清华大学电子工程系

媒体与认知 课堂 2

2023-2024 学年春季学期

作业 2

毕嘉仪 2024 年 4 月 11 日

理论部分

1 单选题（15 分）

1.1 C

1.2 D

1.3 D

1.4 C

1.5 B

2 计算题（15 分）

2.1 已知某卷积层的输入为 X(该批量中样本数目为 1，输入样本通道数为 1)，采用一个卷积核 W，即卷积输出通道数为 1，卷积核尺寸为 2×2，卷积的步长为 1，无边界延拓，偏置量为 b：

$$X = \begin{bmatrix} 0.5 & -0.2 & 0.3 \\ 0.6 & 0.4 & -0.1 \\ -0.4 & 0.5 & 0.2 \end{bmatrix}, W = \begin{bmatrix} 0.1 & -0.2 \\ -0.3 & 0.4 \end{bmatrix}, b = 0.04$$

2.1.1 请计算卷积层的输出 Y。

$$y_{11} = 0.05 + 0.04 - 0.18 + 0.16 + 0.04 = 0.11$$
$$y_{12} = -0.02 - 0.02 - 0.12 - 0.04 + 0.04 = -0.20$$
$$y_{21} = 0.06 - 0.08 + 0.12 + 0.20 + 0.04 = 0.34$$
$$y_{22} = 0.04 + 0.02 - 0.15 + 0.08 + 0.04 = 0.03$$
$$\therefore Y = \begin{bmatrix} 0.11 & -0.20 \\ 0.34 & 0.03 \end{bmatrix}$$

2.1.2 若训练过程中的目标函数为 L，且已知 $\frac{\partial L}{\partial Y} = \begin{bmatrix} 0.3 & 0.1 \\ -0.4 & 0.2 \end{bmatrix}$，请计算 $\frac{\partial L}{\partial X}$。

$$
\begin{aligned}
\frac{\partial L}{\partial X} &= \text{zero\_padded}(Y) * W^{\mathrm{T}} \\
&= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.11 & -0.20 & 0 \\ 0 & 0.34 & 0.03 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0.4 & -0.3 \\ -0.2 & 0.1 \end{bmatrix} \\
&= \begin{bmatrix} 0.011 & 0.104 & 0.004 \\ 0.001 & 0.039 & -0.086 \\ -0.102 & 0.127 & 0.012 \end{bmatrix}
\end{aligned}
$$

注：本题的计算方式不限，但需要提供计算过程以及各步骤的结果。

编程部分

3 编程作业报告

(1) 探究 batch normalization 和 dropout 的作用

1) 使用默认配置（不启用 BN 和 dropout），训练 baseline 模型:
经过多次试验与测试，发现默认参数配置下的模型效果就已经是最好的了。
训练模型：
input:

```
python train.py --ckpt_path checkpoints/default
```

output:

```
training ...
Epoch 01: loss = 2.726, accuracy on validation set = 0.248
Model saved in checkpoints/default\ckpt_epoch_1.pth

Epoch 02: loss = 1.923, accuracy on validation set = 0.431
Model saved in checkpoints/default\ckpt_epoch_2.pth

Epoch 03: loss = 1.284, accuracy on validation set = 0.581
```

```
Model saved in checkpoints/default\ckpt_epoch_3.pth

Epoch 04: loss = 0.857, accuracy on validation set = 0.727
Model saved in checkpoints/default\ckpt_epoch_4.pth

Epoch 05: loss = 0.485, accuracy on validation set = 0.802
Model saved in checkpoints/default\ckpt_epoch_5.pth

Epoch 06: loss = 0.305, accuracy on validation set = 0.821
Model saved in checkpoints/default\ckpt_epoch_6.pth

Epoch 07: loss = 0.207, accuracy on validation set = 0.844
Model saved in checkpoints/default\ckpt_epoch_7.pth

Epoch 08: loss = 0.117, accuracy on validation set = 0.858
Model saved in checkpoints/default\ckpt_epoch_8.pth

Epoch 09: loss = 0.050, accuracy on validation set = 0.885
Model saved in checkpoints/default\ckpt_epoch_9.pth

Epoch 10: loss = 0.077, accuracy on validation set = 0.835
Model saved in checkpoints/default\ckpt_epoch_10.pth

Epoch 11: loss = 0.058, accuracy on validation set = 0.867
Model saved in checkpoints/default\ckpt_epoch_11.pth

Epoch 12: loss = 0.034, accuracy on validation set = 0.887
Model saved in checkpoints/default\ckpt_epoch_12.pth

Epoch 13: loss = 0.028, accuracy on validation set = 0.881
Model saved in checkpoints/default\ckpt_epoch_13.pth

Epoch 14: loss = 0.032, accuracy on validation set = 0.879
Model saved in checkpoints/default\ckpt_epoch_14.pth

Epoch 15: loss = 0.042, accuracy on validation set = 0.890
Model saved in checkpoints/default\ckpt_epoch_15.pth
```

