Semantische Suche

Hannah Bast

Einleitung

Volltextsuche

In der klassischen Volltextsuche gibt die Benutzerin eine Reihe von (typischerweise wenigen) Suchwörtern ein. Die Suchmaschine gibt dann eine Reihe von Dokumenten zurück, die diese Suchwörter (oder Schreibvarianten davon) enthalten. Diese Dokumente sind typischerweise danach sortiert, wie *prominent* die Suchwörter in ihnen vorkommen. Dabei spielen zum Beispiel eine Rolle: die Anzahl der Vorkommen im Dokument (relativ zur Anzahl der Vorkommen insgesamt), Vorkommen im Titel oder einer Überschrift, die relative Nähe der Suchwörter im Dokument, die absolute Wichtigkeit / Popularität des Dokumentes, usw.

Diese Volltextsuche funktioniert gut, solange hinreichend viele der von der Benutzerin gesuchten Dokumente die eingegebenen Schlüsselwörter (oder Varianten davon) enthalten, und die erwähnte Prominenz der Vorkommen für die relevantesten Dokumente am höchsten ist. Zum Beispiel wird eine Google-Suche nach *broccoli* die entsprechende Wikipedia-Seite als ersten Treffer liefern, weil es eine populäre Seite ist, die das Suchwort im Titel enthält. Eine Anfrage nach *broccoli gardening* wird auch funktionieren, weil viele relevante Dokumente beide Suchwörter enthalten, und genügend davon im Titel oder in einer Überschrift und nah beieinander.

Die Volltextsuche erreicht ihre Grenzen bei einer Suchanfrage der Art *plants with edible leaves*[1].

Das erste Problem ist wie folgt. Viele relevante Dokumente werden sehr wohl die Worte *edible leaves* enthalten, aber es gibt keinen Grund anzunehmen, dass sie das Wort *plants* enthalten, und auch keine Variationen davon wie *plant* oder *botany*. Stattdessen werden sie den Namen einer bestimmten Pflanze enthalten, zum Beispiel *broccoli*. Hier würde man also das Wissen benötigen, dass es sich bei *broccoli* um eine spezielle *plant* handelt. Das ist genau die Art von Wissen, wie man sie in Ontologien findet, siehe Abschn. „Ontologien".

Ein noch schwerwiegenderes Problem ist, dass für eine Anfrage wie *plants with edible leaves* die gewünschten Ergebnisse keine Dokumente sind. Sondern gefragt ist nach einer Liste von sogenannten *Entitäten*, in dem Fall Pflanzen, und zwar einer möglichst vollständigen. Mit den Techniken der gewöhnlichen Volltextsuche wäre aber die Wahrscheinlichkeit hoch, dass wir zahlreiche Treffer für ein- und dieselbe Pflanze bekämen, während andere Pflanzen gar nicht vorkommen.

Das größte Problem bei solchen Suchanfragen ist aber, dass die relevanten Information gar nicht unbedingt alle in einem Dokument vorkommen. Betrachten wir die Suchanfrage *plants with edible leaves and native to europe*. Die Information, dass eine bestimmte Pflanze, zum Beispiel *broccoli*, essbare Blätter hat mag in einem Dokument stehen, die Information, dass sie einheimisch in Europa ist,

[1] Wir wollen hier den untypischen Fall ausschließen, dass es ein Dokument bzw. eine Webseite gibt, die diese Suchwörter und die dazugehörige Ergebnisliste enthält.

DOI 10.1007/s00287-013-0678-z

Hannah Bast
Institut für Informatik, Albert-Ludwigs-Universität Freiburg, Freiburg
E-Mail: bast@informatik.uni-freiburg.de

Zusammenfassung
Die klassische Volltextsuche sucht nach Vorkommen der eingegebenen Suchwörter in einer gegebenen Menge von Texten. Dieser Ansatz funktioniert bei vielen Anfragen sehr gut, hat aber auch seine offensichtlichen Grenzen. Bei der semantischen Suche versucht man, sowohl die Suchanfrage als auch die Texte in denen gesucht wird zu „verstehen“. Dieser Artikel gibt einen Überblick über dieses sehr aktuelle und sehr breite Forschungsgebiet, und die dabei auftretenden in der Praxis relevanten Teilprobleme. Viele dieser Probleme sind auch algorithmisch interessant, in der Algorithmenforschung aber wenig bekannt.

in einem ganz anderen. Und die Information, dass es eine Pflanze ist, noch einmal an anderer Stelle. Die Funktionalität, derart für einzelne Treffer Inhalte aus mehreren Dokumenten zu kombinieren, ist vollkommen jenseits dessen, was Volltextsuche zu leisten in der Lage ist.

Das Ziel der semantischen Suche sind genau solche Suchanfragen jenseits der Möglichkeiten der Volltextsuche. Es gibt dabei keine einheitliche Definition, genau welche Art von Suchanfragen eine solche Suche unterstützen soll, aber eine Anfrage von der Art *plants with edible leaves and native to europe* ist typisch.

