目录

第一章 作品概述

1.1 项目理念

“FlowVision 高速流量洞察”，是一个可以可视化高速公路数据信息同时可以动态地预测道路流量的信息可视化平台，其中的“Flow”表示流量，如今随着人们出行频繁，出行的方式也多种多样，但是道路的堵塞是人们无法避免的一个问题，通过我们的后端模型预测可以在一定程度上避免走入拥堵区域，为人们的出行带来更多的便捷，同样对于道路流量预测同样可以防止由于过于拥堵而发生的交通事故，通过我们的预测可以在一定程度上提前做出相应的防护减少诸如这类的情况发生。名称中的 Vision 便是可视化的意思，我们致力于将数据以多元化的形式进行展示，通过这样的可视化展示可以更好地帮助我们洞察数据与数据之间的联系，总体来讲“FlowVision 高速流量洞察”可视化平台致力于可视化相应的数据并结合道路流量预测为高速公路、城市的管理与事故预防提供相应的合理化建议，是现代信息化管理层面的重要组成部分。

1.2 功能简述

“FlowVision 高速流量洞察”集成数据收集、处理、分析和可视化功能。平台对于全国高速公路的数据信息使用多种方式进行处理并将数据多元化表达，平台的整体交互性良好，可以下载相应所呈现的数据以及图表，进而了解其中的数据内涵。此外本平台支持流量预测功能，支持用户提交相应的数据文件，借助机器学习和深度学习算法，使用后端的预训练模型将结果以图表的形式进行呈现，使用者可以根据所预测的数据做出更加合理的决策。

第二章 问题分析

2.1 问题来源

在如今的社会中对于道路拥堵已然是一个不可回避的问题，道路的拥堵很有可能会引发严重的交通事故，同时还会拖延一天的行程，同时对于相应的道路信息数据我们不可避免地要进行分析处理预测，进而作出相应的预防与举措从而避免上述的情况发生，所以需要一个可以提供数据信息可视化良好的平台，同时要具有良好的交互性以及相应的拥堵情况的预测功能，“FlowVision 高速流量洞察”便可以解决上述的问题。

2.2 问题解决

高速数据信息可视化：平台通过 echarts 图表对处理过后的数据进行展示

交通流量的有效预测：通过用户所提交的数据利用模型进行预测，通过可视化的手段直观地呈现给用户

交通事故预防：预测可能出现的高风险交通区域有助于预防交通事故。管理部门可以根据预测结果及时调整交通信号，采取必要的安全措施，比如设置临时限速或警告标志。

出行规划与建议： 通过预测未来的交通流量，平台为用户提供了出行规划建议，这样用户可以选择最佳出行时间和路线，避免堵塞，减少出行压力

高效的交通管理： 管理者可以使用平台提供的数据和分析结果优化交通管理决策。比如，根据流量预测调整交通信号灯的配时，或者是规划道路维修和建设的时间，减少对交通的干扰。

数据驱动的决策制定： 通过收集和分析大量数据，平台可以揭示交通流量的模式和趋势，支持基于数据的决策制定。政策制定者和规划者可以利用这些洞察来改善交通基础设施和公共政策。

2.3 竞品分析

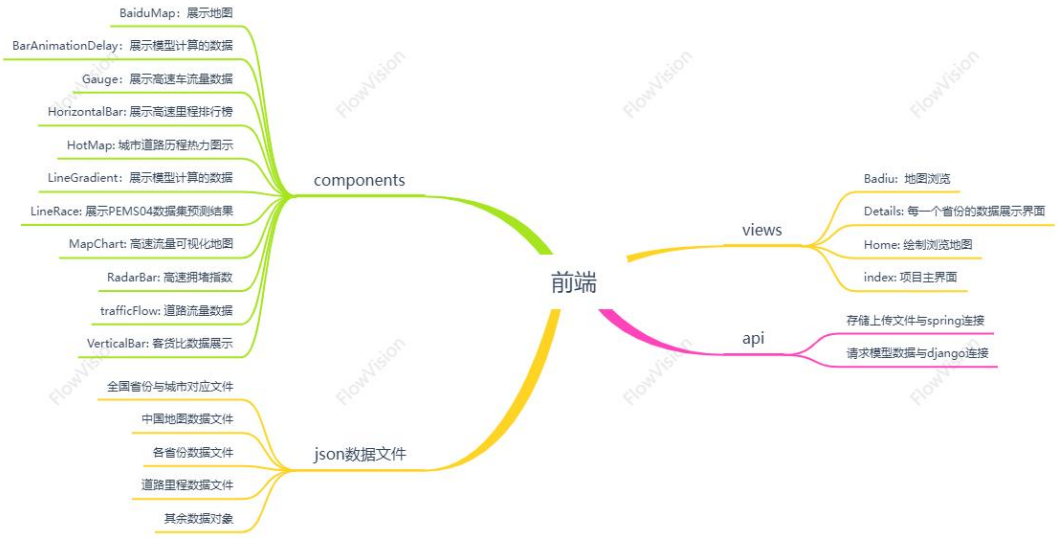
FlowVision 高速流量洞察平台通过集成先进的数据可视化和流量预测技术，提供了一个创新的解决方案，旨在优化道路交通管理并预防交通事故。该平台利用图神经网络，如 ChebNet、GCN 和 GAT，对高速公路数据进行深入分析，并实现动态流量预测，帮助用户和管理者实时了解交通状况。其前端采用 Vue 3 和 ECharts 进行响应式界面设计和复杂图表展示，确保了高交互性和用户体验。后端结合 SpringBoot 和 Django 框架，支持高效数据处理和模型预测，使得 FlowVision 不仅能准确预测交通流量，还能提供基于数据的出行建议和交通管理决策，显著提升城市交通系统的智能化和信息化水平。

第三章 技术方案

3.1 前端实现技术要点：

3.1.1 前端技术方案：

3.1.1.1 前端总体实现脉络图示



前端中设计一系列的组件以及视图，相应的功能已经图示中描述出来了，接下来会详细地介绍前端界面中搭建的要点。

3.1.1.2 技术选择原因

1、响应式布局：通过使用 vue3 中的 ref 以及 reactive 包含数据或是相应的比较复杂的数据对象，我们都可以很方便地实时地修改相应的值，这为我们开发的时候带来了较多的便捷。

2、基于路由的页面跳转：通过前端配置路由我们可以很容易地实现单页面切换，这种切换会更有利于网页的切换的速度与效率，可以在一定程度上提升我们在前端的流畅度

3、基于组件复用以及视图展示的思想：在前端中我们定义了一系列的组件，这些组件是构成页面的重要组成部分，同时这些组件内部存在我们已经定义好的所需的数据格式，从外部传递数据（包含前端中的数据对象以及后端中计算完成的数据）即可显示，在视图中我们定义了项目中比较重要的视图页面，在视图中嵌入上述的组件可以更好地帮助我们展示各种文件之间的关系，这有助于提升我们的前端中的脉络组成。

