تکلیف کامپیوتری پردازش گفتار و تشخیص صدا

نام درس: سیستم های چند رسانه ایی

استاد: دکتر بهزاد بختیاری

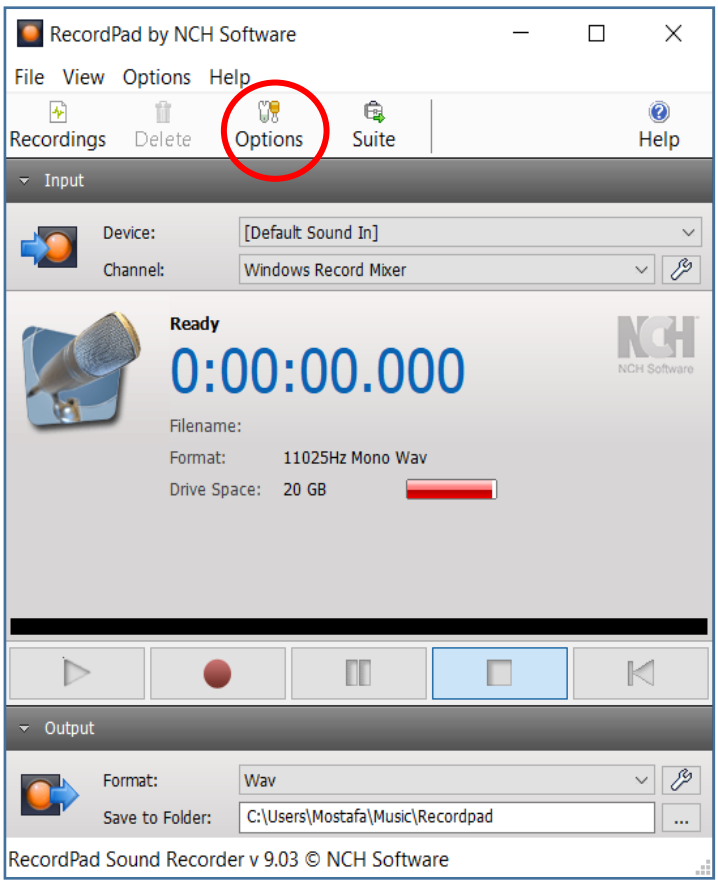
مصطفی نخعی نژاد فرد ۹۸۴۴۰۱۴۹

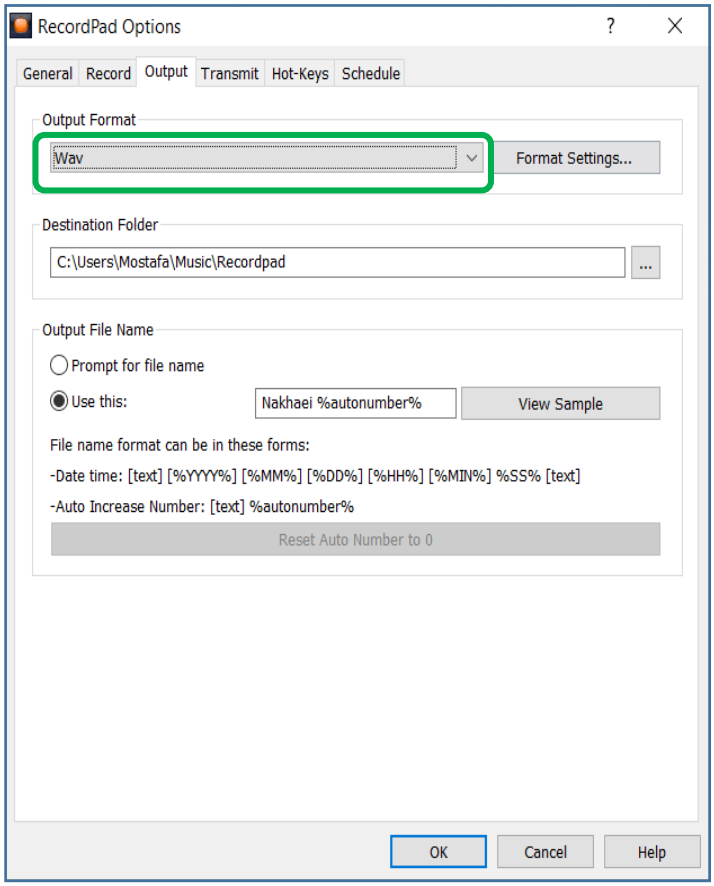
مقدمه:

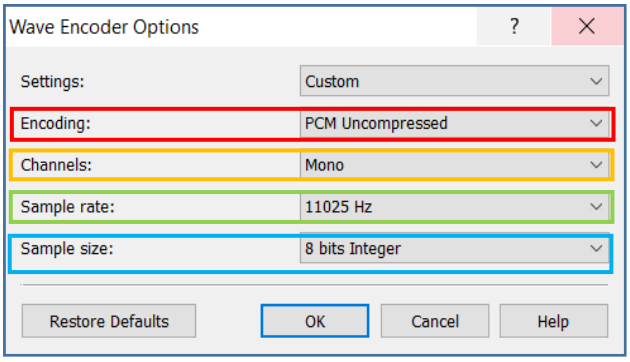
در این پروژه قصد داریم یک سیستم تصدیق هویت به کمک روش DTW بسازیم.

برای این منظور ابتدا لازم است چند مرتبه نام خانوادگی خود را گفته و صدا را ضبط کنیم، برای اینکار از نرم افزار RecordPad Sound Recorder استفاده میکنیم، که لازم است تنظیمات زیر را قبل از ضبط صدا اعمال کنیم:

فرمت باید wav باشد، انکودینگ باید pcm uncompressed باشد، کانال باید mono باشد، فرکانس نمونه برداری 11025 هرتز بوده و در نهایت اندازه نمونه 8 بیت باشد.







برای انجام این پروژه از کتابخانه های زیر استفاده کرده ایم:

Librosa: https://librosa.org/doc/latest/index.html

Numpy: https://numpy.org/

Fastdtw: https://pypi.org/project/fastdtw/

Matplotlib: https://matplotlib.org/

Spafe: https://spafe.readthedocs.io/en/latest/

```
import librosa
import numpy as np
from fastdtw import fastdtw
import matplotlib.pyplot as plt
from spafe.features.lpc import lpc
```

روش مستقیم

ابتدا باید فایل های صوتی را در برنامه بار گذاری کنیم که این کار با کمک تابع librosa.load انجام میشود.

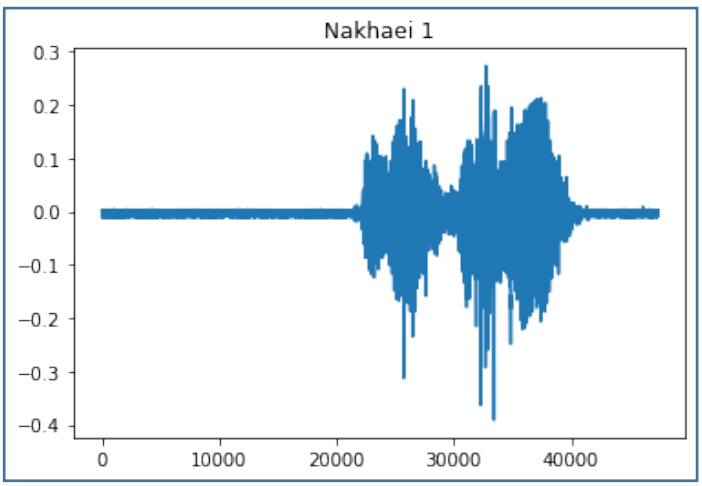
در ادامه باید یکسری پیش پردازش روی سیگنال های صدا انجام شود. مانند نرمال سازی و حذف سکوت و نویز. که اینکار را با پیاده سازی تابع prepare_signal و با کمک تابع librosa.util.normalize انجام داده ایم.

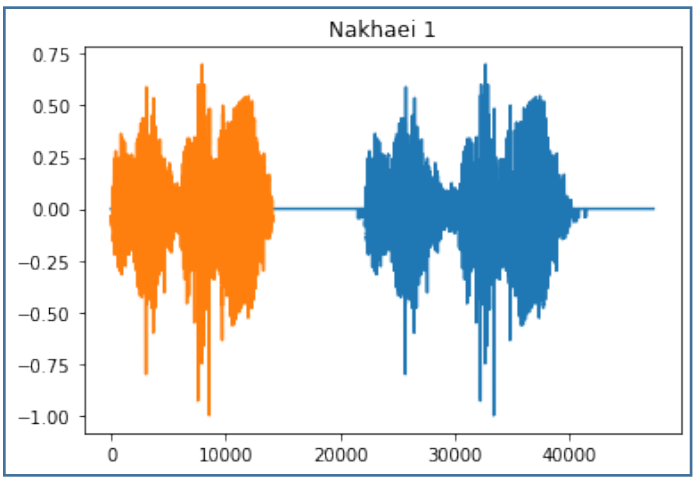
برای مشاهده موج صوت و تغییرات انجام شده بر روی آن از کتابخانه matplot استفاده میکنیم.

