NameSurname=SelimCanYazar

StudentID= 152120191023

Pattern Recognition HW5

```python
import zipfile
import os

# HWData.zip dosyasını çıkartma
zip_path = 'HWData.zip'
extract_path = 'HWData'

# Eğer dosya zaten çıkartılmamışsa çıkartalım
if not os.path.exists(extract_path):
    with zipfile.ZipFile(zip_path, 'r') as zip_ref:
        zip_ref.extractall(extract_path)
    print("Zip dosyası başarıyla çıkartıldı.")
else:
    print("Zip dosyası zaten çıkartılmış.")
```

Elimizde HWData.zip adlı ziplenmiş dataset dosyası var ve bu dosyanın içeriğini zipten çıkartmak istiyoruz. İlk olarak bu işlemi gerçekleştirebilmek için gerekli olan zipfile ve os kütüphanelerini kullanıyoruz. Sıkıştırılmış dosyanın yolu zip_path olarak tanımlandı ve bu dosyanın içeriğinin çıkartılacağı klasörün adı extract_path olarak belirlendi.

Önce extract_path adlı klasörün zaten mevcut olup olmadığını kontrol ediyoruz. Eğer bu klasör zaten varsa dosyaların daha önce çıkartılmış olduğu sonucuna varıp "Zip dosyası zaten çıkartılmış." mesajını yazdırıyoruz.

Eğer extract_path klasörü mevcut değilse, zipfile.ZipFile fonksiyonunu kullanarak HWData.zip dosyasını açıyoruz ve extractall metoduyla tüm içeriği extract_path klasörüne çıkartıyoruz. Bu işlem başarılı bir şekilde tamamlandığında "Zip dosyası başarıyla çıkartıldı." mesajını ekrana yazdırıyoruz.

```python
import numpy as np
import cv2
from skimage.color import rgb2lab, rgb2gray
from sklearn import svm
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.feature_selection import SequentialFeatureSelector as SFS
import matplotlib.pyplot as plt
import os

# OpenCV'nin SIFT modülünü kullanmak için gerekli kütüphane
!pip install opencv-contrib-python

# Veri yollarını belirleme
train_dir = os.path.join('HWData', 'HWData/train')
test_dir = os.path.join('HWData', 'HWData/test')

# Resimleri ve etiketleri yükleme ve özellik çıkarma
def load_data_and_extract_features(data_dir):
    features = []
    labels = []
    sift = cv2.SIFT_create()
    for class_dir in os.listdir(data_dir):
        class_path = os.path.join(data_dir, class_dir)
        if os.path.isdir(class_path):
            for img_name in os.listdir(class_path):
                img_path = os.path.join(class_path, img_name)
                img = cv2.imread(img_path)
                img = cv2.resize(img, (224, 224))
                img_lab = rgb2lab(img)
                img_gray = rgb2gray(img_lab).astype('uint8')

                # CLAHE ile kontrast artırma
                clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
                img_gray = clahe.apply(img_gray)

                keypoints, descriptors = sift.detectAndCompute(img_gray, None)

                if descriptors is not None:
                    features.append(descriptors.flatten())
                    labels.append(class_dir)

    return features, np.array(labels)

# Eğitim ve test verilerini yükleyip özellik çıkarma
X_train, y_train = load_data_and_extract_features(train_dir)
X_test, y_test = load_data_and_extract_features(test_dir)

# Özellik vektörlerini sabit bir boyuta getirme
max_features = 20  # Sabit boyut olarak 500'ü seçiyoruz
X_train_padded = np.array([np.pad(x, (0, max_features - len(x)), 'constant') if len(x) < max_features else x[:max_features] for x in X_train])
X_test_padded = np.array([np.pad(x, (0, max_features - len(x)), 'constant') if len(x) < max_features else x[:max_features] for x in X_test])

# Özellik seçimi (SFS yöntemiyle)
n_features = max_features // 2  # Özellik sayısını yarıya indirme
sfs = SFS(svm.SVC(kernel='linear'), n_features_to_select=n_features, direction='forward')
sfs.fit(X_train_padded, y_train)

X_train_selected = sfs.transform(X_train_padded)
X_test_selected = sfs.transform(X_test_padded)
```

```
# SVM ile eğitim ve test
model = svm.SVC(kernel='linear', decision_function_shape='ovr')
model.fit(X_train_selected, y_train)

# Tahminler
y_pred = model.predict(X_test_selected)

# Karışıklık matrisi ve doğruluk hesaplama
conf_matrix = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)

print('Confusion Matrix:\n', conf_matrix)
print('Accuracy:', accuracy)

# Karışıklık matrisini görselleştirme
plt.figure(figsize=(10, 7))
plt.imshow(conf_matrix, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()
```

