C1 基于 PCA 和全连接神经网络的乳腺癌预测

本案例以威斯康星州乳腺癌（诊断）数据集为数据基础，首先使用主成分分析（PCA）对数据进行降维，然后使用全连接神经网络实现了对乳腺癌良性、恶性的预测。本案例讲解并指导学生完成基于 PCA 和全连接神经网络的乳腺癌预测。通过本案例的学习，学生在理解 PCA 数据降维、全连接神经网络应用的同时，可以进一步加深其对理论知识的理解，并培养学生的自主学习能力、分析总结能力以及动手实践能力。

1 学习目标

⑴ 学习 PCA 降维原理，理解其在数据处理中的作用；

⑵ 掌握设计、训练和测试全连接神经网络的方法；

⑶ 能够针对特定任务，独立查阅资料、设计实验方案，编写和调试程序，并分析实验结果。

2 案例背景

本案例通过设计和构建合适的全连接神经网络模型，利用乳腺癌数据集进行网络训练，从而对乳腺癌良性、恶性进行判断，进而在一定程度上代替人工，提高判断效率。

3 基于全连接神经网络的乳腺癌预测实现原理

⑴ 数据处理

首先读取乳腺癌数据，将除诊断结果外的所有数据进行归一化。由于该数据集数据维数较多，并不是每一个列属性都对本次预测有较大的影响力，所以在数据归一化之后，利用主成分分析(PCA)对数据进行降维，仅提取最有影响的 10 个列属性作为之后的训练输入。

主成分分析(Principal Component Analysis)是一种用于探索高维数据的技术，通常用于高维数据集的探索与可视化，还可用于数据压缩、数据预处理等。PCA 能将可能具有线性相关性的高维变量合成为线性无关的低维变量，称为主成分(principal components)。新的低维数据集会尽可能的保留原始数据的变量，可以将高维数据集映射到低维空间的同时，尽可能的保留更多变量。

需要注意的是，降维就意味着信息的丢失，如果用原始数据在模型上没有效果，进而期望通过数据降维来改善模型的表现力这是不现实的，不过鉴于实际数据本身常常存在的相关性，可以想办法在降维的同时将信息的损失尽可能地降低。只有当模型在原数据上跑了一个比较好的结果，但又嫌它太慢、太复杂，此时才应采取 PCA 降维进行优化。

sklearn 库中封装了 PCA 降维方法，用户可以使用 from sklearn.decomposition import PCA 在代码中导入 sklearn 库，并使用 PCA(n_components=None, copy=True, whiten=False)，其中参数 n_components 表示 PCA 算法中所要保留的主成分个数 n，即保留下来的特征个数 n。参数 copy 表示是否在运行算法时，将原始训练数据复制一份，缺省时默认为 True。参数 whiten 表示白化（白化是一种线性变换，用于对源信号进行去相关，是一种重要的预处理过程，其目的就是降低输入数据的冗余性，使得经过白化处理的输入数据具有消除了特征之间的相关性、所有特征的方差都为 1 的性质。）缺省时默认为 False。

⑵ 神经网络设计

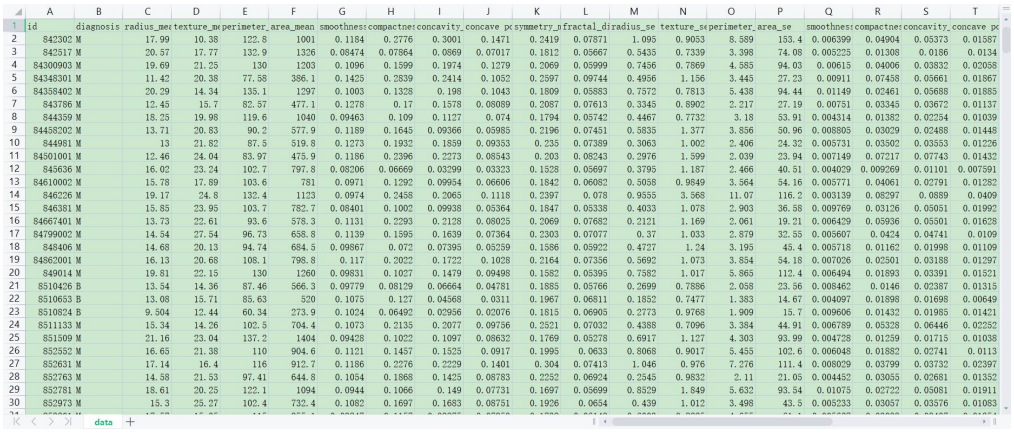
本案例示例代码中所设计的神经网络模型由三个全连接层构成，第一层全连接层的输入为之前 PCA 降维后得到的数据，即大小为 10 的一维数组，该层的输出为大小是 64 的一维

数组；第二层全连接层的输入是第一层的输出，该层的输出依旧为大小是 64 的一维数组；第三层全连接层的输入是第二层的输出，该层的输出为大小是 1 的数组。模型训练的损失函数为二元交叉熵损失函数，优化器为 Adam，共迭代训练 50 轮，最后模型的准确率可达 99%。

学习者也可以改变神经网络的层数、每层中神经元的个数、以及损失函数和优化器等超参数，尝试得到更好、更快的模型和训练策略。

4 数据集

本案例使用的数据集来自威斯康星州乳腺癌（诊断）数据集。该数据共有 569 例乳腺癌良性、恶性患者数据，其中良性患者数据 212 例，恶性患者数据 357 例。每位患者数据包括身份 ID（ID）、诊断结果（diagnosis）、半径均值（radius_mean）、纹理均值（texture_mean）、周长均值（perimeter_mean）、面积均值（area_mean）、平滑度均值（smoothness_mean）、紧致度均值（compactness_mean）、凹度均值（concavity_mean）、凹点均值（concave points_mean）、对称均值（symmetry_mean）、分形维数均值（fractal_dimension_mean）、半径标准差（radius_se）、纹理标准差（texture_se）、周长标准差（perimeter_se）、面积标准差（area_se）、平滑度标准差（smoothness_se）、紧致度标准差（compactness_se）、凹度标准差（concavity_se）、凹点标准差（concave points_se）、对称标准差（symmetry_se）、分形维数标准差（fractal_dimension_se），以及最差半径（radius_worst）、最差纹理（texture_worst）、最差周长（perimeter_worst）、最差面积（area_worst）、最差平滑度（smoothness_worst）、最差平滑度（compactness_worst）、最差紧致度（concavity_worst）、最差凹度（concave points_worst）、最差对称（symmetry_worst）、最差分形维度（fractal_dimension_worst），共 32 个属性特征值，部分数据如图 C1.1 所示。



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	id	diagnosis	radius_mea	texture_m	perimeter_	area_mean	smoothnes	compactne	concavity_	concave pc	symmetry_n	fractal_di	radius_se	texture_se	perimeter_	area_se	smoothnes	compactne	concavity_	concave pc
2	842302	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419	0.07871	1.095	0.9053	8.589	153.4	0.006399	0.04904	0.05373	0.01587
3	842517	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	0.5435	0.7339	3.398	74.08	0.005225	0.01308	0.0186	0.0134
4	84300903	M	19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069	0.05999	0.7456	0.7869	4.585	94.03	0.00615	0.04006	0.03832	0.02058
5	84348301	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1052	0.2597	0.09744	0.4956	1.156	3.445	27.23	0.00911	0.07458	0.05661	0.01867
6	84358402	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1043	0.1809	0.05883	0.7572	0.7813	5.438	94.44	0.01149	0.02461	0.05688	0.01885
7	843786	M	12.45	15.7	82.57	477.1	0.1278	0.17	0.1578	0.08089	0.2087	0.07613	0.3345	0.8902	2.217	27.19	0.00751	0.03345	0.03672	0.01137
8	844359	M	18.25	19.98	119.6	1040	0.09463	0.109	0.1127	0.074	0.1794	0.05742	0.4467	0.7732	3.18	53.91	0.004314	0.01382	0.02254	0.01039
9	84458202	M	13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366	0.05985	0.2196	0.07451	0.5835	1.377	3.856	50.96	0.008805	0.03029	0.02488	0.01448
10	844981	M	13	21.82	87.5	519.8	0.1273	0.1932	0.1859	0.09353	0.235	0.07389	0.3063	1.002	2.406	24.32	0.005731	0.03502	0.03553	0.01226
11	84501001	M	12.46	24.04	83.97	475.9	0.1186	0.2396	0.2273	0.08543	0.203	0.08243	0.2976	1.599	2.039	23.94	0.007149	0.07217	0.07743	0.01432
12	845636	M	16.02	23.24	102.7	797.8	0.08206	0.06669	0.03299	0.03323	0.1528	0.05697	0.3795	1.187	2.466	40.51	0.004029	0.009269	0.01101	0.007591
13	84610002	M	15.78	17.89	103.6	781	0.0971	0.1292	0.09954	0.06606	0.1842	0.06082	0.5058	0.9849	3.564	54.16	0.005771	0.04061	0.02791	0.01282
14	846226	M	19.17	24.8	132.4	1123	0.0974	0.2458	0.2065	0.1118	0.2397	0.078	0.9555	3.568	11.07	116.2	0.003139	0.08297	0.0889	0.0409
15	846381	M	15.85	23.95	103.7	782.7	0.08401	0.1002	0.09938	0.05364	0.1847	0.05338	0.4033	1.078	2.903	36.58	0.009769	0.03126	0.05051	0.01992
16	84667401	M	13.73	22.61	93.6	578.3	0.1131	0.2293	0.2128	0.08025	0.2069	0.07682	0.2121	1.169	2.061	19.21	0.006429	0.05936	0.05501	0.01628
17	84799002	M	14.54	27.54	96.73	658.8	0.1139	0.1595	0.1639	0.07364	0.2303	0.07077	0.37	1.033	2.879	32.55	0.005607	0.0424	0.04741	0.0109
18	848406	M	14.68	20.13	94.74	684.5	0.09867	0.072	0.07395	0.05259	0.1586	0.05922	0.4727	1.24	3.195	45.4	0.005718	0.01162	0.01998	0.01109
19	84862001	M	16.13	20.68	108.1	798.8	0.117	0.2022	0.1722	0.1028	0.2164	0.07356	0.5692	1.073	3.854	54.18	0.007026	0.02501	0.03188	0.01297
20	849014	M	19.81	22.15	130	1260	0.09831	0.1027	0.1479	0.09498	0.1582	0.05395	0.7582	1.017	5.865	112.4	0.006494	0.01893	0.03391	0.01521
21	8510426	B	13.54	14.36	87.46	566.3	0.09779	0.08129	0.06664	0.04781	0.1885	0.05766	0.2699	0.7886	2.058	23.56	0.008462	0.0146	0.02387	0.01315
22	8510653	B	13.08	15.71	85.63	520	0.1075	0.127	0.04568	0.0311	0.1967	0.06811	0.1852	0.7477	1.383	14.67	0.004097	0.01898	0.01698	0.00649
23	8510824	B	9.504	12.44	60.34	273.9	0.1024	0.06492	0.02956	0.02076	0.1815	0.06905	0.2773	0.9768	1.909	15.7	0.009606	0.01432	0.01985	0.01421
24	8511133	M	15.34	14.26	102.5	704.4	0.1073	0.2135	0.2077	0.09756	0.2521	0.07032	0.4388	0.7096	3.384	44.91	0.006789	0.05328	0.06446	0.02252
25	851509	M	21.16	23.04	137.2	1404	0.09428	0.1022	0.1097	0.08632	0.1769	0.05278	0.6917	1.127	4.303	93.99	0.004728	0.01259	0.01715	0.01038
26	852552	M	16.65	21.38	110	904.6	0.1121	0.1457	0.1525	0.0917	0.1995	0.0633	0.8068	0.9017	5.455	102.6	0.006048	0.01882	0.02741	0.0113
27	852631	M	17.14	16.4	116	912.7	0.1186	0.2276	0.2229	0.1401	0.304	0.07413	1.046	0.976	7.276	111.4	0.008029	0.03799	0.03732	0.02397
28	852763	M	14.58	21.53	97.41	644.8	0.1054	0.1868	0.1425	0.08783	0.2252	0.06924	0.2545	0.9832	2.11	21.05	0.004452	0.03055	0.02681	0.01352
29	852781	M	18.61	20.25	122.1	1094	0.0944	0.1066	0.149	0.07731	0.1697	0.05699	0.8529	1.849	5.632	93.54	0.01075	0.02722	0.03081	0.01911
30	852973	M	15.3	25.27	102.4	732.4	0.1082	0.1697	0.1683	0.08751	0.1926	0.0654	0.439	1.012	3.498	43.5	0.005233	0.03057	0.03576	0.01083

图 C1.1 威斯康星州乳腺癌（诊断）数据集部分数据内容展示

本案例为能较好处理和使用数据，首先将数据集无用部分删除（第一行各属性名称、第一列 ID 属性），并将诊断结果进行数值化（M 恶性对应替换为数字 1，B 良性对应替换为数字 0）。

