深度学习报告

July 5, 2025

摘要

TODO: 稍后补充

目录

1 引言

在现代商业环境中，选址决策是品牌扩张和市场布局过程中至关重要的一环。一个优质的门店位置不仅能够提升品牌曝光度和客流量，还直接影响企业的经济效益和市场竞争力。传统的选址方法主要依赖于专家经验和简单的人口统计分析，存在主观性强、难以量化、适应性差等局限。随着大数据、机器学习和深度学习技术的快速发展，基于数据驱动的选址预测方法为选址预测提供更多的思路启发，其能够通过对历史数据的深入分析，挖掘出潜在的空间模式和内在偏好，从而实现更科学、精准、有效的选址决策。

本课题聚焦于“商业智能选址预测”，旨在通过分析品牌历史门店的地理分布数据，建立能够预测品牌下一家门店最优选址的模型。具体而言，课题以网格（Grid）为基本地理单元，利用品牌在城市中的历史开店网格序列及每个网格的地理属性特征，挖掘品牌扩张的空间模式和内在偏好。通过对历史数据的建模与学习，预测品牌未来最有可能选址的网格位置，为企业提供科学、量化的决策依据。

本项目的数据集包含多个品牌的历史门店分布（训练集和测试集），以及覆盖研究区域的网格地理坐标信息。提供数据已划分为训练集和测试集，网格划分保证了空间分析的精度和一致性。研究区域的经纬度范围明确，便于空间特征的提取和建模。

在方法设计上，项目不仅实现了数据预处理、特征增强、模型训练与评估等完整流程，还探索了多种深度学习与空间分析方法，包括序列建模（如 RNN、LSTM、Transformer）、空间关系建模（如图神经网络 GNN）以及多模态信息融合等。针对品牌门店分布的空间依赖性和数据稀疏性等挑战，项目尝试引入迁移学习等先进技术，以提升模型的泛化能力和预测准确率。

本课题的研究不仅具有重要的理论意义，也为实际商业选址提供了可行的智能化解决方案。

A 附录

A.1 项目信息

GitHub 仓库地址： https://github.com/szw0407/DL-project-2025

A.2 核心模型代码

A.2.1 神经网络模型实现 (model.py)

```python
import torch
import torch.nn as nn

class SeqEncoder(nn.Module):
    def __init__(self, vocab_size, embed_dim, lstm_hidden, lstm_layers, dropout):
        super().__init__()
        self.embed = nn.Embedding(vocab_size, embed_dim)
        self.lstm = nn.LSTM(embed_dim, lstm_hidden, lstm_layers,
                            batch_first=True, dropout=dropout if lstm_layers > 1 else 0)
    def forward(self, seq_ids):
        emb = self.embed(seq_ids)
        out, (h, _) = self.lstm(emb)
        return out[:, -1, :]

class MLPEncoder(nn.Module):
    def __init__(self, in_dim, out_dim, hidden_dim=32):
        super().__init__()
        self.model = nn.Sequential(
            nn.Linear(in_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, out_dim),
            nn.ReLU()
        )
    def forward(self, x):
        return self.model(x)

class NextGridPredictor(nn.Module):
    def __init__(self, num_classes, embed_dim=32, lstm_hidden=64, lstm_layers=1,
                 coord_dim=2, poi_dim=10, coord_out_dim=16, poi_out_dim=16, fusion_dim=64, dropout=0.1):
        super().__init__()
        self.seq_encoder = SeqEncoder(num_classes, embed_dim, lstm_hidden, lstm_layers, dropout)
        self.coord_encoder = MLPEncoder(coord_dim, coord_out_dim)
        self.poi_encoder = MLPEncoder(poi_dim, poi_out_dim)
        self.fusion = nn.Sequential(
            nn.Linear(lstm_hidden + coord_out_dim + poi_out_dim, fusion_dim),
            nn.ReLU(),
            nn.Dropout(dropout),
            nn.Linear(fusion_dim, fusion_dim),
            nn.ReLU(),
            nn.Dropout(dropout)
        )
        self.classifier = nn.Linear(fusion_dim, num_classes)
    def forward(self, seq_ids, seq_coords, seq_poi):
        # 输入: (batch, seq_len, dim)
        seq_out = self.seq_encoder(seq_ids)                  # (batch, lstm_hidden)
        coords_out = self.coord_encoder(seq_coords.mean(dim=1)) # (batch, coord_out_dim)
        poi_out = self.poi_encoder(seq_poi.mean(dim=1))     # (batch, poi_out_dim)
        x = torch.cat([seq_out, coords_out, poi_out], dim=-1)
        f = self.fusion(x)
        logits = self.classifier(f)
        return logits
```

