Son Teslim tarihi: 31.05.2023 23.59

Proje

OĞUZHAN TOPALOĞLU
Ç19052025 – Grup 1

Bilgisayar Mühendisliği Bölümü,
Elektrik-Elektronik Fakültesi,
Yıldız Teknik Üniversitesi

Istanbul, 2023

İÇERİK

1. STFT İşlemi

Projede main kısmında çalıştırılabilecek 2 adet fonksiyon bulunmaktadır:

```
if __name__ == '__main__':
    # Hangi main'in çalıştırılması isteniyorsa öbürü yoruma
alınmalı
    # main_create_csv_files()
    main_knn()
```

Bunlardan main_create_csv_files fonksiyonu bütün WAV dosyalarını okuyarak projede istenen 6 adet CSV dosyasını üretmektedir. Main_knn fonksiyonu da bu 6 CSV dosyasını okuyarak yine projede istenen tabloları oluşturmaktadır. (KNN algoritmasını sklearn'den kullanarak).

CSV üreten main fonksiyonu aşağıdaki gibidir:

```
def main_create_csv_files():
    create_specific_csv("Blackman", np.blackman, True)
    create_specific_csv("Hamming", np.hamming, True)
    create_specific_csv("Hanning", np.hanning, True)
    create_specific_csv("Blackman", np.blackman, False)
    create_specific_csv("Hamming", np.hamming, False)
    create_specific_csv("Hanning", np.hanning, False)
```

Görüldüğü gibi, kullanılacak window türünün ismini, fonksiyonunu ve True/False ile test/train olup olmadığını belli ederek başka bir fonksiyonu çağırmaktadır. Bu fonksiyon da aşağıdaki gibidir:

```
def create_specific_csv(window_name, window_func, is_train):
    train_or_test = "Train" if is_train else "Test"
    csv_name = f"{window_name}_{train_or_test}.csv"

    df_list = []
    for music_type in ["classical", "country", "jazz", "metal", "reggae"]:
        folder_path = os.path.join("Data", music_type, train_or_test.lower())
        for music_file_path in os.listdir(folder_path):
            music_file_path = os.path.join(folder_path, music_file_path)

            stft_result = my_stft(music_file_path, window_func)
            features_array = get_features_vector(stft_result)
            features_array.append(music_type)
            df_list.append(features_array)

    result_df = pd.DataFrame(df_list)
    result_df.to_csv(csv_name, index=False)
```

Fonksiyon öncelikle oluşturulacak CSV dosyasının ismini belirlemekte ve sonrasında iki for döngüsü ile seçilen 5 müzik türünde her müzik dosyasını alarak my_stft ile STFT hesaplamakta ve ardından get_features_vector ile de özellik çıkarımı yapmaktadır.

STFT işlemleri aşağıdaki fonksiyonda yapılmaktadır:

```
def my_stft(file_path, window_func):
    # data = işlenecek sinyali içeren bir numpy dizisi
    # fs = verinin örnekleme frekansı olan skalar
    data, fs = read_wav_file(file_path)

    fft_size = 4096 # toplantıda 4000-5000 arası bir şey uygundur denildi
    overlap_fac = 0.2  # raporda hep %20 alınmalı denildi

    hop_size = np.int32(np.floor(fft_size * (1-overlap_fac)))
    pad_end_size = fft_size  # son segment
    total_segments = np.int32(np.ceil(len(data) / np.float32(hop_size)))

    window = window_func(fft_size)
    inner_pad = np.zeros(fft_size) # her segmentin boyutunu iki katına çıkarmak için
kullanılacak sıfırlar

    proc = np.concatenate((data, np.zeros(pad_end_size)))
    result = np.empty((total_segments, fft_size), dtype=np.float32)

    for i in range(total_segments):
        current_hop = hop_size * i
        segment = proc[current_hop:current_hop+fft_size]
        windowed = segment * window
        padded = np.append(windowed, inner_pad)
        spectrum = my_fft(padded) / fft_size              # benim fonksiyonumla FFT al
        autopower = np.abs(spectrum * np.conj(spectrum))
        result[i, :] = autopower[:fft_size]

    result = 20 * np.log10(result)        # dB'ye ölçeklendir
    result = np.clip(result, -40, 200)    # değerleri kırp, linkte neden yaptığını açıklamış
gerek yok aslında
    return result
```

