的增加网络层数并不能使得 ANN 变得容易训练，反而会出现梯度消失的问题，导致误差无法被有效的反向传播，从而使接近输入层的参数无法被很好的调整，最终导致整个 ANN 的表现很差。2006 年 Hinton 提出了 DBN (Deep Belief Network，深度置信网络) ，该方法使用玻尔兹曼机初始化深度 ANN 的参数，可以在一定程度上解决梯度消失的问题，包括随后的 SAE 也是用来初始化 ANN 的参数来解决梯度消失的问题。ReLU 是梯度消失更为理想的解决方案，因为梯度消失的问题源于以 sigmoid 作为激活函数时，其导数在某些情况下会接近 0 导致误差无法有效反向传播。而 ReLU 则在定义域为非负数时导数为 1 使误差可以有效被反向传播，同时定义域为负数时值为零为 ANN 增加了稀疏度限制。目前最为流行的两种深度学习网络结构为 CNN 和 RNN，其中 RNN 多用于处理序列数据，十分适合用于空气质量预测。

RNN 作为一种深度学习的网络结构，也增加了网络的层数，但是与 CNN 和深度 ANN 不同的是，其层数的增加主要在时间轴上而不是直接再网络中增加层数。在每一个时间步上，其权重和偏置都是共享的，这样可以有效减少参数数量并且可以处理变长的序列。要获得一个 RNN 的隐藏层状态，即隐藏层激活值，需要进行一系列的操作，这一些列的操作构成了一个 RNN 的单元，原始 RNN 的单元是最简单的，如图 2-4 所示。由图中可知 RNN 的输入数据不仅包括当前时间步所需要的输入数据，同时还包括上一个时间步的隐藏层状态。因此第s个时间步的 RNN 的隐藏层单元的激活值$\boldsymbol{h}_s$由如下公式计算：

$$\boldsymbol{h}_s = \tanh(\boldsymbol{W} * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \mathbf{b}) \tag{2-11}$$

其中$\boldsymbol{h}_{s-1}$为上一个时间步隐藏层的状态，在第一个时间步往往取 0 矩阵，$\boldsymbol{W}$和$\mathbf{b}$为权重和偏置，中括号表示矩阵的拼接操作。在 RNN 中一般采用 tanh 作为激活函数，因为 sigmoid 会导致时间维度上的梯度消失，而 ReLU 会导致时间维度上的梯度爆炸，所以 tanh 成为最合适的选择。得到隐藏层的状态之后可以利用这个状态来的到最终的预测结果。一般的使用 RNN 最后一个时间步的隐藏层状态来获取最终的结果。最终结果的计算方式如下：

$$\boldsymbol{p} = \boldsymbol{W}_p * \boldsymbol{h}_S + \mathbf{b}_p \tag{2-12}$$

其中$\boldsymbol{p}$是最终的预测结果，$\boldsymbol{h}_S$为最后一个时间步的隐藏层的状态，$\boldsymbol{W}_p$和$\mathbf{b}_p$分别为获得最终结果时所用的权重和偏置。

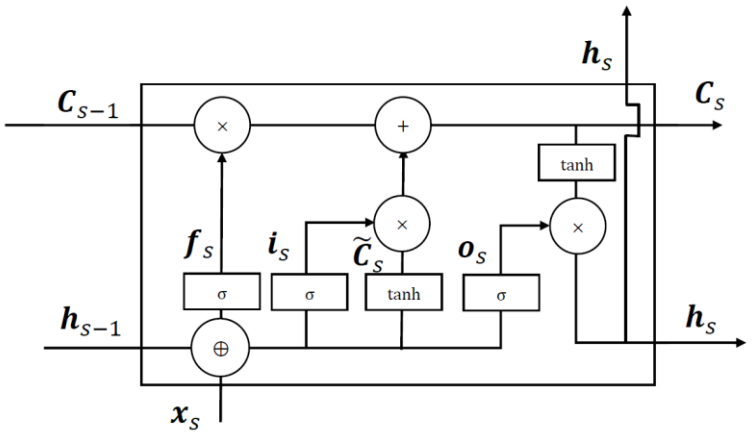


图 2-4　LSTM 的结构。

Figure 2-4 The structure of LSTM.

2.4 长短期记忆单元

然而原始的 RNN 有两个致命的缺陷，一个是梯度爆炸，另一个是梯度消失。即便不使用 sigmoid 和 ReLU，原始的 RNN 还是很容易出现梯度爆炸和梯度消失，从而使得 RNN 变的难以训练。在 RNN 中，梯度爆炸是指梯度的模以指数的速度变大，而梯度消失指的是梯度的模指数下降逼近 0。这两个问题可以通过使用门控单元代替原始的 RNN 单元来解决，其中最初实用的门控单元是 LSTM。LSTM 由输入门，输出门，遗忘门和 cell。LSTM 的结构如图 2-5 所示。LSTM 中的 cell 结构负责存储信息，输入门控制有多少新旧混合的信息可以进入 cell 中，输出门控制有多少信息可以输出为隐藏层的状态，遗忘们控制有多少旧的信息可以进入 cell。LSTM 的隐藏层单元可以通过如下公式计算：

$$\boldsymbol{i}_s = \sigma(\boldsymbol{W}_i * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_i) \tag{2-13}$$

$$\boldsymbol{f}_s = \sigma(\boldsymbol{W}_f * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_f) \tag{2-14}$$

$$\boldsymbol{o}_s = \sigma(\boldsymbol{W}_o * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_o) \tag{2-15}$$

$$\widetilde{\boldsymbol{C}}_s = \tanh(\boldsymbol{W}_c * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_c) \tag{2-15}$$

$$\boldsymbol{C}_s = \boldsymbol{f}_s * \boldsymbol{C}_{s-1} + \boldsymbol{i}_s * \widetilde{\boldsymbol{C}}_s \tag{2-16}$$

$$\boldsymbol{h}_s = \boldsymbol{o}_s * \tanh(\boldsymbol{C}_s) \tag{2-17}$$

其中$\boldsymbol{i}_s$是输入门，$\boldsymbol{f}_s$是遗忘门，$\boldsymbol{o}_s$是输出门，$\widetilde{\boldsymbol{C}}_s$是备选的 cell 值也就是输入信息

和旧信息的混合，$\boldsymbol{C}_s$是 cell 的值。$\boldsymbol{h}_s$是隐藏层的状态，$\boldsymbol{x}_s$是当前时间步的输入，. $\boldsymbol{C}_{s-1}$是上一个时间步的 cell 值并且代表着旧的信息。不同下角标的$\boldsymbol{W}$ 和 $\boldsymbol{b}$代表不同的权重和偏置。$\sigma(\cdot)$表示 sigmoid 函数，由于其值域为(0, 1)所以适合作为门控的开关。当输入门的值接近 0 时，几乎没有新的信息可以被输入；当遗忘门的值接近 0 时所有旧的信息会被舍弃；当输出门接近 0 时，隐藏层的状态接近 0，因为几乎没有信息被输出。当这些门的值接近 1 时，它们的行为与接近 0 时相反。从公式中可知，如果当前时间步的遗忘门接近 1 时，误差可以接近 100%回传给上一个时间步，而这时如果上一个时间步的输入门的值也接近 1 时，误差可以接近 100%传入备选 cell 值，从而使得$\boldsymbol{W}_c$和$\boldsymbol{b}_c$可以被很好的调整。而当遗忘门接近 0 或者上一个时间步的输入门接近 0 时，即便是梯度爆炸也可以使得梯度变得很小。但是门控单元不能完全解决梯度爆炸的问题，所以一般还要配合梯度截断，即限制梯度的最大值使得梯度即便爆炸也不影响训练。

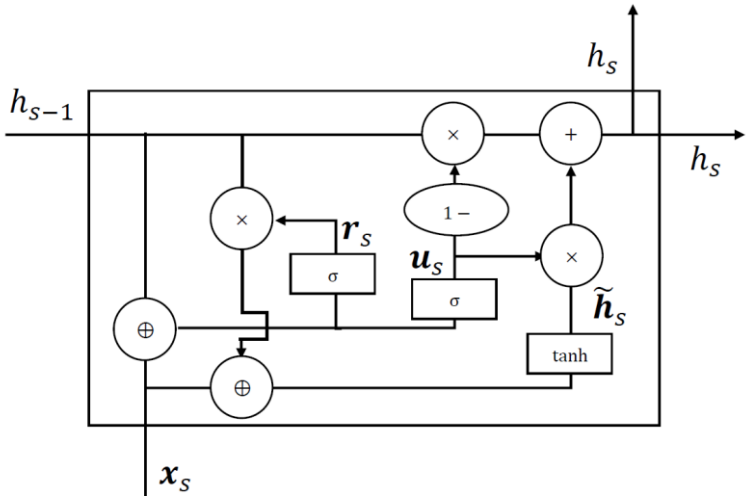


图 2-5 GRU 的结构。

Figure 2-5 The structure of GRU.

2.5 门控循环单元

GRU 也是一种流行的门控单元，虽然它的结构相比 LSTM 有所简化但是其性能有时候更加强大。目前很少有学者使用 GRU 作为 Seq2Seq 模型的编码器和解码器并应用于空气质量预测，所以本文也使用 GRU 并对比 LSTM 的性能。图 2-5 展示了 GRU 的结构。从图中可以看出，GRU 只有两个门，一个更新门和一个重置门。重置门负责控制有多少旧的信息可以参与和输入信息的混合；更新门负责控制有多少新的信息可以进入 cell，同时新的信息越多则旧的信息被保留的越少。GRU 的隐藏层的状态可以由如下公式计算：

$$\boldsymbol{u}_s = \sigma(\boldsymbol{W}_u * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_u) \tag{2-18}$$

$$\boldsymbol{r}_s = \sigma(\boldsymbol{W}_r * [\boldsymbol{h}_{s-1}, \boldsymbol{x}_s] + \boldsymbol{b}_r) \tag{2-19}$$

$$\widetilde{\boldsymbol{h}}_s = \tanh(\boldsymbol{W}_h * [\boldsymbol{r}_s * \boldsymbol{h}_{s-1}, \boldsymbol{x}_{s-1}] + \boldsymbol{b}_h) \tag{2-20}$$

$$\boldsymbol{h}_s = (1 - \boldsymbol{u}_s) * \boldsymbol{h}_{s-1} + \boldsymbol{u}_s * \widetilde{\boldsymbol{h}}_s \tag{2-21}$$

其中$\boldsymbol{u}_s$表示更新门，$\boldsymbol{r}_s$表示重置门，$\widetilde{\boldsymbol{h}}_s$是备选隐藏层状态，其他的符号与 LSTM 类似。当更新门的值接近 0 时，新的信息几乎全部被舍弃，而旧的信息几乎全部被保留；当重置门的值接近 0 时，旧的信息不参与和输入值的混合，所以只有当前的输入信息被用于备选隐藏层状态的计算。当门的值接近 1 时则与接近 0 的时候有着相反的行为。在解决梯度消失问题上，当更新门的值接近 0，上一个时间步更新门的值接近 1 时，或者当前更新门接近 1，重置门接近 1 和上一个时间步的更新门接近 1 时，误差几乎被 100%回传。当上述解决梯度消失的过程中的门有相反的行为时，误差几乎不被回传或者很少被回传，使得梯度爆炸也不影响训练。

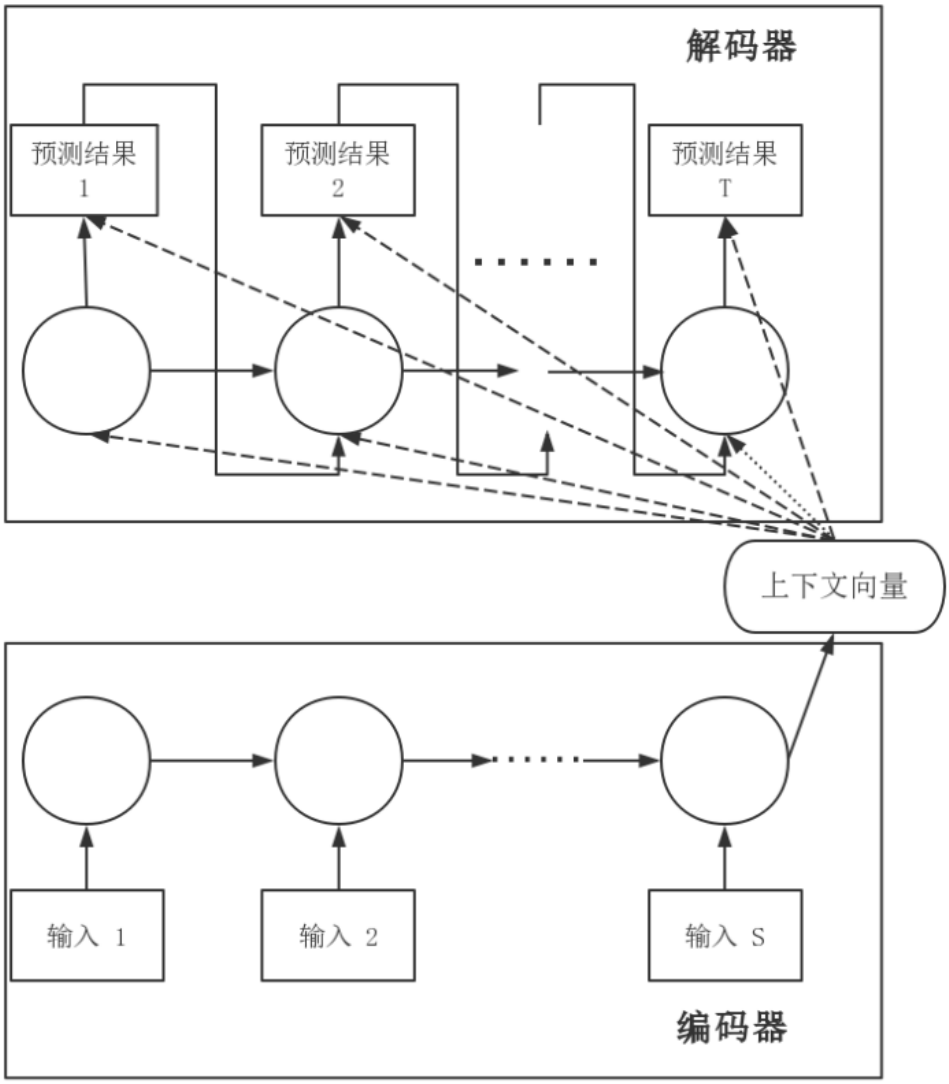


图 2-6 Seq2Seq 的结构。

Figure 2-6 The structure of Seq2Seq.

2.6 序列到序列模型

为了解决机器翻译上的问题，Seq2Seq 模型由 I. Sutskever 等[52]提出，并作为一个端到端的机器翻译解决方案很快流行起来。也有一些改进版被提出，如 K. Cho 等[52]。Seq2Seq 模型由一个编码器和一个解码器组成，其结构如图 2-6 所示。编码器负责将输入序列编码为一个固定长度的上下文向量，而解码器则负责将上下文向量解码成目标序列。Seq2Seq 中的编码器和解码器一般都由 RNN 构成。编码器的行为与一般的 RNN 完全一致，其最后一个时间步隐藏层状态作为

上下文向量输入到解码器中解码。解码器将上下文向量和当前时间步的输入序列拼在一起作为输入。而在空气质量预测中，解码器在当前时间步的输入是上下文向量和上一个时间步对空气质量的预测结果，也可以包括当前时刻的天气预报数据。以 GRU 为例，解码器的隐藏层状态由如下公式得出：

$$\boldsymbol{u}_t = \sigma(\boldsymbol{W}_u * [\boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S] + \boldsymbol{b}_u) \tag{2-22}$$

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r * [\boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S] + \boldsymbol{b}_r) \tag{2-23}$$

$$\widetilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_h * [\boldsymbol{r}_t * \boldsymbol{h}_{t-1}, \boldsymbol{p}_{t-1}, \boldsymbol{h}_S] + \boldsymbol{b}_h) \tag{2-24}$$

$$\boldsymbol{h}_t = (1 - \boldsymbol{u}_t) * \boldsymbol{h}_{t-1} + \boldsymbol{u}_t * \widetilde{\boldsymbol{h}}_t \tag{2-25}$$

其中$\boldsymbol{p}_{t-1}$是上一个时间步的预测结果，并且可由如下方式计算：

$$\boldsymbol{p}_{t-1} = \boldsymbol{W}_p * [\boldsymbol{h}_{t-1}, \boldsymbol{h}_S] + \boldsymbol{b}_p \tag{2-25}$$

2.7 本章小结

本章介绍了基于神经网络的基本理论，包括浅层网络 ANN 和 ELM，深度学习网络 RNN，LSTM，GRU 和 Seq2Seq。通过讲解每个方法的优缺点以及其建模的具体方式，介绍了每个模型的特点。其中 ELM 和 Seq2Seq 是本文工作的基础方法。

第3章 基于集成极限学习机的空气质量预测

ELM 作为一种神经网络具有泛化性能好，训练快的优势，但是若直接将其应用于细粒度预测还需要对其进行预测精度上的提升。因此为了提升 ELM 的预测精度，引入两点改进。首先极限学习机的输入层和隐藏层的参数是随机确定的，而其预测性能依赖这些随机确定的参数，所以其预测结果的方差较大。为了获得较好的预测效果，需要对模型进行多次训练从而获得较好的参数。如果能将多次训练的结果综合起来，不仅可以稳定其输出结果还可以提升预测精度，因此考虑使用集成算法对 ELM 进行集成。梯度提升算法作为一种集成算法在很多数据挖掘竞赛中优胜，因此将其应用于空气质量预测，也可以提升 ELM 的预测精度。另外，simgoid 函数作为 ELM 的激活函数被广泛应用，但是近几年随着神经网络的发展 sigmoid 函数的问题越来越多，逐渐由 ReLU, softplus 等函数取代。ReLU，softplus 等新兴的激活函数可以使神经网络摆脱预训练，对 ELM 而言最重要的是其稀疏度限制，可以将无用特征过滤，提高泛化能力，因此为了进一步提升 ELM 的预测精度，本文将 ReLU 和 softplus 应用于 ELM。

3.1 梯度提升极限学习机

3.1.1 使用 ReLU/softplus 的极限学习机

激活函数是神经网络训练中的重要组成部分，合适的激活函数使得神经网络有更好的表现。传统的极限学习机一般采用 sigmoid 函数作为激活函数，这也是一般神经网络所采用的激活函数，但是随着神经网络的进一步发展，sigmoid 很少再被作为隐藏层神经元的激活函数。目前更多的是将其作为 2 分类问题输出层的激活函数，其地位逐渐被性质更优良的激活函数所取代。Sigmoid 函数是一个 S 型函数，其中心点在(0, 0.5)处，在自变量趋近于无穷时其取值趋近于 1，而自变量趋近于负无穷时其取值接近-1。由于其中心点在(0, 0.5)，因此是一个很好的阈值函数其图像展示在图 2-2 中，其定义为：

$$g(x) = \frac{1}{1 + \exp(-x)} \tag{3-1}$$

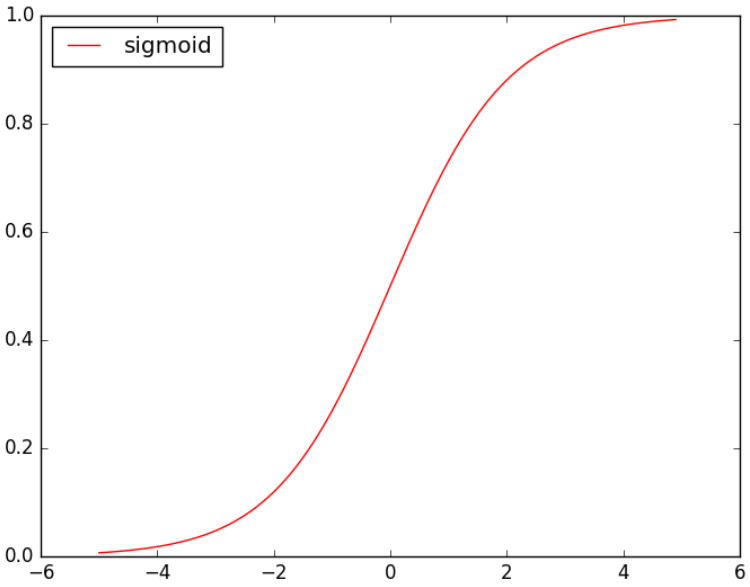


图 3-1　Sigmoid 函数。

Figure 3-1 Sigmoid function.

