Universität Leipzig SWT-Praktikum 2015/16

Recherchebericht

zum Projekt: Datensuche für Linked Data

Inhaltsverzeichnis

1. Begriffe

- *1.1 Linked Data*

  Linked Open Data (LOD) bezeichnet im World Wide Web frei verfügbare Daten, die per Uniform Resource Identifier (URI) identifiziert sind und darüber direkt per HTTP abgerufen werden können und ebenfalls per URI auf andere Daten verweisen. Idealerweise werden zur Kodierung und Verlinkung der Daten das Resource Description Framework (RDF) und darauf aufbauende Abfragesprachen wie SPARQL und die Web Ontology Language (OWL) verwendet, so dass Linked Open Data gleichzeitig Teil des Semantic Web ist. Die miteinander verknüpften Daten ergeben ein weltweites Netz, das auch als „Linked [Open] Data Cloud“ bezeichnet wird. Dort wo der Schwerpunkt weniger auf der freien Nutzbarkeit der Daten wie bei freien Inhalten liegt (Open Data), ist auch die Bezeichnung Linked Data üblich. Die miteinander verknüpften Daten ergeben ein weltweites Netz, das auch als „Linked [Open] Data Cloud“ bezeichnet wird.

- *1.2 Link Discovery*

  Die **Link Discovery** beschäftigt sich mit der Problemstellung zu einem Datensatz bereits existierende ähnliche Datensätze zu finden, um diese miteinander zu verlinken. Durch das Linken werden beide Datensätze mächtiger, da man sie um Informationen aus dem anderen Datensatz angereichert werden, so sollen möglichst konsistente und vollständige Datensätze geschaffen werden. Die Verlinkung wird über Klassen wie *"sameAs", "subClassOf", "subPropertyOf"* und ähnliche Beziehungen realisiert, oder indem man URIs zwei verschiedener Datensätze mit gleichen Themen vergleicht und unifiziert.

- *1.3 Indexierung*

  Bei der **Indexierung** werden dem Dokument Metadaten angehängt, durch die es beschrieben und somit maschinenlesbar gemacht wird. Diese Metadaten enthalten Deskriptoren, welche den Inhalt des Textes beschreiben (z.B. Stichwörter oder Schlagwörter).
  Man unterscheidet zwischen zwei Arten des Indexierens: zum einen die „Gleichgeordnete Indexierung“ (Deskriptoren werden gleichrangig dem betreffenden Dokument zugeordnet), zum anderen die Syntaktische Indexierung (speichert zusätzlich die syntaktische Beziehung der Deskriptoren zueinander).

- *1.4 Mapping*

  Ein Mapping wird bei den meisten Link Discovery Tools verwendet, um eine Ähnlichkeit von Daten bezüglich einer Relation zu prüfen. Gesucht ist die Menge $\{(s, t) \in S \times T : R(s, t)\}$ mit S (sources), T (targets) und $R(s,t)$ (eine Relation), um festzustellen welche Daten zueinander in Relation stehen und in welchem Maße. Dazu wird eine Ähnlichkeitsfunktion $m : S \times T \rightarrow [0, 1]$ erstellt, die **Mappings** (bezeichnet mit $\boldsymbol{M}$, wobei $M \subseteq S \times T \times [0, 1]$) werden verwendet, um die Ergebnisse der Ausführung dieser Ähnlichkeitsfunktion zu speichern. Ein Mapping kann ebenfalls zur Speicherung einer ganzen Link Specification verwendet werden.

- *1.5 Frameworks*

  Ein **Framework** ist ein sogenanntes Programmiergerüst, welches in der Softwaretechnik, vor allem im Bereich der objektorientierten Softwareentwicklung sowie bei komponentenbasierten Entwicklungssätzen, verwendet wird. Es handelt sich um kein fertiges Programm, sondern stellt einen Rahmen zur Verfügung, in dem bereits vorgefertigte Funktionen vorhanden sind. Das Framework definiert insbesondere den Kontrollfluss der Anwendung und Schnittstellen für konkrete Klassen, die vom Programmierer erstellt und registriert werden müssen. Das Ziel bei der Entwicklung und Nutzung von Frameworks ist im Allgemeinen die Wiederverwendung der „architektonischen Muster“, um den Aufwand der Modellierung der Grundstruktur zu erleichtern.

