



中国研究生创新实践系列大赛
“华为杯”第十七届中国研究生数学建模竞赛

学　　校	南京审计大学,天津大学,中国科学院大学
参赛队号	20112870005
队员姓名	1.周尧
	2.郑洁
	3.侯贵洋

中国研究生创新实践系列大赛
“华为杯”第十七届中国研究生数学建模竞赛

题 目 基于深度学习的大雾条件下能见度估计与预测

摘 要：

能见度是了解大气稳定度的天气指标，也是保障交通安全的重要因素。然而雾霾使得能见度大大降低，有时会导致驾驶者和观测者不能对环境作出正确的判读，甚至会引发事故。为了消除安全隐患、提高户外成像系统图像质量，对图像去雾技术进行深入研究很有必要。

对于问题一，首先对机场 AMOS 观测数据进行预处理。第一步检测时间戳缺失，并对有效指标缺失值进行了填充。第二步，对于已有时间戳数据进行完整性检测，数据中的指标均完整。第三步，通过绘制箱线图检测异常值，发现了部分离群点，但是数值较小且到离群点的值连续变化，结合题目背景，保留小部分离群点。我们选取气压、温度、相对湿度、风速四个气象因素，研究它们与能见度的关系。首先分别绘制四个因素与能见度的关系曲线图，可以找出能见度为 3000 时，四个指标的范围，之后对其余数据段分别拟合非线性函数关系，得到的函数关系有三角函数、指数形式等。用拟合得到的函数代替线性函数，进行变换之后，求出回归方程的解析解，得到最终的关系式。

对于问题二，首先将长视频切分成若干个时长为一分钟的短视频，然后将能见度值划分为 6 个区间映射到 6 个类别标签上，进而完成数据集的构造。以能见度估计场景为出发点，采用 Resnet 残差网络提取空间域特征，LSTM 网络提取时间域特征。结合时空域，建立 ResNet + LSTM 融合的深度学习模型来对视频数据的能见度进行估计。通过对交叉熵损失进行可视化，发现网络是收敛的，同时计算出模型在测试集视频上的类别错判率为 17.23%，有效地评估了模型能见度估计的精度。

对于问题三，本文首先利用暗通道先验原理求得透射率，为了消除块状效应，利用引导滤波的方法对求得的透射率进行优化，再通过摄像机与路面之间的几何建模得到道路深度信息，由于大气消光系数可以度量不同能见度下的亮度差异估计能见度，结合大气消散理论，最终得到透射率、道路深度、能见度之间的数学模型，求得在 6:30-7:38 时间段内 100 个时刻点的能见度，6:30 时，能见度为 30.19m，在经过接近一个小时后，大雾慢慢消散，能见度达到 60m，此时路况较之前有所改善。

对于问题四，采用拉格朗日插值和线性插值法进行了数据增强，然后使用擅长捕捉非线性变化的 Seq2Seq+Attention 模型和擅长捕捉线性变化的时间序列 ARIMA 模型结果融合，使用问题三得到的能见度随时间变化规律预测大雾变化趋势，得到了更好的结果，我们发现经过 6637.1s 到 7111.5s 后 MOR 可以达到 150m，对应到当天的时刻为 09:29:48 到 09:37:43。

最后，我们对于模型的优缺点进行评价，提出了一些改进的建议，并且在一定程度上介绍了模型的推广前景。

关键词：联合时空特征 模型融合预测 多项式特征生成 Resnet 残差网络 暗通道先验

目 录

目 录 2
一 问题的背景 3
二 问题的重述 3
三 模型的假设 4
四 符号的说明 4
五 模型的建立与求解 4
5.1 问题一 4
5.1.1 问题一的分析 4
5.1.2 问题一模型的建立和求解 4
5.2 问题二 10
5.2.1 问题二的分析 10
5.2.2 问题二模型的建立 10
5.2.3 问题二模型的求解 12
5.3 问题三 17
5.3.1 问题三的分析 17
5.3.2 能见度测量原理 17
5.3.3 基于暗通道先验的能见度检测算法 19
5.3.4 引导滤波优化算法 21
5.3.5 道路深度计算 21
5.3.6 模型的求解 24
5.4 问题四 26
5.4.1 问题四的分析 26
5.4.2 问题四模型的建立 26
5.4.3 问题四模型的求解 29
六 模型的评价和推广 36
6.1 模型的评价 36
6.2 模型的推广和改进 36
参考文献 37
附录 38

一 问题的背景

能见度是了解大气稳定度的天气指标，也是保障交通安全的重要因素。然而，雾霾的存在使得能见度大大降低，驾驶者和观测者无法用肉眼获得准确的信息，因而不能对环境作出正确的判读，有时会引发事故。当有雾时，大气中存在大量的颗粒介质，如水珠、灰尘等。目标物体的反射光线经过大气时，会被这些介质散射、吸收，光强度衰减，导致图像对比度降低，细节特征被覆盖，色彩失真。图像信息的缺失会给目标判定带来一定的困难，特别是在浓雾天气下，可视度极度下降，直接影响交通运输、室外监控等户外成像系统效用的发挥。为了消除安全隐患、提高户外成像系统图像质量，我们需要对图像去雾技术展开深入研究。

图像去雾算法主要分为两大类：一类是基于物理模型的方法，另一类是基于非物理模型的方法。它们区别主要在于是否利用大气散射模型。

基于物理模型的方法使用大气散射模型。Tan 等人对无雾图像和有雾图像进行对比度统计，发现无雾图像具有更高的对比度，对输入图像进行局部对比度最大化处理实现去雾，但是该算法处理后图像颜色有时会过饱和。Fattal 等人假设光线传输和表面颜色是不相关的，估计出场景的反射率，再估算传输系数。该算法物理可行并且可以取得很好的效果，但不能很好地处理浓雾图像以及假设不成立时的图像。He 等人提出的暗通道先验算法是目前最实用的一种去雾算法，基于暗通道先验估计场景的透射率，得到场景的深度图像，能够很好的恢复无雾清晰图像。但是，当目标场景大部分不满足暗通道先验时，不能有效的恢复图像。

基于非物理模型的方法的提出最初是为了增强图像的对比度和校正图像的颜色，根据视觉感受改善图像的质量。该类方法不采用大气散射模型，按照处理领域分为两类，一类是空域内增强，一类是频域内增强。频域增强利用傅立叶变化转化到频域范围进行处理达到增强目的，包含低通滤波，高通滤波和同态滤波等增强算法；空域增强直接处理图像灰度值实现图像对比度增强，常用的方法是直方图均衡化算法、小波变化、曲波变化、Retinex 算法等。近年来，Retinex 算法成为图像增强算法的热点，它利用色彩恒常性原理实现图像增强。色彩恒常性理论认为，无论在何种照度情况下，人类视觉系统对于物体的颜色感知不会发生改变。该算法将图像分为照度分量和反射分量，照度分量影响图像的动态范围和色彩偏移，反射分量对应于图像自身属性。Retinex 算法通过削弱照度分量保留反射分量来增强图像对比度。最开始由 Land 提出，后续又有人提出了随机路径 Retinex，中心/环绕 Retinex 等。

二 问题的重述

能见度是气象、交通中常见指标，能见度预测是高速公路管理部门和航空公司十分关注的问题。它的影响因素主要是雾和霾。激光能见度仪作为常用的检测能见度的仪器，存在团雾检测精度不高、探测范围小、维护成本高等问题。基于视频的路况（跑道）能见度检测方法在某种程度上能够克服激光能见度仪的不足，但现有的基于视频图像的能见度检测方法很难准确估算能见度。特别地，这些方法并没有充分利用视频的连续信息，所以估计的精度不高，有较大的改进空间。

本项目关注大雾的演化规律。利用视频资料和图片资料，对大雾的消散进行预测。主要解决了以下问题：

（1）建立数学模型描述能见度与地面气象观测因素之间的关系，给出具体的关系式；

（2）根据某机场视频数据和能见度数据，建立基于视频数据的能见度估计深度学习模型，对估计的能见度进行精度评估；

（3）建立不依赖能见度仪观测数据的能见度估计算法，根据高速公路视频截图绘制给定的时间段高速公路能见度随时间变化曲线；

（4）利用问题三得到的能见度随时间变化规律，建立数学模型预测大雾变化趋势、散去的时间。

三 模型的假设

假设 1：假设处理的视频是连续的，不存在跳帧、坏帧等情况。

假设 2：假设大气光全局恒定。

假设 3：《高速公路监控技术要求》中指出高速公路上摄像机采用立柱安装，安装高度在 8~12m 范围内，本文假设安装高度为 10m。

假设 4：假设处理的视频背景结构比较稳定，变化幅度较小。

假设 5：不考虑光照对雾的形成和消散造成的影响。

四 符号的说明

符号	符号说明
$H(x)$	残差网络的输出
$F(x)$	残差映射函数
$C(x)$	目标物与背景的亮度对比度
$L(x)$	目标物的表面亮度
$L_b(x)$	背景亮度
$\Delta L(x)$	目标物与背景的亮度差
J^c	彩色图像的每一个通道
$\Omega(x)$	以像素 x 为中心的一个窗口

五 模型的建立与求解

5.1 问题一

5.1.1 问题一的分析

问题一要求我们建立数学模型描述能见度与地面气象观测因素之间的关系，并给出具体的关系式。对于题目给出的机场 AMOS 观测数据，首先要进行数据预处理。

5.1.2 问题一模型的建立和求解

（1）数据预处理

首先要进行缺失值的检测。

①时间戳缺失检测

第一步，检测时间戳是否存在缺失。我们对题目所给的数据 vis R06-12、wind R06-12、vis R06-15、wind R06-15 进行了时间戳缺失的检测，发现 2019 年 12 月，缺失时间戳为 5 个，取北京时间，为 2019 年 12 月 15 日 9:26、2019 年 12 月 15 日 16:00、2019 年 12 月 15 日 18:36、2019 年 12 月 15 日 20:21 和 2019 年 12 月 16 日 7:18。2020 年 3 月缺失时间戳 5 个，分别是缺失时间戳 5 个，分别是 2020 年 3 月 12 日 9:12、2020 年 3 月 12 日 18:05、2020 年 3 月 12 日 21:29、2020 年 3 月 13 日 1:39 和 2020 年 3 月 13 日 4:11。这里一分钟有三个时间点，都缺失了一个时间点。

对于缺失值进行填充时，只需要对有效指标进行填充即可，RVR_1A 填充的值,t 为缺失时刻点，使用的公式如下：

$$RVR\_1A_t = \frac{RVR\_1A_{t+1} + RVR\_1A_{t-1}}{2}.$$

WS2A 填充值同理。

②已有时间戳数据的完整性检测

第二步，对于已有时间戳数据进行完整性检测。如果存在缺失值，检测结果图中会出现黄色，反之则呈紫色，如图例所示。部分检测结果和图例如下：

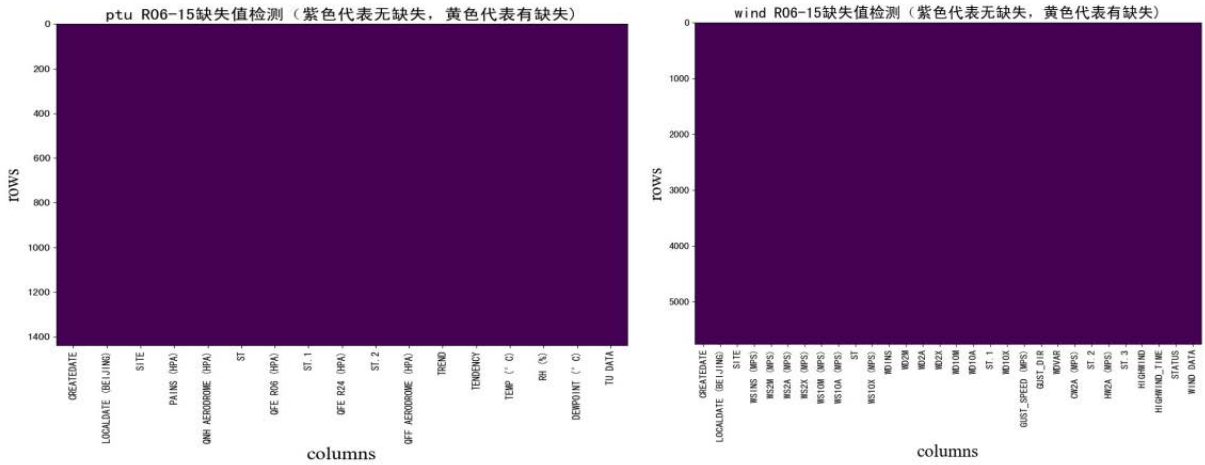

图 1 缺失值检测结果图

图 2 缺失值检测图例

图 1 显示了 ptu R06-15 和 wind R06-15 的缺失值检测结果，图中均为紫色，说明数据中的指标不存在缺失。我们对 vis R06-15、wind R06-12、vis R06-12、ptu R06-12 进行的缺失值检测结果同上，均呈现紫色，说明这些数据中的指标都不存在缺失。

③数据分布检测（异常值检测）

第三步是异常值（离群点）检测。本文通过绘制箱线图的方式进行检测，统计学当中，

根据 3σ 原则，一般将超过箱线图上下四分位数的数值点视为异常值。

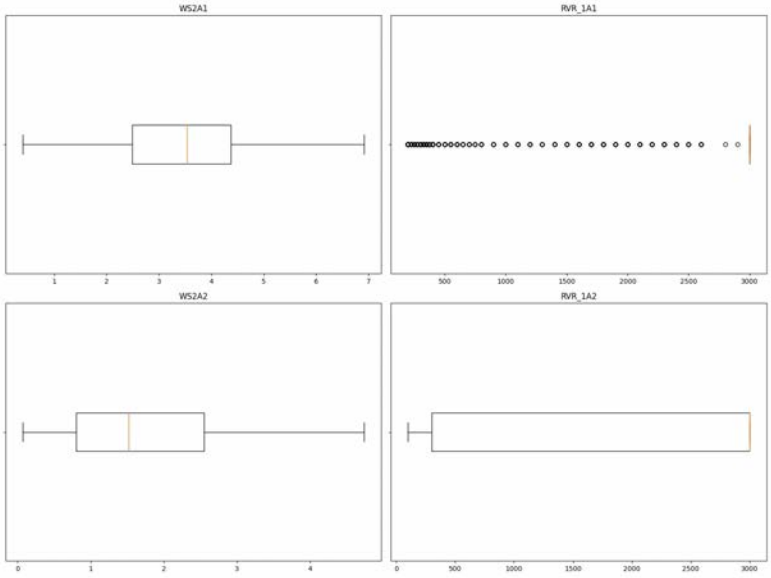

图 3 两个日期下的平均风速以及平均 RAR 分布箱型图

由上图可以看出，两个日期下的平均风速以及平均 RAR 数据分布，数据由明显的左偏或右偏，且部分存在异常值。

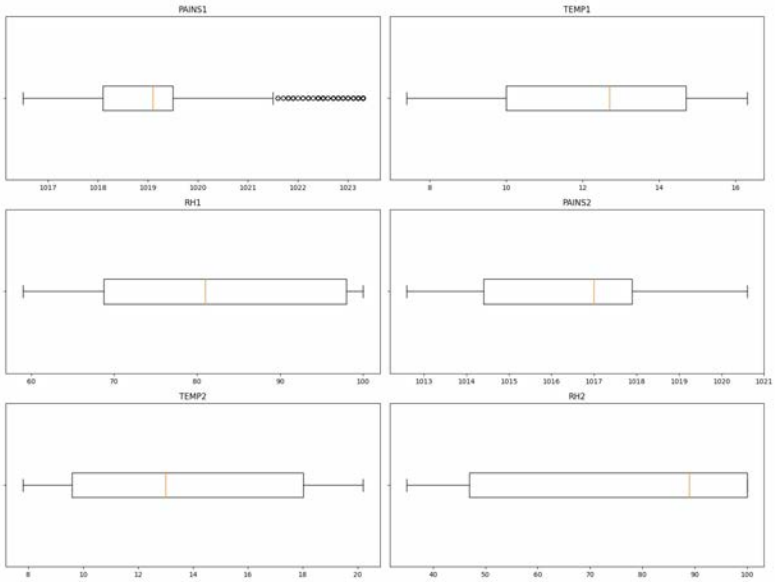

图 4 两个日期下的本站气压、温度、相对湿度分布箱型图

上图是两天的气压、温度、相对湿度数据分布箱线图。通过对各指标数据分布的检测，可以很清晰地看到数据的分布特征。虽然箱型图中出现了部分离群点，将离群点的值记为 k_j，非离群点的值记为 k_i，但是数值较小，并且非离群点到离群点的值是连续变化的，结合题目背景，同时也是为了更好地对题目进行分析计算，保留上图中的小部分离群点。

