# Fachhochschule Schmalkalden


## Bachelorarbeit

in der Fachrichtung
Computer Science

Thema:

# Lernen des Kerns der Wissensbasis für ein Expertensystem zur Tumorbehandlung

### mit Hilfe von Hidden Markov Modellen


| | |
|---|---|
| Eingereicht von: | Fabian Witt |
| | Matrikelnummer: 301577 |
| | fabian.witt@t-online.de |
| Eingereicht im: | 6. Semester |
| Abgabetermin: | 15.07.2014 |
| Betreuer: | Prof. Dr. Berndt Stiefel |
| Zweitprüfer: | Prof. Dr. Martin Golz |
| | Fachhochschule Schmalkalden |
| | Blechhammer 9 |
| | D-98574 Schmalkalden |

# Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

| | |
|---|---|
| Ort, Datum | Unterschrift |

# Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken, die mich bei meiner Bachelorarbeit und während meines Studiums unterstützt haben.

Mein Dank gilt insbesonders meinen Eltern Uwe und Angelika Witt für die Unterstützung während meines Studiums. Ein besonderer Dank geht an meinen Betreuer Prof. Dr. Berndt Stiefel für die zahlreichen Anregungen und Tipps und Herrn Prof. Dr. Martin Golz der sich freundlicherweise als Korreferent zur Verfügung gestellt hat.

Weiter bedanke ich mich bei all denen, die diese Arbeit Korrektur gelesen und wertvolle Tipps gegeben haben.

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# 1. Einleitung

## 1.1. Motivation

Die heutige Schulmedizin kennt für die Behandlung von Tumoren nur drei grundlegende Methoden. Entweder die Behandlung mittels einer Chemotherapie, mittels Bestrahlungstherapie oder das chirurgische Entfernen des Tumors. Alle drei Methoden, insbesondere eine Bestrahlungstherapie, stellen eine sehr starke Belastung für den Körper dar. Die dabei verwendeten Mittel beschränken ihre Wirkung nicht nur auf den Tumor selbst, sondern schädigen den gesammten Organismus des Patienten.

In seinem Buch [Bur12] schreibt J. Compton Burnett:

> „Es ist mein Standpunkt, dass ein Tumor das Produkt des Organismus ist, und um ihn wirklich zu heilen, muss die Kraft, die ihn hervorbringt, eliminiert, entfernt werden; wenn man ihn wegschneidet, wird lediglich der Organismus das Produkt los, die Kraft aber, die ihn hervorbringt, bleibt dort, wo sie zuvor war, und der operative Eingriff wirkt oft nur wie das Beschneiden einer Rebe, d.h. die tumorbildende Kraft wird angefacht und das fatale Ende dadurch nähergebracht.“
> [Bur12, S.12]

Gleiches gilt auch für heutige Bestrahlungstherapien. Bei dieser Art der Therapie wird der Tumor selbst und nicht dessen Auslöser behandelt. In der Homöopathie steckt hinter jedem Tumor eine Kraft, die diesen zum Vorschein bringt. Der Tumor selbst ist dabei nur ein Zeichen für die Existenz einer Erkrankung, jedoch nicht die eigentliche Erkrankung. Um den Tumor erfolgreich und langfristig zu bekämpfen, muss die Kraft eliminiert werden, die diesen hervorgebracht hat.

Weitläufig werden homöopathische Mittel zur Behandlung von kleineren Erkrankungen, wie

Erkältung, Husten oder Schnupfen, eingesetzt. Diese Selbstbehandlung ohne einen erfahrenen Homöopathen und nur auf Basis von Büchern mag bei solch kleinen Erkrankungen reichen. Aber bei einer komplexen Erkrankung, wie die eines Tumors, wird das Können und die Erfahrung eines Experten benötigt. Um die Arbeit eines solchen Experten zu erleichtern wäre es hilfreich ein System zu schaffen, das zu einer bestimmten Symptombeschreibung eine Empfehlung in Form von Arzneimitteln gibt.

## 1.2. Zielsetzung und Überblick

> „Um eine Krankheit mit Arzneimitteln zu heilen, müssen die Mittel in einer gewissen Beziehung zu dem Krankheitsprozess selbst stehen, und da spielt es keine Rolle, ob die Symptome den Prozess erkennen lassen oder nicht. Wenn die Symptome den krankhaften Prozess richtig beschreiben, dann genügen die Symptome.“
> [Bur12, S.25 f.]

Diese Aussage beschreibt das Ziel der folgenden Arbeit. Die im Buch [Bur12] beschriebenen Fälle sollen anhand des Tumors, der Symptome und der verabreichten Arzneimittel beschrieben werden. Die so erhaltenen Behandlungsprotokolle werden als Lerndaten für ein Expertensystem genutzt. Auf Basis dieser erfolgreich behandelten Tumorerkrankungen können für zukünftige Behandlungen Empfehlung, im Bezug auf die zu verabreichenden Arzneimittel, gegeben werden.

Dabei liegt die Hauptaufgabe nicht in der Implementierung des Expertensystems. Die im Laufe der Arbeit benötigte Implementierung dienst nur zu Testzwecken und beachtet keine Fehler- oder Ausnahmebehandlungen. Auch performencetechnische Aspekte wurden bei der Implementierung nicht beachtet. Das Hauptaugenmerk liegt bei dem zugrundeliegenden Modell und den genutzten Trainings- und Testdaten.

Neben der Wahl des richtigen Modells ist es wichtig Gemeinsamkeiten der Fälle zu finden. Nur durch eine Verallgemeinerung, die einerseits Parallelen der Fälle aufzeigt und andererseits nicht zu viele Informationen verwirft, kann es möglich sein Gemeinsamkeit der Behandlungen zu finden.

## 1.3. Was ist ein Expertensystem?

In der Literatur findet man viele unterschiedliche Definitionen für den Begriff des Expertensystems. [BKI03] fasst diese zusammen und formuliert folgende Definition:

> „Ein Expertensystem ist ein Computersystem (Hardware und Software), das in einem gegebenen Spezialgebiet menschliche Experten in Bezug auf ihr Wissen und ihre Schlussfolgerungsfähigkeit nachbildet.“ [BKI03, S.11]

Der zentrale Punkt ist dabei der Bezug auf einen menschlichen Experten. Laut [BKI03] zeichnet sich ein solcher Experte durch eine lange Fachausbildung und umfassende praktische Erfahrung in seinem Themengebiet aus, die er auch bei neuen Problemen einsetzen kann, für die keine oder keine eindeutige Lösung vorhanden ist. Neben vielen positiven Eigenschaften werden aber auch die negativen Eigenschaften von menschlichen Expertensystemen betrachtet. Darunter zählen unter anderem:

- Expertenwissen kann verloren gehen bzw. kann oft nicht in benötigtem Umfang weitergegeben werden.

- Die Leistung eines Menschen ist nicht konstant und zu jedem Zeitpunkt abrufbar.

Um die Nachteile eines menschlichen Experten auszugleichen, wurde schon früh versucht Systeme zu schaffen, die die positiven Eigenschaften eines menschlichen Experten mit den Vorteilen von computergestützten Systemen kombinieren. Die wichtigsten Eigenschaften eines computergestützten Expertensystems sind dabei die hohe Verfügbarkeit, die leichte Erweiterbarkeit im Bezug auf neues Wissen und die Möglichkeit Wissen von mehreren Experten zu kombinieren.

In der folgenden Arbeit wird hierfür das Lernen aus Beispielen[1] genutzt, um dem System Expertenwissen anzueignen. Durch eine Verallgemeinerung der Beispiele wird versucht Ähnlichkeiten zu finden und diese für neue, für das System noch unbekannte Beispiele nutzbar zu machen.

[1] vgl. [BKI03, S.101]

## 1.4. Warum HMM

Hidden-Markov-Modelle kommen meist dort zum Einsatz, wo die Länge der Merkmalsvektoren von einem Fall zum anderen variieren. Diese Eigenschaft wird auch für das folgende Projekt benötigt. Die Anzahl der Behandlungsschritte variiert stark von Fall zu Fall und es gibt keine Lösung alle Protokolle auf eine Länge zu bringen, ohne wichtige Informationen zu entfernen.

Der Ursprung der Hidden-Markov-Modelle liegt in der Sprachverarbeitung. In [Rab89] wird dazu anschaulich die Theorie und Praxis des Modells vorgestellt. Ein großer Vorteil für die Sprachverarbeitung ist dabei, dass die Zustände des Modells in diskreten Zeitschritten durchlaufen werden können. Wie in [Wen04, S.137] beschrieben, können somit beliebige zeitliche Dehnungen von einzelnen Lauten modelliert werden. Wird ein Laut lang ausgesprochen, so verbleibt das Modell im aktuellen Zustand. „Verschluckt“ der Sprecher den Laut jedoch, so wird der jeweilige Zustand einfach übersprungen. Auch diese Eigenschaft kann für die Darstellung der Fälle genutzt werden. Die einzelnen Behandlungsschritte laufen immer in einer zeitlichen Reihenfolge ab. Verändert sich der Zustand des Patienten nicht bzw. ergeben sich keine Änderungen der Symptome, so wird der Zustand für eine bestimmte Anzahl an Behandlungsschritten nicht verlassen.

Eine weitere Eigenschaft, die sich die Sprachverarbeitung zu Nutze macht, ist die Tatsache, dass nicht der Zustand selbst, sondern nur eine Ausgabe für den Anwender zu sehen ist. Das in [Wen04, S.137] beschriebene Beispiel mit dem Wort „heben“ zeigt diese Eigenschaft beim letzten Laut. Mit einer Wahrscheinlichkeit von 0,7 wird beim Erreichen des letzten Lautes ein „n“ ausgegeben. Wird das vorherige „e“ aber kurz ausgesprochen, so klingt das Wort „heben“ wie „hebm“. Für diesen Fall wird mit der Wahrscheinlichkeit von 0,3 der Buchstabe „m“ ausgegeben. Dieses Merkmal ist wesentlich für die Modellierung des Expertensystems. Der Zustand beschreibt den Tumor und die Symptome des Patienten. Was aber für den Anwender von Interesse ist, ist das Arzneimittel mit dem in einem bestimmten Zustand die Behandlung fortgesetzt werden sollte. Dabei kann nicht nur ein Arzneimittel in Frage kommen, sondern die Wahrscheinlichkeit ist verteilt über mehrere Elemente des Ausgabealphabets.

Die Symptome und die Veränderung des Tumors werden zu einem Zustand zusammengefasst. Nach der Gabe eines Arzneimittels durch den Homöopathen wechselt das Modell, mit ei-

ner bestimmten Wahrscheinlichkeit, in einen neuen Zustand. Diese Zustandswechsel werden durch die Zustandübergänge repräsentiert. Ist ein Zustand erreicht, liegen neue Symptome und eine neue Beschreibung des Tumors vor. Jetzt wird mit einer gewissen Wahrscheinlichkeit ein Arzneimittel ausgegeben, mit dem der aktuelle Zustand behandelt wird.

Die Gesamtheit der oben beschriebenen Eigenschaften des Hidden-Markov-Modells hat dazu beigetragen, dieses Modell als Basis des Expertensystems zu nutzen.

## 1.5. Prinzipien der Homöopathie

Um die Extrahierung und Vereinfachung der Behandlungsprotokolle nachvollziehen zu können, werden im folgenden Abschnitt einige Prinzipien der Homöopathie erläutert.

Bei der Behandlung einer Krankheit wird jeder Patient als individuelle Persönlichkeit betrachtet. Dabei ist für die Wahl der Arzneimittel nicht die Krankheit selbst ausschlaggebend, sondern viel mehr die Umstände wie sich die Beschwerden äußern und Symptome sich verschlechtern oder verbessern. Eine Krankheit wird dabei immer als globale Erkrankung des kompletten Organismus angesehen und nicht auf einzelne Organe beschränkt.

Um die Arzneimittel herzustellen wird der Grundstoff „verdünnt“. Besser ausgedrückt, er wird dynamisiert. In mehreren Schritten wird die Grundsubstanz mit Wasser oder Alkohol verschüttelt oder mit Milchzucker verrieben. So erhält man zum Beispiel mit einem Verhältnis von 1:10 (1 Teil Urtinktur und 9 Teilen Lösungsmittel), aus der Urtinktur die Potenz[2] D1. Das D steht dabei für das lateinische Wort decem und bedeutet 10.

Bei der Behandlung von miasmatischen Symptomen[3] kommen in der Regel hohe Potenzen zum Einsatz. Hingegen bei der gezielten Behandlung von Organen kommen meist niedrige Potenzen oder die Urtinktur zum Einsatz. Die verwendeten Arzneimittel werden dabei für gewöhnlich einzeln und nicht in Kombination mit anderen Arzneimitteln verordnet.

[2] vgl. [Pot]
[3] Miasma: Lehre, dass alle chronischen Krankheiten ihren Ursprung in einem „Urübel“, einem Miasma haben. Miasmatische Symptome sind alle Symptome, die einem bestimmten Miasma zugeordnet sind.

