余冰芯

女 | 38岁 15867114649 mqzzxmdw@163.com
13年工作经验 | 算法工程师 | 期望薪资：30-60K | 期望城市：上海

个人优势

1、精通 MYSQL和 ORACLE 数据库,大学学过C++.
2、熟悉 KETTLE
3、熟悉数仓模型设计.
4、 数据中台的搭建 hdfs,mapreduce,yarn hive ,spark ， 实现从0-1搭建。
5、熟练 python numpy ， k-mean ,神经网络 cnn,lstm ， 自然语言(NLP算法模型)，申请评分卡算法模型，使用 requests 等实现数据爬取,pytorch和tensorflow，opencv,chatGML的算法模型的使用，基于gpt-4和chat-gpt的算法模型的使用。
6、熟练 hivesql , impala，kafka,flink 。
7、tableau ， power bi
8、存储过程的开发
9、数据迁移

工作经历

上海神州数码有限公司 算法开发 2023.05-2025.05

内容:

算法开发，不限于数据生成，提示词数据的整理，（上海银行项目）

业绩:

完成相应工作

中航控股集团有限公司 算法工程师 2021.03-2023.03

1.数据挖掘算法工作工作:决策树,kmean分类,神经网络lstm预测,pyechart可视化,业绩:完成相应工作
参与完成数仓的搭建和后期的数据挖掘
bi数据可视化
2.分布式大数据平台的搭建，包括hive，mapreduce，hdfs，yarn，spark，免密设置，后台用mysql数据库。
3.数仓的建模，数据表的设计，关系的设计
4.数据的加载
5.数据倾斜等问题的解决，表分区的设计，桶的设计，全量表和增量表的设计。
6.存储格式的选取
7.优化设计，包括sql优化和加载效率的模式优化

天阳宏业科技股份有限公司 算法工程师 2020.03-2020.12

数据挖掘,1.评分卡模型的开发
2.利用louvain进行欺诈模型的开发，用对抗网络将数据转化成图数据，再用community包中的Louvain算法进行社区划分，将一个全局社交网络拆分成了一个个社区，节点协同分类算法(Collective Classification)并用它来预测每个非欺诈节点（未知节点）的欺诈概率，节点数适中且欺诈节点占比比较大的社区为欺诈团伙。
①初始化网络中所有节点的欺诈概率，已确认欺诈的节点的欺诈概率为1，未知节点的欺诈概率为0；②设定迭代次数，一般设置为6就可以。③待迭代终止后，每个未知节点对应的概率即为其欺诈概率。
宁波银行项目

华信智原教育科技有限公司 算法工程师 2017.05-2019.12

内容:

项目描述:

为了是患者选择更加自主便捷的眼科寻医问药、医疗器材信息,使患者做出更加适合自己的选择。

根据距离,价位,以往案例,医院等级,医生等级,患者病症描述,患者医疗需求和意向,推荐相应的医院、医生和医疗设备,职责:

1、Python医疗数据的爬取

2、Python数据的分析

Python推荐算法的编写

业绩:

作为项目经理,带领部门12个人一起合作完成项目

杭州中奥科技有限公司 etl 2015.05-2017.02

数据的抽取，数据的清洗，数据的加载

安庆326地质队 土木/土建/结构工程师 2012.09-2015.04

内容:

矿产数据的采集、分析、开发

业绩:

完成项目，提交报告

项目经历

3d实时瘦脸 开发 2025.04-至今

内容:

训练一段3d视频，使得实时拍摄的人脸瘦成视频里面的轮廓，进行情感迁移学习，骨骼迁移学习，轮廓迁移学习，3dmm+wdcgan+三角破分+mls变形+tensirboard监控

业绩:

完成全部项目开发

多模态医学器官图像分割系统 开发 2025.04-2025.06

内容:

利用利用人体器官和血管数据，对器官图进行三维分割，加入注意力机制进行消融实验，计算dice系数，网格分割加入曼巴模块，利用u-net构建神经网络。

加载nifti文件，读取数据组并且确保形状相同，利用z-zero归一化数据，寻找包含足够血管素的patch位置，计算血管占比图，生成候选坐标。然后创建固定大小的数组，计算实际可提取的范围，计算在patch中的位置。安全提取数据块填充到patch里面去。

然后应用数据增强 - 仅使用安全增强方法

if self.augment and random.random() > 0.5:

body_patch, vessel_patch = self._augment(body_patch, vessel_patch)

接着转换成tensor数据，确保样本具有相同的维度顺序。随机翻转，随机进行亮度调整，随机进行对比度调整，网络结构去下

class MambaBlock(nn.Module):

