哈爾濱工業大學

实验报告

实 验（三）

题　　目 命令词识别实验报告

专　　业 人工智能 视听觉信息处理

学　　号 1190202107

班　　级 1903602

学　　生 姚舜宇

指 导 教 师 郑铁然

实 验 地 点 格物 213

实 验 日 期 2021.12.18

计算机科学与技术学院

一、 设计命令词识别任务

1.1 描述所设计的命令词识别任务

假设有一个机器人，可以通过语音进行控制它的行动。通过说话给出命令，让机器人可以直行、后退、左转、右转、上台阶、踢球、唱歌。这些命令使得机器人能够供人类进行相关情景下的工作或者娱乐。

实验步骤：

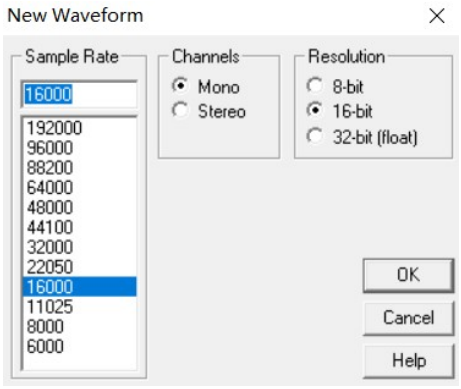
1. 录制语料。总共有 7 个命令词，每个词录制 10 遍，每个词去一个作为模板，其余 9 个作为测试用例，共有 7*9=63 个测试用例。使用 cooledit 进行录制，参数为：采样率 16000hz，单通道，16-bit 分辨率。
2. 对语料进行去除静音，使用实验一中的程序进行处理。
3. 使用 HTK 中的 Hcopy 工具对语料进行特征提取，将提取出的 MFCC 特征保存在.mfc 文件中。
4. 在代码中从 mfc 文件提取出语料的数据，使用 DTW 算法将每个测试样例和每个模板进行比较，选择距离最小的类别作为预测的结果，并且统计正确的个数以及计算准确率。

1.2 列出词表

直行
后退
左转
右转
上台阶
踢球
唱歌

1.3 介绍语料采集方法和规模，

使用 cooledit 进行采集，参数为：采样率 16000hz，单通道，16-bit 分辨率。



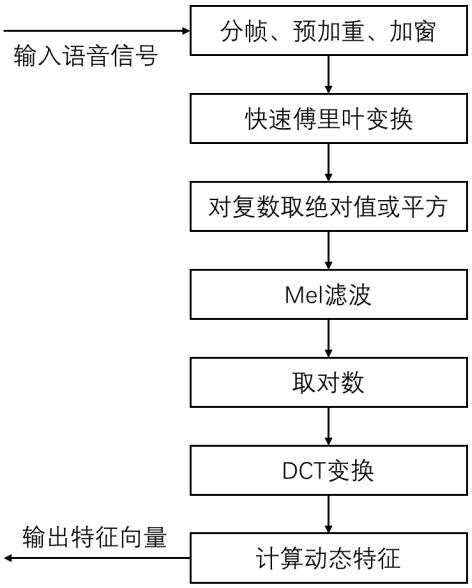
7 个命令词，每个命令词录制 10 遍，每个语音约为 2 秒。其中选择 1 个作为模板，其余 9 个作为测试用例。

二、特征提取

2.1 详细描述你所采用的特征和提取算法

Mfcc 特征提取算法：

算法流程：



分帧：由于原始 wav 音频文件是不定长的，所以首先需要将其按照一定的方法切分成为固定长度的多个小段。一般每一帧的长度要使得有足够多的周期，并且变化不会过于剧烈。为了避免时间窗的边界导致信息遗漏的问题，在对从信号中取每一帧的时间窗进行偏移时，帧和帧之间需要有一部分重叠区域，一般取帧长的一半。

