中国科学院大学
University of Chinese Academy of Sciences

课程作业报告

基于卷积神经网络的手写数字识别

作者姓名：陈子轩　2019E8015061013

学科专业：软件工程

所在单位：中国科学院大学计算机科学与技术学院

2020 年 5 月

基于卷积神经网络的手写数字识别

陈子轩

摘　　要

通过识别手写数字 MNIST 数据集，尝试构建五种不同的网络模型、从简单的仅有全连接层，逐步增加卷积层、池化层、Dropout 层，最终尝试了数据增强操作，取得了在测试集上正确率 99.68% 的成绩，最后分析了 32 张图片预测失败的原因。

1　介绍

(1) 本实验要解决的问题为识别手写数字。以 MNIST 数据集进行训练和测试。

(2) 从 1998 年到目前为止，http://yann.lecun.com/exdb/mnist/中记录了已发表的关于 MNIST 手写数字识别的论文。因为数据集很小，不存在识别效率的对比，所以正确率是衡量识别方法的唯一指标。在线性分类器、KNN、SVM、NN 和 CNN 中，目前识别率最高的模型是 CNN。

(3) 作为神经网络中的入门级实验，并没有进行特别复杂的设计。在使用全连接网络的基础上逐步增加卷积层、池化层、Dropout 层，尝试了数据增强，最终取得了在测试集上正确率 99.68% 的成绩。

2　深度学习平台及数据集

2.1　深度学习平台

TensorFlow == 2.0.0 + keras == 2.2.4 + plaidml == 0.7.0。

关于 plaidml 介绍见Machine Learning/AI on macOS Catalina with Metal GPU Support。这是一个能够使 A 卡也能用 GPU 加速的方法。使用时加入如下代码：

```
import plaidml.keras
plaidml.keras.install_backend()
import os
os.environ["KERAS_BACKEND"] = "plaidml.keras.backend"
import keras
```

2.2 数据集介绍

MNIST 数据集，包括了 60000 个训练样本和 10000 个测试样本，共 70000 张黑底白字手写数字图片。每张图片大小为 28×28 像素，图片中纯黑色像素值为 0，纯白色像素值为 1。使用 one_hot 编码后，数据集的标签是长度为 10 的一维数组，数组中每个元素索引号表示对应数字出现的概率。若未使用 one_hot 编码，则数据集的标签为标量，表示对应数字真实值。例如对于**图 1** 的样本，使用 one_hot 编码后该图片的标签为 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]，未使用 one_hot 编码，则该图片的标签为 8。

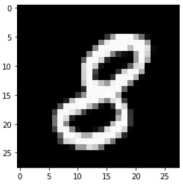


图 1: MNIST 数据集中的一个样本

3 实验结果和分析

3.1 实验 1

因为手写数字图片的大小是固定的，因此无需对输入图片进行 reshape 操作，直接以 28×28 = 784 维向量作为输入。输入图片编码方式为 one_hot = False。只加一层全连接层 Dense，128 个神经元，激活函数为 tanh 函数。输出层为 10 个神经元的分类，激活函数为 softmax 函数。损失函数为多分类交叉熵损失函数 sparse_categorical_crossentropy。指标设置为准确度 accuracy。网络结构设计如**图 2** 所示。模型信息见**表 1** 所示。

训练结果如**图 3** 所示。在训练集上训练 20 轮后，loss 值为 0.0102，accuracy 值为 0.9977；在测试集上 loss 值为 0.0438，accuracy 值为 0.9763。在训练集上训练 50 轮后，loss 值为 8.8444e-04，accuracy 值为 0.9998；在测试集上 loss 值为 0.0727，accuracy 值为 0.9753。可以看到虽然训练集上的 accuracy 值提高，甚至多增几轮后会到 1，但是在测试集上的 accuracy 值几乎没有变化。

此结果在激活函数换为 sigmoid 和 relu 函数后类似出现。

表 1: 模型信息

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 784)]	0
dense (Dense)	(None, 128)	100480
dense_1 (Dense)	(None, 10)	1290