原始威斯康星州乳腺癌（诊断）数据集下载地址（数据格式为.data，部分数据内容如图 C1.2 所示）：

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

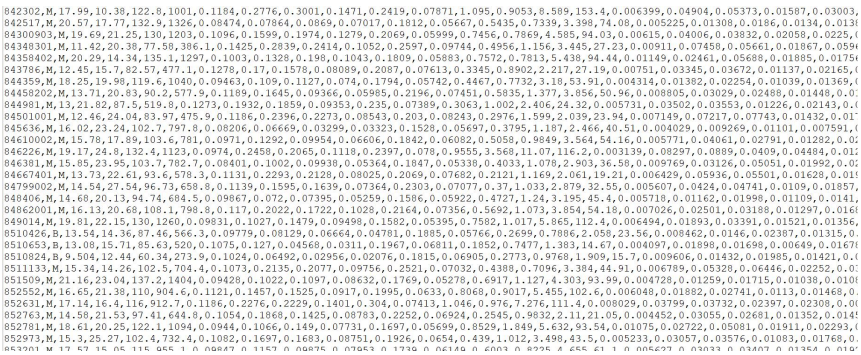


图 C1.2 威斯康星州乳腺癌（诊断）原始数据集部分数据内容展示

Kaggle 威斯康星州乳腺癌（诊断）数据集下载地址（数据格式为.csv，部分数据内容如图 C1.1 所示）：

https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data?resource=download

经处理的威斯康星州乳腺癌（诊断）数据集下载地址（数据格式为.csv，数据内容如图 C1.3 所示）：

数据集下载地址：https://pan.baidu.com/s/1bagitzF-MjIp1CN3DImHVg?pwd=1234

链接密码：1234

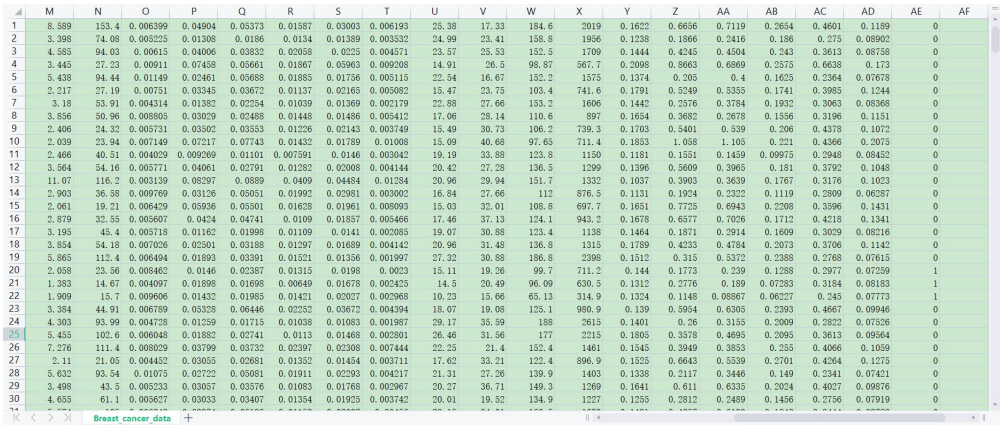


图 C1.3 经处理的威斯康星州乳腺癌（诊断）数据集部分数据内容展示

5 要求

5.1 基本要求

⑴ 使用数据归一化、PCA 等技术完成必要的数据预处理操作；

⑵ 设计神经网络模型，调整超参数和模型结构，体会参数调整对模型训练的影响。

5.2 拓展要求

通过查询资料，独立使用机器学习算法实现乳腺癌预测，并分析对比利用机器学习和本案例两种不同实现方式的各自区别和特点。

5.3 撰写相关实验文档

包括源代码说明文档、实验结果分析文档等。

C2 基于 CNN 的影评文本分类

本案例以《神经网络与深度学习——TensorFlow 实践》第 14 讲“卷积神经网络”为基础，实现基于 CNN 的影评文本分类，介绍了影评文本分类的原理及案例的实现流程，对核心源码进行了解析，并指导学生完成相关实验，分析总结实验结果。通过本案例的学习，可以拓宽视野，培养学生自主学习能力、工程实践能力和分析总结能力。

1 学习目标

⑴ 理解文本信息的处理方法和流程；

⑵ 理解应用卷积神经网络对文本信息分类的原理与方法；

⑶ 能够针对特定任务，独立查阅资料、设计实验方案，编写和调试程序，并分析实验结果。

2 案例背景

影评是观众对电影最直观的感受，不仅能够反映观众对一部电影的接受程度，还可以更好地帮助电影制作方及时获取反馈、提高创作水平。然而，在电影票房上千万的情况下，影评的数量非常多，使用人工的方式去逐条浏览评论，判断其为正面评价还是负面评价，不仅浪费时间，而且成本过高。为了解决上述问题，可以尝试使用机器学习的方式来分析影评。

3 数据集

IMDB 数据集来源于计算语言学协会第 49 届年会(ACL2011 会议)收录的 *Learning Word Vectors for Sentiment Analysis* 论文，数据集中包含来自互联网电影数据库的 50000 条影评数据，每条影评都是一个独立的 txt 文本文件，影评内容由多个英文单词组成，长度不等。所有影评数据被平均划分为训练集和测试集（各 25000 条数据），分别存放在 train 和 test 文件夹中。训练数据集和测试数据集中各包括 12500 条正面影评和 12500 条负面影评，分别存放在文件夹 pos 和 neg 中。

在 TensorFlow 的 keras 模块中集成了 IMDB 数据集，并对该数据集进行了预处理，依据 imdb_word_index.json 字典，将影评文本中的单词转换成为单词的数字索引。即每条影评文本被处理为长度不一的数字列表，其中每个数字表示影评文本中的单词；同时，标签表示为整数 0 或 1，0 表示负面影评，1 表示正面影评。

数据集下载地址：https://ai.stanford.edu/~amaas/data/sentiment/

说明：imdb_word_index.json 文件是一个字典，其中共包含 88584 个单词。单词及其单词数字索引以键-值的格式存放，其中键代表单词，值代表单词的数字索引。字典中单词的顺序按照词频（单词在文本中出现次数）进行排列，词频越高，单词的数字索引越小。

下载/调用 imdb_word_index.json 字典文件：imdb.get_word_index()

4 要求

4.1 基本要求

构建神经网络模型，使用 TensorFlow 中集成的 IMDB 数据集训练并测试模型，完成对正面影评和负面影评的分类。

⑴ 下载 IMDB 数据集

下载文本形式的 IMDB 原始数据集，了解数据集的组成和特点。

⑵ 下载并浏览 keras 中内置的 IMDB 数据集

编写代码，下载 keras 中内置的 IMDB 数据集，分别查看训练集和测试集中的样本数；随机输出训练集中的 3 条样本和对应的标签值，并查看其长度。

⑶ 输出影评文本

Tensorflow 中集成的 IMDB 数据集中已将每条影评处理为数字列表，无法直观展示出影评的文本信息。编写代码，将数字化的影评转化为影评文本，如图 C2.1 所示，并与原始数据集中对应的样本对比。

```
处理前的评论 [1, 194, 1153, 194, 8255, 78, 228, 5, 6, 1463, 4369, 5012, 134, 26, 4, 715, 8, 118, 1634, 14, 394, 20, 13, 119, 954, 1
89, 102, 5, 207, 110, 3103, 21, 14, 69, 188, 8, 30, 23, 7, 4, 249, 126, 93, 4, 114, 9, 2300, 1523, 5, 647, 4, 116, 9, 35, 8163, 4,
229, 9, 340, 1322, 4, 118, 9, 4, 130, 4901, 19, 4, 1002, 5, 89, 29, 952, 46, 37, 4, 455, 9, 45, 43, 38, 1543, 1905, 398, 4, 1649, 2
6, 6853, 5, 163, 11, 3215, 2, 4, 1153, 9, 194, 775, 7, 8255, 2, 349, 2637, 148, 605, 2, 8003, 15, 123, 125, 68, 2, 6853, 15, 349, 1
65, 4362, 98, 5, 4, 228, 9, 43, 2, 1157, 15, 299, 120, 5, 120, 174, 11, 220, 175, 136, 50, 9, 4373, 228, 8255, 5, 2, 656, 245, 235
0, 5, 4, 9837, 131, 152, 491, 18, 2, 32, 7464, 1212, 14, 9, 6, 371, 78, 22, 625, 64, 1382, 9, 8, 168, 145, 23, 4, 1690, 15, 16, 4,
1355, 5, 28, 6, 52, 154, 462, 33, 89, 78, 285, 16, 145, 95]
处理后的评论 <START> big hair big boobs bad music and a giant safety pin these are the words to best describe this terrible movie i
love cheesy horror movies and i've seen hundreds but this had got to be on of the worst ever made the plot is paper thin and ridicu
lous the acting is an abomination the script is completely laughable the best is the end showdown with the cop and how he worked ou
t who the killer is it's just so damn terribly written the clothes are sickening and funny in equal <UNK> the hair is big lots of b
oobs <UNK> men wear those cut <UNK> shirts that show off their <UNK> sickening that men actually wore them and the music is just <U
NK> trash that plays over and over again in almost every scene there is trashy music boobs and <UNK> taking away bodies and the gym
still doesn't close for <UNK> all joking aside this is a truly bad film whose only charm is to look back on the disaster that was t
he 80's and have a good old laugh at how bad everything was back then
```

图 C2.1 输出函数处理前后的评论

⑷ 统一每条影评的长度

由于不同影评文本的长度不统一，在将影评数据送入模型之前，应首先统一每条影评的长度。编写代码，统一每条影评数据的长度。

提示：填充列表函数 tf.keras.preprocessing.sequence.pad_sequences()

⑸ 建立、训练和测试模型

编写代码，建立学习模型，使用 IMDB 数据集训练并测试模型。尝试改变模型结构、超参数和迭代次数，综合考虑模型损失、准确率和训练时间等因素，寻找最佳方案，记录实验结果。

⑹ 可视化输出

编写代码，以可视化的形式，展示训练过程和结果。

⑺ 优化代码

将“要求⑹”中需要重复执行的代码采用函数优化，并提供优化后完整的代码。

4.2 撰写项目文档

以表格或其他适当的图表形式，对比不同的模型/方案的结果，分析上述实验结果，并进行总结，撰写报告。

C3 基于卷积神经网络的验证码识别

本案例实现了一个验证码自动识别程序，并对实现步骤、核心源码进行了解析，指导学生完成训练模型的建立、分析实验结果。通过本案例的学习，可以了解卷积神经网络的在数字图像处理领域的应用，通过自主设计和搭建卷积神经网络，培养学生自主学习的能力、工程实践能力和分析总结能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 理解验证码识别的原理及其方法；

⑵ 掌握设计、训练和测试卷积神经网络的流程和方法；

⑶ 能够针对特定任务，独立查阅资料、设计实验方案，编写和调试程序，并分析实验结果。

2 案例背景

验证码全称为全自动区分计算机和人类的图灵测试（Completely Automated Public Turing test to tell Computers and Humans Apart，简称 CAPTCHA），是一种区分用户是计算机和人的公共全自动程序。网站为了防止用户利用机器人自动注册、登录、灌水，通过强制人机交互来抵御机器自动化攻击，从而确保服务器系统的稳定和用户信息的安全。图片验证码是目前最常用的一种，将一串随机产生的数字或符号，生成一幅图片，图片里加上一些干扰象素（防止 OCR），由用户肉眼识别其中的验证码信息，输入表单提交网站验证，验证成功后才能使用网站中的某些功能。

本案例利用 Pytorch 建立卷积神经网络，实现对验证码的识别。主要内容包括：数据集生成、datasets 数据集加载、label 格式转化、CNN 神经网络构建、模型训练和预测等。

3 数据集生成

通过在原有 conda 环境中直接下载 captcha 库，即：pip install captcha，则可在代码中调用 ImageCaptcha()方法来随机生成验证码测试集与训练集，本案例通过循环语句随机生成 10000 张训练集与 300 张测试集分别放入./datasets/train 与./datasets/test 中，部分数据集图片如图 C3.1 所示。

图 C3.1 部分数据集展示

4 datasets 数据加载

为了准确加载出验证码数据集的图片和标签信息，案例在 mydataset.py 中用到了 img_lable=image_name.split("_")[0]方法来提取标签中的内容信息，例如，原标签文件信息为 0m9e_1660807267，则经过 split("_")[0]之后，所加载到的标签文件信息则为 0m9e。

然而为了使验证码图片数据能够加载到神经网络当中，案例同时也用到了 ToTensor 方法将原本的图片信息转化为 Tensor 格式，并且为了在之后的卷积神经网络中更易于提取特征值，将图片经过 Resize((60,160))方法，和 Grayscale()，对其进行固定参数大小至 60*160 操作，以及灰度化操作。