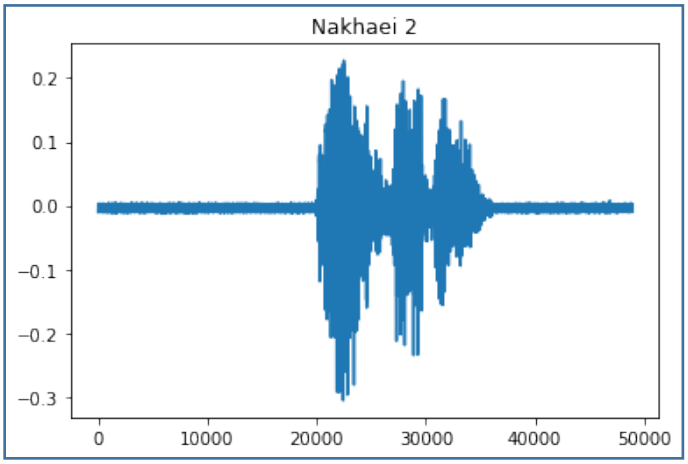
در روش مستقیم فایل های صوتی را مستقیما به کتابخانه fastdtw میدهیم و دو به دو با هم مقایسه میکنیم. و خروجی آن یک ماتریس میشود که قطر اصلی آن صفر است.

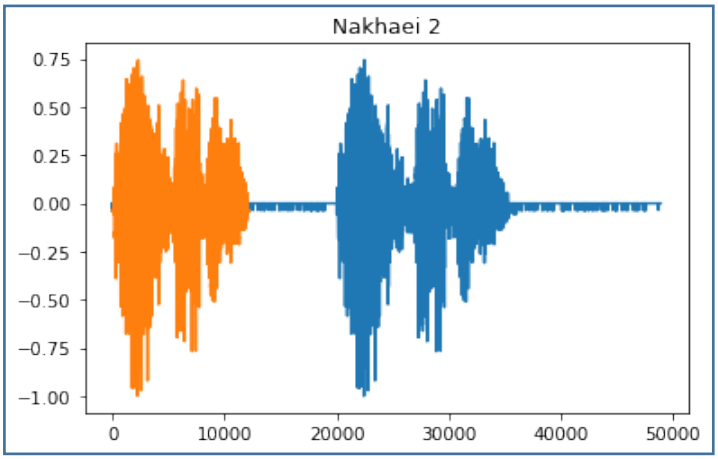
```
######################## DTW WITHOUT FEATURES ##############################################

lastname_count = 15
lastname_signals = []
figure_count = 5
def show_figure(signal,title,figure_number):
    plt.figure(figure_number)
    plt.title(title)
    plt.plot(signal)

for i in range(lastname_count):
    s, sample_rate = librosa.load("/content/Learn/Nakhaei ({}).wav".format(i+1))
    if i<figure_count:
      show_figure(s,"Nakhaei {}".format(i+1),i+1)
    lastname_signals.append(s)



def prepare_signals(signals,show=False):
    for i in range(len(signals)):
        temp = librosa.util.normalize(signals[i])
        for j in range(len(temp)):
            if abs(temp[j]) < 0.035 :
                temp[j] = 0
        if show and i<figure_count:
            show_figure(temp,"Nakhaei {}".format(i+1),figure_count+i+1)
        temp2 = []
        for data in temp:
            if abs(data) > 0.03:
                temp2.append(data)
        temp2 = np.array(temp2)
        if show and i<figure_count:
            show_figure(temp2,"Nakhaei {}".format(i+1),figure_count+i+1)
        signals[i] = temp2

    return signals

lastname_signals = prepare_signals(lastname_signals,True)

dtw = np.zeros((lastname_count,lastname_count))
for i in range(lastname_count):
    for j in range(lastname_count):
        dtw[i][j] , path = fastdtw(lastname_signals[i], lastname_signals[j])
```

