Hochschule
Ravensburg-Weingarten

Hochschule
Ravensburg-Weingarten
Technik | Wirtschaft | Sozialwesen

Entwicklung einer Anwendung zur automatisierten Beschaffung von personenbezogenen Daten im Internet und deren Integration in Phishing-Mails

Bachelorarbeit

Wintersemester 2018/2019

im Studiengang Angewandte Informatik

an der Hochschule Ravensburg - Weingarten

von

Marco Lang Matr.-Nr.: 27416

Abgabedatum : 12. April 2019

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit mit dem Titel

Entwicklung einer Anwendung zur automatisierten Beschaffung von personenbezogenen Daten im Internet und deren Integration in Phishing-Mails

selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt, keine anderen als die angegebenen Hilfsmittel benutzt und wörtliche sowie sinngemäße Zitate als solche gekennzeichnet habe.

Weingarten, 12. April 2019

Autor Name

_____

Inhaltsverzeichnis

Kurzfassung

Es wird gezeigt, wie eine automatisierte Suche nach personenbezogenen Daten im Internet aussehen kann und wie diese Daten für einen Phishing-Mail-Angriff verwendet werden können.

Danksagung

1 Einleitung

1.1 Motivation

70% der Internetnutzer sehen sich laut einer Umfrage durch das Risiko einer missbräuchlichen Verwendung ihrer Daten nach einem Hack nicht gefährdet [Ang18]
Das Ergebnis dieser Umfrage spricht für die Behauptung, dass viele Personen Informationen über die eigenen Person im Internet preis geben, da keine Ängste vorhanden sind. Doch diese Informationspreisgabe kann in den falschen Händen schwerwiegende Folgen haben. So kann beispielsweise bei einem Phishing-Mail-Angriff diese Art von Information genutzt werden, um ein potentielles Opfer zu täuschen oder zu manipulieren. Ein Beispiel dafür, sind die gefälschten DSGVO-E-Mails, bei denen der Angreifer das Opfer durch scheinbar echte Mails der Sparkasse täuscht. Dabei wird die Zielpersonen persönlich mit ihrem Namen angesprochen, wodurch die Mail an Glaubwürdigkeit gewinnt. [NW18]
Solch ein Angriff benötigt allerdings im Voraus eine ausführliche Recherche über das Opfer. Als Informationsquelle für die Recherche dienen beliebig viele Medien. Doch in der heutigen Zeit ist das Internet die meistgenutzte Informationsquelle für Menschen und birgt dadurch Gefahren für jeden einzelnen Internetnutzer, der personenbezogene Daten im Internet teilt. [All18] Diese Gefahr wird unter anderem durch die Entwicklung von kostenlosen OSINT-Tools erhöht. Diese Tools sammeln Informationen über Opfer von öffentlichen und frei zugänglichen Medien. Dadurch wird die Recherche im Internet nach persönlichen Informationen deutlich einfacher. Das hat zu Folge, dass jeder Internetnutzer ohne großen Aufwand OSINT im Internet betreiben kann.

1.2 Zielsetzung und Forschungsfragen

Ziel ist es eine OSINT-Anwendung zu entwickeln, welche automatisiert nach personenbezogenen Daten im Internet sucht. Die gewonnen Daten werden anschließend in eine Phishing-Mail integriert. Dabei soll der Fokus auf der automatisierten Informationsbeschaffung liegen. Für die Beschaffung der Daten gibt es zwei Arten von Suchfunktionen, siehe Ziel 1 und Ziel 2.
Unter anderem sollen Antworten auf die folgenden Fragen gefunden werden. Mit welchem Aufwand ist ein automatisierte Spear-Phishing-Mail-Angriff verbunden? Ist es möglich ein Personenprofil zu erstellen, bei dem ausschließlich korrekte Informationen vorhanden sind?

Ziel 1 *Informationen zu einer ausgewählten Person im Internet suchen.*

Die erste Suchfunktion beinhaltet die Suche nach Informationen einer bestimmten Person. Dadurch können bereits bekannte Daten über die Person angegeben und somit die Suche verfeinert beziehungsweise verbessert werden. Hierbei ist es wichtig zu erkenne wann es sich um eine Information der gesuchten Person handelt.

Ziel 2 *Nach Informationen einer großen Anzahl von unbekannten Personen suchen, indem eine festgesetzte Webseite vollständig durchsucht wird.*

Bei dieser Suchfunktion soll eine bestimmte Webseiten vorgegeben werden, welche durchsucht, analysiert und ausgelesen wird. Dadurch ist es möglich einen weitläufigen "*real-world*" Phishing-Mail-Angriff zu simulieren.

Ziel 3 *E-Mail-Adressen aus den gewonnen Daten generieren.*

Durch die Zusammensetzung von Vorname, Name und Geburtsjahr werden die E-Mail-Adressen generiert. Außerdem kann der Arbeitgeber, falls er bekannt ist, mit in den Generierungsprozess einfließen.

Ziel 4 *Phishing-Mail-Muster erstellt*

Abhängig von den gefundenen Informationen, soll mit Hilfe dieser Muster, eine Phishing-Mail mit glaubhaftem und sinnvollem Inhalt erstellt werden.

Ziel 5 *Phishing-Mail erzeugen.*

Mit der vorhandenen Information, der E-Mail-Adresse und einem passende Muster, soll eine Phishing-Mail erzeugt und versendet werden können.

1.3 Eigene Leistung

In dieser Arbeit wird ein Programm erstellt, welches personenbezogene Daten automatisiert aus dem Internet heraussucht und diese in potentielle Opferprofile ablegt. Die gewonnenen Informationen werden automatisiert in eine personalisierte Phishing-E-Mail eingebaut. Für einen höheren Erfolg werden E-Mail-Muster konzeptioniert und realisiert.
Damit ein kompletter Ablauf eines Phishing-Mail-Angriffs simuliert werden kann, wird zu jeder Personensuche eine passende E-Mail-Adresse benötigt. Allerdings kann nicht bei jeder Suche eine korrekte E-Mail gefunden werden. Aus diesem Grund wird zusätzlich ein Algorithmus entwickelt, der im Fall, dass keine E-Mail-Adresse zu der Zielperson gefunden wurde, eine Adresse aus den gefundenen Informationen generiert.

1.4 Aufbau der Arbeit

Die Arbeit gliedert sich in einen theoretischen und praktischen Teil auf. Die Theorie beginnt im zweiten Kapitel und beschreibt die Grundlagen 2 im Bereich von personenbezogenen Daten, Social Engineering und der Informationsbeschaffung im Internet. In Kapitel 3 wird das Problem aufgezeigt, auf welches in dieser Arbeit eingegangen wird. Darauf folgt die ethische und rechtliche Betrachtung in Kapitel 4. Die Anforderungsanalyse 5 beschreibt das nächste Kapitel, in welchem die Anforderungen und Prioritäten der Arbeit festgelegt werden. Darauf folgen die Lösungsvorschläge im Kapitel 6 und die Auswahl der Lösung anhand den Anforderungen im Kapitel **??**. Anschließend wird bei der Umsetzung auf den

Praktischen Teil eingegangen. Dieser unterteilt sich in die Themen Informationsbeschaffung einer ausgewählten Person 8, Informationsbeschaffung einer großen Menge an unbekannten Personen 9 und die Erstellung einer Phishing-Mail 10. Am Ende dieser Arbeit befindet sich die Evaluation der Implementation in Kapitel 11 und die Schlussbemerkung und der Ausblick in Kapitel 12.

2 Grundlagen

2.1 Personenbezogene Daten

Laut der DSGVO sind **personenbezogene Daten**, alle Informationen, die sich auf eine identifizierbare Person beziehen. Als identifizierbar wird eine natürliche Person angesehen, die mittels einem oder mehreren Merkmalen direkt oder indirekt identifiziert werden kann. Mögliche Kennungen für die Unterscheidung der Merkmale sind der Name, eine Kennnummer, Standortdaten, eine Online-Kennung, et cetera von der Person. Dabei dienen diese Kennungen als Ausdruck der physischen, physiologischen, genetischen, psychischen, wirtschaftlichen, kulturellen oder sozialen Identitäten dieser natürlichen Person. [DSG]

2.2 Social Engineering

Die Definition von Social Engineering ist nicht eindeutig, da es sehr verschiedene Ansichten davon gibt. Jedoch ist der Grundgedanke von Social Engineering, eine Zielperson so zu manipulieren, damit sie für den Angreifer bessere Entscheidung trifft. [Had11]
Kevin D. Mitnick definiert Social Engineering wie folgt:
"Social Engineering uses influence and persuasion to deceive people by convincing them that the social engineer is someone he is not, or by manipulation. As a result, the social engineer is able to take advantage of people to obtain information with or without the use of technology" [Mit01]

Social Engineering wird Menschen von Geburt an beigebracht und begegnet einem beinahe jeden Tag. Schon ein Baby muss wissen wie es die Eltern manipulieren kann, damit es Dinge

wie Essen, Zuneigung, oder ähnliches bekommt. Darüber hinaus ist Social Engineering in vielen Berufen ein täglicher Bestandteil.

Im Bereich der Informationssicherheit, wird von Social Engineering gesprochen, wenn Angreifer durch die Manipulierung und Täuschung von Menschen vertrauliche Informationen oder Zugänge zu Systemen bekommen. Die bekanntesten Angriffsmethoden sind Phishing, Pretexting, Baiting und Quid Pro Quo. Bei dieser Arbeit wird hauptsächlich auf das Thema E-Mail-Phishing eingegangen.

Der Aufbau eines Social Engineering-Angriffes ist definiert in mehrere Phasen. Das wohl bekannteste Modell für einen Social Engineering-Angriffszyklus ist in dem Buch von Kevin D. Mitnicks [Mit01] definiert. Dieser Zyklus besteht aus den 4 Phasen **Research, Developing rapport and trust, Exploiting trust** und **Utilize information**.
In der **Research-Phase** geht es um die Informationsbeschaffung. Bei dieser Phase will der Angreifer möglichst viele Informationen über das Ziel herausfinden. Die **Developing Rapport and Trust-Phase** beschreibt den Kontaktaufbau zum Ziel, da wenn das Opfer dem Angreifer vertraut, hat dieser ein leichteres Spiel in den kommenden Phasen. Das nun erzeugte Vertrauen wird in der **Exploitung Trust-Phase** ausgenutzt. Hier will der Angreifer die eigentlich Information vom Opfer herausfinden. Dies geschieht einerseits durch bestimmtes Nachfragen oder durch Manipulation. **Utilize Information** ist die letzte Phase. Dort wird die gewonnene Information genutzt um das eigentliche Ziel des Angreifers zu erreichen.
Grundsätzlich werden bei einem Social Engineering Angriff menschliche Wünsche, Ängste und verbreitete Verhaltensmuster verwendet um ein Opfer zu manipulieren. [uDsiNe15]

2.2.1 Phishing

Das Wort Phishing wird von dem Wort “fishing“ abgeleitet, da die Angreifer nach Informationen fischen. Das “Ph“ kommt von “sophisticated“ und meint damit, dass die Angreifer ausgeklügelte Techniken verwenden um an Informationen heranzukommen. [Jam05]
Die wohl bekannteste Angriffsmethode von Phishing ist das E-Mail-Phishing. Bei diesem Verfahren, versendet ein Angreifer meist eine gefälschte E-Mail, um ein Opfer zu täuschen

und dadurch sein Ziel zu erreichen. Die sogenannten Phishing-Mails enthalten meist eine Aufforderung einen Link zu öffnen und sehen täuschend echt aus.
Ein reales Beispiel könnte sein, dass der Angreifer eine gefälschte E-Mail von Amazon an das Opfer versendet und es dabei auffordert, einen Link in der Mail zu öffnen. Nachdem die Zielperson auf den Link geklickt hat, muss Sie sich anmelden. Hier könnte der Angreifer ein täuschend echtes Anmeldeformular erstellt haben, um die Anmeldedaten der Zielperson zu bekommen. Sobald die Anmeldedaten eingegeben wurden, könnte eine Fehlermeldung erscheinen, die einen Authentifizierungsfehler beinhaltet und das Opfer auffordert sich erneut anzumelden. Jedoch wird während diesem Prozess das originale Anmeldeformular geladen und das Opfer kann sich korrekt bei der entsprechenden Webseite anmelden.
Dieser Verfahren ermöglicht Angreifern die Anmeldedaten von einer Zielperson ohne großen Aufwand zu beschaffen. Allerdings benötigt der Angreifer für diese Methode nicht nur Social Engineering sondern auch technische Fähigkeiten. [CH15]

2.2.2 Spear-Phishing

Das Spear-Phishing ist eine erweiterte Methode des herkömmlichen E-Mail-Phishings. Hierbei wird anstatt das Versenden etlicher Phishing-Mails an unbekannte Opfer, eine gezielte Mail an eine ausgewählte Person versendet. [Fir]
Bei dieser Form von E-Mail-Phishing spielt die Opferauswahl und die Informationsbeschaffung eine sehr große Rolle, da diese Information später für personalisierte E-Mails oder vorgetäuschte Identitäten verwendet werden können. Durch diese Art von Täuschung kann ein Opfer dazu bewegt werden auf einen Link zu klicken und dadurch eine Schadsoftware herunterzuladen. [Fir]
Der Aufwand für die Informationsbeschaffung wird oft in Kauf genommen, da der Erfolg bei dieser Methode vielversprechender ist als beim herkömmlichen E-Mail-Phishing.
91% der Advanced Persistent Threat (APT) Angriffe auf Firmen beginnen mit einer Spear-Phishing-E-Mail. Die Schadsoftware wir meisten als Remote Access Trojans (RATs) in einem Zip-Datei überliefert. [Cal13]

2.3 Open Source Intelligence

2.3.1 Definition OSINT

Open Source Intelligence (OSINT) ist definiert in eine Intelligenz, welche aus öffentlich zugänglichen Informationen gewonnen wird. Allerdings kann sich die Bedeutung fallspezifisch ändern. So bedeutet OSINT für die CIA die Informationsgewinnung aus ausländischen Nachrichtensendungen. Doch für die meisten Menschen bedeutet OSINT die Gewinnung eines öffentlichen Inhalts aus dem Internet. [Baz18]
Unter Open Source wird die öffentlich zugängliche Information, die in gedruckter oder elektronischer Form vorliegt, bezeichnet. [Ste96] Eine Verbindung mit dem Begriff Open-Source-Software besteht nicht.

2.3.2 Web Crawler

Web Crawler, auch Robot oder Spider genannt, sind Computerprogramme, die mit Hilfe der Hypertextstruktur das Internet durchlaufen. [The01] Dabei können sie in einen **internen** und **externen Web Crawler** unterschieden werden. Der interne Web Crawler durchsucht ausschließliche interne Seiten einer Webseite und der externe Web Crawler durchsucht unbekannte Webseiten im ganzen Netz. [SG12]

In anderen Worten besteht die Funktionsweise darin, dass in den meisten Fällen ein automatisiertes Programm, Web Crawler, erstellt wird. Dieser lädt Webinhalte herunter und durchsucht den Inhalt nach Hyperlinks. Den gefundenen Links wird gefolgt, um neue Webseiten mit weiteren Links zu laden. So hangelt sich ein Web Crawler von Link zu Link durch das Internet. [Mit15] Dieser Ablauf ist in dem Bild 2.1 noch einmal verdeutlicht.