- *1.6 DB-pedia*
  Durch **DBpedia** werden strukturierte Informationen aus Wikipedia extrahiert, wodurch Web-Anwendungen zugänglicher gemacht werden. Als Quellen dienen Wikipedia-Artikel in verschiedenen Sprachen, wobei diese Artikel neben Fließtexten ebenfalls aus strukturierten Informationen, wie z.B Tabellen oder Weblinks bestehen, die extrahiert und als Basis für komplexe Fragen verwendet werden können. Diese Daten werden im RDF-Format gespeichert und über SPARQL ausgelesen.

- *1.7 Metadaten(-Extraktion)*
  Als **Metadaten** werden strukturiete Daten bezeichnet, die lediglich die Informationen über andere Daten oder gar Datensammlungen wie z.B. Datenbanken oder Dateien enthalten, aber nicht die Daten selbst. Somit können auch Angaben von Eigenschaften eines einzelnen Objektes (z.B. Name) als dessen Metadaten bezeichnet werden. Die Metadaten können zur Beschreibung von Informationsressourcen eingesetzt werden, wodurch diese besser auffindbar sind, und um Beziehungen zwischen Datensätzen herzustellen. Es erfolgt meist keine bewusste Trennung zwischen der Objekt-und Metadatenebene.
  Bei der **Metadaten Extraction** sollen neue zum Datensatz passende Metadaten gefunden werden um diese anzureichern. Dabei spielen die Eigenschaften und Klassen, die in den Metadaten des Datensatzes verwendet werden, eine wichtige Rolle. Diese werden verglichen um inhaltliche Übereinstimmungen zwischen Datensätzen festzustellen. Um aus einem RDF-Datensatz ein Metadatendokument zu generieren geht TAPIOCA wie folgt vor:
  1. Zähle alle URIs, die entweder "Classes" (vgl. Röder) oder "Properties" (vgl. Röder) sind.
  2. Entferne alle Classes und Properties, die keine Aussagen über das Thema des Datensatzes enthalten (also Vokubular aus OWL, RDFS, ... sind, vgl. Röder).
  3. Verwandele die URIs anhand von Labels in Schlagworte (siehe "Label retrieval", vgl. Röder)
  4. Entferne alle "Stopwords" (vgl. Röder)

  Eine URI kann hierbei als Klasse (Prädikate „rdf:type“ das Objekt ist eine Klasse / “rdfs:Class“ als Subjekt einer Klasse/ "rdfs:subClassOf" Objekt und Subjekt sind Klassen) oder als Eigenschaft (Prädikat: URI selbst/ "rdf:property/ "rdfs:subPropertyOf" Subjekt und Objekt sind Eigenschaften) auftreten. Außerdem wird die Anzahl aller verschiedener Entitäten sowie die Anzahl von Tripeln pro Klasse und Eigenschaft erfasst. Alle Metadaten die zum Vokabular von OWL, RDFS, RDF, SKOS und VOID gehören, werden wieder gelöscht um nur Metadaten mit inhaltlichen Informationen zum eigentlichen Thema des Datensatzes zu erhalten.

- *1.8 Ontologien*
  Ontologien gelten als wichtiger Bestandteil des Semantic Webs.
  Sie dienen als Mittel der Strukturierung und zum Datenaustausch, um bereits bestehende Wissensbestände zusammenzufügen, in diesen zu suchen und diese zu editieren um aus Typen von Wissensbeständen neue Instanzen zu generieren. Genau wie bei einer Datenbank gehören bei einer Ontologie die Regeln und Begriffe zusammen. Ontologien besitzen eine formale Beschreibung der Daten sowie Regeln über deren Zusammenhang. Diese Regeln ermöglichen es, Rückschlüsse aus den vorhandenen Daten zu ziehen, Widersprüche zu erkennen und fehlendes Wissen aus dem Vorhandenen zu ergänzen. Diese Rückschlüsse werden durch logisches Folgern abgeleitet. Zur Beschreibung von Ontologien zählen unter anderem die formalen Sprachen RDF-Schema und OWL.

- *1.9 URI*
  Ein **URI (uniform resource identifier)** besteht aus Zeichenfolgen und dient zur Identifizierung von Ressourcen. Die URI unterscheidet sich zur URL dadurch, dass sie nicht zwangsweise im Netzwerk erreichbar sein muss. Die URI lässt sich in 5 Teile unterteilen (wobei einige optional sind):
  Das Schema (eng. **scheme**) definiert den Kontext, bezeichnet den Typ des URI s (z.B. http) und wird mit einem Doppel Punkt abgeschlossen. Darauf folgt (optional) der Anbieter (eng. **authority**), welcher die Benutzerdaten des Zugreifenden enthält.

