

PROBABILISTISCHES LATENT SEMANTISCHES
INDEXIEREN MIT MEHREREN VIEWS FÜR
INFORMATION RETRIEVAL

Studienarbeit
bei Prof. Dr. Tobias Scheffer
Betreuer Steffen Bickel
am Lehrstuhl Wissenmanagement

vorgelegt von Sebastian Ordyniak
am Fachbereich Informatik der
Humboldt Universität Berlin

Berlin, 5. September 2005

Inhaltsverzeichnis

Einleitung	2
1 Aspekt-Modell	3
1.1 Einführung	3
1.2 Likelihood und Kreuzentropie	5
1.3 Latent Semantische Analyse	6
1.4 Finden der Parameter des Aspekt-Modells	7
1.5 EM-Algorithmus	8
1.6 Schätzen der Parameter des Aspekt-Modells mit dem EM-Algorithmus	10
1.6.1 Einführung in die Methode der Lagrangemultiplikatoren .	11
1.6.2 Herleitung der Updateschritte für das symmetrische Modell	12
1.6.3 Herleitung der Updateschritte für das asymmetrische Modell	16
1.6.4 Tempered EM	17
2 Anwendungen des Aspekt-Modells	20
2.1 Probabilistische Latent Semantische Analyse	20
2.2 Probabilistisches Latent Semantisches Indexieren	21
2.3 Probabilistische Hypertext Induzierte Themen Selektion	22
3 Anwendung auf verlinkte Dokumentensammlungen	23
3.1 Einführung	23
3.2 Single-View	23

	1
3.3 Single-View mit unterschiedlicher Gewichtung der Sichten	24
3.4 Multi-View	25
3.4.1 Einführung	25
3.4.2 Co-EM	26
4 Experimente	28
4.1 WebKB	29
4.2 Cora	29
4.3 News2x2	30
4.4 Auswertung	30
4.5 Schlussfolgerungen	31
Literaturverzeichnis	37

Einleitung

Für Information Retrieval werden oft semantische Indexierungsverfahren eingesetzt, um die Bedeutung von Dokumenten besser zu erfassen. Wir möchten uns mit Indexierungsverfahren beschäftigen, die auf der Probabilistischen Latent Semantischen Analyse (Hofmann, 1998) aufbauen. Es soll der Spezialfall betrachtet werden, dass die Dokumente untereinander verlinkt sind. Dies ist beispielsweise bei Webseiten (Hyperlinks) oder bei Publikationen (Literaturverweise) der Fall. Als Beschreibung des Themas eines Dokumentes können sowohl der Inhalt, als auch die ausgehenden Verweise verwendet werden.

Wenn mehrere solcher Sichten (Views) vorhanden sind, werden für andere Lernprobleme erfolgreich Multi-View-Verfahren verwendet. Bisher wurde Multi-View-Lernen nur für Klassifikation und Clustering eingesetzt. In der Studienarbeit soll untersucht werden, wie man ein mehrstufiges generatives Modell (wie PLSA), das auf einer Sicht arbeitet, mit Multi-View-Lernverfahren lernen kann, wenn mehrere Sichten vorliegen. Die empirische Untersuchung wird sich auf Information Retrieval Probleme beziehen, bei denen man eine verlinkte Dokumentensammlung und eine Anfrage in Form von Suchwörtern UND Suchlinks hat. Dies ist eine spezielle Form von zweiteiligen Suchanfragen, die nach unserem Wissen bisher noch nicht untersucht worden ist.

Die Studienarbeit ist in vier Kapitel unterteilt. Im ersten Kapitel stellen wir das PLSA zugrundeliegende generative Modell (das Aspekt-Modell) vor. Hier gehen wir insbesondere darauf ein, wieso dieses Modell für die semantische Indexierung von Dokumenten geeignet ist. Ein weiterer Schwerpunkt dieses Kapitels bildet die Darstellung der mathematischen Methoden für die Bestimmung der Parameter des Modelles. Um das Aspekt-Modell besser verstehen zu können, stellen wir im zweiten Kapitel einige Anwendungen des Aspekt-Modells vor. Hier gehen wir insbesondere auf die zur Indexierung von Dokumenten verwendeten Verfahren ein. Das dritte Kapitel beschäftigt sich mit dem Lernen des Modells, wenn mehrere Sichten vorhanden sind. Hier stellen wir unterschiedliche Methoden vor, die Informationen der Sichten miteinander zu kombinieren. Im vierten und letzten Kapitel stellen wir die im zweiten Kapitel entwickelten Verfahren einander anhand von drei ausgewählten verlinkten Dokumentsammlungen empirisch gegenüber.

Kapitel 1

Aspekt-Modell

1.1 Einführung

Das Aspekt-Modell wurde von Thomas Hofmann und Jan Puzicha in [14] als ein Mischmodell für zweigeteilte Daten vorgestellt. Zweigeteilte Daten bezieht sich dabei auf Lernprobleme, bei denen die Beobachtungen immer für Paare von Objekten (aus unterschiedlichen, endlichen Objektmengen) gemacht werden. Seien dazu $X = \{x_1, \dots, x_I\}$ und $Y = \{y_1, \dots, y_J\}$ zwei endliche Objektmengen mit unterschiedlichen Domänen. Eine Beobachtung besteht im einfachsten Fall aus einem Paar (x, y) von Objekten aus X bzw. Y . Zusätzlich zu der Existenz eines solchen Paares, kann mit jedem Paar auch ein Wert $w(x, y)$ verknüpft werden. Dieser Wert gibt dann je nach Anwendungsgebiet z.B. die Stärke der Verknüpfung bzw. die Häufigkeit eines solchen Paares wieder.

Im Falle einer Sammlung von Dokumenten ist X die Menge der Dokumente und Y das Vokabular - also die Menge der Worte. Ein Paar (x, y) steht für das Auftreten des Wortes y im Dokument x . Eine Anwendung, die aus zweigeteilten Daten lernt, muss hauptsächlich zwei Aufgaben erfüllen:

1. Lernen eines Wahrscheinlichkeitsmodelles über $X \times Y$.
2. Identifizieren von Strukturen in den gegebenen Daten, z.B. Cluster oder Hierarchien.

In [14] stellen Thomas Hofmann und Jan Puzicha verschiedene statistische Modelle vor, mit deren Hilfe sich die Daten auf verschiedene Art und Weise strukturieren lassen. Ein solches statistisches Modell ist das Aspekt-Modell. Das Aspekt-Modell weist jeder Beobachtung (jedem Paar von Objekten) eine verborgene (engl. latent) Klasse zu. Jedes Paar wird also mit einer (verborgenen) Variablen verknüpft, die angibt zu welcher Klasse das Paar gehört. Paare, die zur gleichen Klasse

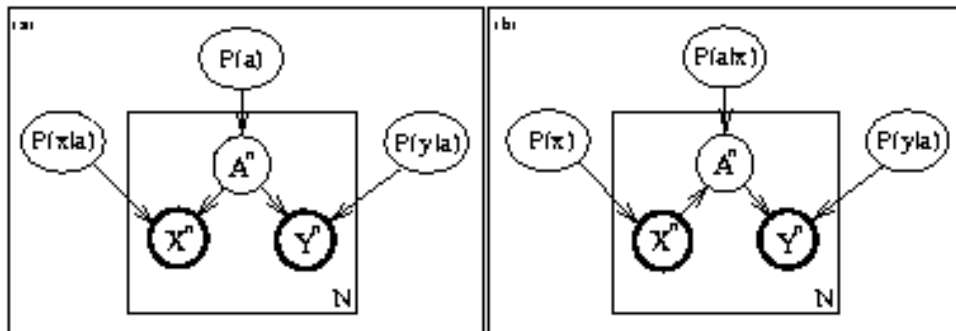


Abbildung 1.1: Graphische Darstellung des Aspekt-Modelles: (a) symmetrische Parametrisierung (b) asymmetrische Parametrisierung. Kreise repräsentieren Zufallsvariablen (fette Umrandungen entsprechen den beobachtbaren Daten), Ellipsen repräsentieren Parameter und Rechtecke symbolisieren mehrere Instanzen. Pfeile symbolisieren Abhängigkeiten zwischen den Zufallsvariablen.

gehören bezeichnet man auch als Aspekt (daher der Name Aspekt-Modell). Bevor wir das Aspekt-Modell einführen, geben wir zunächst eine formale Problembeschreibung an. Startpunkt beim Lernen aus zweigeteilten Daten ist eine Abfolge von Beobachtungen $S = ((x^1, y^1), \dots, (x^N, y^N))$ mit $x^n \in X$ und $y^n \in Y$. Diese Abfolge ist eine Realisierung der zugrundeliegende Abfolge von Zufallsvariablen $((X^1, Y^1), \dots, (X^N, Y^N))$. Das Aspekt-Modell ordnet nun jedem Paar von Zufallsvariablen (X^n, Y^n) eine (verborgene) Klassenvariable über die Zufallsvariable A^n zu. Dem Aspekt-Modell werden zwei Grundannahmen über den Generierungsprozess zugrundegelegt:

1. Die Verteilungen der Zufallsvariablen X^n, Y^n sind für alle n gleich und unabhängig von anderen Paaren. Dies entspricht der Annahme, dass die Beobachtungen unabhängig voneinander und gleichverteilt gemacht werden.
2. X^n und Y^n sind unabhängig voneinander gegebenen A^n .

Der zugrundeliegende Datengenerierungsprozess lässt sich wie folgt beschreiben:

1. Ziehe eine Klasse (Aspekt) mit Wahrscheinlichkeit $P(a)$.
2. Wähle ein Objekt x aus der Menge X , mit Wahrscheinlichkeit $P(x|a)$.
3. Wähle ein Objekt y aus der Menge Y , mit Wahrscheinlichkeit $P(y|a)$.