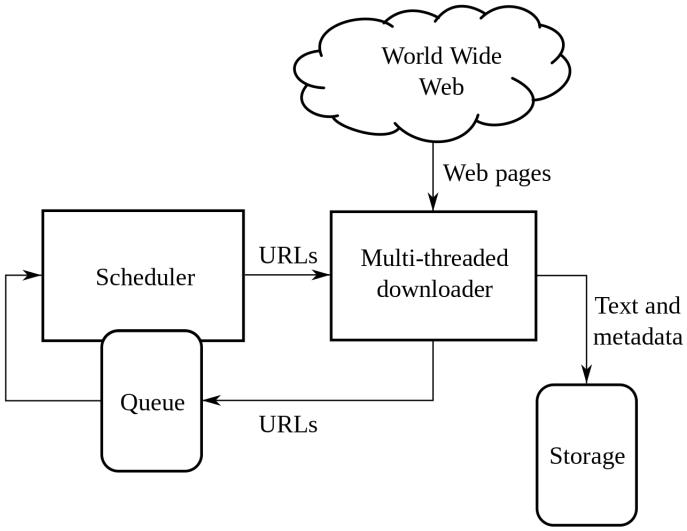

Bild 2.1: Architektur eines Web Crawlers

2.3.3 Web Scraper

In der Theorie bedeutet *web scraping* die Informationsbeschaffung im Internet mit unterschiedlichsten Mitteln. [Mit15]
Meist wird dies mit einem automatisierten Programm realisiert, welches Daten von einem Webserver anfragt, entgegen nimmt, analysiert und auswertet. In der Praxis gibt es ein großes Feld von Programmiertechniken und Einsatzmöglichkeiten. Mit Hilfe eines Web Scrapers ist es möglich, große Datenmengen zu erfassen und zu verarbeiten. [Mit15]

Natural Language Processing

Natural Language Processing (NLP) beschreibt eine Technologie, für die Kommunikation zwischen Mensch und Computer. Mit dem Ziel, dass ein Computer die natürliche Sprache verstehen und verarbeiten kann. Dafür werden verschiedenste Methoden aus der Sprach- und Computerwissenschaft sowie aus der künstliche Intelligenz verwendet. Unter anderem hat eine NLP-Anwendung die Aufgabe von **Stemming**. [Lit16]

Stemming ist eine Methode der Wortstandardisierung, bei der verwandte Wörter auf ihrer Stammform reduziert werden. Dabei wird bei dem Rechenvorgang auf den Stamm und die

Semantik eines Wortes geachtet. Aus diesem Grund fällt der Name Stammformreduktion öfters in Verbindung mit Stemming. [EAD09] Ein Beispiel hierfür wären die Worte “Wetter“ und “Wetten“, welche auf den Stamm “Wett“ reduziert werden könnten. [PH]
Die Verwendung von Stemming, kann bei der Schlüsselwortgenerierung von Texten sehr hilfreich sein, da die Anzahl der möglichen Schlüsselwörter reduziert werden können.

3 Problembeschreibung

Persönliche Daten sind im Internet oft frei zugänglich. Das heißt, dass unterschiedlichste Webseiten persönliche Information von Menschen öffentlich bereitstellen. Die bekanntesten Webseiten sind die Social Media Seiten wie Twitter, Facebook und Instagram. Allerdings wird auch auf anderen Webseiten personenbezogene Daten in großen Mengen bereitgestellt. Ein Beispiel dafür ist das Fußballportal *"www.fupa.net"*. Diese Art von Webseiten sind perfekte Informationsquellen für Phisher, da im Bereich von Social Engineering, diese Informationen oft genutzt werden um ein Opfer zu täuschen oder zu manipulieren.
Dass hier beschriebene Problem zeigt, dass der Zugang für persönliche Information durch das Internet für die Öffentlichkeit einfacher gemacht wird. Es soll mit einem kritisch Blick darauf gezeigt werden, mit welchem Aufwand, personenbezogene Daten aus dem Internet herausgelesen, analysiert und für einen Phishing-Mail-Angriff verwendet werden können.

4 Ethische und rechtliche Betrachtung

Das Sammeln von personenbezogenen Daten auf sozialen Netzwerken ist ethisch und rechtlich gesehen ein sehr sensibles Thema. Jedoch werden in dieser Arbeit ausschließlich die Daten verwendet, die öffentlich frei zugänglich sind. Das heißt, unter den Informationen befinden sich keine Passwörter oder Informationen die nicht an die Öffentlichkeit gehören. Des Weiteren ist der hier verwendete Crawler nicht stark genug, um die Leistung eines Servers von einem sozialen Netzwerk zu beeinflussen.
Mit diesem realen Experiment, soll die Privatsphäre der Benutzer geschützt werden, indem aufgezeigt wird, wozu veröffentlichte Daten über eine Person im negativen Sinn verwendet werden können. Genau aus diesem Grund ist es wichtig, dass das Experiment in der realen Welt durchgeführt wird.
Die gefundenen Daten werden in einer verschlüsselten Datei gespeichert, um die Privatsphäre der Internetnutzer zu schützen.

5 Anforderungsanalyse

Die im Kapitel 1.2 definierten Ziele sollen mit den folgenden Anforderungen gewährleistet werden.

5.1 Anforderung an OSINT

Die Anforderung an OSINT lässt sich in zwei Teile gliedern. Der erste Teil beinhaltet das OSINT einer ausgewählten Personen und der zweite Teil OSINT einer großen Menge unbekannter Personen.

5.1.1 OSINT einer ausgewählten Person

Bei dieser Informationsbeschaffung soll eine Suchfunktion entwickelt werden, welche Daten zu einer angegeben Person im Internet sucht. Hierbei sollen so viele Daten wie möglich gefunden und gespeichert werden.
Das zu entwickelnde Programm soll für die Suche bekannte Daten wie Vorname, Nachname, Geburtsjahr, Ort und Benutzernamen von Social Media Plattformen einlesen können. Die Eingabe kann mit Hilfe einer Konsole oder einer grafische Oberfläche realisiert werden.
Die Herausforderung besteht darin, zu erkennen, wann und ob es sich um die Information der gesuchten Person handelt. Sowie die Analyse und das Herauslesen dieser Daten.

5.1.2 OSINT einer großen Menge von unbekannten Personen

Für die *real-world* Simulation eines Phishing-Mail-Angriffs, soll eine Suchfunktion entwickelt werden, die OSINT einer kompletten Webseite betreiben kann. Dabei sollen möglichst viele Informationen von möglichst vielen Personen herausgefunden werden. Jedoch sind diese Personen dem Programm-Anwender unbekannt. Die Informationen sollen von einer festgesetzten Webseite herausgelesen werden. Hierfür wird manuell nach einer Webseite gesucht, die eine große Menge an personenbezogenen Daten enthält und sich dadurch gut für OSINT eignet.
Zusätzlich soll der zu entwickelnde Web Scraper möglichst performant arbeiten.

5.2 Anforderung an die Datenverwaltung/-speicherung

Ausgelesene Daten sollen vor dem speichern formatiert und klassifiziert werden, damit die Daten später korrekt in die Phishing-Mails eingesetzt werden können. Die Schwierigkeit besteht darin, zu erkennen, um welche Art von Information es sich handelt. Zusätzlich sollen die Daten in einer gut übersichtlichen Struktur gespeichert werden und müssen beliebig erweiterbar sein.

5.3 Anforderung an die Generierung der E-Mail-Adressen

Da nicht zu jeder Suche eine E-Mail-Adresse im Internet gefunden werden kann, muss die E-Mail-Adresse aus den vorhandenen Informationen generiert werden. Es soll eine größere Anzahl von möglichen E-Mail-Adressen erzeugt werden. Durch den Pool an erzeugten E-Mail-Adressen soll die Wahrscheinlichkeit erhöht werden, dass die richtige E-Mail-Adresse dabei ist. Des Weiteren sollen die Adresse auf Verfügbarkeit und Gültigkeit geprüft werden.

5.4 Anforderung an die E-Mail-Muster

Bei der Erstellung der E-Mail-Muster handelt es sich ausschließlich um das Erstellen potentieller Inhalte einer E-Mail, welche mit den gewonnenen Informationen über eine Person erweitert werden kann. Die Muster sollen erstellt werden und so klassifiziert sein, dass für jedes gefundene Opferprofil ein passendes Muster vorhanden ist. Des Weiteren soll der E-Mail-Text mit den eingesetzten Informationen Sinn ergeben und eine korrekte Grammatik beinhalten. Weiterführend können SE-Fähigkeiten genutzt werden um die Zielperson tatsächlich zu manipulieren und zu täuschen. Hierfür können beispielsweise Gefühle wie Freude und Angst ausgenützt oder gefälschte E-Mails von bekannten Firmen in Betracht gezogen werden.

5.5 Anforderung an die Erstellung der Phishing-Mail

Die Phishing-Mails sollen automatisiert erstellt werden. Die Auswahl des richtigen E-Mail-Musters zu der gewonnenen Opferinformation soll ebenfalls automatisiert ablaufen.

6 Lösungsideen

In diesem Kapitel werden die Lösungsideen für die Umsetzung der im Kapitel 1.2 definierten Ziele beschrieben.

6.1 Methoden für das OSINT einer ausgewählten Person

6.1.1 Verwendung von OSINT-Tools

Die Personensuche wird durch die Verwendung kostenloser OSINT-Tools durchgeführt. Eine entsprechende Webseite die mehrere OSINT-Methoden bereit stellt, ist unter dem URL *"https://inteltechniques.com/index.html"* erreichbar. Sie stellt Methoden zur Suche nach E-Mail-Adressen, Benutzernamen, Social-Media-Profilen, und noch viele mehr zu Verfügung. Allerdings werden nicht nur selbstentwickelt OSINT-Methoden von Michael Bazzell bereitgestellt, sondern auch andere Webseiten mit weiteren OSINT-Tools vorgeschlagen.

6.1.2 Algorithmus für OSINT entwickeln

Es wird ein Algorithmus für OSINT entwickelt, der aus einem Web Crawler und Web Scraper besteht. Mit diesem ist es möglich eigenständig nach Information zu suchen. Hierfür wird eine Suchmaschine, wie die von Google, verwendet.
Die Suchergebnisse können mit Hilfe des Web Crawlers verfolgt werden. Anschließen wird

der Webseitentext, durch den Web Scraper, ausgelesen. Im letzten Schritt, wird der Text analysiert und interpretiert.
All diese Prozesse laufen unabhängig von den vorgeschlagenen Webseiten voll automatisiert ab.

6.2 Mögliche Webseiten für OSINT mehrerer unbekannter Personen

Für das OSINT mehrere unbekannter Personen stehen die Webseiten von FuPa, Xing und LinkedIn zu Auswahl.

6.2.1 XING

XING ist ein soziales Netzwerk für Berufstätige mit über 15 Millionen Mitgliedern. Hier vernetzen sich Kontakte aus allen Branchen um Jobs, Mitarbeiter, Aufträge oder ähnliches zu suchen und zu finden. [SE]
XING bietet allerdings viele Möglichkeiten zum Schutz der Privatsphäre. So kann ein Nutzer einstellen, ob er von einer Suchmaschine gefunden werden oder nur für Xing-Mitglieder sichtbar sein will.

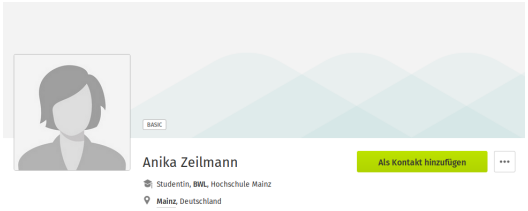

Bild 6.1: Profil von der Webseite XING

6.2.2 LinkedIn

LinkedIn ist das weltweit größte soziale Netzwerk für Berufstätige mit mehreren Millionen Mitgliedern. Es vernetzt berufliche Kontakte der ganzen Welt und stellt ebenfalls Möglichkeiten zum Schutz der Privatsphäre zu Verfügung. [Cor17]

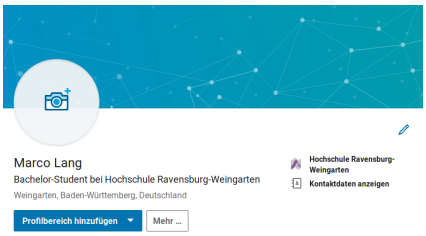

Bild 6.2: Profilkarte von der Webseite LinkedIn

6.2.3 Fupa

Die Webseite Fupa stellt ein regionales Fußballportal dar, welches zur Berichterstattung des Amateurfußballs vorhanden ist. Allerdings werden nicht nur Berichte sondern auch aussagekräftige Spielerprofile zur Verfügung gestellt. [Gmb16] Außer dem verzeichnet FuPa eine Mitgliederzahl von über 200.000. [Wik19]
Das Bild 6.3 zeigt ein Spielerprofil, wie es auf dieser Webseite angezeigt wird. Allerdings kann sich die Vollständigkeit eines Profils variieren.

Bild 6.3: Spielerprofil von der Webseite FuPa

6.3 Konzept für die Erstellung einer Phishing-Mail

Die Generierung einer realen Phishing-Mail benötigt eine korrekte E-Mail-Adresse der Zielperson. Darüber hinaus sollten die gewonnen Informationen in einem sinnvollen E-Mail-Text eingebunden werden. Die Generierung einer Phishing-Mail läuft voll automatisch ab. Das bedeutet, dass das Programm eigenständig die E-Mail-Adressen generiert und passende E-Mail-Muster auswählt.

6.3.1 Methoden zur Generierung der E-Mail-Adresse

Beim OSINT einer ausgewählten Person wird bereits nach E-Mail-Adressen der Zielperson gesucht. Dadurch kann eine bis jetzt unbekannte Anzahl von Adressen gefunden werden. Die Methoden zur Generierung einer E-Mail-Adresse muss dadurch nicht für jede Zielperson durchgeführt werden. In dem Fall, dass keine E-Mail-Adresse gefunden wurde, wird diese Methode verwendet.

Algorithmus entwickeln zum generieren

Es kann ein Algorithmus entwickelt werden, der mögliche E-Mail-Adressen aus den gewonnen Daten generiert. Dies ist durch die Kombination aus Vorname, Nachname, Geburtsjahr und den bekanntesten E-Mail-Providern realisierbar. Für den Fall, dass der Arbeitgeber der Zielperson bekannt ist, kann auf der Firmenwebseite nach E-Mail-Adressen gesucht werden. Dadurch ist es möglich die Domain einer Firmen-Mailadresse zu bestimmen und eine Anzahl möglicher Firmenadressen für die Zielperson zu generieren. Durch diese Methode wird eine Pool mit möglichen Mailadressen erstellt. Dabei muss jede einzelne E-Mail-Adresse auf Validität geprüft werden.

Automatisierbare OSINT-Tools verwenden

Für die Generierung der E-Mail-Adressen kann ein kostenloses OSINT-Tools von Michael Bazzel verwendet werden. Diese Tool ermöglicht es, die gewonnenen Informationen über

eine Formular einzugeben. Anschließend werden draus mögliche E-Mail-Adressen generiert. Auch hier entsteht ein Adresspool, bei dem die E-Mail-Adressen auf Validität geprüft werden. Zu dem bringt das Tool eine weitere Funktion mit sich. Es wird automatisch nach Einträgen, der generierten E-Mail-Adressen, im Internet gesucht und angezeigt. [Baz]

6.3.2 Methoden zur Generierung des E-Mail-Textes

Muster für den E-Mail-Text erstellen

Die zu erstellenden E-Mail-Muster entsprechen hier kategorisierten Lückentexten. Abhängig von den gefundenen Daten, wird ein Lückentext ausgewählt und anschließend mit den Daten an den passenden Stellen ergänzt.
Die Lückentexte werden so kategorisiert, dass für jede gefundene Information ein passender Lückentext vorhanden ist. Eine denkbare Unterteilung wären die Kategorien Privat und Geschäftlich.

E-Mail-Text aus Fragmenten erzeugen

Bei dieser Methode besteht der E-Mail-Text aus zusammengesetzten Fragmenten. Dafür wird zu jeder gefundenen Information ein Fragment erstellt. Anschließend werden alle Fragmente zu einem Text zusammengefügt. Der Unterschied zur Methode 6.3.2 besteht darin, dass der E-Mail-Text dynamisch erzeugt wird. Das bedeutet, der endgültige Text ist nicht vorgeben. Er kann aus einer variierenden Anzahl von Fragmenten besteht. Diese Anzahl kann variieren, da sie abhängig von der gefundenen Information über die Zielperson ist.

7 Bewertung der Lösungsideen anhand der Anforderung

7.1 Bewertung der OSINT-Methoden für eine ausgewählte Person

Hierfür gibt es zwei verschiedene Methoden um OSINT zu betreiben. Die erste Lösungsidee beschreibt die Verwendung von einem öffentlich frei zugänglichen OSINT-Tool. Diese Tool bietet sehr viele Möglichkeiten um eine Person beziehungsweise Daten über eine Person zu finden. Allerdings ist es auf dieser Webseite nicht möglich ein zu suchendes Profil anzugeben, um eine Person zu finden. Die Suchen sind aufgeteilt in verschiedenste Daten wie Name, E-Mail, et cetera. Aus diesem Grund wird bei einer Suche ausschließlich nach dem Namen oder einer E-Mail gesucht. Dadurch ist das Suchergebnis am Ende kein vollständiges Personen-Profil, sondern lediglich Verweise auf weiterer Webseiten mit möglichen Einträgen. Dazu ist die Eingabemöglichkeiten der im Voraus bekannten, Daten begrenzt, da die Formulare nicht individuell erweiterbar sind.
Im Gegensatz zu diesem Tool, nutzt der eigenen Algorithmus alle im Vorfeld bekannten Daten für eine Suche. Des Weiteren kann die Laufzeit verbessert werden und bekannten Suchtechniken dieses Tools, mit Hilfe des Buches [Baz18] verwendet werden. Durch die eigene Anwendung wird die Suche beliebig erweiterbar programmiert. Dadurch kann jede Information zur Personensuche verwendet werden.