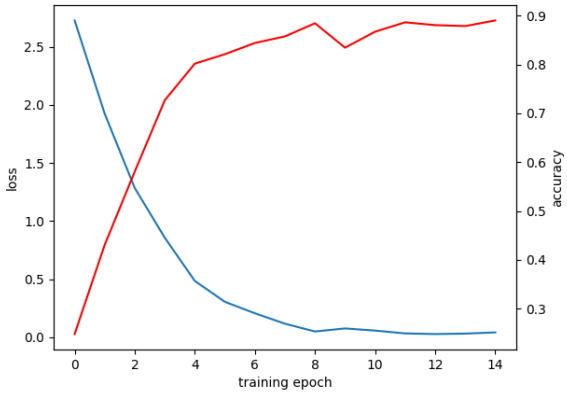

图 1

测试模型：

input:

```
python test.py --ckpt_path checkpoints/default --epoch 10
```

output:

```
[Info] loading checkpoint from checkpoints/default\ckpt_epoch_10.
    pth ...
accuracy on the test set: 0.806
```

2) 启用 batch normalization：

经过多次试验与测试，发现默认参数配置下的模型效果就已经是最好的了。

训练模型：

input:

```
python train.py --ckpt_path checkpoints/bn --bn
```

output:

```
training ...
Epoch 01: loss = 2.257, accuracy on validation set = 0.588
Model saved in checkpoints/bn\ckpt_epoch_1.pth

Epoch 02: loss = 0.854, accuracy on validation set = 0.835
Model saved in checkpoints/bn\ckpt_epoch_2.pth

Epoch 03: loss = 0.325, accuracy on validation set = 0.896
Model saved in checkpoints/bn\ckpt_epoch_3.pth
```

```
10
11 Epoch 04: loss = 0.129, accuracy on validation set = 0.931
12 Model saved in checkpoints/bn\ckpt_epoch_4.pth
13
14 Epoch 05: loss = 0.041, accuracy on validation set = 0.938
15 Model saved in checkpoints/bn\ckpt_epoch_5.pth
16
17 Epoch 06: loss = 0.014, accuracy on validation set = 0.958
18 Model saved in checkpoints/bn\ckpt_epoch_6.pth
19
20 Epoch 07: loss = 0.007, accuracy on validation set = 0.963
21 Model saved in checkpoints/bn\ckpt_epoch_7.pth
22
23 Epoch 08: loss = 0.005, accuracy on validation set = 0.960
24 Model saved in checkpoints/bn\ckpt_epoch_8.pth
25
26 Epoch 09: loss = 0.004, accuracy on validation set = 0.963
27 Model saved in checkpoints/bn\ckpt_epoch_9.pth
28
29 Epoch 10: loss = 0.003, accuracy on validation set = 0.960
30 Model saved in checkpoints/bn\ckpt_epoch_10.pth
31
32 Epoch 11: loss = 0.003, accuracy on validation set = 0.960
33 Model saved in checkpoints/bn\ckpt_epoch_11.pth
34
35 Epoch 12: loss = 0.002, accuracy on validation set = 0.960
36 Model saved in checkpoints/bn\ckpt_epoch_12.pth
37
38 Epoch 13: loss = 0.002, accuracy on validation set = 0.963
39 Model saved in checkpoints/bn\ckpt_epoch_13.pth
40
41 Epoch 14: loss = 0.002, accuracy on validation set = 0.960
42 Model saved in checkpoints/bn\ckpt_epoch_14.pth
43
44 Epoch 15: loss = 0.001, accuracy on validation set = 0.962
45 Model saved in checkpoints/bn\ckpt_epoch_15.pth
```

