多观测点联合的 PM2.5 预测

张猛

学号：2015011463

2020 年 6 月 2 日

摘要

本文提出了一种联合多观测点数据进行 PM2.5 的预测的算法，利用了北京地区 35 个观测点的 24 小时数据，对这 35 个观测点的未来 6 小时的 PM2.5 数值进行预测。本文提出的方法使用了典型的 Seq2Seq 的结构，网络结构由 24 个 GRU 单元组成的编码器和 6 个 GRU 单元组成的解码器构成，并添加了注意力机制提升效果。本文所使用的北京空气质量网得到的真实数据集进行测试，并得到了良好的结果。

关键词： PM2.5 预测；空气质量；Seq2Seq；注意力机制

1 简介

随着工业化、城市化的逐步推进，空气污染问题越发严重，在国内各大城市尤其如此，污染指数屡屡超标。而另一方面，人民对美好生活环境的需求随着物质水平的提高而日益迫切，空气质量就是其中最重要的一块，关系着所有人的日常生活。所以，精确的空气质量预报，可以帮助人们合理安排行程，享受美好生活。

过去的空气质量预测多使用观测点的车流量、街道的几何构造、气候风力信息等使用计算流体动力学的方法进行预测，但是这类方法需要大量的先验数据，所以成本很高，而且缺乏推广性。近些年，随着数据采集数量和质量的增长，使用机器学习方法随机森林或者神经网络例如 LSTM 和 Seq2Seq 的方法已经成为主流。GeoMAN 提出一种多层次的空间时间注意力机制的模型，是 2018 年最先进的方法。

城市的空气污染分布在空间和时间上都有不同的分布，这对空气质量的预测的问题带来了复杂性。另一方面，不同空间的空气质量之间也存在联系，相互影响。对此，本文提出联合多个观测点的数据，融合空间信息，进行 PM2.5 的预测。本文以 35 个观测点的特征作为输入，提取联合特征，输入到编码器中，最终由解码器得到 35 个观测点未来 6 小时的 PM2.5 预测值。本文的贡献主要在于：

- 本文对原始数据进行合理的清洗和特征选择。

- 本文联合多个观测点数据，得到观测点之间相互影响的信息。
- 本文使用 Seq2Seq 结构，得到了较为准确的 PM2.5 预测结果。

2 任务定义

本文研究的任务为利用当前和过去一段时间的各个观测点的观测值，预测未来 6 个小时 PM2.5 的等级。本文方法选取了当前和过去 23 小时，一共 24 小时的观测数据。我们使用 O_k^t 表示 t 时刻第 k 个观测点的观测数据，第 t 时刻的所有观测点表示为 O^t。观测数据为多个数值，表示为 $O_k^t = (o_{k0}^t, o_{k1}^t, \cdots)$ 。而预测的 PM2.5 等级表示为 $P_k^t \in \{0, 1, 2, 3, 4, 5\}$，同样所有观测点表示为 P^t。我们的任务使用数学符号定义即为，设计一个系统 $\mathcal{H}$，使得：

$$\mathcal{H}(O^{-23}, \cdots, O^0) = (P^1 \cdots P^6) \tag{1}$$

3 数据处理

本文所使用的数据来自于北京空气质量网[1]。数据总共为 20140101 至 20200509 的 SO2, SO2_24h, NO2, NO2_24h, O3, O3_24h, CO, CO_24h, PM2.5, PM2.5_24h, PM10, PM10_24h, AQI 的特征。我们对数据分别做了相应处理。

3.1 文件清理

本文对爬取的数据的部分文件进行了文件层面的处理，以保证顺利读取

1. 20140401 及之前的 extra 特征的文件缺失，即缺少 SO2, SO2_24h, NO2, NO2_24h, O3, O3_24h, CO, CO_24h 等特征，所以本文直接删除对应的 20140101 及之前的 all 标识文件。
2. 20140402 至 20140428 的 extra 特征只有七维，相对于之后数据缺少了 O3_24h 特征，导致特征无法对齐。对此，我们选择删除了对应天数的文件，保证每天 extra 文件中每小时 8 个特征。
3. 20141231 的对应的 all 文件损坏，清理删除了 20141231 当日的 all 文件和 extra 文件。

