





Time Series Library (TSLib)

《 Deep Time Series Models: A Comprehensive Survey and Benchmark 》

2024/9/22





I Long-term Forecasting



I Long-term Forecasting

数据集	数据集简介	数据输入特征数	样本长度	时间粒度
ETTh1,ETTh2	2个电力变压器的一小时级别粒度数据	7	17420	1 hour
ETTm1,ETTm2	2个电力变压器的15分钟级别粒度数据	7	69680	15 mins
Electricity	321个客户的用电量数据	321	26304	1 hour
Weather	Jena Climate时间序列数据集中基于多变量的历史气象数据	21	52696	10 mins
Traffic	旧金山高速公路传感器记录的交通统计数据	862	17544	1 hour
Exchange	1990年至2016年8个国家的每日汇率数据	8	7588	1 day
ILI	美国疾病控制和预防中心每周流感统计数据	7	966	1 week

	A	B	C	D	E	F	G	H	I
1	date	澳大利亚汇率0	英国汇率 1	加拿大汇率 2	瑞士汇率 3	中国汇率 4	日本汇率 5	新西兰汇率6	OT新加坡汇率
2	1990/1/1 0:00	0.7855	1.611	0.861698	0.634196	0.211242	0.006838	0.525486	0.593
3	1990/1/2 0:00	0.7818	1.61	0.861104	0.633513	0.211242	0.006863	0.523972	0.594
4	1990/1/3 0:00	0.7867	1.6293	0.86103	0.648508	0.211242	0.006975	0.526316	0.5973
5	1990/1/4 0:00	0.786	1.637	0.862069	0.650618	0.211242	0.006953	0.523834	0.597
6	1990/1/5 0:00	0.7849	1.653	0.861995	0.656254	0.211242	0.00694	0.527426	0.5985
7	1990/1/6 0:00	0.7866	1.6537	0.86103	0.654879	0.211242	0.006887	0.526177	0.604
8	1990/1/7 0:00	0.7886	1.662	0.862887	0.661157	0.211242	0.006885	0.527565	0.607

exchange_rate数据集

收集了1990年至2016年，8个国家的每日汇率值数据

I Long-term Forecasting

ETTh1.csv数据集文件:

- 8维特征
- 1小时级
- 两年数据
- .csv格式

	A	B	C	D	E	F	G	H
1	date	HUFL	HULL	MUFL	MULL	LUFL	LULL	OT
2	2016/7/1 0:00	5.827	2.009	1.599	0.462	4.203	1.34	30.531
3	2016/7/1 1:00	5.693	2.076	1.492	0.426	4.142	1.371	27.787
4	2016/7/1 2:00	5.157	1.741	1.279	0.355	3.777	1.218	27.787
5	2016/7/1 3:00	5.09	1.942	1.279	0.391	3.807	1.279	25.044
6	2016/7/1 4:00	5.358	1.942	1.492	0.462	3.868	1.279	21.948
7	2016/7/1 5:00	5.626	2.143	1.528	0.533	4.051	1.371	21.174
8	2016/7/1 6:00	7.167	2.947	2.132	0.782	5.026	1.858	22.792
9	2016/7/1 7:00	7.435	3.282	2.31	1.031	5.087	2.224	23.144
10	2016/7/1 8:00	5.559	3.014	2.452	1.173	2.955	1.432	21.667
11	2016/7/1 9:00	4.555	2.545	1.919	0.817	2.68	1.371	17.446

数据集各列描述:

date	HUFL	HULL	MUFL	MULL	LUFL	LULL	OT
记录日期	高有用负载	高无用负载	中等有用负载	中等无用负载	低有用负载	低无用负载	油温度 (预测目标值)