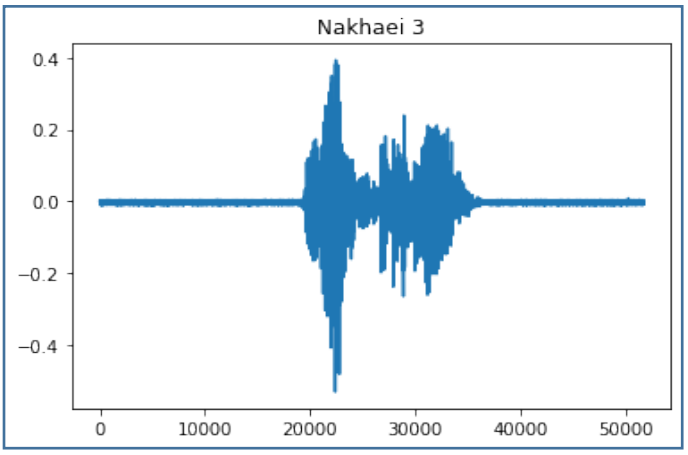
در ادامه برای نمونه چند نمودار از صدا های ضبط شده قبل و بعد از نرمال سازی و حذف سکوت و حذف نویز آورده میشود:

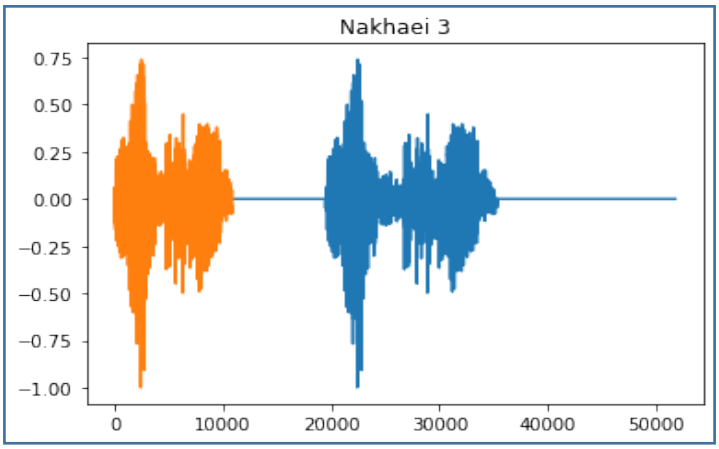












ماتریس خروجی حاصل از روش مستقیم به صورت زیر است:

/	voice 1	voice 2	voice 3	voice 4	voice 5	voice 6	voice 7	voice 8	voice 9	voice 10	voice 11	voice 12	voice 13	voice 14	voice 15
voice 1	0	1106.969	1031.182	1103.898	1156.071	976.7134	1017.577	1307.775	1318.136	997.9009	1054.406	995.79	1104.553	1175.442	2059.772
voice 2	1106.969	0	1293.176	2040.054	1194.133	1144.484	1064.978	1065.392	1304.454	1005.944	1147.904	1055.278	1142.523	947.2509	1024.175
voice 3	1031.182	1293.176	0	1568.521	819.8444	1017.462	1141.888	1162.349	1126.2	1031.644	1437.345	787.0176	1016.921	885.5487	1060.105
voice 4	1103.898	2040.054	1568.521	0	1100.901	1542.09	840.0695	974.6048	1122.796	1047.916	1160.822	970.8917	1155.422	971.8409	1895.96
voice 5	1156.071	1194.133	819.8444	1100.901	0	1183.032	1175.066	1233.376	1077.526	931.4883	1024.844	1046.992	1062.414	1050.198	1191.19
voice 6	976.7134	1144.484	1017.462	1542.09	1183.032	0	1201.444	1476.291	856.2214	879.9408	1005.772	1296.566	887.7904	1004.774	1076.73
voice 7	1017.577	1064.978	1141.888	840.0695	1175.066	1201.444	0	903.928	1072.738	956.4069	879.7117	1202.443	1036.994	863.0677	1064.91
voice 8	1307.775	1065.392	1162.349	974.6048	1233.376	1476.291	903.928	0	1281.915	1223.001	1371.778	990.1322	1509.93	2275.825	1128.639
voice 9	1318.136	1304.454	1126.2	1122.796	1077.526	856.2214	1072.738	1281.915	0	961.3832	890.9725	1347.855	1267.141	993.513	1541.645
voice 10	997.9009	1005.944	1031.644	1047.916	931.4883	879.9408	956.4069	1223.001	961.3832	0	973.8648	832.1069	1136.397	1097.613	965.6368
voice 11	1054.406	1147.904	1437.345	1160.822	1024.844	1005.772	879.7117	1371.778	890.9725	973.8648	0	1673.502	907.5552	943.5897	1976.391
voice 12	995.79	1055.278	787.0176	970.8917	1046.992	1296.566	1202.443	990.1322	1347.855	832.1069	1673.502	0	1291.087	905.3109	1219.329
voice 13	1104.553	1142.523	1016.921	1155.422	1062.414	887.7904	1036.994	1509.93	1267.141	1136.397	907.5552	1291.087	0	1217.498	1156.997
voice 14	1175.442	947.2509	885.5487	971.8409	1050.198	1004.774	863.0677	2275.825	993.513	1097.613	943.5897	905.3109	1217.498	0	864.4217
voice 15	2059.772	1024.175	1060.105	1895.96	1191.19	1076.73	1064.91	1128.639	1541.645	965.6368	1976.391	1219.329	1156.997	864.4217	0

