State of the Art – Web Scraping

Bachelorarbeit zur Erlangung des akademischen Grades
Bachelor of Science im Studiengang Angewandte Informationswissenschaft
an der Fakultät für Informations- und Kommunikationswissenschaften
der Technischen Hochschule Köln

vorgelegt von: Kolja Maxim Günther

eingereicht bei: Prof. Dr. Philip Schaer
Zweitgutachterin: Mandy Neumann, M.A.

Köln, 29.08.2019

Eidesstattliche Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer oder der Verfasserin/des Verfassers selbst entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Köln, 29.08.2019

Ort, Datum

Rechtsverbindliche Unterschrift

Abstract

Die vorliegende Bachelorarbeit hat zum Ziel, dem allgemeinen Leser die aktuell genutzten Methoden des Web Scraping zur Extraktion von relevanten Daten aus vorher definierten Webseiten darzustellen und miteinander zu vergleichen.

Dafür wurde zunächst der Forschungsgegenstand in seiner Art und Funktionsweise definiert und gegen andere Informationsextraktionsverfahren abgegrenzt. Anschließend galt es grundlegende, aktuelle und innovative Techniken des Web Scraping anhand von ausgewählten Beispielen vorzustellen. Anhand von drei Kategorisierungsansätzen (Glez-Pena, Ferrara, Chang) wurden die verschiedenen Ebenen, auf denen ein Web Scraper zu betrachten ist, herausgearbeitet und in einer Gegenüberstellung der Ansätze auf Parallelen und Gegensätze dieser Kategorien überprüft.

Web Scraping bietet eine probate Methode, relevante Inhalte aus dem World Wide Web ohne erforderlichen Zugriff auf den jeweiligen Webserver der Zielseite zu extrahieren. Dabei führt die zunehmende Entwicklung von Web Scrapern hin zu einer benutzerfreundlichen visuellen Umgebung und einfachen Bedienung dazu, dass die Tools sich einem immer breiteren Anwenderpublikum öffnen.

Schlagwörter/Schlüsselwörter: Web Data Extraction, Web Harvesting, Datenextraktion, Web Scraping, (Web) Information Extraction

Inhalt

Abbildungsverzeichnis

Abkürzungsverzeichnis

API ..... Application Programming Interface
CSS ..... Cascading Stylesheet
DOM ..... Document Object Model
HTML ..... Hypertext Markup Language
HTTP ..... Hypertext Transfer Protocol
REST ..... Representational State Transfer
SOAP ..... Simple Object Access Protocol
XML ..... Extensible Markup Language

Grundlegende Begriffe und Kürzel

Webserver

Der Webserver stellt einem Client, wie zum Beispiel einem Webbrowser, alle notwendigen Inhalte zur Verfügung, mit denen dieser eine Webseite entsprechend den Vorgaben des Servers auf dem Endgerät des Nutzers anzeigen kann.

Client

Der Client ist das Gegenstück zum Server und schickt (im Falle eines Webbrowsers) HTTP-Requests zur Datenkommunikation an diesen, die von Server zum Beispiel mit HTML-Inhalten zur Darstellung einer Webseite beantwortet und entsprechend der Servervorgaben auf dem Endgerät des Nutzers verarbeitet werden.

HTTP

Das Hypertext Transfer Protocol ist ein Kommunikationsprotokoll zwischen Server und Client, mit dem diese Daten austauschen können. Das HTTP gilt weithin als Standardprotokoll zur Übermittlung von Webseiten und webseitenrelevanten Daten im Internet.

HTML

Mithilfe der Hypertext Markup Language werden die Struktur und die textuellen Inhalte einer Webseite definiert. Bekommt ein Client ein HTML-Dokument vom Server geschickt, kann er daraus die Inhalte der Seite und ihre Position im Webdokument rendern. Die Struktur gibt HTML dabei über Tags vor, die jedoch darüber hinaus keinerlei Auskunft über den umschlossenen Inhalt geben.

XML

Die Extensible Markup Language ist eine Sprache zur Darstellung von strukturierten Daten. Wie HTML wird die Struktur der Inhalte über Tags definiert, jedoch sagen XML-Tags aus, welchen Inhalt sie umschließen und stehen in hierarchischer Relation zueinander.

Daten

Daten sind Inhalte, die, für sich alleinstehend noch keinen spezifischen Informationswert haben und erst im weiteren Prozess zu Informationen verarbeitet werden können. Dafür bedarf es eines zum Informationsbedürfnis eines Nutzers stehenden Kontexts.

Informationen

Informationen sind Daten, die im Kontext eines spezifischen Informationsbedürfnisses so aufbereitet wurden, dass die Inhalte der Daten dem Nutzer zur Befriedigung dieses Informationsbedürfnisses dienen.

Website/Webseite

Eine Website beschreibt die Gesamtheit der innerhalb einer Top-Level-Domain vorkommenden Inhalte, also beispielsweise den gesamten Internetauftritt eines Unternehmens.

Webseiten sind im Gegensatz dazu einzelne Seiten innerhalb dieses Auftritts, also beispielsweise das Impressum oder die Homepage des Unternehmens.

1 Einleitung

1.1 Ausgangsproblematik

Aktuell umfasst das indexierte Internet laut worldwidewebsize.com etwa 5 Milliarden erfasste Webseiten[1] und das weltweite Datenaufkommen betrug im Jahr 2018 um die 33 Zettabyte. Dieses Volumen entspricht etwa 33 Billionen Gigabye und soll sich Prognosen zufolge bis zum Jahr 2025 verfünffachen.[2]

Mit dem damit verbundenen rapiden Wachstum des Informationsvolumens, welches im Internet heutzutage zur Verfügung gestellt wird, haben sich – neben den neuen Möglichkeiten der Nutzung – auch die Problemstellungen der immer stärker werdenden Heterogenität der Webseiteninhalte in Struktur, Form und Inhalt sowie die wachsende Unkontrollierbarkeit der Masse und Relevanz von Daten im World Wide Web entwickelt.

Dabei können beispielsweise in den Bereichen der Sozialforschung, der naturwissenschaftlichen Forschung und der Unternehmensführung allgemein zugängliche Daten aus dem Internet genutzt werden, um daraus nützliche Informationen zur Forschung oder Strategieentwicklung abzuleiten. So bieten extrahierte Social-Media-Daten im Kontext der Sozialfoschung beispielsweise die Möglichkeit, Informationen zu gewinnen, die auf menschliche Verhaltensmuster und Beziehungsgeflechte einzelner Personen oder Gruppen hin analysiert werden können, um daraus neue wissenschaftliche Erkenntnisse zu ziehen.

Biomedizinische Fachdatenbanken unterliegen oft keiner einheitlichen Struktur, sodass es sich als fast unlösbare Aufgabe erweist, die Informationen in einer einheitlichen Datenbank zusammenzufassen, zumal der Fachbereich eine enorme Masse an Datenbanken aufweist, die sich ständig in ihrer Struktur verändern und wachsen. 2011 wurde dabei die Anzahl auf 1380 biomedizinische Datenbanken geschätzt.[3] Die Herausforderung in diesem Bereich besteht darin, gesuchte Informationen aus verschiedenen biomedizinischen Quellen zusammenzutragen, sodass sie von Nutzern in einer gemeinsamen Ansicht abrufbar sind.

Betrachtet man den allgemeinen Wirtschaftssektor in seiner aktuellen Entwicklung, wächst die Rolle der Business Intelligence zunehmend. Ein Unternehmen, das sich heutzutage sowohl im lokalen aber insbesondere auch im globalen Markt langfristig positionieren will, kommt nicht um die Aufgabe herum, frei zugängliche Daten von Wettbewerbern, Kundenmeinungen zu einer bestimmten Marke oder einem bestimmten Produkt sowie Marktdaten und marktrelevante Nachrichten aus dem Web zu sammeln und

[1] WORLDWIDEWEBSIZE.COM (De Kunder, Maurice): The size of the World Wide Web (The Internet). URL: https://www.worldwidewebsize.com, letzter Aufruf: 28.04.2019.

[2] DYCK, Andreas (12.03.2019): Wie das Internet das Surfen lernte. In: general-anzeiger-bonn.de, URL: http://www.general-anzeiger-bonn.de/news/digitale-welt/Wie-das-Internet-das-Surfen-lernte-article4065601.html, letzter Aufruf: 28.04.2019.

[3] GALPERIN, Michael Y. / Fernández-Suárez, Xosé M. (2011): The 2012 Nucleic Acids Research Database Issue and the online Molecular Biology Database Collection. In: Nucleic Acids Research 40 (D1), S. D1-D8. DOI: https://doi.org/10.1093/nar/gkr1196.

auszuwerten. Die gewonnenen Informationen kann das Unternehmen zum Beispiel nutzen, um sein Produktportfolio entsprechend der Kundenmeinungen zu überarbeiten oder weitreichende Entscheidungen zur Unternehmensausrichtung und -strategie mit detaillierten und aktuellen Markt- und Wettbewerbsinformationen zu fällen.

Konfrontiert mit den beschriebenen Herausforderungen und Chancen des heutigen Datenaufkommens, hat sich in den vergangenen Jahren aus den Bedürfnissen von Wissenschaft und Wirtschaft die Disziplin der Web Data Extraction entwickelt, welche verschiedene Methoden zur kontrollierten Extraktion und Aufbereitung von Daten aus dem World Wide Web anbietet.

1.2 Ziel der Arbeit

Ziel der Arbeit ist es, den Begriff Web Scraping als eine Methode der Web Data Extraction zu definieren und den aktuellen Stand der Wissenschaft sowie aktuelle Technologien in diesem Feld vorzustellen. Der technische Aufbau eines Web Scrapers wird anhand unterschiedlicher Herangehensweisen erklärt sowie aktuelle Kategorisierungsansätze des Web Scrapings beschrieben und gegenübergestellt.

1.3 Aufbau und Vorgehensweise

Zunächst soll der Forschungsgegenstand des Web Scraping im Umfeld der Methoden zur Datenextraktion aus dem Internet definiert, der technische Prozess schematisch dargestellt und das Profil des Web Scraping im Kontext der Web Data Extraction gegenüber alternativen Methoden geschärft werden.

Im Fokus der vorliegenden Arbeit steht die Aufgabe, einen aktuellen Überblick über verschiedene Web-Scraping-Technologien zu schaffen. Anhand von ausgewählten Beispielen soll der momentane Stand der Wissenschaft skizziert und dabei die Vor- und Nachteile der Herangehensweisen sowie einzelner Tools gegenübergestellt und diskutiert werden. Die Vorstellung verschiedener Kategorisierungsansätze für Web Scraper soll zusätzlich einen Überblick über die Bandbreite und Entwicklungsstufen der möglichen Umsetzungsmethoden geben.

Ein grober Überblick über die verschiedenen Anwendungsfelder des Web Scraping dient dazu, ein differenziertes Bild davon vermitteln zu können, wie Web Scraping praktisch genutzt werden kann.

Folgende Fragestellungen lassen sich für die Arbeit ableiten:

- Inwiefern unterscheidet sich Web Scraping von anderen webbasierten Informationsextraktionsverfahren, wie zum Beispiel Web APIs?
- Welche aktuellen Web-Scraping-Verfahren gibt es und worin liegen die Vor- bzw. Nachteile dieser?
- Unter welchen Aspekten lassen sich die vorgestellten Web-Scraping-Verfahren kategorisieren?

- Wo liegen die aktuellen Anwendungsfelder des Web Scraping?

1.4 Methodik

Die Arbeit wird als Literaturarbeit verfasst, in der die vorhandene Fachliteratur identifiziert, einzelne Positionen und Darstellungen gegenübergestellt und Schlüsse aus diesen gezogen werden. Dies kommt besonders in der Definition von Web Scraping, der Vorstellung der Methoden und Technologien und der Gegenüberstellung der vorhandenen Web-Scraping-Kategorisierungsansätze zum Tragen.

2 Begriffserklärung Web Scraping

2.1 Definition von Web Scraping und Abgrenzung

In der heutigen Zeit spielt die Extraktion von Daten aus dem Word Wide Web zur Gewinnung spezifischer Informationen und Erkenntnisse verschiedenster Art eine immer bedeutendere Rolle. Die schiere Masse an frei zugänglichen Daten macht es Forschungsinstitutionen und Wirtschaftsunternehmen fast unmöglich, diese Ressource zu ignorieren.

Aus diesem Potenzial heraus hat sich in den vergangenen Jahren eine Branche entwickelt, die sich auf die Gewinnung dieser Daten spezialisiert und gleichzeitig die Entwicklung der Methoden zur Extraktion von Daten aus dem Web vorangetrieben hat.

Im Folgenden soll das Profil des Web Scraping als eine dieser Methoden geschärft und die Unterschiede in der Herangehensweise gegenüber dem Ansatz der Informationsgewinnung mit Hilfe von Web API abgegrenzt werden.