I Long-term Forecasting

Dataset_ETT_hour类:

```
run.py × exp_long_term_forecasting.py × TimesNet.py × data_factory.py × data_loader.py × TimesNet_ETTh1.sh ×
19 # 读取ETTh1、ETTh2数据集类
    3个用法
20 class Dataset_ETT_hour(Dataset):
21     # 预测任务, features选项: [M, S, MS]; M: 多元变量预测多元变量, S: 单变量预测单变量, MS: 多元变量预测单变量
22     # target: 目标变量的名称、scale: 是否对数据进行标准化
23     # timeenc: 时间特征编码方式, 0 表示简单编码, 1 表示时间特征提取、freq: 时间序列的频率、seasonal_patterns: 季节性模式
24     def __init__(self, args, root_path, flag='train', size=None,
25                 features='S', data_path='ETTh1.csv',
26                 target='OT', scale=True, timeenc=0, freq='h', seasonal_patterns=None):
27         # size [seq_len, label_len, pred_len]
28         self.args = args
```

数据集构成:

```
# 根据特征, 选择不同的数据列
if self.features == 'M' or self.features == 'MS':
    # 多特征: 排除日期列
    cols_data = df_raw.columns[1:]
    df_data = df_raw[cols_data]
elif self.features == 'S':
    # 单特征: 仅选择目标列
    df_data = df_raw[[self.target]]
```

长时预测示例脚本:

```
model_name=TimesNet
python -u run.py \
--task_name long_term_forecast \
--is_training 1 \
--root_path ./dataset/ETT-small/ \
--data_path ETTh1.csv \
--model_id ETTh1_96_96 \
--model $model_name \
--data ETTh1 \
--features M \
--seq_len 96 \
--label_len 48 \
--pred_len 96 \
--e_layers 2 \
--d_layers 1 \
--factor 3 \
--enc_in 7 \
--dec_in 7 \
--c_out 7 \
--d_model 16 \
--d_ff 32 \
--des 'Exp' \
--itr 1 \
--top_k 5
```

