svm_cifar10

December 17, 2021

[ ]:
```
import pickle

from train_model import train_step, test_step
from utils.load_data import get_data
from utils.make_dict import train_bow, get_bow
```

[ ]:
```
args ={'dataset': 'MNIST',
       'dataroot': './data',
       'model': 'custom_SVM',
       'kernel': 'gaussian',
       'validation': 0.1,
       'C': 5.0,
       'sigma': 1.0,
       'batch': 1000,
       'dict_size': 100,
       'train': True,
       'load_cluster': False,
       'cuda': True
       }
```

[ ]:
```
hyper_C = [0.8, 0.9, 0.95, 1.0, 1.25]
hyper_sigma = [1.0, 1.25, 1.5]
```

[ ]:
```
trainX, trainy = get_data(dataset=args['dataset'], train=True,␣
 ↪dataroot=args['dataroot'])

if args['load_cluster']:
    with open("./cluster.dump", "rb") as f:
        cluster = pickle.load(f)
else:
    cluster = train_bow(trainX, num_dict=args['dict_size'], num_select=10000)
    with open("./cluster.dump", "wb") as f:
        pickle.dump(cluster, f)

trainFeature = get_bow(trainX, cluster, num_dict=args['dict_size'])
```

[ ]:
```
best_C = None
best_sigma = None
best_valid = 0.0

for C in hyper_C:
    for sigma in hyper_sigma:
        # Test hyperparameter
        args['C'] = C
        args['sigma'] = sigma

        # Get result
        _, train_acc_list, valid_acc_list  = \
            train_step(args, trainFeature, trainy)

        # Evaluation parameter
        tra = sum(train_acc_list) / len(train_acc_list)
        val = sum(valid_acc_list) / len(valid_acc_list)

        if val > best_valid:
            best_valid = val
            best_C = C
            best_sigma = sigma

        # Print result
        print("C: %f Sigma: %f Train accuracy: %f Valid accuracy: %f"%(C,␣
 ↪sigma, tra, val))

print("Best C: %f Best sigma: %f"%(best_C, best_sigma))
```

```
100%|      | 10/10 [00:08<00:00,  1.25it/s]
C: 0.800000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 42.730000
100%|      | 10/10 [00:05<00:00,  1.69it/s]
C: 0.800000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 42.956667
100%|      | 10/10 [00:07<00:00,  1.30it/s]
C: 0.800000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 43.125000
100%|      | 10/10 [00:03<00:00,  3.26it/s]
C: 0.900000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 34.775000
100%|      | 10/10 [00:05<00:00,  1.94it/s]
C: 0.900000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 42.770000
100%|      | 10/10 [00:06<00:00,  1.57it/s]
C: 0.900000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 34.825000
100%|      | 10/10 [00:03<00:00,  3.04it/s]
```

```
C: 0.950000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 50.628333
100%|      | 10/10 [00:04<00:00,  2.21it/s]
C: 0.950000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 34.656667
100%|      | 10/10 [00:05<00:00,  1.92it/s]
C: 0.950000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 50.438333
100%|      | 10/10 [00:02<00:00,  3.63it/s]
C: 1.000000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 51.018333
100%|      | 10/10 [00:03<00:00,  2.82it/s]
C: 1.000000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 43.033333
100%|      | 10/10 [00:04<00:00,  2.13it/s]
C: 1.000000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 26.896667
100%|      | 10/10 [00:02<00:00,  3.54it/s]
C: 1.250000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 51.193333
100%|      | 10/10 [00:03<00:00,  2.67it/s]
C: 1.250000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 66.788333
100%|      | 10/10 [00:04<00:00,  2.18it/s]
C: 1.250000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 35.015000
Best C: 1.250000 Best sigma: 1.250000
```

[ ]:
```
args['C'] = best_C
args['sigma'] = best_sigma
args['part'] = False
models, train_acc_list, valid_acc_list = \
            train_step(args, trainFeature, trainy)
```

```
100%|      | 10/10 [00:03<00:00,  2.76it/s]
```

[ ]:
```
testX, testy = get_data(dataset=args['dataset'], train=False,␣
 ↪dataroot=args['dataroot'])
testFeature = get_bow(testX, cluster, num_dict=args['dict_size'])
```

[ ]:
```
test_acc_list = test_step(args, testFeature, testy, models)
```

```
 90%|      | 9/10 [00:01<00:00,  5.63it/s]
```

[ ]:
```
print("Test average accuracy:", sum(test_acc_list) / len(test_acc_list))
```

```
Test average accuracy: 82.056
```