Storyfinder: Graphen zur Verbesserung des Textverständnisses auf Webseiten

Storyfinder: Using graphs to improve the understanding of text on websites
Master-Thesis von Manuel Kaufmann aus Miltenberg
Tag der Einreichung:

1. Gutachten: Prof. Dr. Chris Biemann
2. Gutachten: Dr. Tatiana von Landesberger

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Fachgebiet Sprachtechnologie

Storyfinder: Graphen zur Verbesserung des Textverständnisses auf Webseiten
Storyfinder: Using graphs to improve the understanding of text on websites

Vorgelegte Master-Thesis von Manuel Kaufmann aus Miltenberg

1. Gutachten: Prof. Dr. Chris Biemann
2. Gutachten: Dr. Tatiana von Landesberger

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 29. Januar 2017

(M. Kaufmann)

Zusammenfassung

In dieser Arbeit wird untersucht, ob sich das Textverständnis auf Webseiten durch einen interdisziplinären Ansatz aus sprachtechnologischen Methoden und Visual Analytics verbessern lässt. Hierfür wird betrachtet, welche Elemente einer Webseite zum Textverständnis beitragen können und wie diese Daten über sprachtechnologische Methoden extrahiert werden können. Es wird zudem beschrieben, wie diese Informationen in einer auf Graphen basierenden Visualisierung dargestellt und durch entsprechende Recherchefunktionen ergänzt werden können. Zu diesem Zweck wurde das Browserplugin *Storyfinder* entwickelt, das im Rahmen dieser Arbeit vorgestellt wird. *Storyfinder* ist eine Erweiterung für den Browser *Mozilla Firefox*, die Informationen wie Schlüsselwörter, Eigennamen und Beziehungen zwischen diesen Elementen beim Öffnen einer Webseite extrahiert. Diese Daten werden mit Informationen aus zuvor besuchten Webseiten kombiniert und durch das Plugin in einem Wissensgraphen in der Seitenleiste des Browsers visualisiert. Über verschiedene Recherchefunktionen können darüber hinaus die Quellen und Ursprungswebseiten erneut gefunden und aufgerufen werden. Im Rahmen einer Evaluation wird durch eine gruppenvergleichende Analyse gezeigt, dass ein solches Plugin das Textverständnis auf Webseiten verbessern kann.

Abstract

We investigate in this paper, whether and how the understanding of text on websites can be improved by combining methods of language technology and visual analytics. Therefor, we identify elements of webpages, which could be used to improve the understanding of websites and present methods of language technology to extract these elements. Furthermore, we investigate and describe how to show the extracted information in a graph based visualization and how to complement it with methods for investigation. For this purpose the browser plugin *Storyfinder* was developed, which is an extension for *Mozilla Firefox*. When opening a webpage, *Storyfinder* extracts keywords, named entities and relations between these elements. These data get combined with data of previously visited websites and visualized through a knowledge graph in the sidebar of the browser. Additionally the plugin provides methods to show the sources of the information and to find and return to the corresponding websites. Finally the plugin is evaluated in an between-subjects experiment and it is shown that the plugin can improve the understanding of text on websites.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Nachrichten und Informationen jeglicher Art werden heutzutage überwiegend über das Internet konsumiert, das sich weiterhin eines stetigen Wachstums erfreut [69].

Durch den einfachen Zugriff auf unzählige Informationsquellen und die starke weltweite Vernetzung stehen wir als Konsumenten hierbei jedoch vor einem Überfluss an Wissen. Dieser Überfluss wird durch die zahlreichen Informationen erzeugt, die täglich im Internet auf uns einströmen. Dasselbe Wissen steht auf unterschiedlichen Seiten, in verschiedenster Granularität zur Verfügung. War es früher ein Problem, die benötigten Informationen zu erhalten oder zu finden, stehen wir heute vielmehr vor der Aufgabe, in diesem Meer aus Wissen, das uns zur Verfügung steht, den Überblick zu bewahren und das Relevante, die entscheidenden Informationstropfen zu sehen und später wiederzufinden.
Ein Zitat von Konfuzius beschreibt dieses Grundproblem des Informationsüberflusses sehr passend:

> *„Zu wissen, was man weiß, und zu wissen, was man tut, das ist Wissen."*
> *– Konfuzius*

Wie Konfuzius feststellt, besteht Wissen nicht allein in der Anhäufung von Informationen, sondern insbesondere im Wissensmanagement, d.h. sich innerhalb dieses Wissens zurechtzufinden und den Überblick über die Informationen zu bewahren, also „zu wissen, was man weiß".
Bei der Recherche im Internet ergeben sich im Wesentlichen drei Anforderungen für dieses Wissensmanagement:

- Beim Besuch einer Webseite möchten wir möglichst schnell erkennen, ob und falls ja, welche relevanten Informationen enthalten sind. Hierdurch ist es uns zum einen möglich, uninteressante Seiten rasch auszusortieren und nur tatsächlich interessante Seiten zu lesen. Zum anderen können wir beim Lesen den Fokus direkt auf diese relevanten Informationen legen.
- Zu einem späteren Zeitpunkt soll auf das zuvor gefundenen Wissen zurückgegriffen werden können. Es ist daher nötig, dass wir bereits gesammelte Informationen möglichst einfach wiederfinden und deren Quellen erneut aufrufen können.
- Zusätzlich ist es nötig, das Wissen im Kontext zu betrachten und Informationen nicht nur auf eine Webseite zu beschränken, sondern beim Lesen auch bereits bekanntes Wissen einfließen zu lassen. Es muss also eine Verknüpfung zwischen den Informationen der besuchten Webseiten stattfinden.

Alle drei Punkte wenden wir intuitiv bei der Recherche an. Wir überfliegen Artikel, suchen nach Zwischenüberschriften oder markanten Textelementen um die Relevanz einer Seite zu beurteilen. Beim Lesen der Seiten erinnern wir uns an Informationen aus zuvor gelesenen Artikeln und setzen diese mit den Informationen des Artikels in Verbindung. Zusätzlich versuchen wir die Quellen unseres gesammelten Wissens durch Lesezeichen oder Notizen festzuhalten und später wiederzufinden.
Diese Methoden funktionieren bis zu einem gewissen Maß, stoßen jedoch auf Grund des heutigen Informationsüberflusses immer häufiger an ihre Grenzen.

Grundidee von *Storyfinder*

Auf Grundlage dieser Beobachtung wurde im Rahmen dieser Arbeit das Browserplugin *Storyfinder* entwickelt. Bei *Storyfinder* handelt es sich um eine Erweiterung für den Webbrowser *Mozilla Firefox*, die den Nutzer bei der Recherche und dem Wiederfinden von bereits gelesenen Informationen unterstützt.
Hierfür werden beim Besuch von beliebigen Webseiten im Artikeltext enthaltene Personen, Organisationen, Orte und sonstige wichtige Schlüsselwörter hervorgehoben. Zusätzlich werden diese Entitäten und deren Beziehungen durch eine interaktive Visualisierung in der Seitenleiste des Browsers dargestellt.

Wissen des einzelnen Benutzers steht im Mittelpunkt

Entscheidend bei der Entwicklung von *Storyfinder* war die Idee, dass anders als bei vielen bereits existierenden Systemen (siehe Kapitel 2) das Wissen bzw. die gelesenen Webseiten eines einzelnen Nutzers erfasst und bereits während dem Lesen visualisierst werden sollen. Durch diese Anforderung, dass eine Visualisierung bereits direkt beim Lesen eines Artikels zur Verfügung stehen soll und keine Vorverarbeitung möglich ist, musste das System besonders auf eine schnelle Verarbeitung der Artikel ausgelegt werden.

Verwendung von sprachtechnologischen Methoden und Visual Analytics

Für die Extraktion der Daten aus den besuchten Webseiten werden verschiedene sprachtechnologische Methoden verwendet. Hierzu zählen unter anderem die Erkennung von Eigennamen über einen Named Entity Recognizer (kurz: NER), die Extraktion von Schlüsselwörtern über einen Keyword Extraction Algorithmus sowie die Ermittlung von Beziehungen über Kookkurrenzen auf Satzebene.
Da es sich um sehr viele Daten und Informationen handelt, die beim Besuch von Webseiten auftreten, wurde die *Visual-Analytic Methode* [40] verwendet, um die extrahierten Daten für den Benutzer verständlich und übersichtlich anzuzeigen. Ziel hierbei war es, dass der Benutzer sehr schnell einen Überblick über die Inhalte erhält und die Details bei Bedarf weiter explorieren kann.

1.2 Ziele/Forschungsfragen

In dieser Arbeit wird untersucht, wie sich der im vorherigen Kapitel beschriebene Informationsüberfluss adressieren lässt. Hierfür wurde im Rahmen einer Studie evaluiert, ob sich mithilfe von sprachtechnologischen und der *Visual-Analytic Methode* das allgemeine Verständnis von Text auf Webseiten verbessern lässt, wofür das Browserplugin *Storyfinder* eingesetzt wurde.
Im Detail werden die folgenden Forschungsfragen untersucht:

1. Wie können die Informationen einer Webseite und die Beziehung zwischen Webseiten erfasst und visualisiert werden?
   Diese Frage lässt sich in die folgenden drei Teilfragen unterteilen:
   a) Welche Informationen einer Webseite können zum Textverständnis beitragen?
   b) Wie können diese Daten im Browser extrahiert werden?
   c) Wie können diese Informationen grafisch aufbereitet und visualisiert werden?
2. Lässt sich das Textverständnis eines Artikels auf einer Webseite verbessern, wenn dem Leser parallel zur Webseite eine Graphen-basierte Visualisierung der Seiteninhalte sowie Informationen aus anderen bisher gelesenen Seiten zur Verfügung stehen?

Zur Klärung der ersten Forschungsfrage (Frage 1a) werden die einzelnen Elemente und Inhalte einer Webseite betrachtet (siehe Kapitel 5.1) und untersucht, wie diese Informationen zum Textverständnis beitragen können. Es werden hierfür sowohl die Textinhalte der Seiten als auch Meta- und Strukturdaten analysiert. Zusätzlich wird eine Methode entwickelt, um anhand der relevanten Elemente zu bestimmen, ob eine Webseite überhaupt für *Storyfinder* geeignet ist, da nicht alle Seiten diese relevanten Elemente enthalten.
Nach Ermittlung der relevanten Elemente, werden zur Beantwortung der zweiten Teilfrage (Frage 1b) verschiedene Methoden betrachtet, wie diese Elemente aus den Webseiten extrahiert werden können. Hierbei wird als ein Aspekt die Einbindung der Extraktionsmethoden in den Webbrowser des Benutzers betrachtet (siehe Kapitel 4.4), sowie als weiterer Aspekt die verschiedenen sprachtechnologischen Methoden zur Datenextraktion erläutert.
Zur Klärung der dritten Teilfrage (Frage 1c) wird anschließend untersucht, wie die extrahierten Daten nach dem Prinzip der *Visual-Analytic Methode* aufbereitet und visualisiert werden können. Hierbei ist ebenfalls die Einbindung der Visualisierung in den Webbrowser (siehe Kapitel 4.4) ein Aspekt. Als weiterer Aspekt werden die Methoden zur Visualisierung der Daten und die nötigen Interaktionsmethoden untersucht, um diese Daten zu explorieren (siehe Kapitel 6).
Zur Beantwortung der zweiten Forschungsfrage (Frage 2) wurde eine Nutzerstudie durchgeführt. In deren Rahmen mussten Probanden verschiedene Fragen zu zuvor gelesenen Artikeln auf Webseiten beantworten. Die Probanden wurden hierfür in eine Untersuchungs- und eine Kontrollgruppe unterteilt und mussten dieselbe Aufgabe mit- bzw. ohne das Plugin durchführen. Anhand der Ergebnisse dieses Experimentes wird in Kapitel 7 analysiert, ob bzw. in welchen Fällen sich das Textverständnis durch das Plugin verbessert.

1.3 Aufbau der Arbeit

Im Folgenden wird der Aufbau der Arbeit skizziert sowie ein Überblick über den Inhalt der neun Kapitel gegeben.
Zu Beginn wird in Kapitel 1 das Thema motiviert. Hieraus werden die Ziele und Forschungsfragen erarbeitet und formuliert. Abschließend erfolgt ein Überblick über den Aufbau der Arbeit.
Kapitel 2 befasst sich mit ähnlichen Arbeiten und betrachtet relevante Forschungen und Projekte aus dem Bereich der Sprachtechnologien und Visualisierung.
Bevor im weiteren Verlauf der Arbeit die Details von *Storyfinder* beschrieben werden, werden in Kapitel 3 wichtige Grundbegriffe eingeführt. Diese sind für das weitere Verständnis der Arbeit dringend erforderlich.

In Kapitel 4 wird ein Überblick über *Storyfinder* sowie die verwendeten Techniken gegeben. Hierfür wird zuerst der Aufbau von *Storyfinder* und dessen Funktionsumfang aus Sicht des Anwenders beschrieben. Anschließend wird der technische Aufbau und die verwendeten Bibliotheken der serverseitigen Anwendungen zur Datenextraktion und Verarbeitung erläutert. Das Kapitel beschreibt hierbei auch die Verbindungen zwischen den einzelnen Komponenten und erläutert den für die Datenspeicherung verwendeten Aufbau der relationalen Datenbank sowie die Struktur zur Speicherung binärer Daten im Dateisystem. Abschließend wird die clientseitige Integration von *Storyfinder* in den Browser betrachtet. Hierfür werden die verschiedenen Schnittstellen der Standardbrowser verglichen und erläutert, warum und wie die Umsetzung des Plugins für *Mozilla Firefox* über das *Add-on SDK* erfolgte.
In Kapitel 5 werden die verschiedenen, überwiegend sprachtechnologischen Komponenten der Datenextraktion, die zur Erfassung der für *Storyfinder* relevanten Daten aus Webseiten genutzt werden, erläutert. Hierfür wird zuerst ein Überblick über die Elemente einer Webseite gegeben und deren Relevanz für *Storyfinder* untersucht. Da nicht alle Webseiten für *Storyfinder* relevant sind, wird anschließend eine Methode vorgestellt, die wir zur Erkennung von relevanten Webseiten verwenden. Daran anschließend werden die einzelnen Extraktionsschritte beschrieben, beginnend mit der Extraktion allgemeiner Webseitendaten wie beispielsweise Seitentitel und Metadaten. Danach werden verschiedene Methoden zur Erkennung des Artikels innerhalb einer Webseite vorgestellt, sowie hierauf aufbauend die Extraktion von Eigennamen, Schlüsselwörtern und weiterer relevanter Textelemente aus dem Artikeltext beschrieben. Nachfolgend werden Methoden zur Erkennung von Beziehungen zwischen diesen Daten betrachtet bevor abschließend erläutert wird, wie markante Änderungen auf Webseiten erkannt werden können.
Nachdem die Datenextraktion erläutert wurde, beschreibt Kapitel 6 die Komponenten der Visualisierung, insbesondere den Aufbau des Wissensgraphen. Zuerst werden jedoch allgemeine Konzepte der Visualisierung sowie die Darstellungen direkt auf den Webseiten erläutert. Anschließend folgt die Beschreibung des Wissensgraphen sowie der Quellen- und Recherchemöglichkeiten. Am Ende des Kapitels wird die Verwendung eines weiteren Gerätes, wie beispielsweise eines Smartphones, als „Second Screen" zur Anzeige des Wissensgraphen vorgestellt.
Zur Klärung der Forschungsfrage wurde im Rahmen dieser Arbeit eine Evaluation durchgeführt. Kapitel 7 beschreibt die hierbei verwendete Between Subjects Evaluationsmethode, stellt die Zusammensetzung der Versuchsgruppen dar und präsentiert die Ergebnisse der Evaluation.
Schlussendlich werden in Kapitel 8 die Ergebnisse zusammengefasst und die Forschungsfragen beantwortet. Abschließend werden in diesem Kapitel in einem Ausblick offene Fragen für zukünftige Forschungen sowie Ideen zur Erweiterung von *Storyfinder* vorgestellt.

2 Related Work

Für die Erfassung und Visualisierung der Daten von besuchten Webseiten, existieren verschiedene Programme, die sich direkt in den Browser integrieren und das Surfverhalten der Benutzer analysieren. In Abschnitt 2.1 werden einige solcher relevanten Browserplugins vorgestellt.
Neben diesen Programmen, die direkt das Surfverhalten überwachen, gibt es noch weitere Tools für das Wissensmanagement, über die Informationen manuell erfasst werden können. Abschnitt 2.2 gibt einen Überblick über diese Tools.
In *Storyfinder* wird ein Benutzer-zentrierter Wissensgraph erstellt. Die hierbei verwendeten Konzepte sind jedoch ähnlich wie bei allgemeinen Wissensgraphen, die nicht speziell das Wissen eines einzelnen Nutzers präsentieren. Für die Erstellung und Visualisierung solcher allgemeiner Wissensgraphen gibt es zahlreiche Tools. Einige wichtige Vertreter werden in Abschnitt 2.3 aufgeführt.

2.1 Browserplugins zur Erfassung und Visualisierung von besuchten Webseiten

Browserplugins ermöglichen das direkte Erfassen des Surfverhaltens und die Visualisierung im Browser, sowie die Visualisierung direkt auf den besuchten Webseiten. Im Folgenden werden zwei Browserplugins vorgestellt, die Beziehungen zwischen besuchten Seiten visualisieren (WebReview und LightBeam) sowie zwei weitere Plugins (Magpie und Espotter) die Eigennamen (Named Entities, kurz: NE) direkt auf der Webseite kennzeichnen.

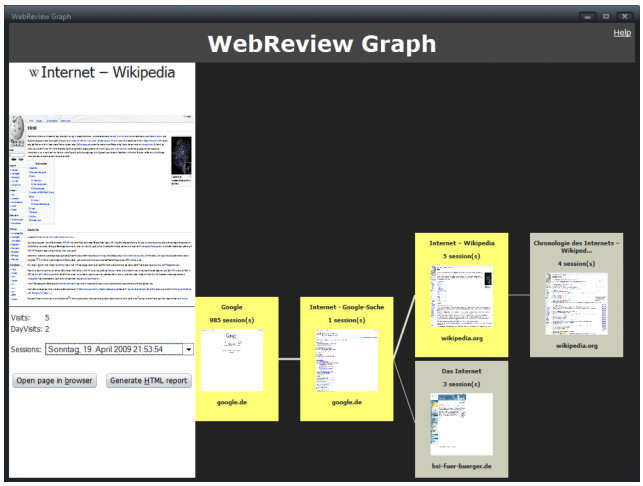


Abbildung 2.1.: WebReview: Visualisierung des Browserverlaufs als Baumstruktur

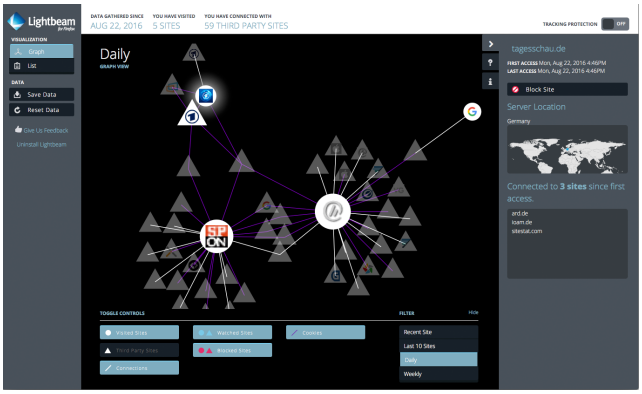


Abbildung 2.2.: LightBeam: Visualisierung der Beziehungen zwischen besuchten Domains und third-party Webseiten

WebReview

WebReview [2] ist ein Browserplugin für *Mozilla Firefox* zum Auffinden bereits besuchter Webseiten. Hierfür wird der Verlauf des Browsers als Baumstruktur visualisiert und eine Suche innerhalb dieses Verlaufs ermöglicht. Der Graph visualisiert sowohl die besuchten Webseiten durch kleine Vorschaubilder als auch den Weg zu diesen Webseiten durch Kanten zwischen den Webseiten. Die Liste der besuchten Webseiten können über verschiedene Suchfunktionen gefiltert werden. Bei der Suche werden jedoch keine Seiteninhalte sondern nur Metadaten wie Host, Title, URL oder Abrufdatum berücksichtigt.

LightBeam

Die Firefox-Erweiterung LightBeam [28] überwacht das Surfverhalten und stellt die besuchten Domains der Webseiten in einem Graphen dar. Neben den eigentlich besuchten Domains, die anhand ihres Favoriten-Icons repräsentiert werden, werden auch third-party Seiten angezeigt, also Inhalte von anderen Seiten, die auf den besuchten Domains eingebunden sind. Bei solchen Inhalten von third-party Seiten handelt es sich häufig um Werbung von Werbenetzwerken oder Widgets von anderen Seiten wie beispielsweise Facebook oder Twitter, die in die besuchten Webseiten integriert sind.

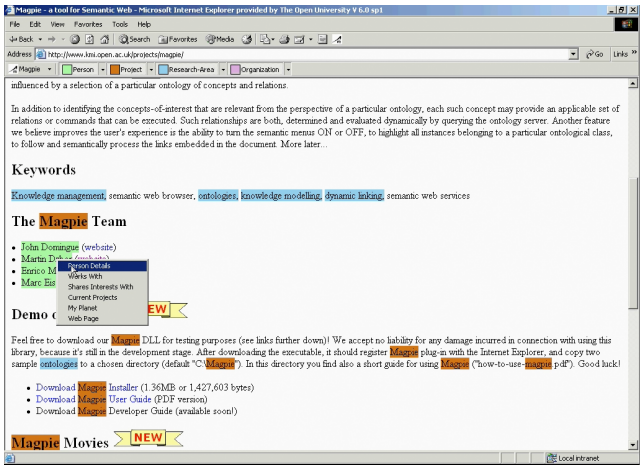


Abbildung 2.3.: Magpie: Bekannte Entitäten werden beim Aufrufen von Webseiten hervorgehoben.

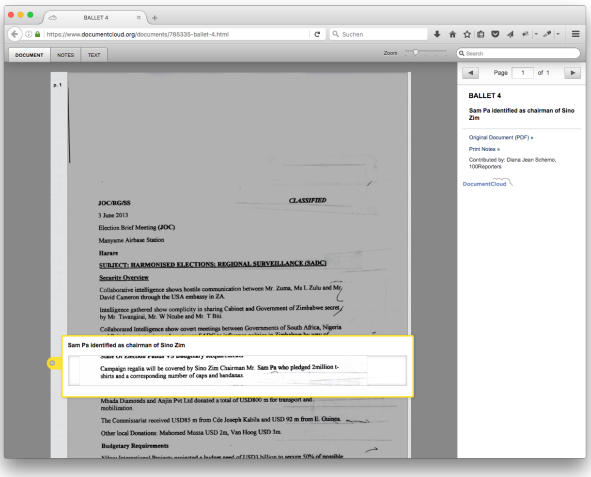


Abbildung 2.4.: DocumentCloud: Eine Annotation zur Person „Sam Pa" wird im zugehörigen Dokument angezeigt.

Magpie und ESpotter

Magpie [19] ist eine Browser-Erweiterung für den Microsoft Internet Explorer zum Hervorheben semantischer Informationen auf Webseiten. Es werden hierfür vorgefertigte Ontologien aus verschiedenen Domains verwendet. Beim Aufrufen einer Webseite werden die auf der Seite enthaltenen Elemente der ausgewählten Domains farbig hervorgehoben. Durch einen Rechtsklick auf die Elemente können weitere in der Ontologie hinterlegte Informationen abgerufen werden, wie beispielsweise Publikationen zu einem bestimmten Forschungsprojekt oder Details zu einer bestimmten Person.
ESpotter [71] erweitert die Idee von Magpie durch eine domainabhängige Erkennung von Eigennamen (Named Entity Recognition, kurz: NER) und manuelle Annotierungen durch den Benutzer. Die domainabhängige Named Entity Recognition ermittelt abhängig von der Domain die Wahrscheinlichkeit, dass es sich bei den Elementen eines bestimmten Lexikons tatsächlich um Eigennamen handelt. Es wird außerdem dem Benutzer ermöglicht, fehlende Eigennamen manuell hinzuzufügen und die Genauigkeit bzw. das gewünschte Verhältnis zwischen Precision und Recall (siehe Kapitel 3.2) bei der Erkennung der Eigennamen frei zu wählen.

2.2 Allgemeine Tools für das Wissensmanagement

Neben Browserplugins existieren weitere Tools für das Wissensmanagement, die meist als Webanwendungen innerhalb des Browsers aufgerufen werden.

Content-Management Systeme und Wikis

Content-Management Systeme (CMS) ermöglichen das Anzeigen und Bearbeiten von Webseiten direkt im Browser. Inhalte können somit einfach geändert und ergänzt sowie neue Informationen ohne größeren Aufwand erfasst werden.
Eine Spezialform von Content-Management Systemen sind Wikis, mit ihrem bekanntesten Vertreter, der Enzyklopädie Wikipedia[1]. Solche Systeme ermöglichen das kollaborative Bearbeiten von Webseiten innerhalb des Browsers. Hierdurch eignen sich diese besonders zur einfachen Erfassung von Wissen und Informationen jeglicher Art in strukturierter oder unstrukturierter Form.
Innerhalb von Organisationen und Firmen werden häufig interne Wikis eingesetzt, die einem eingeschränkten Kreis von Benutzern als Basis für das gemeinsame Wissensmanagement dienen [11].

DocumentCloud

DocumentCloud [18] ist ein Webdienst zur Erstellung und Veröffentlichung von Dokumentsammlungen. Alle Dokumente werden mit Hilfe von OpenCalais analysiert [31]. Hierdurch werden Entitäten wie Orte, Zeiten und Personen in den Dokumenten erkannt und indexiert, wodurch die Dokumente einfacher wiedergefunden werden können. Zudem werden die erkannten Entitäten innerhalb der Dokumente beim Lesen hervorgehoben. Über eine Texterkennung un-

[1] https://www.wikipedia.org

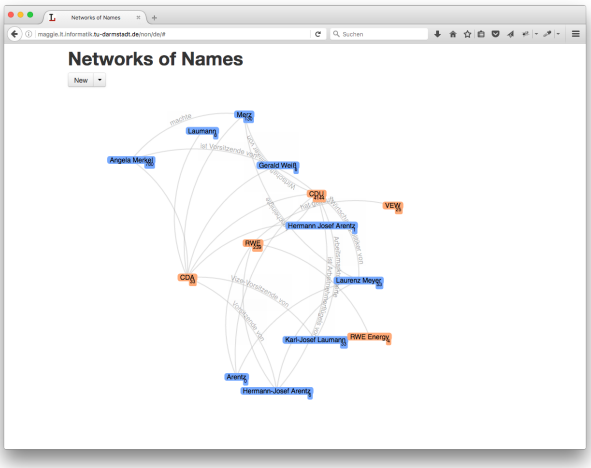


Abbildung 2.5.: Darstellung der RWE Affäre in Network Of Names

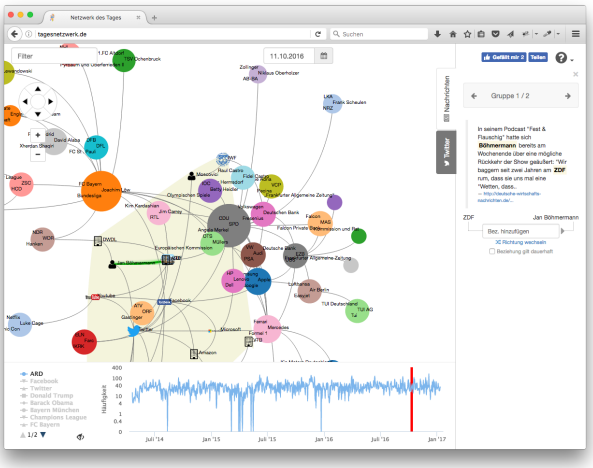


Abbildung 2.6.: Tagesnetzwerk vom 11.10.2016 in „Network of the day" mit der ausgewählten Beziehung zwischen Jan Böhmermann und dem ZDF

terstützt DocumentCloud auch die Analyse von Dokumenten, die nur in gerasterter Form vorliegen, wie beispielsweise eingescannten Textseiten.

2.3 Benutzerunabhängige Visualisierung von Websitedaten

Network of Names

Das Network of Names [42] ermittelt Beziehungen zwischen Personen und Organisationen anhand von Informationen, die aus einem Textkorpus von Zeitungsartikeln extrahiert werden. Die Erkennung der Personen und Organisationen erfolgt in einem Vorverarbeitungsschritt mithilfe von Named Entity Recognition. Hierbei werden aus den unstrukturierten Texten des Textkorpus die Eigennamen extrahiert sowie die Beziehungen zwischen diesen Elementen ermittelt. Hierfür werden Kookkurrenzen auf Satzebene gebildet.
Nach diesem Vorverarbeitungsschritt werden die Beziehungen in einer Web-basierenden interaktiven Visualisierung dargestellt. Für die Visualisierungen wird ein Knoten-Link-Diagramm verwendet, in dem die Entitäten als Knoten und die Beziehungen zwischen diesen Entitäten als Links dargestellt werden. Im Rahmen dieser Visualisierung können die Beziehungen weiter erforscht und beschriftet werden. Aus diesen Beschriftungen erlernt das System automatisch Suchmuster, die diese Beschriftungen auch auf ähnliche Beziehungen in anderen Sätzen anwendet.

Network of the day

Network of the day [7] ist eine Webseite, die wichtige Begriffe eines einzelnen Tages in einer Graphen-basierten Visualisierung darstellt. Diese wichtigen Begriffe werden mit Hilfe verschiedener sprachtechnologischer Methoden wie beispielsweise Named Entity Recognition aus Twitter Tweets und Artikeln von Nachrichtenseiten extrahiert. Ähnlich wie beim Network of Names werden Beziehungen zwischen den extrahierten Elementen über Kookkurrenzen auf Satzebene ermittelt. Über einen Clustering-Algorithmus werden zusammengehörige Begriffe gruppiert und in der Visualisierung entsprechend als Gruppe dargestellt. Analysefunktionen bieten zudem detaillierte Informationen über die zeitliche Auftretungshäufigkeit von einzelnen oder mehreren Begriffen.

Lobbyradar

Der Lobbyradar [20] stellt auf der Webseite http://www.lobbyradar.org Beziehungen zwischen Politikern, Firmen, Journalisten, Vereinen und Verbänden dar. Die Seite wurde von mehreren freien Journalisten aufgebaut, wobei die Informationen aus festen Quellen, wie beispielsweise der Lobbyliste des Bundestages, ausgelesen wurden. Über diese Quellen werden die Daten auch regelmäßig aktualisiert. Hierfür wurden spezielle „Scraper" entwickelt, um die Daten aus diesen Quellen entsprechend zu extrahieren. Zusätzlich wurden zahlreiche Daten manuell erfasst und in die Datenbank eingetragen.
Auf der Webseite werden die gesammelten Lobbyverbindungen als Netzwerk dargestellt, das die einzelnen Personen, Organisation usw. als Knoten enthält und die Beziehungen zwischen diesen Elementen als Kanten darstellt. Hierüber lassen sich auch weitere Details zu den einzelnen Elementen, wie beispielsweise Quellen und Verbindungen aufrufen.

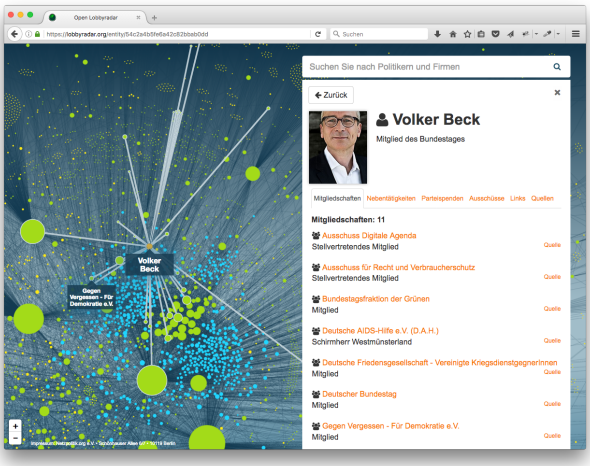


Abbildung 2.7.: Lobbyradar: Browseransicht des Lobbyradar Netzwerkes vom 12.01.2017. Der Politiker „Volker Beck" wurde im Netzwerk ausgewählt.

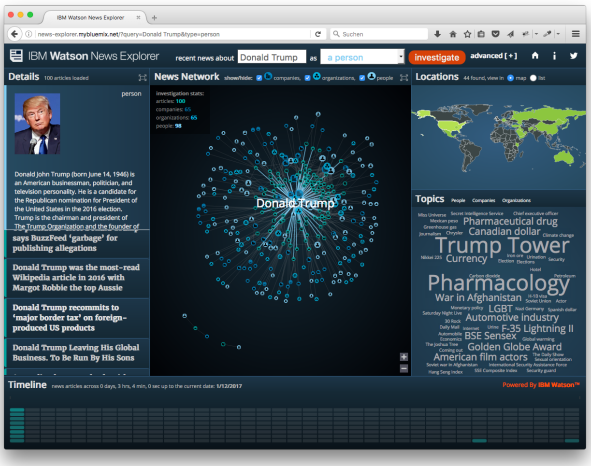


Abbildung 2.8.: IBM Watson News Explorer: Für die Abbildung wurde „Donald Trump" als zentrale Entität ausgewählt (abgerufen am 12.01.2017).

Ein Browserplugin bietet zudem die Möglichkeit, die im Lobbyradar enthaltenen Elemente automatisch auf den besuchten Webseiten hervorheben zu lassen und direkt auf den Seiten weitere Informationen zu diesen Elementen aufzurufen. Lobbyradar verwendet hierbei eine lokale Datenbank auf Benutzerseite, in denen die Entitäten gespeichert sind. Nur für den Aufruf von Details wird auf die Online-Datenbank zugegriffen. Das Highlighting erfolgt hierbei im kompletten sichtbaren Bereich der Webseite, also beispielsweise auch innerhalb der Navigation. Die Seiten, auf den das Plugin arbeiten soll, wird durch eine Whitelist angegeben und umfasst überwiegend große deutsche Nachrichtenseiten.

IBM Watson News Explorer

Der IBM Watson News Explorer [48] verwendet die Technik von IBM Watson [26] zur Analyse von Nachrichtentexten. Aus diesen Texten werden durch sprachtechnologische Methoden wichtige Entitäten sowie deren Beziehungen extrahiert. Diese Daten werden zu verschiedenen Themengebieten gruppiert und auf der Webseite http://news-explorer.mybluemix.net dargestellt. Durch die Auswahl einer Entität werden in verschiedenen Elementen die zugehörigen Entitäten visualisiert. Hierbei wird sowohl ein Knoten-Link-Diagramm eingesetzt, das die Entitäten und der Beziehungen darstellt, als auch eine Word Cloud, in denen die Entitäten nach Relevanz skaliert dargestellt werden. Eine Weltkarte ermöglicht zudem die Filterung der Daten anhand bestimmter Orte. Ähnlich wie im Netzwerk des Tages wird über eine Timeline der zeitliche Verlauf des Auftretens der ausgewählten Entität dargestellt.

3 Grundlagen

Im Folgenden werden verschiedene Grundbegriffe erläutert, die für das Verständnis der weiteren Kapitel benötigt werden. Das Kapitel ist in die Bereiche Sprachtechnologie, maschinelles Lernen, Visualisierung und Web-Technologien unterteilt, sowie einem abschließenden Abschnitt der einige sonstige Begriffe erläutert, die sich keinem der anderen Bereiche zuordnen lassen.

3.1 Sprachtechnologie

Das *Storyfinder-Plugin* und insbesondere die Extraktion der Daten aus Artikeltexten basiert auf verschiedenen Methoden aus dem Bereich der Sprachtechnologie. Im Folgenden werden einige Grundbegriffe aus diesem Bereich erläutert.

Token, Lemma und Tokenisierung

Viele der vorgestellten Methoden arbeiten auf Token-Ebene. Hierfür ist es wichtig, die Begriffe Token, Lemma und Tokenisierung zu definieren. Nach [65] ist ein *Token* „[...] eine dem Wort weitgehend entsprechende orthografische Einheit, die man an einer exakten Stelle im Text verorten kann". Die Unterteilung eines Textes in einzelne Tokens bezeichnet man als *Tokenisierung*. Im folgenden Beispiel sind die einzelnen Tokens durch Klammern gekennzeichnet:

> *[Mercedes-Pilot] [Nico] [Rosberg] [hat] [sich] [die] [Pole] [Position] [für] [sein] [Heimrennen] [in] [Hockenheim] [gesichert] [.]*[1]

Zahlreiche weitere Verarbeitungsschritte, wie beispielsweise POS-Tagging (Zuordnung von Wortarten), Named Entity Recognition (Erkennung von Eigennamen) oder Keyword Extraction (Erkennung von Schlüsselwörtern), arbeiten auf Tokenebene. Die Tokenisierung bildet daher die Grundlage für diese Methoden.
Als *Lemma* werden „die Grundformen im Text bezeichnet" [65]. Das Lemma des Tokens „gesichert" wäre somit beispielsweise die Grundform „sichern".

Korpus

Als *Korpus* oder Textkorpus bezeichnet man eine Kollektion von mehreren Texten. In *Storyfinder* bilden die Artikel auf den besuchten Webseiten das Korpus. Methoden, wie beispielsweise die im folgenden Abschnitt beschriebene Berechnung der Dokumentfrequenz bzw. des TFIDF Wertes, die eine Berechnung auf einem Korpus durchführen, verwenden hierfür dieses Korpus aus besuchten Webseiten.

Termfrequenz, Dokumentenfrequenz und TFIDF

Die Häufigkeit bzw. die Frequenz, mit der ein bestimmtes Token oder Lemma in einem Text auftritt, kann Aufschluss über die Relevanz dieses Wortes liefern. Hierfür werden die drei Maße *Termfrequenz*, *Dokumentfrequenz* sowie *TFIDF* verwendet, über die sich der Informationsgehalt eines Terms messen lässt. Häufig erfolgt die Berechnung dieser Werte auf Basis der Lemmata innerhalb eines Textes oder Korpus.
Die *Termfrequenz* TF gibt hierbei an, wie häufig ein bestimmter Term innerhalb eines Dokumentes enthalten ist. Die *Dokumentfrequenz* DF gibt hingegen an, in wie vielen Dokumenten ein Term enthalten ist.
Da bestimmte Wörter deutlich häufiger auftreten wie andere Wörter, lässt sich die Termfrequenz verschiedener Wörter nicht direkt vergleichen, um hierüber auf den Informationsgehalt eines Wortes zu schließen. Es wird deshalb der $TFIDF$-*Wert* als das Verhältnis von Termfrequenz und Dokumentenfreqenz bzw. das Produkt aus Termfrequenz und dem Inversen der Dokumentenfrequenz (IDF) gebildet:

$$TFIDF = TF \times IDF$$

Dieser Wert ist bei Wörtern, die in einem Dokument sehr häufig, im gesamten Korpus jedoch nur selten vorkommen, sehr hoch. Ein hoher $TFIDF$-Wert zeigt somit einen hohen Informationsgehalt eines Terms innerhalb eines Dokumentes an. Insbesondere für die Erkennung der im folgenden Abschnitt definierten Schlüsselwörter ist der $TFIDF$-Wert entscheidend.

[1] Quelle: http://www.sportschau.de/formel1/formel-eins-qualifying-hockenheim-bericht-100.html

Schlüsselwörter, Eigennamen und Entitäten

Als *Schlüsselwörter* (engl. „Keyword") eines Textes bezeichnet man wichtige Wörter innerhalb eines Textes. Eine Möglichkeit solche wichtigen Wörter zu erkennen, basiert auf der Häufigkeit des Auftretens dieser Wörter im Text. Erscheint ein bestimmtes Lemmata oder n-Gramm überverhältnismäßig häufig in einem Text, so kann man davon ausgehen, dass dieses Wort einen hohen Informationsgehalt in diesem Kontext besitzt. Zu beachten ist hierbei jedoch, dass sich ein überverhältnismäßig häufiges Auftreten nicht alleine durch die Termfrequenz bestimmen lässt. Bestimmte Wörter, wie beispielsweise „die", treten allgemein häufiger auf wie andere Wörter (z.B. „Mercedes-Pilot"). Da das Wort „die" jedoch im gesamten Korpus sehr häufig vorkommt, ist das häufige Auftreten in einem Text nicht überverhältnismäßig. Das Wort „Mercedes-Pilot" kommt jedoch vermutlich im gesamten Korpus nur sehr selten vor. Tritt dieses Wort innerhalb eines Textes mehrmals auf, so ist das Auftreten überverhältnismäßig häufig und es handelt sich um ein Schlüsselwort.
Zur Erkennung von Schlüsselwörtern kann der TFIDF-Wert eines Wortes verwendet werden, da dieser die Häufigkeit des Auftretens im Verhältnis zwischen Dokument und Korpus betrachtet.
Als *Eigennamen* (engl. „Named Entity" oder kurz „NE") bezeichnet man benennbare Objekte. Hierzu zählen beispielsweise Personen, Orte und Organisationen. Die „Eigennamenerkennung findet und normalisiert spezielle Ausdrücke wie Personen-, Firmen-, Produktnamen, komplexe Datums-, Zeit-, und Maßausdrücke." [33]
Bei der Eigennamenerkennung werden die Eigennamen im Text extrahiert und anschließend verschiedenen Kategorien zugeordnet. In dieser Arbeit wird zur Eigennamenerkennung die Software *GermaNER* [8] eingesetzt, die eine Einteilung in die vier Kategorien „Person", „Ort", „Organisation" und „Sonstiges" vornimmt.
Eigennamen und Schlüsselwörter bilden die Grundlage des Wissensgraphen in *Storyfinder.* Terme aus diesen beiden Kategorien werden in den weiteren Kapiteln dieser Arbeit als *Entität* bezeichnet.

Kookkurrenz

Neben Eigennamen und Schlüsselwörtern sind insbesondere die Beziehungen zwischen diesen Wörtern für die Visualisierung in *Storyfinder* relevant. Grundlage hierfür sind *Kookkurrenzen*.

> *„Als Kookkurrenz wird das gemeinsame Vorkommen zweier oder mehrerer Wörter in einem Kontext von fest definierter Größe bezeichnet." [45]*

Zwei Wörter sind somit beispielsweise kookkurrent, wenn diese innerhalb desselben Satzes (Kookkurrenz auf Satzebene) oder innerhalb eines Abschnittes (Kookkurrenz auf Abschnittsebene) vorkommen. Im folgenden Beispiel sind „Nico Rosberg" und „Lewis Hamilton" kookkurrent auf Abschnittsebene, jedoch nicht auf Satzebene. „Nico Rosberg" und „Hockenheim" sind hingegen auf Satzebene (und somit auch auf Abschnittsebene) kookkurrent:

> *„Mercedes-Pilot **Nico Rosberg** hat sich die Pole Position für sein Heimrennen in **Hockenheim** gesichert. Der 31-Jährige verwies seinen Teamrivalen **Lewis Hamilton** auf den zweiten Platz."*

In *Storyfinder* werden Kookkurrenzen auf Satzebene verwendet, um eine Beziehung zwischen zwei Entitäten herzustellen.

3.2 Maschinelles Lernen

In *Storyfinder* wird eine Methode verwendet, um Webseiten zu klassifizieren. Hierfür werden Methoden des *maschinellen Lernens* verwendet. Beim maschinellen Lernen wird Wissen in Form eines Modells automatisch oder teil-automatisch aus Trainingsbeispielen gelernt und so verallgemeinert, dass es auf andere Beispiele angewendet werden kann. Wichtige Grundbegriffe aus diesem Bereich werden im Folgenden erläutert.

Trainings- und Evaluationskorpus

Zum Erlernen von Wissen werden im Bereich des maschinellen Lernens Trainingsbeispiele benötigt. Diese befinden sich in einem annotierten Trainingskorpus. Zur Überprüfung des erlernten Wissens wird zusätzlich ein Evaluationskorpus benötigt. Die Korpora enthalten zusätzlich zu den Trainingsdaten das gewünschte Ergebnis. Die Annotation der Korpora erfolgt meist manuell. Bei einem Klassifizierungsproblem sind beispielsweise neben den zu klassifizierenden Elementen auch die zugehörigen korrekten Klassen angegeben.
Aus dem Trainingskorpus lernt der Algorithmus, welche Ergebnisse korrekt sind.
Um die Qualität des aus dem Trainingskorpus gelernten Modells zu beurteilen, wird ein Evaluationskorpus verwendet. Zur Evaluation wird das gelernte Modell auf diesen Evaluationskorpus angewendet und das Ergebnis anschließend mit den im Korpus annotierten Ergebnissen verglichen. Hieraus lassen sich verschiedene Evaluationsmaße berechnen, die im nächsten Abschnitt erläutert werden.

