Ruprecht-Karls-Universität Heidelberg
Institut für Informatik
Sommersemester 2010
Seminar: Text Mining
Dozenten: Jannik Strötgen
Prof. Dr. Michael Gertz

Seminararbeit

Coreference of Named Entities

Name:	Philipp Schäfer, Daniel Kruck
Matrikelnummer:	2612579 (Philipp), 2440234 (Daniel)
Studiengang:	Angewandte Informatik (6. Semester)
Email:	trashzopf@googlemail.com (Philipp), daniel.kruck@gmx.net (Daniel)
Datum der Abgabe:	28. Juli 2010

Hiermit versichere ich **Philipp Schäfer, Daniel Kruck**, dass ich die Hausarbeit mit dem Titel **Coreference of Named Entities** im Seminar **Text Mining** im **Sommersemester 2010** bei **Jannik Strötgen** und **Prof. Dr. Michael Gertz** selbstständig und nur mit den in der Arbeit angegebenen Hilfsmitteln verfasst habe. Zitate sowie der Gebrauch fremder Quellen, Texte und Hilfsmittel habe ich nach den Regeln wissenschaftlicher Praxis eindeutig gekennzeichnet. Mir ist bewusst, dass ich fremde Texte und Textpassagen nicht als meine eigenen ausgeben darf und dass ein Verstoß gegen diese Grundregel des wissenschaftlichen Arbeitens als Täuschungs- und Betrugsversuch gilt, der entsprechende Konsequenzen nach sich zieht. Diese bestehen in der Bewertung der Prüfungsleistung mit "nicht ausreichend"(5,0) sowie ggf. weiteren Maßnahmen.
Außerdem bestätige ich, dass diese Arbeit in gleicher oder ähnlicher Form noch in keinem anderen Seminar vorgelegt wurde.

Heidelberg, den 28. Juli 2010 _____

Inhaltsverzeichnis

1 Einleitung

In letzter Zeit hat sich das Sortier- und Suchverhalten der Menschheit geändert. Sortierte man früher noch seine Dokumente in Ordner, ist man heute glücklich, wenn man mit leistungsstarken Suchalgorithmen schnell und präzise das gewünschte Dokument findet.
Dabei beschränken sich Suchanfragen nicht nur auf lokale Daten, sondern werden sogar großteils ans Web gestellt. Eine häufige Anfrageform an Suchmaschinen sind hierbei die *named entity queries*[1].

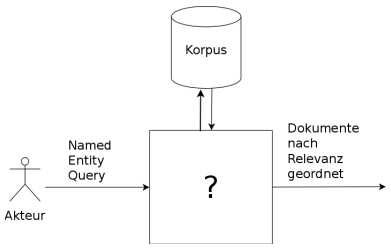

Abbildung 1: Häufige Suchanfrage an Suchmaschinen

Ein Problem bei solchen Suchanfragen ist es, die Häufigkeit der **named entity** in Dokumenten richtig zu erfassen[6]. Denn die Referenzierung des Objektes, das sich hinter einem Eigennamen verbirgt, wird sowohl mit dem Eigennamen selbst, als auch mit kompakteren Ausdrücken vorgenommen.
So wird beispielsweise in einem Text, der von Peter Chen handelt, die Person *Peter Chen* auch mit *„er“* referenziert.

> „**Chen** has recieved serveral awards in the fields of Information Technology. **He** received the Data Resource Management Technology Award [. . .]“ [4]

In der folgenden Hausarbeit wird ein statistisches Verfahren erklärt, welches die Häufigkeit von Eigennamen in Dokumenten schätzen soll. Grundlage für diese Ausarbeitung ist ein Paper der Herren Na und Ng über „A 2-Poisson Model for Probabilistic Coreference of Named Entities for Improved Text Retrieval“[6].

2 Co-referentially Enhanced Entity Frequency (CEEF)

2.1 Plausible Annahme über die Verteilung von Koreferenzen

Die Häufigkeit von Eigennamen in Dokumenten ist schwer zu erfassen. Zählt man nur die *named entities* selbst, erhält man die raw entity frequency:

$$tf\,(e;d) = \sharp(\text{named entity})$$

Die raw entity frequency beachtet noch keine zusätzlichen Koreferenzen auf das Objekt hinter dem gesuchten Eigennamen. Um später ein besseres Ranking ausführen zu können, möchten wir möglichst alle Referenzen erfassen. Also addieren wir die Häufigkeit der Koreferenzen zu unserer raw entity frequency:

$$tf\,(e;d) \leq tf_{true}\,(e;d) = tf\,(e;d) + \underbrace{atf(e_Q;A,d)}_{Koreferenzen}$$