Total params: 101,770

Trainable params: 101,770

Non-trainable params: 0

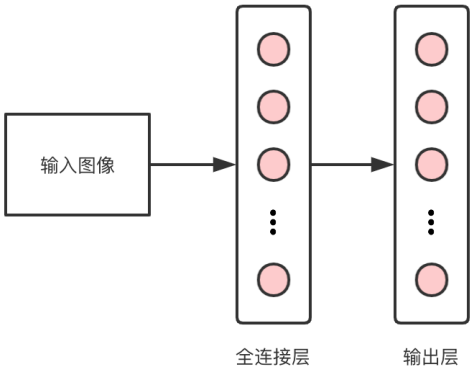


图 2: 网络结构设计

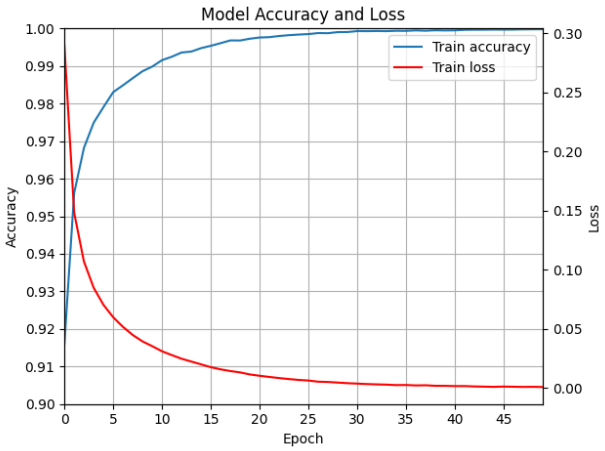


图 3: 训练结果

3.2 实验 2

采用两层全连接网络，并在每层全连接后加入 Dropout 层防止过拟合。依次加入

(1) 全连接层 fc1：512 个神经元，激活函数为 relu 函数；

(2) Dropout 层 do1：每个神经元被丢弃的概率 rate=0.2；

(3) 全连接层 fc2，同 fc1；

(4) Droupout 层 do2：同 do1；

(5) 输出层为 10 个神经元的分类，激活函数为 softmax 函数。

损失函数为多分类交叉熵损失函数 categorical_crossentropy。指标设置为准确度 accuracy。批大小设置为 128，轮数设置为 100，并从总训练集中划分出验证集 validation_splite = 0.2。网络结构设计见图 4 所示。模型信息如表 2 所示。优化器选择 rmsprop。训练过程中加入 callbacks=[checkpointer]，其中 checkpointer = tf.keras.callbacks.ModelCheckpoint("best_model/mnist_epoch:02d-val_accuracy:.2f.hdf5", save_weights_only=True, verbose=1)。训练结果如图 5 所示。在训练集上训练 100 轮后，训练集 loss 值为 0.0080、accuracy 值为 0.9989；验证集 loss 值为 0.3461，accuracy 值为 0.9825；测试集 loss 值为 0.1241、accuracy 值为 0.9837。显然，因为数据集过小且网络结构过于复杂，所以从训练结果看出此结果严重过拟合。

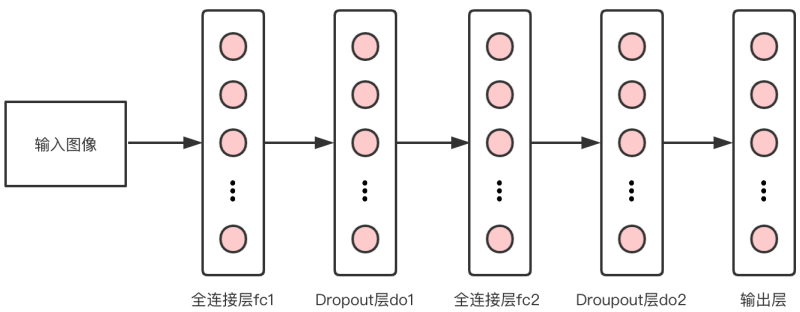


图 4: 网络结构设计

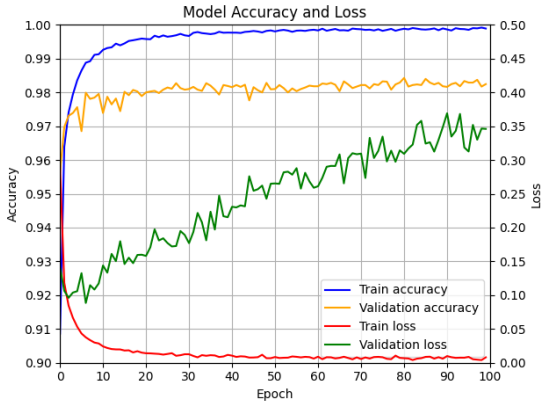


图 5: 训练结果

表 2: 模型信息

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 784)]	0
dense (Dense)	(None, 512)	401920
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 512)	262656
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 10)	5130