在文件清理后，测试集剩余 20140429——20141230 时段的完整数据。

[1] http://zx.bjmemc.com.cn/

3.2 数据清洗

在文件清理之后，我们对所有特征进行可视化，其训练测试集和测试集的分布直方图如图 1和图 2。

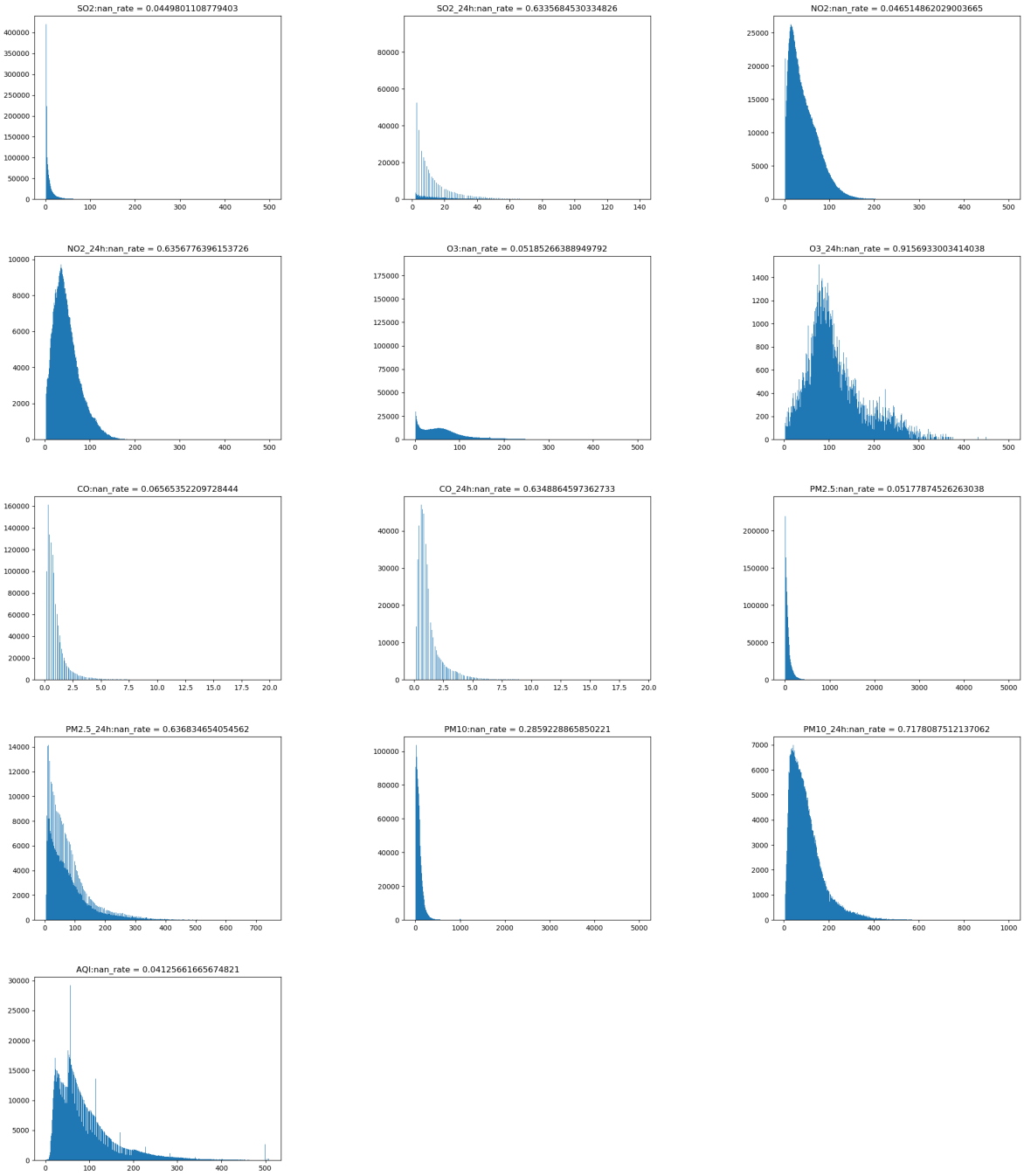

图 1: 训练验证集的特征分布直方图

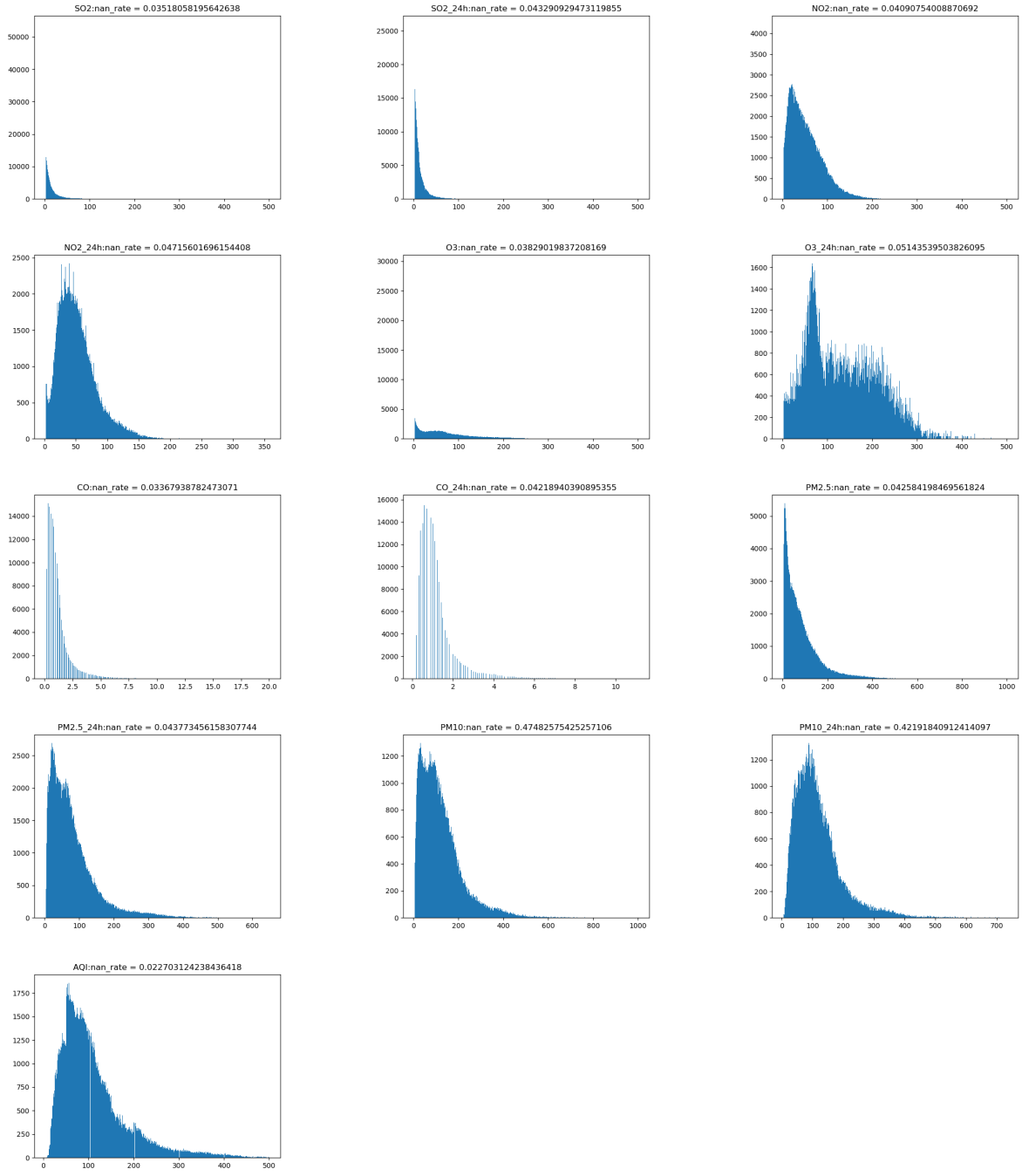

图 2: 测试集的特征分布直方图

我们首先将所有数据读入，为了取得连续时间，我们对部分数据做了如下处理：

1. 若某一天的时间不足 24 个小时，认为连续时间在此中断。不读入不足 24 小时的数据，同时在程序中记录中断位置 breakpoints。

2. 20170527 左右的数据文件为空，同样进行中断处理。

3. 20170702 左右的数据文件为爬取失败的字符串文件，同样认为中断，记录中断点。