Der Pfad (eng. **path**) beinhaltet Angaben die mit dem Abfrageteil zur Identifizierung einer Ressource dienen, er wir mit einem „/„ eingeleitet.
Der Abfrageteil (eng. **query**), beinhaltet Daten zur Identifizierung, die durch Pfadangaben nicht eindeutig identifiziert werden können. Letztlich kann die URI auch auf eine Stelle in einer Ressource verweisen, was durch den (optionalen) Teil (eng. **fragment**) ermöglicht wird.

- *1.10 API*
  Die **API (application programming interface)** oder auch Programmierschnittstelle ermöglicht Kommunikation zwischen zwei Programmen und versetzt Programmierer und andere Dritte in die Lage, zusätzlich Software für das System zu entwickeln (Software development kits). Sie lässt sich in 4 verschiedene Typklassen einteilen: **Funktionsorientierte Klasse** ruft nur Funktionen auf und bekommt handle s und Rückgabewerte, welche die Kommunikation ausmachen. **Dateiorientierte Klasse** wird durch die Dateisystemaufrufe open, read, write und close angesprochen und dient zur Manipulation von Daten. Die **Objektorientierte Klasse** ist ein System und arbeitet auf der Basis von Typbibliotheken. Die **Protokollorientierte Klasse** verwaltet Protokolle, wobei sie zwischen algemeingültigen und spezifischen Protokollen unterscheidet.

- *1.11 OWL*
  Die **Web Ontology Language** (kurz OWL) ist eine Spezifikation des World Wide Web Consortiums (W3C), um Ontologien anhand einer formalen Beschreibungssprache erstellen, publizieren und verteilen zu können. Sie klassifiziert im Semantic Web Informationen und bildet dadurch eine Hierarchie.

- *1.12 CBL*
  Durch die **Contextual Browsing Language** (CBL) werden im Semantic Web Informationen in eine Relation zueinander gestellt und deren Verknüpfungen gewichtet.

- *1.13 TURTLE*
  Turtle (Terse RDF Triple Language) ist eine Serialisierung für RDF-Graphen. Es ist unter Semantic Web-Entwicklern verbreitet, weil es als benutzerfreundliche Alternative zu RDF/XML gilt.

- *1.14 Web-Crawler*
  Ein **Webcrawler** ist ein Computerprogramm, das automatisch das World Wide Web durchsucht und Webseiten analysiert. Webcrawler werden vor allem von Suchmaschinen eingesetzt. Weitere Anwendungen sind das Sammeln von Web-Feeds, E-Mail-Adressen oder von anderen Informationen. Webcrawler sind eine spezielle Art von Bots, d. h. Computerprogrammen, die weitgehend autonom sich wiederholenden Aufgaben nachgehen.

2. Konzepte

- *2.1 Semantic Web*
  Das **Semantic Web** ist eine Erweiterung des bestehenden Web (2.0), bei der die Informationen und Daten zueinander in Beziehung gesetzt werden, diese Beziehungen sollen vom Computer eigenständig ausgewertet. Um dies zu ermöglichen müssen die Daten in einer strukturierten Form aufgearbeitet werden, um sie für Maschinen interpretierbar zu machen (siehe RDF/OWL/CBL).
  Das Semantic Web besitzt dafür eine Metadatenebene, auf der die Informationen in ihrer Semantik beschrieben werden. Dies ermöglicht z.B. einen besseren Umgang mit natürlichsprachlichen Eingaben.

- *2.2 TAPIOCA*

  TAPIOCA ist eine Engine zur Suche von RDF-Daten. TAPIOCA besitzt einen Index aller ihm bekannten Datensätze aus der LOD-Cloud und generiert aus den RDF's ein Modell, welches die Themen des RDFs beschreibt. Dieses Modell ordnet jedem RDF-Datansatz einen einzigartigen „Fingerprint" (Themenvektor) zu. Das Erstellen des Fingerprints geschieht, indem aus den bereits vorhandenen RDF's die Metadaten extrahiert werden und diesem Metadata Document auf Basis der enthaltenen Schlagworte ein Themenvektor zugeordnet werden kann. Dies wird über ein LDA-Modell realisiert. Für die Suche wird nun folgendes Prinzip angewandt: aus dem Eingabedatensatz werden zunächst die Metadaten extrahiert, anschließend wird für diese Metadaten ein Themenvektor berechnet. Dieser Themenvektor wird nun paarweise mit allen Themenvektoren im Index der Suchmaschine verglichen. Ist die Ähnlichkeit groß genug, so ist ein Match gefunden.