"""轻量级曼巴模块 (3D) - 保持通道数不变"""

def __init__(self, channels):

super(MambaBlock, self).__init__()

```
# 深度可分离卷积
self.dw_conv = spectral_norm(nn.Conv3d(
channels, channels, kernel_size=3,
padding=1, groups=channels
))
# 点卷积: 输出通道数改为channels（与输入相同）
self.pw_conv = spectral_norm(nn.Conv3d(channels, channels, 1))
self.norm = nn.InstanceNorm3d(channels)
self.activation = nn.GELU()

def forward(self, x):
identity = x
x = self.dw_conv(x)
x = self.pw_conv(x)
x = self.norm(x)
return self.activation(x + identity) # 输出通道数仍然是channels
class ChannelAttention(nn.Module):
"""通道注意力机制"""
def __init__(self, channels, reduction=8):
super(ChannelAttention, self).__init__()
self.avg_pool = nn.AdaptiveAvgPool3d(1)
self.max_pool = nn.AdaptiveMaxPool3d(1)

self.fc = nn.Sequential(
spectral_norm(nn.Conv3d(channels, channels // reduction, 1)),
nn.ReLU(inplace=True),
spectral_norm(nn.Conv3d(channels // reduction, channels, 1)),
nn.Sigmoid()
)

def forward(self, x):
avg_out = self.fc(self.avg_pool(x))
max_out = self.fc(self.max_pool(x))
return x * (avg_out + max_out)
class ResidualBlock(nn.Module):
"""带谱归一化和通道注意力的残差块 """
def __init__(self, in_channels, out_channels, use_attention=True):
super(ResidualBlock, self).__init__()
self.conv1 = spectral_norm(nn.Conv3d(in_channels, out_channels, 3, padding=1))
self.norm1 = nn.InstanceNorm3d(out_channels)
self.conv2 = spectral_norm(nn.Conv3d(out_channels, out_channels, 3, padding=1))
self.norm2 = nn.InstanceNorm3d(out_channels)
self.activation = nn.GELU()

self.attention = ChannelAttention(out_channels) if use_attention else None
```

```
# 快捷路径
if in_channels != out_channels:
self.shortcut = spectral_norm(nn.Conv3d(in_channels, out_channels, 1))
else:
self.shortcut = nn.Identity()

def forward(self, x):
identity = self.shortcut(x) # 如果通道数需要调整，shortcut会处理

out = self.activation(self.norm1(self.conv1(x)))
out = self.norm2(self.conv2(out))

if self.attention:
out = self.attention(out)

# 注意：这里我们假设空间尺寸没有变化（因为padding=1），所以不需要调整空间尺寸。
# 但可能出现通道数相同但空间尺寸不同（例如输入输出尺寸不一致的情况）？在VesselSegNet中，残差块后面通常是下采样，所以同一残差块内空间尺寸不变。
# 因此，我们去掉之前的尺寸调整，仅使用短路连接（shortcut）来匹配通道数。

return self.activation(out + identity)
class VesselSegNet(nn.Module):
"""轻量级3D血管分割网络"""
def __init__(self, in_channels=1, out_channels=1, init_features=16,
use_mamba=True, use_attention=True):
super(VesselSegNet, self).__init__()
features = init_features

# 编码器路径
self.enc1 = ResidualBlock(in_channels, features, use_attention) # 1->16
self.pool1 = nn.MaxPool3d(2)

self.enc2 = ResidualBlock(features, features*2, use_attention) # 16->32
self.pool2 = nn.MaxPool3d(2)

self.enc3 = ResidualBlock(features*2, features*4, use_attention) # 32->64
self.pool3 = nn.MaxPool3d(2)

self.bottleneck = nn.Sequential(
ResidualBlock(features*4, features*8, use_attention), # 64->128
MambaBlock(features*8) if use_mamba else nn.Identity(), # 输入128，输出128
ResidualBlock(features*8, features*8, use_attention) # 输入128，输出128
)
```

# 解码器路径（修复通道不匹配问题）
# 上采样3：128->64
self.up3 = nn.Sequential(
nn.Upsample(scale_factor=2, mode='trilinear', align_corners=True),
spectral_norm(nn.Conv3d(features*8, features*4, 1))
)
# 拼接后通道：64(up3输出) + 64(e3) = 128
self.dec3 = ResidualBlock(features*8, features*4, use_attention) # 128->64

# 上采样2：64->32
self.up2 = nn.Sequential(
nn.Upsample(scale_factor=2, mode='trilinear', align_corners=True),
spectral_norm(nn.Conv3d(features*4, features*2, 1))
)
# 拼接后通道：32(up2输出) + 32(e2) = 64
self.dec2 = ResidualBlock(features*4, features*2, use_attention) # 64->32

# 上采样1：32->16
self.up1 = nn.Sequential(
nn.Upsample(scale_factor=2, mode='trilinear', align_corners=True),
spectral_norm(nn.Conv3d(features*2, features, 1))
)

