Lösungen für Aufgabenblatt 11 –

Einführung in die Computerlinguistik (WS19/20) Abgabe bis –

Aufgabe 1) Clustering-Evaluation **Punkte: x**

Gegeben sei die folgende Menge an Elementen:

$$M = \{a, b, c, d, e, f, g, h, i, j, k, l\}$$

Zudem ist ein Gold-Clustering G vorgegeben:

$$G = \{a, b, c, d\}, \{e, f, g, h\}, \{i, j, k, l\}$$

Die Elemente dieser Menge werden von einem System S wie folgt geclustert:

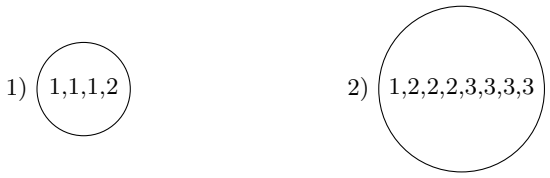
$$S = \{a, b, c, e\}, \{d, f, g, h, i, j, k, l\}$$

Berechnen Sie für das System S

1. Rand Index
2. Purity
3. Normalized Mutual Information (NMI)

in Bezug auf Gold-Cluster G.

Lösung zu 1)
Darstellung des Clusterings aus Sicht von S:



1.

$$\begin{aligned} RI =& \frac{\text{true positive links + true negative links}}{\text{all original links}} \\ =& \frac{12 + 26}{\binom{12}{2}} \\ =& \frac{38}{66} \\ =& 0.58 \end{aligned}$$

2. $P = \frac{3+4}{4+8} = \frac{7}{12} = 0.583$

3. Wir bezeichnen die Inidizes für G mit j und für S mit k:

$$\begin{aligned}
P(j=1) = P(j=2) = P(j=3) &= \frac{4}{12} \\
P(k=1) &= \frac{4}{12} \\
P(k=2) &= \frac{8}{12} \\
P(j=1, k=1) &= \frac{3}{12} \\
P(j=1, k=2) &= \frac{1}{12} \\
P(j=2, k=1) &= \frac{1}{12} \\
P(j=2, k=2) &= \frac{3}{12} \\
P(j=3, k=1) &= \frac{0}{12} \\
P(j=3, k=2) &= \frac{4}{12}
\end{aligned}$$

Somit beträgt die Mutual Information:

$$\begin{aligned}
MI = P(j=1,k=1) &\cdot \log \frac{P(j=1,k=1)}{P(j=1)\cdot P(k=1)} \\
+ P(j=1,k=2) &\cdot \log \frac{P(j=1,k=2)}{P(j=1)\cdot P(k=2)} \\
+ P(j=2,k=1) &\cdot \log \frac{P(j=2,k=1)}{P(j=2)\cdot P(k=1)} \\
+ P(j=2,k=2) &\cdot \log \frac{P(j=2,k=2)}{P(j=2)\cdot P(k=2)} \\
+ P(j=3,k=1) &\cdot \log \frac{P(j=3,k=1)}{P(j=3)\cdot P(k=1)} \\
+ P(j=3,k=2) &\cdot \log \frac{P(j=3,k=2)}{P(j=3)\cdot P(k=2)} \\
&= 0.377
\end{aligned}$$

Normalisierung mit Entropie von S und G:

$$\begin{aligned}
H_G &= -\left(3 \cdot \frac{4}{12} \cdot \log \frac{4}{12}\right) = 1.58 \\
H_S &= -\left(\frac{4}{12} \cdot \log \cdot \frac{4}{12} + \frac{8}{12} \cdot \log \frac{8}{12}\right) = 0.92 \\
HMI &= \frac{0.377}{\frac{H_G + H_S}{2}} = \frac{0.377}{1.25} = 0.30
\end{aligned}$$

Aufgabe 2) IR-Evaluation (aus einer alten Klausur) **Punkte:** –

Ein Korpus enthält eine Menge von Dokumenten, von denen 8 Dokumente für eine gegebene Query relevant sind.

Ein Information Retrieval System gibt die folgenden 20 Dokumente zurück, wobei das Ranking von links nach rechts angeben ist (d.h. ganz links steht das Dokument, das das System als das relevanteste erachtet). Die 5er-Gruppen haben keine semantische Bedeutung, sondern sollen nur die Lesbarkeit verbessern.

```
R R N N N    N N N R N    R N N N R    N N N N R
```

R bedeutet, dass das gefundene Dokument wirklich relevant ist, N bedeutet nicht-relevant.

1. Berechnen Sie precision und recall des Systems.
2. Berechnen Sie die Average Precision des Systems für diese Query.
3. Warum wird in IR meist Average Precision oder Precision at K verwendet, anstatt Precision und Recall zu verwenden? (1-Satz-Antwort)
4. Warum ist für Websuche Precision at K leichter zu verwenden als (Mean) Average Precision? (1-Satz-Antwort)

Lösung zu 2)

1. $P = 6/20$, $R = 6/8$
2. Average Precision (average for precision at all recall levels for 8 relevant documents)

   $$1/8 \cdot (1/1 + 2/2 + 3/9 + 4/11 + 5/15 + 6/20 + 0 + 0)$$

   Wichtig: Man wird dafür bestraft, dass man zwei Dokumente gar nicht gefunden hat!
3. Damit man auch das Ranking der gefundenen Dokumente berücksichtigen kann.
4. Für (Mean) Average Precision braucht man auch die Menge aller relevanten Dokumente der gesamten Dokumentenmenge. Dies ist bei einer sehr großen Dokumentmenge (zu) aufwändig zu bestimmen.