Die Parameter des Aspekt-Modelles sind also die Wahrscheinlichkeiten $P(a)$, $P(x|a)$ und $P(y|a)$. Die Wahrscheinlichkeit einer gegebenen Menge von Paaren S und den zugehörigen (verborgenen) Klassen ist dann:

$$\begin{aligned}
 P(S, (a^1, \dots, a^N)) &= \prod_{n=1}^N P(x^n, y^n, a^n) \\
 P(x^n, y^n, a^n) &= P(a^n) P(x^n | a^n) P(y^n | a^n)
 \end{aligned}$$

Eine Graphische Darstellung des Aspekt-Modelles ist in Abbildung 1.1 a) zu sehen. Summiert man für jedes Paar über alle möglichen Klassen und gruppiert identische Paare, so erhält man die Wahrscheinlichkeit der beobachteten Paare:

$$\begin{aligned}
 P(S) &= \prod_{(x,y) \in X \times Y} P(x,y)^{n(x,y)} & (1.1) \\
 P(x,y) &= \sum_{a \in A} P(a) P(x|a) P(y|a)
 \end{aligned}$$

$n(x,y)$ steht dabei für die (u.U. gewichtete) Häufigkeit des Paares (x,y) in S . Die Wahrscheinlichkeit der beobachteten Paare bezeichnet man auch als Likelihood. Für eine gegebene Menge S bestimmt man die Parameter $P(a)$, $P(x|a)$ und $P(y|a)$, die die Likelihood maximieren. Ein solches Vorgehen wird auch als Maximum-Likelihood-Methode bezeichnet. Mit der Maximum-Likelihood-Methode findet man also die Parameter, die am besten mit den gegebenen Beobachtungen übereinstimmen. Ist die Anzahl der Beobachtungen gross, so entsprechen diese Parameter genau der zugrundeliegenden Verteilung. Formt man die symmetrische Parametrisierung mit Hilfe des Bayesschen Satzes um, so erhält man:

$$\sum_{a \in A} P(y|a) P(x|a) P(a) = \sum_{a \in A} P(y|a) P(x,a) = \sum_{a \in A} P(y|a) P(a|x) P(x) \quad (1.2)$$

Diese Parametrisierung nennt man auch asymmetrische Parametrisierung des Aspekt-Modells (wegen der Asymmetrie bezüglich der verborgenen Klassenvariable). Die asymmetrische Parametrisierung des Aspekt-Modells ist in Abbildung 1.1 (b) dargestellt.

1.2 Likelihood und Kreuzentropie

Wir wollen in diesem Kapitel eine anschauliche Bedeutung für die Lösungsparameter des Aspekt-Modelles herleiten. Wie beschrieben findet man mit der Maximum-Likelihood-Methode die Parameter, die die Likelihood der Daten maximieren. Diese Parameter maximieren dann auch die logarithmierte Likelihood:

$$\log(P(S)) = \sum_{(x,y) \in X \times Y} n(x,y) \log \left(\sum_{a \in A} P(a) P(x|a) P(y|a) \right)$$

Teilt man diese Gleichung durch die Anzahl (bzw. das Gewicht) der N Paare und fasst $\frac{n(x,y)}{N}$ als empirische Verteilung $\hat{P}(x,y)$ der Paare auf, so erhält man:

$$\frac{\log P(S)}{N} = \sum_{(x,y) \in X \times Y} \hat{P}(x,y) \log P(x,y)$$

Das ist die negierte Kreuzentropie der Verteilungen $\hat{P}(x,y)$ und $P(x,y)$. D.h. das Finden der Parameter, die die Likelihood der Daten maximieren ist äquivalent zum Finden der Parameter, die die Kreuzentropie der Verteilungen $\hat{P}(x,y)$ und $P(x,y)$ minimieren.

Um die durch das Aspekt-Modell beschriebene Lösung besser zu verstehen, schauen wir uns nochmal die Parametrisierung der Verteilung $P(x,y)$ für das asymmetrische Modell aus Gleichung 1.2 an. Wegen $\sum_{a \in A} P(a|x) = 1$ ist jeder der Vektoren $P(x, \cdot)$ eine konvexe Kombination der $|A|$ Vektoren $P(\cdot|a)$. D.h. alle I Vektoren $P(x, \cdot)$ kommen aus dem $|A|$ -dimensionalen Raum (genauer Simplex, wegen $\sum_{y \in Y} P(y|a) = 1$), der von den Vektoren $P(\cdot|a)$ aufgespannt wird. Lösungen für $P(x,y)$ sind also nur für diesen reduzierten Suchraum optimal.

Die $I \times J$ -Matrix L der gefundenen Verteilung $P(x,y)$ hat also Rank höchstens $|A|$. Bezeichnet man die Matrix der empirischen Verteilung $\hat{P}(x,y)$ mit M , dann kann man sagen:

Die Maximum-Likelihood-Methode findet zu einer gegebenen Matrix M von Beobachtungen, die Matrix L mit Rank höchstens $|A|$, deren Einträge die Kreuzentropie (zu den Einträgen der Matrix M) minimieren.

1.3 Latent Semantische Analyse

Wir wollen in diesem Kapitel auf die Ähnlichkeit von PLSA zur Latent Semantischen Analyse (LSA) eingehen. Die Latent Semantische Analyse [7] ist wie PLSA ein Verfahren zur semantischen Analyse von Textsammlungen.

LSA findet zu einer gegebenen Matrix M von Beobachtungen, die Matrix vom Rank k , die der Matrix M bezüglich der L_2 -Norm¹ am nächsten steht. Setzt man $k = |A|$ unterscheiden sich beide Methoden nur noch durch die gewählte Norm für

¹Die L_2 -Norm entspricht der Summe der quadratischen Abstände zwischen den Einträgen der Matrizen

die Optimierung der Lösung. Thomas Hofmann und Jan Puzicha weisen in [14] auf folgende Vorteile ihres Verfahrens gegenüber dem LSA-Verfahren hin:

- Die Kreuzentropie ist wegen Ihrer mathematischen Eigenschaften besser für die Modellierung von Häufigkeiten geeignet als die von LSA verwendete L_2 -Norm.
- Die Lösung des Aspekt-Modells hat im Gegensatz zur Lösung von LSA (die Lösungsmatrix von LSA kann sogar negative Einträge enthalten) eine wohldefinierte statistische Bedeutung.

1.4 Finden der Parameter des Aspekt-Modells

Wie bereits beschrieben ist die Lösung des Aspekt-Modells die Menge der Parameter, die die Likelihood der Beobachtungen maximieren. Für das Finden der Parameter muss man also die Likelihood-Funktion des Aspekt-Modells maximieren. Die Likelihood ergibt sich wie beschrieben je nach Modell zu:

1. asymmetrisches Modell:

$$L(\theta) = \prod_{(x,y) \in X \times Y} \left(\sum_{a \in A} P(x) P(a|x) P(y|a) \right)^{n(x,y)}$$

2. symmetrisches Modell:

$$L(\theta) = \prod_{(x,y) \in X \times Y} \left(\sum_{a \in A} P(a) P(x|a) P(y|a) \right)^{n(x,y)}$$

Dabei bezeichnet θ den Vektor der Parameter des Aspekt-Modelles. Zur Vereinfachung schreibe ich im folgenden für $z \in Z$ nur noch z , wenn die Menge aus dem Zusammenhang ersichtlich ist. Das Optimieren der Likelihood ist äquivalent zum Optimieren der logarithmierten Likelihood:

1. asymmetrisches Modell:

$$\begin{aligned} L_{\log}(\theta) &= \sum_{(x,y)} n(x,y) \log \left(\sum_a P(x) P(a|x) P(y|a) \right) \\ &= \sum_{(x,y)} n(x,y) \left(\log(P(x)) + \log \left(\sum_a P(a|x) P(y|a) \right) \right) \end{aligned} \quad (1.3)$$

2. symmetrisches Modell:

$$L_{\log}(\theta) = \sum_{(x,y)} n(x,y) \log \left(\sum_a P(a) P(x|a) P(y|a) \right)$$

Diese Funktion ist wegen der Summe im Logarithmus schwierig zu optimieren. Zum Finden der Parameter des Aspekt-Modells verwenden wir deshalb den im nächsten Abschnitt beschriebenen EM-Algorithmus.

1.5 EM-Algorithmus

Der EM-Algorithmus ist eine Methode zum Schätzen der Parameter, die eine Likelihoodfunktion optimieren², wenn nicht alle Werte der Attribute der Beispiele bekannt sind [2]. Sei dazu x ein Beispiel aus X , y_x der zugehörige Vektor mit den Werten der bekannten Attribute von x und z_x der Vektor mit den Werten der unbekannt Attribute von x . Da es für unsere Zwecke ausreichend ist, wenn die Aufteilung der Attribute für alle Beispiele dieselbe ist, gehen wir im folgenden davon aus, weisen aber darauf hin, dass sich der EM-Algorithmus auch anwenden lässt, wenn dies nicht der Fall ist. Die logarithmierte Likelihood ist dann:

$$L_{\log}(\theta) = \sum_x \log(p(y_x, z_x | \theta))$$

Dabei bezeichnet θ die Parameter des zugrundeliegenden Wahrscheinlichkeitsmodells. Die Maximumlikelihoodmethode besteht nun darin, die Parameter θ zu finden, die diese Likelihood maximieren. Da z_x aber nicht bekannt ist, ist ein Maximieren dieser Funktion über θ nicht möglich. Die Idee, die hinter dem EM-Algorithmus steckt, ist es nun für jeden unbekanntes Attributwert eine Zufallsvariable einzuführen, die die Werte des zugehörigen Attributes repräsentiert. Die Verteilung dieser Zufallsvariablen wird dabei so gewählt, dass sie möglichst gut mit den auf dem Beispiel bekannten Attributwerten übereinstimmt. Der EM-Algorithmus ist ein iterativer Algorithmus. Jeder Schritt des EM-Algorithmus ist dabei in zwei Phasen eingeteilt:

1. (Expectation): Berechne die Verteilung der Zufallsvariablen der unbekanntes Attributwerte. Benutze dazu das Modell des vorangegangenen Schrittes.
2. (Maximization): Finde ein lokales Optimum des Erwartungswertes der Likelihood über die unbekanntes Attributwerte. Benutze dazu die Verteilung der Zufallsvariablen der unbekanntes Attributwerte aus Phase 1.

²Optimieren ist hier nicht im mathematischen Sinne gemeint, sondern vielmehr ein der Problemstellung angepasster Begriff. Wir werden im diesem Kapitel noch die Funktion kennenlernen, die der EM-Algorithmus mathematisch optimiert.

