Example questions for Übung-Exam

You should try to solve it by your self and add your solution there.

We can solve some questions together in the tutorium.

$$p_{backoff}(w|C) = \frac{f^*(C,w)}{\sum_{w'} f^*(C,w')}$$

Was bedeutet diese Formel? Wo benutzen wie sie? Was brauchen wir um p_backoff zu berechnen?
Gib ein konkretes Beispiel an. (also was kann w and C sein?)

Was ist die Formel für die Berechnung von ein N-1 Gram Häufigkeit? (im Backoff Schritt). Nenne beide Variante (standard and kneser-ney)

Aufgabe 1) Welche Python-Datenstruktur eignet sich zur Repräsentation der bereits berechneten Häufigkeiten?

Hinweis: Es gibt hier mehrere Möglichkeiten. Wählen Sie eine aus, mit der Sie dann die folgenden Aufgaben lösen. (1 Punkt)

Aufgabe 2) Schreiben Sie eine Funktion mit dem Namen **discount**, welche die N-Gramm-Häufigkeiten als Argument bekommt und den Discount zurückgibt.

Hinweis: Den Discount erhalten Sie, indem Sie N1 durch N1 plus 2 mal N2 teilen, wobei N1 die Zahl der N-Gramme mit Häufigkeit 1 ist. (4 Punkte)

Aufgabe 3) Schreiben Sie eine Funktion **estimate_prob**, welche die N-Gramm-Häufigkeiten als Argument bekommt und dann von jeder N-Gramm-Häufigkeit den Discount abzieht und das Ergebnis durch die Kontexthäufigkeit teilt. Die Kontexthäufigkeit eines N-Grammes **g** bekommen Sie, indem Sie die Häufigkeiten aller N-Gramme summieren, die sich höchstens im letzten Element von **g** unterscheiden.

Die Werte, die für jedes N-Gramm berechnet wurden, geben Sie in einer geeigneten Datenstruktur zurück. Diese Werte werden im Folgenden als *angepasste relative Häufigkeiten* bezeichnet. (10 Punkte)

Aufgabe 4) Schreiben Sie eine Funktion **compute_backoff_factors**, welche die Tabelle mit den angepassten relativen Häufigkeiten aus der vorherigen Aufgabe als Argument erhält und die Backoff-Faktoren für die verschiedenen Kontexte berechnet, indem Sie die Wahrscheinlichkeiten aller N-Gramme in der Tabelle summiert, die bis auf ein zusätzliches letztes Element mit dem Kontext-N-Gramm identisch sind. Die Summe wird dann von 1 abgezogen, um den Backoff-Faktor des Kontextes zu erhalten. Die Tabelle mit den Backoff-Faktoren geben Sie zurück. (6 Punkte)

Aufgabe 5) Schreiben Sie eine Funktion **get_prob**, welche ein N-Gramm als Argument erhält und die geglättete Wahrscheinlichkeit des letztes Elementes des N-Grammes gegeben die vorherigen Elemente zurückgibt. Dazu muss zu der angepassten relativen Häufigkeit des N-Grammes das Produkt aus Backoff-Faktor und geglätteter Backoff-Wahrscheinlichkeit addiert werden. Die Backoff-Wahrscheinlichkeit wird dabei rekursiv mit derselben Funktion **get_prob** berechnet.

Die angepasste relative Häufigkeit jedes N-Grammes kann in dem globalen Dictionary **prob** nachgeschlagen werden. Diese Werte wurden in der vorherigen Aufgabe berechnet. Das Dictionary **prob** enthält aber auch die Werte für alle kürzeren N-Gramme. Die Backoff-Faktoren des Kontextes jedes N-Grammes können in einem Dictionary **backoff** nachgeschlagen werden. Die Rekursion zur Berechnung der geglätteten Wahrscheinlichkeiten soll bei Unigrammen enden.

Geben Sie bei dieser Aufgabe zusätzlich an, welche Defaultwerte die beiden Dictionaries prob und backoff liefern sollten, falls ein Key nicht definiert ist.

(9 Punkte)

Aufgabe 5) Erklären Sie wie man mit Hilfe von Markowmodellen einen Sprachidentifizierer realisieren kann, der die Sprache eines Textes bestimmt. (3 Punkte)

Aufgabe 1) Implementieren Sie eine Funktion **compute_discount(freq)**. Der Funktion wird ein Dictionary of Dictionaries übergeben mit Werten freq[word][tag] (oder analog ein Perl-Hash %freq mit $freq{word}{tag}). Sie soll nach der Formel

$$\delta = \frac{n_1}{n_1 + 2n_2}$$

den Discount berechnen und zurückgeben. n_1 (n_2) ist die Zahl der einmal (zweimal) aufgetretenen Wort-Tag-Paare. (8 Punkte)

Aufgabe 2) Implementieren Sie eine weitere Funktion **estimate_prob**, welche dasselbe Argument freq erhält und die folgenden Werte berechnet:

$$p^*(t|w) = \frac{f(w,t) - \delta}{\sum_{t'} f(w,t')}$$

und ein Dictionary of Dictionaries (oder einen Hash) mit den berechneten Werten zurückgibt. Zur Berechnung des Discounts δ können Sie die Funktion aus Aufgabe 1 aufrufen. (10 Punkte)

Aufgabe 3) Implementieren Sie nun eine Funktion **compute_backoff**, welche ein Dictionary of Dictionaries prob mit prob[word][tag] (oder einen analogen Perl-Hash) als Argument erhält, die Backoff-Faktoren berechnet und in einem Dictionary zurückgibt.

$$\text{backoff}(w) = 1 - \sum_{t} \overset{\circ}{p}^{*}(t|w)$$

(6 Punkte)