# SVM

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': 'cifar10',
       '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
       }
```

[ ]:
```
hyper_C = [0.5, 0.8, 0.9, 0.95, 1.0, 1.25]
hyper_sigma = [1.0, 1.25, 1.5, 2.0]
```

[ ]:
```
trainX, trainy = get_data(dataset=args['dataset'], train=True,␣
 ↪dataroot=args['dataroot'])
trainX = trainX.reshape((-1, 32, 32, 3), order='F')

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 [00:11<00:00,  1.13s/it]
C: 0.500000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 43.058000
100%|      | 10/10 [00:07<00:00,  1.30it/s]
C: 0.500000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 58.838000
100%|      | 10/10 [00:07<00:00,  1.29it/s]
C: 0.500000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 66.930000
100%|      | 10/10 [00:07<00:00,  1.32it/s]
C: 0.500000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 58.878000
100%|      | 10/10 [00:07<00:00,  1.33it/s]
C: 0.800000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 50.902000
100%|      | 10/10 [00:07<00:00,  1.34it/s]
C: 0.800000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 67.056000
100%|      | 10/10 [00:07<00:00,  1.30it/s]
```

```
C: 0.800000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 58.832000
100%|      | 10/10 [00:07<00:00,  1.28it/s]
C: 0.800000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 50.812000
100%|      | 10/10 [00:07<00:00,  1.28it/s]
C: 0.900000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 34.796000
100%|      | 10/10 [00:07<00:00,  1.33it/s]
C: 0.900000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 42.034000
100%|      | 10/10 [00:07<00:00,  1.30it/s]
C: 0.900000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 51.206000
100%|      | 10/10 [00:07<00:00,  1.35it/s]
C: 0.900000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 67.044000
100%|      | 10/10 [00:07<00:00,  1.36it/s]
C: 0.950000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 66.776000
100%|      | 10/10 [00:07<00:00,  1.38it/s]
C: 0.950000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 58.732000
100%|      | 10/10 [00:07<00:00,  1.41it/s]
C: 0.950000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 50.088000
100%|      | 10/10 [00:07<00:00,  1.38it/s]
C: 0.950000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 58.938000
100%|      | 10/10 [00:07<00:00,  1.38it/s]
C: 1.000000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 50.854000
100%|      | 10/10 [00:07<00:00,  1.35it/s]
C: 1.000000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 42.774000
100%|      | 10/10 [00:07<00:00,  1.26it/s]
C: 1.000000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 51.014000
100%|      | 10/10 [00:08<00:00,  1.13it/s]
C: 1.000000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 42.766000
100%|      | 10/10 [00:07<00:00,  1.32it/s]
C: 1.250000 Sigma: 1.000000 Train accuracy: 100.000000 Valid accuracy: 58.900000
100%|      | 10/10 [00:07<00:00,  1.33it/s]
C: 1.250000 Sigma: 1.250000 Train accuracy: 100.000000 Valid accuracy: 51.180000
100%|      | 10/10 [00:07<00:00,  1.35it/s]
```

```
C: 1.250000 Sigma: 1.500000 Train accuracy: 100.000000 Valid accuracy: 35.260000
100%|      | 10/10 [00:07<00:00,  1.37it/s]
C: 1.250000 Sigma: 2.000000 Train accuracy: 100.000000 Valid accuracy: 42.994000
Best C: 0.800000 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 [00:07<00:00,  1.29it/s]
```

[ ]:
```
testX, testy = get_data(dataset=args['dataset'], train=False,␣
↪dataroot=args['dataroot'])
testX = testX.reshape((-1, 32, 32, 3), order='F')
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,  4.94it/s]
```

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

```
Test average accuracy: 66.0
```