- *2.3 RDF*

  Das Dateiformat **RDF (Resource Description Framework)** ist der Baustein des Semantik Webs und wird als Tripel (Subjekt, Prädikat, Objekt) präsentiert. Die Tripel werden in RDF-Datein abgelegt, welche auf XML oder TURTLE basierende Textdateien sind. Diese Tripel sind Formulierungen von logischen Aussagen über Ressourcen. Eine Ressource stellt eine Information dar (z.B. Bild, Webseite) und kann durch eine zugewiesene URI eindeutig identifiziert werden. Die Menge der Tripel kann in einem RDF-Modell dargestellt werden, die meist genutzte Anfragesprache ist SPARQL (Beispiele sind zu finden im „Semantik Web Crashkurs").

- *2.4 XML*

  Die **Extensible Markup Language** ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien. XML wird u.a. für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt, insbesondere über das Internet. XML wurde eine Zeit lang als Standard zur Speicherung von RDF Dateien verwendet, wird aber vorraussichtlich von TURTLE abgelöst, da XML mit seiner strikten Baumstruktur nicht unbedingt den Anforderungen von RDF entspricht.

- *2.5 SPARQL*

  SPARQL ist eine Graph-basierte Anfragesprache, welche genutzt wird, um aus RDF-Dokumenten einen RDF-Graphen bzw. Teilgraphen abzuleiten. Die Abfrageweise von SPARQL funktioniert wie die einer SQL-Abfrage, wobei SPARQL jedoch RDF-Daten im Web findet und diese nutzt, um eine konkrete Antwort auf die Anfrage zu geben.
  Die Syntax orientiert sich an der Turtle-Notation von RDF-Dokumenten. Die RDF-Tripel werden in der Reihenfolge Subjekt, Prädikat, Objekt notiert und durch einen Punkt abgeschlossen. URIs sind dabei in spitzen Klammern und Literale in Anführungszeichen zu setzen. Genutzte Variablen werden durch $ oder ? gekennzeichnet, wobei diese als Subjekt, Prädikat oder Objekt zulässig sind.
  Durch SPARQL wird es ermöglicht, komplexere Anfrage zu stellen, als es eine reine Textsuchmaschine kann. Das Ziel von *SPARQL* ist es, die Inhalte des Internets nicht nur maschinenlesbar zu machen, sondern auch maschinenverständlich.

- *2.6 MVC*

  Das Team einigte sich auf das **Model-View-Controller Konzept**, eines der beliebtesten Architekturmodelle der objektorientierten Programmierung. Eine klare Rollenverteilung des MVC-Konzeptes in 3 Einheiten, ein **model** ***(observable** – Verhalten von view und controller sind von Änderungen im model abhängig)* realisieriert die Kernfunktionalität, ein **view** (***observer** – überwacht das model und reagiert auf Veränderungen)* ist die Schnittstelle für die Bildschirmpräsentation und ein **controller** ***(observer** – reagiert auf Veränderungen des models, Eingabeschnittstelle zwischen Benutzer und model)* verwaltet die Benutzereingaben. Dies ermöglicht eine strikte Trennung von Verarbeitung, Präsentation und Datenmanipulation und dient nicht nur der Besseren Wartbarkeit, sondern ermöglicht auch eine einfache Erweiterung des Programms, da dem model mehrere views und mehrere controller zur Verfügung gestellt werden können. Für den Überwachungsmechanismus stellt java die Klassen java.util.Observable (vererbt an zu überwachende Klassen) und java.util.Observer (überwachende Klassen implementieren das Interface Observer) zur Verfügung.