Listing 1: 多模态神经网络模型实现

A.2.2 主程序入口 (main.py)

```python
import torch
from data_preprocessing import load_all_data
from model import NextGridPredictor
from train import train_model
from evaluate import evaluate_model

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

train_csv = 'data/train_data.csv'
test_csv = 'data/test_data.csv'
grid_csv = 'data/grid_coordinates-2.csv'
if __name__ == '__main__':
    print("加载并处理数据...")
    train_set, val_set, test_set, num_classes, grid2idx = load_all_data(
        train_csv, test_csv, grid_csv, val_size=0.2
    )

    print(f"训练样本数: {len(train_set)}, 验证样本数: {len(val_set)}, 测试样本数: {len(
    test_set)}")
    model = NextGridPredictor(num_classes=num_classes)

    print("开始训练...")
    model = train_model(model, train_set, val_set, device, num_epochs=40, batch_size=32,
     lr=1e-3, patience=5)

    print("在测试集上评估...")
    acc_k, mrr = evaluate_model(model, test_set, device)
    print(f"Test_MRR: {mrr:.4f}")
    for k in [1, 5, 10]:
        print(f"Test_Acc@{k}: {acc_k[k]:.3f}")
```

Listing 2: 主程序实现

A.2.3 训练逻辑实现 (train.py)

```python
import torch
import torch.nn as nn
import torch.optim as optim
import numpy as np
from evaluate import evaluate_model

def batch_iter(samples, batch_size=32, shuffle=True):
    idxs = np.arange(len(samples))
    if shuffle:
        np.random.shuffle(idxs)
    for i in range(0, len(samples), batch_size):
        batch = [samples[j] for j in idxs[i:i+batch_size]]
        maxlen = max(len(x[0]) for x in batch)
        # pad to maxlen
        seq_ids = np.zeros((len(batch), maxlen), dtype=np.int64)
        seq_coords = np.zeros((len(batch), maxlen, 2), dtype=np.float32)
        seq_poi = np.zeros((len(batch), maxlen, 10), dtype=np.float32)
        for i, (pidx, pcoord, ppoi, _) in enumerate(batch):
            L = len(pidx)
            seq_ids[i, -L:] = pidx
            seq_coords[i, -L:, :] = pcoord
            seq_poi[i, -L:, :] = ppoi
        targets = np.array([x[3] for x in batch], dtype=np.int64)
        yield (
            torch.from_numpy(seq_ids),
            torch.from_numpy(seq_coords),
            torch.from_numpy(seq_poi),
            torch.from_numpy(targets)
        )

def train_model(model, train_set, val_set, device, num_epochs=40, batch_size=32, lr=1e-3, patience=5):
    model = model.to(device)
    opt = optim.Adam(model.parameters(), lr=lr)
    criterion = nn.CrossEntropyLoss()
    best_mrr = -1
    best_state = None
    no_improve = 0
    for epoch in range(1, num_epochs+1):
        model.train()
        losses = []
        for seq_ids, seq_coords, seq_poi, targets in batch_iter(train_set, batch_size):
            seq_ids, seq_coords, seq_poi, targets = (
                seq_ids.to(device), seq_coords.to(device), seq_poi.to(device), targets.to(device)
            )
            opt.zero_grad()
            logits = model(seq_ids, seq_coords, seq_poi)
            loss = criterion(logits, targets)
            loss.backward()
            opt.step()
            losses.append(loss.item())
        val_acc_k, val_mrr = evaluate_model(model, val_set, device)
        print(f"Epoch {epoch} | loss={np.mean(losses):.4f} | Val_MRR={val_mrr:.4f} | Acc@1={val_acc_k[1]:.3f} Acc@5={val_acc_k[5]:.3f} Acc@10={val_acc_k[10]:.3f}")
        if val_mrr > best_mrr:
            best_mrr = val_mrr
            best_state = model.state_dict()
            no_improve = 0
        else:
            no_improve += 1
            if no_improve >= patience:
                print("Early stop triggered.")
                break
    if best_state is not None:
        model.load_state_dict(best_state)
    return model
```