Fazit: Semantische Suche erfordert eine Funktionalität jenseits dessen, was Techniken zur Volltextsuche zu leisten im Stande sind: Suche nach Entitäten einer gegebenen Klasse anstatt nach Worten, Diversität der Ergebnisse, pro Treffer Kombination von Informationen aus verschiedenen Quellen.

Semantische Suche

Folgende Probleme treten bei der im vorangegangenen Abschnitt motivierten und lose definierten semantischen Suche auf.

(1) Wie kommt man an sogenanntes *ontologisches* Wissen, zum Beispiel, dass *Broccoli* eine

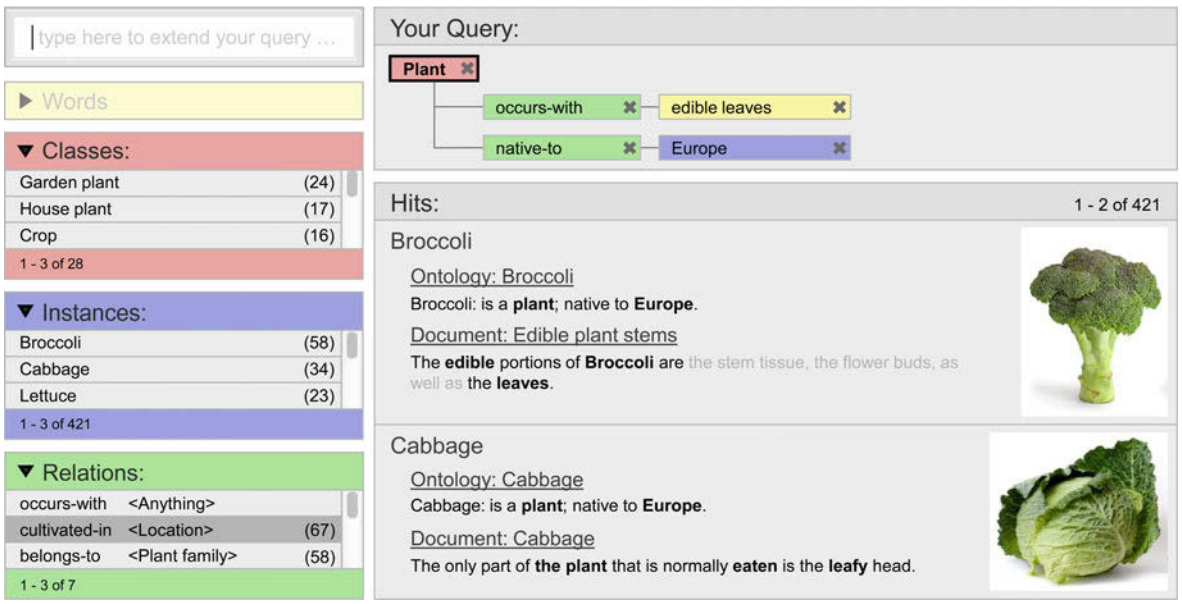

Abb. 1 Die semantische Suchmaschine* Broccoli *in Aktion für die im Text wiederholt erwähnte Suchanfrage* plants with edible leaves and native to europe. *Die aktuelle Suchanfrage wird in dem Feld oben rechts als Baum grafisch dargestellt. Getippt wird ausschließlich in das Suchfeld oben links. In den vier Boxen darunter werden, nach jedem Tastendruck, Vorschläge für die Erweiterung der Suchanfrage gemacht (die Farben der Boxen entsprechen den Farben der Knoten im Suchanfragebaum). Ein Klick auf einen der Vorschläge erweitert die Suchanfrage entsprechend. Einer der Vorschläge ist immer grau hervorgehoben. Möchte man diesen übernehmen, reicht ein Druck auf die Taste* Return. *Die Abbildung oben ist nur ein Schnappschuss. Der wohl schnellste Weg die Suchmaschinen und die wesentlichen Prinzipien dahinter zu verstehen ist unsere Online-Demo unter broccoli.informatik.uni-freiburg.de

Abstract
Classical full-text search looks for occurrences of the query words in a given text collection. This approach works well for many queries, but also has its obvious limits. Semantic search aims at really "understanding" both the query as well as the searched texts. This article gives an overview over this very active and broad research area, and the various sub-problems relevant in practice. Many of these problems are also algorithmically interesting, but little-known in traditional algorithms research.

Pflanze ist? Und wie repräsentiert und durchsucht man es?

(2) Wie erkennt man, dass ein bestimmtes Wort oder eine bestimmte Folge von Wörtern im Text eine bestimmte Entität meint. Selbst für ein vermeintlich eindeutiges Wort wie *broccoli* ist die Antwort nicht klar, denn es gibt beispielsweise zahlreiche Persönlichkeiten mit diesem Nachnamen und ein so benanntes japanisches Medienunternehmen. Nicht zu vergessen die gleichnamige Suchmaschine.