Evaluationsmaße

Es gibt zahlreiche Maße zur Beurteilung der Evaluationsergebnisse einer Methode, die je nach Problemstellung unterschiedlich gut zur Beurteilung der Ergebnisse geeignet sind. Wir stellen deshalb im Folgenden nur die beiden in dieser Arbeit verwendeten Maße Precision und Recall sowie den hieraus errechneten F-Score vor.

false positive, true positive, false negative und true negative

Die Evaluationsmaße berechnen sich aus der Anzahl der korrekt bzw. inkorrekt klassifizierten Elemente. Ein Element kann hierbei zu einer der folgenden vier Mengen gehören:

- Als true positive (TP) wird ein korrekt zugeordnetes Element bezeichnet, das zur Klasse zugeordnet wurde und auch zu dieser Klasse gehört.
- Als true negative (TN) wird ein korrekt zugeordnetes Element bezeichnet, das korrekterweise nicht zur Klasse zugeordnet wurde und auch tatsächlich nicht zu dieser Klasse gehört.
- Als false positive (FP) wird ein Element bezeichnet, das zur Klasse zugeordnet wurde, jedoch nicht zur Klasse gehört.
- Als false negative (FN) wird ein Element bezeichnet, das nicht zur Klasse zugeordnet wurde, obwohl es tatsächlich zur Klasse gehören würde.

Precision

Über die Precision wird die Genauigkeit der Ergebnisse gemessen. Die Precision gibt an, wie viele der zur Klasse zugeordneten Elemente tatsächlich zu dieser Klasse gehören. Es wird hierbei nicht berücksichtigt, ob noch weitere relevante Elemente existieren, die nicht zur Klasse zugeordnet wurden. Die Precision ergibt sich als Verhältnis zwischen der Anzahl der true positives und der Summe aus true positives und false positives:

$$precision = \frac{tp}{tp + fp}$$

Eine hohe Precision gibt an, dass die klassifizierten Elemente mit hoher Wahrscheinlichkeit tatsächlich zur vorhergesagten Klasse gehören.

Recall

Der Recall gibt an, wie viele der relevanten Elemente einer Klasse erkannt wurden. Es wird hierbei nicht berücksichtigt, ob noch weitere Elemente erkannt wurden, die nicht zur Klasse gehören. Der Recall ist definiert, als das Verhältnis zwischen korrekt erkannten Elementen (true positives) und aller relevanten Elemente (Summe aus true positives und false negatives):

$$recall = \frac{tp}{tp + fn}$$

Ein hoher Recall gibt an, dass sehr viele relevante Elemente zur jeweiligen Klasse zugeordnet werden. Ein perfekter Recall für eine Klasse kann immer erzielt werden, indem alle Elemente zu dieser Klasse zugeordnet werden.

F-Score

Da in den meisten Fällen sowohl eine hohe Precision als auch ein hoher Recall gewünscht ist, also idealerweise dass alle relevanten Elemente (maximaler Recall) und keine falschen Elemente (maximale Precision) zu einer Klasse zugeordnet werden, verwendet man als Evaluationsmaß das harmonische Mittel aus diesen beiden Werten, den F_1-Score:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Bei diesem F_1-Score werden Precision und Recall gleichermaßen gewichtet.

n-Fold Evaluation

Da der Aufwand für die Erstellung separater Trainings- und Evaluationskorpora sehr aufwendig ist, kann eine sogenannte n-Fold Evaluation durchgeführt werden. Bei dieser Art der Evaluation wird das Korpus in n gleichgroße Teile (Folds) aufgeteilt. Anschließend werden n Evaluationsdurchgänge ausgeführt, bei denen jeweils einer der n Teile als Evaluationskorpus verwendet wird und die restlichen (n-1) Teile als Trainingskorpus. In den n Durchgängen wird somit jeder Teil einmal als Evaluationsmenge verwendet.
Abschließend wird der Durchschnitt der einzelnen Ergebnisse als Gesamtergebnis der Evaluation verwendet.

Die Evaluation dauert bei dieser Methode deutlich länger, als bei einer einfachen Evaluation mit einem Trainings- und Evaluationskorpus. Der Vorteil liegt jedoch in dem größeren Trainingskorpus, da immer auf (n-1) Teilen des Trainingskorpus trainiert werden kann.
Bei der Evaluation in Kapitel 5.3 wurde eine 10-Fold Evaluation durchgeführt.

3.3 Visualisierung

Für die Visualisierung in *Storyfinder* werden Graphen eingesetzt. Dieses Unterkapitel definiert den Begriff, sowie weitere Begriffe die zur Erzeugung eines Diagramms benötigt werden.

Graph, Knoten und Kanten

Ein (ungerichteter) Graph ist definiert als „einer Menge E von Ecken und einer Menge K von Kanten", wobei eine Kante definiert wird „durch ein ungeordnetes Paar von Ecken, den Endecken" oder auch Knoten [64].

Graphendiagramm und Layout

Beim *Graphendiagramm* handelt es sich um eine Visualisierung von Graphen. In dieser Arbeit verwenden wir ausschließlich zweidimensionale Darstellungen von Graphen. Jedem Knoten eines Graphen muss in diesem Fall eine Position im zweidimensionalen Raum zugeordnet werden. Der Graph erhält hierdurch ein *Layout*. Diese Zuordnung ist im Allgemeinen nicht eindeutig und muss deshalb über einen *Layout-Algorithmus* erstellt werden.

Layout-Algorithmus und force-directed Layout

Ein *Layout-Algorithmus* erstellt ein Layout für einen vordefinierten Graphen. Die Algorithmen können hierbei verschiedene Ziele verfolgen, wie beispielsweise die Vermeidung von Überschneidungen zwischen Kanten, geringer Platzbedarf oder Gruppierungen der Knoten in logische Einheiten. Wir verwenden in dieser Arbeit ein Graphendiagramm das mit Hilfe eines *force-directed Layout* erstellt wurde. Hierbei wird über die Simulation von Krafteinwirken auf die einzelnen Knoten und die zeitliche Verringerung dieser Kräfte (Abkühlung) ein Layout berechnet.

3.4 Web- und Webtechnologien

Das *Storyfinder-Plugin* sowie die Visualisierung basiert auf verschiedenen Techniken, die im Web-Bereich zum Einsatz kommen, sogenannten *Web-Technologien*. Im Folgenden werden wichtige Grundbegriffe aus diesem Bereich erläutert.

Hypertext Markup Language

Die *Hypertext Markup Language* [9] oder kurz *HTML* ist eine Auszeichnungssprache für strukturierte Textdokumente, die neben Text noch weitere Elemente wie Grafiken, Tabellen und Hyperlinks zu anderen Dokumenten enthalten können. HTML ist die Standard-Auszeichnungssprache für Webdokumente und bildet die Grundlage nahezu aller Webseiten. Die Sprache beschreibt hierbei nur die Stuktur und den Inhalt eines Dokumentes bzw. einer Webseite, jedoch nicht deren Aussehen. Das Aussehen wird über Cascading Style Sheets definiert.
Die Elemente eines HTML Dokumentes werden als Baumstruktur definiert und enthalten neben Inhaltselementen auch Metadaten wie Seitentitel oder Autor.

Cascading Style Sheets

Cascading Style Sheets (kurz: „CSS") beschreiben das Aussehen der Elemente eines HTML-Dokumentes. Die Darstellung jedes einzelnen Elementes kann hierdurch nahezu beliebig angepasst werden. Alle Browser enthalten ein Standardset an Style Sheets, die den HTML-Elementen ein vordefiniertes Aussehen geben, so dass ein HTML Dokument auch ohne die explizite Angabe von Style Sheets angezeigt werden kann. Durch seitenspezifische Style Sheets können diese Standardwerte überschrieben oder ergänzt werden, um das gewünschte Aussehen zu definieren.

Javscript und Javascript Engine

Javascript (kurz: „JS") ist eine Skriptsprache, die im Kontext einer Webseite innerhalb des Browsers ausgeführt wird. Über Javascript können die Inhalte einer Webseite dynamisch verändert und Benutzereingaben verarbeitet werden. Zudem können über Javascript externe Resourcen über HTTP nachgeladen oder Daten an Webservices gesendet werden.
Javascript findet mittlerweile nicht mehr nur als Skriptsprache im Browser Einsatz, sondern kann beispielsweise auch serverseitig in Webservern oder anderen Anwendungen eingesetzt werden.
In *Storyfinder* verwenden wir Javascript sowohl für die Visualisierung im Browser als auch für den serverseitigen *Storyfinder-Webservice*.

Über die *Javascript Engine* wird Javascript Code interpretiert und ausgeführt. Gängige Javascript Engines sind *SpiderMonkey* (*Mozilla Firefox*), *V8* (u.a. Google Chrome und NodeJS) und *WebKit* (u.a. Apple Safari). Mit *NodeJS* ist es möglich Javascript Code auch serverseitig auszuführen.

Webbrowser

Ein *Webbrowser* (häufig auch nur *Browser*) ist ein Programm zum Aufrufen und Anzeigen von Webseiten. Er stellt die Benutzeroberfläche dar, bindet Engines zum Darstellen von HTML und anderen Inhalten (z.B. PDF oder SVG), sowie zum Ausführen von Javascript-Code ein.
Gängige Browser für Desktop-Systeme sind Micorsoft Internet Explorer, Google Chrome, *Mozilla Firefox* und Apple Safari. Bei den mobilen Systemen sind Apple Safari, Google Chrome und der Android Browser die am stärksten vertretenen Varianten.

Render Engine

Die *Render Engine* eines Browsers verarbeitet das Markup einer Webseite (z.B. HTML und SVG) und visualisiert die Seite (*Rendern*). Das Aussehen einer Webseite wird hierbei meist durch *Cascading Style Sheets* (CSS) beschrieben. Weit verbreitete Render Engines sind Gecko (u.a. Mozilla Firefox) und WebKit (u.a. Apple Safari und Google Chrome).
Nach dem Parsen des Quelltextes wird ein *Document Object Model* (DOM) erstellt, das die einzelnen Elemente einer Webseite in einer hierarchischen Struktur enthält. Hieraus wird anschließend der Frame Tree (Gecko) bzw. Render Tree (WebKit) erstellt, der den Elementen (Frames) ihr tatsächliches Layout mit Position, Größe, Farbe usw. zuordnet. Dieser Frame-Tree kann anschließend gezeichnet und auf dem Bildschirm ausgegeben werden (vgl. [30]).
Das DOM kann nach dem erstmaligen Rendern einer Webseite über *Javascript* verändert werden. Hierdurch ändert sich die Darstellung einer Webseite und diese muss neu gezeichnet werden.

Same-origin Policy und CORS

Durch das Sicherheitskonzept *Same-origin Policy* wird innerhalb eines Browsers sichergestellt, dass eine Webseite oder ein darin ausgeführtes Script nicht unerlaubterweise auf die Daten einer fremden Webseite zugreift. Die Same-origin Policy stellt dies sicher, indem domainübergreifende Zugriffe unterbunden werden. Ein Script auf der Webseite *example.com/index.html* kann beispielsweise Daten von der Seite *example.com/data/test.html* laden, da beide Seiten in der gleichen Domain (example.com) liegen. Das Script kann jedoch keine Daten von der Seite *anotherexample.com/data.html* laden, da hierfür ein Zugriff über die Domaingrenze hinweg nötig wäre.
In *Storyfinder* werden durch das Plugin, das innerhalb einer speziellen lokalen Domain läuft, Resourcen des *Storyfinder-Servers* abgerufen, der sich innerhalb einer anderen Domain befindet. Um diese Anfragen trotz der Same-origin Policy zu ermöglichen, wird *CORS* verwendet.
Unter Cross-Origin Resource Sharing (kurz: CORS) versteht man eine Methode, um die Einschränkung der Same-origin Policy zu umgehen. Ein Webserver kann hierbei durch die Ausgabe von speziellen HTTP Headern den Zugriff für bestimmte oder beliebige Webseiten auf die angeforderte Resource erlauben.

3.5 Sonstiges

Plugin

Ein *Plugin* (auch als *Add-on* oder *Extension* bezeichnet) ist eine Softwarekomponente, die ein Programm um neue Funktionen ergänzt und meist von unabhängigen Entwicklern zur Verfügung gestellt wird. Das Programm, das durch das Plugin erweitert werden soll, muss diese Möglichkeit der Erweiterung unterstützen und hierfür eine Schnittstelle bereitstellen. Diese Schnittstelle regelt, welche Veränderungen bzw. Ergänzungen durch das Plugin möglich sind.
Storyfinder erweitert die Funktionen des Webbrowsers und arbeitet deshalb als Browser Plugin. Die meisten Webbrowser unterstützen Plugins, die Schnittstellen unterscheiden sich jedoch deutlich zwischen den einzelnen Webbrowsern. In Kapitel 4.4.1 erfolgt ein Überblick über die Plugin Schnittstellen der verschiedenen Browser.

Linux Container

Bei *Linux Containern* (LXC) handelt es sich um eine Art der Virtualisierung, die jedoch nicht auf einzelnen virtuellen Maschinen (VMs) aufbaut. Es werden stattdessen mehrere Prozesse in voneinander abgeschotteten Umgebungen (Container) ausgeführt, die im Gegensatz zu virtuellen Maschinen alle einen gemeinsamen Kernel verwenden. Hierdurch ist diese Art der Virtualisierung deutlich Resourcen schonender.
In *Storyfinder* werden Linux Container für alle serverseitigen Komponenten genutzt.

Second Screen

Als Second Screen wird ein zusätzliches Gerät, wie beispielsweise ein Smartphone oder Tablet bezeichnet, das die Inhalte eines anderen Gerätes (z.B. PC, TV, Laptop) durch zusätzliche Informationen ergänzt. Häufig wird der Begriff im

Bereich des interaktiven Fernsehens verwendet, wenn zusätzlich zum aktuell ausgestrahlten TV Programm noch weitere Informationen oder Interaktionsmöglichkeiten zur laufenden Sendung über ein internetfähiges Gerät zur Verfügung stehen. Zusätzliche Informationen können beispielsweise Hintergrundinformationen zu den Schauspielern in der aktuellen Szene, zusätzliche Kameraperspektiven bei Sportereignissen oder Tweets zur Sendung sein. Der Begriff ist jedoch nicht nur auf den TV Bereich beschränkt.
Storyfinder kann nicht nur im Browser des Hauptgerätes sondern auch zusätzlich auf weiteren Geräten angezeigt werden. Hierdurch wird die Darstellung des Wissensgraphen und die Interaktion mit diesem auf einem Smartphone oder anderen Geräten ermöglicht, während das eigentliche Aufrufen der Webseiten auf dem Hauptgerät erfolgt.

4 Systemübersicht

Die im Rahmen dieser Arbeit entwickelte Software besteht aus mehreren Komponenten für die Textanalyse und Visualisierung. In diesem Kapitel erfolgt eine Übersicht über das System. Hierfür wird zuerst in Unterkapitel 4.1 das System aus Anwendersicht beschrieben. Anschließend wird die technische Seite von *Storyfinder* betrachtet. Hierfür werden in Unterkapitel 4.2 die serverseitigen Komponenten vorgestellt und in Unterkapitel 4.3 der Aufbau der Datenbank beschrieben. Abschließend werden die clientseitigen Komponenten von *Storyfinder*, insbesondere die Einbindung in den Browser, in Unterkapitel 4.4 erläutert.

4.1 Storyfinder aus Anwendersicht

Abbildung 4.1.: Storyfinder eingebunden in eine Entwicklerversion von *Mozilla Firefox*. Im linken Bereich ist die Seitenleiste mit dem lokalen Wissensgraphen zu sehen, der die Entitäten der daneben angezeigten Webseite (*http://www.tagesschau.de/inland/fifa-dfb-wm-101.html*; abgerufen am 05.09.2016) visualisiert. Die Entitäten sind zusätzlich auf der Webseite durch farbige Hinterlegungen hervorgehoben.

Aus Anwendersicht besteht *Storyfinder* aus einem Browserplugin (vgl. Abb. 4.1), welches das Surfverhalten des Benutzers kontinuierlich überwacht und die Inhalte nach dem *Visual-Analytic Prinzip* grafisch aufbereitet anzeigt. Hierfür analysiert das Plugin den Artikeltext auf besuchten Webseiten und hebt darin enthaltene relevante Textteile (im Folgenden „Entitäten") direkt auf der Webseite hervor. Welche Textteile relevant sind und wie diese erkannt werden, wird ausführlich in Kapitel 5 beschrieben.
Neben der eigentlichen Webseite werden in einer Seitenleiste des Browsers die Entitäten sowie Beziehung zwischen den Entitäten visualisiert. Hierfür wird ein Knoten-Link-Diagramm verwendet, das zusätzlich zu den Entitäten der aktuellen Webseite auch weitere relevante Entitäten aus zuvor besuchten Seiten anzeigt. Dieses Knoten-Link-Diagramm bildet den sogenannten Wissensgraphen in *Storyfinder*.
Die im Wissensgraphen dargestellten Entitäten und Beziehungen sowie deren Quellen können vom Benutzer weiter untersucht, bearbeitet und ergänzt werden. Neben diesem lokalen Wissengraphen, der gezielt die Entitäten einer bestimmten

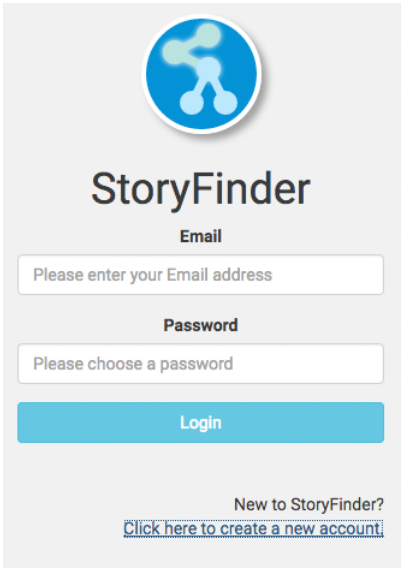


Abbildung 4.2.: Anmeldemaske von Storyfinder

Webseite in den Fokus rückt, kann zusätzlich ein globaler Wissensgraph angezeigt werden, der alle bisher erkannten Entitäten darstellt und somit das im Rahmen der Recherche gesammelte Wissen des Nutzers repräsentiert.
Auf Grund des beschränkten Bildschirmplatzes kann die Darstellung des Wissensgraphen auch auf einem oder mehreren externen Geräten ausgelagert werden und beispielsweise auf einem Tablet oder Smartphone im sogenannten „Second Screen Modus" angezeigt werden.
Im Folgenden werden die verschiedenen Bereiche von *Storyfinder* - die Einrichtung, der Seitenaufruf, die Exploration und die Bearbeitungsfunktionen - im Detail erläutert.

4.1.1 Einrichtung von Storyfinder

Aus Benutzersicht besteht *Storyfinder* nur aus einem Browserplugin für *Mozilla Firefox*. Dieses Plugin lässt sich per Drag & Drop in den Browser bewegen und wird anschließend automatisch installiert. Neben diesem Browserplugin gibt es noch mehrere serverseitige Komponenten. Eine detaillierte Installationsanleitung für diese Komponenten findet sich in Anhang A.1. Nach der Installation des Plugins wird in *Mozilla Firefox* eine Seitenleiste mit dem Anmeldebildschirm angezeigt (siehe Abb. 4.2).

Benutzeranmeldung

Da die serverseitigen Komponenten von *Storyfinder* als Mehrbenutzersystem konzipiert sind, also für die Verwendung durch mehrere Benutzer, erfolgt nach der Installation von *Storyfinder* zuerst die Registrierung und Anmeldung am System. Durch die Anmeldung können die Seiteninhalte in *Storyfinder* den einzelnen Benutzern zugeordnet werden und es wird sichergestellt, dass keine anderen Benutzer Zugriff auf den eigenen Wissensgraphen erhalten.
Benutzer werden in *Storyfinder* anhand Ihrer E-Mailadresse und einem Passwort authentifiziert. Benutzer können sich beim ersten Start des Plugins mit ihren Zugangsdaten anmelden, falls sie bereits ein *Storyfinder* Benutzerkonto besitzen. Andernfalls können sie auf der Registrierungsseite ein neues Benutzerkonto erstellen.
Die Zugangsdaten werden nach der Anmeldung dauerhaft im Browser gespeichert und müssen danach nicht erneut eingegeben werden.

Nach der erstmaligen Anmeldung zeigt *Storyfinder* einen leeren globalen Graphen an und wartet darauf, dass der Benutzer eine Seite aufruft.

4.1.2 Seitenaufruf

Sobald der Benutzer eine Seite im Browser aufruft, beginnt *Storyfinder* mit der Verarbeitung des Artikeltextes, wie ausführlich in Kapitel 4.2.1 beschrieben wird. Da die Verarbeitung einige Sekunden in Anspruch nehmen kann, signalisiert die Seitenleiste die Verarbeitung des Artikeltextes. Der Benutzer kann die Webseite in dieser Zeit bereits wie gewohnt lesen.

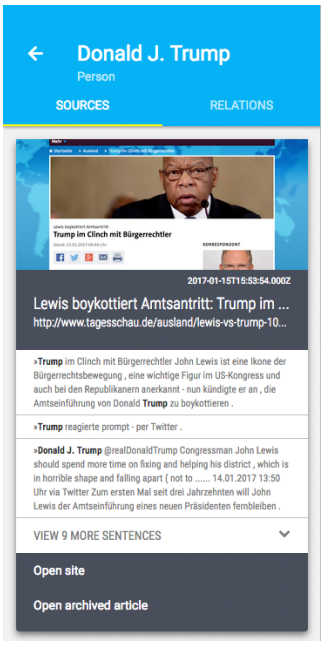


Abbildung 4.3.: Detailseite der Entität „Donald J. Trump"

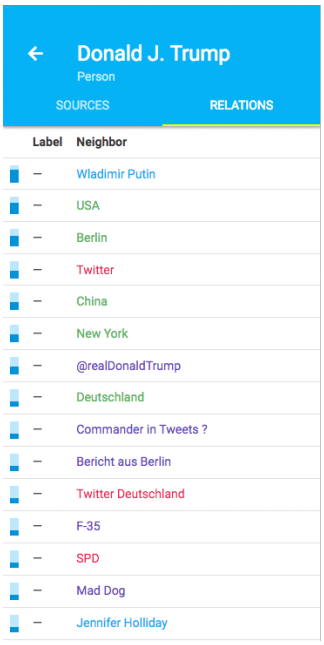


Abbildung 4.4.: Liste der Beziehungen der Entität „Donald J. Trump"

Sobald die Verarbeitung abgeschlossen ist, werden die Entitäten auf der Webseite farbig hinterlegt und hierdurch hervorgehoben. Die Farbe der Hinterlegung gibt hierbei die Art der Entität an (siehe Kapitel 6.2):

- Personen: blau
- Organisationen: rosa
- Orte: grün
- Schlüsselwörter: orange
- Sonstiges: lila

Zeitgleich wird in der Seitenleiste der Wissensgraph dieser Webseite angezeigt. Dort sind die wichtigsten Entitäten der Webseite sowie ggf. weitere relevante Entitäten aus anderen Webseiten enthalten. Die Entitäten, dargestellt als Kreise, bilden die Knoten eines Knoten-Link Diagramms. Stehen zwei Entitäten in Beziehung (siehe Kapitel 5.8), so werden diese beiden Knoten durch eine Linie verbunden dargestellt.
Da auf Grund des beschränkten Platzangebotes in der Seitenleiste oftmals nicht alle Entitäten angezeigt werden können, wird automatisch eine Vorauswahl getroffen (siehe Kapitel 6.4.1), um nur die wichtigsten Knoten anzuzeigen. Weitere Entitäten können auf Wunsch manuell eingeblendet werden oder erscheinen automatisch im Graphen, sobald sich der Mauszeiger über dem entsprechenden Textabschnitt auf der Webseite befindet.
Der Wissensgraph bietet zu diesem Zeitpunkt einen Überblick über den Artikel und ermöglicht von hier aus die weitere Exploration der Informationen, wie im nächsten Abschnitt erläutert wird.

4.1.3 Exploration

Der Wissensgraph bietet die Möglichkeit zur interaktiven Exploration der angezeigten Entitäten und Beziehungen. Durch das Anklicken einer Entität wird diese ausgewählt und dem Benutzer stehen weitere Aktionen zur Verfügung.

Detailansicht einzelner Entitäten und Beziehungen

Wurde eine Entität ausgewählt, so kann der Benutzer weitere Informationen zu diesem Element in der Detailansicht aufrufen. In dieser Detailansicht werden alle Webseiten und Sätze angezeigt, die diese Entität enthalten (vgl. Abb. 4.3). Von hier aus lassen sich die zugehörigen Webseiten aufrufen. Zusätzlich kann über die Archivfunktion der ursprüngliche Artikeltext erneut angezeigt werden, auch wenn der Text auf der Quellwebseite mittlerweile geändert oder entfernt wurde.

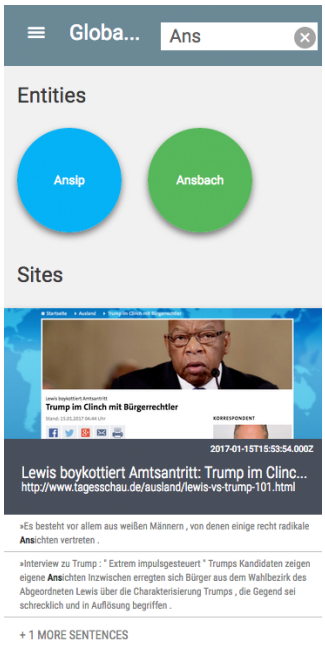


Abbildung 4.5.: Ergebnisse der Suche nach dem Suchbegriff „Ans". Die Ergebnisliste enthält sowohl Entitäten („Ansip" und „Ansbach") als auch Seiten mit Sätzen, die diesen Begriff beinhalten.

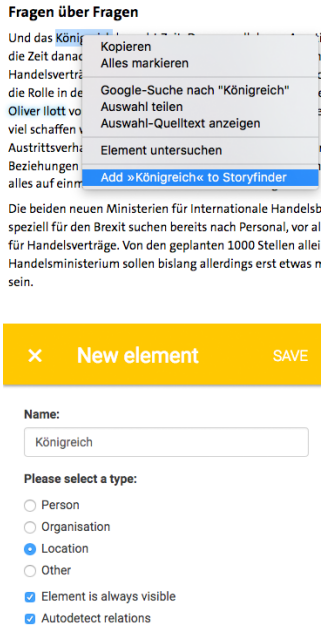


Abbildung 4.6.: Bild oben: Kontextmenü zum Hinzufügen einer neuen Entität; Bild unten: Eingabemaske in *Storyfinder* zur Auswahl weiterer Details beim Hinzufügen

Neben den Webseiten kann in der Detailansicht einer Entität die Liste aller verknüpften Entitäten angezeigt werden (vgl. S. 21, Abb. 4.4). Diese Liste ist nach Häufigkeit (siehe Kapitel 6.5) sortiert und zeigt über ein kleines Diagramm vor der Entität deren Relevanz an. Wurde die Beziehung bereits beschriftet, so wird auch die Beschriftung in dieser Ansicht angezeigt.
Klickt der Benutzer in dieser Detailansicht auf eine verknüpfte Entität oder im Wissensgraphen auf eine Kante zwischen zwei Entitäten, so werden die Quellen für diese Beziehung angezeigt. Zusätzlich zu den Quellen werden alle Sätze angezeigt, in denen diese Beziehung gefunden wurde.

Globaler Graph und Gruppen von Graphen

Neben des Wissensgraphen einer einzelnen Webseite (Lokaler Graph) kann auch ein Graph von mehreren Webseiten (Gruppengraph) oder aller bisher besuchter Webseiten (Globaler Graph) angezeigt werden. Diese Graphen verhalten sich analog zum Gruppengraph, enthalten jedoch Entitäten aus mehreren Webseiten.
Der globale Graph kann direkt vom lokalen Graphen aus aufgerufen werden. Von hier aus erhält der Benutzer auch Zugriff auf den Verlauf, eine chronologisch sortierte Liste aller bisher besuchten Webseiten. Aus diesem Verlauf heraus können die lokalen Graphen der einzelnen Webseiten erneut aufgerufen werden oder durch Auswahl mehrerer Seiten ein Gruppengraph dieser Webseiten erstellt werden.

Suche

Zum Wiederfinden bestimmter Entitäten oder beliebigen Textausschnitten auf den besuchten Seiten, steht eine Suchfunktion zur Verfügung. Diese Suchfunktion durchsucht alle bisher gelesenen Artikel nach dem eingegebenen Suchbegriff. In den Ergebnissen (vgl. Abb. 4.5) werden sowohl passende Entitäten als auch relevante Sätze angezeigt. Von hier aus können die Details der Entitäten, zugehörigen Graphen der Seiten oder die Webseiten selbst aufgerufen werden.

4.1.4 Bearbeitungsfunktionen

Neben den Explorationsfunktionen stehen dem Benutzer verschiedene Methoden zur Bearbeitung des Netzwerkes zur Verfügung:
Nach Auswahl einer Entität kann diese gelöscht werden, in dem der Knoten per Drag & Drop auf den angezeigten Papierkorb gezogen wird. Ebenso kann der Knoten auf einen anderen Knoten gezogen werden, um die beiden Entitäten zu einer Entität zu verschmelzen, falls es sich hierbei um Synonyme handelt. In der Detailansicht der Beziehung zweier Knoten können diese Beziehungen beschriftet werden. Dies ist sowohl für die Beziehungen im Allgemeinen möglich, als

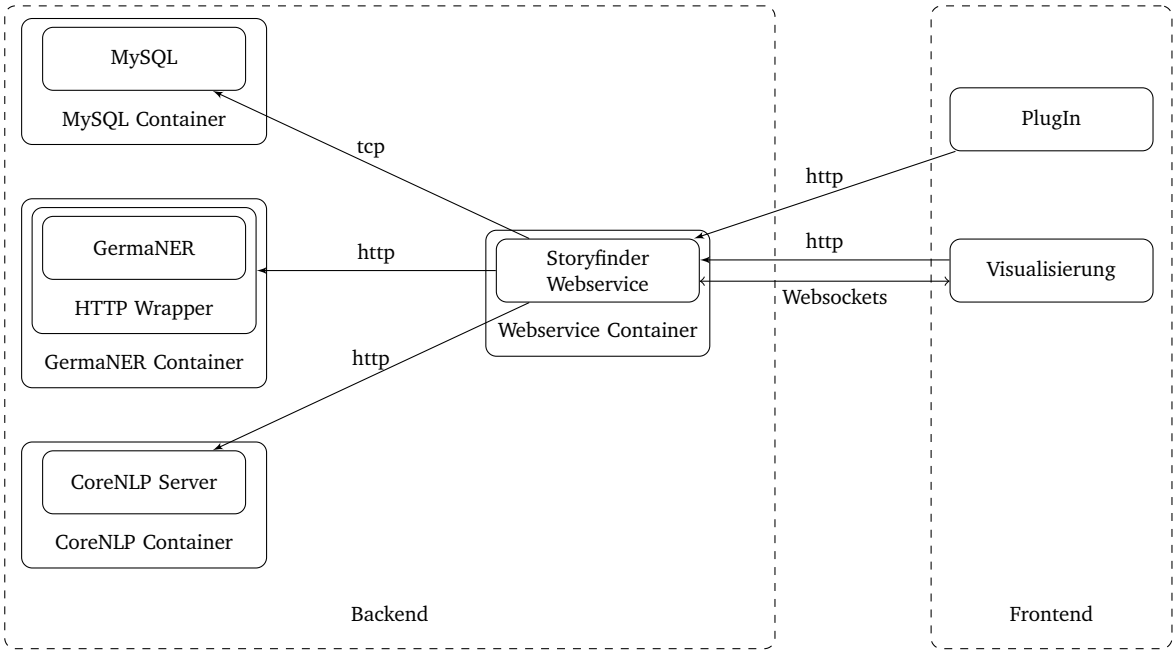


Abbildung 4.7.: Komponenten von *Storyfinder*

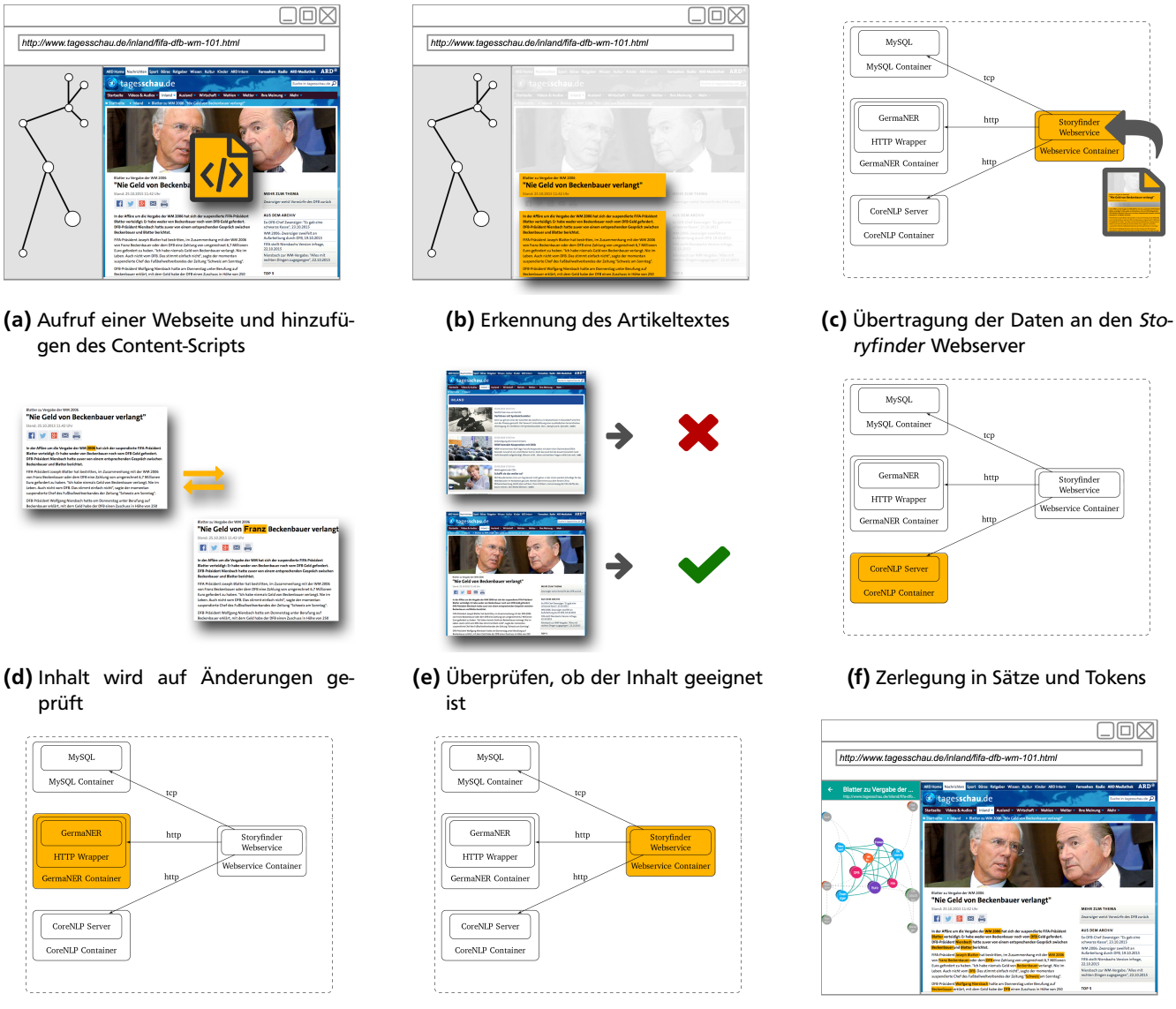
auch für die Beziehung in einem einzelnen Satz.
Für den Benutzer relevante Elemente, die von *Storyfinder* nicht automatisch erkannt wurden, können manuell zum Graphen hinzugefügt werden. Hierfür kann der Benutzer entweder den entsprechenden Button in der Seitenleiste anklicken und den Begriff eingeben oder er markiert den Begriff auf der Webseite und fügt diesen über das Kontextmenü zum Netzwerk hinzu (vgl. S. 22, Abb. 4.6). Anschließend erhält der Benutzer ein Eingabeformular, in dem er den Typ der Entität auswählen kann und festlegen kann, ob auch bereits gelesene Artikel nach diesem Element durchsucht werden sollen. Sobald der Benutzer den Button zum Speichern anklickt, wird die Entität zum Netzwerk hinzugefügt und ggf. auch in allen bereits gelesenen Artikeln hervorgehoben. Ebenso wird das Element zukünftig automatisch hervorgehoben, falls ein neuer Artikel diese Entität enthält.

4.2 Serverseitige Komponenten von Storyfinder

Nachdem im vorherigen Unterkapitel *Storyfinder* aus Benutzersicht betrachtet wurde, wird in diesem Unterkapitel die technische Seite von *Storyfinder* beschrieben. Bei *Storyfinder* handelt es sich um eine Mehrbenutzer Client-Server Anwendung auf Basis von Webtechnologien. Das Storyfinder-Browserplugin stellt die clientseitige Komponente dar und wird ausführlich im Kapitel 4.4 beschrieben. Serverseitig besteht *Storyfinder* aus vier Komponenten, wobei hierbei der *Storyfinder-Webserver* im Mittelpunkt steht. Diese Komponente steuert die Kommunikation zwischen allen weiteren Komponenten, führt die Datenextraktion aus, liefert alle nötigen Daten für die Visualisierung an den Browser und stellt die Benutzerverwaltung zur Verfügung. Bei den weiteren Komponenten handelt es sich um den *Stanford CoreNLP Webservice*, die Eigennamenerkennung *GermaNER* sowie eine *MySQL-Datenbank* für die Datenspeicherung. In Abbildung 4.7 sind die Komponenten dargestellt.

4.2.1 Beziehung zwischen den Storyfinder-Komponenten

Das Zusammenspiel und die Funktion der verschiedenen Komponenten lässt sich am besten anhand des Aufrufs einer Webseite und der damit verbundenen Extraktion der Entitäten erläutern.
In einem ersten Schritt ruft der Benutzer eine Webseite im Browser auf, in dem das *Storyfinder-Plugin* installiert ist. Nachdem die Seite geladen wurde, fügt das Plugin ein Content-Script in die aufgerufene Webseite ein und kann somit im Kontext der Webseite arbeiten, d.h. auf die Seiteninhalte zugreifen und diese verändern (vgl. S. 24, Abb. 4.8a).
Dieses Content-Script extrahiert den Artikeltext der Webseite, trennt also den Artikeltext von anderen Elementen wie beispielsweise Navigation und Werbung (vgl. S. 24, Abb. 4.8b), erstellt Bilder der Webseite und des Artikels und sendet diese Daten an den *Storyfinder-Webserver* (vgl. S. 24, Abb. 4.8c).
Der *Storyfinder-Webserver* empfängt diese Daten und prüft zuerst, ob die Seite bereits in der Datenbank vorhanden ist und falls ja, ob sich deren Inhalt markant gegenüber der gespeicherten Version verändert hat (vgl. S. 24, Abb. 4.8d).



(a) Aufruf einer Webseite und hinzufügen des Content-Scripts

(b) Erkennung des Artikeltextes

(c) Übertragung der Daten an den *Storyfinder* Webserver

(d) Inhalt wird auf Änderungen geprüft

(e) Überprüfen, ob der Inhalt geeignet ist

(f) Zerlegung in Sätze und Tokens

(g) Erkennung von Eigennamen

(h) Extraktion vom Schlüsselwörtern und Beziehungen

(i) Highlighting auf Webseite und Anzeige des Wissensgraphen

Abbildung 4.8.: Beziehung zwischen den Storyfinder-Komponenten beim Aufruf einer Webseite

Hierfür wird die Kosinus-Distanz zwischen den im Artikeltext enthaltenen Tokens und den bereits gespeicherten Tokens berechnet und über einen Grenzwert entschieden, ob eine markante Änderung vorliegt (siehe Kapitel 5.4).
Falls es sich um eine neue Seite handelt oder eine markante Veränderung vorliegt, wird geprüft, ob die Seite für *Storyfinder* geeignet ist (vgl. Abb. 4.8e). Webseiten, die keine einzelnen Artikel enthalten, wie beispielsweise Übersichtsseiten, sind für *Storyfinder* ungeeignet und sollen nicht weiter verarbeitet werden. Falls die Seite als relevant eingestuft wird, beginnt die eigentliche Datenextraktion. Andernfalls hat der Benutzer die Möglichkeit, eine Datenextraktion manuell zu starten.
Für die Datenextraktion wird zuerst eine linguistische Vorverarbeitung des Artikeltextes vorgenommen. Hierfür wird der Text an den *CoreNLP Webservice* gesendet und dort in einzelne Sätze und Tokens unterteilt (vgl. Abb. 4.8f). Anschließend werden die Daten für die Extraktion von Eigennamen an *GermaNER* gesendet (vgl. Abb. 4.8g).
Nachdem die Eigennamen extrahiert wurden, werden die Daten im *Storyfinder* Webserver zusammengeführt und mit gespeicherten Daten aus der Datenbank ergänzt. Zudem wird eine Extraktion von Schlüsselwörtern über die TFIDF-Werte der Mono-, Bi- und Trigrammen vorgenommen.
Im Anschluss werden die Beziehungen der Entitäten über Satzkookurrenzen ermittelt und die extrahierten Daten in der Datenbank gespeichert (vgl. Abb. 4.8h). Abschließend werden die gewonnenen Daten an das Plugin gesendet, wo die

Entitäten auf der Webseite hervorgehoben und der Wissensgraph für die Webseite angezeigt wird (vgl. S. 24, Abb. 4.8i). Im folgenden Abschnitt werden technische Details der verschiedenen Komponenten erläutert.

4.2.2 Technische Details der Komponenten

Der in Javascript geschriebene und über NodeJS ausgeführte *Storyfinder* Webserver bildet die Kernkomponenten des *Storyfinder-Servers*. Der Server enthält die folgenden Resourcen:

- User: Über diese Resource erfolgt die Registrierung, Anmeldung und Sitzungsverwaltung der einzelnen Benutzer. Für die Registrierung werden Benutzer anhand ihrer E-Mailadresse und eines Passwortes authentifiziert.
- Entity: Diese Resource ermöglicht das manuelle Hinzufügen, Ändern und Löschen von Entitäten. Zusätzlich können hierüber Daten einzelner Entitäten, wie beispielsweise deren Quellen oder Relationen abgerufen werden. Die Suche nach bestimmten Entitäten ist ebenfalls über diese Resource verfügbar.
- Relation: Die Beziehung zwischen Entitäten können über diese Resource verwaltet werden.
- Site: Das Verwalten und Abrufen von Daten über besuchte Seiten ist über diese Resource möglich. Beim Aufruf einer neuen Seite wird hierüber die neue Seite registriert und die Datenextraktion gestartet.
- Graph: Die Graph-Resource stellt die Daten für die verschiedenen Graphen-Visualisierungen zur Verfügung.
- Group: Über die Group-Resource können Gruppen von Seiten erstellt oder abgerufen werden.

Eine detaillierte Auflistung aller Resourcen ist in Anhang A.2 zu finden. Die Definition der Resourcen wurde über das Express-Framework [29] realisiert. Die Kommunikation zwischen Plugin, Visualisierung und allen weiteren Komponenten erfolgt immer über den *Storyfinder* Webserver und niemals direkt zwischen den einzelnen Komponenten.

CoreNLP und GermaNER

Die komplexeren sprachtechnologischen Extraktionsschritte werden durch zwei externe Komponenten durchgeführt. Für das Sentence Splitting und zur Tokenisierung wird der *Stanford CoreNLP Server* [47] verwendet. Der *CoreNLP Server* stellt selbst einen eigenen Webservice bereit, der vom *Storyfinder* Webserver aufgerufen wird.
Die Erkennung von Eigennamen erfolgt über *GermaNER* [8] und wird ebenfalls vom *Storyfinder* Webserver aufgerufen. Im Rahmen dieser Arbeit wurde ein einfacher Wrapper für *GermaNER* implementiert, der Texte über eine HTTP Schnittstelle empfängt und als Antwort die erkannten Eigennamen zurückliefert.

Deployment

Die serverseitigen Komponenten *Storyfinder-Webserver*, *CoreNLP*, *GermaNER* und *MySQL* sind in einzelne *Docker Container* [49] ausgelagert und können somit einfach installiert und nahezu unabhängig voneinander betrieben werden.