经过以上步骤之后对图片的 shape 以及标签进行输出，可以得到以下结果：

torch.Size（[1，60，160]）

其中 1，代表此图片为灰度图片，后两位为图片的宽和长，而 012r 代表这张图片的标签，为了更加直观看见这张图片，案例通过 tensorboard 来展示所需加载的图片，如下图所示。

图 C3.2 调整大小和灰度后的验证码图片

5 label 格式转化

同样，为了让模型能够识别正确的标签内容，也需要将第 3 小节中加载的 label 数据进行 Tensor 转化即，将 label 中的数字和字母内容转化为矩阵，本案例将建立 4 列 36 行的矩阵来存放标签内容。

```
captcha_array=list("0123456789abcdefghijklmnopqrstuvwxyz")
captcha_size=4
#One_hot
#建立 36 行 4 列的二维数组
vectors=torch.zeros((common.captcha_size,common.captcha_array.__len__()))
for i in range(len(text)):
#将所对应的 list 位置赋值为 1
    vectors[i,common.captcha_array.index(text[i])]=1 return vectors
```

以 common.captcha_array 为序列表，例如，若将标签内容为“aaab”的标签转化格式，则最终输出转化为 tensor 格式的内容应该为：

```
tensor([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
     0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
     0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
     0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
     0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
torch.Size([4, 36])
```

最终，为了能够进行线性输出，需要将原本的[4,36]数组进行乘积，合并为一个一维数组，并且由于最终的 CNN 模型所输出的尺寸为验证码长度*验证码大小（4*36=144），即利用 img_lable.view(1,-1)[0]，所以要将原数组尺寸展平成 torch.size[144]的格式。

6 CNN 的建立

卷积神经网络（CNN）是一类包含卷积计算且具有深度结构的神经网络，具有良好的特征抽取和分类输出能力，是深度学习的代表算法之一。同时，卷积神经网络具有较强的学习表征的能力，能够按其阶层结构对输入信息进行平移不变分类，因此也被称为“平移不变人工神经网络”。

卷积神经网络主要由三部分组成，分别是卷积层，池化层，和全连接层。卷积神经网络可以分为，一维卷积神经网络（Conv1d）、二维卷积神经网络（Conv2d）和三维卷积神经网络（Conv3d）。由于本案例所需要的输入为图片转化后的二维矩阵，所以采用二维卷积神经网络，同时二维卷积神经网络对图片具有良好的提取特征以及压缩功能。

输入层：将图片输入后，还需转化成二维矩阵的格式，以便于卷积层的读取和功能的实现。

卷积层：主要作用是提取输入层图片的主要特征值，在本层需要设置输入通道个数，输出通道个数，以及卷积核的大小、滑动步长等参数。

最大池化层：主要作用是压缩图片的参数，本层需要设置的参数是最大池化层 kernel 的尺寸。

全连接层：全连接层的主要工作是将从前面网络所提取到的特征值输出到到样本标记空间中，即实现维度的转换。

输出层：将全连接层的输出图片进行叠加操作，与输入层的功能相反，输出层主要工作是将二维的矩阵转化为一张可视化的提取过特征值轮廓的图片。

案例采用 4 层卷积层，3 层最大池化层，以及全连接层构建卷积神经网络，具体网络架构如图 C3.3 所示。

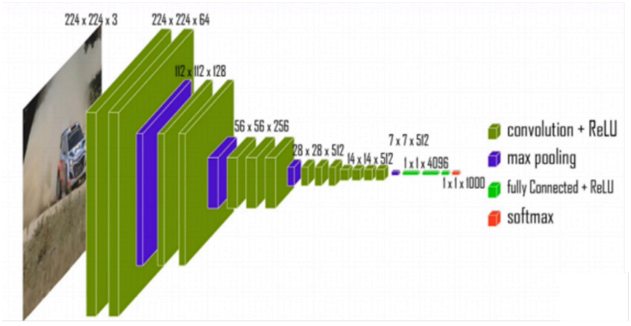


图 C3.3 CNN 网络架构

7 模型训练

经过了上述数据集生成，datasets 数据集加载，label 格式转化，cnn 神经网络构建等几个步骤之后，下面就是要将上述步骤所得到的输出进行加载与训练的过程，本次案例用到的损失函数为 MultiLabelSoftMarginLoss 多标签交叉熵损失函数，优化器为 Adam。训练迭代次数设置为 140 轮，批次设置为 64，并规定每训练 100 次（即训练 6400 张图片）输出一次损失值，并记录，因而最终会输出至第 280 次训练结果的 loss 值。

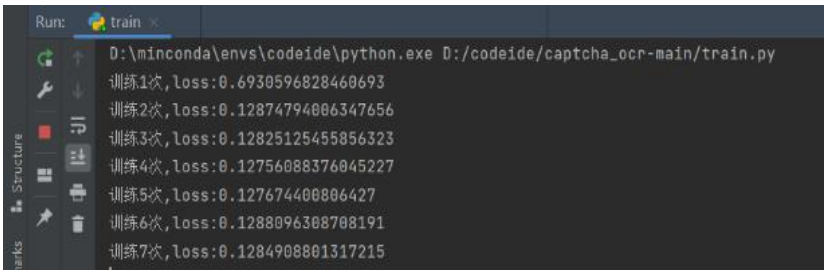
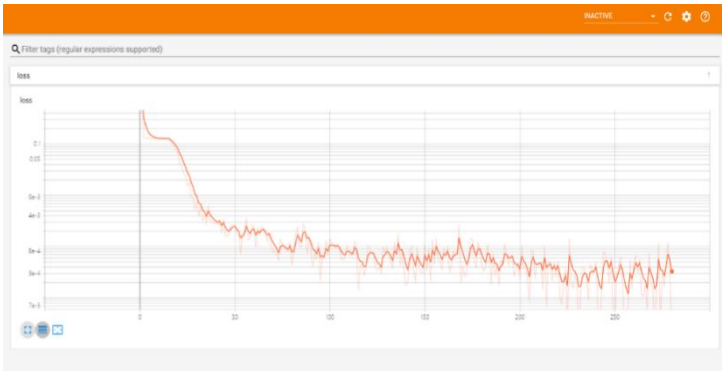


图 C3.4 训练过程展示

为了更加直观显示 loss 的变化，案例利用 tensorboard 对其进行折线图绘制，如图 C3.5



所示。

图 C3.5 280 次训练过程的 loss 变化

8 测试数据集预测

案例通过利用 captcha 库随机引入 300 张验证码数据放入./datasets/test 中，用于测试集的建立，而训练好的模型则保存为 model.pth 文件，同样根据 my_datasets 的 mydatasets 方法进行图片的灰度处理和固定参数后进行加载，以及 one_hot 中对标签的格式转化后再对数据

进行预测，下面分别实现对整个测试集的图片测试以及单张图片的测试。

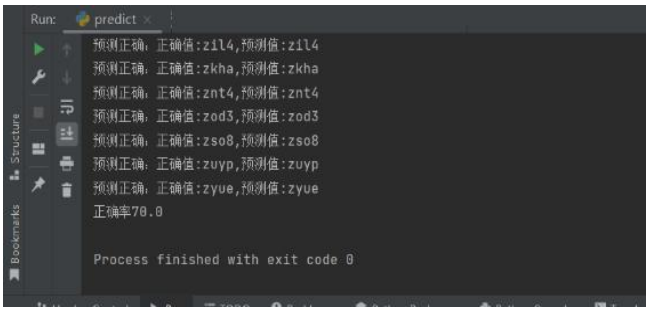


图 C3.6 测试数据预测

在单张预测时，利用 pred_pic 方法对单张图片进行灰度处理并利用 model.pth 模型进行预测，并打印出图片的 tensor 大小和预测的标签内容。

单张预测结果和真实标签内容一致，预测成功，其中 torch.size[1,60,160]表明了预测图片的信息为灰度图片，同时宽度为 60，长度为 160。

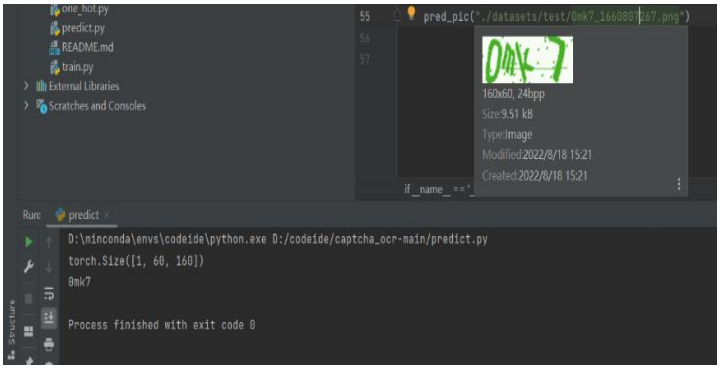


图 C3.7 单张图片预测结果成功

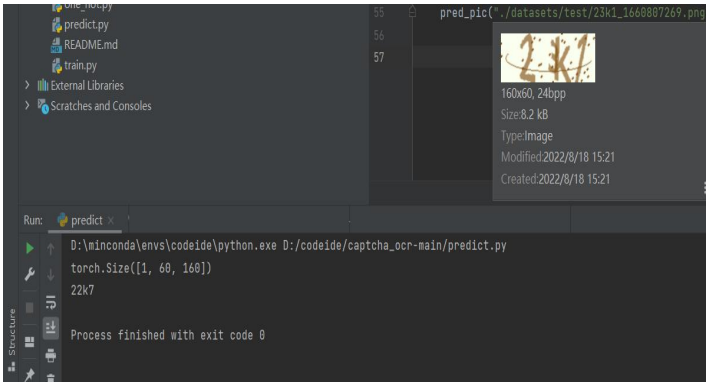


图 C3.8 单张图片预测结果失败

9 要求

9.1 基本要求

⑴ 构建项目工程，正确配置源代码运行环境，并成功运行源代码；

⑵ 实验结果分析：根据设置不同训练集个数与其他训练参数建立正交对比实验组，并分析实验结果。

9.2 撰写项目文档

项目文档包括基于卷积神经网络的验证码识别原理与流程、源代码说明文档、实验过程及结果分析等。

C4 基于 TensorFlow.Lite 的手写数字识别

案例将 TensorFlow 模型部署在安卓系统中，在移动端实现数字 0-9 的手写输入和识别。通过本案例的学习，掌握 TensorFlow.Lite 的应用方法、流程和项目结构，提高工程实践能力，提升创新意识。

1 学习目标

⑴ 掌握应用 TensorFlow.Lite 的流程、步骤和项目结构；

⑵ 能够针对特定任务，独立查阅资料、设计实验方案，编写和调试程序，并分析实验结果。

2 案例背景

为适应近几年移动文化浪潮和交互方式的改变，同时降低深度学习门槛，TensorFlow 团队在 2017 年底开源了 TFLite。这是一款轻量、快速、兼容性高的专门针对移动应用场景的深度学习工具，用于在移动设备和嵌入式设备上部署模型，主要由 Converter 和 Interpreter 组成。

在中国，在移动应用方面，TFLite 已经运用的十分成熟。网易使用 TFLite 做 OCR 处理；爱奇艺使用 TFLite 实现视频中的 AR 效果；WPS 用它来做一系列文字处理等等。TFLite 支持 Java、C++、Python 等多种语言的 API，为开发者提供了极大的方便。在安卓平台下构建 TFLite 更是具有轻量级、快速启动和内存利用率高的特点。

本次案例训练好的手写数字识别模型进行保存、转换，并在安卓系统中部署，实现在移动端进行数字 0-9 的手写输入和识别。

3 案例数据集

手写数字数据集是美国国家标准与技术研究院收集整理的大型手写数字数据库，由纽约大学的 Yann LeCun 等人维护，这个数据集包括了 60000 条训练数据和 10000 条测试数据，由 250 个不同的人手写而成，数据集中的手写数字图片，都是 28×28 像素的灰度图像，并将每幅图片存储在一个 28×28 的二维数组中，数组中的每个元素对应于图片中的一个像素，如图 C4.1 所示。TensorFlow 的 Keras 模块中集成了手写数字数据集。

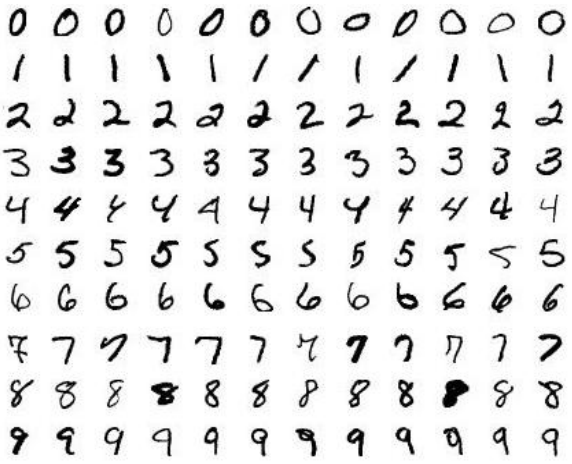


图 C4.1 手写数字识别数据集

4 项目结构

图 C4.2 展示了在 TensorFlow2.0 中的 TFLite 模型转换过程，用户在自己的工作台中使用 TensorFlow API 构造 TensorFlow 模型，然后使用 TFLite 文件格式（FlatBuffers 格式）。在移动端，TFLite 解释器接受 TFLite 模型，调用不同的硬件加速器例如 GPU 进行执行。

