Karlsruhe Institute of Technology

Clustering von Internetdiensten für aktive Ontologien

Masterarbeit
von

Philipp Lingel

An der Fakultät für Informatik
Institut für Programmstrukturen
und Datenorganisation (IPD)

Erstgutachter:	Prof. Dr. Walter F. Tichy
Zweitgutachter:	Prof. Dr. Ralf H. Reussner
Betreuender Mitarbeiter:	Dipl.-Inform. Martin Blersch
Zweiter betreuender Mitarbeiter:	Dipl.-Inform.Wirt. Mathias Landhäußer

Bearbeitungszeit: 23.03.2015 – 22.09.2015

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Die Regeln zur Sicherung guter wissenschaftlicher Praxis im Karlsruher Institut für Technologie (KIT) habe ich befolgt.

Karlsruhe, 22.09.2015

. . . . . .

(**Philipp Lingel**)

Kurzfassung

Für die Verwendung von Internetdiensten ist die sofortige Benutzbarkeit ein entscheidendes Kriterium. Der Nutzer soll ohne zeitaufwendige Schulungen einen Dienst in Anspruch nehmen können. Um diesem Ziel gerecht zu werden, wird der Ansatz verfolgt, dass der Nutzer den Dienst über die natürliche Sprache bedienen kann. Dazu müssen beim Aufruf einer Internetseite zunächst die verschiedenen Dienste erkannt und klassifiziert werden, um anschließend für den jeweiligen Dienst eine aktive Ontologie zu entwerfen, die die interaktive Kommunikation zwischen Nutzer und Dienst ermöglicht. Zielsetzung dieser Ausarbeitung ist zunächst die Erzeugung eines Werkzeugs, das autonom die verschiedenen Internetdienste gruppiert und anschließend der Entwurf eines Konstruktionsplans zur Modellierung einer aktiven Ontologie. Dafür werden autonom Merkmale eines Datensatzes an Internetdiensten erzeugt, die sich auf die Webseitenbeschreibung, die Syntax und Semantik der Dienstelemente sowie die Anzahl der Elemente beziehen. Auf Basis dieser Merkmale werden die Internetdienste mittels eines Clusteringalgorithmus klassifiziert. Die dabei entstandenen Internetdienstkategorien werden anschließend analysiert und ein Konstruktionsplan entworfen, der Regen für das Ableiten der einzelnen Formularelemente, eines Internetdienstes und einer ganzen Internetdienstkategorie enthält. Auf dieser Grundlage kann für die jeweilige Kategorie eine aktive Ontologie modelliert werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Die computergestützte Erkennung und Verarbeitung von natürlich gesprochener Sprache ist ein stark wachsender Bereich. Sie bieten anstatt der üblichen „*klicken-und-ausführen*“ Interaktion eine intuitive Kommunikation mit dem System, die dem Nutzer die Bedienbarkeit erleichtert. So kann beispielsweise mit dem Handy bereits über eine Konversation in natürlicher Sprache ein Termin erstellt und in den Kalender eingetragen werden. Der Nutzer kann somit alltägliche Dienstleistungen (Wecker stellen, Termin erstellen, usw.), die das Assistenzprogramm unterstützt, mit natürlicher Sprache bedienen.

Die Assistenzsoftware *Siri* von Apple implementiert diese Funktionalität über die Modellierung von aktiven Ontologien. Dazu wird manuell von einem Experten für jede dieser Dienstleistungen eine aktive Ontologie nach der Arbeit [Guz08] von D. Guzzoni erstellt. Anhand der aktiven Ontologie werden Nutzeranfragen bis zu ihrer Verarbeitung gespeichert. Durch die Speicherung kann einerseits bei einer unvollständigen Anfrage das System die noch benötigten Informationen nachfragen und andererseits können Angaben korrigiert werden.

Beim Entwurf einer aktiven Ontologie müssen alle Dienstelemente, die Nutzerinformationen abfragen, modelliert werden. Der Aufwand dieser manuellen Modellierung stellt jedoch ein Problem dar. Daher entstand dieses Projekt, das die manuelle Modellierung automatisiert. Abbildung 1.1 skizziert die einzelnen Schritte von einem Internetdienst zu einer aktiven Ontologie. Dabei ist die Grundidee, einen Internetdienst zu klassifizieren und anschließend für die Internetdienste einer Kategorie eine gemeinsame aktive Ontologie zu erstellen.

1.1. Zielsetzung

Ziel dieser Arbeit ist es, gemäß der Abbildung 1.1, den ersten Schritt und zwar die Klassifikation von unbekannten Internetdiensten zu automatisieren. Zudem soll ein Konstruktionsplan entworfen werden, der Regeln für die manuelle Abbildung einer Internetdienstkategorie zu einer aktiven Ontologie vorgibt. Zur vollständigen Automatisierung müsste also noch ein System entwickelt werden, das anhand des Konstruktionsplans die Abbildung von einer Kategorie auf eine aktive Ontologie vornimmt.

Für die Klassifikation von unbekannten Internetdiensten soll ein maschinell lernendes System entwickelt werden. Dies klassifiziert unbekannte Internetdienste anhand bekannter

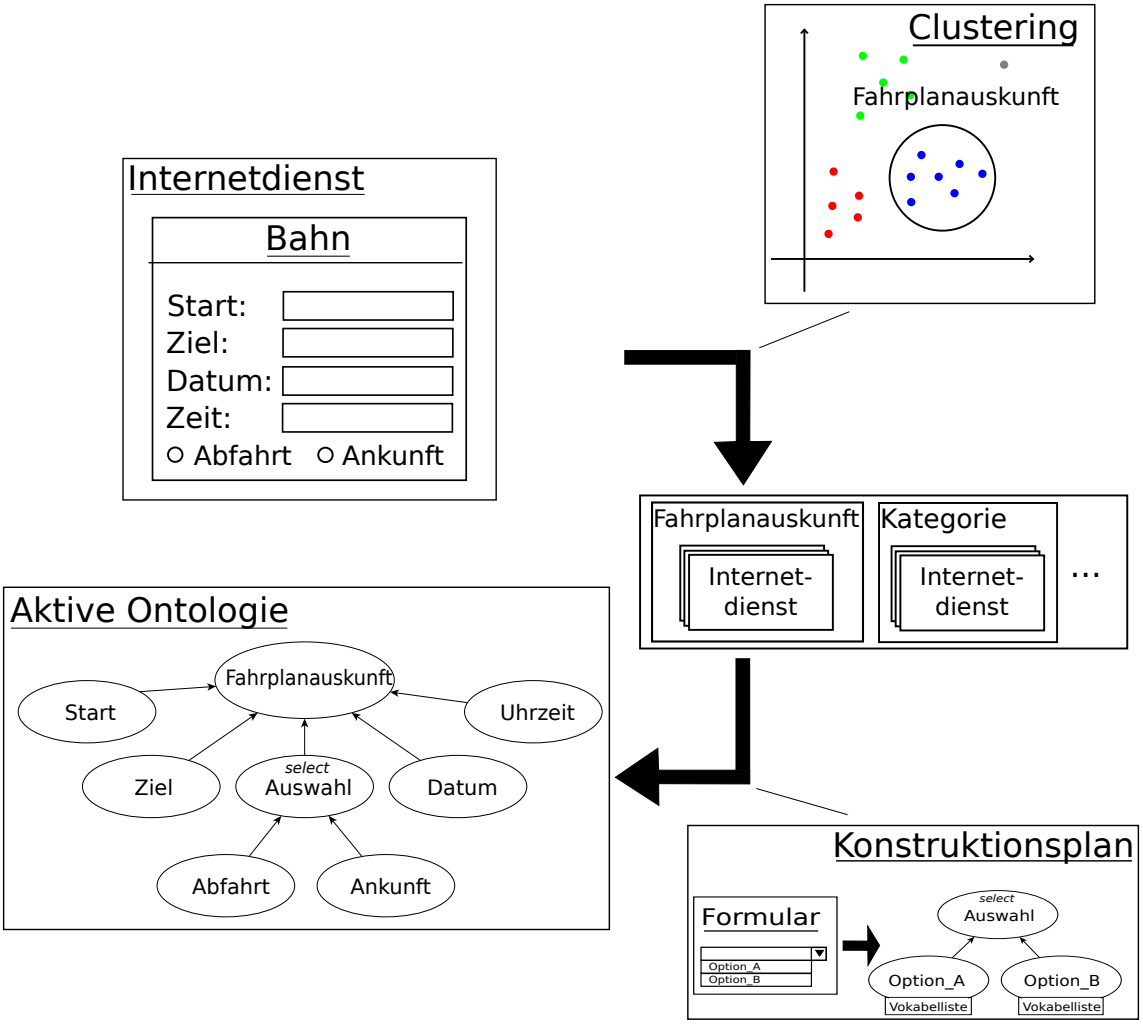


Abbildung 1.1.: Autonome Modellierung einer aktiven Ontologie: Der Internetdienst wird zuerst über einen Clusterer klassifiziert. Danach wird für die jeweils zugeordnete Gruppierung die aktive Ontologie erzeugt. Zusätzlich werden die Beziehungen zwischen den Dienstelementen und der aktiven Ontologie skizziert.

Abbildungen von Internetdiensten auf Kategorien. Somit ist dieses System durch die Hinzunahme bekannter Klassifikationen beliebig erweiterbar. Zu diesem Zweck muss in einem ersten Schritt ein möglichst repräsentativer Datensatz an Diensten aus dem Internet gewonnen werden. Dies erfolgt über einen Webcrawler, der die einzelnen Internetdienste herausfiltert. Durch die flexible Erweiterbarkeit des Systems ist die Sammlung einer vollständigen Menge kein Kriterium für den Webcrawler. Aus den gefundenen Internetdiensten soll das System autonom Merkmale erzeugen. Basis für die Merkmale bilden die Metadaten der Internetseite, auf der der Dienst angeboten wird, die Anzahl an Formularelementen und die Syntax und Semantik der einzelnen Formularelemente. Die daraus erzeugten Merkmale werden von einem Clusteringalgorithmus genutzt, um die Internetdienste einer Kategorie zuzuordnen.

Ziel des zweiten Schritts ist der Entwurf eines Konstruktionsplans, mit dem manuell für die gefundenen Internetdienstkategorien jeweils eine aktive Ontologie modelliert werden kann. Zur Modellierung einer aktiven Ontologie muss bekannt sein, welche Nutzerinformation unbedingt benötigt wird und welche optional angegeben werden kann. Zur Bestimmung dieser Daten müssen die einzelnen Internetdienste einer Kategorie verglichen und ana-

lysiert werden. Zusätzlich soll der Konstruktionsplan noch Regeln für das Ableiten der einzelnen Formularelemente, eines Internetdienstes und einer ganzen Internetdienstkategorie enthalten.

1.2. Beispiel

Im Beispiel der Abbildung 1.1 wird die abstrakte Zielsetzung für einen exemplarischen Internetdienst schematisch erläutert. Dieser Internetdienst liefert einen Zugfahrplan unter der Angabe zweier Orte für den Start und das Ziel, eines Datums und einer Uhrzeit sowie einer Auswahl, ob es sich um die Abfahrt oder Ankunftszeit handelt. Aus dem Internetdienst werden nun Merkmale extrahiert, wie beispielsweise das Vorkommen eines Elements für die Abfrage des Zielorts oder die Anzahl an Formularelementen. Mittels dieser Merkmale klassifiziert der Clusteringalgorithmus den Internetdienst als Fahrplanauskunft und bildet anhand des Konstruktionsplans eine aktive Ontologie für diese Kategorie ab. Diese Ontologie enthält an der Wurzel einen Kombinationsknoten, der die Daten aller seiner Kindknoten sammelt. Die Kindknoten bestehen aus vier obligatorischen Sensorknoten, die für die Erfüllung des Dienstes einen Wert liefern müssen. Außerdem wird über zwei weitere Sensorknoten die Auswahl zwischen Abfahrts- und Ankunftszeit erfasst. Die beiden Knoten werden über einen Selektionsknoten zusammengefasst, wobei dieser auch einen Kindknoten der Wurzel bildet.

1.3. Struktur der Arbeit

Die Arbeit gliedert sich wie folgt auf. Zuerst werden die Grundlagen sowie der aktuelle Forschungsstand im Bereich dieser Arbeit vorgestellt. Danach wird im Analyse-Kapitel das einzelne Vorgehen zur Erfüllung des Zieles beschrieben. Dabei wird der Weg vom Aufbau des Datensatzes an Internetdiensten über die Merkmalsextraktion bis hin zum Klassifikation der Internetdienste. Zudem werden die einzelnen gefundenen Internetdienstkategorien analysiert sowie ein Konstruktionsplan zur Abbildung der Kategorien auf eine aktive Ontologie entworfen. Im nächsten Kapitel wird aufbauend auf der Analyse der dazugehörige Entwurf des Webcrawlers, der den Datensatz an Internetdiensten aufbaut, und der Clusterer, der die Dienste gruppiert, entwickelt. Zusätzlich wird in diesem Kapitel auf die Implementierung der Programme eingegangen. Die Ergebnisse, die der Clusterer liefert, werden im darauffolgenden Kapitel evaluiert. Zum Schluss wird das Ergebnis dieser Arbeit zusammengefasst und ein kurzer Ausblick auf sich daraus ableitende Fragestellungen skizziert.

2. Grundlagen

Dieses Kapitel legt für das weitere Verständnis die relevanten Grundlagen dar. An erster Stelle wird der zentrale Begriff des Internetdienstes eindeutig definiert. Darauf aufbauend wird die Struktur einer URL, die den Zugriff auf einen Internetdienst bietet, erläutert. In diesem Zusammenhang wird danach noch der Aufbau der Auszeichnungssprache HTML, in der ein Internetdienst implementiert ist, skizziert. Zusätzlich zum Aufbau wird der für die Implementierung eines Internetdienstes relevante Teil der Syntax und Semantik von HTML beschrieben.

Im Abschnitt 2.4 wird das Clustering charakterisiert und eine Klassifizierung der einzelnen Verfahren vorgenommen. Einen Teil dieser Clusterverfahren stellt die Data-Mining-Software WEKA in Form einer grafischen Oberfläche sowie einer Java-Bibliothek zur Verfügung. Nach der Beschreibung dieses Werkzeugs wird die Grundstruktur einer aktiven Ontologie erläutert, deren autonome Erzeugung übergeordnetes und weiterführendes Ziel dieser Arbeit ist.