4.2.3 Anbindung des Frontends

Das clientseitige Frontend greift auf den *Storyfinder* Webservice über HTTP und WebSockets zu.
Das Plugin kommuniziert ausschließlich über HTTP-Anfragen mit dem Webservice. Die Anfragen gehen hierbei immer vom Plugin aus. Die Graphen-Visualisierung, welche innerhalb der Seitenleiste oder in einem externen Browser angezeigt wird, verwendet ebenfalls HTTP-Anfragen zur Kommunikation mit dem Webservice.
Da für den Server hierüber keine Möglichkeit besteht, den Client über neue Daten zu informieren (Server PUSH), wird zusätzlich eine WebSocket-Verbindung zwischen Visualisierung und Webservice aufgebaut. WebSocket Verbindungen ermöglichen eine bidirektionale Kommunikation, so dass die verbundenen Clients über Änderungen, wie beispielsweise neue Entitäten oder gelöschte Knoten auf anderen Geräten, durch den Webservice informiert werden können.
Da WebSockets bisher nicht von allen gängigen Browsers unterstützt werden, wurde für die Implementierung der Schnittstelle in *Storyfinder* die socket.io [54] Bibliothek verwendet. Diese ermöglicht neben regulären WebSocket Verbindungen auch verschiedene Fallback Strategien, wie beispielsweise Polling über HTTP.

4.3 Datenspeicherung

Die Datenspeicherung erfolgt in einer relationalen *MySQL Datenbank* und im Dateisystem. Im Folgenden werden zuerst die wichtigsten Datenbanktabellen und die darin gespeicherten Daten beschrieben. Anschließend wird erläutert, wie binäre Daten wie beispielsweise die Vorschaubilder der Webseiten im Dateisystem auf dem Server gespeichert werden.

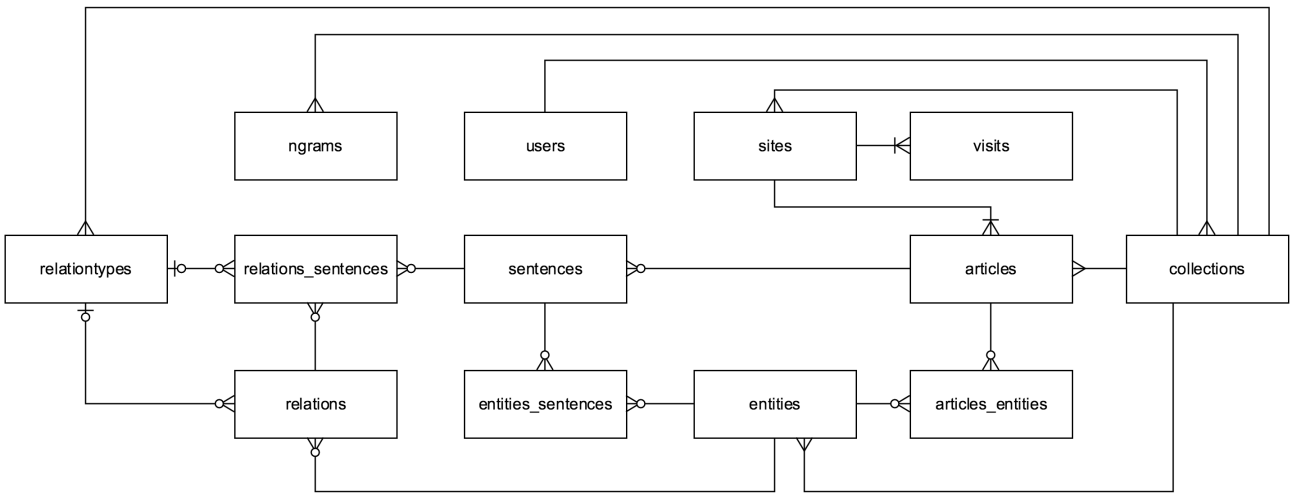


Abbildung 4.9.: Datenbanktabellen und deren Beziehungen

4.3.1 Überblick über die Tabellen in Storyfinder

Für die Datenspeicherung werden die 13 im ERM in Abbildung 4.9 dargestellten Tabellen verwendet. Ein detailliertes Schema der Datenbank befindet sich in Anhang A.3.

Sites

Die Speicherung der aufgerufenen Webseiten erfolgt in der Tabelle „sites". Für jede aufgerufene URL wird ein Eintrag in dieser Tabelle angelegt, der neben der URL den Host, Seitentitel, URL des Favicons, Zeitstempel des letzten Aufrufs, die dominante Farbe der Seite, sowie einen Hash des Artikels enthält.
Bei diesem Hash handelt es sich um den MD5-Hash des reinen Artikeltextes ohne Markup. Wird dieselbe URL erneut aufgerufen, wird der Hash des Artikels der aufgerufenen Seiten mit dem gespeicherten Hash verglichen. Nur falls die beiden Hashes verschiedenen sind, findet eine detaillierte Prüfung auf Änderungen an der Seite statt (siehe Kapitel 5.4). Falls eine solche Änderung vorliegt, wird nur erneut ausgeführt und der enthaltene Artikel als neuer Artikel behandelt.
Der Artikeltext selbst wird nicht in der Tabelle „sites" gespeichert, sondern separat in der Tabelle „articles". Hierdurch ist es möglich, dass es für eine URL und somit einen Eintrag in „sites" mehrere unterschiedliche Artikeltexte gibt, falls sich diese ändern. Ein Beispiel hierfür wäre eine Webseite, die immer den aktuellsten Artikel unter der gleichen URL anzeigt, z.B. eine Zusammenfassung der wichtigsten Ereignisse des heutigen Tages.

Articles

Die Artikeltexte werden in der Tabelle „articles" gespeichert. Die Artikel sind immer mit einer Webseite aus der Tabelle „sites" verknüpft und enthalten sowohl den Text mit HTML Markup sowie den reinen Artikeltext.

Sentences

Falls Entitäten manuell vom Benutzer hinzugefügt werden, muss überprüft werden, in welchen Sätzen diese Entitäten enthalten sind und welche weiteren Entitäten im selben Satz enthalten sind, so dass automatisch die Satzkookkurrenzen und hierüber die Beziehungen zwischen diesen Entitäten extrahiert werden können. Hierfür werden die einzelnen Sätze eines Artikels in der Tabelle „sentences" gespeichert.

Visits

Bei jedem Seitenaufruf wird ein Eintrag mit Zeitstempel in der Tabelle „visits" angelegt, der den Besuch der Webseite speichert. Hierfür wird der Eintrag mit der aufgerufenen Seite in der Tabelle „sites" verknüpft. Zudem wird die ID der Referrer Seite gespeichert, also der Seite, von der aus die aktuelle Seite bei diesem Besuch aufgerufen wurde.

Entities

Die extrahierten oder manuell hinzugefügten Entitäten werden in der Tabelle „entities" abgelegt. Neben dem String der Entität wird die Anzahl der Wörter die diese Entität umfasst, der Entitätstyp, eine Bezeichnung für die Entität sowie der Zeitstempel, wann diese Entität zuletzt in einem aufgerufenen Artikel enthalten war, gespeichert. Über diesen Zeitstempel lassen sich die Entität nach deren Aktualität gewichtet. Im Rahmen dieser Arbeit wurde nur die Gewichtung Anhang des

TFIDF Wertes implementiert, der Zeitstempel wird jedoch bereits für eine mögliche zukünftige Erweiterung gespeichert (vgl. 8.2).
Besteht eine Entität aus mehreren Wörtern, so enthält das Feld „value" nur den ersten Token, da im ersten Schritt bei der Suche nach Entitäten nur einzelne Tokens verglichen werden. In einem zusätzlichen Feld („multiword") ist der komplette String der Entität gespeichert.
Falls zwei oder mehrere Entitäten zu einer Entität zusammengefasst werden, so wird eine Entität zum Master und alle anderen Entitäten zum Slave. Nur die Master-Entität wird später in der Visualisierung angezeigt. Bei den Slave-Entitäten wird im Feld „master_id" die ID der Master-Entität eingetragen. Wird eine Slave Entität in einem Artikel gefunden, so wird anhand dieser master_id erkannt, dass anstelle dieser Entität die Master-Entität im Graphen ausgegeben werden soll.

Relations

Die Beziehungen zwischen zwei Entitäten werden in der Tabelle „relations" gespeichert. Für jede Beziehung wird der primäre Relationstyp (Bezeichnung der Beziehung), sowie ob die Beziehung automatisch erkannt wurde oder vom Benutzer hinzugefügt wurde, gespeichert. Die Speicherung, ob es sich um eine von Benutzer hinzugefügte Beziehung handelt, ermöglicht eine höhere Gewichtung dieser Beziehung, falls es mehrere Beziehungen zwischen zwei Entitäten gibt, da davon auszugehen ist, dass eine manuell hinzugefügte Beziehung für den Benutzer wichtiger ist wie automatisch erkannte Beziehungen.
Neben diesen allgemeinen Beziehungen, die unabhängig vom zugehörigen Satz sind, wird in der Tabelle „relations_sentences" für jeden Satz in dem diese Beziehung zwischen zwei Entitäten enthalten ist, ein Eintrag erzeugt. Diese satzabhängigen Beziehungen können je nach Satz anderen Relationstyp (Bezeichnung) verwenden, wie in den folgenden beiden Beispielsätzen deutlich wird:

1) „Angela Merkel ist zum Staatsbesuch in Frankreich."

2) „Angela Merkel verurteilt die Anschläge in Frankreich."

In beiden Fällen stehen „Angela Merkel" und „Frankreich" in Beziehung. Die Bezeichnungen für die Beziehungen sind jedoch unterschiedlich. Im ersten Beispiel wäre eine mögliche Bezeichnung „[Angela Merkel] besucht [Frankreich]", wogegen im zweiten Beispiel z.B. die Bezeichnung „[Angela Merkel] verurteilt Anschläge in [Frankreich]" möglich wäre.
Da aufgrund der Übersichtlichkeit im Wissensgraph nur eine Beziehung angezeigt werden kann, muss bei mehreren Beziehungen zwischen zwei Entitäten eine Beziehung als primärer Relationstyp festgelegt werden. Dieser primäre Relationstyp ist direkt in der Tabellen „relations" angegeben.
Die Relationstypen werden in der separaten Tabelle „relationtypes" gespeichert. Für jede Bezeichnung wird ein Eintrag in dieser Tabelle erstellt. Mehrere Beziehungen zwischen verschiedenen unterschiedlichen Entitäten mit gleicher Bezeichnung sind mit demselben Entitätstyp verknüpft. Hierüber wäre es möglich, ähnlich wie in [42] Suchmuster für diese Relationstypen zu lernen. Bisher werden solche Suchmuster in *Storyfinder* jedoch nicht verwendet.

Collections

Da *Storyfinder* als Mehrbenutzer-System konzipiert ist, muss eine Zuordnung der Daten zu den einzelnen Benutzern erfolgen. In *Storyfinder* verwenden wir hierfür Collections, die in der gleichnamigen Tabelle gespeichert sind. Jede Collection ist einem Benutzer zugeordnet, wobei jeder Benutzer mehrere solcher Collections verwenden kann. Die restlichen Daten (Sites, Entities, Relationtypes usw.) sind an eine bestimmte Collection gebunden, so dass jede Collection eine eigene, unabhängige Sammlung von Seiten und Entitäten darstellt.

Ngrams

Die Speicherung der Mono-, Bi- und Trigramme für die Schlüsselworterkennung erfolgt in der Tabelle Ngrams. Jedes erkannte N-Gramm in den Artikeltexten wird in dieser Tabelle gespeichert. Neben den N-Grammen selbst wird zusätzlich die Anzahl der Dokumente gespeichert, in denen das jeweilige N-Gramm enthalten ist. Über diese Dokumentfrequenz und die Termfrequenz aus den jeweiligen Artikeln ist eine Schlüsselworterkennung möglich, wie in Kapitel 5.7 erläutert wird.

4.3.2 Speicherung binärer Daten

Neben den zuvor beschriebenen Daten die in der Datenbank gespeichert werden, speichert *Storyfinder* auch binäre Daten. Bei diesen Daten handelt es sich um die Vorschaubilder der Webseiten und Artikel, die als PNG Datei gespeichert werden. Diese Bilder werden nicht in der Datenbank sondern direkt als einzelne Dateien im Dateisystem abgelegt. Die Speicherung erfolgt für jede Collection in einem Unterordner mit der Collection-ID. In diesem Ordner befinden sich jeweils zwei weitere Unterordner „websites" und „articles", in denen die zugehörigen Bilder im PNG-Format gespeichert sind. Als Dateiname wird die ID des zugehörigen Datenbankeintrags in der Tabelle „sites" bzw. „articles" verwendet.

Browser	Marktanteil	WebExtensions	Binäre Erweiterungen	Sonstige
Mozilla Firefox	34,23 %	√ *Beta*	√	√ *Add-on SDK* Legacy/XUL *(deprecated)*
Google Chrome	32,67 %	√	×	×
Microsoft IE	12,62 %	×	√	×
Apple Safari	11,29 %	√	×	×
Microsoft Edge	3,37 %	√ *geschlossene Beta*	×	×

Zeichenerklärung: √ unterstützt × nicht unterstützt

Tabelle 4.1.: Unterstützte Schnittstellen für Erweiterungen in den Desktopversionen der aktuellen Standardbrowser

4.4 Einbindung von Storyfinder in den Browser

Als Basis für die Darstellung von *Storyfinder* und die Anbindung des Plugins wird ein Webbrowser benötigt. Die Anzeige des globalen Graphen ist in allen aktuellen Standardbrowsern möglich und erfolgt wie bei anderen Webseiten durch Aufruf der URL.
Das Plugin zur Überwachung des Surfverhaltens und der direkten Darstellung auf den aufgerufenen Webseiten muss hingegen im Browser installiert werden und ist deshalb browserabhängig. Die Schnittstellen für die Plugin-Entwicklung unterscheiden sich bei den verschiedenen Browsern. Zudem stehen teils auch innerhalb eines Browsers mehrere Plugin Schnittstellen zur Verfügung, die verschieden umfangreiche Änderungen am Browser ermöglichen.
In diesem Unterkapitel wird die Einbindung des *Storyfinder-Plugins* in den Browser erläutert. Hierfür erfolgt zuerst ein Überblick über die Schnittstellen der Standardbrowser[1], die in Tabelle 4.1 aufgelistet sind. Anschließend wird das für *Storyfinder* verwendete *Add-on SDK* von *Mozilla Firefox* sowie die Einbindung von *Storyfinder* in den Browser im Detail erläutert.

4.4.1 Vergleich verschiedener Browserschnittstellen

Die meisten Browser stellen eine oder mehrere Schnittstellen für Erweiterungen zur Verfügung, die jedoch meist untereinander nicht kompatibel sind. Im Folgenden werden die gängigen Schnittstellen betrachtet und analysiert, inwieweit diese für *Storyfinder* geeignet sind.

WebExtensions

Unter der Bezeichnung WebExtensions [1] bieten die WebKit basierenden Browser *Apple Safari* und *Google Chrome* sowie seit jüngstem auch *Mozilla Firefox* und *Microsoft Edge* eine einheitliche Schnittstelle zur Entwicklung von Browserplugins an. Während der Entwicklung von *Storyfinder* befand sich die Schnittstelle für *Mozilla Firefox* noch im Beta-Status und auch in *Microsoft Edge* war diese für Standarduser noch nicht verfügbar:

> *We're currently building and validating our extension platform and documentation with the help of a small number of extension authors. Please stay tuned for future updates!*[2]

WebExtensions basieren auf Webtechnologien und verwenden HTML und Javascript als Basis für die Plugin Entwicklung. Über diese Schnittstelle können die aufgerufenen Webseiten ausgelesen und verändert, das Kontextmenü ergänzt und Buttons sowie Panels zum Browserinterface hinzugefügt werden. WebExtensions bieten jedoch keine Möglichkeit zur dauerhaften Anzeige einer Seitenleiste für die Darstellung des *Storyfinder* Wissensgraphen.

Im Rahmen dieser Arbeit wurde deshalb getestet, ob sich der Wissensgraph auch direkt auf einer Webseite einbinden lässt, anstelle von der Einbindung in einer separaten Seitenleiste. Die Einbindung direkt auf der Webseite wäre auch bei der Verwendung von WebExtensions möglich. In diesem Fall wird beim Laden jeder Webseite durch das Plugin eine

[1] Gemessen anhand der fünf meistgenutzten Destkop, Tablet und Konsolenbrowser in Deutschland im Jahr 2016 [62].
[2] https://developer.microsoft.com/en-us/microsoft-edge/platform/documentation/extensions/

Javascript-Datei in den Seitenkontext eingefügt. Über dieses sogenannte Content-Script ist es anschließend möglich, die Inhalte der Webseite auszulesen und zu verändern. Darüber hinaus kann das Content-Script über Nachrichten mit dem Plugin kommunizieren und hierüber auf API Funktionen zugreifen.
Zur Anzeige des Wissensgraphen wird durch das Content-Script ein IFrame zur Webseite hinzugefügt. Innerhalb dieses IFrames wird anschließend eine HTML-Seite angezeigt, die den Wissensgraphen enthält.
Bei dieser Art der Einbindung wurden jedoch zwei Probleme festgestellt:

- Der IFrame befindet sich innerhalb des Document Object Models der Webseite und kann somit von anderen Elementen auf der Seite überlagert werden. Um dieses Verhalten zu steuern, kann jedem Element einer Webseite über die CSS Eigenschaft „z-Index" eine Tiefe (Stapel-Index) zugeordnet werden. Diese Eigenschaft kann einen beliebigen Integer Wert annehmen. Stellt man sich die einzelnen Elemente einer Webseite als Stapel vor, auf den man von oben schaut, so gibt der z-Index an, in welcher Höhe das aktuelle Element auf dem Stapel liegt. Überschneiden sich zwei Elemente, so wird das Element mit dem höheren z-Index vor dem Element mit niedrigerem z-Index angezeigt. Bei gleichem oder nicht gesetzten z-Index entscheidet die Position im DOM, so dass Elemente die weiter unten im DOM stehen vor Elemente die weiter oben im DOM stehen, angezeigt werden.
  Da die CSS Spezifikation keinen Wertebereich für Integer Werte definiert, sind je nach System unterschiedliche Maximalwerte möglich. In den meisten Fällen wird ein 32 Bit signed Integer verwendet, in bestimmten Browsern[3] sind auch 64 Bit signed Integer Werte möglich. Durch die Verwendung eines sehr hohen z-Index kann dennoch sichergestellt werden, dass der Wissensgraph vor anderen Elementen auf der Webseite angezeigt wird.
  Problematisch ist jedoch die Anzeige von Werbung oder anderen dynamischen Inhalten, die nachträglich durch Javascript nachgeladen werden. Für solche Elemente wird oftmals vor der Einbindung der höchste gesetzte z-Index auf der Seite ermittelt. Anschließend wird das neue Element mit einem höheren z-Index in das Dokument eingefügt. Es wird hierdurch sichergestellt, dass diese Elemente, also z.B. Werbebanner, vor allen anderen Elementen der Seite liegen und somit auch tatsächlich sichtbar sind.
  Für die Einbindung des Wissensgraphen auf der Seite ist dieses Verhalten jedoch problematisch, da dynamisch nachgeladene Inhalte, den zu diesem Zeitpunkt bereits in die Seite eingebundenen Wissensgraphen, überlagern können. Hierdurch wird der Wissensgraph häufig von Werbung überdeckt.

- Bei einer regulären Seitenleiste eines Browsers wird automatisch der Inhaltsbereich der Webseite verkleinert, sobald die Seitenleiste angezeigt wird. Bei der Einbindung des Wissensgraphen direkt auf der Webseite findet eine solche Verkleinerung des Inhaltsbereiches nicht automatisch statt. Über verschiedene CSS Eigenschaften kann zwar ein ähnlicher Effekt erzielt werden, bei absolut positionierten Elementen auf der Webseite oder spätestens bei dynamisch per Javascript eingefügten Elementen kommt es jedoch zu Fehlern in der Darstellung. In diesen Fällen kann es passieren, dass Teile der Webseite vom Wissensgraphen überlagert werden oder bestimmte Elemente nicht mehr an der korrekten Stelle im Seitenlayout angezeigt werden.

WebExtensions stellen ein vielversprechendes Konzept dar, sind aber auf Grund der fehlenden Seitenleiste und des Beta-Status in einigen Browsern für die Einbindung von *Storyfinder* nicht ideal. Da für die beiden Browser *Apple Safari* und *Google Chrome* keine weiteren Schnittstellen zur Verfügung stehen, müsste für eine Einbindung von *Storyfinder* in diese Browser dennoch WebExtensions verwendet werden. Die meisten Komponenten von *Storyfinder* könnten hierfür direkt verwendet werden. Eine Portierung auf WebExtensions wurde im Rahmen dieser Arbeit jedoch nicht durchgeführt.

Microsoft Internet Explorer

Erweiterungen für den *Microsoft Internet Explorer* werden nicht mit Web-Technologien entwickelt, sondern in Form von dynamischen Bibliotheken (DLL). Die Entwicklung ist daher prinzipiell in allen Programmiersprachen möglich, die entsprechende Bibliotheken erzeugen können, wie z.B. C++, C#, Visual Basic oder Delphi.
Durch diese Form der Einbindung sind die Erweiterungen sehr performant, besonders im Vergleich zu Erweiterungen die über Scriptsprachen (z.B. WebExtensions) entwickelt wurden. Zudem ist ein nahezu unbeschränkter Zugriff auf alle Systemresourcen möglich. Die Erweiterungen sind jedoch in den meisten Fällen nicht mit anderen Browsern und anderen Systemen kompatibel und müssen daher in verschiedenen Versionen gepflegt werden.
Die Schnittstellen für den Internet Explorer unterteilen sich in mehrere unterschiedliche Komponenten. Für *Storyfinder* sind die beiden Komponenten Browser Helper Objects (BHO) [25] und Explorer Bars [51] relevant.
Über Browser Helper Objects lassen sich Änderungen am eigentlichen Seiteninhalt vornehmen, um beispielsweise gefundene Entitäten direkt auf der Seite hervorzuheben. BHOs werden als COM-Server implementiert und können auf Browserevents reagieren, wie beispielsweise das Laden einer neuen Seite. Zudem können BHOs auf das Document Object Model der aufgerufenen Seite zugreifen und dieses verändern. Das Hinzufügen einer Seitenleiste die den *Storyfinder* Graphen enthält, ist über Explorer Bars möglich. Explorer Bars sind ebenfalls in Form von COM-Servern realisiert und verwenden Band Objects zur Implementierung des Inhalts.

[3] z.B. 64 Bit Webkit Browser

Auf Grund des vergleichsweisen hohen Aufwands für die Implementierung der *Storyfinder* Benutzeroberfläche über Band Objects und die Inkompatibilität der Technik mit anderen Browsern, wurde *Storyfinder* nicht als Erweiterung für den Internet Explorer entwickelt. Microsoft rät zudem selbst zur Entwicklung von Erweiterungen mit WebExtensions für den neuen Browser Microsoft Edge:

> *„Developers are encouraged to use Microsoft Edge, the new default browser built for Windows 10." [16]*

Mozilla Firefox

Neben WebExtensions stehen in der aktuellen Version von *Mozilla Firefox*[4] drei weitere Schnittstellen für Erweiterungen zur Verfügung [50]. Über die Legacy Extensions ist ein tiefer Eingriff in die Benutzeroberfläche und Funktionen des Browsers möglich. GUI Elemente können, wie in *Mozilla Firefox* selbst, über die von Mozilla entwickelte XML User Interface Language (XUL) erstellt werden. Die Anwendungslogik wird durch Javascript beschrieben. Da Mozilla jedoch bereits angekündigt hat, diese Schnittstelle in Zukunft nicht mehr zu unterstützen, ist diese für *Storyfinder* nicht geeignet:

> *We are planning to deprecate the use, in Firefox, of the techniques described in this document.*
> *Don't use these techniques to develop new add-ons. Use WebExtensions or the Add-on SDK instead [68].*

Als weitere Schnittstelle sind binäre Erweiterungen in C++ möglich, die einen noch tieferen Eingriff in den Browser ermöglichen, wie dies durch Legacy Extensions möglich ist. Mozilla nennt in der Dokumentation dieser Schnittstelle drei Fälle, in denen binäre Erweiterungen sinnvoll sind [21]:

- Höhere Anforderungen an die Performance, die durch JavaScript nicht erfüllt werden können.
- Die Verwendung von externen Bibliotheken in C bzw. C++.
- Die Verwendung von Mozilla Schnittstellen, die nicht über XPCOM angesprochen werden können.

Im Allgemeinen rät Mozilla jedoch von der Verwendung ab, da sehr viele PlugIns in JavaScript entwickelt werden können und somit nicht plattformabhängig sind bzw. für jede neue *Mozilla Firefox* Version neu kompiliert werden müssen:

> *Note: With the modern JIT Javascript Engine in Gecko and js-ctypes more extension code can be written only in JavaScript than ever before. Please consider all of your options carefully before deciding to use native code (C++) in your extension. In addition, binary components need to be recompiled for every major Firefox release, which can be frustrating. [21]*

Als vierte Schnittstelle bietet *Mozilla Firefox* das *Add-on SDK*, das ähnlich wie WebExtensions die Entwicklung von Plugins über Webtechnologien ermöglicht. Hierüber ist sowohl die Einbindung einer Seitenleiste in die Browseroberfläche als auch der Zugriff auf alle für *Storyfinder* benötigten Inhalte möglich. Für die Einbindung von *Storyfinder* wurde deshalb diese Schnittstelle gewählt, die im folgenden Abschnitt detailliert beschrieben wird.

4.4.2 Einbindung von Storyfinder über das Add-on SDK in Mozilla Firefox

Das *Add-on SDK* von *Mozilla Firefox* ermöglicht die Entwicklung von Plugins in Javascript. Das SDK bietet hierfür Zugriff auf High- und Low-Level APIs des Browsers. *Storyfinder* verwendet die folgenden neun High-Level APIs:

- page-mod: Die Page-Mod API ermöglicht das Ausführen von Javascript Dateien im Kontext einer Webseite. Beim Aufruf bestimmter Webseiten, die über entsprechende URL-Filter festgelegt werden können, wird eine Javascript Datei zur Webseite hinzugefügt. Dieses „Content-Script" kann auf das Document Object Model der Seite zugreifen und dieses auslesen oder verändern. Das Plugin selbst hat keinen direkten Zugriff auf die Inhalte einer Webseite und jeder Zugriff auf die Seiteninhalte ist deshalb nur über ein Content-Script möglich. Da Content-Scripte aus Sicherheitsgründen keinen direkten Zugriff auf die Browser APIs haben, erfolgt die Kommunikation zwischen Content-Script und Plugin über frei definierbare Nachrichten (Port-Messages).
- request: Über die Request API ist der Aufruf externer Resourcen über HTTP(s) möglich. Es können hierüber HTTP Anfragen an externe Webservices gestellt werden und somit Daten von externen Quellen gelesen oder geschrieben werden.
- context-menu: *Storyfinder* fügt in das Kontext-Menü des Browsers einen Eintrag zum Hinzufügen neuer Entitäten hinzu. Dieser Eintrag wird nur bei Textmarkierungen angezeigt. Die context-menu API stellt die nötigen Funktionen hierfür zur Verfügung und ermöglicht die kontextbezogene Anzeige des Eintrags.

[4] Version 47.0, Stand 15.07.2016

- tabs: Um den zum ausgewählten Tab passenden Wissensgraphen anzuzeigen, sowie zur Überwachung von Tab-Wechseln und dem Öffnen von neuen Tabs, nutzt *Storyfinder* die tabs-API. Hierüber wird das Plugin über Änderungen an Tabs informiert und kann auch selbst neue Tabs erzeugen oder eine bestimmte URL in einem Tab öffnen. Diese Funktion wird für das Öffnen des Globalen Graphen benötigt.

- ui: Die UI API ermöglicht das Hinzufügen neuer Elemente zur Browseroberfläche. Zur Auswahl stehen Action- und Togglebuttons, Frames mit HTML Inhalten, die in der Browsertoolbar angezeigt werden, sowie eigene Toolbars und Sidebars. *Storyfinder* verwendet diese API zum Hinzufügen der Sidebar, die den Wissensgraphen enthält. Position und Größe der Seitenleiste können jedoch nur vom Benutzer und nicht über die API festgelegt werden.

- simple-prefs: Zur Speicherung von Einstellungen stellt das *Add-on SDK* die simple-prefs API zur Verfügung. In *Storyfinder* verwenden wir diese API zur Speicherung des Plugin Status und für allgemeine Einstellungen wie Server-URL und Benutzer-Konto. Diese Einstellungen können vom Benutzer in den Add-On Optionen angepasst werden. Das Plugin wird zudem bei jeder Änderung dieser Eigenschaften benachrichtigt und kann entsprechend auf solche Ereignisse reagieren.

- window: Die Window-API bietet Zugriff auf das aktuelle Browserfenster. *Storyfinder* benötigt diese API zur Erstellung von Vorschaubildern der aufgerufenen Webseiten. Die Erstellung von Screenshots wurde analog zur in [32] beschriebenen Methode implementiert.

- url, base64: Diese beiden APIs stellen Hilfsfunktionen zum Parsen von URLs und für die Base64 Kodierung zur Verfügung. Die Übertragung von Vorschaubildern an den *Storyfinder* Server erfolgt Base64 kodiert.

Die Entwicklung der Erweiterungen erfolgt in Javascript und unterstützt die Aufteilung des Quelltextes in Module. Über *JPM*, eine auf *NPM* [52] basierte Paketverwaltung, können zusätzlich externe Module eingebunden werden. Ebenfalls über *JPM* erfolgt das für die Installation im Browser notwendige Packen der Erweiterung, bei dem alle zur Plugin gehörenden Dateien zu einer XPI-Datei zusammengefasst werden.
Bei der Initialisierung des *Storyfinder-Plugins*, wird zuerst geprüft, ob *Storyfinder* aktiviert wurde. Der Status wird über die simple-prefs API gespeichert und kann hierüber auch abgefragt werden. Falls das Plugin im aktiven Status ist, wird die Sidebar initialisiert und angezeigt. Als Inhalt einer Sidebar in *Mozilla Firefox* wird eine reguläre HTML Seite verwendet. Es kann jedoch nur eine Datei eingebunden werden, die lokal im Plugin Packet enthalten ist. Somit kann der Wissensgraph nicht direkt in der Sidebar angezeigt werden, da dieser nicht lokal vorliegt, sondern vom *Storyfinder* Server generiert wird. Um dennoch den *Storyfinder* Wissensgraphen in der Sidebar anzeigen zu können, verwenden wir eine lokale HTML Seite als Zwischenschicht. Bei dieser Zwischenschicht handelt es sich um eine HTML Seite die direkt in der Sidebar angezeigt werden kann, da diese im Plugin Paket enthalten ist. Diese Seite enthält einen sichtbaren IFrame, der die komplette Größe der Seite und somit der Sidebar einnimmt.
Nach dem Laden der Seite wird innerhalb dieses IFrames der Wissensgraph vom *Storyfinder* Server geladen. Aufgrund von Sicherheitsmechanismen ist jedoch keine direkte Kommunikation zwischen dem Wissensgraphen im IFrame und dem Plugin möglich. Die Zwischenschicht kann jedoch sowohl mit dem Plugin kommunizieren als auch mit dem Wissensgraphen und dient deshalb als Schnittstelle zur Nachrichtenübermittlung zwischen diesen beiden Komponenten. Hierfür werden Nachrichten vom Wissensgraphen über Window-Messages empfangen und über Port-Messages an das Plugin weitergeleitet. Die Kommunikation in die entgegengesetzte Richtung erfolgt analog hierzu. Neben diesem sichtbaren IFrame, enthält die Zwischenschicht noch weitere ausgeblendete IFrames, so dass für jeden Tab im aktuellen Browserfenster ein IFrame in der Zwischenschicht enthalten ist. Bei einem Tab-Wechsel wird immer nur der zum jeweiligen Tab gehörende IFrame eingeblendet, so dass für jeden Tab bzw. die darin enthaltene Seite ein separater Wissensgraph zur Verfügung steht. Auf Seite 32 in Abbildung 4.10 ist der Aufbau der *Storyfinder* Sidebar skizziert.
Nach der Initialisierung der Sidebar wird das Kontext-Menü des Browsers erweitert. Über die context-menu API wird ein Eintrag zum manuellen Hinzufügen neuer Entitäten hinzugefügt, der nur bei Textmarkierungen angezeigt wird. Abschließend wird über page-mod ein Content-Script registriert, das bei jedem Seitenaufruf geladen wird, sowie ein Content-Stylesheet für das Highlighting der Entitäten. Das Content-Script versucht nach erfolgreichem Laden der Webseite den eigentlichen Artikel auf der Seite zu erkennen, wie in Kapitel 5.2 erläutert wird. Dieser wird anschließend an das Plugin gesendet. Von dort wird der Artikel zusammen mit einigen zusätzlichen Metadaten wie Favicon und Seitentitel über einen HTTP Request an den *Storyfinder* Server gesendet. Zeitgleich wird die Sidebar über den Aufruf der neuen Seite informiert und zeigt den Verarbeitungsstatus an.
Der *Storyfinder* Server prüft, ob die Seite relevant ist (siehe Kapitel 5.3) und übernimmt gegebenenfalls die eigentliche Extraktion der Entitäten (vgl. Kapitel 5). Nach der Extraktion übermittelt der Server die extrahierten Daten der ausgewählten Seite an die, über WebSockets verbundenen Wissensgraphen. Dies führt zu einer Aktualisierung und zur Anzeige des Wissensgraphen in der Seitenleiste und auf externen Geräten.
Parallel hierzu antwortet der *Storyfinder* Server auf die vom Plugin gesendete HTTP Anfrage mit der Liste der extrahierten Entitäten.

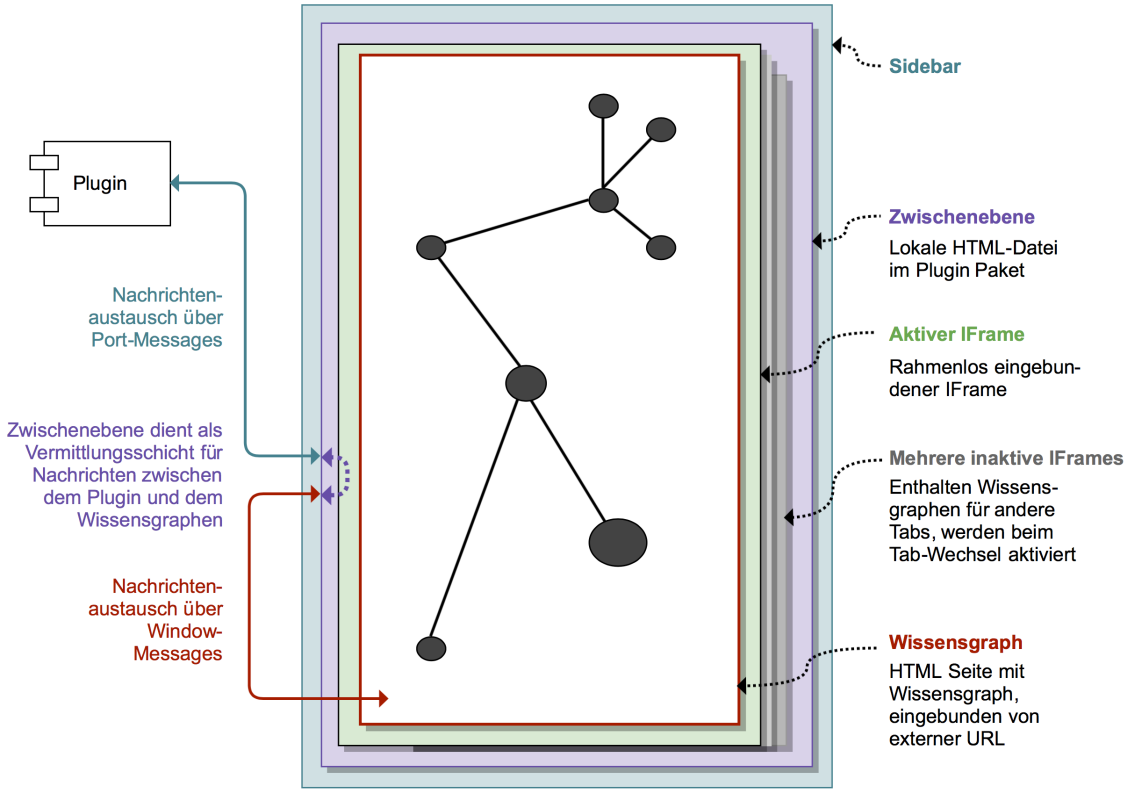


Abbildung 4.10.: Aufbau und Kommunikation der Komponenten in der *Storyfinder* Sidebar

Das Plugin sendet diese Daten an das Content-Script auf der Webseite, wo anhand dieser Informationen ein Highlighting dieser Elemente erfolgt. Zusätzlich erzeugt das Plugin ein Screenshot der Webseite und des Artikels. Diese werden wieder per HTTP Request an den *Storyfinder-Server* gesendet und dort für die Seitenvorschau und das Artikel-Archiv gespeichert. Im nächsten Kapitel werden ausführlich die Extraktionsmethoden zur Ermittlung der Entitäten und Beziehungen erläutert.

5 Datenextraktion in Storyfinder

Bei der Datenextraktion werden die Artikel- und die Meta-Daten der aufgerufenen Webseiten extrahiert. Anschließend werden hieraus in mehreren Aufbereitungsschritten die für die Visualisierung relevanten Daten gewonnen. Dieses Kapitel beschreibt diese Extraktionsschritte, die teilweise clientseitig durch das Plugin, jedoch größtenteils serverseitig durch den *Storyfinder* Server und die verbundenen Komponenten durchgeführt werden. Zu Beginn erfolgt in Unterkapitel 5.1 ein Überblick über die Elemente einer Webseite. Hierbei wird insbesondere betrachtet, inwieweit diese Elemente für das Textverständnis und somit für *Storyfinder* relevant sind. Besonders wichtig ist hierbei der eigentliche Artikeltext auf der Seite. Es werden deshalb hieran anschließend in Unterkapitel 5.2 Methoden zur Erkennung und Extraktion des Artikeltextes erläutert.
Da nicht alle Seiten geeignete Artikel in Textform enthalten, wird in Unterkapitel 5.3 anhand der 100 meistbesuchten Webseiten betrachtet, welche Seiten für *Storyfinder* relevant sind. Anschließend wird hieraus eine Methode entwickelt, die eine entsprechende Klassifizierung der Webseiten ermöglicht.
Um zu vermeiden, dass dieselben Seiten bei mehrmaligem Aufruf jedes Mal erneut zu *Storyfinder* hinzugefügt werden, prüft *Storyfinder* die Seiten auf markante Änderungen (siehe Unterkapitel 5.4). Die weitere Extraktion wird nur durchgeführt, falls eine solche markante Änderung der Seite vorliegt.
Hierauf aufbauend werden in den folgenden Unterkapiteln die einzelnen Extraktionsschritte im Detail beschrieben. Zuerst wird die Extraktion allgemeiner Daten der Webseite wie Metadaten oder Vorschaubilder in Unterkapitel 5.9 betrachtet. Anschließend erfolgt die Extraktion der Daten für den Wissensgraphen. Die hierfür nötige linguistische Vorverarbeitung des Textes wird in Kapitel 5.5 beschrieben. Danach kann die eigentliche Extraktion beginnen. Diese setzt sich aus der Extraktion von Eigennamen (siehe Unterkapitel 5.6) und Schlüsselwörtern (siehe Unterkapitel 5.7) aus dem Artikeltext, sowie der Erkennung von Beziehung zwischen diesen Elementen (Kapitel 5.8) zusammen. Diese Daten bilden die Grundlage für den Wissensgraphen. In Unterkapitel 5.9 wird abschließend erläutert, welche zusätzlichen Meta-Daten der Webseiten für die Quellenangaben extrahiert werden.

5.1 Elemente einer Webseite und deren Relevanz

Die Elemente, der vom Benutzer besuchten Webseiten, bilden die Quelle für die Datenextraktion. Im Folgenden werden die verschiedenen Elemente einer Webseite aufgeführt und deren Verwendungszweck analysiert.
Neben dem eigentlichen Seiteninhalt bestehen viele Webseiten aus zahlreichen weiteren Komponenten wie Navigation, Werbung, Footer usw., die für das Verständnis des Inhalts nicht oder kaum relevant sind. Wir konzentrieren uns deshalb bei der folgenden Aufstellung auf Elemente aus dem Seiteninhalt sowie auf die Metadaten einer Webseite. In Unterkapitel 5.2 werden Methoden beschrieben, wie der eigentliche Seiteninhalt von den restlichen Komponenten getrennt werden kann.

5.1.1 Artikeltext

Der Artikeltext stellt das wichtigste Textelement und die Kernkomponente vieler Webseiten dar. Hieraus lassen sich über Methoden aus dem Bereich der Sprachtechnologien wie Named Entity Recognition oder Keyword Extraction wichtige Elemente und Aussagen des Artikels extrahieren. Da nicht alle Wörter eines Textes die gleiche Relevanz besitzen, müssen diejenigen Wörter oder Phrasen ermittelt werden, die einen hohen Informationsgehalt besitzen. Zu diesen Elementen zählen Eigennamen und Keywords.

Objekte mit Eigennamen

Unter Objekten mit Eigennamen (Named Entities) versteht man einzelne Wörter oder Phrasen, die ein Element aus der realen Welt beschreiben. Hierzu zählen Personen, Orte, Organisationen, Zeitangaben, Marken- oder Produktnamen sowie Zahlen und Währungen. Da Eigennamen eine Benennungsfunktion haben, sind sie wesentlich für das Verständnis und die Aussage eines Satzes. In folgendem Beispielsatz wird dies deutlich:

> *Zahlreiche Menschen haben auf dem **Trafalgar Square** in **London** (**Großbritannien**) bei einer Gedenkveranstaltung für die ermordete **Labour**-Politikerin **Jo Cox** teilgenommen.*[1]

[1] Quelle: http://www.tagesschau.de/multimedia/bilder/blickpunkte-2191.html (zuletzt abgerufen am 22.06.2016)

In diesem Satz sind drei Orte (Trafalgar Square, London, Großbritannien), eine Person (Jo Cox) sowie eine Organisation (Labour) enthalten. Ohne diese Elemente, wie im folgenden Satz dargestellt, würden die wesentlichen Informationen fehlen und die Aussage des Satzes würde verloren gehen.

Zahlreiche Menschen haben bei einer Gedenkveranstaltung für die ermordete Politikerin teilgenommen.

Im Beispielsatz erkennt man außerdem, dass es neben Eigennamen noch weitere Wörter bzw. Phrasen gibt, beispielsweise „Gedenkveranstaltung" und „ermordete Politikerin", die wichtige Informationen enthalten. Diese Schlüsselwörter werden im folgenden Abschnitt näher beschrieben.

Keywords

Bei Schlüsselwörtern handelt es sich um einzelne Wörter oder auch Phrasen, die einen hohen Informationsgehalt innerhalb eines Satzes oder Textes besitzen und daher für das Verständnis und die Aussage von hoher Bedeutung sind. Im Gegensatz zu Eigennamen beschreiben die Schlüsselwörter nicht zwangsläufig ein benanntes Element aus der realen Welt und sind immer vom jeweiligen Textkorpus abhängig.
Betrachtet man beispielsweise das deutsche WikiWars Korpus [63], das aus 22 Wikipedia Artikeln über verschiedene Kriege besteht, so ist das Wort „Militär" 185 Mal enthalten, also im Schnitt etwa 8,4 Mal je Artikel. Für einen einzelnen Artikel ist der Informationsgewinn aus diesem Wort somit nur sehr niedrig und das Wort eignet sich kaum als Schlüsselwort.
Vergleicht man hierzu das Tagesspiegel Korpus von 1996-2005 [41], das aus allen 330.790 online veröffentlichten Artikeln aus dem Tagesspiegel besteht, so kommt das Wort „Militär" insgesamt nur 4.043 Mal vor[2]. Im Schnitt ist der Begriff also nur etwa einmal in jedem hundertsten Artikel enthalten. Kommt der Begriff in einem Artikel dieses Korpus mehrmals vor, wie z.B. im Artikel „Stille Macht, heilige Macht" vom 07.01.1997, in dem der Begriff vier Mal enthalten ist, so hat das Wort in diesem Artikel einen hohen Informationsgehalt und ist daher ein mögliches Schlüsselwort für diesen Artikel. In Kapitel 5.7 wird im Detail auf die Extraktion von Schlüsselwörtern eingegangen.

Im Gegensatz zu nicht annotiertem Text, werden Textabschnitte oder einzelne Wörter durch die HTML Auszeichnungen auf Webseiten gesondert ausgezeichnet. Hierüber lassen sich weitere relevante Textelemente erkennen und extrahieren. Zu diesen Elementen zählen insbesondere Hyperlinks und Überschriften.