Total params: 669,706

Trainable params: 669,706

Non-trainable params: 0

3.3 实验 3

将输入图片进行 reshape 操作，变为 28×28 的数组，输入图片编码方式为 one_hot = True。加入

(1) 卷积层 conv1：16 个卷积核，尺寸为 3，步长为 2，激活函数为 relu 函数；

(2) 卷积层 conv2：32 个卷积核，尺寸为 3，步长为 2，激活函数为 relu 函数；

(3) 卷积层 conv3：64 个卷积核，尺寸为 3，步长为 2，激活函数为 relu 函数；

(4) 全连接层 fc1：128 个神经元，激活函数为 relu 函数；

(5) 输出层为 10 个神经元的分类，激活函数为 softmax 函数。

损失函数为多分类交叉熵损失函数 categorical_crossentropy。指标设置为准确度 accuracy。批次大小 batch_size 设置为 32。网络结构设计如**图 6** 所示。模型信息如**表 3** 所示。

训练结果如**图 7** 所示。在训练集上训练 20 轮后，loss 值为 0.0105，accuracy 值为 0.9972；在测试集上 loss 值为 0.0425，accuracy 值为 0.9879。在训练集上训练 50 轮后，loss 值 0。0070，accuracy 值为 0.9989；在测试集上 loss 值为 0.1679，accuracy 值为 0.9859。已经无法提高准确性。

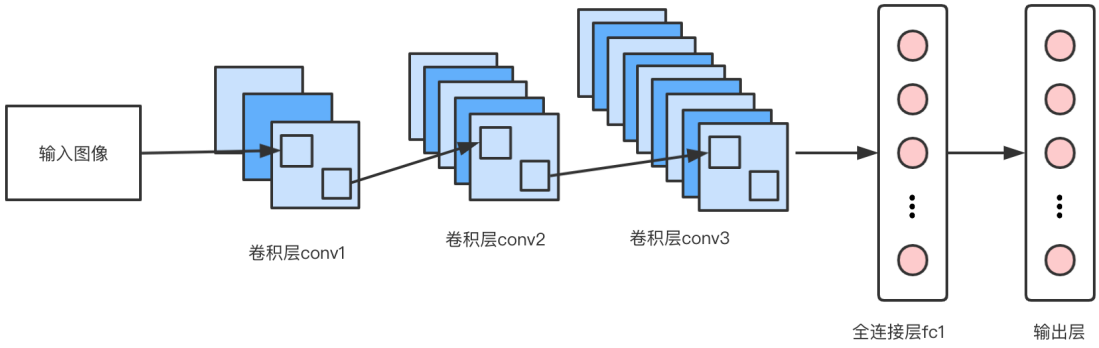


图 6: 网络结构设计

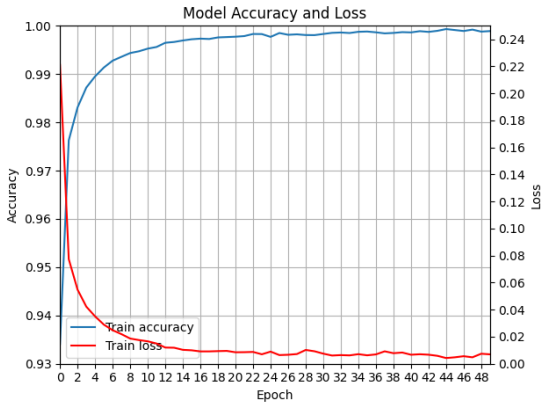


图 7: 训练结果

表 3: 模型信息

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 784)]	0
tf_op_layer_Reshape	[(None, 28, 28, 1)]	0
conv2d (Conv2D)	(None, 13, 13, 16)	160
conv2d_1 (Conv2D)	(None, 6, 6, 32)	4640
conv2d_2 (Conv2D)	(None, 2, 2, 64)	18496
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 128)	32896
dense_1 (Dense)	(None, 10)	1290