(3) Wie erkennt man, welche Worte in einem Dokument semantisch „zusammen gehören". Betrachten wir dazu unsere Anfrage von oben und den Satz *The stalks of rhubarb are edible, but its leaves are toxic*. Dieser enthält den Namen einer Pflanze und die Worte *edible* und *leaves*. Rhabarber ist aber gerade kein Treffer, denn *edible* bezieht sich nur auf die Stängel, während die *leaves* giftig sind.

(4) Wie integriert man alle diese Informationen in einen *Suchindex*, der nicht zu viel Platz verbraucht und die von der Volltextsuche gewohnten schnellen Antwortzeiten (wenige 100 Millisekunden, unabhängig von der Größe der Datenmenge) ermöglicht.

(5) Ein wesentliches Element einer Benutzerschnittstelle für semantische Suche ist, dass nicht nur Trefferentitäten angezeigt werden, sondern auch eine Begründung, warum die entsprechende Entität als Treffer erkannt wurde. Das ist ähnlich zu den Textauszügen (engl.: result snippets) bei der Volltextsuche, nur bei der semantischen Suche weitaus komplexer.

Jeder der folgenden vier Abschnitte wird sich mit einem dieser Probleme beschäftigen. Insbesondere wird dabei auf die algorithmisch interessanten Aspekte eingegangen (die sonst in dieser Art Forschung häufig zu kurz kommen).

Abbildung 1 zeigt unsere eigene neue semantische Suchmaschine „Broccoli" in Aktion für die Suchanfrage *plants with edible leaves and native to europe*. In diesem Prototypen sind alle im Artikel vorgestellten Konzepte berücksichtigt. Für ein unmittelbares intuitives Verständnis empfehlen wir unsere Online-Demo unter http://broccoli.informatik.uni-freiburg.de. Details zur Realisierung sowie weiterführende Informationen und Literatur finden sich in [5].

Fazit: Folgende Kernprobleme gilt es bei der semantischen Suche zu lösen: Ontologieerstellung und -suche, Entitätserkennung, semantische Sprachanalyse der Volltexte, ein Suchindex mit möglichst wenig Platzverbrauch und schnellen Anfragezeiten, sowie eine intuitive Benutzerschnittstelle mit aussagekräftigen Textauszügen für jeden Treffer.

Ontologien

Für die Zwecke dieser Arbeit sei eine Ontologie einfach eine Menge von Subjekt-Prädikat-Objekt-Tripeln, auch *Fakten* genannt. Es ist dabei wichtig, dass die gleiche Entität bzw. Relation in verschiedenen Fakten auch gleich benannt ist, dazu mehr in Abschn. „Entitätserkennung". Drei solcher Fakten sind zum Beispiel: *Broccoli is-a Vegetable* und *Vegetable is-subclass-of Plant* und *Broccoli is-native-to Europe*.

Der Vorteil einer Ontologie (mit einem hinreichenden Schatz an Fakten) ist, dass sich damit auch komplexe Anfragen präzise beantworten lassen. So erfordert die Anfrage *plants native to europe* die Verknüpfung aller drei Fakten aus dem vorherigen Absatz.

Manuell erstellte Ontologien

Es stellt sich die Frage, wie man an solche Fakten kommt. Im besten Fall stehen sie bereits in einer Datenbank, aus der sie sich leicht im gewünschten Format extrahieren lassen. Das ist bei von Natur aus stark strukturierten Daten der Fall. Zum Beispiel alle Mitarbeiter eines Unternehmens oder einer Behörde, und für jeden: Name, Adresse, Telefonnummern und berufliche Stellung. Oder eine Datenbank wie www.geonames.org, die das Wissen über alle geografische Örtlichkeiten der Welt enthält, zum Beispiel, welche Stadt in welchem Land liegt. Oder die Fakten aus den Infoboxen der Wiki-

pedia, die als *DBpedia* strukturiert zur Verfügung stehen [3].

Darüber hinaus gibt es inzwischen zahlreiche Formate, in denen Benutzer Fakten direkt im Rahmen einer Webseite formulieren können. Die etabliertesten davon sind, nach ihrem „Alter" geordnet: RDFa, Microformats und Microdata [10]. Zwei typische Verwendungszwecke sind: (1) Angaben zu Name, Telefonnummer, Institution auf einer persönlichen Webseite; (2) Angabe von Metainformationen zu einer Webseite oder einem Produkt für strukturierte Informationen in den Trefferlisten von Suchmaschinen, zum Beispiel mittels *Google Rich Snippets* [9].

Schließlich gibt es die große Intiative der *Linked Open Data (LOD)* [6]. Diese hat zum Ziel, ontologisches Wissen (sprich: Fakten) möglichst einheitlich und in einem möglichst einfachen Format (basierend auf RDF und HTTP) zur Verfügung zu stellen. Außerdem sind Verweise zwischen auf verschiedenen Seiten zur Verfügung gestellten Fakten möglich und erwünscht, ähnlich zu den Links zwischen gewöhnlichen Webseiten.