Hyperlinks

Durch Hyperlinks wird auf andere Webseiten verwiesen. Für Textelemente, die durch einen Hyperlink gekennzeichnet sind („Anchortext"), existieren weiterführende Informationen. Besonders bei redaktionellen Inhalten werden solche Auszeichnungen häufig gezielt vom Autor ausgewählt und vorgenommen. Die Textabschnitte werden für den Leser meist deutlich gegenüber dem restlichen Text hervorgehoben und bilden somit sowohl aus Sicht des Autors als auch aus Lesersicht wichtige Elemente innerhalb eines Textes. Es erscheint daher sinnvoll, die über einen Hyperlink markierten Textabschnitte zu extrahieren und für die Visualisierung zu verwenden.
Im folgenden Beispiel ist ein Hyperlink (hier durch Unterstreichung gekennzeichnet) enthalten:

Zu all diesen Themen hatte [...] die Global Commission on Internet Governance in einer Studie 66 Empfehlungen veröffentlicht [...].[3]

Der Textausschnitt „66 Empfehlungen" ist keine Named Entity und würde vermutlich auch nicht als Schlüsselwort erkannt werden, beinhaltet für diesen Satz allerdings trotzdem wichtige Informationen und sollte daher im Wissensgraphen aufgeführt werden. Bei der Verwendung von Hyperlinks in der Visualisierung von *Storyfinder* hat sich jedoch gezeigt, dass hierdurch sehr häufig irrelevante Elemente oder sogar ganze Sätze extrahiert werden, wogegen es sich bei relevanten Hyperlinks meist um Named Entities oder Schlüsselwörter handelt, die bereits durch diese Extraktionsschritte erkannt wurden. Aus diesem Grund werden für die Datenextraktion von *Storyfinder* keine Hyperlinks verwendet.

Überschriften und Textauszeichnungen

Neben Hyperlinks sind insbesondere Überschriften und Zwischenüberschriften besonders hervorgehoben und enthalten oftmals wichtige Informationen für den Text. Für die Darstellung im Wissensgraphen sind die vollständigen Überschriften allerdings auf Grund ihrer Länge meist nicht geeignet. Ebenso verhält es sich mit weiteren Auszeichnungen durch Änderung des Schriftgewichtes („fett"), des Schriftschnittes (z.B. „kursiv") oder Unterstreichungen. Auch hier werden häufig komplette Sätze entsprechend gekennzeichnet, so dass sich diese Daten nicht direkt für die Extraktion eignen. In *Storyfinder* werden diese Daten, ebenso wie alle anderen Textabschnitte, in die Extraktion von Eigennamen und die Schlüsselworterkennung einbezogen, jedoch ansonsten nicht gesondert verarbeitet.

[2] https://www.dwds.de/r?q=Milit%C3%A4r;corpus=tagesspiegel

[3] http://www.heise.de/newsticker/meldung/OECD-Ministerkonferenz-US-Handelsministerin-Pritzker-warnt-vor-digitalem-Protektionismus-3246515.html (Zuletzt abgerufen am 23.06.2016)

5.1.2 Metadaten

Webseiten können neben den Seiteninhalten noch zusätzliche Metadaten enthalten. Diese sind teilweise für den Benutzer sichtbar (Seitentitel, Favicon), können aber auch unsichtbar sein (Meta-Keywords, Meta-Description) und dienen dann meist der Suchmaschinenoptimierung. Für einige dieser Angaben existieren spezifische HTML Elemente (z.B. title). Die HTML Spezifikation [9] sieht allerdings zusätzlich auch die Angabe von Metadaten in frei definierbaren Meta-Tags vor, ohne jedoch die möglichen Meta-Angaben und deren Bedeutung zu spezifizieren:

> *The META element can be used to identify properties of a document (e.g., author, expiration date, a list of key words, etc.) and assign values to those properties. This specification does not define a normative set of properties.[9]*

Im Folgenden werden daher neben Seitentitel und Favicon nur die beiden häufig verwendeten Metaangaben *Keywords* und *Description* betrachtet und deren mögliche Verwendung in *Storyfinder* untersucht.

Seitentitel

Der Seitentitel wird in Browsern als Tab-Beschriftung verwendet, als Titel für Lesezeichen und als Überschrift in der Trefferliste von Suchmaschinen. Der Titel ist häufig eine Kombination aus der Artikelüberschrift und dem Namen des Webportals. Die Informationen aus dem Titel sind somit meist bereits im Text enthalten und müssen daher nicht gesondert analysiert werden. Der Titel kann allerdings in der Visualisierung für die Beschriftung der besuchten Webseiten verwendet werden.

Favoriten Icon

Bei einem Favoriten Icon (kurz „Favicon") handelt es sich um ein kleines Bild, das für alle Seiten einer Domain identisch ist und daher meist das Logo des Webseitenbetreibers enthält. Ursprünglich wurden diese Bilder lediglich vor Einträgen in der Lesezeichenliste angezeigt. Mittlerweile werden diese Bilder in den meisten Browsern zusammen mit dem Seitentitel in der Tableiste angezeigt. Da das Bild für alle Seiten einer Domain identisch ist, ist es nicht auf den Artikelinhalt bezogen und eignet sich daher nicht zur Visualisierung des Inhaltes. In *Storyfinder* werden Favicons jedoch zur kompakten Kennzeichnung der Quelle eines Artikels genutzt.

Meta-Keywords und Meta-Description

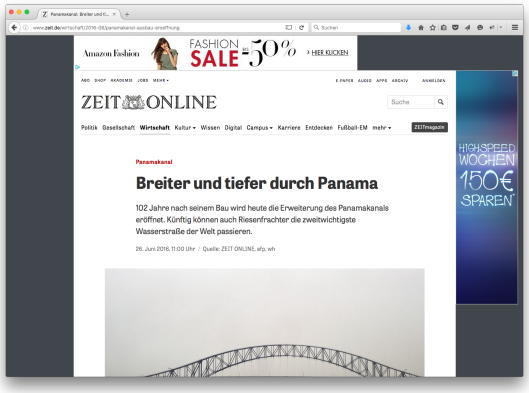
Durch die Angabe von Meta-Keywords im Kopf einer Webseite können die Inhalte der aktuellen Seite durch Schlüsselbegriffe näher beschrieben werden. Zusätzlich kann die Meta-Description eine kurze Zusammenfassung des Seiteninhaltes beinhalten. Diese Angaben wären also für die Verwendung im Wissensgraphen von *Storyfinder* sehr gut geeignet. Häufig werden diese Daten jedoch gezielt zur Verbesserung der Platzierung in Suchmaschinen verwendet und enthalten daher nur teilweise geeignete Daten. Zudem sind die enthaltenen Daten meist auch im Artikeltext enthalten und können dort extrahiert werden. Die Angaben werden deshalb in *Storyfinder* nicht verwendet.

5.2 Erkennung des Artikeltextes auf einer Webseite

Webseiten enthalten neben den eigentlichen Inhalten noch zahlreiche weitere, oftmals als Boilerplate bezeichnete Elemente, wie beispielsweise Navigation oder Werbung. Diese Elemente müssen vor den weiteren Extraktionsschritten entfernt werden. Im Folgenden werden verschiedene Algorithmen für die Säuberung von Webseiten vorgestellt. Anschließend wird die Funktionsweise des in *Storyfinder* eingesetzten Algorithmus Readability erläutert und dessen Einbindung in *Storyfinder* beschrieben.

5.2.1 Algorithmen zur Inhaltsextraktion

Verschiedene Algorithmen für die Inhaltsextraktion wurden 2007 im CleanEval Wettbewerb [5] evaluiert. Hierbei wurden insgesamt 9 verschiedene Systeme auf einem Korpus aus 684 englischsprachigen Webseiten getestet.
Das beste Ergebnis, mit 84.1% wurde von Victor: the Web-Page Cleaning Tool [61] erzielt. Die einzelnen Elemente einer Webseite werden bei diesem Ansatz durch Sequence Labeling auf Basis von Conditional Random Fields als Inhalt oder kein Inhalt klassifiziert. Als Elemente werden einzelne zusammenhängende Textinhalte innerhalb eines HTML Blocks betrachtet. Jedem Element werden hierbei mehrere Features zugeordnet, die sich auf den jeweiligen Block (z.B. Container-Typ), den Textinhalt (z.B. Anzahl der Leerzeichen, Anzahl der Sätze, Group-Word-Ratio) und das gesamte Dokument (z.B. Anzahl der Wörter und Sätze im Dokument und die maximale Word-Group-Ratio für DIV und TD Elemente) beziehen.



(a) Webseite in der Standardansicht von Mozilla Firefox

(b) Webseite in der Leseansicht von Mozilla Firefox

Abbildung 5.1.: Vergleich der Darstellung einer Webseite in der Standardansicht und im Lesemodus

Der in [43] vorgestellte Ansatz, der in Boilerpipe[4] verwendet wird, basiert auf einfachen Features wie der durchschnittlichen Wort- und Satzlänge und der Position der Elemente im Dokument. Zusätzlich wird noch die Textdichte [44] eines Blocks betrachtet.

5.2.2 Inhaltsextraktion in Storyfinder durch Readabilty

In *Storyfinder* wird der Inhalt nicht nur für die weitere Datenextraktion benötigt, sondern es wird auch eine Verknüpfung zwischen dem Textinhalt auf der Webseite und den Wissensgraphen hergestellt. Es ist somit erforderlich, dass die Inhaltsbereiche in der aufgerufenen Seite identifiziert werden können. Wir verwenden deshalb zur Erkennung der Seiteninhalte das ursprünglich von AC90 entwickelte Readability [4]. Die Methode wird unter anderem in *Mozilla Firefox*[5] und Apple Safari für den Lesemodus [67, 3] verwendet, um dem Benutzer eine übersichtliche Darstellung des Inhalts zu präsentieren. In Abbildung 5.1a ist eine Webseite[6] in der normalen Browser-Ansicht von *Mozilla Firefox* dargestellt und zum Vergleich hierzu in Abbildung 5.1b dieselbe Seite im Lesemodus. Wie sich gut erkennen lässt, wurden Elemente wie Kopfzeile, Navigation und Werbung vollständig entfernt und es wird nur der Artikel angezeigt.

Funktionsweise von Readability

Zur Extraktion des Artikeltextes löscht Readability zuerst überflüssige Elemente wie Formulare oder Footer. Zusätzlich werden weitere nicht benötigte Elemente anhand des Klassennamens (z.B. „comment", „banner" oder „sidebar") aussortiert.
Da Elemente vom Typ „div" oftmals wie Abschnitte (p-Element) verwendet werden, wird in einem Vorverarbeitungsschritt jedes div-Element untersucht und ggf. in einen Abschnitt umgewandelt.
Anschließend teilt der Algorithmus allen Elementen vom Typ section, h2 - h6, p, td und pre einen Content-Score zu. Dieser Content-Score errechnet sich aus verschiedenen Faktoren. Es wird hierbei unter anderem die Anzahl an Kommata, die Textlänge, der Klassenname und der Elementtyp selbst berücksichtigt. Content-Scores werden außerdem an übergeordnete Elemente weitergereicht, wobei zu direkten Elternelemente der gesamte Content-Score addiert wird und zu

[4] https://github.com/kohlschutter/boilerpipe
[5] Quellcode unter: https://github.com/mozilla/readability
[6] http://www.zeit.de/wirtschaft/2016-06/panamakanal-ausbau-eroeffnung (Abgerufen am 26.06.2016 in *Mozilla Firefox* 47.0 unter OS X 10.11.5)

deren Elternelement nur noch die Hälfte des Content-Scores. Danach wird der Content-Score je weiterer Ebene durch die Anzahl der Ebenen × 3 geteilt.
Sobald die Content-Scores verteilt wurden, wird hieraus der Top-Kandidat ermittelt, also das Element mit dem höchsten Content-Score. Anschließend werden benachbarte Elemente um diesen Top-Kandidaten gesucht, die ggf. auch noch zum Artikel gehören könnten. Hierzu könnten beispielsweise Textabschnitte zählen, die durch Werbung vom Top-Kandidaten getrennt sind. Abschließend werden der Top-Kandidat und gegebenenfalls relevante benachbarte Elemente als Artikel-Container verwendet.

Einbindung in Storyfinder

Readability ist vollständig in Javascript geschrieben und kann daher als Content-Script im Kontext einer Webseite ausgeführt werden. Es arbeitet dort direkt auf dem DOM der jeweiligen Webseite und ermöglicht somit die Erkennung und Verknüpfung der relevanten Elemente in *Storyfinder*. Hierdurch ist nicht nur die Extraktion des Artikeltextes möglich, sondern auch das Highlighting der gefundenen Entitäten innerhalb des Artikels.

5.3 Ermittlung relevanter Webseiten für Storyfinder

Beim Surfen im Internet ruft der Benutzer zahlreiche Webseiten auf, die für den Aufbau eines Wissensgraphen in *Storyfinder* nicht geeignet sind. Hierzu zählen beispielsweise Suchmaschinen, Online-Shops oder Übersichtsseiten, die keine vollständigen Artikel enthalten. *Storyfinder* speichert für jede Seite die gefundenen Entitäten und Beziehungen und fügt diese zum Wissensgraphen hinzu. Durch Seiten die keine Artikel enthalten, sondern beispielsweise eine Übersicht über mehrere Artikel, werden häufig falsche oder irrelevante Beziehungen erkannt.
Für *Storyfinder* wurde deshalb eine Methode entwickelt, um Webseiten als geeignet bzw. ungeeignet zu klassifizieren. Bei geeigneten Seiten wird die Extraktion automatisch gestartet, wohingegen auf ungeeigneten Seiten die Extraktion manuell vom Benutzer gestartet werden muss.

Für das Training und die Evaluation der Klassifizierung wurde ein Korpus von 449 Webseiten erstellt. Die Erstellung und der Aufbau des Korpus wird in Unterkapitel 5.3.1 erläutert. Anschließend wird die verwendete Methode zur Klassifizierung in Unterkapitel 5.3.2 beschrieben und abschließend in Unterkapitel 5.3.3 evaluiert.

5.3.1 Erstellung eines Korpus aus Webseiten

Zur Erstellung des Korpus wurden die 100 meistbesuchten deutschen Webseiten verwendet. Für die Ermittlung der meistbesuchten Seiten wurde auf die Daten von Alexa vom Mai 2016[7] zurückgegriffen und anschließend ein Webcrawler verwendet, der für alle Webseiten folgenden Algorithmus ausführt und somit einen Korpus aus Bildern und Quelltexten der besuchten Seiten erstellt:

1: $link \leftarrow Host$ {URL als Startlink wählen}
2: **for** $i = 0$ **to** 9 **do**
3: $website \leftarrow$ load($link$) {Link aufrufen}
4: $screenshot \leftarrow website$.takeScreenshot() {Screenshot erstellen}
5: $screenshot \leftarrow website$.readability() {Readability ausführen}
6: $source \leftarrow website$.getArticle() {Artikel auslesen}
7: $filesystem$.saveAs($link$.txt, $source$) {Artikel speichern}
8: $filesystem$.saveAs($link$.png, $screenshot$) {Screenshot speichern}
9: $candidates \leftarrow website$.getAllLinksForHost($Host$) {Alle Links die auf die gleiche Domain zeigen auslesen}
10: $link \leftarrow candidates$.getRandomItem() {Zufälligen Link als nächstes Ziel auswählen}
11: **end for**

Als Basis des Webcrawlers wurde SlimerJS [37] verwendet, ein Headless Browser auf der Basis vom *Mozilla Firefox*. Insgesamt wurden 449 Seiten erfolgreich aufgerufen und gespeichert. Diese Seiten wurden anschließend manuell als für *Storyfinder* geeignet oder ungeeignet klassifiziert. Geeignete Webseiten mussten hierbei die folgenden beiden Kriterien erfüllen:

- Hauptinhalt der Seite ist ein einzelner Artikel zu einem bestimmten Thema.
- Der Artikel ist vollständig (keine Teaser), kann aber ggf. auf mehrere Seiten verteilt sein (Pagination).

Anhand dieser Kriterien wurden 59 Seiten (13,14%) für *Storyfinder* geeignet und 390 Seiten (86,86%) als ungeeignet annotiert.

[7] www.alexa.com/topsites/countries;0/DE

5.3.2 Klassifizierung der Webseiten

Für die Klassifizierung der Webseiten werden mehrere Features verwendet, die auf dem durch Readability (siehe Kapitel 5.2.2) erkannten Artikel berechnet werden. Da die Klassifizierung bei jedem Seitenaufruf ausgeführt werden muss, wurden die Features bewusst einfach gewählt, um den Rechenaufwand und somit die Ladezeit möglichst gering zu halten. Es werden insgesamt neun Features verwendet:
Als erstes Feature wird die Angabe verwenden, ob Readability überhaupt einen Artikel auf der Seite erkennen konnte.
Das Textverhältnis dient als zweites Feature und gibt an, wie viele Sätze aus dem erkannten Artikeltext relevant sind. Die Grundidee hierbei ist, dass alle Sätze in geeigneten Artikeln eine gewisse Länge besitzen und nur wenige Sonderzeichen oder Zahlen beinhalten. Der Artikeltext wird deshalb in Sätze unterteilt und anschließend für jeden Satz geprüft, ob die beiden folgenden Kriterien erfüllt sind:

- Mindestens 90% der Zeichen im Satz sind aus der Gruppe {„A-Z", „a-z", „-", „„", {Leerzeichen}}
- Der Satz enthält mindestens sechs und maximal 25 Tokens

Die restlichen sieben Features geben die Anzahl der folgenden HTML Elemente an: Formulare („<form>"), Eingabefelder („<input>"), Überschriften („<h*>"), Abschnitte („<p>"), Zeilenumbrüche („
"), Bilder („<img>") und Tabellenzeilen („<tr>").

Klassifizierer

Für die Klassifizierung wurden zwei unterschiedliche Machine Learning Ansätze mit Hilfe der WEKA Data Mining Software [34] evaluiert. Der Regellerner JRip verwendet eine Methode namens Repeated Incremental Pruning to Reduce Error Reduction (RIPPER) [15]. Es werden hierbei zuerst einzelne Regeln aus den Trainingsbeispielen gelernt. Anschließend werden diese Regeln in mehreren Pruning Schritten vereinfacht, um ein Overfitting zu verhindern.
Die zweite Methode „Random Forest" basiert auf einer Kombination von mehreren Entscheidungsbäumen. Jeder Entscheidungsbaum wird während des Trainings aus mehreren zufällig gewählten Features erstellt (Random Tree). Bei der Klassifizierung wird durch jeden dieser Entscheidungsbäume eine Klasse vorhergesagt und anschließend die am häufigsten vorhergesagte Klasse verwendet (Random Forest [13]).

5.3.3 Training und Evaluation

Für das Training und die Evaluation wurde eine Cross-Validation mit zehn Folds verwendet. Hierbei wird das Korpus in zehn Teilmengen gleicher Größe aufgeteilt. Anschließend wird der Klassifizierer jeweils auf neun Teilmengen trainiert und auf der verbleibenden Teilmenge evaluiert. Dies wird zehn mal wiederholt, so dass jede Teilmenge einmal zur Evaluation und neunmal zum Training verwendet wird. Schlussendlich wird der Durchschnitt der Ergebnisse aus allen zehn Durchgängen zum Gesamtergebnis kombiniert.
Die Ergebnisse sind auf Seite 39 in Abbildung 5.2 dargestellt. Wie man erkennen kann, wurde durch Random Forest ein F-Score von 88,1% erzielt, wogegen JRip nur 86,4% erreichte. Besonders wichtig sind auch Recall und Precision bei der Vorhersage der gültigen Dokumente. Der Recall liegt bei beiden Methoden bei etwa 88%. Die Precision von Random Forest liegt bei 88,1% und bei JRip bei 85,8%. Der Recall bei relevanten Seiten liegt jedoch nur bei 37,3% (JRip) bzw. 39,0% (RandomForest). Auch die Precision liegt bei den relevanten Seiten deutlich niedriger bei 52,4% (JRip) bzw. 65,7% (Random Forest).

Diskussion der Ergebnisse

Besonders der Recall der geeigneten Seiten von nur etwa 38% ist sehr niedrig. Die Überprüfung, ob eine Seite relevant ist, bevor *Storyfinder* gestartet wird, ist dennoch sinnvoll. Würde *Storyfinder* stattdessen immer automatisch gestartet werden, so läge die Precision für relevante Seiten nur bei 13,1%, d.h. bei 86,9% der aufgerufenen Seiten würden Entitäten und Beziehungen von ungeeigneten Seiten in den Wissensgraphen aufgenommen werden. Der Wissensgraph würde hierdurch extrem aufgebläht und es wären zahlreiche manuelle Korrekturen durch den Benutzer notwendig. Eine andere Alternative wäre es, *Storyfinder* nie automatisch zu starten, sondern immer nur manuell durch den Benutzer. Es würde somit verhindert, dass Elemente von ungeeigneten Seiten übernommen werden. Die zusätzliche Hürde, bei jedem Aufruf einer relevanten Seite *Storyfinder* manuell zu starten, würde sich allerdings sehr schlecht auf die User Experience auswirken. Besonders der automatische Aufbau eines globalen Wissensgraphen wäre hierdurch nicht mehr gegeben.
Wir verwenden deshalb in *Storyfinder* eine Klassifizierung der Seiten. Da Random Forest sowohl insgesamt als auch besonders bei der Precision der relevanten Seiten die besseren Ergebnisse erzielte, verwenden wir in *Storyfinder* diese Methode mit insgesamt zehn Entscheidungsbäumen und jeweils vier Features.

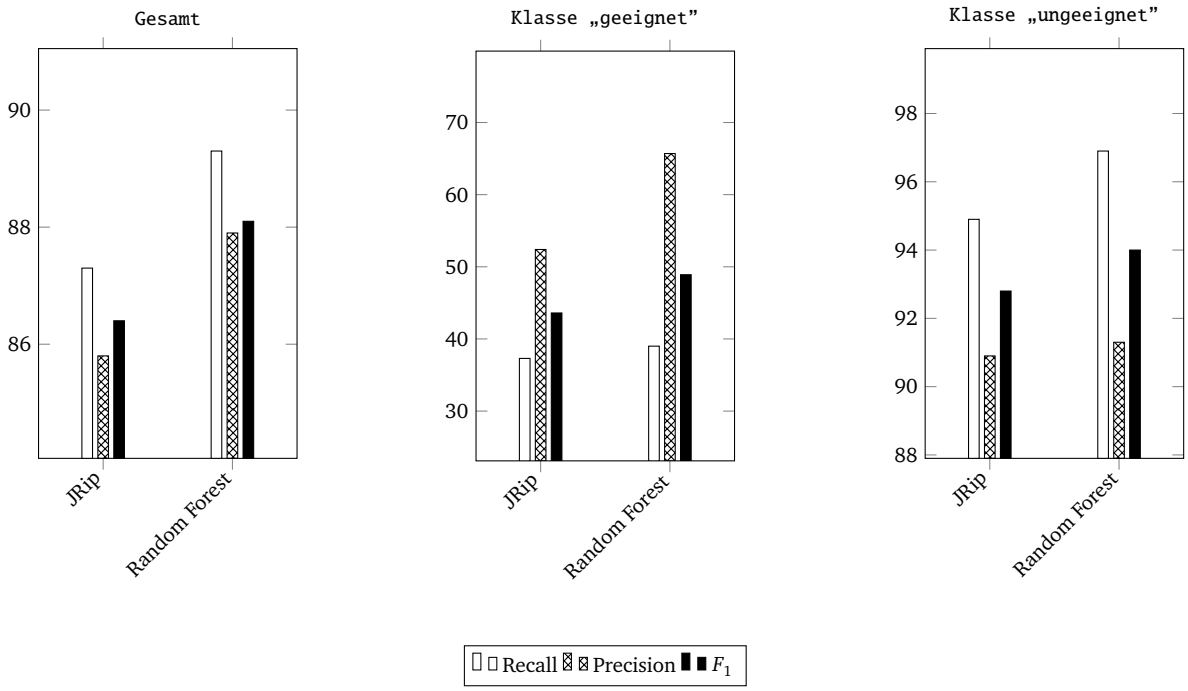


Abbildung 5.2.: Evaluationsergebnisse

5.4 Erkennung neuer und geänderter Webseiten

Wenn ein Nutzer eine bereits besuchte Seite erneut öffnet, soll diese Seite und die enthaltenen Entitäten und Beziehungen nur erneut in die Datenbank aufgenommen werden, falls eine markante Änderung des Inhaltes vorliegt. Andernfalls würde eine doppelte Extraktion stattfinden, die aus zwei Gründen problematisch wäre: Zum einen würde für solche doppelten Seiten die gesamte, rechenintensive Extraktion unnötigerweise erneut ausgeführt werden. Öffnet der Nutzer beispielsweise eine bereits bekannte Seite aus *Storyfinder* heraus, um den Artikel erneut zu lesen oder nach bestimmten Informationen zu suchen, so müsste er jedes Mal abwarten, bis die Datenextraktion erneut erfolgt ist und der Wissensgraph sowie das Highlighting auf der Webseite angezeigt wird. Zum anderen ist die Häufigkeit des Auftretens von Entitäten und Beziehungen ein Hauptfaktor für die Gewichtung der Knoten innerhalb des Graphen. Hierüber wird sowohl entschieden, welche Knoten angezeigt werden sollen und wie groß diese Knoten dargestellt werden. Würde das Vorkommen der Elemente bei jedem wiederholten Aufruf einer Seite erneut gezählt werden, so würden diese Werte verfälscht.
Dennoch soll selbstverständlich eine neue Extraktion ausgeführt werden, falls sich der Artikel deutlich geändert hat, also beispielsweise ganze Absätze ergänzt oder verändert wurden oder gar ein ganz anderer Artikel unter der bisherigen URL zu finden ist. Besonders bei Online-Auftritten von Tageszeitungen gibt es solche Änderungen häufig, da Artikel bei Veröffentlichungsstrategien wie „Online First” möglichst schnell online erscheinen und nach und nach ergänzt werden. Der Artikel für die gedruckte Ausgabe reift somit vor dem Erscheinungstag bereits in der Online-Ausgabe und ändert sich dort häufig.
Um einerseits Doppelungen von Artikeln und die damit verbundenen Probleme zu vermeiden, aber anderseits trotzdem geänderte Artikel zu erkennen, prüft *Storyfinder* die Artikel vor der Extraktion auf markante Änderungen. Diese Überprüfung erfolgt in zwei Schritten, die nachfolgend erläutert werden.

Exakte Übereinstimmung

Im ersten Schritt wird geprüft, ob der Artikel exakt mit der bereits gespeichert Version übereinstimmt. Hierfür wird für jeden Artikel eine MD5-Prüfsumme [57] gespeichert, die aus dem Artikeltext ohne Markup errechnet wird. Wird eine neue Seite aufgerufen, so prüft der *Storyfinder-Server*, ob für diese URL bereits ein Eintrag in der Datenbank existiert und ob die gespeicherte Prüfsumme mit der Prüfsumme des neuen Artikels übereinstimmt. Stimmen die beiden Prüfsummen überein, wurde der Artikel nicht verändert und es findet keine weitere Überprüfung und Extraktion statt. Da jedoch selbst bei kleinsten Änderungen die Prüfsummen verschiedenen sind, wird bei unterschiedlichen Prüfsummen ein zweiter Überprüfungsschritt durchgeführt.

Bestimmung der Artikelähnlichkeit über die Kosinus-Distanz

Im zweiten Überprüfungsschritt wird die Ähnlichkeit der beiden Artikel anhand der Kosinus Distanz im Vector-Space Model betrachtet wie in [38] erläutert wird. Hierfür wird der zuletzt gespeicherte Artikel mit dem neuen Artikel verglichen. Beide Artikel werden über einen einfachen Whitespace-Tokenizer in einzelne Tokens zerlegt. Aus den Tokens der beiden Artikel wird jeweils ein Vektor für ein Vector-Space-Model [59] erstellt. Die Tokens stellen hierbei die einzelnen Dimensionen dar und die Häufigkeit des Auftretens dieses Tokens im Artikel steht für die Länge des Vektors in der jeweiligen Dimension. Zwischen diesen beiden Vektoren wird die Kosinusdistanz nach der folgenden Formel aus [38] berechnet:

$$sim_{cosine}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

Der hieraus ermittelte Wert liegt zwischen 0 und 1, wobei ein Wert von 0 für keine Ähnlichkeit und ein Wert von 1 für eine komplette Übereinstimmung der Dokumente steht. Die Kosinusdistanz liefert hierbei einen vergleichbaren Wert, unabhängig von der Artikellänge.
Es lässt sich somit über einen Grenzwert entscheiden, ob eine markante Änderungen vorliegt und die Extraktion erneut gestartet werden soll.

5.5 Vorverarbeitung des Artikeltextes

Nachdem der Artikel extrahiert, die Seite als relevant eingestuft sowie festgestellt wurde, dass es sich um eine neue Seite oder um einen Artikel mit markanten Änderungen handelt, kann die eigentliche Extraktion der Daten beginnen. Hierfür muss der Artikeltext entsprechend vorbereitet werden.

Tokenisierung und Sentence Splitting

Für die Named Entity Recognition muss der Text in einzelne Sätze (Sentence Splitting) und die Sätze wiederum in einzelne Tokens (Tokenization) unterteilt werden. Ebenso setzt die Erkennung der Schlüsselwörter einer Aufteilung in einzelne Tokens voraus.
In *Storyfinder* greifen wir hierfür auf die Funktionen des *CoreNLP Toolkits* [47] zurück. Eingebunden in den *Stanford CoreNLP Server* stellt dieses Toolkit verschiedene sprachtechnologische Funktionen zur Verfügung und lässt sich als Webservice über ein einfaches HTTP API ansprechen. Beim API Aufruf werden per HTTP Request der Dokumenttext sowie die gewünschten Annotatoren spezifiziert und als Antwort der annotierte Text im JSON Format zurückgeliefert.
Für das Sentence Splitting und die Tokenisierung verwendet *CoreNLP* den *PTBTokenizer*, der sowohl die Aufteilung in Tokens als auch die Aufteilung in Sätze ermöglicht. Die Integration in *Storyfinder* erfolgt über den TokenizerAnnotator des *CoreNLP Servers*.
Beim *PTBTokenizer* handelt es sich um einen deterministischen endlichen Automaten, der über zusätzliche Heuristiken bestimmte Sonderfälle wie Anführungszeichen oder Interpunktion innerhalb von Sätzen korrekt zuordnen kann. Das Sentence-Splitting erfolgt nach der Tokenisierung anhand der Satzzeichen, die zu keinem anderen Token zugeordnet sind [46].
Unter bestimmten Umständen extrahiert Readability gegebenenfalls in die Seite eingebundenen Javascript-Code fälschlicherweise als Teil des Artikeltextes. Diese Abschnitte können zu Problemen bei der Verarbeitung durch den *CoreNLP Server* führen, so dass gegebenenfalls die Verarbeitung des gesamten Artikels fehlschlägt. Da dieses Problem bei der Entwicklung von *Storyfinder* mehrmals aufgetreten ist, findet vor der Verarbeitung durch *CoreNLP* eine Aufteilung des Textes in einzelne Absätze statt. Die Aufteilung erfolgt anhand von zwei oder mehr aufeinander folgenden Zeilenumbrüchen.
Jeder Abschnitt durchläuft anschließend einen Filter, um Abschnitte, die Javascript-Code enthalten, zu entfernen. Hierfür wird geprüft, ob mindestens 90% der im Abschnitt enthaltenen Zeichen in der folgenden Liste enthalten sind: {„A-Z", „a-z", „-", „„, {Leerzeichen}}. Zudem wird überprüft, ob die durchschnittliche Wortlänge, ermittelt durch eine einfache Whitespace-Tokenisierung, nicht mehr als zehn Zeichen beträgt. Nur Abschnitte, die diesen Kriterien entsprechen, werden bei der weiteren Verarbeitung durch *CoreNLP* verwendet.
Jeder dieser Abschnitte wird separat zur Verarbeitung an den *CoreNLP-Server* gesendet. Sollte die Verarbeitung eines Abschnittes fehlschlagen, können dennoch die anderen Abschnitte für die weiteren Extraktionsschritte verwendet werden.

Stopword Filtering

Um die Extraktion von Stoppwörtern, also von sehr häufig vorkommenden Wörtern mit niedrigem Informationsgehalt wie z.B. „der" oder „ist", zu vermeiden, wird ein Stopword Filtering auf die Tokens des Artikels angewendet. Wir verwenden hierfür die deutsche Stoppwort-Liste[8] des Snowball-Stemmers [53]. Vor der Extraktion von Schlüsselwörtern werden alle Tokens des Artikels mit dieser Liste abgeglichen und bei Übereinstimmung entfernt.

[8] https://snowballstem.org/algorithms/german/stop.txt

5.6 Extraktion von Named Entities aus dem Artikeltext

Eigennamen bilden die wichtigsten Elemente innerhalb des Wissensgraphen von *Storyfinder*. Für die Extraktion von Eigennamen (Named Entity Recognition, NER) existieren zahlreiche verschiedene Ansätze, wie beispielsweise die Verwendung eines Hidden Markov Modells [10], regelbasierte Ansätze [60] oder Conditional Random Fields [27].

GermaNER

In *Storyfinder* verwenden wir den ebenfalls auf CRF basierenden *GermaNER* [8] für die Extraktion der Eigennamen aus deutschen Texten. *GermaNER* kann Eigennamen aus den vier Klassen Organisation, Ort, Person und Sonstiges erkennen und extrahieren. Hierfür werden verschiedene Features zur Klassifizierung verwendet, wie beispielsweise N-Gramme der Tokens und Wortarten, Lexika (u.a. Orte und Währungen), die vier ähnlichsten Worte zu jedem Token, Topic Clusters sowie weitere Features [8].

Einbindung in Storyfinder

GermaNER steht als UIMA Komponente oder direkt als Programm für die Kommandozeile zur Verfügung. Da die Verarbeitung über die Kommandozeile sehr lange benötigt, wurde für *Storyfinder* ein Wrapper für die UIMA Komponente geschrieben. Der Wrapper stellt hierbei eine Web-API zur Verfügung, so dass die Extraktion über HTTP aufgerufen werden kann. Dies ermöglicht die Auslagerung von *GermaNER* in einen eigenen *Docker Container*. Hierdurch wäre ein Load-Balancing sehr einfach durch die Verwendung mehrerer Instanzen möglich.
Der Wrapper führt direkt beim Starten die Initialisierung durch und lädt die Daten aus den einzelnen Modellen. Dieser Schritt dauert auf unserem Testsystem mehr als eine Minute und müsste bei der direkten Verwendung des Kommandozeilen Programms für jede Extraktion erneut ausgeführt werden. Anschließend stellt das System die Resource „germaner" zur Verfügung. Diese Resource erwartet per HTTP-POST Request das Dokument direkt als Body der Anfrage im *GermaNER* Format. Hierbei wird jeweils ein Token je Zeile angegeben und Sätze durch eine Leerzeile getrennt.
Sobald eine entsprechende Anfrage eingeht, wird die UIMA Pipeline zur Klassifizierung aufgerufen. Da die Initialisierung der Modelle bereits beim Start ausgeführt wurde, dauert die Verarbeitung je nach Länge des Artikels nur wenige Sekunden. Sobald die Extraktion abgeschlossen ist, sendet der Wrapper die Ergebnisse als Antwort der HTTP-Anfrage. Die Antwort erfolgt in dem von *GermaNER* verwendeten BIO-Schema Ausgabeformat [56]. Für jeden Token wird hierbei eine Zeile verwendet, wobei jede Zeile durch einen Tab in zwei Spalten getrennt ist. Die erste Spalte enthält das Token selbst und die zweite Spalte gibt an, ob das Token den Beginn („B-[Type]"), ein Folgeelement („I-[Type]") oder keine Named-Entity („O") darstellt, sowie um welchen Typ („PER" für Person, „LOC" für Ort, „ORG" für Organisation und „OTH" für Sonstiges) es sich handelt.
Der *Storyfinder-Server* wertet die Antwort aus und speichert die gefundenen Entitäten in der Datenbank.

Extraktion bereits gespeicherter Entitäten

Neben der Extraktion von Named Entities über *GermaNER* wird auch geprüft, ob in der Datenbank bereits gespeicherte Entitäten vorhanden sind, die ebenfalls im Artikel vorkommen. Hierbei kann es sich beispielsweise um manuell durch den Benutzer hinzugefügte Entitäten handeln.
Für jede Entität ist in der Datenbank, neben der vollständigen Bezeichnung, jeweils das erste Token sowie die Anzahl der Tokens, aus denen diese Entität besteht, gespeichert. Für die Entität „Jean Baptiste Joseph Fourier" sind beispielsweise zusätzlich zur vollständigen Bezeichnung noch das Starttoken „Jean" sowie die Anzahl der Tokens („4") gespeichert.
Für jeden Token des Artikels wird in der Datenbank geprüft, ob eine oder mehrere Entitäten mit diesem Starttoken existieren. Falls entsprechende Kandidaten gefunden wurden, wird anschließend für jeden Treffer überprüft, ob der aktuelle Token sowie die *(n - 1)* folgenden Tokens mit der Entität übereinstimmt. Der Wert *n* ergibt sich aus der gespeicherte Länge der gesuchten Entität. Würde also im Artikel das Token „Jean" gefunden, so werden die gespeicherten Entitäten, die mit diesem Anfangstoken beginnen, gesucht. Werden hier beispielsweise die Treffer „Jean Baptiste Joseph Fourier" und „Jean Paul" gefunden, so wird zuerst geprüft, ob das aktuelle Token und die folgende drei Tokens „Jean Baptiste Joseph Fourier" lauten. Ist das nicht der Fall, so wird geprüft, ob das aktuelle sowie das folgende Token „Jean Paul" lauten. Falls ein Treffer gefunden wurde, wird diese Entität extrahiert und im Artikeltext später auch entsprechend hervorgehoben.

5.7 Extraktion von Schlüsselwörtern aus dem Artikeltext

Bei Schlüsselwörtern handelt es sich um einzelne Wörter oder Phrasen im Text, die einen hohen Informationsgehalt besitzen. Wie in 5.1 erläutert wurde, ist der Informationsgehalt eines Wortes vom jeweiligen Korpus abhängig. Im Gegensatz zu Eigennamen können Schlüsselwörter deshalb nicht anhand allgemeiner Features oder Lookup Tabellen extrahiert werden.
Für die Extraktion der Schlüsselwörter in *Storyfinder* werden N-Gramme mit $n \leq 3$ verwendet, also einzelne Tokens sowie Gruppen aus zwei oder drei aufeinander folgenden Tokens. Es können somit keine Schlüsselwörter erkannt werden,

die aus mehr als drei Tokens bestehen. Die Bewertung der N-Gramme erfolgt anhand des TFIDF Wertes. Für dessen Berechnung wird die Häufigkeit der N-Gramme auf der aktuellen Webseite benötigt, sowie zusätzlich deren Häufigkeit im gesamten Korpus, also auf allen bisher erfassten Webseiten.

Extraktion der N-Gramme

Die Extraktion der N-Gramme erfolgt nach dem Vorverarbeitungsschritt auf Satzebene. Es wird über alle Tokens im Satz iteriert und hierbei jeweils das Monogramm (einzelnes Token), das Bigramm (Token und folgendes Token) sowie Trigramm (Token und zwei folgende Tokens) gebildet. Dieses Vorgehen wird für alle Sätze wiederholt. Abschließend werden identische N-Gramme summiert, so dass sich hieraus die Häufigkeit der N-Gramme ergibt.

Speicherung der Daten

Zur Berechnung des TFIDF Wertes müssen alle N-Gramme und deren Häufigkeit auf den einzelnen Webseiten dauerhaft gespeichert werden. In der Datenbank werden hierfür die einzelnen N-Gramme, sowie die Anzahl der Dokumente in denen dieses N-Gramm enthalten ist, gespeichert.

Berechnung des TFIDF Wertes

Für alle auf der Webseite enthaltenen N-Gramme wird der TFIDF Wert berechnet. Der TFIDF Wert errechnet sich aus der Termfrequency (TF) und der Documentfreqency (DF). Die Termfrequency gibt die Häufigkeit eines N-Gramms auf der aktuellen Webseite an. Hierdurch erhalten N-Gramme, die häufiger im Dokument enthalten sind einen höheren Wert, wie N-Gramme, die nur selten im Dokument enthalten sind.
Es werden somit jedoch auch N-Gramme, die in jedem Dokument häufig enthalten sind, als sehr wichtig eingestuft, obwohl der Informationsgehalt vergleichsweise gering ist. Die Termfrequency wird deshalb mit der inversen Documentfrequency (IDF) multipliziert:

$$TFIDF = \frac{TF}{DF}$$

Die Documentfrequency gibt an, auf wie vielen der besuchten Webseiten das N-Gramm enthalten ist. N-Gramme, die auf vielen Webseiten enthalten sind, besitzen somit eine höhere Documentfrequency bzw. eine geringere inverse Documentfrequency, werden also schlechter bewertet wie N-Gramme die nur auf wenigen Webseiten enthalten sind.
Durch die Multiplikation beider Werte ergibt sich eine Gewichtung, bei der N-Gramme eine hohe Bewertung erhalten, die häufig auf der aktuellen Webseite, aber selten im gesamten Korpus enthalten sind. Diese Elemente haben einen hohen Informationsgehalt und sind daher geeignete Kandidaten für Schlüsselwörter.

Ermittlung der Schlüsselwörter

Über die TFIDF Werte ergibt sich eine Gewichtung der N-Gramme. In *Storyfinder* verwenden wir für die Auswahl der Schlüsselwörter einen Grenzwert $TFIDF_{min} = 3$. Nur N-Gramme mit einem TFIDF-Wert über diesem Grenzwert werden als Schlüsselwort Kandidaten verwendet. Anschließend werden die Kandidaten auf Übereinstimmung mit bereits extrahierten Named Entities geprüft. Schlüsselwörter, die teilweise oder vollständig in einer Named Entity enthalten sind oder selbst eine Named Entity enthalten, werden nicht extrahiert. Von den verbleibenden Kandidaten extrahiert *Storyfinder* maximal die fünf am höchsten gewichteten Kandidaten als Schlüsselwörter.

5.8 Erkennung von Beziehungen zwischen den extrahierten Entitäten eines Artikels

Sobald die Erkennung der Named Entities und Schlüsselwörter abgeschlossen ist, können die Beziehungen zwischen den Entitäten ermittelt werden. *Storyfinder* extrahiert Beziehungen anhand von Satzkookkurrenz, d.h. sobald zwei Entitäten gemeinsam in einem Satz enthalten sind, wird eine Beziehung zwischen diesen Entitäten hergestellt. Diese Beziehung wird anschließend in der Visualisierung als Kante zwischen den Entitäten gekennzeichnet.
Zur Extraktion der Kookkurrenzen wird über alle Entitäten des Artikels iteriert. Für jede dieser Entitäten wird jeweils über alle folgenden Entitäten iteriert und hierbei geprüft, ob beide Entitäten im selben Satz enthalten sind. Ist dies der Fall, so wird eine Relation für diese Beziehung angelegt.
Eine Ausnahme hiervon bilden zwei Entitäten vom Typ „Person". Bei diesen Entitäten wird zuerst geprüft, ob eine der beiden Entitäten dem letzten Token der anderen Entität entspricht. In diesem Fall wird keine Beziehung zwischen den Entitäten hergestellt, sondern die beiden Entitäten werden zu einer Entität verschmolzen. Werden beispielsweise in einem Satz die beiden Entitäten „Merkel" und „Angela Merkel" gefunden, so wird geprüft, ob der letzte Token der längeren Entität, in diesem Fall das Token „Merkel" aus „Angela Merkel" mit der kürzen Entität, in diesem Fall „Merkel" übereinstimmt. Andernfalls wird auch für diese Entitäten eine Relation erstellt.
Zur Speicherung der Beziehung werden die beiden Entitäten alphabetisch sortiert. In der Datenbank wird als erste Entität der Beziehung, die alphabetisch erste Entität verwendet und als zweite Entität, die alphabetisch zweite. Hierdurch wird

sichergestellt, dass die Relation nicht doppelt angelegt wird, falls diese in verschiedenen Sätzen in unterschiedlicher Reihenfolge gefunden wird. Neben den eigentlichen Beziehungen wird in der Datenbank zusätzlich eine Referenz auf jeden Satz gespeichert, in dem die Beziehung gefunden wurde. In der Visualisierung wird hierüber die Anzeige der Quellen einer Beziehung ermöglicht.

