Uygulama 6. 3-Boyutlu Evrişimsel Sinir Ağı ile Görüntü Verilerinin Analizi

Bu uygulamada kedi ve köpeklere ait görüntülerin sınıflandırılması ve analizi yapılacaktır.

Öncelikle sistemin çalışması için gerekli olan kütüphaneler yüklenir.

```
#3DESA ile görüntü sınıflandırma

#gerekli kütüphanelerin yüklenmesi
import numpy as np
import pandas as pd
from keras.preprocessing.image import ImageDataGenerator, load_img
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import random
import os
```

Ardından kullanılacak klasörün yolu belirtilir.

```
#klasörün gösterilmesi
print(os.listdir("C:/Users/burak/Masaüstü/CNN Görüntü Sınıflandırma/kediKöpek"))

#gerekli sabitlerin belirlenmesi
```

Daha sonra gerekli olan sabit değerler belirlenir.

```
#gerekli sabitlerin belirlenmesi
IMAGE_WIDTH=128
IMAGE_HEIGHT=128
IMAGE_SIZE=(IMAGE_WIDTH, IMAGE_HEIGHT)
IMAGE_CHANNELS=3

#eğitim verisinin hazırlanması
```

Eğitim verisi hazırlanır.

```
#eğitim verisinin hazırlanması
filenames = os.listdir("D:/CNN Görüntü Sınıflandırma/kediKöpek/train")
categories = []
for filename in filenames:
    category = filename.split('.')[0]
    if category == 'dog':   #1 numaralı sınıf köpek
        categories.append(1)
    else:
        categories.append(0) #0 numaralı sınıf kedi
df = pd.DataFrame({
    'filename': filenames,
    'category': categories
})
```

Eğitim verileri belirlendikten sonra model oluşturulur ve derlenir.

```
#modelin oluşturulması
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten, Dense, Activation, BatchNormalization

model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(IMAGE_WIDTH, IMAGE_HEIGHT, IMAGE_CHANNELS)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax')) # 2 because we have cat and dog classes
model.summary()

#modelin derlenmesi
model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
```

Ardından eğitim ve doğrulama verileri hazırlanır.

```
#eğitim ve doğrulama verisinin hazırlanması
total_train = train_df.shape[0]
total_validate = validate_df.shape[0]
batch_size=15
```

Daha sonra eğitim ve doğrulama verileri çoğaltılır.

```
#eğitim verilerinin çoğaltılması
train_datagen = ImageDataGenerator(
    rotation_range=15,
    rescale=1./255,
    shear_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
    width_shift_range=0.1,
    height_shift_range=0.1
)

train_generator = train_datagen.flow_from_dataframe(
    train_df,
    "C:/Users/burak/Masaüstü/CNN Görüntü Sınıflandırma/kediKöpek/train/",
    x_col='filename',
    y_col='category',
    target_size=IMAGE_SIZE,
    class_mode='categorical',
    batch_size=batch_size
)

#doğrulama verilerinin çoğaltılması
validation_datagen = ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_dataframe(
    validate_df,
    "C:/Users/burak/Masaüstü/CNN Görüntü Sınıflandırma/kediKöpek/train/",
    x_col='filename',
    y_col='category',
    target_size=IMAGE_SIZE,
    class_mode='categorical',
    batch_size=batch_size
)
```

Model eğitilerek sınıflandırma işlemi tamamlanmış olur.

```
#modelin eğitilmesi
epochs=10
history = model.fit_generator(
    train_generator,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=total_validate//batch_size,
    steps_per_epoch=total_train//batch_size,
)
```

Eğer daha sonra eğittiğimiz modele ait verileri kullanmak istersek modelimizi kaydetmemiz gerekmektedir.

```
#oluşturulan modelin kaydedilmesi
model.save_weights("model1.h5")
```

Eğitim işlemi bittiğinde tahmin işlemi gerçekleştirilir.

```
#tahmin işleminin yapılması
predict = model.predict_generator(test_generator, steps=np.ceil(nb_samples/batch_size))
```

Uygulama 6. Değerlendirme Soruları

1) Verilen kodu inceleyerek kendi modelinizi oluşturunuz **(10p).**

2) Verilen kodu inceleyerek farklı epok değerlerini kullanınız. Kullandığınız bu epoklar içerisinde en iyi epok değerini belirleyiniz **(10p).**

3) Yaprak verisetine ait verileri kullanarak bir ESA mimarisi geliştiriniz. Geliştirmiş olduğunuz modelin doğrulama (accuracy), F1-skor, ve kayıp değerlerini bulunuz. Bulduğunuz bu değerleri grafik üzerinde gösteriniz. Bunlara ek olarak tahmin işlemi gerçekleştiriniz **(40p)**.

4) Beyin tümörüne ait verileri kullanarak bir ESA mimarisi geliştiriniz. Geliştirmiş olduğunuz modelin doğrulama (accuracy) ve kayıp değerlerini bulunuz. Bulduğunuz bu değerleri grafik üzerinde gösteriniz. Bunlara ek olarak karmaşıklık matrisi oluşturunuz **(40p)**.