2.1. Internetdienst

Zur Definition der Komposition aus Internet und Dienst müssen zunächst die beiden Teile getrennt betrachtet werden. Unter einem Dienst versteht man das Anbieten einer Leistung. Durch den Zusatz „Internet“ wird die Leistung auf Onlineangebote eingeschränkt. Jedoch entspricht diese Einschränkung noch nicht der Definition eines Internetdienstes in dieser Arbeit.

In dieser Arbeit wird unter einem Internetdienst das Angebot einer Dienstleistung auf einer Internetseite in Form eines HTML-Formulars verstanden. Diese Art des Internetdienstes lässt über verschiedenste Eingabefelder eine Interaktion zwischen Dienst und Nutzer zu, anhand derer verschiedenste Internetdienste entworfen werden können. Der detailliertere Aufbau eines solchen Formulars wird im Unterabschnitt 2.3.1 erläutert. Ein Internetdienst in diesem Sinne wäre beispielsweise eine Wetterauskunft, bei der über das Formular der Ort oder die Postleitzahl eingegeben wird und daraufhin der 3-Tages-Trend für diesen Ort dargestellt wird.

2.2. Uniform Resource Locator (URL)

Der *Uniform Resource Locator* ist eine Technologie, mit der über das Netzwerk eine Datei eindeutig aufgerufen werden kann. Sein Standard wird im RFC 1738 ([BLMM+94])

URL-Schemata	Protokoll
http	Hypertext Transport Protocol
ftp	File Transport Protocol
mailto	Single Mail Transfer Protocol(SMTP)

Tabelle 2.1.: URL-Schema: Ein Ausschnitt an URL-Schemata und dem dazugehörigen Protokoll, das am häufigsten dabei verwendet wird.

definiert. Die URL besteht aus zwei Grundblöcken, wobei die Gestalt des zweiten Teils abhängig von der Wahl des ersten Teils (des Schemas) variiert. Die beiden Teile sind stets durch einen Doppelpunkt getrennt.

[Schema] : [Schemata-spezifischer-Teil]

Das Schema spezifiziert welches Protokoll für den Netzwerkzugriff verwendet wird. Die Tabelle 2.1 skizziert einen Ausschnitt möglicher Netzwerkprotokolle.

2.2.1. Absolute HTTP-URL

Für den Aufruf von Internetseiten wird als URL-Schema das *Hypertext Transport Protocol* verwendet. Durch die Wahl des Schemas kann der erste Teil der URL durch die Abkürzung des Protokolls ersetzt werden:

http:[Schemata-spezifischer-Teil]

Zusätzlich kann von dieser Wahl auch noch der schemata-spezifische Teil abgeleitet werden, der sich wie folgt zusammensetzt:

http://[host]:[port]/[url-pfad]?[anfrage]

Die einzelnen spezifischen Teile sind bis auf den Hostbereich optional und werden bei Verwendung durch die verschiedenen reservierten Präfixsymbole unterschieden. Für die Dateiwahl über das Netzwerk sind die beiden Bereiche [host] und [url-pfad] entscheidend. Daher werden in der folgenden Betrachtung die beiden optionalen Teile [port] und [anfrage] weggelassen:

http://[host]/[url-pfad]

Der [host] gibt die Domäne einer Internetseite an und innerhalb der Domäne wird über den [url-pfad] die Quelldatei angesteuert.

http://www.example.de/index.html

2.2.2. Relative HTTP-URL

Die relative Pfadangabe wird zur Referenzierung innerhalb einer Seite verwendet, d.h. ausgehend von der absoluten Adresse, die bereits aufgerufen wurde, kann mit der relativen URL ein Speicherort innerhalb der Domäne spezifiziert werden. Daraus folgt, dass durch die aktuelle URL und die relative Referenzierung eben wieder eine absolute Adresse gewonnen werden kann. Bei der relativen Referenzierung gibt es zwei Varianten. Die Abbildung 2.1 präsentiert die beiden Referenzierungsarten, die sich durch das beginnende Literal / unterscheiden und somit entweder vom Wurzelverzeichnis oder vom aktuellen Verzeichnis aus referenziert werden.

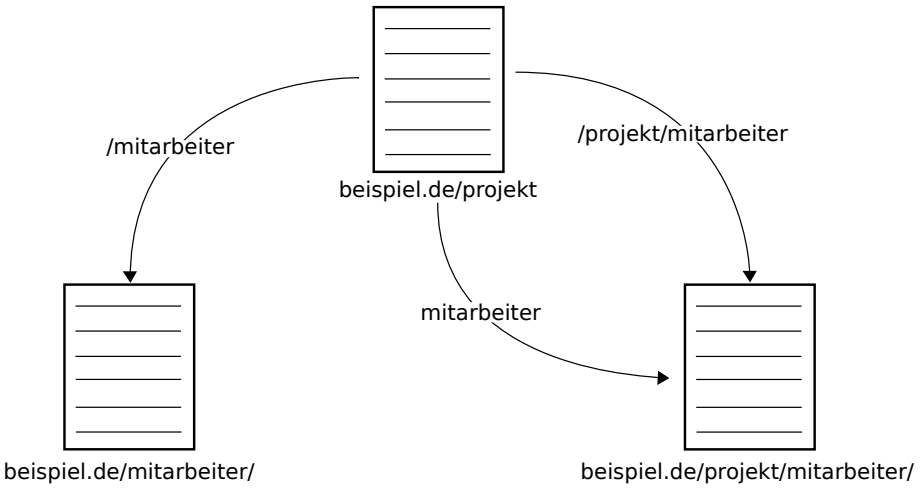


Abbildung 2.1.: Relative Referenzierung: Darstellung der Unterschiede zwischen den beiden relativen Referenzierungsarten.

2.3. Hypertext Markup Language (HTML)

Die Computersprache *Hypertext Markup Language (HTML)* ist der Standard für die Darstellung von digitalen Dokumenten und wird daher von allen gängigen Browsern unterstützt. HTML ist eine Auszeichnungssprache, die die Gliederung und Formatierung von Texten und anderen Daten beschreibt. Jeder Befehl (sog. *Tag*) wird von spitzen Klammern umhüllt. Zudem besteht jeder Befehl aus einem Start- und einem End-Tag, wobei das End-Tag als Präfix vor dem Tag einen Slash(/) enthält.

<tag> ... </tag>

Alles innerhalb des Befehlspaares wird gemäß des Befehls formatiert und so entsteht durch die gewählte Syntax eine Baumstruktur. Für Befehle, die ohne Inhalt sind (sog. „Standalone“-Tags), kann einer der folgenden Kurzschreibweisen ohne End-Tag gewählt werden:

<tag />

<tag>

Weiterhin können einem Befehl im Start-Tag noch verschiedene Attribute zugewiesen werden. Je nach Befehl unterscheiden sich die Attribute.

<tag attribute1=”...” attribute2=”...”> ... </tag>

2.3.1. Formular

Zur Erstellung von Formularen wird das Form-Element <*form*> verwendet. Innerhalb dieses Elements wird das Formular spezifiziert. Es bietet dem Entwickler die Möglichkeit, mit dem Nutzer zu interagieren, denn über verschiedenste Eingabefelder kann der Entwickler Informationen des Nutzers abfragen. Diese Informationen werden an eine Server geschickt und je Informationsangabe und Dienst, der vom Server angeboten wird, erhält der Nutzer eine Antwort. Die Angabe der Netzwerkadresse, an die das Formular gesendet werden soll, wird über das Attribut *action* spezifiziert. Das *action*-Attribut ist ein optionales Attribut und falls dieses nicht angegeben ist, wird als Netzwerkadresse die Adresse verwendet, auf der sich das Formular befindet.

<form action=”http://www.example.de/index.html”> ... </form>

Formularelement	Bedeutung	Seite
<input>	einzeiliges Eingabeelement	8
<textarea>	mehrzeiliges Eingabeelement	8
<select>	Auswahlelement	8
<button>	Schaltflächenelement	8
<label>	Bezeichnungselement	10

Tabelle 2.2.: Forumlarelemente: Eine Auswahl an Elementen, die innerhalb eines HTML-Formulars spezifiziert werden können.

In diesem Beispiel wurde die absolute Adresse im *action*-Attribut angegeben. Es kann aber auch bei passenden Voraussetzungen, wie im Unterabschnitt 2.2.2 erläutert, die relative URL zur Spezifizierung der Adresse verwendet werden.

2.3.2. Formularelemente

Für das Formular existieren spezielle Eingabeelemente, deren Werte beim Absenden des Formulars übermittelt werden. Voraussetzung dafür ist, dass sich die Elemente innerhalb des Form-Tags befinden. Die Tabelle 2.2 listet alle formularspezifischen Elemente auf.

Element: Input

Mit dem Input-Element <*input*> werden Benutzereingaben abgefragt. Je nach Zweck kann das Eingabeelement verschieden typisiert werden und durch die Spezialisierung kann der Browser das Element passender darstellen. Die Tabelle 2.3 listet alle möglichen Typen auf. Das Eingabeelement mit dem Typ *date* wird beispielsweise je nach gewünschtem Datumsformat mit Platzhaltern, die vom Nutzer ersetzt werden sollen, vordefiniert. Bei fehlender Typisierung wird das Texteingabefeld als Standardelement gewählt.

Element: Textarea

Das Textarea-Element ist eine Erweiterung des Texteingabefeldes auf eine mehrzeilige Texteingabe. Es können somit ganze Texte und nicht nur Schlüsselworte eingegeben werden. Eine beispielhafte Verwendung findet das Textarea-Element bei Formularen für das Eintragen von Forenbeiträgen.

Element: Select

Bei der Abfrage eines Wertes, bei dem die Auswahlmöglichkeiten bereits spezifiziert sind, wird das Select-Element <*select*> verwendet. Innerhalb des Select-Elements werden die einzelnen Auswahlmöglichkeiten über das Option-Element <*option*> angegeben. Die Abbildung 2.2 zeigt exemplarisch die Implementierung des Auswahlelements und die dazugehörige Darstellung im Browser. Die dargestellte Implementierung lässt ausschließlich die Wahl einer Option zu, jedoch kann das Select-Element durch die Angabe des Attributs *multiple* auf die Auswahl mehrerer Optionen erweitert werden.

Das

Element: Button

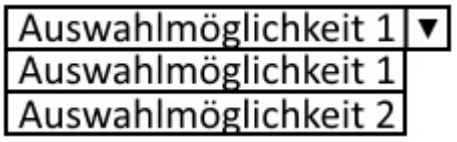
Das Button-Element <*button*> bietet dem Nutzer eine Schaltfläche, die je nach Typisierung eine Aktion auslöst. Es werden die drei Typen *button*, *submit* (Defaultwert) und *reset* unterschieden, die in dieser Reihenfolge betrachtet eine beliebige kundenseitige Aktion auslösen, das Formular absenden oder zurücksetzen.

Gruppierung	Typ	Beschreibung
Eingabe von Texten	text (Standardwert)	einzeiliges Eingabefeld
	email	E-Mail-Adressfeld
	password	Password-Eingabefeld
	search	Suchfeld
	url	Internetadressfeld
Buttons	reset	zum Zurücksetzen des Formulars
	submit	zum Senden des Formulars
	button	zum Aufruf einer clientseitigen Aktion
	image	Schaltfläche in Form eines Bildes
Eingabe von Zahlen	number	Nummer-Eingabefeld
	datetime	Datum- & Uhrzeit-Eingabefeld
	datetime-local	Datum- & Uhrzeit-Eingabefeld
	date	Datum-Eingabefeld
	month	Monat- & Jahres-Eingabefeld
	week	Woche- & Jahres-Eingabefeld
	time	Uhrzeit-Eingabefeld
	tel	Telefonnummer-Eingabefeld
Sonstige Elemente	checkbox	Checkbox
	radio	Radiobutton
	file	Hochladen einer Datei
	hidden	unsichtbarer Text
	color	Farbauswahl
	range	Intervallslider

Tabelle 2.3.: Input-Elemente: Alle Typen, die dem Input-Element zugewiesen werden können und deren Beschreibung.

```
<select>
 <option>
  Auswahlmöglichkeit 1
 </option>
 <option>
  Auswahlmöglichkeit 2
 </option>
</select>
```

(a) HTML-Code



(b) Browser-Darstellung

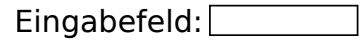
Abbildung 2.2.: Einfache Auswahl: HTML-Code und die dazugehörige Darstellung des Select-Elements im Browser.

```
<label for="refValue">
  Eingabefeld:
</label>
<input id="refValue">
```

(a) Referenzierung mit dem for-Attribut

```
<label >
  Eingabefeld:
  <input id="refValue">
</label>
```

(b) Referenzierung über das Kindelement



(c) Browser-Darstellung

Abbildung 2.3.: Label-Element: Ein beispielhafter HTML-Code für die Referenzierungsarten des Label-Elements auf ein Formularelement und die dazugehörige Darstellung des Label-Elements im Browser.

Element: Label

Das Label-Element <*label*> dient zur Beschreibung bzw. Beschriftung eines der anderen Formularelemente. Für die Zuordnung des Label-Elements zu einem der anderen Formularelemente gibt es zwei Methoden. Zur gegenseitigen Referenzierung muss bei der ersten Variante, der Wert des form-Attributs im Label-Element mit dem Wert des id-Attributs im dazugehörigen Formularelement übereinstimmen. Die zweite Möglichkeit, eine direkte Referenzierung zu gewährleisten, besteht darin, dass das Formularelement innerhalb des Label-Elements definiert wird, also zwischen Start- und End-Tag des Labels. Die Quelltextausschnitte 2.3a und 2.3b demonstrieren exemplarisch beide Referenzierungsarten. In vielen HTML-Dokumenten ist keine dieser beiden Referenzierungsmethoden umgesetzt, jedoch wird das aufeinanderfolgende Auftreten des Label- und Formularelements dort als Zusammengehörigkeit der Elemente betrachtet.

2.3.3. Navigationselemente

Die Navigationselemente übernehmen den elektronischen Verweis zwischen zwei Dokumenten (Hypertexten). Abstrakt betrachtet stellt ein Link eine unidirektionale Verbindung zwischen zwei Dokumenten auf und dadurch entsteht ein Linkgraph (siehe Unterabschnitt 2.3.6). Im optimalen Fall enthält ein Linkgraph, mit der Startseite als Ausgangspunkt, alle Dokumente der Domain. Wird ein Dokument von keinem anderen Dokument verlinkt, dann ist es ausschließlich über die direkte Adresseingabe erreichbar.