# 2. Hidden Markov Models (HMM)

## 2.1. Modellbeschreibung und Notationen

### 2.1.1. Hidden Markov Modell

Das Hidden Markov Model (im Folgenden HMM) ist ein zweistufiger stochastischer Prozess, der eine Erweiterung einer einfachen Markov-Kette[4] darstellt. Das System besitzt eine endliche Anzahl n an Zuständen $S = (s_1, \ldots, s_n)$, die in diskreten Zeitschritten t, wobei $1 \leq t \leq T$, durchlaufen werden. Die durchlaufene Zustandsfolge wird als $X = (X_1, \ldots, X_T)$ dargestellt. Dabei steht jedes $X_t$ für einen der n Zustände. Somit kann das HMM als endlicher Automat betrachtet werden. Zwischen den Zuständen existieren Kanten, die mit einer bestimmten Übergangswahrscheinlichkeit belegt sind.

Die erste Stufe des stochastischen Prozesses ist der Zustandsübergang. Die Wahrscheinlichkeit zum Zeitpunkt t in einen bestimmten Zustand zu gehen hängt nur davon ab, in welchem Zustand sich das System zum Teitpunkt t-1 befunden hat. Alle vorherigen Zustände werden also nicht beachtet. Somit ergibt sich ein Modell mit der Markov-Eigenschaft 1. Ordnung:

$$P(X_t = s_j | X_1, X_2, \ldots, X_{t-1} = s_i) = P(X_t = s_j | X_{t-1} = s_i) \tag{2.1}$$

Als Erweiterung von Markov-Modellen 1. Ordnung gibt es noch Modelle m-ter Ordnung[5]. Diese Modelle treffen eine statistische Aussage über den aktuellen Zustand auf der Basis der Kenntnis von m zurückliegenden Zuständen. Im Bezug auf die Wirkung der Arzneimittel

[4] vgl. [Rab89, S.258ff]
[5] vgl. [Wen04, S.138ff]

könnte man auch ein Modell m-ter Ordnung wählen, da die Arzneimittel eine bestimmte Zeit lang wirken. Bei den betrachteten Fällen liegen jedoch Abstände von ein bis zwei Monaten zwischen den Behandlungen, sodass die Arzneimittel die darauffolgenden Arzneimittel nicht beeinflussen dürften. Auf Grund dieser Tatsache und zur Vereinfachung wurde ein Modell 1. Ordnung gewählt.

Die zweite Stufe des stochastischen Prozesses bewirkt zum Zeitpunkt t eine Ausgabe oder Emission $O_t$, die eine Sequenz $O = (o_1, \ldots, o_T)$ bildet. Dabei ist die Ausgabe nur vom aktuellen Zustand $X_t$ abhängig und wie folgt definiert:

$$P(O_t|O_1, \ldots, O_{t-1}, X_1, \ldots, X_t) = P(O_t|X_t) \tag{2.2}$$

Wichtig ist, dass für einen Beobachter lediglich die ausgegebenen Symbole und nicht die Zustände selbst zu sehen sind. Die Zustände sind also „versteckt“ (engl. hidden), woraus sich der Name Hidden Markov Modell ableiten lässt.

### 2.1.2. Modellbeschreibung

Alle Zustände des Modells sind über Kanten miteinander verbunden. Diese Kanten sind wiederum mit Wahrscheinlichkeiten belegt. Jedes Element beschreibt die Wahrscheinlichkeit vom Zustand i in den Zustand j zu gelangen. Zusammengefasst werden die Übergangswahrscheinlichkeiten in der $N \times N$ Matrix **A**:

$$A = \{a_{ij}|a_{ij} = P(X_t = j|X_{t-1} = i)\}, \quad \sum_{j=1}^{N} a_{ij} = 1, \quad 1 \leq i, j \leq N \tag{2.3}$$

Die Summe aller Wahrscheinlichkeiten die von einem Zustand wegführen summieren sich zu eins. Gibt es Zustände zwischen denen keine Kanten existieren, so wird die Wahrscheinlichkeit auf null gesetzt.

Jeder Zustand besitzt eine Anfangswahrscheinlichkeit $\pi_i$. Somit ist jeder Zustand mit einer Anfangswahrscheinlichkeit $\pi_i > 0$ mit der angegebenen Wahrscheinlichkeit ein „Startzustand“. Möchte man einen Zustand explizit als Startzustand kennzeichnen, so erhält dieser die Wahrscheinlichkeit $\pi_i = 1$ unter der Bedingung $\sum_{i=1}^{N} \pi_i = 1$. Alle betrachteten Behandlungen beginnen mit einem ersten Treffen, das als „Start“ in den Behandlungsprotokollen vermerkt ist. Da dieser Zustand immer an erster Stelle vorkommt, erhält er automatisch die Wahrscheinlichkeit 1,0. Zusammengefasst werden die Wahrscheinlichkeiten in einem Vektor mit folgender Form:

$$\pi = \{\pi_i | \pi_i = P(X_1 = i)\}, \quad \sum_{i=1}^{N} \pi_i = 1, \quad 1 \leq i \leq N \tag{2.4}$$

Die Emissionswahrscheinlichkeit, das bedeutet die Wahrscheinlichkeit, dass ein bestimmtes Symbol aus dem vorgegebenen Alphabet emittiert wird, setzt sich wie folgt zusammen:

$$\{b_i(o_k) | b_i(o_k) = P(O_t = o_k | X_t = i)\}, \quad 1 \leq k \leq M \tag{2.5}$$

Wird, wie in unserem Fall, nur ein diskretes Intervall $\{o_1, o_2, \ldots, o_M\}$ an Emissionen verwendet, so spricht man von einem diskreten HMM. Die Emissionen liegen dabei in symbolischer Darstellung vor. Alle $b_i(o_k)$ (Wahrscheinlichkeit im Zustand i die Emission $o_k$ zu erhalten) sind somit diskrete Wahrscheinlichkeiten und können in einer $N \times M$ Matrix **B** zusammengefasst werden:

$$B = \{b_i(o_k) | b_i(o_k) = P(O_t = o_k | X_t = i)\}, \quad \sum_{i=k}^{M} b_i(o_k) = 1, \quad 1 \leq k \leq M \tag{2.6}$$

Somit ergibt sich für das Hidden Markov Modell folgendes Tupel, das mit dem griechischen Kleinbuchstaben $\theta$ (Theta) bezeichnet wird:

$$\theta = \langle \pi, a, b \rangle$$

## 2.2. Parameteroptimierung

### 2.2.1. Lernstrategie

Der nächste Schritt besteht darin die Parameter des HMM (Start-, Übergangs- und Ausgabewahrscheinlichkeiten) so anzupassen, dass das Modell die Trainingsdaten mit größter Wahrscheinlichkeit erzeugen wird. Dabei stehen zwei grundsätzliche Lernstrategien zur Verfügung.

Beim unüberwachten Lernen (engl. unsupervised learning) sind die Lernstichproben nicht etikettiert, d.h. die Klassenzugehörigkeit der Stichprobe ist vorher nicht bekannt. Im Fall des HMM werden die Lerndaten durch Zustände und Klassen durch die Ausgaben des HMM repräsentiert. Da die Zustände für einen Beobachter beim HMM nicht sichtbar sind, ist nicht bekannt welcher Zustand welche Ausgabe erzeugt hat. Um dennoch die Parameter optimal an die Trainingsdaten anpassen zu können, kommt meist der EM-Algorithmus[6] zum Einsatz. Dabei handelt es sich um ein iteratives Verfahren, bei dem in zwei Schritten die Qualität des Ergebnisses, d.h. die Wahrscheinlichkeit, dass die Trainingsdaten vom Modell generiert wurden, verbessert wird.

Bei den verwendeten Trainingsbeispielen handelt es sich um etikettierte Daten. Das bedeutet, das zu jedem Zustand der Sequenz die jeweilige Ausgabe bekannt ist. Daher nutzen wir das überwachte Lernen (engl. supervised learning) zur Anpassung des Modells. In den folgenden Abschnitten wird näher auf das Lernen mit etikettierten Lernstichproben eingegangen.

### 2.2.2. Trainingsdaten

In den Trainingsdaten ist zu jeder Ausgabe x des HMMs der Zustand y bekannt, der diese Ausgabe erzeugt hat. Damit ergibt sich für m Trainingsbeispiele die folgende Zusammenfassung:

[6] Expectation-Maximization-Algorithmus, vgl. [Wen04]

$$L = (x_1, y_1), \ldots, (x_m, y_m), \quad wobei\ (x_l, y_l) = \{o_1, \ldots, o_T, s_1, \ldots, s_T\}$$

Für jede Trainingssequenz (x, y) kann anhand des HMM dessen Wahrscheinlichkeit berechnet werden. Einfach ausgedrückt, werden die Wahrscheinlichkeiten der Zustandsübergänge (Start-, Übergangs- und Ausgabewahrscheinlichkeiten) für jeden Sequenzschritt multipliziert. Bei der Betrachtung einer Sequenz gilt:

$$P(x, y) = \prod_i \pi_i^{f(i,x,y)} \cdot \prod_{i,j} a_{i,j}^{f(i,j,x,y)} \cdot \prod_{i,o} b_i(o)^{f(i,o,x,y)} \tag{2.7}$$

Um die Formel zu vereinfachen, wurden in den Potenzen folgende Ausdrücke verwendet:

**f(i,x,y)** mal ist i der Startzustand,

**f(i,j,x,y)** mal folgt Zustand j auf Zustand i,

**f(i,o,x,y)** mal gibt Zustand i die Emission o aus.

### 2.2.3. Optimierungsverfahren mit Lagrange-Multiplikatoren

Ziel ist es die Parameter des HMM so anzupassen, dass ein Trainingsbeispiel mit größt möglicher Wahrscheinlichkeit von unserem Modell erzeugt wird. Das bedeutet $P(x, y)$ muss für alle Beispiele maximal werden.

Um von einem Trainingsbeispiel auf alle schließen zu können, muss die Unabhängigkeit der Daten gegeben sein. Das würde bedeuten, dass die verabreichten Arzneimittel eines Patienten nicht den Krankheitsverlauf eines anderen Patienten beeinflussen. Diese Tatsache wird in unserem Fall als gegeben vorausgesetzt. Auf Grund dieser Unabhängigkeit kann folgende Regel der Wahrscheinlichkeitsrechnung genutzt werden:

$$P(A \cap B) = P(A)P(B) \tag{2.8}$$

Aus Formel 2.7 und 2.8 erhält man so die Wahrscheinlichkeit für alle Trainingsbeispiele, bezogen auf das verwendete Modell:

$$P(L) = \prod_{l=1}^{m} P(x_l, y_l) = \prod_{l=1}^{m} \left( \prod_i \pi_i^{f(i,x,y)} \cdot \prod_{i,j} a_{i,j}^{f(i,j,x,y)} \cdot \prod_{i,o} b_i(o)^{f(i,o,x,y)} \right) \tag{2.9}$$

Ein Problem, das bei der Berechnung der Wahrscheinlichkeit auftritt, ist der Wert der einzelnen $P(x_l, y_l)$. Durch das Produkt über alle m Wahrscheinlichkeiten wird das Ergebnis meist so klein, dass es durch Rundungsfehler nicht mehr genau dargestellt werden kann. Um dieses Problem zu umgehen, wird die errechnete Wahrscheinlichkeit logarithmiert. Für die logarithmierte Wahrscheinlichkeit schreibt man kurz $L(\theta)$. $\theta$ steht hierbei für die Parameter des HMM, also $\theta = (\pi, a, b)$. Mit der Logarithmusregel $log(AB) = log(A) + log(B)$ gilt für die Formel 2.9:

$$\begin{aligned} L(\theta) &= \sum_{l=1}^{m} log P(x_l, y_l) \\ &= \sum_{l=1}^{m} \sum_i f(i, x_l, y_l)\ log\ \pi_i + \sum_{i,j} f(i, j, x_l, y_l)\ log\ a_{i,j} + \sum_{i,o} f(i, o, x_l, y_l)\ log\ b_i(o) \end{aligned} \tag{2.10}$$

Für die Formel 2.10 sollen nun die Parameterwerte gefunden werden, die die Wahrscheinlichkeit maximal werden lassen. Dafür wird die erste Ableitung gebildet und diese anschließend gleich null gesetzt:

$$\frac{dL(\theta)}{d\,\theta} = 0 \tag{2.11}$$

Um die Berechnung zu erleichtern und die Herleitung anschaulicher zu gestalten, wird die Ableitung von $L(\theta)$ in drei einzelne Ableitungen gesplittet:

$$\nabla L = \forall\, i, j, o : \frac{\partial\, L(\pi)}{\partial\, \pi_i} = 0, \quad \frac{\partial\, L(a)}{\partial\, a_{i,j}} = 0, \quad \frac{\partial\, L(b)}{\partial\, b_i(o)} = 0 \tag{2.12}$$

Laut Definition des HMM unterliegen die zu maximierenden Parameter den folgenden Bedingungen. Nur Ergebnisse, die diese Bedingungen erfüllen, können für das Modell genutzt werden.

$$\sum_k \pi_k = 1 \tag{2.13}$$

$$\sum_k a_{i,k} = 1 \tag{2.14}$$

$$\sum_o b_i(o) = 1 \tag{2.15}$$

Das Schema zur Berechnung der drei Ableitungen aus 2.12 ist immer das Gleiche. Daher wird die Berechnung der Ableitungen beispielhaft an $L(\pi)$ gezeigt und zum Schluss werden alle Ergebnisse aufgelistet. Ausgangspunkt der Berechnung ist die Formel 2.10 bzw. deren erster Teil, der sich auf $\pi$ bezieht.

$$L(\pi) = \sum_{l=1}^{m} \sum_i f(i, x_l, y_l)\; log\; \pi_i \tag{2.16}$$

Um die Bedingung 2.13 umzusetzen, führen wir eine neue Variable ein. $\lambda$ ist der sog. Lagrange-Multiplikator. Das $\lambda$ „bestraft“ dabei alle Ergebnisse, die nicht der Bedingung folgen, und wird zur Formel 2.16 hinzuaddiert. In der Literatur wird oft diskutiert, ob das $\lambda$ addiert oder subtrahiert werden muss. Dies macht jedoch keinen Unterschied, da $\lambda$ nur eine Hilfsvariable ist, um die Optimallösung zu finden.

$$L(\pi, \lambda) = L(\pi) + \lambda \cdot \ (1 - \sum\nolimits_k \pi_k) \tag{2.17}$$

Bevor das Maximum der Formel 2.17 gesucht wird, soll zunächst der Ausdruck 2.16 vereinfacht werden. Zu diesem Zweck wird die Ableitung für $L(\pi)$ betrachtet.

$$\frac{\partial L(\pi)}{\partial \pi_i} = \frac{\partial}{\partial \pi_i} \underbrace{\sum_{l=1}^{m} f(i, x_l, y_l)\ log\ \pi_i}_{enthält\ \pi_i} + \underbrace{\sum_{l=1}^{m} \sum_{j:j \neq\ i} f(j, x_l, y_l)\ log\ \pi_j}_{enthält\ keine\ \pi_i = Konstante} \tag{2.18}$$

Die Ableitung der Formel 2.16 besteht nun aus zwei Teilen. Der erste Teil der Addition summiert alle $\pi_i$, der zweite Teil enthält dementsprechend keine $\pi_i$ mehr. Da es sich um eine Ableitung nach $\pi_i$ handelt, ist der zweite Teil eine Konstante und fällt weg.