5.9 Erfassung allgemeiner Daten einer Webseite

In *Storyfinder* werden neben den Elementen, die direkt die Knoten und Kanten des Wissensgraphen bilden, noch weitere Daten extrahiert, die für die Referenzierung, Quellenangaben oder das Artikelarchiv benötigt werden. Hierzu zählen verschiedene Meta-Daten der Webseite sowie Screenshots von der Webseite und dem Artikel selbst.

Extraktion von Metadaten

Aus den Metadaten der Webseite wird der Seitentitel, die URL und der Host der Seite sowie das Favoriten Icon extrahiert. Die Extraktion erfolgt bereits durch das Plugin über die Funktionen des *Mozilla Add-on SDK*, die einen direkten Zugriff auf diese Daten ermöglichen. Die Daten werden zusammen mit dem jeweiligen Artikel in der Datenbank gespeichert. Für das Favoriten Icon wird nur die URL, nicht jedoch die Bilddatei selbst gespeichert.

Screenshots

Neben den Meta-Daten werden auch Screenshots von der Webseite sowie vom Artikel erstellt. Die Erstellung der Screenshots erfolgt über das Plugin, wie in Kapitel 4.4.2 beschrieben. Die Screenshots der Webseite werden als Vorschaubilder in den Quellenangaben der Entitäten verwendet. In der Visualisierung wird die Quellenangabe mit der dominanten Farbe der Webseite hinterlegt. Hierfür wird bereits beim Speichern der Seite die dominante Farbe des Screenshots über Color-Thief [17] extrahiert. Die Ermittlung in Color-Thief erfolgt anhand des Zentralwert-Schnittes (Median-Cut Algorithmus) [35], worüber Cluster aus ähnlichen Farben in einem Bild erstellt werden. Anschließend wird aus diesen Clustern der größte Cluster ausgewählt und als dominante Farbe verwendet.
Die dominante Farbe wird zusammen mit den restlichen Seiteninformationen in der Datenbank gespeichert.

6 Visualisierung in Storyfinder

Die Visualisierung der extrahierten Daten liefert einen Überblick über die Inhalte einer Webseite, bezieht darüber hinaus aber auch Wissen aus zuvor besuchten Webseiten ein, um so das bisherige Wissen des Benutzers mit den neuen Informationen der aufgerufenen Seite zu verknüpfen. Zusätzlich bietet die Visualisierung Recherche-Möglichkeiten sowie Werkzeuge zur manuellen Erfassung und Annotation von Informationen. In Abschnitt 6.1 wird ein Überblick über die Komponenten gegeben und anschließend werden in Abschnitt 6.2 allgemeine Konzepte der Visualisierung in Storyfinder erläutert. Die Visualisierung von Storyfinder erfolgt sowohl auf der aufgerufenen Webseite (siehe Unterkapitel 6.3), als auch über den Wissensgraphen. Der Aufbau des Wissensgraphen wird ausführlich in Unterkapitel 6.4 erläutert. Anschließend werden in Unterkapitel 6.5 die hierüber aufrufbaren Recherchemöglichkeiten gezeigt. Da Storyfinder auf unterschiedlichen Geräten mit einer großen Spanne an Bildschirmauflösungen angezeigt werden kann, wird abschließend in Unterkapitel 6.6 die Anpassung an verschiedene Gerätegrößen beschrieben.

6.1 Übersicht über die einzelnen Komponenten

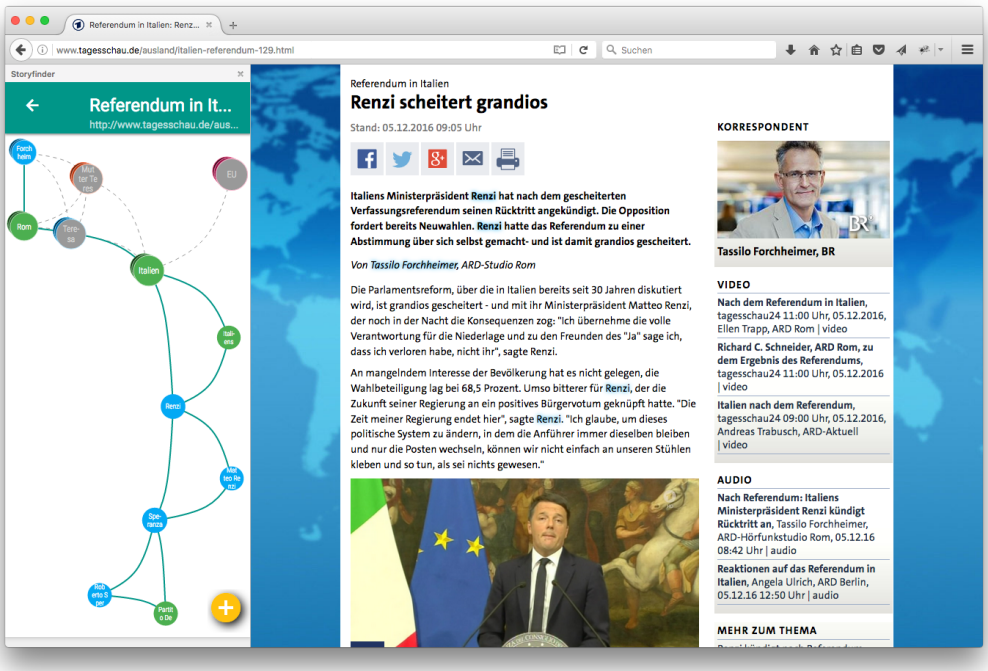


Abbildung 6.1.: Im Browser *Mozilla Firefox* wird eine geöffnete Webseite und der zugehörige *Storyfinder* Wissensgraph in der Seitenleiste angezeigt. Auf der Webseite werden die erkannten Entitäten durch farbige Hinterlegungen hervorgehoben.

Nach der im vorherigen Kapitel ausführlich beschriebenen Extraktion von Entitäten, werden die Entitäten und deren Beziehungen durch das *Storyfinder-Plugin* visualisiert. Die Kernkomponente der Visualisierung bildet der Wissensgraph, der als Netzwerk durch ein Knoten-Link-Diagramm dargestellt wird. Die extrahierten Elemente werden als Knoten im Graphen dargestellt und die Verknüpfungen als Kanten zwischen diesen Knoten. Der Wissensgraph kann sich hierbei sowohl auf eine bestimmte Webseite (Artikelgraph), eine Auswahl aus mehren Seiten (Gruppengraph) oder auf alle besuchten Seiten (Globaler Graph) beziehen. Dieser Wissensgraph kann in der Seitenleiste des Browsers, also direkt neben der jeweiligen Webseite (vgl. Abb. 6.1), in einem zusätzlichen (pinned) Tab des Browsers und auf weiteren Geräten, wie beispielsweise einem Smartphone oder Tablet angezeigt werden („Second Screen"). Durch die Darstellung auf einem zusätzlichen Gerät, kann der Nutzer die Visualisierung parallel zur angezeigten Webseite verwenden, ohne dass die Visualisierung den Anzeigebereich des Browsers bzw. der angezeigten Webseite einschränkt.

Über den Wissensgraphen erhält der Benutzer Zugriff auf zusätzliche Recherchemöglichkeiten. Hierzu zählen Quellennachweise für alle Webseiten und die zugehörigen Sätze, in denen ein bestimmtes Element aus dem Graphen enthalten ist, ebenso wie Quellennachweise für die Verknüpfungen zwischen den Elementen. Außerdem ist hierüber der Aufruf des Archivs möglich, das alle Artikel und die jeweilige Webseite zum Abrufzeitpunkt darstellt.
Neben der Visualisierung im Wissensgraphen werden die relevanten Elemente auch direkt auf der Webseite innerhalb des Artikeltextes hervorgehoben (vgl. S. 44, Abb. 6.1). Diese Hervorhebungen sind mit dem Wissensgraphen verknüpft und ermöglichen eine Interaktion zwischen den Elementen.
Im nächsten Abschnitt werden allgemeine Konzepte der Visualisierung beschrieben, die bei allen Komponenten angewendet wurden.

6.2 Allgemeine Konzepte der Visualisierung in Storyfinder

In *Storyfinder* werden bei der Visualisierung aller Komponenten bestimmte Konzepte der Farbgestaltung, Animation und der Darstellung von Status berücksichtigt. Hierdurch ergibt sich ein einheitliches Bedien- und Visualisierungskonzept über alle Komponenten hinweg. Dies ist wichtig, um beim Benutzer ein verständliches mentales Modell zu schaffen und somit den Lernaufwand für den Benutzer möglichst gering zu halten.
Storyfinder orientiert sich hierbei an den Design Prinzipien des Material Designs [36].

Farbgestaltung

Durch die Verwendung gleicher Farben für zusammengehörige Elemente, lässt sich deren Zusammengehörigkeit sehr schnell erkennen. Außerdem können einzelne Elemente nur anhand der Farbe bereits zu einer bestimmten Gruppe zugeordnet werden, ohne dass gezielt der Inhalt der Elemente betrachtet werden muss.
In *Storyfinder* werden die fünf Klassen Person, Ort, Organisation, Schlüsselwörter und Sonstiges, denen jedes extrahierte Element zugeordnet ist, durch unterschiedliche Farben visualisiert. Die Farben werden innerhalb aller Komponenten und Ansichten beibehalten. Ein Ort wird auf der Webseite in derselben Farbe hervorgehoben, wie der Knoten im Wissensgraph. Auch für die Titelzeile in der Quellenansicht sowie für die Elemente und Verknüpfungen wird diese Farbe verwendet.

Animationen

Durch Animationen können Veränderungen beschrieben werden und der Wechsel zwischen verschiedenen Zuständen ist für den Benutzer leichter zu erkennen [6]. In *Storyfinder* werden deshalb für alle Zustandsänderungen Animationen verwendet. Statusänderungen, die aufgrund einer Benutzerinteraktion erfolgen (z.B. Öffnen eines Elementes), starten immer am Punkt der Interaktion (z.B. an der Position eines Mausklicks), da der Fokus des Benutzers zum Zeitpunkt der Interaktion an dieser Stelle liegt.

Aktive und hervorgehobene Elemente

Aktive und hervorgehobene Elemente werden größer dargestellt und mit einem Schattenwurf gekennzeichnet, so dass eine Tiefenwirkung entsteht. Die Elemente rücken optisch näher in Richtung Benutzer und werden hierdurch hervorgehoben.

6.3 Visualisierungen auf der Webseite

In *Storyfinder* werden die erkannten Entitäten, neben der Darstellung im Wissensgraphen, auch direkt im Artikeltext auf der Webseite hervorgehoben. Für die Hervorhebung der Entitäten direkt auf der Webseite wurden verschiedene Methoden betrachtet. Im Folgenden werden die verschiedenen Ansätze diskutiert und begründet, warum in *Storyfinder* eine farbige Hinterlegung verwendet wird.

Änderung des Schriftschnittes

Eine Möglichkeit der Hervorhebung sind Änderungen am Schriftschnitt, also die Anpassung der Schriftstärke („**fett**") oder Lage („*kursiv*"). Die Hervorhebungen einzelner Wörter durch Veränderung der Schriftlage ist meist nur schwer zu erkennen (vgl. S. 46, Abb. 6.2c). Die Erhöhung der Schriftstärke führt hingegen zu einer deutlichen Abgrenzung zu anderen Textteilen (vgl. S. 46, Abb. 6.2d). Bei beiden Methoden besteht jedoch das Problem, dass bereits bei der Formatierung des Originalartikels häufig Änderungen am Schriftschnitt eingesetzt werden, wie beispielsweise der Lead auf Seite 46 in Abb. 6.2d zeigt. Eine Abgrenzung zwischen den Entitäten und anderen Hervorhebungen im Artikel ist somit nicht gegeben. Zudem führen Änderungen am Schriftschnitt zu Veränderungen am Platzbedarf eines Wortes und somit insgesamt zu Verschiebungen der Wörter und der Zeilenumbrüche. Da *Storyfinder* aufgrund der benötigten Zeit für die Datenextraktion asynchron arbeitet, kann es vorkommen, dass Entitäten erst hervorgehoben werden, wenn der Benutzer

Auf *Amazons Cloud-Computing-Dienst* lässt sich ab sofort die aktuelle Server-Version von *Microsoft Windows* betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf *Microsofts Cloud-Dienst Azure* ausgerichtet, aber auch *Amazon* hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der *Elastic Compute Cloud* (EC2) zu betreiben.

(a) Entitäten in kursiv. Das Highlighting ist kaum zu erkennen.

Auf Amazons Cloud-Computing-Dienst lässt sich ab sofort die aktuelle Server-Version von Microsoft Windows betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf **Microsofts Cloud-Dienst Azure** ausgerichtet, aber auch **Amazon** hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der **Elastic Compute Cloud** (EC2) zu betreiben.

(b) Entitäten in fett. Eine Fettung findet jedoch auch bereits im Lead des Artikels statt.

Auf Amazons Cloud-Computing-Dienst lässt sich ab sofort die aktuelle Server-Version von Microsoft Windows betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf Microsofts Cloud-Dienst Azure ausgerichtet, aber auch Amazon hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der Elastic Compute Cloud (EC2) zu betreiben.

(c) Unterstreichung von Entitäten. Es besteht die Gefahr der Verwechslung von Entitäten mit Links (z.B. „bietet vier Möglichkeiten an"), die ebenfalls häufig unterstrichen werden.

Auf Amazons Cloud-Computing-Dienst lässt sich ab sofort die aktuelle Server-Version von Microsoft Windows betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf Microsofts Cloud-Dienst Azure ausgerichtet, aber auch Amazon hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der Elastic Compute Cloud (EC2) zu betreiben.

(d) Unterstreichung von Entitäten mit Änderung der Schriftfarbe. Auch bei einer zusätzlichen Änderung der Schriftfarbe besteht weiterhin die Gefahr der Verwechslung mit Links.

Auf Amazons Cloud-Computing-Dienst lässt sich ab sofort die aktuelle Server-Version von Microsoft Windows betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf Microsofts Cloud-Dienst Azure ausgerichtet, aber auch Amazon hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der Elastic Compute Cloud (EC2) zu betreiben.

(e) Änderung der Schriftfarbe

Auf Amazons Cloud-Computing-Dienst lässt sich ab sofort die aktuelle Server-Version von Microsoft Windows betreiben.

Windows Server 2016 ist erwartungsgemäß stark auf Microsofts Cloud-Dienst Azure ausgerichtet, aber auch Amazon hat jetzt mitgezogen und bietet vier Möglichkeiten an, das Betriebssystem auf der Elastic Compute Cloud (EC2) zu betreiben.

(f) Änderung der Hintergrundfarbe

Abbildung 6.2.: Vergleich verschiedener Stilmittel zur Hervorhebung von Entitäten

bereits angefangen hat, den Artikel zu lesen. In diesem Fall wäre eine Verschiebung der Wörter jedoch äußerst störend und keinesfalls wünschenswert. In *Storyfinder* werden deshalb keine Änderungen am Schriftschnitt vorgenommen.

Unterstreichungen

Unterstreichungen von Textteilen ist ein häufig verwendetes Stilmittel zur Hervorhebung. Auf Webseiten werden Unterstreichung jedoch meist ausschließlich zur Kennzeichnung von Hyperlinks verwendet, um diese wichtigen Elemente besonders in den Vordergrund zu stellen. Hierdurch wäre die Gefahr der Verwechslung von Entitäten und Hyperlinks sehr groß, falls auch Entitäten durch Unterstreichungen hervorgehoben werden würden. Dies würde zu einem unerwarteten Benutzerverhalten und somit zu einer geringeren Akzeptanz des *Storyfinder-Plugins* führen. Auch die Verwendung einer anderen Schriftfarbe in Kombination mit Unterstreichungen für Hyperlinks und Entitäten verbessert die Problematik nur unzureichend.

Änderung der Schriftfarbe

Durch die Änderung der Schriftfarbe können Textteile hervorgehoben werden, ohne dass es zu Verschiebungen im Textfluss kommt. Durch verschiedene Farben lassen sich zudem unterschiedliche Merkmale, also im Fall von *Storyfinder* beispielsweise der Entitätstyp kennzeichnen. Zwei Probleme sprechen jedoch gegen die Verwendung unterschiedlicher Textfarben: Zum einen wird die Änderung der Textfarbe, ähnlich wie bereits bei den Unterstreichungen, zur Kennzeichnung von Links eingesetzt. Durch die Verwendung einer zur Linkfarbe verschiedenen Farbe, für die Hervorhebung der Entitäten, kann das Problem der Verwechslung entschärft, jedoch nicht komplett vermieden werden. Zum anderen sind die geeigneten Farben zur Hervorhebung abhängig von der jeweiligen Seite, sowohl durch die Linkfarbe der Seite, aber besonders auch durch die Farbe des Hintergrundes. Die Farben müssten also je Seite entsprechend gewählt werden, um einen entsprechend Kontrast zum Hintergrund zu bilden um eine gute Lesbarkeit des Textes zu gewährleisten. Abgesehen von der vergleichsweisen komplexen und rechenaufwendigen Implementierung führt der Wechsel der Farben von Seite zu Seite dazu, dass der Benutzer keine Assoziation zwischen Farbe und Entitätstyp herstellen kann.

Farbige Hinterlegung

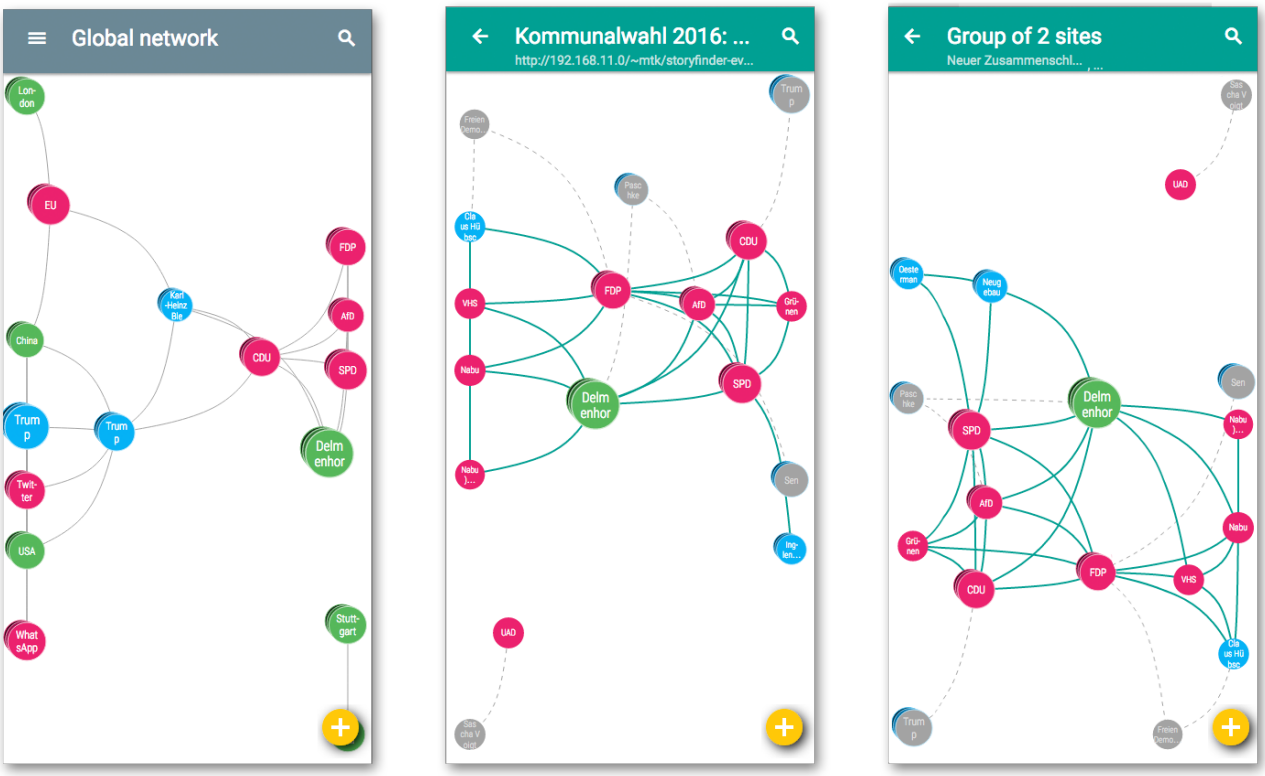
Ähnlich wie bei der Änderung der Schriftfarbe, kann eine farbige Hinterlegung zur Hervorhebung verwendet werden. Im Gegensatz zur Schriftfarbe wird eine farbige Hinterlegung bei Links nur sehr selten eingesetzt, so dass kaum die Gefahr einer Verwechslung zwischen Links und Entitäten besteht. Zudem sind farbige Hinterlegungen auch auf dem Papier ein häufig verwendetes Mittel zur Hervorhebung durch Marker-Stifte, so dass die meisten Benutzer bereits an dieses Stilmittel gewohnt sind und direkt eine Hervorhebung relevanter Textteile hiermit assoziieren.
Wie bei der Schriftfarbe gibt es auch bei den Hinterlegungen eine Abhängigkeit von der Hintergrundfarbe und der Schriftfarbe. Durch eine nur etwa 70 prozentige Deckung der Hintergrundfarbe ist jedoch weiterhin in den meisten Fällen die Lesbarkeit des Textes gegeben.
Wir verwenden in *Storyfinder* deshalb zur Hervorhebung der Entitäten farbige Hinterlegungen. Als Farbe der Hinterlegung dient die Farbe des Entitätstypes. Es wird zudem ein weicher Rand verwendet, um hierdurch eine bessere Assoziation zur Hervorhebung durch Marker-Stifte auf dem Papier herzustellen.

6.4 Wissensgraph

Der Wissensgraph stellt die Kernkomponente der Visualisierung in *Storyfinder* dar. Es handelt sich bei dem Graphen um ein Knoten-Link-Diagramm, das die aus dem Text extrahierten Elemente wie beispielsweise Eigennamen oder Schlüsselwörter als Knoten anzeigt. Die Verknüpfungen zwischen diesen Komponenten, die wie in Kapitel 5.8 beschrieben über Kookurrenzen auf Satzebene gebildet werden, werden als Kanten dargestellt. Das Layout wird mithilfe eines Force-directed Layouts berechnet, wie in Abschnitt 6.4.2 erläutert wird.
Der Wissensgraph kann auf verschiedenen Ebenen betrachtet werden. In der Artikeldarstellung liegt der Fokus auf den Elementen eines einzelnen Artikels. Der globale Wissensgraph berücksichtigt hingegen Elemente aus allen bisher besuchten Webseiten und stellt somit das gesamte über *Storyfinder* erfasste Wissen eines Benutzers dar. Durch Auswahl mehrerer Seiten in der Seitenübersicht können außerdem Gruppen von mehreren Webseiten erstellt und im Wissensgraphen dargestellt werden. Der Gruppengraph visualisiert dann die zu dieser Auswahl aus mehreren Seiten gehörenden Elemente. In Abschnitt 6.4.1 werden die verschiedenen Darstellungsebenen betrachtet sowie die zugrunde liegende Berechnung der anzuzeigenden Elemente erläutert.

6.4.1 Darstellungsebenen

In *Storyfinder* können drei verschiedene Darstellungsebenen für den Graphen ausgewählt werden. Die Darstellungsebenen unterscheiden sich durch die im Graph enthaltenen Elemente bzw. den Fokus der Darstellung auf bestimmte



(a) Der globale Graph zeigt Entitäten aller Seiten.

(b) Der lokale Graph zeigt die Entitäten einer einzelnen Seite.

(c) Der Gruppengraph zeigt die Entitäten mehrere ausgewählter Seiten.

Abbildung 6.3.: Darstellungsebenen des Wissensgraphen

Webseiten. Abhängig von der gewählten Darstellungsebene ändert sich somit die Gewichtung der Elemente und das Highlighting bestimmter Elemente. Der Wechsel zwischen den Ebenen erfolgt hierbei fließend, wie in Abschnitt 6.4.5 ausführlich erläutert wird. Im Folgenden werden die drei Darstellungsebenen beschrieben sowie die zugrunde liegende Berechnung der Elementgewichtung.

Globaler Graph

Der globale Graph zeigt die Gesamtheit der gesammelten Daten, ohne hierbei einen Fokus auf einzelne Webseiten zu legen. Er eignet sich somit besonders als Übersicht und für die allgemeine Recherche. In Abb. 6.3a ist ein globaler Graph dargestellt. Wie zu erkennen ist, erfolgt in dieser Darstellungsebene kein Highlighting einzelner Elemente.
Die Gewichtung und somit die Auswahl der anzuzeigenden Knoten erfolgt auf Basis des Pagerank Algorithmus [14]. Nachdem eine Gewichtung der Knoten berechnet wurde, werden hiervon die $nodes_{top}$ am höchsten gewichteten Knoten als Top-Knoten ausgewählt und im Graphen angezeigt. Für jeden Top-Knoten werden zusätzlich $nodes_{neighbours}$ benachbarte Knoten gewählt, die zusätzlich im Graphen angezeigt werden sollen. Die Auswahl des benachbarten Knoten erfolgt wiederum anhand der Gewichtung. Ist ein Knoten bereits im Graph enthalten, beispielsweise als Top-Knoten oder Nachbar eines anderen Top-Knotens, so wird dieser Knoten bei der Auswahl übersprungen und der nächst niedriger gewichtete Knoten gewählt. Die Anzahl der im Anfangszustand des Graphen anzuzeigenden Top-Knoten ergibt sich als

$$nodes_{top} = \frac{nodes_{max}}{nodes_{neighbours} + 1}$$

Die beiden Variablen $nodes_{max}$ und $nodes_{neighbours}$ werden abhängig von der verfügbaren Anzeigefläche gewählt, wie in Abschnitt 6.6 beschrieben wird.

Artikelgraph

Beim Aufruf einer Webseite im Browser wechselt die Darstellung in *Storyfinder* automatisch auf den Artikelgraphen. Zusätzlich kann der Artikelgraph einer Seite auch jederzeit manuell über die Auswahl einer Webseite in der Seitenübersicht aufgerufen werden. In dieser Darstellung stehen die auf einer Webseite enthaltenen Knoten im Fokus. Knoten, die direkt auf der ausgewählten Seite enthalten sind, werden weiterhin farbig dargestellt. Zusätzlich werden die Kanten zwischen diesen Knoten im Graphen farbig hervorgehoben. Alle anderen Knoten werden in grau mit gestrichelten Kanten

angezeigt. Auf Seite 48 in Abbildung 6.3b ist der Artikelgraph einer Webseite dargestellt.
Für die Gewichtung der Knoten und die Auswahl der $nodes_{top}$ Top-Knoten werden nur die auf der Seite enthaltenen Elemente berücksichtigt. Als Maß wird, anders wie beim globalen Graphen, der TFIDF Wert der Knoten verwendet. Hierdurch werden Knoten mit einem hohen Informationsgehalt, also insbesondere Knoten mit neuen Informationen, bevorzugt. Anhand dieses Wertes werden die $nodes_{top}$ am höchsten bewerteten Knoten als Top-Knoten ausgewählt und im Graphen angezeigt.
Zusätzlich werden für jeden Top-Knoten $nodes_{neighbours}$ benachbarte Knoten ausgewählt. Bei der Auswahl dieser Knoten werden alle Knoten, also nicht nur die Knoten von der ausgewählten Webseite, berücksichtigt. Da somit die Termfrequency alle Knoten von anderen Webseiten Null wäre, wird für die Auswahl der Nachbarknoten wieder auf den PageRank Wert der Knoten zurückgegriffen und für jeden Top-Knoten die $nodes_{neighbours}$ am besten bewerteten Nachbarknoten im Graphen angezeigt.
Der Artikelgraph enthält somit die informativsten Elemente einer Webseite und zusätzlich deren wichtigsten Nachbarknoten von allen Webseiten.

Gruppengraph

Neben der Auswahl einzelner Webseiten können auch mehrere Webseiten ausgewählt werden. Die Auswahl erfolgt durch eine Mehrfachauswahl in der Seitenübersicht.
In dieser Ansicht (siehe S. 48, Abb. 6.3c) werden die Elemente der ausgewählten Seiten fokusiert. Die Darstellung und Gewichtung der Knoten erfolgt analog zum lokalen Graphen. Die Termfrequenz bezieht sich hierbei auf alle ausgewählten Seiten.

6.4.2 Layoutberechnung

Für die Layout-Berechnung wird ein Force-directed Layout [24] verwendet. Hierbei wird in mehreren Iterationsschritten die Einwirkung verschiedener Kräfte auf die einzelnen Knoten simuliert. Die Stärke dieser Kräfte nimmt mit jedem Schritt ab, so dass sich nach einer gewissen Zeit ein festes Layout für den Graphen ergibt.
Obwohl die Darstellung von *Storyfinder* auf der d3 Bibliothek [12] basiert, wird für die Berechnung des Layouts die WebCola Bibliothek [22] verwendet.
WebCola ersetzt das standardmäßig in d3 enthaltene Force-directed Layout durch ein erweitertes Force-directed Layout auf Basis von LibCola [70]. Über stress-majorization [23] wird hierbei die Einhaltung bestimmter separation constraints wie z.B. nicht überlappende Knoten sichergestellt. Zudem ist die Layoutberechnung durch WebCola im Vergleich zu der in d3 enthaltene Methode deutlich stabiler, wodurch insgesamt eine ruhigere Darstellung entsteht.

6.4.3 Knoten im Wissensgraphen und deren Status

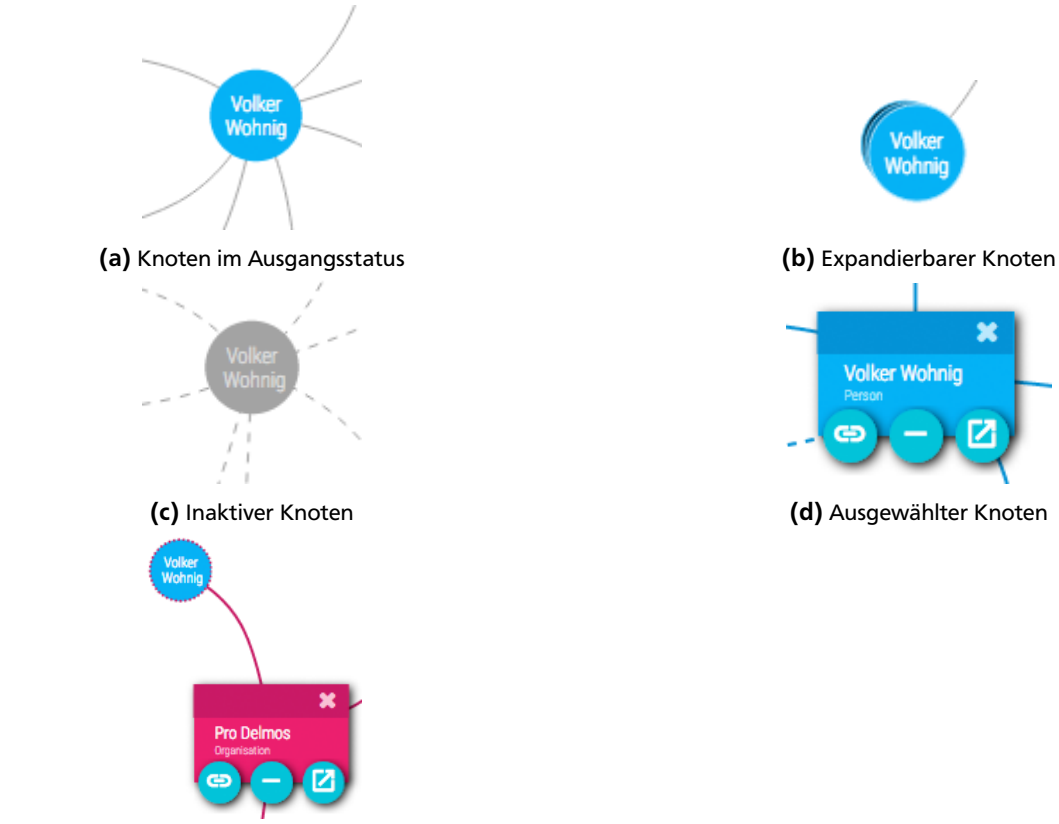
Durch die Knoten im Wissensgraphen werden die extrahierten Textabschnitte dargestellt. Es wird hierbei abhängig vom Elementtyp (Person, Ort, Organisation, Schlüsselwort, Sonstiges), wie in Abschnitt 6.2 beschrieben, jeweils eine andere Hintergrundfarbe verwendet. Diese Farbe wird in allen Status des Elements verwendet, so werden beispielsweise die Kanten zu den Nachbarn eines ausgewählten Knotens in der Farbe des ausgewählten Elements dargestellt.

Form

Bei der Entwicklung von *Storyfinder* wurden neben der verwendeten Darstellung als Kreis, in dem die Beschriftung enthalten ist, zwei weitere Formen für Knoten getestet: Rechtecke und Symbole, die den Elementtyp darstellen. Für die Wahl der Form waren mehrere Kriterien entscheidend:

- Die Beschriftung eines Knotens sollte möglichst vollständig angezeigt werden.
- Bei der Darstellung in der Seitenleiste des Browsers, sowie im Portrait Modus auf einem Smartphone, ist die zur Verfügung stehende Bildschirmbreite eingeschränkt. Die Form sollte somit auch für geringe Bildschirmbreiten geeignet sein.
- Über die Größe der Form soll die Relevanz des Knotens dargestellt werden können.

Anhand dieser Kriterien wurden die verschiedenen Formen beurteilt. Werden die Elemente durch ein Symbol dargestellt, so erhält man neben der Farbe noch ein zweites Kriterium zur Erkennung des Elementtyps. Die Beschriftung der Elemente kann jedoch nicht innerhalb der Form angezeigt werden, wodurch die Lesbarkeit und Zuordnung zu den Knoten schwieriger wird. Zusätzlich spricht gegen die Verwendung von Symbolen, dass sich die Größe von zwei unterschiedlichen Elementtypen (z.B. Person und Organisation) nicht direkt vergleichen lässt.
Dieses Problem besteht auch bei der Verwendung von Rechtecken, falls das Seitenverhältnis nicht bei allen Elementen



(a) Knoten im Ausgangsstatus

(b) Expandierbarer Knoten

(c) Inaktiver Knoten

(d) Ausgewählter Knoten

(e) Nachbarknoten eines ausgewählten Knotens

Abbildung 6.4.: Status eines Knotens im Wissensgraphen

identisch ist. Da zudem die Seitenbreite stark eingeschränkt ist, sollte bei einer Rechteck-Darstellung Quadrate verwendet werden. Die Beschriftung ist innerhalb des Elements möglich und muss gegebenenfalls umgebrochen und gekürzt werden. Problematisch bei der Verwendung von Rechtecken ist die Darstellung des Abstandes zu anderen Elementen, so erscheinen die Kanten zwischen Elementen die diagonal zueinander stehen kürzer, wie die Kanten von Elementen die horizontal oder vertikal auf einer Ebene stehen.
Bei der Darstellung der Elemente als Kreise besteht dieses Problem nicht, da die Abstände in alle Richtungen identisch sind. Zudem lässt sich die Relevanz über den Kreisradius darstellen und gut zwischen den Elementen vergleichen. Wie bei der Darstellung als Quadrate kann die Beschriftung innerhalb des Elements enthalten sein und muss ggf. umgebrochen und gekürzt werden.
Aufgrund der schlechten Vergleichbarkeit von Symbolgrößen und der schwierigeren Darstellung von Abständen zwischen den Elementen bei Rechtecken/Quadraten, verwendet *Storyfinder* Kreise zur Darstellung von Knoten in der Standardansicht. Nur bei geöffneten Knoten wird ein Rechteck verwendet, um das Element deutlich von den anderen Elementen abzugrenzen. Die verschiedenen Status, die ein Knoten annehmen kann, werden im Folgenden beschrieben.

Status

Innerhalb des Wissensgraphen kann ein Knoten unterschiedliche Status annehmen, die den internen Zustand des Knotens (z.B. hervorgehoben oder geöffnet), die Beziehung des Knotens zu seinen Nachbarn (z.B. „der Knoten hat weitere Nachbarn, die noch nicht angezeigt werden" oder „ein Nachbarknoten ist geöffnet"), sowie die Graphzugehörigkeit (z.B. „der Knoten ist auf der ausgewählten Seite enthalten") darstellen. Da sich die Status aus diesen drei Kategorien überschneiden können, visualisieren wir diese in *Storyfinder* durch unterschiedliche Merkmale. Die einzelnen Status sind im Folgenden aufgeführt und in Abbildung 6.4 dargestellt:

- *Standard:* In diesem Modus befinden sich standardmäßig alle Knoten. Der Knoten wird in diesem Fall durch einen farbigen Kreis mit Beschriftung dargestellt (vgl. Abb. 6.4a).
- *Erweiterbar:* Da meistens nicht alle verfügbaren Knoten angezeigt werden können, sind zu besimmten Knoten im Graphen noch weitere ausgeblendete Nachbarn vorhanden. Damit der Benutzer erkennt, dass diese Knoten erweitert werden können, werden hinter den Knoten zwei weitere Kreise angezeigt, so dass der Eindruck eines Stapels aus Knoten entsteht (vgl. Abb. 6.4b).

(a) Vollständiger Name ohne Umbruch

(b) Vollständiger Name mit Silbentrennung

(c) Name mit gekürztem Vornamen und Silbentrennung

(d) Nur Nachname mit Silbentrennung

(e) Nur Nachname ohne Silbentrennung

(f) Abgeschnittener Nachname ohne Silbentrennung

Abbildung 6.5.: Größenabhängige Variationen der Knotenbeschriftungen für den Namen „Franz Beckenbauer"

- *Hervorgehoben:* Wird ein Knoten auf der Webseite mit dem Mauszeiger überfahren (Hover), so wird der zugehörige Knoten im Graphen hervorgehoben. Der Knoten wird in diesem Fall etwas größer dargestellt.
- *Geöffnet:* Klickt der Benutzer auf einen Knoten, um weitere Informationen über diesen Knoten aufzurufen bzw. um weitere Aktionen auf diesem Knoten auszuführen, so wechselt der Knoten in den Status „geöffnet". In diesem Status kann sich jeweils nur ein Knoten befinden. Der Knoten wird in diesem Status deutlich vergrößert und als Rechteck dargestellt. Zusätzlich zur Beschriftung wird der Knotentyp sowie weitere Buttons für die verfügbaren Interaktionsmöglichkeiten angezeigt. Durch einen Schattenwurf erscheint der Knoten als hervorgehoben und näher am Benutzer (vgl. S. 50, Abb. 6.4d).
- *Nachbar geöffnet:* Benachbarte Knoten eines geöffneten Knotens erhalten den Status „Nachbar geöffnet". Die Knoten erhalten einen gestrichelten Rahmen in der Farbe des geöffneten Knotens. Zusätzlich werden auch die Kanten zwischen diesem Knoten und dem geöffneten Nachbarn etwas dicker und in der Farbe des geöffneten Knotens angezeigt (vgl. S. 50, Abb. 6.4e).
- *Kein Nachbar:* Ist ein Knoten im Graphen geöffnet, so erhalten alle anderen Knoten, die kein Nachbar des geöffneten Knotens sind, den Status „kein Nachbar". Die Hintergrundfarbe und Beschriftung dieser Knoten wechselt auf grau und die Kanten werden nur noch gestrichelt dargestellt (vgl. S. 50, Abb. 6.4c).
- *Nicht in der Gruppe:* Falls nicht der globale Graph angezeigt wird, sondern der Fokus auf den Elementen einer bestimmten Webseite oder Gruppe von Webseiten liegt, so werden neben den Elementen aus dieser Gruppe auch wichtige Nachbarn dieser Knoten angezeigt, die nicht in der Gruppe enthalten sind. Für diese Knoten wird ein grauer Hintergrund und graue Schriftfarbe verwendet (vgl. S. 50, Abb. 6.4c).

Größe

Über den Radius des Knotens wird dessen Relevanz innerhalb des Wissensgraphens angezeigt. Die Relevanz ergibt sich aus der Berechnung des Pageranks bzw. TF/IDF Wertes, die in Abschnitt 6.4.1 erläutert wurde. Die Werte werden zwischen 0 und 1 normalisiert, so dass der unwichtigste Knoten den Wert 0 und der wichtigste Knoten den Wert 1 erhält. Abhängig von der verfügbaren Größe für den Wissensgraphen und der Anzahl anzuzeigender Knoten, wird eine untere Schranke für den Knotenradius festgelegt (vgl. Abschnitt 6.6). Hierdurch wird verhindert, dass die Elemente zu klein werden und somit die Beschriftung keinen Platz mehr findet. Zusätzlich wird die doppelte Größe der unteren Schranke als obere Schranke festgelegt. Die Knotenradius wird innerhalb dieser Schranken linear skaliert.

Beschriftung

Die Beschriftung der Knoten wird innerhalb der Knotenkreise angezeigt. Da der zur Verfügung stehende Platz innerhalb der Kreise stark beschränkt ist, wird die Beschriftung gegebenenfalls umgebrochen und in Ausnahmefällen gekürzt. Die Platzierung der Beschriftung erfolgt hierbei in mehreren Iterationsschritten, um das bestmögliche Ergebnis für den verfügbaren Platz zu erzielen:

1. Zuerst wird versucht, den vollständigen Text ohne Umbruch in einer Zeile anzuzeigen.

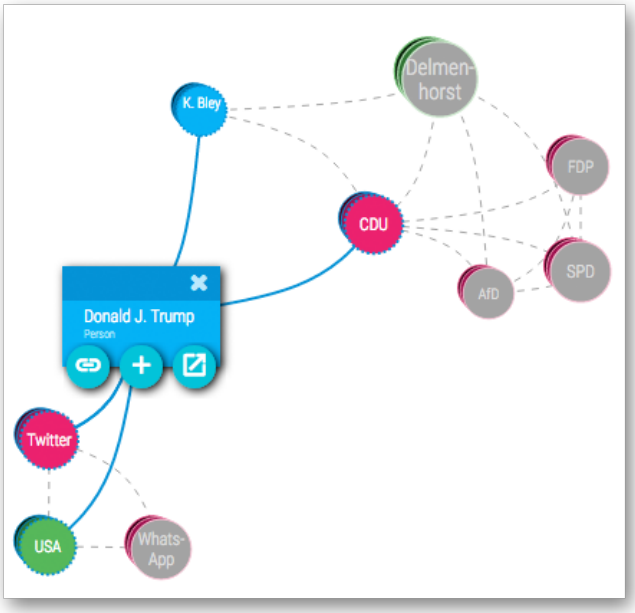


Abbildung 6.6.: Ausgewählter Knoten „Donald J. Trump" mit hervorgehobene Kanten zu den Nachbarknoten. Die Kanten des ausgewählten Knotens sind dicker und in dessen Farbe dargestellt. Kanten zwischen anderen Knoten sind gestrichelt.

2. Im zweiten Schritt wird die Beschriftung umgebrochen. Es wird hierbei eine Silbentrennung verwendet, um passende Stellen für die Umbrüche zu ermitteln. Falls mindestens 50% der Beschriftung im Element angezeigt werden, wird die Silbentrennung verwendet und der überlaufende Text abgeschnitten.

3. Falls weniger als 50% der Beschriftung im Element Platz finden, wird keine Silbentrennung verwendet, sondern an beliebiger Stelle umgebrochen, um den größtmöglichen Teil der Beschriftung anzuzeigen.

Einen Sonderfall bilden Personennamen. In diesem Fall wird der Fokus auf den Nachnamen gelegt. Die zuvor beschriebenen Iterationsschritte werden für verschiedene Namensvarianten durchlaufen, bis eine Variation gefunden wird, bei der kein Text abgeschnitten wird:

1. In der ersten Iteration wird versucht, den vollständigen Namen im Element zu platzieren.

2. Im zweiten Iterationsschritt werden alle Namen außer dem Nachnamen auf ein Zeichen gekürzt. Das letzte Token wird jeweils als Nachname betrachtet. Der Name „Johann Wolfgang von Goethe" wird beispielsweise zu „J. W. v. Goethe". Auch dieser Name wird versucht im Element zu platzieren.

3. Im letzten Iterationsschritt wird nur der Nachname verwendet, also beispielsweise nur „Goethe". Dieser Name wird im Element platziert und ggf. abgeschnitten.

In Abbildung 6.5 auf Seite 51 sind die verschiedenen Variationen der Knotenbeschriftungen für den Namen „Franz Beckenbauer" dargestellt.