2.1.1 API und Web Services zur serverseitigen Datenextraktion

Wenn es darauf ankommt, schnell und einfach Daten von bestimmten Webseiten zu extrahieren, sind Web Services aktuell in einigen Bereichen das erste Mittel der Wahl.[4] Der Nutzer bedient sich dafür einer vom Betreiber der Zielwebseite zur Verfügung gestellten Programmierschnittstelle, welche es ihm ermöglicht, den Server der Seite direkt anzusprechen und auf die darauf gespeicherten Daten zuzugreifen. Diese Schnittstelle, besser bekannt als Application Programming Interface (API), wird von einem Web Service genutzt, um spezifische Anfragen an den Host beziehungsweise Server der Zielseite zu stellen. Die API übermittelt die Anfrage an den Server und gibt die Antwort wiederum an den Web Service zurück. Zur Kommunikation zwischen Web Service und API werden standardmäßig die beiden Protokolle Simple Object Access Protocol (SOAP) und Representational State Transfer (REST) genutzt.[5]

Gerade Fachdatenbanken stellen oft eine Schnittstelle zu themennahen Web Services bereit, über die der Nutzer datenbankübergreifend nach den gewünschten Informationen suchen kann, um sie dann beispielsweise in einer gesammelten Ansicht zur Verfügung gestellt zu bekommen. Dadurch wird die weitverbreitete Heterogenität in Struktur und Aufbereitungsart der Daten, wie sie zum Beispiel im Bereich der biomedizinischen Fachdatenbanken auftritt, durch Datenintegration umgangen.[6]

Gleichzeitig stellen Dienste wie Google APIs zu ihren Anwendungen zur Verfügung. Beim spezifischen Beispiel von Google Maps ist es dadurch Programmierern möglich,

[4] Vgl. GLEZ-PEÑA, D. / Lourenço, A. / López-Fernández, H. u.a. (2014): Web scraping technologies in an API world. In: Briefings in Bioinformatics 15(5), S. 789.

[5] Vgl. EBERSBACH, Anja / Glaser, Markus / Heigl, Richard (2011): Social Web. UVK, Konstanz. S. 175–177.

[6] Vgl. ebd. S. 788.

das zur Verfügung gestellte Kartenmaterial als Mashup – also als Zusammenführung verschiedener Applikationsdaten zum Zweck der Erzeugung von Daten mit neuem Informationsgehalt – in ihre eigene Software zu integrieren und mit zusätzlichen Funktionen beziehungsweise den eigenen Overlays zu versehen.[7] Social-Media-Kanäle wie Facebook oder YouTube bieten APIs, um es Softwareentwicklern zu ermöglichen, auf ihre Daten zuzugreifen. Die seit 2006 implementierte Facebook-API ermöglicht es, extern auf Profilinformationen, Bilder und Freundeslisten zuzugreifen und bietet damit einen großen Pool an sozialwissenschaftlich relevanten Daten.[8]

Die Vorteile dieses Ansatzes liegen maßgeblich in zwei Punkten. Zum einen ist die Web-API-Schnittstelle durch den Zugriff auf den Datenpool des Zielseitenservers unabhängig von Veränderungen in der Struktur der Webseite. Dadurch muss die Schnittstelle, solange es keine strukturellen Veränderungen auf der Serverseite gibt, auf lange Sicht nicht angepasst werden. Der zweite Vorteil liegt folgerichtig darin, dass ein Web API, einmal als Schnittstelle zu einer weiterverarbeitenden Software eingerichtet, die bezogenen Daten automatisch aktualisieren kann, sobald es zu einer Datenveränderung auf dem Server kommt.

Dagegen ist eine weit verbreitete Schwierigkeit von APIs vor allem, dass keine einheitliche Konvention in der Art, wie eine solche Programmierschnittstelle aufgebaut werden soll, besteht. Dadurch steht ein Programmierer, der mehrere APIs zur Stillung seines Informationsbedürfnisses nutzen will, vor dem Problem, sich mit jeder Schnittstelle, ihrer Struktur und ihren Ausführungsbefehlen individuell auseinandersetzen zu müssen. [9]

2.1.2 Web Scraping als Gegenentwurf

Web APIs stellen demnach heutzutage eine Methode dar, um über zur Verfügung gestellte Schnittstellen strukturierte Daten in maschinenlesbarer Sprache direkt von den Servern bestimmter Webseiten abzurufen. Häufig genug steht ein Nutzer aber vor mehreren potenziellen Szenarien, in denen das Nutzen von Web APIs oder Web Services zur Datenextraktion nicht ausreicht oder funktioniert.

Zum einen kommt es gerade bei älteren oder weniger gut gepflegten Websites vor, dass eine Schnittstelle vom Betreiber der Webseite nicht angeboten wird. In diesem Fall hat der Nutzer keine Möglichkeit, serverseitig auf maschinenlesbare Daten dieser Seite zuzugreifen. In einem zweiten potenziellen Szenario existiert zwar eine API, der Umfang und die Qualität der Daten, die der Nutzer über diese beziehen kann, reichen aber nicht aus, um sein spezifisches Informationsbedürfnis zu stillen.[10]

[7] Vgl. EBERSBACH, Anja (2011): Social Web. S. 178.

[8] Vgl. SPICHALE, Kai (2016): API-Design. Praxishandbuch für Java- und Webser-vice-Entwickler. dpunkt.verlag, Heidelberg. S. 6.

[9] Vgl. EBERSBACH, Anja (2011): Social Web. S. 179.

[10] Vgl. GLEZ-PEÑA (2014): Web scraping technologies in an API world. S. 789.

Desweiteren können APIs kostenpflichtig oder auf eine bestimmte Frequenz an Zugriffen beschränkt sein. In beiden Fällen erweist sich ein ressourcensparendes beziehungsweise zeiteffizientes Arbeiten mit den zur Verfügung gestellten Daten als schwierig.[11]

In diesem Fall kann Web Scraping helfen, die gewünschten Daten aus einem völlig anderen Ansatz heraus zu gewinnen, ohne auf die Kommunikation mit den Servern der Zielwebseite und den damit verbundenen Einschränkungen angewiesen zu sein. Diesen Vorteil verschafft sich der Nutzer daraus, dass browserseitig auf die Hypertext Markup Language- (HTML) bzw. Document Object Model-Struktur (DOM) der Webseite zugegriffen wird. Innerhalb dieser Strukturen kann der Web Scraper dann die eingebetteten Inhalte der Seite identifizieren. Die Methode greift also nicht auf Informationen zu, die serverseitig zur Verfügung gestellt werden, sondern „kratzt" clientseitig die Daten ab, die dem menschlichen Leser der Webseite auch zur Verfügung stehen würden. Dieser Ansatz bedeutet im Umkehrschluss aber auch, dass mit Daten gearbeitet wird, die nur für den menschlichen Leser konzipiert und nicht dafür vorgesehen sind, von Maschinen ausgelesen zu werden, wodurch besondere Techniken erforderlich sind, die in dieser Arbeit erörtert werden sollen.

Allgemein ausgedrückt stellt Web Scraping eine Methode dar, spezifische Daten browserseitig von vorher definierten Webseiten zu extrahieren, in ein gewünschtes, strukturiertes Format zu konvertieren und in ein weiterverarbeitendes Tool zu speisen.

Die Auswahl der Tools, welche in nachgelagerten Prozessen die gesammelten Daten weiterverarbeiten, basiert in erster Linie auf dem Informationsgewinnungsziel des Nutzers beziehungsweise der potenziellen Kunden des Nutzers. Oft dienen einfache Datenbanksysteme als Ablage der bis zu diesem Zeitpunkt strukturierten Daten. Besonders im wirtschaftlichen Anwendungsbereich können die Daten in Customer-Relationship-Management-, Enterprise-Ressource-Management- oder Enterprise-Risk-Management-Software eingespeist und genutzt werden, um weitreichende Erkenntnisse hinsichtlich Entscheidungen der Unternehmensstrategie und -ausrichtung, sowie Insights in unternehmens- oder produktbezogene Kundenmeinungen zu gewinnen. Oft bieten kommerzielle Full-Service Web Scraper bereits passende Schnittstellen zu solchen Programmen an, um eine reibungslose Verarbeitungspipeline zu gewährleisten.[12]

Neben der Tatsache, dass die Inhalte einer im Browser aufgerufenen Seite nicht für Maschinen erstellt wurden, stellen Restriktionen des Seitenbetreibers eine weitere Herausforderung dar. Dies geschieht in Form einer robots.txt, eines in einer Website integrierten Textdokuments, das Bots und Crawlern vorschreibt, welche Seiten und Inhalte nicht betrachtet werden dürfen. Gleichzeitig sind Passwortsperren und Captchas ein

[11] Vgl. BAESENS, Bart / vanden Broucke, Seppe (2018): Practical Web Scraping for Data Science: Best Practices and Examples with Python. Apress, New York. S. 5.

[12] Vgl. FERRARA, Emilio / De Meo, Pasquale / Fiumara, Giacomo u.a. (2014): Web data extraction, applications and techniques: A survey. In: Knowledge-Based Systems (2014) Nr. 70, S. 312; GLEZ-PEÑA, D. (2014): Web scraping technologies in an API world. S. 792.

unüberwindbares Hindernis für reguläre Bots. Um diese Einschränkungen zu umgehen, nutzen neuere Scraping-Mechanismen beispielsweise Techniken, um menschliches Nutzerverhalten zu simulieren und diese Barrieren zu umgehen.[13]

2.2 Technischer Ablauf des Web Scraping

Um die Funktionsweise eines Web Scrapers verstehen zu können, ist es notwendig, im Vorfeld auf den allgemeinen Aufbau und die grobe Funktionsweise einer Webseite einzugehen.

Das World Wide Web funktioniert grundsätzlich nach dem Client-Server-Prinzip, in dem ein Client, in diesem Fall der Browser, eine Anfrage an einen Server, in diesem Fall den Webserver, stellt. Der Server sendet eine entsprechende Antwort an den Client zurück. Im Falle einer Webseite sind das in der Regel Anweisungen, in welchem Design und welcher Struktur die Seite dargestellt werden soll. Die allgemeine Sprache zur Kommunikation solcher Requests und Responses im World Wide Web ist das Hypertext Transfer Protocol (HTTP).

Webseiten werden in Form von HTML-Dokumenten im Browser angezeigt, die Informationen in Textform darstellen. Die grundlegende Struktur einer Webseite wird dabei in HTML-Tags erzeugt, welche geschachtelte Ebenen innerhalb der Struktur erzeugen. Der Web Scraper kann solche Strukturelemente über das Document Object Model der Seite erfassen und die Inhalte extrahieren. Enthält eine Webseite beispielsweise eine Tabelle mit relevanten Daten, kann der Scraper das HTML-Tag *<table></table>* identifizieren und alle darin enthaltenen Daten auslesen. Da die Tags selbst keine Informationen über ihren Inhalt geben, sondern nur Inhalte voneinander trennen beziehungsweise vorgeben, wo und wie diese auf der Webseite angeordnet werden sollen, spricht man bei HTML-Dokumenten von semi-strukturierten Daten.

Beim Web Scraping selbst kann zwischen den drei Hauptakteuren Nutzer, World Wide Web und Wrapper unterschieden werden, die jeweils miteinander interagieren, um ein funktionstüchtiges System zu erstellen und Daten aus dem Web in strukturierter Form zu gewinnen. Nachfolgend sollen diese anhand von Abbildung 1 vor dem Hintergrund ihrer Funktion und jeweiligen Aufgabe beschrieben werden.