近来，深度学习的兴起，研究者发现随着网络层数的加深 sigmoid 作为激活函数会引起梯度消失的问题，因此目前在深度学习中，多采用 ReLU 作为激活函数，因为其在非负数定义域内是线性函数所以可以极大的减少梯度消失现象的出现，也因此层数更多的神经网络在使用 ReLU 后便不用再进行预训练了，同时由于其自变量为负数时取值为 0，可以将一些无用特征过滤掉，这样可以提高神经网络的泛化能力，也是其稀疏度限制的体现。ReLU 的定义如下：

$$g(x) = \max(0, x) \tag{3-2}$$

从定义中可知其激活值在x是负值时为 0，而在x是非负数时与x是线性关系。但是因为 ReLU 会强制将一些神经元的激活值变为零，所以会有一些死神经元，也就是说连接该神经元的权重无法被继续调整，从而使得神经网络变得脆弱。为了解决这个问题还有一种激活函数 softplus 不再强制将激活值为负数的神经元归零而是将其接近零，这样便解决了死神经元的问题。ReLU 和 softplus 的图像如图 2-3 所示，softplus 函数的定义如下：

$$g(x) = \ln(1 + \exp(x)) \tag{3-3}$$

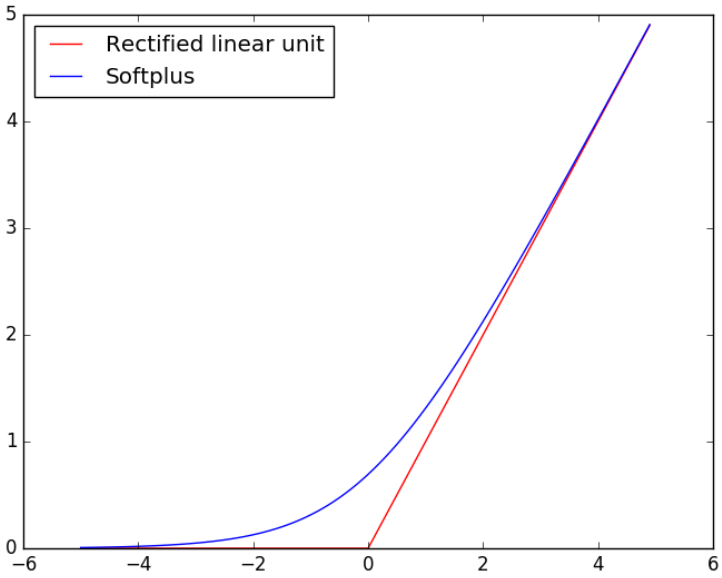


图 3-2 ReLU 和 softplus 函数。

Figure 3-2 ReLU and softplus functions.

3.1.2 使用径向基函数的极限学习机

还有一种 ELM 同样不使用 sigmoid 而是使用 RBF。使用 RBF 的 ANN 是一种通用的机器学习模型，而且有很好的逼近性质，所以采用 RBF 函数的 ANN 往往具有很好的泛化性能和预测精度。RBF 的作用是将低维数据映射到高维空间中，这样原本输入和输出没有线性关系的样本，在高维空间出现了线性的关系，然后再通过 ELM 的输出层得到更精确的预测结果。核函数，尤其是 RBF 对于 SVM 是至关重要的，因为没有核函数的 SVM 本质上是一个线性模型，有了核函数可以使得 SVM 处理非线性问题，而 RBF 则理论上可以让 SVM 处理任意非线性问题。RBF 的使用也提高了 SVM 的预测精度和泛化性能。同样的，将 RBF 函数应用到 ELM 中也可以使得 ELM 获得更好的泛化性能和预测精度。RBF 函数的定义如下：

$$\mathrm{k}(\boldsymbol{x},\boldsymbol{c}) = exp(-\boldsymbol{\gamma}||\boldsymbol{x}-\boldsymbol{c}||_2^2) \tag{3-4}$$

其中$\boldsymbol{x}$是输入向量，$\boldsymbol{c}$是中心向量。$||\boldsymbol{x}-\boldsymbol{c}||_2^2$表示的是每个样本到中心向量欧式距离的平方。当$\boldsymbol{\gamma}=\boldsymbol{\sigma}^{-2}$时，这个函数叫做以方差为$\boldsymbol{\sigma}^2$的高斯核函数，但是在ELM(RBF)中$\boldsymbol{\gamma}=\boldsymbol{\sigma}^{-1}$。另外 ELM(RBF)中的中心向量是随机确定的，一般是(0, 1)之间的随机数。在高斯核函数中σ可由如下公式计算：

$$\boldsymbol{\sigma}_i = \frac{c_{max}}{\sqrt{2h}}, i = 1,2,3 \ldots \ldots h \tag{3-5}$$

其中c_{max}是所有样本与中心向量之间距离的最大值。h是隐藏层神经元的个数。在 RBF-ELM 中$\boldsymbol{\gamma}$的计算方法与之类似：

$$\boldsymbol{\gamma}_i = \frac{\sqrt{2h}}{c_{max}}, i = 1,2,3 \ldots \ldots h \tag{3-6}$$

当确定了中心向量和$\boldsymbol{\gamma}$之后，激活值便可由如下公式计算：

$$\boldsymbol{H} = exp(-\boldsymbol{\gamma}||\boldsymbol{x} - \boldsymbol{c}||_2^2) \tag{3-7}$$

之后可以使用求解一般 ELM 的方法求解 RBF-ELM。

3.1.3 梯度提升极限学习机

ELM 虽然本身有较强的泛化性能能和预测精度，但是其本身由于随机初始化输入层和隐藏层的权重和偏置，导致其本身输出结果不稳定，一旦随机初始化的参数不够好那么就无法得出很好的预测结果，所以为了稳定其输出结果，本文将 ELM 作为梯度提升算法的基学习器，稳定其预测结果的同时还可以进一步提升性能。集成后的算法名为 GB-ELM。梯度提升算法是一种机器学习算法并且可以应用于回归和分类问题。这种算法将一些弱学习器进行集成从而组成一个强学习器。梯度提升算法实际上是一个加法模型，其定义为：

$$\mathrm{F}(x) = \sum_{q=1}^{Q} \gamma_q h_q(x) \tag{3-8}$$

其中Q表示基学习器的数量，$\mathrm{F}(x)$是最终集成后的模型，γ是当前弱学习器的权重，$h_q(x)$是基学习器。在本文中$h_q(x)$即为极限学习机。梯度提升算法在构建最终集成模型时采用前项分步的方式：

$$F_q(x) = F_{q-1}(x) + \gamma_q h_q(x) \tag{3-9}$$

在每一步，当前 ELM 的任务是让其输出结果加上之前所有基学习器的预测结果使得损失函数最小化：

$$h_q(x) = argmin(h)\sum_{i=1}^{M}\sum_{t=1}^{T} L(y_t^i, F_{q-1}(x_t^i) + \gamma_q h(x_t^i)) \tag{3-10}$$

梯度提升算法使用梯度下降来解决这个极小化问题，然而与梯度下降对 ANN 的优化不同，这次要调整的参数不是一个参数向量而是一个函数，在梯度提升中，这个函数时F_{q-1}，也就是说要让F_{q-1}在梯度的负方向，即最速下降方向，前进一步使得损失函数最小。为了使用梯度下降算法，需要先求得梯度并使得F_{q-1}在梯度的负方向上前进一步：

$$F_q(x) = F_{q-1}(x) - \gamma_q \sum_{i=1}^{M}\sum_{t=1}^{T} \nabla_F L(y_t^i, F_{q-1}(x_t^i)) \tag{3-11}$$

对应到梯度下降算法，γ_q则代表步长。$h_q(x)$代表搜索方向，也就是 ELM 要输出的结果应该是梯度的负方向，因此对于每一个 ELM，其拟合的目标值为：

$$\boldsymbol{r} = -\sum_{i=1}^{M}\sum_{t=1}^{T} \nabla_F L(y_t^i, F_{q-1}(x_t^i)) \tag{3-11}$$

接下来只要再通过线搜索的方法确定步长即可：

$$\gamma_q = argmin_{\gamma_q}\sum_{i=1}^{M}\sum_{t=1}^{T} L(y_t^i, F_{q-1}(x_t^i) - \gamma_q \frac{\partial L(y_t^i, F_{q-1}(x_t^i))}{\partial F_{q-1}(x_t^i)}) \tag{3-12}$$

然而在使用 MSE 作为损失函数的情况下，γ_q恒等于 1。而本文则采用 MSE 作为损失函数，所以不必再计算步长，但是为了增加模型的鲁棒性，还需要引入学习速率α，因此梯度提升模型变为：

$$F_q(x) = F_{q-1}(x) + \alpha\gamma_q h_q(x) \tag{3-13}$$

学习速率是超参数，所以需要手动设定。学习速率和基学习器数量有着强相关性，如果学习速率很小则需要更多的基学习器学习。一般来讲使用较小的学习速率有助于模型达到更高的精确率。由于每个基学习器需要学习上一次集成结果的残差，所以需要对第一个预测结果$F_0(x)$进行初始化，而对于 MSE 为损失函数的模型，$F_0(x)$一般采用训练数据目标值的均值，而对于多输出情况，采用每个输出对应目标值的均值。整个算法流程如下所示：

算法 1 梯度提升极限学习机

输入: 数据**X**, 目标值$\boldsymbol{Y}$, 学习速率α , 基学习器数量 M

输出: 集成后的极限学习机 $F_m(x)$

1. 初始化$F_0(x)$ 为训练集的每个输出目标值的平均

2. 对于 q = 1 到 Q:

(a) 对于 i = 1, 2, 3, ……, M 计算负梯度$\boldsymbol{r}$

$$\boldsymbol{r} = -[\frac{\partial L(y, F(x_i))}{\partial F(x_i)}]_{F=F_{m-1}}$$

(b)初始化一个 ELM 为 $h_m(x)$

(c)随机确定$\boldsymbol{W}$ 和 $\boldsymbol{b}$ 并计算隐藏层的激活值$\boldsymbol{H}$

(d)将x_i (i = 1, 2, 3, ……, N)输入 h_m预测 r 并得到$h_m(x)$

(e)$F_m(x) = F_{m-1}(x) + \alpha h_m(x)$

3. 输出$F_m(x)$

3.2 实验设置

3.2.1 数据集

本文使用 2017 年 4 月至 2018 年 3 月北京市逐小时空气质量数据以及逐小时天气数据。空气质量数据记录了SO_2, CO, NO_2, O_3, PM*2.5*, PM10 六项污染物逐小时的浓度。天气数据则记录了未来 24 小时逐小时的天气预报，包括温度，湿度，风力，风向和降水。本文只对雾霾的主要成因 PM2.5 的预测结果进行分析，其中模型的输入是当前时刻过去 24 小时的空气质量数据和天气数据，输出项则为未来 24 个小时的 PM2.5 浓度，也就是可以一次性的得到 24 小时的 PM2.5 的浓度。

3.2.2 模型评价标准

为了评价模型的好坏，本文采用了两种评价指标，一个是 MAE(Mean Absolute Error, 平均绝对误差)，另一个是 R^2(拟合优度)。

MAE 的定义如下：

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=0}^{n}|O_i - P_i| \tag{3-14}$$

其中O表示观测值，也就是真实数据。P代表模型的预测值。n代表样本数量。R^2的定义如下：

$$R^2 = 1 - \frac{\sum_{i=0}^{n}(O_i - P_i)^2}{\sum_{i=0}^{n}(O_i - \bar{O})^2} \tag{3-15}$$

其中$\bar{O}$表示观测值的平均值。

3.2.3 训练模型

实验平台的CPU为I6770HQ, 16GB内存，ANN和SVM则利用scikit-learn0.20实现。ELM系列方法使用numpy1.16.2和scipy1.2.1实现。ELM的四个方法均使用200个隐藏层神经元。GBELM的学习速率设置为0.05，而GBELM(ReLU)和GBELM(softplus)的隐藏层神经元是200个，而GBELM(RBF)则设置为100个。

除了使用ReLU,softplus和RBF的ELM以及GBELM外，SVM, ANN, ELM(sigmoid)被用于进行对比评测。

3.3 预测精度对比

表格 3-1 到表格 3-4 展示了在两个空气质量监测站上对各个方法对未来 24 个小时 PM2.5 预测结果的评价。为了更加细致的了解预测结果，表格中评价了每四个小时的 MAE 和 R^2，而最后一列表示所有时刻的预测结果 MAE 和 R^2 的平均值。表格中将当前列最优秀的数字用带下划线的加粗字体表示。

首先看奥林匹克中心站，GBELM (softplus)总体的精度是最高的，无论是MAE还是R^2都有着最好的值。GBELM (ReLU) 则仅次于GBELM (softplus)的精度，但是这两种方法的表现相差不多。具体地，就MAE而言，在前8个小时，可以发现GBELM (ReLU)略小于GBELM (softplus)，而在之后的16个小时的结果还是GBELM (softplus)高于GBELM (ReLU)，但是其优势并不是很大。就R^2而言可知，所有的指标都是GBELM (softplus)优于GBELM (ReLU)但是其优势也不是非常大。对于三种激活函数和一种核函数的比较中，可以发现 ELM(RBF)函数的表现是最差的，甚至还不如ANN和SVM的表现。三种激活函数中还是softplus表现的最好，ELM (softplus)的精度虽然是最高的，但是它相对ELM(ReLU)

的优势并不大。另外 ELM (softplus)在总体的优势相对 ANN，SVM 和 ELM(simgoid)都不是非常大，不过具体地，在前几个小时还是具有较大的优势的，但是后面时刻的预测结果也没有明显优势。有这些结果可知，ReLU 和 softplus 的确对 ELM 的预测精度有提升，而且再加入梯度提升算法对这些方法进行集成后，方法的精度又进一步提升了。

接下来看东四站。和奥林匹克中心站一样，GBELM (softplus)的总体 MAE 是最小的，总体 R^2 是最大的，所以其表现最好，但是其表现对比 GBELM(ReLU)也没有很大的优势。而且就 R^2 而言，可以看到 GBELM(ReLU)前 8 个小时的预测结果反而比 GBELM(softplus)稍好一些，而后面时刻的预测结果 GBELM(softplus)也只是稍好于 GBELM(ReLU)。和奥利匹克中心站不同，GBELM(RBF)的表现与其他两种 GBELM 相当，尤其是就 MAE 而言，但是其 R^2 相较其他两种方法稍逊一些。而在不使用梯度提升的 ELM 中，ELM(softplus)还是表现最好的，但是比 ELM(ReLU)的表现只是稍好，尤其是在具体 MAE 的对比中，ELM(ReLU)也有比 ELM(softplus)表现更好的预测。ELM(sofplus)的表现则全面优于 ELM(RBF)，尤其是前四个小时的预测结果 ELM(softplus)有明显的优势。与奥林匹克中心站不同，ELM(RBF)的表现相对于 ELM(sigmoid)具有了一定的优势。在 MAE 的比较上，ELM(RBF)的预测结果又明显的优势，但是在 R^2 的对比上优势变得很小，甚至在总体 R^2 的对比上有微弱的劣势。虽然 ELM(RBF)能比 ELM(sigmoid)获得误差更小的预测，但是趋势的预测上却相差不大。ANN 有着和 ELM(sigmoid)表现相似的预测结果，而且更是比 SVM 的预测结果更加出色。

综合两个监测站的测试结果来看，使用了 ReLU 和 softplus 激活函数可以使得 ELM 的性能获得提升，但是 RBF 在当前这两个监测站没有能够提升 ELM 的的性能。使用梯度提升算法对 ELM 进行集成的确达到了提升预测精度的目的，而且无论是 ELM(ReLU), ELM(softplus)还是 ELM(RBF)都获得了提升，尤其是在东四站，其提升效果更加明显。而对于激活函数的选择上，softplus 虽然在两个监测站上无论有没有梯度提升的帮助都有最佳的表现，但是其对使用 ReLU 的 ELM 并没有非常明显的优势，因此无法断定使用 softplus 就一定可以取得更好的结果，最好的做法是在实际问题中通过实验比较两种激活函数的性能之后再决定使用哪种激活函数。