6.4.4 Darstellung von Beziehungen im Wissensgraphen

Die Beziehungen zwischen zwei Elementen bilden die Kanten im Wissensgraphen. Standardmäßig werden die Kanten als einfache graue Linien dargestellt. Die Kanten können jedoch verschiedene Farben, Stärken und Musterungen annehmen, abhängig vom Status der verbundenen Knoten und der aktuellen Darstellungsebene. Im Folgenden sind die verschiedenen Darstellungen aufgeführt:

Status der verbundenen Knoten

Falls ein Knoten im Graphen ausgewählt ist, so werden alle Kanten zwischen dem ausgewählten Knoten und den benachbarten Knoten durch eine stärkere Linie gekennzeichnet. Zudem wird die zum Knotentyp gehörende Farbe als Kantenfarbe verwendet. Alle anderen Kanten im Graphen werden gestrichelt angezeigt (siehe Abbildung 6.6).

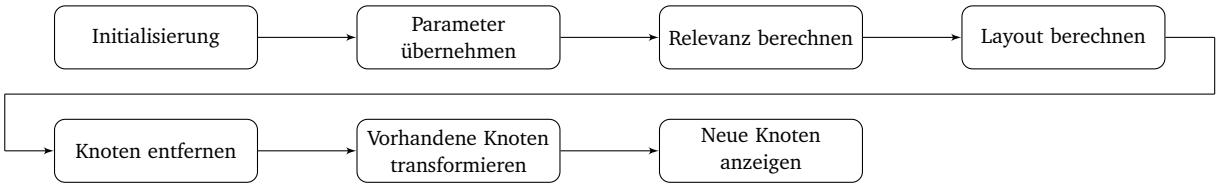


Abbildung 6.7.: Transformationsschritte beim Zustandsübergang

Darstellungsebene

Falls ein lokaler oder Gruppengraph angezeigt wird, werden im Graphen sowohl Knoten angezeigt, die zu der Auswahl gehören, als auch benachbarte Knoten die nicht innerhalb der Auswahl liegen. Die Kanten zwischen zwei zur Gruppe gehörenden Knoten werden in diesen Fällen farbig dargestellt.

Benannte Beziehungen

Beziehungen zwischen zwei Elementen können zudem benannt werden. Es kann hierbei, wie in Kapitel 4.3.1 beschrieben, mehrere Beschriftungen für eine Beziehung geben. Dies ist beispielsweise dann der Fall, wenn die Beziehung in mehreren Sätzen gefunden wurde. Im Graphen wird jedoch jeweils nur eine Beschriftung für die Beziehung angezeigt und es muss deshalb gegebenenfalls ein Label ausgewählt werden. *Storyfinder* verwendet in solchen Fällen die neueste Beschriftung dieser Beziehung.

6.4.5 Übergänge zwischen Graphen

Die Menge der im Wissensgraph angezeigten Elemente kann sich durch Benutzerinteraktion verändern. Storyfinder unterscheidet hierbei zwischen lokalen Änderungen, die sich innerhalb der angezeigten Darstellungsebene bewegen und globalen Änderungen aufgrund eines Wechsels zwischen Darstellungsebenen oder deren Zusammenstellung. Folgende lokalen Änderungen können in *Storyfinder* auftreten:

- Einblenden weiterer Nachbarn eines Knotens.
- Ausblenden der Nachbarn eines Knotens.
- Löschen eines Knotens.
- Zusammenführen zweier Knoten.
- Hinzufügen eines neuen Knotens.

Bei allen lokalen Änderungen werden Änderungen an einzelne Knoten und gegebenenfalls deren Nachbarn durchgeführt. Bei den globalen Änderungen, die im Folgenden aufgeführt sind, werden mehrere unabhängige Knoten bis hin zu allen Knoten des Graphens geändert:

- Wechsel vom lokalen zum globalen Graph.
- Wechsel zwischen globalen und Gruppen-Graph.
- Änderung der anzuzeigenden Webseite im lokalen Graphen.
- Auswahl einer anderen Gruppe für die Anzeige im Gruppengraphen.
- Hinzufügen oder Entfernen von Seiten aus dem Gruppengraphen.

Unabhängig von der Art der Änderung werden immer die im Folgenden aufgeführten Schritte während der Transformation zwischen den beiden Zuständen in der hier angegebenen Reihenfolge abgearbeitet. Die Menge der betroffenen Elemente für einzelne Schritte kann hierbei auch leer sein (z.B. das Entfernen von Knoten beim Erweitern eines Knotens). Der jeweilige Schritt wird in diesem Fall übersprungen. In Abbildung 6.7 ist der Ablauf dargestellt.

Initialisierung

Zu Beginn der Transformation wird diese initialisiert. Hierzu werden zuerst alle aktiven Animationen beendet und die gegebenenfalls noch aktive Layoutberechnung aus der vorherigen Transformation gestoppt. Alle Eingaben und Interaktionsmöglichkeiten werden deaktiviert, geöffnete Knoten werden geschlossen und der Graph wird als inaktiv (alle Knoten und Kanten werden grau angezeigt) dargestellt.

Übernahme der bisherigen Parameter

Im zweiten Schritt werden die Parameter der momentan noch sichtbaren Knoten für die weitere Berechnung übernommen. Hierzu zählen die aktuellen Koordinaten jedes Knotens und die Anzahl der manuell eingeblendeten Nachbarn. Durch die Übernahme der bisherigen Koordinaten als Startwert für die Berechnung des neuen Layouts werden größere Positionsänderungen nach Möglichkeit verhindert.

Relevanz und Layout berechnen

Die Berechnung der Relevanz, die auch entscheidend dafür ist, welche Knoten angezeigt werden, wird wie in Kapitel 6.4.1 erläutert berechnet. Hierdurch ergibt sich die Menge der neu anzuzeigenden und der zu entfernenden Knoten. Durch die Berechnung eines neuen Layouts für die zukünftig anzuzeigenden Elemente, werden die neuen Positionen der Knoten festgelegt.
Abgesehen vom Wechsel zum inaktiven Graphen, finden die ersten drei Schritte ohne Animation statt und deren Dauer ist daher nur von der Rechenleistung abhängig. In der folgenden zweiten Hälfte der Transformation werden die neu berechneten Daten angewendet und dem Benutzer die hierfür nötigen Änderungen durch entsprechende Animationen visualisiert.

Nicht mehr benötigte Knoten entfernen

Die erste für den Benutzer sichtbare Änderung der Transformation ist das Entfernen nicht mehr benötigter Knoten. In diesem Fall bewegen sich die Knoten nach links, bis sie den sichtbaren Bereich vollständig verlassen haben. Die Animation in Richtung des linken Randes wurde bewusst gewählt, da die Sidebar im Browser standardmäßig[1] am linken Fensterrand angezeigt wird. Eine Bewegung der Knoten nach links zeigt somit eine Bewegung von der angezeigten Webseite weg, aus dem Browserfenster heraus.
Eine Ausnahme bildet das Entfernen von Knoten, falls ein zuvor expandierter Knoten wieder geschlossen wird. In diesem Fall bewegen sich die Knoten von ihrer aktuellen Position auf den Elternknoten zu. Hierbei werden die Knoten kleiner und transparent, bis sie am Elternknoten angekommen vollständig ausgeblendet sind. Es wird hierdurch visualisiert, dass die Knoten wieder hinter dem Elternknoten, der als Stapel angezeigt wird, verschwinden.

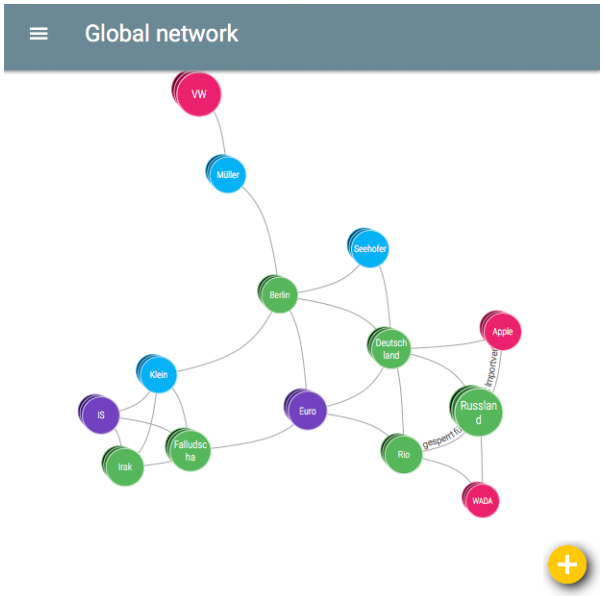
Verbleibende Knoten transformieren

In diesem Schritt werden im Graphen nur die Knoten angezeigt, die sowohl im ursprünglichen Zustand als auch im neuen Zustand angezeigt werden sollen. Die Knoten bewegen sich hierbei zu ihrer neuen Position.

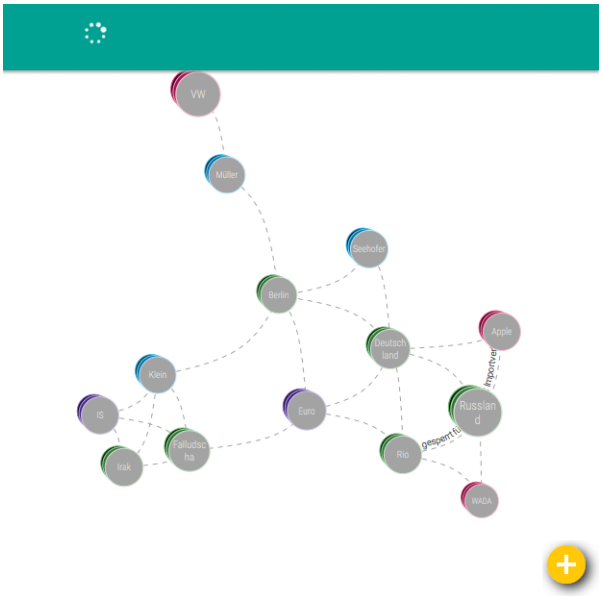
Neue Knoten einblenden

Das Einblenden der neuen, bisher nicht sichtbaren Knoten, stellt den letzten Schritt der Transformation dar. Die Startposition der neuen Knoten ist rechts vom Graphen und somit außerhalb des sichtbaren Bereichs. Von dort bewegen sie sich zu ihrer Zielposition. Wie bereits beim Löschen, ist die Bewegungsrichtung bewusst gewählt und bewegt sich aus Richtung der angezeigten Webseite zum Graphen hin.
Eine Ausnahme bilden neuen Knoten, die aufgrund des Expandierens eines Knotens angezeigt werden. Werden die Nachbarn eines Knotens eingeblendet, so starten diese von der selben Position wie die Elternknoten. Von dort bewegen sie sich zu ihrer Zielposition. Es wird hierdurch visualisiert, dass die Knoten im Elternknoten enthalten waren und dort auch wieder ausgeblendet werden können.
Die Abbildung 6.8 auf den folgenden beiden Seiten zeigt die Änderungen am Graphen in den einzelnen Schritten.

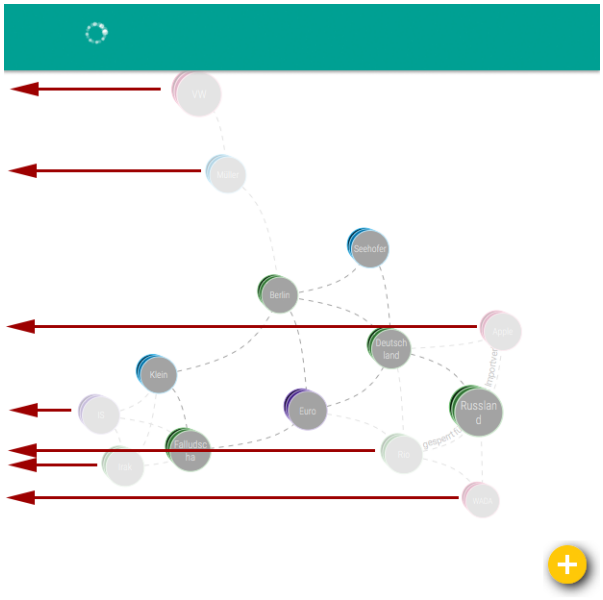
[1] Die Browser-API von *Mozilla Firefox* bieten leider keine Möglichkeit, die Position der Sidebar innerhalb des Browsers zu ermitteln.



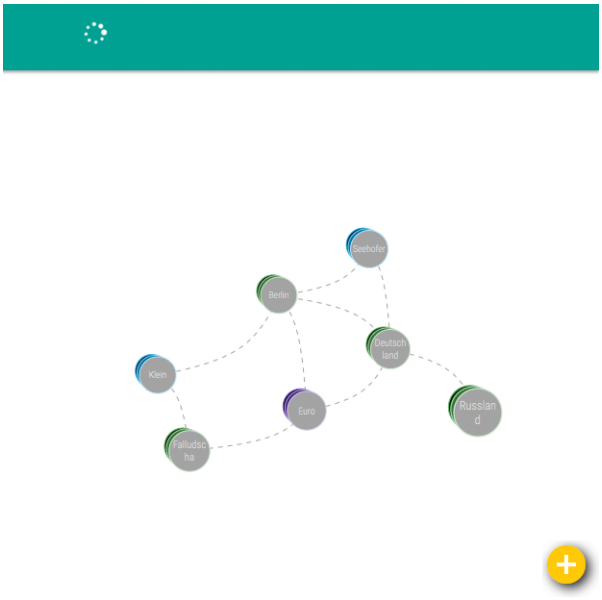
(a) Startzustand



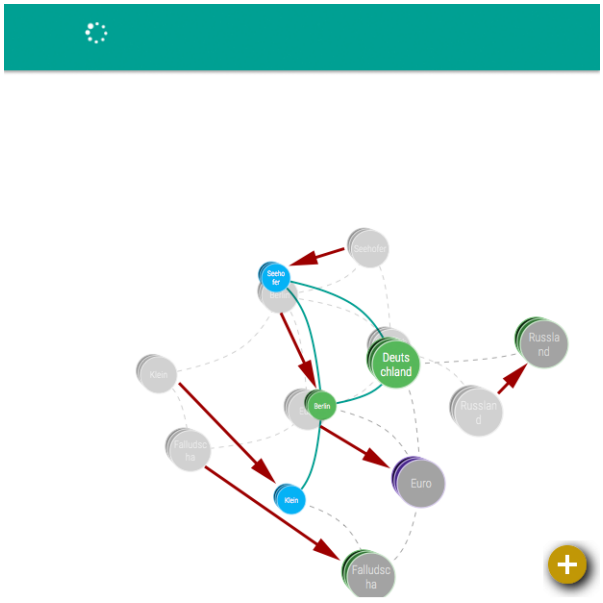
(b) Bei der Initialisierung wird der Graph als inaktiv gekennzeichnet.



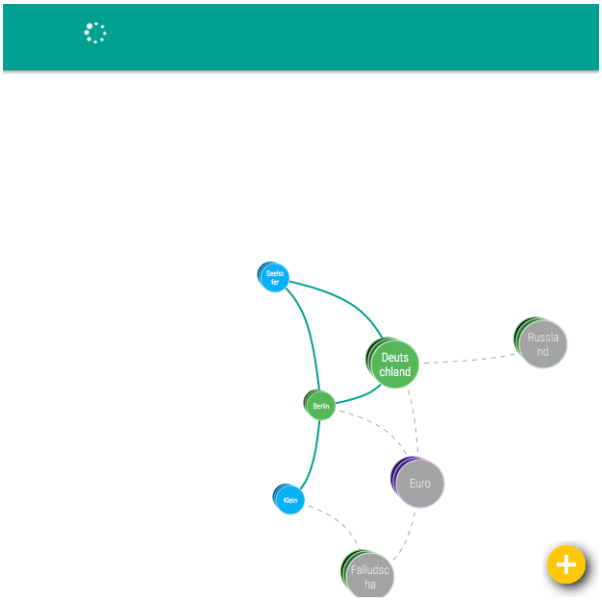
(c) Nicht mehr enthaltene Knoten werden nach links aus dem sichtbaren Bereich hinaus bewegt.



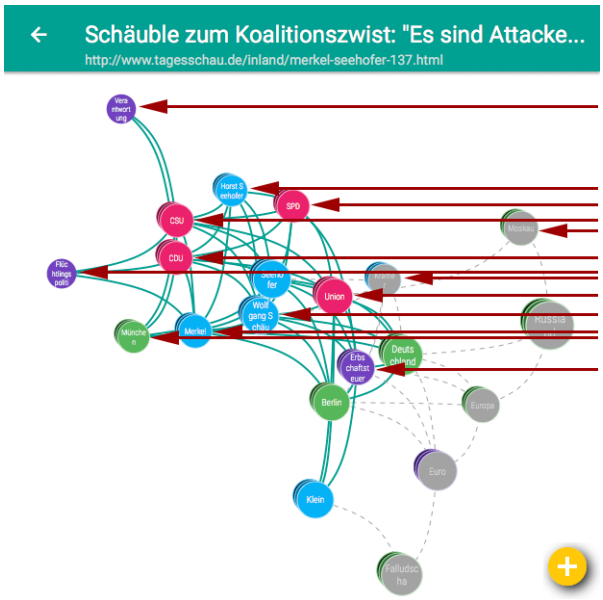
(d) Nach dem Löschen zeigt der Graph nur noch die Knoten an, die in beiden Zuständen enthalten sind.



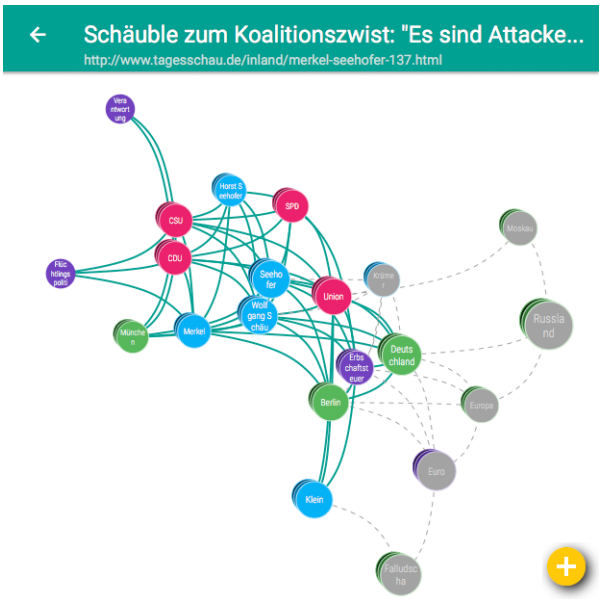
(e) Die verbleibenden Knoten werden an ihre neue Position verschoben.



(f) Die verbleibenden Knoten haben anschließend ihre Endposition erreicht.



(g) Neue Knoten werden vom rechten Rand an ihre neue Position bewegt.



(h) Endzustand des Graphen

Abbildung 6.8.: Änderungen am Zustand des Graphen werden durch animierte Übergänge für den Benutzer nachvollziehbar.

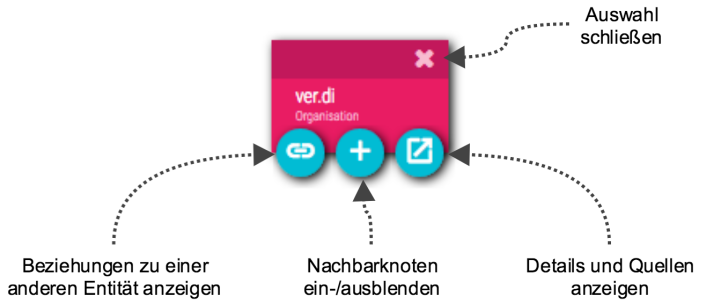


Abbildung 6.9.: Geöffneter Knoten mit den verfügbaren Interaktionsmöglichkeiten

6.4.6 Interaktionsmöglichkeiten

Der Wissensgraph bietet verschiedene Interaktionsmöglichkeiten um im Graphen zu recherchieren und diesen zu bearbeiten, die im Folgenden erläutert werden.

Auswählen eines Knotens

Durch das Auswählen eines Knotens wird dieser Knoten hervorgehoben und bildet das zentrale Element des Graphen. Die Auswahl eines Knotens erfolgt durch einen Klick auf diesen Knoten, wodurch die Darstellung des Knotens von einem Kreis zu einem deutlich größeren Rechteck wechselt (vgl. Abb. 6.9). Der Knoten erhält zudem einen Schattenwurf und rückt somit optisch näher an den Benutzer. Die Kanten zu benachbarten Knoten werden in der Farbe des ausgewählten Knotens dargestellt. Alle nicht benachbarten Knoten werden in grau dargestellt und rücken somit in den Hintergrund. Hierdurch heben sich die Beziehungen und Nachbarn des ausgewählten Knotens deutlich vom restlichen Graphen ab.
Da der ausgewählte Knoten vergrößert wird, benötigt dieser mehr Platz innerhalb des Graphen. Um eine Überlappung mit anderen Knoten zu vermeiden, wird eine tonnenförmige Verzeichnung („Fischauge-Effekt") mit dem ausgewählten Knoten als Zentrum auf den Graphen gelegt. Die Verzeichnung wird hierbei allerdings nur auf den Abstand der Knoten, nicht jedoch auf die Knotenform angewendet. Die Knoten werden somit sternförmig vom ausgewählten Knoten weg bewegt, wobei Knoten nahe des ausgewählten Knotens stärker verschoben werden, als Knoten die weiter weg liegen. Hierdurch wird der zusätzliche Platz um den Knoten geschaffen, ohne die Gesamtabmessung des Graphen deutlich zu vergrößern (siehe S. 58, Abb. 6.10).
Der ausgewählte Knoten lässt sich per Drag & Drop verschieben, um diesen zu löschen oder mit einem anderen Knoten zusammenzuführen. Es werden außerdem drei Schaltflächen angezeigt, um weitere Details zu diesem Knoten oder zu den Beziehungen zu anderen Knoten anzuzeigen, sowie zum Anzeigen weiterer benachbarter Knoten (vgl. Abb. 6.9). Diese Interaktionsmöglichkeiten werden in den folgenden Abschnitten beschrieben.

Öffnen der Knotendetails

Das Öffnen der Knotendetails erfolgt über die Schaltfläche „Details und Quellen anzeigen". Nach dem Öffnen wird der Artikelgraph durch eine Detailansicht mit weiteren Informationen zum ausgewählten Knoten überlagert, die in Kapitel 6.5 beschrieben werden.

Anzeigen weiterer Knoten

Standardmäßig werden nur die wichtigsten Nachbarn eines Knotens angezeigt. Falls darüber hinaus noch weitere Nachbarn vorhanden sind, können diese über die Schaltfläche „Nachbarknoten ein-/ausblenden" eingeblendet werden.

Löschen eines Knotens

Das Löschen eines Knotens erfolgt per Drag & Drop. Nachdem der Knoten geöffnet ist, kann dieser verschoben werden. Beim Verschieben wird ein Papierkorb in der linken oberen Ecke angezeigt. Wird der Knoten auf den Papierkorb gezogen, so wird dieser gelöscht.

Zusammenführen zweier Knoten

Das Zusammenführen zweier Knoten zu einem Knoten erfolgt analog zum Löschen. Per Drag & Drop wird einer der beiden Knoten auf den anderen Knoten gezogen. Die Knoten werden anschließend zusammengeführt, so dass im Graph nur noch ein Knoten angezeigt wird. Alle Relationen und Quellen der beiden Knoten, werden diesem einen Knoten zugeordnet. Ist eine der beiden Entitäten zukünftig in einem Artikel enthalten, so wird diese immer dem gemeinsamen Knoten zugeordnet.

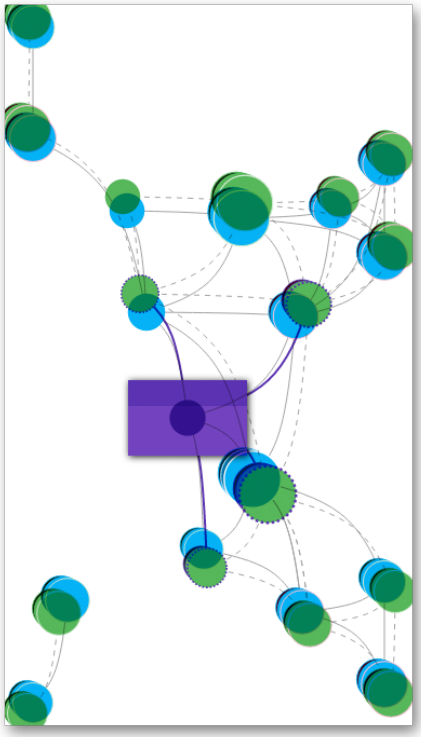


Abbildung 6.10.: Die Abbildung zeigt den „Fischauge-Effekt" beim Öffnen eines Knotens. Zur besseren Veranschaulichung werden in der Abbildung beide Zustände des Graphens, also vor (blau) und nach (grün) dem Öffnen eines Knotens, überlagernd angezeigt. Der kreisförmige, in dunklem lila dargestellte Knoten, wird geöffnet und anschließend als Rechteck (ebenfalls lila) angezeigt. Die ursprüngliche Position der anderen Knoten, hier in blau dargestellt, ändern sich zur neuen Position, hier in grün dargestellt. Hierdurch wird dem geöffneten Knoten mehr Platz zur Verfügung gestellt.

Beziehungen zwischen zwei Entitäten anzeigen

Um die Beziehungen zwischen zwei Knoten anzuzeigen, kann auf die Kante zwischen diesen Knoten geklickt werden. Anschließend öffnet sich die Detailansicht, in der alle Sätze angezeigt werden, in denen diese Beziehung gefunden wurde. Über diese Detailansicht können Beschriftungen für die Beziehung im Allgemeinen oder für die Beziehung innerhalb eines einzelnen Satzes hinzugefügt werden.
Falls für die Entitäten bisher keine Beziehung existiert, wird im Graphen keine Kante zwischen den Elementen angezeigt. Soll eine Beziehung zwischen solchen Knoten erstellt werden, so muss einer der beiden Knoten ausgewählt werden. Anschließend wird die Schaltfläche „Beziehungen zu einer anderen Entität anzeigen" angeklickt und danach der zweite Knoten ausgewählt. *Storyfinder* öffnet dann entweder die Detailansicht der bereits vorhandenen Beziehung oder fragt den Benutzer, ob eine neue Beziehung erstellt werden soll.

Suche im Graphen

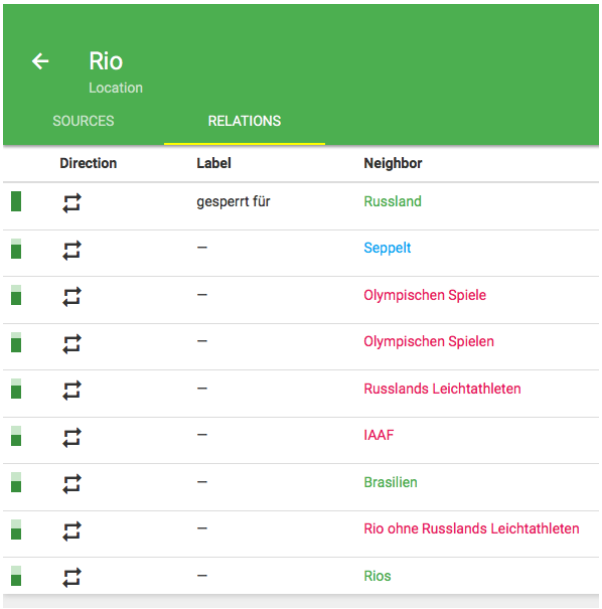
Über die Knotensuche ist eine Suche nach bestimmten Knoten und Seiten möglich. Der Benutzer gibt hierzu in ein Textfeld einen Teil des Knotennamens ein. Die Suche wird direkt während der Eingabe ausgeführt, so dass der Benutzer sofort die möglichen Treffer angezeigt bekommt.
Bei der Suche nach Knoten wird innerhalb des Knotennamens gesucht und die Knoten zurückgeliefert, die den Suchbegriff enthalten. Zusätzlich wird in allen bisher gelesenen Artikeln nach dem Suchbegriff gesucht. Hierbei ist irrelevant, ob der Suchbegriff zuvor als Entität erkannt wurde. Die Seiten, in denen der Begriff enthalten ist, werden ebenfalls in der Ergebnisliste angezeigt, zusammen mit den ersten drei Sätzen, die den Suchbegriff beinhalten.
Die Suche basiert auf einer Volltextsuche in der Datenbank.

6.5 Quellen- und Recherchemöglichkeiten

Innerhalb des Wissensgraphen von *Storyfinder* werden die Entitäten und deren Beziehung dargestellt. Um weitere Informationen über die Quellen der Entitäten und Beziehungen zu erhalten, können diese in *Storyfinder* ausgewählt und

Abbildung 6.11.: Überlagerung des Wissensgraphen durch die Quellenansicht einer Entität.

(a) Webseiten der Entität



(b) Beziehungen zu anderen Entitäten

Abbildung 6.12.: Die Quellenansicht einer Entität („Rio") zeigt in zwei Reitern die Webseiten an, auf denen die Entität enthalten ist (a) sowie die Beziehungen zu anderen Entitäten (b).

die entsprechenden Details aufgerufen werden, wie im vorherigen Abschnitt beschrieben wurde. Die Quellenansicht überlagert hierbei den Wissensgraphen, wie in Abbildung 6.11 dargestellt ist.

Quellen einer Entität

Die Quellenansicht für eine Entität (vgl. Abb. 6.12) enthält in der Kopfzeile den Entitätsnamen und Entitätstyp. Der Hintergrund des Kopfes ist in der zum Entitätstyp gehörenden Farbe eingefärbt.
Der Inhaltsbereich enthält die zwei Reiter „Webseiten" und „Beziehungen".
Unter dem Reiter Webseiten, der standardmäßig angezeigt wird, sind alle Webseiten aufgelistet, in denen diese Entität enthalten ist. Die Sortierung der Seiten erfolgt absteigend chronologisch, so dass die zuletzt aufgerufenen Seite an erster Stelle angezeigt wird. Jede Webseite wird hierbei mit einem Vorschaubild der Seite innerhalb einer eigenen Box dargestellt. Die Hintergrundfarbe der Box entspricht der dominanten Farbe der zugehörigen Webseite. Hierdurch wird die Wiedererkennung einer Webseite erleichtert, so dass diese besser mit den zugehörigen Inhalten assoziiert werden kann. Abhängig von der verwendeten Hintergrundfarbe, muss entweder eine helle oder dunkle Schriftfarbe verwendet werden, damit sich diese gut vom Hintergrund abhebt. Die Farbe wird wie in [55] beschrieben ermittelt. Hierdurch wird der optimale Kontrast nach den W3C Empfehlungen [66] erzielt. Zur Berechnung wird die Leuchtdichte (Luminance) der dominanten Farbe der Webseite mit schwarz oder weiß kombiniert, und hierüber bestimmt, welche Kombination den höheren Kontrast erzielt.
In jeder Box werden das Datum des letzten Seitenaufrufs und bis zu drei Sätzen angezeigt, in denen die Entität enthalten ist. Sind noch weitere Sätze vorhanden, so können diese bei Bedarf zusätzlich eingeblendet werden.
Am unteren Ende der Box werden Links zum Aufrufen der Quelle angezeigt. Es kann hierfür entweder direkt die URL geöffnet werden, so dass die aktuelle Version der Webseite angezeigt wird oder die im Archiv gespeicherte Version des

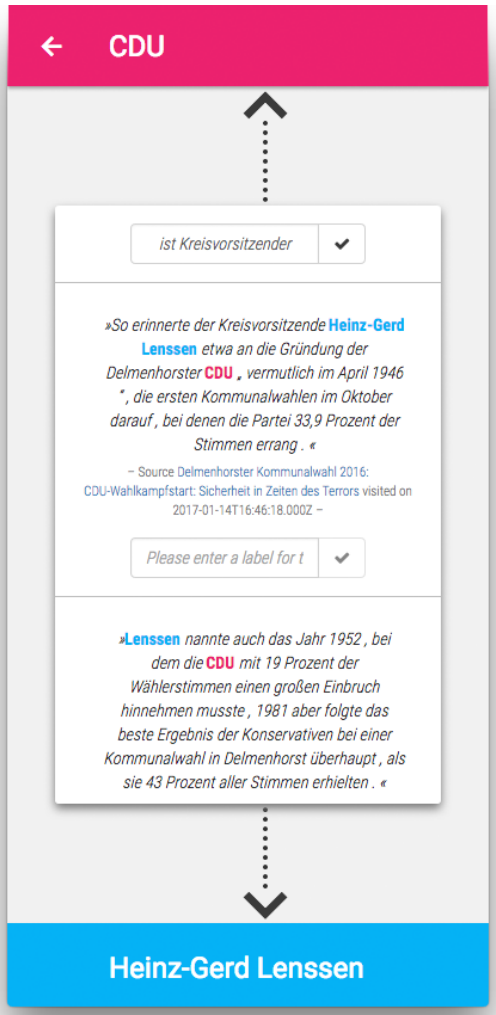


Abbildung 6.13.: Detailansicht der Beziehung zwischen der „CDU" und „Heinz-Gerd Lenssen"

Artikels, die den Stand des Abrufdatums widerspiegelt.
Unter dem Reiter „Beziehungen" werden die Beziehungen der ausgewählten Entität zu anderen Entitäten aufgelistet. Die Sortierung erfolgt absteigend nach Häufigkeit, so dass Beziehungen, die häufig gefunden wurden, vor Beziehungen angezeigt werden, die nur selten gefunden wurden.
Für jede Beziehung wird die primäre Bezeichnung der Beziehung und der Name der anderen Entität angezeigt. Zusätzlich wird über ein kleines Balkendiagramm die Häufigkeit im Verhältnis zu den anderen Beziehungen dieser Entität dargestellt. Durch Anklicken der Beziehung öffnet sich die Quellenansicht für diese Beziehung.

Quellen einer Beziehung

Die Quellenansicht einer Beziehung (siehe Abbildung 6.13) kann entweder aus der Detailansicht einer Entität oder direkt über den Wissensgraphen aufgerufen werden.
Die Ansicht enthält in der Kopfzeile die erste Entität der Beziehung und deren Entitätstyp und in der Fußzeile die zweite Entität und deren Entitätstyp. Der Hintergrund der Kopf- und Fußzeile ist in der zum Entitätstyp gehörenden Farbe eingefärbt.
Zwischen den beiden Entitäten werden die Sätze aufgelistet, in denen diese Beziehung gefunden wurde. Es werden hier zuerst die Sätze mit Beziehungstyp absteigend chronologisch und anschließend die Sätze ohne Beziehungstyp, ebenfalls absteigend chronologisch, angezeigt. Für jeden Satz wird der Titel der Quellwebseite mit Abrufdatum sowie ein Eingabefeld zum Festlegen eines Beziehungstyps angezeigt. Durch einen Klick auf den Titel der Quellwebseite wird die Archivversion dieser Webseite geöffnet und der ausgewählte Satz hervorgehoben.
Zusätzlich zu den satzabhängigen Beziehungstypen kann ein primärer Beziehungstyp festgelegt werden, der im Gra-

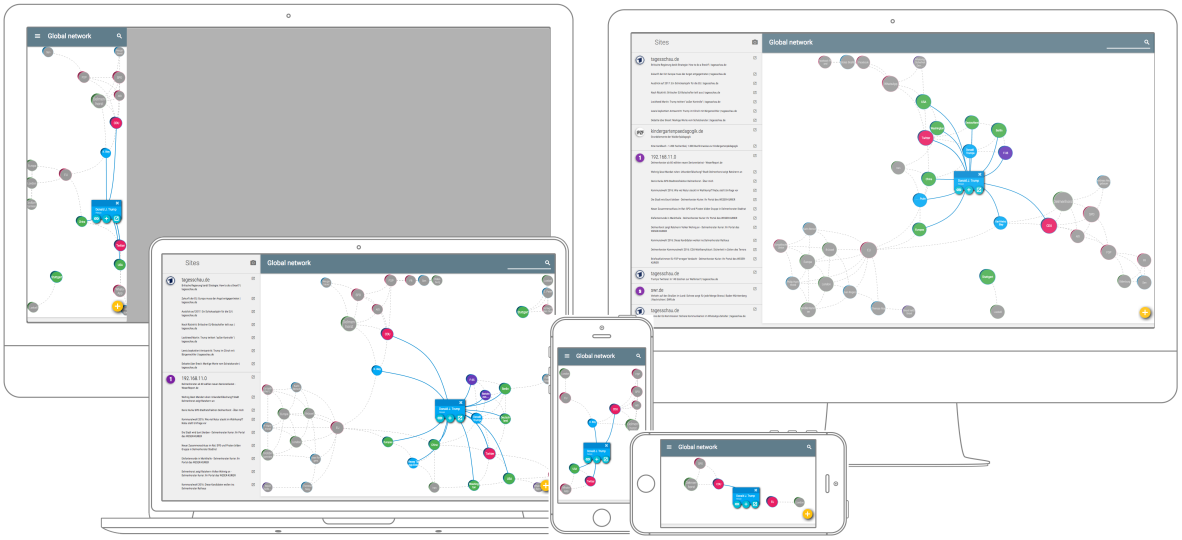


Abbildung 6.14.: Der *Storyfinder* Wissensgraph auf verschiedenen Geräten. Hinten v.l.n.r: Sidebar, Desktop-PC; Vorne v.l.n.r: Notebook, Smartphone vertikal, Smartphone horizontal

phen angezeigt wird. Dieser primäre Beziehungstyp ist unabhängig von einem bestimmten Satz. Wurde kein primärer Beziehungstyp festgelegt, so wird der zuletzt festgelegte satzabhängige Beziehungstyp im Graphen angezeigt.

6.6 Anpassung an unterschiedliche Anzeigegrößen

Eine der schwierigsten Anforderungen bei der Entwicklung von *Storyfinder* war die Anpassung der Visualisierung an unterschiedliche Geräte mit einer großen Spanne verschiedener Bildschirmauflösungen, Punktdichten und Ausrichtungen. Dies war notwendig, um die Darstellung von *Storyfinder* auf verschiedenen Geräten zu ermöglichen. In Abbildung 6.14 sind die verschiedene Darstellungsvarianten von *Storyfinder* abgebildet.

Vergleich verschiedener Anzeigengröße

Im Second-Screen Modus kann *Storyfinder* auf Smartphone Displays angezeigt werden, die besonders im Portraitmodus nur eine sehr eingeschränkte Bildschirmbreite besitzen. Ähnlich viel Platz steht für die Anzeige in der Seitenleiste des Browsers zur Verfügung, wobei hier die Punktdichte meist niedriger und somit die Anzeigengröße deutlich größer ist. Diese Ansicht ist besonders für den Kontextbezug zur dargestellten Webseite wichtig.
Für die Recherche im Wissensgraph bietet sich jedoch auch die Anzeige von *Storyfinder* in einem eigenen Browser-Tab oder Fenster an, da dort mehr Entitäten dargestellt werden können. In diesem Fall steht im Idealfall nahezu die komplette Bildschirmbreite und Höhe zur Verfügung.

Ausrichtung

Nicht nur die reine Größe des zur Verfügung stehenden Platzes wechselt zwischen den unterschiedlichen Ansichten, sondern auch die Ausrichtung. Bei der Anzeige in der Seitenleiste steht normalerweise deutlich mehr vertikaler Anzeigenbereich zur Verfügung, da nur die Breite durch die daneben stehende Webseite eingeschränkt ist. Bei der Nutzung des kompletten Fensters steht hingegen meist deutlich mehr horizontaler Platz zur Verfügung. Zusätzlich ist bei den meisten als Second Screen genutzten Geräten sowohl eine vertikale als auch eine horizontale Ausrichtung des Gerätes möglich.
Die gesamte Benutzeroberfläche von *Storyfinder* und insbesondere der Wissensgraph wird deshalb an die jeweils zur Verfügung stehende Bildschirmauflösung angepasst.

Anpassung der Benutzeroberfläche

Um die Benutzeroberfläche vollständig responsive abzubilden, verwenden wir in *Storyfinder* das Bootstrap CSS Framework und richten uns bei den Abmessungen für Buttons und anderen UI-Elementen nach den Vorgaben aus dem Material Design Guide [36]. Hierdurch wird sichergestellt, dass alle Elemente entsprechend dimensioniert sind, um diese auch auf einem Smartphone gut erkennen und mit dem Finger bedienen zu können.
Bei Desktopauflösungen ab 960px Breite wird die Benutzeroberfläche von *Storyfinder* zweispaltig angezeigt. In der linken, kleineren Spalte findet sich die Liste der besuchten Seiten. In der rechten Spalte wird die Titelleiste und der Wissensgraph angezeigt, sowie die Entitäten- und Beziehungsdetails, wenn diese aufgerufen werden. Bei niedrigeren Auflösungen wird

nur die rechte Spalte angezeigt. Die ursprüngliche Liste der besuchten Seiten ist in diesem Fall standardmäßig ausgeblendet und lässt sich über einen Button in der Titelzeile einblenden, wodurch der Graph von dieser Liste überlagert wird. Die Breite der Titelleiste und des Graphen passt sich an die verfügbare Bildschirmbreite an.
Die Quellenangaben zu einer Entität werden je nach verfügbaren Platz ein- bis dreispaltig angezeigt. Hierdurch wird vermieden, dass bei höheren Auflösungen das Vorschaubild der Webseite zu viel Platz einnimmt.

Anpassung des Graphen

Besonders die Anpassung des Wissensgraphen, der den größten Teil der Benutzeroberfläche einnimmt, ist für die korrekte Darstellung von *Storyfinder* auf unterschiedlichen Bildschirmgrößen entscheidend. Je nach Auflösung ändert sich die Größe und die Anzahl der angezeigten Knoten sowie die Kantenlänge zwischen den Knoten.
Die Ausgangsgröße der Knoten wird zwischen einem fest definierten minimalen Radius r_{min} und maximalen Radius r_{max} linear anhand der zur Verfügung stehenden Breite $graph_{width}$ oder Höhe $graph_{height}$ des Graphen skaliert. Es wird hierbei der größere Wert der beiden Dimensionen verwendet:

$$r_{node} = \min\left(r_{max}, \max\left(r_{min}, \frac{\max(graph_{width}, graph_{height})}{50}\right)\right)$$

Dieser Radius dient als Ausgangsgröße und wird je Knoten abhängig von dessen Gewichtung auf maximal die doppelte Größe skaliert.
Die Anzahl der anzuzeigenden Knoten $nodes_{max}$ wird über den Quotienten der verfügbaren Fläche a_{graph} und der benötigten Fläche für einen Knoten a_{node} berechnet. Da die Knoten somit jedoch den gesamten freien Platz einnehmen würden, wird die Fläche eines Knotens mit dem Faktor s multipliziert. Hierdurch wird eine Abstandsfläche um jeden Knoten berücksichtigt:

$$nodes_{max} = \frac{a_{graph}}{a_{node} \cdot s} \text{ mit } a_{node} = (r_{node} \cdot 2)^2 \text{ und } a_{graph} = graph_{width} \cdot graph_{height}$$

Zusätzlich zur Gesamtzahl der Knoten $nodes_{max}$ ändert sich auch die Menge der anzuzeigenden Top-Knoten $nodes_{top}$ und der Anzahl an Nachbarknoten je Top-Knoten $nodes_{neighbours}$. Die Anzahl der Nachbarknoten je Top-Knoten ergibt sich anhand der Gesamtzahl anzuzeigender Knoten $nodes_{max}$:

$$nodes_{neighbours} = \begin{cases} 0 & \text{für } nodes_{max} < 16 \\ 1 & \text{für } 16 \geq nodes_{max} < 24 \\ 2 & \text{für } nodes_{max} \geq 24 \end{cases}$$

Die Anzahl der Top-Knoten ergibt sich hieraus als die verbleibende Menge der Knoten:

$$nodes_{top} = \frac{nodes_{max}}{nodes_{neighbours} + 1}$$

Es wird somit sichergestellt, dass immer mindestens 8 Top-Knoten angezeigt werden. Bei kleiner Anzeigefläche werden nur die wichtigen Top-Knoten angezeigt und keine Nachbarn, wogegen bei größere Anzeigefläche die Anzahl der Nachbarknoten steigt.

Die Ausgangsgröße für den Knotenabstand (Linklänge) l bewegt sich zwischen einer oberen l_{max} und unteren l_{min} Grenze. Dazwischen wird die Linklänge über die freie Fläche im Graphen berechnet. Hierfür wird von der verfügbaren Gesamtfläche a_{graph} die benötigte Fläche für die Anzahl der tatsächlich im Graphen vorhandenen Knoten $nodes_{in\_graph}$ zzgl. deren Abstandsfläche abgezogen. Anschließend wird die verbleibende Fläche in gleichgroße Quadrate für jeden Knoten eingeteilt. Die Ausgangsgröße für die Linklänge ist die Kantenlänge dieser Quadrate:

$$l = \max\left(l_{min}, \min\left(l_{max}, \sqrt{\frac{a_{graph} - nodes_{in\_graph} \cdot a_{node} \cdot s}{nodes_{in_graph}}}\right)\right)$$

Die tatsächliche Linklänge zwischen zwei Knoten a und b wird ausgehend von diesem Ausgangswert über WebCola berechnet. Hierfür verwendet die Bibliothek die Differenz zwischen der Anzahl der Nachbarknoten der beiden Knoten a und b und der Anzahl gemeinsamer Nachbarknoten:

$$symDiff = |neighbours_a \cap neighbours_b| - |neighbours_a \cup neighbours_b|$$

Hierdurch wird für Knoten mit vielen Nachbarn eine größere Linklänge verwendet. Somit wird diesen Knoten mehr Platz zur Verfügung gestellt.