Total params: 57,482

Trainable params: 57,482

Non-trainable params: 0

3.4 实验 4

将输入图片进行 reshape 操作，变为 28×28 的数组，输入图片编码方式为 one_hot = False。加入

(1) 卷积层 conv1：32 个卷积核，尺寸为 3，步长为 1，激活函数为 relu 函数；

(2) 池化层 MaxPooling1：pool_size = (2, 2)；

(3) 卷积层 conv2：64 个卷积核，尺寸为 3，步长为 1，激活函数为 relu 函数；

(4) 卷积层 conv3：64 个卷积核，尺寸为 3，步长为 1，激活函数为 relu 函数；

(5) 池化层 MaxPooling2：pool_size = (2, 2)；

(6) 全连接层 fc1：64 个神经元，激活函数为 relu 函数；

(7) 输出层为 10 个神经元的分类，激活函数为 softmax 函数。

优化器选择 adam，损失函数为多分类交叉熵损失函数 sparse_categorical_crossentropy。指标设置为准确度 accuracy。网络结构设计见**图 8** 所示。模型信息如**表 4** 所示。训练过程中加入 callbacks=[checkpointer]。

训练结果如**图 9** 所示。在训练集上训练 20 轮后，loss 值为 0.0213，accuracy 值为 0.9927；在测试集上 loss 值为 0.0224，accuracy 值为 0.9868。在训练集上训练 50 轮后，loss 值 0.0213，accuracy 值为 0.9931；在测试集上 loss 值为 0.0297，accuracy 值为 0.9827。已经无法提高准确性。

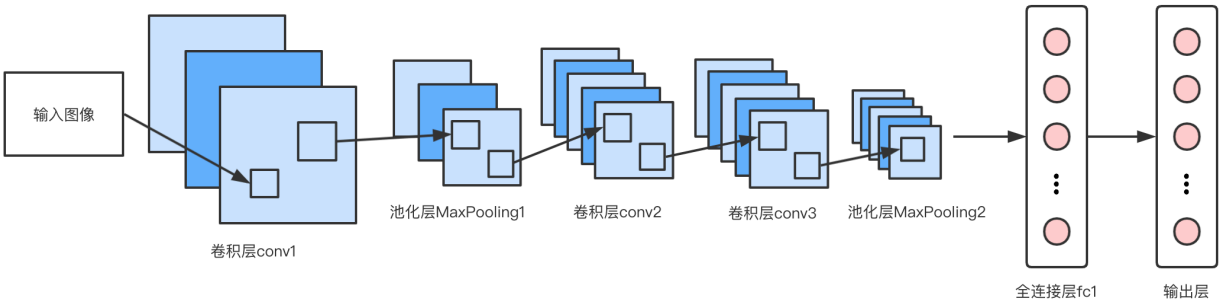


图 8: 网络结构设计

表 4: 模型信息

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d (MaxPooling2D)	(None, 13, 13, 32)	0
conv2d_1 (Conv2D)	(None, 11, 11, 64)	18496
conv2d_2 (Conv2D)	(None, 9, 9, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 4, 4, 64)	0
flatten (Flatten)	(None, 1024)	0
dense (Dense)	(None, 64)	65600
dense_1 (Dense)	(None, 10)	650