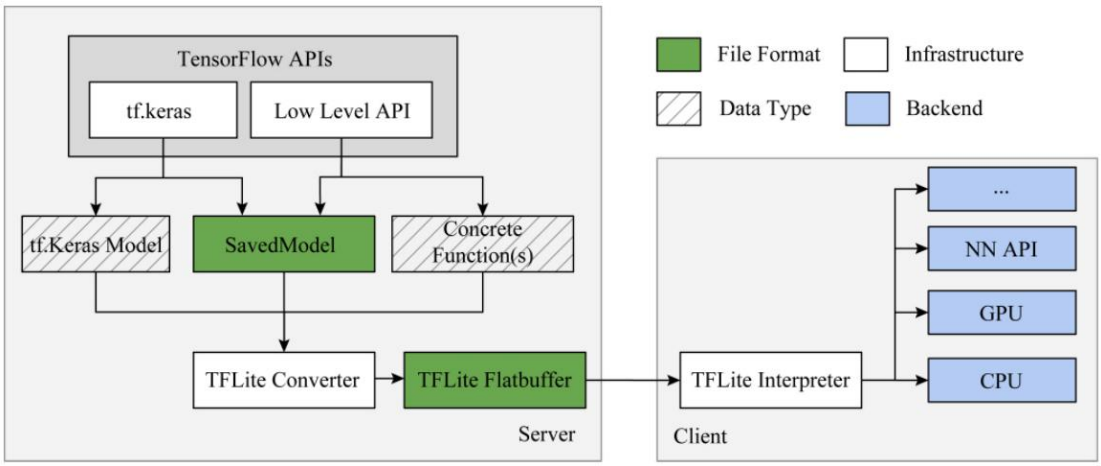


图 C4.2 TFLite 模型转换过程

这是本案例的整体结构图，由 Jupyter NoteBook + Android Studio 共同完成，Jupyter NoteBook 中包括：构建模型、训练模型、保存模型以及转换模型，数据集使用 Keras 模块中集成的 MNIST 数据集；Android Studio 中包括：界面设计、加载模型、识别手写数字等内容。

```
Tflite-Mnist-Digits
├─jupyter notebook
│    ├─datasets
│    │      mnist   #在 Keras 模块中加载 MNIST 数据集
│    │
│    ├─model
│    │      tf.keras.Sequential( ) #构建 Sequential 模型
│    │      model.compile( ) #配置训练方法
│    │      model.summary() #查看摘要
│    │      model.fit( ) #训练模型
│    │
│    ├─save and convert
│    │      save_model( )   #保存整个模型
│    │      tflite_model_file.write_bytes( )   #转换为 tflite 文件
│    │
├─Android Studio
```

```
│   │       manifests   #配置信息
│   │       MainActivity   #核心类
│   │       PaintView     #触碰响应
│   │       DigitsDetector   #手写数字识别的核心方法
│   │       activity_main.xml   #页面布局
│   │
│   ├─assest
│   │       mnist.tflite   #mnist 文件存放于 assest 文件夹下
    ─┤
```

5 要求

5.1 基本要求

⑴ 模型的建立、测试与保存

选择全连接网络或卷积神经网络等任一方法构建手写数字识别模型，使用 MNIST 数据集进行训练及测试，调整模型，使其在测试集上的准确率达到最高，保存准确率最高的模型。保存方式选择 model.save( )方法保存整个模型，保存格式选择 HDF5 格式或 SavedModel 格式。

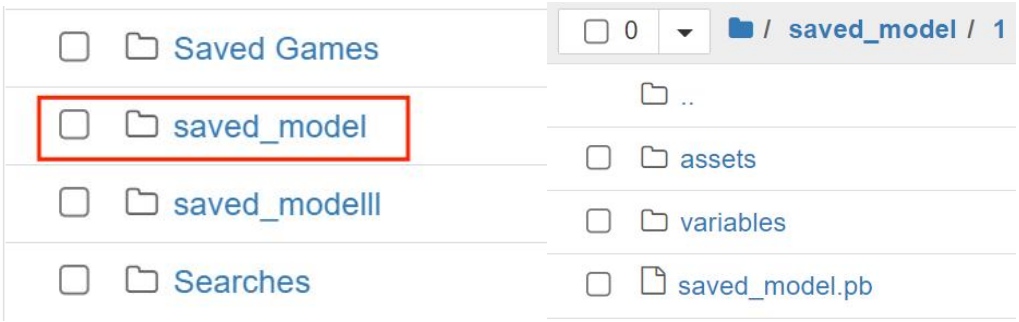


图 C4.3 保存模型

⑵ 模型转换

使用 Python API 实现模型转换，得到手写数字识别的 tflite 文件。

图 C4.4 转换模型

⑶ 安卓部署

使用 Android Studio 创建一个项目，将要求 2 中转换后的模型部署到该项目中，能够在移动端实现数字的手写和识别。选择模拟器或真机进行测试。

界面设计中包括：画布：用于数字的手写；识别按钮：点击该按钮可进行手写数字的识别；清除按钮：用于清空画布；识别结果的展示形式不做限制，图 C4.5 中采用的是直接显示结果的形式。

图 C4.5 手写数字识别示例

⑷ 分析和总结

对实验过程和结果进行分析和总结。

5.2 撰写项目文档

项目文档包括基于 TensorFlow.Lite 的手写数字识别项目结构与流程、源代码说明文档、实验过程及结果分析等。

提示：

① 实例化 converter：tf.lite.TFliteConvrter.from_saved_model(path) ，path 为保存模型的目录

② 创建一个“mnist.tflite”的文件，使用如下语句将 data 写入“mnist.tflite”文件：tflite_model_file.write_bytes(data)

③ 在【build.gradle(Model:app)】中 android-->dependencies 中需要添加如下代码，加载 TensorFlow 依赖。

```
dependencies {

    implementation 'org.tensorflow:tensorflow-lite:1.10.0'

}
```

④ 在【build.gradle(Model:app)】中的 android-->aaptOptions 中需要加入如下代码，设置 tflite 文件不压缩，确保 tflite 文件可以被 Interpreter 正确加载。

```
aaptOptions {
    noCompress "tflite"
}
```

⑤ 需要自己新建 assest 文件夹，存放 mnist.tflite 文件。

C5 基于 TensorFlow.js 的手写数字识别应用

本案例实现在浏览器端进行数字手写输入和识别，介绍了 TensorFlow.js 相关开发技术、工作原理及实现流程，对核心源码进行了解析，并指导学生完成相关实验。通过本案例的学习，可以应用 TensorFlow.js 这一高性能的、易于使用的深度学习工具，培养学生的工程实践能力。

1 学习目标

⑴ 掌握 TensorFlow.js 进行手写数字识别的流程、步骤和项目结构；

⑵ 能够针对特定任务，独立查阅资料、设计实验方案，编写和调试程序，并分析实验结果。

2 案例背景

TensorFlow.js 是 Google 推出的一个开源的基于 JavaScript 的深度学习库，用于训练并部署深度习模型。在 TensorFlow.js 库的支持下，前端开发者可以直接在浏览器环境中来实现深度学习的功能，相对于基于其他语言的 TensorFlow 库，使用 TensorFlow.js 可以在浏览器内进行模型的训练及数据的预测，提高服务器资源利用，增强客户端响应运算结果的速度，由于数据是在浏览器上进行传输的，因此也增加了数据的安全性。

本案例使用卷积神经网络构建手写数字识别模型，并使用 TensorFlow.js 将训练好的模型部署在浏览器上，完成对手写数字的识别。

3 相关开发技术介绍

3.1 HTML

HTML 的全称为超文本标记语言，是一种标记语言。它包括一系列标签，通过这些标签可以将网络上的文档格式统一，使分散的 Internet 资源连接为一个逻辑整体。HTML 文本是由 HTML 命令组成的描述性文本，HTML 命令可以说明文字，图形、动画、声音、表格、链接等。

文件结构及标签：

⑴ HTML 文档以<!DOCTYPE HTML>开头。<!DOCTYPE HTML> 告诉浏览器这是一个 html 文档。

⑵ <html>和</html>之间的内容标记网页。

元数据包含在<head>元素中，<head> 元素也定义了与外部的关系资源(如 CSS 样式表)、内联 CSS 样式以及脚本等。

⑶ <body> 和</body> 之间的内容标记可见的网页内容，通过<body>元素添加 html 文档的内容。

3.2 JavaScript

JavaScript（简称“JS”）是一种轻量级的面向对象的编程语言，也是一种跨平台的解释型语言，不需要提前编译，能在各种操作系统下运行，既能用在浏览器中控制页面交互，也能用在服务器端作为网站后台（借助 Node.js），因此 JavaScript 是一种全栈式的编程语言。本案例使用 JavaScript 在浏览器中控制页面交互。

3.3 Canvas、Fabric.js、Chart.js

Canvas API（画布）是在 HTML5 中新增的标签，用于在网页实时生成图像，并且可以

操作图像内容。它没有自己的行为，但是定义了一个 API 支持脚本化客户端绘图操作，所有的绘制工作必须在 JavaScript 内部完成，其例程代码包含在 HTML<canvas>标签中。在本案例中用于创建一个手写数字画布。

Fabric.js 是专门为简化 Canvas 程序所开发的库。Canvas 为我们提供了一个很好的画布，可以绘制简单图形，但当做一些复杂的图像或编写一些复杂的效果时，就很不方便。Fabric.js 为 Canvas 提供了它所缺少的对象模型、交互和一些必要的工具，用来更方便的构造画布。

Chart.js 是基于 HTML5 的 JavaScript 图形库，它支持多种方式进行数据的可视化，若浏览器改变大小，Chart.js 可以调整图表大小以适应。Chart.js 是模块化的，并且每种图表类型都已拆分，我们可以仅加载项目所需的图表类型。在本案例中用于将预测后的数据在图表上进行可视化。

3.4 TensorFlow.js

Tensorflow.js 是一个开源的基于硬件加速的 JavaScript 库，用于训练和部署深度学习模型。TensorFlow.js 可以为我们提供高性能的、易于使用的深度学习构建模块，允许我们在浏览器上训练模型及数据的预测。此处 TensorFlow.js 将使用训练好的模型加载到浏览器并进行手写数字的预测。

3.5 Flask 库、Flask-cors 库

Flask 是一个使用 Python 编写的轻量级 Web 应用框架，核心构成比较简单，但具有很强的扩展性和兼容性。其运行原理如图 C5.1 所示。

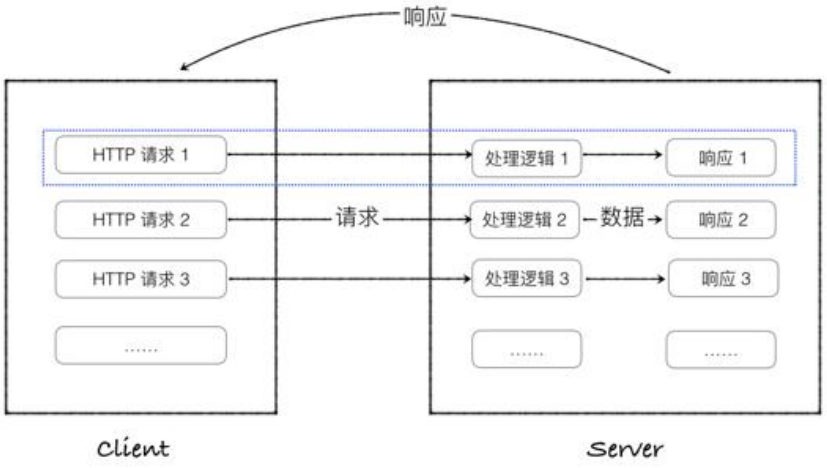


图 C5.1 Flask 运行原理

Flask-cors 库用来处理跨域的需求（跨域：如果在 js 脚本中试图请求拉取的后端资源的协议或域名与 js 本身的不一致，那么浏览器会阻止这个请求——这是一种安全保护策略）。本案例中用于读取本地模型文件时进行跨域的处理。

4 数据集

MNIST 数据集是由纽约大学的 Yann LeCun 等人维护，该数据集包括了 60000 条训练数据和 10000 条测试数据，由 250 个不同的人手写而成，这些人中，有 50%是高中生，50%是人口普查局的工作人员。数据集中的手写数字图片，都是 28×28 像素的灰度图像，但是它们并不是以图像文件的形式存储的，而是将每幅图片存储在一个 28×28 的二维数组中，数

