Speech Recognition

SMO GROUP

29 March 2016

1. Introduction

Blablabla

2. Dynamic Time Warping

Blablabla

2.1 Algorithm

2.2 Modifications

2.3 Applications

3. Speech Recognition

AINA Blablabla, data, blablabla

3.1 Speech Processing

AINA Blablabla

3.2 Dynamic Time Warping in this project

Blablabla

3.3 Results

Blablabla

4. Conclusions

Blablabla

5. Bibliography

6. Annex

6.1 DTW function

```
TimeWarp<-function(x,y,w=4){

  # define distance function
  distance<-function(a,b){
    dist(rbind(a,b))
    }

  # 1. Compute matrix 11xM

  # set parameters
  m<-dim(x)[2]
  n<-dim(y)[2]
  colnames(x)<-1:m
  colnames(y)<-1:n
  w = max(w, abs(n-m))

  # Create matrix
  DTW<-matrix(Inf,n,m)
  rownames(DTW)<-n:1
  colnames(DTW)<-1:m

  # Initial values
  DTW['1','1']<-distance(x[,'1'], y[,'1'])

   # First row
  for(j in 2:(w+1)){
    cost<-distance(x[,as.character(j)], y[,as.character(1)])
    DTW['1',as.character(j)]<- cost + DTW['1', as.character(j-1)]
  }

   # First column
  for(i in 2:(w+1)){
    cost<-distance(x[,as.character(1)],y[,as.character(i)])
    DTW[as.character(i), '1']<- cost + DTW[as.character(i-1), '1']
  }

  # Fill matrix
  for(i in 2:n){
    for(j in (max(2, i-w)):(min(m, i+w))){

      #current cost
      cost<-distance(x[,as.character(j)], y[,as.character(i)])

      #cumulated cost
      d.cost<-min(DTW[as.character(i-1), as.character(j)] ,
                        DTW[as.character(i), as.character(j-1)],
            2*DTW[as.character(i-1), as.character(j-1)])

   #combined cost
   DTW[as.character(i),as.character(j)]<-cost + d.cost

  }
 }

 # 2. Find path
 path<-matrix(c(n,m), 1,2)
 full.path<-(tail(path,1)[1] ==1 & tail(path,1)[2] ==1)

 while(full.path==FALSE ){

  l.path<-tail(path,1)

  if(l.path[1]==1 | l.path[2]==1){
   p<-which(l.path==1)

     if(p==1){new.point<-c(l.path[1], l.path[2]-1)
     }else{
      new.point<-c(l.path[1]-1, l.path[2])
   }

  } else {

  # nearest point
  min.step<-min(DTW[as.character(l.path[1]-1), as.character(l.path[2]-1)],
    DTW[as.character(l.path[1]), as.character(l.path[2]-1)],
    DTW[as.character(l.path[1]-1), as.character(l.path[2])])
  min.step<-which(c(DTW[as.character(l.path[1]-1), as.character(l.path[2]-1)],
          DTW[as.character(l.path[1]), as.character(l.path[2]-1)],
          DTW[as.character(l.path[1]-1), as.character(l.path[2])])==min.step)
  min.step<-min.step[1]

  #path to nearest point
  if(min.step==1){
   new.point<-c(l.path[1]-1, l.path[2]-1)
  } else{
   if(min.step==2){
   new.point<-c(l.path[1], l.path[2]-1)
   } else{
    new.point<-c(l.path[1]-1, l.path[2])
   }
  }
  }
  path<-rbind(path,new.point)
  full.path<-(tail(path,1)[1] ==1 & tail(path,1)[2] ==1)

  }


return(list(path=path, DTW=DTW))
```

```
}
```