Element: Link

In HTML wird ein Link mit dem Element <*a*> erzeugt. Dieses Element ist ein Bild oder Text und wenn der Nutzer darauf klickt, wird das verlinkte Dokument geöffnet. Die Adresse des verlinkten Dokuments muss über das *href*-Attribut erfasst werden. Der Verweis kann ausgehend vom aktuellen Dokument absolut oder relativ angegeben werden, wie dies im Unterabschnitt 2.2.2 erläutert wurde. Exemplarisch dazu zeigt die Abbildung 2.4 einen Link mit dem Text „Kontakt“, bei dem einerseits über eine relative und andererseits über eine absolute Pfadangabe der Verweis auf das Dokument „Kontakt.html“ getätigt wurde.

2.3.4. Metadatenelemente

Das HTML-Dokument kann in zwei Bereiche aufgeteilt werden. Im zweiten Teil, dem „Körper “ <*body*>, wird der vom Browser abgebildete Inhalt spezifiziert. Im Gegensatz dazu werden im ersten Teil, dem „Kopf“ <*head*>, Angaben über das Dokument gemacht. Einige Elemente, die den Inhalt des Dokuments beschreiben, werden im folgenden beschrieben.

```
<a href="Kontakt.html">
 Kontakt
</a>
```

(a) relative Referenzierung

```
<a href="http://www.beispiel.de/
   Kontakt.html">
 Kontakt
</a>
```

(b) absolute Referenzierung

Abbildung 2.4.: Link-Element: Ein Link der ausgehend von der Seite http://www.beispiel.de/index.html über eine relative oder eine absolute Referenzierung auf das Dokuemnt http://www.beispiel.de/Kontakt.html verweist.

Element: / **Attribut:**	Input	Textarea	Select	Button	Link	Meta	Title
name	×	×	×	×	×	×	
title	×	×	×	×	×	×	
placeholder	×	×					
value	×			×			
content						×	
Zugehöriges Label-Element	×	×	×				
Inhalt des Elements		×		×	×		×

Tabelle 2.4.: Attribute: Darstellung einer Abbildungsmatrix, die für jedes Element illustriert, welches Attribut es enthalten kann. Bei dem Ausschnitt handelt es sich um Attribute die potenziell Informationen über die Semantik des Elements liefern.

Element: title

Das Title-Element *<title>* beschreibt den Inhalt eines Dokuments und muss in jedem HTML-Dokument stets im Kopfbereich vorkommen. Das Title-Element ist durch keine Attribute erweiterbar und wird dem Nutzer zur Beschreibung der Webseite im Browsertab angezeigt.

Element: meta

Das Meta-Element *<meta>* enthält Informationen über den Inhalt des Dokuments. Diese Angaben dienen der Verwaltung, damit können beispielsweise Suchmaschinen den Inhalt der Dokumente einem gewissen Themenbereich zuordnen. Das Meta-Element ist ein „Standalone“-Tag und daher werden Informationen über die Attribute übermittelt, wie beispielsweise über die Attribute *name* und *content*.

2.3.5. Attribute der Elemente

Alle zuvor beschriebenen Elemente können durch verschiedene Attribute erweitert werden. Für die folgenden Betrachtungen sind nur Attribute erheblich, die Auskunft darüber geben, welcher Inhalt abgefragt wird. Die Tabelle 2.4 listet alle potenziellen Attribute auf und markiert für jedes Element, mit welchem Attribut es erweitert werden kann. Die beiden Elemente *Label* und *Title* sind nicht dargestellt, da sie mit keinem aufgeführten Attribut erweitert werden können. Jedoch steckt bei diesen beiden Elementen die Information im Text, der zwischen Start- und End-Tag.

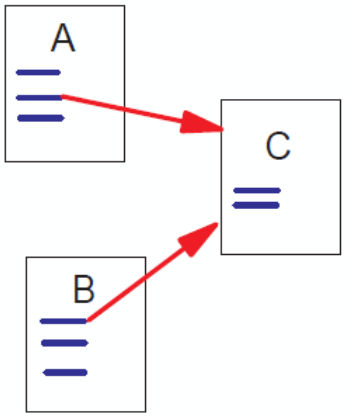


Abbildung 2.5.: Linkgraph: Das Beispiel zeigt einen Linkgraph bei der die Seiten A & B (Knoten) mit je einem Link (Kante) auf die Seite C verweisen. (Quelle:[Pag+99])

2.3.6. Linkgraph

Der Linkgraph wird beispielsweise im Verfahren *PageRank* [Pag+99] verwendet und kann eines von vielen Kriterien sein, nach dem eine Suchmaschine ihre Treffer einstuft. Der Linkgraph ist ein gerichteter Graph, dessen Knoten die einzelnen Seiten im Web sind. Falls eine Seite A auf eine Seite B verlinkt, dann wird im Linkgraph eine Kante vom Knoten A nach B erzeugt. Die Abbildung 2.5 zeigt exemplarisch eine einfache Variante eines Linkgraphen.

Für die Verwendung im Abschnitt 5.1 wird der Linkgraph ausgehend von einer Startseite aufgebaut. Auf der hierarchischen Ebene unterhalb des Wurzelknotens befinden sich alle Seiten (Knoten), die eine eingehende Kante vom Wurzelknoten haben. Eine Ebene darunter befinden sich dann wiederum alle Seiten, die eine eingehende Kante von einem der Knoten der nächst höheren Ebene haben und dies wird rekursiv fortgesetzt. Zusätzlich wird dieser Linkgraph noch dahingehend eingeschränkt, dass er nur Knoten derselben Domäne enthält und sollten Links zu anderen Domänen bestehen, wird für diese Domäne wieder ein neuer Linkgraph erzeugt. Die Abbildung 2.6 zeigt exemplarisch den zuvor definierten Linkgraph mit seinen Hierarchieebenen.

2.4. Clustering

Unter dem Begriff *Clustering* verbirgt sich die Analyse eines Datensatzes auf Ähnlichkeitsstrukturen der einzelnen Daten. Alle Daten, die derselben Äquivalenzklasse zugeordnet werden, bilden ein Cluster und das übergeordnete Verfahren der Datenzuordnung zu einem Cluster wird als Klassifizierung bezeichnet.

Für die Analyse der Datenstruktur kann das Clustering in partitionierende, hierarchische, graphentheoretische und probabilistische Verfahren untergliedert werden.

2.4.1. Partitionierendes Clusterverfahren

Beim partitionierenden Clustering wird die Datenzuordnung mit einer gegebenen festen Anzahl an Clustern vorgenommen. Über das Aufspannen eines N-dimensionalen Merkmalsraums werden anhand der räumlichen Nähe die Objekte zu einem Cluster zusammengefasst. Dabei können einzelne Datenpunkte zur Optimierung der Zuordnung noch einem anderen Cluster zugeteilt werden.

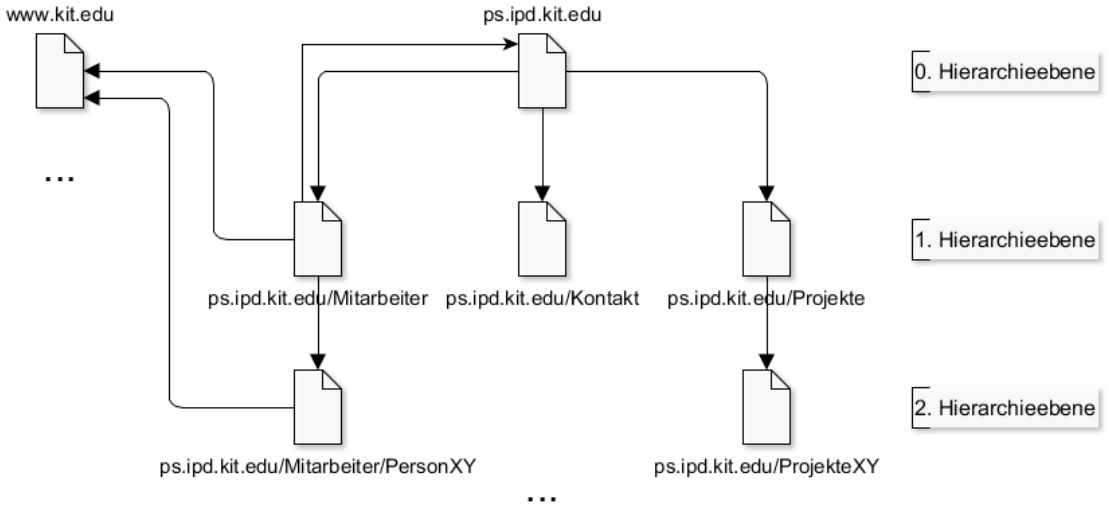


Abbildung 2.6.: Beispielhafter Ausschnitt des Linkgraphs der Subdomain *ps.ipd.kit.edu*. Zusätzlich wird die hierarchische Einteilung der Seiten (Knoten) dargestellt.

2.4.2. Hierarchisches Clusterverfahren

Die hierarchischen Clusteringverfahren ordnen zuerst die gesamte Menge einem Cluster zu und teilen dieses Cluster bis zu einer vorgegebenen Anzahl rekursiv weiter auf. Im Gegensatz zum partitionierenden Clustering ist bei diesem Verfahren die Zuordnung fest und Datenpunkte können nicht verschoben werden. Zusätzlich gibt es noch eine zweite umgekehrte Variante, die ausgehend von einem Cluster pro Datenpunkt, die Cluster bis zu einer vorgegeben Anzahl vereint.

2.4.3. Graphentheoretische Clusterverfahren

Graphentheoretische Clusterverfahren abstrahieren den Datensatz als einen Graphen. Die einzelnen Datenpunkte werden dabei als Knoten abgebildet und die Kanten liefern über ihre Gewichtung Aufschluss über die Ähnlichlichkeit beider Datenpunkte. Anhand der Gewichtung lässt sich die Zusammenhangskomponente berechnen, die eine Möglichkeit bildet, anhand des Graphen die einzelnen Cluster abzuleiten. Vorteile dieser Verfahren sind die Verwendung von graphentheoretischen Algorithmen sowie die Visualisierung, da aus dem n-dimensionalen Merkmalsraum ein Graph wird, der zweidimensional abgebildet werden kann.

2.4.4. Probabilistische Clusterverfahren

Im Gegensatz zu den vorherigen drei Clusterverfahen wird in diesem Fall keine deterministische, sondern eine probabilistische Clusterzuordnung vorgenommen. Für jeden Datenpunkt wird somit die Aussage getroffen mit welcher Wahrscheinlichkeit er zu welchem Cluster gehört. Die Wahrscheinlichkeit für die einzelnen Clusterzugehörigkeiten kann beispielsweise aus verschiedensten Abstandsfunktionen berechnet werden. Die Wahrscheinlichkeiten für einen Datenpunkt liefern ein Maß für die Eindeutigkeit einer Zuordnung.

2.5. WEKA

WEKA ist ein Softwarewerkzeug, das Techniken aus dem Bereich Data-Mining bereitstellt. Mit der grafischen Oberfläche lassen sich selbst in großen Datensätzen Ähnlichkeiten bzw. Unterschiede analysieren. Es liefert einen umfangreichen Satz an Klassifikations-,

Cluster-, Filter- und Evaluationsalgorithmen, deren Auswirkung auf den Datensatz mithilfe der grafischen Oberfläche komfortabel analysiert werden kann. Zusätzlich stellt WEKA die Techniken und Algorithmen noch als Java Bibliothek zur Verfügung. Die Einbindung der WEKA-Bibliothek ermöglicht es, komfortabel und einfach Data-Mining-Software zu schreiben. Als Schnittstelle für die Eingabe der Datensätze agiert das Attribut-Beziehung-Dateiformat (*attribute-relation file format* - arff), dessen Struktur im folgenden detailliert betrachtet wird.

Dateiformat: ARFF

Das ARFF-Dateiformat kann in zwei getrennte Bereiche unterteilt werden. Zum einen der Kopf der Datei, in dem der Datensatz spezifiziert ist und zum anderen der Körper, in dem die Daten selbst stehen. Zur Veranschaulichung ist im Quelltextausschnitt 2.1 ein Datensatz im ARFF-Format skizziert.

Im Kopf steht zu Beginn hinter dem Schlüsselwort *@RELATION* die Bezeichnung des Datensatzes. Danach folgt eine beliebige Anzahl an Attributen, die sich im einzelnen aus dem Schlüsselwort *@ATTRIBUTE*, dem Namen und dem Typ des Attributes zusammensetzen. Diese beiden Angaben genügen um den Datensatz eindeutig zu spezifizieren. Als Typ kann das Attribut die Form *numeric*, *nominal*, *string* und *date* annehmen.

Der zweite Teil des Dateiformats wird mit dem Schlüsselwort *@DATA* eingeläutet. Darauf folgend wird pro Zeile ein Datum erfasst, wobei die einzelnen Attribute des Datums durch ein Komma getrennt werden. Im Falle eines String-Attributs wird der String in Anführungszeichen gesetzt.

```
@RELATION ortsabhängige Temperaturmessung

@ATTRIBUTE Temperatur numeric
@ATTRIBUTE Ort  string
@ATTRIBUTE Datum date [yyyy-MM-dd]

@DATA
20.5, "Housten", 2015-06-15
21.3, "New York", 2015-06-16
...
```

Quelltextausschnitt 2.1: ARFF-Datei: Ein Ausschnitt eines Datensatzes im ARFF-Dateiformat.

