Tugas Signal 1 Dimensi

Nama : Muhammad Fakhri Ajrillah

NIM : 22/498417/PA/21515

Tahap 1 - Preproses

Kode ini mengekstrak fitur MFCC (Mel-Frequency Cepstral Coefficients) dari file audio dalam dataset perintah suara ("maju", "mundur", "kanan", "kiri") untuk digunakan dalam pembelajaran mesin. Setiap file audio dibaca menggunakan Librosa, diubah menjadi fitur MFCC, lalu dinormalisasi panjangnya hingga 30 timestep dengan padding atau pemangkasan agar seragam. Fitur ini kemudian diratakan menjadi array satu dimensi dan dikumpulkan dalam X sebagai fitur serta y sebagai labelnya. Setelah seluruh dataset diproses, data disimpan dalam file "features.pkl".

Code :

```
import librosa
import librosa.display
import numpy as np
import os
import pickle

DATASET_PATH = "dataset/mentah/"
LABELS = ["maju", "mundur", "kanan", "kiri"]
OUTPUT_FILE = "dataset/features.pkl"

def extract_features(file_path, max_length=30):
   audio, sr = librosa.load(file_path, sr=16000)
   mfcc = librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=13)

   if mfcc.shape[1] < max_length:
       pad_width = max_length - mfcc.shape[1]
       mfcc = np.pad(mfcc, ((0,0), (0,pad_width)), mode='constant')
   else:
       mfcc = mfcc[:, :max_length]

   return mfcc.flatten()

X, y = [], []
```

```
for label in LABELS:
    folder_path = os.path.join(DATASET_PATH, label)
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        features = extract_features(file_path)
        X.append(features)
        y.append(label)

X = np.array(X)
y = np.array(y)

with open(OUTPUT_FILE, "wb") as f:
    pickle.dump((X, y), f)

print(f"Dataset disimpan di {OUTPUT_FILE} dengan {len(X)} sampel.")
```

Tahap 2 - Pelatihan

Kode ini melatih model K-Nearest Neighbors (KNN) untuk mengklasifikasikan perintah suara berdasarkan fitur MFCC yang telah diekstrak sebelumnya. Dataset yang tersimpan dalam file "features.pkl" dimuat, kemudian labelnya dikonversi ke format numerik menggunakan LabelEncoder. Selanjutnya, data dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Terakhir, model KNN beserta encoder label disimpan dalam file "model_knn.pkl".

Code :

```
import pickle
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

with open("dataset/features.pkl", "rb") as f:
    X, y = pickle.load(f)

encoder = LabelEncoder()
y = encoder.fit_transform(y)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

y_pred = knn.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Akurasi model: {accuracy:.2f}")

with open("model_knn.pkl", "wb") as f:
   pickle.dump((knn, encoder), f)

print("Model KNN berhasil disimpan!")
```

Tahap 3 - Prediksi

Kode ini digunakan untuk memprediksi perintah suara dari file audio menggunakan model **K-Nearest Neighbors (KNN)** yang telah dilatih sebelumnya. Model KNN dan encoder label dimuat dari file **"model_knn.pkl"**, kemudian file audio dibaca menggunakan **Librosa** dan dikonversi menjadi fitur **MFCC**. Setelah itu, fitur diubah menjadi **array satu dimensi** dan diberikan ke model KNN untuk melakukan prediksi. Hasil prediksi berupa label numerik dikembalikan ke bentuk aslinya menggunakan **encoder**, lalu ditampilkan sebagai output. Terakhir, model diuji dengan file audio **"2.mp3"**, dan hasilnya dicetak sebagai perintah yang terdeteksi.

Code :

```
import librosa
import pickle
import numpy as np

with open("model_knn.pkl", "rb") as f:
   knn, encoder = pickle.load(f)

def predict_audio(file_path):
   """Memprediksi perintah dari file audio."""
   audio, sr = librosa.load(file_path, sr=16000)
```

```
    mfcc = librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=13)

    max_length = 30
    if mfcc.shape[1] < max_length:
        pad_width = max_length - mfcc.shape[1]
        mfcc = np.pad(mfcc, ((0,0), (0,pad_width)), mode='constant')
    else:
        mfcc = mfcc[:, :max_length]

    mfcc = mfcc.flatten().reshape(1, -1)
    prediction = knn.predict(mfcc)
    label = encoder.inverse_transform(prediction)[0]

    return label

file_path = "2.mp3"
predicted_command = predict_audio(file_path)
print(f"Prediksi perintah: {predicted_command}")
```

Hasil

```
(py38) aj@icardo:~/workspace/kuliah/pengenalanPola/code$ /home/aj/anaconda3/envs/py38/bin/pyth
on /home/aj/workspace/kuliah/pengenalanPola/code/predict.py
Prediksi perintah: mundur
```