预加重：由于声音信号从人的声门发出后，存在 12dB/倍频程的衰减，在通过口唇辐射后，还存在 6dB/倍频程的衰减，所以在进行 FFT 后，高频信号部分中的成分较少，所以需要对语音信号进行预加重操作。主要目的是加强语音信号的每一帧中高频部分的信号，以提高高频信号的分辨率。可以采用以下的一阶高通滤波器进行预加重操作：

$$H(z)=1-\alpha\times z^{-1}$$

$$S(n)=S(n)-\alpha\times S(n-1)\ \ \forall n\in N$$

在上式中，α 是预加重的系数，n 表示当前处理的是第 n 帧，n=0 的帧需要特别处理。

加窗：在分帧过程中，直接将一个连续的语音信号切分为若干个片段，会造成截断效应产生的频谱泄露，加窗的目的是消除每个帧的短时信号在其两端边缘处出现的信号不连续性问题。在这里一般选择哈明窗。

哈明窗的窗函数为：

$$W(n)=\begin{cases}0.54-0.46\cos[2\pi n/(N-1)] & 0\le n\le N-1\\ 0 & otherwise\end{cases}$$

加窗过程为：

$$S'(n)=W(n)\times S(n)$$

快速傅里叶变换：

在经过以上处理后，需要将时域信号转化成为频域信号。使用快速傅里叶变换来实现。

$$P(n)=\sum_{k=0}^{N-1}S(n)\times e^{-j\cdot\frac{2\pi kn}{N}}\ (0<n<N)$$

根据奈奎斯特定理，如果要再次从离散的数字信号无损转换到模拟信号上，需要采用模拟信号最高频率值的 2 倍以上的采样率。

计算幅度谱：

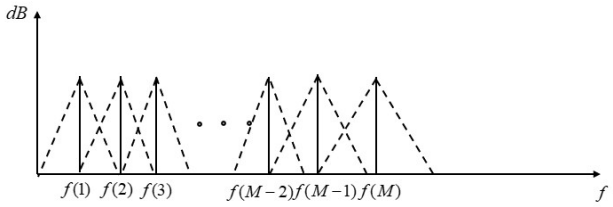
完成 FFT 后，得到的特征是一个复数矩阵，为一个能量谱。由于能量谱中的相位谱包含的信息量较少，所以一般只保留幅度谱。方法一般有两种，对复数求绝对值或平方：

$$P'(n)=\sqrt{P^2(n)}$$

$$P'(n)=P^2(n)$$

Mel 滤波：

Mel 滤波器是由 20 个三角形带通滤波器组成的，将线性频率转换为非线性分布的 Mel 频率。



Mel 倒谱公式：

$$Mel(f)=2595\times\log_{10}\left(1+\frac{f}{700}\right)=1125\times\ln\left(1+\frac{f}{700}\right)$$

Mel 滤波器：

$$B_m[k]=\begin{cases}0 & k<f_{m-1}\ or\ k>f_{m+1}\\ \dfrac{k-f_{m-1}}{f_m-f_{m-1}} & f_{m-1}\le k\le f_m\\ \dfrac{f_{m-1}-k}{f_{m+1}-f_m} & f_m\le k\le f_{m+!}\end{cases}$$

Mel 滤波公式：

$$E_m=\ln\left(\sum_{k=0}^{N-1}P(k)\times H_m(k)\right)$$

经过 Mel 滤波后，E_m 即为得到的 fBank 特征。

取对数：

得到上一步的特征后，由于人耳对声音的感受是成对数增长的，所以需要将数值进行一次对数运算以模拟人耳的感受。

离散余弦变换：

在取了对数后，需要对得到的 N 维特征向量值再进行一次离散余弦变换。原因是不同阶数的信号值之间具有一定的相关性，现需要去除这些相关性，将信号映射到低维空间中。一般保留前 12-20 个结果值，一般取 13 个。公式如下：

$$C_i=\sqrt{\frac{2}{N}}\sum_{j=1}^{N}E_j\times\cos\left(\frac{\pi\times i}{N}\cdot(j-0.5)\right),\forall i\in[1,M]$$