2.6. Aktive Ontologien

Der Begriff der Ontologie bezeichnet in der Informatik eine Datenstruktur, die für eine Menge an Begrifflichkeiten eine formal geordnete Darstellungsmöglichkeit bietet. Mit einer Ontologie lässt sich beispielsweise die Aussage, dass ein Architekt einen Plan zeichnet, formal modellieren. Die aktiven Ontologien erweitern die konventionellen Ontologien um eine dynamische Komponente und so wird aus einer Datenstruktur (Ontologie) eine Ausführungsumgebung (aktive Ontologie). Die Abbildung 2.7 skizziert die einzelnen Elemente einer aktiven Ontologie, die im folgenden kurz beschrieben werden:

2.6.1. Konzept

Für die Beschreibung der Konzepte und Beziehungen soll die Abbildung 2.8, die eine exemplarische Modellierung einer aktiven Ontologie darstellt, als graphischer Leitfaden dienen. Die Grunddatenstruktur einer aktiven Ontologie ähnelt der eines Graphen, wobei die Knoten als Konzepte bezeichnet werden. Jedes Konzept hat einen Namen, einen Satz

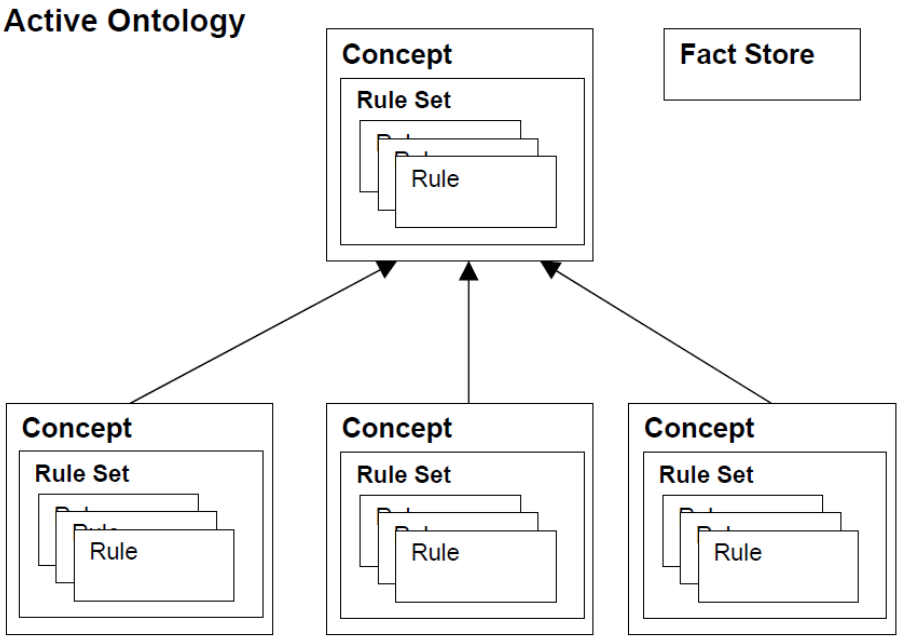


Abbildung 2.7.: Aktive Ontologie: Konzeptionelle Darstellung der einzelnen Komponenten einer aktiven Ontologie. ([Guz08])

an Regeln und einen Wert für die Wahrscheinlichkeit (Ranking) des Konzepts. Das Ranking gibt Auskunft mit welcher Wahrscheinlichkeit eine Aussage zu diesem Konzept vom Nutzer getätigt wurde. Für die Berechnung dieses Rankings müssen die Konzepte differenzierter betrachtet werden. Gemäß der Graphenanalogie entsprechen die Blattknoten den Sensorknoten und die inneren Knoten den Konzeptknoten.

Die *Sensorknoten* agieren als eine Art Filter und je nach Konzept werden die Wörter der Aussage des Nutzers sequentiell analysiert. Für die Filterung gibt es die folgenden fünf verschiedenen Sensorknoten:

Vokabularliste: Dieser Sensorknoten filtert aus einer Aussage des Nutzers alle Worte, die in der Vokabularliste gespeichert sind. Die Vokabularliste kann manuell erzeugt werden oder extern angebunden werden.

Prefix: Für ein bestimmtes Präfix werden alle darauf folgenden n Worte gefiltert.

Postfix: Für ein bestimmtes Postfix werden alle zuvor genannten n Worte gefiltert.

Regulärer Ausdruck: Alle Worte, die einem bestimmten regulären Ausdruck genügen werden von diesem Sensorknoten erkannt.

Spezialisierung: Dieser Sensorknoten erkennt fest implementierte Logik. So kann beispielsweise der Begriff *morgen* erkannt und in ein Datum transformiert werden.

Der Filter kann beispielsweise über den Abgleich von Vokabularlisten oder über externe Dienste realisiert werden. In der vorherigen Abbildung 2.8 werden fünf Sensorknoten dargestellt. Für den Stadt-Sensorknoten kann beispielsweise eine externe Vokabularliste mit allen Städten verwendet werden. Auf Basis der Filterung wird dann ein Ranking berechnet, das die Wahrscheinlichkeit angibt, mit der eine Stadt in der Aussage genannt wurde.

Bei den Konzeptknoten gibt es zwei Typen, den Selektionsknoten und den Kombinationsknoten, die jeweils über die Beziehung zu ihren Kindkonzepten unterschieden werden. Der Selektionsknoten wählt aus seinen Kindern das Konzept mit dem höchsten Ranking aus. In der Abbildung spiegelt sich dieser Typ im Gebäude-Konzept wider, da sowohl

ein Theater als auch ein Restaurant ein Gebäude sind. Im Gegensatz dazu berechnet der Kombinationsknoten sein Ranking anhand aller Rankingwerte seiner Kindkonzepte. Es wird also eine Beziehung zu den Kindkonzepten aufgebaut, die beschreibt was ein Elternkonzept alles besitzt. Passend dazu besteht beispielsweise das Adresse-Konzept aus einer Postleitzahl, einer Stadt und einem Bundesland.

2.6.2. Beziehungen

Eine Beziehung verbindet stets ein Quellkonzept mit einem Zielkonzept. Jede Beziehung hat einen Typ und einen Satz an Attributen. Je nach Beziehung der Elternkonzepte zu ihren Kindkonzepten können die verschiedenen Konzepttypen unterschieden werden.

Wie bereits erwähnt sind die Sensorknoten die Konzepte, die keine weiteren Kindskonzepte mehr enthalten. Die anderen beiden Konzepte werden durch unterschiedliche Beziehungstypen unterschieden. Zum einen gibt es die strukturelle Beziehung, der Eigenschaften eines Konzepts, wie beispielsweise eine Kardinalität oder die Optionalität, zugewiesen werden können. Diese Beziehungen führen stets zu Kombinationsknoten und bauen eine „hatte eine“ Beziehung auf. Zum anderen gibt es die klassifizierende Beziehung, die Konzepte untergliedert und als „ist eine“ Beziehung betrachtet werden kann. Dieser Beziehung können keine Eigenschaften zugewiesen werden. Zudem bilden sie stets auf Selektionsknoten ab.

2.6.3. Regeln

Aktive Ontologien enthalten zusätzlich zu Konzepten und Beziehungen noch Verarbeitungselemente, die über Regeln modelliert werden. Jedes Konzept hat einen Satz an Regeln gemäß der Abbildung 2.7. Jede Regel besteht aus einer Bedingung und einer Aktion. Bei jeder Zustandsänderung werden alle Regeln anhand ihrer Bedingung und des aktuellen Zustandes evaluiert und bei positiver Auswertung die dazugehörige Aktion ausgeführt, wobei diese wieder eine Zustandsänderung hervorrufen kann.

2.6.4. Fakten

Die Fakten bilden den aktuellen Zustand der aktiven Ontologie und werden in einem gemeinsamen Faktenspeicher gehalten. Ein Fakt könnte beispielsweise sein, dass der Nutzer in seiner Anfrage eine Telefonnummer genannt hat. Außerdem können über Benutzeräußerungen und damit verbunden dem Auslösen der Aktionen innerhalb einer Regel, Fakten geändert, gelöscht oder neue Fakten hinzugefügt werden.

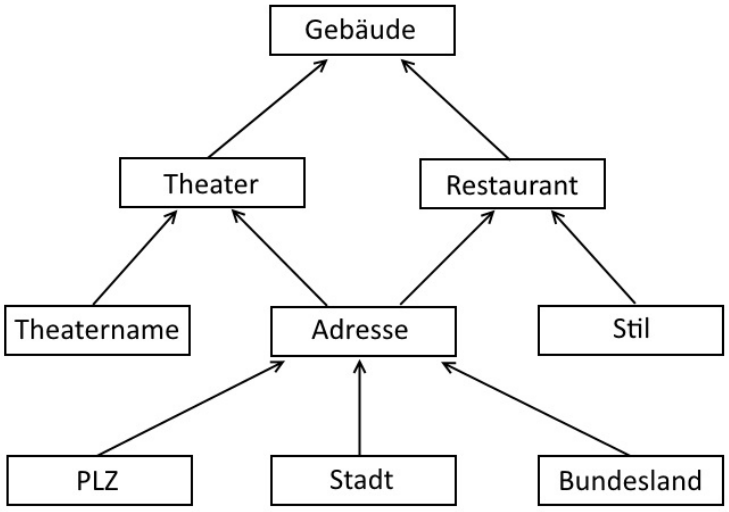


Abbildung 2.8.: Aktive Ontologie: Beispielhafte Modellierung einer aktiven Ontologie mit dem Einsatz von Konzepten und Beziehungen.

3. Verwandte Arbeiten

In Anlehnung an das Ziel dieser Arbeit kann eine Aufgabenteilung in die Bereiche:

- Webcrawling, zum Finden von Internetdiensten
- Clustering, zur Klassifizierung der Internetdienste
- aktive Ontologie, zum Ableiten einer aktiven Ontologie

vorgenommen werden. Für die einzelnen Bereiche wurden bereits andere Arbeiten mit ähnlichen Zielsetzungen oder Herangehensweisen durchgeführt. Im folgenden werden für die einzelnen Bereiche einige verwandte Arbeiten vorgestellt.

3.1. Webcrawler

Im Bereich des Webcrawling werden Programme geschrieben, die Informationen aus dem Internet extrahieren. Diese Programme wurden nahezu zeitgleich mit der Einführung des Internets entworfen und somit wurde von Beginn an das Informationspotenzial des Internets erkannt. Eine der ersten Arbeiten [Pin94] im Bereich des Webcrawlings ist von B. Pinkerton und wurde auf der zweiten *World Wide Web-Konferenz* vorgestellt. Er entwarf einen Webcrawler, dessen Ziel es war, den Inhalt des Netzes zu indexieren und so über Suchmaschinen dem Nutzer zugänglich zu machen. Für die Umsetzung dieses Zieles mussten alle bereits aufgerufenen Seiten markiert bzw. gespeichert, alle ausgehenden Links gesammelt und wie bereits erwähnt der Inhalt des Dokuments indexiert werden. Anhand der ersten Aufgabe wird ersichtlich, dass von Beginn an das Problem der zyklischen Verlinkung bestand. Jedoch war dieser Webcrawler aufgrund des damals kleinen Internets noch relativ einfach. Dennoch wuchs das Netz im Laufe der Jahre annähernd exponentiell und so wurde aus einer aktiven Webseite im Jahre 1991 fast eine Milliarde aktiver Webseiten im Jahre 2014. Die Abbildung 3.1 zeigt die Gesamtzahl an Internetseiten in den letzten 15 Jahren. Dieser exponentielle Anstieg an aktiven Webseiten ist analog auch auf den Informationsgehalt im Netz zu übertragen und dient als Motivation zum Entwurf eines Webcrawlers.

3.1.1. Zentraler, skalierbarer Webcrawler (*Mercator*)

Im Paper [HN99] von A. Heydon und M. Najork mit dem Titel *„Mercator: A scalable, extensible web crawler“* wird der Entwurf eines skalierbaren und erweiterbaren Webcrawlers (sog. *Mercator*) beschrieben. Die Skalierbarkeit gibt Auskunft darüber, dass trotz einer

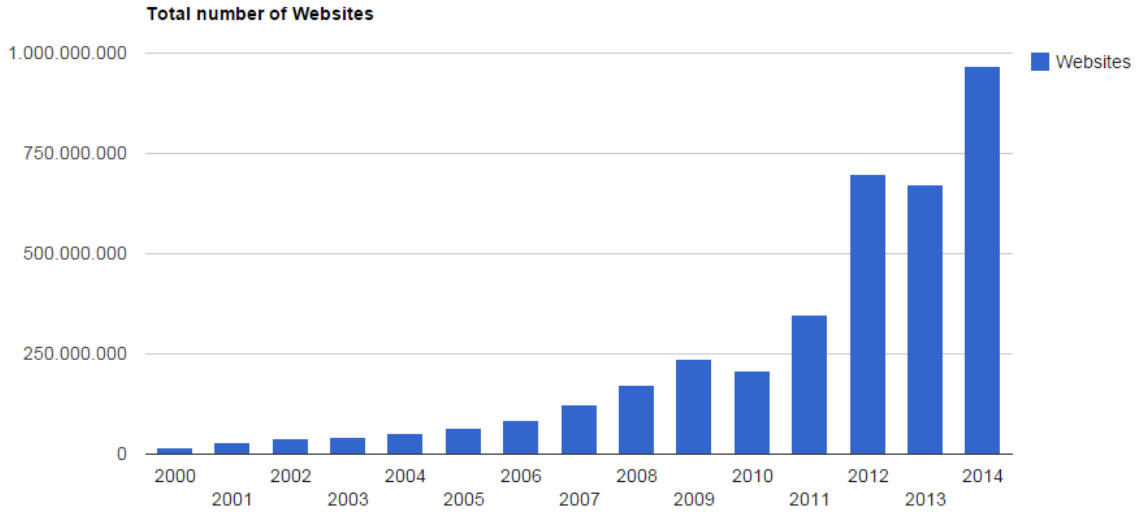


Abbildung 3.1.: Gesamtzahl der aktiven Webseiten in den letzten 15 Jahren

großen Anzahl an untersuchten Webseiten, der Speicherbedarf im Rahmen bleibt. Aufgrund der immensen Webseitenanzahl, die zuvor beschrieben wurde, ist die Charakteristik der Skalierbarkeit ein wichtiges Kriterium. *Mercator* basiert auf einer mehrfädigen Prozessarchitektur, bei der ausgehend von den vorgegebenen Initialwebseiten, eine Breitensuche ausgeführt wird. Dazu erhält jeder Prozessfaden seine eigene FIFO-Warteschlange, die die URLs der nächsten Dokumente enthält. Beim Befüllen der Warteschlangen wird jedoch darauf geachtet, dass selten zwei Prozessfäden beim selben Server ein Dokument anfordern, um den Server nicht zu überlasten und somit die Antwortzeit des Servers möglichst gering zu halten.

Im nächsten Schritt wird das Dokument geladen und es wird getestet, ob der Inhalt schon einmal geladen wurde (Dokument-gesehen-Test). Falls dies nicht der Fall ist, werden Informationen sowie Links extrahiert. Dabei werden die URLs der Links wiederum darauf getestet, ob diese URL bereits extrahiert wurde(URL-gesehen-Test).