7.2 Bewertung der OSINT-Methoden für eine großen Anzahl unbekannter Personen

Bei den sozialen Netzwerken XING und LinkedIn lässt sich die Privatsphäre eines Benutzers in den Kontoeinstellungen konfigurieren. Dies hat zu folge, dass viele Profile nicht von Suchmaschinen sondern ausschließlich von angemeldeten Mitgliedern gefunden werden kann. Somit muss für das vollständige Auslesen dieser Webseite, ein Benutzerkonto angelegt werden. Das anlegen eines Kontos ist ein negativer Aspekt, da zusätzliche Arbeit für ein anonymes Konto entsteht.
Beide Netzwerke stellen eine Mitgliedersuche zur Verfügung. Des Weiteren lassen sich von bekannten Firmen Mitgliederliste anzeigen, wodurch Personenprofile mit zugehörigem Arbeitgeber angezeigt werden. Dies kann später für die Generierung einer E-Mail-Adresse sehr hilfreich sein.
Auf der Webseite Xing ist es möglich, ein Mitgliederverzeichnis anzuzeigen, ohne ein Benutzerkonto zu erstellen. Jedoch sind dort nur die Mitglieder aufgelistet, welche keinen besonderen Privatsphären-Schutz eingestellt haben. Trotzdem ist dieses Verzeichnis eine gute Informationsquelle, da es eine große Liste mit vielen Verweisen zu persönlichen Profilen ist. Dennoch werden die Profile nicht vollständig angezeigt, da eine Anmeldung von Nöten ist. Dies könnte allerdings im folgenden Schritt getan werden.
Die Webseite Fupa, bietet kein Mitgliederverzeichnis, welches jedes Spielerprofil auf einer Seite auflistet. Jedoch benötigt FuPa keine Anmeldung und es besteht keine Möglichkeit, den Schutz der Privatsphäre von Spielerprofilen zu verstärken. Die einfach Struktur der Webseite ist ebenfalls ein Vorteil. Des Weiteren spricht die Gewinnung des Geburtsjahres für beinahe jeden Spieler ebenso für FuPa, da das Geburtsjahr für die E-Mail-Generierung sehr wichtig ist. Darüber hinaus benötigt FuPa kein JavaScript um angezeigt zu werden. Dies kann beim OSINT zu einer Laufzeitverbesserung gegenüber den Konkurrenten führen, da kein automatisierter Browser benötigt wird.

7.3 Bewertung der Methoden zur Erstellung einer Phishing-Mail

Generierung der E-Mail-Adressen

Bei der Verwendung eines bereits fertigen OSINT-Tools wird keine große Arbeit mehr benötigt, es ist ein komplettes System was funktioniert. Lediglich die Automatisierung muss entwickelt werden. Allerdings kann nicht jede individuelle Information für die Generierung genutzt werden. Dies ist für die zu entwickelnde Anwendung ein großer Nachteil. Die Wahrscheinlichkeit, dass sich die richtige E-Mail-Adresse darunter befindet, wird dadurch kleiner.
Im Gegensatz dazu, kann ein eigener Algorithmus jegliche Information mit in die Generierung einer E-Mail-Adresse einfließen lassen. Ein Beispiel hierfür wäre das Geburtsjahr einer Zielperson. Das OSINT-Tool [Baz] verwendet das nicht. Allerdings können die möglichen Adressen, welche von dem OSINT-Tool generiert wurden, als Anregung und Ideengeber für den eigenen Algorithmus dienen.
Für die erfolgreiche Simulation eines Phishing-Mail-Angriffes, wird die richtige E-Mail-Adresse benötigt. Aus diesem Grund wird der eigene Algorithmus verwendet, damit die Wahrscheinlichkeit erhöht wird, dass sich die korrekte Maildresse in dem Pool befindet.

E-Mail-Text

Der Inhalt einer E-Mail ist sehr wichtig für die Glaubwürdigkeit einer Phishing-Mail. Aus diesem Grund ist es von Bedeutung, dass der E-Mail-Text Sinn ergibt und mit einer guten Grammatik geschrieben wurde. Bei der dynamischen Texterzeugung mit der Fragment-basierten Methode 6.3.2, kann die Grammatik und der Zusammenhang des Textes zu einer Problematik führen. Durch die Verkettung von verschiedensten Fragmenten kann ein Text erzeugt werden, welcher kein sinnvoller Zusammenhang hat. Allerdings muss hierbei nicht für jede Kombination aus gewonnen Daten ein vollständiges Fragment-Muster erstellt werden. Wogegen bei der Verwendung von fertigen Lückentexten, ein Muster für jede Kombination aus gewonnen Daten vorhanden sein muss. Dennoch ist die Glaubwürdigkeit

durch einen sinnvollen E-Mail-Text höher. Aus diesem Grund werden die vollständigen E-Mail-Muster umgesetzt.

8 OSINT einer ausgewählten Person

8.1 Auswahl der Programmiersprache

Damit das Programm anhand den Lösungsideen umgesetzt werden kann, ist der erste Schritt die Auswahl der Programmiersprache.
Hierbei wird keine Anforderung an die Geschwindigkeit der Sprache gestellt, da beim web scraping das Internet den zeitlichen Engpass darstellt. Allerdings wäre es von Vorteil wenn bereits entwickelte Bibliotheken für das OSINT vorhanden sind. Die Eingabe der Information für die Suche kann über eine Konsole oder über eine graphische Benutzeroberfläche möglich sein.
Als mögliche Programmiersprachen zählen Python, Ruby, C++.
Für web-basierende Anwendung eignet sich eine dynamische Programmsprache. Im Gegensatz zu Python und Ruby zählt C++ nicht zur Familie der dynamischen Programmiersprachen und fällt aus diesem Grund als mögliche Lösung heraus.
Python und Ruby können beide Webseiten, die JavaScript zum rendern benötigen, laden. Dies ist mit Hilfe eines automatisierten Webbrowsers möglich. Des Weiteren lässt sich die Anwendung durch beide Sprachen, entsprechend den Anforderungen entwickeln. Es kann sowohl eine Oberflächenanwendung als auch eine Konsolenanwendung programmiert werden. Zusätzlich bringen beide Sprachen Module mit sich, um das Pojekt mit den vorgegebenen Zielen umzusetzen. Somit haben beide Programmiersprachen die Voraussetzungen für die Entwicklung der Anwendung. Allerdings bietet Python in diesem Bereich eine große Community und eignet sich sehr gut für die Bearbeitung von linguistischen Daten. [BKL09]
Aus diesen Gründen wird die zu erstellende Anwendung mit der Programmiersprache Python entwickelt.

8.2 Methoden zur Suche nach einer Person im Internet

Die Art der Personensuche wird abhängig von den eingegeben Daten variiert. Das heißt, dass die eingegebenen Daten über die Zielperson vor der Suche analysiert werden und dementsprechend angepasst wird. Nachstehen werden zwei grundsätzliche Methoden für die Art der Personensuche beschrieben.

8.2.1 Personensuche mit Hilfe einer Suchmaschine

Hier wird mit Hilfe einer Suchmaschine nach Informationen gesucht. Mögliche Suchmaschinen sind die von Google und Bing. Allerdings muss nicht für jede Suche eine Suchmaschine verwendet werden. Die nachfolgenden Fälle sollen diesen Ansatz verdeutlichen.

Im Fall, dass der Vorname, Nachname und Wohnort der gesuchten Person eingegeben wird, kann mit Hilfe der festgelegten Suchmaschine nach Information gesucht werden. Die von der Suchmaschine vorgeschlagenen Seiten werden anschließend analysiert, ausgelesen und gespeichert. Dadurch können weitere Informationen gewonnen werden. Falls Benutzernamen von anderen Webseiten wie Instagram, Facebook oder ähnliches vorgeschlagen werden, kann somit die Suche mit diesen Daten speziell auf den entsprechenden Seiten erweitert werden.

Ein weiterer Fall beschreibt das Szenario, wenn ein Benutzername der gesuchten Person in das Programm eingegeben wird. Hierbei handelt es sich um einen Benutzernamen von Social-Media-Webseiten wie Facebook, Instagram, LinkedIn, et cetera.
Zuallererst, wird hier nach Einträgen auf der entsprechende Webseite zu dem angegebenen Benutzername gesucht. Dadurch können zusätzliche Daten herausgefunden werden. Diese sind bei der weiteren Suche von Vorteil.
Sobald die Webseite mit Hilfe des Nutzernamens durchsucht und ausgewertet wurde, kann die Suche mit einer Suchmaschine und den gewonnen Daten erweitert werden.

8.2.2 Personensuche auf festgelegten Webseiten

Unabhängig von den eingegebenen Daten, wird eine festgesetzte Anzahl von Webseiten durchsucht. Als potentielle Kandidaten-Webseiten eigenen sich die Social-Media-Seiten wie Facebook, Instagram, Twitter, LinkedIn, et cetera. Diese Art der Personensuche arbeitet allerdings ohne die Verwendung einer Suchmaschine.

8.3 Bewertung der Methoden zur Personensuche

Um möglichst viele Informationen über eine Person im Internet zu finden, bietet die Personensuche mit der Verwendung einer Suchmaschine die beste Lösung. Es wird anstatt ausschließlich festgelegten Seiten das ganze Internet durchsucht. Dadurch können wesentlich mehr individuelle Einträge gefunden werden. Des Weiteren wird keine Logik zur Suche nach Einträgen im Internet benötigt, da lediglich den vorgeschlagenen Suchergebnissen gefolgt werden kann.
Allerdings muss beachtet werden, dass Benutzer bei verschiedensten Social-Media-Seiten auswählen können, ob das Benutzerprofil von einer Suchmaschine gefunden werden kann oder nicht. Bekannte Webseiten die diese Einstellungsmöglichkeiten unterstützen sind XING und LinkedIn. Aus diesem Grund, werden zu Beginn der Suche die Social-Media-Seiten durchsucht. Dadurch können vor der Google-Suche zusätzliche Informationen herausgefunden werden, die für das später OSINT von Vorteil sind. Falls sich eine Social-Media-Seite unter den Google-Suchergebnissen befindet, kann diese nachträglich ebenfalls durchsucht werden.

8.3.1 Auswahl der Suchmaschine

Laut Expertenaussage sucht Bing tiefgreifender nach Information auf Social Media Seiten wie Facebook, Twitter und LinkedIn. Allerdings finden nur 3,5% aller Suchanfragen in Deutschland über Bing statt. Im Gegensatz dazu hat Google einen Marktanteil von 91,2% in Deutschland. Diese Zahlen sprechen eindeutig für Google. Durch die höhere Anzahl von Suchanfragen, können mehr Daten erfasst und die Ergebnislisten besser gerankt werden.

Dies hat zu Folge, dass Bing bei einer konkreten Suche schlechter abschneidet. [Boh14] Grundsätzlich stellt die Verwendung von zwei Suchmaschinen die beste Lösung dar, da die Wahrscheinlichkeit für einen Suchtreffer erhöht wird. Dennoch wird in dieser Arbeit ausschließlich die Suchmaschine von Google verwendet, da sie gegenüber dem Konkurrenten keine Nachteile hat. Selbst die detailliertere Suche auf Sozialen Netzwerken, bringt bei der hier verwendeten Personensuche keinen großen Vorteil für Bing. Das heißt, durch die Analyse der Suchergebnisse, wird erkannt ob sich die bekannten Social Media Webseiten darunter befinden. Falls diese es nicht tun, wird die Suche auf den entsprechenden Sozialen Netzwerken erweitert.

8.4 Umsetzung: Personensuche mit Hilfe der Google-Suchmaschine im Internet

Die Suchmaschine von Google wird für die Personensuche im Internet verwendet. Gesucht wird nach den eingegebenen Daten, welche über die Konsole eingelesen werden.

8.4.1 Eingabe der bekannten Daten

Es besteht die Möglichkeit den **Vorname, Nachname, Wohnort, Arbeitgeber, Instagram Benutzername, Facebook Benutzername, Twitter Benutzername, E-Mail-Adresse**, und das genaue beziehungsweise geschätzte **Geburtsjahr** der gesuchten Person über eine Konsole einzugeben. Falls der genaue Jahrgang der Zielperson nicht bekannt ist, kann ein geschätztes Geburtsjahr eingetragen werden. Dies kann später bei der Identifizierung der gesuchten Person hilfreich sein.
Zu Beginn werden alle Personen-Variablen mit einem leeren String initialisiert. Das bedeutet, alle Variablen, zu denen keine Information eingegeben wurde, enthalten einen leeren String.

Verarbeitung der Daten

Im ersten Schritt wird kontrolliert, welche Informationen vom Progamm-Nutzer eingegeben wurden. Der Vorname und Nachname sind nicht ausreichend für die Suche. Es wird mindestens ein weiteres Attribute benötigt. Dagegen ist der Benutzernamen von Instagram und Twitter sowie die E-Mail-Adresse einzigartig. Dadurch kann mit einem dieser Attribute gesucht werden.
Bei der Eingabe des Wohnortes, kann dieser vor der Suche mit der entsprechenden Wortsammlung verglichen werden. Falls sich der Wohnort nicht in der Datenbank befindet, wird er nachträglich ergänzt. Für die Personenerkennung ist es wichtig, dass sich der korrekt Wohnort in der Datenbank befindet.
Daraufhin werden mit diesen Eingaben Kombinationen für die Suche und die URL-Generierung erstellt. Mögliche Such-Kombinationen für erfolgreiche Ergebnisse sind:

Vorname, Nachname, Wohnort;
Vorname, Nachname, Geburtsjahr;
Vorname, Nachname, Institution;
Vorname, Nachname, Wohnort, Geburtsjahr;
Vorname, Nachname, Wohnort, Institution;
Benutzername einer Social-Media-Seite;

Die Kombination aus vielen oder allen Daten ist ebenfalls eine mögliche Option. Allerdings wird dadurch oft kein Ergebnis gefunden, da nicht zur jeder Information ein Eintrag im Internet besteht.
Sobald die Kombinationen aus den Daten bekannt sind, werden die Such-URLs für die Google-Suchmaschine generiert.

8.4.2 Generierung der Google-Such-URLs

Aufbau eines URLs

Ein Uniform Resource Locator (URL) lokalisiert eine Ressource, indem eine abstrakte Identifikation der Lokalisierung verwendet wird. Dabei wird ein URL grundsätzlich im

folgenden Format angegeben. [RFC94]

$< scheme >:< scheme - specific - part >$ [RFC94]

Das Schema gleicht hierbei meist dem verwendeten Protokoll wie HTTP oder FTP. Der Doppelpunkt stellt die Trennung zum Schema-spezifischen Teil dar. Ein Beispiel für ein HTTP-URL-Aufbau ist im Folgenden definiert. [RFC94]

$http : // < host >:< port > / < path >? < searchpart >$ [RFC94]

Hier wird das Protokoll HTTP als Schema verwendet, wobei sich der Aufbau bei der Verwendung des HTTPS-Protokolls kaum unterscheidet. Lediglich das Schema und der Port verändert sich.
Für den <host> kann der FQDN oder die IP-Adresse des Hostrechners eingetragen werden. Wenn der Port nicht angegeben wird, ist der Standardport voreingestellt. Bei HTTP wäre dies Port 80 und bei HTTPS Port 443. Der <path> stellt ein HTTP-Selektor dar und ist mit einem Fragezeichen von der Suchzeichenkette getrennt. [RFC94]
Im Bereich des <searchpart> lassen sich URL-Parameter einfügen um Informationen an die entsprechende Webseite mitzugeben. Die Parameter bestehen aus einem Schlüssel und aus einem Wert, welche durch ein Gleichheitszeichen getrennt werden. Um mehrere Parameter hinzuzufügen und zu kombinieren wird das kaufmännische Und-Zeichen verwendet. [AH19]
Ein URL für die Google-Suche von *Max Mustermann* ist in dem folgenden Beispiel gegeben.