表 3-1　奥林匹克中心站 MAE

Table 3-1 MAE of Olympic Center Station

方法	4	8	12	16	20	24	总体
SVM	23.68	27.26	31.17	34.84	38.15	41.31	32.73
ANN	24.69	29.33	33.08	34.90	38.10	40.24	33.39
ELM (sigmoid)	24.49	28.63	32.07	34.32	38.21	39.75	32.91
ELM (ReLU)	19.12	26.77	31.22	33.45	36.47	39.06	31.01
ELM (softplus)	19.00	24.65	29.88	33.72	35.45	38.19	30.15
ELM(RBF)	22.56	29.35	34.75	39.81	43.65	47.73	36.31
GBELM (ReLU)	**14.93**	**22.52**	27.69	31.50	35.02	38.74	28.40
GBELM (softplus)	14.97	23.02	**27.28**	**30.10**	**32.44**	**35.71**	**27.25**
GBELM (RBF)	19.22	25.88	31.93	36.72	40.69	44.89	33.22

表 3-2　东四站 MAE

Table 3-1 MAE of Dong Si Station

方法	4	8	12	16	20	24	总体
SVM	43.84	44.50	42.06	45.48	54.34	58.52	48.12
ANN	34.90	40.42	41.29	45.25	49.56	47.76	43.20
ELM (sigmoid)	37.14	39.99	43.53	46.32	47.22	46.23	43.40
ELM (ReLU)	27.74	36.14	38.80	41.88	47.24	49.26	40.18
ELM (softplus)	26.42	34.97	39.33	43.68	45.53	47.80	39.62
ELM(RBF)	30.19	35.25	40.32	43.89	47.37	50.39	41.23
GBELM (ReLU)	22.66	31.26	35.44	39.54	42.71	44.81	36.07
GBELM (softplus)	**22.53**	31.04	34.15	**37.81**	**41.02**	**42.92**	**34.91**
GBELM (RBF)	25.52	**30.37**	**33.62**	37.83	44.17	48.80	36.72

表 3-3　奥林匹克中心站 R^2

Table 3-3 R^2 of Olympic Center Station

方法	4	8	12	16	20	24	总体
SVM	0.690	0.589	0.514	0.408	0.305	0.189	0.449
ANN	0.585	0.427	0.336	0.268	0.141	0.048	0.301
ELM (sigmoid)	0.625	0.529	0.452	0.393	0.253	0.185	0.406
ELM (ReLU)	0.710	0.522	0.401	0.348	0.287	0.172	0.407
ELM (softplus)	0.720	0.560	0.494	0.398	0.336	0.223	0.455
ELM(RBF)	0.617	0.391	0.276	0.164	0.037	-0.111	0.229

GBELM (ReLU)	0.806	0.650	0.603	0.506	0.400	0.278	0.541
GBELM (softplus)	**0.808**	**0.654**	**0.619**	**0.541**	**0.465**	**0.357**	**0.574**
GBELM (RBF)	0.659	0.456	0.389	0.276	0.169	0.015	0.327

表 3-4 东四站 R^2

Table 3-4 R^2 of Dong Si Station

方法	4	8	12	16	20	24	总体
SVM	0.432	0.341	0.398	0.323	0.174	0.025	0.282
ANN	0.536	0.356	0.291	0.220	0.102	0.064	0.261
ELM (sigmoid)	0.555	0.459	0.411	0.335	0.258	0.206	0.371
ELM (ReLU)	0.651	0.520	0.426	0.342	0.248	0.142	0.388
ELM (softplus)	0.687	0.540	0.419	0.340	0.285	0.190	0.410
ELM(RBF)	0.595	0.516	0.402	0.302	0.233	0.100	0.358
GBELM (ReLU)	**0.732**	**0.606**	0.531	0.476	0.391	0.284	0.503
GBELM (softplus)	0.728	0.596	**0.536**	**0.491**	**0.420**	**0.319**	**0.515**
GBELM (RBF)	0.641	0.556	0.499	0.431	0.329	0.177	0.439

3.4 时间对比

表3-5展示了不同方法的训练时间和预测时间。虽然实验在两个监测站进行，但是两个检测站的数据集大小相同，使用特征和模型参数相同，因此时间差异很小，表中的数值是两个监测站模型的训练时间和预测时间的平均值。就训练时间而言，ANN 的训练时间最快，比 ELM 中训练最快的 ELM(ReLU)还要快。虽然 ELM 的理论上说要比 ANN 快，但是这往往是的规模较小的数据集上，因为在 ELM 的计算中还需要进行激活值矩阵伪逆的计算，在规模较小的数据集上伪逆的计算不是非常耗时，然而如果到了较多的数据集上伪逆的计算相当耗时，而 ANN 采用 mini-batch 梯度下降算法将大型矩阵的计算分割为多个小矩阵的运算，所以 ANN 的训练速度取得了更大的优势。不过 ANN 虽然快一些，但是其优势也仅仅有 1 秒，而且 ELM 的预测精度更高，所以在实际应用中 ELM 仍然是首选。而在未使用梯度提升算法ELM的四个方法中，ELM(ReLU)的训练速度最快，原因是其激活函数的计算非常容易，因为只是简单的比大小，然而 sigmoid 和 softplus 都涉及到指数运算，所以会相对耗时一些，不过也仅仅是相差了 1 秒以内的时间，所以并没有很大的优势。使用梯度提升算法后所有的 ELM 训练时间

明显变长，因为需要训练很多个 ELM 所以其所需时间变多。ELM(RBF)与 ELM(softplus)有着相似的训练时间，但是 GBELM(RBF)的训练时间明显比 ELM(RBF)要短，原因是 GBELM(RBF)使用了 100 个隐藏层神经元所以其训练时长较短。但是 GBLEM(softplus)的训练时长明显多于了 GBELM(ReLU)，但是两者的在空气质量预测的性能相似，所以如果需要对大量监测站建模时可以考虑使用 GBELM(ReLU)。预测时间上，ELM(sigmoid), ELM(ReLU), ELM(sofplus)和 ANN 相近，因为这四种方法的网络结构是一样的，只是训练方法不同，所以预测时间相差极小。所有的方法都可以在 1 秒之内得到预测结果，完全可以满足实际应用需要。

表 3-5 训练时间和预测时间

Table 3-5 Training time and predicting time

方法	训练时间	预测时间
SVM	15.484	0.555
ANN	**2.097**	**0.001**
ELM (sigmoid)	4.004	**0.001**
ELM (ReLU)	3.235	**0.001**
ELM (softplus)	4.022	0.002
ELM(RBF)	4.084	0.015
GBELM (ReLU)	360.932	0.097
GBELM (softplus)	423.389	0.135
GBELM (RBF)	373.820	0.880

3.5 结果可视化

图 3-1 和图 3-2 展示了 GBELM(softplus)预测结果的可视化，即将其预测的曲线和真实的曲线进行对比。本文只给出了模型输出的前 9 个时刻的预测结果，因为 9 小时之后的预测结果精度很差。在奥林匹克中心站的预测上，第 1 个时刻的输出几乎可以完美与真实值相匹配，可是在第 2 个输出时刻出现了轻微的滞后现象，而第三个时刻开始滞后现象更加严重尤其是对于突变的预测上，滞后较为严重。在东四站的预测结果中，同样是在第 2 个输出时刻出现了轻微的滞后现象，并且在第三个输出时刻变得更加严重。东四站的 PM2.5 变化更为剧烈，除了第一个输出时刻以外，其他的输出时刻不能将突变很好的描述出来。总的来说 GBELM(softplus)可以在两个站点的大多数情况下给出可信的预测结果，但是对于很大的突然变化无法给出十分精确地预测。

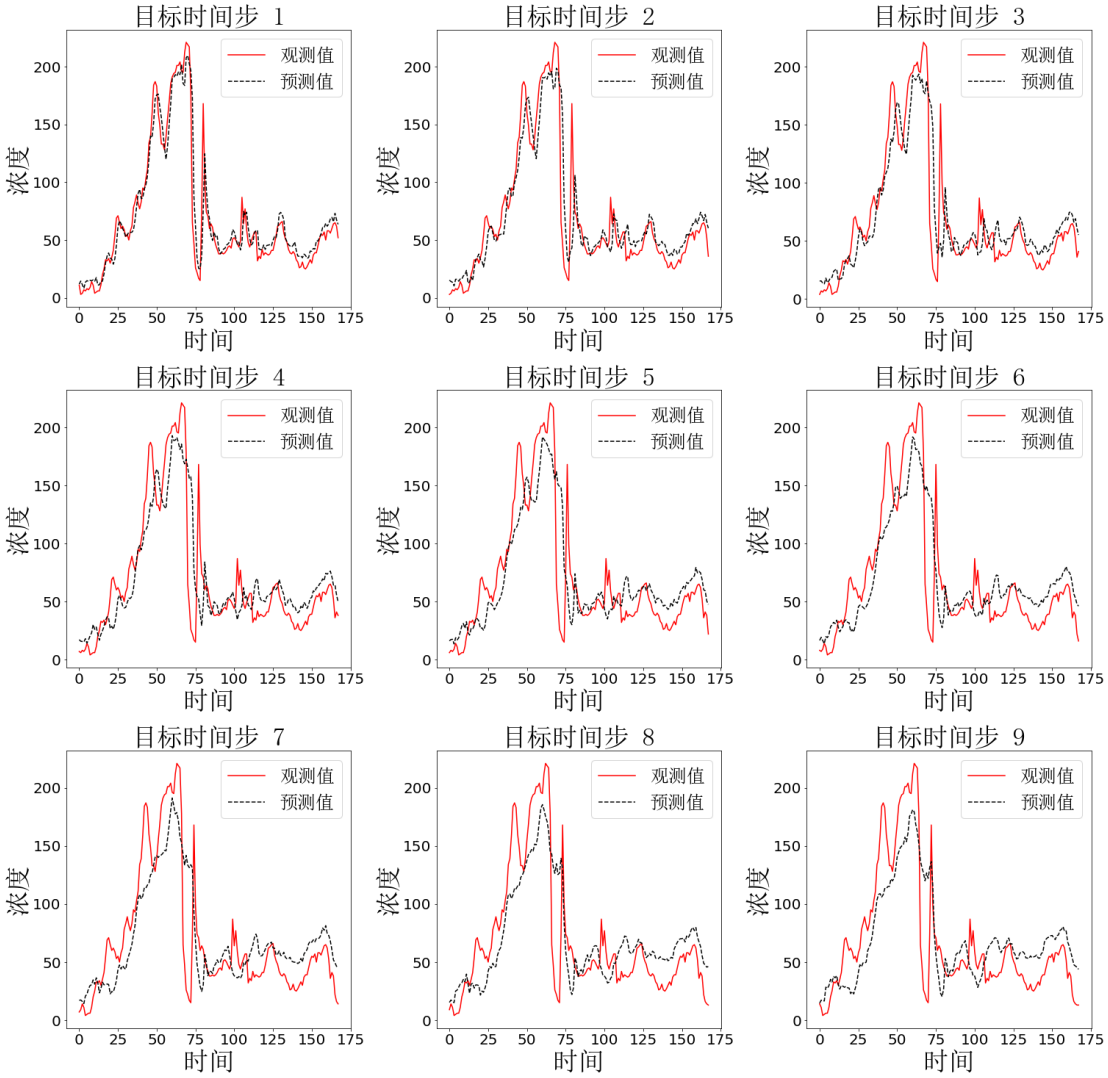


图 3-3　奥林匹克中心预测结果的可视化

Figure 3-3 The visualization of the predictions at Olympic Center station.

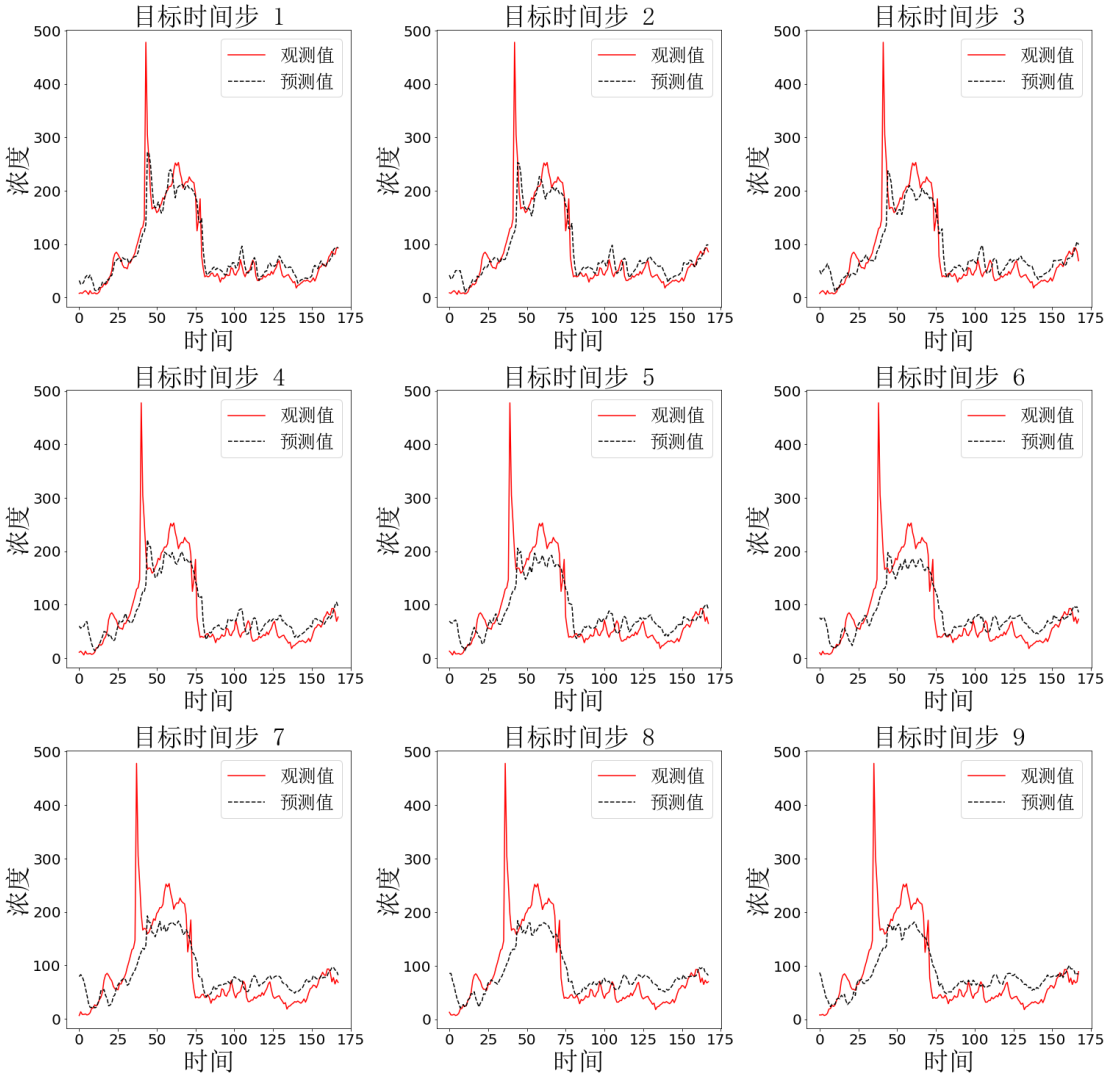


图 3-4 东四站预测结果的可视化

Figure 3-4 The visualization of the predictions at Dong Si station.

3.6 本章小结

这一章对比了应用不同激活函数的 GBELM，ELM 和 ANN 以及 SVM 算法在奥林匹克中心站和东四站的预测结果。结果表明 GBELM(softplus)有着最好的表现，而 GBELM(ReLU)有着稍弱一些的表现，但是其训练时间要明显少于 GBELM(softplus)。另外，改用 ReLU 和 softplus 作为激活函数可以提升 ELM 的性能，再加上梯度提升算法，可以进一步提升 ELM 的性能。GBELM(softplus)除了在较大突变时无法给出精确地预测，其他情况下都能给出可信的预测。ELM 的训练速度在这个问题上没有体现，反而比 ANN 训练速度慢，因为数据较多时，ELM 涉及到求大型矩阵的逆，这是个十分耗时的工作。

第4章 基于序列到序列模型的空气质量预测

基于非深度学习的空气质量预测方法虽然能够达到还不错的精度，但是想要进一步提升预测精度变得非常困难，主要是非深度学习对于非线性的拟合能力有限，想要进一步挖掘空气质量数据中的非线性关系需要借助深度学习。虽然深度学习这样的神经网络方法常常出现过拟合的问题，但是目前有如 dropout 等很多有效的方法防治过拟合。在众多深度学习模型中 Seq2Seq 是非常适合空气质量预测预测任务的，因为其将一个序列转化为另一个序列的特性十分符合空气质量预测的特点，即将通过空气质量数据的历史序列对未来一段时间的空气质量预测做出预测。原始的 Seq2Seq 将输入序列编码为一个上下文向量的方式不能将所有有用的信息全部编码，因此为了更好的利用序列信息，AM 被用来计算上下文向量。AM 可以将所有编码器隐藏层状态考虑到上下文向量中，并且给每个编码器隐藏层状态不同的权值。因此，使用 AM 的 Seq2Seq 能够取得更好的结果。另外 Seq2Seq 的训练速度很慢，其主要原因在于编码器和解码器的 RNN 无法并行计算。因此本文采用全连接层代替编码器的 RNN，同时为了弥补全连接层对序列信息提取能力有限的缺陷，加入位置编码以加强全连接编码器对序列信息的提取。对于解码器，除了无法并行之外，其递归式的预测方式会产生误差积累。因为每一个时间步的预测都存在误差，所以将预测结果输入下一个时间步就会引入误差。为了减弱误差积累，本文使用 n-step 递归预测的方法，减少时间步数量，从而减少误差积累以提高预测精度。同时由于时间步的减少，解码器 RNN 的训练速度将会获得提升。