2.2.1 Der Nutzer

Der Nutzer tritt mit einem spezifischen Informationsbedürfnis auf, welches er mit Hilfe von Web Scraping stillen möchte. Dafür erstellt er einen sogenannten Wrapper, welcher die konkreten Anforderungen des Nutzers bezüglich Art und Inhalt der gewünschten Daten in Form von Extraktionsregeln auf zuvor definierte Webseiten beziehungsweise Seitenelemente anwenden soll. Dieser kann dabei ein Experte im wissenschaftlichen Bereich der Informationsextraktion sein, aber auch ein Mitarbeiter eines Unternehmens

[13] Vgl. BARANOVSKIY, Evgeny (2011): Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten. Stuttgart, Universität Stuttgart [Diplomarbeit]. S. 16–17.

oder einer Institution, welcher mit limitierter Expertise in Bezug auf die Wrappererstellung auf benutzerfreundliche Web-Scraping-Systeme zurückgreift, um ein für seinen Arbeitgeber relevantes Informationsbedürfnis zu stillen. Die Bandbreite der potenziellen Nutzer von Web Scraping ist dank der aktuellen Vielfalt an Scraping Tools kaum eingeschränkt.[14]

2.2.2 Das World Wide Web

Das World Wide Web selbst stellt einen unerschöpflichen Datenpool bereit und enthält die Webseiten, deren Inhalte potenziell von Interesse für das Informationsbedürfnis des Nutzers sind. Dabei kann zwischen Trainingsseiten und tatsächlichen Zielseiten unterschieden werden. Trainingsseiten werden in bestimmten Web-Scraping-Ansätzen vom Nutzer definiert, um festzulegen, wie ein bestimmter Webdokumenttyp strukturiert ist. So kann dem Wrapper mitgeteilt werden, welcher Teil des Textes dem Titel, dem Abstract, dem Veröffentlichungsdatum oder anderen spezifischen Elementen entspricht. Die tatsächlichen Zielseiten sind im Umkehrschluss die Webseiten, die potenziell relevante Daten beinhalten und vom Nutzer als Anwendungsziele für den Wrapper festgelegt werden.[15]

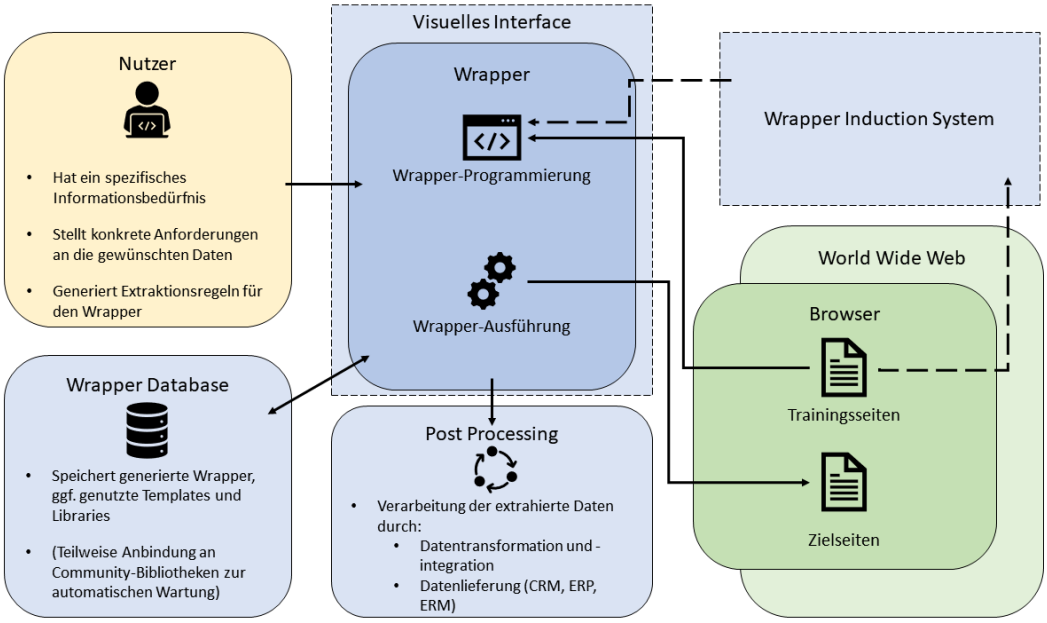

Abbildung 1: Schema eines Web Scrapers
(Quelle: Eigene Darstellung nach BAUMGARTNER, R. (2009): Web data extraction system. S. 3467.)

2.2.3 Der Wrapper

Der Wrapper beschreibt das eigentliche Softwareelement des Web Scraping und kann in etlichen Varianten auftauchen. Der Aufbau und die Funktionsweise eines Wrappers

[14] Vgl. BAUMGARTNER, R. / Gatterbauer, W. / Gottlob, G. (2009): Web data extraction system. In: Encyclopedia of Database Systems (5), S. 3467.

[15] Vgl. ebd.

hängen dabei davon ab, nach welchem Ansatz dieser erstellt und welches Informationsbedürfnis mit ihm gestillt werden soll.

Unabhängig davon kann ein allgemeines Schema erstellt werden, nach dem die allermeisten Wrapper erzeugt und ausgeführt werden. Im Folgenden sollen die einzelnen Elemente genauer untersucht werden.

Die Wrapper-Programmierung beschreibt das Element, in dem der Nutzer den Wrapper erstellt und alle Extraktions- und Transformationsparameter definiert. In dieser Phase wird festgelegt, welche Daten extrahiert werden sollen, und in welches Format diese strukturiert werden. Ist der Wrapper in ein visuelles Interface eingebettet, gibt es in der Regel mehrere, nachfolgend beschriebene Ansichten.[16]

Das Browserfenster ist eines der Kernelemente des visuellen Web Scrapers, in dem die Trainings- beziehungsweise Zielwebseiten live gerendert und angezeigt werden. Der Nutzer kann in dieser Ansicht ohne weitreichende Programmierkenntnisse relevante Seitenelemente markieren und definieren, wie der Wrapper mit diesen umgehen soll. Das Parse-Tree-Fenster dient dazu, die HTML-Struktur der aktuellen Webseite anzeigen zu lassen und relevante HTML-Tags zu markieren. Im Kontrollfenster hat der Nutzer eine Übersicht über alle Bereiche des Erstellungsprozesses und kann z.B. Extraktionsattribute oder -Tags festlegen. Das Programmfenster, in dem der Quellcode des Wrappers angezeigt wird, dient dazu, etwaige Detailanpassungen im Programm vorzunehmen, sofern entsprechende Programmierkenntnisse vorhanden sind.[17]

Nutzt der Web Scraper Machine-Learning-Mechanismen, um relevante Daten zu identifizieren, ist der Wrapper-Programmierung in der Regel ein sogenanntes Wrapper-Induction-System vorgelagert. Dieses interpretiert positive und negative Extraktionsbeispiele aus Trainingsseiten, um anhand von diesen eigene Extraktionsregeln zu erstellen und in den Wrapper zu integrieren. In Kapitel 3.2 wird noch einmal genauer auf diese Technik eingegangen.[18]

In der Wrapper-Ausführung interpretiert der Scraper die festgelegten Nutzervorgaben und konstruiert den fertigen Wrapper. Anschließend erfolgt die eigentliche Ausführung des programmierten Wrappers auf die vorher definierten Webseiten. Dabei können, je nach Tool, verschiedene Navigations- und Extraktionsmethoden, wie zum Beispiel Deep Web Navigation, Identifikation von relevanten Elementen durch deklarative Regeln oder prozedurale Scripts und das Erstellen eines strukturierten Ausgabeformats zum Einsatz kommen.[19]

[16] Vgl. ebd.

[17] Vgl. ebd.

[18] Vgl. GOEBEL, Max / Ceresna, M. (2009): Wrapper Induction. In: Encyclopedia of Database Systems (5), S. 3629.

[19] Vgl. BAUMGARTNER, R. (2009): Web data extraction system. S. 3468.

Die Wrapper Database wird genutzt, um den erstellten Wrapper, Metadaten und ggf. Templates und Libraries abzuspeichern. Durch diesen Vorgang kann der Wrapper jederzeit reproduziert und, falls nötig, entsprechend der Veränderungen der Zielwebseite gewartet und angepasst werden. Manche Web Scraper bieten eine direkte Anbindung des Speicherelements an genutzte Community-Bibliotheken an, wodurch im Wrapper verbaute Bibliothekselemente automatisch gewartet werden können.[20]

Im Post Processing werden die Ergebnisse verschiedener Suchprozesse gesammelt und zu einem strukturierten, homogenen Ergebnis im entsprechenden Zielformat zusammengefasst. Eine Strukturierung erfolgt durch Integration, Transformation und Bereinigung der gesammelten Daten. Die Datenlieferungseinheit bildet das Hauptausgabeelement des Post Processing. Hier können vom Nutzer die Ausgabekanäle für die strukturierten Daten festgelegt und Systemkonnektoren bereitgestellt werden, mit denen die Daten direkt in weiterverarbeitende Programme oder Datenbanken übergeben werden können. Besonders im wirtschaftlichen Kontext bildet die Datenlieferungseinheit ein wichtiges Kernelement, welches Daten direkt in Prozessentscheidungssoftware, wie zum Beispiel Customer-Relationship-Management-, Enterprise-Ressource-Management- oder Enterprise-Risk-Management-Software speist.[21]

[20] Vgl. ebd.

[21] Vgl. ebd.

3 Aktueller Stand der Wissenschaft

3.1 Momentaner Forschungsstand und aktuelle Web-Scraping-Technologien

Das Forschungsfeld des Web Scraping bietet eine vielfältige Bandbreite an Methoden und Technologien, um einen geeigneten Wrapper zu konstruieren und die gewünschten Daten zu extrahieren. Im Folgenden sollen grundlegende und heutzutage weit verbreitete, mit der Browserless Web Data Extraction aber auch solche Technologien vorgestellt werden, die mit ihrem Ansatz richtungsweisend für eine neue Herangehensweise der Datenextraktion aus dem Web werden könnten. Da die Möglichkeiten, einen Wrapper zu gestalten, sehr umfangreich sind, stellt dieses Kapitel einen Versuch dar, repräsentativ die wichtigsten beziehungsweise interessantesten Methoden zu beschreiben, um so ein angemessenes Bild des aktuellen Stands von Web-Scraping-Technologien zu erstellen.

3.1.1 Reguläre Ausdrücke

Reguläre Ausdrücke sind eine sehr einfache Methode, die ein Wrapper nutzen kann, um relevante Inhalte in vom Server erhaltenen Textdokumenten zu ermitteln. Trotz des vergleichweise simplen Ansatzes bilden sie eine der Grundlagenmethoden der Web Data Extraction und sollen daher hier besprochen werden.

Das Ziel von regulären Ausdrücken ist es, vorher festgelegte Buchstabenketten, sogenannte Strings, oder bestimmte Muster in Texten wiederzuerkennen und als relevant zu identifizieren. Die Strings können dabei zum Beispiel Schlüsselwörter sein, die, sobald sie in einem Textelement vorkommen, dieses als relevant ausweisen.[22] Sie werden in Form von Klassen kategorisiert, die ausdrücken, ob das Stringelement ein Buchstabe, eine Ziffer oder ein Sonderzeichen ist. Mithilfe dieser Klassen lassen sich dann Muster erstellen, die eine bestimmte Art Inhalt definieren. Beispielsweise kann die Regel aufgestellt werden, dass alle Strings mit der Abfolge Ziffer-Ziffer-Doppelpunkt-Ziffer-Ziffer eine Uhrzeit darstellen.[23]

Im Vorfeld müssen diese Strings und Muster mithilfe von Extraktionsregeln festgelegt werden. Da das Erstellen solcher Regeln oft komplex ist, fordert eine manuelle Vorgehensweise sehr viel Zeit und Expertise. Daher steht dem Nutzer bei Web Scrapern, die auf dem Ansatz der regulären Ausdrücke basieren, oft die Möglichkeit zur Verfügung, diese Regeln dynamisch zu generieren. Dabei markiert der Nutzer auf Testseiten

[22] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 306.

[23] Vgl. BARANOVSKIY, Evgeny (2011): Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten. S. 35–36.

gewünschte Elemente und der Scraper generiert daraus automatisch einen Wrapper mit entsprechenden Regeln, die er auf Seiten mit gleicher Struktur anwenden kann.[24]

Zu diesen Tools, die zur Erstellung von Extraktionsregeln reguläre Ausdrücke verwenden, gehört zum Beispiel W4F. Es stellt einen sogenannten Wizard zur Verfügung, mit dessen Hilfe der Nutzer relevante Elemente direkt markieren kann. Aus diesen Annotationen erstellt der Wizard dann die regulären Ausdrücke und stellt sie dem Nutzer zur Verfügung.[25]

Neben der relativ einfachen Handhabe solcher Wrapper auf Basis von regulären Ausdrücken, sticht in der praktischen Nutzung jedoch auch ein markanter Nachteil heraus. Dadurch, dass die Erstellung der Regeln auf der Struktur einer dafür festgelegten Testwebseite aufbaut und diese auch nur auf solche Zielwebseiten mit gleicher Struktur angewandt werden können, funktionieren die regulären Ausdrücke ab dem Moment nicht mehr, in dem eine Veränderung in der Struktur der Seite vorgenommen wird.[26] Das hat zur Folge, dass die Regeln neu geschrieben werden müssen und der ganze Ansatz in einem sich ständig verändernden Web ein hohes Maß an Wartungskosten mit sich bringt.

Eine weitere Schwierigkeit liegt in der Handhabe der erstellten Muster. Wenn mehrere zu betrachtende Webdokumente eines Scrapingvorgangs verschiedenen länderspezifischen Konformitäten unterliegen, können beispielsweise Datums- oder Währungsformate unterschiedlich ausfallen. In diesem Fall würde ein Muster, welches das Dokument zum Beispiel nach einem kommagetrennten Währungsformat durchsucht, ein punktgetrenntes Währungsformat nicht erkennen und verliert somit seine Allgemeingültigkeit.[27] Das führt dazu, dass mehrere Regeln desselben Typs angelegt werden müssen, was zu einer Aufblähung des Wrappers führt.