Fazit: Von Natur aus stark strukturierte Daten stehen bereits in großer Menge als Ontologien zur Verfügung. Darüber hinaus gibt es diverse Initiativen, die manuelle Eingabe ontologischer Daten zu ermöglichen, und alle diese Daten auch miteinander zu verknüpfen.

Informationsextraktion

Trotz dieser Möglichkeiten und Anstrengungen ist aber absehbar, dass auch auf lange Sicht ein großer (und wahrscheinlich der größte) Teil aller Informationen ausschließlich in Form von natürlichsprachigem Text zur Verfügung steht. Aus dem einfachen Grund, dass dies die primäre Kommunikationsform zwischen Menschen ist. Zahlreiche Forschungsarbeiten beschäftigen sich darum mit der *Informationsextraktion*, der Gewinnung von Faktenwissen (im obigen Sinne) aus gewöhnlichem Text bzw. Webseiten im Allgemeinen [14].

Die einfachsten Methoden basieren dabei auf einfacher Mustererkennung [2]. Steht zum Beispiel in einem Satz *<word 1> , the capital of <word 2>*, so ist anzunehmen, dass *<word 1>* eine Stadt ist, und *<word 2>* ein Land, und erstere die Hauptstadt von Letzterem ist. Findet man diese Sequenz öfter, ist die Wahrscheinlichkeit hoch, dass die Information auch korrekt ist.

Ein zentrales Problem schon bei diesem einfachen Ansatz ist das der bereits erwähnten *Entitätserkennung*; dazu mehr im Abschn. „Entitätserkennung". Ein zentrales Problem fortgeschrittenerer Ansätze ist die Erkennung von tieferen Satzstrukturen und von deren semantischen Zusammenhang. Dazu mehr im Abschn. „Sprachverstehen".

Ein weiteres großes Problem sind einheitliche Namen für die Entitäten. Wie bereits bei unserer Definition am Anfang von Abschn. „Ontologien" erwähnt, erfordern typische Ontologiefragen die Kombination mehrerer Fakten. Das kann nur funktionieren, wenn ein und dieselbe Entität in verschiedenen Fakten auch genau gleich benannt ist.

Bei der verteilten Bereitstellung von Ontologiedaten, wie bei der *Linked Open Data (LOD)*, sind a priori vereinheitlichte Namen aber unrealistisch. Es steht dazu eine spezielle Relation zur Verfügung, *owl:sameAs*[2], mit der Benutzer explizit machen können, dass zwei verschieden benannte Entitäten genau dasselbe meinen. Mit demselben Argument wie in Abschn. „Manuell erstellte Ontologien" wird diese explizite Information aber immer inhärent unvollständig sein.

Automatische Informationsextraktion ist ein schweres Problem. In der Benchmarkserie ACE (*Automatic Content Extraction*) wurde zuletzt 2008 die Performanz der besten Verfahren für sieben vorgegebene Relationen (zum Beispiel: *<person> married-to <person>*) auf diversen Textkorpora (zum Beispiel: Nachrichtentexten) gemessen. Die einheitliche Benennung von Entitäten war dabei ein Teil des Problems. Die besten Systeme erreichten dabei eine Erfolgsrate von lediglich 50 % [1].

Fazit: Die vollautomatische Extraktion von Fakten (in unserem Sinne) aus natürlich-sprachlichem Text ist ein sehr schweres Problem. Die Erfolgsrate der zurzeit besten Systeme liegt nur bei etwa 50 %.

Entitätserkennung

Bei der Entitätserkennung gilt es, in einem gegebenen Text Referenzen auf Entitäten aus einer (implizit oder explizit) gegebenen Ontologie zu erkennen. Nehmen wir zum Beispiel die bereits in Abschn. „Manuell erstellte Ontologien" erwähnte DBpedia als Referenz-Ontologie und

[2] OWL = Web Ontology Language, eine Spezifikation des World Wide Web Consortiums (W3C) für formale Sprachen zur Beschreibung von Ontologien.

betrachten wir den Satz *The stalks of rhubarb, a plant native to Eastern Asia, are edible, however, its leaves are toxic.* Darin sollten dann sowohl *rhubarb* als auch das Pronomen *its* als dbpedia.org/resource/Rhubarb erkannt werden, und *Eastern Asia* als dbpedia.org/resource/East_Asia.

Bei der Entitätserkennung gilt es insbesondere folgende Teilprobleme zu lösen:

(1) Manche Namen/Bezeichnungen bestehen aus vielen Wörtern, zum Beispiel *Eastern Asia* oben. Bei welchem Wort fängt der Name/die Bezeichnung an und bei welchem hört sie auf? Das Problem wird dadurch erschwert, dass Teile des Namens ganz andere Entitäten bezeichnen können. Zum Beispiel könnte *Eastern* ja auch eine Variation von *Easter* (Ostern) sein. Bei einer großen Referenz-Ontologie ist dieses Problem bei Mehrwortbezeichnung nicht die Ausnahme sondern die Regel.