计算动态特征：上述算法仅体现出 MFCC 静态特征，而动态特征需要使用静态特征的差分来表示。通过将得到的动态的特征，和前一步得到的静态特征相结合，可以有效提高语音识别系统的性能。差分参数计算公式如下：

$$d_t=\begin{cases}C_{t+1}-C_t & t<K\\ \dfrac{\sum_{k=1}^{K}k(C_{t+k}-C_{t-k})}{\sqrt{2\sum_{k=1}^{K}k^2}} & otherwise\\ C_t-C_{t-1} & t\ge Q-K\end{cases}$$

其中d_t是第 t 个一阶差分值，C_t是第 t 个倒谱系数值，Q 是倒谱系数的最大阶数，K 是一阶差分的时间差。二阶差分将上式的结果再代入进行计算可以得到。

最后，再将静态特征和动态特征的一阶、二阶差分值合并起来，当静态特征是 13 维的特征向量时，合并动态特征后总共有 39 维特征。

Mfcc 特征提取方法：

使用 HTK 工具包进行特征提取。

首先配置好 list.scp 文件，包括.wav 文件的目录和生成的.mfc 特征文件的目录。如下图所示：

```
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行1.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行2.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行3.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行4.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行5.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行6.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行7.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行8.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行9.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词i
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行10.wav D:\2021秋\视听觉信号处理\听觉部分\实验\命令词
```

使用命令 hcopy -A -D -T 1 -C tr_wav.cfg -S list.scp 运行，即可得到所有.mfc 文件。

读取.mfc 文件的方法如下

```python
def readmfc(filename):
    file = open(filename, 'rb')
    nframes, frate, nbytes, feakind = struct.unpack('>IIHH', file.read(12))
    ndim = nbytes // 4
    ret = np.empty((nframes, ndim))
    for i in range(nframes):
        for j in range(ndim):
            mf = file.read(4)
            c = struct.unpack('>f', mf)
            ret[i][j] = c[0]
    file.close()
    return ret
```

2.2 给出特征提取部分运行结果的截图

```
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\MFCC>HCopy -A -D -T 1 -C tr_wav.cfg -S list.scp
HCopy -A -D -T 1 -C tr_wav.cfg -S list.scp

HTK Configuration Parameters[21]
  Module/Tool     Parameter                Value
# HREC            FORCEOUT                 TRUE
# HNET            TRACE                       1
# HLABEL          TRACE                       8
# HPARM           TRACE                      65
# HSHELL          TRACE                       2
#                 FORCECXTEXP              TRUE
#                 ALLOWXWRDEXP             TRUE
#                 ENORMALIZE               TRUE
#                 NUMCEPS                    12
#                 CEPLIFTER                  22
#                 NUMCHANS                   26
#                 PREEMCOEF           0.970000
#                 USEHAMMING               TRUE
#                 WINDOWSIZE     250000.000000
#                 SAVEWITHCRC              TRUE
#                 SAVECOMPRESSSED          TRUE
#                 TARGETRATE     100000.000000
#                 TARGETKIND          MFCC_E_D_A_Z
#                 ZMEANSOURCE             FALSE
#                 SOURCEFORMAT              WAV
#                 SOURCEKIND            WAVEFORM

HParm: Parm tab type MFCC_E_D_A_Z saved to D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行1.mfc [sampSize=156,nSamples=195] with CRC
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行1.wav ->
    D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行1.mfc
HParm: Parm tab type MFCC_E_D_A_Z saved to D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行2.mfc [sampSize=156,nSamples=261] with CRC
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行2.wav ->
    D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行2.mfc
HParm: Parm tab type MFCC_E_D_A_Z saved to D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行3.mfc [sampSize=156,nSamples=257] with CRC
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行3.wav ->
    D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行3.mfc
HParm: Parm tab type MFCC_E_D_A_Z saved to D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行4.mfc [sampSize=156,nSamples=238] with CRC
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行4.wav ->
    D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行4.mfc
HParm: Parm tab type MFCC_E_D_A_Z saved to D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行5.mfc [sampSize=156,nSamples=252] with CRC
D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\语料\直行5.wav ->
    D:\2021秋\视听觉信号处理\听觉部分\实验\命令词识别实验3\mfc\直行5.mfc
```