Listing 3: 模型训练实现

A.2.4 数据预处理模块 (data_preprocessing.py)

```python
import pandas as pd
import numpy as np
import ast
from sklearn.model_selection import train_test_split

def parse_list(s):
    try:
        return ast.literal_eval(s)
    except Exception:
        return []

def load_grid_info(grid_csv_path):
    grid_df = pd.read_csv(grid_csv_path, encoding='gbk')
    coords_map = {}
    poi_feat_map = {}
    poi_columns = ['医疗', '住宿', '摩托', '体育', '餐饮', '公司', '购物', '生活', '科教', '汽车']
    for _, row in grid_df.iterrows():
        gid = int(row["grid_id"])
        x = (row["grid_lon_min"] + row["grid_lon_max"]) / 2.0
        y = (row["grid_lat_min"] + row["grid_lat_max"]) / 2.0
        coords_map[gid] = (x, y)
        poi_feat_map[gid] = row[poi_columns].values.astype(float)
    # 归一化空间和poi
    xs, ys = zip(*coords_map.values())
    x_min, x_max = min(xs), max(xs)
    y_min, y_max = min(ys), max(ys)
    for gid in coords_map:
        x, y = coords_map[gid]
        coords_map[gid] = [(x - x_min) / (x_max - x_min + 1e-8), (y - y_min) / (y_max - y_min + 1e-8)]
    all_poi = np.stack(list(poi_feat_map.values()))
    poi_min, poi_max = all_poi.min(axis=0), all_poi.max(axis=0)
    for gid in poi_feat_map:
        poi_feat_map[gid] = (poi_feat_map[gid] - poi_min) / (poi_max - poi_min + 1e-8)
    return coords_map, poi_feat_map

def sort_by_density(gid_list, coords_map):
    if len(gid_list) <= 1: return gid_list[:]
    k = 3
    locs = [coords_map[g] for g in gid_list]
    scores = []
    for i, g in enumerate(gid_list):
        xi, yi = locs[i]
        dists = [np.linalg.norm([xi-xj, yi-yj]) for j, (xj, yj) in enumerate(locs) if i != j]
        avg = np.mean(sorted(dists)[:min(k, len(dists))]) if dists else 1e5
        scores.append((avg, g))
    scores.sort()
    return [g for _, g in scores]

def make_samples(data_csv_path, coords_map, poi_feat_map, grid2idx, max_seq_len=10):
    data_df = pd.read_csv(data_csv_path)
    brand_samples = []
    for _, row in data_df.iterrows():
        brand = row['brand_name']
        gid_list = parse_list(row['grid_id_list'])
        seq = sort_by_density(gid_list, coords_map)
        if len(seq) < 2: continue
        for l in range(1, len(seq)):
            prefix = seq[:l]
            target = seq[l]
            if len(prefix) > max_seq_len:
                prefix = prefix[-max_seq_len:]
            prefix_idx = [grid2idx[g] for g in prefix]
            prefix_coords = [coords_map[g] for g in prefix]
            prefix_poi = [poi_feat_map[g] for g in prefix]
            target_idx = grid2idx[target]
            brand_samples.append((prefix_idx, prefix_coords, prefix_poi, target_idx))
    return brand_samples

def load_all_data(train_csv, test_csv, grid_csv, val_size=0.2):
    coords_map, poi_feat_map = load_grid_info(grid_csv)
    # 构造全网格字典
    all_grids = set()
    for csvf in [train_csv, test_csv]:
        df = pd.read_csv(csvf)
        for _, row in df.iterrows():
            all_grids.update(parse_list(row['grid_id_list']))
    grid2idx = {gid: idx for idx, gid in enumerate(sorted(all_grids))}
    num_classes = len(grid2idx)
    train_samples = make_samples(train_csv, coords_map, poi_feat_map, grid2idx)
    test_samples = make_samples(test_csv, coords_map, poi_feat_map, grid2idx)
    # 再在train_samples中拆分出val
    train_idx, val_idx = train_test_split(np.arange(len(train_samples)), test_size=val_size, random_state=42)
    train_set = [train_samples[i] for i in train_idx]
    val_set = [train_samples[i] for i in val_idx]
    return train_set, val_set, test_samples, num_classes, grid2idx
```