3.1.2 Logischer Programmieransatz

Der logische Ansatz stellt eigens für den Zweck der Erstellung eines Wrappers vorgesehene Porgrammiersprachen zur Verfügung, deren Funktionen auf die Datenextraktion zugeschnitten sind. Scraper, die mit solchen Sprachen programmiert wurden, erfassen eine Webseite nicht nur als reine Abfolge von Zeichenketten, sondern auch die dahinterliegende Struktur in Form eines DOM.[28]

XPath wurde ursprünglich als Sprache zur Adressierung von XML-Dokumenten entwickelt, die sich hierarchische Relationen zwischen einzelnen Elementen des Dokuments zunutze macht. Über das DOM ist es aber auch möglich, XPath für HTML-Dokumente

[24] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 306.

[25] Vgl. ebd.

[26] Vgl. ebd.

[27] Vgl. BARANOVSKIY, Evgeny (2011): Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten. S. 36.

[28] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 306.

zugänglich zu machen. Dabei nutzt die Sprache drei Wege, um Strukturelemente einer Webseite zu adressieren. Ausdrücke geben einen Adresspfad innerhalb des DOM an, der alle Elemente, welche dem Pfad entsprechen, anspricht. So sucht der Ausdruck /html/head/title beispielsweise nach Titeln innerhalb des <head>…</head>-Elements. Knotentests definieren gesuchte Elemente noch einmal tiefer, indem sie beispielsweise nicht aussagen, welche Elemente selbst gesucht werden, sondern innerhalb welcher Knoten sie vorkommen. Die Attributsuche nutzt nicht mehr allgemeine Knotentypen, sondern schränkt die Suche auf Elemente mit bestimmten Attributen, z.B. einer bestimmten Id ein. Diese Funktion ermöglicht es, Elemente auch dann wiederzufinden, wenn sich die DOM-Struktur einer Webseite verändert hat.[29]

3.1.3 Wrapper Induction

Die sogenannte Wrapper Induction beschreibt Machine-Learning-Ansätze im Kontext der Wrappererstellung, bei dem ein Wrapper Generation System Trennzeichen-basierte[30] Extraktions- und Transformationsregeln anhand von Beispielen und Gegenbeispielen lernt und diese in den ausführenden Wrapper implementiert.[31] Das Ziel der Wrapper Induction ist es dabei, den Wrapper automatisiert und mit so wenig Hilfe durch menschliche Nutzer wie möglich zu generieren.[32]
Bei den meisten Wrapper-Induction-Systemen werden die Extraktionsbeispiele aus einem Set an Trainingsseiten abgeleitet, die vom menschlichen Nutzer entsprechend dem Extraktionsziel belabelt werden und Formatierungsmerkmale nutzen, die die Struktur der gesuchten Daten beschreiben.[33] Neuere Ansätze verfolgen das Ziel, sogar auf diesen Labelingvorgang zu verzichten und mit unbelabelten Trainingsseiten automatisiert Regeln zu erzeugen.[34]

Chang unterteilt Wrapper Induction Systeme in drei Stufen des Automatisierungsgrades.[35] Bei voll beaufsichtigten Tools belabelt der Nutzer die Trainingsseiten händisch und das System erzeugt anhand der Beispiele einen Wrapper, dessen Extraktionsregeln die gewünschten Daten liefern. Neben der Tatsache, dass ohne diese Nutzer-System-Interaktion das Tool nicht in der Lage ist, einen Wrapper zu generieren, stellen vor allem die Menge an benötigten Trainingsdaten und die Unflexibilität in Bezug auf andere Aufgabenbereiche einen gewichtigen Nachteil dar. Viele vollbeaufsichtigte Wrapper

[29] Vgl. BARANOVSKIY, Evgeny (2011): Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten. S. 31.

[30] Vgl. LAENDER, Alberto H. F. / Ribeiro-Neto, Berthier A. / da Silva, Altigran S. / Teixeira, Juliana S. (2002): A Brief Survey of Web Data Extraction Tools. In: ACM SIGMOD Record 31 (2), S. 85.

[31] Vgl. GOEBEL, Max (2009): Wrapper Induction. S. 3629.

[32] Vgl. CHANG, CHIA-HUI / Kayed, M. / Girgis, M. R. u.a. (2006): A Survey of Web Information Extraction Systems. In: IEEE Transactions on Knowledge and Data Engineering 18(10), S. 1415.

[33] Vgl. LAENDER, Alberto H. F. (2002): A Brief Survey of Web Data Extraction Tools. S. 85.

[34] Vgl. CHANG, CHIA-HUI (2006): A Survey of Web Information Extraction Systems. S. 1419.

[35] Vgl. ebd. S. 1413

Induction Tools benötigen eine große Menge an belabelten Daten für ein vergleichsweise kleines Extraktionsziel.[36]

Teilbeaufsichtigte Wrapper Induction Systeme können über zwei verschiedene Ansätze vorgehen. IEPAD erstellt beispielsweise automatisch Extraktionsregeln, indem in der Lernphase wiederkehrende Muster einer Webseite identifiziert und in musterbasierte Extraktionsregeln umgewandelt werden. Anschließend wählt der Nutzer aus den gefundenen Mustern jene aus, die relevante Daten enthalten, woraus letztendlich der fertige Wrapper generiert wird.[37]

Unbeaufsichtigte Systeme entwickelten sich aus dem Bedürfnis, die hohen Kosten der manuellen Belabelung für beaufsichtigte Systeme zu minimieren. Sie bieten den aktuell höchsten Automationsgrad in der Wrappererstellung und benötigen keinerlei Eingreifen durch den Nutzer und keine belabelten Trainingsseiten, sondern arbeiten ausschließlich mit der Aussage des Nutzers über sein Informationsbedürfnis.[38]

Als einer der ersten induktiven Wrapperansätze gilt WIEN, welcher dafür konzipiert ist, mit wenig menschlichem Einsatz innerhalb einer Webseite strukturierte Textelemente in Tabellenform zu identifizieren. Dafür nutzt das Tool einheitliche Begrenzungen der Tabellenstruktur, um die Tabelle selbst vom restlichen Text zu separieren und die einzelnen Felder innerhalb der Tabelle auszumachen. Schwächen zeigt WIEN vor allem im Umgang mit fehlenden Werten oder variablen Parameterpositionen.[39]

RoadRunner dagegen stellt ein Wrapper Induction System dar, das gänzlich auf unbeaufsichtigten Lernalgorithmen beruht. Das System vergleicht die HTML-Struktur von Trainingsseiten desselben Typs auf ihre Ähnlichkeit und erzeugt daraus ein Schema der Daten, die in diesen Seiten vorhanden sind. Aus dieser Position heraus werden anschließend reguläre Ausdrücke zur Datenextraktion auf Basis von Ähnlichkeiten und Unterschieden der Beispielseiten erstellt.[40]

3.1.4 Visuelle Web Scraping Interfaces

Mit der Verbreitung des Web Scraping in Bereichen, die außerhalb der Forschung liegen, wuchsen schnell die Ansprüche, die Tools auch ohne ausgeprägtes Expertenwissen nutzen zu können, um sie zum Beispiel in der Wirtschaft oder in anderen Forschungsdisziplinen, wie der Sozialforschung anzuwenden.

Aus den ersten Web Scrapern, die in allgemeinen Programmiersprachen verfasst wurden, entwickelten sich spezifische Web-Scraping-Sprachen, die auf das Extrahieren von

[36] Vgl. ebd.; KAISER, Katharina / Miksch, Silvia (2005): Information Extraction. A survey. Technische Universität Wien, Wien. Institut für Softwaretechnik und Interaktive Systeme. S. 15.

[37] Vgl. CHANG, CHIA-HUI (2006): A Survey of Web Information Extraction Systems. S. 1419.

[38] Vgl. KAISER, Katharina (2005): Information Extraction. A survey. S. 15–17.

[39] Vgl. ebd. S. 17–18.

[40] Vgl. FIUMARA, Giacomo (2007): Automated Information Extraction from Web Sources: a Survey. In: Michigan State University (Hg.): Between Ontologies and Folksonomies Workshop. S. 4–5.

Daten zugeschnitten waren und schließlich Wizards, die, wie im *Abschnitt 3.2.1 Reguläre Ausdrücke* beschrieben, die Definition von Anfrageregeln automatisieren.[41] Im nächsten logischen Schritt wurden schließlich Web Scraping Tools mit visuellen Benutzeroberflächen entwickelt, in denen Laien nicht mehr mit spezifischen Konfigurationen wie der manuellen Definition von Anfragen oder regulären Ausdrücken in Kontakt kommen sollten.[42]

Die ersten Tools in diesem Bereich erschienen mit implementiertem Graphical User Interface (GUI), welche eine visuelle Interaktion mit der gerenderten Browserseite ermöglichen, in denen der Nutzer die Extraktionsziele auswählen und mit Operatoren der integrierten Extraktionstools verknüpfen kann. Je nach Scraper wird die Seite dabei in einer eigenen Software-Umgebung live gerendert oder die Extraktionstools werden als Extension in den Standardbrowser integriert. Diese Entwicklung machte die Methode des Web Scraping nun auch Nutzern zugänglich, die nur über Laienwissen im Bereich Web Data Extraction verfügen, und eröffnete damit die Möglichkeit des breiten kommerziellen Vertriebs von Web Scraping Tools. Den aktuellsten Entwicklungsschritt stellen Web Scraper mit einer Integrated Development Environment (IDE) dar. Die Funktionalitäten von IDEs gehen weit über die reine Darstellung der Webseite durch eine GUI hinaus, indem sie eine Deep Web Navigation ermöglichen. Der Nutzer kann zum Beispiel im gerenderten Browser Formulare auswählen und entsprechende Parameter, wie zum Beispiel Login-Daten festlegen, die der Wrapper nutzt, um in passwortgeschützte Bereiche der Website zu gelangen.[43]
LiXto ermöglicht es beispielsweise dem Nutzer, Extraktionsmuster interaktiv in der visualisierten Beispielseite zu definieren.[44]

3.1.5 Browserless Web Data Extraction

In einem 2018 veröffentlichten Konferenzpapier stellte sich eine Forschergruppe der University of Oxford unter Leitung von Ruslan R. Fayzrakhmanov der Frage, wie es zukünftig möglich wäre, Web Scraping noch zeiteffizienter zu gestalten. Die Problematik, aus der sich diese Idee entwickelt hat, basiert vor allem auf einer der Kernelemente der aktuellen Web Scraper: dem implementierten Browserfenster des visuellen Desktopinterfaces.

Da dieser Ansatz relativ neu ist und verwertbare weiterführende Literatur noch nicht identifiziert werden konnte, bezieht sich die folgende Darstellung ausschließlich auf das Ursprungswerk der Forscher, die die Methode begründet haben.

[41] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 311.

[42] Vgl. GLEZ-PEÑA (2014): Web scraping technologies in an API world. S. 791.

[43] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 310-311.

[44] Vgl. FIUMARA, Giacomo (2007): Automated Information Extraction from Web Sources: a Survey. S. 3.

Fayzrakhmanovs Team stellt in seinen Forschungsergebnissen fest, dass alleine das Rendern der Zielwebseite im integrierten Browser etwa 85%[45] der Laufzeit des Wrappers beansprucht, wohingegen die eigentliche Datenextraktion nur circa 2%[46] des Gesamtprozesses ausmacht. Gleichzeitig wird die Renderphase dadurch aufgebläht, dass mit dem Laden der Zielwebseite nicht nur das eigentliche Extraktionsziel, sondern alle Seiteninhalte inklusive Bilder, geschalteter Werbung, CSS-Stylesheets und vieler weiterer, für das Informationsbedürfnis irrelevanter Inhalte, gerendert werden. Dieser Prozess führt im Umkehrschluss zu der Erkenntnis, dass die extrem hohe Laufzeit, die das Rendern in Anspruch nimmt, größtenteils durch überflüssige Browserinhalte erzeugt wird.[47]

Gerade im wirtschaftlichen Kontext wird die Laufzeit der visuellen Wrapper, wie sie heutzutage am verbreitetsten sind, als maßgebliches Problem identifiziert. Echtzeitabfragen, wie auf Preisvergleichsportalen, bei denen sogar mehrere Wrapper gleichzeitig arbeiten, laufen nicht schnell genug, was sich direkt auf die Kundenzufriedenheit des entsprechenden Unternehmens auswirkt. Aus dieser Situation heraus wurden Data-Extraction-Ansätze entwickelt, die über HTTP-Requests direkt mit dem Web Server der jeweiligen Seite kommunizieren und gänzlich auf einen Renderprozess verzichten. Bei diesen Ansätzen werden die Serverresponses, die an den Browser zurückgeschickt werden, von Analysten untersucht und diejenigen Statements identifiziert, welche konkret die relevanten Daten liefern. Aus dieser Kommunikation heraus werden anschließend nur noch Requests gesendet, die auch datenrelevante Antworten liefern, sodass die Requests, welche normalerweise dazu dienen, die angesprochenen überflüssigen Inhalte zu laden, gar nicht erst mitgeschickt werden.[48]

Diese Maßnahmen führen dazu, dass der Wrapper auf Basis von HTTP-Requests eine bis zu 23,8[49] mal schnellere Laufzeit hervorbringen kann als ein Wrapper, der mit visuellen Browserimplementationen arbeitet und über den Browser kommuniziert.