Nach der Vereinfachung der Formel 2.16 wird jetzt der Lagrange-Multiplikator wieder eingesetzt und der so erhaltene Ausdruck gleich null gesetzt. Wir suchen nun den Wert von $\pi_i$, sodass gilt: $\frac{\partial L(\pi,\lambda)}{\partial \pi_i} = 0$.

$$\frac{\partial}{\partial\ \pi_i} \sum_{l=1}^{m} \left( f(i, x_l, y_l)\ log\ \pi_i + \lambda(1 - \sum_k \pi_k) \right) = 0$$

$$\frac{\partial}{\partial\ \pi_i} \sum_{l=1}^{m} \left( \underbrace{f(i, x_l, y_l)\ log\ \pi_i}_{\frac{\partial}{\partial \pi_i} = \frac{f(i,x_l,y_l)}{\pi_i}} + \lambda - \underbrace{\lambda\ \pi_i}_{\frac{\partial}{\partial \pi_i} = \lambda} - \lambda \sum_{j:j \neq\ i} \pi_j \right) = 0 \tag{2.19}$$

Die markierten Teile der Ableitung bleiben erhalten. Alle anderen Teile sind wiederum Konstanten und können vernachlässigt werden.

$$\frac{\partial L(\pi)}{\partial \pi_i} = \sum_{l=1}^{m} \frac{f(i, x_l, y_l)}{\pi_i} - \lambda = 0 \tag{2.20}$$

$$\pi_i = \frac{\sum_{l=1}^{m} f(i, x_l, y_l)}{\lambda} \tag{2.21}$$

Die erhaltene Formel für $\pi_i$ beinhaltet noch die Variable $\lambda$. Setzt man jetzt die Bedingung 2.13 in die Formel 2.21 ein und stellt auf $\lambda$ um, so erhält man folgenden Ausdruck:

$$\lambda = \sum_{k} \sum_{l=1}^{m} f(k, x_l, y_l) \tag{2.22}$$

Mit Formel 2.21 und 2.22 können wir nun $\pi_i$ berechnen, sodass dessen Wert maximal wird.

$$\pi_i = \frac{\sum_{l=1}^{m} f(i, x_l, y_l)}{\sum_{k} \sum_{l=1}^{m} f(k, x_l, y_l)} \tag{2.23}$$

Beim Betrachten der Formel 2.23 wird klar, dass es sich hier um eine relative Häufigkeit handelt. Gleiches gilt auch für die anderen zu maximierenden Parameter des Modells. Die Werte für $\pi_i$, $a_{i,j}$ und $b_i(o)$, die $L(\theta)$ maximieren, ergeben sich aus der Herleitung wie folgt:

$$\pi_i = \frac{\sum_{l} f(i, x_l, y_l)}{\sum_{l} \sum_{k} f(k, x_l, y_l)} \tag{2.24}$$

$$a_{i,j} = \frac{\sum_{l} f(i, j, x_l, y_l)}{\sum_{l} \sum_{k} f(i, k, x_l, y_l)} \tag{2.25}$$

$$b_i(o) = \frac{\sum_{l} f(i, o, x_l, y_l)}{\sum_{l} \sum_{o' \in V} f(i, o', x_l, y_l)} \tag{2.26}$$

# 3. HMM als Kern der Wissensbasis für ein Expertensystem

## 3.1. Trainingsdaten

### 3.1.1. Die Datenquelle

Die verwendeten Daten werden aus dem Buch „Die Heilbarkeit von Tumoren durch Arzneimittel“[7] von J. Compton Burnett bezogen und liegen in Textform vor. Dabei handelt es sich um Krankheitsfälle die Ende des 19. Jahrhunderts behandelt wurden. Der Autor beschreibt die Fälle in einer zeitlichen Abfolge, wobei zwischen den Behandlungsschritten Abstände von ca. einem Monat liegen. Zum Ende der jeweiligen Behandlung kommt es öfter vor, dass die Patienten längere Zeit nicht erscheinen, da sie sich nach eigener Aussage sehr gut fühlen.

Um sich ein Bild des Patienten machen zu können, sind einige Angaben zur Person gegeben: Alter, Geschlecht, Familienstand und Anzahl der Kinder. Es folgt eine Beschreibung des Tumors mit Lage, Größe und aufgetretenen Symptomen der Vergangenheit. Teilweise werden auch Todesfälle der Familie oder Verwandten erwähnt, wie zum Beispiel der Tod der Mutter durch Tuberkulose. Darauf folgend sind die jeweiligen Treffen beschrieben, wobei kurz die aufgetretenen Symptome, Verbesserungen oder Verschlechterungen des Tumors und die neuverordneten Arzneimittel aufgelistet werden. Das Buch enthält Fälle mit einer sehr detaillierten Beschreibung, aber auch Fälle die nur sporadische Verordungen enthalten. Für die Trainingsbeispiele wurden gut beschriebene Fälle gewählt. Die Behandlungsdauer bzw. die Anzahl der verordneten Arzneimittel variiert dabei sehr.

Die für die Wissensbasis genutzen Behandlungsprotokolle stehen in Form von CSV-Dateien

[7] [Bur12]

unter folgendem Link zum Download zur Verfügung:
`https://www.dropbox.com/sh/4pc636rbplp4fmy/AAAIPMk5vrpm2v8YpA6U2Z9pa`

### 3.1.2. Aufbereitung der Daten

Um die Fälle als Trainingsbeispiele nutzen zu können, mussten die Daten aufbereitet werden. In einem ersten Schritt wurden die verordneten Arzneimittel in ihrer zeitlichen Reihenfolge erfasst. Es wurden jeweils der Arzneimittelname und die verordneten Potenzen gespeichert. Durch die Zusammenfassung der Arzneimittel mit den verordneten Potenzen treten viele Arzneimittel öfter auf, die Potenzen unterscheiden sich aber oft nur gering. Bei der Verordnung der Arzneimittel werden die Potenzen an den jeweiligen Patienten angepasst, die Wirkung des Mittels bleibt dabei gleich. Durch eine niedrigere Potenz soll eine negative Reaktion des Körpers vermieden werden. Zur späteren Analyse wird der Name und die Potenz seperat gespeichert. Um eine Vergleichbarkeit zu schaffen, wurden die Potenzen in fünf Gruppen aufgeteilt: Die Urtinktur, C und D Potenzen bilden dabei die Obergruppen. Unter den C und D Potenzen wurde wiederum zwischen den Verdünnungsstufen 1 bis 30 und 100 bis 10.000 unterschieden. Mit dieser Unterscheidung wird versucht eine Überanpassung (engl. overfitting) zu vermeiden.

Das Buch enthält Fälle, bei denen zwei oder mehr Arzneimittel gleichzeitig verordnet wurden. Diese Arzneimittel wurden in unserer Betrachtung wiederum zu einem neuen Arzneimittel zusammengefasst, um die kombinierte Wirkung dieser darzustellen.

Als nächstes wurden aus den Fallbeschreibungen der Zustand des Tumors, bezogen auf dessen Größe nach der Arzneimittelgabe ermittelt. Es wurden vier Hauptzustände extrahiert, zwischen denen unterschieden wird:

- „Tumor kleiner“,
- „Tumor gleich“,
- „Tumor größer“,
- „Tumor weg“.

Zum Schluss wurden noch die aufgetretenen Symptome erfasst. Einerseits alle Symptome die der Patient zu Beginn der Behandlung zeigte, so wie die Symptome die nach einer Arzneimittelgabe hinzugekommen oder weggefallen sind.

Um Symptome bei einer Analyse besser vergleichen zu können wurde eine Taxonomie[8] erstellt. Jedes Symptom wurde zunächst einer der vier Oberklassen Psora[9], Sykose[10], Syphilinie[11] oder Schmerzen zugeordnet. Der zweite Teil eines Symptoms ist die Veränderung durch das Arzneimittel. Dabei gibt es zwei Kategorien: stärker oder schwächer. Ist bei einem Treffen nichts über ein vorher erwähntes Symptom beschrieben, wird davon ausgegangen, dass es sich gebessert hat und zum jetzigen Zeitpunkt keinen Einfluss mehr hat. Dieser Schritt dient außerdem zur Vereinfachung der Zustände. Würde man eine dritte Beschreibung mit dem Wert „gleich“ einführen, so ergäbe sich nur eine geringe Ähnlichkeit der Fälle durch eine zu genaue Beschreibung der Zustände. Ein weiterer Grund auf einen dritten Wert zu verzichten ist die Tatsache, dass sich der behandelnde Homöopath auf die Aussagen des Patienten verlassen muss. Eine objektive Einschätzung des Patienten, ob ein Symptom gleich geblieben ist, gibt es meistens nicht. Hat sich ein Symptom verbessert oder verschlechtert, so merkt der Patient das sofort. Symptome die aber gleich geblieben sind, werden oft nicht erwähnt.

Die folgende Abbildung zeigt beispielhaft die erhaltene Taxonomie der vier Oberklassen. Die Zuordnung der einzelnen Symptome zu den Miasmen sind im Anhang A zu finden.

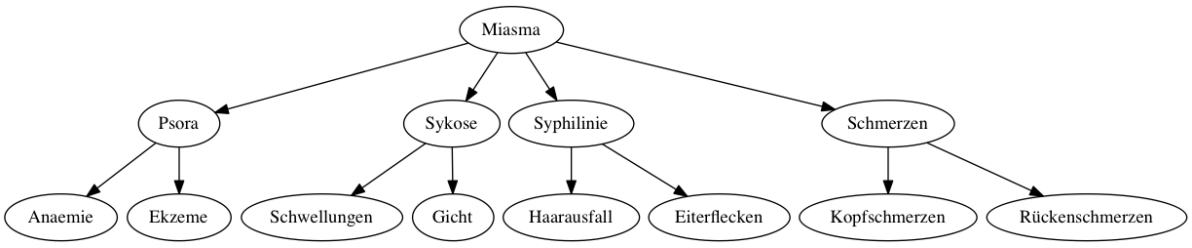


Abbildung 3.1.: Symptomoberklassen mit Ausprägungen

[8] Eine Taxonomie ist in der Regel eine hierarchische Klassifikation mit Klassen, Unterklassen usw.
[9] Erhöhung oder Erniedrigung der Funktion von Zellen und Organen. Dazu gehören z.B. Entzündungen, Allergien und Hautausschläge
[10] Zunahme und Neubildung von Gewebe wie Warzen, Hautverdickungen und Vergrößerungen von Organen
[11] Zerstörung von Gewebe wie Läsionen und Blutungen

## 3.2. Modelltopologie

### 3.2.1. Zustände

Um den Verlauf einer Behandlung zu verallgemeinern, mussten aus den Beschreibungen verschiedene Arten von Zuständen extrahiert werden. Die erhaltenen Zustände werden in der Behandlungsreihenfolge beschrieben.

Die Beschreibung der Fälle beginnt in der Regel mit einem ersten Treffen. Dabei erhält man einige Informationen zum Patienten. Der nächste Schritt besteht in einer genaueren Untersuchung und Befragung des Patienten. Die Art und Größe des Tumors, sowie bisher aufgetretene Symptome werden beschrieben. Auch wird kurz auf die vergangenen Erkrankungen des Patienten, sowie Krankheiten und Todesfälle in der Familie eingegangen.