در قطعه کد زیر ابتدا ماتریس بدست آمده را در یک فایل با پسوند **.CSV** ذخیره میکنیم. سپس برای پیدا کردن سیگنال مرجع ابتدا میانگین تمام سطر ها را محاسبه کرده و سپس میانگین کلی را محاسبه میکنیم، در نهایت سیگنالی به عنوان مرجع انتخاب میشود که میانگین سطر آن کمترین فاصله را با میانگین کلی داشته باشد.

```
file=open('DTW without Feature.csv','w')

for i in range(lastname_count+1):
    line = ''
    if i==0 :
        line = '/'
        for j in range(lastname_count):
            line += ',voice {}'.format(j+1)
    else :
        line = 'voice {},'.format(i) + ','.join(map(str, dtw[i-1]))
    line +='\n'
    file.write(line)
file.close()
dtw = np.array(dtw)

row_average = []
for i in range(lastname_count):
    row_average.append(np.sum(dtw[i])/(lastname_count-1))

total_average = np.average(row_average)
row_max = max(row_average)
row_min = min(row_average)
print("\nAverage = ",total_average)
print("Max = ",row_max)
print("Min = ",row_min,"\n")
for i in range(len(row_average)):
    print("Row {} Average = ".format(i+1),row_average[i])


def find_reference_signal(row_average,total_average):
    min_distance = abs(total_average - row_average[0])
    index = 0
    for i in range(len(row_average)):
        dist = abs(total_average - row_average[i])
        if dist < min_distance:
            min_distance = dist
            index = i
    return index


index = find_reference_signal(row_average,total_average)
reference_signal = lastname_signals[index]
print("Reference Signal = Last Name ({})".format(index+1))
```

```
Average =  1144.4106293422126
Max =  1413.680191164304
Min =  975.0790404781167

Row 1 Average =  1171.870378928525
Row 2 Average =  1244.4376573536013
Row 3 Average =  1084.458995508296
Row 4 Average =  1171.3990395364485
Row 5 Average =  1086.017518190933
Row 6 Average =  1122.7233877932388
Row 7 Average =  998.5833871872829
Row 8 Average =  1327.9462599102408
Row 9 Average =  1109.994025467496
Row 10 Average =  1038.4974174185522
Row 11 Average =  1144.3467728921346
Row 12 Average =  1121.2611470507193
Row 13 Average =  1155.8642212532993
Row 14 Average =  975.0790404781167
Row 15 Average =  1413.680191164304
Reference Signal = Last Name (11)
```

در ادامه چند دو مرتبه چند فایل صوتی از نام خانوادگی خود ضبط میکنیم و میخواهیم آزمایش کنیم که برنامه این صدا ها را تصدیق هویت میکند یا خیر.

برای این کار مانند قبل باید فایل ها در برنامه بار گذاری شده و پیش پردازش شوند.

در نهایت فایل های صوتی جدید با فایل سیگنال مرجع بدست آمده مقایسه میشوند و اگر اعداد بدست آمده در بازه مینیمم و ماکسیممی باشد که در مرحله قبل بدست آمدند تصدیق هویت انجام میشود. (1 یعنی تصدیق انجام شده و 0 یعنی تصدیق صورت نپذیرفته است.)

```
##################### TEST LAST NAME (MY VOICE) ################################################

test_lastname_count = 10
test_lastname_signals = []

for i in range(test_lastname_count):
    s, sample_rate = librosa.load("/content/Test/Test Nakhaei ({}).wav".format(i+1))
    test_lastname_signals.append(s)

test_lastname_signals = prepare_signals(test_lastname_signals)

dtw_test_ln = np.zeros(test_lastname_count)
result_test_ln = []
for i in range(test_lastname_count):
    dtw_test_ln[i] , path = fastdtw(reference_signal, test_lastname_signals[i])
    if row_min <= dtw_test_ln[i] and dtw_test_ln[i] <= row_max:
        result_test_ln.append(1)
    else:
        result_test_ln.append(0)

print("Result Last Name Test = ",dtw_test_ln)
print("Recognition Result = ",result_test_ln)
```

```
Result Last Name Test =  [1267.27855448 1144.57025251 1403.42340543 1514.23642429 1204.19851447
 1360.60492796 1173.28071896 1137.65361643 1184.91024324 1370.21862067]
Recognition Result =  [1, 1, 1, 0, 1, 1, 1, 1, 1, 1]
```

در ادامه میخواهیم این آزمایش را برای فایل های صوتی که افراد دیگر کلمات دیگری را گفته اند انجام دهیم.