3.1.1.3 echarts 开源图表介绍

强大的绘图能力：支持常见的图表类型，并且可以通过组合不同的图表构成复杂的数据可视化界面。支持多种坐标系绘制，例如笛卡尔坐标系、极坐标系和地理坐标系（地图）。

高度定制性：提供了多种配置项，用户可以轻松自定义图表的各个组成部分，如图例、坐标轴、提示框、视觉映射组件等。支持主题自定义，可以创建符合个人或公司品牌风格的图表。

交互性：图表交互丰富，例如缩放、拖拽、点击、鼠标悬浮等，提高用户体验。事件处理 API 允许开发者定制用户与图表的交互行为。

动态数据：支持动态数据更新，能够处理实时数据。

提供数据流图表，比如实时动态折线图。

多平台支持：能够在所有现代浏览器上运行，也兼容老版本的 IE 浏览器（使用 canvas 绘制）。

提供了移动优化：可以在不同尺寸的屏幕上保持良好的显示效果。

3.1.1.4 tailwindcss 开源项目介绍

工具类优先：Tailwind 通过提供成百上千的工具类(如 text-center, pt-4, md:w-1/2）让你可以直接在 HTML 中指定元素的样式。

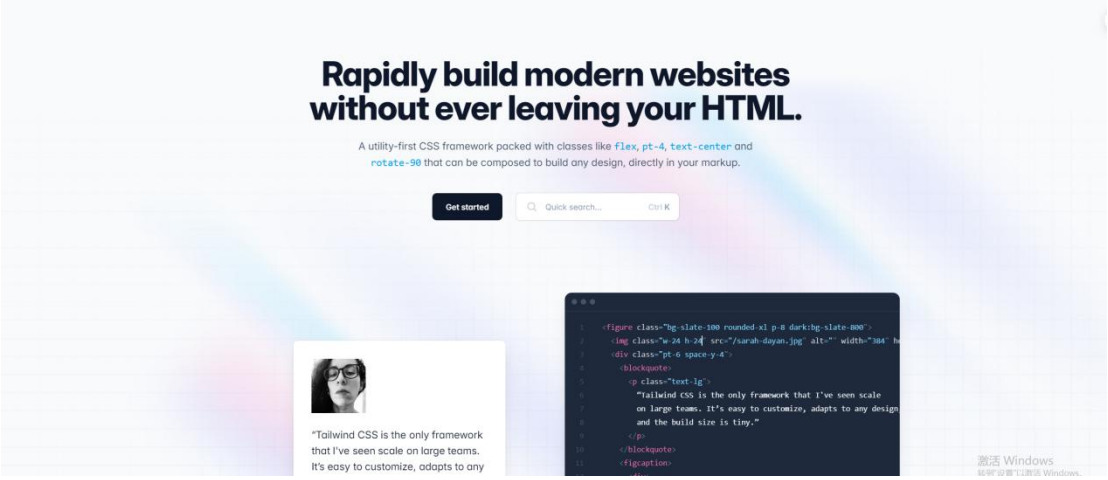
高度定制： 通过配置文件 tailwind.config.js，你可以自定义你的设计系统，包括颜色、字体、间距等，使其符合你的品牌和设计美学。

响应式设计： Tailwind 提供了基于视口大小的响应式工具类，允许你快速为不同的屏幕大小定制样式。

状态变体： 能够为不同的状态（如 hover、focus、disabled）和媒体查询提供样式，使得交互效果的实现更为简单。

插件系统： 可以通过插件扩展 Tailwind 的功能，比如添加新的工具类、组件、动画效果等。

原子性： 每个工具类都是单一的 CSS 声明，这意味着更高的重用性和更小的样式文件。



3.2 后端实现要点

3.2.1 后端实现脉络图示：

在后端中我们主要涉及的是两个流行的开发框架以及相应的流量预测模型，接下来将详细地介绍相应的框架以及模型的工作原理

3.2.2 框架的选择原因：

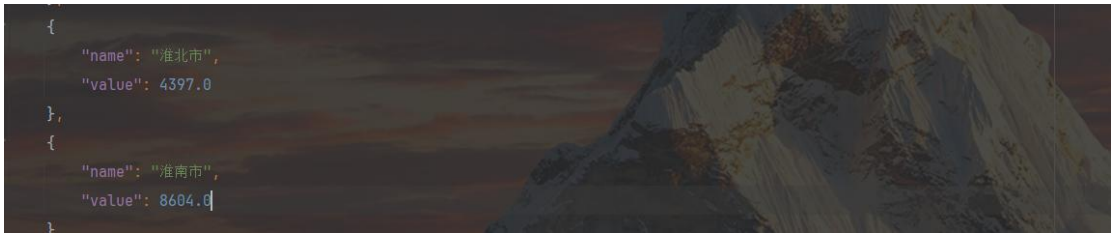
本项目的后端开发是基于 springboot 以及 django 框架开发的，其中 spring 项目主要是处理前端上传的数据文件以及与数据库如何交互，django 主要是利用模型处理前端的请求，并返回相应的计算结果。

由于对于 springboot 中强大的以及便捷的请求处理能力，这有助于我们的平台后续的功能性的迭代升级

对于 django,由于我们的后端流量预测模型是基于深度学习库 pytorch 实现的所以使用 django 框架来集成我们的后端模型，同时 django 较为轻量使用起来比较方便，基于种种原因使用 django 进行开发

3.2.3 数据处理脚本

Django 中存在处理我们所获取到的一系列的数据，这些数据大多为 json 数据以及.csv 格式的数据文件，例如在 django 我们使用了 convert 文件处理的全国市级的道路里程数据



最终将其转换为了这种及其好处理的 json 格式，通过这样的转换一定程度上减轻了前端中 echarts 数据格式的定义以及数据嵌入等工作的任务量。

3.3 流量预测原理

3.3.1. ChebNet (Chebyshev Spectral Graph Convolutional Network)

3.3.1.1 技术原理：

ChebNet 利用切比雪夫多项式 $T_k(x)$ 近似图拉普拉斯算子的谱滤波器。这种方法允许图卷积网络在不同尺寸的图上应用，且不依赖于图的精确特征分解，大幅提升计算效率。