Um der Anforderung der Skalierbarkeit gerecht zu werden, muss der Speicherbedarf minimiert werden. Dabei belegen die zuvor vorgestellten Tests, der Dokument-gesehen-Test und der URL-gesehen-Test, mit steigender Anzahl an durchsuchten Dokumenten den größten Speicherplatz. Für beide Tests wird jeweils nicht das gesamte Dokument oder die absolute URL gespeichert, sondern ein Fingerabdruck in Form einer 64-Bit langen Prüfsumme. Anhand des Vergleichs der Prüfsumme können die beiden Tests ausgeführt werden. Zusätzlich werden im *Mercator* noch die populärsten und zuletzt untersuchten Dokumente und auch URLs in einem Cache gespeichert, der durch seine hohe Trefferrate eine Geschwindigkeitssteigerung zur Folge hat. *Mercator* verknüpft somit die Anforderungen, einen Dokument- und url-seen-Test mit der Eigenschaft der Skalierbarkeit durchzuführen.

3.1.2. Verteilter Webcrawler

In der Arbeit [SS02] von V. Shkapenyuk und T. Suel wird ein verteilter Webcrawler entworfen, d.h. die einzelnen Komponenten des Webcrawlers können sich über mehrere Rechner verteilen und kommunizieren über Sockets oder Dateisysteme.

Ein Ziel der Autoren war es einen Höchstleistungswebcrawler zu entwerfen, der ca. 100 Webseiten pro Sekunde auswertet. Diese Anforderung hat zur Folge, dass der Webcrawler sich robust gegenüber Serverausfällen und „schlecht“ entworfenen HTML-Seiten verhält. Außerdem sollten die Kosten bei Höchstleistung möglichst gering sein und er muss für

verschiedenste Gebrauchsszenarien erweiterbar sein, damit er häufig verwendet werden kann.

Für die verteilte Architektur ist der Webcrawler in eine Crawling-Anwendung und in ein Crawlingsystem geteilt. Das Crawlingsystem erhält als Eingabe eine URL und liefert die heruntergeladene Datei der URL zurück. Zudem testet diese Komponente, ob die URL bereits schon einmal angefordert wurde. Falls die URL noch nicht besucht wurde, wird sie aufgelöst und anschließend die Datei heruntergeladen. Die Crawling-Anwendung übernimmt die Aufgaben, die auf dem Inhalt des Dokumentes basieren. In dieser Komponente werden folglich die benötigten Informationen herausgefiltert sowie die Links extrahiert. Das gesamte System ist, bis auf das Downloader-Modul (Python), in C++ geschrieben.

Durch die verteilte Architektur kann jede Teilaufgabe auf einem anderen System berechnet werden und somit sind die vorhandenen Recheneinheiten in ihrer Anzahl unbegrenzt. Dem entgegen steht jedoch der erhöhte Kommunikationsbedarf und die steigende Komplexität.

3.1.3. Diskussion

Die beiden vorgestellten Arbeiten haben mit sehr unterschiedlichen Ansätzen einen Webcrawler entworfen. Im folgenden werden die Charakteristiken der vorgestellten Webcrawler verglichen und es wird auf die Verwendbarkeit in dieser Arbeit eingegangen.

Komplexität

Die beiden vorgestellten Webcrawler sind entworfen worden, um große Mengen des Internets zu durchsuchen und somit große Datensätze zu generieren. Aufgrund dieser Anforderung sind die einzelnen Systeme sehr komplex, damit eine möglichst allgemeingültige Lösung für die beim Webcrawling entstehenden Probleme geboten werden kann. Der verteilte Webcrawler ist durch die zusätzliche Kommunikationsstruktur zum Austausch der Daten noch etwas komplexer. Für den Entwurf eines Webcrawlers, dessen Anforderung nicht der Vollständigkeit genügen muss, können einige Abstraktionen vorgenommen werden, die die Komplexität des Systems vereinfachen.

URL-gesehen-Test

Mercator implementiert den URL-gesehen-Test durch den Abgleich der Fingerabdrücke der URL und versucht diesen über Caches zu beschleunigen. Bei der verteilten Variante werden die URLs in einem Rot-Schwarz-Baum im Arbeitsspeicher gehalten und dabei stets im ganzen Block ab einer gewissen Größe in den Speicher geladen. Beide Webcrawler haben gemeinsam, dass sie unabhängig davon wie die URLs gespeichert werden, stets alle URLs speichern. Diese allgemeingültige Lösung kann im Falle der Zielsetzung dieser Arbeit durch Einschränkungen mit weniger Speicherbedarf gelöst werden. Dazu wird der URL-gesehen-Test um einen Host-gesehen-Test erweitert. Sobald alle URLs eines Hosts durchsucht wurden, können alle URLs des Hosts gelöscht werden und es muss nur noch der Host gespeichert und getestet werden. Damit ein Host als vollständig durchsucht angenommen werden kann, gilt folgende Einschränkung. Für einen Host werden nur die URLs betrachtet, die von der Startseite mit maximal 5 Links erreichbar sind.

Dokument-gesehen-Test

Ein Problem, das beim Entwurf eines Webcrawlers betrachtet werden muss, sind URL-Alias-Effekte, d.h. dass verschiedene URLs auf den selben Inhalt verweisen. Auf dieses Problem wird nur in der *Mercator*-Arbeit eingegangen und durch den Dokument-gesehen-Test mit einem Prüfsummenvergleich gelöst. Aufgrund der Komplexität und des Speicherbedarfs wird beim Entwurf eines Webcrawlers für diese Arbeit auf diesen Test verzichtet,

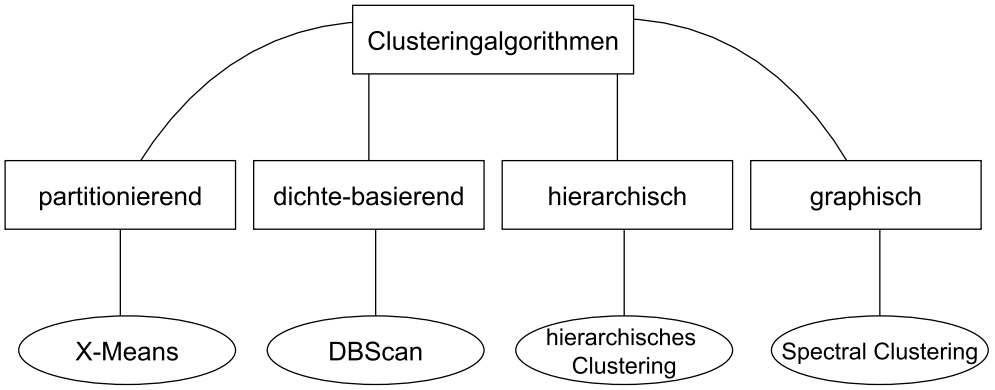


Abbildung 3.2.: Clusteringalgorithmen: Eine Übersicht und Klassifizierung der betrachteten Clusteringalgorithmen.

jedoch werden die gefundenen Internetdienste anhand einiger Merkmale innerhalb eines Hosts auf Duplikate untersucht. Somit wird die Duplikatserkennung vom Inhalt der Webseite auf den Inhalt des gefundenen Internetdienstes verlagert.

Fazit

Die vorgestellten Arbeiten bieten allgemeingültige Lösungen zur Extraktion für beliebige Informationen an. Die präsentierten Lösungswege können durch die konkrete Informationsextraktion von Internetdiensten und gewissen konkreten Einschränkungen in vielen Komponenten vereinfacht werden.

3.2. Clustering

Das Clustering wird in vielen Disziplinen der Informatik verwendet, da es eine generalisierte Betrachtung auf einen Datensatz ermöglicht. Nachfolgend werden einige verwandte Arbeiten beschrieben und in Bezug zu dieser Ausarbeitung gesetzt. Die Abbildung 3.2 zeigt die verwendeten Clusteringalgorithmen und klassifiziert sie gemäß des Kapitels 2.4.

3.2.1. Partitionierendes Clustering

Partitionierende Clusteringalgorithmen zeichnen sich durch ihre Partitionierung anhand einer vorgegebenen Clusteranzahl aus und je nach Forschungsfrage ist die Clusteranzahl bekannt. Dennoch wird in der Arbeit [Lia+09] mit dem Titel *„Clustering Web Services for Automatic Categorization“* ein Anwendungsfall gezeigt, in dem trotz unbekannter Clusteranzahl, ein partitionierendes Clusteringverfahren verwendet wird. Denn in dieser Arbeit wird zum einen ein schnelles Clusteringverfahren zur groben Vorgruppierung, und zum anderen danach ein langsameres Clusteringverfahren zur Verfeinerung der ersten Gruppierung angewandt. Für die grobe Gruppierung wird der partitionierende Clusteringalgorithmus k-Means verwendet, der die Daten in k-Cluster einteilt. Als Daten dienen die in der Beschreibung enthaltenen Eingabe- und Ausgabeparameter, sowie deren latente Zwischenbeziehungen des Webdienstes. Diese Daten sollen ähnlich funktionale Webdienste gruppieren.

Durch das zweigeteilte Clustering kann schnell eine grobes Ergebnis berechnet und damit verbunden ein grober Überblick geschaffen werden. Allerdings ist für die Berechnung einer „guten“ Lösung der zweite Clusteringalgorithmus verantwortlich. Die Aufteilung ist somit nur bei zeitkritischen Programmen sinnvoll, da sie in den anderen Fällen nur die

Komplexität des Clusterings erhöht. Schlussfolgernd wird in dieser Arbeit kein partitionierender Clusteringalgorithmus verwendet, da der zu entwickelnde Klassifikator einerseits keine zeitkritische Anforderung besitzt und andererseits die Clusteranzahl aufgrund der Forschungsfrage variabel ist.

3.2.2. Dichte-basiertes Clustering

In der Arbeit [RD12] mit dem Titel *„Web Services Discovery based on Semantic Similarity Clustering“* von P. Ravinder Reddy und A. Damodaram sollen IT-Dienste gruppiert werden. Als IT-Dienste werden alle Dienste, die mit dem dienstorientierten Architekturmuster (SOA) implementiert sind, spezifiziert. Jeder dieser Dienste verfügt über eine Dienstbeschreibung und ist in einem zentralen Verzeichnis registriert. Bisher wurde zur Auswahl einer dieser registrierten Dienste eine Anfrage anhand einer Schlüsselwortübereinstimmung ausgewertet. Die Grundidee der Autoren ist das Ranking für die Auswahl nicht anhand der Syntax zu bestimmen, sondern anhand der Semantik der Anfrage. Dies bedeutet, dass die Dienste anhand der semantischen Bedeutung ihrer Beschreibung gruppiert und ausgewertet werden sollen.

Für die Gruppierung wird ein dichte-basierender Clustering-Algorithmus(DBSCAN) verwendet, der anhand einer Ähnlichkeits- bzw. Distanzfunktion Daten gruppiert. Als Eingabe erhält dieser Algorithmus zwei Parameter. Der erste Parameter *epsilon* spiegelt den Radius wider. Dazu wird von einem Datenpunkt aus ein Kreis mit dem Radius *epsilon* aufgespannt und innerhalb dessen wird nach benachbarten Datenpunkten gesucht. Der zweite Parameter $minPoints$ liefert die Mindestanzahl an Datenpunkten, die innerhalb dieses Radius liegen müssen, damit die Datenpunkte im selben Cluster liegen. Mit diesen beiden Kenngrößen lässt sich die zu erkennende Dichte des Datensatzes regulieren.

Bevor jedoch der Clusteringalgorithmus auf den Datensatz angewandt werden kann, müssen die Daten erst in einem Merkmalsraum abgebildet werden. In dieser Arbeit wird eine Ähnlichkeitsfunktion verwendet, die die semantische Ähnlichkeit zweier Dienste misst. Die semantische Ähnlichkeit wird über den Abgleich der semantischen Bedeutung einzelner in der Beschreibung auftauchender Begriffe berechnet. Für den Abgleich wird eine externe Datenbank aufgerufen, die anhand von Synonymlisten die Anfrage bearbeitet. Wie bereits erwähnt, muss für die Nutzung der Datenbank die Beschreibung auf einzelne Worte reduziert werden. Dazu wird die Beschreibung zuerst in Wörter geteilt, in einem zweiten Schritt werden die unwichtigen Wörter entfernt, dann die Endungen angepasst (Bsp. Plural wird zu Singular) und zuletzt die zu generellen Begriffe entfernt.

Beim Vergleich der vorgestellten Arbeit mit der Zielsetzung dieser Ausarbeitung lassen sich einige Parallelen finden. Die Aufgabenstellung, Informationen aus einem Netz zu extrahieren und diese nach einem Kriterium zu klassifizieren, sind identisch. Somit herrschen durch die gleiche Art der Aufgabenstellung für das Clustering dieselben Voraussetzungen. In der vorgestellten Arbeit wurde ein dichte-basierter Clusteralgorithmus verwendet, der mit einer unbekannten Clusteranzahl umgehen kann. Ein weiterer Vorteil ist, dass die Cluster keine spezielle Form haben müssen, sondern nur anhand der Dichte die Cluster gebildet werden. Dies kann jedoch auch ein Nachteil sein, wenn die einzelnen Cluster verschiedene Dichten aufweisen, denn dann können mit diesem Algorithmus die Cluster nicht erkannt werden.

3.2.3. Spektrales Clustering

Das spektrale Clustering basiert auf einer Ähnlichkeitsmatrix, die von einem Graphen abgeleitet wird. Daraufhin wird die Ähnlichkeitsmatrix in eine Laplacian-Matrix transformiert und anhand dieser die k-kleinsten Eigenvektoren berechnet. Der Parameter k

bezeichnet die Anzahl der Cluster und wird dem Algorithmus als Eingabe zur Verfügung gestellt. Durch die Einschränkung auf die k-kleinsten Eigenvektoren wird die Merkmalsmatrix in Form der Laplacian-Matrix in der Anzahl ihrer Dimensionen reduziert. Im letzten Schritt werden mit dem k-Means Clusteringalgorithmus auf dem reduzierten Merkmalsraum die Cluster gebildet. Diese Art des Clusterings wurde auch in der Arbeit [Zha+09] mit dem Titel *„Web Service Community Discovery Based on Spectrum Clustering“* angewandt. Die Grundidee der Autoren ist es, ähnliche Webdienste, die in Form des dienstorientierten Architekturmusters (SOA) implementiert sind, anhand der Interaktionslogdaten zu gruppieren. Dazu werden die Logdaten ausgelesen und ein Graph erstellt, der als Knoten die Operationen der Webdienste und als gerichtete Kanten den Fernzugriff einer Operation auf die andere enthält. Aus dem Graphen wird daraufhin eine Ähnlichkeitsmatrix abgeleitet und danach wie zuvor beschrieben der spektrale Clusteringalgorithmus angewandt.