- *2.7 Gold Standard*

  Um die Eindeutigkeit und Konsistent der Referenzstrukturen zu gewährleisten, müssen einige feste Referenzen etabliert werden, anhand derer verglichen werden kann. Solche feste Referenzen werden als „**Goldstandard**“ bezeichnet. Im Kontext des DSL-16 Projektes, wurde für das zugrundeliegende Projekt TAPIOCA mit 1680 Datensätzen aus der LOD-Cloud gearbeitet. Von diesen wurden zufällig 100 Datensätze ausgewählt und unabhängig von 2 verschiedenen Testpersonen auf thematische Ähnlichkeit überprüft. Anschließend wurden die TAPIOCA Ergebnisse mit den Ergebnissen der menschlichen Link-Discoverer verglichen um so festzustellen, wie gut Tapioca unter verschiedenen Bedingungen arbeitet. Um diesen „Goldstandard“ zu quantifizieren, wird ein Messwert festgelegt, in diesem Fall der sogenannte **F1-Score**. Ist der F1-Score nahe 0, hat TAPIOCA den Gold Standard stark verfehlt, ist der Messwert nahe 1, hat TAPIOCA den Gold Standard gut approximiert.

- *2.8 Topic Model*

  Topic Models bieten eine effiziente Möglichkeit, große Mengen von Text zu analysieren. Es gibt viele verschiedene Arten von Topic Models, das häufigste und nützlichste Topic Model für Suchmaschinen ist die Latent Dirichlet Allocation, welche im nächsten Punkt ausgeführt wird.

- *2.9 LDA Latent Dirichlet Allocation*

  Die **Latent Dirichlet Allocation** (LDA) ist ein generatives Warscheinlichkeitsmodell für Dokumente wie Text- oder Bildkorpora. Es dient dazu, ein oder mehrer Themen von Sätzen oder Wörtern zu entdecken, denen sie jeweils eine Verteilung über Worte zuordnet. Einem Dokument wird so eine Verteilung von Themen zugeordnet, welche den Fingerprint/ Themenvektor darstellt. Dabei wird jedem Wort in einem Dokument ein Thema zugeorndet, zu welchenm es warscheinlich gehört.
  Zur Veranschaulichung ein kleines Beispiel:
  Gegeben seien folgende Sätze:
  1. I like to eat broccoli and bananas.
  2. I ate a banana and spinach smoothie for breakfast.
  3. Chinchillas and kittens are cute.
  4. My sister adopted a kitten yesterday.
  5. Look at this cute hamster munching on a piece of broccoli.

  Dann könnte LDA in etwa produzieren:
  - Satz 1 und 2: 100% *Thema A*
  - Satz 3 und 4: 100% *Thema B*
  - Satz 5: 60% *Thema A*, 40% *Thema B*
  - *Thema A*: 30% broccoli, 15% bananas, 10% munching, …
    (hier kann interpretiert werden, dass es in Thema A um 'Essen' geht)
  - *Thema B*: 20% chinchillas, 20% kittens, 20% cute, 15% hamster …
    (hier kann interpretiert werden, dass es bei Thema B um 'süße Tiere' geht)

3. Aspekte

- *3.1* Ausgangspunkt des Projektes

  Das Linked Data Web hat sich als Erweiterung des Webs etabliert. Verschiedene Organsisationen nutzen mehrere tausend Datensätze, um ihre Web-Inhalt auch für Maschinen verständlich zu gestalten. Jedoch erweist sich das Auffinden dieser Datensätze als ein schwieriges Unterfangen. Das Ziel der Pratikumsgruppe DSL-16 ist es, eine Suchmaschine für Datensätze zu erstellen, die dieses Auffinden anhand von keyword-basierter Suche nach Datensätzen sowie der Suche nach ähnlichen Datensätzen, erleichtert. Das Semantic Web wächst - wie das WWW - exponentiell und umfasst bereits mehrere tausend Datensätze. Gleichzeitig ist es für die Veröffentlichung neuer Daten wichtig, dass man diese mit bereits im Web vorhandenen Daten verlinkt. Für diesen Schritt muss man allerdings bereits im Web existierende Datensätze kennen, die mit dem eigenen verlinkt werden können. Daher wird eine entsprechende Suche nach thematisch ähnlichen Datensätzen benötigt.