进行了数据连续性的处理之后，我们分别得到了训练验证集和测试集的的 35 个检测点的观测值。观察数据分布，粉们可以发现，几乎所有的特征都是**长尾分布**，数据分布严重的偏度会导致算法的效果较差。所以我们对特征进行了取对数处理：

$$o_k^t = \ln(o_k^t + 1) \tag{2}$$

使得特征分布被矫正到接近于正态分布。此时，各个特征分布仍然具有一定偏置，我们对其进行标准化，使其矫正到 $\mathcal{N}(0,1)$ 的正态分布，标准化后的特征更有利于训练的收敛。需要注意的是，我们进行标准化所使用的的均值和标准差为训练验证集的样本所计算的均值和标准差，为保证结果的公平性，在对测试集进行标准化时，使用相同的参数，并没有使用测试集的任何先验信息。

$$o_{ki}^t = \frac{o_{ki}^t - \mu_i}{\sigma_i} \tag{3}$$

此时，数据中仍然存在大量的 nan 值。我们对于数据缺失值进行如下处理：

1. 首先对所有的缺失值利用均值补全。由于我们进行了标准化，所以补充的为零值。
2. 对于任何一个缺失值 o_{ki}^t，我们使用前后各 10 个小时的特征进行 B 样条插值，得到最终的缺失补充值。

