

Sprachverarbeitung: Übung 22

Ziffernerkennung mit CDHMM-Wortmodellen

In den Übungen 19 bis 21 sind 2-dimensionale Merkmalsvektoren, nämlich die beiden ersten Elemente des MFCC-Vektors (*mel frequency cepstral coefficients*), also $\check{c}(1)$ und $\check{c}(2)$ verwendet worden. Um für DDHMM (*discrete density HMM*) geeignet zu sein, mussten die Vektoren zudem quantisiert werden. Trotz der groben Quantisierung (Codebuchgrösse $K = 10$) konnten die fünf künstlich erzeugten Laute [a], [i], [n], [f] und [ʃ] bzw. die daraus gebildeten “Wörter” ziemlich gut unterschieden werden.

Einen Spracherkennung für natürliche Wörter, z.B. für die deutschen Ziffern, kann man jedoch nicht so bauen: Es müssten erstens bessere Merkmalsvektoren¹ verwendet werden, und falls DDHMM zur Anwendung kommen sollen, dann wäre ein grösseres Codebuch erforderlich.

In dieser Übung werden nun aber CDHMM (*continuous density HMM*) eingesetzt, bei denen die Beobachtungen $b_j(\mathbf{x})$ im Zustand S_j als gewichtete Summe multivariater Normalverteilungen dargestellt werden (siehe auch Abschnitt 5.1.2.2 im Buch):

$$b_j(\mathbf{x}) = \sum_{k=1}^M c_{jk} b_{jk}(\mathbf{x}) = \sum_{k=1}^M c_{jk} \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}), \quad (1)$$

wobei die Normalverteilung $\mathcal{N}(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ ist, mit $-\infty < x < \infty$.

Ein lineares CDHMM² mit $N-2$ emittierenden Zuständen, M Mischkomponenten und D Elementen in den Merkmalsvektoren hat also $2N - 4 + (N - 2) * M(1 + D + D^2)$ Parameter. Für $N = 10$, $M = 8$ und $D = 12$ macht dies 10064. Angenommen ein solches CDHMM soll ein Wort beschreiben, das etwa 0.5 sec lang ist und aus dem alle 10 msec ein Merkmalsvektor berechnet wird, dann resultieren 50 Merkmalsvektoren (bzw. Beobachtungen) pro Wort. Mit der Faustregel, dass mindestens 10 Beobachtungen pro Parameter notwendig sind, folgt, dass das Trainingsset eines solchen CDHMMs mindestens $10064/50 \cdot 10 = 2013$ Wörter umfassen muss.

Da dies den Rahmen einer Übung bei weitem sprengen würde, werden hier einfachere CDHMM verwendet: nur eine Mischkomponente ($M = 1$) und zudem wird angenommen, dass die Elemente des Beobachtungsvektors statistisch unabhängig³ sind und somit von der Kovarianzmatrix nur

¹Bessere Merkmalsvektoren müssen insbesondere mehr Information umfassen. Wie anhand von Übung 11, Aufgabe 2 einfach zu verifizieren ist, entsprechen nur die beiden ersten Koeffizienten des Mel-Cepstrums einer viel zu groben Approximation des Mel-Spektrums.

²Ein lineares HMM kann nur im aktuellen Zustand bleiben oder in den nächsten wechseln, jedoch keine Zustände überspringen (vergl. Buch, Abschnitt 5.1.1.2). Es hat deshalb nur $2N - 4$ Zustandsübergangswahrscheinlichkeiten, die nicht fix 0 oder 1 sind.

³Trifft für Cepstren näherungsweise zu.

noch die Diagonalelemente geschätzt werden müssen. Die Zahl der Parameter reduziert sich somit auf $16 + 8 \cdot (12 + 12) = 208$ und damit die Mindestgrösse des Trainingssets auf etwa 54 Wörter.⁴

In dieser Übung arbeiten Sie mit solchen Wort-CDHMM für die Ziffern “Eins” bis “Neun”, “Null” und “Zwo”. Die Sprachsignale sind im Directory `Uebung22/Gegebenes/ueb22_sigs/` abgelegt, jeweils eine Ziffer pro File. Die Files sind in ein Trainings- und ein Testset aufgeteilt, wobei die Files mit `zXX_rY_sZZ_train.wav` bzw. `zXX_rY_sZZ_test.wav` bezeichnet sind. Dabei bedeuten:

XX	Ziffer	01	→	“eins”
		02	→	“zwei”
		:		:
		10	→	“null”
		11	→	“zwo”
Y	Repetitionsnummer	1, 2, ... 5		
ZZ	Sprechernummer	01, 02, ...		