(2) Oft findet sich im Text nur ein Teil des vollen Namens der Entität, zum Beispiel nur der Nachname einer Person. Das ist insbesondere häufig, wenn die Entität vorher schon einmal mit vollem Namen bezeichnet wurde.

(3) Verweise sind oft mehrdeutig und nur aus dem Kontext heraus eindeutig einer Entität zuzuordnen. Ein Beispiel wären wieder Nachnamen von Personen. Aber wie wir in Abschn. „Semantische Suche“ schon gesehen haben, sind bei genügend großen Textmengen selbst so vermeintlich eindeutige Worte wie *Broccoli* stark mehrdeutig.

Im Vergleich zur vollständigen Informationsextraktion ist die Entitätserkennung ein relativ einfaches Problem. Bei den erwähnten ACE Benchmarks wurde regelmäßig eine Qualität von etwa 90 % erreicht [1]. Die Effizienz ist bei diesen Arbeiten aber oft sekundär oder wird gar nicht beachtet, mit wenigen Ausnahmen [16]. Dabei spielt sie bei der Verarbeitung sehr großer Textmengen eine große Rolle.

Fazit: Entitätserkennung ist mit einer hohen Genauigkeit von etwa 90 % lösbar. In puncto Effizienz und Skalierbarkeit auf sehr große Datenmengen gibt es aber noch großen Forschungsbedarf.

Suchindexe

Sowohl bei der Volltextsuche als auch bei der Ontologiesuche sind die Datenmengen in der Regel sehr groß. Beispielsweise liegt die (geschätzte) Anzahl der von Google indizierten Webseiten derzeit bei etwa 50 Milliarden (Quelle: www.worldwidewebsize.com) und die Anzahl der Tripel im BTC 2012 Datensatz, dem derzeit größten seiner Art[3], bei etwa 1,4 Milliarden (Quelle: http://km.aifb.kit.edu/projects/btc-2012).

Für beide Arten von Suchen ist dabei ein vorberechneter *Suchindex* vonnöten. Vorrangige Effizienzkriterien sind dabei die Zeit für eine Anfrage und der Platzverbrauch des Suchindexes (im Verhältnis zur indizierten Datenmenge). Für beide Arten von Suchen gibt es bereits Suchindexe mit schnellen Anfragezeiten und verhältnismäßig geringem Platzverbrauch [4, 11].

Für die semantische Suche ist eine Kombination dieser beiden Arten von Suchen erforderlich. Viele Ansätze teilen die Suche dabei einfach in einen Volltextteil und einen Ontologieteil auf. Insbesondere gilt das für praktisch alle Verfahren, die an der TREC Benchmark Serie zur Entitätssuche teilgenommen haben [15]. Dieser einfache Ansatz erlaubt es, für die jeweiligen Suchen auf Standardverfahren zurückzugreifen. Er ist aber für die Anfragezeit aus zwei Gründen problematisch.

(1) Um die Ergebnisse der beiden Teilsuchen zu verbinden, müssen in den Ergebnissen der Volltextsuche Entitäten aus den Ergebnissen der Ontologiesuche erkannt werden, und zwar *zur Anfragezeit*. Das ist teuer, vor allem wenn die Anzahl der Treffer der Volltextsuche groß ist.

(2) Selbst wenn das Ergebnis der kombinierten Suche relativ klein ist, können die Ergebnismengen der Teilsuchen sehr groß, und sie zu materialisieren sehr teuer sein.

Für unsere eigene Suchmaschine *Broccoli* lösen wir deshalb das Entitätserkennungsproblem in der Vorberechnung, und wir bauen spezielle *hybride Indexlisten* für Vorkommen von sowohl Wörtern als auch Entitäten. Die Schilderung der Details unseres Ansatzes würde den Rahmen dieses Artikels sprengen, siehe [5]. Für die englische Wikipedia ergibt das einen Suchindex, der etwa vier mal so groß ist wie für gewöhnliche Volltextsuche mit Anfragezeiten unter 100 Millisekunden. Nach unserem Wissen ist das der derzeit mit Abstand effizienteste Index für diese Art von Suche.

Wir sehen allerdings noch viel Verbesserungspotenzial, in Hinblick auf sowohl Platzverbrauch

[3] Die BTC (*billion triple challenge*) Datensätze bestehen aus einer Mischung der erwähnten LOD-Daten, Daten aus den in Abschn. „Manuell erstellte Ontologien“ erwähnten eingebetteten Formaten, und Fakten aus fertig strukturierten Datensätzen wie geonames.org und dbpedia.org.

als auch Anfragezeit. Standardtechniken für die Volltextsuche (wie komprimierte Indexlisten und Caching-Strategien) sind dabei ein guter Ausgangspunkt, können aber aufgrund des komplexeren Problems nicht 1 : 1 übernommen werden.