Aufgabe 3) Term Frequency - Inverse Document Frequency **Punkte:** —

Gegeben sind 9 Zeilen aus verschiedenen Gedichten von Emily Dickinson:

d_1 life is but life, and death but death !
d_2 you left me, sweet, two legacies,
d_3 sweet litigants for life.
d_4 the quiet nonchalance of death
d_5 my business, just a life i left,
d_6 sweet debt of life, each night to owe,
d_7 at least to pray is left, is left.
d_8 a death blow is a life blow to some
d_9 so quiet, oh, how quiet!

1. Erstellen Sie eine Term-Dokument-Matrix M (mit Term Frequency als Werte) für die 5 Types *life*, *sweet*, *death*, *quiet* und *left* für alle Dokumente.
2. Errechnen Sie die Inverse Document Frequency für die selben 5 Types und verwandeln Sie dann M in eine TF-IDF-Matrix.
3. Wie würden Sie die Ähnlichkeit zwischen zwei Dokumenten berechnen?
4. Wie würden Sie die Ähnlichkeit zwischen der query *death, life* und den einzelnen Dokumenten berechnen?

Lösung zu 3)

1.

d_i	life	sweet	death	quiet	left
d_1	2	0	2	0	0
d_2	0	1	0	0	1
d_3	1	1	0	0	0
d_4	0	0	1	1	0
d_5	1	0	0	0	1
d_6	1	1	0	0	0
d_7	0	0	0	0	2
d_8	1	0	1	0	0
d_9	0	0	0	2	0

2.

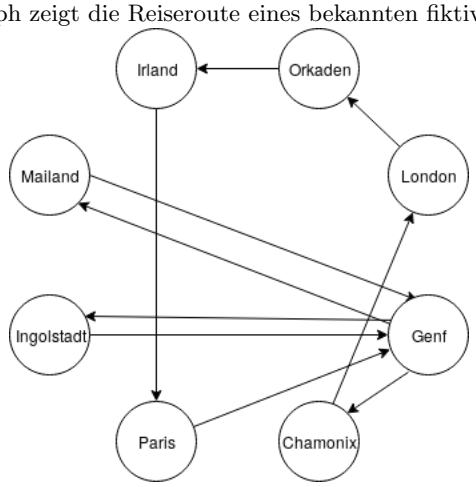
$$idf(\text{life}) = \log \frac{9}{5} \approx 0.5878$$
$$idf(\text{sweet}) = \log \frac{9}{3} \approx 1.0986$$
$$idf(\text{death}) = \log \frac{9}{3} \approx 1.0986$$
$$idf(\text{quiet}) = \log \frac{9}{2} \approx 1.5041$$
$$idf(\text{left}) = \log \frac{9}{3} \approx 1.0986$$

d_i	life	sweet	death	quiet	left
d_1	1.1756	0	2.1972	0	0
d_2	0	1.0986	0	0	1.0986
d_3	0.5878	1.0986	0	0	0
d_4	0	0	1.0986	1.5041	0
d_5	0.5878	0	0	0	1.0986
d_6	0.5878	1.0986	0	0	0
d_7	0	0	0	0	2.1972
d_8	0.5878	0	1.0986	0	0
d_9	0	0	0	3.0082	0

3. Cosinus zwischen zwei Zeilen.
4. Stelle query als weiteres Dokument und TFIDF-Vektor dar (ergibt $(0.5878, 0, 1.0986, 0, 0)$ als query Vektor). Dann Cosinus zu den einzelnen Dokumenten zum Finden der relevantesten Dokumente. Insbesondere beträgt die Cosinusähnlichkeit sowohl zu d_1 als auch d_8 gleich 1 (Bitte in Tutorial durchrechnen).

Aufgabe 4) Page Rank Iterativ **Punkte:** –

Der folgende Graph zeigt die Reiseroute eines bekannten fiktiven Charakters:



Stellen Sie den Graphen durch eine Adjazenzmatrix dar.

Verwandeln Sie diese in eine stochastische Transition Matrix, wie im PageRank-Algorithmus beschrieben (mit Sprungwahrscheinlichkeit $\alpha = 0.1$).

Benutzen Sie dann die iterative Berechnung für den steady-state vector, um die Approximation des PageRank-Vektors zu berechnen, bis sich dieser stabilisiert hat. Programmieren Sie diesen Algorithmus, um Zeit zu sparen! Initialisieren Sie den Vektor v mit einem stochastischen Vektor (was für die Konvergenz notwendig ist). Ein *stochastischer Vektor* besitzt ausschließlich nicht-negative Werte, die sich auf 1 aufaddieren, wie (0.5, 0, 0, 0.5, 0) oder (1,0,0).