Die Ziffern des Trainingssets sind von 61 Personen (`s16 ... s76`) je einmal gesprochen worden. Die bereits trainierten Wort-CDHMM können Sie mit dem Befehl `load('cdhmm')` laden. Mit diesen CDHMM sind die folgenden Experimente durchzuführen: Zuerst ist der Viterbi-Algorithmus für CDHMM zu realisieren. Dann sind für die trainierten CDHMM die Erkennungsraten pro Wort bzw. pro Sprecher (Sprecherabhängigkeit) zu ermitteln.

Aufgabe 1: Viterbi-Algorithmus für den kontinuierlichen Fall

In der Übung 20 haben Sie den Viterbi-Algorithmus für den diskreten Fall implementiert. Ändern Sie diese Matlab-Funktion (Sie können auch die Musterlösung verwenden) so, dass sie für CDHMM funktioniert, also: `[logP,optQ] = log_cont_viterbi_alg(a,mu,sigma,c,X)`, wobei `logP` die logarithmierte Viterbi-Wahrscheinlichkeit⁵ für die kontinuierliche Beobachtungssequenz `X` ist und `optQ` wiederum die optimale Zustandssequenz bezeichnet. Achten Sie darauf, dass Sie in der Funktion mit dem Logarithmus der Wahrscheinlichkeiten rechnen müssen, um Underflows zu vermeiden (vergl. Buch, Abschnitt 5.7). Die Eingabeargumente der Funktion sind:

a	Zustandsübergangs-Wahrscheinlichkeiten ($N \times N$ -Matrix)
mu	Mittelwertvektor der Beobachtungs-Wahrscheinlichkeitsdichten ($N \times M \times D$ -Matrix)
sigma	Kovarianzmatrix, von der nur die Diagonalelemente als Vektor gegeben sind ($N \times M \times D$ -Matrix)
c	Gewichtungskoeffizienten der Mischkomponenten ($N \times M$ -Matrix)
X	Sequenz von Merkmalsvektoren ($T \times D$ -Matrix)

Um mit dem Programmieren zügig voranzukommen, wird die Beachtung der folgenden Hinweise empfohlen:

⁴Selbstverständlich wird ein Worterkenner aus derart vereinfachten CDHMM nicht gleich gut arbeiten, weil die Modelle die Wörter und ihre Variabilität statistisch zu grob beschreiben. Insbesondere ist ein solcher Spracherkennung noch nicht sprecherunabhängig. Um dies zu erreichen müsste für jedes Wort-CDHMM das entsprechende Wort von mindestens 1000 Personen gesprochen werden und die Personen müssten erst noch statistisch repräsentativ sein.

⁵Weil die Beobachtungswahrscheinlichkeiten bei CDHMM als Wahrscheinlichkeitsdichten gegeben sind, kann der Wert der Produktionswahrscheinlichkeit grösser als 1 werden. Man bezeichnet deshalb `logP` oft als Likelihood, oder, weil hier der Logarithmus verwendet wird, als Log-Likelihood der Beobachtungssequenz und der optimalen Zustandssequenz des CDHMM.

- Da Wahrscheinlichkeiten null sein können (insb. Zustandsübergangs-Wahrscheinlichkeiten), bietet das Rechnen mit logarithmierten Grössen i.a. Schwierigkeiten. Matlab kann jedoch mit dem Logarithmus von null umgehen (`log(0)` ergibt `-Inf`) und auch mit unendlichen Grössen rechnen. Um jedoch die vielen Warnings zu vermeiden, kann die Funktion `logProb` benutzt werden.
- Die Kingsbury-Rayner-Formel (siehe Buch, Gleichung 151) für die Addition zweier Grössen im logarithmischen Bereich müssen Sie nicht implementieren. Sie können die Funktion `logSum` benutzen.
- Zur Berechnung der Likelihoods $b_j(\mathbf{x}_t)$ für alle Zustände und Beobachtungen kann entweder die Formel für die multivariate Normalverteilung (Buch, Gleichung 231) oder die Matlab-Funktion `normpdf` verwendet werden.⁶ Wenn Sie `normpdf` mit nicht-skalaren Eingangsgrössen benutzen, dann müssen Sie dafür sorgen, dass alle Eingangsgrössen dieselben Dimensionen haben. Um beispielsweise die Likelihood für den Zustand S_j und die Beobachtung \mathbf{x}_t für die Mischkomponente k zu bestimmen, werden u.a. `X(t,:)` und `mu(j,k,:)` als Eingangsgrössen gebraucht, die 2- bzw. 3-dimensional sind. Eine mögliche Abhilfe ist, zuerst die Mittelwerte für den Zustand S_j in eine 2-dimensionale Matrix umzuformen mit `mu_j = shiftdim(mu(j,:,:),1)`. Dann hat `mu_j(k,:)` die gleiche Dimension wie `X(t,:)`.