مراحل انجام کار مانند قبل است.

```
##################### TEST OTHER WORDS (NOT MY VOICE) ##########################################

test_others_count = 10
test_others_signals = []

for i in range(test_others_count):
    s, sample_rate = librosa.load("/content/Test/Test Others ({}).wav".format(i+1))
    test_others_signals.append(s)

test_others_signals = prepare_signals(test_others_signals)

dtw_test_others = np.zeros(test_others_count)
result_test_others = []
for i in range(test_others_count):
    dtw_test_others[i] , path = fastdtw(reference_signal, test_others_signals[i])
    if row_min <= dtw_test_others[i] and dtw_test_others[i] <= row_max:
        result_test_others.append(1)
    else:
        result_test_others.append(0)

print("Result Others Test = ",dtw_test_others)
print("Recognition Result = ",result_test_others)
```

```
Result Others Test =  [1747.73749425 1598.63425955 2272.89654504 1856.20648333 1664.07502133
 2929.92354551 2119.0191853  3260.0789389  3416.4742026  2315.9752529 ]
Recognition Result =  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

روش استخراج ویژگی

در این روش فایل های صوتی باید فریم بندی شوند سپس اعمال استخراج ویژگی روی هر یک از فریم ها به صورت جداگانه انجام میشود.

ویژگی های اعمال شده عبارتند از:

- MFCC
- Zerocrossing
- Energy
- RMS
- LPC

در نهایت آرایه بدست آمده از ویژگی ها را به تابع fastdtw میدهیم و سیگنال ها را بر اساس ویژگی های استخراج شده آن ها دو به دو با هم مقایسه میکنیم.

```
###################### FEATURE EXTARCTION FUNCTIONS ############################################

def stride_trick(a, stride_length, stride_step):
    """
    apply framing using the stride trick from numpy.
    Args:
        a (array) : signal array.
        stride_length (int) : length of the stride.
        stride_step (int) : stride step.

    Returns:
        blocked/framed array.
    """
    nrows = ((a.size - stride_length) // stride_step) + 1
    n = a.strides[0]
    return np.lib.stride_tricks.as_strided(a,shape=(nrows, stride_length),strides=(stride_step*n, n))

def framing(sig, fs=11025, win_len=0.2, win_hop=0.1):
    """
    transform a signal into a series of overlapping frames (=Frame blocking).
    Args:
        sig     (array) : a mono audio signal (Nx1) from which to compute features.
        fs        (int) : the sampling frequency of the signal we are working with.
                          Default is 16000.
        win_len (float) : window length in sec.
                          Default is 0.025.
        win_hop (float) : step between successive windows in sec.
                          Default is 0.01.

    Returns:
        array of frames.
        frame length.

    Notes:
    ------
        Uses the stride trick to accelerate the processing.
    """
    # run checks and assertions
    if win_len < win_hop: print("ParameterError: win_len must be larger than win_hop.")

    # compute frame length and frame step (convert from seconds to samples)
    frame_length = win_len * fs
    frame_step = win_hop * fs
    signal_length = len(sig)
    frames_overlap = frame_length - frame_step

    # compute number of frames and left sample in order to pad if needed to make
    # sure all frames have equal number of samples  without truncating any samples
    # from the original signal
    rest_samples = np.abs(signal_length - frames_overlap) % np.abs(frame_length - frames_overlap)
    pad_signal = np.append(sig, np.array([0] * int(frame_step - rest_samples) * int(rest_samples != 0.)))