Der vorgestellte Algorithmus bietet den Vorteil, dass die Ähnlichkeitsmatrix auf Basis des Graphen abgeleitet wird. Somit entstehen keine Ähnlichkeiten zwischen zwei Knoten, die nicht miteinander verbunden sind. Zusätzlich wird der Merkmalsraum, dessen Dimension der Anzahl der Knoten entspricht, auf k-Dimensionen reduziert. Die Dimensionenreduktion entspricht der Hauptkomponentenanalyse. Allerdings hat die Hauptkomponentenanalyse den Nachteil, dass die neuen reduzierten Merkmale aus einer Linearkombination der alten Merkmale gebildet wird und somit die Intuition und Anschaulichkeit der Merkmale verloren geht.

Aufgrund der geforderten Clusteranzahl-Eingabe ist die direkte Umsetzung dieses Algorithmus aufgrund der Aufgabenstellung dieser Arbeit nicht möglich. Jedoch existieren abgewandelte spektrale Clusteringalgorithmen, die anhand der Werte der Ähnlichkeitsmatrix, ohne Eingabe einer festen Clusteranzahl, Daten gruppieren und somit der Anforderung der Forschungsfrage genügen.

3.2.4. Hierarchisches Clustering

Bei dieser Arbeit [JBL08] mit dem Titel *„Hierarchical Business Process Clustering“* werden Geschäftsprozesse gruppiert. Ein Geschäftsprozess besteht aus einer Menge an Aktivitäten und gerichteten Übergängen, die die Aktivitäten in Beziehung setzen. Um die Geschäftsprozesse vergleichen zu können, werden die Aktivitäten und die Übergänge jeweils auf einen Vektor abgebildet. Jeder dieser Vektoren wird über eine Ähnlichkeitsfunktion mit den passenden Vektoren eines anderen Geschäftsprozesses verglichen. Die resultierenden Ähnlichkeitsmaße werden daraufhin mit passender Gewichtung aufsummiert. Mit diesem Verfahren können alle Geschäftsprozesse miteinander verglichen und somit gruppiert werden. Für die Gruppierung wird ein hierarchischer Clusteralgorithmus verwendet, der als Grundmenge jeden Geschäftsprozess in einen eigenem Cluster abbildet. In einem zweiten Schritt wird iterativ die Clusteranzahl auf k-Cluster verringert, indem ähnliche Cluster vereinigt werden.

In der vorgestellten Arbeit sollen die Geschäftsprozesse auf eine bestimmte Anzahl an Clustern gruppiert werden. Die leicht veränderte Aufgabenstellung ändert die Voraussetzung für das Clustering. Bei einer festen Clusteranzahl sind die dichte-basierenden Algorithmen suboptimal, da diese Information vom Algorithmus nicht genutzt wird. In diesem Fall sind hierarchische Clusteringalgorithmen passender, da sie abhängig von der vorgegebenen Clusteranzahl ihr Resultat liefern. Ihre Berechtigung hat die vorgestellte Arbeit bei den verwandten Arbeiten aufgrund der detaillierten Beschreibung der Ähnlichkeitsfunktion. Für die Ähnlichkeitsfunktion werden die Geschäftsprozessaktivitäten und -übergänge auf je einen Vektor abgebildet und anschließend die Ähnlichkeit über das normierte Skalarprodukt der Vektoren berechnet. Alternativ dazu könnten die Datenpunkte im aufgespannten Vektorraum über ein Distanzmaß, wie die euklidische Distanz, berechnet werden.

3.2.5. Diskussion

An den einzelnen Arbeiten werden die Vor- und Nachteile der einzelnen Clusteralgorithmen deutlich. Je nach Aufgabenstellung und je nach Datensatz sind unterschiedliche Algorithmen geeignet. Der dichte-basierende Algorithmus gruppiert den Datensatz in unbekannt viele Cluster und ignoriert dabei Ausreißer. Im Gegensatz dazu kategorisiert der hierarchische Clusteralgorithmus stets alle Datenpunkte und ist variabel in seiner Clusteranzahl. Ziel dieser Arbeit ist es, die verschiedenen Internetdienste autonom zu klassifizieren, daher sollte aufgrund der anzunehmenden Unvollständigkeit des Datensatzes auf einen Clusteralgorithmus mit einer variablen Clusteranzahl zurückgegriffen werden.

Außerdem wurde in den vorgestellten Arbeiten ersichtlich, dass die Wahl der Distanz- oder Ähnlichkeitsfunktion Basis für ein gutes Clusteringresultat ist. Je nach Ähnlichkeitsfunktion werden verschiedene Unterschiede sichtbar und nur diese können beim Clustering berücksichtigt werden.

3.3. Ableiten von Ontologien aus Webformularen

Um die Webformulare interaktiv ausfüllen zu können, müssen die Daten der Internetdienste durch eine Ontologie repräsentiert werden. Im folgenden werden zwei Arbeiten vorgestellt, die eine Abbildung der Webformulare auf eine Ontologie erzeugen.

3.3.1. Matching Ansatz

Ontologien sind aufgrund ihrer Flexibilität ein beliebtes Werkzeug um Internetinhalte zu repräsentieren und zu interpretieren. In der Arbeit [GMJ04] mit dem Titel *„OntoBuilder: Fully Automatic Extraction and Consolidation of Ontologies from Web Sources“* wird daher ein Programm entworfen, das Webformulare auf Ontologien abbildet. Zusätzlich werden je zwei Ontologien von verschiedenen Webformularen bestmöglichst aufeinander abgebildet, um ihre Ähnlichkeit bzw. ihre ähnliche Funktionalität bestimmen zu können. Die Übereinstimmungsalgorithmen basieren einerseits auf dem Vergleich der Konzepte (Begriffe) der Ontologie und andererseits auf der Anordnung der Konzepte innerhalb des Formulars. Die Algorithmen werden in der Arbeit [Gal+05] mit dem Titel *„Automatic Ontology Matching Using Application Semantics“* detaillierter beschrieben.

Bei der Anwendungssemantik handelt es sich um die zuvor erwähnte Anordnung der Konzepte bzw. Formularelemente innerhalb des Formulars. Anhand der Anwendungssemantik werden algorithmisch Rückschlüsse über eine ähnliche semantische Bedeutung eines Formularelements gezogen. Für den Vergleich der Konzepte wird der Begriffsvergleich und der Wertevergleich vorgestellt. Beim Begriffsvergleich werden die Begriffe, die ein Formularelement beschreiben, verglichen, wie beispielsweise das *name*-Attribut des *input*-Elements. Im Gegensatz dazu werden beim Wertevergleich die Werte verglichen, die ein Formularelement annehmen kann. Beispielsweise sind beim Auswahlelement die möglichen Werte vorgegeben und können verglichen werden. Anhand dieses Programmes kann nun also ein Webformular auf ein anderes abgebildet werden.

3.3.2. Maschinell lernender Ansatz

Die Arbeit [AHS12] mit dem Titel *„Learning to Discover Complex Mappings from Web Forms to Ontologies“* leitet Abbildungen von einem Webformular zu einer Ontologie über ein maschinell lernendes Programm her. Als Eingabe für diese Abbildung erhält die Software das Webformular, eine Ontologie und einfache Zuweisungen. Anhand dieser Eingaben müssen die komplexeren Abbildungen hergeleitet werden. Dafür verfolgt diese Arbeit den Ansatz, anhand bestehender vollständiger Abbildungen zu lernen und daraus verschiedene

Lösungen für die angefragte Abbildung zu finden und zusätzlich den Lösungen eine Wahrscheinlichkeit zuzuordnen. In der Evaluierung dieses Ansatzes erhielten 80% der erwarteten Abbildungen die höchste Wahrscheinlichkeit und bei den anderen war es mindestens unter den besten drei Abbildungsvarianten.

3.3.3. Diskussion

Beide Ansätze erzeugen aus einem Webformular eine Ontologie. Jedoch versuchen beide für ähnliche Webformulare eine passende vereinigte Ontologie zu erstellen. Der Ansatz dieser Arbeit vereinigt jedoch nicht die Ontologien, sondern die Webformulare und anschließend wird für die vereinigten bzw. geclusterten Webformulare eine Ontologie erzeugt. Somit wird der Ansatz des maschinellen Lernens einen Schritt vorher beim Clustering umgesetzt.

Der Ansatz, die Abbildung aufgrund von Matchingalgorithmen zu erzeugen, nutzt zusätzlich zur Syntax des Formulars noch die Semantik, die in den Attributen der einzelnen Formularelemente spezifiziert ist. Dieser vielversprechende Ansatz wird in dieser Arbeit in ähnlicher Form angewandt.

3.4. Zusammenfassung

In diesem Kapitel wurde ein Ausschnitt an Forschungsergebnissen in den Bereichen Webcrawling, Clustering und Ontologieherleitung vorgestellt. Anschließend wurden in einer Diskussion die einzelnen Arbeiten in Bezug auf die Aufgabenstellung, autonom Internetdienste klassifizieren zu können, eingeordnet und bewertet. In den folgenden Kapiteln werden diese Ergebnisse mit in die Analyse und den Entwurf eines Programms zur Klassifizierung von Internetdiensten eingebracht.

4. Analyse

In diesem Kapitel werden die einzelnen Teilaufgaben der Zielsetzung untersucht, die in der Abbildung 4.1 skizziert sind. Betrachtet man den Ablauf der Analyse in umgekehrter Richtung, dann muss zur interaktiven und natürlichen Sprachkommunikation zwischen Nutzer und Dienst eine aktive Ontologie für den Dienst erzeugt werden. Zur Erzeugung dieser aktiven Ontologien wird in dieser Arbeit ein Konstruktionsplan entworfen, der Regeln für die Abbildung von Formularelementen, Internetdiensten und Internetdienstkategorien angibt. In einer weiterführenden Arbeit soll aus diesem Konstruktionsplan für jede klassifizierte Internetdienstkategorie eine allgemeingültige aktive Ontologie entworfen werden. Dafür werden in dieser Arbeit die einzelnen Internetdienste anhand eines Clusteringalgorithmus gruppiert. Der Algorithmus und die verwendeten Merkmale zur Klassifizierung bilden den Kern der Analyse. Voraussetzung für diesen Schritt ist jedoch das Vorhandensein eines Datensatzes an Internetdiensten. Dazu wird mittels eines Webcrawlers das Internet nach Internetdiensten durchsucht. Im folgenden werden die einzelnen beschriebenen Schritte in korrekter Ablaufrichtung detailliert analysiert.

4.1. Sammeln von Internetdiensten

Nach der Definition im Abschnitt 2.1 handelt es sich bei Internetdiensten um HTML-Formulare, die innerhalb der <*form*>-Umgebung spezifiziert sind. Das Auffinden dieser Dienste wird durch einen selbst entworfenen Webcrawler übernommen. Dazu sucht der Webcrawler nacheinander in verschiedenen HTML-Dokumenten nach Implementierungen eines Internetdienstes und bei einem Treffer werden die für das Clustering benötigte Informationen extrahiert. In diesem Fall beziehen sich die benötigten Informationen auf das Formular selbst und die Webseitenbeschreibung, auf der sich der Internetdienst befindet, die der <*head*>-Umgebung entspricht. Detaillierte Informationen zum Entwurf und der Implementierung des Webcrawlers können im Kapitel 5 nachgelesen werden.

4.2. Merkmalsmuster

Nachdem mit dem Webcrawler eine Sammlung an Internetdiensten gefunden wurde, ist gemäß der Abbildung 4.1 der nächste Schritt die Extraktion von Merkmalen. Diese wird für jeden Internetdienst anhand der Daten des Webcrawlers vorgenommen.

Für die Extraktion von Merkmalen wurden Merkmalsmuster entwickelt, die Regeln für die Erzeugung eines Merkmals definieren. Die einzelnen Merkmalsmuster wurden dabei aus

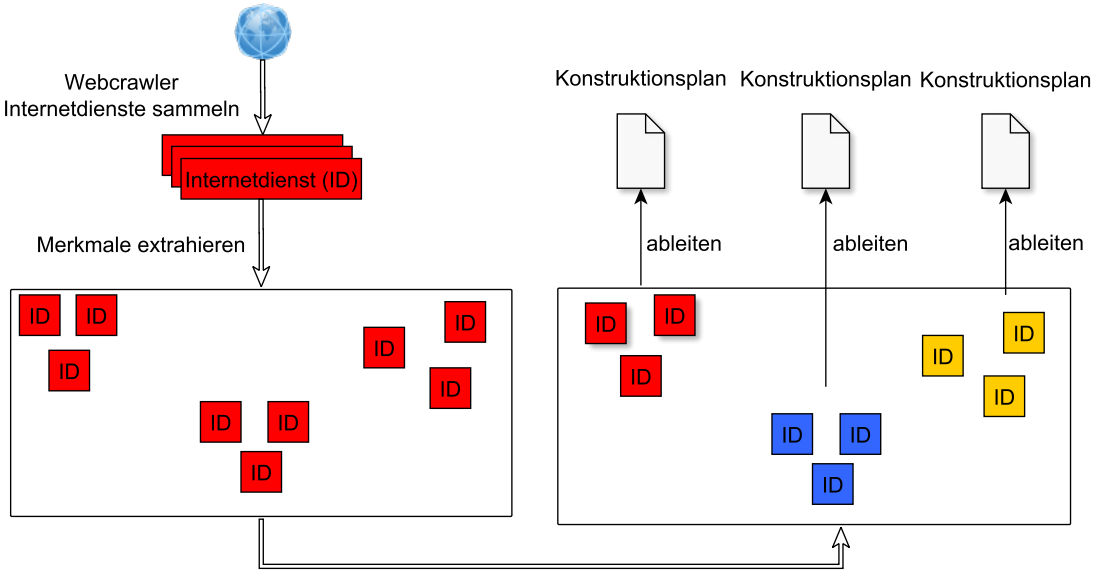


Abbildung 4.1.: Die einzelnen Schritte vom Aufbau einer Stichprobe über die Kategorisierung der Internetdienste bis zum Entwurf des Konstruktionsplans.

den Formular- und Metadatenelementen, die der Webcrawler pro Internetdienst herausfiltert, hergeleitet. Bei jedem Merkmalsmuster entstehen ausschließlich binäre Merkmale. Die binären Merkmale geben Auskunft darüber, ob das Merkmal, unabhängig von seiner Häufigkeit, bei einem Internetdienst auftritt.