Listing 4: 数据预处理实现

A.2.5 数据特征增强 (测试数据文件.py)

```python
import pandas as pd
import ast
from collections import defaultdict

# 读取 train_data.csv 文件
df_train = pd.read_csv('data/train_data.csv')

# 初始化一个默认字典来存储 grid_id 和 brand_type 的计数
grid_brand_counts = defaultdict(lambda: defaultdict(int))

# 已知的 brand_type 前两个字符集合
brand_types = {'住宿', '摩托', '公司', '餐饮', '体育', '购物', '生活', '汽车', '医疗', '科教'}

# 遍历 train_data.csv 的每一行
for _, row in df_train.iterrows():
    brand = row['brand_type'][:2]  # 取 brand_type 的前两个字符
    # 将 grid_id_list 从字符串解析为列表
    grid_ids = ast.literal_eval(row['grid_id_list'])

    # 为每个 grid_id 统计 brand_type 的数量
    for grid_id in grid_ids:
        grid_brand_counts[grid_id][brand] += 1

df_grid = pd.read_csv('data/grid_coordinates.csv')

# 为每个 brand_type 添加一列，初始化为 0
for brand in brand_types:
    df_grid[brand] = 0

# 更新 df_grid 中每个 grid_id 对应的 brand_type 数量
for grid_id in grid_brand_counts:
    # 确保 grid_id 存在于 df_grid 中
    if grid_id in df_grid['grid_id'].values:
        for brand, count in grid_brand_counts[grid_id].items():
            df_grid.loc[df_grid['grid_id'] == grid_id, brand] = count

# 保存更新后的表格到新的 Excel 文件
df_grid.to_csv(
    "data/grid_coordinates-2.csv", encoding='gbk'
)
```

Listing 5: 数据特征增强实现

A.2.6 模型评估模块 (evaluate.py)

```python
import torch
import numpy as np

@torch.no_grad()
def evaluate_model(model, dataset, device, k_list=[1, 5, 10]):
    model.eval()
    acc_k = {k: 0 for k in k_list}
    mrr_sum = 0
    total = 0
    for seq_ids, seq_coords, seq_poi, targets in batcher(dataset, 64):
        seq_ids, seq_coords, seq_poi, targets = (
            seq_ids.to(device), seq_coords.to(device), seq_poi.to(device), targets.to(device)
        )
        logits = model(seq_ids, seq_coords, seq_poi)
        topk = torch.topk(logits, max(k_list), dim=1).indices.cpu().numpy()
        targets_np = targets.cpu().numpy()
        for i, target in enumerate(targets_np):
            rank = np.where(topk[i] == target)[0]
            if len(rank) > 0:
                rank = rank[0] + 1
                mrr_sum += 1.0 / rank
                for k in k_list:
                    if rank <= k:
                        acc_k[k] += 1
            total += 1
    mrr = mrr_sum / total if total else 0
    acc_k = {k: acc_k[k]/total for k in k_list}
    return acc_k, mrr

def batcher(samples, batch_size=64):
    for i in range(0, len(samples), batch_size):
        batch = samples[i:i+batch_size]
        maxlen = max(len(x[0]) for x in batch)
        seq_ids = np.zeros((len(batch), maxlen), dtype=np.int64)
        seq_coords = np.zeros((len(batch), maxlen, 2), dtype=np.float32)
        seq_poi = np.zeros((len(batch), maxlen, 10), dtype=np.float32)
        for j, (pidx, pcoord, ppoi, _) in enumerate(batch):
            L = len(pidx)
            seq_ids[j, -L:] = pidx
            seq_coords[j, -L:, :] = pcoord
            seq_poi[j, -L:, :] = ppoi
        targets = np.array([x[3] for x in batch], dtype=np.int64)
        yield (
            torch.from_numpy(seq_ids),
            torch.from_numpy(seq_coords),
            torch.from_numpy(seq_poi),
            torch.from_numpy(targets)
        )
```

