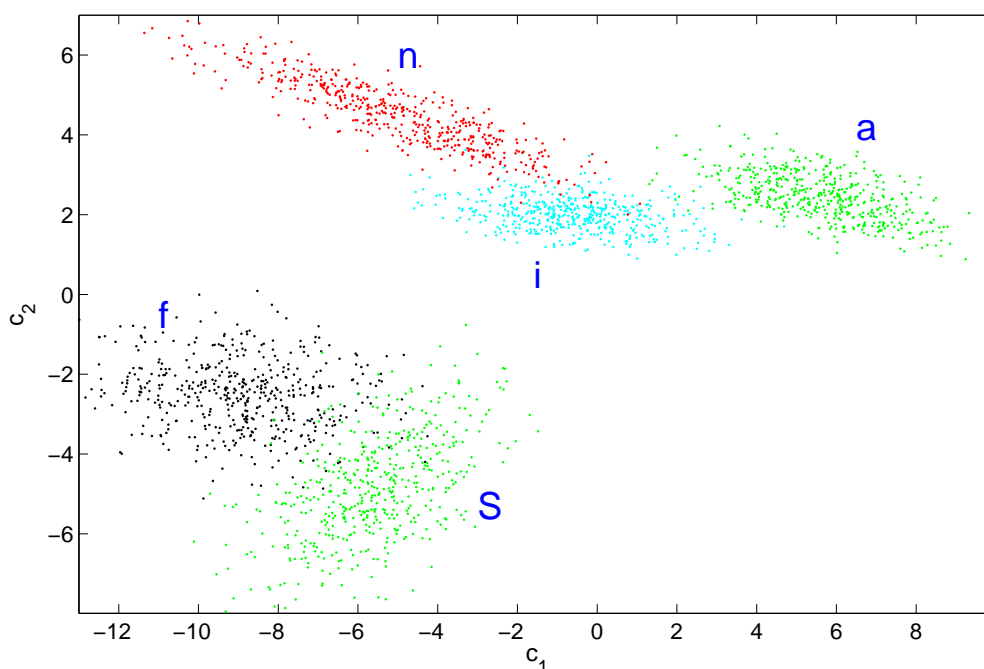


Sprachverarbeitung: Übung 19

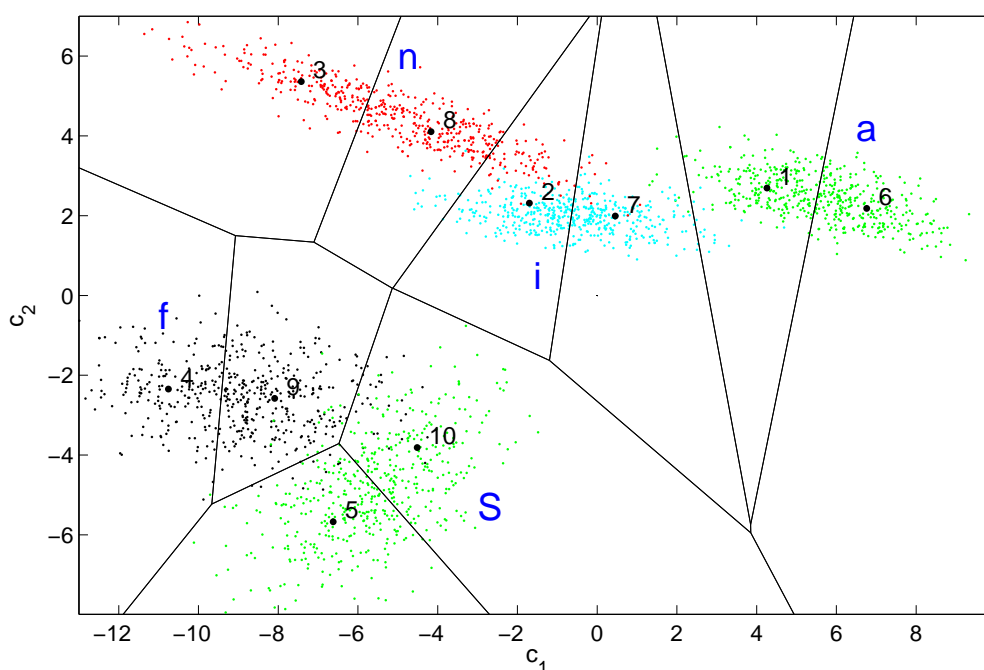
Generieren und Erkennen diskreter Beobachtungssequenzen