Total params: 121,994

Trainable params: 121,994

Non-trainable params: 0

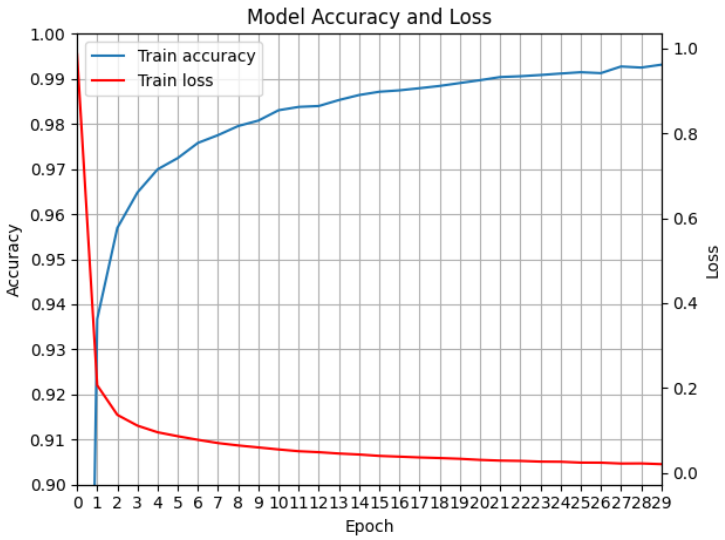


图 9: 训练结果

3.5 实验 5

将输入图片进行 reshape 操作，变为 28×28 的数组，输入图片编码方式为 one_hot = True。加入

(1) 卷积层 conv1：32 个卷积核，大小为 5，填充 padding = "same"，激活函数为 relu 函数；

(2) 卷积层 conv2：同 conv1；

(3) 池化层 MaxPooling1：pool_size = (2, 2)；

(4) Dropout 层 do1：每个神经元被丢弃的概率 rate = 0.25；

(5) 卷积层 conv3：64 个卷积核，大小为 3，填充 padding = "same"，激活函数为 relu 函数；

(6) 卷积层 conv4：同 conv3；

(7) 池化层 MaxPooling2：pool_size = (2, 2)，步长 strides = (2, 2)；

(8) Dropout 层 do2：同 do1；

(9) 全连接层 fc1：256 个神经元，激活函数为 relu 函数；

(10) 输出层为 10 个神经元的分类，激活函数为 softmax 函数。

损失函数为多分类交叉熵损失函数 categorical_crossentropy。指标设置为准确度 accuracy。优化器为 adam。网络结构设计见**图 10** 所示。模型信息如**表 5** 所示。数据增强设计为：

(1) 随机旋转角度范围 rotation_range = 10；

(2) 图片长宽放缩 zoom_range = 0.1；

(3) 水平位置平移 width_shift_range = 0.1；

(4) 上下位置平移 height_shift_range = 0.1。

表 5: 模型信息

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 32)	832
conv2d_2 (Conv2D)	(None, 28, 28, 32)	25632
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
dropout_1 (Dropout)	(None, 14, 14, 32)	0
conv2d_3 (Conv2D)	(None, 14, 14, 64)	18496
conv2d_4 (Conv2D)	(None, 14, 14, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(None, 7, 7, 64)	0
dropout_2 (Dropout)	(None, 7, 7, 64)	0
flatten_1 (Flatten)	(None, 3136)	0
dense_1 (Dense)	(None, 256)	803072
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 10)	2570

Total params: 887,530

Trainable params: 887,530

Non-trainable params: 0

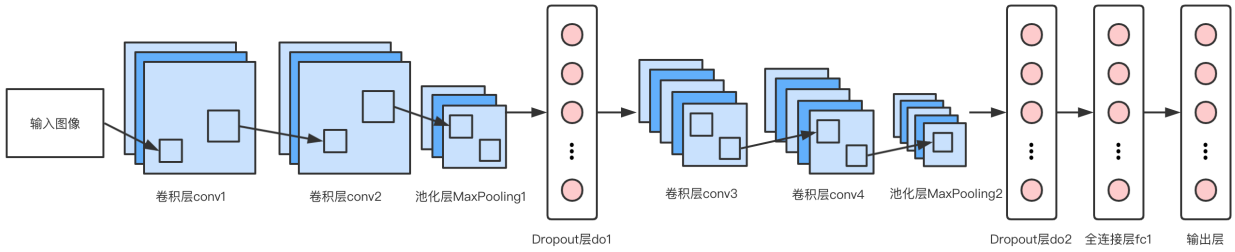


图 10: 网络结构设计

在数据增强中，steps_per_epoch = train_images.shape[0] // BS 整除保证了每轮均能完整遍历训练集中每一张图片。轮数 epochs 确定了新生成的数量，例如 epchs = 50，则可视为在同一张图基础上进行数据增强 50 次，生成 50 张新图送入网络训练。数据增强样例如**图 11** 所示。

