

refs

DTW core code from <https://github.com/Linzecong/MFCC-DTW>

Audio dataset from <https://www.kaggle.com/datasets/sripaadsrinivasan/audio-mnist>, using python librosa to extract MFCC feature

Python FastDTW

Accuarcy

97%

```
问题 36 输出 调试控制台 终端 JUPYTER
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

尝试新的跨平台 PowerShell https://aka.ms/powershell

PS C:\Users\28147> python -u "g:\hls\py_mfcc_n_dtw\dtw.py"
RECOGNITION CORRECT RATE: 0.97
TOTAL EXECUTE TIME: [3055.6016016 2714.78509426 2613.37055659 2749.54712939 2764.4760685
2826.98189378 3156.58651423 3132.39928794 2696.43909335 2943.10387087]
PS C:\Users\28147>
```

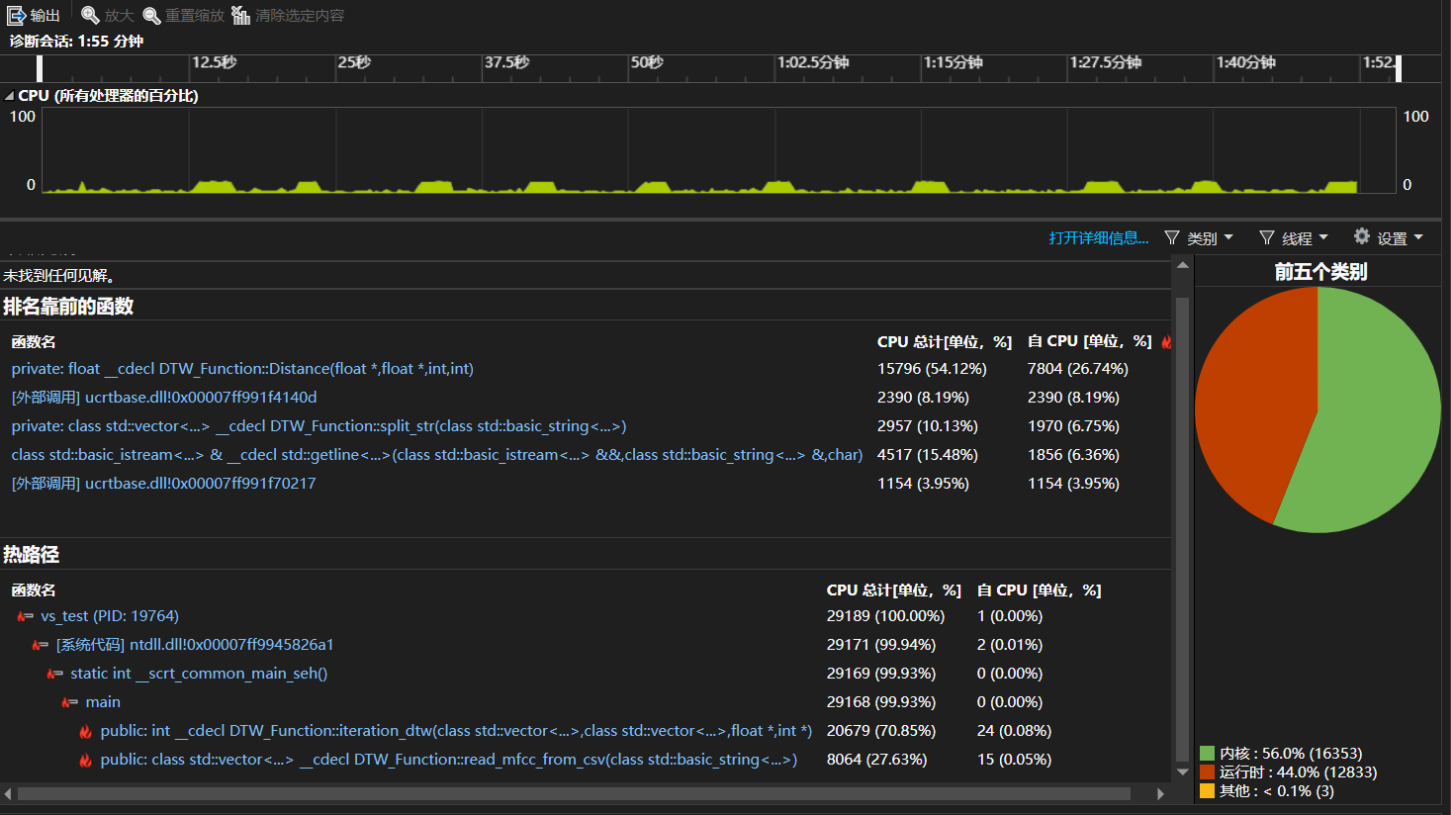
C++ DTW

Accuarcy

Only 10% 😊

```
Microsoft Visual Studio 调试控制台
Recognition execute time:2947s
correct times:10
wrong times:90
Recognition Correct rate:10%
```

Utilization



CPU Details

```
216     vector<vector<float> > match_mfcc_feat;
1 (0.00%) 217     for (int i = 0; i < match_csv_content.size(); i++)
218     {
3 (0.01%) 219         if((match_csv_content[i].size() % 13) != 0){
220             cout << "WRONG READ!" << "MFCC Length is: " << match_csv_content[i].size() << endl;
221             /* for (int j = 0; j < match_csv_content[i].size(); j++)
222             {
223                 cout << match_csv_content[i][j] << endl;
224             } */
225             break;
226         }
227         else{
228             float distance = 0;
229             float normal_distance = 0;
3372 (11.55%) 230             match_mfcc_feat = rebuilt_mfcc_feat(match_csv_content[i]);
17208 (58.95%) 231             distance = ComputeDTW(query_mfcc_feat,match_mfcc_feat);
232             normal_dist = distance / (query_mfcc_feat.size() + match_mfcc_feat.size());
233             eud_distance.push_back(normal_dist);
234         }
235     }
236     //eud_distance = standard_normal(eud_distance); //no standard
237     auto min_pos = std::min_element(eud_distance.begin(), eud_distance.end());
238     *min_idx = (min_pos - eud_distance.begin());
239     *min_dist = *min_pos;
```

```
132
59 (0.20%) 133 float DTW_Function::Distance(float *ps1, float *ps2, int k1, int k2) {
134     int i=0;
135     float sum=0;
42 (0.14%) 136     for(i=0;i<MFCC_P;i++)
15593 (53.42%) 137         sum+=(1+MFCC_Pf/2*(float)sin(PI*i/MFCC_Pf))*(ps1[k1+i]-ps2[k2+i])*(ps1[k1+i]-ps2[k2+i]);
138
67 (0.23%) 139     return sum;
35 (0.12%) 140 }
141
```

Function Distance used 53.42%.

Memory Details