Der EM-Algorithmus startet also mit einem beliebigen (meist zufällig initialisierten) Modell und verbessert dieses solange, bis sich das Modell nicht mehr ändert. Die Verteilung der Zufallsvariablen für die unbekanntes Attributwerte ergibt sich aus dem vorherigen Modell θ^{i-1} wie folgt:

$$P(z_x|y_x, \theta^{i-1}) = \frac{P(z_x, y_x | \theta^{i-1})}{\int_{z_x \in \Upsilon} P(z_x, y_x | \theta^{i-1})} \quad (1.4)$$

Dabei bezeichnet Υ die Menge an Werten, die z_x annehmen kann. In der Literatur über den EM-Algorithmus findet man oft diese Schreibweise für die vom EM-Algorithmus im M-Schritt maximierte Funktion:

$$Q(\theta^i, \theta^{i-1}) = E[\log p(Y, Z | \theta) | Y, \theta^{i-1}] = \sum_x \int_{z_x \in \Upsilon} \log(p(y_x, z_x | \theta^i)) p(z_x | y_x, \theta^{i-1}) \quad (1.5)$$

Y und Z sind die Mengen aller y_x bzw. z_x . Die beiden Phasen lassen sich dann abgekürzt auch so verstehen:

1. Berechne die Verteilung der unbekanntes Attribute, benutze dazu Gleichung 1.4.
2. Finde ein lokales Maximum der Q-Funktion über θ^i gegeben $p(z_x|y_x, \theta^{i-1})$ aus Phase 1.

Kommen wir nun zu der Funktion, für die der EM-Algorithmus ein lokales Optimum findet [8]:

$$Q'(\theta) = \sum_x \int_{z_x \in \Upsilon} \log\left(\frac{p(y_x, z_x)}{p(z_x|y_x)}\right) p(z_x|y_x)$$

Es gilt:

Der EM-Algorithmus findet ein lokales Maximum von $Q'(\theta)$ bzgl. des Modells und der möglichen Verteilungen der Zufallsvariablen z_x .

Die Funktion Q' wird dabei oft mit der negierten Funktion der freien Energie aus der statistischen Physik identifiziert. Man sagt deshalb der EM-Algorithmus minimiert die freie Energie. Die beiden Phasen des EM-Algorithmus lassen sich mithilfe der Funktion Q' auch so ausdrücken:

1. Finde ein lokales Maximum der Q' -Funktion über die Verteilungen $p(z_x|y_x)$. Die Wahrscheinlichkeiten der Beispiele ergeben sich dabei aus dem vorherigen Modell.

2. Finde ein lokales Maximum der Q' -Funktion über die Modellparameter, gegeben die Verteilungen $p(z_x|y_x)$ aus Phase 1.

Eine wichtige Anwendung des EM-Algorithmus ist das Optimieren der Likelihood von misch- bzw. faktorisierten Wahrscheinlichkeitsmodellen (wie beim Aspekt-Modell). Im nächsten Kapitel widmen wir uns deshalb der Anwendung des EM-Algorithmus zum Finden der Parameter des Aspekt-Modells.

1.6 Schätzen der Parameter des Aspekt-Modells mit dem EM-Algorithmus

Kommen wir also nun zur Anwendung des EM-Algorithmus auf das Aspekt-Modell. Beim Aspekt-Modell wird jedem Beispiel (Paar (x, y)) eine Klasse zugeordnet. Fasst man diese Klasse als verstecktes Attribut des Beispiels auf, dann ergibt sich die Likelihood der Klassen und Beispiele zu:

1. asymmetrisches Modell:

$$L_{\log}(\theta) = \sum_{(x,y)} n(x,y) \log(P(a)P(a|x)P(y|a))$$

2. symmetrisches Modell:

$$L_{\log}(\theta) = \sum_{(x,y)} n(x,y) \log(P(a)P(x|a)P(y|a))$$

Auf diese Likelihood (der Daten und verborgenen Klassen) lässt sich dann der EM-Algorithmus anwenden und es gilt:

Satz 1.1. *Der EM-Algorithmus findet ein lokales Optimum der ursprünglichen Likelihoodfunktion (der Likelihood der Paare (x, y)).*

Diesen Satz werden wir hier nicht beweisen. Damit ergibt sich für die Q' -Funktion des Aspekt-Modells:

1. asymmetrisches Modell:

$$Q' = \sum_{x,y} n(x,y) \sum_a \log\left(\frac{P(y|a)P(a|x)P(x)}{P(a|x,y)}\right) P(a|x,y) \quad (1.6)$$

2. symmetrisches Modell:

$$Q' = \sum_{x,y} n(x,y) \sum_a \log\left(\frac{P(y|a)P(x|a)P(a)}{P(a|x,y)}\right) P(a|x,y) \quad (1.7)$$

Diese Funktion ist analytisch leichter zu optimieren. Die Ergebnisse der beiden Phasen des EM-Algorithmus findet man für diese Funktion durch 0-Setzen der 1. Ableitung nach den Parametern der entsprechenden Phase. Die hier vorgestellte Vorgehensweise lässt sich auch auf andere Wahrscheinlichkeitsmodelle anwenden (z.B. einer Mischerteilung). Die folgenden Kapitel beschäftigen sich mit der Herleitung der für die Modelle des Aspekt-Modells notwendigen Updateschritte. Dazu ist es aber zunächst notwendig eine Einführung in die Methode der Lagrangemultiplikatoren zu geben, da diese Methode für die Herleitung von grosser Bedeutung ist.

1.6.1 Einführung in die Methode der Lagrangemultiplikatoren

Für die Herleitung der E- und M-Schritte müssen wir die Q' -Funktion über die jeweils freien Variablen maximieren. Für diese freien Variablen gelten Nebenbedingungen, die wir beim Maximieren berücksichtigen müssen. Formal lässt sich dieses Problem wie folgt ausdrücken:

Maximiere eine Funktion $f(x_1, \dots, x_n) : D \rightarrow \mathbf{R}$ mit $D \subseteq \mathbf{R}^n$ unter den Nebenbedingungen $g_i(x_1, \dots, x_n) = 0$ mit $g_i(x_1, \dots, x_n) : D \rightarrow \mathbf{R}$ und $i = 1 \dots m$.

Um ein solches Problem zu lösen, verwenden wir die Methode der Lagrangemultiplikatoren. Die Grundlage dieses Verfahrens bildet der folgende Satz (siehe [13]):

Satz 1.2. $D \subseteq \mathbf{R}^n$ sei eine offene Menge und $f, g_1, \dots, g_m : D \rightarrow \mathbf{R}$ seien stetig differenzierbare Funktion auf D . Falls f im Punkt (a_1, \dots, a_n) ein Extrempunkt hat unter den Nebenbedingungen $g_i(x_1, \dots, x_n) = 0$ für alle $i = 1 \dots m$, so ist die Menge $(\text{grad}(f(a_1, \dots, a_n)), \text{grad}(g_1(a_1, \dots, a_n)), \dots, \text{grad}(g_m(a_1, \dots, a_n)))$ linear abhängig.

Die Lineare Abhängigkeit der Gradienten der Nebenbedingungen und der zu maximierenden Funktion ist also ein notwendiges Kriterium für die Existenz eines lokalen Extrempunktes von f unter den gegebenen Nebenbedingungen. D.h. für einen Extrempunkt müssen $(\lambda_1, \dots, \lambda_m)$ existieren, die das folgende Gleichungssystem erfüllen:

$$\begin{aligned} 0 &= \text{grad}(f(x_1, \dots, x_n)) + \sum_{i=1}^m \lambda_i \text{grad}(g_i(x_1, \dots, x_n)) \\ 0 &= g_1(x_1, \dots, x_n) \\ &\vdots \\ 0 &= g_m(x_1, \dots, x_n) \end{aligned}$$

Ein Punkt, für den diese Gleichungen erfüllt sind, ist aber gerade ein stationärer Punkt der sogenannten Lagrangefunktion:

$$H(x_1, \dots, x_m, \lambda_1, \dots, \lambda_m) = f(x_1, \dots, x_n) + \sum_{i=1}^m \lambda_i g_i(x_1, \dots, x_n)$$

D.h. für das Finden der lokalen Extrempunkte einer Funktion mit Nebenbedingungen, genügt es die stationären Punkte der zugehörigen Lagrangefunktion auszurechnen und zu schauen, ob diese auch wirklich Extrempunkte darstellen. In den von uns zu untersuchenden Funktionen, ist die lineare Abhängigkeit der Gradienten sogar hinreichend für die Frage, ob ein gegebener Punkt einen Extrempunkt darstellt. D.h. alle stationären Punkte der Lagrangefunktion sind auch gleichzeitige lokale Extrema.

1.6.2 Herleitung der Updateschritte für das symmetrische Modell

Wir werden zunächst die sich aus dem E-Schritt für die Wahrscheinlichkeiten $P(a|x, y)$ ergebenden Updateschritte herleiten. Dazu müssen wir die Q' -Funktion aus Gleichung 1.7 nach den Parametern $P(a|x, y)$ ableiten, wobei wir die eigentlichen Modellparameter konstant halten.

Wollen wir diese Funktion nach den Wahrscheinlichkeiten $P(a|x, y)$ ableiten müssen wir die Nebenbedingung berücksichtigen, die besagen, dass jedes Paar (x, y) mit Wahrscheinlichkeit eins aus irgendeiner Klasse kommt, d.h. :

$$\forall_{x,y} : \sum_a P(a|x, y) - 1 = 0$$

Für das Maximieren mit Nebenbedingungen verwenden wir die im letzten Abschnitt beschriebene Methode der Lagrangemultiplikatoren. Die Lagrangefunktion ergibt sich dann zu:

$$H(P(a|x, y), \lambda_{xy}) = \left(\sum_{x,y} n(x, y) \sum_a \log \left(\frac{P(y|a) P(x|a) P(a)}{P(a|x, y)} \right) P(a|x, y) \right) + \sum_{x,y} \lambda_{xy} \left(\left(\sum_a P(a|x, y) \right) - 1 \right)$$

Die Ableitungen nach $P(a|x, y)$ und λ_{xy} ergeben:

$$0 = n(x, y) \log \left(\frac{P(y|a) P(x|a) P(a)}{P(a|x, y)} \right) - 1 + \lambda_{xy} \quad (1.8)$$

$$0 = \left(\sum_a P(a|x, y) \right) - 1 \quad (1.9)$$

Formt man die Gleichung 1.8 etwas um erhält man für $P(a|x, y)$:

$$P(a|x, y) = \frac{P(y|a)P(x|a)P(a)}{e^{\frac{-\lambda_{xy}+n(x,y)}{n(x,y)}}} \quad (1.10)$$

Um einen Wert für $e^{\frac{-\lambda_{xy}+n(x,y)}{n(x,y)}}$ zu erhalten, formen wir die Gleichung 1.8 um zu:

$$P(y|a)P(x|a)P(a) = e^{\frac{-\lambda_{xy}+n(x,y)}{n(x,y)}} P(a|x, y) \quad (1.11)$$