2.3 给出特征文件内容的截图（二进制打开或逐维读出到文本文件）

直行1.mfc

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	ANSI ASCII
00000000	00	00	00	5A	00	01	86	A0	00	9C	1B	46	C1	6E	E0	FF	Z † œ FÁnàÿ
00000016	C1	3C	69	AE	C1	0C	4B	DE	BF	46	9E	18	BF	64	4D	C8	Á<i®Á KÞ¿Fž ¿dMÈ
00000032	C0	93	C9	2C	40	B7	3A	B4	BE	E8	E4	9F	C0	D3	95	63	À"É,@·:´¾èäŸÀÓ•c
00000048	BF	D2	DA	1C	40	B8	2C	8C	40	8F	48	7C	3E	B2	59	93	¿ÒÚ @¸,Œ@ H|>²Y"
00000064	BF	8D	95	05	40	1F	43	AE	40	4E	A6	A9	BD	11	93	9A	¿ • @ C®@N¦©½ "š
00000080	3F	1B	80	5F	3F	D8	64	46	3F	B9	66	54	3F	9F	A9	24	? €_?ØdF?¹fT?Ÿ©$
00000096	40	2C	78	CB	BE	4F	F6	23	BF	39	76	2E	3F	15	55	38	@,xË¾Oö#¿9v.? U8
00000112	3C	5A	C6	DD	3F	42	46	BB	3E	2C	0C	E5	BD	79	04	2D	<ZÆÝ?BF», å½y -
00000128	3E	37	C1	3C	3D	EA	91	32	3E	9A	DD	5B	BE	65	D9	3F	>7Á<=ê'2>šÝ[¾eÙ?
00000144	BF	5A	A5	C8	BE	DB	FB	6B	3E	1C	E1	95	BE	41	F9	40	¿Z¥È¾ÛÛk> á•¾Aù@
00000160	BE	CD	0D	15	3B	DB	F4	9E	C1	9B	6D	1C	C0	8D	8C	44	¾Í ;ÛôžÁ›m À ŒD
00000176	40	09	3A	A0	3F	A2	C8	DE	3F	3B	6C	40	BF	74	8E	72	@ : ?¢È Þ?;l@¿tŽr
00000192	41	22	2A	84	40	C0	E5	F1	40	6D	29	37	BF	6C	2B	10	A"*„@Àåñ@m)7¿l+
00000208	BF	AA	CB	BE	40	89	DA	BD	3E	C5	ED	06	3E	A6	C1	53	¿ªË¾@‰Ú½>Åí >¦ÁS
00000224	40	2B	AA	F3	40	42	75	A4	BD	B8	42	BB	3F	7F	65	24	@+ªó@Bu¤½¸B»? e$
00000240	40	4D	45	7B	3F	BE	A1	33	3C	FD	FD	4D	40	03	3B	D9	@ME{?¾¡3<ýýM@ ;Ù
00000256	3E	A3	67	BA	BF	68	46	AF	BF	2F	24	BB	3C	D1	43	6E	>£g°¿hF¯¿/$»<ÑCn
00000272	3F	A1	97	15	3E	0D	4F	18	BE	B0	9D	66	3E	A0	AB	7D	?¡— > O ¾° f> «}
00000288	BE	91	57	E4	BE	C2	EC	A1	BF	39	2F	65	BF	AA	21	5C	¾'Wä¾Âì¡¿9/e¿ª!\
00000304	BF	64	ED	0C	3E	F4	DB	38	BE	CD	BF	86	BE	B4	C9	CF	¿dí >ôÛ8¾Í¿†¾´ÉÏ
00000320	3C	23	EA	D3	C1	91	B0	C2	C0	40	9B	6F	3F	F5	B9	90	<#êÓÁ'°ÂÀ@›o?õ¹
00000336	BF	FD	1A	11	3F	AA	AB	88	40	00	30	FA	41	2C	36	BA	¿ý ?ª«ˆ@ 0úA,6°
00000352	40	22	96	0C	3F	DA	0E	68	C0	41	9B	1E	40	B5	B2	6D	@"– ?Ú hÀA› @µ²m
00000368	40	EF	54	9E	3E	CA	BE	EC	3F	FC	79	22	40	4E	D4	13	@ïTž>Ê¾ì?üy"@NÔ
00000384	40	41	4A	D8	3F	63	8F	E0	3F	7C	28	BC	40	1C	72	D0	@AJØ?c à?|(¼@ rÐ
00000400	3E	9C	84	74	C0	1A	8E	5C	3F	5E	68	84	3E	9A	87	7E	>œ„tÀ Ž\?^h„>š‡~
00000416	BF	CA	42	BF	BF	49	0E	84	3D	27	32	9A	3F	A9	56	33	¿ÊB¿¿I „='2š?©V3
00000432	BD	8E	90	F3	BE	F3	AA	FE	3E	DE	6C	88	BF	54	2A	C7	½Ž ó¾óªþ>Þlˆ¿T*Ç
00000448	BF	A4	B9	EC	BF	8A	AD	D4	BF	67	17	03	BF	63	96	3C	¿¤¹ì¿Š-Ô¿g ¿c–<
00000464	3F	40	4A	65	BF	32	20	7C	BE	92	93	6E	3C	26	EB	5A	?@Je¿2 |¾'"n<&ëZ
00000480	C1	3A	92	87	C0	2F	D5	A0	3F	8A	2D	34	BF	1F	FC	C8	Á:'‡À/Õ ?Š-4¿ üÈ
00000496	40	3E	F6	60	41	01	B9	DA	41	2A	75	6A	BF	E6	13	FE	@>ö`A ¹ÚA*uj¿æ þ