组中的每个元素对应于图片中的一个像素。部分数据集如图 C5.2 所示。

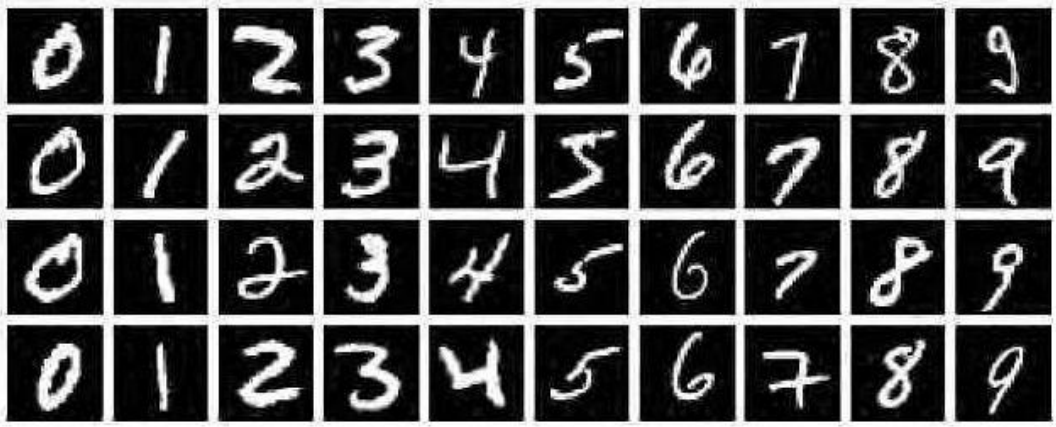


图 C5.2 MNIST 数据集

数据集下载地址：

链接：http://yann.lecun.com/exdb/mnist/

5 要求

5.1 基本要求

在浏览器端实现基于 TensorFlow.js 的手写数字识别案例。在网页中创建一个用于手写数字的画布，点击识别按钮对手写数字进行识别，将识别结果进行可视化显示，点击清除按钮可清空画布。具体要求如下：

图 C5.3 手写数字识别界面

手写数字识别案例界面设计及功能需包括：

⑴ 画布：用于数字的手写。

⑵ 识别按钮：点击该按钮可进行手写数字的识别。

⑶ 清除按钮：用于清空画布。

⑷ 识别结果可视化：用于将识别的结果进行可视化显示。

要求：界面设计美观，控件布局合理，可参考图 3，也可自行设计。

5.2 拓展要求

⑴ 历史记录：每次运行该案例时，将所有已识别过的手写数字及识别结果进行记录并显示，关闭浏览器会清空记录。

⑵ 手写数字准确率：每次运行该案例时，能够进行准确率的计算并显示，例如：若只进行了一次手写数字识别且识别准确，则准确率为 100%；若一共进行两次手写数字识别，一次识别正确而一次识别错误，则识别准确率为 50%。准确率根据手写数字次数的增加而动态变化，以此类推，关闭浏览器会清空记录。

⑶ 将识别错误的手写数字图片保存至本地，便于分析该手写数字图片识别错误的原因。

5.3 撰写项目文档

包括手写数字识别案例原理与流程、源代码说明文档、实验过程及结果分析等。

C6 猫狗大战——数据增强在深度学习中的应用

本案例使用 Kaggle 竞赛中“猫狗大战(Dogs vs Cats)”数据集，设计卷积神经网络，实现了对数据集中猫、狗图像的识别。

在本案例中，包括数据增强、卷积神经网络的应用等内容，对网络和代码的关键部分进行了解析，并提供了猫狗分类的评估结果和识别实例。通过本案例的学习，可以掌握深度学习中数据增强的方法，加深对卷积神经网络理解，培养学生自主学习的能力、工程实践能力和分析解决问题的能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 理解数据增强的原理，掌握使用使用 ImageDataGenerator 进行数据增强的方法。

⑵ 根据案例提供的数据集和要求，完成猫狗分类实例。

⑶ 培养学生根据特定任务独立查阅资料，分析问题并解决问题的能力。

2 案例背景

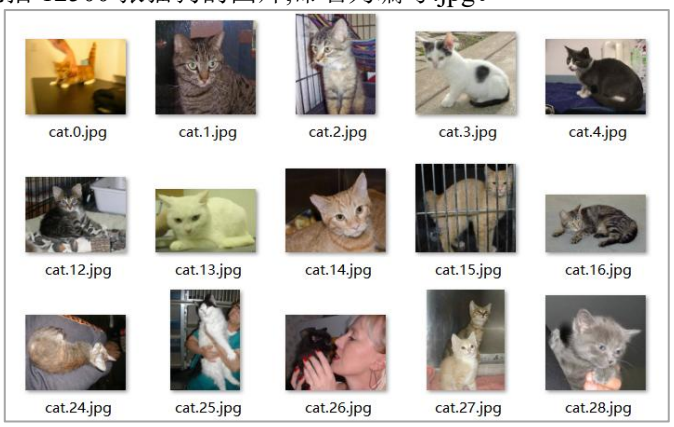
随着卷积神经网络的不断发展，其强大的特征提取能力和较快的速度得到了越来越多人们的青睐，卷积神经网络也被不断应用于目标跟踪，目标检测等多个领域。

猫狗大战(Dogs vs Cats)是 Kaggle 竞赛中的赛题之一，主要是利用给定的数据集，构建神经网络来实现猫、狗图像的识别。利用神经网络强大的特征提取能力和学习性强的优点作为猫狗识别的方法,可以提高识别准确率，达到良好的分类效果。

3 猫狗大战数据集

以猫狗大战数据集为例，共分为两个文件夹，Train 和 Test，Train 文件夹里包括各种姿态的猫狗图片各 12500 张,命名为 cat/dog+编号.jpg。例如 cat.1000.jpg。

Test 文件夹里包括 12500 张猫狗的图片,命名为编号.jpg。



C6.1 猫狗大战数据集

要获取数据集，首先需要登录 Kaggle 网站，并按照要求注册一个 Kaggle 账号。

然后进入到猫狗大战竞赛页面。

选择 Rules，并点击我接受。

选择 Data 并点击下载即可完成猫狗大战数据集的下载。

下载成功后，我们即可在文件夹中看到整个猫狗大战的数据集。

4 要求

4.1 基本要求

⑴ 使用卷积神经神经网络构建猫狗分类模型。

⑵ 使用猫狗大战数据集完成模型训练和测试，并输出结果。

⑶ 应用猫狗分类模型：将图片导入模型，进行分类，完成猫狗识别。

4.2 拓展要求

⑴ 为猫狗识别网络设置可视化页面，要求界面美观，简介。

⑵ 使用 ImageDataGenerator 类中多种方法，为模型增加数据增强处理

4.3 撰写项目文档

包括基于 CNN 的猫狗识别网络的简介、原理与方法，程序流程与源码解析、实验过程及结果分析等。

C7 基于深度学习的泰坦尼克号旅客生存概率预测

本案例介绍了对泰坦尼克号旅客生存概率预测模型的构建、测试的基本原理和实验流程，对关键步骤进行了解析，并指导学生完成实验、分析实验结果。通过本案例的学习，可以扩展学生对课程内容的理解与运用，熟练掌握深度学习网络的一般流程；通过动手实践和对实验结果的思考与改进，培养学生自主学习的能力、工程实践能力和分析总结能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 理解本案例中基于深度学习的生存概率预测网络的构建流程，利用已有数据集，掌握建立、训练一个预测网络的能力；

⑵ 调试运行程序，培养自主学习、研究的能力。

2 案例背景

1912 年 4 月 15 日，泰坦尼克号在首次航行期间撞上冰山后沉没，船上共有 2224 名人员（包括乘客和机组人员），共有 1502 人不幸遇难。造成海难失事的原因之一是乘客和机组人员没有足够的救生艇。这场耸人听闻的悲剧震惊了国际社会，并导致了更好的船舶安全条例。本次案例将根据已有数据集预测泰坦尼克号上的旅客生存概率。

3 数据集

Titanic 数据集来自 kaggle 官方，分为两组：训练集（train.csv）和测试集（test.csv）。该数据集包含 11 个特征，每个特征包含 1309 条数据，部分特征中存在空值。这 11 个特征分别是：

Survived：0 代表死亡，1 代表存活；Pclass：乘客所持票类，有三种值(1,2,3)；Name：乘客姓名；Sex：乘客性别；Age：乘客年龄(有缺失)；SibSp：乘客兄弟姐妹/配偶的个数(整数值)；Parch：乘客父母/孩子的个数(整数值)；Ticket：票号(字符串)；Fare：乘客所持票的价格(浮点数，0-500 不等)；Cabin：乘客所在船舱(有缺失)；Embark：乘客登船港口:S、C、Q(有缺失)，如表 C7.1 所示。

表 C7.1 数据集特征介绍

特征	特征说明	数据说明	数据类型
pclass	舱等	1:头等舱，2：二等舱，3：三等舱	整型
survival	是否生存	0：否，1：是	整型
name	姓名		字符串类型
sex	性别	Female：女性，male：男	字符串类型
age	年龄		整型
sibsp	兄弟姐妹或者配偶也在船上的数量		整型
parch	双亲或者子女也在船上的数量		整型
ticked	船票号码		字符串类型
fare	船票费用		整型
cabin	舱位号码		字符串类型
embarked	登船港口	C：Cherbourg, Q：Queenstown,	字符串类型

4 要求

4.1 基本要求

⑴ 下载数据集

百度网盘链接: https://pan.baidu.com/s/1bagitzF-MjIp1CN3DImHVg?pwd=1234

⑵ 数据预处理

包括：空值处理（均值、中位数等）；设置特征与标签；数据打乱；数据集分割以及将性别与港口等特征用数字进行编码（例：男，0；女，1）。

⑶ 模型建立、训练与测试

选择全连接网络或卷积神经网络等任一方法构建模型，调整模型，寻找最佳方案，记录实验结果。

⑷ 可视化结果

编写代码，以可视化的形式，展示训练过程和结果。

4.2 拓展要求

将“基本要求 1-4”中需要重复执行的代码采用函数优化，并记录优化后完整的代码。

4.3 撰写项目文档

包括基于深度学习的泰坦尼克号旅客生存概率预测网络的结构与流程、实验过程及结果分析等。

C8 美国爱荷华州艾姆斯房价预测

本案例以 Kaggle 数据竞赛平台的“美国爱荷华州艾姆斯房价数据集”为基础，应用回归模型完成了对房价的预测。通过本案例的学习，可以掌握应用数据清洗、降维等数据分析方法，并应用回归模型完成预测任务。有利于扩展学生对课程内容的理解，通过动手实践和对实验结果的思考与改进，培养学生自主学习的能力、工程实践能力和分析总结能力，提升科研兴趣喝创新意识。

1 学习目标

⑴ 理解回归问题的基本原理；

⑵ 掌握数据缺失值处理、降维、数值化协方差等数据处理和数据分析方法；

⑶ 自主查阅资料，完成指定任务，培养培养自主学习、解决问题的能力。

2 背景

房价预测一直是人们持续关注的问题，而对于房价的影响因素中，除了外在经济环境以及政策调控之外，更关键的因素是房屋本身的硬件及软件设施的水平。

房价预测是一个典型的回归问题，其主要是根据房屋的相关特征，去科学估计未来的房屋价格信息数据，并使得它的误差最小。

本案例将运用来源于 Kaggle 数据竞赛平台的“美国爱荷华州艾姆斯房价数据集”构建神经网络模型，实现一个房价预测网络。

3 数据集

“美国爱荷华州艾姆斯房价数据集”来源于 Kaggle 数据竞赛平台，包括训练集(train.csv)和测试集(test.csv)。

训练集有 1460 行，每行数据包含 80 个属性和 1 个房价信息的标签。

测试集有 1459 行，每行数据包含 80 个属性，不含标签。

属性分为 38 个数值类型和 43 个类别类型。属性有建筑类型（MSSubClass）、一般分区类别（MSZoning）、街道距离（LotFrontage）、地块尺寸（LotArea）、道路通道的类型（Street）、胡同通道的类型（Alley）、其他配备（Utilities）等。

数据集下载地址:

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

4 要求

构建神经网络模型，使用 Kaggle 数据竞赛平台的“美国爱荷华州艾姆斯房价数据集”，训练并测试模型，完成对房价的预测。

4.1 基本要求

⑴ 下载数据集

下载 csv 格式的原始数据集，了解数据集的组成和特点；

了解数据集基本的缺失情况并可视化，了解离群点情况、异常值情况；

可以采用自定义缺失数据方法，实现缺失数据可视化。首先，获取各个特征缺失数据的总个数；接着，获取各个特征缺失数据的百分比；然后，通过类型搜索合并缺失特征数据；最后，输出缺失数据对象。