4.1 N-step AAQP

4.1.1 位置编码

位置编码(Position Embedding)[54]的引入是为了帮助全连接编码器提取输入序列中的信息。原本在 RNN 编码器中，每一个时刻的数据依次输入到 RNN 中并获得最后的上下文向量。然而全连接编码器不需要将每个时刻的数据依次输入，而是将每个时刻的数据同时输入编码器中编码，因此编码器不知道每个时刻数据的先后顺序，也就无法对时间序列信息进行提取。为了解决这个问题，位置编码被用来存储每个序列中每个时刻数据的绝对位置，位置编码和输入序列同时被输入到编码器中，编码器通过辨识不同的位置编码获取序列中每个时刻信息的正确位置。设一个输入数据的样本为：$\boldsymbol{x} = [\boldsymbol{x}_1, \ldots . \boldsymbol{x}_S]$，其中$\boldsymbol{x}_j \epsilon \mathrm{R}^N, S$表示序列长度。

输入序列的绝对位置被嵌入到了一个矩阵$\boldsymbol{p} = [\boldsymbol{p}_1, \ldots . \boldsymbol{p}_S]$，其中$\boldsymbol{p}_j \epsilon \mathrm{R}^N$。将$\boldsymbol{x}$和$\boldsymbol{p}$合并在一起后就可以得到编码器的输入：$\boldsymbol{e} = [\boldsymbol{x}_1 + \boldsymbol{p}_1, \ldots . \boldsymbol{x}_S + \boldsymbol{p}_S]$。矩阵$\boldsymbol{p}$不同于输入数据，它不是固定的，而是一个可以调整的参数，随着 AAQP 模型的训练，$\boldsymbol{p}$会不断被调整以适应当前任务，最终可以存储准确的位置信息。

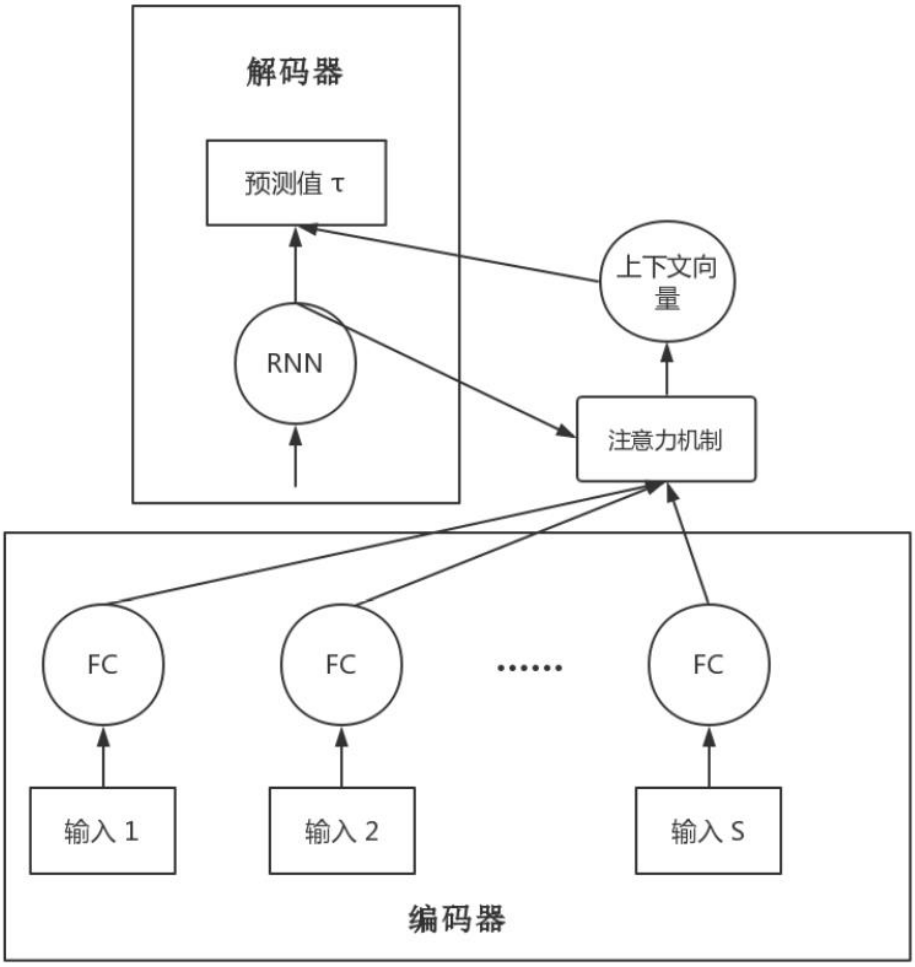


图 4-1　第 τ 个时间步的 n-step AAQP。

Figure 4-1 The illustration of τth time step of the n-step AAQP.

4.1.2　编码过程

因为 Seq2Seq 中的编码器和解码器都是 RNN 所以必须逐个时间步计算才能得到最终的预测结果。所以为了加速训练，首先使用了全连接层代替了编码器的 RNN，但是全连接层无法捕捉序列中的序列信息，因此还引入了位置编码。位置编码是一个与输入序列有着同样维度的变量，它被每个样本共享。位置编码的每一行学习并存储输入序列每个时刻的序列信息。由此编码器的隐藏层状态由如下

方式计算：

$$\bar{\boldsymbol{h}}_s = \tanh(\boldsymbol{W}_e * (\boldsymbol{x}_s + \boldsymbol{PE}_s) + \boldsymbol{b}_e) \tag{4-1}$$

其中$\bar{\boldsymbol{h}}_s$是隐藏层的状态，$\boldsymbol{W}_e$和$\boldsymbol{b}_e$是编码器的权重和偏置，并且由每个时间步共享。$\boldsymbol{PE}_s \in R^Q$是第$s$个时间步的位置编码。注意这里的激活函数使用的是 tanh，要和解码器的激活函数保持一致。因为是全连接层，而且参数由不同的时间步共享，因此可以同时将所有时间步的隐藏层状态计算出来，比使用 RNN 更加快速且高效。位置编码也许无法完全准确地存储所有的序列信息，但是输入序列的序列信息的重要程度相对输出序列要低。

4.1.3　解码过程

解码器的 RNN 被保留了，因为输出序列的序列信息要比输入序列的序列信息更加重要，因为在有了上一个时间步的预测结果再对当前时间步的结果进行预测会有增加很多信息，而如果采用直接预测的方式便不一定可以获取这些信息。然而，由于编码器采用了全连接结果，所以如果还使用编码器最后一个时间的隐藏层状态作为上下文向量则会丢失前面所有时间步的信息。因此本文使用了 AM 来计算上下文向量。AM 除了可以用来计算上下文向量，其另一个重要用途是可以给不同时间的解码器隐藏层状态不同的权值，因为每个编码器的隐藏层状态实际上对编码结果的影响不同。AM 则可以给影响较大的编码器隐藏层状态更大的权重，而给与影响较小的编码器隐藏层状态较小的权重。另外，解码器还采用了 n-step 循环预测，用来减少训练时间的同时减少随着时间的推移造成的误差积累。其中，n 的选取应该可以被目标序列的长度整除。在应用 n-step 循环预测之后，解码器的时间步变为了T/n。

本文使用了由 M. Luong 等[55]提出的改进版本的的 AM。使用 AM 的第τ个时间步的编码器展示在图 4-1 中。在使用 AM 之前，需要先计算解码器隐藏层的状态：

$$\boldsymbol{u}_\tau = \sigma(\boldsymbol{W}_u * [\boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}] + \boldsymbol{b}_u) \tag{4-2}$$

$$\boldsymbol{r}_\tau = \sigma(\boldsymbol{W}_r * [\boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}] + \boldsymbol{b}_r) \tag{4-3}$$

$$\tilde{\boldsymbol{h}}_\tau = \tanh(\boldsymbol{W}_h * [\boldsymbol{r}_\tau * \boldsymbol{h}_{\tau-1}, \boldsymbol{p}_{\tau-1}] + \boldsymbol{b}_h) \tag{4-4}$$

$$\boldsymbol{h}_\tau = (1 - \boldsymbol{u}_\tau) * \boldsymbol{h}_{\tau-1} + \boldsymbol{u}_\tau * \tilde{\boldsymbol{h}}_\tau \tag{4-5}$$

为了给不同的编码器隐藏层状态赋予不同的权重，需要使用一个量度来衡量编码

器隐藏层状态的重要程度，给重要程度高的状态赋予更高的权重。这个度量的定义如下：

$$\text{measure}\left(\boldsymbol{h}_{\tau}, \overline{\boldsymbol{h}}_{s}\right) = \boldsymbol{h}_{\tau}^{T}\overline{\boldsymbol{h}}_{s} \tag{4-6}$$

其中$\boldsymbol{h}_{\tau}$是解码器隐藏层的状态，$\overline{\boldsymbol{h}}_{s}$是编码器隐藏层的状态。这个度量本质上是编码器与解码器隐藏层状态的内积，换句话说，如果编码器的隐藏层状态和解码器的隐藏层状态越接近则编码器的隐藏层状态可以获得越高的权重。权重之和应当等于 1，显然仅仅定义了度量还不能保证权重的和为 1，因此为了保证所有的每个时间不的权重之和等于 1，还需要使用 softmax 函数对其归一化：

$$\boldsymbol{a}_{\boldsymbol{\tau}} = \frac{exp(\text{measure}\left(\boldsymbol{h}_{\tau}, \overline{\boldsymbol{h}}_{s}\right))}{\sum_{s'=1}^{S} exp(\text{measure}\left(\boldsymbol{h}_{\tau}, \overline{\boldsymbol{h}}_{s'}\right))} \tag{4-7}$$

$\boldsymbol{a}_{\boldsymbol{\tau}}$则是权重向量。上下文向量$\boldsymbol{c}_{\tau}$则是通过$\boldsymbol{a}_{\boldsymbol{\tau}}$每个编码器隐藏层的状态加权得到的：

$$\boldsymbol{c}_{\tau} = \sum_{s} \boldsymbol{a}_{\tau,s}\overline{\boldsymbol{h}}_{s} \tag{4-8}$$

最终上下文向量和编码器隐藏层的状态被用来计算最终的预测结果：

$$\boldsymbol{p}_{\tau} = \boldsymbol{W}_{p} * [\boldsymbol{h}_{\tau}, \boldsymbol{c}_{\tau}] + \boldsymbol{b}_{p} \tag{4-9}$$

4.1.4 模型训练

MSE 依然被用作 AAQP 的损失函数，但是需要将输出结果的每一项都纳入损失函数的计算，也就是说每一个时间步的输出结果都要和真实值进行损失函数的计算。在训练深度学习模型时一般采取小批量梯度下降，所以对于一个小批量的数据，其损失函数由如下公式计算：

$$\text{loss} = \frac{1}{m}\sum_{m=1}^{M}\sum_{\tau=1}^{T/n} ||\boldsymbol{p}_{\tau}^{m} - [\boldsymbol{y}_{n(t-1)+1}^{m}, \cdots\cdots, \boldsymbol{y}_{n*t}^{m}]||^{2} \tag{4-10}$$

其中m为一个小批量数据中的样本数量。最终可以使用梯度下降算法调整模型中的参数来最小化损失函数，而梯度下降所用到的梯度可以使用反向传播算法或者自动微分工具计算。AAQP 完整的训练过程被展示在 Algorithm 2。当有数据需要被预测时，执行 Algorithm 2 外层循环的(A)步和(B)步即可。

Algorithm 1. Attention-Based Air Quality Predictor

输入：数据集 D，预测步数 n

初始化所有参数，包括位置编码

对每一个 mini-batch 执行：

(A)计算所有编码器的隐藏层状态：$\overline{\boldsymbol{h}}_s = \tanh(\boldsymbol{W}_e * (\boldsymbol{x}_s + \boldsymbol{PE}_s) + \boldsymbol{b}_e)$

(B)对于 $\tau = 1{:}T/n$：

(a)计算隐藏层的状态$\boldsymbol{h}_\tau$ 可以直接使用公式 (4-2)-(4-5) 也可是使用 LSTM 版本的(4-2)-(4-5)

(b)对于每一个编码器的隐藏层状态，计算度量：$\text{measure}(\boldsymbol{h}_\tau, \overline{\boldsymbol{h}}_s) = \boldsymbol{h}_\tau{}^T \overline{\boldsymbol{h}}_s$

(c)使用公式(4-7)归一化分数向量

(d)根据权重向量$\boldsymbol{a}_\tau$对编码器隐藏层状态球加权和从而得到上下文向量$\boldsymbol{c}_\tau$。

(e)求得预测结果:$\boldsymbol{p}_\tau = \boldsymbol{W}_p * [\boldsymbol{h}_\tau, \boldsymbol{c}_\tau] + \boldsymbol{b}_p$

(D)利用梯度下降算法调整模型中的参数使得损失函数最小化

4.2 实验设置

4.2.1 数据集

实验所用数据集与第 3 章一致。本文使用了递归预测的方式，所以在在所有使用递归预测的模型中不仅 PM2.5 作为输出还包括SO_2，CO，NO_2，O_3，PM10。因为其他污染物的数据对于预测 PM2.5 是至关重要的。增加模型的输出项会影响到训练的过程，因为 Seq2Seq 的优化过程是针对所有的输出项而言的。因此增加的输出项不宜过多否则会产生反效果。另外污染物的信息虽然可以通过 RNN 的隐藏层状态向下一个时间步传递，但是 RNN 本身拥有遗忘的性质，RNN 也许会因为 cell 无法存储更多的数据从而选择遗忘一些相对不重要的信息，所以为了保证信息不丢失，还是将污染物作为输出，并直接输入到下一个时间步。因此目标序列的每个样本的是一个 24×6 的矩阵。然而对于采用使用递归预测的模型也只对其中 PM2.5 的预测结果进行分析。在北京 35 个空气质量监测站中，选用奥林匹克中心站和东四站作为目标站点。因为这两个站点人流较大，同时其数据完整性好。2017 年 4 月 1 日 0 时至 2018 年 3 月 31 日 23 时的数据将作为训练数据。2018 年 4 月 1 日 0 时至 2018 年 4 月 7 日 23 时的数据作为测试数据，因为这段时间的空气质量数据波动较大，适合用于测试模型性能。

另外，由于使用了 48 小时的天气数据和 24 小时的污染无数据，LSTM 和 GRU 需要对这些数据进行特殊处理，如果处理方法不合理会导致性能下降，因

此为了保证公平对比，本文先在不使用天气预报的情况下将所有方法进行对比

4.2.2 模型评价标准

模型评价标准采用与第 3 章相同的标准即平均绝对误差 MAE 和拟合优度R^2。

4.2.3 训练模型

模型超参数设定如下：

隐藏层节点	128
优化器	Adam
Batch size	512
学习速率	0.001
梯度截断	5
Dropout	0.5
epoch	1000(Seq2Seq 系列)/100(GRU,LSTM)
Teacher forcing	否

SVM 和 ANN 的超参数使用 gridsearch 选取。

4.3 预测精度对比

表 4-1　奥林匹克中心站 MAE

Table 4-1 MAE of Olympic Center Station

方法	4	8	12	16	20	24	总体
ANN	22.15	34.11	40.20	43.64	44.18	45.17	38.24
SVM	26.28	34.55	41.10	44.92	43.267	41.71	38.64
GRU	28.45	37.99	41.99	45.05	42.09	41.36	39.49
LSTM	30.26	37.28	41.32	44.48	43.26	42.27	39.81
Seq2Seq (GRU)	21.32	33.59	36.21	40.31	44.06	46.53	37.00
Seq2Seq (LSTM)	19.02	29.52	36.71	44.33	45.25	49.00	37.50
Seq2Seq-mean	18.62	28.61	32.20	37.05	39.05	43.23	33.11
Seq2Seq-mean	21.82	32.63	36.19	35.45	**36.28**	41.24	33.94
Seq2Seq-attention	19.12	**28.57**	**31.97**	**32.76**	40.62	46.71	**33.31**
Seq2Seq-attention	18.53	31.36	36.18	37.78	36.91	40.12	33.48
AAQP (GRU)	**17.81**	30.99	35.62	37.50	39.70	39.70	33.55
AAQP (LSTM)	20.42	29.94	35.73	38.47	39.69	**38.13**	33.73

表 4-2　东四站 MAE

Table 4-2 MAE of Dong Si Station

方法	4	8	12	16	20	24	总体
ANN	40.53	52.58	55.54	57.18	67.49	61.07	55.73
SVM	56.43	67.62	67.73	68.12	67.59	66.30	65.63
GRU	34.96	45.85	46.76	45.66	48.73	54.61	46.09
LSTM	33.02	43.03	49.59	53.08	55.89	55.07	48.28
Seq2Seq (GRU)	28.47	42.46	46.94	55.17	60.44	57.21	48.45
Seq2Seq (LSTM)	27.46	40.93	45.80	50.62	54.17	59.52	46.10
Seq2Seq-mean	26.84	38.75	48.74	51.13	50.88	55.07	46.23
Seq2Seq-mean	26.27	44.70	58.63	66.13	67.90	63.70	54.56
Seq2Seq-attention	26.89	42.77	50.41	47.52	49.09	54.95	45.27
Seq2Seq-attention	24.89	37.85	48.57	57.51	58.46	51.32	46.44
AAQP (GRU)	**24.32**	40.23	**44.48**	**46.70**	**46.85**	**46.20**	**41.46**
AAQP (LSTM)	25.19	**33.65**	41.33	48.29	50.16	53.28	41.98

表 4-3　奥林匹克中心站 R^2

Table 4-3 R^2 of Olympic Center Station

方法	4	8	12	16	20	24	总体
ANN	0.657	0.280	0.081	-0.101	-0.112	-0.291	0.086
SVM	0.609	0.274	0.045	-0.121	-0.097	-0.077	0.105
GRU	0.419	0.069	-0.339	-0.622	-0.378	-0.341	-0.193
LSTM	0.276	0.035	-0.171	-0.400	-0.439	-0.393	-0.177
Seq2Seq (GRU)	0.607	0.143	-0.026	-0.342	-0.500	-0.496	-0.102
Seq2Seq (LSTM)	0.648	0.431	0.195	-0.253	-0.369	-0.393	0.043
Seq2Seq-mean	0.692	0.454	0.395	0.178	-0.035	-0.334	0.225
Seq2Seq-mean	0.680	0.388	0.353	0.322	-0.199	-0.669	0.246
Seq2Seq-attention	0.692	**0.475**	**0.437**	**0.401**	-0.051	-0.433	0.253
Seq2Seq-attention	**0.729**	0.403	0.275	0.094	0.112	-0.101	**0.252**
AAQP (GRU)	0.710	0.251	0.169	0.158	0.031	-0.036	0.213
AAQP (LSTM)	0.678	0.358	0.260	0.122	**0.009**	**-0.007**	0.237