Theoretisch sind wir nun am Ziel. Praktisch ist es aber nicht so einfach, die Häufigkeit der Koreferenzen zu bestimmen. Mit Programmen, die Koreferenzen vollständig auflösen, geht ein überdimensionaler Berechnungsaufwand einher. Zudem sind diese Programme noch nicht besonders Präzise und ordnen weniger als 70% der Koreferenzen richtig zu.[6]
Seung-Hoon Na und Hwee Tou Ng untersuchten deswegen einen statistischen Ansatz zur Schätzung der Koreferenzen. Dafür nehmen sie folgendes an:

> "Our key assumption is that the frequency of anaphoric expressions is distributed over named entities in a document according to the probabilities of whether the document is elite for the named entities."[6]

Man geht also davon aus, dass sich mögliche Koreferenzen in einem Dokument auf die verschiedenen Eigennamen verteilen. Die Häufigkeit der Korelation zwischen Koreferenz und Eigennamen hängt dabei von der Wahrscheinlichkeit ab, dass ein Dokument eben diesen Eigennamen thematisiert. Ausgehend von dieser Annahme wird nun ein statistisches Modell erstellt, das in der Lage ist, die Häufigkeit dieser Korelationen in Dokumenten zu schätzen. Dazu wird der Kontext vernachlässigt und die Anzahl der anaphorischen Ausdrücke mit einem Faktor multipliziert, der die Wahrscheinlichkeit

eines Bezuges auf die gesuchte Named Entity abschätzt.

$$tf_{true}\left(e_Q; d\right) \approx tf\left(e_Q; d\right) + P\left(e_Q|A, d\right) tf\left(A; d\right)$$

Im folgenden gilt es diesen Faktor $P\left(e_Q|A, d\right)$ zu bestimmen.

2.2 Formalisierung

Bevor wir zur eigentlichen Statistik kommen, legen wir noch die mathematische Schreibweise für diese Aufgabe fest. Zunächst sei Q unsere Query. Die Query besteht hier zur Vereinfachung aus einem einzigen Eigennamen.
Weiter sei e ein beliebiger Eigennamen. Entspricht e der Suchanfrage der Query Q, so nennen wir den Eigennamen gesuchte Entität e_Q. Ist e ein beliebiger nicht gesuchter Eigenname, so nennen wir die Entität nicht gesuchter Eigenname e_N.
Die Koreferenzen werden *mögliche Koreferenzen* zu einem Eigennamen genannt, wenn sie semantisch zu dem Eigennamen passen. Unterteilen wir beispielsweise die Eigennamen in Objekte und Personen, so sind {“he”,“she”} mögliche Koreferenzen zu Personen. Eine mögliche Koreferenz auf Objekte ist {“it”}. Die Menge aller möglichen Koreferenzen wird A genannt.
Zählt man alle möglichen Koreferenzen A in einem Dokument d, so schreibt man $tf\left(A; d\right)$. Die Menge aller möglichen Eigennamen zu einer Menge von Koreferenzen in einem Dokument d wird mit $\varepsilon\left(A; d\right)$ beschrieben.

- Q = query (Suche)
- e_Q = query entity (gesuchte Entität)
- e_N = non-query entity
- A = Menge plausibler anaphorischer Ausdrücke
- $tf(A; d)$ = Anzahl von A in Dokument d
- $\varepsilon(A; d)$ = Menge plausibler Entitäten in Dokument d
- $tf(e; d)$ = raw entity frequency
- $atf(e; A, d)$ = anaphoric entity frequency

Ausgerüstet mit einer einheitlichen Schreibweise wird zunächst Eliteness diskutiert und dann der Kern des Ganzen - das Poisson-Modell für statistische Häufigkeit von Koreferenzen - hergeleitet.

2.3 Eliteness

Definition:
A document is elite for a term if the document is "about" the concept represented by the term [5].

Bei genauerer Begutachtung der Definition liegt die grundlegende Annahme von Seung-Hoon Na und Hwee Tou Ng, dass die Bezüge der anaphorischen Ausdrücke auf die Entitäten im Zusammenhang stehen, ob ein Dokument bezüglich dieser Entität „elite" ist, oder nicht, nahe. Wir definieren uns also einen weiteren Faktor:

Definition:
$P(\mathbf{E}(e) = 1|d)$ ist die Wahrscheinlichkeit, dass ein Dokument für eine Entität e „elite" ist.