Bei einigen Fällen wird erwähnt, dass der Auslöser des Tumors ein Trauma[12] oder vorangegangene Impfungen sind. Um die verschiedenen Auslöser des Tumors bei der Auswertung unterscheiden zu können, wurden die Patienten verschiedenen Gruppen zugeordnet. In einem ersten Schritt wird unterschieden, ob der Auslöser des Tumors bekannt ist. Daraus ergeben sich die folgenden drei Zustände „Auslöser: Trauma“, „Auslöser: Impfung“ und „Auslöser: unbekannt“. Der Zustand „Auslöser: Trauma“ fasst dabei physische oder psychische Traumata zusammen. Die Gruppierung der Patienten ist im folgenden Zustandsgrahpen dargestellt.

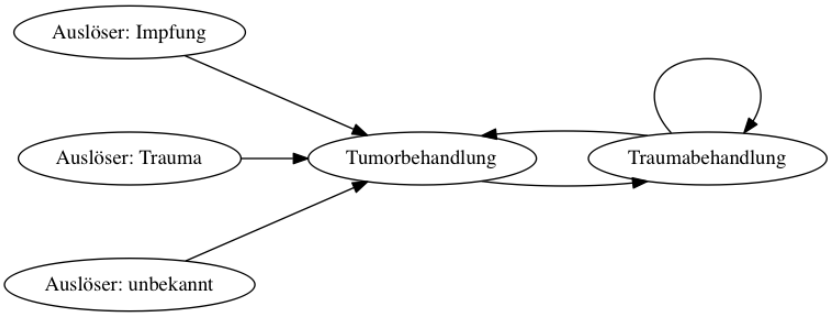


Abbildung 3.2.: Klassifizierung der Patienten

Der im Zustandsgraphen 3.2 gezeigte Zustand mit dem Namen Tumorbehandlung repräsen-

[12]Eine durch physische oder psychische Gewalteinwirkung entstandene Verletzung

tiert die eigentliche Behandlung. Dieser umfasst die Beschreibungen der einzelnen Behandlungsschritte und fasst alle beschreibenden Zustände zusammen. Ein beschreibender Zustand besteht aus der Kombination von Tumorbeschreibung und Liste der derzeitigen Symptome. Dabei bildet die Tumorbeschreibung mit ihren vier Ausprägungen jeweils eine Obergruppe. Diese fasst alle Zustände zusammen, die die gleiche Tumorbeschreibung, aber unterschiedliche Symptomlisten besitzen. Im Folgenden ist die Behandlung des Tumors als Zustandgraph dargestellt.

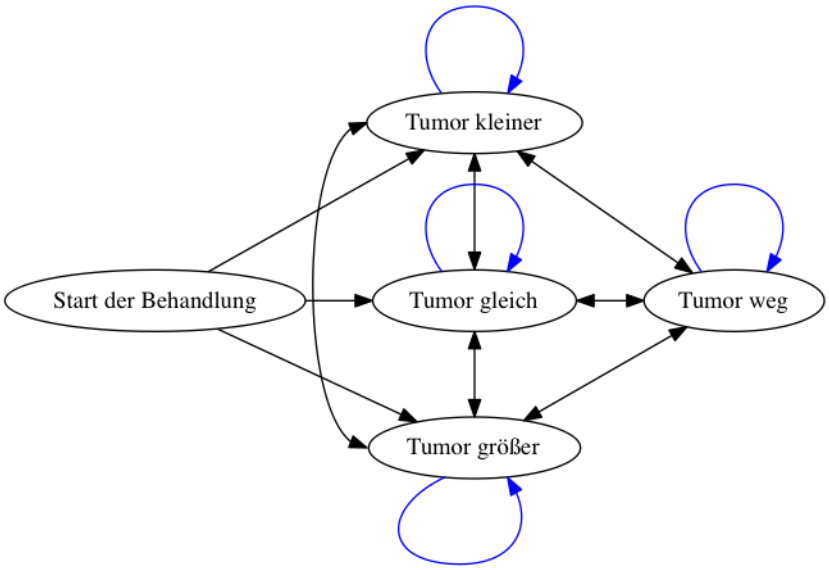


Abbildung 3.3.: Vereinfachter Zustandsgraph der Tumorbehandlung

Aus jedem Zustand des Graphen 3.3 kann in den Zustand der Traumabehandlung gewechelt werden, der im Graphen 3.2 abgebildet ist. Zur Vereinfachung der Abbildungen wurde das Modell auf zwei Graphen verteilt.

### 3.2.2. Sonderzustände

Der in Abbildung 3.2 gezeigte Zustand zur Behandlung eines Traumas ist ein Sonderzustand. Um alle Fälle gleichwertig nutzen zu können, darf immer nur die Therapie des Tumors betrachtet werden. Zu diesem Zweck überpringt dieser Zustand die Phase der Traumabehandlung, um danach ein einheitliches Behandlungsprotokoll zu erhalten. Bei der späteren Anwendung wird nur der Teil der Tumorbehandlung betrachtet.

Wird der in Abbildung 3.3 gezeigte Zustandsgraph als einzelnes Modell betrachtet, kann der erste Zustand ebenfalls als Sonderzustand angesehen werden. Dieser bildet einen sog. Startzustand. Wie in [Fin03, S.133ff] beschrieben, handelt es sich dabei um einen Zustand von dem immer gestartet wird und der nach dem Verlassen nicht wieder erreicht werden kann. Der in Abbildung 3.3 gezeigte Zustandsgraph entspricht dabei dem Modell, das zur späteren Analyse bzw. Betrachtung neuer Fälle genutzt wird.

Einen expliziten Endzustand wie in [Fin03, S.133ff] beschrieben gibt es hingegen nicht. Zwar endet die Behandlung mit der Entlassung des Patienten als gesund, jedoch kann nach einer Heilung der Tumor wieder auftreten. Um einen Endzustand zu erhalten, darf dieser nach Erreichen nicht wieder zu verlassen sein. In der Matrix der Übergangswahrscheinlichkeiten A müssten alle Übergänge vom Endzustand weg mit null belegt sein, was bei uns nicht der Fall ist.

### 3.2.3. Ausgaben

Wird ein neuer Zustand im HMM erreicht, erfolgt eine Emission aus dem Ausgabealphabet. In der Matrix B der Ausgabewahrscheinlichkeiten ist dazu die Verteilung der Wahrscheinlichkeiten auf die einzelnen Arzneimittel je Zustand gespeichert. Das Element mit der höchsten Wahrscheinlichekeit wird ausgegeben und gibt das Arzneimittel an, mit dem die Behandlung fortgesetzt wird.

In manchen Fällen wurde mehreren Arzneimitteln die gleiche Wahrscheinlichkeit zugeordnet. Ist dies der Fall, so werden alle Arzneimittel mit der gleichen Wahrscheinlichkeit ausgegeben. Für die Entscheidung, welches Arzneimittel zum Einsatz kommen soll, wird wiederum ein Experte benötigt.

## 3.3. Implementierung des HMM

### 3.3.1. Daten einlesen

Nachdem die Fallbeschreibungen händisch extrahiert und als CSV-Datei abgespeichert wurden, können diese nun mit Hilfe eines Java-Programms eingelesen werden. Die Trainingsdaten liegen pro Patient in einer einzelnen CSV-Dateien vor. In der folgenden Tabelle ist der Beginn einer Behandlung zu sehen. Es sind drei Behandlungsschritte mit insgesamt zwei Symptomen dargesellt.

| Arzneimittel | Tumor | Symtom 1 | Symptom 2 |
|---|---|---|---|
| thuja_C1_30 | Start | | |
| viscum_album_C1_30 | tumor_gleich | sykose_schwaecher | |
| arsenicum_album_C1_30 | tumor_gleich | sykose_schwaecher | psora_staerker |

Tabelle 3.1.: Struktur der Trainingsdaten in einer CSV-Datei

Die Arzneimittel, die das Alphabet des HMMs bilden, werden zunächst alle in einer Liste gespeichert. Dafür werden alle Patientenprotokolle durchlaufen und die Arzneimittel extrahiert. Als nächstes werden Objekte erstellt, die eine eindeutige, numerisch fortlaufende ID erhalten.

Beim Einlesen der Zustände, bestehend aus Tumorbeschreibung und Symptomliste, wird ähnlich wie bei den Arzneimitteln vorgegangen. Der Unterschied besteht darin, dass zunächst alle Symptome erfasst werden und wiederum in Objekte mit eindeutiger, numerisch fortlaufender ID umgewandelt werden. Im zweiten Schritt wird die Tumorbeschreibung zusammen mit der Liste der Symptomobjekte als neues Zustandsobjekt gespeichert. Dabei erhält auch jeder Zustand wiederum eine ID.

### 3.3.2. Parameterberechnung

Um das Modell zu beschreiben, werden zwei Matrizen und ein Vektor mit Wahrscheinlichkeiten benötigt. Die Variable a[ i ][ j ] entspricht der Matrix A aus der Definition 2.3, Variable

b[ i ][ o ] entspricht der Matrix B aus der Definition 2.6 und pi[ i ] dem Vektor $\pi$ aus der Definition 2.4.

Wie in 3.3.1 beschrieben, erhält jedes Arzneimittel und jeder Zustand eine eindeutige ID. So kann zum Beispiel die Startwahrscheinlichkeit des Zustands „Null“ einfach an der ersten Position des Arrays pi, also pi[ 0 ] gespeichert werden. Das gleiche gilt auch für die Variablen a und b. Der Wert an der Position a[ i ][ j ] gibt somit die Übergangswahrscheinlichkeit an, vom Zustand i in den Zustand j zu wechseln.

Um die Wahrscheinlichkeit zu berechnen, wurden die Zustands- und Arzneimittelsequenzen durch deren IDs ersetzt. Somit ergibt sich automatisch aus der ID auch die Position in den jeweiligen Variablen.

### Startwahrscheinlichkeiten

Um die Startwahrscheinlichkeiten zu berechnen, wurde von jeder Zustandsfolge das jeweils erste Element betrachtet. Der Wert des Elements gibt gleichzeitig die Position in der Variable pi wieder. Somit kann dort der Wert um eins erhöht werden. Zum Schluss werden alle Werte des Arrays noch normalisiert, indem jeder Wert durch die Anzahl der Sequenzen geteilt wird.

Die Methode compute_pi erhält als Parameter eine Liste von Zustandssequenzen, wobei die Zustände durch die jeweilige ID dargestellt sind. Als erstes wird das Array pi_num angelegt, wobei die Länge des Arrays der Anzahl der Zustände des HMMs entspricht. Im zweiten Schritt werden alle Zustandssequenzen durchlaufen und jeweils das erste Element betrachtet. Der an der Position akt_bsp[ 0 ] gespeicherte Wert entspricht der Position des gerade betrachteten Zustands im Array pi_num. Somit wird die Häufigkeit bestimmt, dass ein Zustand auch Startzustand ist. Im dritten und letzten Schritt werden alle Werte durch die Anzahl der Trainingsbeispiele geteilt, um so auf die Wahrscheinlichkeit zu kommen. Die dabei verwendete Methode divide(double n, double d) dividiert die Zahl n durch die Zahl d. Für den Fall das d gleich null ist, wird trotzdem der Wert Null zurück gegeben, um einen Rechenfehler zu umgehen.

```
public void compute_pi(ArrayList<Integer[]> z) {

    int pi_num[] = new int[numStates];
    int pi_denum = z.size();

    Iterator<Integer[]> iter_z = z.iterator();

    while(iter_z.hasNext()) {
        Integer[] akt_bsp = iter_z.next();
        pi_num[ akt_bsp[ 0 ] ] ++;
    }

    for(int i = 0; i < numStates; i++){
        pi[ i ] = divide( pi_num[ i ], pi_denum);
    }
}
```

Listing 3.1: Berechnung der Startwahrscheinlichkeiten in Java

### Übergangswahrscheinlichkeiten

Da es sich bei der Variable a um eine Matrix handelt, werden bei jeder Berechnung immer zwei Elemente der Zustandssequenz betrachtet.

Die Methode compute_a erhält als Parameter die gleiche Liste von Zustandssequenzen, die auch compute_pi übergeben wurde. Zunächst wird das zweidimensionale Array a_num angelegt. Die Größe der ersten und zweiten Dimension entspricht dabei der Anzahl der Zustände des HMMs. In dieser Matrix werden die Häufigkeiten der einzelnen Zustandsübergänge gespeichert. Um später die Häufigkeiten zu normalisieren, werden die Zeilensummen der Matrix a_num in dem Array a_denom gespeichert. Im nächsten Schritt wird, wie bei der Berechnung der Startwahrscheinlichkeiten, jedes Trainingsbeispiel einzeln durchlaufen. In der for-Schleife werden dazu jeweils der aktuelle Zustand akt_z[ j ] und der Folgezustand akt_z[ j+1 ] betrachtet. Diese zwei Werte ergeben die Position für die Matrix a_num, um die Häufigkeit für den

Zustandswechsel von j zu j+1 um eins zu erhöhen. Im darauffolgenden Schritt werden die Zeilensummen durch zwei verschachtelte for-Schleifen berechnet. Zum Schluss werden, ähnlich wie bei den Startwahrscheinlichkeiten, die Häufigkeiten der Matrix a_num normalisiert, um auf die Wahrscheinlichkeiten zu kommen.

```
public void compute_a(ArrayList<Integer[]> z) {

    int a_num[][] = new int[numStates][numStates];
    int a_denom[] = new int[numStates];
    Iterator<Integer[]> iter_z = z.iterator();

    while(iter_z.hasNext()) {

        Integer[] akt_z = iter_z.next();

        for(int j = 0; j < akt_z.length - 1; j++){
            a_num[ akt_z[ j ] ][ akt_z[ j+1 ] ] ++;
        }
    }

    for(int i = 0; i < numStates; i++){
        for(int j = 0; j < numStates; j++){
            a_denom[ i ] += a_num[ i ][ j ];
        }
    }

    for(int i = 0; i < numStates; i++){
        for(int j = 0; j < numStates; j++){
            a[ i ][ j ] = divide( a_num[ i ][ j ], a_denom[ i ]);
        }
    }
}
```

Listing 3.2: Berechnung der Übergangswahrscheinlichkeiten in Java

### Ausgabewahrscheinlichkeiten

Um die Matrix b der Ausgabewahrscheinlichkeiten zu berechnen, benötigt man neben den Zustandssequenzen auch die Folge der Arzneimittel.