7 Evaluation

Die Forschungsfrage, ob sich das Textverständnis eines Artikels auf einer Webseite verbessern lässt, wenn dem Leser parallel zur Webseite eine Graphen-basierte Visualisierung der Seiteninhalte sowie Informationen aus anderen bisher gelesenen Seiten zur Verfügung stehen, wurde im Rahmen einer Studie untersucht. Konkret wurde hierfür das in den vorherigen Kapiteln vorgestellte Browserplugin *Storyfinder* bei einer Between-Subjects Studie verwendet. In Unterkapitel 7.1 wird der Aufbau und Ablauf der Studie im Detail erläutert. Anschließend werden in Unterkapitel 7.2 und 7.3 die Evaluationsgruppen und das Evaluationskorpus vorgestellt. In Unterkapitel 7.5 werden die Ergebnisse der Studie in einer Gruppen-vergleichenden Analyse dargestellt und interpretiert. Abschließend werden in Unterkapitel 7.6 die Ergebnisse zusammengefasst und die Forschungsfrage beantwortet.

7.1 Aufbau der Studie

In einer Studie mit insgesamt zwölf Teilnehmern wurde das Textverständnis mit und ohne das *Storyfinder-Plugin* in einer Versuchs- und einer Kontrollgruppe geprüft. Anschließend wurden die Ergebnisse der beiden Gruppen verglichen, um hieraus mögliche Verbesserungen im Textverständnis durch das *Storyfinder-Plugin* zu erkennen.

7.1.1 Forschungsdesign

Als Designprinzip für die Evaluation stand die Between-Subjects oder die Within-Subjects Methode zur Auswahl [58]. Bei der **Within-Subjects** Evaluation löst jeder Proband jeweils mehrere unterschiedliche Aufgaben, also in diesem Fall die Beantwortung eines Fragebogens mit sowie eines Fragebogens ohne das *Storyfinder-Plugin*. Die persönlichen Leistungen und die Fähigkeiten der einzelnen Teilnehmer spielen bei dieser Evaluationsmethode nahezu keine Rolle, da jeder Teilnehmer beide Experimente durchläuft. Es können in diesem Fall jedoch unerwünschte Lerneffekte sowie Konzentrationsschwächen auftreten, die abhängig von der Reihenfolge, in der die Experimente durchgeführt werden, die Ergebnisse verfälschen. Zudem ist es bei dieser Methode äußerst schwierig, vergleichbare Themen und Fragebögen und somit am Ende ein homogenes und aussagekräftiges Ergebnis zu erhalten.
Für die Studie wurde deshalb das **Between-Subjects** Design gewählt. Bei dieser Methode werden die Studienteilnehmer in zwei möglichst homogene Gruppen eingeteilt, wobei jede Gruppe nur ein Experiment durchführt, also entweder die Beantwortung des Fragebogens mit dem *Storyfinder-Plugin* (Versuchsgruppe) oder die Beantwortung ohne das Plugin (Kontrollgruppe).

7.1.2 Fragebogen zur Erfassung allgemeiner Angaben

Zur Feststellung der Vorkenntnisse und zur Erfassung allgemeiner Angaben wie Alter, Geschlecht, Beruf und Schulabschluss wurde von allen Teilnehmern vor dem Experiment ein Fragebogen (siehe Anhang B.1) ausgefüllt. Der Fragebogen umfasst die folgenden drei Kategorien:

- Allgemeine Angaben zur Person: In den allgemeinen Angaben zur Person wurden Alter, Geschlecht und Wohnort erfasst.
- Schulische und berufliche Ausbildung: Die Angaben zur schulischen und beruflichen Ausbildung erfassten die Art des Schulabschlusses sowie ggf. des Hochschulabschlusses, die berufliche Ausbildung sowie den aktuell ausgeübten Beruf.
- Nutzungsverhalten im Internet: In der dritten Kategorie zum Thema Nutzungsverhalten im Internet wurde zuerst die Häufigkeit und Art der Internetnutzung (privat, Schule/Studium, beruflich) sowie die drei häufigsten besuchten Webseiten, abgesehen von Suchmaschinen, erfasst. Anschließend wurden gezielt Informationen zum Verhalten bei der Recherche und beim Lesen von Nachrichten ermittelt. Hierzu wurden die verwendeten Hilfsmittel, wie beispielsweise Lesezeichen, RSS Reader oder handschriftliche Notizen sowie das eigentliche Leseverhalten abgefragt.

Zeit	Schritt	Dauer
0:00 - 0:05	Cover Story / Erläuterung Evaluationsziel, Aufgaben und Ablauf des Experiments	∼ 5 Min
0:05 - 0:10	Einführung in die Bedienung des Plugins oder die Verwendung von Lesezeichen	∼ 5 Min
0:10 - 0:15	**Einführungsphase:** Einarbeitung in System und Arbeitsumgebung	5 Min
0:15 - 0:18	Präsentation der Fragen	3 Min
0:18 - 0:20	Pause und Zurücksetzen des Browsers	2 Min
0:20 - 0:35	**Trainingsphase:** Einarbeitung in Evaluationskorpus	15 Min
0:35 - 1:05	**Testphase:** Beantwortung der Evaluationsfragen	≤ 30 Min
1:05 - 1:10	Aufklärung der Cover Story	∼ 5 Min
	Zeit für Rückmeldung und Fragen der Teilnehmer	
Gesamt: 1:10 Std.		

Tabelle 7.1.: Zeitlicher Ablauf des Experimentes

7.1.3 Experiment

Die Durchführung des Experiments erfolgte mit jedem Studienteilnehmer jeweils einzeln an einem Büro-Arbeitsplatz an einem Desktop PC. Als Browser wurde *Mozilla Firefox* Version 49.0 verwendet, je nach Versuchsgruppe mit oder ohne das *Storyfinder-Plugin*. Die Teilnehmer hatten während des Experiments keinen Zugriff auf das Internet. Als Informationsquelle zur Beantwortung der Fragen wurde ein vorgefertigtes Korpus aus zwölf lokal gespeicherten Webseiten (siehe Unterkapitel 7.3) zum Thema „Kommunalwahlen 2016 in Delmenhorst" verwendet.
Zu diesem Thema wurden über einen Fragebogen 29 inhaltliche Fragen aus verschiedenen Kategorien gestellt, die jeweils in Form von Multiple Choice Fragen beantwortet werden mussten.

Cover Story

Um ein möglichst realistisches Benutzerverhalten bei der Evaluation hervorzurufen, wurde den Teilnehmern erst nach dem Experiment dessen eigentlicher Zweck mitgeteilt. Vor und während des Experiments wurde eine Cover Story verwendet, um das tatsächliche Motiv der Evaluation zu verschleiern.
Im Rahmen dieser Cover Story wurde den Nutzern mitgeteilt, dass das Ziel der Studie die Untersuchung des Rechercheverhaltens im Internet sei und hierfür unterschiedliche Hilfsmittel existieren. Hierfür wurden die Namen von fünf Programmen, u.a. *Storyfinder* sowie als sechste Option handschriftliche Notizen und Lesezeichen aufgezählt. *Storyfinder* wurde hierbei als frei verfügbares Programm eingeführt, ohne auf die Entwicklung im Rahmen der Masterarbeit hinzuweisen. Anschließend konnten die Benutzer würfeln, um hierüber das verwendete Hilfsmittel zu losen. Die tatsächliche Einteilung stand jedoch bereits vorab fest und war vom Würfelergebnis unabhängig.
Durch die Cover Story wurde sichergestellt, dass keine Verzerrung des Versuchsergebnisses durch sozial erwünschtes Verhalten erfolgt.

Ablauf des Experimentes

Zu Beginn des Experimentes wurde den Probanden im Rahmen der Cover Story der Zweck der Evaluation, der Aufbau und Ablauf des Experiments sowie die Aufgaben mitgeteilt. Nachdem die Benutzer zu einer der beiden Versuchsgruppen zugeordnet wurden, wurde in einer etwa fünf minütigen Einführung die Bedienung des Plugins in der Versuchsgruppe bzw. die Verwendung von Lesezeichen und handschriftlichen Notizen in der Kontrollgruppe erläutert. In den folgenden fünf Minuten hatten die Teilnehmer Zeit, sich mit dem System und der Arbeitsumgebung vertraut zu machen. Hierfür war in dieser Zeit der Zugriff auf die Internetseite tagesschau.de möglich und es bestand die Möglichkeit, Fragen zur Bedienung zu stellen.

Da bei einer Recherche davon ausgegangen werden kann, dass dieser Recherche ein Ziel bzw. eine Fragestellung zugrunde liegt, wurden den Teilnehmern anschließend die zu beantwortenden Fragen präsentiert. Die Fragen wurden ohne die möglichen Antworten, nacheinander auf einem Tablet, für jeweils 5 Sekunden angezeigt. Die Reihenfolge, in der die Fragen angezeigt wurden, wurde jeweils zufällig gewählt. Die kurze Zeit war ausreichend, um die Fragen zu lesen, jedoch war es für die Probanden kaum möglich, alle Fragen im Kopf zu behalten.
Ziel hiervon war es, dass bei der anschließenden Einarbeitung in das Textkorpus bereits nach Informationen recherchiert werden konnte, ohne jedoch gezielt nach den Antworten der einzelnen Fragen zu suchen. Dies war auch erforderlich, um eine Protokollierung der Nutzeraktionen während der eigentlichen Beantwortung der Fragen zu ermöglichen.

Anschließend hatten die Teilnehmer eine kurze Pause von zwei Minuten. In dieser Zeit wurde der Browser und das *Storyfinder-Plugin* zurückgesetzt, so dass alle Lesezeichen und Plugin-Daten aus der Einführungsphase gelöscht wurden. Nach dieser Einführungsphase erhielten die Teilnehmer Zugriff auf das aus zwölf Artikeln bestehende Evaluationskor-

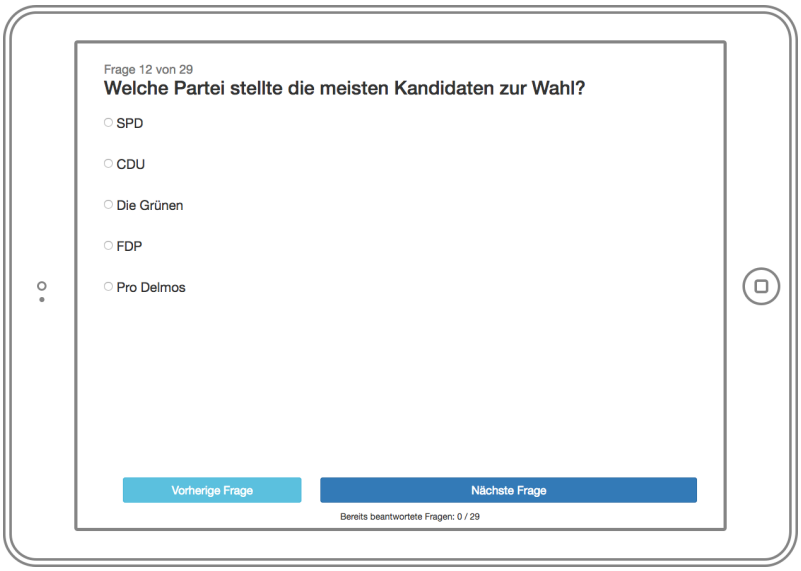


Abbildung 7.1.: Zur Erfassung der Antwortzeit und des Benutzerverhaltens erfolgte die Beantwortung der Multiple Choice Fragen auf einem Tablet.

pus und die Aufgabe, sich für 15 Minuten mit den Inhalten der darin enthaltenen Texte vertraut zu machen. In dieser Trainingsphase wurde bei der Versuchsgruppe das Plugin mit Inhalten aus dem Evaluationskorpus automatisch beim Lesen der Artikel trainiert. Zusätzlich hatten die Testpersonen die Möglichkeit, manuell Entitäten und Beziehungen hinzuzufügen. Die Kontrollgruppe konnte in der Trainingsphase handschriftliche Notizen anfertigen sowie Lesezeichen für voraussichtlich relevante Artikel setzen.
Nach der Trainingsphase folgte die Testphase, in der verschiedene Fragen zu den Texten aus dem Evaluationskorpus beantwortet werden mussten. Die Teilnehmer hatten während dieser Phase weiterhin Zugriff auf das Evaluationskorpus und die zuvor angefertigten Notizen bzw. das *Storyfinder-Plugin*. Für die Beantwortung standen maximal 30 Minuten zur Verfügung. Die Teilnehmer hatten aber auch die Möglichkeit, die Testphase schneller abzuschließen.
Nach der Testphase wurde die Cover Story aufgeklärt. Zudem bestand die Möglichkeit, Fragen zum Experiment zu stellen und Rückmeldung zu geben.
Der zeitliche Ablauf ist in Tabelle 7.1 auf Seite 64 nochmals aufgeführt.

Testphase

Die Fragen der Testphase wurden auf einem Tablet angezeigt und auch dort beantwortet (siehe Abb. 7.1). Es war hierbei jeweils nur eine Frage sichtbar, wobei die Nutzer jederzeit die Möglichkeit hatten, zu einer anderen Frage zu wechseln. Hierdurch war es möglich, die ausgeführten Aktionen im Browser und *Storyfinder* während der Anzeige einer bestimmten Frage aufzuzeichnen. Zudem konnte hierüber die Zeit erfasst werden, die für die Beantwortung einer Frage benötigt wurde. Hierfür wurden während der Testphase verschiedene Aktionen im Browser, im *Storyfinder-Plugin* und auf dem Tablet aufgezeichnet:

- Tablet: Aufruf einer Frage.
- Tablet: Beantwortung einer Frage.
- Browser: Aufruf einer Seite.
- Browser: Schließen/Verlassen einer Seite.
- Storyfinder: Aufruf eines Wissensgraphen.
- Storyfinder: Aufruf von Details einzelner Entitäten.
- Storyfinder: Eingabe von Suchbegriffen.

Die Aufzeichnung der Aktionen erfolgte über ein im Rahmen der Masterarbeit entwickeltes Browser-Plugin (siehe Unterkapitel 7.1.5), das die entsprechenden Aktionen in einer Protokolldatei protokollierte.
Durch diese Daten war es bei der Auswertung möglich, die Aktionen und Aufrufe der Beantwortung einer bestimmten Frage zuzuordnen und diese Informationen in die Auswertung einfließen zu lassen.

7.1.4 Vortest

Vor dem eigentlichen Experiment wurde ein Vortest (Pretest) [39] durchgeführt, um etwaige Probleme mit dem Aufbau des Experiments, dem allgemeinen Fragebogen oder den Multiple Choice Fragen festzustellen. Hierfür wurde das Experiment unter Verwendung des *Storyfinder-Plugin* mit einem zusätzlichen Probanden unter Experiment-Bedingungen durchgeführt.
Beim Vortest wurden zwei Probleme festgestellt: Zum einen war die ursprüngliche Fragestellung nach dem „höchsten Bildungsabschluss" im allgemeinen Fragebogen missverständlich, da hierzu auch Hochschulabschlüsse zählen. Die Fragestellung wurde daraufhin geändert, so dass nach dem „höchsten Schulabschluss" gefragt wurde. Zum anderen hat sich herausgestellt, dass in der Trainingsphase zu viel und in der Testphase zu wenig Zeit zur Verfügung stand. Ursprünglich waren 20 Minuten für die Trainingsphase und 25 Minuten für die Testphase vorgesehen. Die Zeiten wurden daraufhin angepasst, so dass für die Trainingsphase nur noch 15 Minuten zur Verfügung standen und für die Testphase 30 Minuten. Auf Grund der Anpassungen, insbesondere im Zeitplan, wurde ein erneuter Vortest mit einem weiteren Probanden durchgeführt. Hierbei hat sich gezeigt, dass der neue Zeitplan besser geeignet war. Da auch sonst keine Probleme aufgetreten sind, wurde das Experiment mit diesen angepassten Bedingungen durchgeführt.

7.1.5 Technische Details zu den verwendeten Hilfsmitteln

Während des Experiments wurden mehrere technische Hilfsmittel eingesetzt, die im Rahmen dieser Arbeit entwickelt wurden[1].

Fragebogen für Multiple Choice Fragen

Der digitale Fragebogen zur Beantwortung der Multiple Choice Fragen wurde als HTML Seite unter Verwendung von Javascript und dem Bootstrap CSS Framework erstellt. Alle Fragen waren auf einer gemeinsamen HTML Seite enthalten. Über Javascript wurde jeweils die gewünschte Frage ausgewählt und angezeigt, so dass keine spürbare Verzögerung beim Wechsel zwischen den einzelnen Fragen auftrat.
Bei der Auswahl einer Antwort wurden die Antworten des gesamten Fragebogens im Hintergrund per HTTP an einen lokalen Webserver gesendet. Dort wurden die Daten über ein PHP-Script verarbeitet und im JSON-Format lokal abgespeichert. Zusätzlich wurde beim Wechsel der angezeigten Frage oder bei der Auswahl einer Antwort ebenfalls per HTTP eine entsprechende Nachricht an den lokalen Webserver gesendet, der wiederum über ein PHP-Script diese Nachricht entgegen nahm. Diese Nachricht wurde mit einem Zeitstempel versehen und an eine Protokolldatei angehängt.

Fragenrotation

Vor Beginn der Trainingsphase wurde den Probanden jede Frage für fünf Sekunden auf einem Tablet gezeigt. Die Anzeige erfolgte, wie beim Fragebogen, über eine HTML Seite. Alle Fragen waren auf dieser Seite enthalten und wurden bei jedem Aufruf der Seite zufällig sortiert. Über Javascript wurde anschließend jede Frage für 5 Sekunden angezeigt und danach zur nächsten Frage gewechselt.

Protokollierung des Nutzerverhaltens

Zur Protokollierung des Nutzerverhaltens im Browser wurde ein Browserplugin für *Mozilla Firefox* entwickelt. Das Plugin reagiert auf verschiedene Browserevents, wie das Auswählen eines Tabs oder das Öffnen einer Seite. Bei diesen Events wird eine Protokollnachricht an den lokalen Webserver gesendet. Dieser versieht, wie bereits bei der Protokollierung der aufgerufenen Fragen, jede Nachricht mit einen Zeitstempel und hängt die Nachricht an eine Protokolldatei an.

Protokollierung der Interaktion mit Storyfinder

Die Protokollierung der im *Storyfinder-Plugin* ausgeführten Aktionen, erfolgte direkt über eine angepasste Version des *Storyfinder* Servers. Der *Storyfinder* Server legte hierfür alle relevanten Resourcenzugriffe in einer lokalen Protokolldatei ab.

7.2 Zusammensetzung der Evaluationsgruppen

Im Folgenden wird die Auswahl der Studienteilnehmer sowie die Zusammensetzung der beiden Evaluationsgruppen beschrieben. An der Evaluation haben insgesamt zwölf Teilnehmer zwischen 20 und 49 Jahren teilgenommen, die jeweils die folgenden Kriterien erfüllen mussten:

[1] Quellcode und Plugin: https://github.com/mt-k/storyfinder-eval

- Die Studienteilnehmer sind mit der Nutzung des Internets und des Webbrowsers *Mozilla Firefox* vertraut.
- Die Studienteilnehmer nutzen das Internet regelmäßig zum Lesen von Nachrichten und/oder zur Recherche.
- Den Studienteilnehmern ist die Region Delmenhorst nahezu unbekannt und sie verfügen über keinerlei tiefer gehendes Wissen über die dortige Kommunalpolitik.

Für die Zuordnung der Probanden zur Versuchs- oder Kontrollgruppe waren die Antworten aus dem allgemeinen Fragebogen ausschlaggebend. Hierüber wurde versucht, eine möglichst homogene Verteilung zu erzielen. Von den zwölf Teilnehmern waren insgesamt fünf weiblich und sieben männlich. Die Kontrollgruppe setzte sich aus zwei weiblichen und vier männlichen Probanden zusammen, die Versuchsgruppe aus drei weiblichen und drei männlichen Probanden. Das Durchschnittsalter der Kontrollgruppe betrug 30 Jahre. In der Versuchsgruppe lag das Durchschnittsalter etwas höher bei 36 Jahren. Beim höchsten Schulabschluss waren in der Versuchsgruppe drei Teilnehmer mit Abitur oder einem vergleichbaren Abschluss sowie drei Teilnehmer mit mittlerer Reife. Die Versuchsgruppe umfasste ebenfalls drei Probanden mit Abitur, sowie zwei Teilnehmer mit mittlerer Reife und ein Teilnehmer mit einem qualifizierten Hauptschulabschluss. Als letztes Kriterium war der Berufsabschluss relevant, hier war in der Kontrollgruppe ein Proband mit einem Hochschulabschluss, vier Probanden mit einer abgeschlossenen Berufsausbildung sowie ein Proband in der Ausbildung. Bei der Versuchsgruppe hatten drei Probanden einen Hochschulabschluss sowie die restlichen drei Teilnehmer eine abgeschlossene Berufsausbildung.

7.3 Vorstellung des Evaluationskorpus

Das Evaluationskorpus bestand aus zwölf Artikeln zum Thema „Stadtratswahl in Delmenhorst (Niedersachsen)" die am 14. September 2016 stattgefunden hat. Da alle Studienteilnehmer im süddeutschen Raum leben, wurde gezielt ein regionales Thema aus einer entfernten Region gewählt. Hierdurch wurde ein Vorwissen der Teilnehmer zu diesem Thema und somit eine Verfälschung der Ergebnisse ausgeschlossen.
Bei den zwölf Artikeln handelt es sich um Vorberichte zur Wahl, Berichte zu Wahlveranstaltungen oder den Kandidaten sowie um Berichte zum Wahlausgang und dem Wahlverlauf. Quelle für die Artikel waren die folgenden fünf Webseiten: weser-kurier.de, noz.de, delmenews.de, deniz-kurku.de und weser-report.de.
Die durchschnittliche Länge der Artikel beträgt 3113 Zeichen in 439 Wörtern. Die Länge der Artikel variiert zwischen 127 und 871 Wörtern. In Anhang B.2 findet sich eine detaillierte Auflistung der Artikel mit den zugehörigen URLs.

Vorbereitung der Artikel

Um die Verwendung weiterer Quellen auszuschließen, war während des Experiments kein Zugriff auf das Internet möglich. Aus diesem Grund wurden alle zwölf Webseiten über die Speichern-Funktion in *Mozilla Firefox* lokal gespeichert. Aus diesen Webseiten wurden anschließend alle Elemente entfernt, die sich über den eigentlichen Seiteninhalt legen. Zu diesen Elementen zählen u.a. bestimmte Werbung und Hinweise zur Nutzung von Cookies. Es wurden zudem alle Links deaktiviert und mit einem entsprechenden Hinweis versehen, der beim Anklicken erscheint.
Anschließend wurde eine Übersichtsseite für alle Artikel erstellt (siehe S. 68, Abb. 7.2), auf der die Seitentitel der Webseiten aufgeführt sind. Diese Seite verlinkt auf die jeweiligen Artikel. Um eine einfache Navigation zwischen den Artikeln zu ermöglichen, wurde die Übersichtsseite als Startseite im Browser eingestellt und die Probanden darauf hingewiesen, dass sie über den Home-Button im Browser jederzeit zu dieser Seite gelangen.

7.4 Fragen und deren Einteilung in Fragetypen

Die Probanden mussten in der Testphase insgesamt 29 Fragen beantworten. Alle Antworten auf diese Fragen waren in den zwölf Artikeln des Evaluationskorpus enthalten.
Die Fragen wurden in Form von Multiple Choice Fragen beantwortet, wobei 28 der 29 Fragen nur die Auswahl einer Antwort ermöglichten. Nur bei einer Frage war nach mehreren Antworten gefragt. Es wurden jeweils fünf Antwortmöglichkeiten dargestellt, mit Ausnahme einer Frage, bei der nur die Auswahl von „ja" oder „nein" möglich war.
Die Fragen lassen sich grob den folgenden sechs Fragetypen zuordnen:

- Frage nach einer Person (4 Fragen)
- Frage nach einer Begriffserklärung (2 Fragen)
- Frage nach einer Zahl oder Partei, die als wichtige Fakten im Text angegeben waren (9 Fragen)
- Frage nach einer Zahl, die nur beiläufig im Text erwähnt wurde (5 Fragen)

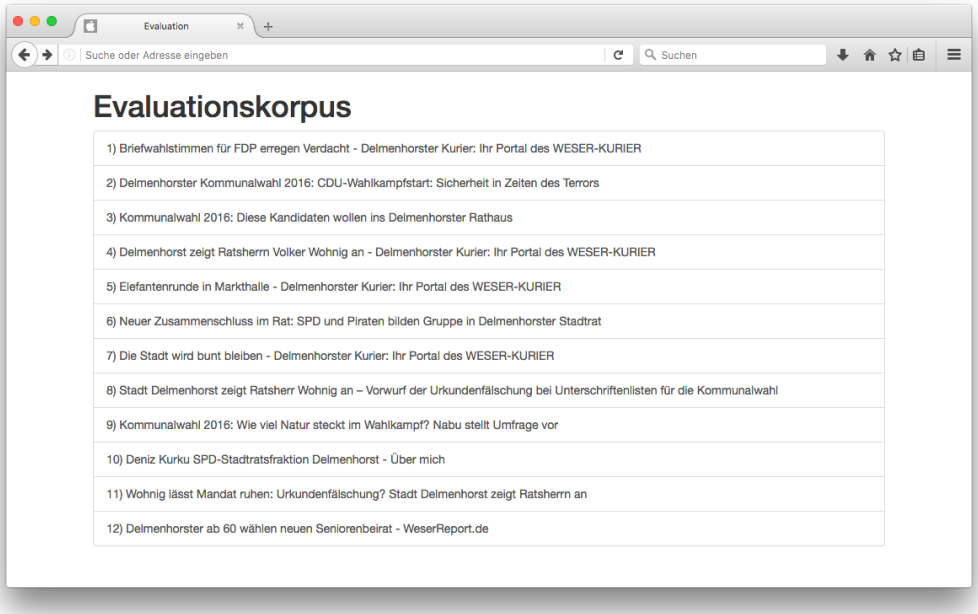


Abbildung 7.2.: Übersichtsseite für die Navigation zwischen den Artikeln im Evaluationskorpus

	Versuchsgruppe				Kontrollgruppe			
	Gesamt	**∅**	**Varianz**	**m.F.**	**Gesamt**	**∅**	**Varianz**	**m.F.**
Beantwortete Fragen	139	23,17	3,29	0,14	156	26	2,16	0,08
...davon korrekt beantwortet	134	22,33	3,86	0,17	135	22,5	3,91	0,17
...davon falsch beantwortet	5	0,83	0,68	0,81	21	3,5	1,7	0,48
Unbeantwortete Fragen	35	5,83	3,29	0,56	18	3	2,16	0,72
Genauigkeit		96,4 %	0,04	0,05		86,5 %	0,09	0,11

Tabelle 7.2.: Ergebnisse der Beantwortung der Multiple Choice Fragen. In der Tabelle ist je Gruppe jeweils die Gesamtmenge, der Mittelwert, die unkorrigierte Stichprobenvarianz sowie die Standardabweichung angegeben.

- Frage nach Informationen, die keine Entität darstellen (5 Fragen)
- Frage nach Informationen, die über mehrere Sätze verteilt sind (4 Fragen)

Eine detaillierte Auflistung der Fragen findet sich in Anhang B.3.

7.5 Darstellung und Interpretation der Ergebnisse

Im Folgenden werden die Ergebnisse des Experimentes dargestellt und interpretiert. Hierbei werden sowohl die Antworten der Multiple Choice Fragen als auch die Seitenaufrufe der Benutzer während der Beantwortung analysiert. Auf Grund des Between-Subjects Design wird für die Auswertung ein Gruppen-vergleichendes Analyseverfahren verwendet.

7.5.1 Ergebnisse der Multiple Choice Fragen

Die Beantwortung der 29 Multiple Choice Fragen stellt das wichtigste Kriterium zur Beurteilung des Textverständnisses dar. In Tabelle 7.2 sind die Ergebnisse aus der Beantwortung der Fragen aufgelistet. Da 28 der 29 Fragen jeweils von der Mehrheit der Teilnehmer korrekt beantwortet wurden, lässt sich im Allgemeinen feststellen, dass die Fragen klar gestellt und die richtige Antwort eindeutig zu erkennen war.

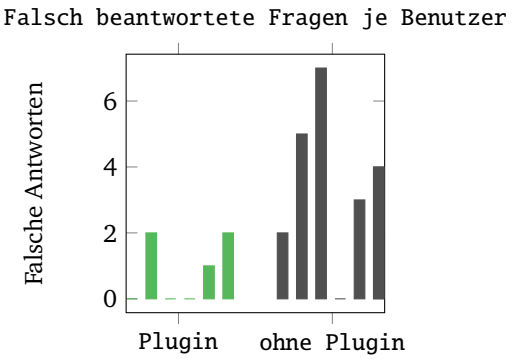


Abbildung 7.3.: Vergleich der Anzahl an falsch beantworteten Fragen in den beiden Gruppen

Korrekte Antworten

Die Kontrollgruppe hat im Schnitt 26 der 29 Fragen beantwortet, wogegen die Probanden in der Versuchsgruppe durchschnittlich nur etwa 23 Fragen, also drei Fragen weniger beantwortet haben. Bei der Anzahl der korrekt beantworteten Fragen gibt es hingegen nahezu keinen Unterschied. Sowohl die Kontroll- als auch die Versuchsgruppe haben im Mittel etwa 22,5 Fragen richtig beantwortet. Aus der Standardabweichung sind deutliche, aber keine ungewöhnlich hohen Varianzen zwischen den einzelnen Gruppenmitgliedern erkennbar. Da die Standardabweichung in beiden Gruppen gleichermaßen bei 0,17 liegt, ist diese Varianz beim Vergleich der Ergebnisse irrelevant.
Betrachtet man alleine die korrekten Antworten, so lässt sich keine Veränderung durch das Plugin feststellen.

Genauigkeit

Bezieht man neben den korrekt beantworteten Fragen auch die Anzahl der falschen Antworten in die Analyse mit ein, so lässt sich erkennen, dass in der Kontrollgruppe durchschnittlich über vier Mal so viele Fragen falsch beantwortet wurden wie in der Versuchsgruppe. Jeder Proband in der Kontrollgruppe hat durchschnittlich 3,5 Fragen falsch beantwortet, wogegen in der Versuchsgruppe im Mittel nur 0,83 Fragen je Proband falsch beantwortet wurden. Dies zeigt sich auch in der Genauigkeit, die den Anteil der korrekt beantworteten Fragen unter allen beantworteten Fragen angibt:

$$Genauigkeit = \frac{\#korrekt}{\#beantwortet}$$

Die Genauigkeit mit Plugin in der Versuchsgruppe lag bei 96,4% und somit fast zehn Prozentpunkte höher als in der Kontrollgruppe mit 86,5%.

Es ist jedoch zu beachten, dass die Standardabweichung bei der Anzahl der falschen Antworten in der Kontrollgruppe bei sehr hohen 0.43 liegt. Hier ist deshalb eine Betrachtung der Einzelergebnisse in beiden Gruppen sinnvoll. In Abb. 7.3 ist die Anzahl der falschen Antworten je Teilnehmer dargestellt. Hieraus lässt sich erkennen, dass fünf der sechs Teilnehmer aus der Kontrollgruppe mehrere Fragen falsch beantwortet haben. Nur ein Proband in der Kontrollgruppe hat keine Frage falsch beantwortet, einmal wurden zwei falsche Antworten gegeben und die restlichen vier Teilnehmer haben auf drei bis max. sieben Fragen falsch geantwortet. Im Vergleich hierzu gab es in der Versuchsgruppe nur zwei Teilnehmer, die zwei Fragen falsch beantwortet haben, ein Teilnehmer mit einer falschen Antwort und drei Teilnehmer ohne falsche Antwort. Es zeigt sich also, dass trotz der hohen Standardabweichung durch die Probanden in der Kontrollgruppe im Allgemeinen deutlich mehr Fragen falsch beantwortet wurden.
Hieraus lässt sich schließen, dass die Genauigkeit und somit auch das Textverständnis unter Verwendung des *Storyfinder-Plugins* steigt.

7.5.2 Seitenaufrufe während der Beantwortung der Fragen

Als zweites Kriterium werden die getätigten Seitenaufrufe während der Beantwortung der Fragen betrachtet. Insgesamt wurden in der Versuchsgruppe 209 Seiten geöffnet. In der Kontrollgruppe wurden mit 268 Seiten etwa 30% mehr Fragen aufgerufen. Im Schnitt ergeben sich somit je Proband und Frage 1,20 Seitenaufrufe in der Versuchsgruppe und 1,54 Seitenaufrufe in der Kontrollgruppe. In Abb. 7.4 auf Seite 70 sind die durchschnittlichen Seitenaufrufe je Frage aufgeführt. Wie sich aus der Abbildung sehr deutlich erkennen lässt, gibt es starke Schwankungen zwischen den einzelnen Fragen und Versuchsgruppen. Es gibt sowohl Fragen, bei denen die Kontrollgruppe signifikant mehr Seitenaufrufe benötigte wie die Versuchsgruppe (Frage 1, 3, 10, 11, 12, 16 und 24) sowie auch den umgekehrten Fall, bei dem die Kontrollgruppe erheblich mehr Seitenaufrufe verzeichnete (Frage 5 und 23).

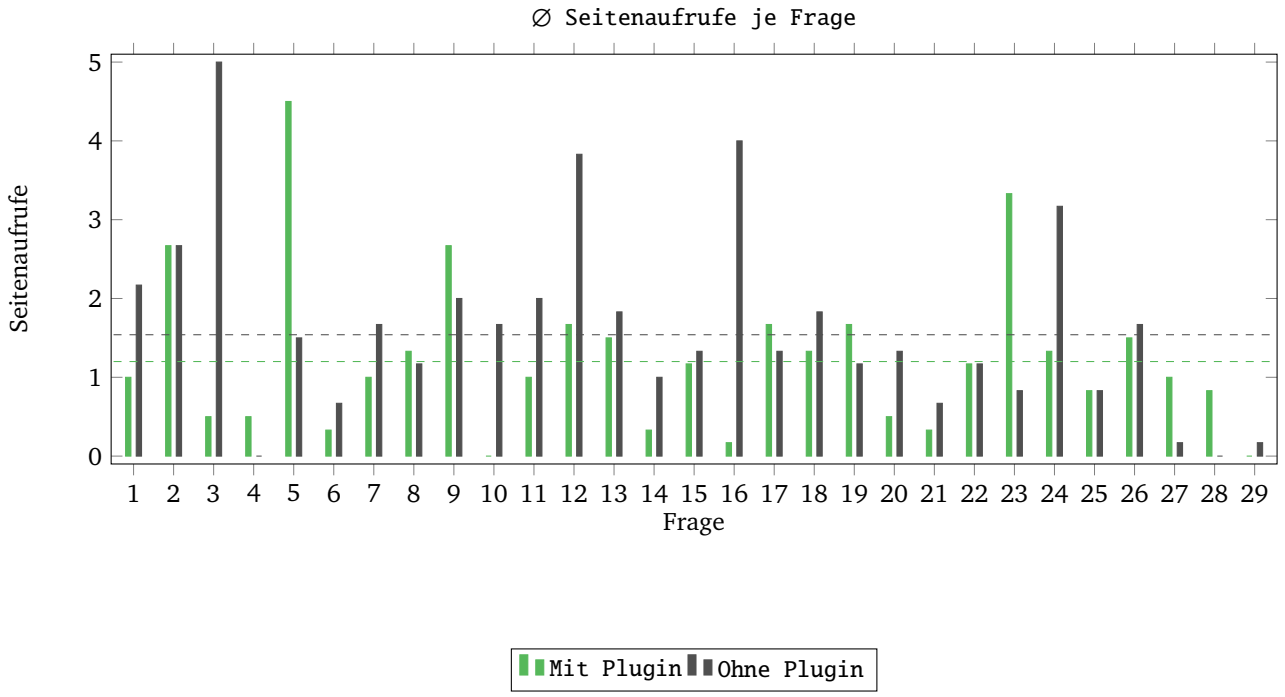


Abbildung 7.4.: Durchschnittliche Seitenaufrufe zur Beantwortung der Fragen. Die gestrichelten horizontalen Linien zeigen den jeweiligen Mittelwert der beiden Gruppen an.

Fragen am Ende des Fragebogens

Bei Frage 27 und 28 zeigen sich zwar grundsätzlich auch deutliche Schwankungen zwischen den Gruppen, jedoch konnten die Fragen in beiden Gruppen mit unterdurchschnittlich wenigen Seitenaufrufen beantwortet werden. Dies lässt sich eher dadurch erklären, dass die Fragen am Ende des Fragebogens standen, als anhand von Unterschieden zwischen den Gruppen. Durch die Position der Fragen wurden bereits mehrere andere Fragen zu denselben Artikeln gestellt und diese beiden Fragen konnten, ebenso wie Frage 29, überwiegend beantwortet werden, ohne dass hierfür erneut eine Seite aufgerufen werden musste. Die beiden Fragen werden deshalb nicht im Detail betrachtet.

Fragen mit Ausreißern

Betrachtet man bei den anderen neun Fragen mit großen Differenzen zwischen den beiden Gruppen, die Seitenaufrufe je Benutzer (siehe S. 71, Abb. 7.5), so lässt sich feststellen, dass es einige Fragen gibt, bei denen nur einzelne Probanden signifikant mehr Seitenaufrufe benötigt haben. Konkret lässt sich dies bei den Fragen 10, 12, 16 und 24 beobachten. Diese vier Fragen sind somit für den Vergleich zwischen den beiden Gruppen ungeeignet.

Fragen, bei denen die Kontrollgruppe signifikant mehr Seitenaufrufe benötigte

Bei den Fragen 1, 3 und 11 ist zu erkennen, dass nahezu alle Probanden der Kontrollgruppe im Vergleich zur Versuchsgruppe mehr Seiten aufrufen mussten, um die jeweilige Frage zu beantworten. Die drei zugehörigen Fragen lauteten:

- Frage 1) Wie heißt der Fraktionschef der CDU?
- Frage 3) Wie heißt die amtierende Bürgermeisterin?
- Frage 11) Um welchen Faktor möchte Karl-Heinz Bley die Abschieberate von Flüchtlingen verändern?

Die Antworten zu Frage 1 und 11 standen beide in einem Artikel zum Wahlkampfstart der CDU. Aus dem Seitentitel „Delmenhorster Kommunalwahl 2016: CDU-Wahlkampfstart: Sicherheit in Zeiten des Terrors" lässt sich bereits auf der Übersichtseite erkennen, dass die gesuchte Antwort zur Frage nach dem Fraktionschef der CDU in diesem Artikel enthalten sein könnte. Es handelt sich jedoch mit 628 Wörtern um den zweit längsten Artikel im Korpus. Die Artikellänge liegt somit um 200 Wörter über der durchschnittlichen Textlänge im Korpus. Zudem enthält der Text sehr viele verschiedene Informationen zu unterschiedlichen Wahlkampfthemen. Der Text besteht aus sechs Abschnitten mit unterschiedlichen Themen, wie z.B. Sicherheit, Elektroautos, Brexit, Flüchtlingspolitik oder dem Straßenausbau. Die Information zum Fraktionschef und zur Abschieberate von Flüchtlingen kann hier leicht übersehen werden. Zudem kommt bei der Frage nach der Abschieberate noch die Schwierigkeit hinzu, den Namen Karl-Heinz Bley mit diesem Artikel zu verknüpfen und hierüber auf die Information in diesem Artikel zu schließen.

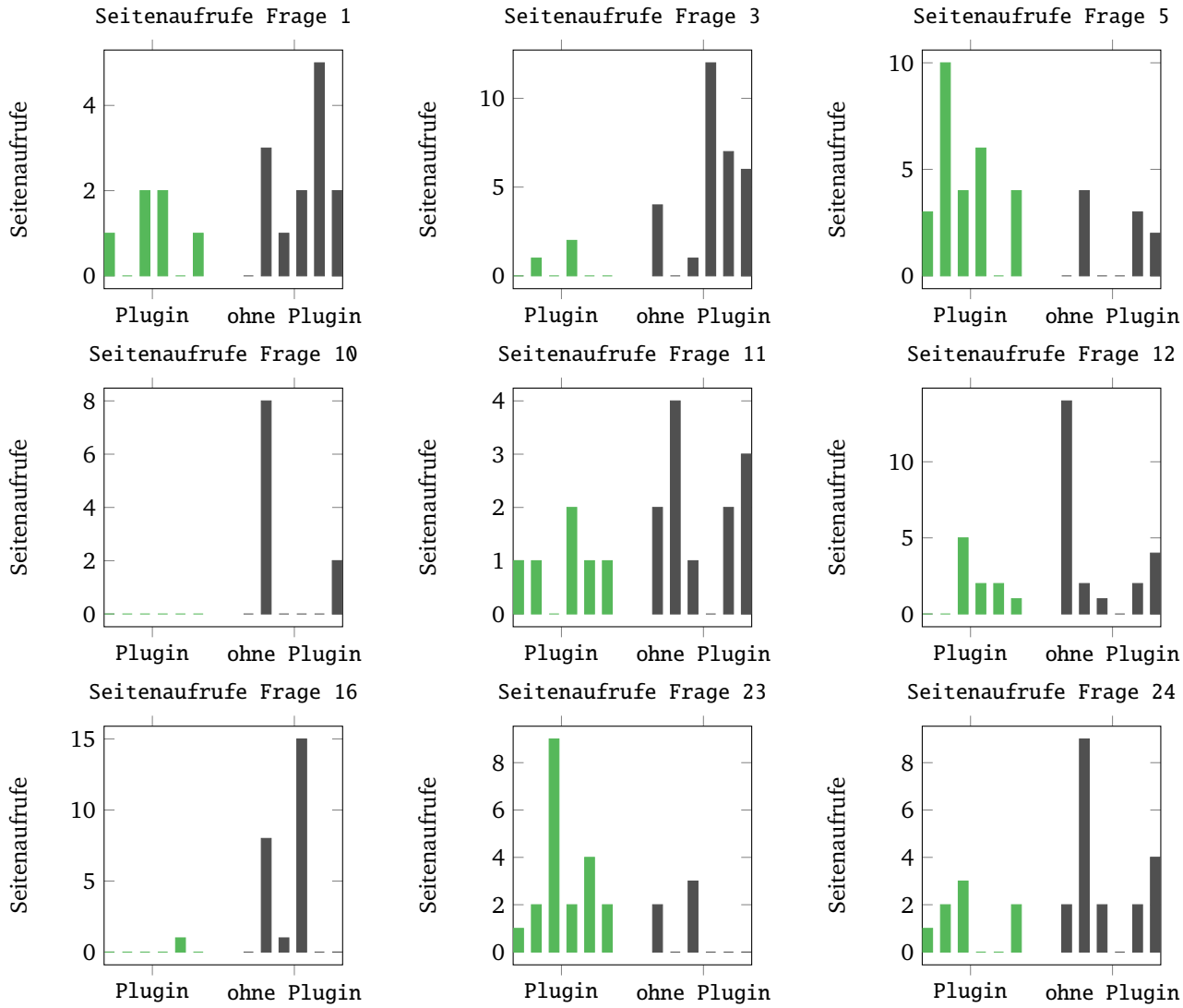


Abbildung 7.5.: Seitenaufrufe der einzelnen Probanden für die neun Fragen mit markanten Unterschieden zwischen den beiden Gruppen

Ein ähnliches Bild zeigt sich bei Frage 3. Die Antwort stand hier ebenfalls in einem vergleichsweise langen (538 Wörter) und unübersichtlichen Artikel, der auch verschiedene Wahlkampfthemen beinhaltete. Der Name der Bürgermeisterin wurde nur beiläufig in einem Absatz erwähnte. Zudem gab in diesem Fall auch der Titel „Elefantenrunde in Markthalle" keinen Aufschluss darüber, dass der Name in diesem Artikel enthalten sein könnte.
Hier zeigt sich, dass die Versuchsgruppe mit *Storyfinder-Plugin* bei diesen drei Fragen sehr stark von den Funktionen des Plugins profitieren konnte. Die Hervorhebung der Namen auf der Seite, die Zusammenfassung des Artikels im Wissensgraphen und die seitenübergreifende Suchfunktion haben mit hoher Wahrscheinlichkeit zu den deutlichen Unterschieden zwischen den Gruppen geführt.