Jetzt können wir folgendes setzen:

$$P(e_Q|A, d) = \frac{P(\mathbf{E}(e) = 1|d)}{\sum_{e \in \varepsilon(A;d)} P(\mathbf{E}(e) = 1|d)} \tag{1}$$

Wir machen also nichts anderes, als die Wahrscheinlichkeit der Eliteness des Dokumentes bezogen auf unsere gesuchte Entität in Relation mit den Wahrscheinlichkeiten der anderen Named Entities zu stellen.
Jetzt vereinfachen wir die Formel noch ein wenig, indem wir zuerst alle non-query Entitäten nach der Wahrscheinlichkeit der Eliteness sortieren. Danach werden die K non-query Entitäten mit der höchsten Wahrscheinlichkeit selektiert, sodass sich unsere Formel auf Folgendes beschränkt:

$$P(e_Q|A, d) \approx \frac{P(\mathbf{E}(e) = 1|d)}{P(\mathbf{E}(e) = 1|d) + \sum_{i=1}^{K} P(\mathbf{E}(e_n^{(i)}) = 1|d)}$$

K ist in dem Fall frei gewählt und kann entsprechend angepasst werden. So dass tatsächlich nur Wahrscheinlichkeiten vernachlässigt werden, die nicht ins Gewicht fallen.
Nun suchen wir uns eine repräsentative non-query Entität heraus und setzen die Wahrscheinlichkeit der Eliteness aller anderen der Top-K Entitäten mit

der Wahrscheinlichkeit der Herausgesuchten gleich. Somit erhalten wir eine stark vereinfachte, aber trotzdem recht genaue Alternative von (1).

$$P(e_Q|A,d) \approx \frac{P(\mathbf{E}(e)=1|d)}{P(\mathbf{E}(e)=1|d) + K \cdot P(\mathbf{E}(e_N)=1|d)}$$

Im Folgenden werden wir eine performante und einfache Methode beschreiben, mit der es möglich ist, den Faktor $P(\mathbf{E}(e)=1|d)$ zu berechnen.

2.4 Poisson

Aufgrund ihrer Wichtigkeit, nochmal die grundlegende Annahme von Seung-Hoon Na und Hwee Tou Ng:

> "Our key assumption is that the frequency of anaphoric expressions is distributed over named entities in a document according to the probabilities of whether the document is elite for the named entities."[6]

Um diese Annahme in ein mathematisches Modell zu übersetzen, muss man die Wahrscheinlichkeit abschätzen, dass ein Dokument "elite" ist. Wir suchen also eine Formel für

$$P\left(E\left(e\right)=1|d\right) = ?$$

Zunächst interessiert uns hierfür die Wahrscheinlichkeit, dass eine beliebige Entität tf-fach in einem Dokument vorkommt. Dafür wird ein 2-Poisson Mixture Modell herangezogen.

$$P\left(tf\right) = \pi_e \frac{e^{-\lambda_e}\lambda_e^{tf}}{tf!} + \left(1-\pi_e\right)\frac{e^{-\mu_e}\mu_e^{tf}}{tf!} \tag{2}$$

Der erste Term repräsentiert hierbei die Wahrscheinlichkeit des tf-fachen Auftretens der Entität in der Funktion als "elite"-Eigennamen, der zweite Term steht für die Wahrscheinlichkeit des tf-fachen Auftretens der Entität als "non-elite"-Eigennamen.

$$P\left(tf\right) = \underbrace{\pi_e \frac{e^{-\lambda_e}\lambda_e^{tf}}{tf!}}_{Wahrscheinlichkeitsanteil-elite} + \underbrace{\left(1-\pi_e\right)\frac{e^{-\mu_e}\mu_e^{tf}}{tf!}}_{Wahrscheinlichkeitsanteil-nonelite}$$

Dabei deutet π_e auf den Erwartungswert hin, ob ein Dokument "elite" ist oder nicht.