Trotz der beachtlichen Erfolgsquote bringt das Erstellen eines HTTP-Wrappers einen entscheidenden Nachteil mit sich. Um die Browser-Server-Kommunikation zu analysieren und relevante Requests daraus zu extrahieren sowie die Serverresponses wiederum auf relevante Daten zu durchsuchen, erfordert es ein hohes Maß an Expertise durch geschulte Fachkräfte und einen hohen zeitlichen Aufwand, da der Wrapper händisch geschrieben werden muss. Fayzrakhmanov geht davon aus, dass das Entwickeln und Testen eines ausgereiften HTTP-Wrappers mehrere Tage in Anspruch nimmt, und da dieser individuell auf eine entsprechende Zielseite zugeschnitten ist, fällt der Wrapper

[45] FAYZRAKHMANOV, Ruslan R. / Sallinger, Emanuel / Spencer, Ben u.a. (2018): Browserless Web Data Extraction: Chal-lenges and Opportunities. In: Proceedings of the 2018 World Wide Web Conference, S. 1096.

[46] ebd.

[47] Vgl. ebd.

[48] Vgl. ebd.

[49] ebd.

weniger robust aus und Wartungskosten im Falle einer Strukturänderung der Zielseite wären sehr hoch.[50]

Motiviert durch diese Erkenntnis und das Optimierungsbedürfnis aus der realen Wirtschaft, hat sich das Team um Fayzrakhmanov zum Ziel gesetzt, einen Weg zu finden, HTTP-Wrapper automatisiert aus visuellen Wrappern zu erzeugen, um so die Vorteile beider Ansätze zu kombinieren und damit eine schnellere, robuste und gleichzeitig realistisch umsetzbare Web-Scraping-Methode zu entwickeln.

In dem Papier werden zwei maßgebliche Ziele zur Konstruktion eines browserlosen Web Scrapers herausgestellt. Das Hauptziel liegt darin, den Wrapper gänzlich auf simulierte Interaktionen zwischen Nutzer und Browser verzichten zu lassen und ihn stattdessen darauf zu programmieren, HTTP-Requests direkt an den Webserver zu senden. In zweiter Konsequenz sollen alle erfassten Requests darauf geprüft werden, ob die entsprechenden Responses des Servers für das Informationsbedürfnis relevante Daten liefern. Im Anschluss an diese Prüfung werden all die Requests entfernt, die nicht direkt eine Antwort mit relevanten Daten erhalten. Dieser Mechanismus minimiert die Menge der vom Wrapper gestellten Anfragen, der vom Server gelieferten Antworten und verhindert das Liefern von überflüssigen Browserinhalten, was zu einer Verschlankung der Wrapperprozesse und Reduzierung der gelieferten Daten auf das Wesentliche führt.[51]

Die Kernidee des Ansatzes zur automatisierten Wrappererzeugung ist, einen Mechanismus zu entwickeln, der den Wrapper eines bereits erstellten, visuellen Web Scrapers in einen browserlosen HTTP-Wrapper transformiert. Fayzrakhmanov kombiniert dadurch die Vorteile der visuellen Web Scraper mit der Effizienz des HTTP-Wrappers, indem das benutzerfreundliche Wrapperdesign und die Wartung über die Elemente des ersteren vorgenommen werden, die eigentliche Ausführung des Wrappers jedoch ausschließlich über HTTP-Requests läuft.[52]

Dieser Konverter wurde FastWrap getauft und verfolgt den Ansatz, die Interaktionen eines visuellen Wrappers auf http-Ebene nachzubauen. Fayzrakhmanov spricht dabei bei dem visuellen Wrapper vom sogenannten Input-Wrapper, wohingegen der zu erzeugende HTTP-Wrapper als Output-Wrapper bezeichnet wird. Der Output-Wrapper soll nach erfolgreicher Konvertierung durch direkte Interaktion mit dem Server der Zielwebseite exakt dieselben Daten wie der Input-Wrapper liefern, ohne dabei auf das Laden und Rendern der Seite selbst zurückzugreifen.[53]

Wie in Abbildung 2 zu sehen, wird dafür zunächst der Input Wrapper mit den für ihn vorgesehenen Parametern ausgeführt. FastWrap sammelt und analysiert die dadurch generierte Sequenz aus Requests des Wrappers und Responses des Servers, woraus sich ein Kommunikationsverlauf zwischen Client und Server ergibt. Innerhalb dieses

[50] Vgl. ebd.

[51] Vgl. ebd.

[52] Vgl. ebd.

[53] Vgl. ebd. S. 1097-1098.

Verlaufs kommen Responses vor, die relevante Daten in verschiedenen Formaten liefern. Ziel ist es, den Kommunikationsverlauf auf diese Statements hin zu analysieren und die relevanten Responses zu extrahieren. Aus diesen relevanten Responses werden dann die äquivalenten Requests abgeleitet, die wiederum zu einer neuen Sequenz zusammengefasst werden. Diese Sequenz bildet anschließend das Kommunikationsprotokoll für den neu generierten HTTP-Wrapper.[54]

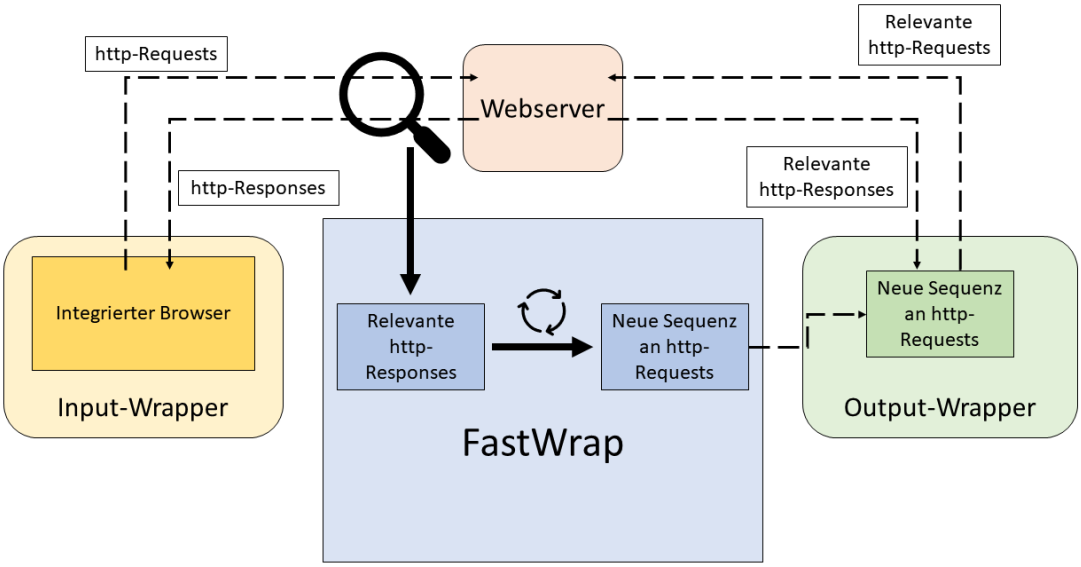

Abbildung 2: Schema des FastWrap-Konvertierungsvorgangs
(Quelle: Eigene Darstellung)

Die Forschungsergebnisse, die zum Zeitpunkt der Veröffentlichung vorlagen, sind unter dem Aspekt zu bewerten, dass für die Ausführungsphase des Wrappers Laborbedingungen geschaffen wurden, die einschränken, unter welchen Voraussetzungen der Wrapper transformierbar ist. Konkret sollen nur solche relevanten Inhalte als Zieldaten ausgewählt werden, die auf einer einzigen Seite auffindbar und nicht über mehrere Seiten einer Website verteilt sind. Fayzrakhmanov argumentiert, dass die überwiegende Mehrheit der Wrapper im industriellen Anwendungsbereich auf solche Single Target Pages ausgerichtet sind und so zum Testen der Funktionstüchtigkeit von FastWrap die Ausweitung der Grundgesamtheit auf Multi-Target Pages lediglich einen unnötigen Mehraufwand mit sich bringen würde. [55]

Eine weitere Einschränkung wird in Bezug auf die relevanten Daten selbst vorgenommen. So sollen als Zieldaten nur solche angesprochen werden, die eine flache Struktur aufweisen und damit ohne Verschachtelung und optionale Felder auskommen sowie nicht aus zusammengesetzten Datenwerten bestehen. Fayzrakhmanov spricht von flachen Datensätzen, merkt aber auch an, dass diese Einschränkung durch die

[54] Vgl. ebd. S. 1098-1099.

[55] Vgl. ebd. S. 1097.

Verwendung von Techniken zur Strukturidentifikation in Zukunft leicht aufgehoben werden kann.[56]

Aus Gründen der Vergleichbarkeit der Ergebnisse wurden für die Ausführung des Wrappers außerdem nur solche Webseiten als Zielseiten ausgewählt, die reproduzierbare Ergebnisse liefern. Webseiten mit variablen Inhalten, wie zum Beispiel Preisvergleichsportale, wurden bewusst aus der Grundgesamtheit ausgeschlossen.[57]

Schließlich schränkt Fayzrakhmanov die Grundgesamtheit noch einmal sehr spezifisch ein, indem solche Webseiten ausgeschlossen werden, welche die Google Maps Geocoding API nutzen. Der Grund dafür liegt darin, dass die API aktiv Anfragewerte in die Parameter der HTTP-Requests schreibt und damit den Wrapper in seiner Ausführung stören würde.[58]

Die beschriebenen Einschränkungen schaffen Bedingungen, unter denen der FastWrap-Ansatz vergleichbare Ergebnisse liefert und die Entwicklung und Optimierung der Methode in den Vordergrund stellt. Trotzdem sollten sie bei einer späteren Einschätzung des browserlosen Scraping-Ansatzes auf seine zukünftige potenzielle Tauglichkeit unter realen Bedingungen berücksichtigt werden.

3.2 Kategorisierungsansätze von Web Scraping Systemen

Forscher im Bereich der Web Data Extraction setzen immer wieder an, aktuelle Web Scraper einem Kategorisierungsschema unterzuordnen. Inzwischen gibt es dafür viele Ansätze, denen teilweise sehr unterschiedliche Ideen zugrunde liegen, andere liegen wiederum sehr nahe beieinander und lassen den Versuch eines Vergleichs beziehungsweise einer Verknüpfung zu.

Im folgenden Abschnitt werden drei unterschiedliche Kategorisierungsansätze vorgestellt. Ziel des Vergleichs ist es, die vielfältige Landschaft der Web Scraper unter verschiedenen Strukturen und aus unterschiedlichen Blickwinkeln zu betrachten und zu analysieren.

3.2.1 Kategorisierung nach Grad des Programmieraufwands

Daniel Glez-Peña geht bei der Unterteilung von Web Scrapern nach dem Aufwand, der sich für den Nutzer bei der Programmierung des Wrappers ergibt. Dabei fasst er Web-Scraping-Systeme in drei Kategorien:

Zum einen werden Systeme zusammengefasst, denen eine vollständige Programmierung des Wrappers in einer herkömmlichen Programmiersprache zugrunde liegt und lediglich mit der Unterstützung durch Codefragmente aus Programmbibliotheken

[56] Vgl. ebd.

[57] Vgl. ebd.