Lösung zu 4)
Mit der Reihenfolge: London, Orkaden, Genf, Paris, Ingolstadt, Mailand, Chamonix, Irland sowie Ausgangspunkt in den Zeilen

$$r = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$r = \begin{bmatrix} 0.0125 & 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \\ 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.9125 \\ 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.3125 & 0.3125 & 0.3125 & 0.0125 \\ 0.0125 & 0.0125 & 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \\ 0.0125 & 0.0125 & 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \\ 0.0125 & 0.0125 & 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \\ 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \\ 0.0125 & 0.0125 & 0.0125 & 0.9125 & 0.0125 & 0.0125 & 0.0125 & 0.0125 \end{bmatrix}$$

Resultierender steady state vector v

$v = [0.1056, 0.1035, 0.2866, 0.0985, 0.0985, 0.0985, 0.1076, 0.1011]$

Aufgabe 5) PageRank: Lösung durch Gleichungssystem **Punkte: -**

Anstatt iterativ kann der PageRank auch manchmal einfach als lineares Gleichungssystem berechnet werden. Dazu wird der Haupteigenvektor der Matrix berechnet, der durch die Matrix wieder auf sich selbst transformiert wird, denn der steady state vector hat die Eigenschaft $v*T = v$. Stellen Sie für folgende Matrix das Gleichungssystem auf und berechnen Sie den Eigenvektor. Berücksichtigen Sie bei der Berechnung, dass v auch ein stochastischer Vektor ist.

$$\begin{pmatrix} 0.33 & 0.33 & 0.33 \\ 0.475 & 0.05 & 0.475 \\ 0.05 & 0.9 & 0.05 \end{pmatrix}$$

Lösung zu 5)

$$0.33v_1 + 0.475v_2 + 0.05v_3 = v_1 \quad (1)$$
$$0.33v_1 + 0.05v_2 + 0.9v_3 = v_2 \quad (2)$$
$$0.33v_1 + 0.475v_2 + 0.05v_3 = v_3 \quad (3)$$

Da v ein stochastischer Vektor ist, haben wir als zusätzlichen Constraint:

$$v_1 + v_2 + v_3 = 1 \quad (4)$$

- durch (1) und (3): $v_1 = v_3$ (5)
- (5) in (4): $v_2 = 1 - 2v_1$ (6)
- (6) und (3) in (1): $0.33v_1 + 0.475 * (1 - 2v_1) + 0.05v_1 = v_1$

$\Rightarrow$ $v_1 = 0.303$ (7)

- (7) in (5): $v_3 = 0.303$
- (7) in (6): $v_2 = 0.394$

$\Rightarrow v = [0.303, 0.394, 0.303]$

Aufgabe 6) PageRank Advanced **Punkte:** –

a) Eigenschaften der Iteration (Advanced) *Punkte:* –

Zeigen Sie, dass solange man bei PageRank mit einem stochastischen Vektor beginnt, man bei jeder Iteration wieder einen stochastischen Vektor erhält, sprich dass das Produkt eines stochastischen Vektors mit einer stochastischen Matrix wieder ein stochastischer Vektor ist.

Lösung zu 6a)

Beweise finden Sie sehr viele online unter *Multiplikation/Produkt zweier stochastischer Matrixen.* Beispiel: `https://www.staff.uni-oldenburg.de/wiland.schmale/Modul_Lineare_Algebra/Aufgabe8.pdf` für zwei Matrizen in Aufgabe 8b. Der Beweis von Vektor und Matrix ist analog..

b) Mindestgröße PageRank (Advanced) *Punkte:*

Überlegen Sie sich, dass der page rank jeder Seite mindestens α/N ist (wobei α die "Sprungwahrscheinlichkeit" ist), d.h. der steady state vector hat nur Einträge von mindestens α/N.

Was bedeutet das für den steady state Vektor und damit den Page Rank, wenn α nah bei 1 liegt (also man meistens springt)?

Ich empfehle auch die Seite `http://www.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture3/lecture3.html` als alternative PageRankerklärung. Vorsicht: Auf dieser Seite sind die Reihen und Spalten vertauscht (im Gegensatz zur Vorlesung). Dies geht natürlich genauso. Man muss dann aber den Vektor rechts, nicht links ran multiplizieren bei der Iteration.

Lösung zu 6b)

Man schaue sich die Originalidee von PageRank an. Hier sieht man, dass der Surfer entweder springen kann mit Wahrscheinlichkeit α oder laufen kann. Das heisst, dass die Wahrscheinlichkeit zu einer Seite zu gelangen immer größer oder gleich α/N sein muss (also er immer mindestens zu der Seite springen kann). D.h der Pagerank einer Seite ist immer die Wahrscheinlichkeit dorthin zu springen (α/N) plus die Wahrscheinlichkeit dorthin zu laufen.

Sollte α also näher an 1 liegen, dann sind alle Vektoreinträge immer näher an der Gleichverteilung (da es sich ja um einen stochastischen Vektor handeln muss).. Würde man also fast nur springen, dann ist das Linksytem unwichtig und alle Seiten werden gleich wichtig.