π_e, λ_e und μ_e werden festgelegt als:

$$\pi_e = \frac{df\,(e)}{df\,(e) + N},\ \lambda_e = \frac{cf\,(e)}{df\,(e)},\ \mu_e = \frac{cf(e)}{N}$$

wobei

- N: Anzahl der Dokumente
- $df(e)$: Anzahl der Dokumente, in denen e vorkommt
- $cf(e)$: Anzahl von e in allen Dokumenten

Damit haben wir ein Modell, dass die Wahrscheinlichkeit der Häufigkeit von Entitäten in Dokumenten schätzt. Wichtig ist hierbei, dass sowohl das Auftreten von "elite"-Entitäten in Dokumenten mit einer Poissonverteilung modelliert wird, als auch das "zufällige" Auftreten der "nonelite"-Entitäten.
Eine einfache Poissonverteilung sieht in Formel und im Diagramm wie folgt aus:

$$P_\lambda(k) = \frac{\lambda^k}{k!}e^{-\lambda}$$

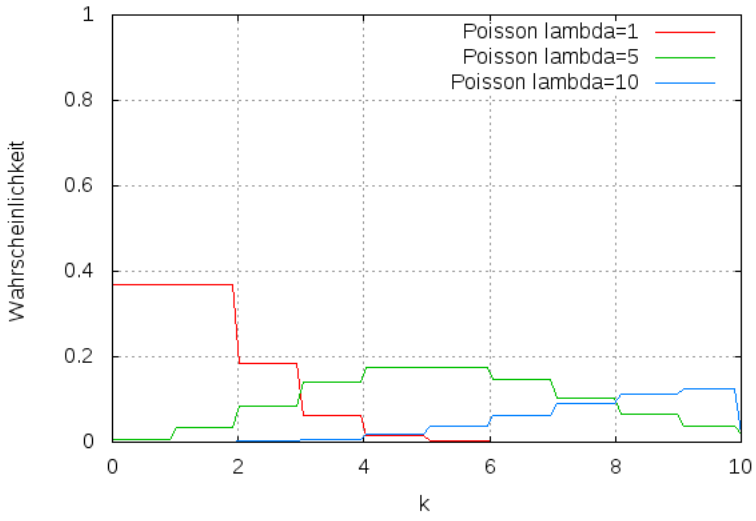

Abbildung 2: Einfache Poissonverteilung

Mit Hilfe von (2) können wir nun $P(\mathbf{E}(e) = 1|d)$ bestimmen. Wir setzen

$$P(\mathbf{E}(e) = 1|d) = \frac{\pi_e P(tf(e;d)|\mathbf{E}(e) = 1)}{\pi_e P(tf(e;d)|\mathbf{E}(e) = 1) + (1 - \pi_e)P(tf(e;d)|\mathbf{E}(e) = 0)}$$

mit

$$\begin{aligned} P(tf(e;d)|\mathbf{E}(e)=1) &= \frac{e^{-\lambda_e}\lambda_e^{tf}}{tf!} \\ P(tf(e;d)|\mathbf{E}(e)=0) &= \frac{e^{-\mu_e}\mu_e^{tf}}{tf!} \end{aligned}$$

und erhalten so

$$\begin{aligned} P(\mathbf{E}(e)=1|d) &= \frac{\pi_e P(tf(e;d)|\mathbf{E}(e)=1)}{\pi_e P(tf(e;d)|\mathbf{E}(e)=1)+(1-\pi_e)P(tf(e;d)|\mathbf{E}(e)=0)} \\ &= \frac{\pi_e \frac{e^{-\lambda_e}\lambda_e^{tf}}{tf!}}{\pi_e \frac{e^{-\lambda_e}\lambda_e^{tf}}{tf!}+(1-\pi_e)\frac{e^{-\mu_e}\mu_e^{tf}}{tf!}} \\ &= \frac{\pi_e e^{-\lambda_e}\lambda_e^{tf}}{\pi_e e^{-\lambda_e}\lambda_e^{tf}+(1-\pi_e)e^{-\mu_e}\mu_e^{tf}} \\ &= \frac{\pi_e}{\pi_e+(1-\pi_e)\frac{e^{-\mu_e}\mu_e^{tf}}{e^{-\lambda_e}\lambda_e^{tf}}} \\ &= \frac{\pi_e}{\pi_e+(1-\pi_e)e^{\lambda_e-\mu_e}\left(\frac{\mu_e}{\lambda_e}\right)^{tf(e;d)}} \end{aligned}$$

Jetzt werden 2 Fälle unterschieden:

1. $e = e_Q$
   Wir machen die plausible Annahme, dass $tf(e;d) \geq 1$, wenn das Dokument d elite ist für die gesuchte Entität e. Alle anderen Dokumente im Test seien non-elite bezüglich e.
   Dann können wir die π_e, λ_e und μ_e wie oben beschrieben ersetzen und erhalten

$$\begin{aligned} P(\mathbf{E}(e_Q)=1|d) &= \frac{\pi_e}{\pi_e+(1-\pi_e)e^{\lambda_e-\mu_e}\left(\frac{\mu_e}{\lambda_e}\right)^{tf(e;d)}} \\ &= \frac{\frac{df(e)}{df(e)+N}}{\frac{df(e)}{df(e)+N}+\left(1-\frac{df(e)}{df(e)+N}\right)e^{\frac{cf(e)}{df(e)}-\frac{cf(e)}{N}}\left(\frac{\frac{cf(e)}{N}}{\frac{cf(e)}{df(e)}}\right)^{tf(e;d)}} \\ &= \frac{1}{1+\left(\frac{df(e)}{N}\right)^{tf(e;d)-1}e^{\frac{cf(e)}{df(e)}-\frac{cf(e)}{N}}} \end{aligned}$$

2. $e \neq e_Q$
   Auch hier machen wir zu Beginn zwei Annahmen:

   (a) Es seien alle non-query Entitäten vom selben Typ, wie die query Entität

   (b) Es sei die Termfrequency ($tf(e;d)$) einer Top-K non-query Entität gleich dem Durschnittsvorkommen einer query Entität in einem elite Dokument

   Unter diesen Vorraussetzungen dürfen die λ_e, μ_e und π_e wie oben gewählt werden. Dies führt uns erneut zu der Formel

$$P(\mathbf{E}(e) = 1|d) = \frac{1}{1 + \left(\frac{df(e)}{N}\right)^{tf(e;d)-1} e^{\frac{cf(e)}{df(e)} - \frac{cf(e)}{N}}}$$

   Aufgrund von (b) gilt jetzt aber, dass $tf(e;d) = \frac{cf(e)}{df(e)}$, weshalb wir in diesem Fall zu dem Ergebnis kommen, dass

$$P(\mathbf{E}(e_N) = 1|d) = \frac{1}{1 + \left(\frac{df(e)}{N}\right)^{\frac{cf(e)}{df(e)}-1} e^{\frac{cf(e)}{df(e)} - \frac{cf(e)}{N}}}$$

Mit diesen Ergebnissen können wir jetzt unseren Wahrscheinlichkeitsfaktor

$$P(e_Q|A,d) \approx \frac{P(\mathbf{E}(e) = 1|d)}{P(\mathbf{E}(e) = 1|d) + K \cdot P(\mathbf{E}(e_N) = 1|d)}$$

abschätzen und damit die gesuchte wahre Termfrequency

$$tf_{true}(e_Q;d) \approx tf(e_Q;d) + P(e_Q|a,d)tf(A;d)$$

Wir wollen zum Schluss noch darauf aufmerksam machen, dass es sinnvoller ist, statt der normalen Termfrequency, die so genannte „normalized Termfrequency“ zu verwenden.

Definition: **normalized Termfrequency**

$$ntf(e;d) = \frac{tf(e;d) \cdot avglen}{len(d)}$$

Wobei *avglen* die Durchschnittslänge eines Dokuments im Test ist und $len(d)$ die Länge des aktuellen Dokuments.

Die normalized Termfrequency setzt also das Vorkommen der Entitäten in Relation mit der Länge der Dokumente. Das ist wichtig, da die Termfrequency in langen Dokumenten ja meist höher ist als in Kurzen, dies aber das Dokument nicht unbedingt relevanter macht. Somit setzen wir also

$$ntf_{true}(e_Q; d) \approx ntf(e_Q; d) + P(e_Q|a, d)ntf(A; d)$$

und sind fertig.

3 Zusätzliche Bearbeitungsschritte

3.1 Automatische Klassifizierung der named entities

Nachdem man nun die Wahrscheinlichkeit schätzen kann, mit der ein Dokument "elite" ist, benötigt man nun noch die Häufigkeit der möglichen Koreferenzen.
Dafür müssen wir zunächst herausfinden, von welchem Typ die Entity Query ist. Für Personen wird

$$A_P = \{\text{'he', 'she', 'his', 'her', 'himself', 'herself'}\}$$

als sinnvolle Menge von Koreferenzen angesehen. Für Objekte wird eine erweiterbare Menge an Koreferenzen um

$$A_0 = \{\text{'it', 'its', } \dots\}$$

bei Bedarf verwendet. Das hat den Grund, dass Objekte oft mit Nomen referenziert werden. So wird beispielsweise "Google" eventuell mit "Suchmaschine", "Internetriese" oder ähnlichem bezeichnet.