Listing 6: 模型评估实现

A.3 数据样本展示

A.3.1 训练数据格式

以下直接展示训练数据文件的前 10 行内容：

```
brand_name,brand_type,longitude_list,latitude_list,grid_id_list
四季啤酒屋,体育休闲服务;娱乐场所;酒吧,"[116.928274525344, 117.043320008084, 116.9422963244636, 117.0778221078006]","[36.69774403270743,
       36.69430636057923, 36.69940378336072, 36.65480559942927]","[71, 77, 72, 25]"
携程旅游,生活服务;生活服务场所;生活服务场所,"[117.0099652551294, 117.0083278100292, 116.9029874347601, 117.1258233788835, 117.1318264911793,
       117.130692205233, 117.0276045368049, 116.9643505011395, 117.0700358356303, 116.9762270603227, 117.0415532841417]","[36.61515271727929,
       36.64534980906663, 36.67805564190155, 36.65345336291342, 36.66021335891504, 36.676643613408, 36.65479701387262, 36.65681312305596,
       36.64246444489299, 36.61303461884545, 36.68037991342308]","[4, 21, 51, 28, 45, 64, 22, 19, 25, 3, 59]"
幸福时光KTV,体育休闲服务;娱乐场所;KTV,"[116.9786056408031, 116.9167728180969, 117.0098096139525, 116.9590453043534,
       117.0495090946829]","[36.66161180954602, 36.65290255903322, 36.6710547488655, 36.67043566438539, 36.66060909605174]","[37, 16, 38, 36,
       41]"
小电,生活服务;共享设备;充电宝,"[116.946966962845, 117.0165404020267, 117.0442775933112, 117.0251578749105, 117.0251360192659,
       117.0703663502994, 116.9679460276465, 117.1241885471427, 117.0454510621802, 116.9908858098853, 117.0164047908445, 117.0595249194539,
       117.1492552466137, 117.0387305279752, 117.0026990920988, 116.9833750721236, 117.0259239931365, 117.1239399242847, 116.9688828178708,
       117.0047431014325, 116.9807610358491, 117.0817309886335]","[36.70479258447448, 36.66309812601193, 36.64531065644391, 36.66396584217707,
        36.66344022296101, 36.67629346501127, 36.72221597591807, 36.65502138143246, 36.67928044322549, 36.66917117386832, 36.65574600192463,
       36.68491232846421, 36.70537523707548, 36.65513173060003, 36.61779391991816, 36.68115049015188, 36.66329622415145, 36.69306744935307,
       36.6628680099412, 36.61928189507721, 36.66546105430051, 36.70016712172751]","[72, 39, 23, 39, 39, 61, 89, 28, 59, 37, 22, 60, 83, 23,
       4, 56, 39, 64, 36, 4, 37, 79]"
康明眼镜,购物服务;专卖店;眼镜店,"[117.1474925587893, 116.9899725033711, 117.0496668536213, 117.0009462913962, 116.9162896279549,
       117.1410070518251]","[36.72060927039779, 36.69860146269658, 36.67619517995269, 36.67480000224907, 36.65303092587671,
       36.73127828863626]","[98, 74, 60, 38, 16, 105]"
中财管道,购物服务;家居建材市场;建材五金市场,"[117.0334533368044, 116.9703254004627, 116.9880174777508, 116.9558548711589, 116.992987253108,
       116.9364190460066, 117.143078283607, 116.9750794367394, 117.0050266022032, 117.0778553647915, 117.031448155798, 117.0102340663978,
       116.9578983721978, 117.0242489307601, 117.039477183328, 117.0215823229305, 116.9612181682283, 116.9550518132942, 116.9431431740838,
       116.9606835793541, 117.0478167565158]","[36.68642073855862, 36.7131882069229, 36.66762686110791, 36.71761468549576, 36.68152706900885,
       36.68485510759297, 36.7221657010541, 36.716158675651, 36.70174365839906, 36.71288009941338, 36.70789597294158, 36.69540520391379,
       36.69043405380769, 36.69456713613525, 36.68870724027503, 36.71320839471499, 36.70611753046003, 36.66261143837973, 36.71354397295313,
       36.69362291074891, 36.70676831386894]","[59, 89, 37, 88, 56, 53, 98, 89, 75, 94, 77, 75, 55, 76, 59, 91, 73, 35, 88, 73, 78]"
美利达自行车,购物服务;专卖店;自行车专卖店,"[116.9524783955336, 116.9154449123776, 117.0606450771626, 117.2299536114407, 116.9908572113357,
       117.0750930872932, 117.2280146835699, 117.1371797256954, 116.956813342363, 117.038657173431]","[36.67667890606948, 36.65417998169556,
       36.73888578931914, 36.68444825883368, 36.65774427341513, 36.68036777295877, 36.72384491010683, 36.65061320002995, 36.64512170736571,
       36.67791786120331]","[54, 16, 102, 68, 37, 61, 101, 29, 18, 59]"
酒快到,购物服务;专卖店;烟酒专卖店,"[117.1316969616346, 117.3016293700698, 117.0343191810293, 116.9759775287286,
       117.101368983480 4]","[36.65471352253594, 36.67943315359312, 36.67821879365496, 36.64535937215268, 36.685721602042]","[28, 70, 59, 20,
       63]"
望京烧烤,餐饮服务;中餐厅;特色/地方风味餐厅,"[116.9464233558445, 117.0335236445724, 117.0455703904242, 117.218120082778, 116.9148782739088,
       117.1755609416811]","[36.61671256380603, 36.68151265323061, 36.71683623281009, 36.70442907328825, 36.67787789073074,
       36.67421657284811]","[1, 59, 92, 85, 52, 48]"
```