使用散点图观察数据之间的关系以及数据分布情况；

使用折线图观察 train.csv 房价（标签）走势；

⑵ 数据预处理

选择一种方法分别填充字符型数据和数值型数据中的缺失值（如均值、中位数填充等）；

数值化协方差热力图并观察特征的多重共线性问题；

消除多重共线性（如使用方差膨胀系数）；

⑶ 建立、训练和测试模型

编写代码，建立学习模型，使用美国爱荷华州艾姆斯房价数据集训练并测试模型。调整网络结构，寻找最佳方案，进行模型预测。

⑷ 可视化输出。

编写代码，以可视化的形式，展示训练过程和结果。

4.2 撰写项目文档

包括基于深度学习的基于深度学习的房价预测网络的结构与流程、实验过程及结果分析等。

提示：

计算各变量之间的相关系数:corr = data.corr()

画热力图:ax = sns.heatmap()

提示：导入计算膨胀因子的库：

from statsmodels.stats.outliers_influence import variance_inflation_factor

C9 基于卷积神经网络的表情识别

本案例实现了一个人脸表情识别案例。首先使用 Harr 分类器进行人脸检测，然后使用卷积神经网络实现表情识别。通过本案例的学习，可以拓展视野，提升工程实践能力和分析总结能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 理解 Harr 分类器的原理和应用；

⑵ 掌握卷积神经网络在表情识别中的应用；

⑶ 能够根据特定任务，独立查阅资料，编写和调试程序，分析对比实验结果，培养自主学习、研究的能力。

2 案例背景

人的脸部表情传递着一个人的喜怒哀乐，是人们与外界交互的一种重要手段，也是外界捕捉一个人情绪变化的重要依据。表情识别（Facial Expression Recognition, FER）是利用计算机作为辅助工具并结合特定的算法，通过人的面部表情对其内在情绪进行判断的一种技术手段，在日常生活中的各个领域都有着广泛的应用。

将人脸表情识别应用于医疗领域，在自闭症儿童的治疗过程中，表情识别可以用于辅助解读自闭症儿童的情感，帮助医生了解自闭症儿童的心理变化，从而制定更精准的治疗方案。将表情识别应用于教学领域，可以让教学系统捕捉和记录学生学习的情绪变化，为老师应材施教提供更好的参考。将表情识别应用于交通领域，可以用来判别飞行员或驾驶员的疲劳状态，通过技术手段避免交通险情的发生。将表情识别应用于日常生活，可以了解人们的精神状态。

人脸图像具有丰富的表情数据信息，如：眼睛睁大、眉毛皱起、嘴巴张大、嘴角上扬等，这些都是人心情通过面部的具象表达。本案例将表情识别视作一个七分类问题，分别对应生气、厌恶、高兴、害怕、正常、伤心、惊讶七种表情。本案例采用 OpenCV 的 haarcascade 分类器检测人脸部分，利用卷积神经网络来提取人脸表情特征，实现实时人脸表情识别。

3 基于卷积神经网络的表情识别原理

基于卷积神经网络进行人脸表情识别流程如下：

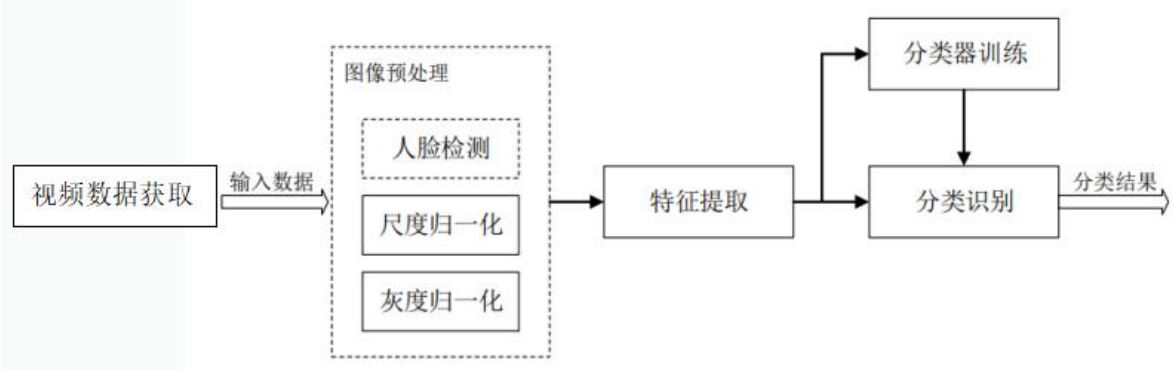


图 C9.1 表情识别流程图

在打开摄像头获取实时视频数据后，人脸检测是实现人脸表情识别的首要环节，使用 Opencv 库中自带的 Haar 分类器实现人脸的检测并架构面部表情图像区域与其他无关区域进

行分割，保留单一的面部区域，从而保证卷积神经网络模型表情识别的精准程度。

Haar 分类器以 xml 文档的形式保存于 OpenCV 安装目录下的 Lib\site-packages\cv2\data 路径，在使用时需要提前加载相应的 xml 文件。在人脸检测阶段需要加载 haarcascade_frontalface_default.xml 人脸正面检测器文件。在将获取的图像输入分类器前要进行灰度处理，使用 python 的切片功能按照分类器返回的人脸面部图像的区域对灰度图像进行切割，只保留面部区域，再对分割后得到的面部图像进行灰度处理并将大小调整为 48*48，之后才可以输入卷积神经网络进行分析。

初始卷积层对局部线条或图形边界进行数据特征提取，识别局部图像中基本的曲线、边界等内容；中间层级的卷积层将初始卷积层识别到的特征信息进行整合，实现人脸局部特征的识别，比如眼睛、嘴、鼻子等；较深层级的卷积层对眼睛、鼻子等人脸局部特征进行更上层整体分析判断，最终对人脸面部表情进行分类完成人脸的表情识别。

3.1 Haar 分类器算法

Haar 分类器=Haar-like 特征+AdaBoost 算法+级联+积分图快速计算

⑴ 使用 Haar-like 特征做检测。

⑵ 使用积分图（Integral Image）对 Haar-like 特征求值进行加速。

积分图（Integral image）是 Haar 分类器能够实时检测人脸的保证。积分图是一种能够描述全局信息的矩阵表示方法。

⑶ 使用 AdaBoost 算法训练区分人脸和非人脸的强分类器。

AdaBoost 是一种具有一般性的分类器提升算法，利用 AdaBoost 算法可以帮助选择更好的矩阵特征组合。

⑷ 使用筛选式级联把强分类器级联到一起，提高准确率。

通过组合多个强分类器来进一步处理图片信息，增加识别的准确率，级联的强分类器的复杂度逐渐增加来提高算法的识别准确度

3.2 卷积神经网络

卷积神经网络主要由输入层、卷积层、激活函数、池化层、全连接层组成。

⑴ 卷积层

卷积核是一系列的滤波器，用来提取某一种特征。用卷积核来处理一个图片，当图像特征与过滤器表示的特征相似时，卷积操作可以得到一个比较大的值，当图像特征与过滤器不相似时，卷积操作可以得到一个比较小的值。实际上，卷积的结果特征映射图显示的是对应卷积核所代表的特征在原始特征图上的分布情况。每个卷积核生成一个特征图，这些特征图堆叠起来组成整个卷积层的输出结果。卷积计算体现了参数共享和局部连接的模式。每个卷积核的大小代表了一个感受野的大小。

⑵ 池化层

依据图像局部相关性的原理，即局部相邻的像素点间存在一定的共性和联系，一定数值的像素点可以代表局部区域所有像素点的特征。常见的池化层思想认为最大值或者均值代表了这个局部的特征，从局部区域选择最有代表性的像素点数值代替该区域，可以减少数据处理量同时保留最具特征的数据信息。池化操作可以逐渐降低数据体的空间尺寸并保持一定的数据关联性，能有效减少网络中参数的数量，使得计算资源耗费降低，也能在一定程度上预

防过拟合。

⑶ 全连接层

全连接层将特征图转化为类别输出。全连接层不止一层，为了防止过拟合，会在各全连接层之间引入 DropOut 操作。

4 数据集

案例所使用的数据集 Fer2013 人脸表情库，该数据源于 kaggle 竞赛平台 2013 年的一个比赛项目。

该数据集共包含 28709 张训练图片和 7178 张测试图片，每一张都是 48×48 大小的灰度图片，每张人脸都或多或少位于图片居中位置。数据集中一共包含 7 种表情类别，用数字 0-6 表示，分别是：0=生气（Angry），1=厌恶(Disgust)， 2=害怕（Fear），3=开 心（Happy），4=难过（Sad），5=惊讶（Surprise），6=无表情（Neutral）。

图 C9.2 部分数据集展示

5 要求

5.1 基本要求

⑴ 构建项目工程，正确配置源代码运行环境，并成功运行源代码；

⑵ 输出表情识别结果

⑶ 实验结果分析

5.2 拓展要求

⑴ 采用不同于 Fer2013 的数据集进行测试，并分析实验结果。

⑵ 通过实验对于数据进行增强，分析实验结果，并客观评价实验结果。

5.3 撰写项目文档

包括基于卷积神经网络的表情识别的原理与流程、源代码说明文档、实验过程及结果分析等。

C10 基于深度学习的车牌识别

1 学习目标

⑴ 理解车牌识别的原理及其应用；

⑵ 自主查阅资料、编写车牌识别程序、培养自主学习、研究的能力。

2 案例背景

随着汽车数量在我国大幅度的增加，城市交通状况逐渐受到人们的重视，如何进行有效的交通管理更是成为了人们关注的焦点。针对此问题，人们运用新的科学技术，相继研制开发出了各种交通道路监视、管理系统。智能交通系统已成为世界交通领域研究的重要课题。车牌识别系统作为智能交通系统的核心，起着非常关键的作用。

车牌识别技术(Vehicle License Plate Recognition，VLPR) 是指能够检测到受监控路面的车辆并自动提取车辆牌照信息（含汉字字符、英文字母、阿拉伯数字及号牌颜色）进行处理的技术。

车牌识别是现代智能交通系统中的重要组成部分之一，应用十分广泛。它以数字图像处理、模式识别、计算机视觉等技术为基础，对摄像机所拍摄的车辆图像或者视频序列进行分析，得到每一辆汽车唯一的车牌号码，从而完成识别过程。车牌识别为停车场收费管理，交通流量控制指标测量，车辆定位，汽车防盗，高速公路超速自动化监管、闯红灯电子警察、公路收费站等等功能提供了技术支持。对于维护交通安全和城市治安，防止交通堵塞，实现交通自动化管理有着现实的意义。

基于传统的数字图像处理的车牌识别技术对于车牌位置相对固定、光线条件良好等理想环境下的车牌识别较为准确，然而在异常天气、背景复杂、车牌倾斜等复杂场景下的车牌识别仍存在可以改进的空间。为提升复杂场景下的车牌识别的性能，本案例提出了基于深度学习的车牌检测与识别方法，从而极大的提升车牌检测与识别的准确率。

2 基于深度学习的车牌识别实现过程

⑴ 车牌图像获取

车牌图像获取是进行车牌识别的首要环节，车牌图像可以从摄像机拍摄的车辆图像或者视频图像中进行抽取，车牌图像的获取也可由用户手机拍摄后传入车牌识别系统。

⑵ 车牌区域定位

车牌区域的定位通常采用基于形状的方法，因为车牌都是固定的矩形形状，通过首先寻找图像上拥有矩形特征的区域，然后再抽取这些区域，再结合车牌的长宽的比例特征可以筛选出相应的矩形区域，从而实现对车牌的准确定位。

本案例的车牌定位模块依赖于 HyperLRP 已经训练好的模型文件 cascade.xml，该文件是一个由基于 OpenCV 的 Haar 级联分类器训练出来的模型。HyperLRp 是一个开源、基于深度学习的高性能中文车牌识别库。

图 C10.1 车牌的灰度图像

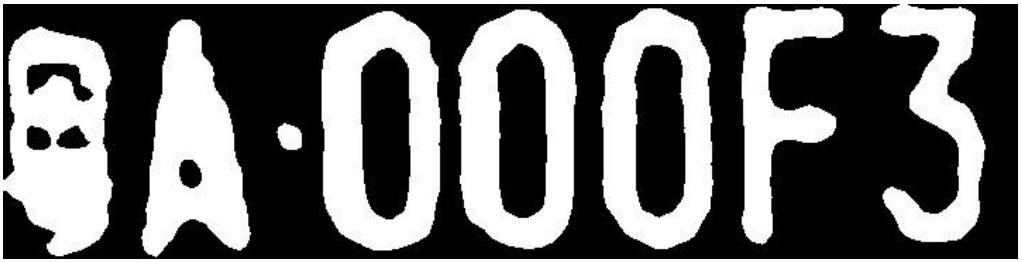


图 C10.2 车牌的二值图像

⑶ 单个字符切割

完成牌照区域的定位后，再将牌照区域分割成单个字符，然后进行识别。字符分割一般采用垂直投影法。由于字符在垂直方向上的投影必然在字符间或字符内的间隙处取得局部最小值的附近，并且这个位置满足牌照的字符书写格式。利用垂直投影法对复杂环境下的汽车图像中的字符分割有较好的效果。