对数据预处理之后，要建立模型描述能见度与地面气象观测因素之间的关系，我们阅读了参考文献，选取了公认对能见度影响较大的四个因素气压、温度、相对湿度、风速进行研究。由于本问题中要使用的是机场数据，我们选择在机场使用较多的 RVR 定义下的能见度做为因变量。在关系分析中，共使用两个不同日期的缺失值填充后的能见度数据共 11520 个样本。平均风速一共使用 11520 个数据，温度、相对湿度以及气压一分钟使用一

个数据样本，共 2880 数据。因此在画能见度值和温度、相对湿度、气压关系时，对能见度值一分钟四个点的数值计算平均值来代表这一分钟的能见度，取完平均后共 2880 个数据，与和温度、相对湿度、气压一一对应。

综合了两个日期的数据，对四个气象因素和能见度的关系绘制的关系曲线图分别如下。

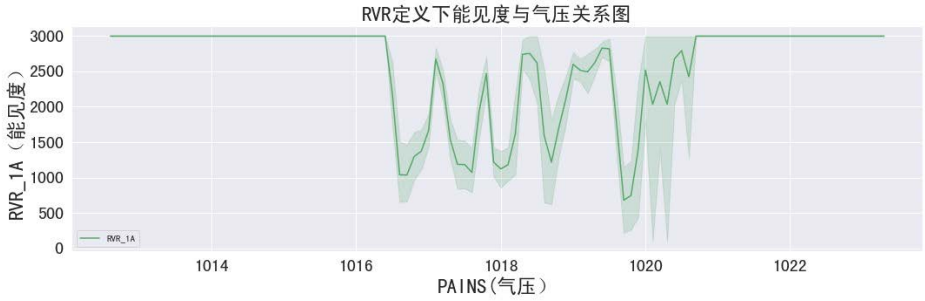

图 5 RVR 定义下能见度与气压关系

首先是 RVR 定义下能见度与气压关系图，数据中的气压相关指标有三个，分别是本站气压、飞机着陆地区最高点气压、修正海平面气压，由于第一问考虑的是近地面的气象因素，我们在这里选择本站气压进行分析。当气压位于 0-1014、大约 1021 以上时，能见度为 3000。当气压（Pains）在大约 1016 到 1021 之间时，它与能见度之间的关系曲线呈现震荡形式。

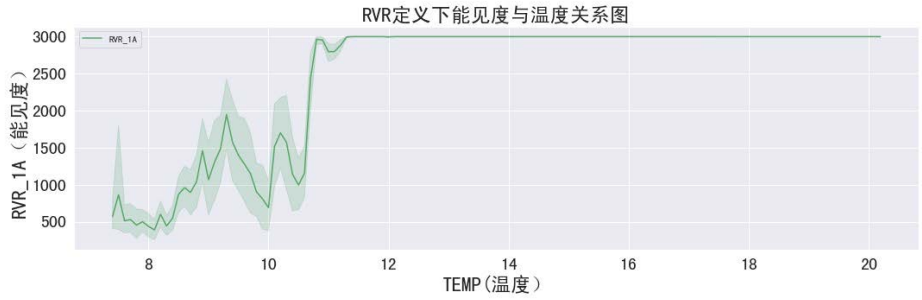

图 6 RVR 定义下能见度与温度关系

上图是 RVR 定义下能见度与温度关系图，当温度位于大约 11 以上时，能见度为 3000。当温度在大约 7 到 11 之间时，随着温度升高，能见度整体呈现上升趋势，温度在 9 到 11 之间时有部分震荡。

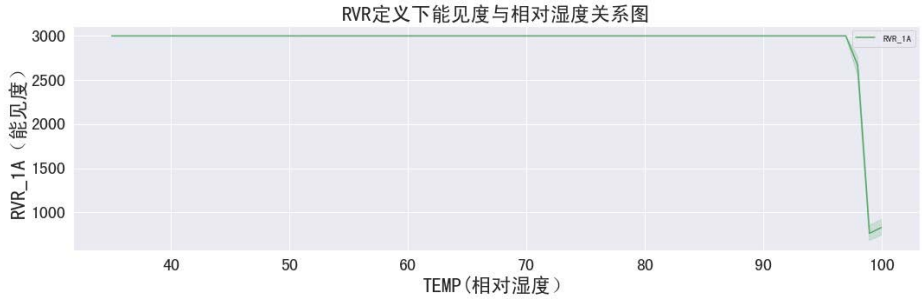

图 7 RVR 定义下能见度与相对湿度关系

上图为 RVR 定义下能见度与相对湿度的关系图，当温度位于大约 96 以下时，能见度为 3000。当相对湿度在大约 95 到 100 之间时，随着温度升高，能见度整体呈现上升趋势。

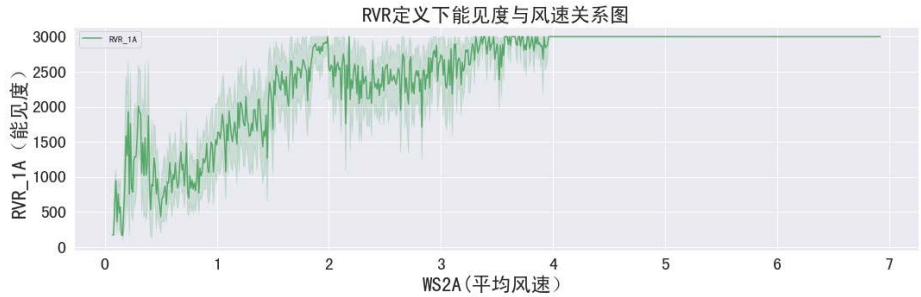

图 8 RVR 定义下能见度与风速关系

在分析RVR定义下能见度与风速的关系时，本题中给出的风速数据有2分钟平均风速、2 分钟平均风向、2 分钟平均垂直风速，我们选平均风速作为代表，上图是析 RVR 定义下能见度与 2 分钟平均风速的关系图，当风速位于大约 4 以上时，能见度为 3000。当平均风速在 0-4 之间时，随着平均风速增大，能见度整体呈现出上升趋势。

根据以上四幅图进行分段，当气压位于 0-1014、大约 1021 以上，温度位于大约 11 以上，温度位于大约 96 以下，风速位于大约 4 以上时，能见度为 3000。接下来对其余数据段求解关系式，拟合非线性函数关系。

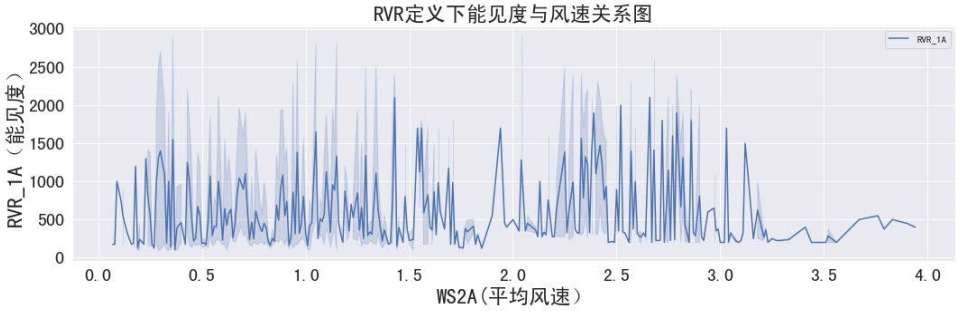

图 9 RVR 定义下能见度与风速关系

上图是对于能见度不等于 3000 时的区间进行拟合的关系图，我们得到的能见度与平均风速的关系式为

$$f(x) = -153.2\sin(x-\pi) - 4.231(x-10)^2 + 824.$$

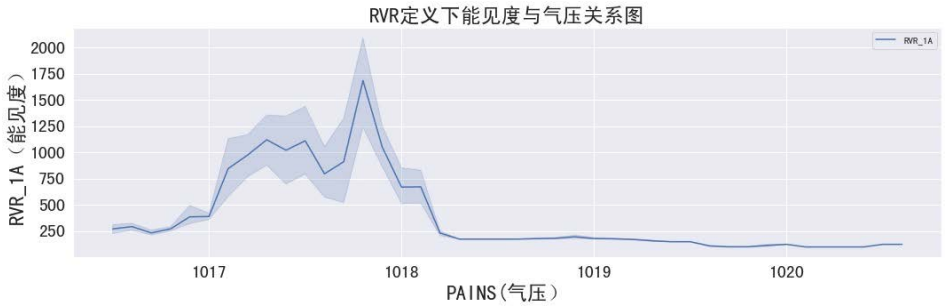

图 10 RVR 定义下能见度与气压关系

能见度与气压的关系式为

$$f(x) = 850.6\sin(0.9376x - 173.7).$$

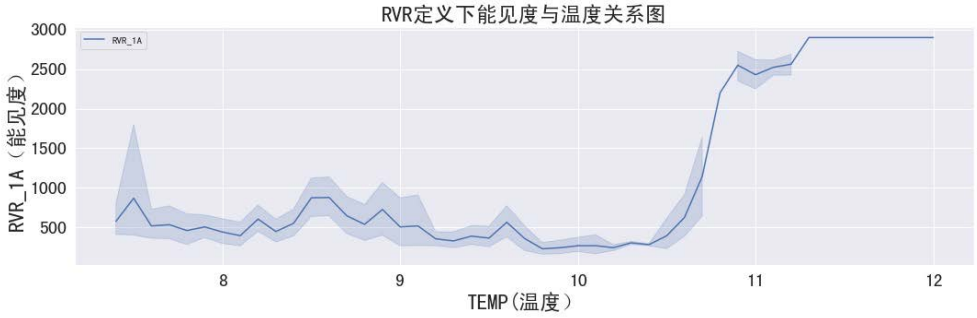

图 11 RVR 定义下能见度与温度关系

能见度与温度的关系式为

$$f(x)=0.7408x^{0.3014}.$$

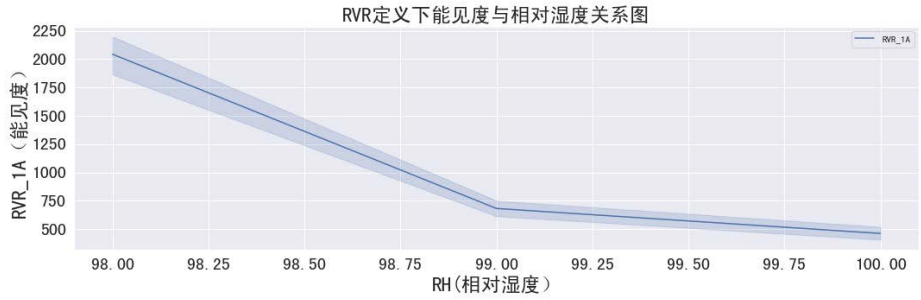

图 12 RVR 定义下能见度与相对湿度

能见度与相对湿度的关系式为

$$f(x)=\frac{157.2x-1.416\times10^4}{x-97.61}.$$

能见度与以上四个气象因素即风速、气压、温度、相对湿度的关系式拟合精度用 R-square 来衡量，依次是 0.6622、0.7745、0.8317、0.8843，拟合效果均令人满意，优度依次递增。所以可以用拟合得到的函数代替线性函数。进行完变换之后，接下来求回归方程的解析解。通过对最小二乘函数求导，当导函数为 0 时，就是最小二乘的解。

先对最小二乘进行变形，得到矩阵形式：

$$J(\theta)=\frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2=\frac{1}{2}(X\theta-Y)^T(X\theta-Y)$$

展开矩阵函数之后，对 $J(\theta)$ 求导并令导数为 0，得到

$$\frac{\partial J(\theta)}{\partial\theta}=\frac{1}{2}(2X^TX\theta-X^TY-X^TY)=0$$

得

$$\theta=(X^TX)^{-1}.$$

使用 sklearn 包实现上述求解解析解的过程，得到最终关系式为

$$y=2.3405\times10^3-0.3562x_1+1.1581x_2+1.3468x_3-2.8767x_4$$

常数项后 x_1、x_2、x_3、x_4 分别对应风速、气压、温度、相对湿度，各个 bint 回归系数的区间估计如下表（此处的系数是经过变换后的）。

表 1 各个 bint 回归系数的置信区间

系数	置信下限	置信上限
常数项	1.55E+03	3.13E+03
风速	-0.997	0.2846
气压	1.027	1.2892
温度	1.149	1.5477
相对湿度	-3.3673	-2.3862

对最终的关系式进行检验，统计量 $R^2 = 0.8018$ ，P 值< 0.0001，拟合优度很高。

5.2 问题二

5.2.1 问题二的分析

问题二要求我们建立基于视频数据的能见度估计深度学习模型，并对估计的能见度进行精度评估。为了提高估计的能见度，不仅要选择适合该题目场景的网络，而且也要将时域、空域特征联合起来。

5.2.2 问题二模型的建立

近年来，基于计算机深度学习模型的特征学习引起研究人员的广泛关注，基于深度学习模型的特征提取也成为一个重点的研究对象，作为深度学习模型之一的卷积神经网络(convolution neural network, CNN)在图像识别、语音识别、视频处理等领域取得了巨大成功，基于卷积神经网络的特征提取，可直接以图像矩阵作为模型的输入，避免了像传统机器学习那样前期对图像数据的各种复杂的预处理，实现了监督式的学习，由局部到全局、由低级到高级的特征提取。CNN 一般由输人层、输出层和多个隐藏层组成，隐藏层一般包括卷积层、池化层、激活层和全连层。

CNN 在视频中应用的一个方法是对每一帧用 CNN 进行识别，但这种方法只考虑到了空间上的视觉效果，没有考虑到行为运动是一个序列，在时间维度上还有关联，连续帧之间有一定耦合，是相互关联的。本文针对视频分析中空间和时间两个维度的特征，提出一种卷积神经网络、循环神经网络的融合模型(ResNet-LSTM)，.对于空间维度，将单帧图像输人 ResNet 模型，提取空间维度特征；对于时间维度，将多帧叠加后的空间维度特征作为输人，输人到循环神经网络(LSTM)模型，然后将 ResNet-LSTM 模型的输出经过 Softmax 输出作为结果，得到一个多模型融合的视频能见度检测网络。

本文提出网络模型优势在于将空间域和时间域的特征分开提取，首先在静止的图片上提取特征，随后在时间序列上分别进行拟合，模型的两层神经网络相互独立，训练过程分开进行，所以提取时间域的 LSTM 的反向传播不会贯穿到 ResNet，从而一定程度上避免时间域和空间域上特征提取的混淆。

（1）空间域的 ResNet 模型

由于神经网络的梯度在反向传播的过程中会不断衰减，如果一个神经网络过深的话就会出现梯度消失现象，导致该网络的性能急剧下降。

残差网络(Residual network, ResNet)可以有效解决以上问题，残差网络一个单元的结构如图所示。在该图中可以看到残差网络在普通的神经网络中增加了一个跃层的连接，使得网络的输出从原来的 $H(x) = F(x)$，变成 $H(x) = F(x) + x$。这样做的好处是使得残差网络在每一个局部模块的训练目标转化成残差值 $H(x) - x$，这也是残差网络中“残差”这一名字