    # apply stride trick
    frames = stride_trick(pad_signal, int(frame_length), int(frame_step))
    return frames, frame_length

def add_feature(features,f):
    features.append(np.min(f))
    features.append(np.max(f))
    features.append(np.average(f))
    features.append(np.mean(f))
    return features

def feature_extraction(signals_frames):
    signals_features = []
    for signalframes in signals_frames:
        features=[]
        for frame in signalframes:
            ff = []

            # MFCC
            ff = add_feature(ff,librosa.feature.mfcc(frame,11025))
            # ZeroCrossing
            ff = add_feature(ff,
                librosa.feature.zero_crossing_rate(frame, 11025))
            # Energy
            ff = add_feature(ff,np.array([data * data for data in frame]))
            # RMS
            ff = add_feature(ff,librosa.feature.rms(frame,11025))
            # LPC
            ff = add_feature(ff,
                lpc(frame, fs=11025, num_ceps=13,  win_len=0.2, win_hop=0.1))

            features.append(ff)

        signals_features.append(features)
    return signals_features
```

بعد از فریم بندی و استخراج ویژگی ها و مقایسه آن ها توسط تابع fastdtw مانند قبل ماتریس بدست آمده را در فایل ذخیره کرده و میانگین های سطری و میانگین کلی را محاسبه میکنیم و بر اساس آن سیگنال مرجع را انتخاب میکنیم.

```
###################### FEATURE EXTARCTION ##########################################

lastnames_frames = []

for s in lastname_signals:
    frames, length = framing(s,11025)
    lastnames_frames.append(frames)

lastnames_features = feature_extraction(lastnames_frames)

dtw_with_feature = np.zeros((lastname_count,lastname_count))
for i in range(lastname_count):
    for j in range(lastname_count):
        dtw_with_feature[i][j] , path = fastdtw(lastnames_features[i], lastnames_features[j])


file=open('DTW with Feature.csv','w')

for i in range(lastname_count+1):
    line = ''
    if i==0 :
        line = '/'
        for j in range(lastname_count):
            line += ',voice {}'.format(j+1)
    else :
        line = 'voice {},'.format(i) + ','.join(map(str, dtw_with_feature[i-1]))
    line +='\n'
    file.write(line)
file.close()
dtw_with_feature = np.array(dtw_with_feature)

row_average = []
for i in range(lastname_count):
    row_average.append(np.sum(dtw_with_feature[i])/(lastname_count-1))

total_average = np.average(row_average)
row_max = max(row_average)
row_min = min(row_average)
print("\nAverage = ",total_average)
print("Max = ",row_max)
print("Min = ",row_min,"\n")
for i in range(len(row_average)):
    print("Row {} Average = ".format(i+1),row_average[i])

index = find_reference_signal(row_average,total_average)
reference_signal = lastname_signals[index]
print("Reference Signal = LastName",index+1)
```

/	voice 1	voice 2	voice 3	voice 4	voice 5	voice 6	voice 7	voice 8	voice 9	voice 10	voice 11	voice 12	voice 13	voice 14	voice 15
voice 1	0	761.3151	271.8558	371.5373	675.1367	325.9949	591.3871	380.8726	282.7153	296.314	327.5974	383.5336	416.2191	447.4134	504.1578
voice 2	761.3151	0	722.2048	611.3597	303.0794	678.5418	369.5392	558.8525	638.6858	716.5038	574.0763	521.0241	709.1031	543.6373	553.4039
voice 3	271.8558	722.2048	0	273.1428	607.9039	286.5866	609.5177	429.8608	270.8899	214.2696	385.4322	326.968	354.4461	288.6148	360.4552
voice 4	371.5373	611.3597	273.1428	0	530.2942	260.471	450.2045	267.3152	284.1239	274.4826	260.036	264.7695	220.6356	253.1172	248.7661
voice 5	675.1367	303.0794	607.9039	530.2942	0	634.9896	294.7656	501.2263	627.8438	640.7351	542.958	483.3196	551.8944	499.38	533.0017
voice 6	325.9949	678.5418	286.5866	260.471	634.9896	0	566.556	286.4897	165.7464	244.8573	274.1756	314.4178	331.002	404.4499	419.6595
voice 7	591.3871	369.5392	609.5177	450.2045	294.7656	566.556	0	410.2424	531.9329	566.0816	463.612	458.245	488.1689	455.1351	468.0242
voice 8	380.8726	558.8525	429.8608	267.3152	501.2263	286.4897	410.2424	0	297.0815	353.0772	184.9282	277.3651	317.978	332.4967	277.4977
voice 9	282.7153	638.6858	270.8899	284.1239	627.8438	165.7464	531.9329	297.0815	0	304.8449	288.6758	321.5818	363.268	424.688	378.153
voice 10	296.314	716.5038	214.2696	274.4826	640.7351	244.8573	566.0816	353.0772	304.8449	0	311.4526	306.7489	409.1183	355.8386	408.0343
voice 11	327.5974	574.0763	385.4322	260.036	542.958	274.1756	463.612	184.9282	288.6758	311.4526	0	226.0376	359.4275	328.6594	326.7468
voice 12	383.5336	521.0241	326.968	264.7695	483.3196	314.4178	458.245	277.3651	321.5818	306.7489	226.0376	0	344.1292	182.0402	261.1025
voice 13	416.2191	709.1031	354.4461	220.6356	551.8944	331.002	488.1689	317.978	363.268	409.1183	359.4275	344.1292	0	326.3201	311.728
voice 14	447.4134	543.6373	288.6148	253.1172	499.38	404.4499	455.1351	332.4967	424.688	355.8386	328.6594	182.0402	326.3201	0	227.6333
voice 15	504.1578	553.4039	360.4552	248.7661	533.0017	419.6595	468.0242	277.4977	378.153	408.0343	326.7468	261.1025	311.728	227.6333	0