[58] Vgl. ebd.

geschrieben werden. Diese Herangehensweise bietet die Möglichkeit, den Scraper von Grund auf in der favorisierten Programmiersprache zu schreiben und auf ein sehr spezifisches Extraktionsziel zuzuschneiden. Spezielle Programmbibliotheken bieten Codefragmente, die Zugang zu den gewünschten Zielseiten erlauben, indem sie die Client-Ebene des HTTP-Protokolls in das Programm implementieren.[59]

Die Vorteile der Programmierung des Web Scrapers in einer allgemeinen Programmiersprache liegt in der großen Konstruktionsfreiheit und der Möglichkeit, maßgeschneiderte Wrapper für sehr spezifische Extraktionsziele zu schreiben. Die Nachteile dieser Herangehensweise finden sich in dem hohen Maß an erforderlichen Programmierkenntnissen, die diese Methode für Laien unzugänglich machen, sowie der Unflexibilität des Programms, sobald sich die zu scrapende Seite in ihrer Struktur ändert. Gänzlich von Hand geschriebene Scraper, die auf ein sehr spezifisches Informationsbedürfnis zugeschnitten wurden, sind extrem anfällig für jegliche Änderung in der Struktur der anvisierten Webseite und im Umkehrschluss sehr wartungsbedürftig, da bei jeder dieser strukturellen Änderungen der Programmcode händisch angepasst werden muss.[60]

Das Schreiben des Wrappers mithilfe von Frameworks, die auf eigens für die Programmierung von Wrappern eingeführten Sprachen beruhen, bieten eine Vereinfachung von Bibliotheken, die dynamischer auf Veränderungen in HTML-Strukturen reagieren können. Mit Frameworks können dem Scraper z.B. prozedurale Anweisungen, Loops und variable Definitionen mitgegeben werden, um auf solche Veränderungen reagieren zu können.[61]

Full Service Scraper, die in eine Desktop-basierte Umgebung eingebettet sind, bieten eine grafische Bedienoberfläche mit integriertem Browser, die es dem Nutzer ermöglicht, interaktiv zu scrapende Elemente einer Webseite auszuwählen und mit Befehlsmodulen vorzugeben, wie mit den entsprechenden Elementen umgegangen werden soll. Dieses Modell zeichnet sich durch seine hohe Benutzerfreundlichkeit aus, der Nutzer selbst muss oft keinerlei Programmierkenntnisse mitbringen. Die Software selbst ist aber oft nur kommerziell zu erwerben und schwer in andere Applikationen zu integrieren, da sie bereits selbst ein in sich geschlossenes System anbieten.[62]

3.2.2 Kategorisierung nach Grad der technologischen Entwicklung

Ein weiterer Kategorisierungsansatz, der 2014 von Emilio Ferrara vorgestellt wurde, betrachtet Web Scraping unter dem Aspekt der technologischen Entwicklung unter Berücksichtigung von sechs Schichtmodellen, auf denen der entsprechende Fortschritt der Wrapperentwicklung beschrieben wird. Die Problematiken, welche die Modelle betrachten, beziehen sich dabei hauptsächlich auf Aspekte der Wrappererstellung und -

[59] Vgl. GLEZ-PEÑA (2014): Web scraping technologies in an API world. S. 790.

[60] Vgl. ebd. S. 791.

[61] Vgl. ebd.

[62] Vgl. ebd.

bedienung, der Begrenztheit in Bezug auf extrahierbare Daten und der Möglichkeiten im Post Processing.

Ein Schichtmodell, das in diesem Ansatz betrachtet wird, befasst sich mit der Einfachheit der Wrappererstellung und weist ein hohes Maß an Schnittpunkten mit dem Kategorisierungsansatz Glez-Peñas auf. Die technische Weiterentwicklung von Web-Scraping-Systemen wird hier äquivalent zum zuvor beschriebenen Ansatz beschrieben, beginnend mit dem Programmieren von Wrappern in gängigen Sprachen über spezielle Wrapper-Programmiersprachen und Wizards, die bei der Erstellung von Extraktionsregeln unterstützend agieren, bis hin zu Web Scraping Systemen mit integrierten visuellen Interfaces und Entwicklungsumgebungen auf der obersten Hierarchieebene.[63]

Im zweiten Modell wird die Frage aufgegriffen, welche Mittel das Web Scraping System dem Nutzer zur Verfügung stellt, um einen Wrapper mit robusten Extraktionsregeln zu erstellen. Dabei findet sich die Einfachheit der Bedienung aus der vorangegangenen Betrachtung als Maß der technologischen Entwicklung in diesem Ansatz wieder und die vorgestellten Mittel haben die Benutzerfreundlichkeit bei der Erstellung von Regeln als ein wichtiges Bewertungskriterium im Fokus. Der primitivste technische Ansatz ist, die Anfragen manuell zu erstellen und jeweils einzeln an Beispielseiten zu testen. Da diese Methode aber gleichzeitig die für den Nutzer aufwändigste ist, wurden folgende Web-Scraping-Systeme in Konsequenz mit systemgestützter Anfrageprogrammierung ausgestattet, die es ermöglicht, den Nutzer bei der Erstellung von Extraktionsregeln beispielsweise durch Keyword Highlighting oder Autovervollständigungsmechanismen zu entlasten. Auch bei der manuellen Erstellung von Wrappern mithilfe von Programmiersprachen hielten Extensions Einzug, die es ermöglichten, den Programmierer mit Debugging-Tools oder visuellen Verkörperungen/Abstraktionen von Programmkonstrukten zu unterstützen.[64]

Die Möglichkeiten der Deep Web Navigation spielen vor allem seit der Entwicklung des Internet zum Web 2.0 und der Möglichkeit, Webseiten dynamisch zu gestalten, eine immer wichtigere Rolle. Daher betrachtet ein Schichtmodell Ferraras exakt diesen Aspekt. Neben dem einfachen Navigieren durch eine Abfolge von festgelegten Links bietet das automatische Ausfüllen von Formluaren auf Basis von analysierten Client/Server-Kommunikationen eine grundlegende Funktion der Deep Web Navigation. Weiter entwickelte Web Scraper sind in der Lage, skriptbasierte DOM-Events zu analysieren und nachzuahmen. Die höchste Stufe der Deep Web Navigation findet sich in Wrappern, die in der Lage sind, Klick-Events, die der Nutzer vorgegeben hat, zu speichern und äquivalent zum Navigationsverhalten eines Menschen abzuspielen.[65]

Mit dem Vergleichsmodell, das die Möglichkeiten der Web Data Extraction betrachtet, wird die Kernkompetenz eines Web Scrapers aufgegriffen und in ein skalierbares Muster

Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 311.

[64] Vgl. ebd.

[65] Vgl. ebd.

gebracht. Die Methode der untersten Hierarchiestufe, die ein Web Scraper hier nutzen kann, ist das Verarbeiten der einfachen Textstrings, die vom Webserver zurückgeschickt wurden, beispielsweise mithilfe der zuvor beschriebenen regulären Ausdrücke. In der nächsthöheren Stufe stehen Scraper, deren Wrapper die HTML- beziehungsweise DOM-Struktur nutzen, um relevante Inhalte zu identifizieren.[66]

Im vorletzten Modell wird die technische Umsetzung des Parsers und die Einbindung eines speziellen oder herkömmlichen Webbrowsers betrachtet. Während auf der einfachsten Stufe keine Browserimplementierung vorgesehen ist und selbsterstellte Parser eingesetzt werden, die lediglich die HTML-Struktur auf Tag-Ebene analysieren, greifen weiterentwickelte Systeme auf sogenannte DOM-Bibliotheken zurück. Spezielle Browser bieten eine Möglichkeit, in Kombination mit den passenden Frameworks genutzt zu werden, wohingegen reguläre Browser wie Firefox oder Chrome den Vorteil bieten, neben der DOM-Analyse auch Strukturen wie das CSS-Stylesheet anbieten zu können, um darüber die Webseite zu parsen. Andere Web Scraper greifen das Problem von einer ganz anderen Seite her auf und werden direkt als Browser Extension angeboten.[67]

Die Komplexität von unterstützten zusätzlichen Tools befasst sich mit der Frage, welche Services ein Web Scraper implementiert hat, um den Extraktionsvorgang selbst, aber auch vor- und nachgelagerte Prozesse zu optimieren. Sogenannte Batch-Processing-Implementationen ermöglichen das wiederholte Ausführen bestimmter Extraktionsaufgaben mit variablen Parameterwerten, was zum Beispiel beim automatischen Ausfüllen von Webformularen genutzt werden kann. Hochentwickelte Scraper bieten beispielsweise Tools zur zeitlichen Planung und Anonymisierung der Extraktionen selbst, um die Zielseiten nicht mit zu vielen gleichzeitigen Requests zu überlasten und einen reibungslosen Extraktionsvorgang zu gewährleisten. Gleichzeitig bieten solche Systeme oft auch Konnektoren zu nachgelagerter Business-Intelligence-Software.[68]

Ferraras Ansatz bietet die Möglichkeit, Web-Scraping-Systeme, je nachdem, auf welchem Entwicklungsstand sie sich auf den jeweiligen Schichtmodellen befinden, objektiv nach ihren technischen Fähigkeiten und der Nähe zum aktuellen Stand der Technik zu kategorisieren und vergleichen zu können. Das Modell beschreibt dabei sehr detailliert die verschiedenen Möglichkeiten, nach welchen Methoden Web-Scraping-Systeme unter bestimmten Gesichtspunkten aufgebaut werden können, erfordert gleichzeitig aber ein tieferes technisches Verständnis der Materie, sobald auf einzelne Techniken genauer eingegangen wird.

[66] Vgl. ebd. S. 311–312.

[67] Vgl. ebd. S. 312.

[68] Vgl. ebd.

3.2.3 Dreidimensionale Kategorisierung von Wrapper-Induction-Systemen

Chia-Hui Chang teilt Web-Information-Extraction-Systeme in drei Dimensionen auf. Dabei beschränkt sich die Evaluation allerdings exklusiv auf Wrapper-Induction-Systeme.[69]

Die erste Dimension beschreibt die Schwierigkeit der Aufgabe, mit der ein Web Scraper konfrontiert wird und befasst sich im Kern mit der Frage, warum ein System daran scheitert, Webseiten mit einer bestimmten Struktur erfolgreich zu scrapen. Dabei folgt die Evaluierung der Schwierigkeit auf zwei separaten Ebenen.[70]

Auf der ersten Ebene wird die Schwierigkeit der Aufgabe nach der Struktur des Eingabedokuments klassifiziert. Dieser Überlegung geht die anfängliche Feststellung voran, dass Dokumente, die ein Web Scraper betrachten soll, unterschiedlich strukturiert sein können. Chang legt dafür eine eigene Skala an, die sich nach den Anforderungen eines Web Scrapers richtet. Innerhalb dieser Skala gelten XML-Dokumente und Inhalte einer Datenbank als strukturierte Daten, da ihnen ein XML-Schema oder generell eine Dokumenttypdeklaration zugrunde liegt, welche die Inhalte beschreiben. Ein HTML-Dokument ist semi-strukturiert, da die enthaltenen Daten zwar durch Benutzung von HTML-Tags in einen einheitlichen Rahmen gebracht werden, der vom Wrapper angesprochen werden kann, diese Tags aber nichts über ihren Inhalt aussagen. Zuletzt werden Freitextdokumente betrachtet, die nur aus Stringelementen bestehen und denen keine übergeordnete Struktur zugrunde liegt. Diese Dokumente können vom Wrapper nur über Natural-Language-Processing-Verfahren angesprochen werden und gelten daher als unstrukturierte Dokumente.[71]

Aus dieser Einstufung ergibt sich die Feststellung, dass sich die Schwierigkeit der Extraktionsaufgabe unter anderem aus der Beschaffenheit des Eingabedokuments ergibt. Je strukturierter das Dokument ist, desto einfacher ist es für den Wrapper, dieses zu erfassen.

Die zweite Ebene befasst sich mit dem Extraktionsziel selbst, wobei hier noch einmal zwischen der strukturellen Beschaffenheit des Extraktionsziels und dem Status, die ein Attribut haben kann, unterschieden wird. Die Struktur des Extraktionsziels wird mithilfe eines hierarchischen Baumdiagramms verdeutlicht, welches flach oder geschachtelt ausfallen kann. Weist das Ziel eine flache Struktur auf, besteht es nur aus dem Wurzelknoten und maximal zwei Blattknoten. In einer geschachtelten Struktur verzweigt sich der Baum über mehr als diese zwei Ebenen und hat mehrere interne Knoten. Je verschachtelter ein Objekt ist, desto schwieriger wird es für den Wrapper, dieses vollständig zu erfassen.[72]

[69] Vgl. CHANG, CHIA-HUI (2006): A Survey of Web Information Extraction Systems. S. 1412.

[70] Vgl. ebd. S. 1413

[71] Vgl. ebd.

[72] Vgl. ebd.

Darüber hinaus können die Attribute eines Extraktionsziels Wertevariationen haben, mit denen der Wrapper unter Umständen nicht umgehen kann. So kann ein Attribut beispielsweise keinen oder gleich mehrere Werte. Weitere Beispiele für Permutationen eines Attributs sind die Möglichkeit, dass dieses bei mehreren Dokumenten an verschiedenen Positionen steht oder ein und dasselbe Attribut in verschiedenen Dokumenten unterschiedlich formatiert ist beziehungsweise unterschiedliche Attribute, wie zum Beispiel Tabellen, das gleiche Format aufweisen.[73]

Die Kombination aus den beschriebenen Varianten, wie ein Extraktionsziel beschaffen sein kann, und den verschieden strukturierten Eingabedokumenten erzeugen unterschiedliche Grade der Aufgabenschwierigkeit und klassifizieren Wrappersysteme über die Fähigkeit, mit solchen Variationen umgehen zu können.