4 特征选择

原始数据中一共有 SO2, SO2_24h, NO2, NO2_24h, O3, O3_24h, CO, CO_24h, PM2.5, PM2.5_24h, PM10, PM10_24h, AQI 以及时间共 14 维特征。这 14 维特征的训练验证集分布如图 1，并且我们统计了这些特征丢失的比例，即 nan 值所占的比例，并标明在分布上方。我们认为，丢失比例大于 20% 的特征对最终的预测的帮助不大，nan 值的插值多补充成为均值，反而影响数据分布。于是，我们只保留了 nan 值比例小于 20% 的特征，总共 7 维，保留的特征为：

[hour]	当前的小时时刻
[SO2]	SO2 取对数的标准化后指数
[NO2]	NO2 取对数的标准化后指数
[O3]	O3 取对数的标准化后指数
[CO]	CO 取对数的标准化后指数
[PM2.5]	PM2.5 取对数的标准化后指数
[AQI]	AQI 取对数的标准化后指数

对于选择后的除时间以外的特征，我们的处理如式 2和式 2所示。对于 hour 特征，原始特征为 0——23 的离散整形值，注意到这个数值并不能完全代表时间的距离的远

近，例如 0 时和 23 时在特征上相差 23，但是实际上只有 1 个小时的差距。对此，我们利用正弦函数将时间轮接合起来：

$$hour = \sin(\frac{\pi}{12} hour) \tag{4}$$

这样时间 hour 特征被限制到-1 到 1 之间，同时，时间的距离使用新特征的差值进行度量。

5 模型设计

本文使用了典型的 Seq2Seq 模型进行 PM2.5 的回归，再利用得到的回归值按分类标准进行分类，得到最终的分类结果。Seq2Seq 的模型分为 encoder 和 decoder 两个部分，分别有 24 个 GRU 单元和 6 个解码的 GRU 单元组成。

5.1 编码器

编码器的 24 个 GRU 单元按顺序分别输入 t 时刻及之前 24 小时的特征，每个小时的输入 O^t 维度为 35×7，分别代表 35 个观测点的 7 个特征。在进入 GRU 模块之前，35×7 特征首先经过全连接网络进行空间特征的融合，网络结构如图 3。全连接的结构分别为 fc(7,16),fc(16,8),fc(280,280)，每个全连接层后都为 relu 的激活层和概率 0.5 的 dropout 层。

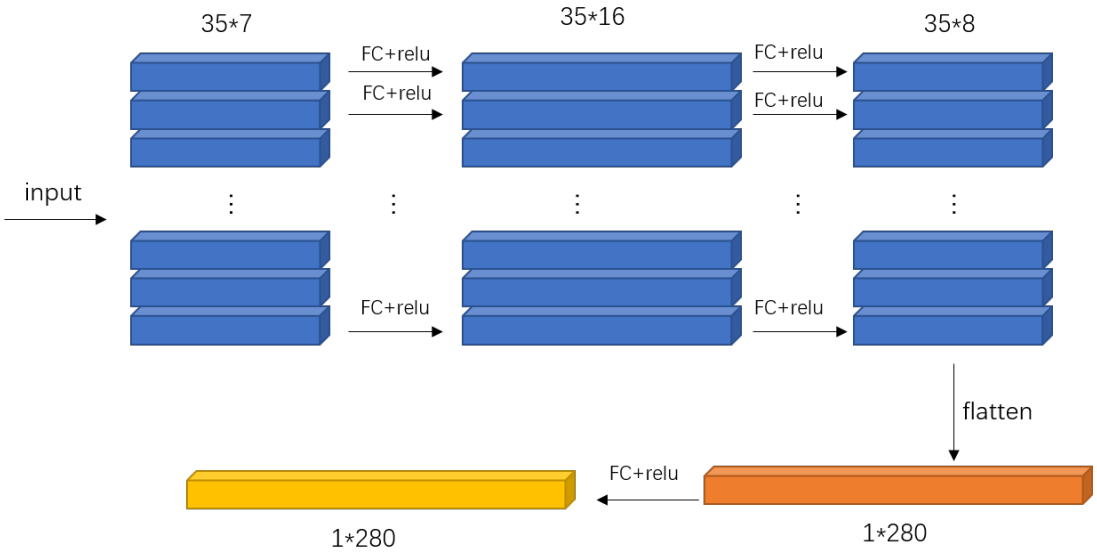

图 3: 多观测点特征融合

5.2 解码器

每个 encoder 部分的 GRU 的输入大小和隐层向量大小都设置为 280，经过 24 个 GRU 的编码得到最终 280 维的向量。解码器部分本文使用注意力机制的 GRU 网络，网