模型预测输出:

```
~~~~~preds~~~~~
(2785, 96, 7)
[[[-0.10912043  0.9184426  0.07979733 ... -0.4020451  0.5361992
   -0.72009236]
 [ 0.09899503  0.9421642  0.23975384 ... -0.27258646  0.6385776
   -0.72398686]
 [ 0.32184052  0.9100838  0.33293867 ... -0.20767187  0.6255687
   -0.72224313]
 ...
 [ 0.27041888  0.1737634  0.20396459 ... -0.07382867  0.35848248
   -0.72356147]
 [ 0.19573367  0.36036006 -0.17921153 ... -0.16663224  0.37533462
   -0.7247473 ]
 [ 0.42512244  0.6815027  0.24572605 ... -0.1927617  0.51490444
   -0.7554367 ]]
```



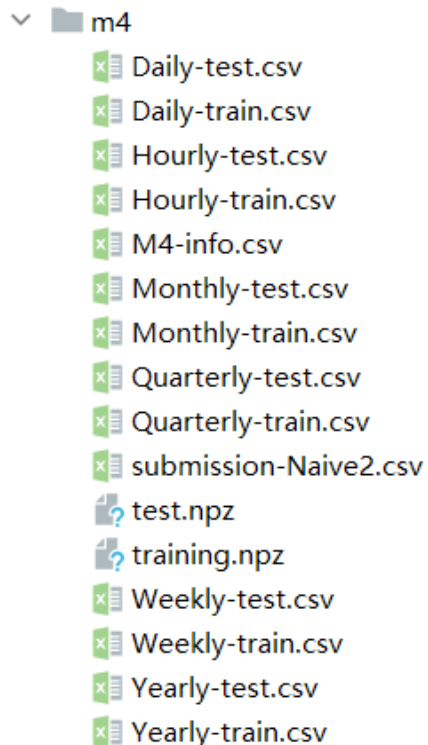
II Short-term Forecasting



II Short-term Forecasting

M4数据集:

- 六种采样频率
- 训练集+测试集
- .csv格式
- 单变量时间序列预测
- 历史观测值长度不定



数据集	输入特征数	预测长度
M4-Yearly	1	6
M4-Quarterly	1	8
M4-Monthly	1	18
M4-Weakly	1	13
M4-Daily	1	14
M4-Hourly	1	48

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22
2	Y1	5172.1	5133.5	5186.9	5084.6	5182	5414.3	5576.2	5752.9	5955.2	6087.8	6238.9	6317.2	6262.7	6361	6427.4	6654.9	6835.4	6925.5	7073.5	7144	7230.6
3	Y2	2070	2104	2394	1651	1492	1348	1198	1192	1105	1008	994	1420	1582	1286	1133	1322	1157	1244	1843	1794	1426
4	Y3	2760	2980	3200	3450	3670	3850	4000	4160	4290	4530	4720	4890	5070	5220	5420	5660	5820	5940	6070	6200	6520
5	Y4	3380	3670	3960	4190	4440	4700	4890	5060	5200	5490	5690	5910	6120	6340	6530	6710	6850	6980	7130	7320	7630
6	Y5	1980	2030	2220	2530	2610	2720	2970	2980	3100	3230	3340	3430	3810	3780	3930	3840	3890	3840	3930	4050	4040
7	Y6	1988.65	2073.59	2144.04	2147.85	2180.73	2240.89	2311.21	2392.71	2470.71	2572.71	2718.48	2778.76	2903.94	2918.9	2977.39	3062.05	3151.34	3361.47	4009.21		
8	Y7	2158.96	2216.71	2252.75	2217.93	2215.01	2235.28	2282	2328.25	2343.31	2405.12	2502.4	2521.06	2604.74	2620.9	2640.29	2698.93	2748.12	2912.29	3669.46		
9	Y8	2164.71	2169.77	2240.05	2294.09	2327.14	2351.64	2368.54	2364.42	2344.55	2347.72	2419.65	2396.56	2374.46	2371.94	2367.11	2358.06	2343.01	2347.41	2400.96		

II Short-term Forecasting

年频率预测数据:

▼

m4_results

▼

TimesNet

📄

Daily_forecast.csv

📄

Hourly_forecast.csv

📄

Monthly_forecast.csv

📄

Quarterly_forecast.csv