Listing 7: 训练数据样本 (train_data.csv)

A.3.2 网格坐标映射

以下直接展示网格坐标映射文件的前 10 行内容：

```
,grid_id,grid_lon_min,grid_lat_min,grid_lon_max,grid_lat_max,   ,¹º  ,²   ,¿  ,Ħ  ,¹«  ,  »  ,  ³µ,    ,
0,1,116.8315,36.71129734099892,116.84946622349985,36.72926356449874,11,47,63,4,0,1,19,3,1,7
1,2,116.8315,36.72926356449874,116.84946622349985,36.74722978799856,5,15,12,0,1,0,8,3,0,0
2,3,116.84946622349985,36.6035,116.86743244699966,36.621466223499816,1,25,25,2,0,0,13,5,1,0
3,4,116.84946622349985,36.621466223499816,116.86743244699966,36.63943244699964,15,76,71,7,3,0,36,3,3,5
4,5,116.84946622349985,36.63943244699964,116.86743244699966,36.657398670499454,6,11,11,3,0,0,10,2,3,0
5,6,116.90336489399928,36.6035,116.92133111749912,36.621466223499816,10,26,34,2,0,0,10,0,2,0
6,7,116.90336489399928,36.63943244699964,116.92133111749912,36.657398670499454,3,19,16,2,0,0,6,3,0,0
7,8,116.90336489399928,36.65739867049946,116.92133111749912,36.67536489399928,16,43,28,1,0,0,14,3,1,1
8,9,116.90336489399928,36.67536489399928,116.92133111749912,36.6933311174991,16,43,34,4,1,0,26,1,2,0
```

Listing 8: 网格坐标映射数据 (grid_coordinates-2.csv)

