Musterlösung zur Wiederholungsklausur
Vorlesung Statistische Methoden in der Sprachverarbeitung
Sommersemester 2020

Da es sich um eine Open-Book-Klausur handelte, lagen den Studenten die Vorlesungsfolien mit allen Formeln vor.

Aufgabe 1) Wie berechnen Sie bei einem Naive-Bayes-Modell die gemeinsame Wahrscheinlichkeit der Klasse Sport und der Wörter Bayern, schlägt, Manchester?

Antwort:

$p(Sport, Bayern, schlaegt, Manchester) = p(Sport) \cdot p(Bayern|Sport) \cdot p(schlaegt|Sport) \cdot p(Manchester|Sport)$

Aufgabe 2) Wie berechnen Sie bei einem Markowmodell 2. Ordnung (Trigramm-Modell) über Buchstaben die Wahrscheinlichkeit der Buchstabenfolge CIS?

Antwort:

$p(C, I, S) = p(C|\langle s\rangle, \langle s\rangle) \cdot p(I|\langle s\rangle, C) \cdot p(S|C, I) \cdot p(\langle s\rangle|I, S)$

$\langle s\rangle$ ist hier das Grenzsymbol.

Aufgabe 3) Wie berechnen Sie bei einem Hidden-Markowmodell 1. Ordnung die gemeinsame Wahrscheinlichkeit der Wortfolge Bayern, schlägt, Manchester und der Tagfolge NE, VVFIN, NE?

Antwort:

$p(Bayern, schlaegt, Manchester, NE, VVFIN, NE) =$
$p(NE|\langle s\rangle) \cdot p(Bayern|NE) \cdot$
$p(VVFIN|NE) \cdot p(schlaegt|VVFIN) \cdot$
$p(NE|VVFIN) \cdot p(Manchester|NE) \cdot$
$p(\langle s\rangle|NE) \cdot p(\epsilon|\langle s\rangle)$

$\langle s\rangle$ ist hier das Grenzsymbol und ϵ das Grenztoken. $p(\epsilon|\langle s\rangle)$ ist immer 1 und kann weggelassen werden.

Aufgabe 4) Wie berechnen Sie bei einem Linear Chain CRF die bedingte Wahrscheinlichkeit der Tagfolge NE, VVFIN, NE gegeben die Wortfolge Bayern, schlägt, Manchester, wenn das CRF-Modell nur Tag-Tag-Paare und Wort-Tag-Paare als Merkmale verwendet? Sie können w(NE,VVFIN) für das Gewicht des Merkmales des Tagpaares (NE, VVFIN) schreiben und w(Bayern,NE) für das Gewicht des Merkmales des Wort-Tag-Paares (Bayern,NE). Erklären Sie außerdem, wie hier die Normalisierungskonstante Z berechnet wird.

Antwort:

$p(NE, VVFIN, NE|Bayern, schlaegt, Manchester) = \frac{1}{Z} \cdot exp(w(\langle s\rangle, NE) + w(Bayern, NE) +$
$w(NE, VVFIN) + w(schlaegt, VVFIN) + w(VVFIN, NE) + w(Manchester, NE) + w(NE, \langle s\rangle)$

Die Normalisierunskonstante Z erhält man, indem man den Exponential-Ausdruck für alle möglichen Tagfolgen berechnet und aufsummiert.

Aufgabe 5) Wie berechnen Sie mit interpoliertem Backoff die geglättete Wahrscheinlichkeit $p(Manchester|Bayern, schlaegt)$ aus den Häufigkeiten der Wort-Unigramme (bspw. f(Manchester)), -Bigramme (bspw. f(schlägt, Manchester)) und -Trigramme (bspw. f(Bayern, schlägt, Manchester)), sowie den Discount-Werten (δ_1 etc.) und den Backoff-Faktoren (α(Bayern, schlägt) etc.)? Bei den Unigrammen sollen die Wahrscheinlichkeiten nicht mehr geglättet werden.

Antwort:

$p(Manchester|Bayern, schlaegt) =$

$\frac{f(Bayern,schlaegt,Manchester)-\delta_2}{\sum_w f(Bayern,schlaegt,w)} + \alpha(Bayern, schlaegt) \cdot (\frac{f(schlaegt,Manchester)-\delta_1}{\sum_w f(schlaegt,w)} + \alpha(schlaegt) \cdot \frac{f(Manchester)}{\sum_w f(w)})$

Aufgabe 6) Erklären Sie in Worten, wie die Discount-Werte δ_i und die Backoff-Faktoren $\alpha(...)$ bei der interpolierten Backoff-Glättung berechnet werden.