的来源。当网络比较深的时候，对网络进行训练以使得每个模块的残差 $H(x)-x$ 尽量向 0 靠拢，此时网络每个模块的输出基本与输入相同，即 $H(x)=x$ 。基于这样的跃层连接模块，残差网络中浅层输出能够直接传递到更深的层次，以使神经网络的性能不会因为层数的增长而急剧下降。通过这种跃层连接模式，我们可以利用更深的网络得到输入信号的深层次特征。

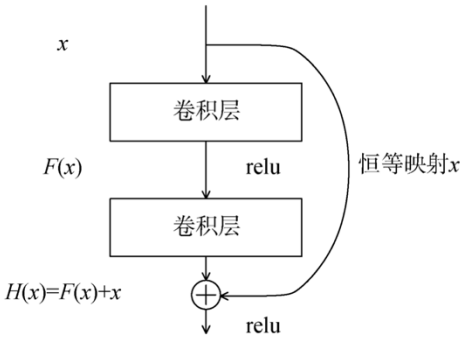

图 13 残差网络一个单元的结构图

残差块结构如上图所示，其中 x 为输入，$H(x)$ 为输出，$F(x)$ 为残差映射函数，weight layer 为卷积层。

（2）处理时间域的 LSTM 模型

LSTM（长短期记忆网络）模型的核心在于细胞的状态和穿过细胞的那条水平线。细胞状态类似传送带，直接在整个链上运行，只有少量的线性交互。因此信息流传容易保持不变。

LSTM 模型通过三个这样的基本结构来实现信息的保护和控制。三个门分别输入门、遗忘门和输出门。因为拥有遗忘门、记忆门来更新内部参数，长短期记忆网络模型更擅长处理语言、股票等长序列数据，同时能模拟变量之间的非线性关系。它的前向计算和后向计算功能，分别用于结果预测和参数。典型的 LSTM 神经网络结构如下。

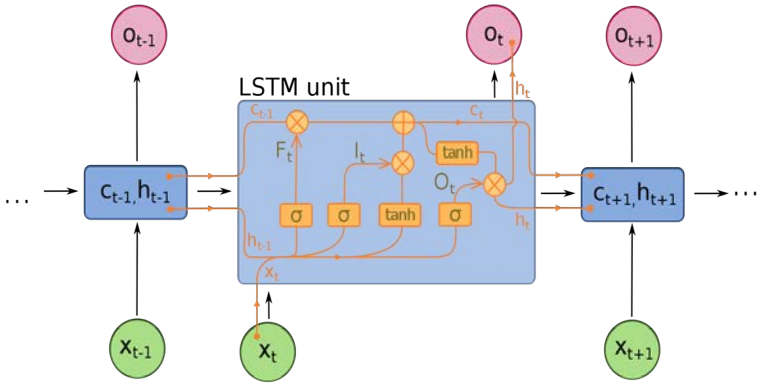

图 14 LSTM 模型示意图

前向计算过程在每个时刻 t 输入 X_t，得到预测值 o_t 和被送往下一个时刻参与计算的 c_t 和隐层变量 h_t。完整的前向计算过程为

$$F_t=\sigma_g(W_fX_t+U_fh_{t-1}+b_f)$$
$$i_t=\sigma_g(W_iX_t+U_ih_{t-1}+b_i)$$
$$o_t=\sigma_g(W_0X_t+U_0h_{t-1}+b_0)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \sigma_c(W_c X_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \cdot \sigma_h(c_t)$$

后向计算可以使用 BPTT 算法，反向传播梯度来更新参数，可以使用均方根误差（RMSE）作为损失函数 L，其中 y_t 为真实序列值。

$$L = \sqrt[2]{\frac{1}{n}\sum_{t=1}^{n}(o_t - y_t)^2}$$

参数 W（包括$W_f, W_i, W_0, W_c, U_f, U_i, U_0, U_c, b_f, b_i, b_0, b_c,$）的更新公式为

$$W \leftarrow W - \frac{\partial L}{\partial W}\eta.$$

LSTM 模型的预测评价指标，可以使用 RMSE 和 MAPE，其中 RMSE 为均方根误差，MAPE 为平均绝对百分比误差，它们的数值越小，模型效果越好。具体计算的公式如下：

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right|$$

5.2.3 问题二模型的求解

5.2.3.1 数据集的构造以及数据预处理

在构造数据集时，将长达 11 个多小时的长视频切分成若干个时长为 1 分钟的短视频，视频的帧率为 25 帧/s，即每个短视频有 60*25 = 1500 帧。

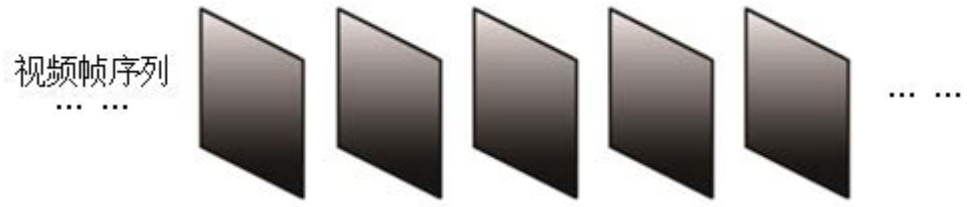

图 15 视频帧序列示意图

在完成了视频切分的基础上，接下来对视频打标签，对题目给的每分钟四个能见度数据取平均即为视频对应能见度数据。由于能见度数据取值较多、范围较广，将一定范围内的能见度映射到 1 个类别标签上，能见度范围与类别标签以及 one-hot 编码的对应关系如下表所示。

表 2 能见度范围与类别标签以及 one-hot 编码的对应关系

类别标签	能见度 L 范围/m	one-hot 编码表示
1	50<L<100	[1,0,0,0,0,0]
2	100<L<200	[0,1,0,0,0,0]
3	200<L<500	[0,0,1,0,0,0]
4	500<L<1000	[0,0,0,1,0,0]
5	1000<L<3000	[0,0,0,0,1,0]
6	L>3000	[0,0,0,0,0,1]

拥有了视频与类别标签的对应关系，进而可以训练模型学习视频到类别的映射。数据集的构造如下表所示。

表 3 数据集构造

被切分后的视频		类别标签
视频 1		类别标签 1
视频 2		类别标签 2
视频 3		类别标签 3
……	⇒	……
……		……
视频 n-1		类别标签 n-1
视频 n		类别标签 n

5.2.3.2 网络模型结构及各层参数

在能见度估计的场景下，Resnet 的残差思想在空间域可以较好的体现出来，“雾浓—雾细”、“晴天—雾天”等场景的特征可以很理想的被 Resnet 网络所关注到并提取出来。同时由于视频帧与帧之间存在着较为紧密的联系，采用 LSTM 模型提取视频时间域上的特征。结合空域和时域，建立 ResNet + LSTM 融合的深度学习模型来对视频数据的能见度进行估计。

网络的输入部分：输入尺寸为 Batchsize*1500*352*288*3，Batchsize 为网络一次训练所选取的样本数（视频数量），1500 为每个视频中包含帧的数目，352*288 代表着每一帧图像的尺寸，同时每一帧的图像划分 RGB 三个通道。

Resnet34 网络部分：视频经过预处理之后，会进入到 Resnet34 网络结构中，Resnet34 网络中除了包含卷积模块的结构，还包含了 skip connection 结构，残差的思想就是依靠 skip connection 结构来实现的。

全连接层部分：全连接层输出的尺寸为 Batchsize*1500*m，m 为对应到每一帧图像上的特征编码表示，这里 m = 6。

LSTM 网络输入部分：通过 Resnet34 网络提取了空间域中的特征后，接下来提取时域中的特征。全连接层输出的每一帧图像的特征编码作为 LSTM 网络的输入，即 LSTM 网络的输入尺寸为 Batchsize*1500*6。

LSTM 网络部分：共三层结构，由 LSTMs 单元叠加构成，每一层 LSTM 的隐藏单元都进行初始化，前一层的输出作为后一层的输入，如此迭代得到 LSTM 三层循环神经网络。最后的输出维度为 Batchsize*1500*12。

全连接层部分：输入参数为 Batchsize*1500*12，权重矩阵维度为 6*12，输出维度为 Batchsize*1500*6，这里的 6 对应到类别标签的数目。

Loss Function 计算部分：将类别标签的 one-hot 编码以及网络输出的类别概率编码代入交叉熵损失函数中计算交叉熵损失。

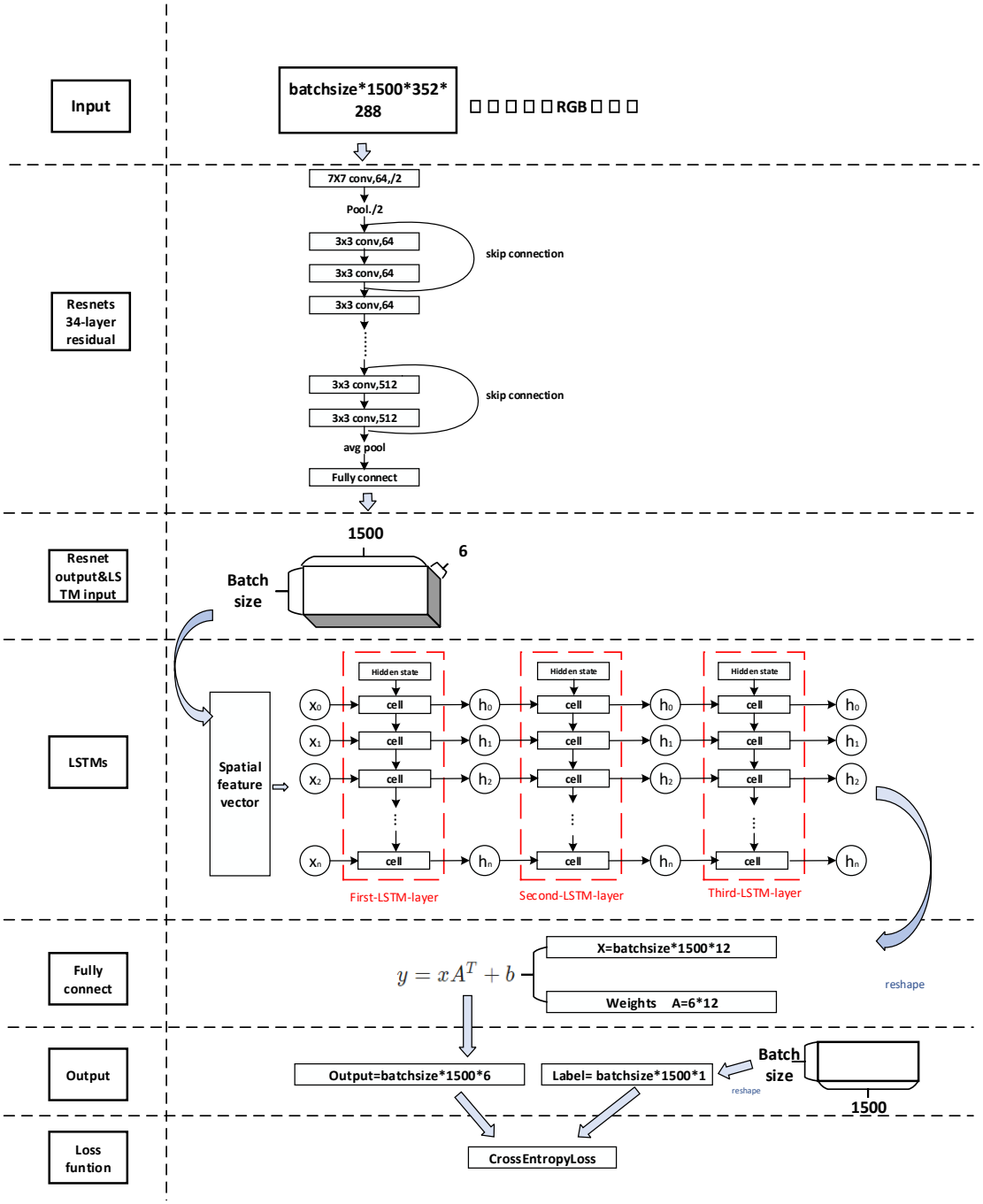

图 16 Resnet+LSTM 融合的网络模型结构图

Resnet+LSTM 融合的网络模型结构图如上图所示。

5.2.3.3 网络训练参数

（1）训练集、验证集、测试集的划分

以 6:2:2 的比例将数据集划分为训练集、验证集、测试集，如下图所示。

图 17 数据集划分

（2）初始学习率的选择

初始学习率 lr_init = 0.01。

（3）ReduceLROnPlateau

当准确率不再发生变化时，学习率就会发生改变。本文设置将学习率的原始值降低为原来的 1/2。

（4）Batch_size

网络一次训练所选取的样本数 Batch_size = 64

（5）epoches

算法设置的迭代轮数 epoches = 100

（6）是否预训练

算法的 Resnet34 网络部分调用了 pytorch 官方提供的预训练的模型。

5.2.3.4 损失函数

Softmax()函数是非线性函数，输入是一个实数向量并返回概率分布。输出是概率分布，元素都是非负的，且所有元素之和为 1。LogSoftMax()函数是在 Softmax()函数计算结果基础上，计算 Log 值。

$$soft\max(x_i)=\frac{\exp(x_i)}{\Sigma_j \exp(x_j)}$$

$$LogSoft\max=\log(soft\max(x_i))=\log(\frac{\exp(x_i)}{\Sigma_j \exp(x_j)})$$

NLLLoss()函数即负对数似然损失函数，对 LogSoftMax()的结果进行取负对数似然。NLLLoss()中，是输入向量，为分类标签。

$$NLLLoss(x,class)=-x[class]$$

交叉熵函数将 LogSoftMax()函数与 NLLLoss()函数相结合得到

$$loss(x,class)=-\log(\frac{\exp(x[class])}{\Sigma_j \exp(x[j])})=-x[class]+\log(\Sigma_j \exp(x[j]))$$

5.2.3.5 优化器选择

损失函数是用来对计算分类结果的错误程度进行衡量的，优化器在于将对损失函数进行求导，最小化损失函数值。网络计算中，参数主要是权重信息作为求导的参数，权重数值下降的幅度为梯度，梯度下降策略种类很多，本文中选择 Adam 优化器。

$$\begin{cases} m_t=\beta_1 m_{t-1}+(1-\beta_1)g_t \\ v_t=\beta_2 v_{t-1}+(1-\beta_2)g_t^2 \\ \hat{m}_t=\dfrac{m_t}{1-\beta_1^t},\hat{v}_t=\dfrac{v_t}{1-\beta_2^t} \\ W_{t+1}=W_t-\dfrac{\eta_0}{\sqrt{\hat{v}_t}+\varepsilon}\hat{m}_t \end{cases}$$

Adam 中动量直接并入梯度一阶矩的估计。上式中，m_t, v_t 分别是一阶动量项和二阶动量项。β_1, β_2 是动力值大小，通常分别取 0.9 和 0.999，$\hat{m}_t, \hat{v}_t$ 是各自的修正值。W_t 为 t 时刻

也即是第次迭代模型的参数，$g_t = \Delta J(W_t)$ 为 t 次迭代代价函数关于 W 的梯度值，ε 为取值很小的值，一般取 1e-8，避免分母等于 0。Adam 算法伪代码如下表所示。