三、 基于 DTW 的命令词识别

3.1 介绍你所设计 DTW 算法，标明所采用的开发工具

开发工具：
Python3.9，pycharm2021.1.3，Windows 10 64 位

DTW 算法：

```
def DWT(template, test):
    """
    计算两个序列的 DWT 距离
    :param template: 模板序列 shape:n*39
    :param test: 测试序列 shape:m*39
    :return: DWT 距离
    """
    len_template = template.shape[0]
    len_test = test.shape[0]
    phi = np.empty((len_template + 1, len_test + 1))
    for i in range(1, len_template + 1):
        phi[i][0] = np.inf
    for j in range(1, len_test + 1):
        phi[0][j] = np.inf
    phi[0][0] = 0
    sum_w = 0
    for i in range(1, len_template + 1):
        for j in range(1, len_test + 1):
            d = distance(template[i - 1], test[j - 1])  # 欧氏距离
            a1 = phi[i - 1][j] + d
            a2 = phi[i][j - 1] + d
            a3 = phi[i - 1][j - 1] + 2 * d
            minimum = min(a1, a2, a3)
            phi[i][j] = minimum
            if minimum == a1:
                sum_w += 1
            elif minimum == a2:
                sum_w += 1
            else:
                sum_w += 2
    return phi[len_template][len_test] / sum_w
```

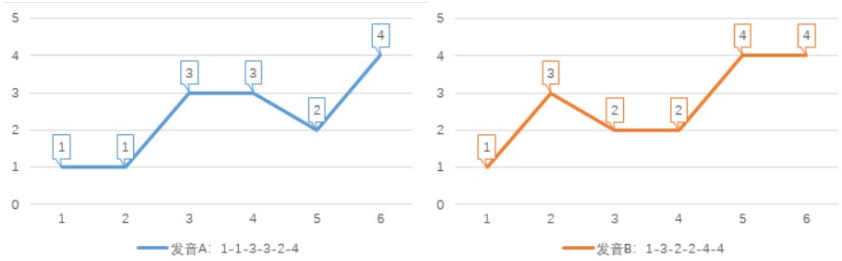
DTW 算法是一种模板匹配技术，是基于相似度计算与匹配实现的识别方法。一般情况下，计算相似度的方法有以下几种：
计算两个标量x_1与x_2的相似度：$d=|x_1-x_2|$
计算两个矢量$\vec{x}_1=\{x_{11},...,x_{1n}\}$和$\vec{x}_2=\{x_{21},...,x_{2n}\}$的相似度，即用欧氏距离进行表