Summiert man alle zu einem Paar (x, y) gehörenden Gleichungen 1.11, so erhält man:

$$\sum_z P(y|a)P(x|a)P(a) = e^{\frac{-\lambda_{xy}+n(x,y)}{n(x,y)}} \sum_z P(a|x, y) \quad (1.12)$$

Zusammen mit Gleichung 1.9 ergibt das:

$$e^{\frac{-\lambda_{xy}+n(x,y)}{n(x,y)}} = \sum_z P(y|a)P(x|a)P(a) \quad (1.13)$$

Setzt man nun Gleichung 1.13 in Gleichung 1.10 ein, erhält man für den E-Schritt:

$$P(a|x, y) = \frac{P(y|a)P(x|a)P(a)}{\sum_z P(y|a)P(x|a)P(a)}$$

Kommen wir nun zum M-Schritt des Algorithmus und somit zu den Update-schritten für die Parameter des Modells $P(y|a)$, $P(x|a)$ und $P(a)$. Um die Herleitung der Schritte zu vereinfachen, führen wir zunächst eine Variablentrennung an Q' durch:

$$Q' = \sum_{x,y,a} \log(P(y|a)) + \sum_{x,y,a} \log(P(x|a)) + \sum_{x,y,a} \log(P(a)) - \sum_{x,y,a} \log(P(a|x, y)) n(x, y) P(a|x, y) \quad (1.14)$$

Wie man sieht hängt der Term $\sum_{x,y,a} \log(P(a|x, y))$ nicht von den Modellparametern ab und die Q' -Funktion vereinfacht sich für den M-Schritt, wie zu erwarten zur Q -Funktion aus Gleichung 1.5.

$$Q = \left(\sum_{x,y,a} \log(P(y|a)) + \sum_{x,y,a} \log(P(x|a)) + \sum_{x,y,a} \log(P(a)) \right) * n(x,y)P(a|x,y) \quad (1.15)$$

Um diese Funktion nach den Modellparameter zu maximieren können wir jede Summe einzeln maximieren. Daraus ergeben sich die Abhängigkeiten der Q -Funktion zu:

$$Q(P(y|a)) = \sum_{x,y,a} \log(P(y|a)) n(x,y)P(a|x,y) \quad (1.16)$$

$$Q(P(x|a)) = \sum_{x,y,a} \log(P(x|a)) n(x,y)P(a|x,y) \quad (1.17)$$

$$Q(P(a)) = \sum_{x,y,a} \log(P(a)) n(x,y)P(a|x,y) \quad (1.18)$$

Für das Maximieren dieser Funktionen sind für uns noch die folgenden Nebenbedingungen für $P(y|a)$, $P(x|a)$ sowie $P(a)$ des symmetrischen Modelles interessant.

$$\forall a : \sum_y P(y|a) = 1$$

$$\forall a : \sum_x P(x|a) = 1$$

$$\sum_a P(a) = 1$$

Das Maximieren mit Nebenbedingungen führen wir mit Hilfe der Lagrange-Multiplikatoren durch. Jeder der zu maximierenden Q 's entspricht dabei einer neuen Funktion H , welche durch Addition des Lagrange-Multiplikators aus Q entsteht. Für die Gleichungen 1.16-1.18 ergeben sich dabei die folgenden Lagrangefunktionen:

$$H(P(y|a), \lambda_a) = \sum_{x,y,a} \log(P(y|a)) n(x,y) P(a|x,y) + \sum_z \lambda_a \left(\sum_w (P(y|a)) - 1 \right) \quad (1.19)$$

$$H(P(x|a), \lambda_a) = \sum_{x,y,a} \log(P(x|a)) n(x,y) P(a|x,y) + \sum_z \lambda_a \left(\sum_d (P(x|a)) - 1 \right) \quad (1.20)$$

$$H(P(a), \lambda) = \sum_{x,y,a} \log(P(a)) n(x,y) P(a|x,y) + \lambda \left(\sum_a (P(a)) - 1 \right) \quad (1.21)$$

Für die Funktion 1.19 ergibt sich bei Ableitung nach den freien Parametern also folgendes Gleichungssystem (für jedes Paar (x, y)):

$$\begin{aligned} 0 &= \sum_x \frac{n(x,y) P(a|x,y)}{P(y|a)} + \lambda_a \\ &= \sum_x n(x,y) P(a|x,y) + \lambda_a P(y|a) \end{aligned} \quad (1.22)$$

$$0 = \sum_y (P(y|a)) - 1 \quad (1.23)$$

Summiert man alle zu 1.22 zur selben Klasse a gehörenden Gleichungen und benutzt 1.23, dann ergibt sich für λ_a :

$$\lambda_a = - \sum_{y,x} n(x,y) P(a|x,y)$$

Eingesetzt in 1.22 ergibt sich für $P(y|a)$:

$$P(y|a) = \frac{\sum_x n(x,y) P(a|x,y)}{\sum_{x,y'} n(x,y') P(a|x,y')}$$

Die restlichen Updateschritte ergeben sich analog. D.h. für $P(x|a)$ ergibt sich:

$$P(x|a) = \frac{\sum_y n(x,y) P(a|x,y)}{\sum_{y,x'} n(x',y) P(a|x',y)}$$

Und für $P(a)$:

$$P(a) = \frac{\sum_{x,y} n(x,y) P(a|x,y)}{\sum_{y,x,a'} n(x,y) P(a|x,y')}$$

Zusammengefasst ergeben sich für das symmetrische Modell die folgenden Updateschritte:

E-Schritt:

$$P(a|x,y) = \frac{P(y|a) P(x|a) P(a)}{\sum_{a'} P(y|a') P(x|a') P(a')} \quad (1.24)$$

M-Schritt:

$$P(y|a) = \frac{\sum_x n(x,y) P(a|x,y)}{\sum_{x,y'} n(x,y') P(a|x,y')} \quad (1.25)$$

$$P(x|a) = \frac{\sum_y n(x,y) P(a|x,y)}{\sum_{y,x'} n(x',y) P(a|x',y')} \quad (1.26)$$

$$P(a) = \frac{\sum_{x,y} n(x,y) P(a|x,y)}{\sum_{y,x,a'} n(x,y) P(a'|x,y)} \quad (1.27)$$

Dies entspricht den von Hofmann in [9] angegebenen Updateschritten für das symmetrische Modell.

1.6.3 Herleitung der Updateschritte für das asymmetrische Modell

Die Updateschritte für das asymmetrische Modell ergeben sich völlig analog zu denen des symmetrischen Modells. D.h. mit der entsprechenden Q' -Funktion:

$$Q' = \sum_{x,y} n(x,y) \sum_a \log \left(\frac{P(y|a) P(x|a) P(a)}{P(a|x,y)} \right) P(a|x,y)$$

und den Nebenbedingungen des asymmetrischen Modells:

$$\forall a : \sum_y P(y|a) = 1$$

$$\forall x : \sum_a P(a|x) = 1$$

$$\sum_x P(x) = 1$$

kommt man auf die folgenden Updateschritte für das asymmetrische Modell:

E-Schritt:

$$P(a|x,y) = \frac{P(y|a)P(a|x)P(x)}{\sum_{a'} P(y|a')P(a'|x)P(x)}$$

M-Schritt:

$$P(y|a) = \frac{\sum_x n(x,y)P(a|xy)}{\sum_{x,y'} n(x,y')P(a|xy')}$$

$$P(a|x) = \frac{\sum_y n(x,y)P(a|xy)}{\sum_{y,a'} n(x,y)P(a'|xy')}$$

$$P(x) = \frac{\sum_{w,a} n(x,y)P(a|xy)}{\sum_{y,x',a} n(x',y)P(a|x'y')}$$

1.6.4 Tempered EM

Mit der Maximum-Likelihood-Methode findet man Modellparameter, die sehr gut mit den Trainingsdaten übereinstimmen. Ein Problem dieser Methode ist die Anwendung des gefundenen Modells auf ungesehene Daten. Im Allgemeinen kann man nicht davon ausgehen, dass sich ein Modell, welches exakt mit den Trainingsdaten übereinstimmt, auch auf andere Daten übertragen lässt. Um dieses Problem abzuschwächen, unterteilt man die vorhandenen Daten oft in Trainings- und Tuningmenge und optimiert die Performance des Modells für die Tuningmenge. Tempered EM ist eine von Thomas Hoffmann entwickelte Methode, die diese Idee auf den EM-Algorithmus überträgt. Dazu misst man nach jeder Iteration die Likelihood auf einer zuvor definierten Tuningmenge. Wenn sich diese nicht mehr verbessert hört man entweder auf, oder passt die vom EM-Algorithmus maximierte Funktion an. Um dies zu verstehen schauen wir uns nochmal die von EM maximierte Funktion an. Hoffmann führt einen zusätzlichen Parameter β ein und definiert die Q'_{temp} -Funktionen wie folgt:

1. asymmetrisches Modell:

$$Q'_{temp} = \sum_{x,y} n(x,y) \sum_a \log \left(\frac{(P(y|a)P(a|x)P(x))^\beta}{P(a|x,y)} \right) P(a|x,y)$$

2. symmetrisches Modell:

$$Q'_{temp} = \sum_{x,y} n(x,y) \sum_a \log \left(\frac{(P(y|a)P(x|a)P(a))^\beta}{P(a|x,y)} \right) P(a|x,y)$$

Um die Wirkung von β besser zu veranschaulichen formen wir dies um und erhalten:

1. asymmetrisches Modell:

$$Q'_{temp} = \beta \sum_{x,y,a} n(x,y) \log(P(y|a)P(a|x)P(x))P(a|x,y) - \sum_{x,y,a} n(x,y) \log(P(a|x,y))P(a|x,y)$$

2. symmetrisches Modell:

$$Q'_{temp} = \beta \sum_{x,y,a} n(x,y) \log(P(y|a)P(x|a)P(a))P(a|x,y) - \sum_{x,y,a} n(x,y) \log(P(a|x,y))P(a|x,y)$$

Für den EM-Algorithmus ändert sich mit der neuen Funktion lediglich der E-Schritt zu:

1. asymmetrisches Modell:

$$P(a|x,y) = \frac{(P(y|a)P(a|x)P(x))^\beta}{\sum_{z'} (P(w|z')P(z'|d)P(x))^\beta}$$

2. symmetrisches Modell:

$$P(a|x,y) = \frac{(P(y|a)P(x|a)P(a))^\beta}{\sum_{z'} (P(w|z')P(d|z')P(z'))^\beta}$$

Für $\beta = 1$ entspricht die Q'_{temp} -Funktion der ursprünglichen Q' -Funktion. Für $\beta < 1$ erhöht man den Anteil der Entropy der Posterior, d.h. es werden Posterior bevorzugt die eine grössere Entropy aufweisen. Für $\beta = 0$ sind die Posterior gleichverteilt. Tempered EM arbeitet nun wie folgt:

1. Starte den EM-Algorithmus mit $\beta = 1$ und lasse ihn solange laufen, bis die Verbesserung der Likelihood unterhalb eines Schwellwertes (*likelihoodborder*) liegt.
2. Erniedrige β (setze $\beta = \beta\mu$ für ein $\mu < 1$) und setze den EM-Algorithmus solange fort, bis β unterhalb von *betaend* liegt.
3. Mache *finaliterations* Iterationen mit $\beta = 1$.