Die Methode compute_b erhält als Parameter neben der Liste von Zustandssequenzen z auch die dazugehörige Arzneimittelsequenz o für jedes Trainingsbeispiel. Der Ablauf der Berechnungen ist ähnlich denen der Übergangswahrscheinlichkeiten.

Der Unterschied besteht einerseits in der Größe des zweidimensionalen Arrays b_num und andererseits in deren Berechnung. Die Anzahl der Zeilen wird wie gewohnt durch die Anzahl der Zustände des HMMs vorgegeben. Anders als bei der Berechnung der Übergangswahrscheinlichkeiten steht hier jede Spalte für ein Element des Ausgabealphabets des HMMs, was bei unserem Beispiel den Arzneimitteln entspricht. Aus diesem Grund wird bei der Häufigkeitszählung die Spalte des zweidimensionalen Arrays b_num durch die Arzneimittel-ID in der Variable akt_o[ j ] bestimmt. Die Normalisierung der Häufigkeiten verläuft hier genauso wie bei der vorher gezeigten Methode compute_a.

```
public void compute_b(ArrayList<Integer[]> o,
                      ArrayList<Integer[]> z) {

    int b_num[][] = new int[numStates][sigmaSize];
    int b_denom[] = new int[numStates];

    Iterator<Integer[]> iter_o = o.iterator();
    Iterator<Integer[]> iter_z = z.iterator();

    while(iter_o.hasNext() && iter_z.hasNext()) {

        Integer[] akt_o = iter_o.next();
        Integer[] akt_z = iter_z.next();

        for(int j = 0; j < akt_o.length; j++){
            b_num[ akt_z[ j ] ][ akt_o[ j ] ] ++;
```

```
        }
    }

    for(int i = 0; i < numStates; i++){
        for(int j = 0; j < sigmaSize; j++){
            b_denom[ i ] += b_num[ i ][ j ];
        }
    }

    for(int i = 0; i < numStates; i++){
        for(int j = 0; j < sigmaSize; j++){
            b[ i ][ j ] = divide( b_num[ i ][ j ], b_denom[ i ]);
        }
    }
}
```

Listing 3.3: Berechnung der Ausgabewahrscheinlichkeiten in Java

## 3.4. Test der Wissensbasis

### 3.4.1. Inhalt der Tests

Um die Genauigkeit des Modells beurteilen zu können, soll mit Hilfe von Tests ein Wert ermittelt werden, der eine Aussage über die Qualität des Modells erlaubt. Eine Möglichkeit wäre die Berechnung der Sequenzwahrscheinlichkeit, die in Abschnitt 2.2.2 Formel 2.7 beschrieben wird. Dieser Wert ist aber für eine genauere Betrachtung zu allgemein, da die Modellstruktur nicht beachtet wird. Bei der Auswertung sollen für jedes Sequenzelement drei Aspekte betrachtet werden:

- Besitzt der durch die Testsequenz gegebene Zustandsübergang, laut Modell, die höchste Wahrscheinlichkeit? Falls nicht, wie weit ist der Wert, relativ zur maximalen Wahrscheinlichkeit, vom Maximum entfernt?

- Besitzt das durch die Testsequenz gegebene Arzneimittel, des aktuellen Zustands, die höchste Ausgabewahrscheinlichkeit im Modell? Falls nicht, wie weit ist der Wert, relativ zur maximalen Wahrscheinlichkeit, vom Maximum entfernt?

- Wurde der Zustandsübergang bzw. das Arzneimittel durch das Modell richtig prognostiziert, so wird als Sicherheit der Abstand zwischen der größten und zweitgrößten Wahrscheinlichkeit angegeben.

Der dritte Punkt der Betrachtung ist dabei sehr wichtig. Für den Fall, dass von einem Zustand nur eine Kante wegführt, ist die Wahrscheinlichkeit für diesen Übergang gleich eins. Das bedeutet, der Wert, der sich für die Sicherheit ergibt, ist gleich eins.

### 3.4.2. Testdurchführung

Der Testablauf ist in der Java-Datei „test.java“ im Anhang D zu finden. Die einzelnen Schritte des Tests werden im Folgenden anhand einer Testsequenz gezeigt, deren Ergebnisse in der Tabelle 3.2 zu finden sind. Die dargestellte Sequenz besitzt 14 Elemente, bestehend aus einem Startzustand, 13 Zustandsübergängen und 14 Ausgaben. Zur besseren Übersicht wurden die Werte, falls nötig, auf drei Stellen nach dem Komma gerundet.

Die erste Spalte vergleicht die Vorgabe der Sequenz mit der Prognose des Modells, bezüglich der Übergangswahrscheinlichkeiten. Ist der Übergang von Zustand i nach Zustand j der Testsequenz im Modell mit der maximalen Wahrscheinlichkeit belegt, so wird der Wert 1,0 gespeichert. Ist das nicht der Fall, wird die Differenz zwischen maximaler Wahrscheinlichkeit laut Modell, und der Wahrscheinlichkeit bezüglich der Sequenz berechnet. Mit der Division durch die maximale Wahrscheinlichkeit erhält man einen Wert, der die Annäherung an das Maximum angibt.

Gleiches gilt bei der Berechnung der Werte für die Spalte „Ausgabe“. Der Unterschied liegt darin, dass nicht die Übergangswahrscheinlichkeiten, sondern die Ausgabewahrscheinlichkeiten betrachtet werden. Ist das ausgegebene Arzneimittel des aktuellen Zustands identisch mit der Ausgabe des Modells, erhält man den Wert 1,0. Liegt aber eine Abweichung vor, wird wie bereits beschrieben die Annäherung an das Maximum angegeben.

Die zweite Spalte „Sicherheit des Übergangs“ gibt die Differenz zwischen größter und zweitgrößter Wahrscheinlichkeit des Modells an. Dieser Wert ist interessant, wenn das Modell den Zustandsübergang richtig prognostiziert hat. In Tabelle 3.2 und 3.4 sind die Sicherheiten nur angegeben, wenn eine Wahrscheinlichkeit des Übergangs oder der Ausgabe von 1,0 vorliegt. In allen anderen Fällen liegt eine Verteilung der Wahrscheinlichkeiten vor und der Wert wird nicht benötigt. Durch die geringe Anzahl an Trainingsbeispielen kommt es vor, dass von manchen Zuständen nur eine Kante wegführt. In diesem Fall liegt keine direkte Prognose vor, sondern das Modell folgt einem Trainingsbeispiel. Das Auftreten von solchen Zustandsübergängen muss in der späteren Betrachtung der Ergebnisse einzeln bewertet werden. Auch bei der „Sicherheit der Ausgabe“ werden alle Prognosen des Modells mit einer 1,0 markiert, bei denen nur ein Arzneimittel als Ausgabe zur Verfügung steht.

| **Zustandsübergang** | **Sicherheit des Übergangs** | **Ausgabe** | **Sicherheit der Ausgabe** |
|---|---|---|---|
| 1.000 | 1.000 | 0.800 | - |
| 0.750 | - | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 0.500 |
| 1.000 | 0.500 | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 0.143 |
| 0.667 | - | 1.000 | 0.250 |
| 1.000 | 0.250 | 1.000 | 0.200 |
| 1.000 | 0.200 | 0.500 | - |
| 0.643 | - | 1.000 | 0.083 |
| 1.000 | 0.500 | 0.500 | - |
| 1.000 | 0.167 | 1.000 | 0.042 |
| 0.571 | - | 0.500 | - |
| 0.500 | - | 0.929 | - |
| 1.000 | 0.333 | 1.000 | 0.765 |
| 0.866 | 0.214 | 0.873 | 0.143 |

Tabelle 3.2.: Testdurchlauf mit einer Sequenz

In der letzen Zeile der Tabelle werden die Ergebnisse der Testsequenz zusammengefasst. Die zwei Werte für Zustandsübergänge und Ausgaben geben die Annäherung des Modells an das gegebene Testbeispiel an. Der Wert der Sicherheit gibt an, in wie viel Prozent der Fälle nur

ein Weg bzw. eine Ausgabe möglich war.

### 3.4.3. Test mit allen Sequenzen

Im ersten Teil der Tests werden alle Behandlungsprotokolle für die Wissensbasis des Expertensystems genutzt. Die erhaltenen Ergebnisse dienen als Referenz für die nachfolgenden Auswertungen mit der Gruppierung in Auslöserklassen. Die folgende Tabelle zeigt alle Ergebnisse der 15 Sequenzen.

| Nr. | Übergänge | Zustandsübergang | Sicherheit Überg. | Ausgabe | Sicherheit Ausg. |
|---|---|---|---|---|---|
| 01 | 24 | 0.871 | 0.208 | 0.864 | 0.167 |
| 02 | 28 | 0.919 | 0.107 | 0.804 | 0.071 |
| 03 | 25 | 0.829 | 0.040 | 0.832 | 0.000 |
| 04 | 3 | 0.750 | 0.333 | 1.000 | 0.333 |
| 05 | 14 | 0.866 | 0.214 | 0.873 | 0.143 |
| 06 | 7 | 0.964 | 0.714 | 1.000 | 0.571 |
| 07 | 11 | 0.886 | 0.273 | 0.794 | 0.182 |
| 08 | 33 | 0.829 | 0.030 | 0.879 | 0.000 |
| 09 | 14 | 0.918 | 0.357 | 0.926 | 0.286 |
| 10 | 12 | 0.899 | 0.083 | 0.858 | 0.000 |
| 11 | 11 | 0.762 | 0.091 | 0.770 | 0.000 |
| 12 | 6 | 0.923 | 0.167 | 0.856 | 0.000 |
| 13 | 7 | 0.918 | 0.143 | 0.833 | 0.000 |
| 14 | 10 | 0.889 | 0.100 | 0.813 | 0.000 |
| 15 | 4 | 0.583 | 0.250 | 0.700 | 0.000 |
| | 209 | 0.865 | 0.158 | 0.850 | 0.091 |

Tabelle 3.3.: Testergebnisse aller Sequenzen

Der Zustandsgraph für den Test mit allen Sequenen ist im Anhang B und unter folgendem Link zu finden:
`https://www.dropbox.com/s/tfwab1ufl9mkd9t/graphAlleSequenzen.png`

### Zustandsübergang

Das Modell prognostiziert die Zustandsübergänge mit einer Abweichung von durchschnittlich 13,5 Prozent. Die Ergebnisse der einzelnen Sequenzen liegen dabei nah beieinander. Eine Ausnahme bildet die Sequenz mit der Nummer 15. Dieses Behandlungsprotokoll enthält nur drei Zustandsübergänge, wobei jeder Zustand nur die Tumorveränderung ohne Angabe von Symptomen beschreibt. Der Fall zeigt, dass die Beschreibung der Zustände sehr wichtig ist. Zustände, die nur die Tumorveränderung beschreiben, sind für eine genaue Darstellung der Behandlungen zu allgemein und führen zu Fehlern.

### Sicherheit des Übergangs

Der Wert der Spalte „Sicherheit des Übergangs“ scheint zunächst mit 15,8 Prozent relativ gut. Dieser Wert zeigt aber, dass es dennoch viele Zustände gibt, bei denen keine Wahrscheinlichkeitsverteilung vorhanden ist. Durch mehr Trainingsbeispiele ist es möglich, diesen Wert zu verbessern. Eins der Hauptziele für eine Optimierung des Modells muss sein, dass es keine Übergänge mit der Wahrscheinlichkeit 1,0 mehr gibt. Nur so kann gewährleistet werden, dass das Modell eine allgemeine Tumorbehandlung repräsentiert und keinem einzelnen Beispiel mehr folgt.

### Ausgabe

Die Ergebnisse in der Spalte „Ausgabe“ sind ähnlich wie die der Spalte „Zustandsübergang“. Die Abweichung zu den einzelnen Sequenzen liegt meistens zwischen 10 und 20 Prozent. Auch hier bildet das Beispiel mit der Nummer 15 eine Ausnahme. Die bei der Behandlung verwendeten Arzneimittel gehören nicht zu den am häufigsten verwendeten Mitteln und besitzen daher meist nicht die maximale Ausgabewahrscheinlichkeit. Auch hier kann mit einer größeren Anzahl an Trainingsbeispielen eine Verbesserung erreicht werden.