1.1.1 拉普拉斯算子和切比雪夫多项式

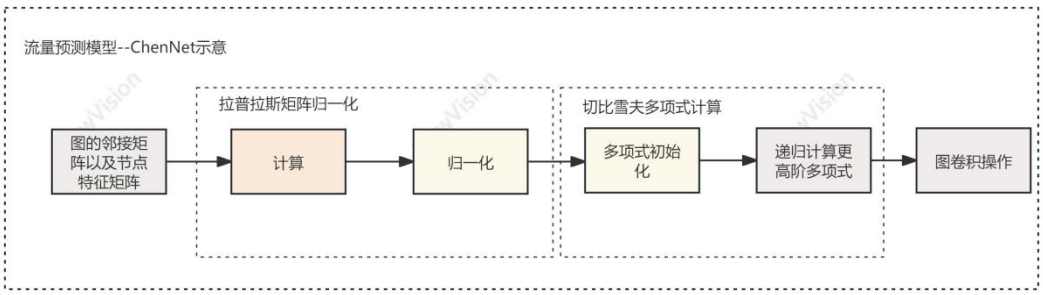
拉普拉斯算子 $L = D - A$ 是图的基本矩阵表示，其中 A 是邻接矩阵，D 是度矩阵。切比雪夫多项式是一种正交多项式系列，可以有效地递归计算，用于近似滤波函数。

1.1.2 图卷积公式

ChebNet 使用切比雪夫多项式对滤波器 g 进行参数化，从而定义图卷积：
$g_\theta(L) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})$
其中，$\tilde{L} = 2L/\lambda_{max} - I$ 是归一化拉普拉斯矩阵，λ_{max} 是 L 的最大特征值，I 是单位矩阵，θ 是滤波器系数。

3.3.1.2 工作流程：

1. **拉普拉斯矩阵归一化**
   - 输入图的邻接矩阵 A 和度矩阵 D。
   - 计算归一化拉普拉斯矩阵 $\tilde{L} = 2L/\lambda_{max} - I$ 其中 $L = D - A$ 是未归一化的拉普拉斯矩阵，I 是单位矩阵，λ_{max} 是 L 的最大特征值。
2. **切比雪夫多项式递归计算**
   - 使用切比雪夫多项式 $T_k(x)$ 的递归定义来计算 $T_k(\tilde{L})$：
     $T_0(x) = 1$
     $T_1(x) = x$
     $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$
   - 这一步骤不需要计算 L 的特征值分解，从而大大降低计算复杂性。
3. **图卷积操作**
   - 定义图卷积核 $g_\theta(L)$ 为多项式的线性组合：
     $g_\theta(L) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})$
   - 对于输入的特征矩阵 X，图卷积输出计算为：
     $Y = g_\theta(L)X = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})X$
   - 其中，θ_k 是模型可学习的参数，代表不同多项式项的权重。



3.3.2 GCN (Graph Convolutional Network)

3.3.2.1 技术原理

GCN 通过直接在图的邻接矩阵上应用卷积操作，以利用节点的邻域信息。GCN 的目的是学习图顶点的低维表示，这可以用于多种预测任务。

2.1.1 图卷积操作

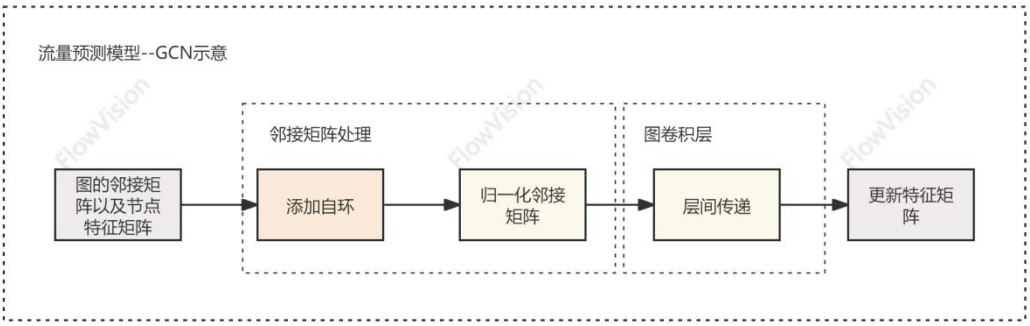
GCN 的图卷积操作定义为：

$H^{(l+1)} = \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H^{(l)}W^{(l)})$

其中，$\hat{A} = A + I$，$\hat{D}$ 是 $\hat{A}$ 的度矩阵，$H^{(l)}$ 是第 l 层的节点表示，$W^{(l)}$ 是该层的权重矩阵，σ 是激活函数。

3.3.2.2 工作流程

1. **邻接矩阵处理**
   - 计算 $\hat{A} = A + I$，其中 I 是单位矩阵，用来加入自环，使得每个节点在聚合邻居信息时也考虑自身特征。
2. **度矩阵归一化**
   - 计算 $\hat{D}$，$\hat{D}$ 中每个元素 $\hat{D}_{ii}$ 是 $\hat{A}$ 第 i 行的和。
   - 归一化邻接矩阵，$\hat{A}_{\text{norm}} = \hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}$。
3. **图卷积层**
   - 对每个图卷积层，计算：
     $H^{(l+1)} = \sigma(\hat{A}_{\text{norm}}H^{(l)}W^{(l)})$
   - $H^{(l)}$ 是第 l 层的节点特征矩阵，$W^{(l)}$ 是该层的权重矩阵，σ 是非线性激活函数（通常是 ReLU）。