📄

Weekly_forecast.csv

📄

Yearly_forecast.csv

223

224

225

226

227

228

229

230

OS.makedirs

将预测结果

forecasts_

forecasts_

forecasts_

forecasts_

forecasts_

ExcelReader

Yearly_forecast.csv

Q

↻

Cc

W

*

V1	V2	V3	V4	V5	V6
7290.556...	7231.8667	7394.7686	7399.8164	7380.5728	7390.552
1627.583...	1566.2559	1357.9553	1529.8916	1590.653	1704.0415
8657.404...	8845.364	9027.532	9271.184	9447.65	9468.54
9992.932...	10186.751	10393.691	10670.256	10832.951	10843.502
7217.581...	7451.6006	7715.683	8040.0664	8211.755	8247.694
4216.046...	4242.0566	4256.294	4374.5967	4392.4272	4399.2227
3737.940...	3741.0063	3673.734	3797.2637	3829.4106	3840.2524
2406.765...	2413.651	2409.812	2422.3508	2420.8386	2424.4902
2332.002...	2338.3765	2344.0234	2364.7183	2367.4648	2367.355
2321.285...	2323.4514	2290.9465	2327.6528	2339.9053	2339.2395
1499.546	1501.787	1502.4625	1512.8696	1515.0487	1515.6825

中

结果

年频率真实数据:

test.npz

training.npz

Weekly-test.csv

Weekly-train.csv

Yearly-test.csv

Yearly-train.csv

MSL

PEMS-SF

223

224

225

226

227

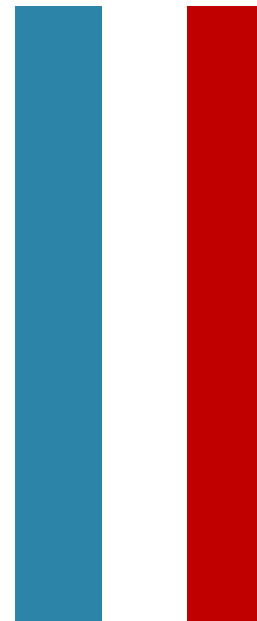
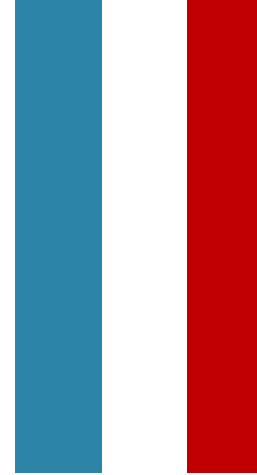
228

229

230

os.makedirs(1000

III Imputation



III Imputation

数据集	数据集简介	数据输入特征数	样本长度	时间粒度
ETTh1,ETTh2	2个电力变压器的一小时级别粒度数据	7	17420	1 hour
ETTm1,ETTm2	2个电力变压器的15分钟级别粒度数据	7	69680	15 mins
Electricity	321个客户的用电量数据	321	26304	1 hour
Weather	Jena Climate时间序列数据集中基于多变量的历史气象数据	21	52696	10 mins

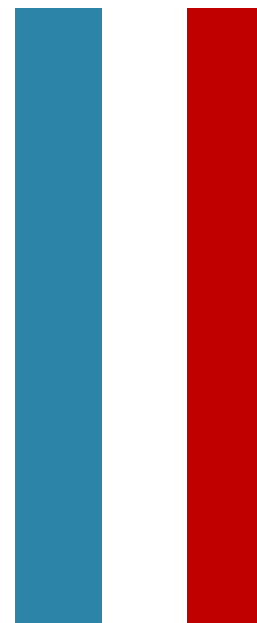
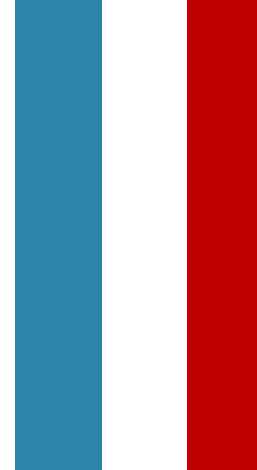
```
class Exp_Imputation(Exp_Basic):
    def train(self, setting):
        # random mask
        # 获取输入数据的形状, B: 16, T: 96, N: 7
        B, T, N = batch_x.shape
        print("~~~~~B、T、N~~~~~")
        print(B, T, N)
        # 生成随机掩码
        mask = torch.rand((B, T, N)).to(self.device)
        # 根据掩码率设置掩码,
        mask[mask <= self.args.mask_rate] = 0 # masked
        mask[mask > self.args.mask_rate] = 1 # remained
        # 对输入数据进行掩码填充, 填充0
        inp = batch_x.masked_fill(mask == 0, 0)
        # 打印输入数据
        print("~~~~~inp~~~~~")
        print(inp)