Jedes Mal, wenn sich die Likelihood auf den Tuningdaten nicht mehr verbessert, werden Posterior mit höherer Entropy bevorzugt. Tempered EM hat für die von Hoffmann durchgeführten Experimente folgende Verbesserung gegenüber dem einfachen EM gezeigt:

- Tempered EM braucht laut Hoffmann weniger Iterationen.
- Das gelernte Modell eignet sich besser, für die Anwendung auf ungesehene Daten.

Kapitel 2

Anwendungen des Aspekt-Modells

Um die Möglichkeiten des Aspekt-Modells zu illustrieren, werden wir in diesem Kapitel einige Anwendungen des Aspekt-Modells vorstellen. Insbesondere werden wir das Probabilistische Latent Semantische Indexieren vorstellen und damit die für die Indexierung von Dokumenten mit dem Aspekt-Modell notwendigen Verfahrensweisen kennenlernen. Um zu zeigen, dass sich das Aspekt-Modell nicht nur für die Analyse von reinen Textdaten eignet stellen wir im letzten Abschnitt dieses Kapitels die Probabilistische Hypertext Induzierte Themen Selektion (PHITS) vor.

2.1 Probabilistische Latent Semantische Analyse

Probabilistische Latent Semantische Analyse (PLSA) bezeichnet die Anwendung des Aspekt-Modells für die Analyse von zweigeteilten Daten. Eine Beispielanwendung ist die Analyse von Dokumentsammlungen, die Hofmann in [9] beschreibt. Eine Beobachtung besteht in diesem Fall, aus einem Wort-Dokument-Paar, durch das ein Auftreten eines Wortes in einem Dokument repräsentiert wird. Die aus der sogenannten Wort-Dokument-Matrix gelernten Parameter des Aspekt-Modells ermöglichen eine semantische Analyse der in der Dokumentsammlung enthaltenen Themengebiete. Um das zu verstehen schaue man sich nochmal die Erläuterungen aus Kapitel 1.2 an. Identifiziert man die Menge der Dokumente mit X und die Menge der Worte (das Vokabular) mit Y , dann ergibt sich die Wahrscheinlichkeit der Worte eines Dokumentes als konvexe Kombination der Wortvektoren $P(\cdot|a)$. Ein Vektor $P(\cdot|a)$ lässt sich somit auch als die für ein Thema repräsentative Menge an Worten auffassen. Der Vektor $P(\cdot|d)$ gibt dann, die für ein Dokument spezifische Themensammlung wieder. Mittels PLSA lassen sich eine Reihe interessanter Fragen beantworten, wie z.B.:

- Was sind die in einer Dokumentsammlung enthaltenden Themengebiete und durch welche Worte ist ein solches Themengebiet gekennzeichnet?

- Aus welchen Themengebieten setzt sich ein bestimmtes Dokument zusammen?
- Welches sind die Dokumente, die sich hauptsächlich mit einem der Themengebiete auseinandersetzen?

Hofmann vergleicht in [9] PLSA mit LSA anhand einer (bereits vorher von Hand) klassifizierten Dokumentsammlung.

2.2 Probabilistisches Latent Semantisches Indexieren

Probabilistisches Latent Semantisches Indexieren (PLSI) bezeichnet die Anwendung des Aspekt-Modells für die Indexierung von zweigeteilten Daten. In [10] wendet Hofmann PLSI für die Indexierung von Dokumentsammlungen an. PLSI benutzt die gelernten Parameter des Aspekt-Modells zum Aufbau einer vektoriellen Repräsentation der Dokumente. Die Nutzung der Parameter zur vektoriellen Repräsentation der Dokumente erfolgt mithilfe eines sogenannten Vektorraummodells. Ein Vektorraummodell besteht aus drei Komponenten:

1. Einer Funktion, welche ein Dokument auf einen (lokal) repräsentativen Vektor abbildet (lokale Wortgewichtung).
2. Einer Funktion, welche ein Dokument auf einen (global) repräsentativen Vektor abbildet (globale Wortgewichtung).
3. Ein Ähnlichkeitsmass, welches die Ähnlichkeit von Dokumenten anhand derer lokalen bzw. globalen Repräsentation beschreibt.

Eine weit verbreitete Variante ist das sogenannte $TF \times IDF$. Ein Dokument (bzw. eine Anfrage) wird bei diesem Verfahren durch einen Vektor beschrieben der zu jedem Wort einen Eintrag der Form $n(w, d) \times idf(w)$ enthält. Dabei bezeichnet $n(d, w)$ die Häufigkeit des Wortes w im Dokument d (lokale Gewichtung) und $idf(w)$ den Logarithmus der inversen Dokumenthäufigkeit (globale Gewichtung). Die Dokumenthäufigkeit eines Wortes ist die Anzahl der das Wort enthaltenden dividiert durch die Gesamtanzahl der Dokumente. Der Abstand zwischen zwei Dokumenten (bzw. einer Anfrage und einem Dokument) berechnet sich dann häufig nach dem sogenannten Cosinusmass (Dies ist der Winkel zwischen den entsprechenden Vektoren). Basierend auf diesem Verfahren und den gelernten Parametern der Dokumentsammlung schlägt Hofmann zwei Varianten von PLSI vor:

PLSI-U: Bei PLSI-U werden die empirischen Worthäufigkeiten $(n(w, d) / n(d))$ von $TF \times IDF$ durch die gelernten Wahrscheinlichkeiten $P(w, d)$ ersetzt. Die

Ähnlichkeit zwischen Dokumenten wird dann durch die Berechnung des Cosinusabstandes zwischen den Vektoren bestimmt. Die Anfrage erhält dabei eine Repräsentation durch die empirischen Worthäufigkeiten.

PLSI-Q: Bei PLSI-Q wird ein Dokument durch die gelernten Wahrscheinlichkeiten $P(a|d)$ repräsentiert. Für eine Anfrage schlägt Hofmann vor, diese Parameter durch einen eigenen EM-Lauf zu berechnen. Dabei werden dann die Wortwahrscheinlichkeiten $P(w|a)$ als konstant angenommen. Hofmann lässt offen, inwieweit bei diesem Verfahren globale Maße (wie z.B. $idf(w)$) berücksichtigt werden können. Als eine Möglichkeit schlägt er vor, $\sum_w P(w|a) idf(w)$ als globales Maß zu verwenden.

Eine Erweiterung dieser Verfahren kann durch die Kombination von verschiedenen Modellen erreicht werden. Dazu variiert man die Anzahl der Komponenten (Klassen) und errechnet die Ähnlichkeit über die zugehörigen Modelle. Hofmann bezeichnet die sich daraus ergebenden erweiterten Varianten als PLSI-U* bzw. PLSI-Q*.

2.3 Probabilistische Hypertext Induzierte Themen Selektion

Probabilistische Hypertext Induzierte Themen Selektion (PHITS) (siehe [5]) ist eine Anwendung von PLSA auf verlinkte Dokumentsammlungen. Anstelle der in einem Dokument enthaltenen Worte benutzt PHITS auf ein Dokument verweisende Links für die Analyse der Dokumentsammlung. Über das von PHITS gelernte Modell lassen sich dann die folgenden Fragen beantworten:

- Was sind die in einer Dokumentsammlung enthaltenen Themengebiete?
- Welche Dokumente werden innerhalb eines Themengebietes am häufigsten zitiert?
- Aus welchen Themengebieten besteht ein Dokument?

Kapitel 3

Anwendung auf verlinkte Dokumentsammlungen

3.1 Einführung

In einer verlinkten Dokumentsammlung besteht jedes Dokument aus einer Menge von im Dokument enthaltenen Worten und einer Menge von Verweisen auf andere Dokumente. Diese Mengen lassen sich als verschiedene Sichten auf ein Dokument auffassen.

Für die Herleitung verschiedener Methoden zur Kombination der Sichten mit dem Aspekt-Modell werden wir zunächst etwas Notation einführen. Im folgenden bezeichnen c und d Dokumente und w Worte über dem Vokabular der Dokumentsammlung. Die zu c , d bzw. w gehörenden Mengen bezeichnen wir mit C , D bzw. W . Die Menge der Beobachtungen für die im Dokument enthaltenen Worte ist dann die Menge aller Paare (w, d) für die gilt: Das Wort w ist $n(d, w)$ -mal im Dokument d enthalten. Analog ergibt sich die Menge der Beobachtungen für die zweite Sicht als Menge aller Paare (c, d) für die gilt: Aus dem Dokument d wird $l(c, d)$ -mal auf das Dokument c verwiesen. Für die folgenden Darstellungen benutzen wir das asymmetrische Modell. Weiterhin lassen wir die Wahrscheinlichkeiten $P(d)$ weg, da diese für die Indexierung mit PLSI-Q nicht benötigt werden. Das dies keinen Einfluss auf die Bestimmung der Parameter $P(w|a)$ bzw. $P(a|d)$ hat, sieht man leicht an der Gleichung für die logarithmierte Likelihood des asymmetrischen Modells 1.3.