Fazit: Semantische Suche mit schnellen Anfragezeiten benötigt spezielle Indexdatenstrukturen. Dazu gibt es bisher noch kaum Forschungsarbeiten.

Sprachverstehen

Die gewöhnliche Volltextsuche ist auch 2012 noch ohne tieferes Sprachverständnis sehr erfolgreich. Wie in Abschn. „Volltextsuche" erklärt, wird ja nach wie vor einfach nach Vorkommen der Schlüsselwörter, oder leichten Variationen davon, gesucht. Und die (in der Regel sehr langen) Ergebnislisten werden nach der dort beschriebenen „Prominenz" geordnet.

Dies funktioniert bei sehr großen Textmengen und populären Suchanfragen sehr gut (also gerade bei dem Markt, den Websuchmaschinen bedienen). Der Grund dafür ist, vereinfacht gesprochen, dass es zu hinreichend populären Suchanfragen nicht nur eine, sondern sehr viele relevante Dokumente gibt, und die Wahrscheinlichkeit groß ist, dass es einige Dokumente gibt, die sehr genau zu dem gesuchten Thema passen und genau die eingegebenen Schlüsselwörter (oder leichte Variationen) davon prominent enthalten, das heißt im Titel und/oder einer Überschrift und/oder überproportional häufig und/oder in der Nähe voneinander.

Wichtig zu verstehen ist auch, dass es bei der Volltextsuche, insbesondere auf dem Web, meistens nicht wichtig ist *alle* relevanten Dokument zu finden. Das heißt, der sogenannte *recall* (Prozentsatz aller relevanten Dokumente, die gefunden werden) ist sekundär. Viel wichtiger ist, dass die Dokumente in den oberen Rängen relevant sind. Das heißt, die *precision* unter den ersten Treffern[4] (Prozentsatz der relevanten Dokumente unter diesen) sollte möglichst hoch sein.

Ganz anders verhält sich dies bei einer typischen semantischen Suchanfrage. Nehmen wir wieder unsere Beispielanfrage *plants with edible leaves*. Die Anzahl der relevanten Entitäten (Pflanzen mit essbaren Blättern) ist hier überschaubar; in der gesamten englischen Wikipedia finden sich beispielsweise nur an die 100 Treffer. Durch zusätzliche ontologische Einschränkungen, wie zum Beispiel *native to Europe*, wird die Treffermenge noch kleiner.

Jetzt ist *recall* wichtig: man möchte möglichst alle relevanten Entitäten finden. Die *precision* ist zwar nicht egal, aber nicht mehr so zentral. Es wäre durchaus akzteptabel, wenn beispielsweise die Hälfte der Treffer nicht relevant wäre. Vorausgesetzt, es werden ähnlich wie bei der Volltextsuche Textauszüge (sogenannte *result snippets*) angezeigt, mit Hilfe derer sich das leicht feststellen lässt; dazu mehr in Abschn. „Benutzerschnittstelle".

Außerdem müssen jetzt für *jede* der relevanten Entitäten Treffer gefunden werden. Für populäre Trefferentitäten, im Fall unserer Suchanfrage etwa *cabbage* (Kohl), gibt es – ähnlich wie bei unserer Diskussion zu populären Suchanfragen oben – viele Treffer, und dann auch einige, die die Worte *edible leaves* enthalten. Für exotischere Trefferentitäten, wie etwa *malva* (Malve), gibt es aber insgesamt nur sehr wenige Treffer.

Jetzt wird Sprachverständnis erforderlich, wie folgendes Beispiel zeigt. Nehmen wir wieder den Satz *The stalks of rhubarb are edible, however, its leaves are toxic*. In diesem Satz stehen eine Pflanze, *rhubarb* (Rhabarber), und die Wörter *edible* und *leaves*. Der Rhabarber ist aber gerade kein Treffer, denn *edible* bezieht sich auf *stalks* und nicht auf *leaves* und die Blätter sind in dem Fall sogar giftig. Nun ist eine solche zufällige Ko-Okkurenz der Suchwörter unwahrscheinlich. Würde man sich auf Entitäten mit vielen Treffern beschränken, könnte man solche falschen Treffer sehr effektiv aussortieren. Man würde dann aber auch die richtigen Treffer für viele der seltenen Pflanzen verlieren.

Fazit: Anders als bei der Volltextsuche muss bei der semantischen Suche für qualitativ hochwertige Ergebnisse verstanden werden, welche Worte in einem Text semantisch „zusammen gehören". Dazu braucht es ein gewisses Maß an Sprachverständnis.