图 C10.3 车牌单个字符

⑷ 车牌字符模块识别

对车牌中的省份简称、城市代号、车牌编号均使用两个卷积层加一个全连接层进行训练。

3 数据集

本案例为学习者提供了车牌识别与检测数据集，包括训练车牌检测模型数据和训练字符识别模型数据。其中训练字符识别模型数据包括数字：0-9；字母 A-Z；省市简称：京、津、晋、冀、蒙、辽、吉、黑、沪、苏、浙、皖、闽、赣、鲁、豫、鄂、湘、粤、桂、琼、川、贵、云、藏、陕、甘、青、宁、新、渝。部分数据集如下所示：

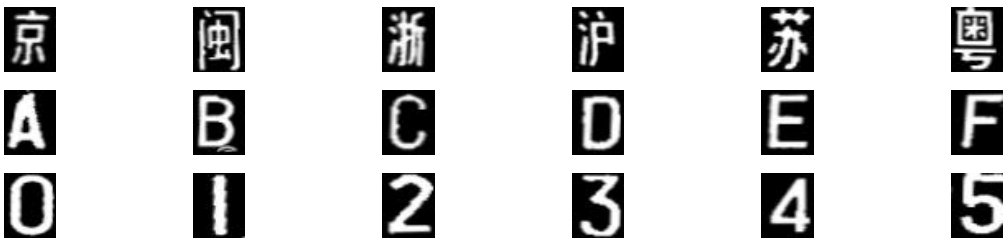


图 C10. 4 部分数据集

数据集百度网盘链接：https://pan.baidu.com/s/1bagitzF-MjIp1CN3DImHVg?pwd=1234

5 要求

5.1 基本要求

⑴ 构建项目工程，编写车牌识别代码，成功识别车牌；

⑵ 识别结果分析：选取不同环境下的车牌识别结果进行对比，并分析程序。

5.2 拓展要求

⑴ 实现对车牌颜色的检测，从而识别出蓝牌、黄牌、绿牌汽车；

⑵ 任选一种工具设计可视化界面，至少包括获取图像、识别结果结果显示两大部分，要求界面整齐美观，布局合理。

5.3 撰写项目文档

包括车牌识别流程、源代码说明文档、实验过程及结果分析等。

C11 基于 CNN 的手写汉字识别

本案例使用卷积神经网络搭建了手写汉字识别网络并完成了对汉字库中汉字的识别。案例主要介绍了汉字识别网络的基本结构，对汉字识别代码的结构和关键部分进行了解析，并提供了汉字识别网络的评估结果和识别实例。

通过本案例的学习，可以增强对卷积神经网络工作原理的了解；通过实现具体的手写汉字识别实例，从而培养学生自主学习的能力、工程实践能力和分析解决问题的能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 了解卷积神经网络的工作原理；

⑵ 能够使用使用 ImageDataGenerator 进行数据增强。

⑶ 能够根据特定任务，自主查阅资料，分析问题并解决问题。

2 案例背景

汉字作为记录科学知识,记载历史事件,传播哲学思想的重要载体,在中国历史中一直扮演着十分重要的作用。

随着信息技术和网络技术的进步，手写汉字识别在各类使用汉字作为信息传递的应用场景中有非常巨大的潜在需求，因此成为众多学者研究的热点之一。从 70 年代开始至今，汉字识别研究经过了多年研究，从最简单的印刷体，过渡到手写体识别、从单机版的脱机识别过渡到连接云端的联机识别、从单个汉字的识别过渡到长篇的识别，汉字识别技术已经应用在了车牌识别，支票签字识别等多个领域，明显提高了工作效率。

汉字相比于数字和英文字母来说,具有千变万化的特点。利用神经网络强大的特征提取能力和学习性强的优点作为手写汉字识别的方法,可以提高文字的识别准确率，达到良好的识别效果。

3 手写汉字识别数据集 HWDB

HWDB 是一个手写汉字数据集，来自于中科院自动化研究所，一共有三个版本，分别为 HWDB1.0、HWDB1.1 和 HWDB1.2。各版本的详情如图 C11.1 所示。这里我们选择 HWDB1.1 版本的数据集。

Table 1. Statistics of offline isolated character datasets

Dataset	#writers	#character samples		
		total	Symbol	Chinese/#class
HWDB1.0	420	1,680,258	71,122	1,609,136/3,866
HWDB1.1	300	1,172,907	51,158	1,121,749/3,755
HWDB1.2	300	1,041,970	50,981	990,989/3,319
Total	1,020	3,895,135	173,261	3,721,874/7,185

C11.1 HWDB 手写汉字数据集

HWDB1.1 来自 300 个不同的人，我们其中的 3755 种汉字样本作为本案例手写汉字识别的数据集，其中将每个字的 240 个样本作为训练集，而其他 60 个样本作为测试集。

HWDB 数据集官方下载地址：

http://www.nlpr.ia.ac.cn/databases/handwriting/Offline_database.html

Item	Type	Length	Instance	Comment
Sample size	unsigned int	4B		Number of bytes for one sample (byte count to next sample)
Tag code (GB)	char	2B	"啊"=0xb0a1 Stored as 0xa1b0	
Width	unsigned short	2B		Number of pixels in a row
Height	unsigned short	2B		Number of rows
Bitmap	unsigned char	Width*Height bytes		Stored row by row

C11.2 HWDB 手写汉字数据集格式

官方下载的数据格式为自定的 gnt 格式。包括图像尺寸，字符编码，宽度，高度，以及图像五个部分组成。在官方文档中提供了一个软件来可视化数据集和一份 C++代码帮助将.gnt 图像的格式转换为 png 格式的图像。

Kaggle 数据集

在 Kaggle 网站上，有人将其中的汉字图片单独提取出来组成了数据库，并标上序号。从 00000 标记到 03754，我们可以下载这个数据集到本地或者直接使用它在 Kaggle 平台上进行训练。

HWDB1.1 训练集：

https://www.kaggle.com/datasets/z1518787384/hwdb-train

HWDB1.1 测试集：

https://www.kaggle.com/datasets/z1518787384/hwdb-test

下图是数据集“碧”字的部分样本。

C11.3 HWDB“碧”字部分样本

4 要求

4.1 基本要求

(1) 使用卷积神经神经网络构建手写汉字识别模型。

(2) 使用 HWDB1.1 数据集完成模型训练和测试，并输出结果。

(3) 应用汉字识别模型：将手写汉字图片导入模型，进行分类，完成对手写汉字的识别。

4.2 拓展要求

(1) 调整汉字识别模型网络结构和超参数，对比实验结果并进行分析。

(2) 使用 ImageDataGenerator 类中多种方法，为汉字识别模型增加数据增强处理

4.3 撰写项目文档

包括基于 CNN 手写汉字识别的简介、原理与方法，程序流程与源码解析、实验过程及结果分析等。

C12 基于 TensorFlow.js 的人体姿态估计

本案例使用 TensorFlow.js 搭建 PoseNet 模型并完成了对人体姿态的识别。案例介绍了 tf.js 的相关知识，对 PoseNet 的结构和关键部分进行了解析，并提供了 PoseNet 的演示实例。通过本案例的学习，可以帮助学生掌握 TensorFlow.js 开发，培养学生自主学习的能力、工程实践能力和分析解决问题的能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 了解 TensorFlow.js 的工作原理和相关应用。

⑵ 了解人体姿态估计，能够部署并运行 PoseNet。

⑶ 培养学生自主查阅资料、分析问题并解决问题的能力。

2 案例背景

TensorFlow.js 是一个开源硬件加速 JavaScript 库，用于训练和部署机器学习模型。PoseNet 姿态估计是一项检测图像和视频中人物的计算机视觉技术，例如可以确定某人的肘部出现在图像中的位置。PoseNet 既可估算单个姿势，也可估算多个姿势。PoseNet 运行在 TensorFlow.js 上，任何拥有摄像头的 PC 或手机的人都可以在网络浏览器中体验这种技术。这种技术不会探测任何个人身份信息，只是估算关键身体关节的位置。现要求在基于 TensorFlow.js 的 PoseNet 基础之上，进行二次开发，增加一个功能性模块，实现模型的应用。

3 PoseNet 简介

输入：RGB 图片或者视频

输出：姿势的置信度得分和一个关键点数组，数组中包括每个关键点的名称、置信度分数和位置。其中关键点（keypoint）如图 1 所示，在估算人体姿势的时候，PoseNet 为人体选择了 17 个关键点；关键点位置（keypoint positions）表示为关键点的 x 和 y 的坐标值。

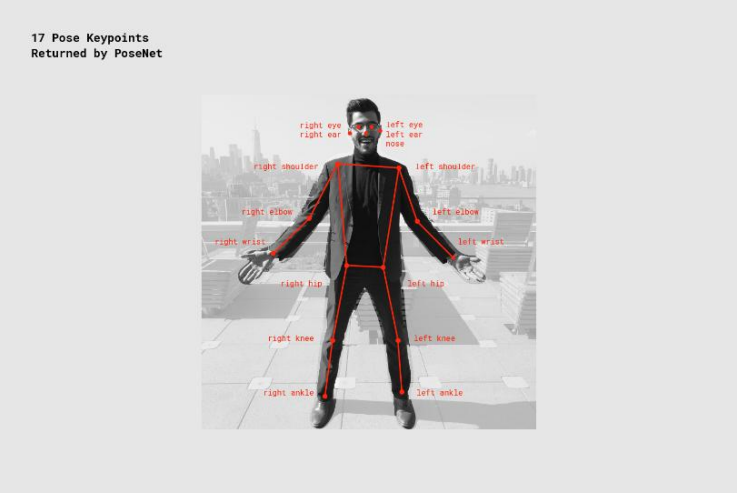


图 C12.1 人体姿态识别关键点示意图

PoseNet 人体姿态估计分为单人姿态估计和多人姿态估计。在实现两种姿态估计之前，首先要加载预先训练好的 PoseNet 模型。其中包括 MobileNet v1 架构和 ResNet50 架构。

使用 posenet.load()函数进行调用，函数中包含参数如表 C12.1 所示。

表 C12.1 posenet.load()函数参数说明

参数名称	参数说明
Architecture	该参数取值为 MobileNetV1 或 ResNet50，它确定要加载的 PoseNet 模型。
inputResolutin	该参数指定图像输入到 PoseNet 模型之前的图像大小。值越大，输入图像越大，精度越高，速度越慢。如果提供数字，图像将被调整大小并填充为具有相同宽度和高度的正方形。如果提供 width 和 height，将调整图像大小并将其填充到指定的宽度和高度。
outputStride	该参数指定了 PoseNet 模型的输出步幅。MobileNetV1 支持步幅 8、16、32；ResNet50 支持步幅 16、32。步幅越大，分辨率越低，速度越快。
multiplier	该参数影响 feature map 的数量。有多少个卷积核，经过卷积就会产生多少个 feature map。取值为 1.0、0.75 或 0.5。数值越高，精度越高，但速度越慢;数值越低，速度越快，但精度越低。
quantBytes	该参数影响 ResNet50 模型的权重量化。可用的选项有 1 字节、2 字节和 4 字节。通常是将 32 位浮点数量化为 8 位、16 位或 32 位整型。数值越高，模型尺寸越大，加载时间越长;数值越低，加载时间越短，但精度越低。

单个姿态估算相较于多人姿态估计更简单快速，但要求帧图像中只有一人。 如果图像中有多个人，那么来自两个人的关键点可能被估计为同一个单一姿势的一部分。例如，第一个人的左臂和第 2 个人的右膝可能会混淆。

PoseNet 官方 Demo　https://storage.googleapis.com/tfjs-models/demos/posenet/camera.html

4 要求

4.1 基本要求

(1) 查看官方示例，进入官方 demo 链接，尝试改变每个参数的值，观察效果。

(2) 在本地运行 PoseNet 的 demo，学习源代码并了解其中文件、主要函数、参数以及调用 API 的含义。

4.2 拓展要求

(1) 应用 PoseNet 模型，开发一款基于 TensorFlow.js 的应用实例。

4.3 撰写项目文档

包括基于基于 TensorFlow.js 的 PoseNet 模型应用简介、原理与方法，程序流程与源码解析、实验过程及结果分析等。

C13 基于 YOLOv5 的农田杂草识别

本案例实现了一个基于 YOLOv5 的田间杂草自动识别模型。案例内容包括数据集的制作、YOLOv5 的网络结构等，并对核心源码进行了解析，指导学生完成训练集的制作、建立和训练模型、以及模型评估等。通过本案例的学习，可以使学生理解目标检测的基本原理和应用，拓展视野，培养自主学习的能力、工程实践能力和分析总结能力，提升科研兴趣和创新意识。