```
Average =  402.1326553015249
Max =  590.0947732683524
Min =  326.4468240538327

Row 1 Average =  431.14643669603
Row 2 Average =  590.0947732683524
Row 3 Average =  385.86774086018943
Row 4 Average =  326.4468240538327
Row 5 Average =  530.4662969834266
Row 6 Average =  370.99556760636534
Row 7 Average =  480.24372466352236
Row 8 Average =  348.23456254243246
Row 9 Average =  370.01649369020146
Row 10 Average =  385.88276940553084
Row 11 Average =  346.7011030559083
Row 12 Average =  333.66307488131105
Row 13 Average =  393.1027444819906
Row 14 Average =  362.1017144436046
Row 15 Average =  377.02600289017494
Reference Signal = LastName 13
```

حال دو آزمایش قبل را با استخراج ویژگی ها تکرار میکنیم.

```
#################### TEST LAST NAME WITH FEATURE (MY VOICE) ################################################

test_ln_frames = []

for s in test_lastname_signals:
    frames, length = framing(s,11025)
    test_ln_frames.append(frames)

test_ln_features = feature_extraction(test_ln_frames)

dtw_test_ln_feature = np.zeros(test_lastname_count)
result_test_ln_feature = []
for i in range(test_lastname_count):
    dtw_test_ln_feature[i] , path = fastdtw(lastnames_features[index], test_ln_features[i])
    if row_min <= dtw_test_ln_feature[i] and dtw_test_ln_feature[i] <= row_max:
        result_test_ln_feature.append(1)
    else:
        result_test_ln_feature.append(0)

print("Result Last Name Feature Test = ",dtw_test_ln_feature)
print("Recognition Result = ",result_test_ln_feature)
```

```
Result Last Name Feature Test =  [350.00244533 375.81143182 435.73137398 423.11136747 488.62640057
 395.89321872 375.06995412 366.28341491 421.73251121 388.65656812]
Recognition Result =  [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

```
#################### TEST OTHER WORDS WITH FEATURE (NOT MY VOICE) ###########################################


test_others_frames = []

for s in test_others_signals:
    frames, length = framing(s,11025)
    test_others_frames.append(frames)

test_others_features = feature_extraction(test_others_frames)

dtw_test_others_feature = np.zeros(test_others_count)
result_test_others_feature = []
for i in range(test_others_count):
    dtw_test_others_feature[i] , path = fastdtw(lastnames_features[index], test_others_features[i])
    if row_min <= dtw_test_others_feature[i] and dtw_test_others_feature[i] <= row_max:
        result_test_others_feature.append(1)
    else:
        result_test_others_feature.append(0)

print("Result Others Feature Test = ",dtw_test_others_feature)
print("Recognition Result = ",result_test_others_feature)
```

```
Result Others Feature Test =  [ 609.64178273  802.56435736 1010.56206984  686.08454729  640.75902657
  749.23625528  852.34978099  688.54621938  749.59503038  823.78872832]
Recognition Result =  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

نتایج

- ✓ نتایج بدست آمده در حالت پیش از حذف سکوت به دلیل اینکه درصد زیادی از حجم فایل صوتی سکوت میباشد به هم شباهت داشته که این دور از واقعیت است و باید برای دقت بیشتر قبل از انجام محاسبات سکوت ها و نویز های فایل های صوتی حذف شوند.
- ✓ روش استخراج ویژگی نسبت به روش مستقیم اعداد بدست آمده در جدول کوچکتر هستند که این کوچکتر بودن نشان از شباهت بالا و فاصله کم و دقت بیشتر است.
- ✓ سرعت انجام روش استخراج ویژگی نسبت به روش مستقیم بیشتر است البته زمان استخراج ویژگی ها بدلیل انجام محاسبات ممکن است زمان گیر بوده و زمان کلی را افزایش دهد.