        # 通过模型进行预测
        outputs = self.model(inp, batch_x_mark, None, None, mask)
```

```
~~~~~B、T、N~~~~~
16 96 7
~~~~~inp~~~~~
tensor([[[[-0.2363,  0.3469, -0.1284, ..., -0.5156, -0.1388,  0.7553],
          [-0.1095,  0.6354,  0.1227, ..., -0.8429, -0.1880,  0.7475],
          [ 0.0000,  1.0200,  0.3739, ..., -1.2005, -0.3815,  0.7093],
          ...,
          [ 0.1669,  0.0000,  0.0776, ...,  0.7047,  1.0211, -0.0190],
          [-0.0634,  0.5392, -0.0833, ...,  0.1693,  0.6816,  0.0000],
          [-0.1326,  0.0000, -0.1092, ..., -0.0994,  0.0000,  0.1803]],

          ~~~~~outputs~~~~~
torch.Size([16, 96, 7])
tensor([[[[ 0.1917,  1.1854,  0.2043, ...,  0.0821,  0.3506,  0.2195],
          [ 0.4900,  0.5696,  0.0346, ..., -0.3302,  0.3620, -0.5090],
          [ 0.4743,  0.5061,  0.4429, ..., -0.2648,  0.3130,  0.0427],
          ...,
          [-0.2734,  1.1172,  0.0734, ..., -0.6112,  0.2719,  0.1856],
          [ 0.0118,  1.1298, -0.1555, ..., -0.4783,  0.0689, -0.2087],
          [ 0.2862,  1.1872,  0.2556, ..., -0.5008,  0.2046,  0.2912]],
```

IV Anomaly Detection

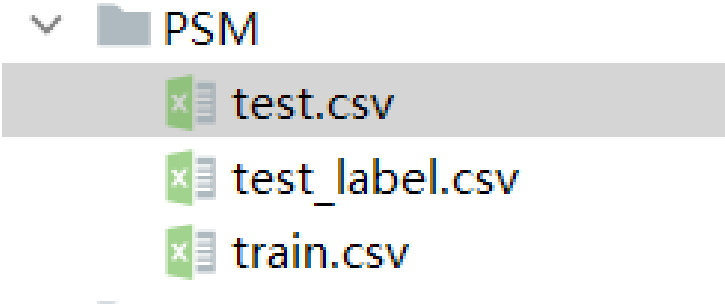


IV Anomaly Detection

数据集	数据集简介	数据输入特征数	序列长度	时间粒度
SMD	服务器机组时序异常数据	38	100	1 min
MSL	包含火星好奇号探测器的异常情况	55	100	1 min
SMAP	NASA航天器系统遥测异常数据	25	100	1 min
SwaT	水处理传感器时序异常数据	51	100	1 min
PSM	服务器节点时序异常数据	25	100	1 min

- PSM数据集:
- 训练集: 132481
 - 验证集: 26398
 - 测试集: 87841
 - 25个特征
 - 0: 无异常、1: 有异常

- PSM数据目录:
- test.csv: 测试集
 - test_label.csv: 测试集标签
 - train.csv: 训练集



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	timestamp_(min)	feature_0	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8	feature_9	feature_1	feature_1	feature_1	feature_1	feature_1	feature_1	feature_1	feature_1	feature_1	feature_1	feature_2	feature_2	feature_2	feature_2	feature_24
2	0	0.73269	0.76175	0.60685	0.48875	0.42431	0.40361	0.51932	0.39879	0.45145	0.44708	0.46334	0.48732	0.15193	0.13846	0.20147	0.3188	0.45186	0.5715	0.46972	0.60988	0.00843	0	0.48184	0.00654	0.13825
3	1	0.7328	0.76185	0.60713	0.48878	0.43201	0.41026	0.51136	0.40257	0.45566	0.44947	0.45927	0.49466	0.15149	0.13801	0.20211	0.32146	0.45612	0.56223	0.46653	0.62981	0.00843	0	0.47722	0.00654	0.11521
4	2	0.73294	0.76159	0.6069	0.48879	0.41886	0.40772	0.48864	0.39653	0.4561	0.45128	0.47159	0.49033	0.15367	0.14076	0.20335	0.34722	0.45669	0.572	0.48785	0.6436	0.00675	0	0.49262	0.00871	0.09217
5	3	0.73289	0.76166	0.60648	0.4888	0.4179	0.40424	0.5	0.40559	0.46002	0.45663	0.47691	0.48086	0.15343	0.14121	0.20135	0.3619	0.46053	0.56335	0.47951	0.64469	0.00843	0	0.45706	0.00871	0.14286
6	4	0.73279	0.76157	0.60678	0.4888	0.4211	0.40741	0.51136	0.39955	0.45851	0.45461	0.45103	0.4588	0.15333	0.13972	0.2031	0.35977	0.45883	0.56335	0.4483	0.62995	0.00675	0	0.47222	0.00654	0.17051
7	5	0.73265	0.7613	0.60782	0.48875	0.40314	0.40203	0.52841	0.38369	0.44968	0.45066	0.4458	0.43763	0.1513	0.13873	0.20305	0.35735	0.45015	0.57777	0.46276	0.62695	0.00675	0	0.47596	0.01307	0.13825
8	6	0.73254	0.76142	0.60759	0.48875	0.42591	0.40836	0.52841	0.41956	0.45338	0.44598	0.46135	0.44798	0.15583	0.14095	0.20206	0.36461	0.45371	0.56661	0.47529	0.6294	0.00675	0.00356	0.48732	0.00871	0.12903
9	7	0.73257	0.76148	0.60864	0.48877	0.42848	0.41532	0.53409	0.41125	0.46479	0.46006	0.46515	0.47246	0.14885	0.13532	0.20218	0.30791	0.46508	0.56724	0.4882	0.63131	0.00843	0.00356	0.4738	0.01089	0.14286
10	8	0.73244	0.7616	0.60888	0.48881	0.42752	0.41089	0.51705	0.4071	0.46406	0.46224	0.46796	0.48128	0.15048	0.1371	0.2003	0.31	0.46451	0.57689	0.49319	0.63636	0.00843	0	0.48132	0.00871	0.14747

IV Anomaly Detection

模型训练阶段

输入结构:

```
~~~~~batch_x~~~~~
torch.Size([128, 100, 25])
tensor([[[[-1.1722,  0.1990, -0.3303, ..., -0.8413, -0.7627, -0.2037],
          [-1.1738,  0.2007, -0.3349, ..., -0.8751, -0.7627, -0.5933],
          [-1.1755,  0.2023, -0.3350, ..., -0.8367, -0.7802, -0.4634],
          ...,
          [-1.0143,  0.2785, -0.3139, ..., -0.4399, -0.7802,  0.4456],
          [-1.0131,  0.2761, -0.3167, ..., -0.4964, -0.7976, -0.1388],
          [-1.0115,  0.2820, -0.3198, ..., -0.7050, -0.7453, -0.8530]],
```

输出结构:

```
~~~~~outputs~~~~~
torch.Size([128, 100, 25])
tensor([[[[-1.1511,  0.2545, -0.3352, ..., -0.9620, -0.7726, -0.4616],
          [-1.0806,  0.2635, -0.3355, ..., -0.8023, -0.7792, -0.4573],
          [-1.0907,  0.2625, -0.3385, ..., -0.7142, -0.7878, -0.6661],
          ...,
          [-1.1455,  0.2586, -0.3297, ..., -0.9209, -0.7518, -0.6857],
          [-1.1216,  0.2487, -0.3297, ..., -1.0084, -0.7526, -0.6415],
          [-1.1057,  0.2491, -0.3323, ..., -0.9575, -0.7379, -0.5738]],
```

模型测试阶段

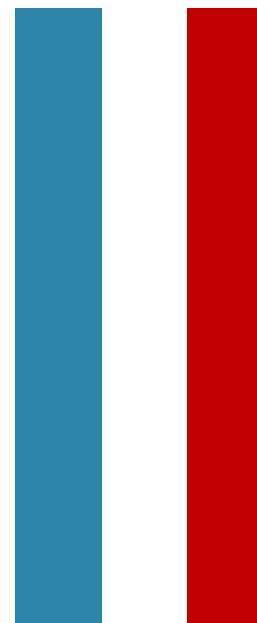
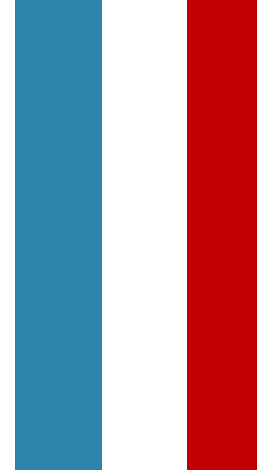
测试集真实标签:

	A	B	C	D
1	timestamp_(min)	label		
2	132480	0		
3	132481	0		
4	132482	0		
5	132483	0		
6	132484	0		
7	132485	0		
8	132486	0		
9	132487	0		
10	132488	0		

测试集预测标签:

```
~~~~~预测标签~~~~~
pred:  [0 0 0 ... 1 1 1]
Accuracy : 0.9803, Precision : 0.9845, Recall : 0.9438, F-score : 0.9637
```

V Classification



V Classification

数据集	数据集简介	输入数据特征数	训练样本长度	分类类别数
EthanolConcentration	乙醇浓度光谱时序数据	3	261个(1751, 3)的特征向量	4
FaceDetection	面部图像时序数据	144	5890个(62, 144)的特征向量	2
Handwriting	手写UCR时序数据	3	150个(152,3)的特征向量	26
Heartbeat	心音记录时序数据	61	204个(405,61)的特征向量	2
JapaneseVowels	UCI档案时序数据	12	270个(29,12)的特征向量	9
PEMS-SF	旧金山湾区高速公路车道占用率数据	963	267个(144,963)的特征向量	7
SelfRegulationSCP1	皮层慢电位时序数据	6	268个(896,6)的特征向量	2
SelfRegulationSCP2	皮层慢电位时序数据	7	200个(1152,7)的特征向量	2
SpokenArabicDigits	声音时序数据	13	6599个(93,13)的特征向量	10
UWaveGestureLibrary	手势运动加速器时序数据	3	120个(315,3)的特征向量	8

