5. Übung zur Vorlesung "NLP – Analyse des Wissensrohstoffes Text" im Sommersemester 2008
– mit Musterlösungen –

Dr. Andreas Hotho, Dipl.-Inform. Dominik Benz, Wi.-Inf. Beate Krause

11. Juni 2008

1 Probabilistic Context Free Grammars (PCFG)

1.1 Grundlagen

1. Grenzen Sie den Vorgang des Parsing von PoS-Tagging ab. Welche sprachlichen Eigenschaften lassen sich mit Hilfe von Parsing analysieren?

   Lösungsvorschlag

   Beim PoS-Tagging werden den einzelnen Wörtern eines Satzes S ihre jeweilige Wortart zugeordnet. Ein Parser strukturiert einen Satz, indem er den Satz mit Hilfe von Regeln einer vorgegebenen Grammatik zerlegt. Sprachliche Eigenschaften, die durch die Baumstruktur erfasst werden können:

   a) Struktur und Semantik: Sätze bestehen aus einer semantischen Struktur, die kritisch für das Verarbeiten von natürlicher Sprache ist. Beispiel: Foxes eat rabbits vs. Rabbits eat foxes.
   b) Übereinstimmung: In vielen Sprachen wird die Übereinstimmung zwischen Subjekt, Verb usw. verlangt.
   c) Rekursion: Viele Sätze haben eine rekursive Struktur, die sich durch das Parsing darstellen lässt. Beispiel: Foxes that eat chickens that eat corn eat rabbits.
   d) Abhängigkeiten über lange Distanzen: Obiges Beispiel: Es kann erkannt werden, dass "eat rabbits" von "Foxes" abhängig ist.

2. Entwickeln Sie eine probabilistische kontextfreie Grammatik, welche für den Satz "People saw the man on the hill" mindestens zwei (möglichst linguistisch sinnvolle) Analysen liefert. Nutzen Sie die Chomsky Normalform.

Lösungsvorschlag

Die Chomsky Normalform beschreibt zwei Arten von Regeln:

a) $X -> YZ$

b) $X -> w$ wobei w ein Wort ist.

Folgende Regeln könnten dann beispielsweise aufgestellt werden.

```
S -> NP VP 1.0
VP -> V NP 0.6
VP -> VP NP 0.4
NP -> NP PP 0.35
NP -> Det NP 0.35
PP -> P NP 1.0
NP -> people 0.1
V -> saw 1.0
Det -> the 1.0
NP -> man 0.1
P -> on 1.0
NP -> hill 0.1
```

Für eine probabilistische, kontextfreie Grammatik muss gelten: $sum_{rs}P(N^j -> N^r N^s) + sum_k P(N^j -> w^k) = 1$

1.2 Wahrscheinlichkeiten eines Satzes

1. Welche Algorithmen lassen sich verwenden, um die Wahrscheinlichkeit für einen Satz w_{1m} gegeben die Grammatik G zu errechnen?

Lösungsvorschlag

Der naive Ansatz wäre, jeden einzelnen, möglichen Baum für einen Satz zu generieren und die Wahrscheinlichkeiten der einzelnen Bäume zu addieren. Ein effizienterer Ansatz besteht in der Verwendung der beiden Algorithmen zur Errechnung der "Inside" und "Outside" Wahrscheinlichkeiten.

2. Berechnen Sie mit den Wahrscheinlichkeiten der S.384 Tabelle 11.2 die Outside Wahrscheinlichkeiten für den Satz "astronomers saw stars with ears".

   a) Erstellen Sie eine passende Tabelle.

   b) Beginnen Sie mit der Startwahrscheinlichkeit im rechten, oberen Feld.

   c) Arbeiten Sie sich diagonal weiter vor.

   d) Errechnen Sie $P(w_{1m}|G)$.

	1	2	3	4	5
1	$\alpha_{NP} = 0.015876$	-	-		$\alpha_s = 1$
2		$\alpha_V = 0.0015876$	$\alpha_{VP} = 0.0054$		$\alpha_{VP} = 0.1$
3			$\alpha_{NP} = 0.00882$	-	$\alpha_{NP} = 0.07$
4				$\alpha_P = 0.0015876$	$\alpha_{PP} = 0.00882$
5					$\alpha_{NP} = 0.00882$
	astronomers	saw	stars	with	ears

Lösungsvorschlag

Rechenschritte:

$\alpha_{VP}(2,5) = \alpha_S(1,5)P(N^S -> N^{NP}N^{VP})\beta_{NP}(1,1) = 1.0 * 1.0 * 0.1 = 0.1$