Semantischer Kontext

Um dieses Problem zu lösen, muss der *semantische Kontext* der Wörter beachtet werden. Denn wie das obige Beispiel zeigt, gehört nicht alles, was in einem Satz steht auch faktisch zusammen. Für unsere eigene Suchmaschine lösen wir das Problem durch zwei Maßnahmen:

(1) Wir berücksichtigen bei der Entitätserkennung auch sogenannte Anaphora, im Englischen etwa *he*, *she*, *it*, *his*, *her*, *its*, etc. Diese verweisen ja

[4] Häufig betrachtet man die ersten 10 oder 100 Treffer.

auf eine vorher genannte Entität, und wir wollen sie dann auch als diese Entität erkennen. In unserem Beispielsatz *The stalks of rhubarb are edible, however, its leaves are toxic* soll zum Beispiel auch das Pronomen *its* als die Pflanze Rhabarber erkannt werden. Berücksichtigt man bei der Entitätserkennung Anaphora, kann man davon ausgehen, dass die meisten in einem Text enthaltenen Fakten jeder für sich innerhalb von ein- und demselben Satz stehen.

(2) In einem Satz können aber, wie das obige Beispiel zeigt, mehrere Fakten stehen. Wir lösen dies, indem wir jeden Satz in seine „faktischen Bestandteile" zerlegen. In unserem Beispiel wären das, nach der Auflösung der Anaphora: *The stalks of rhubarb are edible* und *however, rhubarb leaves are toxic*.

Zurzeit lösen wir das Problem noch auf der Basis eines sogenannten *full parse* für jeden Satz. Ein solcher *full parse* zerlegt einen Satz in seine kleinsten grammatikalischen Bestandteile (zum Beispiel: *noun phrases* = *NP*, *verb phrases* = *VP*, *prepositions* = *PP*, usw.) und stellt sie in einer Baumstruktur in Beziehung. Aus Platzgründen sei hier zum weiteren Verständnis einfach die Online-Demo des Stanford Parsers, der qualitativ dem Stand der Kunst entspricht, empfohlen: http://nlp.stanford.edu:8080/parser. Auf der Basis eines *full parse* können wir die „faktischen Bestandteile" eines Satzes mit Hilfe einer relativ kleinen Menge von Regeln mit hoher Genauigkeit erkennen. Für Details, siehe wieder [5].

Ein *full parse* ist allerdings sehr teuer. So benötigt der Stanford Parser pro Satz im Durchschnitt etwas über eine Sekunde. Für die etwa 250 Millionen Sätze der englischen Wikipedia wären das etwa acht Jahre. Selbst parallelisiert auf 100 Rechenkernen würde es noch einen ganzen Monat dauern. Außerdem sind *full parses* sehr fehleranfällig, vor allem bei langen Sätzen. Das ist für die semantische Suche besonders problematisch, weil gerade in langen Sätzen sehr viele Wörter nur „zufällig" aber nicht im „faktischen Zusammenhang" miteinander vorkommen, und diese somit zu sehr vielen falschen Treffern führen können.

Zur Ermittlung der faktischen Bestandteile ist aber gar kein vollständiger *full parse* vonnöten. Ein wichtiges offenes Problem ist daher, einen gegebenen Satz direkt (ohne den Umweg über einen *full parse*) in seine faktischen Bestandteile zu zerlegen. Ehrgeiziges Ziel dabei wäre eine Verarbeitungszeit von wenigen Millisekunden pro Satz, bei einer Zerlegungskorrekheit von über 80 %.

Fazit: Ein korrekter full parse *ist hinreichend zur Berechnung von „semantischem Zusammenhang", aber sehr teuer. Eine direktere Lösung des Problems mit einer Rate von wenigen Millisekunden/Satz wäre wünschenswert und aus unserer Sicht möglich.*

Benutzerschnittstelle

Die Benutzerschnittstelle ist ein zentrales Element jeder Suchmaschine. Für die semantische Suche gilt das aufgrund der Komplexität der Anfragen erst recht. Die Standardanfragesprache für die Ontologiesuche, die ja ein Teil der semantischen Suche ist, ist SPARQL [13]. SPARQL ist, sowohl vom Namen als auch von der Syntax her, stark an die Datenbankanfragesprache SQL angelehnt. Die Sprache ist damit sehr mächtig, aber auch sehr kompliziert. Als Anfragesprache für gewöhnliche Benutzer ist sie nicht geeignet, und auch für geschulte Benutzer ist sie äußerst unkomfortabel.

Beim sogenannten *ad-hoc object retrieval* [12] wird dagegen einfach das Prinzip der Schlüsselwortsuche von der Volltextsuche auf die semantische Suche übertragen. Eine Benutzerin würde dann zum Beispiel einfach *plants with edible leaves and native to europe* so eingeben. Das macht es der Suchmaschine allerdings sehr schwer, die semantischen Bezüge zu erkennen.

In unserem eigenen Prototypen gibt es wie bei der Volltextsuche nur ein einziges Suchfeld. Während des Tippvorgangs wird die Benutzerin allerdings (durch proaktive Vorschläge) dabei unterstützt, die semantische Struktur ihrer Suchanfrage explizit zu machen. Siehe dazu Abb. 1 und die Erklärungen dazu. Am besten lässt sich diese Benutzerschnittstelle durch Ausprobieren verstehen, siehe dazu die erwähnte Online-Demo unter broccoli.informatik.uni-freiburg.de.