In der zweiten Dimension betrachtet Chang die Technik, die von einem Web Scraper genutzt wird, um die gewünschten Daten zu extrahieren. Vorab wird jedoch herausgestellt, dass diese Dimension keine wirkliche Möglichkeit der Wrapperevaluation bietet, sondern nur dazu dient, die Vielfalt der möglichen Herangehensweisen darzustellen.[74] Dafür wird der gesamte Extraktionsprozess auf drei Arbeitsschritte heruntergebrochen und anschließend die Variationen innerhalb der ersten beiden Schritte beschrieben.

Im ersten Schritt, bei dem die Input-Strings in einzelne Tokens aufgebrochen werden, bedienen sich Wrapper verschiedener Tokenisierungs-Schemata, je nachdem, auf welche Art Input-String sie spezialisiert sind. Die Bandbreite reicht von Tokenisierung auf Wortebene, bei der jedes Wort eines Dokuments als eigenständiges Token übersetzt wird, hin zur Tokenisierung auf Tag-Ebene, die speziell auf strukturierte und semi-strukturierte Webdokumente zugeschnitten ist und jedes HTML-Tag als Token sowie die Strings zwischen den Tags als Special Token übersetzt.[75]

Bei der Wahl der Verfahren zur Erstellung von Extraktionsregeln steht den Tools ebenfalls eine Bandbreite an Möglichkeiten zur Verfügung, die entsprechend der Beschaffenheit des Extraktionsziels gewählt werden sollten. Neben den beschriebenen regulären Ausdrücken für reine Stringabfragen und den logischen Programmiersprachen, gibt es semantische und syntaktische Ansätze, Baumstrukturansätze und viele mehr.[76]

Zusammenfassend werden fünf Ebenen zum Vergleich der genutzten Technik herausgestellt. Neben den bereits angesprochenen Tokenisierungsschemata und den vielfältigen Arten, Extraktionsregeln zu erstellen, stehen die benötigte Anzahl an Scandurchläufen der Zieldokumente, bis dieses vollständig erfasst wurde, die Features, welche die

[73] Vgl. ebd. S. 1414.

[74] Vgl. ebd. S. 1413.

[75] Vgl. ebd. S. 1414.

[76] Vgl. ebd.

Extraktionsregeln spezialisieren und die genutzten Lernalgorithmen im Fokus dieser Betrachtung.[77]

Zuletzt wird mit der dritten Dimension bewertet, wie hoch der Automationsgrad des entsprechenden Wrappers anzusehen ist. Dabei wird die Dimensionierung vor allem auf der Ebene der Belabelung von Trainingsdaten durch den Nutzer vorgenommen. Chang betrachtet hierbei die Nutzerexpertise, die benötigt wird, um Testseiten zu belabeln und Extraktionsregeln manuell zu erstellen. Der Automationsgrad wird dadurch definiert, ob ein Wrapper völlig auf händisch belabelte Trainingsseiten angewiesen ist, das Belabeln nicht zwingend zur Erstellung von Extraktionsregeln erforderlich ist und damit im Anschluss erfolgen kann oder ein Wrapper in der Lernphase völlig ohne Trainingsbeispiele auskommt. Der Preis für diese automatisierte Regelerstellung ist, dass der fertige Wrapper, je höher der Grad der Automation ist, umso spezifischer auf den zuvor definierten Aufgabenbereich festgelegt ist und damit die Kompatibilität mit anders gestalteten Aufgaben in Mitleidenschaft gezogen wird.[78]

Changs Kategorisierungsansatz ist klar in drei unabhängig voneinander zu betrachtende Dimensionen unterteilt, innerhalb derer bestimmte Vergleichspunkte festgelegt wurden. Trotz der Tatsache, dass die Dimensionierung für Wrapper-Induction-Systeme vorgesehen ist, lassen die beiden ersten Dimensionen eine Vergleichsmöglichkeit unter herkömmlichen Web Scrapern ebenfalls zu. Würde man in der dritten Dimension eine weitere Hierarchiestufe implementieren, die unter vollbeaufsichtigten Systemen steht, bestünde die Möglichkeit, Systeme ohne Wrapper Induction dieser untersten Ebene zuzuordnen und hätte damit die Kategorisierung für alle Web-Scraping-Ansätze geöffnet.

[77] Vgl. ebd.

[78] Vgl. ebd.

4 Anwendungsfelder des Web Scraping

Web Scraping hat eine vielfältige Palette an möglichen Anwendungsfeldern. Im vorangegangenen Text wurde dieses Thema bereits an verschiedenen Stellen angeschnitten. So kann es von Unternehmen genutzt werden, um die Reputation des eigenen Produkts auf Social-Media-Kanälen zu ermitteln, Trends im Markt und Tendenzen der Zielgruppe zu identifizieren oder Web-sites von Wettbewerbern langfristig zu beobachten.

Im folgenden Abschnitt sollen konkrete Beispiele genannt werden, in denen Web Scraping zur Anwendung kommt, wobei die Bereiche in einen wissenschaftlichen und einen wirtschaftlichen Teil aufgeteilt werden.

4.1 Anwendung in der Wirtschaft

4.1.1 Business Intelligence, Markt- und Wettbewerbsanalyse

In den heutigen Märkten, die sich immer rasanter entwickeln und immer stärker vernetzen, ist es für ein Unternehmen überlebenswichtig, den Markt und die darin agierenden Wettbewerber genau im Blick zu haben.

Extracting, Transforming and Loading-Software (ETL) nutzt dabei intern vorhandene Daten eines Unternehmens, um diese für Business Intelligence Data Warehouses aufzubereiten und entsprechende Handlungsempfehlungen zur Prozessoptimierung abzuleiten.[79]

Doch in diesem Szenario wird nur die eigene Unternehmenssituation betrachtet und das Bedürfnis, aus externen Daten Wettbewerbsinformationen zu gewinnen, ist immer größer geworden. Web Scraper können hier als Tool fungieren und legal zugängliche Wettbewerbsdaten wie Jahresabschlussberichte, Pressemeldungen oder generell Webauftritte und Social-Media-Kanäle von Wettbewerbern automatisiert nach relevanten Daten durchsuchen, diese extrahieren und in einem entsprechenden Format in nachgelagerte Business-Intelligence-Software, wie Microsoft Analysis Software oder SAP beziehungsweise Data Warehouses speisen.[80]

Die Maximen des Extraktionsprozesses im Bereich der Business Intelligence beruhen dabei auf Skalierbarkeit und Effizienz und fordern von einem Web Scraping System, so viele relevante Daten wie möglich in einem möglichst kurzen Zeitraum und unter Aufwand von so wenig Ressourcen wie möglich zu sammeln.[81] Gerade die letzten beiden Aspekte sind im Sinne des unternehmerischen Denkens von besonderer Bedeutung, da ein Web Scraper seinen Vorteil verliert, sobald die Daten weniger Wert sind als der finanzielle und zeitliche Aufwand, welcher in sie investiert wurde.

[79] Vgl. BAUMGARTNER, R. (2009): Web data extraction system. S. 3469.

[80] Vgl. ebd.; Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 314.

[81] Vgl. ebd.

Besondere Beispiele für ein solches Nutzen von Web Scraping zur Markt- und Wettbewerbsanalyse findet man bei Banken und Investmentfirmen. Erstere scrapen die Seiten von Wettbewerbern, um beispielsweise deren aktuelle Zinssätze tracken zu können oder das jeweilige Filialnetzwerk der Konkurrenz zu analysieren. So können die eigene Zinspolitik bewertet oder Lücken im Filialnetzwerk genutzt werden, um diese mit eigenen Niederlassungen zu schließen. Investmentfirmen nutzen Web Scraping, um Nachrichten zu Branchen oder Produkten aus ihrem Portfolio beobachten und frühzeitig auf etwaige Krisen, die den Kurs ihrer Produkte beeinflussen könnten, reagieren zu können.[82]

4.1.2 Preis- und Produktabfragen durch Vergleichsportale

Mit dem rasanten Wachstum des e-Commerce-Sektors, aus dem sich wenige dominante Marktführer wie Amazon herausgebildet haben, sind Serviceportale, die Vergleichsmöglichkeiten für Produkte und Produktpreise anbieten, ein weiteres Anwendungsfeld für Web-Scraping-Systeme geworden. Neben dem reinen Preisvergleich können mithilfe von Web Scraping auch Produktfunktionen, technische Daten oder Nutzerreviews aus verschiedenen Quellen zusammengetragen und zu einer Vergleichsansicht integriert werden.[83]

Ein bekanntes Beispiel für eine solche Anwendung sind Preisvergleichsportale für Flüge. Der Nutzer gibt Keywords wie Datum, Startflughafen, Zielflughafen oder Preisspannen in die Suchmaske des Portals ein und das Portal ermittelt passende Flugangebote von externen Quellen. Web Scraping kann dabei zum Einsatz kommen, wenn Fluganbieter keine entsprechende API anbieten, indem beispielsweise mithilfe der Struktur und der Stringabfolge der Keywords passende Angebote gesammelt und extrahiert werden können.

4.2 Anwendung in der Wissenschaft

4.2.1 Datenextraktion in der Bioinformatik

In der biomedizinischen Datenextraktion kommt es immer wieder vor, dass Datenbanken keine API zur Verfügung stellen oder die vorhandenen Web Services das Informationsbedürfnis des Nutzers nicht ausreichend stillen können. Daher hat sich Web Scraping in verschiedenen bioinformatischen Szenarien als Methode zur Datenextraktion etabliert. Im Folgenden soll einer dieser Fälle genauer beschrieben werden.

Das Beispiel befasst sich mit dem Ansatz, dass mit den wachsenden Resistenzen mikrobakterieller Erreger gegen herkömmliche Behandlungen ein kontinuierliches Screening und Testen antibakterieller Wirkstoffe nötig ist, um passende Therapeutika zu ermitteln. Um ein dafür erforderliches ausführliches Portfolio eines Wirkstoffs erstellen zu können, müssen möglichst viele vorhandene Daten aus mehreren Quellen extrahiert und

[82] Vgl. BAESENS, Bart (2018): Practical Web Scraping for Data Science: Best Practices and Examples with Python. S. 7.

[83] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. 315.

zusammengetragen werden. Mit speziell dafür konzipierten Web Services könnte eine schnelle und konstante Lösung für diese Aufgabe geschaffen sein, doch in der Realität bieten die meisten relevanten Datenbanken in diesem Bereich keine Schnittstellen zur Implementierung eines solchen Web Services.[84]

Mithilfe eines Web Scrapers können biomedizinische Datenbanken sowohl nach passenden Antibiotika als auch nach zugehörigen Informationen, wie der minimale Konzentration des Stoffes zur wirksamen Bekämpfung des Erregers, dem pflanzlichen Ursprung, bibliographischen Referenzen oder der Bandbreite an Wirksamkeiten gegen Erreger, durchsucht werden. Anschließend lassen sich die gesammelten Daten zu einem umfassenden Portfolio zusammenführen, wodurch eine breite Basis zur Wirkungsforschung geschaffen ist.[85]

Gerade in der Biomedizin bieten standardisierte Formatierungscodes von bestimmten Fachdaten einen dankbaren Ansatz für Web Scraper, um die Seiten nach den entsprechenden Formaten zu durchsuchen.

4.2.2 Social Media Harvesting

Social-Media-Plattformen werden heutzutage so intensiv genutzt, dass sich aus dem Potenzial, welches sich aus den erzeugten Nutzerdaten entwickelt hat, neue Forschungsmöglichkeiten ergeben haben. Mithilfe dieses riesigen Pools an Interaktionsdaten zwischen Menschen können heute Antworten auf Fragen der Sozialforschung, zum Beispiel wie sich menschliche Beziehungen entwickeln oder wie sich Nachrichten im Social Web verbreiten, beantwortet werden.[86]

Grundsätzlich bieten die meisten großen Social-Media-Plattformen Schnittstellen zur Extraktion und Auswertung ihrer Daten an, doch gerade Facebook hat im Zuge der Cambridge-Analytica-Affäre, in der Nutzerdaten für Wahlkampfzwecke der US-Wahlen 2016 missbraucht wurden, die Nutzung dieser Daten über ihre API stark eingeschränkt.[87]

Daher wurde Web Scraping als Mittel zur Extraktion von Social-Media-Daten immer probater. Allerdings sieht sich die Methode auch mit großen Herausforderungen konfrontiert. Besonders Social-Media-Plattformen weisen ein hohes Maß an dynamischen Veränderungen in Inhalten und Struktur der einzelnen Seiten auf. Web Scraper in diesem Bereich müssen also besonders flexibel und gleichzeitig in der Lage sein, eine große Menge an Daten in kurzer Zeit zu sammeln.[88]

[84] Vgl. GLEZ-PEÑA (2014): Web scraping technologies in an API world. S. 792–793.