In dieser Übung geht es darum, Experimente mit DDHMM (*discrete density hidden Markov models*) zu machen, wobei diese sowohl zum Generieren als auch zum Klassifizieren (bzw. zum Erkennen) von Beobachtungssequenzen eingesetzt werden. Ein DDHMM λ mit den Zuständen $S_i, i = 1 \dots N$, wobei die Zustände S_1 und S_N nicht emittierend sind, wird durch die Zustandsübergangswahrscheinlichkeiten A und die Beobachtungswahrscheinlichkeiten B beschrieben. A ist eine $N \times N$ -Matrix und B wird in den Übungen als Matrix mit $N \times M$ Elementen angegeben, wobei M die Zahl der diskreten Beobachtungen ist. Für die nicht emittierenden Zustände sind alle Beobachtungswahrscheinlichkeiten null.

Wenn HMM zur statistischen Beschreibung von Sprachsignalen eingesetzt werden, dann entsprechen die Beobachtungssequenzen der HMM den aus den Sprachsignalen ermittelten Folgen von Merkmalsvektoren. Als Merkmalsvektoren werden in dieser Übung die MFCC (*Mel frequency cepstral coefficients*) verwendet. Damit die Verteilungen graphisch dargestellt werden können, werden in dieser Übung nur 2-dimensionale Merkmalsvektoren gebraucht, nämlich Vektoren mit den beiden Elementen $\check{c}(1)$ und $\check{c}(2)$. In Figur 1 sind die Laute [a], [i], [n], [f] und [ʃ] je als Punktwolke in der $\check{c}(1)/\check{c}(2)$ -Ebene dargestellt, wobei jeder Punkt einem Wertepaar $(\check{c}(1), \check{c}(2))$, also einem Vektor entspricht. In den Figuren und in den Programmen sind die Laute in EThPA-Notation bezeichnet. So steht beispielsweise für den Laut [ʃ] also [S].



Figur 1: Verteilung der Laute [a], [i], [n], [f] und [ʃ] in der $\check{c}(1)/\check{c}(2)$ -Ebene. Die Verteilungen der Punkte für die Laute sind zwar künstlich erzeugt worden, entsprechen jedoch ungefähr der natürlichen Sprache.



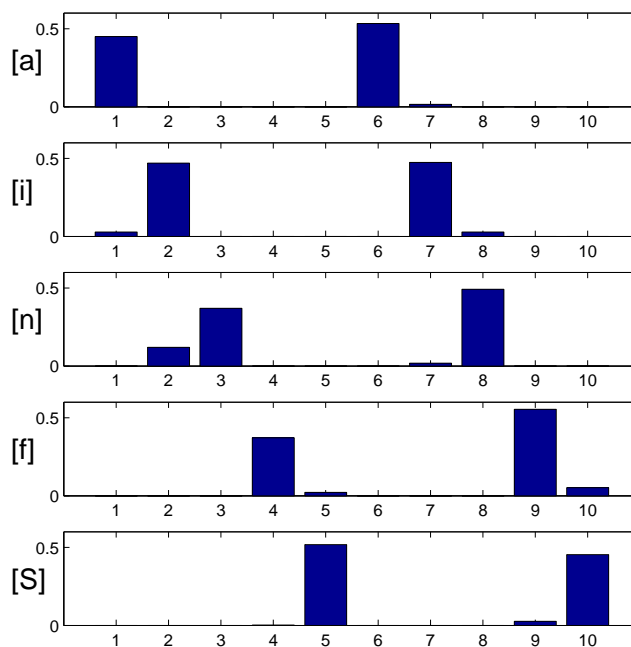
Figur 2: In zehn Partitionen unterteilter Merkmalsraum (also $\check{c}(1)/\check{c}(2)$ -Ebene)

Da in dieser Übung (und auch in den beiden nächsten) mit diskreten Markov-Modellen gearbeitet wird, d.h. die HMM generieren oder erkennen Sequenzen diskreter Beobachtungen, müssen die kontinuierlichen, zweidimensionalen Merkmalsvektoren der Laute vektorquantisiert werden, wobei die Codebuchgröße M auf 10 gesetzt worden ist, wie Figur 2 zeigt.