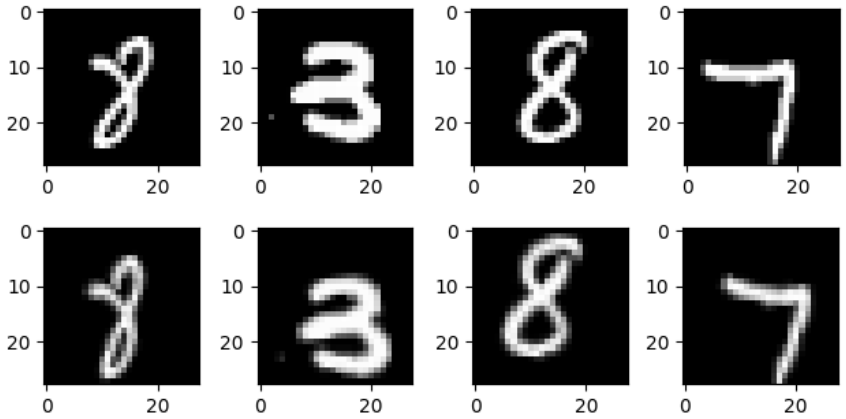


图 11: 数据增强样例

保持网络结构不变，进行如**表 6** 中的 12 个子实验。实验结果如**表 6** 及**图 12** 所示。对比蓝色线和绿色线、橙色线和红色线可以看出，在有数据增强的条件下训练出模型的准确率要明显高于没有数据增强的准确率；对比蓝色线和橙色线、绿色线和红色线，在轮数较高时训练效果更好，但 50 轮应该没有达到过拟合的程度，时间原因，没有尝试更高轮次；整体来看批大小 batch_size 可以认为是正相关于精确度，这可以理解为 batch_size 越大，每步训练中批次下降的方向越精确。

对训练 50 轮的子实验 12 中识别失败的图片进行显示，如**表 7** 所示。

可以看到，这些未能正确识别的图片里，只有少量（2182 号、3985 号、4823 号、6597 号、8408 号）可以被认为是网络确实没有预测正确，其他的图片均可以认为是图片本身的原因使得网络不可能预测正确。由此可以得出结论

(1) 该网络达到 99.68% 的正确率已足够高，但尚有 5 张图片可以预测正确的提升空间；

(2) 图片预处理阶段对网络识别很重要，如 582 号、6576 号、6597 号、6625 号存在明显的预处理过度造成数字部分残缺；

(3) 625 号、716 号、947 号、1901 号、3422 号、3558 号、4201 号、4740 号、5654 号、6576 号、

9679 号、9725 号共 12 张图可以认为是正常条件下不可能预测正确的图片，因此可以推断当前 MNIST 测试集的正确率上限为 99.78%，这一推断也符合当前错误率最低的模型 MCDNN[1] 所给出的错误率 0.23%。

表 6: 实验结果

编号	数据增强	批次大小 batch_size	轮数 epochs = 20				轮数 epochs = 50			
			训练集		测试集		训练集		测试集	
			loss	accuracy	loss	accuracy	loss	accuracy	loss	accuracy
1	×	15	0.0262	0.9931	0.0262	0.9931	0. 0279	0. 9942	0.0482	0.9939
2	×	30	0.015	0.9957	0.0247	0.9943	0.0133	0.997	0.0362	0.9943
3	×	60	0.0106	0.9965	0.0328	0.9926	0.0086	0.9978	0.0293	0.9942
4	×	120	0.0084	0.9973	0.0179	0.9948	0.0074	0.998	0.0268	0.9947
5	×	250	0.0087	0.9969	0.0179	0.9943	0.0042	0.9985	0.0235	0.9948
6	×	500	0.0095	0.9968	0.0182	0.9944	0.0038	0.9986	0.0227	0.9947
7	✓	15	0.0513	0.9862	0.0204	0.9944	0.0555	0.9855	0.0233	0.9937
8	✓	30	0.0371	0.9891	0.0158	0.9945	0.0327	0.9907	0.0195	0.9954
9	✓	60	0.0271	0.9917	0.017	0.9943	0.0223	0.9935	0.0191	0.994
10	✓	120	0.0257	0.9924	0.0131	0.9957	0.0174	0.9943	0.0154	0.9954
11	✓	250	0.0262	0.9919	0.0155	0.9945	0.0166	0.9949	0.0128	0.9957
12	✓	500	0.0286	0.9911	0.0121	0.9957	0.0159	0.9952	0.0101	0.9968

