Übung 10

Statistischer Parser

Schreiben Sie ein Programm, welches eine Grammatik und ein Lexikon einliest und damit Sätze parst.

Die Grammatik enthält PCFG-Regeln der Form

```
1.0 S NP VP
0.2 VP VP PP
0.3 VP V NP
0.2 VP V
0.2 VP V PP
0.1 VP V NP PP
0.2 NP NP PP
0.5 NP DT N1
0.3 NP N1
0.3 N1 A N1
0.7 N1 N
1.0 PP P NP
```

wobei die erste Zeile die Regel S → NP VP mit ihrer Wahrscheinlichkeit enthält.

Das Lexikon enthält Einträge der Form

```
0.6 DT the
0.4 DT a
0.2 A old
0.3 A young
0.2 A big
0.3 A small
0.2 N man
0.3 N hill
0.2 N telescope
0.2 N girl
0.1 N saw
0.4 V saw
0.6 V slept
0.6 P on
0.4 P with
```

In der ersten Spalte steht jeweils die Wahrscheinlichkeit und in der zweiten die linke Seite der Regel.

Verwenden Sie den Left-Corner-Algorithmus zum Parsen und den Viterbi-Algorithmus, um den besten Parse zu berechnen. Beim Parsen des Satzes "the man slept" mit der obigen Grammatik trägt der Scanner zunächst die Punktregel (DT $\rightarrow$ the $\cdot$, 0, 1) in die Chart ein. Dann trägt der Predictor die partielle Konstituente (NP $\rightarrow$ DT $\cdot$ N1, 0, 1) ein. Im nächsten Schritt speichert der Scanner dann (N $\rightarrow$ man $\cdot$, 1, 2). Dann trägt der Predictor die Konstituente (N1 $\rightarrow$ N $\cdot$, 1, 2) ein. Rekursiv trägt der Predictor (NP $\rightarrow$ N1 $\cdot$, 1, 2) und (S $\rightarrow$ NP $\cdot$ VP, 1, 2) und (NP $\rightarrow$ NP $\cdot$ PP, 1, 2) ein. Der Completer trägt dann (NP $\rightarrow$ DT N1 $\cdot$, 0, 2) ein. Dann wird wieder der Predictor aktiv und so weiter.

Sie können bei der Implementierung folgendermaßen vorgehen: Sie verwenden als zentrale Datenstruktur zur Speicherung der Chart eine Liste von Dictionaries z.B. mit Namen *logvitprob*. Darin speichern Sie Informationen über – eventuell nur partiell – erkannte Konstituenten (Items) bestehend aus der linken und rechten Seite der Grammatikregel, der aktuellen Position des "Punktes" auf der rechten Seite der Regel, der Start- und Endposition der Konstituente und der Viterbi-Wahrscheinlichkeit.

Der folgende Python-Befehl speichert eine partielle S-Konstituente (S $\rightarrow$ NP $\cdot$ VP) mit der logarithmierten Wahrscheinlichkeit -75.38, von der die erste Tochterkonstituente NP, welche die ersten 5 Wörter überdeckt (Wortposition 0-5), bereits erkannt wurde (Punktposition 1):

logvitprob[5]['S', ('NP', 'VP'), 1, 0, 5] = -75.38

Schritte:

- Definieren Sie eine Klasse `Parser` und eine `init`-Methode, welche die Namen einer Grammatikdatei und einer Lexikondatei als Argument erhält, und die Methode `read_file` zweimal aufruft, um erst die Grammatik und dann das Lexikon in `self.gramrules` und `self.lexrules` einzulesen. Die Parser-Klasse soll die Grammatik, das Lexikon und die Chart als Attribute speichern.

- Schreiben Sie eine Methode `read_file`, welche die Grammatik (bzw. das Lexikon) einliest. Um die predict- und scan-Schritte zu vereinfachen, speichern Sie die Grammatik in einem Dictionary of Lists, das mit dem ersten Symbol auf der rechten Seite indiziert wird und die Liste der Regeln (mit den logarithmierten Wahrscheinlichkeiten) liefert, deren rechte Seite mit diesem Symbol beginnt:

  logruleprobs['DT'].append(('NP',('DT','N1'), log(0.2)))

  Das Dictionary geben Sie zurück.