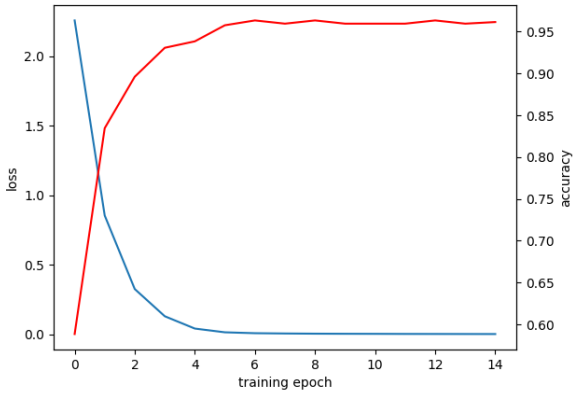

图 2

测试模型：

input:

```
python test.py --ckpt_path checkpoints/bn --epoch 10
```

output:

```
[Info] loading checkpoint from checkpoints/bn\ckpt_epoch_10.pth ...
accuracy on the test set: 0.962
```

分析：测试准确率明显提升，收敛速度也大大加快。

3) 启用 dropout 并设置概率为 0.3：训练模型：

input:

```
python train.py --ckpt_path checkpoints/dropout_epoch25 --dropout
    0.3 --epoch 25
```

output:

```
training ...
Epoch 01: loss = 2.881, accuracy on validation set = 0.206
Model saved in checkpoints/dropout_epoch25\ckpt_epoch_1.pth

Epoch 02: loss = 2.147, accuracy on validation set = 0.413
Model saved in checkpoints/dropout_epoch25\ckpt_epoch_2.pth

Epoch 03: loss = 1.667, accuracy on validation set = 0.498
Model saved in checkpoints/dropout_epoch25\ckpt_epoch_3.pth

Epoch 04: loss = 1.283, accuracy on validation set = 0.612
Model saved in checkpoints/dropout_epoch25\ckpt_epoch_4.pth
```

```
13
14 Epoch 05: loss = 0.928, accuracy on validation set = 0.715
15 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_5.pth
16
17 Epoch 06: loss = 0.656, accuracy on validation set = 0.817
18 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_6.pth
19
20 Epoch 07: loss = 0.474, accuracy on validation set = 0.800
21 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_7.pth
22
23 Epoch 08: loss = 0.380, accuracy on validation set = 0.833
24 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_8.pth
25
26 Epoch 09: loss = 0.268, accuracy on validation set = 0.863
27 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_9.pth
28
29 Epoch 10: loss = 0.177, accuracy on validation set = 0.869
30 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_10.pth
31
32 Epoch 11: loss = 0.153, accuracy on validation set = 0.879
33 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_11.pth
34
35 Epoch 12: loss = 0.134, accuracy on validation set = 0.888
36 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_12.pth
37
38 Epoch 13: loss = 0.103, accuracy on validation set = 0.908
39 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_13.pth
40
41 Epoch 14: loss = 0.062, accuracy on validation set = 0.896
42 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_14.pth
43
44 Epoch 15: loss = 0.083, accuracy on validation set = 0.929
45 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_15.pth
46
47 Epoch 16: loss = 0.070, accuracy on validation set = 0.919
48 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_16.pth
49
50 Epoch 17: loss = 0.047, accuracy on validation set = 0.931
51 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_17.pth
52
53 Epoch 18: loss = 0.050, accuracy on validation set = 0.931
54 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_18.pth
55
56 Epoch 19: loss = 0.105, accuracy on validation set = 0.921
57 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_19.pth
58
59 Epoch 20: loss = 0.107, accuracy on validation set = 0.906
60 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_20.pth
61
62 Epoch 21: loss = 0.067, accuracy on validation set = 0.912
63 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_21.pth
```

```
64
65 Epoch 22: loss = 0.042, accuracy on validation set = 0.938
66 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_22.pth
67
68 Epoch 23: loss = 0.014, accuracy on validation set = 0.933
69 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_23.pth
70
71 Epoch 24: loss = 0.045, accuracy on validation set = 0.917
72 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_24.pth
73
74 Epoch 25: loss = 0.043, accuracy on validation set = 0.925
75 Model saved in checkpoints/dropout_epoch25\ckpt_epoch_25.pth
```

