Pattern-Recognition final

學號：311553032 NAME:年皓鳴

environment

I use colab to train my model, and use tourchvision to get model and transform the image pre processing

use model: Resnet

resnet is a cnn model, but the traditional cnn model with the increase of layers, the gradient of the previous layer may Gradient Vanishing/Exploding during the back propagation, which makes the network become very difficult to train.
But resnet uses Residual Mapping to overcome the Gradient Vanishing/Exploding problem in deep neural networks.
Finally, I tried to use resnet18 and resnet50 for this final project.

Hyperparameters

TASK 1

	resnet18	resnet50
epoch	100	150
batch size	100	32
optimize	Adam	Adam
loss	CrossEntropyLoss	CrossEntropyLoss
lr	1e-3	1e-3

TASK 2

	resnet18	resnet50
epoch	150	300
batch size	100	32
optimize	Adam	Adam
loss	MultiLabelSoftMarginLoss	MultiLabelSoftMargi
lr	1e-3	1e-3

TASK 3

	resnet18	resnet50
epoch	150	300
batch size	100	32
optimize	Adam	Adam
loss	MultiLabelSoftMarginLoss	MultiLabelSoftMargi
lr	1e-3	1e-3

LOSS FINCTION : MULTILABELSOFTMARGINLOSS

Because the later task2 task3 will need to generate multiple letters and to order the results, but if you use the arrangement of the combination of words, because the number of labels may be too much may lead to the accuracy rate is not good，so I used MultiLabelSoftMarginLoss to predict the position of the letters while predicting the letters.

Preprocessing

I also used the following three pre-processing methods in this project to try to see if the effect of different pre-processing is good for the training of the model and to have good results,so I tries to use original 、gray and LoG filiter to train my model.

ORIGINAL

In the original picture, I only used normalization to process my picture because it would have the following benefits: Uniform data range, faster learning, improved model performance

```
transform_1= transforms.Compose([
    transforms.ToPILImage(),
    transforms.Resize(256),
    transforms.CenterCrop(224),

    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.2
])
```

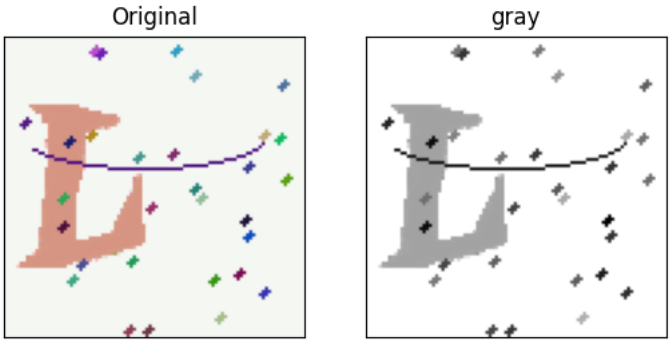
RGBTOGRAY

The original image is in color, but I think color is not a very important feature in this operation, so I converted the image to grayscale to try to see if it would improve its accuracy.

```
transform_1= transforms.Compose([
    transforms.ToPILImage(),
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.Grayscale(),

    transforms.ToTensor(),

    transforms.Normalize(mean=[0.485], std=[0.229]),
])
```



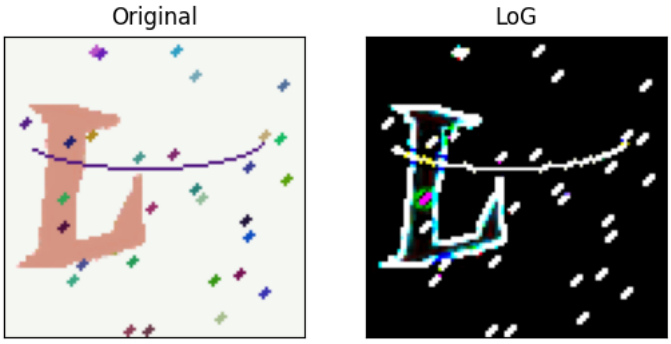
LoG FILITER

First, I used GaussianBlur to remove the noise, and then I used Laplacian to remove the outline of the object. I originally wanted to say that I could remove the noise from the non-letter part of the photo smoothly, but probably because my sigma was not set large enough, the result of the picture did not remove the noise smoothly.

```
from scipy.ndimage import gaussian_laplace
class LoGFilter(object):
    def __init__(self, sigma):
        self.sigma = sigma

    def __call__(self, img):
        img_np = np.array(img)
        img_filtered = gaussian_laplace(img_np, self.sigma)
        return img_filtered
```

result :



outcome

preprocessing	original	gray	LoG
resnet18	94%	96%	88.04%
resnet50	95%	95.6%	x

This is the result of my experiment in terms of preprocessing, the accuracy of gray is much higher compared to other methods, but in terms of the model because the biggest difference between resnet18 and resnet50 is in their network depth, perhaps because of this resnet18 I can use less epoch to train to better values, while resnet50 is to set a larger number to achieve better values, but in the process of experimentation, I found that the results of training resnet50 each time the accuracy range of training is more stable.