业绩：

完成项目的需求分析，代码开发

Agent 3：推理与行动的协同——通过LangChain中的ReAct框架实现自动定价　开发人员

2025.03-2025.06

利用SerpApi和llm-math（这是通过大模型进行数学计算的工具，用于计算售价）使用create_react_agent函数来创建ReAct Agent，并在初始化的过程中指定大模型、工具和提示词
vLLM在linux上环境安装：pip install vllm，vllm serve root/autodl-tmp/DeepSeek-R1-Distill-Qwen-7B.然后用ssh代理工具，将代理窗口改为8000。调用是输入密钥和base_url，建立模型时要指定模型路径
sglang的用法与vllm类似。

纳米级的图片融入到微米级图像分析系统　ai开发工程师

2025.04-2025.05

内容：

读取文件夹1下的每一张微米ct图片，二维，图片路径，里面的图片放大1.2倍。读取文件夹2纳米二维ct图片，纳米图片缩小1000倍。做跨尺度融合。提取缩小后的纳米图和扩大后的微米图的特征，如果最相像的位置是空的将纳米图贴到特征最相像的微米图上，如果最相像的位置不为空，只考虑最相似的位置，相似度要求大于0.8
，白色的矿体和大空隙不可贴和黑色部位，孔隙度一半来自纳米图，一半来自微米图。每张微米图都要文件夹2里面的所有纳米图进行特征提取，对符合条件的融合，
进行空隙连通性评价，可视化融合前后空隙图，要空隙图上保持原始缝隙，不需要过程，空隙图要灰度图，空隙为黑色，其他为白色，可视化连通性评价，matplotlib可以显示中文，加速训练，优化内存，跨模态融合
配置日志
logging.basicConfig(level=logging.INFO,

format='%(asctime)s - %(levelname)s - %(message)s',
handlers=[
logging.FileHandler("ct_fusion.log"),
logging.StreamHandler()
])

设置Matplotlib支持中文显示
mpl.rcParams['font.sans-serif'] = ['SimHei', 'Arial Unicode MS', 'Microsoft YaHei']
mpl.rcParams['axes.unicode_minus'] = False
plt.switch_backend('Agg') # 使用非交互式后端提高性能
模型中配置了日志，以便监控，使用非交互式提高性能。
使用imageio作为健壮的图像读取方法，先将图像转化成灰度图，用tifffile读取tiiff图像。安全缩放图像避免OOM。
计算余弦相似度，在相似度大于0.8且相似度最高的位置选择最佳位置融合随机采样位置，并且验证位置是否有效。然后更新最佳位置。最后，无缝融合纳米图和微米背景，使用99%背景+1%纳米图的比例进行融合。创建边界检查，创建空隙掩模。用形态学操作去掉小噪点。创建羽化掩模实现平滑过渡，使用高斯模糊创建渐变效果。边缘区应用渐变融合，使用自适应阈值处理，计算空隙度，进行连通性分析。过滤小空隙。然后可视化融合前后的图像，可视化连通性分析图像

业绩:
完成项目，对项目可视化

电力设备红外监控系统　开发　2025.04-2025.05

内容:
利用红外图像对电力设备进行自动监控，识别温度异常的电力器件，并且进行识别，输出器件名称，危险程度，以及处理建议

业绩:
完成项目，交付结果

利用对抗网络生成自动驾驶的轨迹的数据生成，用碰撞等级验证　开发人员　2025.04-2025.04

内容:
检测车道线，为了验证极端情况下车辆自动驾驶的安全问题，gan和wdcgan进行对比。检索增强生成，生成器格判别器都是rag,优化器为adam.模型包括数据预处理，残差连接，谱归一化，消融实验。数据进行剪枝，优化，调整学习率，数据的隐藏层的数值和epoch进行模型优化，及时清理缓存，便于模型快速运行，gpu加速训练，整个模型进行，整个模型进行剪枝处理
训练时，将数据迁移到gpu上，定时清理内存，采用half数据结构减少数据量，分批次训练，异常值要进行处理，利用箱形图。然后归一化数据。对数据进行10hz采样，设定每帧最大横像位移，然后检测车道跳变点，检查是否为相邻车道变化，然后提取变道窗口，前后扩展2秒。验证横向运动连续性。标记车道线。然后计算有效车道范围，过滤静止车辆，过滤异常纵向跳跃，然后按照车辆分组排序，计算加速度，使用savitzky-golay过滤器处理平滑横向位置。然后，用速度作为颜色通道，将数据预处理为图像，作为输入数据。整个神经网络，加入谱归一化，注意力机制（包括通道注意力机制，空间注意力机制，应用注意力机制），同时加入残差，用残差连接，保证模型稳定。生成器中包括输入层，初始层，上采样层，输出层判别器中包括输入层，下采样层，输出层，损失函数用wasserstein_loss，优化器用RMSprop,学习率是0.000002。然后计算碰撞等级进行验证，用5种等级，极低风险，低风险，中风险，高风险，极高风险。最后对生成的轨迹和原始轨迹进行可视化对比