$https : //www.google.com/search?q = Max + Mustermann$

Allerdings können URLs nur mit ASCII-Zeichen erzeugt und versendet werden. Aus diesem Grund müssen Zeichen die nicht im ASCII vorkommen, in ein gültiges Format umgewandelt werden. Dies wird realisiert, indem die URL-Kodierung das nicht enthaltende ASCII-Zeichen durch ein “%“, gefolgt von zwei Hexadezimalen Ziffern, ersetzt. Beispielsweise repräsentiert “%20“ ein Leerzeichen und “%22“ ein Anführungszeichen. [W3S]

Erstellen der Such-URLs

Dieser Absatz beschreibt die Erstellung der Such-URLs für Google, mit dem Wissen aus Kapitel 8.4.2.
Für jede genannte Kombination aus den eingegebenen Daten werden Link-Muster erzeugt. Diese entsprechen einem Lückentext. Sobald die entsprechenden Muster ausgewählt wurden, werden die Lücken mit den Daten befüllt. Dadurch wird eine Liste mit einer variierende Menge von Suchlinks erstellt. Diese Liste wird anschließende von dem Web Crawler verwendet um die Suche zu starten. Ein URL für die Suche nach Information auf beliebigen Webseiten wird wie folgt dargestellt.

https://www.google.com/search?q=%22Max+Mustermann%22+%22Weingarten%22

Wenn allerdings der Benutzername einer Social-Media-Seite bekannt ist, werden zwei unterschiedliche URLs verwendet. Mit Hilfe des ersten URLs, wird speziell nach Einträgen auf der entsprechenden Webseite gesucht. Dazu kann der Operator “site“ verwendet werden. Dieser beschränkt die Suchergebnisse soweit, dass die vorgeschlagenen Einträge ausschließlich auf einer festgelegten Webseite vorkommen. Das folgende Beispiel beschreibt die Suche nach dem Benutzer “Mustermann“ auf der Webseite “Instagram.com“. Dabei ersetzt die ASCII-Zeichenkette “%3A“ den Doppelpunkt. [W3S]

https://www.google.com/search?q=site%3Ainstagram.com+%22Mustermann%22

Der zweite URL wird für eine Social-Media-Suche verwendet. Bei dieser Suche werden Social-Media-Seiten nach Einträgen durchsucht. Dafür wird kein zusätzlicher Operator benötigt. Es wird lediglich ein @-Zeichen, welches mit der Zeichenkette “%40“ dargestellt wird, vor dem zu suchenden Wort eingefügt. Die Social-Media-Suche nach dem Benutzernamen “Mustermann“ sieht folgendermaßen aus. [Goo19]

https://www.google.de/search?q=%40Mustermann

Optimierung der Such-URLs

Um die Suchergebnisse von Google zu verbessern, können die Suchbegriffe in Anführungszeichen gesetzt werden. Dadurch wird eine Phrasensuche gestartet, die nach einer

Zeichenfolge sucht. Das bedeutet, es wird ausschließlich nach diesen Zeichenfolgen gesucht und nicht nach einer Abwandlung. Ein Beispiel hierfür ist die Suche nach “Mike Bazzell“. Wenn diese Suche ohne Anführungszeichen durchgeführt wird, werden zusätzlich Webseiten vorgeschlagen die den Namen Mike Bazzell anstatt Micheal Bazzell beinhalten. Diese erweiterte Suche kann dazu führen, dass unzählige Webseiten vorgeschlagen werden, die nicht unbedingt was mit dem Thema der Suchbegriffe zu tun hat. Um dem vorzubeugen können Anführungszeichen verwendet werden, welche die Anzahl der Suchergebnisse um einen sehr großen Teil verringern. [Baz18]
Für die Suche nach **Marco Lang** werden ungefähr **96.400.000** Ergebnisse mit Hilfe der Google-Suchmaschine gefunden. Wird die Suche mit den Anführungszeichen verfeinert indem nach **“Marco“ “Lang“** gesucht wird, werden etwa **55.600.000** Ergebnisse gefunden. Allerdings werden hier Webseiten vorgeschlagen, welche die Wörter “Marco“ und “Lang“ beinhalten, jedoch müssen diese nicht direkt nebeneinander und auch nicht in der Reihenfolge vorkommen. Es wäre Möglich, dass bei dieser Suche, Webseite mit Verweisen auf die Namen “Marco Mustermann“ und “Max Lang“ beinhaltet. Aus diesem Grund kann nach **“Marco Lang“** gegoogelt werden. Dadurch wird die Anzahl der Suchergebnisse auf **45.500** Ergebnisse reduziert. Der Grund für die starke Reduzierung ist, dass ausschließlich die Webseiten vorgeschlagen werden, die den kompletten String “Marco Lang“ beinhalten. Für eine weitere Optimierung der Ergebnisse, wird der Wohnort hinzugefügt, wie in dem Beispiel **“Marco Lang“ “Tettnang“**. Dadurch werden die Suchvorschläge auf lediglich **95** Ergebnisse reduziert. Die URL zu dieser optimierten Suche lautet:

https://www.google.com/search?q=%22Marco+Lang%22+%22Tettnang%22

Nicht nur die Reduzierung der Suchergebnisse, sondern auch das herausfiltern von unerwünschten Webseiten hat einen positiven Effekt auf die zu erstellende Anwendung, da die vorgeschlagenen Seiten in den folgenden Schritten analysiert werden müssen. Das bedeutet, dass jede unerwünschte Seite die allein durch die Suche herausgefiltert werden kann, einen großen Laufzeitvorteil mit sich bringt.

8.4.3 Mit welcher Bibliothek werden Serveranfragen umgesetzt?

Damit eine Person im Internet gesucht werden kann, muss das Programm in der Lage sein, Anfragen an einen Server zu versenden und die dazugehörigen Antwort zu empfangen.
Im Folgenden werden drei Möglichkeiten beschrieben, um Anfragen an einen Server zu versenden. Zum einen ist das die Python Request-Bibliothek, welche sich optimal für HTTP-Anfragen eignet. [Mit15] Zum anderen bietet sich die Verwendung eines automatisierten Webbrowsers an, was mit Hilfe der Selenium Python API realisierbar ist. [Law15] Über diese API ist es möglich auf alle Funktionen des Selenium WebDrivers zuzugreifen. [Mut18] Eine Alternative dazu, ist das Python Framework Scrapy, welches zum Crawlen von Webseiten und Extrahieren von Daten verwendet werden kann. [dev18] Die letzte Möglichkeit stellt die Scrapy Middleware Scrapy-Selenium dar. [Fou19] Dadurch wird die Kommunikation von Scrapy und Selenium ermöglicht.

Für komplizierte Anfragen an einen Server eignet sich die Request-Bibliothek von Python sehr gut. Der Umgang mit Cookies, Header und vielem mehr ist sehr einfach gestaltet. Auch die Generierung des Such-URLs wird von dieser Bibliothek übernommen. Des Weiteren hat Requests einen großen Laufzeit-Vorteil gegenüber dem automatisierten Webbrowser und kann HTTP-Fehlermeldungen empfangen. Allerdings lässt sich mit der Request-Bibliothek keine Javascript-Seite auslesen.
Wenn das Framework Scrapy standardmäßig verwendet wird, können ebenfalls keine Javascript-Seiten ausgelesen werden. Doch in Scrapy lässt sich ein automatisierter Webbrowser einfügen, mit welchem das Auslesen von Javascript-Webseiten möglich ist. Zusätzlich lässt sich mit Scrapy ein effektiver Web Crawler und Web Scraper entwickeln, was für die nächsten Schritte ein erheblicher Vorteil ist.
Aus den erläuternden Gründen, wird das Framework Scrapy mit der Verbindung eines automatisierten Webbrowsers für die Personensuche verwendet. Der automatisierte Webbrowser muss in dem Framework implementiert werden, da auf bestimmte Webseiten mit Javascript direkt zugegriffen wird. Infolgedessen wird die Middelware Scrapy-Selenium verwendet, da sie die eine kompakte Möglichkeit bietet, den automatisierten Webbrowser in Scrapy zu implementieren. Durch diese Kombination aus Scrapy und dem Selenium WebDriver, lassen sich Javascript-Seiten problemlos auslesen.
Zusätzlich zu diesem Framework wird ein unabhängiger Selenium-Wedriver implementiert.

Dieser wird für den Umgang mit den Social-Media-Seiten benötigt, da auf diesen Seiten ein Login vollzogen werden muss. Das hat den Vorteil, dass die Anmeldung in der Session gespeichert wird. Somit muss bei einem erneuten Zugriff auf die selbe Seite keine neue Anmeldung vollzogen werden.

8.4.4 Web Crawler erstellen

Nachdem der Selenium WebDriver in das Scrapy Framework implementiert wurde, kann mit dem crawling begonnen werden. Der Web Crawler hat die Aufgabe ausgewählte Social-Media-Seiten zu durchstöbern und den von Google vorgeschlagenen Webseiten zu folgen. Wie in Bild 8.1 gezeigt, werden zuerst die Informationen über die Zielperson eingelesen. Anschließend werden die Social-Media-Seiten behandelt, wodurch weitere Informationen über die Person gefunden werden können. Die bekannten Informationen über die gesuchte Person, werden zur Generierung der Google-Such-Links verwendet. Mit diesen Links und der Hilfe von der Google-Suchmaschine, werden Webseiten gesucht, die mögliche Inhalte betreffend der Zielperson enthalten. Im nächsten Schritt wird die Google-Webseite mit den Suchergebnissen analysiert und ausgelesen. Dadurch können die URLs für die entsprechenden Webseiten gewonnen werden. Diesen URLs wird anschließend gefolgt, um Informationen über die Zielperson zu gewinnen.

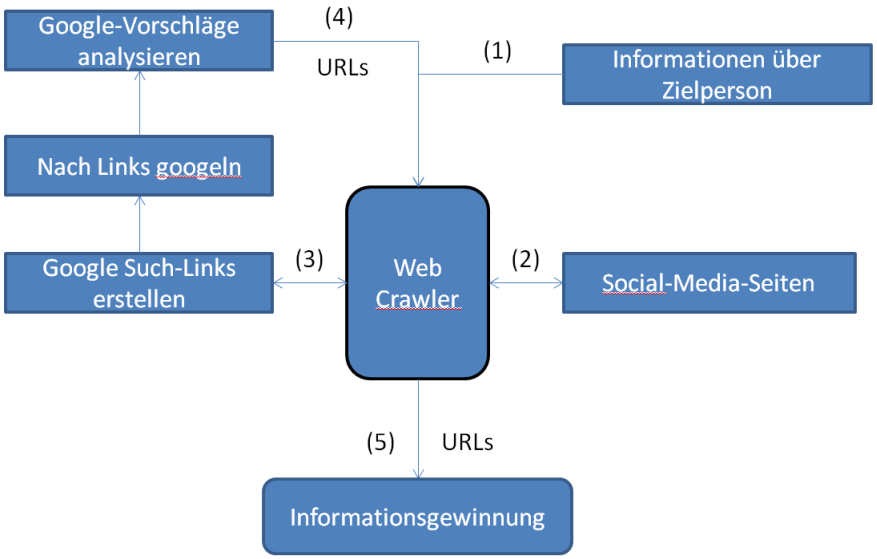

Bild 8.1: Aufbau des Web Crawlers

Social-Media-Seiten

Zu den verwendeten Social-Media-Webseiten gehören Instagram, Facebook, Twitter, Xing und LinkedIn. Für diese Webseiten werden Fake-Accounts und ein eigener Selenium WebDriver erstellt. Dieser automatisierte Webbrowser ist ausschließlich für die Social-Media-Seiten zuständig. Damit vollständige Profile angezeigt werden können, loggt sich dieser automatisch ein. Die gewonnen Informationen werden dem Profil der gesuchten Person hinzugefügt und zusätzlich für die Google-Suche verwendet.
Es kann passieren, dass ausschließlich der Benutzername von einer Person bekannt ist. Weiter Attribute wie Vorname, Nachname und Wohnort sind nicht bekannt. In dem Fall, kann mit einem einzigartigen Benutzername nach diesen Attributen auf der entsprechenden Webseite gesucht werden. Jedoch verwendet nur Instagram und Twitter ein einmaligen Benutzernamen. Aus diesem Grund kann die erweiterte Suche nur auf diesen beiden Plattformen umgesetzt werden.

Login-Formulare

Der Selenium WebDriver muss sich nicht bei jeder Social-Medi-Webseite einloggen. Zu beginn wird kontrolliert, zu welcher Plattform ein Benutzername eingegeben wurde. Dazu gibt es bei dieser Anwendung die Möglichkeit einen Facebook-, Instagram- oder

Twitter-Benutzername einzugeben. Je nach Eingabe meldet sich der Browser auf den entsprechenden Seiten an. Auf den Seiten Xing und LinkedIn wird sich standardmäßig angemeldet. Das hat den Grund, dass diese beiden Seiten immer durchsucht werden sollen. Bei der Umsetzung wird im ersten Schritt die Login-Seite der entsprechenden Plattform angefordert. Die Antwort wird mit Hilfe der BeautifulSoup-Bibliothek nach dem HTML-Tag <input> durchsucht. Allerdings hat nicht jede Webseite den komplett identischen Aufbau. Das bedeutet, dass sich bei diesen Tags die Attribute unterscheiden können. Aus diesem Grund muss nach der Angeforderten-Webseite die Suche der <input>-Tags unterschieden werden.
Die Anmeldung von Instagram, Facebook und LinkedIn sind nahezu identisch. Hier können die zwei gesuchten <input>-Felder mit Hilfe des Attributs *type* identifiziert und gefunden werden. Bei der Twitter-Login-Seite muss anstatt dem *type* nach dem Attribute *class* gesucht werden. Andernfalls findet der Browser keine interaktiven Elemente.
Zum übertragen der Benutzernamen und Passwörter, benötigt der Selenium WebDriver ein Element mit eindeutigem Attribut zur Referenzierung. Dafür dient bei Instragram, LinkedIn und Facebook das vorher gefundene Element mit dem Attribute “id“. Bei Twitter ist das dass Attribut “class[0]“ und bei Xing “name“.

Instagram
Instagram und Twitter verwenden beide einen einzigartigen Benutzername. Dies ist ein riesigen Vorteil für die Identifikation einer Person, wenn der Benutzername bekannt ist. Allerdings ist es ebenfalls möglich eine Person mit ihrem offiziellen Namen zu suchen und zu finden.
Im Fall, dass der Instagram-Benutzername eingegeben wurde, wird die dazugehörige Profilseite angezeigt und nach Informationen durchsucht. Im nächsten Schritt dienen die vorgeschlagenen Freunde, welche in Beziehung zu diesem Profil stehen, als weitere Informationsquelle. Das bedeutet, dass diese Kontakte ebenfalls durchsucht werden. Bei Übereinstimmungen wie zum Beispiel der selben Universität, kann später eine glaubwürdige Phishing-Mail generiert werden.
Falls sich jedoch eine Instagram-Seite unter den Google-Vorschlägen befindet, und die Übereinstimmung des Profils mit der Zielperson nicht klar ist, können die Vorgeschlagenen Kontakte für die Identifizierung der Zielperson genutzt werden. So wird beispielsweise erkannt, wenn ein Freunden aus der selben Stadt vorgeschlagen werd, dass es sich um die

gesuchte Person handeln kann. Diese Methode erzielt kein sicheres Ergebnis. Jedoch kann die Wahrscheinlichkeit erhöht werden, dass es sich um die richtige Person handelt.
Die Profilseite wird mit dem folgenden Link *https://www.instagram.com/username/* angefordert. Weitere Seiten und Einträge der gesuchten Person, unabhängig von der Profilseite,können mit dem Suchbefehl **site:instagram.com "username" -site:instagram.com/username** angezeigt werden. [Baz18] In der Anwendung wird dies mit dem folgenden URL umgesetzt.

https://www.google.com/search?q=site%3Ainstagram.com+%22username%22+-site%3Ainstagram.com%2Fusername&oq=site%3Ainstagram.com+%22username%22+-site%3Ainstagram.com%2Fusername

Die dazugehörigen Suchergebnisse werden anschließend gleich den normalen Google-Suchergebnissen behandelt.

Twitter
Auf der Webseite Twitter wird ausschließlich die Profilseite nach Informationen durchsucht. Dies ist mit dem Link **https://twitter.com/username** möglich.

Facebook
Facebook bietet ein großes Potential um OSINT zu betreiben. Allerdings hat Facebook optimal Algorithmen zur Erkennung von automatisierten Crawlern entwickelt. Aus diesem Grund wurde das Fake-Konto nach wenigen Versuchen gesperrt. Das hat zu Folge, dass auf dieser Plattform nur begrenzt gesucht werden kann, da keine Anmeldung vorgenommen wird. Um das Konto zu entsperren müssten eine Kopie des Ausweises, ein Bild mit erkennbarem Gesicht und eine Handynummer an Facebook übermittelt werden.
Aus diesen Gründen wird Facebook nur dann und ohne Anmeldung verwendet, wenn sich ein Vorschlag unter den Google-Suchergebnissen befindet.