$\alpha_{VP}(2,3) = \alpha_{VP}(2,5)P(N^{VP} -> N^{VP}N^{PP})\beta_{PP}(4,5) = 0.1 * 0.3 * 0.18 = 0.0054$

$\alpha_{PP}(4,5)$

$= \alpha_{VP}(2,5)P(N^{VP} -> N^{VP}N^{PP})\beta_{VP}(2,3)$

$+ \alpha_{NP}(3,5)P(N^{NP} -> N^{NP}N^{PP})\beta_{NP}(3,3)$

Um die Wahrscheinlichkeit für den Satz zu erhalten, kann man für jedes **beliebige** k errechnen: $\sum_j \alpha_j(k,k)P(N^j -> w_k)$. Beispielsweise gilt für $k = 1$: $0.1 * 0.015876 = 0.0015876$.

3. Wie kann der Inside Algorithmus verändert werden, um den wahrscheinlichsten Parsbaum für einen gegebenen Satz zu finden? Welche Werte würden sich in der Tabelle 11.3 ändern und wie?

Lösungsvorschlag

1. Initialisierung

Die Initialisierung bleibt die Gleiche: Jedem Wurzelknoten (die Diagonale in der Tabelle) wird die Wahrscheinlichkeit der unären Regel zugewiesen.

2. Induktion

Anstelle einer Speicherung der Wahrscheinlichkeiten aller möglichen Regeln, wird nur noch die wahrscheinlichste Regel gespeichert. In der Tabelle 11.2 würde sich $P(\beta_{VP}(2,5)$ ändern: $P(\beta_{VP}(2,5) = 0.009072$ Die Gesamtwahrscheinlichkeit für den besten Parsebaum ist dann $\beta_S = 0.9072^{-3}$.

Zusätzlich müssen die ausgewählten Regeln in jedem Schritt abgespeichert werden. Dadurch kann zum Schluss der Baum rekonstruiert werden.

1.3 Trainieren einer PCFG

1. Welche Informationen liegen für das Trainieren von PCFGs vor? Was genau umfasst der Prozess des Trainierens?

   Lösungsvorschlag

   Für das Lernen ist folgendes bekannt:

   a) Anzahl der Terminale und Nichtterminale
   b) Name des Startsymbols
   c) Menge der Regeln

   Das Trainieren einer Grammatik umfasst die Berechnung der Wahrscheinlichkeiten für die einzelnen Regeln in der Grammatik.

2. Benutzen Sie wieder das Beispiel aus dem Buch und die Outside Ergebnisse aus der vorherigen Aufgabe, um die Wahrscheinlichkeiten für die Regeln aus der ersten Spalte der Tabelle 11.2 neu zu bestimmen. Wenden Sie dabei den Inside-Outside Algorithmus unter der Berücksichtigung eines Satzes an. Eine Iteration genügt.

   Lösungsvorschlag

   $P(N^S -> N^{NP}N^{VP}) = \frac{0.0015876*1*0.1*0.015876}{1*0.0015876} = 0.0015876$
   $P(N^{PP} -> N^P N^{NP} = \frac{0.0082*1.0*1.0*0.18}{0.0082*0.18} = 1$
   $P(N^{VP} -> N^V N^{NP} = \frac{0.0054*0.7*1.0*0.18+0.1*0.7*1.0*0.01296}{0.0054*0.126+0.1*0.015876} = \frac{0.0015876}{0.002268} = 0.7$
   $P(N^{VP} -> N^{VP}N^{PP} = \frac{0.1*0.3*0.126*0.18}{0.0054*0.0126+0.1*0.015876} = 0.3$
   $P(N^P -> with) = \frac{1.0*0.0015876}{1.0*0.0015876} = 1$
   $P(N^v -> saw) = \frac{1.0*0.0015876}{1.0*0.0015876} = 1$

2 Praxisübung - Parsing (Abgabe: Mittwoch, 25. Juni 2008

Das NLTK bietet verschiedene Parser. In dieser Übung sollen die vorhanden Parser genutzt und erweitert werden.

1. Schreiben Sie ein Program, welches Sätze mit Hilfe des Recursive Descendent Parser und des Shift Reduce Parsers aus dem NLTK parsen kann.
2. Implementieren Sie selbst einen probabilistischen Left-Corner Parser.
3. Vergleichen Sie die Parser. Verwenden Sie dabei verschiedene Sätze und protokollieren Sie kurz, welche Hauptunterschiede Ihnen auffallen.