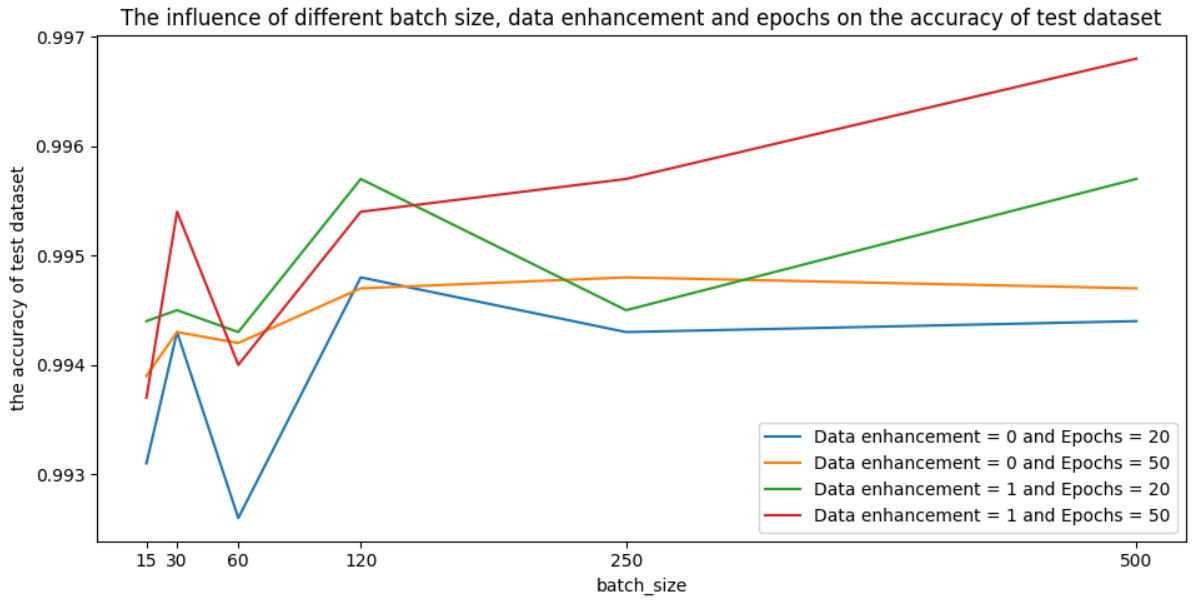
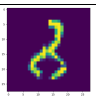
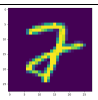
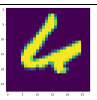
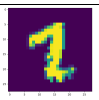
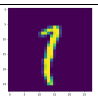
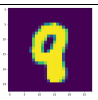
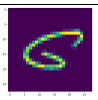
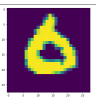
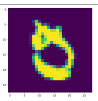
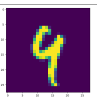
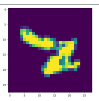
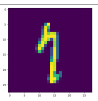
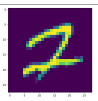
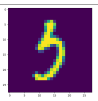
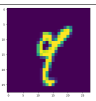
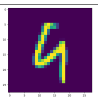
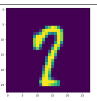
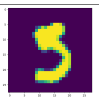
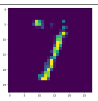
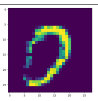
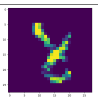
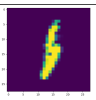
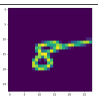
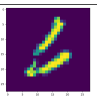
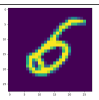


图 12: 实验结果

表 7: 32 张识别错误的输入图片

输入图片							
图片序号	582	583	625	659	716	947	1014
真实标签	8	2	6	2	1	8	6
网络预测	3	7	4	1	7	9	8
输入图片							
图片序号	1232	1260	1901	2182	2597	2939	3225
真实标签	9	7	9	1	5	9	7
网络预测	4	1	4	3	3	5	9
输入图片							
图片序号	3422	3558	3985	4176	4201	4504	4740
真实标签	6	5	9	2	1	2	3
网络预测	0	3	4	7	7	7	5
输入图片							
图片序号	4761	4823	5654	5937	6576	6597	6625
真实标签	9	9	7	5	7	0	8
网络预测	4	4	2	3	1	7	2
输入图片							
图片序号	8376	8408	9679	9725			
真实标签	1	8	6	5			
网络预测	6	5	2	6			