LinkedIn und XING
LinkedIn und XING bieten eine optimale Informationsquelle bezüglich der schulischen und beruflichen Laufbahn der Zielperson. Allerdings gibt es hier keinen einzigartigen Benutzernamen. Demzufolgem, wird eine Person mit dem vollen Namen und dem aktuellen Wohnort gesucht. Dabei wird auf LinkedIn ein Filter angewendet, bei dem ausschließlich Personen aus Deutschland angezeigt werden. Wenn genau eine Person vorgeschlagen wird, wird

dieses gefundene Personenprofil durchsucht. Bei einer Mehrzahl von gefunden Personen werden diese nicht auf Informationen durchsucht, da keine Identifikation möglich ist.
Die Personensuche bei LinkedIn, mit angwandtem Filter, wird mit dem URL *https://www.linkedin.com/search/results/people/?facetGeoRegion=%5B%22de%3A0%22%5D &keywords=vorname%20nachname%20wohnort&origin=FACETED_SEARCH* dargetellt.
Bei Xing sieht der Such-URL wie folgt aus.
https://www.xing.com/search/old/members?hdr=1&keywords=vorname+nachname++wohnort

Webseite mit den Suchergebnissen von Google analysieren

Zur Analyse der Webseite mit den Suchergebnissen von Google, wird der Seitenquelltext benötigt. Mit Hilfe des Quelltextes, können die entsprechenden Links erkannt werden. Der Seitenquelltext wird mit Hilfe der BeautifulSoup-Bibliothek angezeigt.
Das Bild 8.2 stellt ein Suchergebnis von Google dar. Der dazugehörige Quelltext befindet sich in der Darstellung 8.1.

Bild 8.2: Google-Suchergebnis [LLC19]

Im Ausschnitt des Seitenquelltextes 8.1 ist zu sehen, dass der -Container mit der Klasse “g“ einen Hyperlink enthält. URLs oder Links werden mit dem HTML-Tag <a> dargestellt. Dieser Link wird für die Suche benötigt. Deswegen wird genau nach diesem Link gesucht.
Da der -Container bei jedem Suchergebnis identisch ist, kann bei jedem Ergebnis nach dem entsprechenden -Container gesucht werden. Anschließend kann der erste Link in diesem -Tag ausgelesen werden. Dies wird mit Hilfe der BeautifulSoup-Bibliothek umgesetzt.
Um zu erkennen, ob mehrere Seiten mit Suchergebnissen existieren, wird nach bestimmten Hyperlinks gesucht. Diese Links werden über das Attribute “class“ identifiziert. Mit Hilfe

der BeautifulSoup-Bibliothek wird nach dem Klassennamen “fl“ gesucht. Falls weiter Seiten mit Suchergebnissen vorhanden sind, werden die dazugehörigen Links in einer Liste gespeichert. Anschließend wird ihnen gefolgt und die neue Seite wird nach weiteren URLs durchsucht.

```
<div class="g">
  <h3 class="r">
    <a href="/url?q=https://www.fupa.net/spieler/marco-lang
      -1261543.html&amp;sa=U&amp;ved=0ahUKEwiZ3PDGqMvhAhWt
      URUIHU7VAcwQFggUMAA&amp;usg=AOvVaw2QiSMFzScB0Jcvo
      PCisBGw"><b>Marco Lang</b>- Spieler - FuPa - FuPa
    </a>
  </h3>
```

Listing 8.1: Ausschnitt des Quelltextes von einem einem Google-Suchergebnis [LLC19]

8.5 Methoden zum Erkennen von wichtigen Informationen auf einer Webseite

Bei der Suche nach einer ausgewählten Person können verschiedenste Arten von Webseiten gefunden werden. Aus diesem Grund muss das Programm eine gewisse Intelligenz mit sich bringen um die wichtigsten Daten aus einer Seite herauszufiltern. Dabei ist es nicht möglich eine Hartkodierung zu verwenden, um festgelegte Bereiche einer Webseite auszulesen, da jede Webseite eine individuelle Struktur hat.
Die Grundidee zur Lösung diese Problems ist die Analyse des vorliegenden Webseiten-Textes. Eine Methode zur Textanalyse ist die automatisierte Schlüsselwort-Gewinnung. Hierbei wird die HTML-Seite zu einem verwendbaren Text formatiert, wobei alle Sonderzeichen herausgefiltert werden. Für die E-Mail-Erkennung wird der unformatierte Text verwendet, da Sonderzeichen wie “.“ und “@“ dabei von Nöten sind. Im nächsten Schritt werden Schlüsselwörter aus dem formatierten Webseitentext generiert. Möglichkeiten zur automatisierten Schlüsselwortgenerierung sind die Verfahren RAKE 8.5.1 und die Automatic Keyword Extraction mit NLP 8.5.2, welche im Laufe dieser Arbeit detailliert

beschrieben werden.
Nachdem die Schlüsselwörter generiert und in Listen gespeichert wurden, werden Wortsammlung erstellt. Diese Wortsammlungen sind Listen, welche aussagekräftige Schlüsselwörter enthalten und nach Themen kategorisiert sind. Beispiele für den Inhalt der Listen sind alle Hochschulen und Universitäten in Deutschland, Berufsbezeichnungen und Tätigkeiten, Studiengänge, Hobbybezeichnungen und alle Städte und Gemeinden in Deutschland.
Mit diesen Wortsammlungen kann nun die Liste mit den bereits generierten Schlüsselwörtern aus dem Webseitentext verglichen werden. Bei einer Übereinstimmung eines Schlüsselwortes wird das Wort mit der entsprechenden Kategorie vorgemerkt und später in die verwendete Speicherstruktur eingetragen.
Die Wortsammlungen werden mit Hilfe von bekannten Listen im Internet eigenständig befüllt. Als Informationsquelle dienen alle öffentlich frei zugänglichen Listen, die hilfreiche Informationen enthalten.

8.5.1 RAKE

RAKE steht für *Rapid Automatic Keyword Extraction* und stellt eine sehr effiziente Methode zur Schlüsselwortgenerierung dar. Die Funktion von RAKE basiert darin, dass Schlüsselwörter mehrere Wörter mit inhaltlicher Relevanz enthalten, allerdings selten Stoppwörter und Sonderzeichen. [RECC10]
Als Stoppwörter werden Wörter bezeichnet, die sehr oft auftreten und keinen großen Informationsgewinn mit sich bringen. Beispiele dafür sind *und*, *weil*, *der* oder *als*. [Sla]

In einer jungen Wissenschaft wie der Informatik mit ihrer Vielschichtigkeit und ihrer unüberschaubaren Anwendungsvielfalt ist man oftmals noch bestrebt, eine Charakterisierung des Wesens dieser Wissenschaft und Gemeinsamkeiten und Abgrenzungen zu anderen Wissenschaften zu finden. Etablierte Wissenschaften haben es da leichter, sei es, dass sie es aufgegeben haben, sich zu definieren, oder sei es, dass ihre Struktur und ihre Inhalte allgemein bekannt sind.

Bild 8.3: Beispieltext [SS11]

Zu Beginn wird der zu analysierende Text, hier der Beispieltext in Bild 8.3, durch einen Worttrenner in ein Array, bestehen aus möglichen Schlüsselwörtern, aufgeteilt. Das erzeugte Array wird anschließend in Sequenzen von zusammenhängenden Wörtern unterteilt. Dabei erhalten die Wörter in einer Sequenz die gleiche Position und Reihenfolge wie im Ursprungstext und dienen gemeinsam als Kandidatenschlüsselwort. [RECC10]
Nachdem die möglichen Schlüsselwörter identifiziert sind, wird für jeden einzelnen Kandidaten ein Score ausgerechnet. Dieser besteht aus dem Quotient des Grades $deg(w)$ und der Häufigkeit des Vorkommens eines Wortes innerhalb der Kandidaten $freq(w)$. Daraus ergibt sich die Formel:

$$deg(w)/freq(w)$$

Dabei beschreibt der Grad eines Wortes, dass gemeinsame Auftreten mit sich selbst und anderen Schlüsselwörtern. In der Tabelle 8.5.1 ist der Grad für jedes Wort ablesbar, indem die Einträge in der entsprechenden Reihe summiert werden. Beispielsweise Beträgt der Grad des Wortes *"Wissenschaft"* den Wert *3*. Dies ergibt sich aus der Rechnung:

$$2 + 1 = 3$$

Das Wort *"Wissenschaft"* kommt hier selbst zweimal in dem Kandidaten-Array vor und davon einmal in Verbindung mit dem Worten "jungen".
Die Häufigkeit des Vorkommens eines Wortes lässt sich ebenfalls in der Tabelle 8.5.1 ablesen. Allerdings muss hier in der Reihe und Spalte des jeweiligen Wortes nachgeschaut werden. Für das Wort *"Wissenschaft"* beträgt die Häufigkeit des Vorkommens den Wert *3*.
Zusammenfassend kann gesagt werden, dass *deg(w)* die Kandidaten bevorzugt, welche oft und in langen Schlüsselwörtern, die mehrere Wörter enthalten, vorkommen. Dies bedeutet, dass beispielsweise *deg(etabliert)* eine höhere Bewertung als *deg(informatik)* bekommt, obwohl beide Wörter gleich oft im Text vorkommen. Dagegen wird bei *freq(w)*, ausschließlich die Häufigkeit des Vorkommens bewertet. Bei der Formel *deg(w)/freq(w)* werden die Wörter bevorzugt, welche überwiegend in langen Kandidatenwörtern vorkommen. Diese Formel bietet dadurch einen guten Mittelweg zur Schlüsselwortgewinnung. Ein Beispiel dafür sind die Wörter *"Wissenschaften* und *"allgemein"*. Hier ist der Quotient von *deg(allgemein)/freq(allgemein)* höher als von *deg(Wissenschaften)/freq(Wissenschaften)*,

obwohl die Häufigkeit des Wortes *“Wissenschaften“* höher und der Grad gleich hoch ist. [RECC10]

Durch das genannte Verfahren und der Formel *deg(w)/freq(w)* für die Bewertung, ergeben sich die im Bild 8.4 befindenden Kandidaten mit den dazugehörigem Endbewertungen. [RECC10]

	wissenschaften	wissenschaft	sei	etablierte	informatik	aufgegeben	gemeinsamkeiten	oftmals	charakterisierung	jungen	inhalte	allgemein	bekannt	struktur	wesens	bestrebt	unüberschaubaren	anwendungsvielfalt	definieren	abgrenzungen	leichter	finden	vielschichtigkeit
wissenschaften	2			1																			
wissenschaft		2								1													
sei			1																				
etablierte	1			1																			
informatik					1																		
aufgegeben						1																	
gemeinsamkeiten							1																
oftmals								1															
charakterisierung									1														
jungen		1								1													
inhalte											1	1	1										
allgemein											1	1	1										
bekannt											1	1	1										
struktur														1									
wesens															1								
bestrebt																1							
unüberschaubaren																	1	1					
anwendungsvielfalt																	1	1					
definieren																			1				
abgrenzungen																				1			
leichter																					1		
finden																						1	
vielschichtigkeit																							1

Tabelle 8.1: Co-occurance

inhalte allgemein bekannt (9.0), unüberschaubaren anwendungsvielfalt (4.0), jungen wissenschaft(3.5), etablierte wissenschaften (3.5), wissenschaften (1.5), wissenschaft (1.5), wesens (1.0), vielschichtigkeit (1.0), struktur (1.0), sei (1.0), oftmals (1.0), leichter (1.0), informatik (1.0), gemeinsamkeiten (1.0), finden (1.0), definieren (1.0), dass (1.0), charakterisierung (1.0), bestrebt (1.0), aufgegeben (1.0), abgrenzungen (1.0)

Bild 8.4: Schlüsselwörter mit zugehörigem Score

8.5.2 Automatic Keyword Extraction mit NLP

Bei dieser Methode wird der vorliegende Text in die einzelnen Wörter unterteilt. Dabei wird eine Liste mit potentiellen Schlüsselwörtern erstellt, in der *Stoppwörter* und Sonderzeichen herausgefiltert werden. Bei den Schlüsselwörtern handelt es sich nicht ausschließlich um ein Wort sondern auch um Wortsequenzen. Sogenannte N-Gramme bestehen aus einer festgelegten Anzahl von Wörtern. Dies hat den Vorteil, dass nicht nur Schlüsselwörter bestehend aus einem Wort erstellt werden können, sondern auch Schlüsselwörter mit Fragmenten eines Textes. Diese Art von Schlüsselwort wird benötigt um Informationen wie *Hochschule Ravensburg-Weingarten* herauszulesen.
Erweiternd kann die Anzahl der Schlüsselwörter mit dem Verfahren von Stemming reduziert werden. Durch die Verwendung von ergänzende Regeln wie, eine Mindestanzahl von Buchstaben in einem Wort, können die Schlüsselwörter weiter begrenzen.

8.6 Bewertung der Methoden zum Herausfiltern von wichtigen Informationen auf einer Webseite

RAKE stellt eine fertige Methode dar, um Schlüsselwörter, die den Inhalt eines Textes in kurz wiedergeben, zu erstellen. Dabei hat ein Anwender kaum Möglichkeiten eigene Implementierungen vorzunehmen, da vieles vorgegeben ist. In der zu erstellenden Anwendung soll jedoch nicht der Inhalt eines Textes in Schlüsselwörter zusammengefasst werden, sondern es wird nach informationsreichen Wörtern gesucht. Aus diesem Grund ist jedes einzelne Wort aus dem Webseiten-Text von Bedeutung. Dies spricht gegen RAKE, da es nur die selbst errechnenden Favoriten-Schlüsselwörter zur Verfügung stellt. Dadurch werden viele Wörter nicht in Betracht gezogen oder für weiterführende Bearbeitungen

nicht bereitgestellt. Darüber hinaus ist die Berechnung eines Scores für diese Anwendung nicht notwendig.
Die Methode zur automatisierten Schlüsselwortgenerierung mit NLP bringt dagegen ein eigene Implementationsmöglichkeit mit sich. Das bedeutet, es kann selbst festgelegt werden, aus wie vielen Wörtern die Schlüsselwörter bestehen sollen. Des Weiteren wird jedes einzelne Wort in Betracht gezogen und verwendet.
Die Suche nach einer E-Mail-Adresse im Text lässt sich bei beiden Methoden hinzufügen. Jedoch wird aus den eben genannten Vorteilen, die Information mit Hilfe der Methode zur automatisierten Schlüsselwortgewinnung mit NLP herausgefiltert.

8.7 Implementierung der Methoden zum Herausfiltern von wichtigen Informationen auf einer Webseite

8.7.1 Text formatieren

Bevor die Schlüsselwörter generiert werden können, muss der Text in ein verwertbares Format umgewandelt werden. Aus diesem Grund wird der Seitenquelltext zuallererst mit Hilfe des Python-Skripts html2text zu einem ASCII Plaintext umgewandelt. [Fou18] Anschließend werden Zeilenumbrüche und Sonderzeichen aus diesem Text herausgefiltert. Einzelne Wörter und Zahlen die weniger als 2 Zeichen beinhalten, werden ebenfalls aussortiert. Nachdem der Text in ein verwertbares Format umgewandelt wurde, kann mit der Umsetzung für die automatisierte Schlüsselwortgenerierung mit NLP begonnen werden.

8.7.2 Wortsammlungen erstellen

Die Ausführlichkeit der Wortsammlungen ist sehr wichtig für das Projekt, da die Informationsgewinnung abhängig von diesen Listen ist. Das bedeutet, umso besser diese Datenbanken erstellt werden, umso mehr Informationen können gewonnen werden.

Erstellung der Wortsammlungen

Eine Wortsammlung ist eine CSV-Datei die manuell erstellt und befüllt wird. Es gibt eine Wortsammlung mit allen Universität- und Hochschulnamen, Städten und Gemeinden in Deutschland, Vereinskürzel, Bezeichnungen für Tätigkeiten und Hobbys sowie einer Liste mit den bekanntesten Firmennamen.

Wie werden sie am effektivsten verglichen?

Es gibt keine Methode die den Vergleich der Schlüsselwörter mit den Wörtern der Datenbanken verbessern kann, da jedes einzelne Schlüsselwort mit jedem einzelnen Wort aus der Datenbank verglichen werden muss. Suchalgorithmen

8.7.3 Automatic Keyword Extraction mit NLP

Durch das *Natural Language Toolkit* (NLTK) von Pyhton ist es möglich, den vorhandenen Webseitentext zu analysieren.
Zu Beginn wird der vorhandene Text in einzelne Wörter zerlegt und in eine Liste gespeichert. Aus diesen Wörtern werden die *stopwords* der deutschen als auch der englischen Sprache herausgefiltert. Dadurch verringert sich die Anzahl der gesamten Wörter im Text um einen sehr großen Teil.
Im nächsten Schritt kann die Liste mit den entsprechenden Wortsammlungen verglichen werden. Die Wortsammlung, welche die möglichen Institutionen enthält, wird nicht mit dieser Liste vergleichen. Die Problematik besteht darin, dass sich die Anzahl der Wörter für die Institutionen variieren kann. Für diesen bestimmten Fall, wird das Wort in dem Webseitentext ohne Fragmentierung und Formatierung gesucht. Dadurch kann bei dieser Suche ein Laufzeitnachteil entstehen, welcher aber nicht von Bedeutung ist.