Fragen mit deutlich mehr Seitenaufrufen in der Versuchsgruppe

Es gab jedoch auch die folgenden beiden Fragen, bei denen die Versuchsgruppe trotz Plugin mehr Seitenaufrufe benötigte:

- Frage 5) Wie viele Sitze gibt es im Stadtrat insgesamt?
- Frage 23) Wie viele Sitze hatte die Partei „Piratenpartei" in der Legislaturperiode 2011-2016?

Bei den Fragen fällt auf, dass hier jeweils nach einer Anzahl gefragt wurde. Zahlen, sowohl als Ziffer als auch als Wort, werden jedoch durch das *Storyfinder-Plugin* weder im Wissensgraph dargestellt noch im Artikeltext hervorgehoben.
Zudem ist in diesen Fällen auch die Suchfunktion des Plugins kaum hilfreich, da die Suche nach „Stadtrat" sechs Artikel und die Suche nach „Sitze" zehn Artikel als Treffer findet. Bei der Suche nach „Piratenpartei" werden zwar nur zwei

Artikel als Treffer gefunden, diese beiden Artikel enthalten jedoch nicht die gesuchte Information. Stattdessen steht die Antwort in einem mit 198 Wörtern vergleichsweise kurzen Artikel, in dem jedoch nur die Bezeichnung „Piraten" verwendet wird. Die Antwort ist zudem etwas versteckt im Text eingearbeitet:

> „Kurz nach der Kommunalwahl am 11. September hatte Neugebauer keinen Hehl über seine Enttäuschung zum Abschneiden der Piraten gemacht, die einen von zwei Sitzen verloren hatten."[2]

Die Schwierigkeit der Frage wird auch daran deutlich, dass sowohl in der Versuchs- als auch in der Kontrollgruppe jeweils nur zwei Probanden die Frage 23 korrekt beantworten konnten.
Es lässt sich hieraus allerdings erkennen, dass bei bestimmten Fakten, die vom Plugin nicht extrahiert werden, das Plugin keine Vorteile bietet.

Abschließende Bemerkung zur Auswertung der aufgerufenen Seiten

Der Vergleich der Seitenaufrufe ist nur dann aussagekräftig, wenn die jeweiligen Fragen auch in beiden Gruppen bearbeitet und korrekt beantwortet wurden. Aus diesem Grund wurden die Antworten aller fünf Fragen im Detail überprüft: In der jeweiligen Gruppe, die weniger Seitenaufrufe benötigt hatte, konnte die Frage von mindestens genauso vielen Probanden korrekt beantwortet werden, wie in der Gruppe mit mehr Seitenaufrufen. Von daher sind die getroffenen Aussagen und die daraus gezogenen Rückschlüsse bei diesen Fragen relevant.
Zu beachten ist jedoch, dass in der Protokolldatei nur Seitenaufrufe zur jeweiligen Frage verzeichnet wurden, wenn diese Frage aktuell angezeigt wurde. Antworten, die beiläufig bei der Suche nach Antworten zu anderen Fragen gefunden wurden, konnten hier weder erkannt noch berücksichtigt werden.

7.6 Fazit aus den Evaluationsergebnissen und Beantwortung der Forschungsfrage

In der Auswertung der Evaluationsergebnisse hat sich gezeigt, dass die Versuchsgruppe, die mit dem *Storyfinder-Plugin* gearbeitet hat, als auch die Kontrollgruppe, die nur mit Lesezeichen und handschriftlichen Notizen arbeitete, nahezu gleich viele Fragen korrekt beantwortet haben. Die Versuchsgruppe hat jedoch deutlich weniger Fragen falsch beantwortet und somit eine höhere Genauigkeit gezeigt. In der Betrachtung der Seitenaufrufe hat sich zudem gezeigt, dass besonders bei langen und unübersichtlichen Texten die Versuchsgruppe von den Funktionen des Plugins profitieren konnte. Es hat sich allerdings auch herausgestellt, dass bestimmte Fakten, wie beispielsweise Zahlen, von *Storyfinder* nicht erkannt werden und entsprechend für die Beantwortung dieser Fragen mehr Seitenaufrufe nötig sind.
Insgesamt hat sich dennoch gezeigt, dass sich das Textverständnis eines Artikels auf einer Webseite verbessern lässt, wenn dem Leser parallel zur Webseite eine Graphen-basierte Visualisierung der Seiteninhalte sowie Informationen aus anderen bisher gelesenen Seiten zur Verfügung stehen.

[2] http://www.noz.de/lokales-dk/delmenhorst/artikel/786864

8 Abschluss

In diesem Kapitel werden die Ergebnisse der Arbeit zusammengefasst und ein Fazit aus den Erkenntnissen gezogen. Der abschließende Ausblick stellt einige mögliche Themen für weitere Forschungen vor.

8.1 Zusammenfassung

In dieser Arbeit wurde untersucht, wie sich die Informationen auf einer Webseite erfassen und visualisieren lassen und ob eine solche Visualisierung zum Textverständnis beiträgt. Hierfür wurde das Browser Plugin *Storyfinder* entwickelt, welches das Textverständnis beim Nutzer verbessern soll. Es wurden hierfür die Schnittstellen der verschiedenen Browser verglichen und anschließend ein Plugin für den Browser *Mozilla Firefox* entwickelt. Das Plugin besteht aus mehreren server- und clientseitigen Komponenten, die Informationen aus Webseiten extrahieren, diese grafisch darstellen und verschiedene Recherchemöglichkeiten zur Verfügung stellen.
Zur Ermittlung der relevanten Informationen, wurde zuerst analysiert, welche Elemente einer Webseite geeignet sind und wie diese Elemente aus der Webseite extrahiert werden können. Als relevante Elemente wurden Schlüsselwörter und Eigennamen, sowie verschiedene Meta-Daten, wie Seitentitel oder Favoriten Icon, identifiziert. Da nicht alle Webseiten geeignete Artikel enthalten, wurde zudem eine Methode vorgestellt, um Seiten mit relevanten Artikeln zu identifizieren. Hierfür wurden Webseiten aus einem Korpus mit etwa 500 Webseiten manuell als geeignet oder ungeeignet gekennzeichnet und hieraus über Methoden des maschinellen Lernens ein Modell zur Klassifizierung der Seiten erlernt.
Anschließend wurde das in verschiedenen Browsern für den Lesemodus eingesetzte *Readability* vorgestellt und beschrieben, wie sich hierüber der Artikeltext vom restlichen Inhalt einer Webseite trennen lässt. Nachfolgend wurde erläutert, wie aus diesen Artikeln in *Storyfinder* Eigennamen mit Hilfe von *GermaNER*, Schlüsselwörter über den TFIDF-Wert von n-Grammen, sowie die Beziehungen zwischen diesen Elementen über Kookkurrenzen auf Satzebene extrahiert werden können.
Auf Grundlage dieser Daten wurde eine Visualisierung entwickelt, die diese Informationen in einem Knoten-Link-Diagramm darstellt. Hierbei wurde besonderer Wert auf ein einheitliches Bedienungskonzept sowie die Erzeugung eines guten mentalen Modells gelegt. Um dies zu erreichen, wurden verschiedene Konzepte, wie einheitliche Farbgestaltung oder fließende und somit nachvollziehbare Übergänge zwischen den verschiedenen Zuständen des Graphen verwendet. Es wurde zudem gezeigt, wie die Visualisierung und die Benutzeroberfläche an verschiedene Bildschirmgrößen angepasst wurde, damit eine Darstellung auf unterschiedlichen Geräten möglich ist.
Zur Evaluation der Forschungsfrage, ob ein Verbesserung des Textverständnisses auf Webseiten mit einem Plugin wie *Storyfinder* möglich ist, wurde eine Studie mit zwölf Teilnehmern durchgeführt. Diese Teilnehmer wurden in eine Versuchs- und eine Kontrollgruppe aufgeteilt, die mit bzw. ohne das Plugin Fragen zu vorgefertigten Texten beantworten mussten. Bei einer gruppenvergleichenden Analyse wurde festgestellt, dass die Genauigkeit bei der Beantwortung unter Verwendung des Plugins steigt. Zudem hat sich gezeigt, dass besonders bei langen und unübersichtlichen Texten mit dem Plugin weniger Seitenaufrufe benötigt werden. In der Studie ist allerdings auch aufgefallen, dass bestimmte Informationen nicht vom Plugin erfasst werden können und somit in diesen Fällen das Plugin nicht hilfreich ist.
Insgesamt hat sich gezeigt, dass es wichtige Elemente wie Schlüsselwörter und Eigennamen gibt, die aus einer Webseite innerhalb kurzer Zeit extrahiert werden können. Diese Daten tragen durch eine entsprechende Visualisierung und Recherchefunktionen zu einem besseren Textverständnis bei.

8.2 Ausblick

Bei der Entwicklung von *Storyfinder* und der anschließenden Evaluation sind verschiedene Themen und Fragestellungen aufgetreten, die im Rahmen dieser Arbeit nicht weiter untersucht werden konnten. Im Folgenden wird deshalb ein Ausblick auf mögliche weiteren Forschungen in diesem Bereich gegeben.

Datenextraktion

Im Bereich der Datenextraktion erfolgt momentan zwar eine Extraktion von Eigennamen, es werden jedoch keine Koreferenzen aufgelöst. Über eine solche Auflösung von Koreferenzen könnte die Erkennung von Beziehungen deutlich verbessert werden. Zudem könnten die Koreferenzen auf der Webseite gekennzeichnet werden, wodurch der Bezug für den Leser klar ersichtlich und somit das Textverständnis vermutlich weiter verbessert werden würde.
Eine weitere Verbesserung des Textverständnisses und der Recherchefunktionen könnte durch die Anreicherung der gefundenen Daten mit zusätzlichen Informationen erzielt werden. Über Methoden wie Named Entity Resolution könnten

beispielsweise weitere Daten zu den gefundenen Eigennamen aus anderen Quellen wie Wikipedia extrahiert und in der Visualisierung angezeigt werden. Hierdurch würden dem Benutzer, neben den Daten aus den eigenen Wissensgraphen, noch weiterführende Informationen zur Verfügung gestellt werden.
Die Daten im Wissensgraphen haben momentan, neben dem Abrufdatum der Seite, keinen weiteren zeitlichen Bezug. Die Beziehungen, wie beispielsweise der Bezug zwischen einer Person und einer Organisation und auch die Relevanz bestimmter Entitäten können sich jedoch mit der Zeit ändern. Durch die Extraktion von Zeitangaben und eine entsprechende Verknüpfung der extrahierten Daten mit diesen Zeitangaben, könnten diese Veränderungen erkannt werden. In der Visualisierung besteht die Möglichkeit solche Veränderungen, insbesondere durch die bereits verwendeten weichen Übergänge zwischen verschiedenen Graphen, darzustellen.

WebExtensions

Das *Storyfinder-Plugin* verwendet das *AddOn-SDK* von *Mozilla Firefox* zur Anbindung des Plugins an den Browser. Hierdurch ist das Plugin an *Mozilla Firefox* gebunden und funktioniert nicht in anderen Browsern. Durch *WebExtensions* steht eine Schnittstelle zur Verfügung, die in den meisten modernen Browsern unterstützt wird und auf ähnliche Web-Technologien wie das *AddOn-SDK* setzt. Einzig die fehlende Unterstützung einer Seitenleiste sprach gegen die Verwendung dieser Schnittstelle. Über eine Integration des Wissensgraphens auf der Webseite könnte das Plugin auch mit dieser Schnittstelle verwendet werden. Hierfür müsste jedoch weiter untersucht werden, wie die hieraus resultierenden und in Kapitel 4.4 beschriebenen Probleme bei der Einbindung des Graphen in eine Webseite, gelöst werden können.

8.3 Quellcode und Docker-Container

Der Quellcode des *Storyfinder-Webservers*, des *Storyfinder-Plugins* sowie die verschiedenen *Storyfinder Evaluation Tools* sind auf *GitHub*[1] unter den folgenden URLs abrufbar:

- https://github.com/mt-k/storyfinder-webserver
- https://github.com/mt-k/storyfinder-plugin
- https://github.com/mt-k/storyfinder-eval

Die folgenden drei *Docker-Container* zur direkten Installation des *Storyfinder-Servers* sind über das *Docker Hub*[2] verfügbar. Eine Installationsanleitung findet sich in Anhang A.1.

- kisad/storyfinder-webserver
- kisad/storyfinder-germaner
- kisad/corenlp-german

[1] https://github.com
[2] https://hub.docker.com

Literaturverzeichnis

[1] Daniel Abromeit. Webextensions. Technical report, Mozilla Foundation, 2016. https://developer.mozilla.org/de/Add-ons/WebExtensions.

[2] Andreas Wagner. WebReview: Eine verbesserte Benutzerschnittstelle zum Wiederfinden bereits besuchter Webseiten mittels automatischer Erfassung, Analyse und Visualisierung von Webzugriffen. Diplomarbeit, Fachhochschule Kaiserslautern, 2009.

[3] Apple Inc. Safari für Mac: Ungestörtes Lesen von Artikeln in Safari. Published online, 2016. https://support.apple.com/kb/PH21467?locale=de_DE.

[4] Arc90. Readability - An Arc90 Lab Experiment. Published online, 2009. https://code.google.com/archive/p/arc90labs-readability/.

[5] Marco Baroni, Francis Chantree, Adam Kilgarriff, and Serge Sharoff. Cleaneval: a competition for cleaning web pages. In Proc. of the Sixth Language Resources and Evaluation Conference, LREC 2008, Marrakech, 2008.

[6] Benjamin B. Bederson and Angela Boltman. Does animation help users build mental maps of spatial information? In Proceedings of the 1999 IEEE Symposium on Information Visualization, INFOVIS '99, Washington, DC, USA, 1999. IEEE Computer Society.

[7] Darina Benikova, Uli Fahrer, Alexander Gabriel, Manuel Kaufmann, Seid Muhie Yimam, Tatiana von Landesberger, and Chris Biemann. Network of the day: Aggregating and visualizing entity networks from online sources. In Proceedings of the NLP4CMC Workshop at KONVENS, Hildesheim, Germany, 2014.

[8] Darina Benikova, Seid Muhie Yimam, Prabhakaran Santhanam, and Chris Biemann. Germaner: Free open german named entity recognition tool. In International Conference of the German Society for Computational Linguistics and Language Technology (GSCL-2015), 2015. to appear.

[9] Robin Berjon, Steve Faulkner, Travis Leithead, Silvia Pfeiffer, Edward O'Connor, and Erika Doyle Navara. HTML5. Candidate recommendation, W3C, July 2014. http://www.w3.org/TR/2014/CR-html5-20140731/.

[10] Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An Algorithm that Learns What's in a Name. Machine Learning, 34(1):211–231, 1999.

[11] Steffen Blaschke. Wikis in Organisationen: Von Kommunikation zu Kollaboration, pages 183–203. Vieweg+Teubner, Wiesbaden, 2008.

[12] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3 Data-Driven Documents. IEEE Transactions on Visualization and Computer Graphics, 17(12):2301–2309, December 2011.

[13] Leo Breiman. Random forests. Mach. Learn., 45(1):5–32, October 2001.

[14] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30(1–7):107–117, 1998.

[15] William Weston Cohen. Fast effective rule induction. In Proceedings of the 12th International Conference on Machine Learning (ICML'95), pages 115–123, 1995.

[16] Microsoft Corporation. MSDN - Internet Explorer Browser Extensions. Technical report. https://msdn.microsoft.com/en-us/library/aa753587(v=vs.85).aspx.

[17] Lokesh Dhakar. Color Thief. Published online, 2011. http://lokeshdhakar.com/color-thief.

[18] DocumentCloud. Published online, 2011. https://www.documentcloud.org.

[19] J. Domingue, M. Dzbor, and E. Motta. Magpie: Supporting Browsing and Navigation on the Semantic Web. In N. Nunes and C. Rich, editors, Proceedings ACM Conference on Intelligent User Interfaces (IUI), pages 191–197, 2004.

[20] Michael Hartlep Dominik Wurnig, Jan Schneider. How to do... LOBBYRADAR - Datenjournalismus und eine Browser-Erweiterung als Form des Storytellings, 2015.

[21] Dria. Creating custom firefox extensions with the mozilla build system. Technical report, Mozilla Foundation, 2006. https://developer.mozilla.org/en-US/Add-ons/Creating_Custom_Firefox_Extensions_with_the_Mozilla_Build_System.

[22] Tim Dwyer. cola.js - Constraint-Based Layout in the Browser. Published online, 2013. http://marvl.infotech.monash.edu/webcola.

[23] Tim Dwyer and Kim Marriott. Constrained Stress Majorization Using Diagonally Scaled Gradient Projection, pages 219–230. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[24] Peter Eades. A Heuristic for Graph Drawing. Congressus Numerantium, 42:149–160, 1984.

[25] Dino Esposito. MSDN - Browser Helper Objects: The Browser the Way You Want It. Technical report, Microsoft Corporation, 1999. https://msdn.microsoft.com/en-us/library/ms976373.aspx.

[26] David A. Ferrucci. Ibm's watson/deepqa. SIGARCH Comput. Archit. News, 39(3):–, June 2011.

[27] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[28] Mozilla Foundation. Lightbeam. Published online, 2013. https://www.mozilla.org/en-US/lightbeam.

[29] Node.js Foundation. Express - fast, unopinionated, minimalist web framework for node. Published online, 2016. http://expressjs.com/.

[30] Tali Garsiel. How browsers work behind the scenes of modern web browsers. Published online, 2009. http://taligarsiel.com/Projects/howbrowserswork1.htm.

[31] Fréderic Godin, Pedro Debevere, Erik Mannens, Wesley De Neve, and Rik Van de Walle. Leveraging Existing Tools for Named Entity Recognition in Microposts. In Proceedings of the Third Workshop on Making Sense of Microposts (#MSM2013), pages 36–39, May 2013.

[32] Michael Große. drawWindow() broken with multiprocess Firefox (e10s)? Published online, 2016. http://stackoverflow.com/questions/34596722/drawwindow-broken-with-multiprocess-firefox-e10s.

[33] Günter Neumann. Informationsextraktion. In Computerlinguistik und Sprachtechnologie - Eine Einführung, pages 448–455. Spektrum Akademischer Verlag, Heidelberg, 2001.

[34] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. SIGKDD Explor. Newsl., 11(1):10–18, November 2009.

[35] Paul Heckbert. Color image quantization for frame buffer display. SIGGRAPH Comput. Graph., 16(3):297–307, July 1982.

[36] Google Inc. Material design - introduction. Published online, 2016. https://material.io/guidelines/material-design.

[37] Laurent Jouanneau. Slimerjs - a scriptable browser for web developers. Published online, 2012. https://slimerjs.org.

[38] Daniel Jurafsky and James H. Martin. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Prentice Hall, Pearson Education International, Upper Saddle River, 2014.

[39] Maria Kaya. Verfahren der Datenerhebung, pages 49–64. Gabler, Wiesbaden, 2007.

[40] D. A. Keim, C. Panse, M. Sips, and S. C. North. Visual data mining in large geospatial point sets. IEEE Computer Graphics and Applications, 24(5):36–44, Sept 2004.

[41] Wolfgang Klein and Alexander Geyken. Das Digitale Wörterbuch der Deutschen Sprache (DWDS). Volume 26:79–96, December 2010.

[42] Artjom Kochtchi, Tatiana von Landesberger, and Chris Biemann. Networks of Names: Visual Exploration and Semi-Automatic Tagging of Social Networks from Newspaper Articles. Computer Graphics Forum, 2014.

[43] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10, pages 441–450, New York, NY, USA, 2010. ACM.

[44] Christian Kohlschütter and Wolfgang Nejdl. A densitometric approach to web page segmentation. In Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08, pages 1173–1182, New York, NY, USA, 2008. ACM.

[45] Kunze, Claudia and Lemnitzer, Lothar. Computerlexikographie. Eine Einfuehrung. Tuebingen, 1 edition, 2007.

[46] Christopher Manning. Stanford tokenizer. Technical report, Stanford NLP Group, 2015. http://nlp.stanford.edu/software/tokenizer.shtml.

[47] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 55–60, 2014.

[48] Mauro Martino. Presenting watson news explorer. Published online, 2015.

[49] Dirk Merkel. Docker: Lightweight Linux Containers for Consistent Development and Deployment. Linux J., 2014(239), March 2014.

[50] Mossop. Add-ons. Technical report, Mozilla Foundation, 2010. https://developer.mozilla.org/en-US/Add-ons.

[51] MSDN - Creating Custom Explorer Bars, Tool Bands, and Desk Bands. Technical report, Microsoft Corporation. https://msdn.microsoft.com/en-us/library/bb776819(v=vs.85).aspx.

[52] npm, Inc. About npm. Published online, 2016. https://www.npmjs.com/about.

[53] Martin F. Porter. Snowball: A language for stemming algorithms. Published online, October 2001. Accessed 03.08.2016.

[54] Rohit Rai. Socket.IO Real-time Web Application Development. Packt Publishing, 2013.

[55] Mark Ransom. How to decide font color in white or black depending on background color? Published online, 2010. http://stackoverflow.com/questions/3942878.

[56] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In Proceedings of the Thirteenth Conference on Computational Natural Language Learning, CoNLL '09, pages 147–155, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[57] R. Rivest. The MD5 Message-Digest Algorithm, 1992.

[58] R. Rosenthal and D. B. Rubin. Comparing within- and between-subjects studies. Sociological Methods & Research, 9:127–136, 1980.

[59] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. Commun. ACM, 18(11):613–620, nov 1975.

[60] Satoshi Sekine and Chikashi Nobata. Definition, dictionaries and tagger for extended named entity hierarchy. In LREC. European Language Resources Association, 2004.

[61] Miroslav Spousta, Michal Marek, and Pavel Pecina. Victor: the Web-Page Cleaning Tool. In Proceedings of the 4th Web as Corpus Workshop, LREC, 2008.

[62] StatCounter. Top 5 desktop, tablet & console browsers in germany from dec 2015 to dec 2016. Published online, 2016. http://gs.statcounter.com/#browser-DE-monthly-201512-201612-bar.

[63] Jannik Strötgen and Michael Gertz. WikiWarsDE: A German corpus of narratives annotated with temporal expressions. In In Proceedings of the conference of the German society for computational linguistics and language technology (GSCL 2011, pages 129–134, 2011.

[64] Turau, Volker. Algorithmische Graphentheorie (2. Aufl.). Oldenbourg, 2004.

[65] Ulrike Haš and Petra Storjohann. Einführung. Das Wort und der Wortschatz. In Handbuch Wort und Wortschatz. 2015.

[66] Web Content Accessibility Guidelines (WCAG) 2.0. http://www.w3.org/Translations/WCAG20-de/WCAG20-de-20091029/, W3C, October 2009.

[67] Wawuschel and Pollti. Übersichtliche Darstellung von Webseiten mit der Leseansicht. Published online, 2015. https://support.mozilla.org/de/kb/Uebersichtliche-Darstellung-von-Webseiten-mit-der-Leseansicht.

[68] WBamberg. Legacy extensions. Technical report, Mozilla Foundation, 2013. https://developer.mozilla.org/en-US/Add-ons/Overlay_Extensions.

[69] Wolfgang Koch, Beate Frees. ARD-ZDF-Onlinestudie 2016. Media Perspektiven, 2016(9):418–437, September 2001.

[70] Michael Wybrow and Tim Dwyer. Adaptagrams - libcola. 2013. http://www.adaptagrams.org.

[71] Jianhan Zhu, Victoria Uren, and Enrico Motta. Espotter: Adaptive named entity recognition for web browsing. In Proceedings of the Third Biennial Conference on Professional Knowledge Management, WM'05, pages 518–529, Berlin, Heidelberg, 2005. Springer-Verlag.

A Technische Komponenten

A.1 Installationsanleitung

A.1.1 Installation des Plugins

Das *Storyfinder-Plugin* ist unter der folgenden URL abrufbar:
https://github.com/mt-k/storyfinder-plugin/releases

Zur Installation kann der Link für das passende XPI in Mozilla Firefox angeklickt werden (z.B. storyfinder-0.0.4-fx-mac.xpi). Das Plugin installiert sich anschließend automatisch. Nach der Installation muss in den Einstellungen des Plugins (Menü: Extras / Add-ons) die Server-URL des Storyfinder-Webservers eingetragen werden.

A.1.2 Installation der serverseitigen Komponenten

Die Komponenten des Storyfinder-Servers stehen als Docker Container zur Verfügung. Der Storyfinder-Server benötigt die folgenden vier Komponenten: MySQL Datenbank, GermaNER für die NER, Stanford CoreNLP und den Storyfinder-Webserver.

MySQL Container

Ein MySQL 5.5 Container kann mit folgendem Befehl erzeugt werden:

```
docker run --name storyfinder-mysql -e MYSQL_RANDOM_ROOT_PASSWORD=yes \
-e MYSQL_DATABASE=storyfinder -e MYSQL_USER=storyfinder \
-e MYSQL_PASSWORD=storyfinder -d mysql:5.5.51
```

CoreNLP

Ein CoreNLP Container kann mit folgendem Befehl erzeugt werden:

```
docker run --name storyfinder-corenlp -d kisad/corenlp-german
```

GermaNER

Ein GermaNER Container kann mit folgendem Befehl erzeugt werden:

```
docker run -d -m 4g --name storyfinder-germaner kisad/storyfinder-germaner
```

Das Starten des GermaNER Containers kann mehrere Minuten dauern. Der Status kann in den Container-Logs ermittelt werden.
Hinweis: GermaNER benötigt mindestens 4GB Arbeitsspeicher. Bei der Verwendung von Kitematic oder der Docker Toolbox stehen der VM standardmäßig nur 2GB Arbeitsspeicher zur Verfügung. Der VM muss in diesem Fall mehr Speicher zugewiesen werden, wie unter folgendem Link beschrieben wird: http://stackoverflow.com/questions/32834082/how-to-increase-docker-machine-memory-mac

Storyfinder-Webserver

Der Storyfinder-Webserver kann mit folgendem Befehl erzeugt werden:

```
docker run -d --name storyfinder-server --link storyfinder-mysql:mysql
--link storyfinder-germaner:germaner --link storyfinder-corenlp:corenlp
-p 3055:3055 -e "COOKIE_SECRET=YOUR_COOKIE_SECRET" -e "MYSQL_PORT=3306"
kisad/storyfinder-webserver
```

Storyfinder ist anschließend unter der URL http:*docker_host_ip*:3055 erreichbar.

Konfiguration

Der Storyfinder-Webserver lässt sich über die folgenden Umgebungsvariablen konfigurieren. Die Standardwerte funktionieren mit den verlinkten Containern:

- COOKIE_SECRET: Secret zur Verschlüsselung und Signierung der Cookie Inhalte.
- MYSQL_HOST: Host der MySQL Datenbank mit dem Standardwert mysql.
- MYSQL_PORT: Port der MySQL Datenbank mit dem Standardwert 3306. Wird der MySQL Container über einen Link eingebunden, muss diese Umgebungsvariable immer angegeben werden.
- MYSQL_DATABASE: Name der MySQL Datenbank mit dem Standardwert storyfinder.
- MYSQL_USER: Benutzername der MySQL Datenbank mit dem Standardwert storyfinder.
- MYSQL_PASSWORD: Passwort der MySQL Datenbank mit dem Standardwert storyfinder.

A.2 Resourcen des Storyfinder-Webservers

Resource	HTTP-Methode	Pfad	Beschreibung
ArticlesController	GET	/Articles/:articleId	Archivierten Artikel öffnen
	PUT	/Articles/:articleId/image	Artikelbild speichern
EntitiesController	GET	/Entities/search	Entitäten oder Webseiten suchen
	GET	/Entities/:entityId	Entitätdetails laden
	PUT	/Entities	Neue Entität hinzufügen
	PUT	/Entities/:targetId/:sourceId	Zwei Entitäten verbinden
	DELETE	/Entities/:entityId	Entität löschen
GraphsController	GET	/Graphs	Daten des global Graphen abrufen
	GET	/Graphs/Site/:siteId	Daten eines lokalen Graphen abrufen
	GET	/Graphs/Group/:siteIds	Daten eines Gruppengraphen abrufen
RelationsController	GET	/Relations/Entity/:entityId	Beziehungen einer Entität abrufen
	GET	/Relations/:entity1Id/:entity2Id	Beziehung und Quellen zwischen zwei Entitäten abrufen
	PUT	/Relations/:entity1Id/:entity2Id	Beziehung zwischen zwei Entitäten setzen
SitesController	GET	/Sites	Liste der besuchten Seiten abrufen
	PUT	/Sites	Neue Seite hinzufügen/parsen
	PUT	/Sites/:siteId/image	Bild der Webseite speichern
UsersController	PUT	/Users	Benutzer registrieren
	GET	/register	Registrierungsmaske anzeigen
	GET	/login	Loginmaske anzeigen
	POST	/login	Anmeldung
	GET	/logout	Abmeldung

A.3 Datenbankschema

Tabelle	Feld	Typ	Beschreibung
ngrams	id	int	Eindeutige ID
	value	varchar	Text den N-Gramms
	collection_id	int	ID der Collection
	docs	smallint	Anzahl der Dokumente
	created	datetime	Erstellungszeitpunkt
users	id	int	Eindeutige ID
	username	varchar	E-Mailadresse des Benutzers
	password	text	Passwort als Bcrypt Hash
	created	datetime	Erstellungszeitpunkt
	modified	datetime	Änderungsdatum
	is_active	boolean	Ist das Benutzerkonto aktiv?
	is_deleted	boolean	Soft-Delete Flag
sites	id	int	Eindeutige ID
	url	varchar	URL der Seite
	hash	varchar	MD5 Hash des Seiteninhalts beim letzten Aufruf
	host	varchar	Host der Seite
	title	text	Titel der Seite
	favicon	text	URL des Favoriten Icons
	last_visited	datetime	Datum des letzten Seitenaufrufs
	primary_color	varchar	Dominante Farbe der Seite
	collection_id	int	ID der Collection
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag
visits	id	int	Eindeutige ID
	site_id	int	ID der aufgerufenen Seite
	referrer	int	ID der vorherigen Seite
	created	datetime	Erstellungszeitpunkt
	user_id	int	ID des Benutzers
	is_deleted	boolean	Soft-Delete Flag
relationtypes	id	int	Eindeutige ID
	collection_id	int	ID der Collection
	label	text	Beschriftung der Beziehung
	pattern	text	Noch nicht genutzt
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag

Tabelle	Feld	Typ	Beschreibung
relations_sentences	id	int	Eindeutige ID
	relation_id	int	ID der Beziehung
	sentence_id	int	ID des Satzes
	relationtype_id	int	ID des Beziehungstyps
	direction	tinyint	Noch nicht genutzt
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag
sentences	id	int	Eindeutige ID
	article_id	int	ID des Artikels
	text	text	Inhalt des Satzes
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag
articles	id	int	Eindeutige ID
	site_id	int	ID der Seite
	raw	text	HTML Quelltext des Artikels
	text	text	Text des Artikels
	excerpt	text	Zusammenfassung des Artikels (von Readability erzeugt)
	byline	text	Autor des Artikels (von Readability ermittelt)
	title	text	Titel des Artikels
	created	datetime	Erstellungszeitpunkt
	modified	datetime	Änderungsdatum
	collection_id	int	ID der Collection
	is_deleted	boolean	Soft-Delete Flag
collections	id	int	Eindeutige ID
	user_id	int	ID des Benutzers
	is_default	boolean	Handelt es sich um die Standardcollection des Benutzers?
	name	varchar	Name der Collection
	created	datetime	Erstellungszeitpunkt
	modified	datetime	Änderungsdatum
	is_deleted	boolean	Soft-Delete Flag

Tabelle	Feld	Typ	Beschreibung
relations	id	int	Eindeutige ID
	entity1_id	int	ID der alph. ersten Entität
	entity2_id	int	ID der alph. zweiten Entität
	relationtype_id	int	ID des Relationstyps oder null
	direction	tinyint	Noch nicht genutzt
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag
	user_generated	boolean	Wurde die Beziehung manuell hinzugefügt?
entities_sentences	id	int	Eindeutige ID
	entity_id	int	ID der Entität
	sentence_id	int	ID des Satzes
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag
entities	id	int	Eindeutige ID
	value	varchar	Erstes Token der Entität
	tokens	tinyint	Anzahl der Tokens
	multiword	varchar	Vollständige Entität, falls diese aus mehreren Tokens besteht
	caption	text	Beschriftung/Anzeigename der Entität
	type	varchar	Art der Entität
	master_id	int	ID der Master-Entität, falls diese Entität mit einer anderen Entität zusammengefasst wurde
	created	datetime	Erstellungszeitpunkt
	modified	datetime	Änderungszeitpunkt
	collection_id	int	ID der Collection
	last_seen	datetime	Datum, an dem die Entität zuletzt auf einer Seite gesehen wurde
	is_deleted	boolean	Soft-Delete Flag
	show_always	boolean	Soll die Entität immer angezeigt werden?
articles_entites	id	int	Eindeutige ID
	article_id	int	ID des Artikels
	entity_id	int	ID der Entität
	count	smallint	Häufigkeit des Auftretens der Entität im Artikel
	created	datetime	Erstellungszeitpunkt
	is_deleted	boolean	Soft-Delete Flag

B Evaluation

B.1 Fragebogen zur Erfassung allgemeiner Daten

Allgemeine Angaben zur Versuchsperson

A1) ID: 27

A2) Geburtsjahr

z.B. 1986

A3) Geschlecht

○ männlich ○ weiblich

A4) Wohnort

PLZ | Ort

Ausbildung und Beruf

B1) Höchster Schulabschluss

z.B. Qualifizierter Hauptschulabschluss oder Abitur

B2) Berufsabschluss

z.B. Mechatroniker oder Bachelor of Science

B3) Zur Zeit ausgeübter Beruf

z.B. Lehrer

Nutzungsverhalten im Internet

C1) Wie häufig nutzen Sie das Internet pro Woche?

○ unregelmäßig
○ an 1 - 2 Tagen pro Woche
○ mehrmals wöchentlich
○ 1 - 2 Stunden täglich
○ 2 - 5 Stunden täglich
○ mehr als 5 Stunden täglich

C2) In welchem Bereich nutzen Sie das Internet überwiegend?

○ privat
○ beruflich
○ Studium/Ausbildung

C3) Welche drei Webseiten (ohne Suchmaschinen wie z.B. Google.de oder Bing.de) verwenden Sie am häufigsten?

z.B. wikipedia.de, tu-darmstadt.de und heise.de

C4) Welche der folgenden Hilfsmittel verwenden Sie bei der Recherche oder beim Lesen von Nachrichten im Internet?

☐ Lesezeichen
☐ Feed Reader (RSS, Atom)
☐ Handschriftliche Notizen

Sonstiges

C5) Welches Leseverhalten entspricht am ehesten Ihrem Vorgehen beim Lesen von Artikeln auf Webseiten?

○ Ich überfliege den Artikel und lese nur wichtige Abschnitte.
○ Ich lese nur die Überschrift und Einleitung und suche dann nach wichtigen Begriffen im Text.
○ Ich lese den kompletten Artikel der Reihe nach.

Absenden

B.2 Webseiten im Evaluationskorpus

VÖ-Datum	Titel	URL	Zeichen	Wörter
20.09.16	Briefwahlstimmen für FDP erregen Verdacht	http://www.weser-kurier.de/region/delmenhorster-kurier_artikel,-Briefwahlstimmen-fuer-FDP-erregen-Verdacht-_arid,1460825.html	3095	441
30.07.16	CDU-Wahlkampfstart: Sicherheit in Zeiten des Terrors	http://www.noz.de/lokales-dk/delmenhorst/artikel/751916/cdu-wahlkampfstart-sicherheit-in-zeiten-des-terrors	4507	628
06.09.16	Diese Kandidaten wollen ins Delmenhorster Rathaus	http://www.noz.de/lokales-dk/delmenhorst/artikel/770458/diese-kandidaten-wollen-ins-delmenhorster-rathaus	871	127
16.08.16	Urkundenfälschung? Stadt Delmenhorst zeigt Ratsherrn an	http://www.noz.de/lokales-dk/delmenhorst/artikel/759810/urkundenfalschung-stadt-delmenhorst-zeigt-ratsherrn-an	3843	545
30.08.16	Elefantenrunde in Markthalle	http://www.weser-kurier.de/region/delmenhorster-kurier_artikel,-Elefantenrunde-in-Markthalle-_arid,1447100.html	3776	538
08.10.16	SPD und Piraten bilden Gruppe in Delmenhorster Stadtrat	http://www.noz.de/lokales-dk/delmenhorst/artikel/786864/spd-und-piraten-bilden-gruppe-in-delmenhorster-stadtrat#gallery&0&0&786864	1369	198
06.09.16	Die Stadt wird bunt bleiben	http://www.weser-kurier.de/region/delmenhorster-kurier_artikel,-Die-Stadt-wird-bunt-bleiben-_arid,1451625.html	5020	762
16.08.16	Stadt Delmenhorst zeigt Ratsherr Wohnig an – Vorwurf der Urkundenfälschung bei Unterschriftenlisten für die Kommunalwahl	http://delmenews.de/stadt-delmenhorst-zeigt-ratsherr-wohnig-an-vorwurf-der-urkundenfaelschung-bei-unterschriftenlisten-fuer-die-kommunalwahl/	2569	336
09.09.16	Wie viel Natur steckt im Wahlkampf? Nabu stellt Umfrage vor	http://www.noz.de/lokales-dk/delmenhorst/artikel/771710/wie-viel-natur-steckt-im-wahlkampf-nabu-stellt-umfrage-vor	2747	368
unbekannt	Deniz Kurku - Über mich	http://deniz-kurku.de/ueber-mich/	3097	423
16.08.16	Delmenhorst zeigt Ratsherrn Volker Wohnig an	http://www.weser-kurier.de/region/delmenhorster-kurier_artikel,-Delmenhorst-zeigt-Ratsherrn-Volker-Wohnig-an-_arid,1437429.html	2793	389
04.10.16	Delmenhorster ab 60 wählen neuen Seniorenbeirat	https://weserreport.de/2016/10/delmenhorst/delmenhorster-waehlen-neuen-seniorenbeirat/	3669	508
			∅ 3113	∅ 438,58

B.3 Fragen zu den Texten im Evaluationskorpus

Die Fragen sind im Folgenden nach Fragetypen gruppiert.
Die Zahl vor der Frage gibt die Reihenfolge im Fragebogen an.
Die richtige Antwort ist durch Unterstreichung gekennzeichnet.

Fragen nach einer Person:

1. Wie heißt der Fraktionschef der CDU?

Pedro Benjamin Becerra	Tobias Ziegler	Sven Hoover	Marko Jung	Kristof Ogonovski

2. Wie heißt der Chef der Jungen Union?

Marko Jung	Werner Lindemann	Nils Pagel	Axel Jahnz	Tobias Ziegler

3. Wie heißt die amtierende Bürgermeisterin?

Eva Sassen	Antje Beilemann	Marlis Düßmann	Bettina Oestermann	Gudrun Sievers

21. Wie heißt die Fraktionschefin der SPD?

Eva Sassen	Daniela Bar	Janina Koehler	Bettina Oestermann	Antje Beilemann

Fragen nach einer Begriffserklärung:

4. Was versteht man unter „panaschieren"?

Die Verunglimpfung eines anderen Kandidaten durch falsche Aussagen	Die Aufteilung von Wahlstimmen über mehrere Listen oder Kandidaten hinweg	Die Umverteilung von Sitzen im Stadtrat durch Fraktionswechsel von Ratsherren	Den Wechsel eines Ratherrens während der Legislaturperiode zu einer anderen Fraktion	Die Anhäufung von Wahlstimmen für einen Kandidaten

25. Wie bezeichnet man das Verfahren zur Aufteilung von Restplätzen bei einer Wahl anhand der höheren Nachkommastellen?

BBL-Verfahren	Hagenbach-Bischoff-Verfahren	Fractional part nomination	Quotenausgleichs-Verfahren	Hare-Niemeyer-Verfahren

Fragen nach einer Zahl oder Partei, die als wichtige Fakten im Text angegeben waren:

5. Wie viele Sitze gibt es im Stadtrat insgesamt?

32	36	40	44	48

9. Aus wie vielen Wahlbezirken besteht das Wahlgebiet?

3	4	6	7	9

10. Wie viele Stimmen hat jeder Wähler?

Genau 5	Maximal 3	Genau eine	Maximal 5	Genau 3

12. Welche Partei stellte die meisten Kandidaten zur Wahl?

SPD	CDU	Die Grünen	FDP	Pro Delmos

13. Wie viele Plätze umfasst der Seniorenbeirat?

6	8	9	14	15

15. Wie viele Parteien standen zur Wahl?

7	10	12	13	15

16. Wie viele Kandidaten traten bei der Ratswahl an?

87	112	168	182	214

19. Welche Partei stellte die wenigsten Kandidaten zur Wahl?

Die Grünen	AfD	FDP	Die Piratenpartei	UAD

29. Ab welchem Alter ist man für die Wahl des Seniorenbeirates wahlberechtigt?

55	60	63	65	66

Fragen nach Zahlen, die nur beiläufig im Text erwähnt wurden:

7. Wie viel Prozent der Bausubstanz in Delmenhorst wurde vor 1978 erbaut?

60%	65%	70%	75%	80%

11. Um welchen Faktor möchte Karl-Heinz Bley die Abschieberate von Flüchtlingen verändern?

Auf das Doppelte	Auf das Dreifache	Auf die Hälfte	Auf das Fünffache	Keine Veränderung

14. Wieviele Unterschriften je Wahlbezirk benötigen Parteien, die weder in einem Landtag noch im Bundestag vertreten sind, um zur Wahl zugelassen zu werden?

15	20	25	30	35

23. Wie viele Sitze hatte die Partei „Piratenpartei" in der Legislaturperiode 2011-2016?

1	2	3	4	5

24. In wie vielen Wahlbezirken erreichte die Partei „Pro Delmos" nicht die benötigte Anzahl an Unterschriften für die Zulassung zur Wahl?

1	2	3	4	5

Fragen nach Informationen, die keine Entitäten darstellen:

6. Benötigt man die deutsche Staatsangehörigkeit um für die Wahl des Seniorenbeirates Wahlberechtigt zu sein?

ja	nein

20. Welches Jubiläum feierte der CDU Kreisverband bei der Veranstaltung zum Wahlkampfstart?

60 Geburtstag des Fraktionsvorsitzenden	65 jähries bestehen des Kreisverbandes	70 jähriges bestehen des Kreisverbandes	Seit 60 Jahren dauerhaft im Kreistag vertreten zu sein	65. Geburtstag des Fraktionsvorsitzenden

22. Was studierte der SPD Kandidat Deniz Kurku?

Wirtschaftswissenschaften	Maschinenbau	Politikwissenschaften	Wirtschaftsinformatik	Soziologie

27. Bei welchem Wahlkampf wurde der Wahlslogen der CDU „Keine Experimente!" bereits schon einmal von der CDU verwendet?

Bundestagswahl 1957	Kreistagswahl in Delmenhorst 1977	Landtagswahl in Niedersachsen 1963	Bundestagswahl 1976	Landtagswahl in Niedersachsen 2003

28. Mit welchem Fahrzeug erschien der Chef der Jungen Union bei der Veranstaltung zum Wahlkampfstart?

Elektroauto	Traktor	Autonomes Auto	Fahrrad	Mercedes

Fragen nach Informationen, die über mehrere Sätze verteilt sind:

8. Welcher Partei gehörten die beiden ehemaligen Ratsherren der Partei „Pro Delmos" ursprünglich an?

CDU	SPD	Piratenpartei	Die Linke	AfD

17. Durch wen wurde die Anzeige wegen Wahlbetruges gegen die FDP erstattet?

Stadt Delmenhorst	Bürgermeisterin Antje Beilem	Anonym	Andreas Neugebauer, Fraktionsvorsitzender der Piratenpartei	Wahlleiter Horst Jenocha

18. Wie heißt der einzige Ratsherr der Partei „Die Piraten", der die Wiederwahl schaffte?

Kristof Ogonovski	Horst Jenocha	Henning Suhrkamp	Nils Pagel	Andreas Neugebauer

26. Welche der folgenden Ratsmitglieder sind in der Legislaturperiode 2011-2016 mit dem Gesetz in Konflikt geraten?

Pedro Benjamin Becerra	Henning Suhrkamp	Axel Jahnz	Eva Sassen	Werner Lindemann