表 4 Adam 算法伪代码

Adam 算法
Require：步长 ε（设为 0.001）
Require：矩估计的指数衰减速率，ρ_1，ρ_2 在区间[0,1)内。（默认分别为 0.9 和 0.999）
Require：用于数值未定的小常数 δ（设为 10^{-8}）
Require：初始参数 θ
初始化一阶和二阶矩变量 s=0,r=0
初始化时间步 t=0
while 没有达到停止准则 do
从训练集中采包含 m 个样本 $\{x^{(1)},...,x^{(m)}\}$ 的小批量，对应目标为 $y^{(i)}$
计算梯度：$g \leftarrow \frac{1}{m}\nabla_\theta \Sigma_i L(f(x^{(i)};\theta), y^{(i)})$
$t \leftarrow t+1$
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：$\hat{s} \leftarrow \frac{s}{1-\rho_1^t}$
修正二阶矩的偏差：$\hat{r} \leftarrow \frac{r}{1-\rho_2^t}$
计算更新：$\Delta\theta = -\varepsilon \frac{\hat{s}}{\sqrt{\hat{r}+\delta}}$
应用更新：$\theta \leftarrow \theta + \Delta\theta$
end while

5.2.3.6 网络精度评估

5.2.3.6.1 交叉熵损失可视化

为了验证网络的收敛性，对交叉熵损失的值进行可视化。从下图中可以较为明显的看出，网络是收敛的。

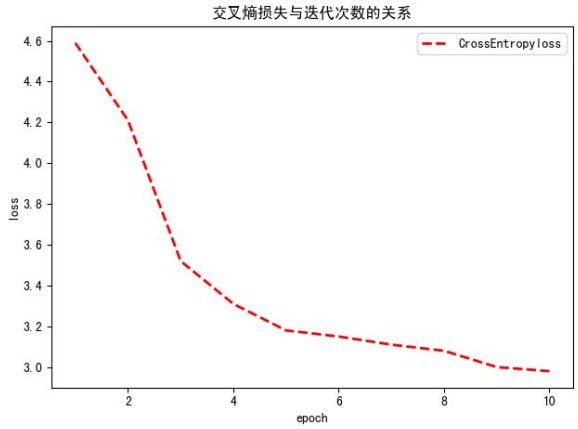

图 18 交叉熵损失值可视化

5.2.3.6.2 类别错判率评估

测试集中共有 138 个视频，将测试集视频输入到训练好的 Resnet+LSTM 融合模型中，模型输出的类别标签记为β_i，数据集中测试集视频对应的类别标签(真实值)记为y_i，定义比较标记ψ_i、标记累加值Q为：

$$\psi_i = \begin{cases} 1, \beta_i = r_i \\ 0, \beta_i \neq r_i \end{cases} \quad i = 1,2,...,138$$

$$Q = \sum_{i=1}^{138} \psi_i$$

根据比较标记ψ_i、标记累加值Q,进而得到错判率η为：

$$\eta = \frac{138 - Q}{138} \times 100\%$$

经过模型计算值与标签真实值比较分析，最终得出模型错判率$\eta = 17.23\%$，错判的区域集中于低能见度视频样本。

5.3 问题三

5.3.1 问题三的分析

问题三要求针对题目提供的一段视频，建立不依仪观测赖能见度数据的能见度估计算法，首先利用暗通道先验原理求得大气透射率，通过引导滤波对求得的大气透射率进行优化，之后结合大气消散模型中消光系数与能见度的关系，建立道路深度、透射率、和能见度之间的数学模型，最终得到能见度随时间变化曲线。

5.3.2 能见度测量原理

5.3.2.1 大气水平能见度的定义

能见度(VIS)作为表征大气透明度的一个重要指标，指目标物的人眼可见距离。其定义为具有正常视力的人在当时天气条件下观测目标物时，能从背景上分辨出目标物轮廓的最大距离。但即使在观测者视力正常的情况下，每个人感受到的能见度不仅和大气光学状态有关，还和各目标物的状态、视角、背景等多方面因素有关。因此，为了使能见度可以单纯地表征大气的光学状态，在气象上的能见度定义为：标准视力眼睛观察水平方向以天空为背景的黑体目标物（0.50°~5°）能从背景上分辨出目标物轮廓的最大水平距离。

5.3.2.2 大气消散模型

根据能见度的定义可以知道能见度决定因素之一为目标物与背景的对比即亮度差，设在距离一目标物为x处，区分目标物与背景之间的亮度对比$C(x)$定义为：

$$C(x) = \frac{|L(x) - L_b(x)|}{L_b(x)} = \frac{|\Delta L(x)|}{L_b(x)}$$

其中，$C(x)$为目标物与背景的亮度对比度，$L(x)$为目标物的表面亮度，$L_b(x)$为背景亮度，$\Delta L(x)$为目标物与背景的亮度差。

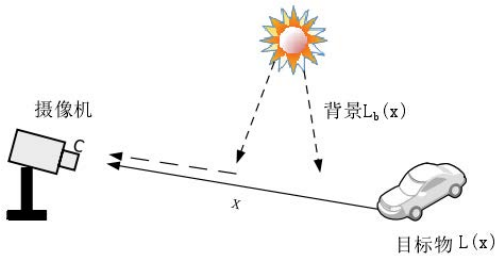

图 19 目标识别条件

若目标物与背景亮度一样，即 $L(x)=L_b(x)$，即 $C(x)=0$，不能分辨出目标物。若目标物为黑体，亮度为 0，即 $L(x)=0$，则 $C(x)=1$，目标物可以很容易的识别出来，显然亮度差是决定目标能否被识别的关键。

大气对光线主要有散射与反射影响，主要是由气体分子与悬浮颗粒造成。大气中悬浮颗粒主要包括空气分子（氮气、氧气、二氧化碳、水蒸气等）与气溶胶微粒（烟、霾、雾、尘埃等）。在高空中分子与气溶胶散射为主要原因，而在地面与低空，气溶胶是散射的主要因素。

从人眼的感官来讲，人眼看到的物体的亮度并非物体的固有亮度，而是由两部分组成，即空气亮度与经过削弱的物体固有亮度。空气亮度为大气光线进入人眼的亮度，一般可认为是观测者与物体之间的空气柱亮度，也可称空气亮度为目标物的背景亮度。而经过削弱的固有亮度是物体固有亮度经空气中颗粒物、气溶胶等散射与吸收后的亮度，该亮度的削弱可以用大气消光系数为度量计算。

Lamber-Beer 定律对大气的消光特性进行了描述，解释了入射光与透射光、消光系数之间的关系。设某光源的入射光通量为 φ_0，经过距离 x 后透射的光通量为 φ_t，则入射光通量与透射光通量之间满足：

$$\varphi_t=\varphi_0\cdot\tau=\varphi_0\cdot e^{-kx}$$

其中 τ 为光传播距离 s 的透射率，k 为大气（介质）的消光系数，kx 为光学厚度。则可以得到距离 x：

$$x=-\frac{1}{k}\ln\frac{\varphi_\tau}{\varphi_0}$$

令 $\varepsilon=\dfrac{\varphi_t}{\varphi_0}$，则 $x=-\dfrac{1}{k}\ln\dfrac{1}{\varepsilon}$。

设在距离观测点 x 处存在目标物，则目标物的视亮度与目标物的固有亮度及背景环境亮度之间满足：

$$L=L_0e^{-\sigma x}+L_b(1-e^{-\sigma x})$$

其中 L 为目标物的视亮度，L_0 为目标物固有亮度，L_b 为背景环境亮度，σ 为大气消光系数，x 为观测点与目标物之间距离。

由上述公式可以看出在大气消光的影响下，目标物固有亮度以 $e^{-\sigma x}$ 倍数衰减，而背景环境亮度反而在以 $1-e^{-\sigma x}$ 倍数衰减，故根据 Koschmieder 定律知，综合固有亮度衰减、背景亮度增强作用下目标物与背景亮度差会降低，影响到视觉效果，即会造成能见度降低，则可进一步得到：

$$L-L_b=(L_0-L_b)e^{-\sigma x}$$

令 $C=L-L_b$，$C_0=L_0-L_b$，那么上述公式变为：

$$C=C_0e^{-\sigma x}$$

其中C为目标物的视亮度与背景亮度差，C_0为目标物的固有亮度与背景亮度差。在此基础上可以得到：

$$\varepsilon = e^{-\sigma x} = \frac{C}{C_0}$$

称ε为视觉对比阈值，其取值存在两种规定：气象上通常为 0.02，CIE 发布标准中规定ε为 0.05。

5.3.2.3 能见度计算

根据能见度的定义，可以看出关于能见度的计算存在白天与夜间差异，由于题目中所提供的高速公路视频截图的拍摄环境为白天，故本文只讨论白天能见度的计算过程。

Koschmieder 定律将大气能见度与大气消光系数联系起来，形成了大气能见度定量计算的理论基础。世界气象组织(World Meteorological Organization, WMO)定义了气象光学距离，即指一个色温为 2700K 的白炽灯发出的平行光通量被大气衰减到 5%所经过的路程，称该水平视程为气象能见度。在这种定义下视觉对比阂值ε为 0.05。

即$\varepsilon = e^{-\sigma x} = 0.05$,将该值代入公式：

$$x = -\frac{1}{k}\ln\frac{1}{\varepsilon}$$

即可得到白天的能见度：

$$MOR = \frac{2.996}{\sigma}$$

其中σ为大气消光系数，可以看出大气消光系数与能见度密切相关。准确检测公路能见度的难题可以转化为通过估计大气消光系数来解决。因此本文将通过大气消光系数来计算最终的大气能见度。

5.3.3 基于暗通道先验的能见度检测算法

暗通道先验原理是由何恺明博士于 2009 年提出的，由前一节提到的能见度计算方法可以知道大气消光系数σ是测算大气能见度不可获取的重要因素。暗通道先验是通过求取暗通道的大气透射率，然后根据衰减定律求出大气消光系数，最后通过柯西米德定律反演得到大气能见度测算值。

5.3.3.1 暗通道先验原理

暗通道先验的原理为：在拍摄的图像中，绝大多数非天空局部区域存在 RGB 三个通道中至少一个通道的值趋近于 0，即该区域光强度极小。下图为暗通道原理示意图，对图像进行分窗口处理，选取图像中以x为中心的区域Ω，先选取窗口内的最小值，再对比 R、G、B 三个通道的值，选取三通道中的最小值为暗通道值，得到暗通道图像。

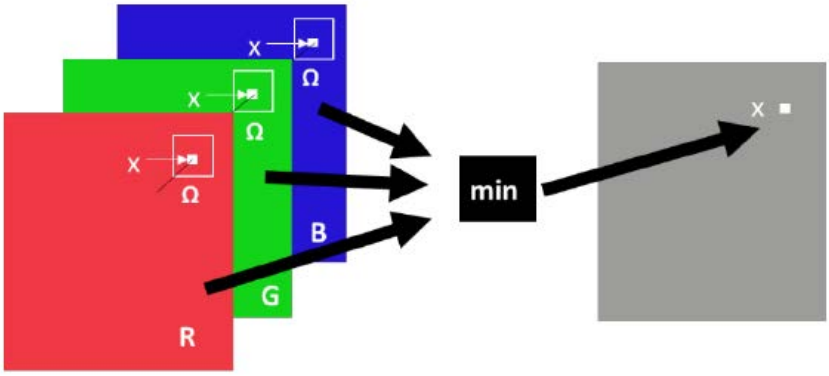

图 20 暗通道原理示意图

具体可以表示为：

$$J^{dark}(x)=\min_{y=\Omega(x)}(\min_{C\in\{r,g,b\}}J^c(y))$$

式中 J^c 指的是彩色图像的每一个通道，$\Omega(x)$ 是以像素 x 为中心的一个窗口。暗原色先验认为，对一幅无雾图像的非天空区域，J^{dark} 具有极低的像素值，即 $J^{dark}\to 0$。

假设以 x 为中心的邻域窗口里面 $\Omega(x)$ 的透射率是相等的，记为 $\tilde{t}(x)$。取邻域内最小值可以得出

$$\min_{y=\Omega(x)}(I^c(y))=\tilde{t}(x)\min_{y=\Omega(x)}(J^c(y))+(1-\tilde{t}(x))A^c$$

这里最小值是对三原色通道分别独立求出的，上式与下式等同：

$$\min_{y=\Omega(x)}(\frac{I^c(y)}{A^c})=\tilde{t}(x)\min_{y=\Omega(x)}(\frac{J^c(y)}{A^c})+(1-\tilde{t}(x))$$

再对上式取三原色通道中的最小值，可以得：

$$\min_c(\min_{y=\Omega(x)}(\frac{I^c(y)}{A^c}))=\tilde{t}(x)\min_c(\min_{y=\Omega(x)}(\frac{J^c(y)}{A^c}))+(1-\tilde{t}(x))$$

根据暗原色先验知识，无雾图像的暗原色通道趋于 0，大气环境光 $A^c>0$，可以得到：

$$\min_c(\min_{y=\Omega(x)}(\frac{J^c(y)}{A^c}))=0$$

将上式代入之前的表达式，可以得到透射率为

$$\tilde{t}(x)=1-\min_c(\min_{y=\Omega(x)}(\frac{I^c(y)}{A^c}))$$

5.3.3.2 能见度求取

光在介质中传播时，一部分被吸收转化为热能释放，另一部分在过程中被颗粒散射，光传播偏离原方向，丢失一部分，余下进入摄像机。此时，透射部分光与入射光强之间符合朗伯一比尔定律。大气消光系数与透射率之间满足如下式的关系：

$$t(x)=e^{-\sigma d(x)}$$

其中，σ 为消光系数，d 为目标物与接收点之间的距离，一般通过标记测算得到。

因此得到能见度与透射率的关系为：

$$MOR=\frac{-2.998\times d(x)}{\ln[t(x)]}$$

5.3.4 引导滤波优化算法

由于图像一定区域块内的透射率$t(x)$不总是常量，所以基于暗通道先验原理获得的透射率较粗糙，这会导致透射率图存在块状效应。为了消除块状效应，本文利用引导滤波法进一步对透射率的计算结果进行优化。

引导滤波是边缘保留的平滑算子，在滤波过后的透射率与原有雾图像满足如下的线性关系：

$$t(i)^{'} = a_k I(i) + b_k, \forall i \in W_k$$

其中，W_k是以像素k为中心的矩形窗口，(a_k, b_k)是矩形窗W_k的线性系数、如果以费用函数来衡量$t(i)$与$t(i)^{'}$的差异，则：

$$E(a_k, b_k) = \sum_{i \in w_k} ((t(i) - a_k I(i) - b_k)^2 + \varepsilon a_k^2)$$

公式中，ε是控制平滑程度的正则化参数，目的是为了防止a_k过大。根据线性回归关系就可以求解系数(a_k, b_k)，图像中的每个像素均被多个线性函数描述。因此计算某一像素i的输出值，只需要将包含像素i的所有输出求算数平均就是透射率，公式描述为：

$$t(i)^{'} = \frac{1}{|w|} \sum_{k:i \in w_k} (a_k I(i) + b_k) = \bar{a_i} I(i) + \bar{b_i}$$

其中，$|w|$表示像素的个数，透射率经引导滤波精细化后，由能见度计算公式得到实际能见度：

$$MOR = \frac{-2.998 \times d(x)}{\ln[t(x)^{'}]}$$

5.3.5 道路深度计算

本文以高速公路监控图像为研究基础，实际场景为高速公路环境，标定关键点可以结合高速公路固有的路面车道线构造尺寸数据。根据《公路交通安全设施设计细则》对车道分界线线长进行规定，每一条车道分界线的长度为6m，相邻车道分界线之间的距离为9m。本文采用车道分界线的数据对摄像机进行标定。在高速公路监控中，可以利用云台转动与变焦功能对摄像机前后一定范围内视频图像采集，确保获取图像中具有车道分界线。

图 21 摄像机标定选点示意图

摄像机标定选点示意图如上图所示，其中红色标记为有雾图像中所能看到的最远距离，上端点和下端点之间的距离表示每一条车道分界线的长度，下端点距离相邻车道分界线为车道分界线之间的距离。