3.3.3. GAT (Graph Attention Network)

3.3.3.1 技术原理

GAT 引入注意力机制来动态调整邻接节点间信息的重要性，从而实现图的卷积操作。注意力机制使得GAT不仅关注邻居节点的特征，还关注邻居之间的相对重要性。

3.1.1 注意力图卷积

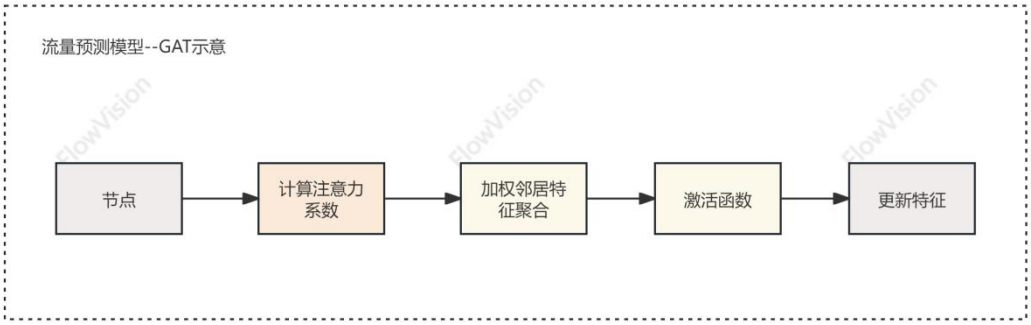
GAT 更新节点特征的方式为：

$h_i' = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W h_j\right)$

其中，α_{ij} 是通过 softmax 函数标准化的注意力系数，用于衡量邻接节点 j 对节点 i 的贡献大小。

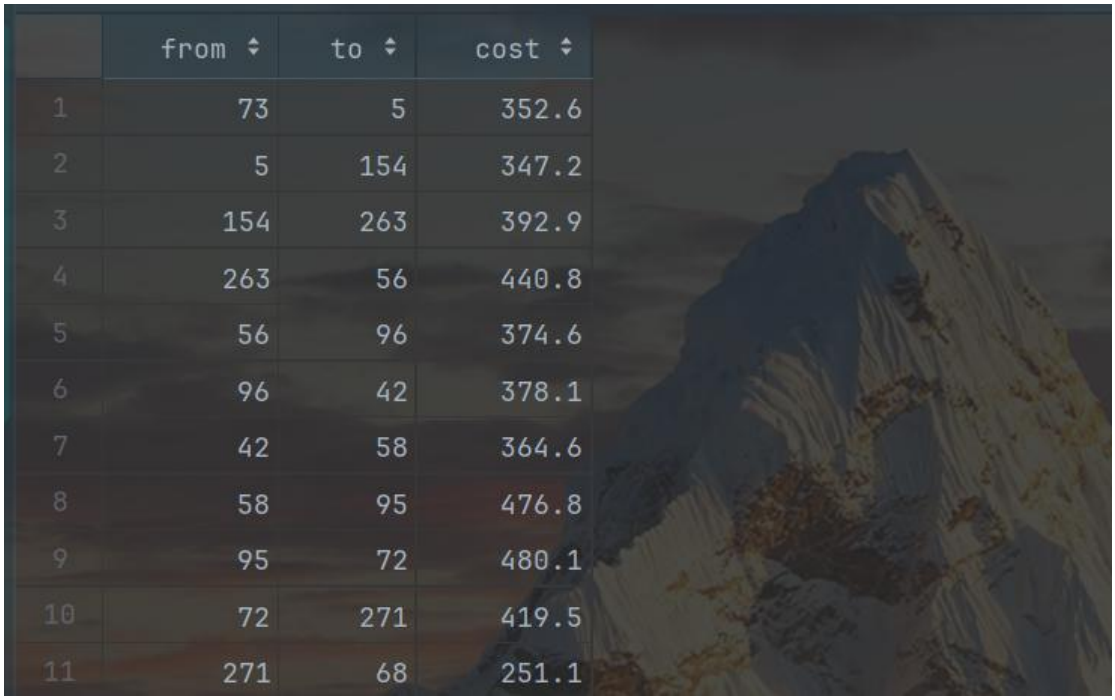
3.3.3.2 工作流程

1. **邻接矩阵处理**
   - 计算 $\hat{A} = A + I$，其中 I 是单位矩阵，用来加入自环，使得每个节点在聚合邻居信息时也考虑自身特征。
2. **度矩阵归一化**
   - 计算 $\hat{D}$，$\hat{D}$ 中每个元素 $\hat{D}_{ii}$ 是 $\hat{A}$ 第 i 行的和。
   - 归一化邻接矩阵，$\hat{A}_{\text{norm}} = \hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}$。
3. **图卷积层**
   - 对每个图卷积层，计算：
     $H^{(l+1)} = \sigma(\hat{A}_{\text{norm}} H^{(l)} W^{(l)})$
   - $H^{(l)}$ 是第 l 层的节点特征矩阵，$W^{(l)}$ 是该层的权重矩阵，σ 是非线性激活函数（通常是 ReLU）。



3.3.4 数据中的网络拓扑结构的表示

数据加载的代码会直接将.csv 中的数据，将其用邻接矩阵的形式表示，其中矩阵的二维坐标可以表示(from, to)，这样我们可以将节点网络的拓扑结构转换为一个矩阵的形式并用于后续模型的预测。



	from	to	cost
1	73	5	352.6
2	5	154	347.2
3	154	263	392.9
4	263	56	440.8
5	56	96	374.6
6	96	42	378.1
7	42	58	364.6
8	58	95	476.8
9	95	72	480.1
10	72	271	419.5
11	271	68	251.1

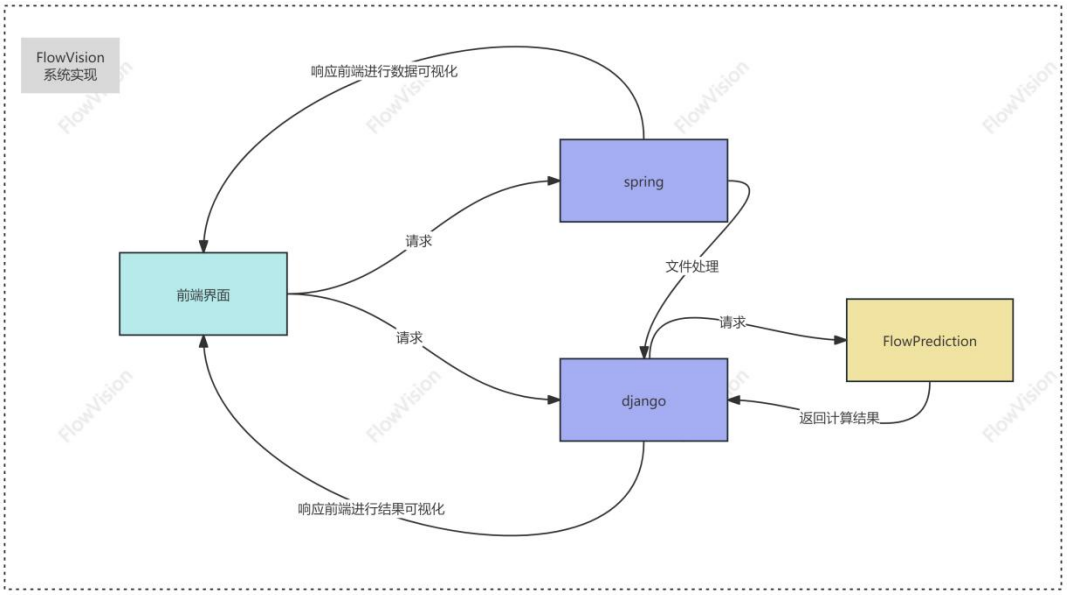
3.3.5、训练模型调参并将保存参数

经过相应的参数调节，我们将模型的参数保留下来，以便前端在进行请求的时候可以直接加载模型（我们这里默认直接加载最优模型，即评估指标最好的模型）以及相应的提交处理过的数据文件，将相应的参数放回给前端。