Für die Unterscheidung zwischen Person und Objekt wird ein Support Vector Machine Klassifikator eingesetzt.
Dafür werden zunächst Entitäten aus einer Trainingsmenge betrachtet. Bei diesen Eigennamen wird dem Computer als zusätzliche Information mitgeteilt, ob es sich um eine Person oder um ein Objekt handelt. Der Computer

versucht dann die besten M Treffer zu den jeweiligen Eigennamen herauszufinden und errechnet charakteristische Werte für diesen Eigennamen. Da wir hier herausbekommen wollen, ob es sich um ein Objekt oder um eine Person handelt, werden bei den Top M Einträgen die Koreferenzen betrachtet. Kommt häufig $A_P = \{$'he', 'she', 'his', 'her', 'himself', 'herself'$\}$ vor, so wird es wahrscheinlicher, dass es sich die betrachtete Entität eine Person ist. Kommt $A_0 = \{$'it', 'its'$\}$ häufig vor, so handelt es sich wahrscheinlich um ein Objekt. Die Quantifizierung erfolgt normiert nach:

$$f_P(Q) = \frac{1}{|F(Q)|} \sum_{d \in F(Q)} \frac{tf(A_P; d)}{len(d)}$$

$$f_O(Q) = \frac{1}{|F(Q)|} \sum_{d \in F(Q)} \frac{tf(A_O; d)}{len(d)}$$

Dabei stellt $F(Q)$ die Menge der besten M Treffer dar, $|F(Q)|$ die Kardinalität dieser Menge, $tf(A_P; d)$ die Häufigkeit der Koreferenzen auf Personen in dem jeweiligen Dokument und $len(d)$ gibt die Anzahl der Worte eines Dokumentes aus.

Der Computer errechnet $f_O(Q)$ und $f_P(Q)$ für jeden Eigennamen der Trainingsmenge. Visualisiert könnte ein Diagramm dazu wie folgt aussehen:

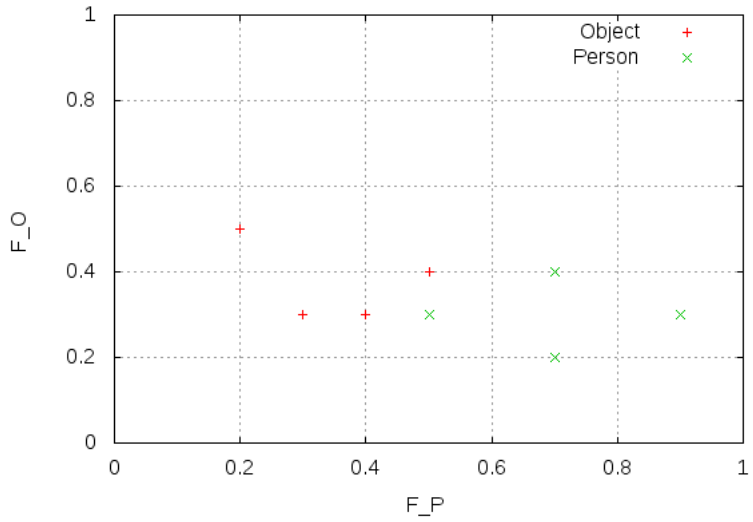

Abbildung 3: Mögliche Visualisierung der SVM nach dem Trainingsschritt ohne Hyperebene

Nachdem sich die Eigenschaften von Personen und Objekten hoffentlich etwas unterscheiden und die verschiedenen Typen im Diagramm getrennt erscheinen, muss man nun noch eine Hyperebene mit möglichst großem Abstand zum Rand zwischen den aufgetrennten Typen einfügen. Dafür wurde im Paper über "A 2-Poisson Model for Probabilistic Coreference of Named Entities for Improved Text Retrieval" [6] eine Radial Basis Function herangezogen.

3.2 Automatische Erkennung von Koreferenzen

Da die Menge an Koreferenzen

$$A_O = \{\text{'it', 'its'}\}$$

für Objekte nicht ausreichend ist, haben die Autoren des Papers "A 2-Poisson Model for Probabilistic Coreference of Named Entities for Improved Text Retrieval"[6] ein Verfahren entwickelt, um die Menge A_O zu erweitern. Sie nennen ihren Algorithmus "Feedback-Based Identification of Anaphoric Expressions".
Man nimmt an, dass häufige auftretende Nomen mit bestimmten Artikeln in der Umgebung von Eigennamen diese referenzieren.