- Schreiben Sie eine Methode `scan(i, word)`, welche das Wort `word` und seine Satzposition `i` als Argumente bekommt und die passenden lexikalischen Regeln in `self.lexrules` nachschlägt und die add-Funktion aufruft, um die Regeln in die Chart einzutragen, bspw. mit

  self.add(('N',('man',), 1, 1, 2), log(0.1))

falls “man” das zweite Wort im Satz war.

- Schreiben Sie eine Methode `predict(child, logprob)`, welche alle Regeln in die Chart einträgt, die auf der rechten Seite mit der Kategorie der Konstituente *child* beginnen. Bspw. wird self.add(('S',('NP','VP'), 1, 0, 2), p) aufgerufen, wenn eine NP von Position 0 bis 2 übergeben wurde und eine Regel S → NP VP existiert. Die Wahrscheinlichkeit p der neuen Punktregel bekommen Sie, indem Sie zur logarithmierten Viterbi-Wahrscheinlichkeit der NP die logarithmierte Wahrscheinlichkeit der Regel S → NP VP addieren.

- Schreiben Sie eine Methode `complete(child, logprob)`, die partiell erkannte Konstituenten vervollständigt, welche die Kategorie der child-Konstituente als Nächstes erwarten.

  Beispielsweise könnten Sie mit self.add(('S',('NP','VP'), 2, 0, 8), p) eine vollständig erkannte S-Konstituente eintragen, nachdem Sie mit einer VP von Position 2 bis 8 eine partielle S-Konstituente ('S',('NP','VP'), 1, 0, 2) vervollständigt haben. Die Wahrscheinlichkeit p ergibt sich als Summe der logarithmierten Wahrscheinlichkeiten der beiden Charteinträge, die kombiniert wurden.

- Schreiben Sie eine Methode `add(item, logprob)`, die von scan, predict und complete aufgerufen wird, um eine Punktregel item in die Chart einzutragen. Wenn die Punktregel bereits in der Chart existiert, wird die neue Regel nur eingetragen, wenn ihre Wahrscheinlichkeit größer ist. Wenn die neu eingetragene Regel den Punkt am Ende hat, ruft die add-Funktion noch die Funktionen predict und complete auf.

- Schreiben Sie eine Methode `parse(words)`, welche die Chart initialisiert und die scan-Funktion nacheinander für jedes Eingabewort aufruft. Die scan-Funktion ruft dann die add-Funktion auf, welche wiederum predict und complete aufruft usw. Nach der letzten scan-Operation ist der Satz geparst.

  Nun muss noch der beste Parsebaum ausgegeben werden. Dazu wird der Graph, der in logvitprob gespeichert ist, top-down durchlaufen. Dies wird vereinfacht, wenn die add-Operation ein zusätzliches Argument `child` erhält, welches den (vollständigen) Chart-Eintrag angibt, mit dem die predict- oder complete-Operation durchgeführt und der neue Charteintrag `item` erzeugt wurde. Im Falle einr Scan-Operation wird None als child übergeben. Die add-Funktion speichert `child` in der Datenstruktur `childitem`. Hier ein Beispiel:

  logvitprob[endpos][item] = logprob
  childitem[endpos][item] = child

  Hier wird die Punktregel *item* mit ihrer Wahrscheinlichkeit in die Chart eingetragen und ein Verweis auf die Tochter *child* in *childitem* gespeichert. (*item* und *child* sind beide Punktregeln.)

  Nach den scan-Operationen suchen Sie den wahrscheinlichsten Eintrag *item* in logvitprob[-1] mit lhs='S' und startpos=0. Dann rufen Sie die Methode *build_tree(item)* auf.

- Die Methode *build_tree(item)* erhält eine Punktregel *item* als Argument und gibt den Teilbaum für die Punktregel zurück. Je nachdem, ob *item* durch eine scan-, predict-, oder complete-Operation entstanden ist, ruft sich *build_tree* 0-mal, 1-mal (mit dem *childitem*) oder 2-mal (mit *childitem* und der Regel, die durch die complete-Operation vervollständigt wurde) rekursiv auf und bekommt jeweils einen Teilbaum zurück. Die Teile des Parsebaumes werden zusammengefügt und das Ergebnis wird zurückgegeben. Bei einer complete-Operation müssen Sie die vervollständigte Punktregel aus der aktuellen Punktregel und der Startposition von *childitem* rekonstruieren.

  Die Ausgabe des Parsebaums soll in Klammernotation erfolgen:

  (S (NP (DT the)(N1 (N man)))(VP (V slept)))