业绩:
完成整个项目的开发

车流量预测 深度学习开发工程师 2025.03-2025.03

内容:

根据路口，车流量，时间，堵车标签预测接下来是否会堵车。利用torch，cnn+transformer框架，进行时序预测。先将堵车文字转换成1-0标签。通过滑动时窗将数据转换成便于torch输入的张量。通过分割点的比例区分训练数据和预测数据。模型包括模型的训练预测和评估

model = Transformer(input_dim = num_features, # 输入特征维度

hidden_dim =80,

num_layers =6,

num_heads =80,

output_dim = 1)

criterion_mse = nn.MSELoss() # 定义均方误差损失函数

criterion_mae = nn.L1Loss() # 定义平均绝对误差损失

optimizer = torch.optim.Adam(model.parameters(), lr=0.0001) # 定义优化器

batch_size, seq_len(time_steps), input_dim

summary(model, (32, time_steps, num_features))

在加速推理和考虑模型准确率的时候尤其要考虑hidden_dim的值，hidden_dim大，可以加速收敛，但是可能会导致收敛速度过快，epoch还在继续而使得模型准确率由一开始的上升，上升到一定位置后又重新下降的问题，同时，学习率大也会导致这种情况，所以，在模型推理时要考虑这三个参数的最优组合。

业绩:

完成整个项目的开发，包括数据预处理，损失函数，模型误差调优，框架的搭建

图像识别系统 开发 2024.10-2024.11

内容:

交通车牌识别计数系统的开发，音频相似度分析

业绩:

完成相应开发工作

电梯双模型 算法开发 2024.09-2024.10

用look算法和qlearning算法开发电梯双模型,要求等待时间最少,能量最少。此项目是强化学习，对电梯的路径规划

问答系统 算法 2024.08-2024.09

知识标注，问答系统的开发

1,将词片段分 类，转换成可以监督的分类，

2,将文本词元话

3用滑动窗口处理长文本，通过查找重复部分od解码

4通过稀疏检索器将使用词频将文本和查询表示为一个稀疏向量。通过计算向量内积来确定查询和文档的相关性。

Elasticsearch计算出的查询分数（分数越高，匹配度越高）

5评估pipeline,评估检索器，计算召回率，F1分数

指纹识别系统 算法 2024.06-2024.06

内容:

matlab指纹识别系统的开发

业绩:

完成指纹识别系统的开发

红蜻蜓包包数仓的搭建 数据建模分析工程师 2022.12-2023.12

1.sqoop和etl抽取数据，加载数据到ods层，ods层设计分区，设计全量表和增量表，设计不同的数据抽取时间和频次。建立数据抽取异常警告方案，定时处理异常问题

2.对ods层数据按照主题进行汇总，同时也会按照不同数据的不同使用周期汇总，

3.对汇总的数据进一步进行加工，以便适应不同的使用场景。并且对数据进行优化包装，使得数据可以更加高效便捷的使用

TF-IDF算法实现相似度专利文本推荐 数据挖掘 2022.04-2022.04

全部代码实现,算法说明2022.04

内容:

指定专利文本,搜寻1万个左右专利中最相似的前10个专利文本,要求使用神经网络的算法,通过算法计算文本的相似度的值。首先计算词频,然后利用稀疏矩阵以及TDIDF算法设置参数,获取相似性系数,然后根据系数或者相应文档,在处理的过程中要注意相应的数据格式的对应

业绩:

使用tensirflow神经网络Tf idf算法,完成相应需求首先使用结巴分词,对文本内容进行处理,再利用算法进行参数的设计,然后的到相似度的值,获取前10的值。然后再把相应的相似度的文本给找出来

中国眼科网app推荐系统 算法 2017.05-2019.12

内容:

为了是患者选择更加自主便捷的眼科寻医问药、医疗器材信息,使患者做出更加适合自己的选择。

根据距离,价位,以往案例,医院等级,医生等级,患者病症描述,患者医疗需求和意向,推荐相应的医院,医生和医疗设备,业绩:

1、医疗数据的采集,爬取

2、利用python编写推荐算法，推荐算法指标的选择，排精排，重排

3，推荐算法的权重计算和调优

4,推荐算法分数的计算

5,推荐算法的维护，运维，监控，优化

6,推荐算法新颖性的计算

Python数据爬取 开发工程师 2016.05-2017.03

内容:

1、数据需求的对接

2、Python数据的爬取清洗,导入、导出业绩:

国标数据爬取

教育经历

中国地质大学 本科 地球物理 2008-2012

资格证书

大学英语六级