Google weiß, wo du bist

Der Datenkrake hat ein enormes Gedächtnis – und will noch mehr erfahren. Nun sollen die Menschen Google auf dem Handy ständig bei sich tragen[...]

Seit Beginn dieser Woche verkaufen der Internetkonzern Google und die Mobilfunkfirma TMobile hierzulande das Handy G1, und wer es sich zulegt, trägt fortan einen cleveren kleinen Helfer mit sich herum.[...]

Denn diese Informationen hat der Internetkonzern bisher nicht, der wie kein anderer davon lebt, zu wissen, was seine Nutzer tun. Wer sie sind. Was sie wollen. Und bald auch, wo sie sind?[...]

Einziger Ausweg für den Internetkonzern war, selbst ein Betriebssystem zu entwickeln, den Wettbewerb um immer leistungsfähigere Mobiltelefone zu verschärfen und nebenbei die Nutzer nicht nur zur eigenen Suche zu lenken, sondern auch zu Google Mail, Google Kalender, Google Stadtpläne, Google News, Google Earth, dem konzerneigenen Videoportal YouTube und vielem mehr.[...][2]

Wie man an dem Beispiel schön sieht, wird Google oft mit einem bestimmten Artikel plus Internetkonzern beschrieben.

Diese Nomen, die als Koreferenzen zu einem bestimmten Eigennamen auftreten können, gilt es nun zu erkennen. Dafür werden lexiko-syntaktische Muster erzeugt, um bestimmten Nomen automatisch zu extrahieren. Dann werden diejenigen Dokumente gesucht, die nach herkömmlichen Rankingmethoden am relevantesten für die Query Entity sind. Aus den relevantesten M Dokumenten werden daraufhin die bestimmten Nomen extrahiert. Zum Schluss werden die Nomen noch nach Häufigkeit sortiert. Die häufigsten M Nomen werden dann zur Menge A_O hinzugefügt. Man kann zuvor noch ein Filterschritt einbauen, der diejenigen Nomen ausblendet, die in allen Dokumenten am häufigsten mit dem bestimmten Artikel auftreten. Diese sind nämlich nicht Query-spezifisch und vermutliche keine Koreferenz auf das Objekt hinter der Query Entity.

3.3 Retrieval Method

Um unsere Methode zu testen, müssen wir uns zunächst auf eine retrieval method bzw. Ranking Methode festlegen. Die Herren Na und Ng verwenden hierfür eine Variante des BM25-Models, welche die CEEF-Methode zur Bestimmung der Termfrequency benutzt und wie folgt definiert ist:

$$BM25_{CEEF}(e_Q; d) = OkapiTF(e_Q; d) \cdot IDF(e_Q)$$

wobei

$$OkapiTF(e_Q; d) = \frac{(k_1 + 1) \cdot tf_{CEEF}(e_Q; d)}{tf_{CEEF}(e_Q; d) + k_1 \left((1 - b) + b\frac{len(d)}{avglen}\right)}$$

b und k_1 dienen hier der Normierung. $IDF(e_Q)$ steht für die inverse Documentfrequency und wird vom BM25-Model zur Gewichtung verwendet und wird wie folgt berechnet:

$$IDF(e_Q) = \log\left(\frac{N}{df(e_Q)}\right) \text{ [3]}$$

Es ist zudem noch zu erwähnen, dass diese Methode mit anderen aktuellen und als „state-of-the-art“ geltenden retrieval methods kombinierbar ist. Dies funktioniert so, dass man sich einen Normierungfaktor $\alpha \in [0, 1]$ wählt und damit die zwei Methoden ($BM25_{CEEF}$ und eine anderen state-of-the-art retrieval method Sim_{BASE}) entsprechend gewichtet und aufaddiert.

$$Sim(Q;d) = (1-\alpha) \cdot Sim_{BASE}(Q;d) + \alpha \cdot BM25_{CEEF}(e_Q;d)$$