Um die realisierte Funktion zu überprüfen, können Sie das Matlab-Skript `ueb22_1` aufrufen, das Ihnen mitteilen wird, ob Sie die Aufgabe 1 als korrekt gelöst betrachten und zur nächsten Aufgabe gehen dürfen.

Aufgabe 2: Ermitteln der Erkennungsrate

Mit der Matlab-Funktion `ueb22_2` können Sie nun den Viterbi-Algorithmus für einen Ziffernerkennung anwenden. Für die Ziffern “Eins” bis “Neun”, “Null” und “Zwo” ist bereits je ein CDHMM trainiert (braucht viel Zeit) und als Datei `cdhmms.mat` gespeichert worden. Die CDHMM können Sie mit dem Befehl `load('cdhmms')` laden. Sie sind anschliessend im Vektor `cdhmm` verfügbar, wobei jedes Element j die vier folgenden Felder aufweist:

<code>cdhmm(j).name</code>	Ziffer (Matlab character array)
<code>cdhmm(j).a</code>	Zustandsübergangs-Wahrscheinlichkeitsmatrix
<code>cdhmm(j).mu</code>	Mittelwertvektor der Beobachtungs-Wahrscheinlichkeitsdichten
<code>cdhmm(j).sigma</code>	Kovarianzmatrix (nur Diagonalelemente als Vektor)

Wenn Sie die Funktion starten, dann werden alle Signale des Testsets verarbeitet, d.h. es werden aus den Signalen zuerst die Merkmalssequenzen berechnet (falls sie nicht bereits im Directory `ueb22_data/` vorliegen) und anschliessend wird mit dem Viterbi-Algorithmus dasjenige Ziffern-CDHMM mit der höchsten Produktionswahrscheinlichkeit (entlang des optimalen Pfades) ermittelt. Das Resultat dieses Ziffernerkenners schreibt die Funktion in den Vektor `result` und speichert ihn als Datei `results.mat`. Jedes Element dieses Vektors hat die Felder:

<code>result(k).file</code>	Name der Datei mit dem Signal der Ziffer
<code>result(k).speaker</code>	Nummer des Sprechers
<code>result(k).is_word_num</code>	gesprochene Ziffer ⁷
<code>result(k).rec_word_num</code>	erkannte Ziffer ⁷

⁶Achtung: Das Argument `sigma` der Matlab-Funktion `normpdf` ist die Standardabweichung, also die Quadratwurzel der Varianz.

⁷Die Ziffern “Eins” bis “Neun”, “Null” und “Zwo” werden hier numerisch angegeben mit 01, 02 ... 11.

Abhängig davon, wie effizient Ihre Matlab-Funktion für den Viterbi-Algorithmus arbeitet, wird das Erkennen des ganzen Testsets mehr oder weniger Zeit brauchen. Das Testset umfasst 15 Personen (`s01 ... s15`), wobei jede Person die Ziffern 5 Mal gesprochen hat. Es sind somit 825 Testsignale vorhanden. Sie können aber probierhalber auch nur einen Teil davon verarbeiten, indem Sie der Funktion einen entsprechenden Dateinamen übergeben. Mit `ueb22_2('*r2*test.wav')` kann beispielsweise nur die zweite Wiederholung jeder Ziffer ausgewählt werden, was 165 Signale ergibt.

Sie können aber auch die Erkennung starten und während sie ausgeführt wird sich mit dem Lösen der nächsten Aufgabe beschäftigen.

Aufgabe 3: Ermitteln der Verwechslungsmatrix

- a) Schreiben Sie ein Matlab-Skript, das aus den Resultaten des Erkenners in `results.mat` die Verwechslungsmatrix für die Ziffern ermittelt und darstellt. Was stellen Sie fest?
- b) Ermitteln Sie aus den Resultaten des Erkenners auch die Erkennungsraten für jede der 15 Personen des Testsets. Was stellen Sie hier fest?