在摄像机标定时取车道分界线为标定特征，几何视觉上分界线在同一条直线上，以几何角度对摄像机标定进行分析。当摄像机光轴在道路平面内的投影内与道路方向基本一致时，可以将摄像机近似为一维标定模型。如图 ，监控图像上的点 M、N、P、Q，映射到道路上各点分别为 A、B、C、D。AB 所在直线在道路上车道分界线位置，MN 所在直线为图像上车道分界线所在位置，OA、OB、OC、OD 为摄像机进行透射时的光线。在下图中，以 A 为端点做辅助线段 AG，满足 MQ//AG，分别交 OB , OC 于点 E、F；以 B 为端点做辅助线段 BK，满足 MQ//BK，交 OC 于点 J。

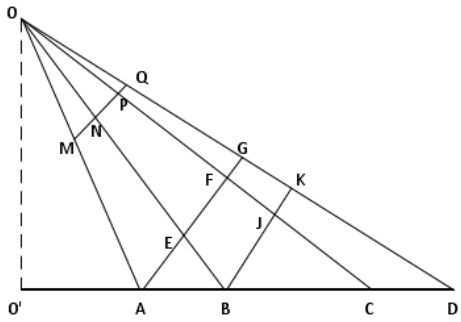

图 22 摄像机光轴与道路方向一致几何推导模型图

由 MQ//AG//BK，可得

$$\frac{MN}{AE}=\frac{MQ}{AG}，\text{即 } AG=\frac{AE\cdot MQ}{MN}$$

$$\frac{NP}{BJ}=\frac{NQ}{BK}，\text{即 } BK=\frac{NQ\cdot BJ}{NP}$$

由上面三个公式联合可得

$$\frac{BD}{AD}=\frac{MQ\cdot NP\cdot AE}{MN\cdot NQ\cdot BJ}$$

由上图可知，$\Delta AFC\sim\Delta BJC$，可得

$$\frac{AF}{BJ}=\frac{AC}{BC}，\text{即 } BJ=\frac{AF\cdot BC}{AC}$$

将此式代入上面一个公式，可得

$$\frac{BD}{AD}=\frac{MQ\cdot NP\cdot AE\cdot AC}{MN\cdot NQ\cdot AF\cdot BC}$$

由 MQ//AG 可得

$$\frac{AE}{AF}=\frac{MN}{MP}$$

再代入上面一个公式，可得

$$\frac{BD}{AD}=\frac{MQ\cdot NP\cdot AC}{MP\cdot NQ\cdot BC}，\text{即 } AD=\frac{AB\cdot MP\cdot NQ\cdot BC}{MP\cdot NQ\cdot BC-MQ\cdot NP\cdot AQ}$$

假设实际世界的距离 $AB=l_1,BC=l_2,AD=l$，图像上的像素点 $M(u_m,v_m)$、$N(u_n,v_n)$，$P(u_p,v_p)$、$Q(u_q,v_q)$，像素与物理尺寸每一个像素所占的物理迟钝分别为 dx 和 dy，那么可得图像上各个像素点之间的物理尺寸：

$$\begin{cases} MQ=(v_q-v_m)\cdot dy \\ MP=(v_p-v_m)\cdot dy \\ NP=(v_p-v_n)\cdot dy \\ NQ=(v_q-v_n)\cdot dy \end{cases}$$

将此式代入上面一个公式，可得

$$l=\frac{l_1 l_2(v_p-v_m)(v_q-v_n)}{l_2(v_p-v_m)(v_q-v_n)-(l_1+l_2)(v_q-v_m)(v_p-v_n)}$$

在这种状态下，沿着摄像机光轴投影的方向任一点$Q(u_q,v_q)$，可求出相应l。在这种情况下，上述公式符合函数模型：

$$y=\frac{ax}{b+cx}$$，其中$(a>0,b>0,c\neq 0)$。

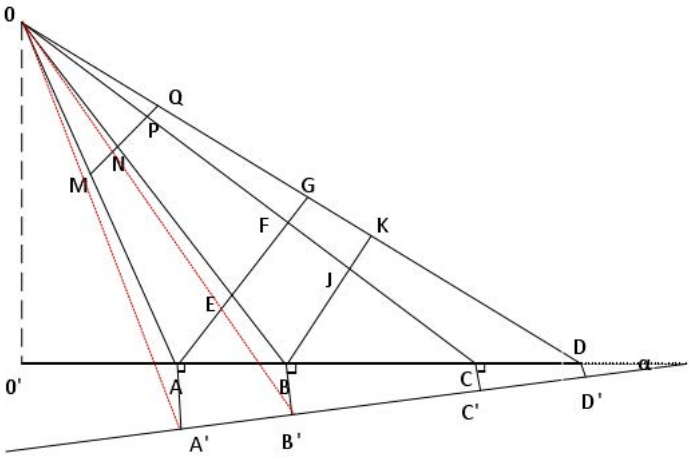

图 23 摄像机光轴与道路方向不一致几何推导模型图

当摄像机光轴在路面上的投影和道路方向存在夹角α时，如上图。假设$A'B'=l_1'$，$B'C'=l_2'$，则$l_1=l_1'\cos\alpha$，$l_2=l_2'\cos\alpha$,仍然符合$y=\frac{ax}{b+cx}$，其中$a>0,b>0,c\neq 0$,所以该函数可以作为标定曲线拟合模型。

《高速公路监控技术要求》中指出高速公路上摄像机采用立柱安装，安装高度在 8~12m 范围内，因此可利用摄像机坐标与世界坐标关系求解距离深度。

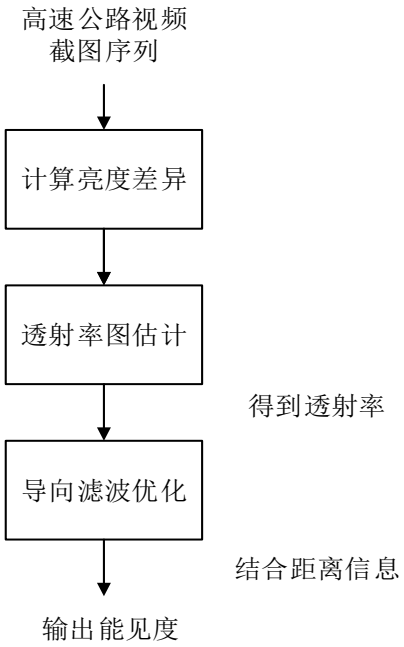

图 24 能见度求解算法流程图

上图展示了能见度求解算法流程，对于高速公路视频的截图序列，首先计算亮度差异，然后估计透射率，得到透射率，之后导向滤波优化，结合距离信息即可输出能见度。

5.3.6 模型的求解

题目中所给出的高速公路监控视频数据，每隔 30-40s 取一帧进行处理，形成测试视频序列。共 100 张，时间为 6:30:26 到 7:39:11，接近一个小时。接下来我们对比不同时刻的高速公路大雾情况，如下图所示。

（a）6：30 高速公路截图　　（b）7：38 高速公路截图

图 25 不同时刻下的高速公路截图

通过对比不同时刻下的高速公路截图可以发现，在 6:30 时刻，能见度较差，中间车道的第二条车道线依稀可见，右侧车道线模糊不清，而在 7:30 时刻，可以清楚的看到中间车道的第二条车道线，且右侧车道线也更明显。

对于两个图像的上半部分来说，均充满浓雾，且由于拍摄角度原因，不是近地面，不会出现景远可视距离，而图像的下半部分包括了路段和雾消散的边界，因此在求取透射率的过程中，我们更多的关注图像的下方，在对暗通道先验后的透射率进行引导滤波优化后，取图像下半方大小即 360×1280，将其取平均得到每副图像的透射率，之后利用几何模型估

算出道路深度，求得能见度结果见下表，完整结果见附录。

表 5 高速公路能见度随时间变化

时间	能见度（m）
6:30:26	30.19
6:31:08	30.31
6:31:50	29.97
6:32:31	30.12
...	...
7:37:48	65.72
7:38:30	67.64
7:39:11	59.17

根据《中华人民共和国气象行业标准》，为了满足高速公路检测场景的服务需求，对雾霾引起的低能见度进行了相应的等级划分，该标准将能见度范围划分为 5 个级别，具体分类如下表所示。当能见度大于 200 米时，即能见度等级为 0 级或 1 级时，高速管理部门无需对车流量和车辆行驶速度进行监管处理，在能见度小于 200 米时，监控人员需要滴交通路况进行管控甚至封路。

表 6 高速公路能见度等级划分与影响程度

等级	能见度 L 范围/m	影响程度
0 级	L>500	-
1 级	200<L<500	交通预警
2 级	100<L<500	预警或交通管制
3 级	50<L<100	交通管制
4 级	L<50	全线或者局部封闭

结合表 可以看出，在 6:30 的时候，能见度只有 30.19m，此时高速公路大雾无法行驶，在经过接近一个小时后，大雾慢慢消散，能见度达到 59.17m，此时路况较之前有所改善，但仍然需要进行交通管制。

绘制该时间段高速公路能见度随时间变化曲线见下图所示。

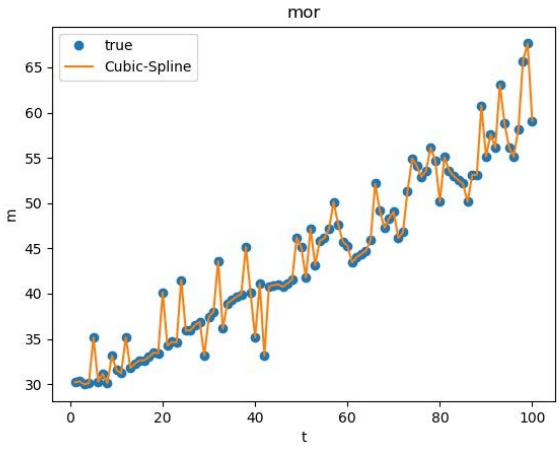

图 26 高速公路能见度随时间变化曲线

可以看出，在前 50 个时间点的能见度增速与后 50 个时间点能见度增速相比较慢，这说明在早上的 6:30-7:00 阶段，大雾消散的速度比 7:00-7:38 慢。观测数据日期为 4 月 14 日，为一年中的春季，随着阳光、温度上升、湿度等因素，大雾将渐渐消散。

5.4 问题四

5.4.1 问题四的分析

能见度是带有趋势和随机性的序列。时间序列方法是使用时间序列建立数学模型，在模型的基础上挖掘数据背后的规律，可以通过分析历史数据对未来数据做出预测。因此，使用之前得到的能见度随时间变化规律，我们可以建立时间序列模型中的 ARIMA 模型和深度学习模型分别预测大雾变化趋势以及散去的时间。

5.4.2 问题四模型的建立

（1）时间序列模型的建立

时间序列模型中我们选择使用经典且应用范围广泛的 ARIMA 模型。ARIMA 模型即差分整合移动平均自回归模型，是时间序列预测分析方法之一。ARIMA(p，d，q)中，AR 是“自回归”，p 为自回归项数；MA 为“滑动平均”，q 为滑动平均项数，d 为使之成为平稳序列所做的差分次数（阶数）。差分是 ARIMA 模型的关键步骤。

自回归（AR）模型是利用自身的历史数据作为回归变量的模型。若时间序列 x_t 是过去时间序列值 $x_{t-i}(i=1,2,...,k)$ 的线性组合：

$$x_t = c + \varphi_1 x_{t-1} + \varphi_2 x_{t-2} + ... + \varphi_p x_{t-p} + \varepsilon_t$$

即 $AR(p)$ 模型。其中，c 为常数，$\varphi_1,\varphi_2,...,\varphi_p$ 为自回归系数，也就是 x_{t-i} 对 x_t 影响程度的表现。ε_t 是误差或白噪声，需要满足如下性质：

$$\begin{cases} E(\varepsilon_t)=0 \\ E(\varepsilon_t,\varepsilon_s)=\begin{cases}\sigma_\varepsilon^2, t=s \\ 0, t\neq s\end{cases} \\ E(\varepsilon_t,\varepsilon_s)=0, \forall s<t \end{cases}$$

引入后移算子 B ，模型可以被简化为：

$$\varphi(B)x_t = c + \varepsilon_t$$

自回归模型的优点在于所需数据量较少，适用于受自身历史因素影响较大的情况。

移动平均（MA）模型是现在干扰值 ε_t 与过去干扰值 ε_{t-i} 的线性组合，$MA(q)$ 的模型为：

$$x_t = c + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + ... + \theta_q\varepsilon_{t-q}$$

其中，c 为常数，$\theta_1,\theta_2,...,\theta_q$ 为滑动平均参数，$\varepsilon_t,\varepsilon_{t-1},...,\varepsilon_{t-q}$ 是误差或白噪声序列，需要满足以下性质：

$$\begin{cases} E(\varepsilon_t)=0 \\ E(\varepsilon_t,\varepsilon_s)=\begin{cases}\sigma_\varepsilon^2, t=s \\ 0, t\neq s\end{cases} \end{cases}$$

引入后移算子 B，该模型可以被简化为：

$$x_t = u + \theta(B) + \varepsilon_t$$

移动平均模型的缺点在于不能很好地反映趋势，适合近期预测。

而 ARMA（p,q）模型是将两者结合，它的结构为

$$\begin{cases} x_t = \phi_0 + \phi_1 x_{t-1} + \ldots + \phi_p x_{t-p} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \ldots - \theta_q \varepsilon_{t-q} \\ \phi_p \neq 0, \theta_q \neq 0 \\ E(\varepsilon_t) = 0, Var(\varepsilon_t) = \sigma_\varepsilon^2, E(\varepsilon_t \varepsilon_s) = 0, s \neq t \\ Ex_s \varepsilon_t = 0, \forall s < t \end{cases}$$

特定当$\phi_0 \neq 0$时，称为中心化 ARMA（p,q）模型。

如果X_t不平稳，而

$$W_t = \nabla^d X_t = (1-B)^d X_t$$

是平稳的，那么整合的 ARMA 模型 ARIMA（p,d,q）为用平稳过程W_t来替代不平稳的X_t在 ARMA 模型中的位置，即

$$\phi(B) W_t = \theta(B) w_t.$$

ARIMA 模型的建模步骤如下：

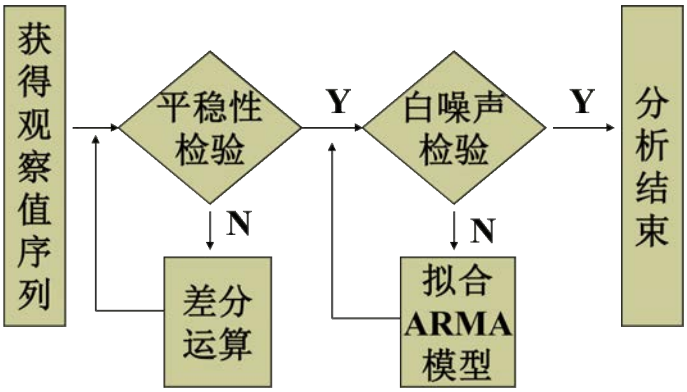

图 27　ARIMA 建模步骤

ARIMA 模型的结构为

$$\begin{cases} \phi(B) \nabla^d x_t = \Theta(B) \varepsilon_t \\ E(\varepsilon_t) = 0, Var(\varepsilon_t) = \sigma_\varepsilon^2, E(\varepsilon_t \varepsilon_s) = 0, s \neq t \\ Ex_s \varepsilon_t = 0, \forall s < t \end{cases}$$

当 d=0 时，ARIMA（p,0,q）模型与 ARMA（p,q）模型等同。

ARIMA 模型即先使时间序列平稳，再构造 ARMA 模型。ARIMA 模型是比 ARMA 模型更通用，多用于处理非平稳时间序列，一般都能取得较好的预测结果。

对不平稳序列进行差分后，得到的平稳序列进行白噪声检验可以利用检验统计量 Q 和 Ljung-Box。它们的计算公式如下：

$$Q = n \sum_{k=1}^{m} \hat{\rho}_k^2 \sim \chi^2(m)$$

$$LB = n(n+2) \sum_{k=1}^{m} \left(\frac{\hat{\rho}_k^2}{n-k} \right) \sim \chi^2(m)$$