In der Abbildung 4.2 ist beispielhaft die Ableitung zweier Merkmale auf Basis eines Merkmalsmusters dargestellt. In diesem Fall definiert das Merkmalsmuster für jedes Texteingabefeld in einem Internetdienst zusammen mit der Semantik des jeweiligen Feldes ein Merkmal. Durch diese dynamische Merkmalserzeugung werden selbst für zukünftige Internetdienste, die nicht im Datensatz enthalten sind, autonom passende Merkmale extrahiert. Nachfolgend werden die in der Tabelle 4.1 aufgeführten Merkmalsmuster erläutert.

4.2.1. Anzahl an Elementen

Diese Gruppe von Merkmalsmustern zählen je nach Muster die Anzahl bestimmter Elemente innerhalb des Internetdiensts. Mittels dieser Anzahl werden daraus von eins bis n binäre Merkmale erzeugt, wobei sich n auf die maximale Anzahl an gezählten Elementen innerhalb der Trainingsmenge bezieht. Jedem dieser Merkmale wird ein Wert $k(1 \leq k \leq n)$ zugewiesen, der Auskunft darüber gibt, ob der jeweilige Dienst mehr als k gezählte Elemente enthält. Für die spätere Vereinigung der Merkmale im Abschnitt 4.3 wird bei dieser Art von Merkmalen die semantische Übereinstimmung bei Gleichheit des Wertes k hergestellt.

In der Testphase wurde auch versucht, anstatt eines n-binären ein numerisches Merkmal zu erzeugen. Dies führte jedoch zu schlechteren Ergebnissen, da Internetdienste sehr variabel implementiert werden können und somit bei den numerischen Merkmalen sehr viel Ausreißer entstehen. Allein durch einen Ausreißer bei einer der drei folgenden Merkmalsmuster lässt sich der Dienst nur noch selten korrekt gruppieren. Im Gegensatz dazu ist bei den binären Merkmalen die Aussagekraft geringer, da nur bestimmt wird, ob etwas mehr oder weniger Elemente realisiert, jedoch entstehen dadurch mehrere Merkmale, von denen später die redundanten und irrelevanten entfernt werden können. Dies hat zur Folge, dass nur die aussagekräftigen Merkmale dieser drei Merkmalsmuster verwendet werden.

Im folgenden werden die drei Merkmalsmuster beschrieben, die die Anzahl an sichtbaren, interaktiven Formularelementen, Eingabeelementen und Navigationselementen zählen.

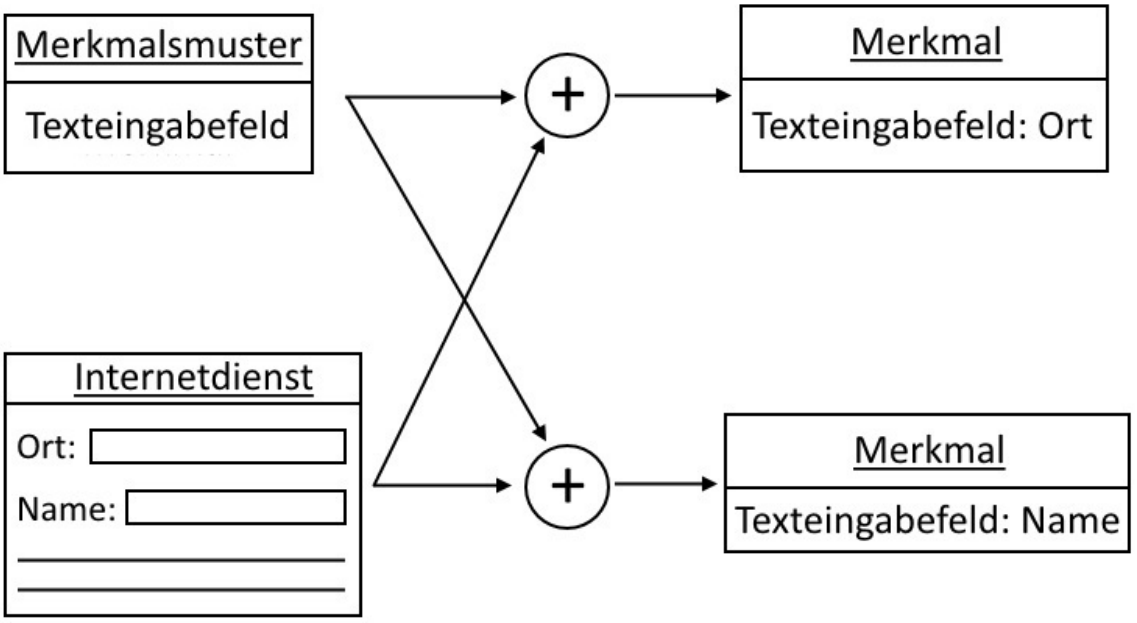


Abbildung 4.2.: Merkmalsmuster: Ableitung zweier Merkmale auf Basis des Merkmalsmusters für ein Texteingabefeld.

4.2.1.1. Anzahl an Eingabeelementen

Bei der Berechnung der Anzahl an Eingabeelementen werden alle Elemente der Abbildung 4.3a mit einbezogen. Basis dieses Merkmalsmusters ist die große Variationsmöglichkeit bei der Implementierung von Auswahl- und Navigationselementen; die Anzahl der Eingabeelemente variiert in vielen Fällen dagegen sehr schwach.

4.2.1.2. Anzahl an sichtbaren, interaktiven Formularelementen

Dieses Merkmalsmuster zählt alle sichtbaren, interaktiven Formularelemente. Diese Begrenzung führt dazu, dass alle Elemente des Merkmalsmusters, die Anzahl an Eingabeelementen plus die in der Abbildung 4.3b aufgeführten Elemente gezählt werden. Durch die zusätzliche Aufnahme der Auswahlelemente wird nun der Fall abgefangen, dass für die Abfrage der gleichen Information je nach Internetdienst einmal ein Eingabeelement und einmal ein Auswahlelement verwendet wird. Ein Beispiel dafür ist die Abfrage der Uhrzeit.

4.2.1.3. Anzahl an Navigationselementen

Navigationselemente liefern im Gegensatz zu den Formularelementen dem Nutzer Informationen. So muss beispielsweise der Dienstanbieter bei sensiblen Daten den Nutzer informieren, was mit den Daten passiert. Daher werden auch innerhalb von Formularen Navigationselemente verwendet, deren Häufigkeit über dieses Merkmalsmuster gezählt wird.

4.2.2. Webseitenbeschreibung

Dieses Merkmalsmuster analysiert die Beschreibung der Internetseite, auf der das Formular spezifiziert ist. Dazu wird der Inhalt der Metadatenelemente untersucht. Die Abbildung 4.2 zeigt die beiden Metadatenelemente, wobei beim *meta*-Element der Inhalt über das *content*-Attribute ausgewertet wird. Die Auswertung erfolgt durch die Erzeugung von jeweils einem Merkmal für jedes der Nomen, die in der Beschreibung enthalten sind.

Merkmalsmuster	Seite
Anzahl an Elementen	
Anzahl an sichtbaren, interaktiven Formularelementen	29
Anzahl an Eingabeelementen	29
Anzahl an Navigationselementen	29
Webseitenbeschreibung	
Webseitenbeschreibung	29
Formularelemente	
Passworteingabe	31
Einfache Auswahl	31
Mehrfache Auswahl	32
Einzeiliges Texteingabefeld	32
Mehrzeiliges Texteingabefeld	32
Spezielle Eingabefelder	33
Link	33

Tabelle 4.1.: Übersicht über alle Merkmale

Element	Typ
input	text
	email
	password
	search
	url
	number
	datetime
	datetime-local
	date
	month
	week
	time
	tel
textarea	-

(a) Anzahl an Eingabeelementen

Element	Typ
input	checkbox
	radio
select	-

(b) Anzahl an sichtbaren interaktiven Formularelementen: Diese Elemente werden zusätzlich zu den Elementen in (a) gezählt.

Abbildung 4.3.: Auflistung aller Elemente, die das jeweilige Merkmalsmuster zählt.

Element	Attributwert
meta	name=„description“
	name=„keywords“
	property=„og:title“
	property=„og:description“
title	-

Tabelle 4.2.: Elemente der Webseitenbeschreibung: Die aufgeführten Elemente mit den jeweiligen Attributwerten geben Auskunft über die jeweilige Webseite.

4.2.3. Formularelemente

Für die Abfrage von Daten werden Formularelemente verwendet. Je nach Daten, die der jeweilige Internetdienst benötigt, können Rückschlüsse auf die Tätigkeit des Dienstes geschlossen werden. Im nachfolgenden werden verschiedene Merkmalsmuster erläutert, die Formular- bzw. Navigationselemente und gegebenenfalls deren Semantik auf Merkmale abbilden. Sollte bei einem Merkmalsmuster die Semantik des jeweiligen Elements benötigt werden, dann wird diese einerseits aus verschiedenen Attributwerten und andererseits aus dem zum Element gehörigen Label ermittelt. Je nach Merkmalsmuster werden die für die Semantikerkennung miteinbezogenen Attribute angegeben.

$$\begin{aligned} &<\text{label} > \text{LABEL} < /\text{label} > \\ &<\text{input attr} = \text{"ATTRIBUTEWERT"} \cdots > \end{aligned} \tag{4.1}$$

Aus den einzelnen Werten werden die Nomen herausgefiltert und dem Merkmal in einer Liste hinzugefügt. Diese Liste bildet die Semantik des Merkmals und anhand dieser können Merkmale auf semantische Gleichheit überprüft werden.

4.2.3.1. Passworteingabe

Für die Eingabe eines Passworts wird das Passworteingabefeld verwendet. Es hat die Eigenschaft, dass jedes eingetippte Symbol durch das gleiche Symbol (meist eine Kugel •) im Eingabefeld angezeigt wird. Intern wird jedoch das unverschlüsselte Passwort als Teil des Inhalts übermittelt.

Für die Erzeugung eines Eingabefelds mit den zuvor genannten Eigenschaften gibt es nur eine Möglichkeit und zwar muss das Eingabefeld vom Typ *password* sein, wie dies nachfolgend illustriert ist.

$$< \text{input type} = \text{"}password\text{"} ... >$$

Aufgrund dieser eindeutigen Typisierung ergibt sich durch die Verwendung automatisch die Semantik dieses Elements. Somit wird dieses Merkmalsmuster beim Auftreten eines Passworteingabefelds stets auf das selbe Merkmal abgebildet.

4.2.3.2. Einfache Auswahl

Bei der einfachen Auswahl kann der Nutzer sich zwischen mehreren Auswahlmöglichkeiten für genau eine entscheiden. Je nach Anwendung kann dies über Radiobuttons

$$< \text{input type} = \text{"}radio\text{" name} = \text{" ..."} \cdots >$$

oder über eine Auswahlliste

$$\begin{aligned} &<\text{select name} = \text{" ..."} \cdots > \\ &\quad < \text{option} > \cdots < /\text{option} > \\ &\quad \cdots \\ &</\text{select} > \end{aligned} \tag{4.2}$$

implementiert werden. Bei den Radiobuttons werden die Auswahlmöglichkeiten anhand des *name*-Attributs zu einer Auswahl gruppiert. Je nach Semantik kann durch die einfache Auswahl eine andere Information abgefragt werden, daher muss diese unterschieden werden. Die Semantik wird einerseits über die Nomen des *name*- und *value*-Attributs und andererseits über die Nomen der Auswahlmöglichkeiten bestimmt.

4.2.3.3. Mehrfache Auswahl

Bei der mehrfachen Auswahl kann der Nutzer aus einer Menge an Auswahlmöglichkeiten eine beliebige Anzahl wählen. Diese Art der Auswahl wird entweder mit Auswahlkästchen,

$$< \text{input type} = \text{"}\textit{checkbox}\text{" name} = \text{" ..." value} = \text{" ..."} \cdots >$$

die über das *name*-Attribute gruppiert werden oder über eine Auswahlliste,

$$\begin{aligned} &<\text{select name} = \text{" ..." multiple} > \\ &\quad < \text{option} > \cdots < /\text{option} > \\ &\quad \ldots \\ &</\text{select} > \end{aligned} \tag{4.3}$$

die mit dem Schlüsselwort *multiple* realisiert wird, implementiert. Ähnlich wie bei der einfachen Auswahl kann den Auswahlelementen je nach Auswahlfrage eine andere Semantik zugeordnet werden. Daher wird je nach Semantik ein Merkmal erzeugt. Um die Bedeutung der jeweiligen mehrfachen Auswahl zu erfassen, werden die Nomen aus den Auswahloptionen und den *name*- und *value*-Attributen gefiltert.

4.2.3.4. Einzeiliges Texteingabefeld

Das einzeilige Texteingabefeld ist, aufgrund seiner vielseitigen Nutzbarkeit, das am häufigsten verwendete Formularelement. Es ist das initiale input-Element und kann somit entweder ohne *type*-Attribut

$$< \text{input name} = \text{" ..." title} = \text{" ..." placeholder} = \text{" ..."} \cdots >$$

oder mit entsprechender Typisierung

$$< \text{input type} = \text{"}\textit{text}\text{" name} = \text{" ..." title} = \text{" ..." placeholder} = \text{" ..."} \cdots >$$

implementiert werden. Das Texteingabeelement kann jede Art von Symbol aufnehmen und folglich auch jede Art von Information abfragen. Daher wird für die Merkmalserzeugung zwischen der Semantik der Texteingabefelder unterschieden. Die Semantik wird über das *name*-, *title*- und *placeholder*-Attribute sowie das dazugehörige Label des Elements bestimmt.

4.2.3.5. Mehrzeiliges Texteingabefeld

Bei dem vorgestellten einzeiligen Eingabefeld können lediglich kurze und knappe Informationen übermittelt werden. Für die Übertragung längerer Texte wird ein mehrzeiliges Eingabefeld benötigt. Das mehrzeilige Eingabefeld lässt einerseits eine übersichtlichere Textformatierung mithilfe des Zeilenumbruchs zu und andererseits erweitert es die Menge des sichtbaren Textes durch die Mehrzeiligkeit.