Bu ödevde resim verileri üzerinde özellik çıkarma(feature extraction), özellik seçimi(feature selection) ve sınıflandırma işlemlerini gerçekleştirmek için çeşitli adımları gerçekleştirdik. İlk olarak, gerekli kütüphaneleri ekledik. Bunlar arasında numpy, cv2 (OpenCV), skimage, sklearn ve matplotlib gibi kütüphaneler bulunuyor. Ayrıca, OpenCV'nin SIFT (Scale-Invariant Feature Transform) modülünü kullanabilmek için `opencv-contrib-python` paketini yüklüyoruz. Bu paket görüntü işleme ve bilgisayarla görme görevlerinde kullanılır.

Veri yollarını belirleme aşamasında, eğitim ve test verilerinin bulunduğu dizinleri tanımlıyoruz. Bu dizinler, 'HWData' adlı ana klasör içinde 'train' ve 'test' olarak ayrılmış durumda. Eğitim verileri 'train' dizininde, test verileri ise 'test' dizininde bulunuyor. Bu dizinler içinde her sınıf için ayrı klasörler bulunuyor.

Veri yükleme ve özellik çıkarma işlemi, `load_data_and_extract_features` fonksiyonu ile gerçekleştiriliyor. Bu fonksiyon, sayesinde belirtilen veri dizinindeki resimleri yükleyerek daha sonra bu resimlerden SIFT kullanarak özellik çıkarıyor. İlk olarak, her sınıfın klasörüne giderek resimleri yüklüyoruz. Resimleri okuma, yeniden boyutlandırma ve gri tonlamaya dönüştürme işlemlerini gerçekleştiriyoruz. Ardından, CLAHE (Contrast Limited Adaptive Histogram Equalization) kullanarak resimlerin

kontrastını artırıyoruz. Bu işlem, özellikle detayların daha belirgin hale gelmesine yardımcı olur. Son olarak, SIFT ile anahtar noktalar ve bu noktalara ait özellikleri çıkarıyoruz. Çıkarılan bu özellikleri ve sınıf etiketlerini bir listeye ekliyoruz.