Bu fonksiyon şu kaynaktan alınmış ve proje boyunca değiştirilmiştir:
https://kevinsprojects.wordpress.com/2014/12/13/short-time-fourier-transform-using-python-and-numpy/

Fonksiyon içinde öncelikle müzik dosyası bir numpy array'ine aşağıdaki fonksiyon ile okunmuştur:

```
# Verilen path'i bir numpy dizisine okur
def read_wav_file(file_path):
    wav = wave.open(file_path, 'r')
    data = wav.readframes(-1)
    data = np.frombuffer(data, dtype=np.int16)
    fs = wav.getframerate()
    wav.close()
    return data, fs
```

Ardından toplantıda denildiği gibi fft_size, overlap_fac belirlenmiş ve SFTF algoritması kodlanmıştır. Algoritmada hazır FFT fonksiyonu kullanılmamış ve aşağıdaki reküsif ve benim tarafımdan yazılmış FFT fonksiyonu kullanılmıştır:

```
# Benim FFT fonksiyonum
def my_fft(data):
    N = len(data)

    if N <= 1:
        return data

    even = my_fft(data[0::2])  # Çift indislere FFT uygula
    odd = my_fft(data[1::2])   # Tek indislere FFT uygula

    T = np.exp(-2j * np.pi * np.arange(N) / N)  # Dönüşüm matrisi
    return np.concatenate([even + T[:N // 2] * odd, even + T[N // 2:] * odd])
```

Bu SFTF sonucu hesaplanan result ile de aşağıdaki fonksiyon sayesinde projede istenilen üç özelliğin ortalaması, medyanı ve standart sapması hesaplanmış ve bir liste ile dönülmüştür:

```
# STFT sonucundan özellik çıkarımı
def get_features_vector(stft_result):
    # v1: frequency_power
    v1 = np.mean(stft_result, axis=1)
    # v2: frequency_amplitude
    v2 = np.mean(np.abs(stft_result), axis=1)
    # v3: weighted_average_frequency
    frequencies = np.fft.fftfreq(stft_result.shape[1])
    v3 = np.sum(np.abs(stft_result) * frequencies, axis=1) /
np.sum(np.abs(stft_result), axis=1)
    return [\
        np.mean(v1), np.median(v1), np.std(v1), \
        np.mean(v2), np.median(v2), np.std(v2), \
        np.mean(v3), np.median(v3), np.std(v3)]
```

2. Tablo Üretimi

KNN algoritması ve tablo üretimi öbür main fonksiyonunda yapılmaktadır:

```
def main_knn():
    create_table(k=1, window_type="Blackman")
    create_table(k=3, window_type="Blackman")
    create_table(k=5, window_type="Blackman")

    create_table(k=1, window_type="Hamming")
    create_table(k=3, window_type="Hamming")
    create_table(k=5, window_type="Hamming")

    create_table(k=1, window_type="Hanning")
    create_table(k=3, window_type="Hanning")
    create_table(k=5, window_type="Hanning")
```