络结构如图 4 [2]。

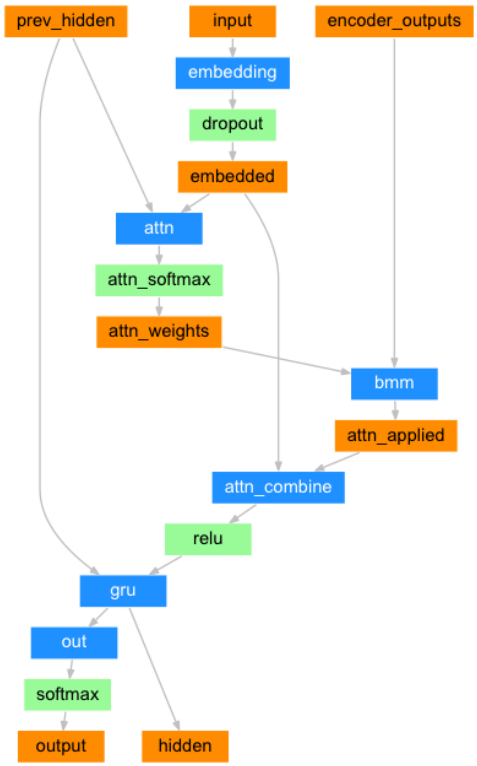

图 4: 带有注意力机制的解码器

带有注意力机制的解码器主要借鉴于 pytorch 的官方教程 [1]。与原本的文本翻译不同，原本的解码器的输入部分为一个 embedding 模块，而本文的编码器和解码器所使用的的 GRU 结构不同，所以对 GRU 的输入进行了修改。本文 GRU 的输出仍然为和隐向量相同尺度的 280 维向量，在输出后我们接了 fc(280,140) 和 fc(140,35) 的全连接层，得到 35 个观测点的预测值。同样，35 个观测点的预测值被作为下一个 GRU 的输入向量输入。解码器的第一个 GRU 的输入使用了 t 小时的 35 个观测点的 PM2.5 数值。这样，我们通过 6 个 GRU 单元，一步步解码得到未来 6 个小时的 PM2.5 预测值。

5.3 损失函数

由于 PM2.5 的分布为长尾分布，所以大部分的 PM2.5 的值都比较小，正负样本的不均衡导致网络收 PM2.5 小的样本影响，从而很难预测 PM2.5 大的样本。对此，我们一方面使用了取对数的方式使得 PM2.5 的分布近似为正态分布，减小偏度。另一方面，我们对自己的损失函数采取了不同的设计。

虽然本问题是分类问题，但是 PM2.5 为数值型变量，所以很自然地先对 PM2.5 的值进行回归，在按标准进行分类。我们使用 MSELoss 作为损失函数，并在其基础上加

[2]图 片 来 自https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html

入权重以实现负样本挖掘。权重的设计为：

$$w = 2 - \sqrt{1 - (\frac{clamp(P, 0, 500)}{500})^2} \tag{5}$$

这样，对每个 PM2.5 的样本进行了 soft 的加权，使得 PM2.5 大的样本的损失权重更大。在实验中注意到，可能由于数据错误的原因，部分 PM2.5 的数值达到了 5000，所以进行了相应的剪切处理。

不过，从最终的实验结果来看，是否添加损失的权重对最终的结果并没有很大影响。

6 实验设计和结果

6.1 数据集划分

本文对数据进行预处理之后，进行数据划分。2014 年的数据为测试集，共 5315 条。剩余数据为训练测试集，我们按照 4:1 的比例随机划分训练集和验证集，为保证可重复性，我们设定随机的初始种子为 0。划分后训练集 32032 个样本，测试集为 7850 个样本。训练集，验证集，测试集分别存储在 train_dataset.h5,val_dataset.h5 和 test_dataset.h5 文件中，以便调用。

6.2 评价指标

分类正确率 未来六个小时的分类正确率 mAP，每个小时的正确率都为 35 个地点的所有类别的平均。

平均绝对误差 对回归的 PM2.5 结果进行评价，平均绝对误差 MAE 即为预测值和真实值差值绝对值的平均。

突变准确率 本文专门对 PM2.5 的值发生突变时的预测准确率进行评价。我们对 PM2.5 的突变的定义是，预测的 6 小时的 PM2.5 的平均值和之前 6 小时的平均值的差大于 100。在这种情况下的分类正确率为突变准确率 SP。