### Sicherheit der Ausgabe

Zum Schluss wird noch die „Sicherheit der Ausgabe“ betrachtet. Dabei fällt auf, dass bei vielen Sequenzen ein Wert von 0,0 auftaucht. Das zeigt, dass die gewählte Verallgemeinerung der Arzneimittel mit den verschiedenen Potenzgruppen eine gute Lösung darstellt. Wie bei allen Werten gibt es auch hier Ausnahmen. Im Fall Nummer 6 werden Arzneimittelkombinationen verordnet, die in dieser Form in keinen anderen Beispielen vorkommen. Betrachtet man alle Werte dieser Sequenz fällt auf, dass alle Ausgaben zu 100 Prozent richtig prognostiziert wurden. Aber bei über 50 Prozent der Fälle lag eine Wahrscheinlichkeit mit dem Wert 1,0 vor. Dieses Beispiel soll im Folgenden näher betrachtet werden.

| Zustandsübergang | Sicherheit des Übergangs | Ausgabe | Sicherheit der Ausgabe |
|---|---|---|---|
| 1.000 | 1.000 | 1.000 | 0.267 |
| 0.750 | - | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 0.500 |
| 1.000 | 0.500 | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 1.000 |
| 1.000 | 1.000 | 1.000 | 0.765 |
| 0.964 | 0.714 | 1.000 | 0.571 |

Tabelle 3.4.: Testergebnisse der 6. Sequenz

Die einzelnen Schritte der Behandlung unterscheiden sich so sehr von den anderen Behandlungen, dass sich eine lineare Zustandsfolge im Modell gebildet hat. An diesem Beispiel ist gut zu erkennen, warum die Angabe der Sicherheit beim Testen mit Trainingsbeispielen so wichtig ist. Lässt man die Werte der Sicherheiten außer Acht, könnte man denken, dass das Modell die Behandlung fast perfekt beschreibt. Mit der Angabe wird aber schnell klar, dass die Behandlung nicht direkt ins Modell aufgenommen wurde, sondern ein extra Modell bildet.

### 3.4.4. Test mit gruppierten Sequenzen

Durch die Gruppierung der Sequenzen nach Tumorauslösern wird versucht, das Modell besser an die jeweilige Behandlungsform anzupassen. Wichtig bei den einzelnen Gruppen ist, dass durch die kleine Anzahl an Beispielen eine Überanpassung stattfindet.

In der folgenden Analyse der Ergebnisse sind die Zustandsgraphen der einzelnen Modelle abgebildet. Die Übergänge sind in drei Kategorien unterteilt:

**Grün:** 0,0 < Wahrscheinlichkeit <= 0,5

**Blau:** 0,5 < Wahrscheinlichkeit < 1,0

**Rot:** Wahrscheinlichkeit = 1,0

Durch diese Darstellung sind lineare Wege im Modell leichter zu finden. Sind nur wenige rote Kanten vorhanden, enthalten die Sequenzen viele Gemeinsamkeiten.

#### Alle Patienten mit physischem Trauma

| Nr. | Übergänge | Zustandsübergang | Sicherheit Überg. | Ausgabe | Sicherheit Ausg. |
|---|---|---|---|---|---|
| 07 | 11 | 0.856 | 0.364 | 0.894 | 0.182 |
| 09 | 14 | 0.929 | 0.571 | 1.000 | 0.429 |
| 10 | 12 | 0.868 | 0.417 | 0.958 | 0.250 |
| 11 | 11 | 0.841 | 0.091 | 0.864 | 0.000 |
| 12 | 6 | 0.917 | 0.500 | 1.000 | 0.333 |
| 14 | 10 | 0.900 | 0.300 | 0.950 | 0.200 |
| | 64 | 0.884 | 0.375 | 0.943 | 0.234 |

Tabelle 3.5.: Testergebnisse Patienten mit physischem Trauma

Bei der Betrachtung der Tabelle 3.5 fällt auf, dass die Ergebnisse insgesamt schlechter ausfallen als in der Tabelle 3.3. Das Ergebnis für den Zustandsübergang hat sich nur minimal

verbessert und die Sicherheiten sind in beiden Fällen gestiegen, also schlechter geworden. Die Genauigkeit der verordneten Arzneimittel ist jedoch viel besser geworden. Die Eigenschaft die diese Fälle verbindet, ist die verwendete Medikation. Auch wenn sich die Zustandsfolgen der einzelnen Behandlungen unterscheiden, so muss bei allen Fällen das Trauma behandelt werden bevor die eigentliche Behandlung des Tumors beginnt. Die Verbesserung ist aber nicht allein auf die Gruppierung der Sequenzen zurückzuführen. Betrachtet man den Zustandsgraph in Abbildung 3.4, sind einige rote Kanten zu erkennen. Für das Auftreten dieser Übergangswahrscheinlichkeiten mit dem Wert 1,0 gibt es neben der Überanpassung durch die geringe Anzahl an Sequenzen noch einen weiteren Grund. In manchen Fällen findet die Behandlung des Traumas erst nach einigen Behandlungschritten statt. Das hat zur Folge, dass sich Sequenzen teilweise oder komplett vom Modell abheben. Dieses Verhalten ist im Zustandsgraphen deutlich zu sehen. Rechts am Graphen hat sich beispielsweise ein Zyklus gebildet. Dieser repräsentiert den Umweg über die Traumabehandlung und führt anschließend wieder in den normalen Behandlungsablauf zurück.

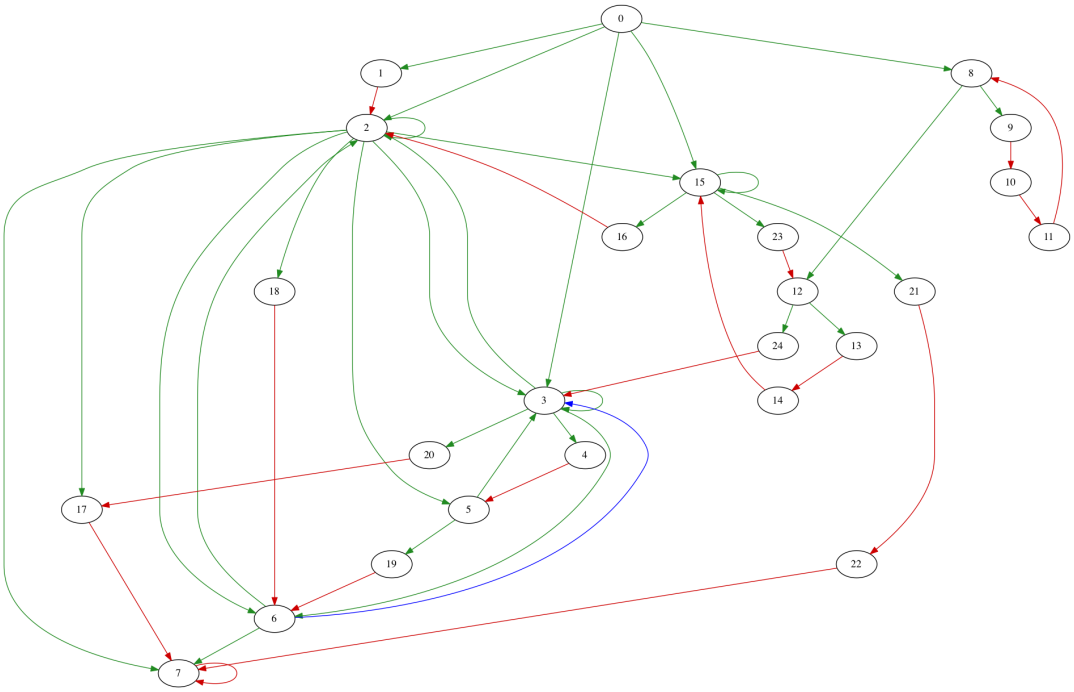


Abbildung 3.4.: Zustandsgraph für Patienten mit physischem Trauma

### Alle Patienten mit psychischem Trauma

| Nr. | Übergänge | Zustandsübergang | Sicherheit Überg. | Ausgabe | Sicherheit Ausg. |
|---|---|---|---|---|---|
| 03 | 25 | 0.793 | 0.120 | 0.867 | 0.120 |
| 08 | 33 | 0.793 | 0.152 | 0.894 | 0.091 |
| 13 | 7 | 0.738 | 0.429 | 0.857 | 0.429 |
| | 65 | 0.787 | 0.169 | 0.879 | 0.138 |

Tabelle 3.6.: Testergebnisse Patienten mit psychischem Trauma

Die Ergebnisse der Tabelle 3.6 zeigen zunächst ein allgemein schlechteres Ergebnis als die Tabelle 3.3. Auch hier spielt die Überanpassung eine große Rolle. Ein Blick auf den Zustandsgraphen in Abbildung 3.5 zeigt aber nur wenige rote Zustandsübergänge. Daraus lässt sich schließen, dass die Sequenzen viele Gemeinsamkeiten besitzen und das schlechte Ergebnis hauptsächlich durch die geringe Anzahl an Trainingsbeispielen erzeugt wird.

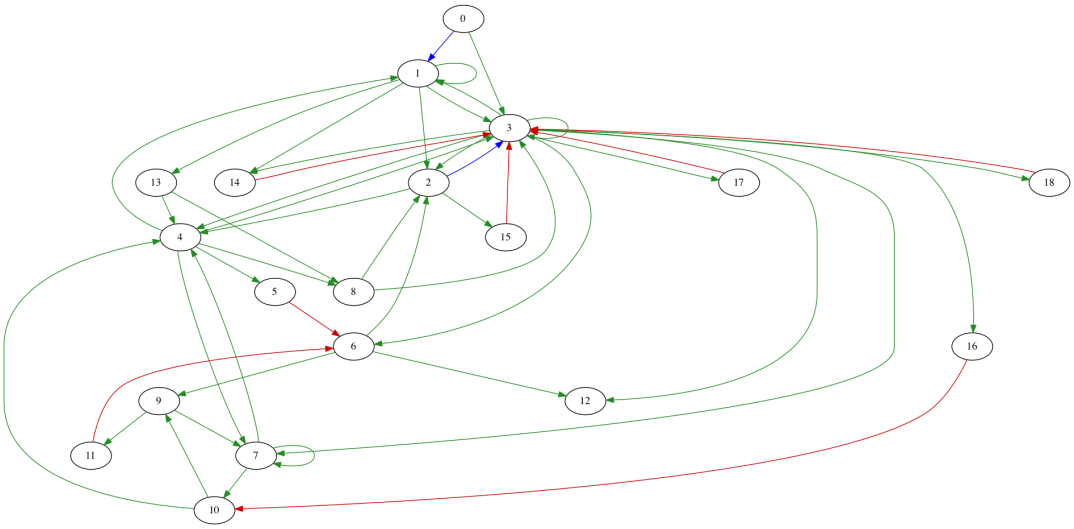


Abbildung 3.5.: Zustandsgraph für Patienten mit psychischem Trauma

### Alle Patienten mit Impfungen

| Nr. | Übergänge | Zustandsübergang | Sicherheit Überg. | Ausgabe | Sicherheit Ausg. |
|---|---|---|---|---|---|
| 02 | 28 | 0.935 | 0.143 | 0.893 | 0.107 |
| 04 | 3 | 0.667 | 0.333 | 1.000 | 0.333 |
| 05 | 14 | 0.893 | 0.429 | 0.904 | 0.286 |
| 06 | 7 | 0.929 | 0.857 | 1.000 | 0.714 |
| 08 | 33 | 0.902 | 0.030 | 0.955 | 0.000 |
| 10 | 12 | 0.938 | 0.250 | 0.956 | 0.167 |
| 13 | 7 | 0.929 | 0.143 | 0.905 | 0.000 |
| 14 | 10 | 0.925 | 0.200 | 0.913 | 0.100 |
| | 114 | 0.912 | 0.211 | 0.931 | 0.140 |

Tabelle 3.7.: Testergebnisse Patienten mit Impfungen

Auch die Gruppe der Patienten mit vorangegangenen Impfungen zeigt ein gutes Ergebnis. Mit Ausnahme der Behandlung Nummer 6 treten bei den Sicherheiten nur geringe Abweichungen auf. Bei dem Fall handelt es sich um die Sequenz, die in Tabelle 3.4 ausführlich dargestellt ist. Wie bereits beschrieben handelt es sich dabei um eine Behandlung, die sich stark von den anderen abhebt. Der Zustandsgraph des Modells ist in Abbildung 3.6 zu sehen. Am rechten Rand verläuft eine Zustandsfolge, die größtenteils rote Kanten besitzt. Bis auf einige Abweichungen zeigt der Zustandsgraph aber eine gute Verallgemeinerung der Sequenzen und bestätigt, dass die Behandlungen der Fälle dem selben Muster folgen.