Görüldüğü gibi bu main fonksiyonu içinde create_table fonksiyonunu çağırmakta ve kullanılan window türü ile k sayısını parametre olarak aktarmaktadır. Bu fonksiyon da aşağıdaki şekildedir:

```
def create_table(k, window_type):
    test_path = os.path.join("Dataset", window_type+"_Test.csv")
    train_path = os.path.join("Dataset", window_type+"_Train.csv")

    train_df = pd.read_csv(train_path)
    train_df.columns = ["freqPowerMean", "freqPowerMed", "freqPowerStd",
"freqAmpMean", \
                "freqAmpMed", "freqAmpStd", "weiFreqMean", "weiFreqMed", "weiFreqStd",
"type"]

    # Train the knn
    X = train_df.drop('type', axis=1)
    y = train_df['type']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    # Test the knn
    test_df = pd.read_csv(test_path)
    test_df.columns = ["freqPowerMean", "freqPowerMed", "freqPowerStd",
"freqAmpMean", \
                "freqAmpMed", "freqAmpStd", "weiFreqMean", "weiFreqMed", "weiFreqStd",
"type"]
    X = test_df.drop('type', axis=1)
    y = test_df['type']
    y_pred = knn.predict(X)

    count = len(y_pred)
    correct = 0
    for i in range(len(y_pred)):
        if y_pred[i] == y[i]:
            correct += 1
```

```
    table_data = []
    for i in range(len(y_pred)):
        table_data.append([y[i], y_pred[i]])

    fig, ax = plt.subplots()
    ax.axis('off')

    table = ax.table(cellText=table_data, colLabels=['Actual', 'Predicted'],
                     cellLoc='center', loc='center', edges='horizontal')

    table.auto_set_font_size(False)
    table.set_fontsize(12)
    table.scale(1.2, 1.2)

    table_title = f"{window_type}, k={k}, Correctness: {correct}/{count}"
    ax.set_title(table_title, fontweight='bold', fontsize=14, y=2)

    # Save the figure as an image
    if not os.path.exists("Tables"):
        os.mkdir("Tables")
    plt.savefig(f"Tables/{window_type}_{k}.png", bbox_inches='tight', dpi=300)
```

Fonksiyonda öncelikle verilen parametrelere göre train ve test CSV dosyalarının path'i bulunmaktadır. Ardından train dosyası okunarak bir KNN algoritması verilen k parametresine göre oluşturulmakta ve train edilmektedir. Model oluşturulduktan sonra da test dosyası okunmakta ve bilinen type değeri silinerek modele tür tahmini yaptırılmaktadır. Ardından da yapılan bu tahminler tablo şeklinde Tables klasöründe png dosyası olarak oluşturulmaktadır. Bu görseller bir sonraki bölümde bulunmaktadır.

3. Ekran Görüntüleri

Hanning, k=3, Correctness: 36/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	jazz
classical	jazz
classical	classical
classical	classical
classical	classical
classical	classical
country	country
country	jazz
country	country
country	jazz
country	reggae
country	reggae
country	reggae
country	country
country	jazz
country	country
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	jazz
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	country
reggae	reggae

Hanning, k=1, Correctness: 38/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	country
country	jazz
country	country
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	country
country	reggae
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	country
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae

Hamming, k=5, Correctness: 35/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	country
country	jazz
country	jazz
country	classical
country	reggae
country	reggae
country	jazz
country	metal
country	jazz
country	jazz
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	jazz
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	jazz
reggae	reggae

Hamming, k=3, Correctness: 36/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	country
country	jazz
country	reggae
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	reggae
country	country
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	country
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	country
reggae	reggae

Hamming, k=1, Correctness: 35/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	metal
country	jazz
country	reggae
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	jazz
country	reggae
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	country
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae

Blackman, k=5, Correctness: 36/50

Actual	Predicted
classical	jazz
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	jazz
classical	classical
classical	classical
classical	classical
country	country
country	classical
country	country
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	jazz
country	jazz
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	jazz
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae

Blackman, k=3, Correctness: 36/50

Actual	Predicted
classical	jazz
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	country
country	classical
country	reggae
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	country
country	country
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	country
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	country
reggae	reggae

Blackman, k=1, Correctness: 38/50

Actual	Predicted
classical	classical
classical	classical
classical	country
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
country	metal
country	jazz
country	country
country	classical
country	reggae
country	reggae
country	reggae
country	country
country	country
country	country
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	reggae
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae

Hanning, k=5, Correctness: 37/50

Actual	Predicted
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	classical
classical	jazz
classical	classical
classical	classical
classical	classical
country	country
country	jazz
country	reggae
country	jazz
country	reggae
country	reggae
country	reggae
country	country
country	country
country	jazz
jazz	classical
jazz	reggae
jazz	jazz
jazz	jazz
jazz	jazz
jazz	jazz
jazz	classical
jazz	jazz
jazz	jazz
jazz	jazz
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
metal	metal
reggae	reggae
reggae	metal
reggae	metal
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae
reggae	reggae