1 学习目标

⑴ 理解图像是如何传输至神经网络的，并且思考卷积神经网络是如何进行提取图片特征值操作的。

⑵ 阅读 YOLOv5 相关文献、调试运行开源程序、自主尝试建立数据集、培养自主研究的能力。

2 案例背景

在世界范围内，每年因杂草危害造成的粮食损失约为 13.2%，相当于每年 10 亿人的口粮。杂草不仅会和农作物争夺水分、还会吸取农作物的养分，会严重影响农作物的生长速度，影响农作产品的质量，综上所述，杂草严重地阻碍了农业生产的发展。

田间杂草的生长严重影响了农作物的产量和质量，目前我国大多数地区对于杂草的清除和预防仍停留在传统的大量农药喷洒的阶段，效率低下且会造成土壤的严重污染。因此，能否研究出高效且轻便的杂草目标检测算法是将杂草识别研究引入现实应用的关键。

本实验采用基于 YOLOv5 目标检测方法实现杂草识别技术，其具有更加轻量级且推理速度较快的优点。因而此类检测方法更加适用于农业机械自动化导航、机械自动化除草、农作物生长环境监测等嵌入式设备和移动设备上。

下面我们将尝试使用 YOLOv5 目标检测算法构建杂草识别模型，实现对田间杂草的目标检测功能。

3 案例实现步骤

⑴ 准备杂草数据集

根据所拍摄到的杂草图片，根据图中是否含有农作物对其进行训练集与测试集分类，并按照 3：1 的大小划分训练集与测试集。

⑵ 对杂草图片进行预处理

建立杂草数据集和预处理工作：为减小杂草图片参数，以及易于在之后的网络模型中提取杂草特征的工作，本案例采用对杂草图片进行旋转、缩放、灰度处理以及输入卷积神经提取特征等预处理，将处理过后的杂草图片数据运用 Labelimg 进行标签处理工作。

⑶ 训练模型的建立

搭建实验环境以及构建 YOLOv5 网络模型，搭建所需的实验运行平台，并确定迭代次数（epoch），批次（batch），权重文件等参数。

⑷ 模型评估

对建立的训练模型进行性能评估测试，主要评估参数有：Map、Precision、Recall、Loss。

⑸ 可视化输出：

尝试编写代码，进行可视化输出，包括静态的图片检测功能，以及实时的视频检测功能。

4 基于 YOLOv5 目标检测算法原理

在了解 YOLOv5 网络架构之前，首先了解一下卷积神经网络的结构以及作用。

卷积神经网络（CNN）是一类包含卷积计算且具有深度结构的神经网络，具有良好的特征抽取和分类输出能力，是深度学习的代表算法之一。同时，卷积神经网络具有较强的学习表征的能力，能够按其阶层结构对输入信息进行平移不变分类，因此也被称为“平移不变人工神经网络”。

卷积神经网络主要由三部分组成，分别是卷积层，池化层，和全连接层，如图 1 所示。卷积神经网络可以分为，一维卷积神经网络（Conv1d）、二维卷积神经网络（Conv2d）和三维卷积神经网络（Conv3d）。由于本案例所需要的输入为图片转化后的二维矩阵，所以采用二维卷积神经网络，同时二维卷积神经网络对图片具有良好的提取特征以及压缩功能。

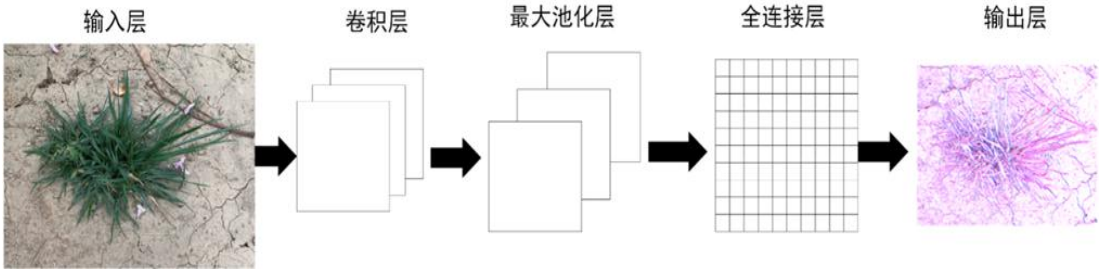


图 C13.1 卷积神经网络结构图

输入层：将图片输入后，还需转化成二维矩阵的格式，以便于卷积层的读取和功能的实现。

卷积层：主要作用是提取输入层图片的主要特征值，在本层需要设置输入通道个数，输出通道个数，以及卷积核的大小、滑动步长等参数。

最大池化层：主要作用是压缩图片的参数，本层需要设置的参数是最大池化层 kernel 的尺寸，Yolov5 中 SPPF 模块中主要用到的 kernel-size 为 5，9，13。

全连接层：全连接层的主要工作是将从前面网络所提取到的特征值输出到到样本标记空间中，即实现维度的转换。

输出层：将全连接层的输出图片进行叠加操作，与输入层的功能相反，输出层主要工作是将二维的矩阵转化为一张可视化的提取过特征值轮廓的图片。

图 C13.2 所示为 YOLOv5 的网络结构图，分为 input, Backbone，neck 和 head 四个部分。其中 Conv=conv2d+BN（Batch Normalization）+Silu，即经过一个普通的卷积层后归一化，在经过 Silu 激活函数输出。其基本网络架构如下：

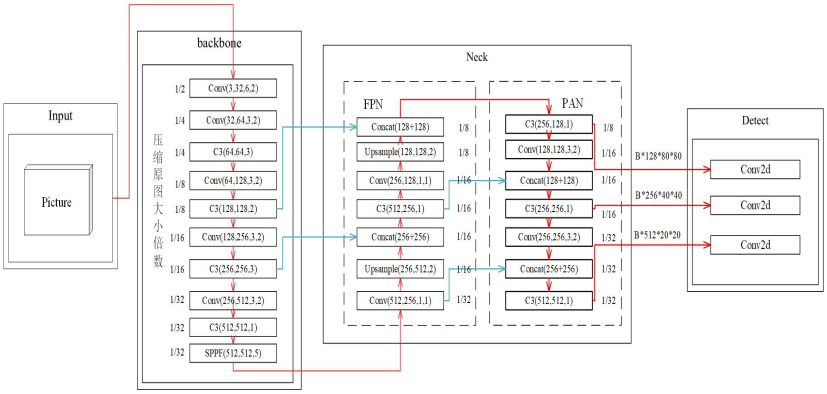


图 C13.2 YOLOV5 网络架构

输入层（Input）：在模型训练阶段，相比较于 YOLOv4 提出了一些改进思路，首先将输入图片在输入端进行 Mosaic 数据增强、自适应图片缩放、自适应锚框计算等数据增强操作

backbone 层：backbone 主要包括 Conv2d 卷积神经网络、C3 网络架构，SPFF 模块，其主要目的是提取特征值并减少图片参数，backbone 层的输出为提取特征值后并将原图缩小至原来的 1/32 的特征图片。其中 C3 架构主要作用为提取特征并提高网络中特征融合的作用，该模块是对残差特征进行学习的主要模块。SPPF 结构又叫空间金字塔，其主要工作为将前面提取特征后的任意大小的特征图转化为固定大小的特征向量。

Neck 网络：Neck 层采用了 FPN（Feature Pyramid Network, 特征金字塔）+PAN(Perceptual Adversarial Network)结合的方式，主要目的是将 backbone 中提取到的特征进行特征融合操作。

Detect：本层的主要工作是将之前网络结构中处理好的数据进行评估分析处理，即得出输入图形中物体的类别和边界框信息。

5 数据集的建立

本次课题图像获取以长茎尖叶杂草为主，主要拍摄作物为草莓和小麦中的杂草，拍摄时间主要为 2022 年 3 月至 5 月期间，时间段为每日 11：00-15：00 之间，采用设备为像素 800 万的智能手机，拍摄高度大约距离地面 30-50cm，大部分杂草图片没有农作物遮挡，并且色泽、光线、角度良好，其中准备杂草图片一共 230 张，部分图片如图 C13.3 所示，其中为了更加严谨丰富实验杂草数据集，本实验大体将图片分为四类，分别为：

第一类，只含有单一较为稀疏杂草图片，此类图片主要用于杂草训练集的制作。

第二类，含有大量较为密集的杂草图片，此类图片主要用于训练集的制作和部分测试集的制作。

第三类，同时含有农作物与杂草，且农作物与杂草相似度较低的杂草图片此类图片主要用于测试集的制作。

第四类，同时含有农作物与杂草，且农作物与杂草相似度较高的杂草图片，此杂草图片主要用于测试集的制作。

图 a 第一类杂草图片

图 c 第三类杂草图片

图 b 第二类杂草图片

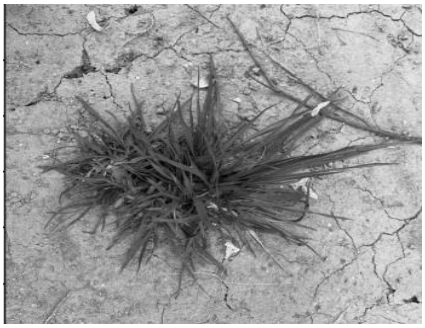
图 d 第四类杂草图片

图 C13.3 拍摄的部分杂草图片

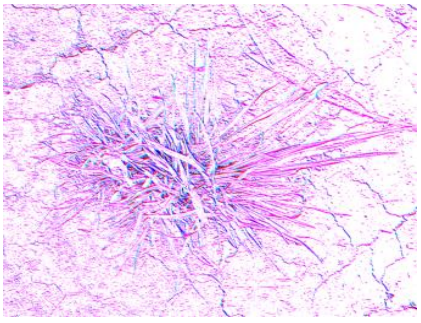
为了增强试验数据集的准确性与可行性，于是使用图像数据增强技术来扩充数据集的规模，降低杂草识别模型对某些图像属性的依赖，减少训练模型过拟合、加强模型的稳定性。主要方式包括随机进行逆时针 90°、180°、270°旋转，水平翻转，以及对图像进行灰度处理，以及将输入自搭建的卷积神经网络提取特征值并压缩后的输出图像放入杂草图片训练集的方式来进行数据增强。四种数据增强后的杂草图片如图 C13.4 所示。

旋转

缩放



进行灰度处理



进行卷积神经网络提起轮廓信息

图 C13.4 进行数据增强后的杂草图片

接下来将预处理过后的杂草图片放入 labelimg 进行标签处理操作，如图所示。其中紫色框为所做的标签区域，而右侧的 grass 选项则表示，框中目标的类别。

图 C13.5　labelimg 中进行标签处理

其中 YOLOv5 下的数据集有严格的文件路径格式，如下图所示。在数据集下必须只能含有两个文件夹且名称必须为 images，和 labels，其中两个文件夹分别存放杂草的图片信息和标签内容信息，并且每个文件下都含有两个子文件，分别为./images/train 和./imgaes/val

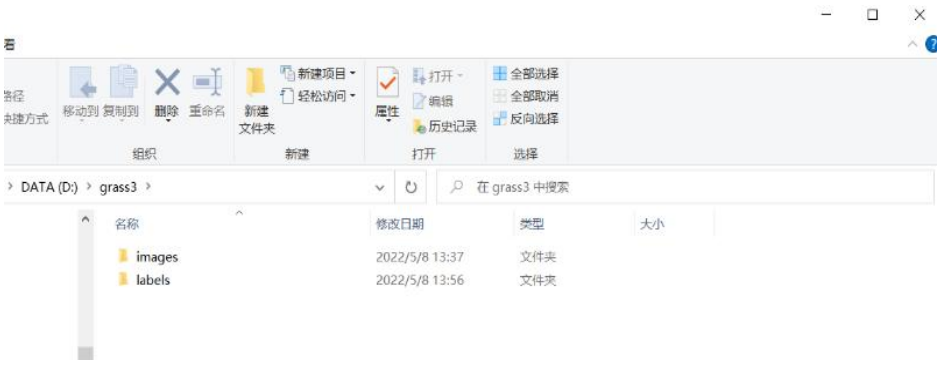


图 C13.6 数据集文件格式

分别存放杂草的训练集和测试集图片，labels 下的子文件也同理。

5 要求

5.1 基本要求

⑴ 构建项目工程，正确配置源代码运行环境，并成功运行源代码；

⑵ 实验结果分析：可以通过更改数据集，建立多组训练模型并进行对比，思考数据集的图片数量对模型影响是否较大。

5.2 拓展要求

⑴ 可以根据上文介绍，尝试自己制作数据集，也可以自己尝试其他的标签软件，并思考 labelimg 的优势在哪？尝试还有什么能够扩增数据集的图片增强方式。

⑵ 在理解 YOLOv5 网络架构和卷积神经网络的基础上，有没有什么办法可以优化网络架构，能够达到提升模型性能的效果，如利用注意力机制，或是采用更加轻量级的网络等。

⑶ 有条件的可以尝试使用 GPU 进行模型的训练和预测，解决实时视频检测时帧率较低的问题。

5.3 撰写项目文档

项目文档包括基于 YOLOv5 的农田杂草识别原理与流程、源代码说明文档、实验过程及结果分析等。