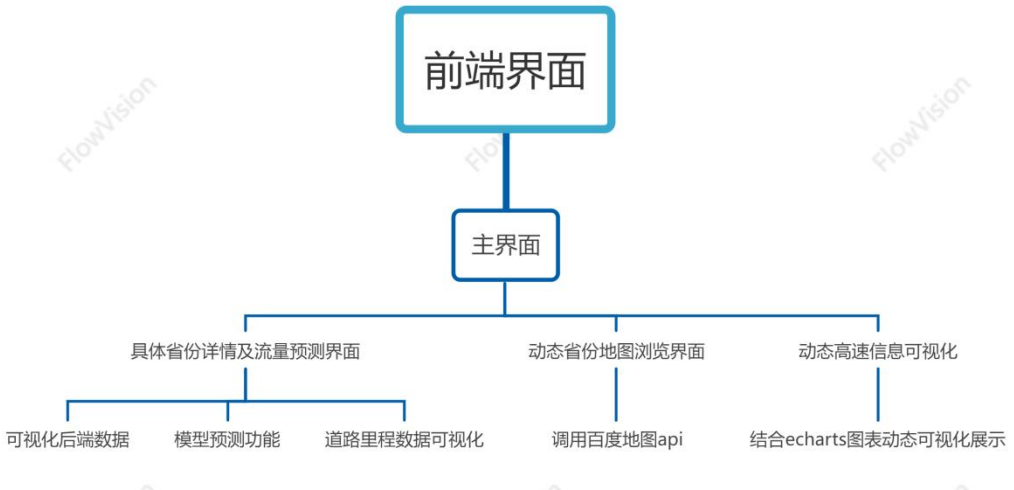
第四章 系统实现

4.1 系统总体实现架构

我们将 FlowVison 平台的总体上结构分为重要的四个模块分别是基于 vue3 框架集成的前端界面以及后端的 spring 框架和 django 框架、FlowPrediction 的预测模型，整体的交互动作以及关系具体如下图所示，下面将详细地讲述整个界面的构成与图表含义以及交互 api 的原理



4.2 界面关系展示



总体上来讲 FlowVison 将数据可视化贯彻到了每一个界面中，在相应的界面中我们都运用了相应的工具来对数据进行系统化的展示，值得一提的是在流量预测界面会使用并感受我们嵌入的深度学习模型，这也是 FlowVison 中的“Flow”的由来，下面将系统地介绍前端的多个界面。

4.3 主界面展示

我们所展示的数据是通过各方搜寻而来的 2019 年到 2021 年的数据，其中在整个界面的正中间有一个展示部分高速公路的车流量动态地图，我们选取了几个关联高速主干道比较多的城市作为 echarts 的关键节点并将其赋予动态的流量展示，同时在下方有一个时间轴，时间序列范围是 2019 年~2021 年，这表明主界面中的相关三年的数据会随着时间轴上的时间节点的变化而动态地展示相应年份的数据。

在界面的左侧有自上而下有：左侧的油表图示是用来展示高速三年的一定时期内的峰值流量（统计的是一天之内的车流量），在其下方的雷达图展示的是相应的高速主干道的拥堵指数（0~10 之间，为衡量拥堵程度的评估指标），可以通过这个雷达图清晰的展示相应的编号的主干道的拥堵指数大小以及逐年变化，在其下方是我们所统计的前六名高速里程的省份数据（单位为公里），可以清晰地看到省份高速里程数据逐年变化的情况。

对于右侧的图示中：最上方的饼状图展示的具有较高流量的高速公路

的比重占比显示，在其下方的图示中我们展示了主干道的客货比（客流量与货运量的比值），在右下角展示的是高速主干道平均流量（取自高速的某一结点，单位为 vph 即辆/小时）数值的柱状排序图。

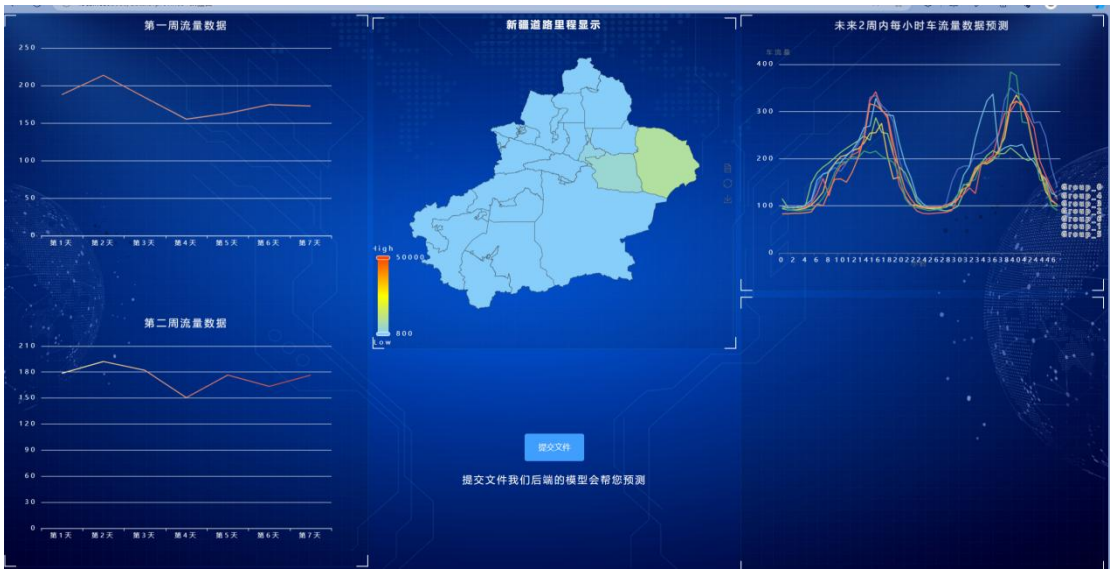
主界面中较为详尽地涵盖了和高速道路相关的数据信息，图表会清晰地动态地展示 2019 年度至 2021 年度的数据，这些数据信息的可视化可以帮助我们较为清晰地捕捉高速数据信息的变化与联系。