8.7.4 Suche nach dem Geburtsjahr der Zielperson

Das Geburtsjahr ist für die Generierung der E-Mail wichtig. Viele Personen verwenden eine Kombination aus dem bürgerlichen Namen und dem Geburtsjahr als lokalen Teil der E-Mail-Adresse. Aus diesem Grund wird speziell nach dem Geburtsjahr in den generierten Schlüsselwörtern aus Kapitel 8.7.3 gesucht.
Dazu wird eine Suche nach einer vierstelligen Zahl, welche größer als 1900 und kleinergleich 2019 ist, durchgeführt. Beim Fund einer Zahl, werden fünfzehn Schlüsselwörter vor und hinter der vermutlichen Jahreszahl kontrolliert. Falls dabei das Wort “Geburtsdatum“, “Alter“, “geboren“, “Geburtsort“, “Geburtstag“, “born“, oder “birth“ vorkommt, wird das entsprechende Jahr als Geburtsjahr der Zielperson festgelegt. Die Implementierung zur Erkennung eines Geburtsjahrs ist in Listing 8.2 dargestellt.

```
regex_string = "(geburtsdatum)|(alter)|(geboren)|(geburtsort)|
        (geburtstag)|(born)|(birth)"
for year in all_years_in_text:
  vistited_elements = 0
  max_number_of_visited_elements = 15
  #  to get all occurrences of this year
  occurrences = [i for i, x in enumerate(keywords) if x == year]
  for position_of_year in occurrences:
    while (position_of_year+vistited_elements) < len(keywords)-1 and
      vistited_elements <= max_number_of_visited_elements and
      position_of_year is not -1:
      index_behind = position_of_year+ vistited_elements
      index_front = position_of_year - vistited_elements
      if re.match(r""+regex_string, keywords[index_behind]):
        print("Behind: Geburtsjahr wurde gefunden", year)
        return year
      elif re.match(r""+regex_string, keywords[index_front]):
        print("Front: Geburtsjahr wurde gefunden", year)
        return year
      vistited_elements += 1
return -1
```

Listing 8.2: Algorithmus zur Suche nach dem Geburtsjahr

8.7.5 E-Mail-Adressen erkennen und herauslesen

Zu beginn wird der unformatierte Webseitentext in Textfragmente zerlegt. Getrennt wird der Text bei einem Leerzeichen. Anschließend werden die erzeugten Fragmente mit einem regulären Ausdruck nach einer gültigen E-Mail-Adresse durchsucht. Bei einer Übereinstimmung des regulären Ausdrucks, wird der korrekte Teilstring, somit die E-Mail-Adresse, ausgelesen. Der Algorithmus zu diesem Vorgang ist in Listing 8.3 aufgezeigt.

```
for fragment in email_words:
  mail_regex = re.search('(.*((@)|(\(at\))).*\.(de|com|net)).*',
  fragment)
  if mail_regex:
    print("Email found:", mail_regex.group(1))
```

Listing 8.3: Teil des Algorithmuses zum auslesen einer E-Mail-Adresse

Es werden nur die E-Mail-Adressen herausgesucht, welche einen Bezug zur Zielperson haben. Aus diesem Grund wird der lokale Teil aller gefunden Adressen mit dem Vor-und Nachnamen der Zielperson vergleichen. Mit Hilfe der difflib von Python, lassen sich diese beide Sequenzen vergleichen und es wird ein prozentuale Übereinstimmung berechnet. Zur Differenzierung, ob eine E-Mailadresse eine Verbindung zum Opfer hat oder nicht wird eine Prozent-Grenze bestimmt. Die Grenze wurde aus den Ergebnissen von zahlreichen Tests auf die Zahl 0,4 % festgelegt. Das folgende Beispiel soll die Methode zur Erkennung von korrekten E-Mail-Adressen verdeutlichen.
In diesem Beispiel heißt die Person “Max Mustermann“ und es werden zwei E-Mail-Adresse gefunden. Die erste Adresse lautet *MusterMax@gmail.com* und die zweite *MartaFrau@gmx.de*. Im ersten Schritt wird der Name “Max Mustermann“ zu einem String “maxmustermann“ umgewandelt. Im nächsten Schritt werden die lokalen Namen aus den E-Mail-Adressen herausgelesen und gleichzeitig in Kleinbuchstaben umgewandelt. In diesem Fall wäre das “mustermax“ und “martafrau“. Anschließend werden die lokalen Namen der E-Mail-Adressen mit dem erzeugten Namensstring der Zielperson vergleichen. Dabei erreicht die lokale Namen *mustermax* eine prozentuale Übereinstimmung von 0,73 % mit dem Namensstring und *martafrau* 0,27 %. Da die Prozent-Grenze bei 0,4 % beträgt, wird die zweite E-Mail-Adresse verworfen.

8.7.6 Auswahl der gewonnenen Information

Die gefundenen Schlüsselwörter einer Webseite, werden in einer Liste gespeichert. Nachdem eine Seite vollständig durchsucht wurde, wird mit der Formel 8.1 eine prozentuale Wertung für das Vorkommen eines Wortes in der Liste berechnet.

$$\frac{\text{Vorkommen eines Wortes}}{\text{Anzahl aller gefundenen Wörter in der Liste}} \tag{8.1}$$

Die Schlüsselwörter werden anschließend mit dem dazugehörigen Score in einer neuen Liste gespeichert. Jedes Wort kommt dabei nur einmal vor. Eine beispielhafte Liste ist nachstehend dargestellt.

```
[['fussball', 0.7],['basketball', 0,2],['fechten', 0.1]]
```

Hierbei ist zu sehen, dass das Wort “Fußball“ sieben Mal öfter als das Wort “Fechten“ auf der Webseite vorgekommen ist. Für jede durchsuchte Seite wird solch eine Liste erstellt und anschließend zu einer großen Liste zusammengefügt. Dabei bleibt die Struktur bestehen, damit erkannt wird, welche Wörter von unterschiedlichen Webseiten kommen. Ein Beispiel hierfür ist die folgende Liste.

```
[[['fussball', 0.7],['basketball', 0,2],['fechten',0.1]], [['fussball',
0.5],['volleyball', 0.5]]
```

In dieser Liste befinden sich die gewonnenen Informationen aller Webseiten für eine Kategorie. Hier wäre es die Kategorie “Hobby“. Für jede dieser kategorisierten Listen, muss nun ein Element bestimmt werden, welches am wahrscheinlichsten eine Verbindung zu der Zielperson hat. Dazu wird die Formel 8.2 verwendet. Wobei beachtet werden muss,

dass nur die Elemente summiert werden, bei denen das Schlüsselwort identisch ist.

$$\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\frac{liste[i][j][1]}{n}$$

mit

$$\begin{aligned} n &= \text{Anzahl der Elemente von Liste} \\ m &= \text{Anzahl der Elemente von Liste[i]} \\ i &= \text{Element in Liste} \\ j &= \text{Element von Liste[i]} \end{aligned} \tag{8.2}$$

In dem aufgezeigten Beispiel würde das zu der folgenden Liste führen.

```
[['fussball', 0.6],['basketball',0,1],['fechten',0.05], ['volleyball', 0.25]]
```

Aus dieser Liste kann nun das Schlüsselwort mit der höchsten Wertung gewählt und dem Personenobjekt hinzugefügt werden. In diesem Fall wäre dass das Wort “Fußball“. Falls zwei Wertungen gleich hoch sind, wird das erste Wort ausgewählt.

8.8 Methoden zum Erkennen einer Person

Bei jeder einzelnen Suche, besteht die Herausforderung darin, zu erkennen, wann es sich um die gesuchte Person handelt. Durch die große Anzahl an verfügbaren Informationen im Internet, besteht eine hohe Wahrscheinlichkeit, dass Personen mit sehr ähnlichen Profilen gefunden werden.
Aus diesem Grund werden Maßnahmen getroffen um die gesuchte Person zu erkennen. Dafür ist der erste Schritt die Anzahl der Suchergebnisse zu reduzieren. Dies ist durch den Ansatz der Personensuche im Kapitel 8.2 möglich. Dabei wird abhängig von der eingegebenen Information die Suche variiert. Des Weiteren kann durch eine Optimierung des Such-URLs 8.4.2, die Personensuche verfeinert und somit die Ergebnisse verbessert werden. Durch diese Maßnahmen steigt die Wahrscheinlichkeit, dass es sich um die richtige

Person handelt.
Im zweiten Schritt können die folgenden Methoden angewendet werden.

8.8.1 Identifikationsschlüssel verwenden

Bei der Personensuche wird mit Hilfe der eingegebenen Daten nach einer Person gesucht. Dabei können fehlerhafte Webseiten von Google vorgeschlagen werden. Fehlerhaft bedeutet hier, dass die Webseiten einen Inhalt repräsentieren, welcher nicht mit der gesuchten Person übereinstimmt.
Um dem entgegenzuwirken können bekannte Informationen als Identifikationsschlüssel verwendet werden. Allerdings müssen diese einzigartige Daten sein. Dazu zählt beispielsweise die E-Mail-Adresse oder Benutzernamen von den Plattformen Instagram und Twitter. Der vollständige Name ist nicht einzigartig und dient deswegen nicht als Identifikationsschlüssel. Dass bedeutet, dass es mehrere Personen mit dem selben vollständigen Namen geben kann.
Um eine Person zu identifizieren, zur welcher keine einzigartigen Informationen bekannt sind, können Kombinationen aus den angegebenen Daten erstellt werden. Diese Kombinationen dienen in dem Fall als Identifikationsschlüssel. Im folgenden sind alle möglichen Kombinationen aufgelistet.

Vorname, Nachname, Wohnort;
Vorname, Nachname, Geburtsjahr;
Vorname, Nachname, Institution;

Der Webseitentext kann anschließend auf das Vorkommen des Identifikationsschlüssels kontrolliert werden. Wenn der Text nur eine dieser Kombination beinhaltet, wird diese Seite für die Informationsgewinnung verwendet. Andernfalls wird die Webseite verworfen.

8.8.2 Kontaktanalyse

Hier kann die Suche erweitert werden, indem auf soziale und berufliche Verbindungen der Zielperson eingegangen wird. Das heißt, dass bekannte Kontakte der gesuchten Person

ebenfalls durchsucht und ausgewertet werden. Als Kontaktquellen können Facebook-Freunden, FuPa-Teammitglieder, Instagram-Follower oder Xing-Kontakte dienen.
Durch die erwähnte Methode können weitere Informationen gewonnen werden. Diese sind zur Unterscheidung von Profilen nützlich.

8.9 Bewertung der Methoden zur Personenidentifizierung

Beide Methoden zur Identifizierung einer Person bringen eine Verbesserungen der Ergebnisse mit sich. Die Wahrscheinlichkeit wird erhöht, dass es sich um die korrekte Person handelt.
Die Methoden unterscheiden sich in der Wirksamkeit und in der Laufzeit. Durch die Verwendung von Identifikationsschlüsseln wird die Anzahl von Fehlinformationen in dem Profil der gesuchten Person reduziert. Allerdings können gleichzeitig wichtige Informationsquellen ignoriert werden, wenn diese den Kriterien nicht entsprechen. Bei der Kontaktanalyse werden jedoch keine Informationsquellen ignoriert. Es werden weitere Informationen gesammelt. Diese sind zusätzlich zur E-Mail-Generierung von Vorteil. Das Ergebnis bei der Verwendung der Kontaktanalyse ist nicht optimal. Es kann nicht davon ausgegangen werden dass das Ergebnis für unmittelbar für die Person spricht. Beim betrachten der Laufzeit, kann davon ausgegangen werden, dass die Kontaktanalyse deutlich mehr Zeit und Ressourcen benötigt.
Es werden beide Methoden umgesetzt, da sie einen positiven Effekt auf die Anwendung haben.

8.10 Implementierung der Personenidentifizierung

8.10.1 Identifikationsschlüssel verwenden

Zu beginn der vorläufigen Inhaltskontrolle werden die Eingaben abgefragt. Dadurch wird erkannt, zu welche Daten Informationen vom Benutzer eingegeben wurden. Anschließend

werden mit diesen Daten alle möglichen Kombinationen aus Kapitel 8.8.1 erstellt. Es sind allerdings nur die Kombinationen mögliche, für die die Daten bekannt sind.
Für die Suche des Vornamen und Nachnamen wird ein String erzeugt der beide Attribute kleingeschrieben beinhaltet. Ein korrekter String ist “max mustermann“. Infolgedessen wird der Webseitentext zu einem String umgewandelt. Anschließend wird kontrolliert ob sich der String bestehend aus Vornamen und Nachnamen und das entsprechenden Attribute, beispielsweise der Wohnort, in dem Webseitentext befindet. Wenn diese Abfrage korrekt ist wird die Webseite weiter behandelt und es kann nach Information gesucht werden.

8.10.2 Kontaktanalyse

Welche Seiten eignen sich zur Kontaktanalyse?

Diese Methode funktioniert auf der Webseite LinkedIn nicht. Es gibt dort keine Möglichkeit die Kontakte der gesuchten Person anzuzeigen. Bei Xing kann ein Nutzer einstellen, ob diese Kontaktanzeige freigegeben wird oder nicht. Dadurch sind die Kontakte bei vielen Usern nicht erkennbar. Facebook, Twitter und Instagram bieten die Möglichkeit die Kontakte der gesuchten Person anzuzeigen. Allerdings wird dafür ein Account benötigt.
Für diese Methode eignen sich somit die Seiten Twitter, Xing, Facebook und Instagram. Wie in Kapitel 8.4.4 beschrieben, wird für Facebook kein Account angelegt. Dadurch ist es nicht möglich Kontakte auf dieser Webseite anzuzeigen. Um die Funktion der Methode aufzuzeigen, wird ausschließlich die Webseite Instagram verwendet.

Instagram Kontakte durchsuchen

Zuallererst wird unterschieden, ob das Profil der gesuchten Person privat oder öffentlich ist. Bei einem öffentlichen Profil, können alle Abonnenten und abonnierte Profile angezeigt werden. Die Abonnenten und abonnierte Profile können sich unterscheiden. Im Gegensatz dazu, werden bei einem privaten Profil, nur eine begrenzte Anzahl von Profilen vorgeschlagen. Des Weiteren kann bei einem privaten Profil nicht unterschieden werden, ob die Abonnenten oder die abonnierten Profile angezeigt werden sollen.

Von den gefunden Followern wird jedes einzelne Profil durchsucht, bis eine Übereinstimmung mit der Zielperson gefunden wurde. Eine Übereinstimmung bedeutet, dass auf diesem Profil ein Teil mit dem Opferprofil identisch ist. Beispielweise kann das die selbe Universität oder der selbe Wohnort sein. Sobald dies gefunden wurde, kann die Suche beendet werden. Wenn keine Profilinformation übereinstimmt, wird nicht ausgeschlossen, dass es sich trotzdem um die gesuchte Person handelt.
Ein Fund einer identischen Information ist keine vollständiger Beweis, dass es sich um die richtige Person handelt. Allerdings erhöht sich die Wahrscheinlichkeit für die Aussage, dass es sich bei diesem Profil um die gesuchte Person handelt.

Wie werden Kontakte ausgelesen

Im ersten Schritt entscheidet der Algorithmus, ob es sich um ein privates oder öffentliches Profil handelt. Dies wird realisiert, indem nach einem String auf der Webseite gesucht wird. Der String lautet “Diese Konto ist privat“. Wenn diese Zeichenfolge gefunden wird, handelt es sich um ein privates Konto. Andernfalls um ein öffentliches.
Damit die Links zu den Kontakt-Profilseiten auf einer privaten Seite herausgelesen werden können, wird ein scrollbarer Container ausgelesen. Dieser Container beinhaltet die vorgeschlagenen Kontakte und zwei Buttons. Wie im Bild 8.5 zu sehen, kann mit den beiden Buttons nach rechts und links gewischt werden. Sobald die Links zu den aktuell angezeigten Profilen ausgelesen wurden, wird auf den rechten Button geklickt. Dies wird mit einem vorgetäuschten Mausklick des Selenium WebDrivers realisiert. Durch diese Schritt-für-Schritt-Methode können alle vorgeschlagenen Kontakte ausgelesen werden. Andernfalls werden nur die aktuell angezeigten Profile geladen und gefunden.