- *3.2* zugrundeliegende Projekte (TAPIOCA)

  Als zugrundeliegende Suchmaschine wird TAPIOCA verwendet. Diese nimmt die Beschreibung eines Datensatzes als Input und liefert dazu ähnliche Datensätze als Output. Dabei bringt TAPIOCA wie oben beschrieben die Themen der Datensätze in Erfahrung und ordnet diese dann gemäß einer bestimmten Logik in Themengruppen ein. Bei der Suche bekommt jeder Datensatz einen Themen-Vektor zugeordnet. Der Input-Themen-Vektor wird dann mit allen anderen Themen-Vektoren im Index verglichen, so dass ähnliche Datensätze identifiziert werden.
  Genaueres kann auf der Webseite des AKSW entnommen werden, dort findet sich auch der Quelltext des Tapioca Projektes
  http://aksw.org/Projects/Tapioca.html

- *3.3* Anforderungen und Zielsetzung

  Mit TAPIOCA lässt sich einwandfrei in der LOD-Cloud arbeiten.
  Da bisher immer nur mit dem eigenen Goldstandard gearbeitet wurde, existiert unbefriedigenderweise noch immer eine zu hohe Anzahl an möglichen Themen.
  In Zukunft soll es ein Tool geben, mit dem man TAPIOCA besser nutzen kann, deswegen wird ein TOOL entwickelt, mit dem es dem Benutzer ermöglicht werden soll, die Metadaten für seine Dokumente automatisch aus seinem RDF-Datensatz auslesen zu lassen. Mit dieser Aufgabe beschäftigt sich das DSL-16 Projekt. Ziel des Projekts ist die Erstellung einer Datensuche auf Linked Data Datensätzen. Dazu soll eine gegebene Menge von Linked Data Datensätzen erst indexiert und dann auf diesen indexierten Datensätzen gesucht werden. Diese Datensuche soll über eine Weboberfläche bedient werden können und als Eingabe sowohl Datensätze also auch Schlagworte erlauben. Die Weboberfläche wird duch ein kleines Commandline Programm ergänzt, in dem die Suche mittels eines Goldstandards gebanchmarkt werden kann. Die Generierung des Themenmodells soll mehrfrach mit verschiedener Anzahl an Themen möglich sein, um so mittels des Goldstandards eine möglichst optimale Anzahl von Themen festzustellen.
  Weitere Spezifikationen des zu realisierenden Projektes sind in der folgenden überarbeiteten ProjectVision als Teil des Arbeitsplans zu finden.

4. Quellenverzeichnis

- https://de.wikipedia.org/wiki/Linked_Open_Data
- "An Incomplete and Simplifying Introduction to Linked Data" - Anja Jentzsch (AKSW)
- "Detecting Similar Linked Datasets Using Topic modelling" - Michael Röder (AKSW)
- *http://wiki.infowiss.net/Indexierung*
- HELIOS – Execution Optimization for Link Discovery by Axel-Cyrille Ngonga Ngomo
- https://de.wikipedia.org/wiki/Framework
- https://de.wikipedia.org/wiki/DBpedia
- http://de.dbpedia.org/
- http://andreas-pfund.de/definition/metadaten/metadaten.php
- https://de.wikipedia.org/wiki/Ontologie_%28Informatik%29
- https://de.wikipedia.org/wiki/Uniform_Resource_Identifier
- http://www.itwissen.info/definition/lexikon/uniform-resource-identifier-URI.html
- https://de.wikipedia.org/wiki/Programmierschnittstelle
- http://www.searchenterprisesoftware.de/definition/Programmierschnittstelle-API
- http://www.itwissen.info/definition/lexikon/application-programming-interface-API-Programmierschnittstelle.html
- http://www.itwissen.info/definition/lexikon/CBL-contextual-browsing-language.html
- https://de.wikipedia.org/wiki/Web_Ontology_Language
- https://en.wikipedia.org/wiki/Turtle_%28syntax%29
- https://de.wikipedia.org/wiki/Webcrawler
- http://www.itwissen.info/definition/lexikon/Semantisches-Web-semantic-web.html
- http://www.w3.org/TR/2013/REC-sparql11-query-20130321/
- https://en.wikipedia.org/wiki/SPARQL
- https://github.com/AKSW/Tapioca
- http://www.dateiendung.com/format/rdf
- https://de.wikipedia.org/wiki/Resource_Description_Framework
- „Crashkurs Semantic Web“ - Konrad Abicht, Präsenation Universität Leipzig
- Einführung in die Objektorientierte Programmierung (Vorlesung) , Universität Leipzig/Institut für Informatk, Dr. Monika Meiler, Kapitel 13
- http://www.mkbergman.com/947/in-search-of-gold-standards-for-the-semantic-web/
- http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/
- https://de.wikipedia.org/wiki/Latent_Dirichlet_Allocation
- https://www.tdktech.com/tech-talks/topic-modeling-explained-lda-to-bayesian-inference