3.2 Single-View

Eine sehr einfache Möglichkeit beide Views zu kombinieren besteht darin, die Beobachtungen beider Sichten in einer Sicht zusammenzufassen. Die logarithmierte

Likelihood ergibt sich für diesen Fall zu:

$$\begin{aligned} L_{\log} &= \sum_{y \in CUW} \sum_{d \in D} \ln(y, d) \sum_a \log(P(y|a)P(a|d)) \\ &= \sum_{w \in W} n(w, d) \sum_a \log(P(w|a)P(a|d)) + \sum_{c \in C} l(c, d) \sum_a \log(P(c|a)P(a|d)) \end{aligned}$$

Dabei ist $\ln(y, d) = n(y, d)$ falls $y \in W$ und $\ln(y, d) = l(y, d)$ falls $y \in C$. Das Zusammenführen der Attribute beider Sichten entspricht also einer Addition der zugehörigen logarithmierten Likelihoods. Als Updateschritte für den EM-Algorithmus erhält man (mit den in Kapitel 1 beschriebenen Methoden):

E-Schritt:

$$P(a|d, w) = \frac{P(w|a)P(a|d)}{\sum_{a'} P(w|a')P(a'|d)} \text{ bzw. } P(a|d, c) = \frac{P(c|a)P(a|d)}{\sum_{a'} P(c|a')P(a'|d)}$$

M-Schritt:

$$\begin{aligned} P(w|a) &= \frac{\sum_d n(d, w)P(a|dw)}{\sum_{d, w'} n(d, w')P(a|dw')} \text{ bzw. } P(c|a) = \frac{\sum_d l(d, c)P(a|dc)}{\sum_{d, c'} l(d, c')P(a|dc')} \\ P(a|d) &= \frac{\sum_w n(d, w)P(a|dw)}{\sum_{w, a'} n(d, w)P(a'|dw)} + \frac{\sum_c l(d, c)P(a|dc)}{\sum_{c, a'} l(d, c)P(a'|dc)} \end{aligned}$$

Diese Methode werden wir im folgenden als Single-View-Methode bezeichnen, da durch das Zusammenführen der beiden Sichten eine gemeinsame Sicht entsteht. Im folgenden Abschnitt werden wir diese Methode etwas verfeinern.

3.3 Single-View mit unterschiedlicher Gewichtung der Sichten

Schaut man sich die Likelihood des Single-View-Verfahrens aus Gleichung 3.1 an, so fällt auf, dass beide Sichten für ein Dokument jeweils mit der Anzahl der für dieses Dokument gemachten Beobachtungen eingehen. Geht man allerdings davon aus, dass die Informationen beider Views für jedes Dokument die gleiche Aussagekraft haben, ist dieses Verhalten unerwünscht. Um dieses Problem zu beheben kann man die absoluten Beobachtungsanzahlen $n(w, d)$ bzw. $l(c, d)$ für jedes Dokument und jede Sicht normieren. Dazu teilt man $n(w, d)$ bzw. $l(c, d)$ durch $n(d)$ bzw. $l(d)$. Daraus ergibt sich eine Likelihood, die für jedes Dokument die Informationen der verschiedenen Views in gleicher Weise einfließen lässt.

$$L_{\log} = \sum_{d,w} \frac{n(w,d)}{n(d)} \sum_a \log(P(w|a)P(a|d)) + \sum_{d,c} \frac{l(c,d)}{l(d)} \sum_a \log(P(c|a)P(a|d))$$

In der Praxis kommt es allerdings eher selten vor, dass alle beteiligten Sichten dieselbe Aussagekraft besitzen. Bei einer ungleichen Verteilung der Information auf die beteiligten Sichten bietet es sich an, auch deren Likelihoods unterschiedlich zu gewichten. Zu diesem Zweck kann man eine Sichtgewichtung α einführen und erhält:

$$L_{\log} = \alpha \sum_{d,w} \frac{n(w,d)}{n(d)} \sum_a \log(P(w|a)P(a|d)) + (1 - \alpha) \sum_{d,c} \frac{l(c,d)}{l(d)} \sum_a \log(P(c|a)P(a|d)) \quad (3.1)$$

Dies ist die von Thomas Hofmann und David Cohn in [6] eingeführte Likelihood. Im folgenden werden wir dieses Verfahren mit Weighted-Single-View bezeichnen. Als Updateschritte für den EM-Algorithmus erhält man (analog zu den in Kapitel 1 beschriebenen Methoden):

E-Schritt:

$$P(a|d,w) = \frac{P(w|a)P(a|d)}{\sum_{a'} P(w|a')P(a'|d)} \text{ bzw. } P(a|d,c) = \frac{P(c|a)P(a|d)}{\sum_{a'} P(c|a')P(a'|d)}$$

M-Schritt:

$$P(w|a) = \frac{\sum_d n(d,w)P(a|dw)}{\sum_{d,w'} n(d,w')P(a|dw')} \text{ bzw. } P(c|a) = \frac{\sum_d l(d,c)P(a|dc)}{\sum_{d,c'} l(d,c')P(a|dc')}$$

$$P(a|d) = \alpha \frac{\sum_w n(d,w)P(a|dw)}{\sum_{w,a'} n(d,w)P(a'|dw)} + (1 - \alpha) \frac{\sum_c l(d,c)P(a|dc)}{\sum_{c,a'} l(d,c)P(a'|dc')}$$

3.4 Multi-View

3.4.1 Einführung

Im Gegensatz zu den bisher vorgestellten Single-View-Algorithmen nutzen Multi-View-Algorithmen die Aufteilung der Informationen in die beteiligten Sichten explizit. Der erste Multi-View-Algorithmus wurde von Blum und Mitchell [4] für die Klassifikation von Webseiten entwickelt. Ein speziell auf den EM-Algorithmus zugeschnittener Multi-View-Algorithmus mit Namen co-EM wurde von Nigam und Ghani in [12] vorgestellt. Wir werden im nächsten Abschnitt zunächst den von Nigam und Ghani entwickelten Co-EM-Algorithmus vorstellen. Darauf aufbauend werden wir eine Variante von Co-EM für den Fall des Lernens in verlinkten Dokumentensammlungen mit dem Aspekt-Modell entwickeln.

3.4.2 Co-EM

Der Co-EM-Algorithmus ist eine Erweiterung des EM-Algorithmus für die Berechnung der Parameter, für den Fall, das sich die Attribute der Beispiele in mehrere Sichten unterteilen lassen. Die Idee von Co-EM ist es die Verteilung der versteckten Attribute (Posterior) zwischen den Sichten auszutauschen. Dazu berechnet man die Posterior der aktuellen Sicht aus den Parametern der jeweils anderen Sicht. Der Pseudocode für Co-EM ist in Abbildung 1 abgebildet.

Data : Die Häufigkeiten der Beispiele für jede Sicht
 init θ_1, θ_2 ;
while *Abbruchbedingung* **do**
 $P(z|y) = \text{ESchritt}(\theta_2)$;
 $\theta_1 = \text{MSchritt}(\theta_1, P(z|y))$;
 $P(z|y) = \text{ESchritt}(\theta_1)$;
 $\theta_2 = \text{MSchritt}(\theta_2, P(z|y))$;
Result : Der Mittelwert der Parameter des Modells der beiden Sichten ($\theta_1,$
 θ_2)

Algorithmus 1 : Co-EM

Dabei steht $P(z|y)$ für die Posterior, also die bedingte Verteilung der Werte der unbekannt Attribute z , gegeben die Werte der bekannte Attribute y . Die Funktionen E- bzw. M-Schritt sind die sich aus der Verteilung des jeweils verwendeten Modelles ergebenden Updateschritte für den E- bzw. M-Schritt des EM-Algorithmus bzgl. einer Sicht.

Bedingung für Co-EM ist es, dass sich die Posterior einer Sicht vollständig aus den Parametern der anderen Sicht berechnen lassen. Dies ist für die Parameter des Aspekt-Modells bei verlinkten Dokumentsammlungen nicht erfüllt, da die Mengen der Worte und Links disjunkt zueinander sind. Eine naheliegende Idee ist es daher, zumindest die gemeinsamen Parameter beider Views für die Berechnung der Posterior zu benutzen. Unter den Voraussetzungen aus Abschnitt 3.1 sind dies die Parameter $P(a|d)$. Der Pseudocode für die sich daraus ergebene Variante von Co-EM ist in Abbildung 2 gegeben. Die Funktionen E- bzw. M-Schritt ergeben sich dabei aus den Gleichungen 1.24-1.27 für das symmetrische bzw. 1.6.3-1.28 für das asymmetrische Modell.

Im Unterschied zu den bisher behandelten Single-View-Verfahren erfolgt die Angleichung der Parameter $P(a|d)$ hier implizit über einen Austausch der Parameter und nicht durch eine Mittelung der Parameter nach jeder Iteration.

Data : Die Häufigkeiten der Beispiele $n(w, d)$ und $l(c, d)$ für jede Sicht
init $P(a|d), P(w|a), P(c|a)$;
while *Abbruchbedingung* **do**
 $P(a|d, w) = \text{ESchritt}(P(a|d), P(w|a))$;
 $\theta_1 = \text{MSchritt}(P(a|d), P(w|a), P(a|d, w))$;
 $P(a|d, c) = \text{ESchritt}(P(a|d), P(c|a))$;
 $\theta_2 = \text{MSchritt}(P(a|d), P(c|a), P(a|d, c))$;
Result : $P(w|a), P(c|a), P(a|d)$

Algorithmus 2 : Co-EM für das Aspekt-Modell

Kapitel 4

Experimente

Wir wollen in diesem Abschnitt die zuletzt vorgestellten Verfahren Single-View, Weighted-Single-View und Co-EM für die Indexierung von Dokumenten verwenden und miteinander vergleichen. Für die Indexierung der Dokumente benutzen wir das in Abschnitt 2.2 vorgestellte Verfahren PLSI-Q. D.h. wir verwenden die Wahrscheinlichkeiten $P(a|d)$ als vektorielle Repräsentation der Dokumente. Als Ähnlichkeitsmass benutzen wir den Cosinusabstand. Zum Lernen des Aspekt-Modells benutzen wir die im letzten Kapitel beschriebenen Verfahren mit dem in Abschnitt 4 vorgestelltem Tempered-EM. Die für Tempered-EM benutzten Parameterwerte sind in Tabelle 4 wiedergegeben.

Parameter	Wert
likelihoodborder	10
μ	0,9
finaliterations	10

Abbildung 4.1: Parameterwerte für Tempered-EM

Um eine möglichst gute Indexierung der Beispiele zu erhalten, haben wir uns bei der Wahl der Anzahl der verborgenen Klassen (Aspekte) an [3] orientiert und die Anzahl der Aspekte relativ zur Dimension (Anzahl der Worte und Links) der jeweiligen Datensammlung gewählt. Der Anteil der verborgenen Klassen beträgt für den News- und WebKB-Datensatz jeweils 1%, für den Cora-Datensatz 2%.