A.3.3 测试数据格式

以下直接展示测试数据文件的前 10 行内容：

```
brand_name,brand_type,longitude_list,latitude_list,grid_id_list
婴贝儿,购物服务;专卖店;儿童用品店,"[117.0996712293962, 117.1379846421835, 116.8932622025578, 117.1499975247358, 117.024877028235,
       116.9530623680675, 116.9175268352235, 116.9058644662969, 116.9726887822101, 116.986423264485, 117.1984933964415, 117.0789651487876,
       117.0068100044073, 117.0645120634065, 117.1805712919167, 117.1291492015525, 117.1229203417397, 116.9822128030304, 116.980556733302,
       117.0470645130879, 116.969951463457, 116.966883782526, 116.932661722804, 116.8956431338762, 117.0294847274234, 117.0925924533216,
       117.1404898036926, 117.067678452798, 116.9632200208925, 117.2130256724302, 117.114841357525, 116.9589876072608, 116.9112198636204,
       117.0535553747456, 116.976233074227, 117.1481918016937, 117.0755583820197, 117.0493036526866, 117.0647861401144, 116.9786000817559,
       117.0318934269682, 116.9474375020692, 117.3023863434853, 116.9989403856787, 116.9286838764815, 117.103153779245, 117.212533931024,
       116.9490482542897, 116.9335158542075, 117.0912068977715, 116.8952493698547, 116.8937014397158, 117.0012938718576, 117.1348022717404,
       116.9934381463388, 116.9248374912076, 117.0338667428631, 117.1245142410434, 116.959728614966, 117.0789611631621, 116.9819612318406,
       116.9761256282631, 117.0663021496076, 116.9598322117709, 116.9543295013623, 117.0760248289039, 116.9373115120875, 116.972206984709,
       116.9759941866322, 117.139236644202, 116.9820209791802, 117.122275277546, 117.006788341595]","[36.68518634034514, 36.65259926084561,
       36.64789102488911, 36.65282137701404, 36.70738025265271, 36.62171064249292, 36.67760566255552, 36.68758359058992, 36.64926352641689,
       36.69574507118392, 36.68805065032451, 36.64994782878281, 36.6923003368949, 36.69080087192778, 36.71362065379788, 36.68683109709343,
       36.68971435585137, 36.68355946553892, 36.65804020687233, 36.71975535800927, 36.69023596420799, 36.65427378356909, 36.66076342911936,
       36.67534239720015, 36.69639228882006, 36.72149586387541, 36.65705057132519, 36.7013634286597, 36.68296497652979, 36.69904041988929,
       36.64470063327149, 36.65144342757935, 36.64945811374917, 36.69934662668804, 36.61268080429876, 36.71987996978413, 36.68042138556957,
       36.69581050100167, 36.73720243585072, 36.67586502415248, 36.70002192573417, 36.7040490371283, 36.68028628058879, 36.65370106096641,
       36.63952835597043, 36.69587093347233, 36.6752718706838, 36.67571039371824, 36.65750701651433, 36.68988022221125, 36.68434475948285,
       36.66258215735837, 36.67399944541669, 36.73042609089215, 36.62910253209033, 36.70510200590964, 36.66111545846772, 36.68652928405672,
       36.67158120624317, 36.6498678793942, 36.63545258838949, 36.71097399359922, 36.66674974427514, 36.67170086583557, 36.63431322611724,
       36.68791932711586, 36.69440347151026, 36.67442049817143, 36.71091641427032, 36.72927374196909, 36.6354204286713, 36.68513449089427,
       36.61908088107396]","[62, 29, 15, 29, 76, 8, 52, 52, 19, 74, 66, 25, 57, 60, 99, 64, 64, 56, 37, 92, 55, 19, 34, 32, 77, 95, 29, 79,
       55, 85, 27, 19, 16, 78, 3, 98, 61, 78, 102, 56, 77, 72, 70, 21, 17, 81, 50, 54, 34, 62, 51, 32, 38, 104, 11, 71, 40, 64, 36, 25, 10,
       74, 42, 36, 8, 61, 71, 36, 74, 105, 10, 64, 4]"
易居房友,生活服务;中介机构;中介机构,"[117.0455418760648, 116.9509171744841, 116.9431907340005, 117.1178271095851, 117.0605752418835,
       116.968580827352, 116.9747873447345, 117.1068710799779, 117.2360849347613, 116.9480072307894, 116.8889581106447, 116.9503722410296,
       116.9435205358656, 117.0798758522983, 117.1183239314952, 116.9211595391622, 116.9208517884814, 117.1028050006402]","[36.69592275616423,
        36.62761021751274, 36.64295591436264, 36.63778862425124, 36.70413759514209, 36.68285644843733, 36.64682295253007, 36.64515329401156,
       36.6844896505136, 36.62532311705273, 36.62305080197009, 36.60744311333485, 36.64456028149461, 36.65115957318063, 36.64249232991678,