表 4-4　东四站 R^2

Table 4-4 R^2 of Dong Si Station

方法	4	8	12	16	20	24	总体
ANN	0.444	0.158	0.058	-0.016	-0.341	-0.250	0.008
SVM	0.111	-0.186	-0.226	-0.266	-0.254	-0.197	-0.170
GRU	0.411	0.247	0.178	0.104	-0.028	-0.100	0.143
LSTM	0.340	0.094	-0.041	-0.170	-0.372	-0.345	-0.071
Seq2Seq (GRU)	0.475	0.029	0.036	-0.203	-0.290	-0.161	-0.018
Seq2Seq (LSTM)	0.507	0.273	0.078	-0.391	-0.383	-0.226	-0.080
Seq2Seq-mean	0.649	0.431	-0.117	-0.188	-0.095	-0.290	0.065
Seq2Seq-mean	0.563	0.107	-0.362	-0.535	-0.617	-0.568	-0.235
Seq2Seq-attention	0.574	0.224	-0.008	0.034	-0.094	-0.288	0.073
Seq2Seq-attention	0.623	0.354	0.029	-0.320	-0.321	-0.044	0.053
AAQP (GRU)	0.650	0.298	0.195	**0.116**	**0.056**	**0.056**	**0.228**
AAQP (LSTM)	**0.663**	**0.492**	**0.308**	0.065	-0.085	-0.229	0.202

表 4-1 至 4-4 展示了在不同站点使用不同方法预测结果的 MAE 和 R^2，其中 AAQP 算法使用的是 1-step，Seq2Seq-mean 是利用编码器隐藏层状态均值作为上下文向量。从这几个表中可以发现，使用 AM 的预测结果比使用编码器最后一个

时间步的隐藏层状态作为上下文向量的结果好，并且简单使用编码器隐藏层状态做为上下文向量的预测结果也没有使用 AM 的效果好。由此可知使用编码器所有隐藏层状态计算的到的上下文向量能增加预测精度，同时给编码器隐藏层状态以不同的权重更有利于预测精度的提升。将输入数据和输出数据同时作为序列处理的 Seq2Seq 比只将输入数据作为序列处理的 LSTM 和 GRU 的预测结果更为准确。LSTM 和 GRU 在奥林匹克中心站的预测结果却没能比均未将输入和输出作为序列处理的 SVM 和 ANN 要好，但是在东四站，LSTM 和 GRU 的预测结果明显好于 SVM 和 ANN。所以原因可能在于奥林匹克中心站的输出数据和输入数据之间的非线性关系不是非常的复杂，所以 LSTM 和 GRU 没有展现出强大的非线性拟合能力，但是东四站输入数据与输出数据之间的非线性关系非常复杂，而深度学习算法非常擅长拟合复杂的非线性关系，所以更加具有优势。

具体地，在奥林匹克中心站，由于其 PM2.5 的波动不是非常大，所以比较容易预测，几种方法的性能差距不是非常大。使用全部编码器隐藏层状态的六种方法的精度相近，但是如果具体看每 4 小时的预测精度可以发现，使用 AM 的方法在前 8 个小时 MAE 要小于使用隐藏层状态均值的方法，并且前 16 小时的 R^2 也比使用隐藏层状态均值的方法要大。越在前面输出时刻做出的预测精度越高，因此越是可以信任的预测，所以具体分析来看还是使用 AM 在奥林匹克中心站的表现要好。1-step AAQP 作为简化的版本的 Seq2Seq 模型，其预测精度与 Seq2Seq 相当，因此这种简化是完全合理的。在东四站，PM2.5 的波动变得很大，因此相较奥林匹克中心站其预测难度更大。AAQP(LSTM)在前 12 个小时的预测有最高的精度，因此其表现是最好的，同时 AAQP 比原始版本的 Seq2Seq 表现的更好，所以这个简化不仅没有将降低预测精度反而可以拟合更加复杂曲线，因此这是一个成功的简化。

尽管 GRU 和 LSTM 可以提取序列信息，但是这两种方法在奥林匹克中心站的表现不如 SVM 和 ANN。这个事实也说明了 SVM 和 ANN 虽然没有将输入数据和输出数据作为序列处理，其本身还是具备一定的提取序列信息的能力，但是显然，在提取复杂序列关系的时候， SVM 和 ANN 的提取能力还是没有 GRU 和 LSTM 强大。因此 AAQP 使用全连接编码器本身就可以在一定程度上提取一些序列信息，只是如果预测复杂的序列关系仅仅依靠全连接编码器是不够的，所以在此基础上的加入的位置编码也帮助了 AAQP 提升了很多提取序列信息的能力，甚至在东四站，AAQP 的表现强于 Seq2Seq。这也许是目前很多研究者选择在构建 Seq2Seq 模型时不选用 RNN 的原因，除了可以减少训练时间还可以增加精度。所以从这个结果可以总结出利用全连接层代替 RNN 编码器再利用位置编码存储序列信息在空气质量预测中是可行的。

4.4 使用不同 step 的 AAQP

表 4-5 奥林匹克中心站 MAE

Table 4-5 MAE of Olympic Center Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	18.68	28.38	**34.87**	40.11	41.93	44.46	34.74
	2	19.73	29.90	35.52	37.48	**36.63**	**34.35**	**32.26**
	3	20.65	31.24	36.02	**36.97**	37.07	40.03	33.66
	4	20.31	31.54	37.69	42.78	40.98	42.15	35.91
	6	18.17	29.33	37.63	38.87	37.71	37.85	33.26
	12	**15.60**	**28.07**	35.42	42.70	43.82	43.30	30.76
AAQP (LSTM)	1	26.39	39.07	44.55	47.75	47.44	47.67	35.26
	2	19.79	27.55	**33.18**	37.31	**36.40**	**35.33**	31.59
	3	20.11	32.32	36.63	38.55	39.14	35.79	33.76
	4	18.40	29.20	33.61	37.75	40.02	38.77	32.96
	6	17.65	**27.26**	33.90	**35.87**	36.42	35.87	**31.16**
	12	**15.41**	28.95	35.28	43.12	44.43	43.45	35.11
Best	—	18.53	28.38	**30.30**	**31.59**	36.91	40.12	32.85

表 4-6 东四站 MAE

Table 4-6 MAE of Dong Si Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	24.32	40.24	44.49	46.70	46.85	46.21	41.47
	2	26.92	38.55	43.93	44.25	44.85	49.37	41.31
	3	25.03	33.94	40.25	47.01	45.81	47.40	39.91
	4	21.65	34.96	44.38	46.42	48.05	46.68	40.36
	6	23.54	36.79	41.82	46.55	**44.21**	**44.18**	39.51
	12	**19.44**	**32.87**	**38.82**	**41.80**	47.59	49.97	**38.41**
AAQP (LSTM)	1	25.20	33.66	41.33	48.30	50.17	53.28	41.99
	2	25.21	33.97	41.98	43.68	43.44	46.72	39.17
	3	24.44	36.12	42.46	44.39	43.91	**43.91**	39.21
	4	21.77	33.08	**37.40**	**41.67**	43.83	46.69	**37.41**
	6	22.20	32.74	39.94	42.00	**43.27**	45.33	37.58

	12	**20.51**	**31.53**	39.16	42.09	46.00	49.89	38.20
Best	—	24.32	37.86	44.49	46.70	46.85	46.21	41.47

表 4-7　奥林匹克中心站 R^2

Table 4-7 R^2 of Olympic Center Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	0.703	**0.444**	**0.275**	-0.022	-0.211	-0.351	0.139
	2	0.682	0.370	0.198	0.126	0.152	**0.233**	**0.294**
	3	0.663	0.354	0.221	**0.173**	0.090	-0.103	0.233
	4	0.657	0.282	0.132	-0.090	-0.075	-0.229	0.112
	6	0.694	0.391	0.106	0.105	**0.167**	0.178	0.273
	12	**0.718**	0.402	0.240	-0.050	-0.117	-0.138	0.175
AAQP (LSTM)	1	0.624	0.327	0.171	0.078	**0.121**	0.070	0.049
	2	0.688	**0.438**	0.235	0.027	0.049	0.124	0.260
	3	0.680	0.244	0.203	0.089	0.039	**0.234**	0.248
	4	0.712	0.377	**0.261**	0.064	-0.094	-0.016	0.217
	6	0.716	0.427	0.230	**0.217**	0.093	0.083	**0.294**
	12	**0.740**	0.395	0.204	-0.106	-0.204	-0.193	0.139
Best	—	0.729	**0.475**	**0.437**	**0.401**	0.121	0.070	0.253

表 4-8　东四站 R^2

Table 4-8 R^2 of Dong Si Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	0.650	0.298	0.195	0.116	0.056	0.056	0.228
	2	0.651	0.376	0.260	**0.171**	**0.196**	0.018	0.279
	3	0.668	0.453	0.294	0.001	0.085	0.073	0.262
	4	0.708	0.444	0.211	0.001	-0.034	0.027	0.226
	6	0.690	0.415	0.219	0.089	0.116	**0.081**	0.269
	12	**0.718**	**0.518**	**0.362**	0.144	0.001	-0.092	**0.275**
AAQP (LSTM)	1	0.663	0.492	0.308	0.065	-0.085	-0.229	0.202
	2	0.635	0.412	0.228	0.150	0.135	-0.003	0.259
	3	0.665	0.360	0.242	0.121	**0.170**	**0.188**	0.291
	4	**0.687**	0.424	**0.350**	0.233	0.091	0.022	0.301

	6	0.678	0.482	0.284	0.190	0.156	0.102	0.315
	12	0.685	**0.513**	0.309	**0.240**	0.150	0.001	**0.316**
Best	—	0.650	0.492	0.308	0.116	0.056	0.056	0.228

使用不同 step 的 AAQP 的预测结果展示在了表格 4-5 至表格 4-8。带下划线的数字代表当前站点中最佳的预测结果，加粗字体表示当前方法不同 step 中表现最好的。“Best”一行表示在 5.1 在当前站点中表现最好结果，只有这一行是整个监测站表现最好时才会标红。从全局看，不同 step 之间的表现比较接近，但是仍然可以看到 12-step AAQP 的表现更好一些，因为其在前面 4 个小时的输出结果要好于其他方法，而 5-8 个小时的预测结果也好于其他方法或者接近最优的结果。另外，几乎所有最优的结果都是由 n-step AAQP 得到的，所以使用 n-step 递归预测的方法的确提升了 AAQP 的预测能力。另外，从结果中可知 n 的取值和预测精度没有单调的关系，也就是说随着 n 的增加预测精度不一定会增加也不一定会降低，在具体使用时需要进行实验来确定 n 的值。不过，大致上还是可以看到较大的 n 能够使得前面 8 个小时的预测结果精度较高。从第 9 个小时的预测开始，小一点的 n，在奥林匹克中心站的预测更有优势，而大一些的 n 在东四站更有优势。原因还是在于奥林匹克中心站和东四站目标序列的那个程度。奥林匹克中心站的波动较小，加上每一个解码时间步的输出项少所以精度较高，因此在解码时时间步变多也不会对预测精度产生很大的负面影响，反而是 n 变大后使得每一次预测变得精度较差。这一点可以从 12-step AAQP 的结果中看出，无论是应用 LSTM 还是 GRU，在奥林匹克中心站的前 12 个小时预测结果很好，但是之后的 12 小时的预测性能出现断崖式下降，可以明显地看出 R^2 变成了负值。相反，东四站的目标序列波动较大，难以预测。较小的 n 会使得解码时间步变多，而每一步预测的精度又较差，所以误差会非常容易积累，导致最终预测结果很差。然而，较大的 n 使得解码时间步变少，加上本来就难以预测，因此解码器输出项变多对精度的影响小，所以误差积累较少，因此最终预测结果也较好。这一点也可以从 12-step AAQP 的预测结果看出，不同于奥林匹克中心站，东四站的预测性能在后 12 个小时没有出现断崖式下降，而是平稳下降。综上所述，如果目标序列的波动较大，则选取一个较大的 n 能够取得更好的结果。但是如果目标序列的波动较小，可以考虑选取稍小一点的 n，虽然奥林匹克中心站的 12-step AAQP 在前 4 个小时的预测结果最好，但是事实上，6-step AAQP 在前 4 个小时的预测结果与 12-step AAQP 没有十分明显的差距，而且在 5-8 小时的预测比 12-step AAQP 更准确。

4.5 时间对比

表 4-9　训练和预测时间

Table 4-9 Training and Testing Time

方法	训练时间（秒）	预测时间（秒）
Seq2Seq (GRU)	392.832	2.378
Seq2Seq (LSTM)	489.456	3.944
Seq2Seq-mean (GRU)	419.904	2.711
Seq2Seq-mean (LSTM)	776.880	3.013
Seq2Seq-attention (GRU)	1577.952	5.520
Seq2Seq-attention (LSTM)	2112.624	5.673
1-step AAQP (GRU)	1239.408	3.576
1-step AAQP (LSTM)	1080.156	4.405
2-step AAQP (GRU)	843.696	1.333
2-step AAQP (LSTM)	890.496	1.163
3-step AAQP (GRU)	663.699	1.295
3-step AAQP (LSTM)	645.264	0.936
4-step AAQP (GRU)	519.735	1.160
4-step AAQP (LSTM)	519.840	0.756
6-step AAQP (GRU)	430.272	0.679
6-step AAQP (LSTM)	435.168	0.612
12-step AAQP (GRU)	**338.544**	0.888
12-step AAQP (LSTM)	341.136	**0.415**

表 4-9 展示了本章所有方法的训练时间和预测时间，其中时间是两个监测站所用时间的平均值。Seq2Seq-Attention 的训练时间和预测时间都是最多的，无论是使用 GRU 还是 LSTM 都需要大量时间训练和超过 5 秒的时间预测。其中一个原因是解码时间步多，逐一解码需要消耗很多时间。而与其有着相同解码时间步的原始 Seq2Seq 所用的训练时间和预测时间都明显少于 Seq2Seq-Attention，其中训练时间仅是 Seq2Seq-Attention 的四分之一，因为原始版本的 Seq2Seq 无需执行额外的计算上下文向量的操作，其上下文向量直接选用编码器最后一个时间步的隐藏层状态。Seq2Seq-mean 的训练时间和预测时间也同样明显少于 Seq2Seq-Attention，因为它采用的是编码器隐藏层状态的均值作为上下文向量，而且这个上下文向量对预测每个解码时间步而言都是一样的，因此只需要计算一

次。然而对于 Seq2Seq-Attention，对于不同的解码时间步，都需要执行一次计算上下文向量的计算，即 AM 的操作，因为使用 AM 计算的上下文向量对于不同的解码器时间步是不一样，因此需要在每个解码时间不都计算一次，大量地消耗了时间。AAQP 使用全连接层代替了 RNN 编码器后，训练时间和预测时间都有了明显的减少，但是依然需要 2-3 倍于原始 Seq2Seq 的训练时间。然而应用 AM 的 Seq2Seq 主要消耗时间在于解码过程，所以在使用 n-step 递归预测后，训练时间又有了大幅减少，在使用 6-step 递归预测时，训练时间已经和原始 Seq2Seq 接近了。原因是 n-step 递归预测可以减少解码时间步，在 n=6 时只需要 4 个解码时间步就可以得到计算结果，同时减少了执行 AM 的次数从而大幅减少了训练时间和预测时间。而在 12-step 递归预测时，训练时间甚至少于原始 Seq2Seq，同时预测的精度又明显高于原始 Seq2Seq，因此在实际使用时，大量训练模型时可以大幅度减少所需时间，使得应用尽快上线。AAQP 中预测时间最长的也只有 4 秒，即便在实际使用也可以在几分钟内获得大量监测站的预测结果，最短的不到 1 秒即可获得预测结果，因此在实际应用中完全可以应用于大量监测站的建模和预测。

4.6 结果可视化

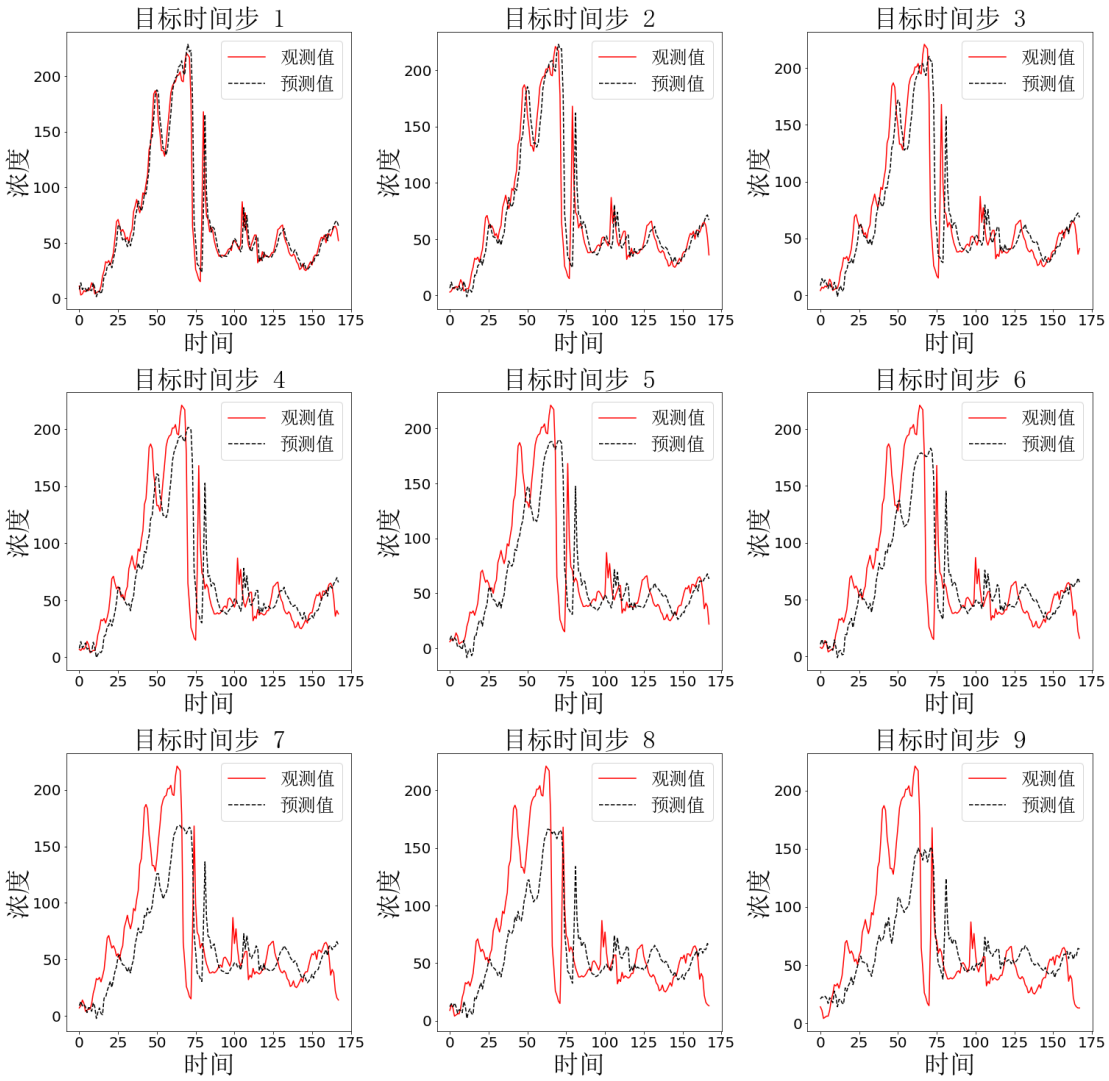


图 4-2　奥林匹克中心站预测结果的可视化

Figure 4-2 The visualization of the predictions at Olympic Center station.

