计算机科学与技术学院神经网络与深度学习课程实验报告

实验题目：体验并实现多种图像分类算法		学号：201918130222
日期：2021/9/30	班级： 智能	姓名： 魏江峰
Email：2257263015@qq.com		

实验目的：

理解并掌握常见图像分类算法的实现

understand the basic **Image Classification pipeline** and the data-driven approach (train/predict stages)

understand the train/val/test **splits** and the use of validation data for **hyperparameter tuning**.

develop proficiency in writing efficient **vectorized** code with numpy

实验软件和硬件环境：

硬件环境：

处理器：Intel core i7 9750-H

电脑：神州 z7m-ct7nk

软件环境：

Pycharm 与 jupyter notebook

实验原理和方法：

利用 Python 的科学计算库，实现常见的图像分类算法

实验步骤：（不要求罗列完整源代码）

1，implement and apply a k-Nearest Neighbor (kNN) classifier

使用两层循环计算 dists:

```
for i in range(num_test):
    for j in range(num_train):
        dists[i, j] = np.sqrt(np.sum((self.X_train[j] - X[i]) ** 2))
```

一层循环：

```
for i in range(num_test):
    m = np.square(X[i] - self.X_train)
    dists[i] = np.sqrt(np.sum(m, axis=1))
```

No loop:

```
M = np.dot(X, self.X_train.T)
nrow = M.shape[0]
ncol = M.shape[1]
te = np.diag(np.dot(X, X.T))
tr = np.diag(np.dot(self.X_train, self.X_train.T))
te = np.reshape(np.repeat(te, ncol), M.shape)
```

```
    tr = np.reshape(np.repeat(tr, nrow), M.T.shape)
    sq = -2 * M + te + tr.T
    dists = np.sqrt(sq)
```

预测 labels：

```
for i in range(num_test):
    dis = np.argsort(dists[i])
    closest_y = self.y_train[dis[0:k]]
    count = np.bincount(closest_y)
    y_pred[i] = np.argmax(count)
```

2，implement and apply a Multiclass Support Vector Machine (SVM) classifier ,

Naive loss 计算：

```
    dW = np.zeros(W.shape)  # initialize the gradient as zero
    num_classes = W.shape[1]
    num_train = X.shape[0]
    loss = 0.0
    for i in range(num_train):
        scores = X[i].dot(W)
        correct_class_score = scores[y[i]]
        for j in range(num_classes):
            if j == y[i]:
                continue
            margin = scores[j] - correct_class_score + 1  # note delta = 1
            if margin > 0:  # max(0,--)操作
                loss += margin
                dW[:, y[i]] += -X[i]  # 对应正确分类的梯度
                dW[:, j] += X[i]  # 对应不正确分类的梯度
    loss /= num_train
    dW /= num_train
    loss += reg * np.sum(W * W)
    dW += 2 * reg * W
```

向量化 loss 计算：

```
    scores = X.dot(W)
    margins = np.maximum(0, scores - scores[np.arange(scores.shape[0]), y][:, np.newaxis] + 1)
    margins[np.arange(margins.shape[0]), y] = 0
    loss = np.sum(margins)
    loss += reg * np.sum(W * W)
    loss /= X.shape[0]

    binary_matrxi = np.zeros(margins.shape)
    binary_matrxi[margins > 0] = 1
    row_sum = np.sum(binary_matrxi, axis=1)
```

```
    binary_matrxi[np.arange(binary_matrxi.shape[0]), y] = -row_sum.T
    dW = np.dot(X.T, binary_matrxi)
    dW /= margins.shape[0]
    dW += 2 * reg * W
```

3，implement and apply a Softmax classifier

向量化 softmax loss 计算：

```
    scores = X.dot(W)
    scores = scores - np.max(scores, axis=1)[:, np.newaxis]
    scores = np.exp(scores)
    scores /= np.sum(scores, axis=1)[:, np.newaxis]
    loss = np.sum(-np.log(scores[np.arange(scores.shape[0]), y]))

    dW = dW.T

    scores_temp = np.zeros(scores.shape)
    for i in range(scores.shape[1]):   # record the cladd lable
        scores_temp[:, i] = i

    scores_temp = [y[:, np.newaxis].flatten() == scores_temp[:, i]
                   for i in range(W.shape[1])]

    scores_temp = np.array(scores_temp)
    scores_temp = scores_temp.T

    dW = ((scores - (scores_temp)).T).dot(X)
    dW = dW.T

    loss /= X.shape[0]
    dW /= X.shape[0]
    loss += reg * np.sum(W * W)
    dW += 2 * reg * W
```

4，implement and apply a Three layer neural network classifier

Loss 计算：

```
    scores = scores - np.max(scores, axis=1)[:, np.newaxis]
    scores = np.exp(scores)
    scores /= np.sum(scores, axis=1)[:, np.newaxis]
    loss = np.sum(-np.log(scores[np.arange(scores.shape[0]), y]))
    loss = loss / X.shape[0] + reg * (np.sum(W1 * W1) + np.sum(W2 * W2) + np.sum(W3 * W3))
```

其他内容可以参见 jupyter 文件

结论分析与体会：

the differences and tradeoffs between these classifiers

KNN：训练快，预测慢，预测准确性较差

SVM: 训练较慢，预测快，准确性较好

Softmax: 训练慢，预测快，准确性较好

三层神经网络：训练极慢，预测快，准确性好

High level 的图像特征表示如 color histogram,histogram of gredient,从图像中提取出了空间较小，更为精炼的特征，可以使训练速度加快，提升分类性能

就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟 1－3 道问答题：

API 不会用？

搜索，看官方文档，测试输入与输出