4.4 流量预测界面展示

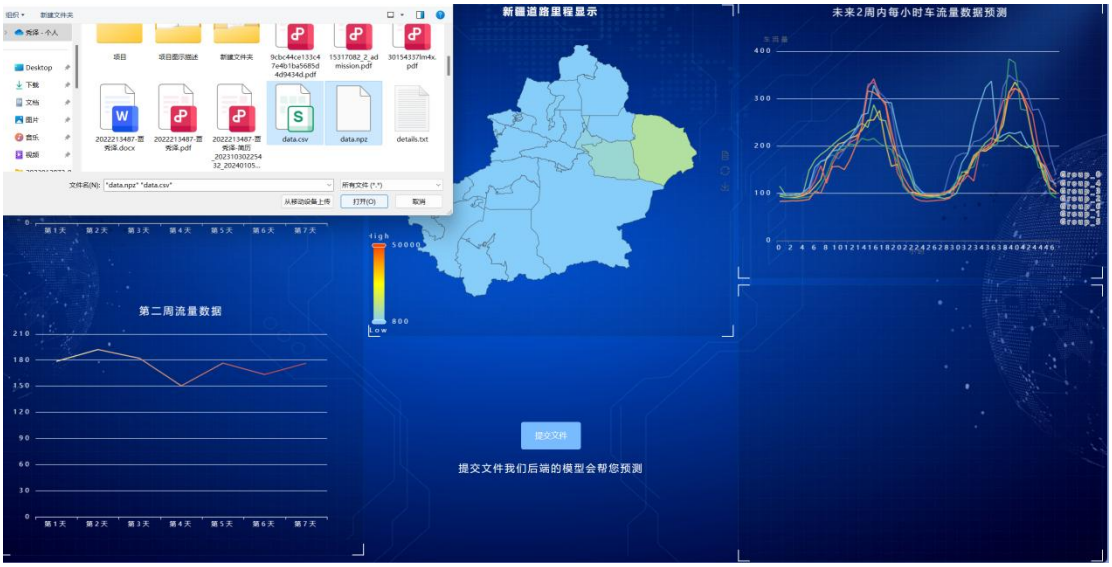
在界面左侧是对于后端计算得到的数据的第一周以及第二周的数据展示，通过简明的线条可以帮助使用者快速地分辨整体的道路流量走势。

在界面的中间部分我们放置了一个热力图示，如果在主界面的地图中点击了不同的省份会跳转到不同的省份的界面，也会展示不同的道路里程数，热力图极为直观地体现了各个省份中的城市中的道路里程数据，这对于相应的流量预测起到了参考参照的作用

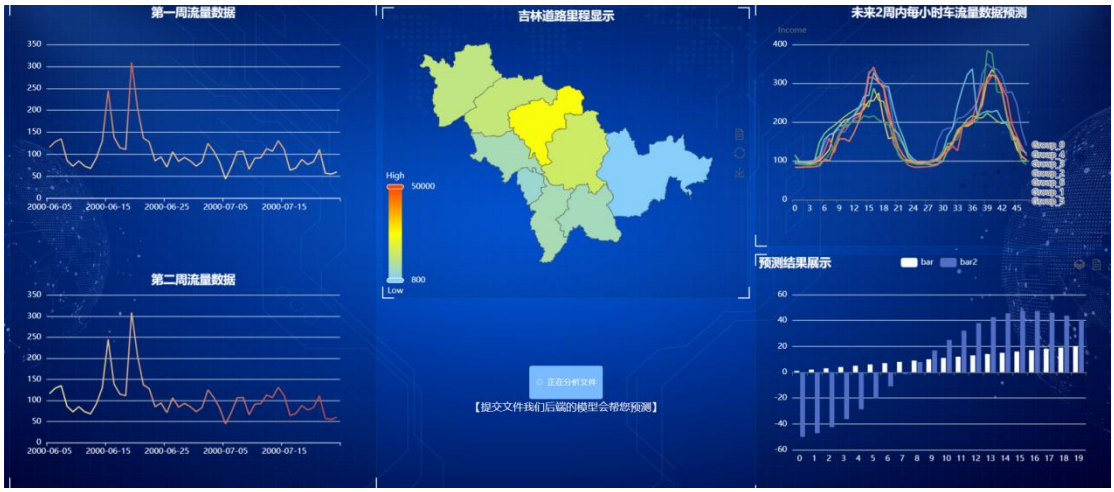
在右侧的图示中，我们在后端中将模型输出的数据做了进一步的处理，将未来两周（设定是 166 号节点的流量数据）的流量结果按照两天分为一组，同时在其中取得了每小时的平均流量组成一组新的数据返回到前端，通过这种形式可以更好地帮助我们了解每个组别中在小时中的流量趋势

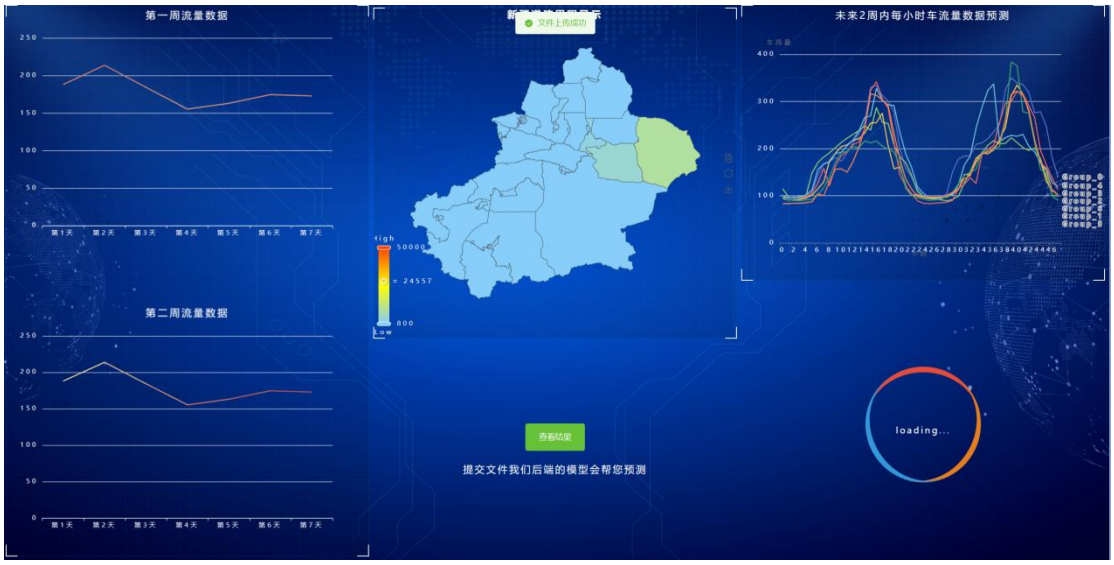


提交文件的时候可以点击选择多个文件（所需要的文件格式是.csv 文件以及.npz 格式的文件），由于在进行提交文件预测的时候右下角的图示会显示相应的预测结果，如果未提交相应的文件会暂时不显示。