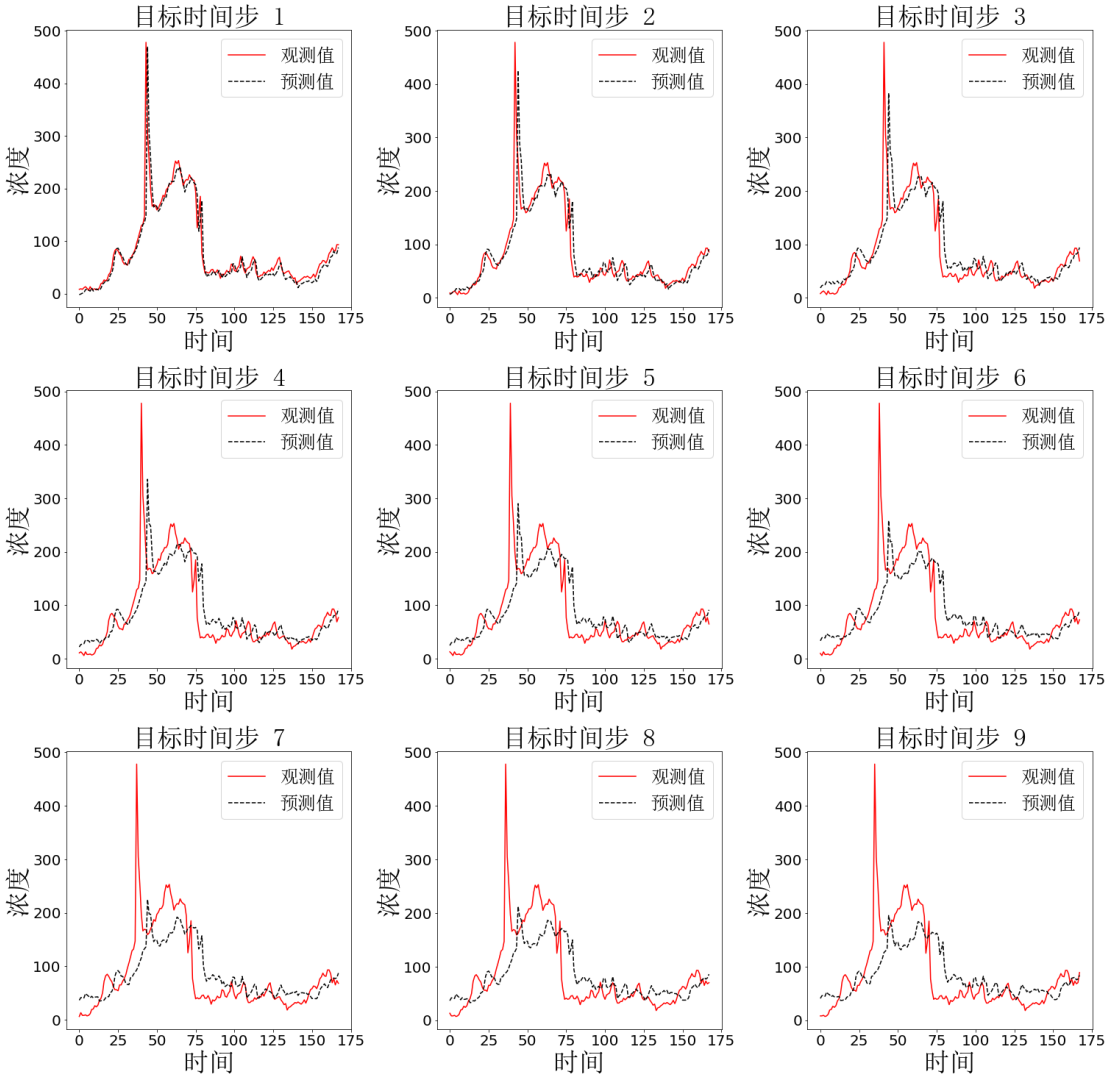


图 4-3　东四站预测结果的可视化

Figure 4-3 The visualization of the predictions at Dong Si station.

图 4-2 和图 4-3 是奥林匹克中心站和东四站的预测结果的可视化，每一个子图是对于未来 24 小时 PM2.5 预测结果中前 9 个小时每个小时的预测结果。奥林匹克中心站选取 12-step AAQP (LSTM)可视化，而东四站则选用 12-step AAQP(GRU)可视化，因为这两个方法能够在较前面的时刻给出更精确的预测，所以认定这两个方法是最好的。在奥林匹克中心站，模型第 1 个小时的预测结果和真实值十分接近，但是从第 2 个小时开始，预测结果相对于真实值出现了轻微滞后的现象。从第 3 个小时的预测开始，滞后变得严重更加严重，预测结果开始变得不可信。在东四站，出现了和奥林匹克中心站一样的情况，模型第 1 个小时的预测结果和真实值几乎一致，但是从第 2 个小时开始预测出现滞后的现象，并逐渐在后面的预测中变得更加严重。和 GBELM 一样，AAQP 也出现了滞后的现象，不过现在的 AAQP 还没有使用天气预报的信息。

4.7 本章小结

本章首先对比了 ANN，SVM，LSTM，GRU，和多种 Seq2Seq 模型以及 AAQP 的性能。实验结果中 1-step AAQP 的预测结果与 Seq2Seq-Attention 的结果相近或者更好。所以使用全连接编码器与位置编码的组合可以取代 Seq2Seq 中的 RNN 编码器。最后对使用不同 step 的 AAQP 进行了对比，发现使用不同的 step 的确可以提升 AAQP 的预测能力。

第5章 性能分析

本章主要对第 3 章 GBELM 的预测结果和第 4 章 n-step AAQP 的预测结果进行对比。GBELM 的预测结果采用其在第 3 章的结果。而 n-step AAQP 的预测结果是加入天气预报信息之后的结果。

5.1 预测精度对比

表 5-1　奥林匹克中心站 MAE

Table 5-1 MAE of Olympic Center Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	22.59	23.77	26.61	29.89	31.71	31.84	27.74
	2	**16.10**	19.95	21.18	22.65	24.72	25.12	21.62
	3	16.24	**19.25**	**21.07**	**21.59**	**22.32**	**22.02**	**20.41**
	4	19.83	24.22	23.58	24.38	25.87	28.63	24.42
	6	14.79	20.33	22.43	24.06	24.85	28.24	22.45
	12	**16.10**	20.29	23.68	24.62	23.89	27.84	22.74
AAQP (LSTM)	1	18.32	23.72	29.01	32.76	36.51	38.25	29.76
	2	22.28	22.85	23.44	24.60	29.13	31.19	25.58
	3	20.87	23.83	24.84	25.25	25.02	**26.16**	24.33
	4	18.34	**23.02**	26.34	28.00	28.32	28.74	25.46
	6	20.02	23.14	**22.77**	**23.43**	**24.79**	26.72	**23.48**
	12	**16.64**	21.72	22.92	25.20	27.99	29.92	24.07
GBELM	—	**14.97**	23.02	27.28	30.10	32.44	35.71	27.25

表 5-2　东四站 MAE

Table 5-2 MAE of Dong Si Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	23.63	30.16	38.69	43.04	45.98	49.40	38.48
	2	21.81	27.55	32.05	38.46	44.00	48.43	35.38
	3	22.84	30.49	30.64	33.26	38.34	42.99	33.09
	4	22.49	24.50	26.36	34.26	39.05	43.08	31.62
	6	22.10	27.23	26.45	**26.14**	**28.05**	**32.81**	**27.13**
	12	**21.59**	**23.12**	**25.87**	29.52	32.22	38.53	28.47
AAQP (LSTM)	1	21.52	27.33	33.76	39.52	45.94	54.24	37.05
	2	26.05	27.94	29.45	32.11	35.02	39.05	31.60
	3	21.05	**27.23**	31.49	37.36	43.06	48.02	34.70
	4	24.96	30.78	29.75	32.41	36.58	**38.28**	32.13
	6	24.28	28.15	**28.69**	30.71	35.08	39.12	31.00
	12	**20.86**	27.61	29.49	**30.68**	**33.75**	39.67	**30.34**
GBELM	—	22.53	31.04	34.15	37.81	41.02	42.92	34.91

表 5-3　奥林匹克中心站 R^2

Table 5-3 R^2 of Olympic Center Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	0.678	0.646	0.529	0.396	0.300	0.256	0.467
	2	0.793	0.732	**0.736**	0.710	0.621	0.612	0.701
	3	0.781	**0.749**	0.734	**0.738**	**0.706**	**0.693**	**0.734**
	4	0.717	0.628	0.668	0.606	0.536	0.442	0.599
	6	**0.797**	0.672	0.683	0.654	0.650	0.590	0.674
	12	0.785	0.704	0.633	0.625	0.612	0.521	0.647
AAQP (LSTM)	1	0.772	0.613	0.472	0.339	0.283	0.183	0.444
	2	0.700	0.664	0.626	0.588	0.418	0.315	0.552
	3	0.717	0.642	0.591	0.573	0.567	0.532	0.604
	4	0.757	0.664	0.545	0.418	0.412	0.439	0.539
	6	**0.793**	0.678	**0.716**	**0.598**	**0.582**	**0.584**	**0.658**
	12	0.775	**0.691**	0.647	0.557	0.512	0.456	0.607
GBELM	—	**0.808**	0.654	0.619	0.541	0.465	0.357	0.574

表 5-4　东四站 R^2

Table 5-4 R^2 of Dong Si Station

方法	steps	4	8	12	16	20	24	总体
AAQP (GRU)	1	0.662	0.588	0.317	0.111	0.124	0.079	0.314
	2	0.695	0.655	0.557	0.403	0.270	0.142	0.454
	3	0.667	0.566	0.536	0.449	0.331	0.183	0.455
	4	0.615	0.646	0.585	0.432	0.264	0.173	0.453
	6	0.663	0.599	0.651	**0.645**	**0.647**	**0.535**	**0.623**
	12	**0.739**	**0.702**	**0.660**	0.582	0.551	0.433	0.611
AAQP (LSTM)	1	**0.713**	0.570	0.369	0.293	0.192	-0.145	0.332
	2	0.666	**0.666**	0.572	0.471	0.452	0.169	0.499
	3	0.691	0.614	0.505	0.349	0.115	-0.050	0.371
	4	0.641	0.602	0.556	0.504	0.365	0.324	0.499
	6	0.651	0.598	0.561	**0.577**	0.457	0.352	0.533
	12	0.710	0.592	**0.597**	0.548	**0.492**	**0.409**	**0.558**
GBELM	—	0.728	0.596	0.536	0.491	0.420	0.319	0.515

表格 5-1 至表格 5-4 是增加天气预报后 AAQP 和 GBELM 的预测结果对比。其中 GBELM 采用的是第 4 章中最优的，也就是使用了 softplus 作为激活函数。首先，使用天气预报的信息后，AAQP 的性能有了很大的提升，所以如果有天气预报信息的情况下应该利用天气预报的信息。只看 AAQP，AAQP(GRU)的预测效果要好于 AAQP(LSMT)。

对于奥林匹克中心站，其中 3-step AAQP(GRU)的效果是最好的，而在 AAQP(LSTM)中，当 n 取 6 时效果是最好的。在前 4 小时的预测中，2-step AAQP(GRU)和 12-step AAQP(GRU)的 MAE 是最小的，6-step AAQP(GRU)的 R^2 是最大的，但是他们相比 3-step AAQP(GRU)在这两个指标上只有微弱的优势，而其他小时的预测上以 3-step AAQP(GRU)为最优的情况最多，并且其有着最好的总体表现。9-12 小时的预测 3-step AAQP(GRU)的 R^2 也不是最优的，但是其与最优的表现方法相比只有微弱劣势，所以在所有的 AAQP(GRU)中 n 取 3 的时候是最优的表现。3-step AAQP(GRU)前 4 小时的 R^2 比 6-stepAAQP(LSTM)要略差，但是差距非常小，而其余的表现无论是总体还是细节都比 AAQP(LSTM)要好，所以 3-step AAQP(GRU)在整个奥林匹克中心站是预测最精准的方法。

在东四站，12-step AAQP(GRU)的表现是最好的，虽然其第 13-24 小时的预

测结果与最优的 6-step AAQP(GRU)有略大的差距，并且在总体的预测效果上相比 6-stepAAQP(GRU)也有微弱的劣势，但是其前 12 小时预测的 MAE 和 R^2 相比 6-step AAQP(GRU)要有明显的优势，而前面 12 小时的预测精度大大高于后面 12 小时的，其可信度高，因此可以认为 12-step AAQP(GRU)的表现是最好的。与 AAQP(LSTM)相比，12-step AAQP(GRU)的 MAE 只有在最后 4 个小时的预测结果不如 AAQP(LSTM),而其他小的的预测结果中，无论是 MAE 还是 R^2 都比 AAQP(LSTM)高。因此所以 6-step AAQP(GRU)是东四站表现最好的方法。

表格中，1-step 的方法往往在后面小时的输出精度较差，而采用 2-step 的方法后，后面小时的输出精度都有所提高，这说明 n-step 的方法的确有助于减小误差的积累，达到比 1-step 递归预测更好的预测效果。但是随着 n 取值的增加，后面小时的输出精度不一定会提高，因为 n 的增加会导致每个解码时间步的输出项变多，从而导致预测精度下降，即便解码时间步的数量减少了，但是每一步的误差变大了，其误差还是会积累，因此在实际使用中需要进行实验来确定 n 的取值。加入天气预报后，两个站点的空气质量都变的更容易预测了，所以应当是较小的 n 更具有优势，但是事实上是只有奥林匹克中心站变成了更小的 n 更加具有优势。然而，东四站依然是 12-step AAQP 的表现最好，所以东四站的 PM2.5 可能并没有真的变得很好预测。

GBELM 的与结果在对比 AAQP 时，是处于劣势的，除了在奥林匹克中心站前 4 个小时的预测结果优于 AAQP 外，其余的预测结果比 AAQP 都要差。即便是前 4 个小时的预测结果也只是有非常小的优势，然而其他的预测往往比最优的方法有着较大的差距。因此 AAQP 的效果比 GBELM 好。GBELM 作为一种使用直接预测的方法，从其预测结果可以发现直接预测可以兼顾整体的预测效果，使得预测结果不会像递归预测一样由于误差的积累或导致后面预测结果很差，但是一次性预测全部时刻结果使得其总体的精度不够高。然而，n-step 预测则成功地兼顾递归预测和直接预测的优势，所以在所有方法中，表现最好的均为 n>1 时的 n-step 递归预测。

5.2 时间对比

表 5-5 展示了性能最优的两种 n-step AAQP 和 GBELM(softplus)的训练时间和预测时间。可以看到 GBELM 所用的训练时间相对 3-step AAQP 明显少，但是其预测精度较低。12-stepAAQP 的训练时间则少于 GBELM 的训练时间，因此 AAQP 在东四站不仅精度高于 GBELM，需要的训练时间也少。如果实际应用中 3-step AAQP 训练确实很耗时的情况下，可以加大 n 的取值，因为实验结果中奥

林匹克中心站 12-step AAQP 的预测精度也是好于 GBELM 的，并且两个监测站模型的输入输出以及参数设定是一致的，因此可以 12-step AAQP 在奥林匹克中心站的训练时间也是少于 GBELM 的。

表 5-5　训练和预测时间

Table 5-5 Training and Testing Time

方法	训练时间（秒）	预测时间（秒）
3-step AAQP (GRU)	701.675	1.193
12-step AAQP (GRU)	363.212	0.459
GBELM(softplus)	423.389	0.135

5.3 结果可视化

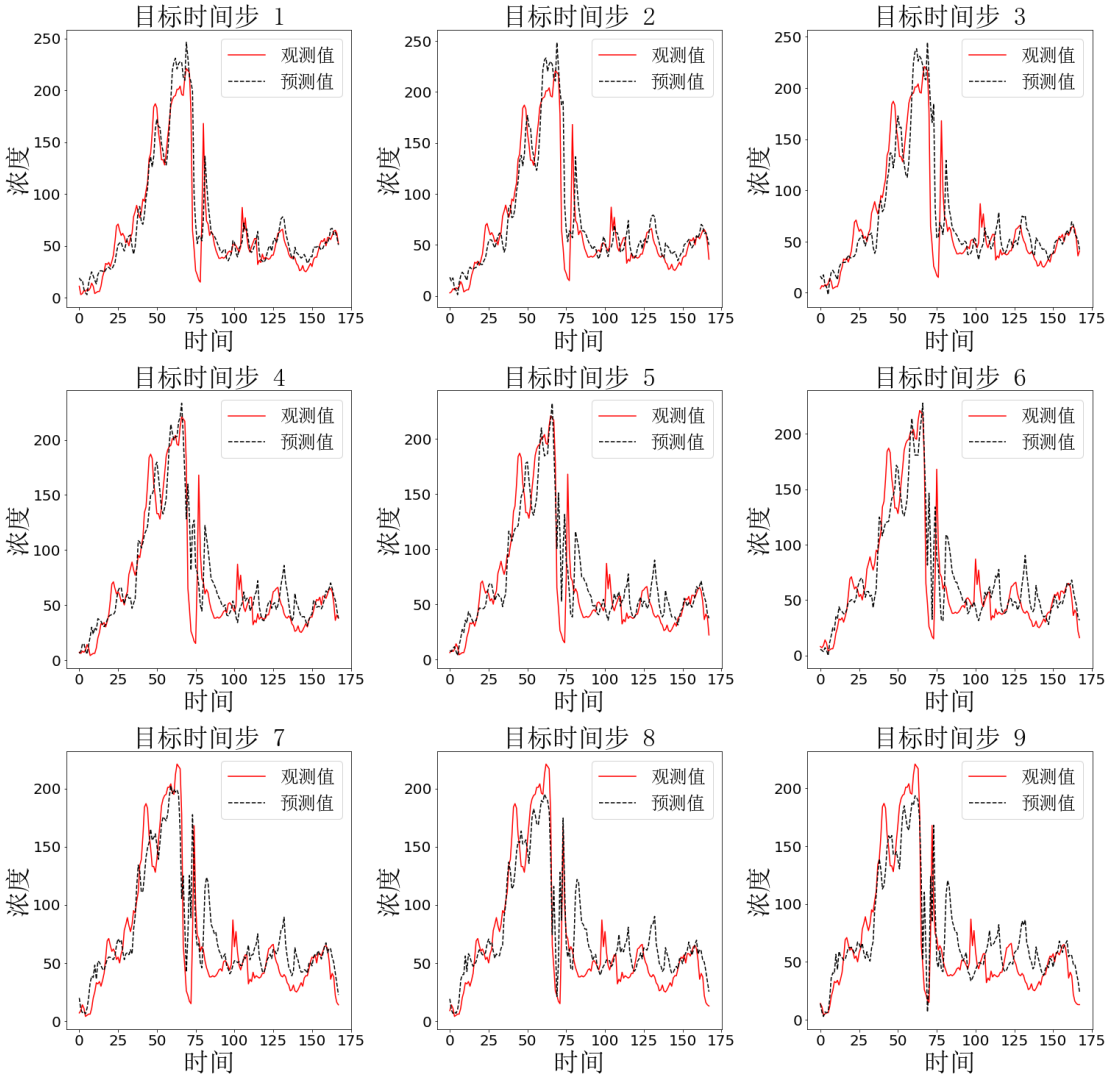


图 5-1　奥林匹克中心站 AAQP 预测结果的可视化

Figure 5-1 The visualization of the predictions of AAQP at Olympic Center station.