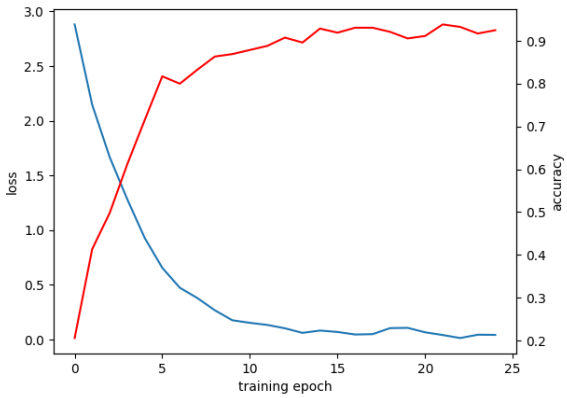

图 3

测试模型：

input:

```
1 python test.py --ckpt_path checkpoints/dropout_epoch25 --epoch 25
```

output:

```
1 [Info] loading checkpoint from checkpoints/dropout_epoch25\
      ckpt_epoch_25.pth ...
2 accuracy on the test set: 0.908
```

分析：测试准确率明显提升，收敛速度有所降低。

(2) 探究数据增广的作用

1) 数据增广变换 input:

```
python unit_test.py data_loader
```

output:

```
I H M H B K J X
Clipping input data to the valid range for imshow with RGB data
    ([0..1] for floats or [0..255] for integers).
```

图 4

其中，所使用的增广变换为

```
if mode == "train" and augment:
    data_transforms.append(transforms.RandomRotation(10))
    data_transforms.append(transforms.RandomPerspective(0.2))
```

选择原因：首先，数据图像已将路牌主体呈现在画面中央，因此 random crop 效果并不会好；其次，由于拍摄角度问题，路牌图像的旋转角度和透视效果不一，因此选择这两种 transform 可以实现比较合理的数据增广。

2) 模型训练与测试训练模型：

input:

```
python train.py --ckpt_path checkpoints/bn_aug --bn --augment --
    epoch 13
```

output:

```
training ...
Epoch 01: loss = 2.478, accuracy on validation set = 0.333
Model saved in checkpoints/bn_aug\ckpt_epoch_1.pth

Epoch 02: loss = 1.306, accuracy on validation set = 0.756
Model saved in checkpoints/bn_aug\ckpt_epoch_2.pth

Epoch 03: loss = 0.661, accuracy on validation set = 0.875
Model saved in checkpoints/bn_aug\ckpt_epoch_3.pth

```

```
Epoch 04: loss = 0.398, accuracy on validation set = 0.887
Model saved in checkpoints/bn_aug\ckpt_epoch_4.pth

Epoch 05: loss = 0.246, accuracy on validation set = 0.935
Model saved in checkpoints/bn_aug\ckpt_epoch_5.pth

Epoch 06: loss = 0.145, accuracy on validation set = 0.904
Model saved in checkpoints/bn_aug\ckpt_epoch_6.pth

Epoch 07: loss = 0.142, accuracy on validation set = 0.960
Model saved in checkpoints/bn_aug\ckpt_epoch_7.pth

Epoch 08: loss = 0.097, accuracy on validation set = 0.954
Model saved in checkpoints/bn_aug\ckpt_epoch_8.pth

Epoch 09: loss = 0.065, accuracy on validation set = 0.967
Model saved in checkpoints/bn_aug\ckpt_epoch_9.pth

Epoch 10: loss = 0.078, accuracy on validation set = 0.963
Model saved in checkpoints/bn_aug\ckpt_epoch_10.pth

Epoch 11: loss = 0.071, accuracy on validation set = 0.967
Model saved in checkpoints/bn_aug\ckpt_epoch_11.pth

Epoch 12: loss = 0.040, accuracy on validation set = 0.967
Model saved in checkpoints/bn_aug\ckpt_epoch_12.pth

Epoch 13: loss = 0.027, accuracy on validation set = 0.962
Model saved in checkpoints/bn_aug\ckpt_epoch_13.pth
```

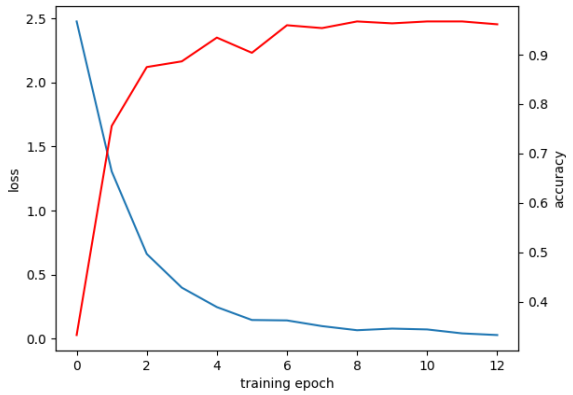

图 5

测试模型：

input:

```
python test.py --ckpt_path checkpoints/bn_aug --epoch 13
```

output:

```
[Info] loading checkpoint from checkpoints/bn_aug\ckpt_epoch_13.pth
      ...
accuracy on the test set: 0.967
```