Fazit: Eine hinreichend mächtige und trotzdem einfache Benutzerschnittstelle ist für semantische Suche von besonderer Bedeutung.

Textauszüge (result snippets)

Wir möchten im Rahmen dieses Artikel ein algorithmisch besonders interessantes Teilproblem herausgreifen: die Anzeige von sogenannten *results snippets*. Das sind Textpassagen, die zeigen, warum eine Entität in der Trefferliste steht. Bei unserer Suchanfrage *plants with edible leaves* wäre dies zum

Beispiel ein Teilsatz in dem eine (Referenz auf eine) Pflanze zusammen mit den Wörtern *edible* und *leaves* vorkommt. Siehe dazu auch wieder Abb. 1 sowie unsere Online-Demo.

Ein Suchindex gibt normalerweise nur die IDs der Treffer(-Entitäten) zurück. Insbesondere gilt dies für den Ansatz, bei dem je ein Standardindex für die Volltextsuche und die Ontologiesuche benutzt wird. Für passende *result snippets* muss dann noch für jede angezeigte Trefferentität in (je nach Suchanfrage) mehreren Dokumenten gesucht werden [7]. Für unseren eigenen Prototypen haben wir die Indexdatenstruktur so modifiziert, dass die Informationen über die Positionen der relevanten *result snippets* bereits vom Index zurückgeliefert werden. Dies führt zu einem größeren Index und längeren Anfragezeiten, ist aber in der Summe um ein Vielfaches schneller als der naive Ansatz.

Auch hier kann aber noch optimiert werden. Interessant sind in diesem Zusammenhang Ideen aus den Arbeiten zu sogenannten *self indices* [8]. Dabei müssen die Originaltexte gar nicht mehr abgespeichert werden, sondern der Index enthält (in stark komprimierter Form) bereits sämtliche Informationen, nicht nur über die Positionen relevanter *result snippets*, sondern auch über die anzuzeigenden Textauszüge.

Fazit: Die effiziente Berechnung passender Textauszüge (result snippets) *bei der semantischen Suche ist zentral wichtig für eine gute Benutzerschnittstelle und ein interessantes algorithmisches Problem.*

Literatur

1. Automatic content extraction (ACE) (2004–2008) http://www.itl.nist.gov/iad/mig/tests/ace, letzter Zugriff 20.2.2013
2. Agichtein E, Gravano L (2000) Snowball: Extracting Relations from Large Plain-Text Collections. In: ACM DL, June 2–7, 2000, San Antonio, TX, USA, pp 85–94
3. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives ZG (2007) DBpedia: A Nucleus for a Web of Open Data. In: ISWC/ASWC, November 11–15, 2007, Busan, Korea, pp 722–735
4. Baeza-Yates RA, Ribeiro-Neto BA (2011) Modern Information Retrieval – The Concepts and Technology Behind Search, 2nd edn. Pearson Education Ltd, Harlow, England
5. Bast H, Bäurle F, Buchhold B, Haussmann E (2012) Broccoli: Semantic full-text search at your fingertips. CoRR abs/1207.2615, http://arxiv.org/abs/1207.2615, letzter Zugriff 20.2.2013
6. Bizer C, Heath T, Berners-Lee T (2009) Linked data – the story so far. Int J Semant Web Inf Syst 5(3):1–22
7. Blanco R, Zaragoza H (2010) Finding support sentences for entities. In: SIGIR, July 19–13, 2010, Geneva, Switzerland, pp 339–346
8. Fariña A, Brisaboa NR, Navarro G, Claude F, Places ÁS, Rodríguez E (2012) Word-based self-indexes for natural language text. ACM Trans Inf Syst 30(1):1
9. Google Rich Snippets (2009) http://www.google.com/webmasters/tools/richsnippets, letzter Zugriff 20.2.2013
10. Hausenblas M (2009) Anreicherung von Webinhalten mit Semantik – Microformats und RDFa. In: Social Semantic Web, ISBN 978-3-540-72215-1, Springer, pp 147–158
11. Neumann T, Weikum G (2008) RDF-3X: a risc-style engine for RDF. PVLDB 1(1): 647–659
12. Pound J, Mika P, Zaragoza H (2010) Ad-hoc object retrieval in the web of data. In: WWW, April 26–30, 2010, Raleigh, NC, USA, pp 771–780
13. Prud'hommeaux E, Seaborne A (2008) SPARQL query language for RDF. W3C recommendation, W3C. http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115, letzter Zugriff 20.2.2013
14. Sarawagi S (2008) Information extraction. Found Trend Datab 1(3):261–377
15. TREC Entity Track 2010 + 2011 (2011) http://trec.nist.gov/data/entity.html, letzter Zugriff 20.2.2013
16. Wang H, Liu Q, Penin T, Fu L, Zhang L, Tran T, Yu Y, Pan Y (2009) Semplore: A scalable IR approach to search the web of data. J Web Semant 7(3):177–188