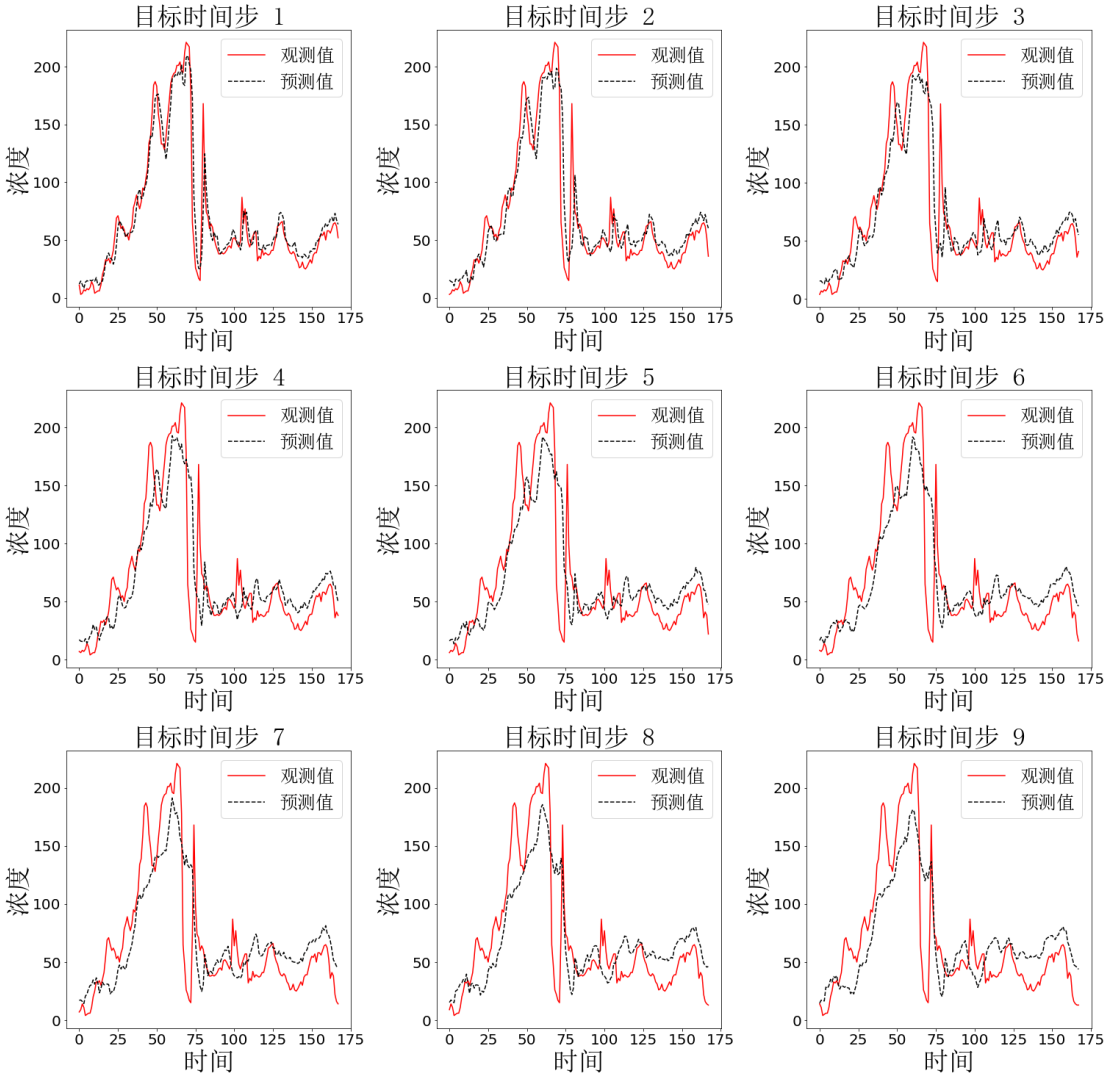


图 5-2　奥林匹克中心 GBELM 预测结果的可视化

Figure 5-2 The visualization of the predictions at Olympic Center station.

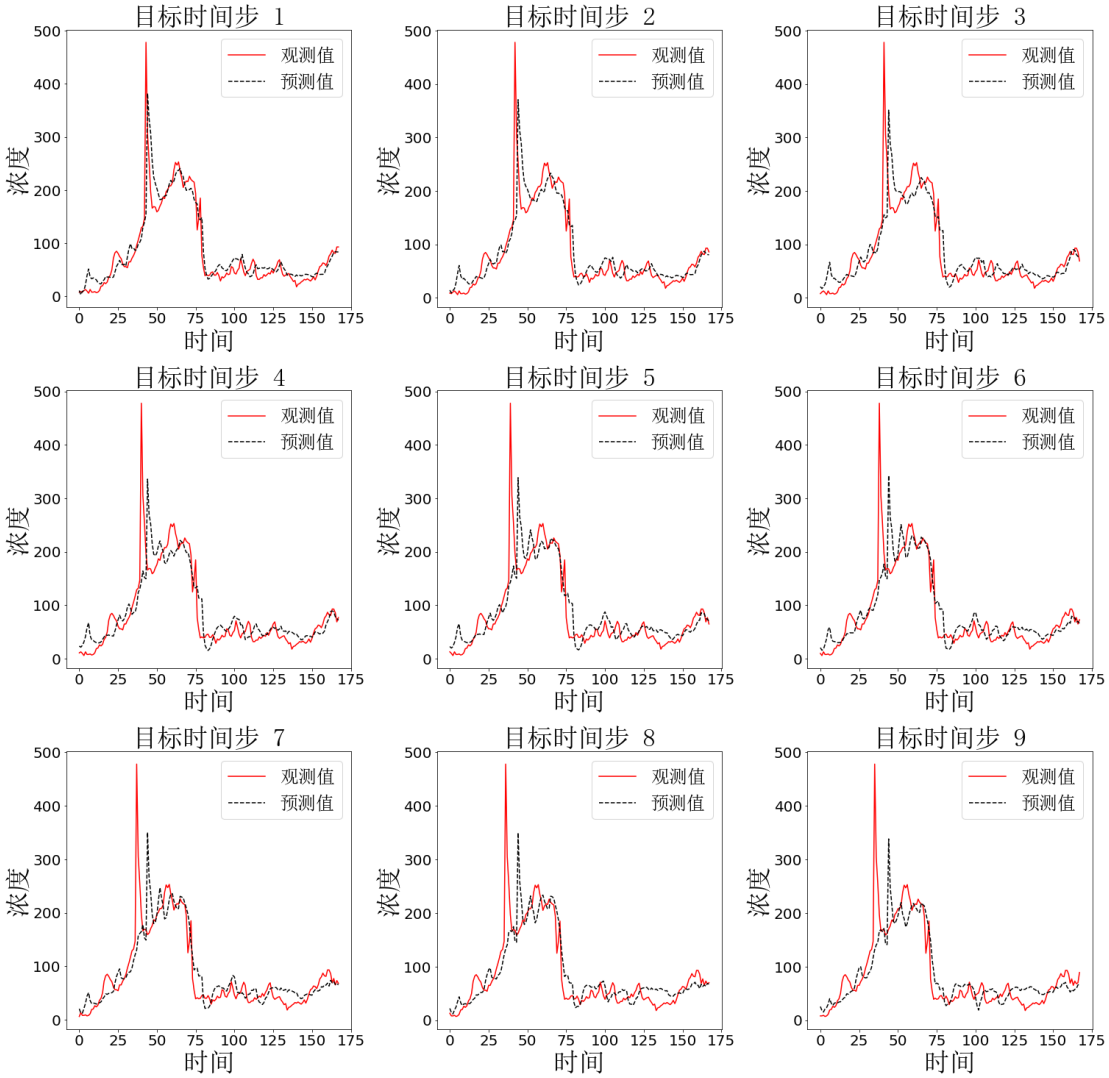


图 5-3　东四站 AAQP 预测结果的可视化

Figure 5-3 The visualization of the predictions of AAQP at Dong Si station.

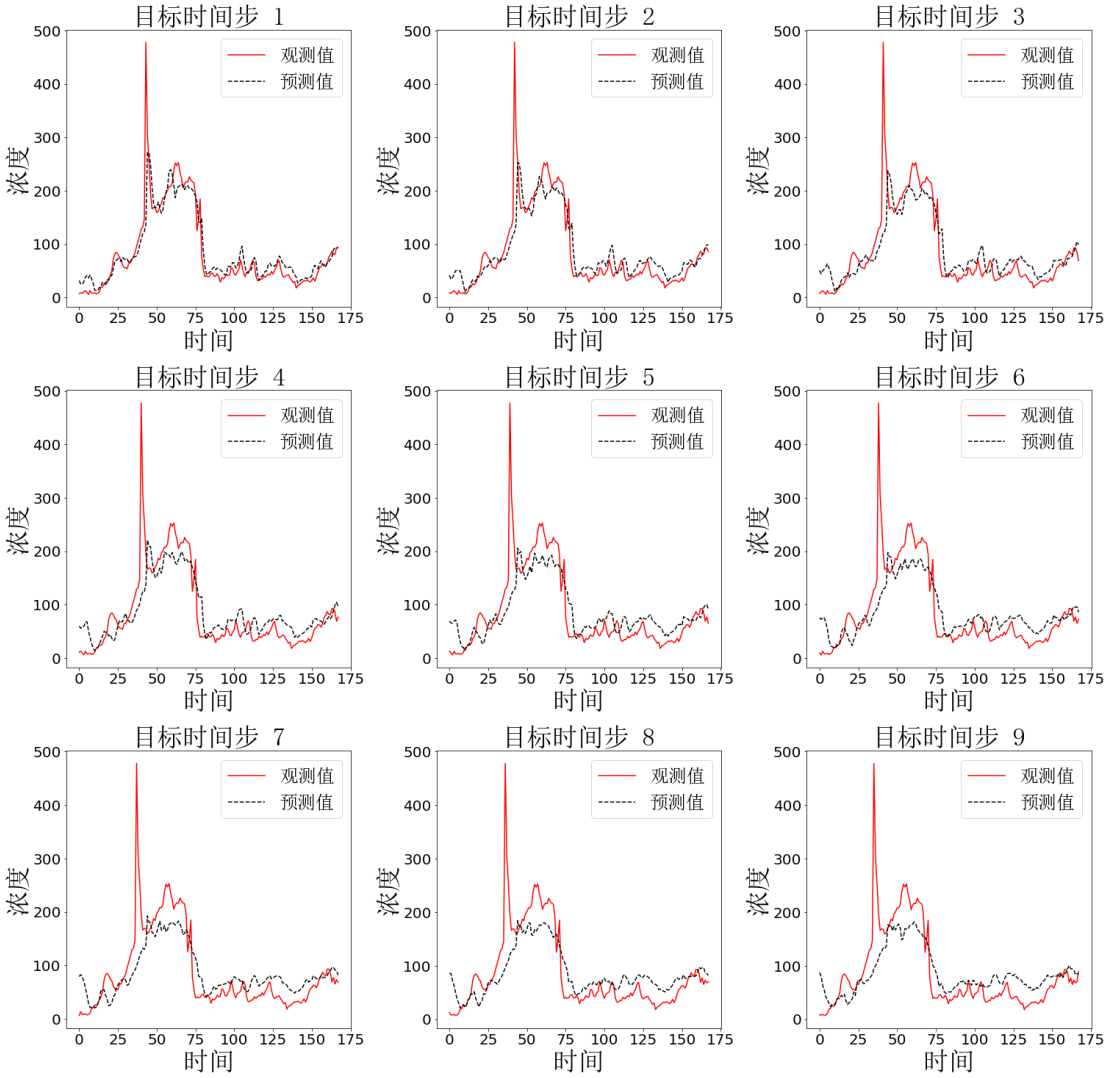


图 5-4　东四站 GBELM 预测结果的可视化。

Figure 5-4 The visualization of the predictions of GBELM at Dong Si station.

图 5-1 至图 5-4 展示了 AAQP 和 GBELM(softplus)预测结果前 9 个小时的可视化。其中对于奥林匹克中心站，选用 3-step AAQP(GRU)进行可视化，而东四站则选用 12-step AAQP(GRU)进行可视化。首先看奥林匹克中心站， GBELM 前 4 小时预测结果的 MAE 和 R^2 都比 AAQP 更好，而且在可视化的结果中也是一样的。但是 AAQP 除了有些预测值偏大以外，前 4 个小时的预测结果与 GBELM 相差无几。到了 5-9 小时的预测，就可以看到 AAQP 的预测结果明显好于 GBELM。AAQP 能够将顶点后的突降以及之后的突然上升拟合出来，但是 GBELM 在这里却出现了滞后，或是完全无法拟合的现象。因此可以看到 AAQP 能够更加使得整体的预测保持高精度，所以使得在后面小时的预测结果也能保持很高的可信度。对于东四站，GBELM 前 4 小时预测结果的 MAE 和 R^2 和 AAQP 很接近，但是从图中可以看到，GBELM 无法很好的将突变的部分拟合出来。虽然两种方法在突

变部分都出现了滞后的现象，但是 GBELM 的结果很难预测出突变的曲线。到了 5-9 小时的预测，GBELM 几乎无法将突变的曲线预测出来，同时其他部分的预测也出现了滞后，并且难以描绘细小的变化。然而 AAQP 的预测结果依然可以表示出突变的部分，虽然突变部分滞后较为严重，但是其他部分的预测几乎没有滞后而且比较精确，而且依然可以描绘出细小的变化。突变部分的滞后，正好可以解释东四站的 AAQP 依然保持着较大的 n 占据优势的原因。因为预测突变部分时，由于有滞后，其误差虽然在前面小时的预测不大，但是这个误差到了第 3,4 个小时的时候就会变得很大，因此如果较小的 n 会将这个误差积累很多次导致最终预测精度较差，但是当 n=12 时，这个误差只会传递一次，因此不会使得误差积累的非常大从而导致预测结果精度差。所以一旦对易产生突然变化的变量进行预测时，选择较大的 n 会更容易的到精确地预测。综上所述，于突然的变化，基于深度学习的 AAQP 更能将其预测出来，但是没有使用深度学神经网络的 GBELM，即便使用了集成算法，在复杂关系的拟合上还是无法与深度学习相比。

另外使用天气预报之前，AAQP 的预测结果无论在奥林匹克中心站还是东四站，两个表现最好的模型都出现了滞后的现象。可是一旦应用了天气预报的信息，滞后的现象明显减少了。事实上，空气质量中滞后的现象是很常见的，一些展示了可视化结果的研究[10, 31, 38, 56] 中都出现了一些滞后的现象，而为给出预测结果可视化的研究中应该也存在同样的现象，因此滞后现象是一个较为普遍的问题。S. Du 等[37]的研究中，当只对未来 1 小时的污染物进行预测时，其结果没有出现滞后，然而一旦对未来一段时间剑的污染物进行预测时，滞后便出现了。结合本文的实验和其他研究者的实验，滞后的出现很可能是因为预测精度低，因为当本文加入天气预报数据后预测精度提高，滞后便减少了很多，但是很难预测的突变部分依然存在滞后的现象。在模型尝试拟合目标序列时，也许会出现难以预测的情况，因此当前的输出会模仿第一个小时的输出结果，这么做会使得当前预测结果的损失函数最小化。因为空气质量数据，如 PM2.5，每个小时的变化不非常大，所以模仿第一个小时的输出就可以使得损失函数最小化。为了使得结果更加的准确，第 1 个小时的输出便很重要，因此出现滞后的预测结果中，第 1 个小时的预测结果非常的准确，但是其他小时的预测结果则出现滞后。加入天气预报之后，应该会使得所有小时的预测结果都有所增加，但是从图中可以看出，使用天气预报后对第一个小时的预测结果无法完全贴合真实值的曲线，反而出现了更大的误差。但是其他小时的预测结果都提高了。这时因为，加入天气预报信息后，其他小时的预测结果不必再完全模仿第 1 个小时的预测结果即可最小化损失函数，所以模型不会全力使得第 1 个小时的预测结果达到最优。为了解决这个问题，应该使用一种全新的损失函数。这个损失函数应该可以限制当前时刻的输出结果模仿