       36.67326156531449, 36.65803927039991, 36.6938970305318]","[77, 8, 18, 13, 78, 55, 19, 27, 68, 8, 6, 1, 18, 25, 27, 33, 33, 81]"
ONLY,购物服务;服装鞋帽皮具店;服装鞋帽皮具店,"[117.0329723390454, 117.126770055059, 117.0069781402267, 116.9158825102825, 116.9680673165032,
       117.1030590123878, 117.0661138170195, 116.9724189265898, 116.9160061067833, 117.0548685526778, 117.0091332642082,
       116.9699238877644]","[36.65771270947515, 36.71707256556891, 36.68470863129098, 36.71590579384737, 36.660916472151, 36.66783446422561,
       36.6729341401392, 36.64897846977063, 36.65301619076054, 36.70229353395882, 36.64640278878978, 36.66027968043058]","[40, 97, 57, 86, 36,
        44, 42, 19, 16, 78, 21, 36]"
马博士婴幼儿游泳馆,生活服务;婴儿服务场所;婴儿游泳馆,"[117.0329544373787, 117.152660325078, 117.1310958460403, 117.0015420162948,
       116.9766451102195, 116.976173324498, 117.0969491625688, 117.1273725892052, 117.0574113167814, 116.903516194149, 117.2276270738628,
       117.2183742367875]","[36.65741593403924, 36.6507404109261, 36.67535198563168, 36.66312840564837, 36.63285286296802, 36.6126849763032,
       36.69106492160663, 36.71812366636688, 36.68266191994277, 36.67560000226136, 36.69304155157505, 36.66937440180349]","[40, 29, 45, 38,
       10, 3, 62, 97, 60, 52, 68, 50]"
东润大药房,医疗保健服务;医药保健销售店;药房,"[116.9189725373848, 117.1798187085277, 117.3004779145081, 117.056191930025, 117.178542302356,
       117.0920242146625, 117.0450441177042, 116.9223930707534, 117.1007288455053, 116.9511416845087, 117.0634469596785, 117.0836888352784,
       116.895060399059, 116.9420764131831, 117.0378625217185, 117.1446505331589]","[36.64703294263889, 36.64582666385298, 36.69132468468406,
       36.69534881889702, 36.69994979724778, 36.70030056941857, 36.71594188138783, 36.67330211662661, 36.72198908101953, 36.66159833949235,
       36.7374749427703, 36.68309610665886, 36.64780522367574, 36.69557866189321, 36.6809817531441, 36.72418103650305]","[16, 31, 70, 78, 84,
        80, 92, 34, 95, 35, 102, 62, 15, 72, 59, 98]"
巴拉巴拉,购物服务;专卖店;儿童用品店,"[116.9184409683297, 117.0487200623952, 117.2265239832789, 116.9699318493467, 116.9162878357841,
       116.9685862509238, 117.1232751782797, 116.9765365659153]","[36.7166918557168, 36.71575676708599, 36.69076631970337, 36.66033462447786,
       36.71584047134091, 36.66072115060988, 36.68534053512016, 36.63285917669518]","[86, 93, 67, 36, 86, 36, 64, 10]"
绿能电动车,购物服务;专卖店;专营店,"[116.9553366474382, 117.0542860517775, 117.137581781354, 117.1150193252241, 117.0674438409209,
       117.1809399707232, 116.9914004966939, 117.1079942801645, 117.0116639683536, 117.0997243007297]","[36.67300333474505, 36.71129970238661,
        36.73335742719549, 36.68928933212071, 36.71061673340619, 36.72203683502129, 36.66977323153279, 36.6977984063399, 36.62508696696515,
       36.69316031721775]","[35, 93, 105, 63, 79, 99, 37, 81, 12, 62]"
杰克琼斯,购物服务;服装鞋帽皮具店;品牌服装店,"[117.0939859999953, 117.1030660151984, 117.1919618382704, 117.122413351018, 117.0581444822647,
       117.2263799631202, 117.127057273006, 116.9530586022749, 117.0165067544813, 117.0168819348173, 116.9161997937927]","[36.65882156309365,
       36.66781148160168, 36.6712140470736, 36.68558436801666, 36.68288265791443, 36.69050839174971, 36.717877414642, 36.61969082307267,
       36.66120438214303, 36.66368279661755, 36.65290008657293]","[43, 44, 49, 64, 60, 67, 97, 1, 39, 39, 16]"
立聪堂助听器,医疗保健服务;医药保健销售店;医疗保健用品,"[117.0743981567748, 116.9504006346039, 117.0300145426286, 117.0138111133497,
       116.9950914008877, 117.0004169856987]","[36.68034533367015, 36.64870287347598, 36.65435010935824, 36.65936424237968, 36.62487595645675,
        36.64456707602362]","[61, 18, 23, 39, 11, 21]"
```

Listing 9: 测试数据样本 (test_data.csv)

A.4 完整源代码文件结构

项目包含以下主要文件：

- `src/model.py` - 神经网络模型定义
- `src/main.py` - 主程序入口
- `src/train.py` - 训练逻辑实现
- `src/evaluate.py` - 模型评估
- `src/data_preprocessing.py` - 数据预处理

- `data/train_data.csv` - 训练数据
- `data/test_data.csv` - 测试数据
- `data/grid_coordinates-2.csv` - 网格坐标映射

详细的代码实现和更多技术细节请参考 GitHub 仓库:https://github.com/szw0407/DL-project-2025