Die diskrete Wahrscheinlichkeitsdichte für einen Laut kann nun ermittelt werden, indem gezählt wird, wie gross der Anteil der Punkte dieses Lautes ist, welche in die Partition k fallen, wobei hier k von 1 bis 10 geht. In Figur 3 sind die resultierenden diskreten Wahrscheinlichkeitsdichten für die synthetischen Laute dargestellt.

Die Figur 3 zeigt, dass die Wahrscheinlichkeitsdichten der Laute zwar verschieden sind, dass jedoch manche Beobachtungen bei mehreren Lauten vorkommen. So kommt z.B. die Beobachtung 2 bei [i] und bei [n] vor, allerdings nicht mit gleicher Wahrscheinlichkeit. Somit werden die Laute [i] und [n] aufgrund der Beobachtung 2 nicht unterscheidbar sein.

Andererseits ist zu sehen, dass die Laute [a] und [f] nie verwechselt werden können, weil jede für [a] mögliche Beobachtung bei [f] ausgeschlossen ist und umgekehrt.



Figur 3: Diskrete Beobachtungs-Wahrscheinlichkeitsdichten der Merkmale für die Laute [a], [i], [n], [f] und [j]

Mit der obigen Beschreibung der Laute können nun die in dieser Übung gebrauchten DDHMM spezifiziert werden. Es sind HMM, die jeweils ein Wort darstellen, wobei hier der Einfachheit halber für jeden Laut ein Zustand gebraucht wird.¹ Da der Anfangs- und der Endzustand nicht emittierend sind, beschreiben die folgenden HMM also Wörter mit 3 Lauten:

$$\begin{array}{ll} \text{DDHMM(1): [fif]} & \begin{array}{l} \text{Zustand 1: Anfangszustand} \\ \text{Zustand 2: Laut [f]} \\ \text{Zustand 3: Laut [i]} \\ \text{Zustand 4: Laut [f]} \\ \text{Zustand 5: Endzustand} \end{array} \end{array} \quad A_1 = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.3 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.3 & 0.7 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.7 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$\begin{array}{ll} \text{DDHMM(2): [fin]} & \begin{array}{l} \text{Zustand 1: Anfangszustand} \\ \text{Zustand 2: Laut [f]} \\ \text{Zustand 3: Laut [i]} \\ \text{Zustand 4: Laut [n]} \\ \text{Zustand 5: Endzustand} \end{array} \end{array} \quad A_2 = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.8 & 0.2 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.8 & 0.2 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 0.2 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$\begin{array}{ll} \text{DDHMM(3): [faf]} & \begin{array}{l} \text{Zustand 1: Anfangszustand} \\ \text{Zustand 2: Laut [f]} \\ \text{Zustand 3: Laut [a]} \\ \text{Zustand 4: Laut [f]} \\ \text{Zustand 5: Endzustand} \end{array} \end{array} \quad A_3 = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.1 & 0.9 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.9 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.1 & 0.9 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$\begin{array}{ll} \text{DDHMM(4): [fif]} & \begin{array}{l} \text{Zustand 1: Anfangszustand} \\ \text{Zustand 2: Laut [f]} \\ \text{Zustand 3: Laut [i]} \\ \text{Zustand 4: Laut [f]} \\ \text{Zustand 5: Endzustand} \end{array} \end{array} \quad A_4 = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.3 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.3 & 0.7 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.4 & 0.6 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$\begin{array}{ll} \text{DDHMM(5): [faf]} & \begin{array}{l} \text{Zustand 1: Anfangszustand} \\ \text{Zustand 2: Laut [f]} \\ \text{Zustand 3: Laut [a]} \\ \text{Zustand 4: Laut [f]} \\ \text{Zustand 5: Endzustand} \end{array} \end{array} \quad A_5 = \begin{bmatrix} 0.0 & 0.1 & 0.4 & 0.5 & 0.0 \\ 0.0 & 0.6 & 0.2 & 0.1 & 0.1 \\ 0.0 & 0.2 & 0.1 & 0.1 & 0.6 \\ 0.0 & 0.5 & 0.2 & 0.2 & 0.1 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

Diese DDHMM können Sie mit dem Befehl `load('ddhmm')` laden. Sie sind anschliessend im Array `ddhmm` verfügbar.² Jedes Element `j` des Arrays `ddhmm` hat die drei folgenden Felder:

<code>ddhmm(j).name</code>	Folge von Lauten in ETHPA-Notation (Character Vector)
<code>ddhmm(j).a</code>	Zustandsübergangs-Wahrscheinlichkeitsmatrix ($N \times N$ -Array)
<code>ddhmm(j).b</code>	Beobachtungs-Wahrscheinlichkeiten ($N \times M$ -Array)

¹In der Spracherkennung werden in der Regel etwa drei Zustände pro Laut verwendet.