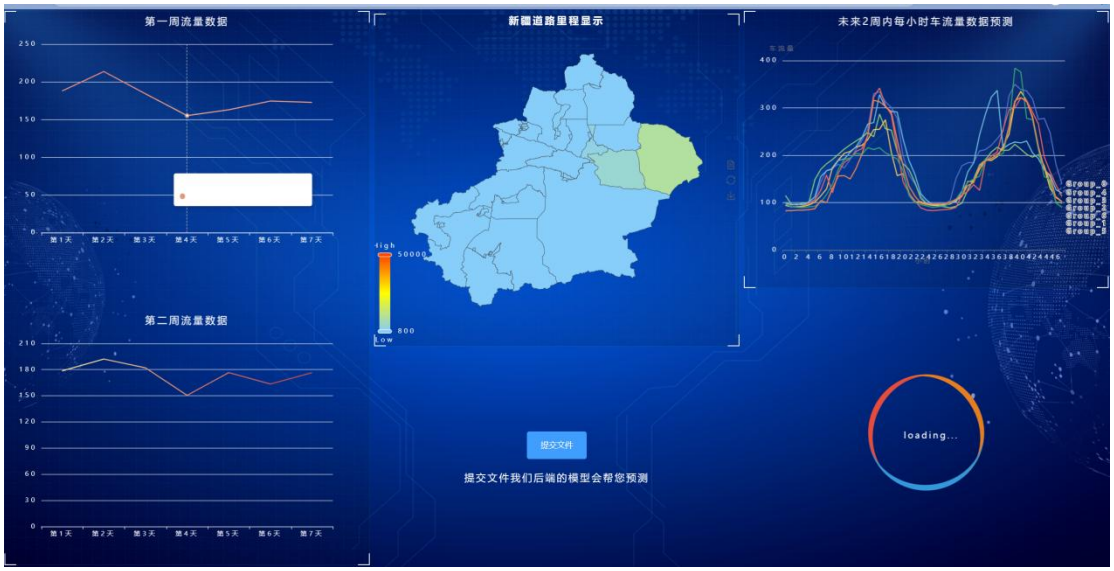


可以通过文件选择器来选择文件，提交格式是.npz 格式的文件以及.csv 文件，前者用于描述相应的时间步内的道路节点的流量数据，后者用于描述整个监控系统的网络拓扑结构，提交之后，平台会向 spring 提交文件处理请求