Antwort: (Die zugehörige Formel konnte nachgeschlagen werden.)

Zur Berechnung des Discountwertes für die Häufigkeit von n-Grammen zählt man zunächst, wieviele n-Gramm-Types genau einmal (N_1) bzw. genau zweimal (N_2) aufgetaucht sind. Dann teilt man N_1 durch die Summe aus N_1 und dem Doppelten von N_2.

Zur Berechnung des Backoff-Faktors $\alpha(C)$ iteriert man über alle möglichen Vorhersagewerte w, subtrahiert den Discount von der Häufigkeit des Paares C,w und teilt das Ergebnis durch die Häufigkeit des Kontextes C. Die erhaltenen Ergebniswerte werden über alle Vorhersagen w summiert und dann von 1 subtrahiert.

Aufgabe 7) Welches Problem gibt es bei der normalen Backoff-Glättung? Wie vermeidet das Kneser-Ney-Verfahren dieses Problem? Worin unterscheidet sich die Berechnung der Backoff-Verteilungen bei den beiden Verfahren?

Antwort:

Ein Sprachmodell, das auf einem Korpus der Zeitung *Los Angeles Times* trainiert wird, weist dem Wort *Angeles* dieselbe Backoff-Wahrscheinlichkeit zu wie dem Wort *Los*, falls die beiden Wörter gleich oft im Korpus auftauchen. Tatsächlich sollte aber die Backoff-Wahrscheinlichkeit von *Angeles* kleiner sein, weil es fast nur nach *Los* auftaucht.

Das Problem wird vermieden, indem die Backoff-Wahrscheinlichkeitsverteilung auf Basis von Type- statt Token-Häufigkeiten berechnet werden. Man zählt nicht, wie häufig das Wort aufgetaucht ist, sondern nach wievielen unterschiedlichen Wörtern es aufgetaucht ist.

Aufgabe 8) Wie schätzen Sie die ungeglättete Wahrscheinlichkeit $p(Haus|NN)$ aus den gemeinsamen Wort-Tag-Häufigkeiten $f(w, t)$ (wobei w ein Wort und t ein Tag ist)?

Antwort:

$p(Haus|NN] = \frac{f(Haus,NN)}{\sum_w f(w,NN)}$

Aufgabe 9) Angenommen Sie vergleichen Ihren neu entwickelten Spam-Klassifizierer *Spammy* mit einem Baseline-Spam-Klassifizierer und erhalten folgende Ergebnisse:

Goldstandard	Baseline	Spammy	Häufigkeit
Spam	Spam	Spam	57
Spam	Spam	NoSpam	5
Spam	NoSpam	Spam	7
Spam	NoSpam	NoSpam	3
NoSpam	Spam	Spam	2
NoSpam	Spam	NoSpam	6
NoSpam	NoSpam	Spam	2
NoSpam	NoSpam	NoSpam	154

Hier gab es bspw. 57 Emails, die sowohl vom Baseline-Tagger als auch von Spammy korrekt als Spam klassifiziert wurden.

Sagen Sie so genau wie möglich, wie Sie hier mit dem **Vorzeichentest** berechnen, ob Spammy signifikant besser als der Baseline-Klassifikator ist.

Antwort:

Man zählt zunächst, wieviele Emails Spammy richtig klassifiziert und der andere Tagger falsch. Das sind 13. Dann zählt man, wieviele Emails Spammy falsch klassifiziert und der andere Tagger richtig. Das sind 7. Nur diese 20 Beispiele sind für den Vorzeichentest relevant. Die Nullhypothese besagt, dass Spammy nicht besser als der andere Tagger ist. Die Wahrscheinlichkeit, dass Spammy ein beliebiges der 20 Beispiele korrekt klassifiziert hat, ist daher unter Annahme der Nullhypothese maximal 0.5. Die Wahrscheinlichkeit, dass man bei Gültigkeit der Nullhypothese das beobachtete Ergebnis (Spammy 13 Mal korrekt) oder ein noch unwahrscheinlicheres Ergebnis (≥ 13) bekommt, ist durch die Summe $\sum_{i=13}^{20} b(i, 0.5, 20)$ gegeben, wobei $b(r, p, n)$ die Binomialverteilung mit Wahrscheinlichkeit p und Stichprobengröße n ist. Wenn diese Summe kleiner als 0.05 ist, kann die Nullhypothese zurückgewiesen werden. Man sagt dann: Spammy hat eine signifikant höhere Genauigkeit.