（2）深度学习的 Seq2seq 模型

Seq2Seq 是一个具有 Encoder-Decoder 结构的神经网络，它的输入是一个序列，输出也是一个序列，因此得名 Seq2Seq。在 Encoder 中，将可变长度的序列转化成固定长度的向量，Decoder 将这个固定长度的向量转换为可变长度目标信号序列。

Seq2Seq 模型工作的流程图如下：

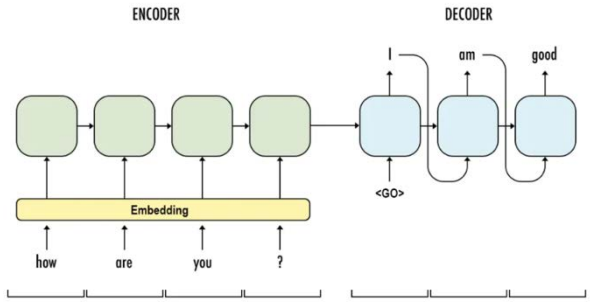

图 28 seq2seq 模型工作流程图

最基础的 Seq2Seq 模型包含了三个部分，即 Encoder、Decoder 以及连接两者的中间状态向量 C，Encoder 通过学习输入，将其编码成一个固定大小的状态向量 C（语义编码），继而将 C 传给 Decoder，Decoder 再通过对状态向量 C 的学习来进行输出对应的序列。

如果单一使用 seq2seq 模型，效果会大打折扣。注意力模型就是基于 Encoder-Decoder 框架下的一种模拟人类注意力直觉的一种模型。

人脑的注意力机制本质上是一种注意力分配的模型，对于一篇论文、一张图片，在任意一时刻我们的注意力分布是不一样的。这便是著名的注意力机制模型的由来。

Encoder-Decoder 作为一种通用框架，在具体的自然语言处理任务上还不够精细化。换句话说，单纯的 Encoder-Decoder 框架并不能有效的聚焦到输入目标上，这使得像 seq2seq 的模型在独自使用时并不能发挥最大功效。而注意力模型要做的事就是根据序列的每个时间步将编码器编码为不同 C，在解码时，结合每个不同的 C 进行解码输出，这样得到的结果会更加准确，如下所示：

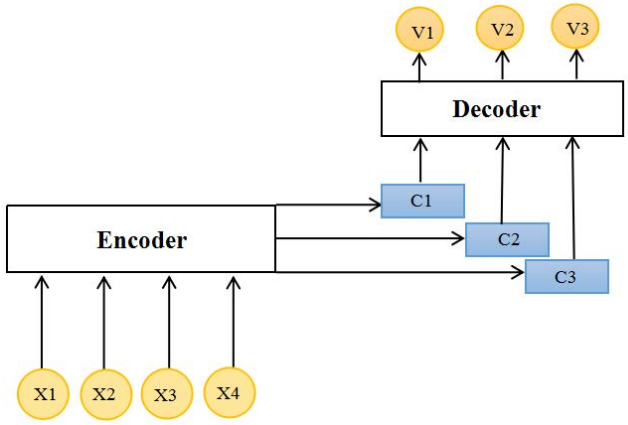

图 28 seq2seq with attention 结构

简单的注意力模型通常由三个公式来描述：

①计算注意力得分；

②进行标准化处；

③结合注意力得分和隐状态值计算上下文状态 C。

具体公式包括：

注意力权重 α_{ts}：

$$\alpha_{ts}=\frac{\exp(score(h_t,\overline{h_s}))}{\sum_{s'=1}^{S}\exp(score(h_t,\overline{h_{s'}}))}$$

注意力输出 c_t：

$$c_t=\sum_{s}\alpha_{t_s}\overline{h}_s$$

最终输出 y_2：

$$y_2=f(c_t,h_t)=\tanh(W_c[c_t;h_t])$$

5.4.3 问题四模型的求解

5.4.3.1 ARIMA 模型的分析和预测

（1）原始数据的分解

由问题 3 得到的高速公路能见度随时间变化曲线图可以看出，能见度数据呈现出波动上升的趋势。为使得预测结果更为准确，先将原始能见度数据进行分解，分解成趋势、周期性部分及残差。分解后的各部分序列如下图所示，分别代表着：趋势、周期性、残差。

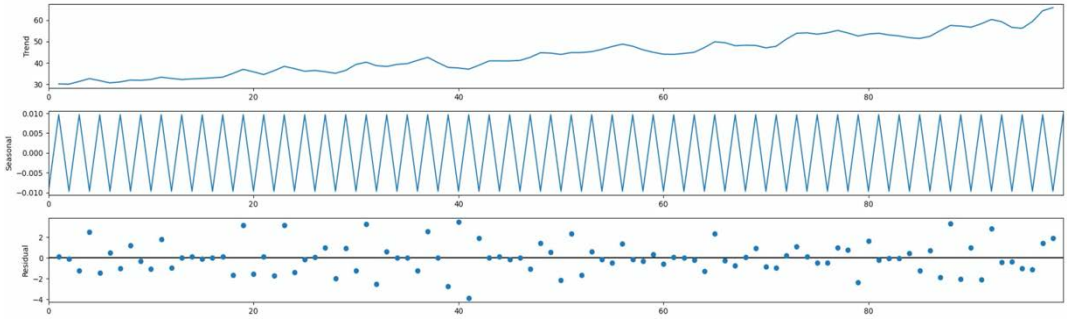

图 29 分解后的时间序列

将时间序列分解之后，可以只对趋势项进行 ARIMA 模型的建模，预测出结果后将周期性部分和预测项相加，从而得到最终的预测数据。

（2）ARIMA 模型的建立

1. 平稳性及白噪声检测

在建立 ARIMA 模型之前，先对趋势数据进行平稳性和白噪声检测。首先对原始趋势序列进行 ADF 单位根校验，原始趋势序列并非平稳序列。 然后对原始序列进行一阶差分，经过一阶差分后的趋势序列通过 ADF 校验，证明其为平稳序列。经过一阶差分的趋势序列图和其 ACF、PACF 图如下图所示。

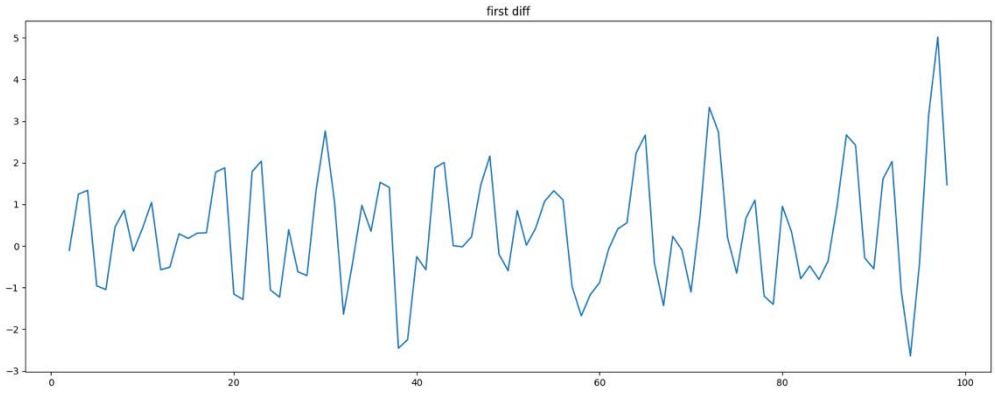

图 29 经过一阶差分后的趋势序列图

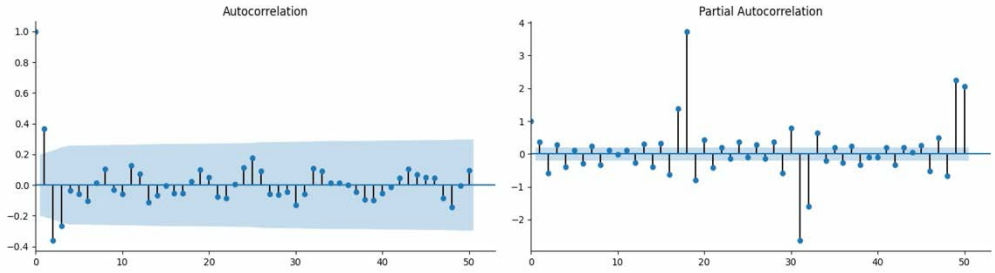

图 31 一阶差分序列的 ACF 图和 PACF 图

构建 ARMA(p,q)的条件是 PACF 图的 p 阶后和 ACF 的 q 阶后均落在置信区间内，由图 4-5 可知，很难选择合适的 p、q 值，因而对一阶差分序列进行再差分，得到经过二阶差分的趋势序列和其 ACF、PACF 图，如下图所示。

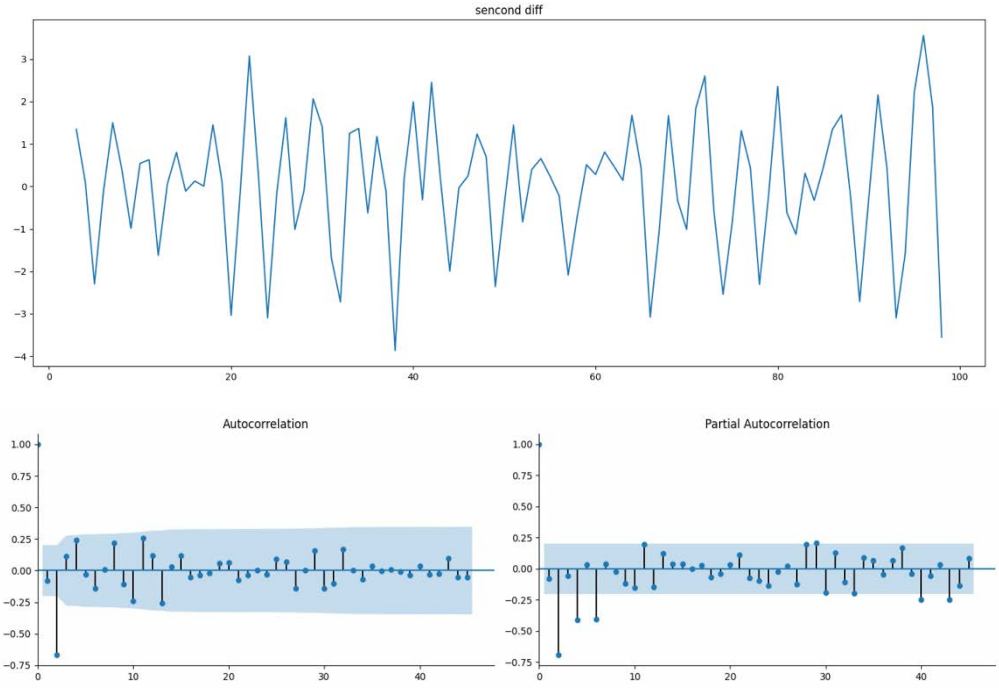

图 32 二阶差分序列的 ACF 图和 PACF 图

由上图可以判断，二阶差分序列的 ACF 图和 PACF 图在某个 p、q 值之后可以满足需求，同时通过构建 LB 统计量对经过二阶差分的趋势序列进行白噪声检测，结果显示原始

趋势序列与经过二阶差分后的趋势序列均通过显著性检验，它们均为非白噪声序列。

2.模型识别

由上图可以判断二阶差分序列的 ACF 图和 PACF 图均显示拖尾，因而确定 d=2。但由于 p 和 q 的值无法用肉眼准确判断，因而实验中使用 AIC 和 BIC 准进行估计。通过穷举法，拟合不同 p、q 值之下的 ARIMA 模型，并获取其 AIC 和 BIC 信息量。部分模型的 AIC 和 BIC 如下表所示

表 7 ARIMA 模型的 AIC 信息量

Lags	MA1	MA2	MA3	MA4	MA5	MA6
AR1	321.075	NaN	NaN	NaN	NaN	NaN
AR2	282.366	NaN	NaN	NaN	NaN	233.146
AR3	281.225	NaN	NaN	NaN	NaN	NaN
AR4	268.279	246.103	NaN	NaN	NaN	NaN
AR5	270.028	NaN	NaN	NaN	NaN	NaN
AR6	262.208	243.178	245.128	NaN	NaN	NaN

表 8 ARIMA 模型的 BIC 信息量

Lags	MA1	MA2	MA3	MA4	MA5	MA6
AR1	331.33	NaN	NaN	NaN	NaN	NaN
AR2	295.18	NaN	NaN	NaN	NaN	258.78
AR3	296.61	NaN	NaN	NaN	NaN	NaN
AR4	286.22	266.61	NaN	NaN	NaN	NaN
AR5	290.54	NaN	NaN	NaN	NaN	NaN
AR6	285.28	268.82	273.33	NaN	NaN	NaN

p=2、q=6 时，AIC 和 BIC 的值均为最小，因而最终选择建立 ARIMA（2，2，6）模型。

3.参数估计与模型检验

在确定 ARIMA 模型的 p,q 值之后，要确定 ARIMA 模型的系数。ARIMA（2，2，6）模型的校验结果如下图所示。

ARIMA Model Results

Dep. Variable:	D2.y	No. Observations:	96
Model:	ARIMA(2, 2, 6)	Log Likelihood	-106.573
Method:	css-mle	S.D. of innovations	0.643
Date:	Mon, 21 Sep 2020	AIC	233.146
Time:	01:56:52	BIC	258.789
Sample:	2	HQIC	243.511

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0020	0.001	2.954	0.003	0.001	0.003
ar.L1.D2.y	-0.4914	0.076	-6.435	0.000	-0.641	-0.342
ar.L2.D2.y	-0.8383	0.063	-13.317	0.000	-0.962	-0.715
ma.L1.D2.y	0.6191	0.096	6.468	0.000	0.432	0.807
ma.L2.D2.y	-0.9913	nan	nan	nan	nan	nan
ma.L3.D2.y	-1.2371	0.076	-16.258	0.000	-1.386	-1.088
ma.L4.D2.y	-0.9922	nan	nan	nan	nan	nan
ma.L5.D2.y	0.6183	0.084	7.326	0.000	0.453	0.784
ma.L6.D2.y	0.9994	0.043	23.282	0.000	0.915	1.084

图 33 ARIMA（2，2，6）模型试验结果图

由图中可知，各参数的 p 值均接近 0，模型的参数通过显著性检验。ARIMA（2，2，6）模型的具体形式为：

$$x_t = -0.49\varphi_1 x_{t-1} - 0.84\varphi_2 x_{t-2} + \varepsilon_t + 0.62\theta_1\varepsilon_{t-1} - 0.99\theta_2\varepsilon_{t-2} - 1.24\theta_3\varepsilon_{t-3} - 0.99\theta_4\varepsilon_{t-4} + 0.62\theta_5\varepsilon_{t-5} + 1.00\theta_6\varepsilon_{t-6} + 0.002$$

最后对模型进行残差检验，由于此模型在模型识别步骤已经根据 AIC 和 BIC 准则进行定阶，因而不再需要进行模型优化，只需进行残差分析即可。通过构建 LB 统计量，结果表明残差序列为白噪声序列。