3.6 其他

采用实验 5 的子实验 12 的网络模型及训练模型文件，对 kaggle 竞赛中手写数字中测试集进行预测，将预测结果上传得到 Score 为 0.99885（如图 **13** 所示）。

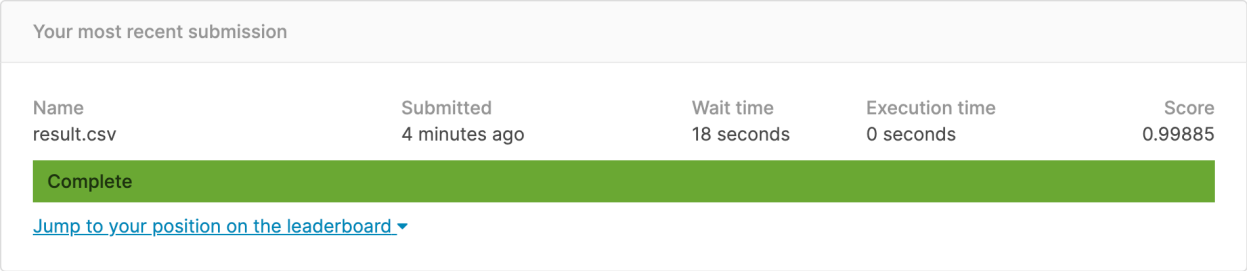


图 **13:** kaggle 中得分

4 结论

通过识别手写数字 MNIST 数据集，我第一次尝试构建网络、从简单的仅有全连接层，逐步增加卷积层、池化层、Dropout 层，最终尝试了数据增强操作，取得了在测试集上正确率 99.68% 的成绩，最后分析了 32 张图片预测失败的原因。

总结 5 个实验中正确率的最高值，如**图 14** 所示。可以看到，网络结构越复杂，最终得到的测试集正确率越高，也即可以认为复杂的在一定复杂程度内越复杂的网络可以越好的挖掘图像的特征。

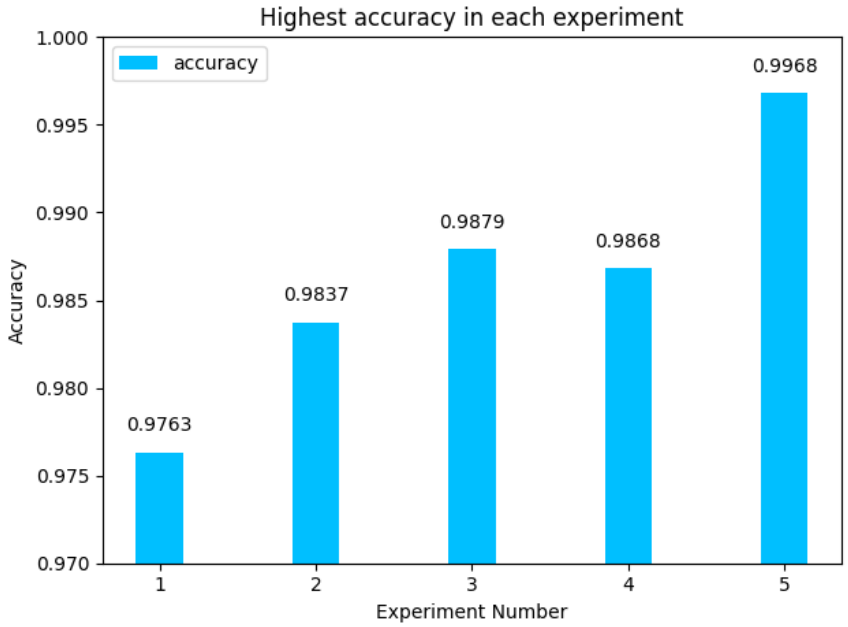


图 **14:** 5 个实验在测试集上最高正确率 accuracy 对比

参考文献

[1] Dan C. Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3642–3649, 2012.