4 Evaluation

Im Paper "A 2-Poisson Model for Probabilistic Coreference of Named Entities for Improved Text Retrieval"[6] wurden noch verschiedene Performancetests durchgeführt. Dafür wurde die TREC Blog Sammlung verwendet. Die meisten Suchanfragen darin sind Eigennamen. Davon sind 33 Personenentitäten und 108 Objekte. Die restlichen 9 Suchanfragen sind allgemeiner Natur. Von den 150 Queries wurden 50 Suchanfragen als Trainingsmenge zum Parametertuning verwendet und die restlichen 100 Queries zum Testen.
Die Support Vector Machine Klassifikatoren wurden ebenfalls mit 50 Queries eingestellt. Damit konnten von den verbleibenden 100 Suchanfragen 93 richtig einem Objekt oder einer Person zugeordnet werden. Von den 7 falsch klassifizierten Suchanfragen wurden 4 Objekte als Personen erkannt und 3 Personen als Objekte.
Zuletzt wurden verschiedene Testläufe mit verschiedenen Modellen gestartet:

- **REF**: Hierfür wurde die raw entity frequency im vorgestellten Retrieval Modell verwendet.
- **CEEF-Thr**: Das Ranking wurde mit dem Coreferntial Enhanced Entity Frequency Modell nach dem threshold Ansatz durchgeführt.
- **CEEF-2Poisson**: Ranking mit CEEF-Modell. Die möglichen Koreferenzen beschränkten sich dabei auf die Mengen
  A_P = {'he', 'she', 'his', 'her', 'himself', 'herself'} und A_0 = {'it', 'its'}.
- **CEEF-FB-2Poisson**: Die möglichen Koreferenzen wurden mit dem unter "Automatische Erkennung von Koreferenzen" vorgestellten Verfahren von Seung-Hoon Na und Hwee Tou Ng ermittelt. Mit den so erhaltenen Mengen an Koreferenzen wurde das CEEF-Modell angewandt.
- **CEEF-FB-2Poisson***: Hier wurde das CEEF-Model mit der "Feedbackbased Identification" verwendet. Allerdings wurden die Entitäten nicht mit dem Support Vector Machine Klassifikator Personen oder Objekten zugeordnet, sondern per Hand.

Methods	MAP	Pr@5	Pr@10
REF	0,3929	0,6707	0,6616
CEEF-Thr	0,3968	0,6788	0,6778
CEEF-2Poisson	0,4139	0,7060	0,6940
CEEF-FB-2Poisson	0,4165	0,7320	0,7200
CEEF-FB-2Poisson*	0,4177	0,7420	0,7270

Tabelle 1: Comparison of MAP, Pr@5, Pr@10
[6]

Methods	LM	LMFB
B (Baseline Run)	0,4043	0,4254
B+REF	0,4665	0,4868
B+CEEF-Thr	0,4831	0,4942
B+CEEF-2Poisson	0,4874	0,5000
B+CEEF-FB-2Poisson	0,5074	0,5090
B+CEEF-FB-2Poisson*	0,5087	0,5106

Tabelle 2: Comparison of MAP of REF and CEEF methods combining with Language Models
[6]

Literatur

[1] Guha, R. und A. Garg. Disambiguating people in search. *In TAP: Building the Semantic Web*, 2003.

[2] Hamann, Götz und Marcus Rohwetter. Google weiß, wo du bist. *Die Zeit*, 26.02.2009.

[3] Modified: 18:48, 23 July 2010 FarzanehSarafraz. tf-idf, 2010.

[4] Modified: 21:02, 21 May 2010 Nineball. Peter chen, 2010.

[5] Robertson, S. E. und S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. *In SIGIR'94*, Seite 232–241, 1994.

[6] Seung-Hoon Na, Hwee Tou Ng. A 2-poisson model for probabilistic coreference of named entities for improved text retrieval. *Annual ACM Conference on Research and Development in Information Retrieval*, 2009.

Query Type	REF	CEEF	CEEF-FB	CEEF-FB*
Person	0,5640	0,5793	0,5761	0,5781
Company, etc.	0,5061	0,5170	0,5305	0,5305
Product	0,5245	0,5401	0,5680	0,5680
TV program, etc	0,3664	0,3754	0,3793	0,3994
Organization	0,4468	0,4682	0,4701	0,4701
Event, etc.	0,5039	0,4987	0,5012	0,5021
Other Types	0,4785	0,4889	0,5040	0,5040
Topic	0,2754	0,3078	0,3094	0,3123
All	0,4868	0,5000	0,5090	0,5106

Tabelle 3: Comparison of MAP of REF and CEEF methods, combining with LMFB each query type.[6]

Query Type	NONE	REF	CEEF	CEEF-FB
TREC '07	0,5406	0,5569	0,5646	0,5733
TREC '08	0,5032	0,5216	0,5248	0,5330

Tabelle 4: Comparison of MAP of TREC08BEST and others. NONE means non-combined baseline run.

[6]