Klas Prütz
Institutionen för lingvistik
Uppsala Universitet
28/11 2001

Dynamic Time Warping

Kurs i talteknologi
Lärare: Kjell Elenius

1 Inledning

Vid automatisk taligenkänning kan grundproblemet sägas vara att en inkommande signal av mänskligt tal ska identifieras. Det innebär att systemet ska kunna relatera det på ett meningsfullt sätt till vad det har lagrat i sin databas. Mänskors tal, vilket är det som ska identifieras, är dynamiskt och varierar över tid och mellan personer. Därför behövs en dynamisk process som kan gissa vad personen som talar säger. Lösningen på det är att hitta det i systemets databas som är mest likt det inkommande talet. Många alternativa lösningar måste jämföras och en funktion för att räkna ut vilken lösning som är den bästa behövs. Denna process består av många delar och många olika problem måste lösas. Även om man isolerar delproblemet att jämföra en given mängd tal i systemets databas med den inkommande ljudsignalen så måste en stor mängd alternativ beaktas. Uttryckt på ett vardagligt sätt kan man säga att om man vill jämföra hur lika talarens ord är med ett av databasens ord så måste man tillåta varierande uttal av orden ifråga. Ett strikt system som endast tillåter ett uttal eller en viss mängd av uttal kommer att misslyckas med igenkännandet eftersom variationen mellan mänskor är så enorm. Dynamic Time Warping (DTW) är en metod som tillåter en viss töjning av tidsskalan, som accepterar lite olika takt i uttalet. Detta kan förstås inte täcka in alla variationer som finns men vidgar ändå perspektivet för systemet.

I denna rapport kommer Dynamic Time Warping att beskrivas och jämföras med en annan populär metod nämligen gömda markovmodeller. DTW är en form av dynamisk programmering och andra former kommer att nämnas med ett exempel hämtat från stränglikhetsmätning.

2 Taligenkänning

Med taligenkänning menas den process som relaterar den inkommande ljudsignalen i, till exempel, ett dialogsystem med ett för systemet tolkningsbart meddelande. Det kan innebära att ljudsignalen jämförs med en databas av ord eller fraser och en speciell enhet eller en mängd av enheter i databasen väljs ut som möjliga ekvivalenter till signalen. Tolkningsbar ska i detta sammanhang förstås som ett vitt begrepp, med tolkning kan menas att ljudsignalen av systemet anses motsvara ett ord eller en fras som finns i databasen och kan analyseras vidare av till exempel en syntaktisk parser.

Ett sätt att lösa detta problem är att systemet har tränats upp med ett antal ord eller fraser som finns representerade i dess databas och som den inkommande ljudsignalen kan relateras till. Det finns olika sätt att lösa själva problemet med att jämföra den inkommande signalen med mönstren i databasen. Två viktiga metoder kommer att beröras här. Huvuddelen av denna rapport kommer att beröra den metod som kallas ’Dynamic Time Warping’ (DTW, Blomberg, Elenius, 1997) men även lösningar med hjälp av gömda markovmodeller (HMM, Hidden Markov Model, Brugnara, De Mori, 2001) kommer att nämnas.

Ett praktiskt sätt att göra igenkänningsproblemet lättare att behandla är att göra insignalen diskret genom att dela upp den i ett antal mätpunkter med ett bestämt men mycket kort tidsavstånd mellan varje punkt. På så sätt kan man betrakta den analoga ljudsignalen som en diskret serie av enheter (mätpunkter) där varje enhet består av ett eller flera mätvärden.

En metod för igenkänning är att använda en HMM (Manning, Schütze, 1999) där de dolda tillstånden i modellen motsvarar en representation av till exempel ett eller flera ord. Tillstånden är då en representation av alla de segment som ingår i enheten. Det är vanligt att denna representation bygger på någon form av träningsmaterial där olika varianter av enheterna förekommer. Detta motsvaras i modellen av att det finns olika vägar från initialtillstånden till sluttillstånden. Träningsdata har dessutom försett transitionerna mellan dessa tillstånd med sannolikhetsvärden Den synliga mängden utdata är de iakttagna mätpunkterna hos insignalen. På motsvarande sätt har träningsdata försett modellen med sannolikheter för att olika dolda tillstånd ska generera de iakttagna tillstånden. Problemet blir då att räkna ut sannolikheten för att den inkommande signalen genererats av de underliggande tillstånden.

Anledningen till att HMM:er nämns bredvid DTW är att båda är viktiga metoder för att lösa igenkänningsproblemet. Båda metoderna har uppenbara likheter med varandra även om de inte är identiska. I båda metoderna används, till exempel, Viterbialgoritmen för att begränsa sökrymden eftersom denna rymd tenderar att öka exponentiellt med längden på de enheter som ska jämföras.