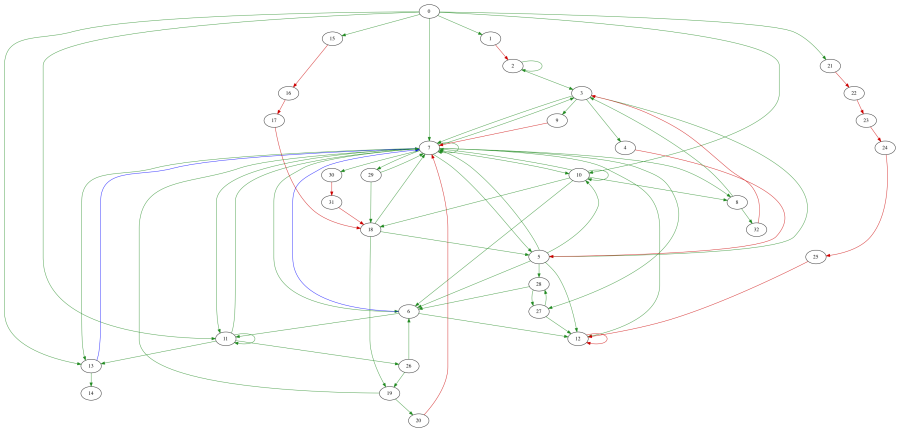

Abbildung 3.6.: Zustandsgraph für Patienten mit Impfungen

### Alle Patienten ohne bekannten Auslöser

| Nr. | Übergänge | Zustandsübergang | Sicherheit Überg. | Ausgabe | Sicherheit Ausg. |
|---|---|---|---|---|---|
| 01 | 24 | 0.847 | 0.375 | 0.854 | 0.292 |
| 15 | 4 | 0.917 | 0.250 | 0.750 | 0.000 |
| | 28 | 0.857 | 0.357 | 0.839 | 0.250 |

Tabelle 3.8.: Testergebnisse Patienten ohne bekannten Auslöser

Die letzte Gruppe beinhaltet alle Fälle bei denen kein Auslöser der Tumore bekannt ist. Der Zustandsgraph in Abbildung 3.7 zeigt im Prinzip nur die erste Sequenz. Die zweite Sequenz mit der Nummer 15 enthält nur Zustände ohne Symptombeschreibung und ist mit nur vier Übergängen sehr kurz. Die in der zweiten Sequenz vorkommenden Zustände sind auch in der ersten Sequenz zu finden, wodurch sich das gute Ergebnis erklären lässt. Auffällig ist, dass sich auch hier ein Zyklus auf der linken Seite des Zustandsgraphen gebildet hat. Das lässt darauf schließen, dass in der ersten Sequenz eine Traumabehandlung stattgefunden hat. Mit einer genaueren Analyse wäre es vielleicht möglich, diese Sequenz einer anderen Auslösergruppe zuzuordnen. Um eine eigenständige Wissensbasis zu erhalten, werden auch hier deutlich mehr Trainingsbeispiele benötigt.

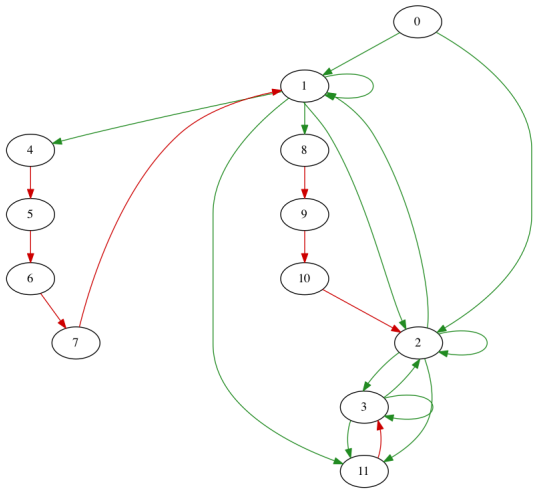


Abbildung 3.7.: Zustandsgraph für Patienten ohne bekannten Auslöser

# 4. Zusammenfassung

Die zentrale Aufgabenstellung der Arbeit, Lernen des Kerns der Wissensbasis für ein Expertensystem zur Tumorbehandlung, wurde gelöst. Es wurde gezeigt, dass mit Hilfe eines Hidden-Markov-Modells die Behandlungen in einer Wissensbasis zusammengefasst werden können.

Die vorangegangenen Tests zeigen, dass die Verwendung einer Wissensbasis auf der Grundlage von allgemeinen Tumorbehandlungen gute Ergebnisse liefert. Um die Genauigkeit zu verbessern, wurden die Fälle in Gruppen eingeteilt. Jede Gruppe steht dabei für einen Auslöser des Tumors. Lässt man die auftretende Überanpassung außer Acht, so wird deutlich, dass die Fälle der einzelnen Gruppen viele Gemeinsamkeiten aufweisen.

Um diese Gemeinsamkeiten sichtbar zu machen, mussten die im Buch [Bur12] beschriebenen Fälle analysiert und bewertet werden. Nur durch eine längere Einarbeitung in das Gebiet der Homöopathie war es möglich die Behandlungprotokolle korrekt zu extrahieren und in nutzbare Trainingsbeispiele umzuwandeln.

Zum Zweck der Verallgemeinerung der Fälle wurden die Symptome verschiedenen Oberklassen zugewiesen. Diese sind in der Homöopathie als Miasmen bekannt und fassen verschiedene Symptome zusammen. Durch diese Verallgemeinerung verkleinert sich die Symptomliste jedes Zustands und es bilden sich mehr Gemeinsamkeiten. Gleiches gilt für die verwendeten Arzneimittel. Neben dem eigentlichen Mittel wurden Gruppen der verordneten Potenzen gebildet, um einerseits die Information der Potenz nicht zu verlieren und andererseits kleinere Unterschiede bei der Verordnung zu verallgemeinern.

Die Herausforderung war, die Beschreibung so zu gestalten, dass nicht zu viele Details verloren gehen und ein gewisser Grad an Verallgemeinerung erreicht wird. Sind die Zustände oder Ausgaben zu detailliert beschrieben, ergeben sich keine Gemeinsamkeiten in den Trainingsdaten. Es entstehen lineare Zustandsfolgen im Modell, die die Trainingssequenzen zwar

abbilden, aber für neue Behandlungen nicht nutzbar sind. Ist die Beschreibung der Fälle jedoch zu allgemein, kann es passieren, dass die Zustandsfolgen der Sequenzen sich zu stark gleichen. Ist das der Fall, wird durch ein neues Trainingsbeispiel nichts neues gelernt, sondern die Wahrscheinlichkeit einer bestehenden Zustandsfolge bzw. Ausgabe nur vergrößert. Die gewählte Beschreibung aus Tumorveränderung und Symptomen beinhaltet genug Informationen, um den Fall genau zu beschreiben und gleichzeitig Ähnlichkeiten zu anderen Fällen zu finden.

Die im Rahmen der Arbeit entstandene Implementierung verfügt über alle Basisfunktionen, die zum Trainieren des Expertensystems notwendig sind. Der im Anhang zu findende Code wurde mittels einiger Kommentare ausführlich dokumentiert, um ihn für anschließende Projekte leichter nutzbar zu machen. Alle Grafiken, Behandlungsprotokolle und die verwendete Implementierung stehen unter folgendem Link zur Verfügung.

`https://www.dropbox.com/sh/sqm5y9yn84dwp1i/`
`AAALlGb629d1UO3A8zlG6ZSda`

# 5. Ausblick

Als Ziel weiterführender Arbeiten ist die Extrahierung und Gruppierung neuer Trainingsdaten zu nennen. Durch eine detaillierte Analyse der Fallbeschreibungen könnten Gruppenzugehörigkeiten gefunden werden, die bisher nicht sichtbar waren. Das Beispiel der Gruppen von Patienten mit vorangegangener Impfung und derer mit einem physischem Trauma zeigt, dass durch die Ähnlichkeit von Fällen bessere Ergebnisse erzeilt werden können.

Ein weiterer Schritt ist die Analyse der Auslöser in den Behandlungen. Für die Wissensbasis ist nur die reine Tumorbehandlung interessant. Wäre es möglich die Behandlung des Auslösers von der Behandlung des Tumors, bezogen auf die Auslösergruppe, zu identifizieren, würde die Wissensbasis an Genauigkeit hinzugewinnen. Dabei liegt die Schwierigkeit aber im Detail. Die Behandlung des Auslösers findet nicht immer zu Beginn der Sequenzen statt. Eine weitere Schwierigkeit liegt bei den verwendeten Arzneimitteln. Oftmals werden Arzneimittel, die für die Auslöserbehandlung genutzt werden, auch in der Tumorbehandlung eingesetzt. Diese Tatsache erschwert die Extrahierung der Auslöserbehandlung zusätzlich.

Ist es möglich mit Hilfe dieser Ansätze die Wissensbasis zu erweitern und zu verbessern, kann ein detailliertes Expertensystem zur Behandlung von Tumorpatienten entstehen.

# A. Zuordnung Symtom und Miasma

| Symptom | Miasma |
|---|---|
| Akne | sykose |
| Anaemie | psora |
| Ausfluss | sykose |
| Auswurf mit Blut | syphilinie |
| Auswurf ohne Blut | psora |
| Diarrhoe | psora |
| durstig | psora |
| Dyspepsie | psora |
| Eiterfleck | syphilinie |
| Ekzeme | psora |
| Entzündug | psora |
| Erbrechen | psora |
| Erkaeltung | psora |
| Flecken | psora |
| Haarausfall | syphilinie |
| Haemorrhoiden | sykose |
| Hände Taub | syphilinie |
| Heuschnupfen | psora |
| Hitzewallungen | psora |
| Husten | psora |
| Mensis | syphilinie |
| Neuralgie | psora |
| Pleurodynie | psora |
| Reizung der Haut | psora |
| Rheumatismus | psora |
| Schwellungen | sykose |
| Verstopfung | psora |
| Gicht | sykose |

Tabelle A.1.: Zuordnung zwischen Symtomen und Miasmen

Die Liste steht außerdem als CSV-Datei zum Download zur Verfügung:
`https://www.dropbox.com/s/b8wlabfh6dnuo0e/symptomliste.csv`