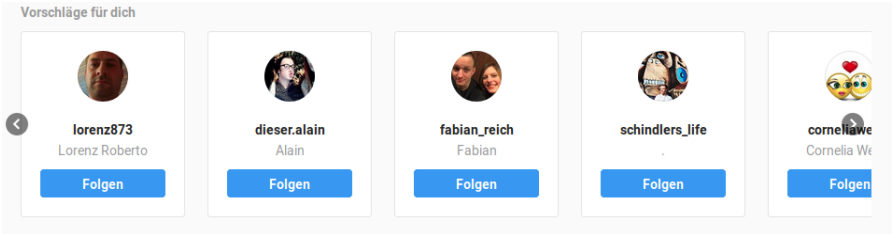

Bild 8.5: Container mit Profil-Vorschlägen

Falls es sich um ein öffentlich frei zugängliches Profil handelt, kann eine Liste der abonnierten Kontakte angezeigt werden. Hierbei handelt es sich um ein scrollbares Pop-Up-Fenster 8.6. Vergleichbar zur Methode bei einer privaten Profilseite, wird hier ebenfalls Schritt-für-Schritt durchgescrollt. Dadurch wird jedes einzelne Profil geladen und der dazugehörig Link, zur dieser Profilseite, kann dadurch ausgelesen werden.

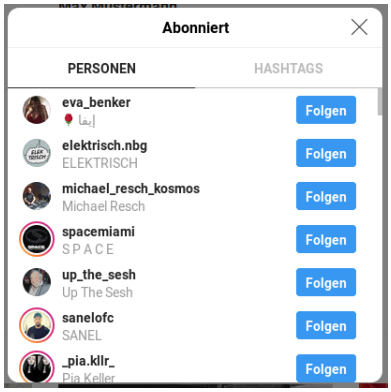

Bild 8.6: Pop-up-Fenster mit abonnierten Profilen

Die Herausforderung besteht darin, dass nicht zu schnell gescrollt werden darf. Aus diesem Grund wird ein Algorithmus 8.4 verwendet, welcher einem menschlichen Verhalten ähneln soll. Hierbei wird zuallererst das Pop-up-Fenster gesucht und festgelegt. Anschließend wir die Anzahl der abonnierten Profile gezählt. Die Anzahl der Profile wird dazu verwendet, dass der Algroithmus weiß wie weit nach unten geblättert werden muss, um alle Profil zu laden.
Im ersten Schritt, wird das Fenster nur ein sechstel des möglichen Bereichs nach unten gescrollt. Dadurch werden weitere Profile geladen. Wenn direkt nach ganz unten geblättert wird, wäre dies beim ersten Scrollvorgang zu schnell. In diesem Fall werden keine Kontakte geladen. Es werden lediglich Profil von sehr bekannten Instagram-Usern angezeigt, die von dieser Person abonniert wurden.
In den nächsten Schritte wird das Fenster jeweils ganz nach unten verschoben. Dadurch werden alle Profile geladen. Sobald alle Links bekannt sind, wird eine URL zu den entsprechenden Profilseiten erstellt. Diese Seiten werden anschließen wie jede andere Seite ausgelesen und nach Information durchsucht. Infolgedessen wird die gewonnene Information jedes Profils mit der Information der Zielperson verglichen. Die Suche wird bei einem

beliebigen Treffer beendet. Anschließend wird die gefunden Information mit dem Namen des Benutzers der Profilseite gespeichert. Diese abgespeicherten Daten können später zur E-Mail-Generierung verwendet werden.

```
# Find the pop-up window
pop_up = self.browser.find_element_by_xpath('/html/body/div[2]
    /div/div[2]')
# find number of followers
all_following = int(self.browser.find_element_by_xpath("//li[2]
    /a/span").text)
# scroll down the page
for i in range(int(all_following / 6)):
  if i == 0:
    self.browser.execute_script("arguments[0].scrollTop =
        arguments[0].scrollHeight/5", pop_up)
    time.sleep(2)
  else:
    self.browser.execute_script("arguments[0].scrollTop =
        arguments[0].scrollHeight", pop_up)
  time.sleep(random.randint(500, 1000) / 1000)
```

Listing 8.4: Herunterscrollen des Pop-up Fensters

8.11 Speicherung der gewonnenen Daten

Die gespeicherten Daten werden von verschiedenen Klassen benötigt. Aus diesem Grund muss es möglich sein, dass andere Klassen auf die Speicherstruktur zugreifen können. Des Weiteren wird eine gute Struktur vorausgesetzt, damit auf einzelne Attribute der Person zugegriffen werden kann. Es ist nicht notwendig, dass die Daten nach Programmende abrufbar sind. Infolgedessen wird keine externe Speicherung in einer Datenbank oder in einer Datei vorausgesetzt.
Eine mögliche Speicherung der Daten wäre in einer SQL-Datenbank. Alternativ könnten die Personendaten in einer externen Datei oder mit Hilfe einem Personenobjekt gespeichert werden.

Eine SQL-Datenbank bringt eine gute Speicherstruktur mit sich. Allerdings sind mit einer SQL-Datenbank aufwendigere Speicher-und Abrufvorgänge verbunden. Die externe Speicherung in einer Datei wie CVS oder TXT ist keine Anforderung. Auf eine SQL-Datenbank sowie auf eine externe Datei, lässt sich mit jeder Klasse darauf zugegriffen. Dennoch wird ein Personenobjekt verwendet. Dadurch werden keine unnötigen Speicher- und Lesezugriffe benötigt. Darüber hinaus lässt sich das Personenobjekt an die entsprechenden Klassen übergeben.

8.11.1 Implementierung der Personenklasse

Die gewonnenen Daten werden in einem Personenobjekt, wie in Listing 8.5 dargestellt, gespeichert. Dabei werden die vom Anwender eingegebenen Daten direkt in das Personenobjekt übertragen. Die gewonnen Informationen werden in Form einer Liste hinzugefügt. Falls bei der Kontaktanalyse ein Treffer gemacht wurde, kann diese Information in dem Attribute “contacts_information gespeichert werden. Hierbei wird zuerst der vollständige Kontaktname und anschließend die übereinstimmende Information gespeichert. Ein Beispiel hierfür wäre [“Max Mustermann“,“Fußball“]

```
class Person(object):
  def __init__(self):
    self.first_name = input("Vorname: ")
    self.second_name = input("Nachname: ")
    self.place_of_residence = input("Wohnort: ")
    self.year_of_birth = input("Geburtsjahr: ")
    self.institution = input("Institution: ")
    self.instagram_name = input("Instagram Benutzername: ")
    self.facebook_name = input("Facebook Benutzername: ")
    self.twitter_name = input("Twitter Benutzername: ")
    self.input_email = input("E-Mail-Adresse: ")
    self.occupation = []
    self.hobbies = []
    self.universities = []
    self.founded_mails = []
    self.locations = []
    self.contacts_information = []
```

Listing 8.5: Personklasse

9 OSINT einer großen Anzahl von Person

Für die *real-world* Simulation eines Phishing-Mail-Angriffs ist die Webseite FuPa die Informationsquelle. Zu Beginn dieser Anwendung muss zuallererst die Webseite analysiert werden.

9.1 Analyse der Webseite FuPa

Damit die Auswahl für die Art des Web Crawlers getroffen werden kann, muss zuerst einmal der komplette Aufbau einer Webseite bekannt sein. Dadurch können beide Möglichkeiten von Web Crawler korrekt bewertet werden.
Für die Analyse der Webseite wird das Entwicklertool eines Webbrowsers verwendet, welches standardmäßig mitgeliefert wird. Es wird von der Startseite begonnen. Diese Startseite stellt eine Karte von Deutschland dar. Struktur geht von Deutschland, Bundesland, Kreisen,...

9.2 Interner Web-Crawler für FuPa

Bei diesem Verfahren gibt es keine automatisierte Suche nach Informationen. Jedoch gibt es eine automatisierte Suche nach internen Links. Diese interne Suche kann mit einem Web Crawler realisiert werden. Dieser hat das Ziel, sich mit den gefundenen Links durch die Webseite zu hangeln. Dadurch soll jedes Personenprofil gefunden werden. Für diese

Aufgabe kann der Web Crawler hartkodiert werden. Eine weitere Möglichkeit ist die Erstellung eines Crawlers, welcher unabhängig von der Webseiten nach Links suchen kann. In Vorbereitung darauf, wird der Aufbau der FuPa-Webseite analysiert.

9.2.1 Methoden zur Suche nach Profilen

Hartkodierung

Webseiten unabhängig

9.2.2 Bewertung der Methoden zum finden von Links

Hartkodierung

9.2.3 Implementierung des internen Web-Crawlers

Damit die Webseite *www.fupa.net* komplett nach Spielerdaten durchsucht werden kann, wird ein interner Web Crawler benötigt. Dieser wird sich anhand den internen Links, über die ganze Seite hinweg, durchhangeln.
Die Funktionsweise des Web Crawlers besteht darin, dass das Programm auf der Startseite von Fupa.net beginnt nach links zu suchen und diesen folgt.

Probleme bei der Erstellung

1. Python hat einen verkürzten und erkennbaren Standard http-Header. Dieser wird von vielen Administratoren geblockt und mit der Fehlermeldung 451 erkennbar gemacht. 451 for legal reason
2. Honeypots gewollt oder ungewollt, hier Kalender darstellung mit links zu neuen Jahren die eine sehr hohe bis überhaupt keine Begrenzung haben.

3. Rekursion erreicht schnell die Maximale tiefe von 1500.
4. Zu langsamer Algorithmus

Lösungen

1. http-Header selber konfigurieren
2. Links mit möglichen Honeypots nicht beachten
3. Stack Klasse schreiben damit keine Rekursion benötigt wird
4. Algorithmus anpassen auf fupa-Webseite

9.2.4 Handhabung mit Links

CVS, Links mit Spielerinformationen speichern.

9.3 Methode zum Auslesen der Informationen

Zum Auslesen einer großen Menge an Daten wird ein Web Scraper erstellt. Dieser könnte für die ausgewählte Webseite hartkodiert werden. Eine Alternative dazu, wäre die Analyse des Webseitentextes, was dem Ansatz 8.5 von der Suchfunktion einer ausgewählten Person entsprechen würde.

9.3.1 Hartkodierung

9.3.2 Gleich OSINT einer Person

9.4 Bewertung der Methoden zum Auslesen der Informationen

Die Suchfunktion für eine große Anzahl von Personen kann *hartkodiert* werden und benötigt dadurch keine Textanalyse, da der Aufbau der Webseite im voraus bekannt ist. Das bedeutet, dass das Programm genau weiß wo welche Information auf einer Webseite steht. Auf der Seite "*www.fupa.net*" befindet sich beispielsweise der Name einer Person immer an der gleichen Position einer Tabelle. Das bringt den Vorteil mit sich, dass der Text nicht analysiert werden muss und das Programm genau weiß, was mit diesen Daten gemacht werden muss. Zusätzlich entsteht eine sehr performante Methode zur Auslesung von personenbezogenen Daten.

9.5 Implementierung der Methode zum Auslesen der Information

9.6 Möglichkeiten zur Speicherung der Daten

Für die Speicherung der gewonnen Daten kann eine SQL-Datenbank erstellt werden. Als Alternative kann eine Datei angelegt werden, bei der alle Daten zu allen Personen gut strukturiert gespeichert werden können. Eine Möglichkeit dafür wäre das Dateiformat *CSV* oder *TXT*.

9.7 Bewertung der Arten zur Speicherung der Daten

9.8 Umsetzung der Datenspeicherung

10 Erstellung einer Phishing-Mail

In diesem Kapitel wird die Umsetzung zur Erstellung einer Phishing-Mail beschrieben.

10.1 Implementierung der Methode zur Generierung der E-Mail-Adressen

Für den den zu entwickelnden Algorithmus wird eine eigene Klasse erstellt. Diese Klasse ist ausschließlich für die Generierung der E-Mail-Adressen zuständig.

10.1.1 Funktion des eigenen Algorithmus

Der lokale Teil einer E-Mail-Adresse befindet sich vor dem At-Zeichen. Dieser kann aus verschiedensten Daten bestehen. Allerdings wird in den meisten Fällen der bürgerlichen Namen verwendet. [Med17] Aus diesem Grund verwendet der Algorithmus die Personenattribute Vorname, Nachname und das Geburtsjahr.
Im ersten Schritt wird kontrolliert, welche Daten bekannt sind. Im Idealfall sind das alle drei Attribute. Im zweiten Schritt wird festgelegt aus welchen Daten der lokale Teil bestehen kann. Im Folgenden sind möglichen Kombinationen aufgezeigt.

Vorname;
Nachname;
Vorname, Nachname;
Vorname, Nachname, vollständiges Geburtsjahr;
Vorname, Nachname, Kurzform von Geburtsjahr;

Ein lokaler Teil kann somit aus mehreren Daten bestehen. Es kann vorkommen, dass anstatt “Max Mustermann“ “Mustermann Max“ als lokaler Namen verwendet wird. Aus diesem Grund wir für jeden lokalen Teil, der aus mehreren Daten besteht, eine Permutation ohne Wiederholung angewendet. Dadurch werden alle möglichen Kombinationen aus den Daten gewonnen, da bei der Zusammensetzung der Daten zusätzlich auf die Reihenfolge geachtet wird. Außerdem werden bei der Zusammensetzung der Daten die bekannten Trennzeichen “.“,“_“ und “-“ hinzugefügt. Jedoch gibt es ebenfalls jede Kombination ohne Trennzeichen. Die lokale Namen werden anschließend in einer Liste gespeichert.
Für den Domainteil werden die bekannte Mailprovider in Deutschland verwendet. Dazu gehören die Provider GMX, WEB.DE, Gmail, T-Online, Freenet und 1&1. [Anb19]. Das bedeutet, es wird für jeden lokalen Namen eine E-Mail-Adresse mit den jeweiligen Mailprovidern und der Landeskennung “de“ erzeugt. Die folgende Tabelle zeigt die erzeugten E-Mail-Adressen des Algorithmus für die Daten “Marco“, “Lang“ und “1995“. Es sind allerdings nur die Mailadressen für die Provider WEB.DE, Gmail und Freenet aufgelistet.

marco@web.de	marco@gmail.com	marco@freenet.de
lang@web.de	lang@gmail.com	lang@freenet.de
marcolang@web.de	marcolang@gmail.com	marcolang@freenet.de
marco.lang@web.de	marco.lang@gmail.com	marco.lang@freenet.de
marco_lang@web.de	marco_lang@gmail.com	marco_lang@freenet.de
marco-lang@web.de	marco-lang@gmail.com	marco-lang@freenet.de
langmarco@web.de	langmarco@gmail.com	langmarco@freenet.de
lang.marco@web.de	lang.marco@gmail.com	lang.marco@freenet.de
lang_marco@web.de	lang_marco@gmail.com	lang_marco@freenet.de
lang-marco@web.de	lang-marco@gmail.com	lang-marco@freenet.de
marcolang1995@web.de	marcolang1995@gmail.com	marcolang1995@freenet.de
marco.lang.1995@web.de	marco.lang.1995@gmail.com	marco.lang.1995@freenet.de
marco_lang_1995@web.de	marco_lang_1995@gmail.com	marco_lang_1995@freenet.de
marco-lang-1995@web.de	marco-lang-1995@gmail.com	marco-lang-1995@freenet.de
marco1995lang@web.de	marco1995lang@gmail.com	marco1995lang@freenet.de
marco.1995.lang@web.de	marco.1995.lang@gmail.com	marco.1995.lang@freenet.de
marco_1995_lang@web.de	marco_1995_lang@gmail.com	marco_1995_lang@freenet.de
marco-1995-lang@web.de	marco-1995-lang@gmail.com	marco-1995-lang@freenet.de
langmarco1995@web.de	langmarco1995@gmail.com	langmarco1995@freenet.de
lang.marco.1995@web.de	lang.marco.1995@gmail.com	lang.marco.1995@freenet.de
lang_marco_1995@web.de	lang_marco_1995@gmail.com	lang_marco_1995@freenet.de
lang-marco-1995@web.de	lang-marco-1995@gmail.com	lang-marco-1995@freenet.de
lang1995marco@web.de	lang1995marco@gmail.com	lang1995marco@freenet.de
lang.1995.marco@web.de	lang.1995.marco@gmail.com	lang.1995.marco@freenet.de
lang_1995_marco@web.de	lang_1995_marco@gmail.com	lang_1995_marco@freenet.de
lang-1995-marco@web.de	lang-1995-marco@gmail.com	lang-1995-marco@freenet.de
1995marcolang@web.de	1995marcolang@gmail.com	1995marcolang@freenet.de