3 ’Dynamic Time Warping’

En alternativ metod är att låta både den inkommande signalen och de i databasen lagrade enheterna representeras av serier av mätpunkter där varje mätpunkt, som tidigare, består av ett eller flera värden. Problemet består då av att jämföra den nya signalen med alla lagrade och

hitta den signal som motsvarar insignalen bäst. För att kunna uppnå detta behöver serier av mätpunkter kunna jämföras och en skillnad eller avstånd dem emellan kunna mätas.

Att jämföra två mätpunkter med varandra (från olika enheter) är ett problem i sig som kräver en del uppmärksamhet. Mätpunkter kan bestå av ett eller flera värden så en funktion som sammanfattar skillnaden mellan två punkter behövs. Består en mätpunkt av en grupp frekvensvärden kan förstås skillnaderna dem emellan summeras, eventuellt samman med någon lämplig viktsfunktion. Detta problem faller utanför arbetet i denna rapport så därför kommer alternativa sätt att mäta avståndet mellan två mätpunkter inte att diskuteras vidare.

Idealfallet är förstås att antalet mätpunkter i båda signalerna är lika stort. Då kan de jämföras i serie, den första mätpunkten i insignalen med den första i jämförelseenheten och så vidare. Avståndet eller skillnaden mellan samtliga mätpunkter summeras och ett totalt avstånd uppnås. Om X och Y är lika långa vektorer av mätpunkter som ska jämföras kan det hela uttryckas som nedan:

$$\Sigma|x_i - y_i|$$

där i = 1 till n där n är antalet mätpunkter i X. Detta är ekvivalent med Hamming Distance eller ’City Block Distance’. Andra avståndsfunktioner skulle förstås kunna vara tänkbara såsom till exempel euklidiskt avstånd:

$$\text{roten ur}(\Sigma(x_i^2 - y_i^2))$$

Problemet är förstås att antalet mätpunkter inte är lika stort. Tiden det tar att säga ett ord varierar och längden av de olika segmenten varierar. Därför behövs en metod som tar hänsyn till detta. Utgående från ’City Block Distance’ används i dessa sammanhang en metod som kallas ’Dynamic Time Warping’ (DTW, Kadouz, 2001). Utgångspunkten är summan av skillnaden mellan enskilda mätpunkter i de båda serierna men metoden tillåter förskjutning av deras inbördes lägen och att en mätpunkt i den ena serien motsvarar fler i den andra. Detta ger möjligheten att hitta den bästa serien i databasen som motsvarar indata.

Eftersom dessa förskjutningar inte är kända på förhand måste olika alternativa jämförelser tillåtas. Olika konfigurationer av jämförelser måste jämföras och den bästa jämförelsen, den med minsta totala skillnad måste hittas. Detta kan lösas med dynamisk programmering och kan illustreras av en matris.

	1	2	3	4	5
4					
3					
2					
1					

Där varje rad motsvaras av en mätpunkt i den ena serien och varje kolumn av en mätpunkt i den andra. Varje cell kommer då att motsvara en kombination av en mätpunkt i de ena och en mätpunkt i den andra serien. Alla tänkbara kombinationer är i detta grundfall tillåtna. I varje cell kan avståndet mellan korresponderande punkter noteras. Problemet är då att hitta en väg från matrisens nedre vänstra hörn till dess övre högra hörn med lägsta möjliga värde hos summan av de passerade cellerna. Man kan begränsa operationen så att om en cell ingår i en möjlig väg så kan inte någon cell motsvarande mätpunkter i serierna som föregår den aktuella punkten tillåtas ingå. Det motsvaras av att de enda stegen i matrisen är rakt uppåt, rakt åt höger eller diagonalt uppåt åt höger.

Trots denna begränsning blir ändå antalet möjliga vägar mycket stort. Om tre alternativ tillåts vid varje steg blir antalet vägar i storleksordningen 3^n där n är antalet mätpunkter i den ena serien. Då fler av dessa vägar överlappar varandra är det inte nödvändigt att i en viss given cell undersöka vägar till denna cell som redan är längre än den väg som undersöks. Om man för varje cell endast tillåter den väg dit som är kortast (minsta ackumulerade skillnad) så är man garanterad att hitta den bästa vägen genom matrisen.

Om a_{ij} är den ackumulerade kostnaden för den kortaste vägen till cellen ij och c_{ij} är skillnaden mellan punkterna i och j i de båda serierna och kan kostnaden för varje cell beräknas som följer:

$$a_{ij} = \min(a_{ij-1}, a_{i-1j}, a_{i-1j-1}) + c_{ij}$$

a_{nm} kommer då att vara den ackumulerade kostnaden för att jämföra den ena strängen av längd n med den andra av längd m.