6.3 模型的最好结果

尽管本文进行了各种各样结构的尝试，以及各种 trick，但是，需要承认的是，大部分的措施都对最终的测试集效果几乎没有影响。所以在此，本文遵循奥卡姆剃刀原则，选择最简单的 baseline 的模型作为最优结果。我们设置 batch_size=256，以初始学习率 lr=1e-3 的 Adam 优化器进行了轮数 epoch=100 的训练。模型结构中，编码器和解码器 GRU 的隐向量大小为 35*16，编码器融合多个观测点的 MLP 网络的 dropout=0.5。最终训练结果如下：

表 1: 分类正确率 mAP

	1-mAP	2-mAP	3-mAP	4-mAP	5-mAP	6-mAP	mAP
验证集	83.69	83.10	83.50	83.70	83.56	82.21	83.29
测试集	76.09	70.33	65.53	61.65	58.53	55.51	64.61

表 2: 平均绝对偏差 MAE

	1-MAE	2-MAE	3-MAE	4-MAE	5-MAE	6-MAE	MAE
验证集	8.718	8.946	8.731	8.601	8.736	9.395	8.855
测试集	13.86	17.53	20.96	24.01	27.78	29.35	22.08

表 3: 突变正确率 SP

验证集	测试集
79.24	51.96

6.4 超参数设计

dropout 我们对编码器的多观测点融合的结构的 dropout 的影响进行了研究，我们分别设置 dropout=0,0.2,0.5，对比其验证集和测试集的正确率和误差。

表 4: dropout 影响

dropout	val-mAP	val-MAE	test-mAP	test-MAE
0	86.01	7.23	61.72	23.85
0.5	84.31	8.24	63.92	22.41
0.5	83.29	8.85	64.61	22.08

从实验结果可以看出，dropout 越小，验证集的正确率越高，而测试集的正确率飞速下降，这说明 droppout 可以防止模型的过拟合。也印证了测试集分布和训练验证集的差别。

学习率 我们对学习率的影响进行了研究，在训练轮数 epoch=100 的情况下，我们分别设置 lr=1e-3,5e-4,1e-4，对比其验证集和测试集的正确率和误差。

表 5: 初始学习率影响

lr	val-mAP	val-MAE	test-mAP	test-MAE
1e-3	83.29	8.85	64.61	22.08
5e-4	82.76	9.12	63.59	22.54
1e-4	73.86	14.78	63.82	23.07

隐向量大小 本文在其他条件不变的基础上，探究了隐向量大小对预测准确率的影响。分别设置 hidden_size=35*8,35*16,35*32，实验结果如下：

表 6: 隐向量大小影响

hidden	val-mAP	val-MAE	test-mAP	test-MAE
35*8	80.60	10.28	64.08	22.23
35*16	83.29	8.85	64.61	22.08
35*32	84.41	8.86	64.30	22.28

隐向量为 35*8 时，不能承载足够的信息，所以正确率较低。而 35*16 和 35*32 的方法正确率比较接近，我们选择参数更少，模型复杂度更低的 35*16 的隐向量作为最终的模型。

7 实验结果分析

本文在对测试集正确率进行改进的过程中，尝试了各种各样的结构，虽然最终都没有得到有效的进展，但是如实记录如下。

7.1 正确率无法提高原因分析

正如实验结果所示，随着时间的推移，预测的准确率在逐渐下降。而统计正确率和类别的关系，可以看出，由于样本不均衡，所以 PM2.5 高的情况，预测的准确率较低。分析突变情况，在 PM2.5 发生突变时，模型很难做出预测，这也是本文没有对突变进行考虑的导致的。

对比验证正确率和测试正确率，可以看出验证正确率明显较高。经过严谨的分析，原因是验证集是从 trainval 中随机选择的，所以分布于训练集更为接近，而测试集是 2014 年的分布差别较大。我们通过将验证集固定为 2015 年，此时，验证集的正确率同样降低；而将测试集设为 2019 年数据，则测试正确率也达到很高。从中可以分析得出，开始 14/15 年的空气质量分布较为不同，联系实际生活，可知在北京治理之前空气质量

较差，而近些年明显好转，所以近几年的 PM2.5 的低等级情况更多，预测更为容易，而比较早时间的样本污染比较严重，所以预测难度较大。

7.2 难样本挖掘

由于样本分布不均，且错误更多出现在 PM2.5 更大的情况中，所以我们尝试了难样本挖掘的策略。在 dataloader 中，我们对难样本进行重采样和简单样本的降采样，样本挖掘的策略如下：