Aufgabe 10) Mit dem Forward-/Backward-Algorithmus können Sie die Wahrscheinlichkeit jedes möglichen Tags bei jedem Eingabewort berechnen. Sie wissen dann beispielsweise, dass das Tag NN beim 3. Wort eines gegebenen Satzes die Wahrscheinlichkeit 0,47 hat. Wie könnten Sie auf Basis dieser Wahrscheinlichkeiten (sinnvoll) eine beste Tagfolge für den Satz definieren? (Die Lösung kennen Sie nicht aus der Vorlesung.) Bekommen Sie mit dieser Methode dieselbe Tagfolge wie mit dem Viterbi-Algorithmus? Versuchen Sie, Ihre Antwort zu begründen.

Antwort:

Man kann einen Satz taggen, indem man an jeder Wortposition das Tag mit der höchsten Aposteriori-Wahrscheinlichkeit auswählt.

Das Ergebnis ist nicht dasselbe wie beim Viterbi-Algorithmus, weil die Rechnungen nicht äquivalent sind. Angenommen ein HMM hat die Wahrscheinlichkeiten:

	A	B	$\langle s \rangle$	a	ϵ
$p(.\|A)$	0.9	0	0.1	1	0
$p(.\|B)$	0	0.9	0.1	1	0
$p(.\|\langle s \rangle)$	0.5	0.5	0	0	1

Dieses HMM generiert eine Folge von a's (plus ein Endesymbol), die entweder alle mit A oder alle mit B getaggt sind. Der Viterbi-Algorithmus kann nur eine dieser beiden Tagfolgen ausgeben. Der Forward-Backward-Tagger könnte dagegen eine beliebige Tagfolge ausgeben, da die Aposteriori-Wahrscheinlichkeit des Tags A (analog B) an jeder Position 0.5 beträgt und damit maximal ist.

Aufgabe 11) Bei der Parser-Evaluierung verwendet man üblicherweise sowohl Precision als auch Recall der ausgegebenen Konstituenten, da jede dieser Metriken alleine ausgetrickst werden kann. Angenommen

das Startsymbol der Grammatik lautet S. Wie könnten Sie bei beliebigen Sätzen eine Precision von 1 erzielen, ohne die Sätze wirklich zu parsen? Wie müssen die ausgegebenen Parsebäume aussehen? Wie könnten Sie bei beliebigen Sätzen einen Recall von 1 erzielen, ohne die Sätze wirklich zu parsen? Wie müssen die ausgegebenen Parsebäume aussehen?

Antwort:

Eine Precision von 1 kann erzielt werden, indem man einen Parsebaum ausgibt, der nur den Wurzelknoten S enthält und alle Terminalsymbole als Tochterknoten hat. Der S-Knoten ist immer richtig.

Ein Recall von 1 kann erzielt werden, indem man einen Parsebaum ausgibt, der alle möglichen Konstituenten enthält. Dazu würden auch überlappende Konstituenten gehören, weshalb es einen solchen Parsebaum nicht in jedem Fall geben kann.

Aufgabe 12) Welche Unterschiede gibt es zwischen generativen und diskriminativen Verfahren? Welche generativen und welche diskriminativen Modelle kennen Sie aus der Vorlesung?

Antwort:

Generative Modelle definieren eine gemeinsame Wahrscheinlichkeit von Klasse und klassifiziertem Objekt. Sie modellieren einen generativen Prozess, der das Paar Schritt für Schritt generiert. Jedem Schritt entspricht ein Wahrscheinlichkeitsparameter des Modelles. Die Wahrscheinlichkeiten werden zur Gesamtwahrscheinlichkeit des Paares multipliziert. Die Wahrscheinlichkeitsparameter des Modelles werden aus Trainingsdatenhäufigkeiten geschätzt.

In der Vorlesung wurden folgende generative Modelle behandelt: Naive Bayes, Markov-Modell, Hidden-Markov-Modell, PCFG

Diskriminative Modelle definieren eine bedingte Wahrscheinlichkeit der Klasse gegeben das klassifizierte Objekt. Dazu werden Merkmalsfunktionen definiert, welche das Objekt auf eine Zahl abbilden. Für jede Merkmalsfunktion umfasst das diskriminative Modell ein trainierbares Gewicht. Die Mermalswerte eines zu klassifizierenden Objektes werden mit den jeweiligen Gewichten multipliziert, aufsummiert und mit Softmax in eine Wahrscheinlichkeit transformiert. Die Gewichte werden iterativ (beispielsweise mit Gradient Ascent) trainiert.

In der Vorlesung wurden nur CRFs als diskriminative Modelle behandelt.