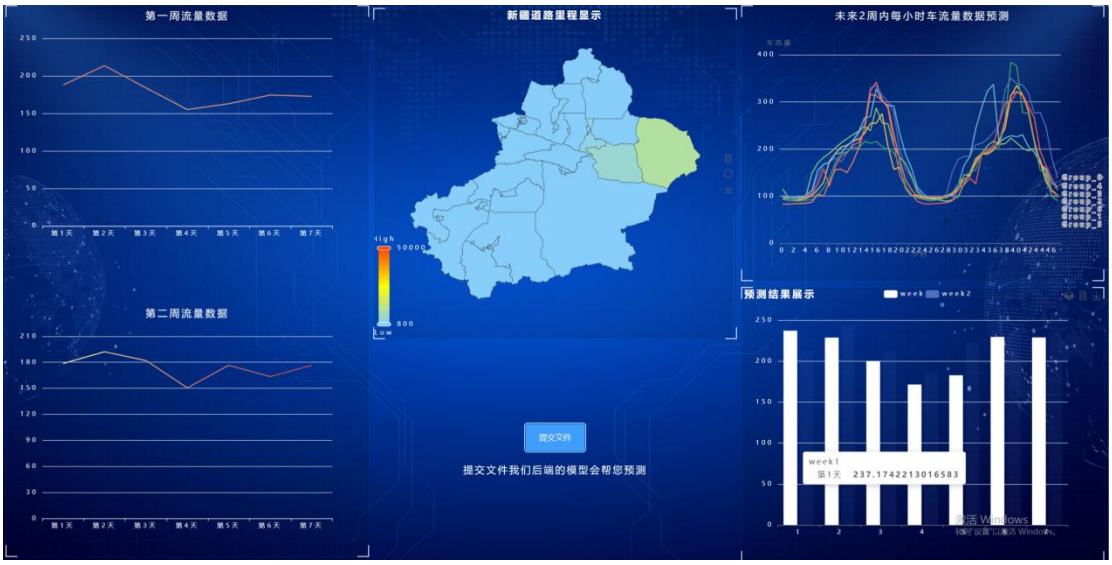




Spring 后端会直接处理这个文件，按钮改变样式为“查看结果”，这时候页面会提示等待

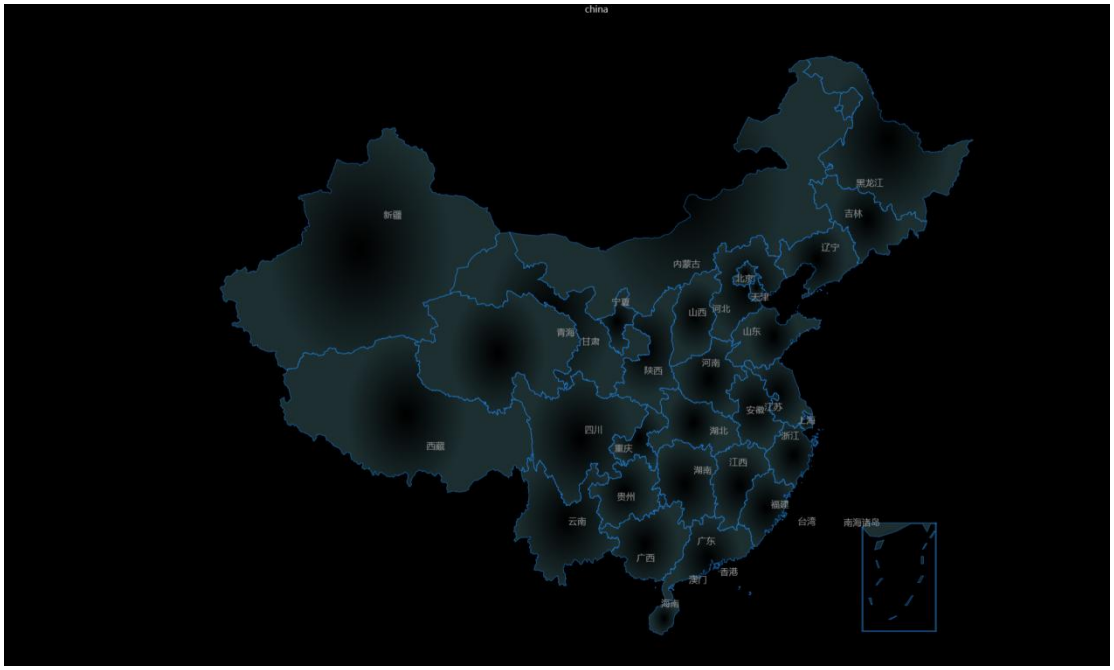


数据会直接显示在右下角的连续的柱状图示中，在右下方使用者可以看到未来所预测的数据结果（后端返回的是前端所提交的网络拓扑结构中的一号节点的两周的流量数据）



4.5 地图浏览功能

在主界面中当点击我们的平台名称的时候会显示我们嵌入的浏览地图的功能，再点击进去之后可以在百度地图中查看相应的高速、城市道路等。



第五章 测试分析

5.1 测试用例

测试编号		01	测试名称	前端界面图表可视化显示测试
测试目的		前端中的逻辑中是否错误，echarts 图表是否配置正确，dom 是否能正确渲染		
预置条件		已经在终端启动项目		
步	简述	测试过程		预期结果
1.	浏览界面	观察界面是否有数据显示，同时使用相应的交互式设计看是否正常		可以正常交互，同时数据可以动态地切换年份正常显示
步骤	简述	异 常 测 试		
		测试过程		预期结果
2.	显示失败	改动数据对象或是配置错误		图表无法正确显示
测试结果		实际测试结果皆符合预期结果， 平台成功通过本测试用例		
测试时间		2023.4.5		

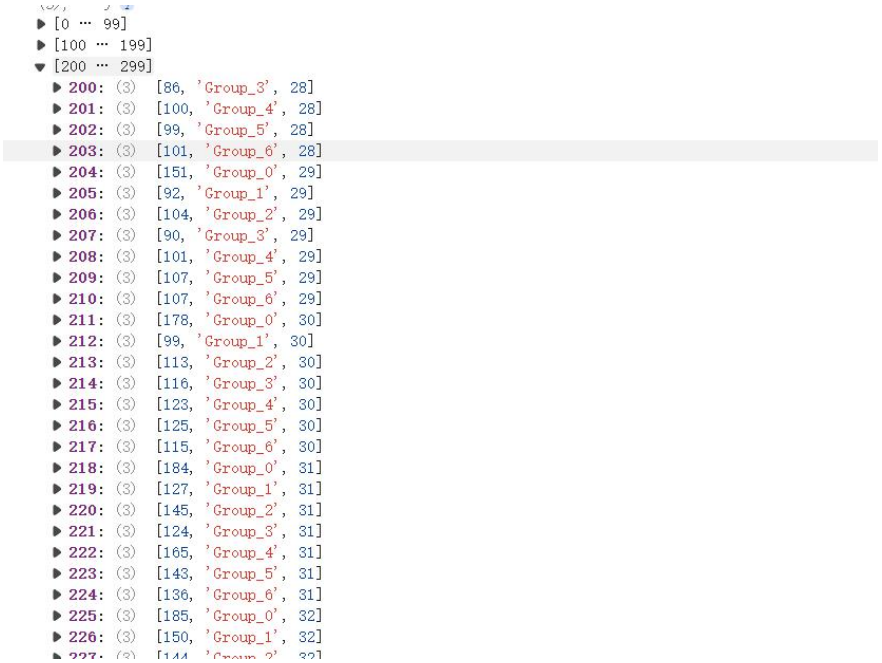
测试编号		02	测试名称	后端接口请求测试
测试目的		后端是否能正常回应相应的网络请求		
预置条件		已经启动后端项目，同时已经使用 postman		
步	简述	测试过程		预期结果
1.	使用 postman	根据返回的数据内容以及预期编写好的打印调试语句进行分析测试，看返回的数据、格式是否正确		Postman 中显示的结果正确可以得到正确的回应以及实现相应的预期效果
步骤	简述	异 常 测 试		
		测试过程		预期结果
2	请求错误	修改 postman 中的请求地址或是修改后端接口的内容		无法找到相应的请求路径或是返回的数据不正确或者 http 状态码返回 500
测试结果		实际测试结果皆符合预期结果， 平台成功通过本测试用例		
测试时间		2023.4.5		

测试编号		03	测试名称	前后端交互测试
测试目的		Vue，django，spring 中的 CORS 配置是否存在问题		
预置条件		所有项目均已开启		
步	简述	测试过程		预期结果
1.	刷新界面	使用浏览器的检查功能，在控制台查看详细信息		可以看到前端打印的后端返回的数据，同时格式内容正确
步骤	简述	异　常　测　试		
		测试过程		预期结果
2	修改配置	再次使用浏览器的刷新功能，在控制台查看信息		发现没有权限访问后端接口
测试结果		实际测试结果皆符合预期结果，　平台成功通过本测试用例		
测试时间		2023.4.5		

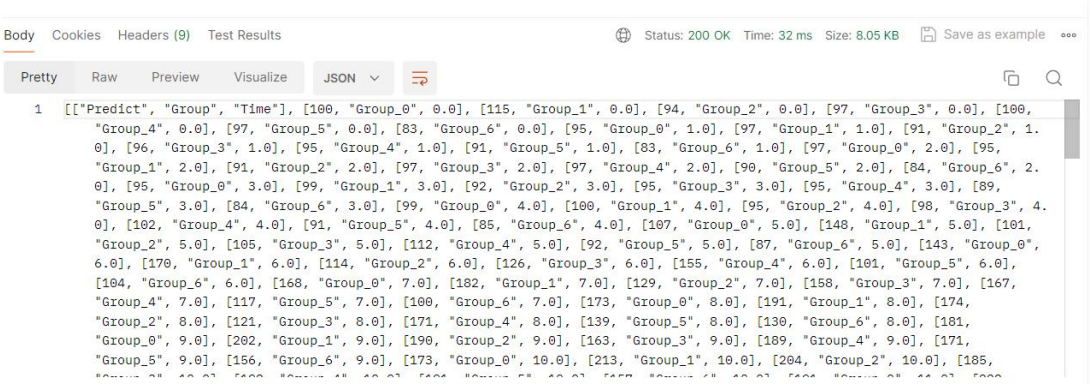
测试编号		04	测试名称	流量预测功能测试
测试目的		测试模型的预测结果		
预置条件		含有相应的评估指标以及可视化结果显示		
步	简述	测试过程		预期结果
1.	单独运行后端预训练模型	等待控制台响应，并查看相关结果		得到了预期的打印内容，查看图表模型运行正常
步骤	简述	异　常　测　试		
		测试过程		预期结果
2	修改部分代码	修改模型的参数或是部分逻辑		报错或差错放大
测试结果		实际测试结果皆符合预期结果，　模型成功通过本测试用例		
测试时间		2023.4.5		

5.2 过程图示

下图是在前后端交互的时候返回 django 模型中计算所返回的结果，并将其在控制台打印输出的结果



使用 postman 测试后端接口



模型运行打印输出