V Classification

```
EthanolConcentration_TRAIN.ts x EthanolConcentration_TEST.ts x
21 @problemName EthanolConcentration
22 @timeStamps false
23 @missing false
24 @univariate false
25 @dimensions 3
26 @equalLength true
27 @seriesLength 1751
28 @classLabel true E35 E38 E40 E45
29 @data
30 1724.8,1726.9,1726.8,1725.0,1725.7,1725.9,1
31 1763.6,1759.5,1759.9,1758.8,1757.7,1756.3,1
32 1775.3,1777.0,1778.2,1776.0,1773.0,1769.8,1
33 1747.8,1745.3,1745.0,1745.9,1746.5,1746.8,1
34 1742.1,1742.3,1743.7,1744.9,1745.1,1745.9,1
35 1756.4,1753.2,1754.7,1754.8,1754.2,1753.8,1
36 1771.2,1768.4,1765.5,1764.3,1767.9,1771.1,1
```

EthanolConcentration数据集:

- 训练集样本：261
- 测试集样本：263
- 序列长度：1751
- 数据特征维度：3
- 类标签：E35、E38、E40、E45

模型输入:

```
~~~~~batch_x~~~~~
torch.Size([16, 1751, 3])
tensor([[[ -8.5956,  -9.4597,  -7.0447],
          [ -8.6056,  -9.4652,  -7.0510],
          [ -8.6166,  -9.4709,  -7.0576],
          ...,
          [ -8.5662,  -9.4135,  -7.0015],
          [ -8.5638,  -9.4079,  -6.9965],
          [ -8.5568,  -9.4019,  -6.9903]],
```

模型预测类别索引输出结果:

```

#####predictions#####
(263,)
[[1 0 1 1 1 0 0 1 1 1 1 1 1 0 0 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1
  1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 0 0 1 0 1 0
  1 1 0 0 1 1 1 1 0 0 1 1 1 1 1 0 1 0 1 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1 0 0 0
  1 1 1 1 1 1 0 0 0 1 1 0 1 1 1 0 0 1 1 0 0 1 1 0 1 1 0 1 0 1 1 1 1 0 0 1 1
  0 0 1 1 0 0 1 0 0 1 1 0 1 0 1 1 1 0 0 1 0 1 1 1 0 1 0 1 1 1 1 1 0 0 1 1 1
  1 1 1 1 1 0 0 1 0 0 0 1 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 1 0 1 1 1 1 1 1
  1 1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 0 0 1 1 1 0 1 0
  0 1 0 1]]



```

真实类别标签索引:

```

~~~~~~trues~~~~~
(263,)
[3 0 3 3 3 0 1 1 2 0 3 2 0 0 1 2 3 2 1 3 1 0 1 3 2 1 0 1 0 1 2 2 0 2 0 2 0
 3 1 1 1 0 1 2 0 1 2 3 0 0 0 3 1 0 3 0 2 2 0 2 2 3 1 1 1 2 3 2 3 1 2 2 1 2
 2 2 3 0 1 3 2 1 0 1 1 1 3 1 3 3 0 1 1 3 2 0 3 0 2 0 1 0 2 1 0 2 2 3 3 2 3
 3 3 0 1 1 3 3 2 2 0 2 1 1 3 3 1 1 2 0 2 1 0 0 2 2 0 0 2 3 0 2 0 1 2 1 1 0
 0 1 3 2 1 0 0 0 2 2 0 1 2 1 3 0 2 0 0 3 3 2 2 2 1 0 0 1 3 2 2 1 1 3 0 3 3
 1 0 2 3 0 3 0 2 1 0 0 2 2 3 3 3 3 1 2 0 3 2 0 2 3 3 3 2 1 3 0 1 1 0 2 0 1
 0 0 2 1 1 3 0 0 3 0 1 1 0 2 3 1 1 1 3 3 1 3 0 3 3 3 3 2 1 2 2 1 1 3 2 2 0
 3 3 2 3]
accuracy:0.2737642585551331

```

谢谢！