Da wir über keinen Multi-View-Datensatz mit Beispielanfragen verfügen, benutzen wir als Anfragen die in der Dokumentsammlung enthaltenen Dokumente. Für jedes Dokument (jede Anfrage) und jedes $k \leq 50$ ermitteln wir mit PLSI-Q die k am besten zur Anfrage passenden Dokumente. Die Bewertung der Anfrageergebnisse erfolgt anhand der, in der Dokumentsammlung, enthaltenen Labels mit Precision und Recall. Die Precision gibt den Anteil der zur Anfrage passenden Dokumente (Dokumente mit dem gleichen Label wie die Anfrage), gegenüber allen

im Anfrageergebnis enthaltenden Dokumente an. Der Recall steht für den Anteil der im Anfrageergebnis enthaltenden passenden, zu allen zur Anfrage passenden Dokumenten.

Die Werte für Precision und Recall haben wir für festes k jeweils über alle in der Dokumentsammlung enthaltenden Dokumente gemittelt. Die Diagramme 4.2-4.5 geben die Recall/Precisionkurven für die verwendeten Verfahren und Dokumentensammlungen wieder. Für jeden Datensatz haben wir zwei Diagramme erstellt. Das jeweils erste Diagramm gibt die Recall/Precision-Kurven für das Weighted-Single-View-Verfahren und verschiedene Gewichtungen der beteiligten Sichten α wieder. Im zweiten Diagramm sind die Precision/Recall-Kurven für Co-EM, Single-View (SV) sowie Weighted-Single-View mit $\alpha = 0,5$ (hof) dargestellt.

Für den Vergleich der Verfahren benutzen wir drei sehr unterschiedliche Datensammlungen (WebKB, Cora und News2x2), die wir in den folgenden Abschnitten vorstellen werden.

4.1 WebKB

Der WebKB-Datensatz [1] besteht aus ca. 6000 verschiedenen Universitätsseiten. Die Seiten wurden von Hand in verschiedene Kategorien unterteilt (studentische Seiten, Kursseiten, Institutsseiten ...). Für jede Seite sind in dem Datensatz zwei verschiedene Arten von Informationen enthalten:

1. Die in der Seite enthaltenden Worte.
2. Alle Worte von auf diese Seite verweisender Links.

In unseren Experimenten stellen 1. und 2. jeweils eine Sicht dar. Dabei ist zu beachten, dass die Gleichheit von Worten in verschiedenen Sichten keine Rolle mehr spielt. Jedes Wort wird also zusätzlich durch die es enthaltende Sicht gekennzeichnet. Von den insgesamt 6 Klassen haben wir 2 Klassen (ca. 1000 Seiten) ausgewählt.

4.2 Cora

Der Cora-Datensatz [11] besteht aus den Referenzen und Abstracts von ca. 34000 Dokumenten der Computerwissenschaften. Jedes Dokument besteht aus den Worten aus dem Abstract und den sich auf das Dokument beziehenden Dokumenten. Auch hier haben wir die Sichten entsprechend dieser natürlichen Aufteilung gewählt. Der von uns gewählte Ausschnitt besteht aus zwei Klassen und ca. 2000 Dokumenten.

4.3 News2x2

Grundlage für den News2x2 Datensatz bildet der 20 Newsgroup Datensatz. Dieser besteht aus den Beiträgen von 20 verschiedenen Newsgroups. Für den News2x2 Datensatz wurden insgesamt 4 Newsgroups ausgewählt und in zwei Gruppen bestehend aus jeweils 2 Newsgroups unterteilt. In jeder Gruppe wurden nun jeweils zwei zufällig gewählte Beiträge (aus verschiedenen Newsgroups) aneinandergehängt. Jedes Beispiel besteht somit aus zwei Beiträgen unterschiedlicher Newsgroups.

4.4 Auswertung

Wie man an den Diagrammen 4.3, 4.5 und 4.7 erkennt, schneiden die drei Verfahren je nach verwendeter Dokumentensammlung sehr unterschiedlich ab. Die Diagramme 4.2, 4.4 sowie 4.6 geben jeweils Aufschluss über die Qualität der beteiligten Sichten, da für $\alpha = 0$ bzw. $\alpha = 1$ jeweils nur die Likelihood einer Sicht maximiert wird.

Der News2x2-Datensatz ist der einzige Datensatz mit annähernd ausgeglichenen Sichten (siehe Abbildung 4.2). Da die beteiligten Sichten hier auch annähernd dieselbe Dimension haben, eignet sich dieser Datensatz am besten für den Vergleich der drei Verfahren. Wie man sieht schneiden hier die beiden Verfahren Co-EM und Weighted-Single-View in etwa gleich und bedeutend besser, als das reine Single-View-Verfahren ab. Dies ist vor allem darauf zurückzuführen, dass beide Verfahren die Informationen in den beteiligten Sichten zu gleichen Teilen nutzen.

Beim Cora-Datensatz schneiden alle Verfahren in etwa gleich ab. Am besten ist hier das Lernen auf der Sicht mit den Abstracts der Dokumente $\alpha = 1$. Da die bessere Sicht (die Abstracts der Dokumente) weniger Dimensionen hat, schneidet das reine Single-View-Verfahren etwas schlechter ab als Co-EM und Weighted-Single-View.

Beim WebKB-Datensatz schneidet das einfache Single-View-Verfahren bedeutend besser ab als Co-EM und Weighted-Single-View. Im Diagramm 4.7 ist zusätzlich noch die Recall/Precision-Kurve für Weighted-Single-View und $\alpha = 1$ eingezeichnet. Wie beim Cora-Datensatz ist auch hier eine Sicht bedeutend besser für die Indexierung geeignet. Im Gegensatz zum Cora-Datensatz hat diese Sicht jedoch mehr Dimensionen. Da beim reinen Single-View-Verfahren die Sichten mit der Anzahl der Dimensionen gewichtet werden, schneidet hier das reine Single-View-Verfahren bedeutend besser ab als Co-EM und Weighted-Single-View.

Zusammenfassend kann man sagen, dass die Effektivität der Verfahren sehr von der Qualität der verwendeten Sichten abhängt. Haben beide Sichten in etwa die gleiche Qualität, so sind Verfahren wie Co-EM und Weighted-Single-View (für $\alpha = 0,5$) gegenüber dem reinen Single-View-Verfahren sicherlich im Vorteil. Ist dies

nicht der Fall, so ist es oft besser entweder nur die bessere Sicht zu nehmen, oder wenn möglich die Sichten entsprechend zu gewichten (wie bei Weighted-Single-View). Vor der Indexierung einer verlinkten Dokumentsammlung sollte man sich auf jeden Fall zuerst Gedanken machen, welche Informationen sich sinnvoll für die Indexierung der Daten mit PLSI-Q verwenden lassen.

4.5 Schlussfolgerungen

Wir haben in dieser Studienarbeit die Eignung verschiedener Methoden zum Lernen eines Aspekt-Modelles anhand der Indexierung von Dokumenten für den Fall untersucht, das sich die Attribute der Dokumente in zwei voneinander disjunkte Mengen unterteilen lassen. Dabei interessierte uns vor allem, ob und wie sich eine solche Aufteilung der Attribute in Sichten nutzen lässt, um das Ergebnis der Indexierung zu verbessern.

Für den Vergleich der drei Methoden, haben wir auf drei sehr unterschiedliche Datensammlungen zurückgegriffen. Die von uns untersuchten Methoden sind:

Single-View Das Single-View-Verfahren ignoriert die Aufteilung der Attribute in Sichten.

Weighted-Single-View Beim Weighted-Single-View-Verfahren werden die Informationen der beiden Sichten über einen Faktor α gewichtet. Für $\alpha = 0,5$ ergibt sich eine Gleichgewichtung der beiden Sichten.

Co-EM Das Co-EM-Verfahren nutzt die Informationen in beiden Sichten gleichermaßen. Co-EM benutzt allerdings einen anderen Mechanismus als Weighted-Single-View zum Angleichen der Parameter der beiden Sichten.

Bei unseren Experimenten konnten wir feststellen, dass Co-EM und Weighted-Single-View mit $\alpha = 0,5$ durchgängig sehr ähnliche Ergebnisse erzielten. Dies ist darauf zurückzuführen, dass diese Verfahren die Informationen beider Sichten in gleicher Weise berücksichtigen. Die Ergebnisse von Co-EM und Weighted-Single-View mit $\alpha = 0,5$ im Vergleich zu Single-View bzw. Weighted-Single-View mit unterschiedlichen α -Werten, hängen jedoch sehr stark von der Qualität der einzelnen Sichten der zugrundeliegenden Datensammlung ab. Weighted-Single-View hat sich hier als das robusteste Verfahren erwiesen, da die Qualität der Sichten explizit über den Parameter α berücksichtigt werden kann. Allerdings lässt sich die Qualität der Sichten für die jeweilige Aufgabe in den meisten Fällen im vorhinein nicht exakt bestimmen.

Zusammenfassend kann man sagen, das sich eine Aufteilung der Attribute in Sichten nur dann sinnvoll zum Indexieren von Dokumentsammlungen mit dem

Aspekt-Modell nutzen lässt, wenn die beteiligten Sichten in etwa die gleiche Qualität aufweisen. Ansonsten kann es sogar manchmal nützlich sein nur eine der beiden Sichten bzw. beide Sichten in Ihrer Gesamtheit zum Lernen heranzuziehen. Wenn sich die Qualität der Sichten im vorhinein gut abschätzen lässt, kann auf das Weighted-Single-View-Verfahren mit einem entsprechenden Wert für α zurückgegriffen werden.