[85] Vgl. ebd. S. 793.

[86] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 316.

[87] Vgl. GOLLA, Dr. Sebastian J. / v. Schönfeld, Dr. Max (2019): Kratzen und Schürfen im Datenmilieu – Web Scraping in sozialen Netzwerken zu wissenschaftlichen Forschungszwecken. In: Kommunikation & Recht (2019), Nr. 1/2019, S. 15.

[88] Vgl. FERRARA, Emilio (2014): Web data extraction, applications and techniques: A survey. S. 317.

Ein Beispiel für diese Art der Verhaltensanalyse erbrachte die kanadische Firma SAS. Sie analysierte Nachrichten und Posts aus Social-Media-Kanälen wie Twitter auf unterschwellige Aussagen. Aus diesen Auswertungen konnte anschließend ein Vorhersagemodell für suizidale Tendenzen bei Nutzern dieser Kanäle entwickelt werden.[89]

[89] Vgl. BAESENS, Bart (2018): Practical Web Scraping for Data Science: Best Practices and Examples with Python. S. 6.

5 Abschließendes Resümee

Zum Abschluss dieser Arbeit soll noch einmal ein Überblick über die gewonnenen Erkenntnisse der einzelnen Kapitel gegeben werden und die zentralen Forschungsfragen aufgegriffen und zusammenfassend beantwortet werden.

Anschließend wird eine persönliche Einschätzung auf Basis dieser Schlüsse abgegeben, die sich damit beschäftigt, wie eine mögliche zukünftige Entwicklungstendenz in der Web Data Extraction aussehen könnte.

5.1 Inwiefern unterscheidet sich Web Scraping von anderen webbasierten Informationsextraktionsverfahren, wie zum Beispiel Web APIs?

Die heutigen Möglichkeiten, über APIs serverseitig an relevante Daten zu gelangen, und deren Extraktionsprozess damit nicht von Strukturänderungen der Zielwebseite betroffen ist, sind vielfältig und weit verbreitet.

Dennoch zeigen die besprochenen Beispiele und Szenarien, dass trotz der API-dominanten Datenwelt Web Scraping immer noch seine Existenzberechtigung als Web-Data-Extraction-Methode hat. Die vielfältigen Individualisierungsmöglichkeiten und die Bandbreite an Techniken, welche es ermöglichen, einen Wrapper zu erstellen, der optimal auf die Zielwebseite zugeschnitten ist und die Daten in genau der Form liefert, die gewünscht ist, machen Web Scraper zu einem mächtigen Tool. Zusätzlich zielen die Bestrebungen in der Entwicklung von Web Scrapern immer weiter auf Benutzerfreundlichkeit und Automatisierung ab, wodurch sich die Tools einer größeren Bandbreite an Nutzern öffnen und damit ein wichtiges Werkzeug in Wirtschaft und Forschung geworden sind.

5.2 Welche aktuellen Web-Scraping-Verfahren gibt es und worin liegen die Vor- bzw. Nachteile dieser?

Die Vielfalt an Anwendungsszenarien für Web Scraping bringt auch eine breite Palette an Methoden und Techniken hervor, mit denen ein Wrapper erstellt werden kann. Je nachdem, welches Extraktionsziel verfolgt wird, wie umfangreich die Ergebnisdaten ausfallen sollen, welcher Zeitaufrahmen zur Verfügung steht und die Beschaffenheit der Zielseiten bestimmte Extraktionsmethoden ausschließen, müssen die Web Scraper entsprechend aufgebaut werden.

Dabei stehen dem Nutzer sowohl grundständige Techniken auf Basis von Zeichenkettenanalysen, wie die Erstellung von regulären Ausdrücken, als auch voll automatisierte Wrappererstellungssysteme auf Basis von Machine-Learning-Ansätzen zur Verfügung.

Mit der browserlosen Web Data Extraction wird eine Extraktionsmethode weiterentwickelt, die bei der Erstellung zukünftiger Web Scraper eine wichtige Rolle spielen könnte. Der FastWrap-Ansatz kombiniert die Vorzüge der Wrappererstellung und -wartung über visuelle Interfaces mit der Effizienz eines Wrappers, der direkt über HTTP-Requests mit

dem Server der Webseite kommuniziert. Besonders im wirtschaftlichen Kontext, in dem Echtzeitprozesse gewinnentscheidend sein können, aber gleichzeitig visuelle Full-Service Suites mit Schnittstelltenanbindung an weiterverarbeitende Prozesssoftware der Standard sind, könnte diese Kombination zukünftig großes Interesse wecken. Dafür muss es jedoch zukünftig Ziel der weiteren Entwicklung sein, daran zu arbeiten, die Limitationen, die zur Erprobung des FastWrap-Konverters geschaffen wurden, zu eliminieren, um den Ansatz für reale Szenarien tauglich zu machen.

Viele aktuelle Web-Scraping-Systeme setzen auf benutzerfreundliche Umgebungen, die eine Expertise und einen möglichst hohen Automationsgrad in der Wrappererstellung nicht benötigen. Besonders letzteres gewinnt im Zusammenhang mit dem stetig wachsenden Datenvolumen im Internet und der immer tiefergehenden Verschachtelung von Webseiten stark an Bedeutung.

5.3 Unter welchen Aspekten lassen sich die vorgestellten Web-Scraping-Verfahren kategorisieren?

Daniel Glez-Peña, Emilio Ferrara und Chia-Hui Chang präsentieren jeweils sehr unterschiedliche Sichtweisen bei der Betrachtung und Kategorisierung von Web-Scraping-Systemen. Während Glez-Peña den Ansatz hat, die Scraper eindimensional und unter dem Blickwinkel des Programmieraufwands zu betrachten, findet sich dieser Aspekt im Schichtmodellansatz von Ferrara in nur einem von insgesamt sechs Modellen wieder. Diese wesentlich komplexere Betrachtungsweise ordnet Web Scraper nach ihrem technologischen Entwicklungsstand innerhalb dieser sechs Schichtmodelle, sodass ein System mehrdimensional betrachtet werden kann. Zuletzt wurde mit Changs dreidimensionalem Modell ein sehr spezifischer Kategorisierungsansatz vorgestellt, der Wrapper-Induction-Systeme auf Basis der Schwierigkeit der Extraktionsaufgabe, der verwendeten Techniken und des Automationsgrads ordnet. Vor allem letztere Dimension spielt dabei in Bezug auf den Machine-Learning-Ansatz von Wrapper-Induction-Systemen eine wichtige Rolle.

5.4 Wo liegen die aktuellen Anwendungsfelder des Web Scraping?

Diese Arbeit hat im letzten Kapitel die vielfältigen Möglichkeiten der Anwendung von Web-Scraping-Tools anhand von ausgewählten Beispielen beleuchtet. Dabei ist Web Scraping nicht auf ein bestimmtes Feld beschränkt, sondern kann sowohl im Forschungskontext als Werkzeug zur Extraktion von relevanten Forschungsdaten, als auch im wirtschaftlichen Bereich als mächtiges Tool zur Gewinnung von Markt- und Wettbewerbsdaten als Ressourcen für Prozessentscheidungssoftware und damit letztendlich als Werkzeug zur Steuerung eines Unternehmens angewandt werden.

Gerade die bereits angesprochene Entwicklung der Tools hin zu benutzerfreundlicher Handhabe und die Vielfalt an möglichen Erstellungsansätzen dienen dazu, Web

Scraping einer Vielzahl an potentiellen Nutzern in verschiedenen Anwendungsbereichen zu öffnen.

5.5 Persönliche Einschätzung

Im Zuge der Erörterungen dieser Arbeit lässt sich feststellen, dass Web Scraping eine elementare Methode der Extraktion von relevanten Daten aus dem Web ist und zukünftig auch bleiben wird. Die Entwicklungen in diesem Forschungsfeld folgen dem allgemeinen Trend aktueller Technik, hin zu mehr Automatisierung und einer höheren Benutzerfreundlichkeit, um Web Scraping effizient für die unterschiedlichsten Anwendungsfelder zu gestalten. Die Vielzahl an unterschiedlichen Strukturen und Beschaffenheiten der Extraktionsziele im Internet wird mit ebenso vielfältigen Möglichkeiten der Gestaltung von Tools beantwortet.

Die browserlosen HTTP-Wrapper als zukünftige Methode der Web Data Extraction umgehen in der eigentlichen Ausführung zwar das „Abkratzen“ der Inhalte aus dem Browser, kommen aber nicht darum herum, Web Scraping Tools als Basis der Wrappererstellung und -wartung zu integrieren. Letztendlich können solche Ansätze dazu beitragen, dass Web Scraping in Kombination mit anderen Web Data Extraction Verfahren noch effizientere Tools hervorbringt.

Besonders im wirtschaftlichen Kontext ist Web Scraping als Werkzeug zur Erzeugung von Wettbewerbsvorteilen durch Business Intelligence und im wissenschaftlichen Kontext zum gezielten Sammeln von strukturierten Massendaten aus semi-strukturierten Webquellen nicht wegzudenken.

Literaturverzeichnis

BAESENS, Bart / vanden Broucke, Seppe (2018): Practical Web Scraping for Data Science: Best Practices and Examples with Python. Apress, New York.

BARANOVSKIY, Evgeny (2011): Methodik zur automatisierten Extraktion und Klassifikation semistrukturierter Produkt- und Adressdaten aus Webseiten. Stuttgart, Universität Stuttgart [Diplomarbeit].

BAUMGARTNER, R. / Gatterbauer, W. / Gottlob, G. (2009): Web data extraction system. In: Encyclopedia of Database Systems (5), S. 3465–3471.

CHANG, CHIA-HUI / Kayed, M. / Girgis, M. R. / Shaalan, K. F. (2006): A Survey of Web Information Extraction Systems. In: IEEE Transactions on Knowledge and Data Engineering, 18(10), S. 1411–1428 https://doi.org/10.1109/TKDE.2006.152.

DYCK, Andreas (12.03.2019): Wie das Internet das Surfen lernte. In: general-anzeiger-bonn.de, URL: http://www.general-anzeiger-bonn.de/news/digitale-welt/Wie-das-Internet-das-Surfen-lernte-article4065601.html, letzter Aufruf: 27.08.2019.

EBERSBACH, Anja / Glaser, Markus / Heigl, Richard (2011): Social Web. UVK, Konstanz.

FAYZRAKHMANOV, Ruslan R. / Sallinger, Emanuel / Spencer, Ben / Furche, Tim / Gottlob, Georg (2018): Browserless Web Data Extraction: Challenges and Opportunities. In: Proceedings of the 2018 World Wide Web Conference, S. 1095-1104. DOI: 10.1145/3178876.3186008.

FERRARA, Emilio / De Meo, Pasquale / Fiumara, Giacomo / Baumgartner, Robert (2014): Web data extraction, applications and techniques: A survey. In: Knowledge-Based Systems (2014) Nr. 70, S. 301–323.

FIUMARA, Giacomo (2007): Automated Information Extraction from Web Sources: a Survey. In: Michigan State University (Hg.): Between Ontologies and Folksonomies Workshop. S. 1–9.

GALPERIN, Michael Y. / Fernández-Suárez, Xosé M. (2011): The 2012 Nucleic Acids Research Database Issue and the online Molecular Biology Database Collection. In: Nucleic Acids Research 40 (D1), S. D1–D8. DOI: https://doi.org/10.1093/nar/gkr1196.

GLEZ-PEÑA, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M., & Fdez-Riverola, F. (2014): Web scraping technologies in an API world. In: Briefings in Bioinformatics 15(5), S. 788–797. DOI: https://doi.org/10.1093/bib/bbt026.

GOEBEL, Max / Ceresna, M. (2009): Wrapper Induction. In: Encyclopedia of Database Systems, (5), S. 3629–3634.

GOLLA, Dr. Sebastian J. / v. Schönfeld, Dr. Max (2019): Kratzen und Schürfen im Datenmilieu – Web Scraping in sozialen Netzwerken zu wissenschaftlichen Forschungszwecken. In: Kommunikation & Recht (2019), Nr. 1/2019, S. 15–21.

KAISER, Katharina / Miksch, Silvia (2005): Information Extraction. A survey. Technische Universität Wien, Wien. Institut für Softwaretechnik und Interaktive Systeme.

LAENDER, Alberto H. F. / Ribeiro-Neto, Berthier A. / da Silva, Altigran S. / Teixeira, Juliana S. (2002): A Brief Survey of Web Data Extraction Tools. In: ACM SIGMOD Record 31 (2), S. 84–93.

SPICHALE, Kai (2016): API-Design. Praxishandbuch für Java- und Webservice-Entwickler. dpunkt.verlag, Heidelberg.

WORDWIDEWEBSIZE.COM (De Kunder, Maurice): The size of the World Wide Web (The Internet). URL: https://www.worldwidewebsize.com, letzter Aufruf: 27.08.2019.