²In diesem Array sind noch weitere DDHMM vorhanden, die jedoch erst in der nächsten Übung gebraucht werden.

Aufgabe 1: Generieren von Beobachtungssequenzen

Starten Sie das Matlab-Skript `ueb19_1.m`. Es erscheint ein neues Fenster mit einer graphischen Benutzerschnittstelle, bei der Sie durch Klicken auf eine der Schaltflächen mit dem betreffenden DDHMM eine Beobachtungssequenz generieren können. Versuchen Sie durch Experimentieren und mit den auf den vorangehenden Seiten gegebenen Spezifikationen der DDHMM die folgenden Fragen zu beantworten:

- a) Generieren Sie mit `ddhmm(1)` und `ddhmm(2)` einige Beobachtungssequenzen und achten Sie auf deren Länge. Woher rührt der Unterschied?
- b) Wieso erzeugt `ddhmm(3)` die Beobachtungen 1 und 6 am häufigsten?
- c) Die Modelle `ddhmm(3)` und `ddhmm(5)` haben je drei Zustände mit den Lauten [S], [a] und [f] und somit die gleichen Beobachtungs-Wahrscheinlichkeitsdichten. Wieso tritt trotzdem bei einer Beobachtungssequenz von `ddhmm(5)` oft zuletzt die 6 auf, bei einer Beobachtungssequenz von `ddhmm(3)` hingegen nie?
- d) Wieso kann `ddhmm(5)` Beobachtungssequenzen erzeugen, die kürzer als drei sind, `ddhmm(3)` hingegen nicht?

Aufgabe 2: Forward-Algorithmus

Schreiben Sie eine Matlab-Funktion `p = discr_forward_alg(a,b,X)`, welche mittels des Forward-Algorithmus die Wahrscheinlichkeit berechnet, dass ein DDHMM die diskrete Beobachtungssequenz `X` generiert. Das DDHMM ist durch die Matrizen `a` und `b`, also die Zustandsübergangs- und die Beobachtungswahrscheinlichkeiten definiert. Die Funktion können Sie mit dem Matlab-Skript `ueb19_2.m` testen.

Zusatz: In der Übung 21 wird die Vorwärtswahrscheinlichkeit $\alpha_t(j)$ gebraucht, die im Forward-Algorithmus berechnet wird. Erweitern Sie deshalb die bereits realisierte Matlab-Funktion auf `[p, alpha] = discr_forward_alg(a,b,X)`, sodass sie mit dem zweiten Ausgabeargument die Vorwärtswahrscheinlichkeit als $T \times N$ -Matrix zurückgibt. Testen Sie diese wiederum mit `ueb19_2.m`.

Aufgabe 3: Klassifizieren von Beobachtungssequenzen

Wenn die in der Aufgabe 2 geschriebene Funktion für den Forward-Algorithmus richtig arbeitet, dann starten Sie das Matlab-Skript `ueb19_3`. Wenn Sie auf eines der generierenden DDHMM klicken, dann wird mit diesem eine Beobachtungssequenz erzeugt und für alle erkennenden DDHMM die Forward-Wahrscheinlichkeit ermittelt und angezeigt. Die Schaltfläche des Modells mit der höchsten Forward-Wahrscheinlichkeit wird eingefärbt. Zudem wird die Erkennungsrate jedes Modells angegeben. Um eine grössere Anzahl von Generierungs- und Erkennungsversuchen zu machen, können Sie nach dem Wählen des Modells auf Start klicken.

Versuchen Sie die folgenden Fragen zu beantworten:

- a) Wie oft werden mit `ddhmm(1)` generierte Beobachtungssequenzen für das Wort [fif] mit den andern Wörtern verwechselt?
- b) Erklären Sie, warum die beobachteten Verwechslungen vorkommen können und warum gewisse Verwechslungen (z.B. Wörter [fif] und [fin]) unmöglich sind.