示：$d(\vec{x}_1,\vec{x}_2)=\sum_{i=1}^{n}(x_{1i}-x_{2i})^2$

经过预处理和特征提取后的语音可以看作矢量的序列：

$X=(\vec{x}_1,\vec{x}_2,...,\vec{x}_M)$

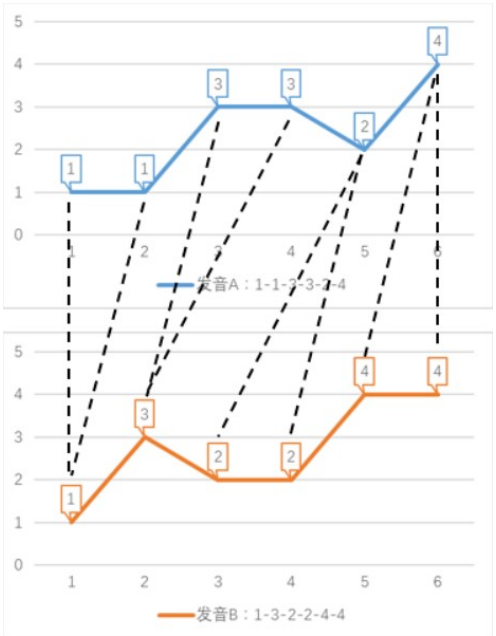
如果用朴素的方法计算相似度会有以下两个问题：长度不同，对不准。另外，在实际情况中，可能会出现每个发音长度不同的情况，如下图所示。



这两个语音虽然每个发音的长度不同，但实际上是同一个含义。如果使用传统的欧氏距离进行衡量，其距离为 6。所以算法对这样的区别不能过于敏感。

DTW 算法将表示两个语音段的矢量序列对准了再计算相似度，或者说在时间上归正后再计算相似度。

以上图为例，如果允许序列的点与另一序列的多个连续的点相对应，然后再计算对应点距离的和，则效果如下图所示：



在这种情况下再进行计算，结果为 0。

将问题进行转化，就成为了求矩阵中的最短路径问题。从矩阵的左下角到右上角，寻找到一条最短的路径。使用动态规划算法进行计算。

定义一个代价函数$\Phi(i,j)$表示从起始点（1，1）出发，到达（i，j）点最小代价路径的累积距离。则存在以下递推式：

$$\Phi(i,j)=\min_{(i',j')\to(i,j)}\{\Phi(i',j')+d(\vec{x}_{1i},\vec{x}_{2j})w_n\}$$

则：$\Phi(M_1,M_2)=\min\begin{Bmatrix}\Phi(M_1-1,M_2)+d(\vec{x}_{1M_1},\vec{x}_{2M_2})w_n,\\ \Phi(M_1,M_2-1)+d(\vec{x}_{1M_1},\vec{x}_{2M_2})w_n,\\ \Phi(M_1-1,M_2-1)+d(\vec{x}_{1M_1},\vec{x}_{2M_2})w_n\end{Bmatrix}$．以此类推，$\Phi(M_1-1,M_2)$、$\Phi(M_1,M_2-1)$、$\Phi(M_1-1,M_2-1)$可以由更低一层的代价函数计算得到。其中加权系数的取值与局部路径有关：$w_n=\begin{cases}2 & (i-1,j-1)\to(i,j)\\ 1 & otherwise\end{cases}$。

在算法的初始化中，$i=j=1,\Phi(1,1)=d(\vec{x}_{11},\vec{x}_{21})$，$\Phi(i,j)=\begin{cases}0 & if\ (i,j)\in \mathrm{Reg}\\ inf & otherwise\end{cases}$。

然后递归求累积距离即可。最后对所求得的$\Phi(M_1,M_2)$使用$\sum W_n$进行规正。

3.2 正确率：

正确率 98.4%

四、 基于 HMM 的命令词识别

4.1 介绍你所实现的算法

4.2 正确率：

五、 总结

5.1 请总结本次实验的收获

了解了 MFCC 特征提取算法的基本流程。
熟悉了 DTW 动态时间归正算法的理论知识，以及动态规划算法的思想。
独立实现了命令词识别的任务。

5.2 请给出对本次实验内容的建议

无。