Denna begränsning av sökvägarna är ekvivalent med Viterbialgoritmen som används för att begränsa sökrymden i en markovmodell. Så istället för att ha ett problem av komplexitetsgraden 3^n så har det reducerats till ett av graden n^2 eller snarare nm där n är längden av den ena serien och m längden av den andra.

4 Dynamisk programmering

Begreppet dynamisk programmering dök upp på femtiotalet och myntades av matematikern Richard Bellman (1957). Denna metod har sedan dess fått ett otal applikationer.

Ett problem snarlikt taligenkänningsproblemet där man använder dynamisk programmering är vid jämförelse av strängar, såsom till exempel jämförelse av ord. En heuristiks strategi för att hitta möjliga översättningsekvivalenter i korpusar av källtexter och dess översättningar som parallellställts är att använda sig av stränglikhetsmått för att hitta ord som är översättningar av varandra (Tiedemann, 1999). Det är förstås en heuristik som fungerar på närbesläktade språk som svenska, tyska och engelska.

Vad som behövs då är en likhetsfunktion, ett mått på likhet eller avstånd mellan ord. En framgångsrik metod är att använda sig av redigeringsavstånd, så kallat Levenshteinavstånd (Kruskal, 1983), vilket innebär att avståndet mellan två textsträngar är den totalt sett minsta summan vikter av det antal redigeringssteg som måste tas för att göra om en sträng till en annan.

Ett antal tillåtna operationer stipuleras såsom borttagande (d(x)), insättande (i(x)) och utbyte (s(x,y)) av symboler. En viktfunktion används så att w(f) innebär vikten av operation f som då kan vara någon av de ovan nämnda tillåtna funktionerna. Avståndet mellan två strängar blir då:

$$\min(\Sigma(w(f_i))$$

där F är mängden av operationer som ändrar den ena strängen till den andra. Eftersom det finns alternativa lösningar, det vill säga endast en delmängd av F behövs för att lösa problemet och den delmängd som eftersöks är den som uppnår det rätta resultatet men har den minsta sammanlagda vikten.

Likt DTW så kan detta lösas med dynamiskt programmering. En matris upprättas:

	x_1	x_2	x_3	x_4	x_5
y_4					
y_3					
y_2					
y_1					

som jämför strängen X med strängen Y eller snarare visar kostnaden för att ersätta sträng X med sträng Y. På motsvarande sätt som i DTW gås matrisen igenom och i varje cell lagras den ackumulerade kostnaden för att komma dit enligt formeln nedan:

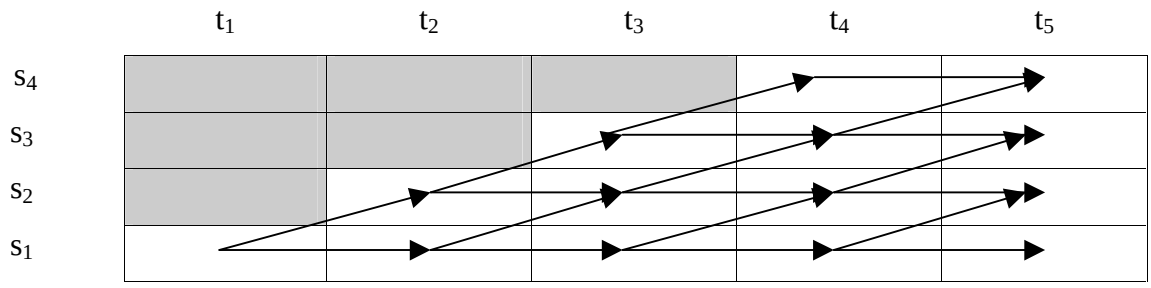
$a_{ij} = \min(a_{ij-1} + w(i(y_j)), a_{i-1j} + w(d(x_i)), a_{i-1j-1} + w(s(x_i,y_j))$

a_{nm} kommer att innehålla den minsta kostnaden för att ersätta sträng X av längden n med sträng Y av längden m.

Denna genomgång av beräkning av Levenshteinavstånd är bara en illustration av ett snarlikt problem som kan lösas med dynamisk programmering. Denna metod är vida spridd och används i många former av applikationer.

5 Jämförelse mellan dynamisk programmering och markovmodeller

Det finns uppenbara likheter mellan markovmodeller (MM) såväl dolda som andra och dynamisk programmering. En tydlig likhet kan illustreras med en trellis som är ett diagram som visar möjliga vägar genom en MM. Om trellisen visas som en matris med riktningen på de tillåtna transitionerna som pilar torde likheten visa sig tydligt.