（3）序列步长的确定

观察题目所给的 100 帧图像截图，发现第一张图片的拍摄时间为 06:30:26 记为t_1，第 100 张图片的拍摄时间为 07:39:11 记为t_{100}，序列步长α定义为：

$$\alpha = \frac{t_{100} - t_1}{100}$$

进而计算出序列步长$\alpha = 41.25s$。

（4）ARIMA 模型的预测

首先使用拟合好的模型预测出未来的趋势序列，再将周期性部分与预测的趋势序列相加，得到最后的能见度预测结果如下图所示。

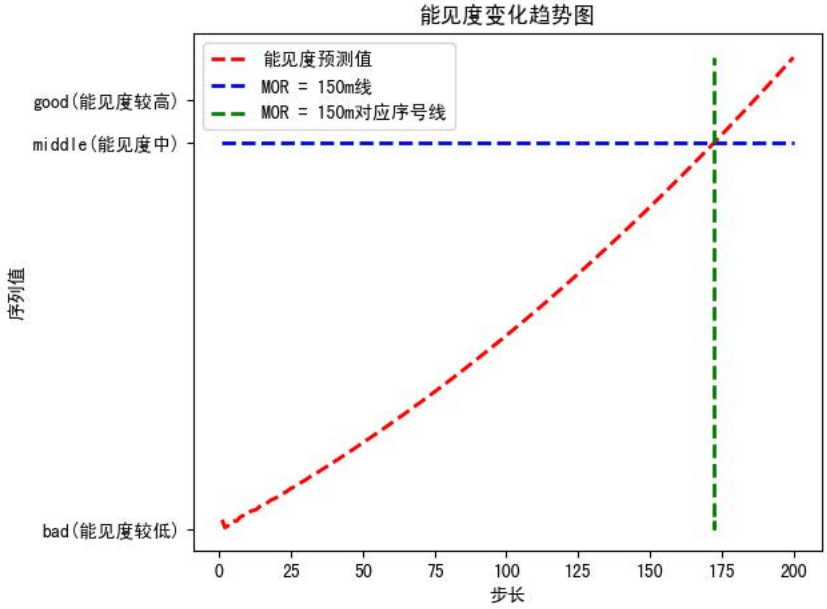

图 30 能见度变化趋势预测图

上图中红线部分即为预测出的能见度变化趋势，可以从中很明显的看出随着时间的推移，能见度越来越高、大雾逐渐减弱，并且减弱的速度呈现出加快的趋势。

上图中红线、蓝线、绿线的交点即为 MOR 到达 150m 的点，该点对应的步长为 172.4，每步长时间 $\alpha = 41.25s$,可以算出经过 7111.5s MOR 的值到达 150m，由于第 100 帧图片拍摄的时间为 07:39:11，从而可以得到在这一天 09:37:43 的时刻雾会散去。

5.4.3.2 Seq2seq+Attention 模型的分析和预测

通过观察 ARIMA 模型对能见度变化趋势预测的结果，发现 ARIMA 模型可以较好的捕捉到能见度趋势的线性变化，但是 ARIMA 模型对能见度趋势的非线性变化不是很敏感。为了进一步的提高预测的精度，引入 Seq2seq+Attention 模型来捕捉能见度趋势的非线性变化。

（1）数据增强

从问题 3 中可以获取到的能见度序列数据只有 100 个，这对于训练深度学习模型是远远不够的，因此需要在问题 3 能见度序列数据的基础上进行数据增强。

通过对问题 3 中 100 帧图片时间的观察，发现帧与帧之间的时间差异集中在 30～60s 之间，为了模拟真实多变的环境，随机采用线性插值法或拉格朗日插值法，在帧与帧之间每隔 3～5s 插一个值，进而完成数据增强。

（2） Seq2seq+Attention 模型的训练

1、Seq2seq 模型的训练参数设定

将能见度数据作为单变量输入到时序模型中，encoder 序列长度设置为 10，decoder 序列长度设置为 2，即用以往 10 个能见度数据来预测下面 2 个能见度数据，同时将预测得到的能见度数据作为下一次预测的输入，模型思想如下图所示。

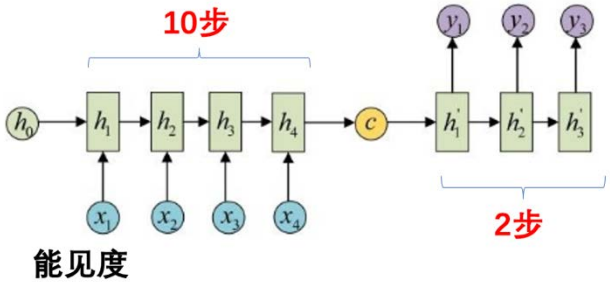

图 31 Seq2seq 模型思想

2、Attention 机制的逻辑设定

将O_t定义为 encoder 的输出，尺寸为$t\times d_h$，h_t定义为 decoder 某一步的隐状态，尺寸为$1\times d_h$,函数$f(O,h)$实现两者的结合，具体关系式如下：

$$f(O,h)=\begin{cases}Oh^T, dot\\ OWh^t, general\end{cases}$$

通过$soft\max$层对$f(O,h)$进行处理进而得到 encoder 的权重矩阵，将权重矩阵与 encoder 输出点乘进而得到加权后的 encoder 输出

$$A=soft\max(f(O,h))$$

$$c_t=A^TO$$

上式中的c_t为加权后的 encoder 输出，尺寸为$1\times d_h$，将加权后的 encoder 输出与 decoder 的隐状态相结合得到注意力机制下网络的最终输出y_t为：

$$y_t=linear(c_t;h_t)$$

加入注意力机制后的模型思想如下图所示。

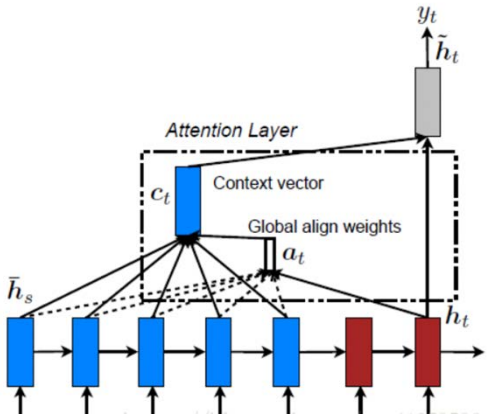

图 32 注意力机制下的模型思想

3.模型的训练

07:39:11 时刻以前的能见度时序数据 70%作为训练集，20%作为验证集，10%作为测试集，用 RMSE 指标来展现模型在验证集和测试集上的效果，经过 500 轮训练迭代后，验证集 RMSE 为 6.16，测试集最小 RMSE 为 10.32，500 轮迭代的损失曲线如下左图所示。同时在模型不断优化迭代的过程中，得出 Attention 机制下 encoder 权重的热力图如下右图所示，从图中可以看出 encoder 后几个时间步具有更大的权重。

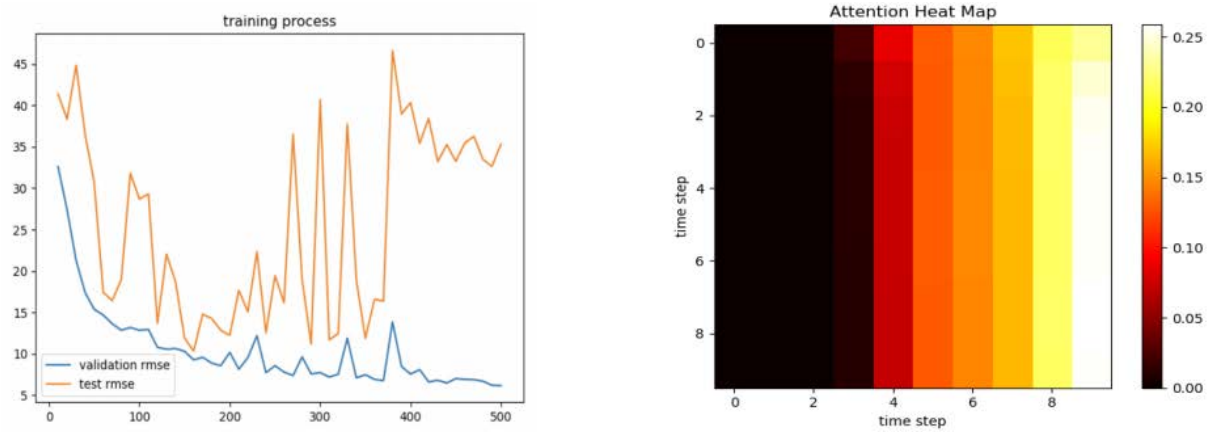

图 33 500 轮迭代损失曲线图和注意力机制下 encoder 权重热力图

（3）Seq2seq+Attention 模型的预测

使用训练好的模型对能见度数据序列进行预测，预测得到的能见度数据会作为下一次预测的输入，最终的能见度预测结果如下图所示。

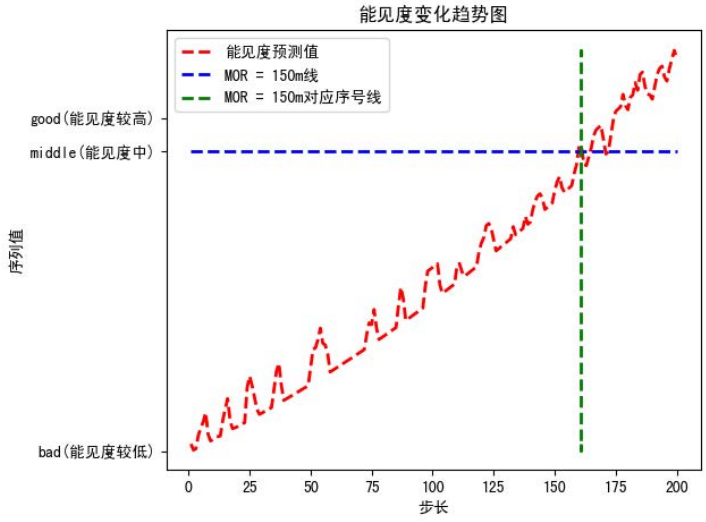

图 34 能见度变化趋势预测图

上图中红线部分即为预测出的能见度变化趋势，可以从中较为明显的看出随着时间的推移，能见度呈现波动上升趋势，大雾在波动中逐渐减弱。

上图中红线、蓝线、绿线的交点即为 MOR 到达 150m 的点，该点对应的步长为 160.9，每步长时间 $\alpha = 41.25s$，可以算出经过 6637.1s MOR 的值到达 150m，由于第 100 帧图片拍摄的时间为 07:39:11，从而可以得到在这一天 09:29:48 的时刻雾会散去。

5.4.3.3 ARIMA 模型和 Seq2seq+Attention 模型融合下的结果分析

ARIMA 模型擅长捕捉能见度数据趋势的线性变化，Seq2seq+Attention 模型擅长捕捉能见度数据趋势的非线性变化，将两个模型的结果进行融合可以得到更好的预测结果：经过 6637.1s～7111.5s 后 MOR 可以达到 150m，对应到当天的时刻为 09:29:48～09:37:43。模型融合后的结果分析如下表所示。

表 9 模型融合后结果分析

模型名称	ARIMA	Seq2seq+Attention
擅长捕捉趋势	线性变化	非线性变化
多久后 MOR 达到 150m	7111.5s	6637.1s
当天何时 MOR 达到 150m	09:37:43	09:29:48
模型融合后 MOR 多久达到 150m	6637.1s~7111.5s	
模型融合后 MOR 何时达到 150m	09:29:48～09:37:43	

模型融合后 6637.1s~7111.5s，MOR 达到 150 米，时间为 09:29:48～09:37:43。

六 模型的评价和推广

6.1 模型的评价

（1）模型的优点

文中联合时空特征，结合题目场景，提出了 ResNet + LSTM 融合的深度学习模型来对视频数据的能见度进行估计，取得了良好的估计效果

文中采用多项式特征生成的方法，将非线性关系变换到了线性关系，准确的回归出了能见度与风速、气压等指标的关系

文中采用模型融合进行预测，充分利用 ARIMA 模型和 Seq2seq+Attention 模型的优点，具备着良好的预测精度

文中提出的能见度求解模型，算法简单，实时性较好，对环境变化有一定的抑制效果，而且不需要考虑背景更新

（2）模型的缺点

文中 Seq2seq 模型的 RMSE 值不是很稳定，其中 encoder 参数和 decoder 参数的值有进一步优化的空间

6.2 模型的推广和改进

（1）模型的推广

模型可以在实时性较强，检测精度要求普通的场景中予以推广拓展

（2）模型的改进

高速公路能见度检测算法对图像噪声比较敏感，即摄像机在采集图像过程中会受到天气环境影响。比如：恶劣天气尤其是风力较大天气，将会导致摄像机采集的图像出现相机抖动现象。如果昆虫或者雨滴等附在镜头前同样影响本文算法的检测结果。故如何提高本文算法的抗噪声性能是后期改进的关注点之一。除此之外，本文算法只适用于道路中无车辆的路况，对于车道线被遮挡或车道线污损情形也不适用。

继续探索 Attention 机制，借助寻优算法，得出使得目标问题精度更高的 encoder 权重值

参考文献

[1]He, K. , Zhang, X. , Ren, S. , & Sun, J. . (2016). Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision & Pattern Recognition. IEEE Computer Society.

[2]苗开超,韩婷婷,王传辉,章军,姚叶青,周建平.基于 LSTM 网络的大雾临近预报模型以及应用[J].计算机系统应用,2019,28(05):215-219.

[3]唐绍恩,李骞,胡磊,马强,顾大权.一种基于迁移学习的能见度检测方法[J].计算机工程,2019,45(09):242-247.

[4]许倩. 基于监控图像的高速公路能见度估计研究[D].长安大学,2016.

[5]周凯. 基于道路监控视频的雾霾能见度检测方法研究[D].南京邮电大学,2017.

[6]董建宁. 雾霾环境下视频图像透雾系统研究[D].长春理工大学,2019.

[7]张炎. 基于数字图像的日间大气能见度估测研究[D].南京信息工程大学,2019.

[8]崔健. 江苏省能见度时空分布特征及其影响因子分析[D].南京信息工程大学,2015.

[9]周奕珂,朱彬,韩志伟,潘晨,郭婷,魏建苏,刘端阳.长江三角洲地区冬季能见度特征及影响因子分析[J].中国环境科学,2016,36(03):660-669.

[10]高嵩.深度学习在机场能见度预测中的应用[J].计算机产品与流通,2020(04):260.

[11]Xiaohong Zhang,Liqing Jiang,Dongxu Yang,Jinyan Yan,Xinhong Lu. Correction to: Urine Sediment Recognition Method Based on Multi-View Deep Residual Learning in Microscopic Image[J]. Journal of Medical Systems,2020,44(4).

[12]Xuelei Li,Liangkui Ding,Li Wang,Fang Cao. FPGA Accelerates Deep Residual Learning for Image Recognition[A]. IEEE Beijing Section、Global Union Academy of Science and Technology、Chongqing Global Union Academy of Science and Technology.Proceedings of 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC 2017)[C].IEEE Beijing Section、Global Union Academy of Science and Technology、Chongqing Global Union Academy of Science and Technology:IEEE BEIJING SECTION,2017:4.

[13]Yan Yang,Chen Zhang,Longlong Liu,Gaoke Chen,Hui Yue. Visibility restoration of single image captured in dust and haze weather conditions[J]. Multidimensional Systems and Signal Processing,2020,31(prepublish).

[14]Liying Lu. Design and Implementation of Fog Visibility Detection System Based on Image Analysis[D].Southeast University,2015.

[15]Suresh Chandra Raikwar,Shashikala Tapaswi. An improved linear depth model for single image fog removal[J]. Multimedia Tools and Applications,2018,77(15).

[16]Nan Dong,Zhen Jia,Jie Shao,Zhipeng Li,Fuqiang Liu,Jianwei Zhao,Pei-Yuan Peng. Adaptive Object Detection and Visibility Improvement in Foggy Image[J]. Journal of Multimedia,2011,6(1).