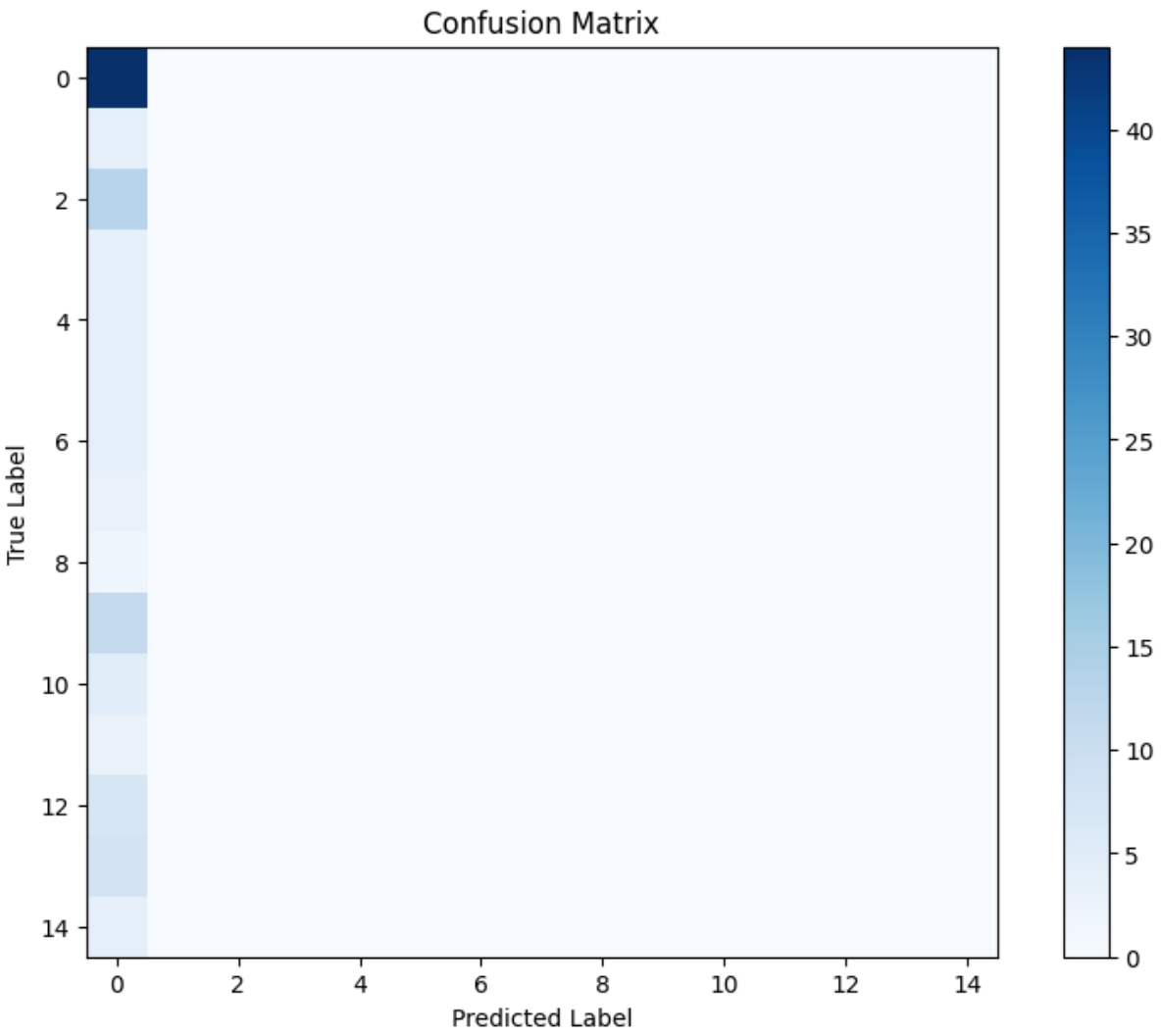
Eğitim ve test verilerini yükleme ve özellik çıkarma işlemi, `load_data_and_extract_features` fonksiyonunu kullanarak gerçekleştiriliyor. Eğitim ve test verilerinden özellik çıkarma işlemi tamamlandıktan sonra, bu verileri birer numpy dizisine dönüştürüyoruz. Özellik vektörlerinin boyutları farklı olabileceğinden, bu vektörleri sabit bir boyuta getiriyoruz. Örneğin, 20 elemanlık sabit bir boyut belirliyoruz. Bu işlemi vektörlerin eksik olan kısımlarını sıfırlarla doldurarak veya fazlalık olan kısımlarını keserek yapıyoruz. Bu sayede, modelimize girdi olarak vereceğimiz verilerin boyutları eşitlenmiş oluyor.

Özellik seçimi aşamasında, Sequential Feature Selector (SFS) kullanarak en iyi özellikleri seçiyoruz. Bu işlem, özellik sayısını yarıya indirme amacıyla yapılır ve en iyi n sayıda özelliği belirlememizi sağlar. SFS, özellik seçimini ileri yönlü yani forward seçim yöntemiyle gerçekleştirir. Bu adımda, SVM modeli ile en iyi özellikleri seçmek için svm.SVC sınıfını kullanıyoruz.

Model eğitim ve test aşamasında, seçilen özellikleri kullanarak SVM modelini eğitiyoruz ve ardından test verileri üzerinde tahminler yapıyoruz. SVM, sınıflandırma problemleri için yaygın olarak kullanılan güçlü bir algoritmadır. Modelin tahmin sonuçlarını kullanarak confusion matrix ve accuracy hesaplıyoruz. Confusion matrisi hangi sınıfların doğru veya yanlış sınıflandırıldığını gösterir ve modelimizin performansını değerlendirmek için kullanılır.

Son olarak da karışıklık matrisini görselleştiriyoruz. Bu görselleştirme, sonuçları daha iyi anlamamıza yardımcı olur. Matris doğru ve yanlış sınıflandırmaların görsel bir temsilini sağlar. Görselleştirme işlemi için matplotlib kütüphanesini kullanıyoruz. Bu şekilde resim verilerini kullanarak makine öğrenimi modeli oluşturma ve değerlendirme sürecini ekstra olarak yapmış oluyoruz.

```
 [[44  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [13  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 3  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 2  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [11  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 5  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 3  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 7  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 8  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 4  0  0  0  0  0  0  0  0  0  0  0  0  0  0]]
Accuracy: 0.36666666666666664
```



Son olarak accuracy değeri , confusion matrix ve confusion matrix grafiği yukarıdaki gibi olacak şekilde sonuç bulunmuştur.