# B. Zustandsgraph aller Sequenzen

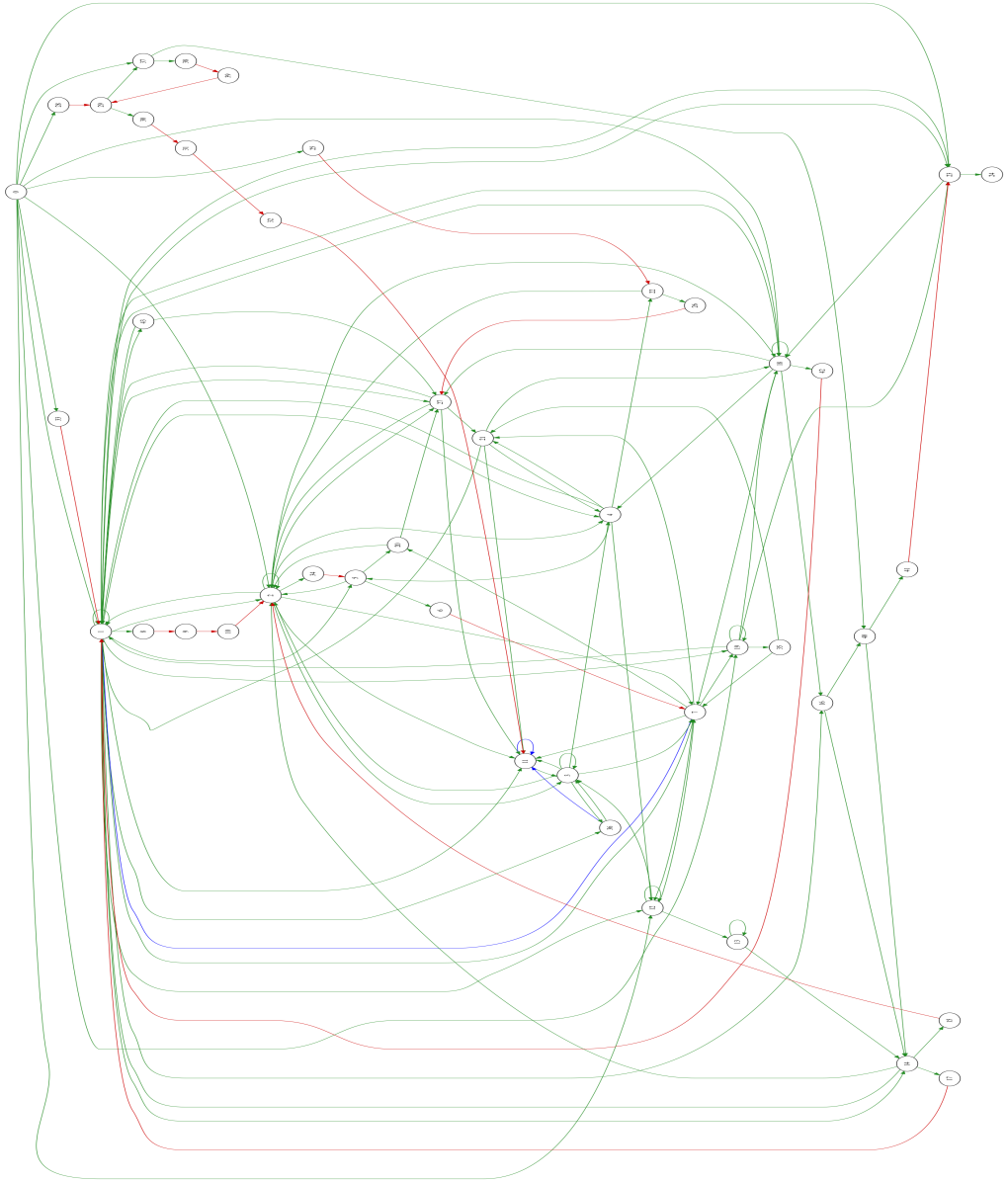

Abbildung B.1.: Zustandsgraph für alle Trainingssequenzen

# C. Hidden Markov Modell: HMM.java

```
import java.util.ArrayList;
import java.util.Iterator;

public class HMM {

    /** Anzahl der Zustaende */
    private final int numStates;
    /** Groesse des Ausgabealphabets */
    private final int sigmaSize;

    /** Startwahrscheinlichkeiten */
    public double pi[];
    /** Uebergangswahrscheinlichkeiten */
    public double a[][];
    /** Ausgabewahrscheinlichkeiten */
    public double b[][];

    /** Initialisiert das HMM */
    public HMM(int numStates, int sigmaSize) {
        this.numStates = numStates;
        this.sigmaSize = sigmaSize;

        pi = new double[numStates];
        a = new double[numStates][numStates];
        b = new double[numStates][sigmaSize];
    }

    public void compute_a( ArrayList<Integer[]> z ) {
        /** 2-Dim-Array fuer Haeufigkeitszaehlung */
```

```
        int a_num[][] = new int[numStates][numStates];
        /** Array fuer Zeilensummen zur Normalisierung */
        int a_denom[] = new int[numStates];

        /** Durchlaufen der Trainingssequenzen */
        Iterator<Integer[]> iter_z = z.iterator();
        while( iter_z.hasNext() ) {
            Integer[] akt_z = iter_z.next();
            for(int j = 0; j < akt_z.length - 1; j++){
                /** Haeufigkeit + 1, fuer Zustandswechsel akt_z[ j ] in akt_z[ j+1 ] */
                a_num[ akt_z[ j ] ][ akt_z[ j+1 ] ] ++;
            }
        }

        /** Berechnen der Zeilensummen der Variable a_num */
        for( int i = 0; i < numStates; i++ ){
            for( int j = 0; j < numStates; j++ ){
                a_denom[ i ] += a_num[ i ][ j ];
            }
        }

        /** Normalisieren der Werte */
        for( int i = 0; i < numStates; i++ ){
            for( int j = 0; j < numStates; j++ ){
                a[ i ][ j ] = divide( a_num[ i ][ j ], a_denom[ i ]);
            }
        }
    }

    public void compute_b(ArrayList<Integer[]> o, ArrayList<Integer[]> z) {
        /** 2-Dim-Array fuer Haeufigkeitszaehlung */
        int b_num[][] = new int[numStates][sigmaSize];
        /** Array fuer Zeilensummen zur Normalisierung */
        int b_denom[] = new int[numStates];
```

```
        /** Durchlaufen der Trainingssequenzen */
        Iterator<Integer[]> iter_o = o.iterator();
        Iterator<Integer[]> iter_z = z.iterator();
        while( iter_o.hasNext() && iter_z.hasNext() ) {
            Integer[] akt_o = iter_o.next();
            Integer[] akt_z = iter_z.next();
            for( int j = 0; j < akt_o.length; j++ ){
                /** Haeufigkeit + 1, fuer Ausgabe akt_o[ j ] in Zustand akt_z[ j ] */
                b_num[ akt_z[ j ] ][ akt_o[ j ] ] ++;
            }
        }

        /** Berechnen der Zeilensummen der Variable b_num */
        for( int i = 0; i < numStates; i++ ){
            for(int j = 0; j < sigmaSize; j++){
                b_denom[ i ] += b_num[ i ][ j ];
            }
        }

        /** Normalisieren der Werte */
        for( int i = 0; i < numStates; i++ ){
            for( int j = 0; j < sigmaSize; j++ ){
                b[ i ][ j ] = divide( b_num[ i ][ j ], b_denom[ i ]);
            }
        }
    }

    public void compute_pi(ArrayList<Integer[]> z) {
        /** Array fuer Haeufigkeitszaehlung */
        int pi_num[] = new int[numStates];
        /** Anzahl der Startzustaende bzw Trainingssequenz zur Normalisierung */
        int pi_denum = z.size();

        /** Durchlaufen der Trainingssequenzen */
        Iterator<Integer[]> iter_z = z.iterator();
```

```
        while( iter_z.hasNext() ) {
            Integer[] akt_bsp = iter_z.next();
            /** Haeufigkeit + 1, fuer jeden ersten Zustand der Sequenz */
            pi_num[ akt_bsp[ 0 ] ] ++;
        }

        /** Normalisieren der Werte */
        for( int i = 0; i < numStates; i++ ){
            pi[ i ] = divide( pi_num[ i ], pi_denum);
        }
    }

    /** Dividiert n durch d => 0 / 0 = 0! */
    public double divide(double n, double d) {
        if ( d == 0 )
          return 0;
        else
          return n / d;
    }

    public double[][] testModellSequenzAbweichung(Integer[] o, Integer[] z){

        double[][] whsMatrix = new double[z.length][ 6 ];

        /** Startwahrscheinlichkeit nach Sequenz */
        whsMatrix[ 0 ][ 0 ] = pi[ z[ 0 ] ];
        /** Ausgabewahrscheinlichkeit nach Sequenz */
        whsMatrix[ 0 ][ 1 ] = b[ z[ 0 ] ][ o[ 0 ] ];

        /** Startwahrscheinlichkeit nach Modell */
        whsMatrix[ 0 ][ 2 ] = getMaxPi();
        /** Ausgabewahrscheinlichkeit nach Modell */
        whsMatrix[ 0 ][ 3 ] = getMaxB( z[ 0 ] );

        /** zweitgroesste Startwahrscheinlichkeit nach Modell */
```

```
        whsMatrix[ 0 ][ 4 ] = getSecondMaxPi();
        /** zweitgroesste Ausgabewahrscheinlichkeit nach Modell */
        whsMatrix[ 0 ][ 5 ] = getSecondMaxB( z[ 0 ] );


        for(int e = 1; e < z.length; e++){
            /** Uebergangswahrscheinlichkeit nach Sequenz */
            whsMatrix[ e ][ 0 ] = a[ z[ e-1 ] ][ z[ e ] ];
            /** Ausgabewahrscheinlichkeit nach Sequenz */
            whsMatrix[ e ][ 1 ] = b[ z[ e ] ][ o[ e ] ];

            /** Uebergangswahrscheinlichkeit nach Modell */
            whsMatrix[ e ][ 2 ] = getMaxA( z[ e-1 ] );
            /** Ausgabewahrscheinlichkeit nach Modell */
            whsMatrix[ e ][ 3 ] = getMaxB( z[ e ] );

            /** zweitgroesste Uebergangswahrscheinlichkeit nach Modell */
            whsMatrix[ e ][ 4 ] = getSecondMaxA( z[ e-1 ] );
            /** zweitgroesste Ausgabewahrscheinlichkeit nach Modell */
            whsMatrix[ e ][ 5 ] = getSecondMaxB( z[ e ] );
        }
        return whsMatrix;
    }

    /** Ermittelt maximale StartWkt */
    private double getMaxPi(){
        double maxPi = -1;
        for(int e = 0; e < pi.length; e++){
            if(pi[ e ] > maxPi)
                maxPi = pi[ e ];
        }
        return maxPi;
    }

    /** Ermittelt zweitgroesste StartWkt */
```

```
private double getSecondMaxPi(){
    double maxPi = −1;
    for(int e = 0; e < pi.length; e++){
        if( pi[ e ] > maxPi && pi[ e ] < getMaxPi() )
            maxPi = pi[ e ];
    }
    return maxPi;
}

/** Ermittelt maximale UebergangsWkt im Zustand z */
private double getMaxA(int z){
    double maxA = −1;
    for(int e = 0; e < a[ z ].length; e++){
        if(a[ z ][ e ] > maxA)
            maxA = a[ z ][ e ];
    }
    return maxA;
}

/** Ermittelt zweitgroesste UebergangsWkt im Zustand z */
private double getSecondMaxA(int z){
    double maxA = −1;
    for(int e = 0; e < a[ z ].length; e++){
        if( a[ z ][ e ] > maxA && a[ z ][ e ] < getMaxA( z ) )
            maxA = a[ z ][ e ];
    }
    return maxA;
}

/** Ermittelt maximale AusgabeWkt im Zustand z */
private double getMaxB(int z){
    double maxB = −1;
    for(int e = 0; e < b[ z ].length; e++){
        if(b[ z ][ e ] > maxB)
            maxB = b[ z ][ e ];
```

```
        }
        return maxB;
    }

    /** Ermittelt zweitgroesste AusgabeWkt im Zustand z */
    private double getSecondMaxB(int z){
        double maxB = -1;
        for(int e = 0; e < b[ z ].length; e++){
            if( b[ z ][ e ] > maxB && b[ z ][ e ] < getMaxB( z ) )
                maxB = b[ z ][ e ];
        }
        return maxB;
    }
}
```

# D. HMM-Test: test.java

```
import java.util.ArrayList;

public class test {
public static void main(String[] args) {
    /** Anzahl der Sequenzen */
    int anzPatienten = 15;

    Data data = new Data( anzPatienten );
    /** Einlesen der Behandlungsprotokolle */
    data.init();

    /** Anlegen des HMM */
    HMM hmm = new HMM( data.getAnzahlZustaende() , data.getAnzahlMedikamente() );

    /** Berechnen der StartWkt */
    hmm.compute_pi( data.getZustandsLerndaten() );
    /** Berechnen der UebergangsWkt */
    hmm.compute_a( data.getZustandsLerndaten() );
    /** Berechnen der AusgabeWkt */
    hmm.compute_b( data.getMedikamenteLerndaten(), data.getZustandsLerndaten() );

    ArrayList<double[][]> whsMatrix = new ArrayList();
    double[] bewertungAusgabe = new double[ anzPatienten ];
    double[] bewertungUebergang = new double[ anzPatienten ];
    double[] sicherheitAusgabe = new double[ anzPatienten ];
    double[] sicherheitUebergang = new double[ anzPatienten ];

    /** Test: Vergleich zwischen Sequenz und Modell */
    for(int p = 1; p <= anzPatienten; p++){
```

```
        whsMatrix.add( hmm.testModellSequenzAbweichung(
                                data.getPatientMediListNumById(p),
                                data.getPatientZustandListNumById(p) ) );
}

/** Durchlaufen aller Sequenzen */
for(int p = 0; p < anzPatienten; p++){
    for(int e = 0; e < whsMatrix.get(p).length; e++){
        /** Zustandsuebergang/Sicherheit */
        if( whsMatrix.get(p)[e][0] < whsMatrix.get(p)[e][2] ){
            bewertungUebergang[ p ] +=
                ( ( whsMatrix.get(p)[e][2] - whsMatrix.get(p)[e][0] ) / whsMatrix.get(p)[e][2] );
        } else {
            bewertungUebergang[ p ] += 1;
            if( ( whsMatrix.get(p)[e][2] - whsMatrix.get(p)[e][4] ) == 1)
                sicherheitUebergang[ p ]++;
        }
        /** Ausgabe/Sicherheit */
        if( whsMatrix.get(p)[e][1] < whsMatrix.get(p)[e][3] ){
            bewertungAusgabe[ p ] +=
                ( ( whsMatrix.get(p)[e][3] - whsMatrix.get(p)[e][1] ) / whsMatrix.get(p)[e][3] );
        } else {
            bewertungAusgabe[ p ] += 1;
           if( ( whsMatrix.get(p)[e][3] - whsMatrix.get(p)[e][5] ) == 1)
                sicherheitAusgabe[ p ]++;
        }
    }
}

double gesBewertungUebergang = 0;
double gesBewertungAusgabe = 0;
double gesSicherheitAusgabe = 0;
double gesSicherheitUebergang = 0;
int gesAnzahl = 0;
```

```
        for(int p = 0; p < anzPatienten; p++){
            gesBewertungUebergang += bewertungUebergang[ p ];
            gesBewertungAusgabe += bewertungAusgabe[ p ];
            gesSicherheitAusgabe += sicherheitAusgabe[ p ];
            gesSicherheitUebergang += sicherheitUebergang[ p ];
            gesAnzahl += whsMatrix.get(p).length;
        }
    }
}
```

# Quellenverzeichnis

## Literatur

[BKI03] Christoph Beierle und Gabriele Kern-Isberner. *Methoden wissensbasierter Systeme*. 2. Aufl. Wiesbaden: Vieweg, 2003.

[Bur12] J. Compton Burnett. *Die Heilbarkeit von Tumoren durch Arzneimittel*. 3. Aufl. Bd. 3. Schriftenreihe der Clemens von Bönninghausen-Akademie. Müller und Steinicke, 2012.

[Fin03] Gernot A. Fink. *Mustererkennung mit Markov-Modellen: Theorie - Praxis - Anwendungsgebiete*. 1. Aufl. Stuttgart: Teubner, 2003.

[Rab89] L.R. Rabiner. „A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.“ In: *Proceedings of IEEE* 77.2 (1989), S. 257–286.

[Wen04] Andreas Wendemuth. *Grundlagen der stochastischen Sprachverarbeitung*. München: Oldenbourg, 2004.

## Internet

[Pot] *Potenzierung*. 2012. URL: `http://www.psiram.com/ge/index.php/Potenzierung`.