附录

代码：
绘图

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="darkgrid")


df = pd.read_csv('/Users/houguiyang/Desktop/2020 国赛/问题 4-1.csv')

plt.rcParams['font.sans-serif']=['SimHei']

font1 = {
'size':23,
}

plt.figure(figsize=(15,5),dpi = 72)

# Add title
plt.title("能见度变化趋势图",font1)

# Line chart showing daily global streams of 'Shape of You'
sns.lineplot(data=df['序列值'], label="能见度",color = 'r')

plt.xticks(fontsize=20)
plt.yticks(fontsize=20)


# Add label for horizontal axis
plt.xlabel("序号",font1)
plt.ylabel("序列值",font1)
plt.show()
```

代码：
问题二

```
import torch.nn as nn
import torch.utils.model_zoo as model_zoo


__all__ = ['ResNet', 'resnet18', 'resnet34', 'resnet50', 'resnet101',
           'resnet152']
```

```
model_urls = {
    'resnet18': 'https://download.pytorch.org/models/resnet18-5c106cde.pth',
    'resnet34': 'https://download.pytorch.org/models/resnet34-333f7ec4.pth',
    'resnet50': 'https://download.pytorch.org/models/resnet50-19c8e357.pth',
    'resnet101': 'https://download.pytorch.org/models/resnet101-5d3b4d8f.pth',
    'resnet152': 'https://download.pytorch.org/models/resnet152-b121ed2d.pth',
}


def conv3x3(in_planes, out_planes, stride=1):
    """3x3 convolution with padding"""
    return nn.Conv2d(in_planes, out_planes, kernel_size=3, stride=stride,
                     padding=1, bias=False)


def conv1x1(in_planes, out_planes, stride=1):
    """1x1 convolution"""
    return nn.Conv2d(in_planes, out_planes, kernel_size=1, stride=stride, bias=False)


class BasicBlock(nn.Module):
    expansion = 1

    def __init__(self, inplanes, planes, stride=1, downsample=None):
        super(BasicBlock, self).__init__()
        self.conv1 = conv3x3(inplanes, planes, stride)
        self.bn1 = nn.BatchNorm2d(planes)
        self.relu = nn.ReLU(inplace=True)
        self.conv2 = conv3x3(planes, planes)
        self.bn2 = nn.BatchNorm2d(planes)
        self.downsample = downsample
        self.stride = stride

    def forward(self, x):
        identity = x

        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)

        out = self.conv2(out)
        out = self.bn2(out)
```

```
        if self.downsample is not None:
            identity = self.downsample(x)

        out += identity
        out = self.relu(out)

        return out


class Bottleneck(nn.Module):
    expansion = 4

    def __init__(self, inplanes, planes, stride=1, downsample=None):
        super(Bottleneck, self).__init__()
        self.conv1 = conv1x1(inplanes, planes)
        self.bn1 = nn.BatchNorm2d(planes)
        self.conv2 = conv3x3(planes, planes, stride)
        self.bn2 = nn.BatchNorm2d(planes)
        self.conv3 = conv1x1(planes, planes * self.expansion)
        self.bn3 = nn.BatchNorm2d(planes * self.expansion)
        self.relu = nn.ReLU(inplace=True)
        self.downsample = downsample
        self.stride = stride

    def forward(self, x):
        identity = x

        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)

        out = self.conv2(out)
        out = self.bn2(out)
        out = self.relu(out)

        out = self.conv3(out)
        out = self.bn3(out)

        if self.downsample is not None:
            identity = self.downsample(x)

        out += identity
        out = self.relu(out)
```

```
        return out


class ResNet(nn.Module):

    def __init__(self, block, layers, num_classes=1000, zero_init_residual=False):
        super(ResNet, self).__init__()
        self.inplanes = 64
        self.conv1 = nn.Conv2d(3, 64, kernel_size=7, stride=2, padding=3,
                               bias=False)
        self.bn1 = nn.BatchNorm2d(64)
        self.relu = nn.ReLU(inplace=True)
        self.maxpool = nn.MaxPool2d(kernel_size=3, stride=2, padding=1)
        self.layer1 = self._make_layer(block, 64, layers[0])
        self.layer2 = self._make_layer(block, 128, layers[1], stride=2)
        self.layer3 = self._make_layer(block, 256, layers[2], stride=2)
        self.layer4 = self._make_layer(block, 512, layers[3], stride=2)
        self.avgpool = nn.AdaptiveAvgPool2d((1, 1))
        self.fc = nn.Linear(512 * block.expansion, num_classes)

        for m in self.modules():
            if isinstance(m, nn.Conv2d):
                nn.init.kaiming_normal_(m.weight, mode='fan_out', nonlinearity='relu')
            elif isinstance(m, nn.BatchNorm2d):
                nn.init.constant_(m.weight, 1)
                nn.init.constant_(m.bias, 0)

        # Zero-initialize the last BN in each residual branch,
        # so that the residual branch starts with zeros, and each residual block behaves
like an identity.
        # This improves the model by 0.2~0.3% according to https://arxiv.org/abs/1706.02677
        if zero_init_residual:
            for m in self.modules():
                if isinstance(m, Bottleneck):
                    nn.init.constant_(m.bn3.weight, 0)
                elif isinstance(m, BasicBlock):
                    nn.init.constant_(m.bn2.weight, 0)

    def _make_layer(self, block, planes, blocks, stride=1):
        downsample = None
        if stride != 1 or self.inplanes != planes * block.expansion:
            downsample = nn.Sequential(
                conv1x1(self.inplanes, planes * block.expansion, stride),
```

```
                nn.BatchNorm2d(planes * block.expansion),
            )

        layers = []
        layers.append(block(self.inplanes, planes, stride, downsample))
        self.inplanes = planes * block.expansion
        for _ in range(1, blocks):
            layers.append(block(self.inplanes, planes))

        return nn.Sequential(*layers)

    def forward(self, x):
        x = self.conv1(x)
        x = self.bn1(x)
        x = self.relu(x)
        x = self.maxpool(x)

        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer3(x)
        x = self.layer4(x)

        x = self.avgpool(x)
        x = x.view(x.size(0), -1)
        x = self.fc(x)

        return x


def resnet18(pretrained=False, **kwargs):
    """Constructs a ResNet-18 model.
    Args:
        pretrained (bool): If True, returns a model pre-trained on ImageNet
    """
    model = ResNet(BasicBlock, [2, 2, 2, 2], **kwargs)
    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['resnet18']))
    return model


def resnet34(pretrained=False, **kwargs):
    """Constructs a ResNet-34 model.
    Args:
        pretrained (bool): If True, returns a model pre-trained on ImageNet
```

```
    """
    model = ResNet(BasicBlock, [3, 4, 6, 3], **kwargs)
    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['resnet34']))
    return model


def resnet50(pretrained=False, **kwargs):
    """Constructs a ResNet-50 model.
    Args:
        pretrained (bool): If True, returns a model pre-trained on ImageNet
    """
    model = ResNet(Bottleneck, [3, 4, 6, 3], **kwargs)
    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['resnet50']))
    return model


def resnet101(pretrained=False, **kwargs):
    """Constructs a ResNet-101 model.
    Args:
        pretrained (bool): If True, returns a model pre-trained on ImageNet
    """
    model = ResNet(Bottleneck, [3, 4, 23, 3], **kwargs)
    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['resnet101']))
    return model


def resnet152(pretrained=False, **kwargs):
    """Constructs a ResNet-152 model.
    Args:
        pretrained (bool): If True, returns a model pre-trained on ImageNet
    """
    model = ResNet(Bottleneck, [3, 8, 36, 3], **kwargs)
    if pretrained:
        model.load_state_dict(model_zoo.load_url(model_urls['resnet152']))
    return model
```

代码：

问题四

```
import pandas as pd #操作数据
import statsmodels.api as sm #统计分析库
import matplotlib.pyplot as plt #可视化
```

```
import matplotlib as mpl
import seaborn as sns
import numpy as np
import itertools #迭代工具
import warnings #设置警告
import sklearn
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf #自相关函数、偏自相关函数
from math import sqrt
from sklearn.metrics import mean_squared_error
from statsmodels.tsa.arima_model import ARIMA
import warnings
from pandas import read_csv
import datetime
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
import statsmodels.api as sm
import csv

#将选取某个小时的
y=pd.read_csv('/Users/houguiyang/Desktop/序列数据预测.csv',index_col=0)
#y=y.drop(['predictOrders'],axis=1)
#y=y[(True^y['hour'].isin(['8:00','9:00','10:00','11:00','12:00','13:00','14:00','15:0
0','16:00','17:00','18:00','19:00','20:00','21:00']))]
#y=y[y['hour'].isin(['12:00'])]
y.index=pd.to_datetime(y.index,format="%Y-%m-%d")
print(y)

y_train=y['2018-10-1':'2019-1-8']
y_test=y['2019-1-9':'2019-1-30']

#print(y_train.trueOrders)

#分解数据
decomposition = sm.tsa.seasonal_decompose(y_train.values,period=2)
fig = plt.figure()
fig = decomposition.plot()
fig.set_size_inches(20, 8)
fig.show
#将分解出的趋势、季节特征以及残差取出
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

#将趋势数据转换成数据框格式
```

```
#trend=trend.dropna()
##处理数据
#trend = pd.dropna(trend)
trend= pd.DataFrame(trend)
trend = trend.dropna()

print(trend)

#对原始趋势数据进行一阶差分 通过差分来看数据变化趋势
trend_diff1=trend.diff()
trend_diff1=trend_diff1.dropna()
plt.figure(figsize=(15,6))
plt.plot(trend_diff1)
plt.title("first diff")
plt.show

#对原始趋势数据进行二阶差分
trend_diff2=trend_diff1.diff()
trend_diff2=trend_diff2.dropna()
plt.figure(figsize=(15,6))
plt.plot(trend_diff2)
plt.title("sencond diff")
plt.show


#对原始数据趋势进行三阶差分
trend_diff3=trend_diff2.diff()
trend_diff3=trend_diff3.dropna()
plt.figure(figsize=(15,6))
plt.plot(trend_diff3)
plt.title("third diff")

#对原始数据趋势进行四阶差分
trend_diff4=trend_diff3.diff()
trend_diff4=trend_diff3.dropna()
plt.figure(figsize=(15,6))
plt.plot(trend_diff4)
plt.title("four diff")

#对原始数据趋势进行五阶差分
trend_diff5=trend_diff4.diff()
trend_diff5=trend_diff4.dropna()
plt.figure(figsize=(15,6))
```

```
plt.plot(trend_diff5)
plt.title("four diff")
```

```
# 将原始图、直方图、acf、pacf、汇总到一个大图上
def tsplot(y, y_diff2, lags=None, title='', figsize=(15, 8)):
    fig = plt.figure(figsize=figsize)
    layout = (2, 2)  # 布局是 2 行 2 列
    ts_ax = plt.subplot2grid(layout, (0, 0))  # subplot2grid 分区函数，（0，0）表起始坐标
    hist_ax = plt.subplot2grid(layout, (0, 1))  # ax:要在上面进行绘制的 matplotlib subplot
对象。
    acf_ax = plt.subplot2grid(layout, (1, 0))
    pacf_ax = plt.subplot2grid(layout, (1, 1))

    y.plot(ax=ts_ax)
    ts_ax.set_title(title)

    y.plot(ax=hist_ax, kind='hist', bins=25)  # hist 表示画柱状图，bins 表示 25 个网格
    hist_ax.set_title('Histogram')

    plot_acf(y_diff2, lags=lags, ax=acf_ax)
    plot_pacf(y_diff2, lags=lags, ax=pacf_ax)

    [ax.set_xlim(0) for ax in [acf_ax, pacf_ax]]  # 设置 acf 和 pacf 的横坐标从 0 开始
    sns.despine()
    plt.tight_layout()
    plt.show()
    return ts_ax, acf_ax, pacf_ax

tsplot(trend,trend_diff3,title="Consumer Sentiment",lags=45)#函数调用

#平稳性检测
def stationarity_test(dataset,number):
    data = dataset.copy()
    data = data.iloc[: len(data)-number] #不检测最后 number 个数据
    from statsmodels.tsa.stattools import adfuller as ADF
    diff = 0
    #adf = ADF[data.setdefault('d',0)]
    adf = ADF(data[0])
    while adf[1] > 0.05:
        diff = diff + 1
        adf = ADF(data[0].diff(diff).dropna())
```

```
    print(u'原始序列经过%s 阶差分后归于平稳，p 值为%s' %(diff, adf[1]))

stationarity_test(trend,4)

#白噪声检测
def whitenoise_test(dataset,number):
    data = dataset.copy()
    data = data.iloc[: len(data)-number] #不使用最后 5 个数据
    from statsmodels.stats.diagnostic import acorr_ljungbox
    [[lb], [p]] = acorr_ljungbox(data[0], lags = 1)
    if p < 0.05:
        print(u'原始序列为非白噪声序列，对应的 p 值为：%s' %p)
    else:
        print(u'原始该序列为白噪声序列，对应的 p 值为：%s' %p)
    [[lb], [p]] = acorr_ljungbox(data[0].diff().dropna(), lags = 1)
    if p < 0.05:
        print(u'一阶差分序列为非白噪声序列，对应的 p 值为：%s' %p)
    else:
        print(u'一阶差分该序列为白噪声序列，对应的 p 值为：%s' %p)

whitenoise_test(trend,4)


#寻找合适参数
p_min=1
p_max=6
d_min=2
d_max=2
q_min=1
q_max=6
#计算不同 ARIMA 模型的 AIC
results_aic=pd.DataFrame(index=['AR{}'.format(i)for i in range(p_min,p_max+1)],
                         columns=['MA{}'.format(i)for i in range(q_min,q_max+1)])
for p,d,q in itertools.product(range(p_min,p_max+1),
                               range(d_min,d_max+1),
                               range(q_min,q_max+1)):
    if p==0 and d==0 and q==0:
        results_aic.loc['AR{}'.format(p), 'MA{}'.format(q)] = np.nan
        continue
    try:
        model=ARIMA(trend[0],order=(p,d,q)) #ARIMA 模型
        #model=sm.tsa.SARIMAX(sentiment_short,order=(p,d,q)) #多元季节性时间序列模型
        results=model.fit()
        results_aic.loc['AR{}'.format(p),'MA{}'.format(q)]=results.aic
```

```
    except:
        continue
print(results_aic)

results_aic=results_aic[results_aic.columns].astype(float)
fig,ax=plt.subplots(figsize=(20,8))
ax=sns.heatmap(results_aic,
               mask=results_aic.isnull(),
               ax=ax,
               annot=True,
               fmt='.2f')
ax.set_title('AIC')

#计算不同 ARIMA 模型的 BIC
results_bic=pd.DataFrame(index=['AR{}'.format(i)for i in range(p_min,p_max+1)],
                         columns=['MA{}'.format(i)for i in range(q_min,q_max+1)])
for p,d,q in itertools.product(range(p_min,p_max+1),
                               range(d_min,d_max+1),
                               range(q_min,q_max+1)):
    if p==0 and d==0 and q==0:
        results_bic.loc['AR{}'.format(p), 'MA{}'.format(q)] = np.nan
        continue
    try:
        model=ARIMA(trend[0],order=(p,d,q)) #ARIMA 模型
        #model=sm.tsa.SARIMAX(sentiment_short,order=(p,d,q)) #多元季节性时间序列模型
        results=model.fit()
        results_bic.loc['AR{}'.format(p),'MA{}'.format(q)]=results.bic
    except:
        continue

results_bic=results_bic[results_bic.columns].astype(float)
print(results_bic)
fig,ax=plt.subplots(figsize=(20,8))
ax=sns.heatmap(results_bic,
               mask=results_bic.isnull(),
               ax=ax,
               annot=True,
               fmt='.2f')
ax.set_title('BIC')



#模型拟合、预测
model=ARIMA(trend[0],order=(2,2,6))
```

```
result=model.fit()
print(result.summary())
pred=result.forecast(200)
trend_predict=pred[0].tolist()
#del(trend_predict[0],trend_predict[-1])
print(trend_predict)

def storFile(data,fileName):
    with open(fileName,'w',newline ='') as f:
        mywrite = csv.writer(f)
        mywrite.writerow(data)

data = trend_predict
storFile(data,'/Users/houguiyang/Desktop/2020 国赛/问题 4-1.csv')
```