第一个小时的预测结果。

5.4 本章小结

本章对 AAQP 使用了天气预报的预测结果与 GBELM(softplus)的预测结果进行了对比，结果发现使用深度学习方法的 AAQP 在曲线的拟合上的确比 GBELM 更具有优势。选取合适 n 的 n-step AAQP 确实可以减少递归预测中的误差积累，同时也可以克服 GBELM 中直接预测的方式无法将目标序列作为序列处理的问题。但是 n 的取值与模型的表现没有单调的关系，较大的 n 不一定会带来性能体能，也可能会使得单个解码时间步的预测精度下降从而导致误差积累变多。一般的，在目标序列难以预测时可以采用较大的 n 避免误差积累过多，而目标序列较容易预测时可以采用较小的 n。另外在训练时间上，使用全连接编码器和位置编码明显减少了训练时间，同时 n-step 递归预测通过减少解码时间步的方式，减少了对 AM 和解码的计算，从而大幅加速了训练速度。

结 论

本文基于神经网络对空气质量预测进行了研究。研究的第一部分是基于浅层神经网络进行的研究，使用的方法 ELM，因为其泛化性能好，训练速度快。研究的另一部分则是基于深度神经网络的研究，选用 Seq2Seq 方法为基础，因为其符合空气质量预测时是将一个序列转化为另一个序列的特点。

对于 ELM，由于传统对 ELM 的激活函数多使用 sigmoid 函数，所以为了能够使其获得更高的预测精度，采用更先进的 ReLU 和 softplus，并且还尝试了使用 RBF 核函数。ReLU 和 softplus 对 ELM 的预测精度有着明显的提升，然而 RBF 和函数对于 ELM 的提升较小，明显不及 ReLU 和 softplus 的提升效果。另外 softplus 的预测结果也稍好于 ReLU。加入梯度提升算法后，无论哪一种 ELM 其预测精度都获得了提高，因此梯度提升算法对于 ELM 来说是一个很有效的改进。虽然经过两个改进，ELM 的预测精度获得了提升，但是其在突然变化情况下的拟合效果欠佳，甚至会在其预测曲线中完全丢失突然变化。ELM 在较大数据集的建模速度不如 ANN，因为 ELM 求解涉及到求隐藏层激活值矩阵的逆，然而求逆是个很耗时的操作，因此在数据量较大时，求大型矩阵的逆会消耗很多时间，反而不如 ANN 基于梯度和反向传播迭代求解速度快。

针对 Seq2Seq 做了两点改进。第一是 Seq2Seq 的训练速度慢，因为编码器和解码器所用的 RNN 无法实现并行，需要逐个时间步进行计算，因此消耗了大量的时间。为了加速训练，编码器的 RNN 被替换为全连接层和位置编码器，从而使得编码器能够同时得到每个时间步的预测结果。对于解码器，其 RNN 被保留，但是对其应用了 n-step 递归预测用于减少递归预测时的误差积累，同时由于解码时间步变少，训练速度也会大幅提升。实验证明，使用全连接编码器和位置编码完全可以代替 RNN，其预测精度与未替换时非常接近，甚至效果更好，同时训练时间明显减少。应用 n-step 递归预测也会使得模型预测精度提高，同时大幅减少了训练时间。此外，实验中还发现，n 的取值与模型的预测精度没有单调关系，即 n 变大，预测精度不一定提升或者下降。在对波动较大的目标序列进行预测时，选择较大的 n 可以获得更好的预测精度，而对于波动较小的目标序列进行预测采用较小的 n 更容易获得高精度预测。相比 GBELM 而言，n-step AAQP 克服了直接预测带来的问题，并且能够更好的对突然变化的情况进行拟合。

两种方法可以完成空气质量的细粒度建模，其中 n-step AAQP 的预测精度更高，但是伴有滞后的现象，在使用天预报的 AAQP 中滞后现象减少。事实上很多空气质量预测的研究中都有滞后现象，这有可能是因为空气质量预测均采用

MSE 作为损失函数。在使用 MSE 作为损失函数时，由于每个时间间隔空气质量变化较小，所以滞后一小时的预测结果可以使得 MSE 最小，这样会使得模型放弃对目标序列的拟合。

对于空气质量预测还有一些可以继续深入研究的内容，本文只使用了当前监测站的污染无数据和天气数据对当前监测站的污染物进行了预测，但是实际上不同监测站之间的污染物数据，天气数据等有相关性。因此未来的研究工作会集中于寻找合适的方式对高相关性的监测站进行选择，同时扩展模型使其可以将其他监测站的数据作为输入信息。另外，对于普遍存在的滞后问题，本文认为是损失函数的问题，今后的工作会继续挖掘这个问题，同时寻找合适的损失函数来减小或消除滞后现象。

参 考 文 献

[1] Sharma T S R N. Multivariate regression analysis of air quality index for Hyderabad city: Forecasting model with hourly frequency [J]. International Journal of Applied Research, 2017,3(8):443-447.

[2] Singh K P, Gupta S, Kumar A, et al. Linear and nonlinear modeling approaches for urban air quality prediction[J]. Science of the Total Environment, 2012,426:244-255.

[3] Dreiseitl S, Ohno-Machado L. Logistic regression and artificial neural network classification models: a methodology review[J]. Journal of Biomedical Informatics, 2002,35(5):352-359.

[4] Cortes C, Vapnik V. Support-vector networks[J]. Machine Learning, 1995,20(3):273-297.

[5] Park S, Kim M, Kim M, et al. Predicting PM10 concentration in Seoul metropolitan subway stations using artificial neural network (ANN)[J]. Journal of hazardous materials, 2018,341:75-82.

[6] Bai Y, Li Y, Wang X, et al. Air pollutants concentrations forecasting using back propagation neural network based on wavelet decomposition with meteorological conditions[J]. Atmospheric pollution research, 2016,7(3):557-566.

[7] Azid A, Juahir H, Toriman M E, et al. Prediction of the level of air pollution using principal component analysis and artificial neural network techniques: A case study in Malaysia[J]. Water, Air, & Soil Pollution, 2014,225(8):2063.

[8] Vito S D. Dynamic multivariate regression for on-field calibration of high speed air quality chemical multi-sensor systems: Aisem Conference, 2015[C].

[9] Kang Z, Qu Z. Application of BP neural network optimized by genetic simulated annealing algorithm to prediction of air quality index in Lanzhou: 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA), 2017[C]. IEEE.

[10] Paoli C, Notton G, Nivet M, et al. A neural network model forecasting for prediction of hourly ozone concentration in Corsica: 2011 10th International Conference on Environment and Electrical Engineering, 2011[C]. IEEE.

[11] Mahajan S, Liu H, Tsai T, et al. Improving the accuracy and efficiency of pm2. 5 forecast service using cluster-based hybrid neural network model[J]. IEEE Access, 2018,6:19193-19204.

[12] Asghari M, Nematzadeh H. Predicting air pollution in Tehran: Genetic algorithm and back propagation neural network[J]. Journal of AI and Data Mining, 2016,4(1):49-54.

[13] Sánchez A S, Nieto P G, Fernández P R, et al. Application of an SVM-based regression model to the air quality study at local scale in the Avilés urban area (Spain)[J]. Mathematical and Computer Modelling, 2011,54(5-6):1453-1466.

[14] Nieto P G, García-Gonzalo E, Sánchez A B, et al. Air quality modeling using the PSO-SVM-based approach, MLP neural network, and M5 model tree in the metropolitan area of Oviedo (Northern Spain)[J]. Environmental Modeling & Assessment, 2017:1-19.

[15] Gu K, Qiao J, Lin W. Recurrent air quality predictor based on meteorology-and pollution-related factors[J]. IEEE Transactions on Industrial Informatics, 2018,14(9):3946-3955.

[16] Huang G, Zhu Q, Siew C. Extreme learning machine: theory and applications[J]. Neurocomputing, 2006,70(1):489-501.

[17] Zhang J, Ding W. Prediction of air pollutants concentration based on an extreme learning machine: the case of Hong Kong[J]. International journal of environmental research and public health, 2017,14(2):114.

[18] Wang D, Wei S, Luo H, et al. A novel hybrid model for air quality index forecasting based on two-phase decomposition technique and modified extreme learning machine[J]. Science of The Total Environment, 2017,580:719-733.

[19] Vong C, Ip W, Chiu C, et al. Imbalanced learning for air pollution by meta-cognitive online sequential extreme learning machine[J]. Cognitive Computation, 2015,7(3):381-391.

[20] Li X, Peng L, Hu Y, et al. Deep learning architecture for air quality predictions[J]. Environmental Science and Pollution Research, 2016,23(22):22408-22417.

[21] Lipton Z C, Berkowitz J, Elkan C. A critical review of recurrent neural networks for sequence learning[J]. arXiv preprint arXiv:1506.00019, 2015.

[22] Ong B T, Sugiura K, Zettsu K. Dynamic pre-training of deep recurrent neural networks for predicting environmental monitoring data: 2014 IEEE International Conference on Big Data (Big Data), 2014[C]. IEEE.

[23] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks: International conference on machine learning, 2013[C].

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

[25] Chaudhary V, Deshbhratar A, Kumar V, et al. Time Series Based LSTM Model to Predict Air Pollutant's Concentration for Prominent Cities in India[J]. 2018.

[26] Pardo E, Malpica N. Air quality forecasting in madrid using long short-term memory networks: International Work-Conference on the Interplay Between Natural and Artificial Computation, 2017[C]. Springer.

[27] Li X, Peng L, Yao X, et al. Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation[J]. Environmental Pollution, 2017,231:997-1004.

[28] Zhao J, Deng F, Cai Y, et al. Long short-term memory-Fully connected (LSTM-FC) neural network for PM2. 5 concentration prediction[J]. Chemosphere, 2019,220:486-492.

[29] Wang J, Song G. A deep spatial-temporal ensemble model for air quality prediction[J]. Neurocomputing, 2018,314:198-206.

[30] Zhou Y, Chang F, Chang L, et al. Explore a deep learning multi-output neural network for regional multi-step-ahead air quality forecasts[J]. Journal of Cleaner Production, 2019,209:134-145.

[31] Kim M, Kim Y, Sung S, et al. Data-driven prediction model of indoor air quality by the preprocessed recurrent neural networks: 2009 ICCAS-SICE, 2009[C]. IEEE.

[32] Kök 0, Şimşek M U, Özdemir S. A deep learning model for air quality prediction in smart cities: 2017 IEEE International Conference on Big Data (Big Data), 2017[C]. IEEE.

[33] Dey R, Salemt F M. Gate-variants of gated recurrent unit (GRU) neural networks: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), 2017[C]. IEEE.

[34] Athira V, Geetha P, Vinayakumar R, et al. Deepairnet: Applying recurrent networks for air quality prediction[J]. Procedia computer science, 2018,132:1394-1403.

[35] Wang B, Yan Z, Lu J, et al. Deep Multi-task Learning for Air Quality Prediction: International Conference on Neural Information Processing, 2018[C]. Springer.

[36] Sun X, Xu W, Jiang H. Spatial-temporal prediction of air quality based on recurrent neural networks: Proceedings of the 52nd Hawaii International Conference on System Sciences, 2019[C].

[37] Du S, Li T, Yang Y, et al. Deep Air Quality Forecasting Using Hybrid Deep Learning Framework[J]. arXiv preprint arXiv:1812.04783, 2018.

[38] Huang C, Kuo P. A deep cnn-lstm model for particulate matter (PM2. 5) forecasting in smart cities[J]. Sensors, 2018,18(7):2220.

[39] Lecun Y, Bengio Y. Convolutional networks for images, speech, and time series[M]. 1998.

[40] Zhang C, Yan J, Li C, et al. On estimating air pollution from photos using convolutional neural network: Proceedings of the 24th ACM international conference on Multimedia, 2016[C]. ACM.

[41] Soh P, Chang J, Huang J. Adaptive deep learning-based air quality prediction model using the most relevant spatial-temporal relations[J]. IEEE Access, 2018,6:38186-38199.

[42] Pan Z, Liang Y, Zhang J, et al. HyperST-Net: Hypernetworks for Spatio-Temporal Forecasting[J]. arXiv preprint arXiv:1809.10889, 2018.

[43] Reddy V, Yedavalli P, Mohanty S, et al. Deep Air: Forecasting Air Pollution in Beijing, China[Z]. 2018.

[44] Bui T, Le V, Cha S. A Deep Learning Approach for Forecasting Air Pollution in South Korea Using LSTM[J]. 2018.

[45] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.

[46] Wang H, Zhuang B, Chen Y, et al. Deep Inferential Spatial-Temporal Network for Forecasting Air Pollution Concentrations[J]. arXiv preprint arXiv:1809.03964, 2018.

[47] Liang Y, Ke S, Zhang J, et al. GeoMAN: Multi-level Attention Networks for Geo-sensory Time Series Prediction.: IJCAI, 2018[C].

[48] Cheng W, Shen Y, Zhu Y, et al. A neural attention model for urban air quality inference: Learning the weights of monitoring stations: Thirty-Second AAAI Conference on Artificial Intelligence, 2018[C].

[49] Glorot X, Bordes A, Bengio Y. Deep Sparse Rectifier Neural Networks.: Aistats, 2011[C].

[50] Huang G B, Siew C K. Extreme learning machine: RBF network case: Control, Automation, Robotics and Vision Conference, 2004. Icarcv 2004, 2004[C].

[51] Natekin A, Knoll A. Gradient boosting machines, a tutorial[J]. Front Neurorobot, 2013,7:21.

[52] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks: Advances in neural information processing systems, 2014[C].

[53] 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.

[54] Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning: Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017[C]. JMLR. org.

[55] Luong M, Pham H, Manning C D. Effective approaches to attention-based neural machine translation[J]. arXiv preprint arXiv:1508.04025, 2015.

[56] Lira T S, Barrozo M A, Assis A J. Air quality prediction in Uberlândia, Brazil, using linear models and neural networks[M]//Computer Aided Chemical Engineering. Elsevier, 2007:51-56.

攻读硕士期间取得的研究成果

SCI期刊论文：

[1] Liu B, Yan S, You H, et al. Road surface temperature prediction based on gradient extreme learning machine boosting[J]. Computers in Industry, 2018,99:294-302. (SCI，IF:2.850, WOS: 000435048200024)

[2] Liu B, Yan S, Li J, et al. A Sequence-to-Sequence Air Quality Predictor Based on the n-step Recurrent Prediction: IEEE Access (SCI，IF=3.557, WOS: 000465630900001)

EI国际会议论文：

[1] Liu B, Yan S, Li J, et al. Forecasting PM2. 5 concentration using spatio-temporal extreme learning machine: 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), 2016[C]. IEEE. (EI检索，检索号：20171203470668)

[2] Liu B, Yan S, You H, et al. An ensembled RBF extreme learning machine to forecast road surface temperature: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), 2017[C]. IEEE. (EI检索，检索号：20182505330888)

[3] Liu B, Yan S, Li J, et al. An Attention-Based Air Quality Forecasting Method: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018[C]. IEEE. (EI检索，检索号：20191006583213)

发明专利：

[1]. 刘博，闫硕. 一种基于集成极限学习机的空气质量预测方法（已公开，CN 107330514 A）

[2]. 刘博，闫硕. 一种基于变分自编码器和极限学习机的空气质量预测方法（已公开，CN 108197736 A）

[3]. 刘博，闫硕. 一种基于多步骤递归预测的空气质量预测方法

软件著作权：

刘博，闫硕. 基于深度学习的空气质量预测系统 （已授权，2017SR092766）

基金：

参与并负责研究生科技基金——关于大气污染成因分析及高精度预测的研究

参与国家自然科学基金(61702021)——时空大数据驱动的大气能见度影响因素分析及精细化预报方法研究

参与北京市自然科学基金(Z5025001201701)——基于因果关联挖掘及语义推理的雾霾追因分析方法研究

获奖

科技创新奖优秀奖

致谢

三年的时间过去，这期间我学到了很多，能力也有了很大的提升。所以我首先要感谢我的导师刘博老师，从本科毕业论文开始就跟着刘博老师学习，在刘博老师的直到下，三年的学术生涯收获颇丰。刘博老师的指导主要体现在对科研能力的培养，不再是如何学习知识，而是如何发现现有知识中的问题，并提出解决方案。另外论文的写作上老师对我的指导也非常的多，从一开始结构混乱的论文，到现在条例清晰的论文，离不开刘博老师的指导。还有老师的督促也至关重要，每次开会都会督促我们的论文进度，所以我才能发表多篇学术论文。所以没有老师的帮助也不会有今天的我。

接下来要感谢我的同门兄弟，刘银星和姚柯璐。我们三人在共同的境地下，所以常常会在一起交流，一起研究论文怎么写，一起研究怎么通过考试，一起研究毕业论文的写法。遇到困难时我们互相鼓励，有了荣誉我们一起祝贺。有了他们，我的研究生生活充满了欢乐。

然后感谢一下宿舍的同学们，一直在家住的李圳，退宿的夏斯彬，本科同学张立波，还有新入住的刘文玉。虽然我们几个人有着不同的导师，甚至完全不同的研究方向。但是在宿舍里，我们一起玩，互相了解不同方向的知识，都对我帮助很大。虽然大多数宿舍同学相处时间短，相逢便是有缘所以还是要感谢一下。

接着要感谢我的一个朋友。虽然我们是来自不同学校的学生，但是对于机器学习都有很高的热情。我们一起讨论算法，研究每个公式的意义，互相讲解自己的所学，这让我获益良多。他甚至影响了我的毕业去向，原本打算工作的，不过在他的劝说下还是决定继续深造。所以他对我的影响也非常大，因此必须感谢一下。

最后要感谢父母，研究生读下来还是需要父母背后的支持，能够完成这些科研成果和父母的支持也是分不开的，所以必须要感谢。

研究生三年确实是一个重要的人生转折点，我找到了我想奋斗终身的理想，所以毕业不是结束，而是真正的开始。