(3) 探究空间变换网络 (STN) 的作用

(4) 可视化

分别输入以下命令：

```
python visualize.py --type filter --ckpt_path checkpoints/bn --
    layer_idx 0
python visualize.py --type filter --ckpt_path checkpoints/bn --
    layer_idx 1
python visualize.py --type filter --ckpt_path checkpoints/bn --
    layer_idx 2
python visualize.py --type filter --ckpt_path checkpoints/bn --
    layer_idx 3
python visualize.py --type filter --ckpt_path checkpoints/bn --
    layer_idx 4

python visualize.py --type feature --ckpt_path checkpoints/bn --
    layer_idx 0 --image_idx 50
python visualize.py --type feature --ckpt_path checkpoints/bn --
    layer_idx 1 --image_idx 50
python visualize.py --type feature --ckpt_path checkpoints/bn --
    layer_idx 2 --image_idx 50
python visualize.py --type feature --ckpt_path checkpoints/bn --
    layer_idx 3 --image_idx 50
python visualize.py --type feature --ckpt_path checkpoints/bn --
    layer_idx 4 --image_idx 50

python visualize.py --type tsne --ckpt_path checkpoints/bn

python visualize.py --type stn --ckpt_path checkpoints/stn
```

1) 可视化各卷积层的卷积核：

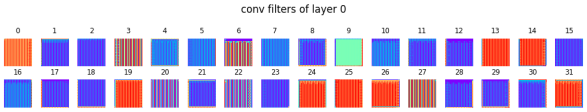
conv filters of layer 0

图 6

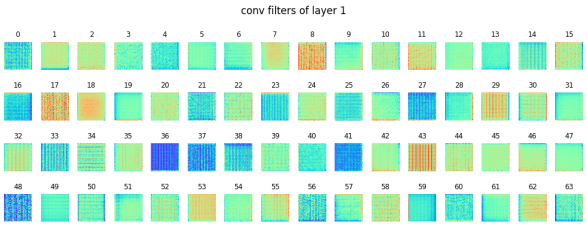
conv filters of layer 1

图 7

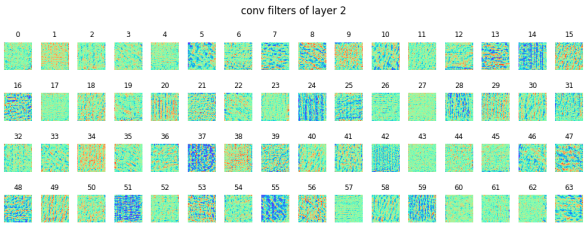
conv filters of layer 2

图 8

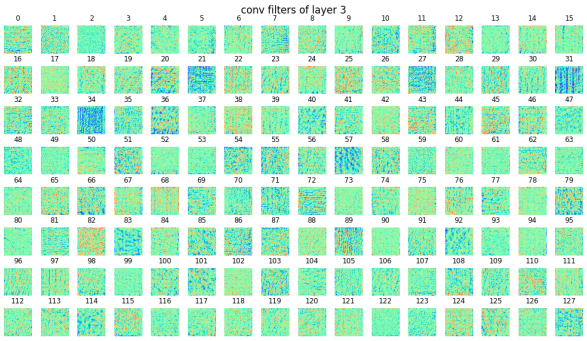
conv filters of layer 3

图 9

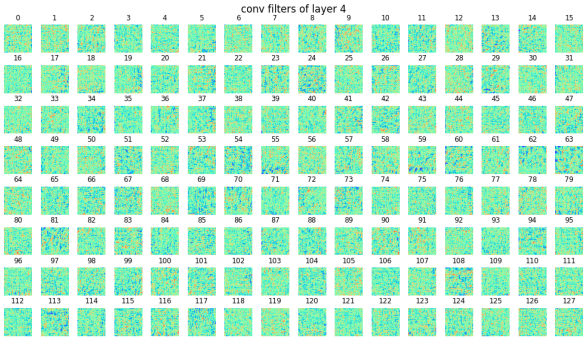

图 10

2) 可视化第 50 张图像个卷积层的输出特征图:

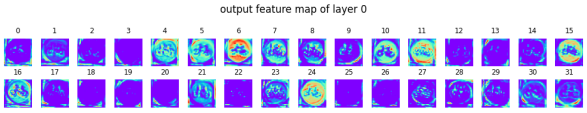

图 11

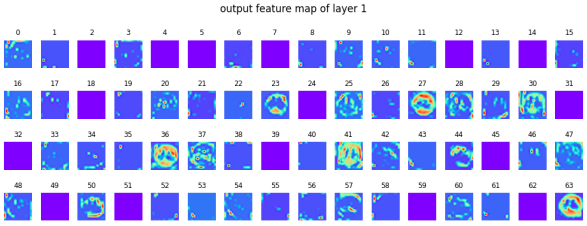

图 12

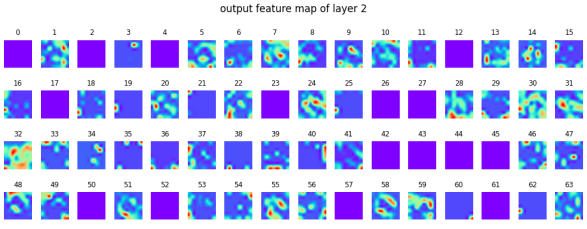

图 13

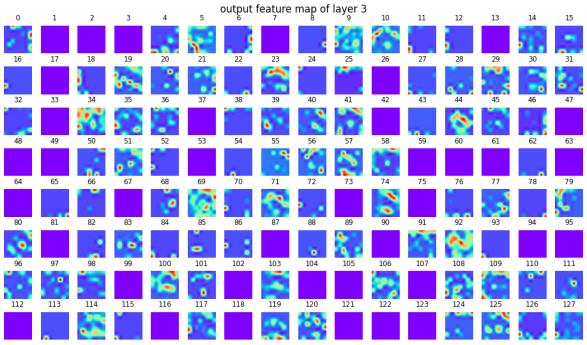

图 14

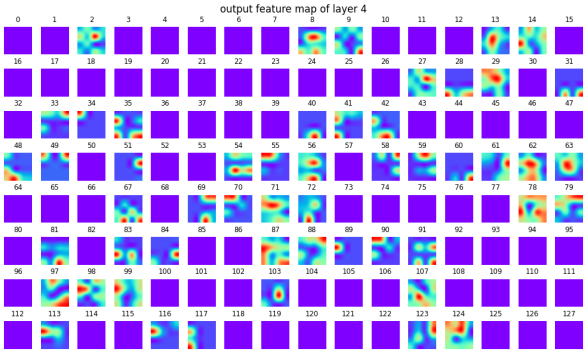

图 15

可见，随着卷积层数的增加，图像的不同特征不断被提取、分离、放大，在最后一张特征图中最为鲜明。

3) t-SNE 可视化最后一层隐藏层的输出特征：

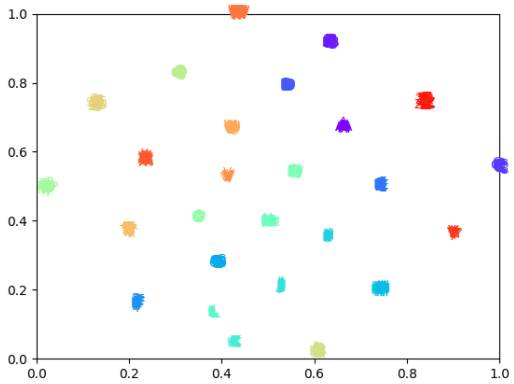

图 16

可见，各个类型的标牌被明确地分成了不同类别。

4) 可视化 STN 学习到的变换：

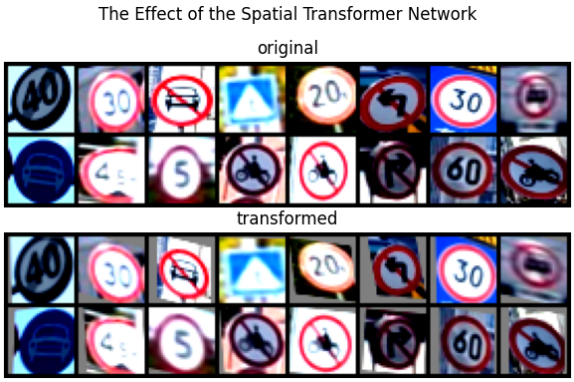

图 17

4 遇到的问题及解决方法

(1)

5 建议