1. 对于 PM2.5 大于阈值的情况，超出阈值越多，越大概率对此样本重采样一次，将此样本重复一次，加入到训练集中。

2. 对于 PM2.5 较小的情况，越接近于 0，越大概率将此样本丢掉。

通过这样的策略，可以达到丢弃一半的简单样本，使 PM2.5 大于等级为 3 以上的样本增加一倍。但是，最终的实验结果证明，调整了样本分布，是的验证集的正确率降低，但是测试正确率并没有得到提高。

表 7: 难样本挖掘影响

	val-mAP	val-MAE	test-mAP	test-MAE
无样本挖掘	83.29	8.85	64.61	22.08
难样本挖掘	70.90	16.31	63.64	22.66

我们同样对加权重的损失函数进行测试，在权重较小时没有影响，权重大时，会严重导致验证集正确率的降低，导致训练的过程非常不稳定。

表 8: 难样本挖掘影响

	val-mAP	val-MAE	test-mAP	test-MAE
MSEloss	83.29	8.85	64.61	22.08
Weighted MSEloss	83.23	8.812	64.10	22.20

7.3 数据增广

同样为了解决测试集正确率过低的问题，增强模型的推广能力，本文尝试对数据增广。由于和图像等不同，多个观测点的时序数据没有合理的增广方式，本文尝试了添加高斯扰动的方式，对于某一个地点的所有观测数据进行平移扰动：

$$o_k = o_k + N(0, \sigma) \tag{6}$$

同样，对应地点的 label 也添加同样的平移：

$$P_k = P_k + N(0, \sigma) \tag{7}$$

这种增广方式并没有考虑数据之间的内部联系，所以自然也没有对最终的效果产生明显的影响。如图 5，橙色、红色、深蓝、浅蓝分别是 $\sigma = 0, 0.001, 0.05, 0.1$ 的正确率变化曲线，说明扰动对训练结果几乎没有影响。

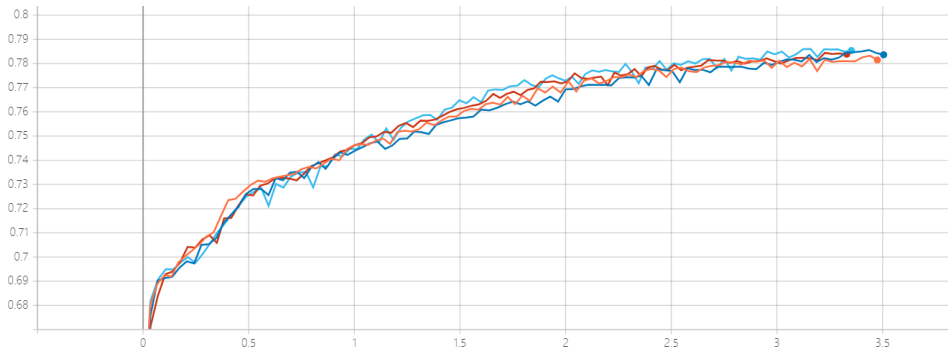

图 5: 扰动增广的效果

7.4 分布拟合可视化

本文使用 tensorboard 对训练过程中的预测值和真实值的分布拟合过程进行了可视化。可以从图中看出，深红色的分布为真实值分布，橙色的分布为预测值分布，预测的 PM2.5 的分布和真实的分布还是比较接近的，主要的差距都在于 PM2.5 较大的部分，而这部分由于比例非常小，很难从图中分辨出来。

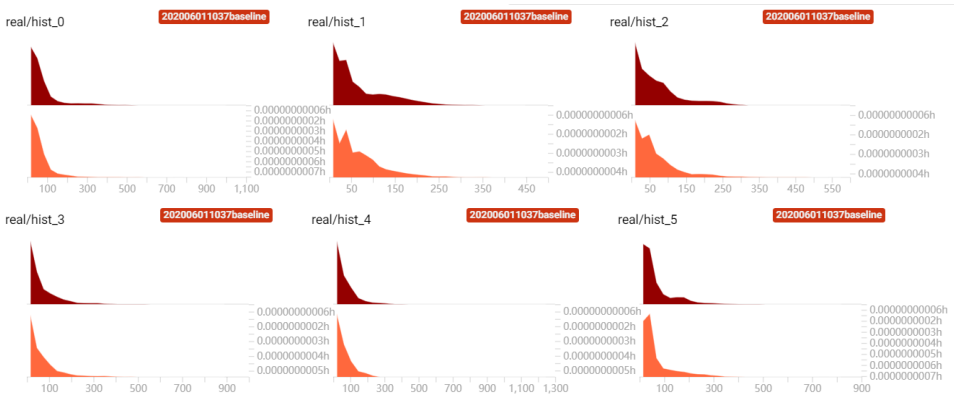

图 6: 预测值和真实值分布

7.5 特征标准化

本文还探究了对于 PM2.5 特征是否进行标准化的影响，即一组实验只取对数，不进行标准化，另一组则标准化到 N(0,1)。从训练的曲线中可以看出，不进行标准化的训练收敛波动更大，更不稳定，也符合我们的预期。图 7中红色的曲线为标准化的正确率