1995.marco.lang@web.de	1995.marco.lang@gmail.com	1995.marco.lang@freenet.de
1995_marco_lang@web.de	1995_marco_lang@gmail.com	1995_marco_lang@freenet.de
1995-marco-lang@web.de	1995-marco-lang@gmail.com	1995-marco-lang@freenet.de
1995langmarco@web.de	1995langmarco@gmail.com	1995langmarco@freenet.de
1995.lang.marco@web.de	1995.lang.marco@gmail.com	1995.lang.marco@freenet.de
1995_lang_marco@web.de	1995_lang_marco@gmail.com	1995_lang_marco@freenet.de
1995-lang-marco@web.de	1995-lang-marco@gmail.com	1995-lang-marco@freenet.de
marcolang95@web.de	marcolang95@gmail.com	marcolang95@freenet.de
marco.lang.95@web.de	marco.lang.95@gmail.com	marco.lang.95@freenet.de
marco_lang_95@web.de	marco_lang_95@gmail.com	marco_lang_95@freenet.de
marco-lang-95@web.de	marco-lang-95@gmail.com	marco-lang-95@freenet.de
marco95lang@web.de	marco95lang@gmail.com	marco95lang@freenet.de
marco.95.lang@web.de	marco.95.lang@gmail.com	marco.95.lang@freenet.de
marco_95_lang@web.de	marco_95_lang@gmail.com	marco_95_lang@freenet.de
marco-95-lang@web.de	marco-95-lang@gmail.com	marco-95-lang@freenet.de
langmarco95@web.de	langmarco95@gmail.com	langmarco95@freenet.de
lang.marco.95@web.de	lang.marco.95@gmail.com	lang.marco.95@freenet.de
lang_marco_95@web.de	lang_marco_95@gmail.com	lang_marco_95@freenet.de
lang-marco-95@web.de	lang-marco-95@gmail.com	lang-marco-95@freenet.de
lang95marco@web.de	lang95marco@gmail.com	lang95marco@freenet.de
lang.95.marco@web.de	lang.95.marco@gmail.com	lang.95.marco@freenet.de
lang_95_marco@web.de	lang_95_marco@gmail.com	lang_95_marco@freenet.de
lang-95-marco@web.de	lang-95-marco@gmail.com	lang-95-marco@freenet.de
95marcolang@web.de	95marcolang@gmail.com	95marcolang@freenet.de
95.marco.lang@web.de	95.marco.lang@gmail.com	95.marco.lang@freenet.de
95_marco_lang@web.de	95_marco_lang@gmail.com	95_marco_lang@freenet.de
95-marco-lang@web.de	95-marco-lang@gmail.com	95-marco-lang@freenet.de
95langmarco@web.de	95langmarco@gmail.com	95langmarco@freenet.de
95.lang.marco@web.de	95.lang.marco@gmail.com	95.lang.marco@freenet.de
95_lang_marco@web.de	95_lang_marco@gmail.com	95_lang_marco@freenet.de
95-lang-marco@web.de	95-lang-marco@gmail.com	95-lang-marco@freenet.de

10.2 Validität der generierten Mail-Adressen prüfen

10.2.1 Methoden zum Prüfen der Validität

Die erzeugten Adressen werden anschließend auf Validität geprüft. Hierfür gab es früher eine *VRFY* Anfrage von SMTP. Mit dieser Anfrage konnte eine angegebene E-Mail-Adresse überprüft werden. Allerdings wurde der Dienst von Spammern ausgenutzt und wird dadurch von den meisten SMTP-Servern nicht mehr zu Verfügung gestellt. [BPH+10]

Demnach muss die Validität auf einem anderen Weg geprüft werden. Eine Möglichkeit zur Prüfung ist die Verwendung bereitgestellter Webseiten, bei der die zu prüfenden E-Mail-Adresse angegeben werden kann. Eine anschließende Rückmeldung verrät dann, ob die Adresse verwendet wird oder nicht. Eine Webseite dafür wäre "*https://centralops.net/co/*". Als Alternative dazu, ist die Entwicklung eines Skriptes, welches die Validität der Adresse prüft.

Im Fall, dass mehrere Adressen von diesem Adresspool gültig sind, kann nach mit Hilfe dieser Mail-Adressen nach Einträgen im Internet gesucht werden. Wenn es eine Übereinstimmung mit der Zielperson gibt, wird diese E-Mail ausgewählt. Andernfalls wird an jede gültige Adresse eine Phishing-Mail gesendet.

10.2.2 Bewertung: Validität Prüfen

Für eine bessere Laufzeit des Programms, wird ein Skript zur Überprüfung der Adressen auf Verfügbarkeit und Gültigkeit, verwendet.

10.3 Implementierung der E-Mail-Muster

Ein E-Mail-Muster entspricht einem Lückentext, bei dem die entsprechenden Lücken mit den gewonnen Daten ergänzt werden. Die Texte müssen so erstellt werden, dass sie die Zielperson ansprechen. Aus diesem Grund, muss für jede Kombination der gewonnenen Daten ein Muster zur Verfügung stehen. Infolgedessen, stellt sich die Frage, wie die E-Mail-Texte möglichst passend kategorisiert werden können.

10.3.1 Kategorien erstellen

Die Muster können in zwei große Kategorien unterteilt werden. Es gibt eine private und eine berufliche Kategorie. Der unterschied zwischen privat und beruflich besteht in der Art und Weise wie ein Text geschrieben wird. Genaugenommen bedeutet das, dass ein privates Muster in einer Alltagssprache und ein berufliches in einer formelleren Sprache

erstellt wird. Dies beiden Kategorien haben weitere Unterkategorien, welche verschiedene Kombinationen aus den personenbezogenen Daten verwenden.
Um die Kategorie zu erkennen, werden zu beginn Abfragen gestartet. Dadurch wird kontrolliert, welche Daten bekannt sind. Im Fall, dass die Firma, Universität oder die Tätigkeit bekannt ist, wird ein berufliches Muster gewählt. Wenn keines dieser Attribute bekannt ist, wird ein privates Muster ausgewählt.

Berufliche Muster erstellen

Private Muster erstellen

10.4 Versenden einer Phishing-E-Mail

11 Evaluation der Implementation

12 Schlussbemerkungen und Ausblick

12.1 Wie kann eine Person weiter identifiziert werden?

Durch die Google Bildersuche ist es möglich, anstatt einem Suchbegriff ein Bild zu verwenden und nach diesem zu suchen. Dabei kann ein zu suchendes Bild selbst hochgeladen oder ein URL angegeben werden. Bei dem Ergebnis kann es sich um ein ähnliches Bild oder eine Webseite, die das Bild enthält, handeln.
Als Alternative zur Google-Bildersuche kann eine Bilderkennungssoftware verwendet werden um Personen zu identifizieren bzw. zu unterscheiden.

12.1.1 Zeitrahmen wird mit Beachtet

Wie kann Alter der Webseite herausgefunden werden

Der Webseitentext kann nach Datums suchen und diese mit dem angegebenen Geburtsjahr verglichen werden. Dabei kann erkannt werden, ob das theoretische Alter des Artikels mit dem Alter der Person übereinstimmen kann. Möglicherweise können Metadaten von der Webseite ausgelesen werden. möglicherweise über domain

Bereits umgesetzt

Jahr nach copyright wird ausgelesen, wenn das nicht vorhanden werden alle Jahreszahlen genommen und ein durchschnitt ausgerechnet

12.1.2 Zeitraum beachten

Eine Methode für das Erkennen von Personen kann das Beachten von Zeiträumen sein. Dabei fließt das Alter der Zielperson mit in die Suche ein. Das bedeutet, dass nach dem Alter der Webseite gesucht wird, indem Jahreszahlen aus dem Webseitentext ausgelesen werden. Dadurch wird erkannt, ob der Zeitrahmen des Artikels oder das Erstellungsdatum einer Webseite mit dem Alter der Person grundsätzlich übereinstimmt.

12.2 Adressgenerierung

12.2.1 Wenn Firma bekannt

12.3 Keyword Extraction mit Hilfe von Machine Learning

In der Theorie ist es möglich, ein Neuronales Netz mit den Begriffen zu trainieren und eine Kategorisierung durchzuführen. Dabei entsteht ein Netz, welches selbst entscheiden würde, in welche Kategorie ein Wort fällt. Das Wort “Fußball“ müsste dadurch in die Kategorie Hobby eingeordnet werden.

12.4 Email-adressen

Adressen von Micheal Bazzel mit verwenden wie ml@web.de

12.5 Absender-Adresse

Spoofing, Kontakte von Fupa oder Instagram nutzen.

A Ein Kapitel des Anhangs

Abkürzungsverzeichnis

3

Literatur

[AH19] Ads-Hilfe, Google: *URL-Parameter*. https://support.google.com/google-ads/answer/6277564?hl=de, 2019. Abrufdatum: 26.02.2019.

[All18] Allensbach, IfD: *Meistgenutzte Informationsquellen der Bevoelkerung in Deutschland im Jahr 2018*. https://de.statista.com/statistik/daten/studie/171257/umfrage/normalerweise-genutzte-quelle-fuer-informationen/, 2018. Abrufdatum: 18.01.2019.

[Anb19] *Bei welchem Anbieter haben Sie Ihr Haupt-E-Mail-Postfach?* https://de.statista.com/statistik/daten/studie/170371/umfrage/nutzung-von-e-mail-domains/, 2019. Abrufdatum: 04.02.2019.

[Ang18] *Haben Sie gro Angst davor, dass Siel Opfer von Datendiebstahl im Internet, also der missbrhlichen Verwendung Ihrer persnlichen Daten durch Dritte, werden?* https://de.statista.com/statistik/daten/studie/886892/umfrage/angst-vor-einem-datendiebstahl-im-internet-in-deutschland/, 2018. Abrufdatum: 22.02.2019.

[Baz] Bazzell, Michael: *Email Assumptions*. https://inteltechniques.com/osint/email.html. Abrufdatum: 01.02.2019.

[Baz18] Bazzell, Michael: *Open Source Intelligence Techniques: Resources for Searching and Analyzing Online Information*. CreateSpace Independent Publishing Platform, USA, 6th , 2018.

[BKL09] Bird, Steven, Ewan Klein Edward Loper: *Natural language processing with Python: analyzing text with the natural language toolkit*. Ö'Reilly Media, Inc.“, 2009.

[Boh14] Bohnensteffen, Marcel: *Die alternativlose Suchmaschine*. https://www.handelsblatt.com/unternehmen/it-medien/google-die-alternativlose-suchmaschine/11061626-all.html, 2014. Abrufdatum: 24.02.2019.

[BPH+10] Balduzzi, Marco, Christian Platzer, Thorsten Holz, Engin Kirda, Davide Balzarotti Christopher Kruegel: *Abusing social networks for automated user profiling. International Workshop on Recent Advances in Intrusion Detection*, 422–441. Springer, 2010.

[Cal13] Caldwell, Tracey: *Spear-phishing: how to spot and mitigate the menace.* Computer Fraud & Security, 2013(1):11–16, 2013.

[CH15] Christopher Hadnagy, Michele Fincher: *Phishing Dark Waters: The Offensive and Defensive Sides of Malicious E-mails.* 2015.

[Cor17] Corp, LinkedIn: *Ueber LinkedIn.* `https://about.linkedin.com/de-de`, 2017. Abrufdatum: 19.02.2019.

[dev18] developers, Scrapy: *Scrapy at a glance.* `http://doc.scrapy.org/en/latest/intro/overview.html`, 2018. Abrufdatum: 28.02.2019.

[DSG] DSGVO: *Art. 4 DSGVO Begriffsbestimmungen.* `https://dsgvo-gesetz.de/art-4-dsgvo/`. Abrufdatum: 09.01.2019.

[EAD09] Eldesouki, Mohamed I, W Arafa K Darwish: *Stemming techniques of Arabic language: Comparative study from the information retrieval perspective.* The Egyptian Computer Journal, 36(1):30–49, 2009.

[Fir] FireEye, Inc: *Spear-Phishing-Angriffe ? Warum sie erfolgreich sind und wie sie gestoppt werden knnen.*

[Fou18] Foundation, Python Sofware: *html2text 2018.1.9.* `https://pypi.org/project/html2text/`, 2018. Abrufdatum: 15.03.2019.

[Fou19] Foundation, Python Sofware: *scrapy-selenium 0.0.7.* `https://pypi.org/project/scrapy-selenium/`, 2019. Abrufdatum: 16.03.2019.

[Gmb16] GmbH, FuPa: *Was ist eingentlich diese "FuPa"?* `https://www.fupa.net/berichte/sachsenliga-was-ist-eigentlich-dieses-fupa-570543.html`, 2016. Abrufdatum: 19.02.2019.

[Goo19] Google: *Refine web searches.* `https://support.google.com/websearch/answer/2466433?hl=en`, 2019. Abrufdatum: 27.02.2019.

[Had11] Hadnagy, Christopher: *Social Engineering: The Art of Human Hacking.* 2011.

[Jam05] James, Lance: *Phshing Exposed: Uncover Secrets from the Dark Side.* 2005.

[Law15] Lawson, Richard: *Web scraping with Python.* Packt Publishing Ltd, 2015.

[Lit16] Litzel, Nico: *Was ist Natural Language Processing?* https://www.bigdata-insider.de/was-ist-natural-language-processing-a-590102/, 2016. Abrufdatum: 10.02.2019.

[LLC19] LLC, Google: *marco lang tettnang - Google-Suche*. https://www.google.com/search?q=marco+lang+tettnang, 2019. Abrufdatum: 12.04.2019.

[Med17] Media, United Internet: *Brgerlicher Name als E-Mail-Adresse in terreich und der Schweiz 2017*. https://de.statista.com/statistik/daten/studie/745611/umfrage/buergerlicher-name-als-e-mail-adresse-in-oesterreich-und-der-schweiz/, 2017. Abrufdatum: 31.10.2018.

[Mit01] Mitnick, Kevin D.: *The art of deception:controlling the human elemnet of security*. 2001.

[Mit15] Mitchell, Ryan: *Web Scraping with Python: Collecting Data from the Modern Web*. 2015.

[Mut18] Muthukadan, Baiju: *Selenium with Python*. https://selenium-python.readthedocs.io/installation.html#introduction, 2018. Abrufdatum: 27.02.2019.

[NW18] Nordrhein-Westfalen, Verbraucherzentrale: *Phishing-Radar: Aktuelle Warnungen*. https://www.verbraucherzentrale.nrw/wissen/digitale-welt/phishingradar/phishingradar-aktuelle-warnungen-6059, 2018. Abrufdatum: 29.10.2018.

[PH] Philipp, Jonas Nathanael Gerhard Heyer: *Multi-Label Klassifikation am Beispiel sozialwissenschaftlicher Texte*.

[RECC10] Rose, Stuart, Dave Engel, Nick Cramer Wendy Cowley: *Automatic keyword extraction from individual documents*. Text Mining: Applications and Theory, 1–20, 2010.

[RFC94] *Uniform Resource Locators (URL)*. https://tools.ietf.org/html/rfc1738#section-3.1, 1994. Abrufdatum: 27.02.2019.

[SE] SE, XING: *Was ist XING?* https://faq.xing.com/de/startseite-allgemeines/was-ist-xing. Abrufdatum: 19.02.2019.

[SG12] Sharma, Arvind Kumar PC Gupta: *Study & Analysis of Web Content Mining Tools to Improve Techniques of Web Data Mining*. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 1(8):pp–287, 2012.

[Sla] Slavin, Tim: *Stop Words*. `https://www.kidscodecs.com/stop-words/`. Abrufdatum: 29.01.2019.

[SS11] Schubert, Sigrid Andreas Schwill: *Didaktik der Informatik. Didaktik der Informatik*, 1–30. Springer, 2011.

[Ste96] Steele, Robert David: *Open Source Intelligence: What Is It? Why Is It Important to the Military?* American Intelligence Journal, 35–41, 1996.

[The01] Thelwall, Mike: *A web crawler design for data mining*. Journal of Information Science, 27(5):319–325, 2001.

[uDsiNe15] Netz e.V., DATEV und Deutschland sicher im: *Verhaltensregeln zum Thema "Social Engineering"*. 2015.

[W3S] W3Schools: *HTML URL Encoding Reference*. `https://www.w3schools.com/tags/ref_urlencode.asp`. Abrufdatum: 27.02.2019.

[Wik19] Wikipedia: *FuPa*. `https://de.wikipedia.org/wiki/FuPa`, 2019. Abrufdatum: 25.02.2019.