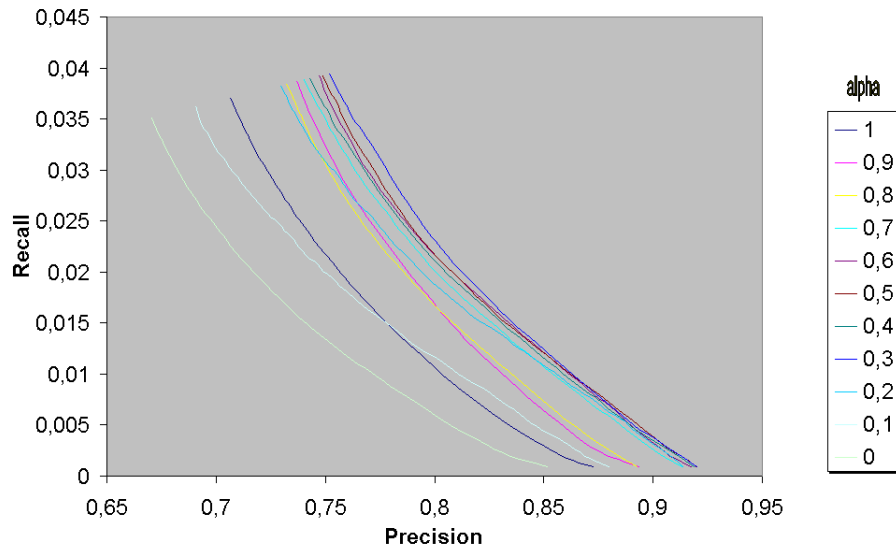


Abbildung 4.2: Weighted-Single-View für den Newsdatensatz

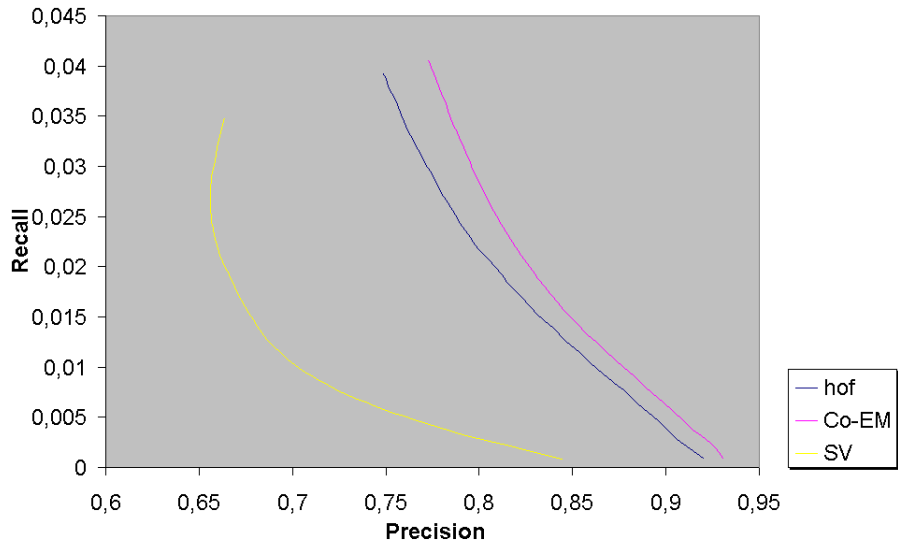


Abbildung 4.3: Vergleich der 3 Verfahren für den Newsdatensatz

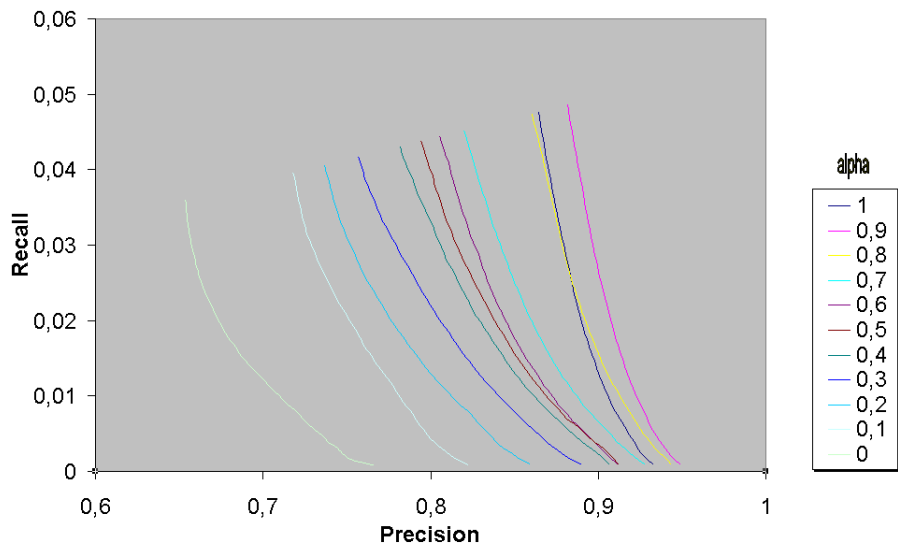


Abbildung 4.4: Weighted-Single-View für den Coradatensatz

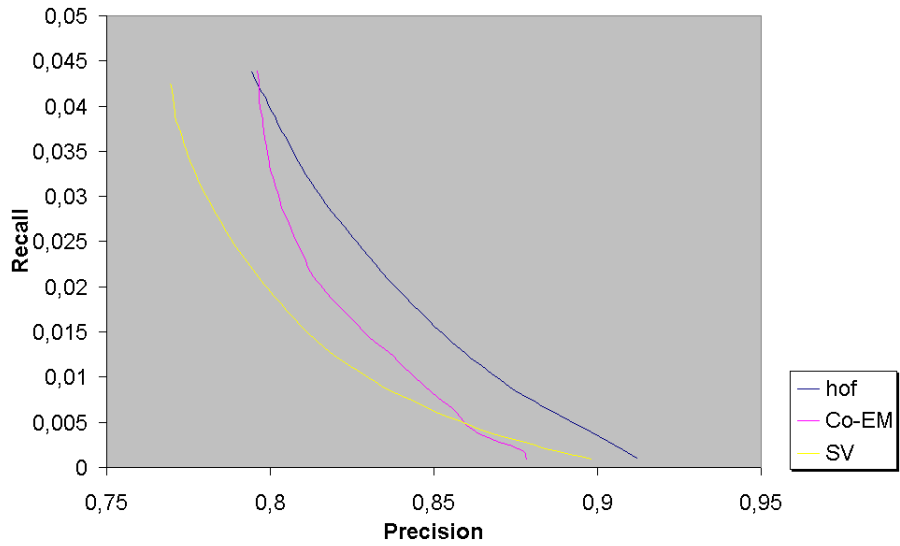


Abbildung 4.5: Vergleich der 3 Verfahren für den Coradatenatz

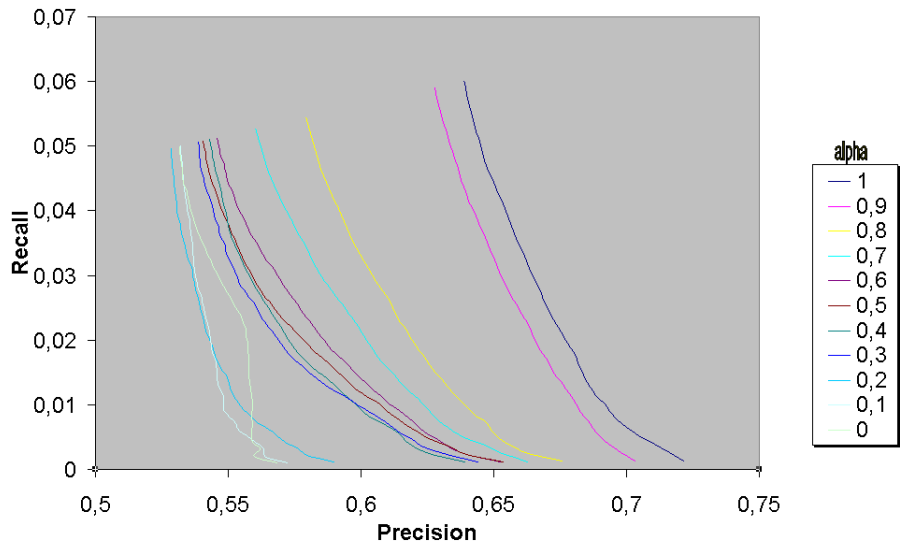


Abbildung 4.6: Weighted-Single-View für den WebKB-Datensatz

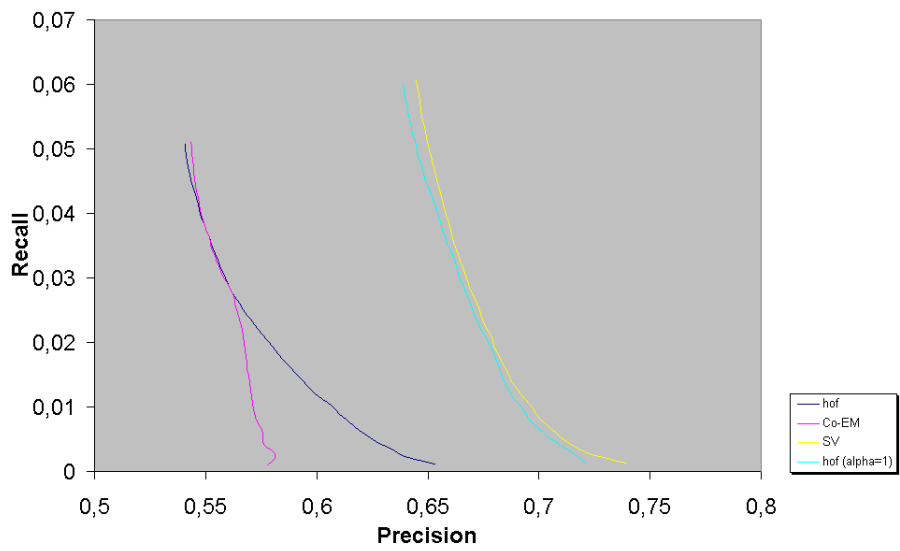


Abbildung 4.7: Vergleich der 3 Verfahren für den WebKB-Datensatz

Literaturverzeichnis

- [1] Webkb siehe <http://www.cs.cmu.edu/webkb/>.
- [2] J. A. Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technischer Bericht.
- [3] D. Blei, A. Ng und M. Jordan. Latent dirichlet allocation, 2002.
- [4] Avrim Blum und Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory*, Morgan Kaufmann Publishers. 1998.
- [5] David Cohn und Huan Chang. Learning to probabilistically identify authoritative documents. In *Proc. 17th International Conf. on Machine Learning*, Seiten 167–174. Morgan Kaufmann, San Francisco, CA, 2000.
- [6] David Cohn und Thomas Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *Neural Information Processing Systems 13*. 2001.
- [7] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas und Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
- [8] F. Dellaert. The expectation maximization algorithm. Technischer Bericht, College of Computing, Georgia Institute of Technology, 2002.
- [9] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI'99*. Stockholm, 1999.
- [10] Thomas Hofmann. Probabilistic Latent Semantic Indexing. In *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, Seiten 50–57. Berkeley, California, August 1999.
- [11] A. McCallum, K. Nigam, J. Rennie und K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval Journal*, 2000.

- [12] Kamal Nigam und Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM*, Seiten 86–93. 2000.
- [13] W. K. Seiler. Skriptum zur vorlesung höhere mathematik ii an der universität mannheim.
- [14] M. I. Jordan T. Hofmann, J. Puzicha. Unsupervised learning from dyadic data. In *Advances in Neural Information Processing Systems, volume 11*.