变化，橘黄色为未标准化的，可以看出，未标准化的波动更大，最终的验证正确率也更低。

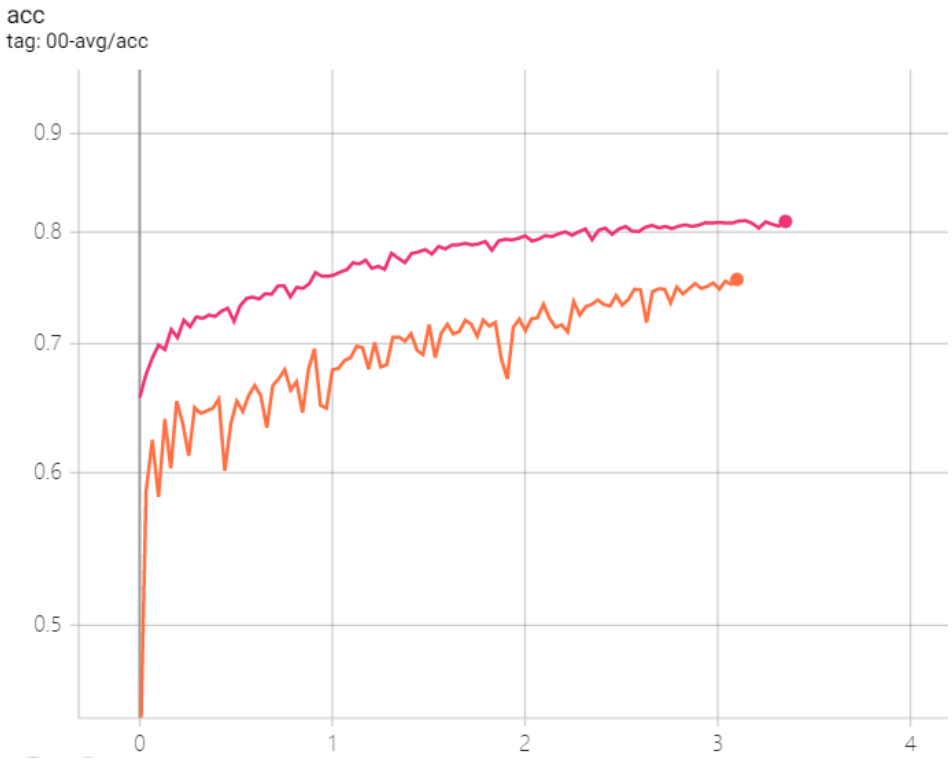

图 7: 特征标准化的对比

8 代码接口

需要说明的是，由于进行了文件层面上的清理，所以本文的方法可能不能使用全自动的代码运行。而需要对 20140429 之前的文件手动删除。

本文首先在 create.py 读入 csv 数据，并转存为 database.h5 文件。之后利用 clean.py 文件对数据进行清洗和相应的正则化，并生成训练所需要的 (24-6) 连续时间的样本，存储到 dataset.h5 中。train.py 中的 train 函数负责训练模型，test.py 中的 evaluate 函数负责得到验证和测试的结果。具体的命令请查看 README 文件。

9 结论

本文提出了一种使用全连接网络融合多个观测点数据，在使用 Seq2Seq 的结构对未来 6 小时的 PM2.5 进行预测的方法，并取得了（并不）良好的效果。经过大量的实验证明，随着时间的发展，北京空气指标的分布已经有了较大的改变。当然，本文的工作有着太多需要改进的地方，例如没有进行突变预测，没有合理使用观测点的几何位置信息，使用 Graph LSTM 可能是个更好的选择。Seq2Seq 部分的 decoder 也可以得到更精

心的设计。相信，如果有充裕的时间，可以做得更好。

参考文献

[1] Sean Robertson. Nlp from scratch: Translation with a sequence to sequence network and attention. https://github.com/pytorch/tutorials/blob/master/intermediate_source/seq2seq_translation_tutorial.py, 2019.