Varje cell motsvarar ett givet tillstånd (s_j) vid varje given tidpunkt (t_i) och modellen avgör förstås vilka transitioner som är tillåtna. En markovmodell är en stokastisk process därför är ’transitionskostnaden’ mellan tillstånden sannolikheter och kostnaden för en av flera alternativa vägar genom trellisen är produkten av dessa transitionssannolikheter. Dessutom är summan av utgående transitionssannolikheter från varje givet tillstånden i modellen 1.

Det är vanligt i olika applikationer av markovmodeller att den bästa, mest sannolika vägen eftersöks vilket gör det till ett sökproblem där man vill hitta en optimal väg genom trellisen.

Detta problem kan lösas med Viterbialgoritmen vilken kan ses som en form av dynamisk programmering (Brugnara, De Mori, 2001). Istället för ackumulerade summor som i exemplen ovan lagras i cellerna den ackumulerade produkten av sannolikheterna ändes vägen dit och istället för att eftersöka den minsta summan eftersträvas här den största sannolikheten. Dessa skillnader är ytliga och berör egentligen bara de funktioner som används för att beräkna den optimala vägen.

För en äkta dold markovmodell (HMM) där varje tillstånd kan producera en hel mängd utdata skulle förstås en tredimensionell matris krävas men i de flesta fall är mängden utdata det som är givet, alltså är det bara meningsfullt att beräkna sannolikheten för att ett givet dolt tillstånd genererar en viss mängd utdata. Sökproblemet inskränker sig så att säga till modellens dolda del varför en tvådimensionell trellis eller matris är nog.

Likheterna mellan att använda HMM:er och DTW:er vid taligenkänning torde därmed vara tydliga. Den metod som använder DTW jämför två uppsättningar mätpunkter med varandra och beräknar det minsta möjliga avståndet mellan dem. I de metoder som använder HMM ersätts denna jämförelse mellan mätpunkter med sannolikheten för att den lagrade identifierbara representationen av till exempel ett ord ska genererar den aktuella insignalen. I båda fallen behöver man lösa ett kombinatoriskt problem där alternativa möjligheter till jämförelse, alternativt sannolikhetsberäkningar, måste beaktas. Båda fallen är optimeringsproblem där många olika alternativ måste jämföras. I båda fallen kan man begränsa mängden beräkningar med Viterbialgoritmen eller med ett annat ord med dynamisk programmering.

6 Sammanfattning

Denna rapport har försökt beskriva vad Dynamic Time Warping (DTW) är och vad det kan användas till. Utgångspunkten är användandet inom taligenkänning. DTW har jämförts med gömda markovmodeller (HMM) och likheter och skillnader har pekats ut.

Skillnaderna mellan användningen av de båda metoderna inom taligenkänning ligger främst i hur problemet struktureras. HMM innebär att sannolikheter för att en viss representation av till exempel ett ord ska generera en viss mängd utdata beräknas, oftast med hjälp av ett stort träningsmaterial. Utdata i dessa fall är i praktiken den mängd indata som taligenkänningssystemet ska behandla. DTW förutsätter inga sannolikheter utan använder sig av jämförelser mellan olika serier av mätningar, den ena lagrad i en databas (vilket

representerar det som systemet kan känna igen) och den andra är den mängd indata som kommer in till systemet när det används.

Grundproblemet är att många olika alternativa lösningar måste jämföras och i båda fallen används Viterbialgoritmen eller med andra ord dynamisk programmering. Olika funktioner för att beräkna den optimala vägen genom sökrymden används men det ändrar inte den fundamentala likheten mellan dessa metoder eller i alla fall delar av metoderna.

Andra former av dynamisk programmering har nämnts och ett exempel gavs på hur det används för att beräkna Levenshteinavståndet mellan två symbolsträngar.

Bibliografi

Bellman, Richard E. 1957. Dynamic Programming. Princeton University Press: Princeton.

Blomberg, Mats och Kjell Elenius. 1997. Automatisk igenkänning av tal.

Brugnara, Fabio och Renato De Mori. 2001. HMM Methods in Speech Recognition.
URL: http://cslu.cse.ogi.edu/HLTsurvey/ch1node7.html
27/11 2001

Manning, Christopher D. och Hinrich Schütze. 1999. Kap 9 Markov Models. In Foundations of Statistical Natural Language Processing. MIT Press.

Kadous, Mohammed Waleed. 2001. Dynamic Time Warping.
URL: http://www.cse.unsw.edu.au/~waleed/phd/tr9806/node14.html
28/11 2001

Kruskal, Joseph B. 1983. An Overview of Sequence Comparison. In Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, David Sankhoff and Joseph B. Kruskal (red). Addison Wesley Publishing: Massachusetts, London.

Tiedemann, Jörg. 1999. Word Alignment - Step by Step. In Proceedings of the 12th Nordic Conference on Computational Linguistics, University of Trondheim/Norway 1999.