Dieses mehrzeilige Eingabefeld wird durch das *textarea*-Element erzeugt:

$$< \text{textarea} > \cdots < /\text{textarea} >$$

Durch die Mehrzeiligkeit ist die Verwendung dieses Elements auf längere Nachrichten eingeschränkt, wie beispielsweise Fragen und Kritiken. Durch den spezialisierten Einsatz dieses Elements wird vom Merkmalsmuster keine semantische Unterscheidung vorgenommen. Somit bildet jedes der mehrzeiligen Texteingabefelder auf das selbe Merkmal ab.

4.2.3.6. Spezielle Eingabefelder

Die speziellen Eingabefelder umfassen alle Typisierungen des *input*-Elements der Tabelle 2.3, ausgenommen die bereits verarbeiteten Typen *text*, *radio* und *checkbox*. Jeder dieser Typen bildet auf ein Merkmal ab, bei denen aufgrund der Spezialisierung die Semantik nicht unterschieden wird. Trotzdem wird die Semantik der einzelnen speziellen Eingabefelder über das *name*-, *title*- und *placeholder*-Attribut ermittelt. Dies hat den Zweck, das Merkmal eines einzeiligen Texteingabefelds anhand der semantischen Übereinstimmung einem Merkmal eines speziellen Eingabefelds zuzuordnen.

4.2.3.7. Link

In manchen Internetdiensten werden auch Links zu anderen Webdokumenten verwendet, beispielsweise bei der Registrierung auf einer Internetseite. In diesem Fall ist der Anbieter gesetzlich dazu verpflichtet, einen Link mit seinen allgemeinen Geschäftsbedingungen dem Nutzer zur Verfügung zu stellen.

Je nach Link wird dem Nutzer eine andere Information bereitgestellt, daher muss jeweils die Semantik der Links unterschieden werden. Zur Bestimmung der Semantik werden die Nomen des Linktexts in einer Liste gespeichert.

4.3. Vereinigung von Merkmalen

Im vorherigen Kapitel wurden die einzelnen Merkmalsmuster und deren Abbildung auf konkrete Merkmale erläutert. Der nächste Schritt ist nun gleiche Merkmale zu vereinigen. Dies wird je nach Merkmalsmuster unterschiedlich vorgenommen. Die Abbildung 4.4 skizziert den Ablauf einer Vereinigung zweier Merkmale. Dabei wird zuerst geschaut, ob sich das untere Merkmal auf das obere Merkmal abbilden lässt. Die verschiedenen Ableitungsregeln der einzelnen Merkmalsmuster sind in der Tabelle 4.3 dargestellt. Der Aufbau dieser Tabelle wird im nächsten Abschnitt detaillierter beschrieben. Falls nun die Abbildung des einen Merkmalsmusters auf das andere zulässig ist, wird als nächstes die Semantik beider Merkmale verglichen. Bei einigen Merkmalen wird die Semantik nicht unterschieden, deshalb entfällt hier die Prüfung der semantischen Übereinstimmung. Sollten die Merkmale semantisch übereinstimmen, dann können sie vereint werden, indem die Begriffslisten vereinigt werden.

Für die semantische Übereinstimmung muss mindestens ein Wortpaar, das sich aus je einem Wort aus den Begriffslisten der beiden Merkmale bildet, der folgenden Ungleichung genügen:

$$0.8 < \frac{\text{Länge des längsten gemeinsamen Teilstrings}}{\text{Länge des kürzeren Strings}}$$

Die Gleichung besagt, dass die Division aus Länge des längsten gemeinsamen Teilstrings durch Länge des kürzeren Strings größer als der Schwellwert von 80% sein muss. Die Länge bezeichnet dabei die Anzahl der Buchstaben.

Die Tabelle 4.3 verdeutlicht, anhand welcher Tests ein Merkmal eines Merkmalsmusters(Spalte) auf ein Merkmal eines anderen oder gleichen Merkmalsmusters (Zeile) abgebildet werden kann. Das •-Symbol signalisiert, dass alle Merkmale des Merkmalsmusters (Spalte) bedingungslos, ohne Test der semantischen Übereinstimmung, mit den Merkmalen des Merkmalsmusters (Zeile) vereint werden können. Somit erzeugen alle Merkmalsmuster, die in ihrer Spalte nur ein •-Symbol auf der Matrixdiagonalen haben, nur ein Merkmal. Im Gegensatz dazu präsentiert das ×-Symbol ausschließlich eine Vereinigung zwischen

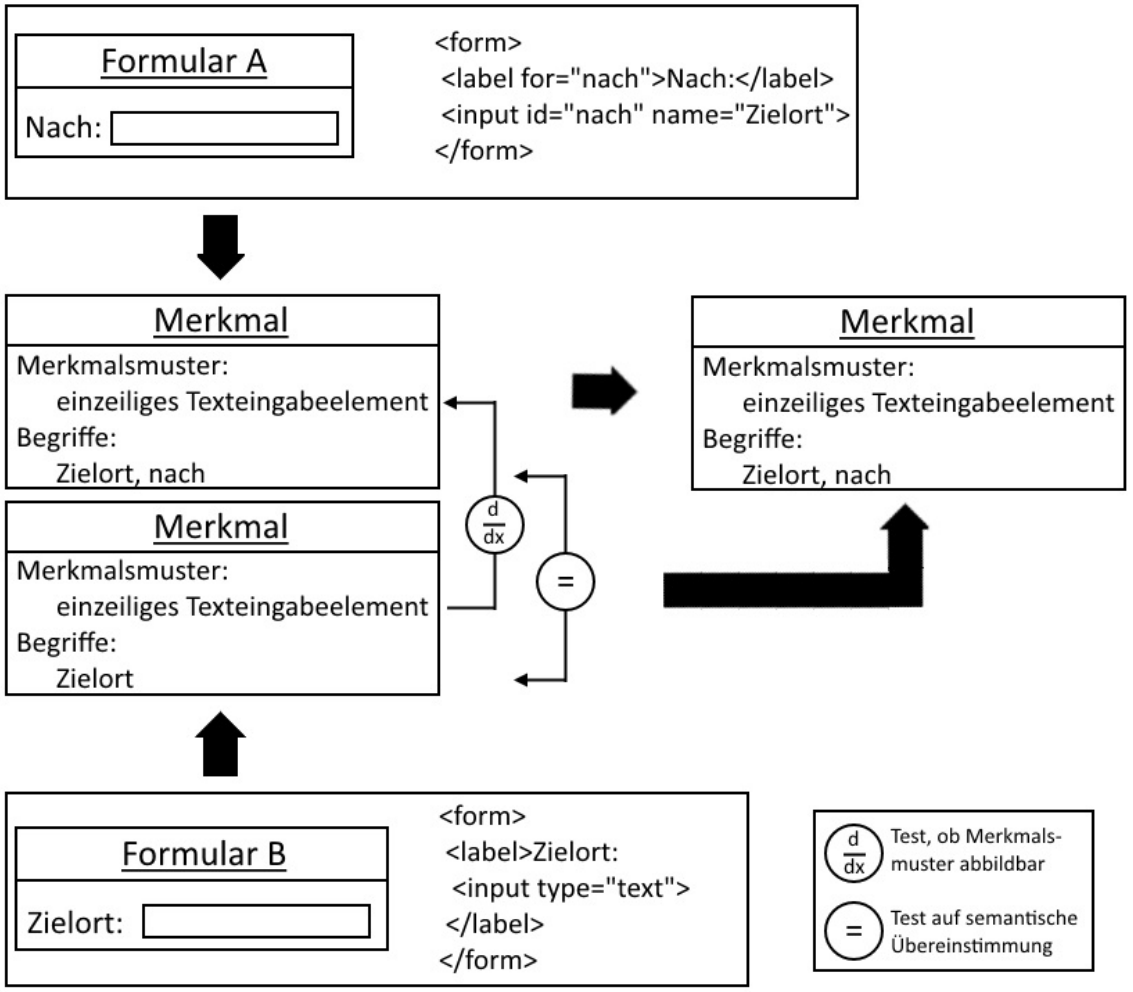


Abbildung 4.4.: Ablauf für die Vereinigung zweier Merkmale: Das Merkmal des Formular B soll mit dem Merkmal von Formular A vereinigt werden. Dafür werden die Merkmalsmuster anhand der Tabelle 4.3 abgeglichen und die semantische Übereinstimmung überprüft.

zwei Merkmalen der jeweiligen Merkmalsmuster, wenn die Semantik übereinstimmt. Die semantische Übereinstimmung wird über einen Begriffsvergleich anhand der Nomenliste jedes Merkmals geprüft. Dazu wird jeder dieser Begriffe der beiden Listen miteinander verglichen und bei Übereinstimmung werden die Merkmale sowie die Liste vereint.

In der Tabelle 4.3 werden drei Abweichungen beim Vereinigen von Merkmalen dargestellt. Die erste Abweichung lässt eine Vereinigung der Merkmale eines einzeiligen Textausgabefeldes mit dem Merkmal eines speziellen Eingabefelds zu. Dies basiert auf der Entwicklung von HTML. Das Texteingabeelement kann jede Art von Symbol aufnehmen. Diese Variabilität hat jedoch bei speziellen Abfragen wie beispielsweise einem Datum seine Nachteile, da stets die Gültigkeit der Eingabe abgefragt werden muss. Deshalb wurden im Laufe des Entwicklungszyklus von HTML für häufige Verwendungsarten speziellere Eingabefelder entwickelt, wie beispielsweise die Uhrzeit- und Datumseingabe. Diese spezielleren Anfragen können jedoch auch weiterhin mit dem einzeiligen Texteingabefeld abgefragt werden. Anhand der Semantik werden nun die Merkmale dieses Elements unterschieden und zusätzlich noch überprüft, ob eines der einzeiligen Texteingabefelder semantisch mit einem spezielleren Eingabefeld übereinstimmt. Ist dies der Fall, dann wird das Merkmal zum Merkmal des spezielleren Elements hinzugefügt.

Merkmalsmuster	# Eingabeelemente	# interaktive Formularelemente	# Navigationselemente	Webseitenbeschreibung	Passworteingabe	Einfache Auswahl	Mehrfache Auswahl	einzeiliges Texteingabeelement	mehrzeiliges Texteingabeelement	Spezielle Eingabefelder	Link
# Eingabeelemente	×										
# interaktive Formularelemente		×									
# Navigationselemente			×								
Webseitenbeschreibung				×							
Passworteingabe					•						
Einfache Auswahl						×		×			
Mehrfache Auswahl							×				
einzeiliges Texteingabeelement								×			
mehrzeiliges Texteingabeelement									•		
Spezielle Eingabefelder						×		×		×	
Link											×

Tabelle 4.3.: Vereinigung von Merkmalen: Die Vereinigungsmatrix gibt an, welche Merkmale eines Merkmalsmusters (Spalte) mit welchem Merkmal eines anderen Merkmalsmusters (Zeile) verschmolzen werden können. Zusätzlich wird unterschieden, ob die Vereinigung bedingungslos ist (•) oder an die semantische Übereinstimmung (×) der Merkmale geknüpft ist.

Die zweite Abweichung ist, dass Merkmale der einfachen Auswahl wiederum auf Merkmale der speziellen Eingabefelder abgebildet werden können. Die Begründung liegt auch hier in der Entwicklung spezieller Eingabefelder zum Beispiel für die Uhrzeit können stattdessen auch zwei Auswahllisten für die Stunde und die Minute verwendet werden.

Als drittes kann ein Merkmal des Texteingabefeldes noch auf ein Merkmal der einfachen Auswahl abgebildet werden. Hintergrund dafür ist, dass in vielen Fällen mit beiden Elementen die selbe Information abgefragt werden kann und somit müssen die entstehenden Merkmale aufeinander abgebildet werden können.

4.4. Selektion der Merkmale

Die Abbildung 4.5 illustriert die Reduktion der Merkmale in diesem Unterkapitel. Als Ausgangssituation existiert eine Abbildung in der für jeden Internetdienst (Zeile) das vorkommen welchen Merkmals (Spalte) festgehalten ist. Diese Abbildung wurde bei der Merkmalsextraktion und -vereinigung erzeugt. Sie enthält eine Vielzahl an Merkmalen, die redundant oder irrelevant sind. Diese Merkmale werden bei der Merkmalsselektion gefiltert und entfernt. Somit entsteht nach der Selektion eine kleinere Menge an Merkmalen.

Abbildung 4.5.: Merkmalsselektion: Die Merkmalsselektion entfernt redundanten und irrelevante Merkmale.

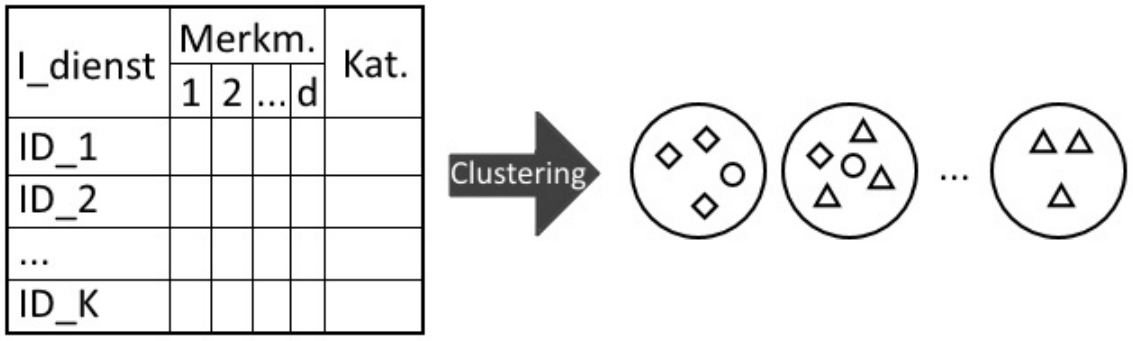


Abbildung 4.6.: Clustering: Ablauf von der Eingabe der Mekmale bis zur Ausgabe der Cluster

4.5. Clustering

Die Grundidee des Clusterings ist, gemäß der Abbildung 4.6, nach Erhalt der Merkmale die verschiedenen Internetdienste einem Cluster zuzuweisen. Dieses Cluster bildet eine Gruppe und ist in der zuvor genannten Abbildung als großer Kreis dargestellt. Für die Abbildung der Internetdienste können verschiedene Algorithmen verwendet werden. Bei dieser Arbeit wurde der *DBScan* und der *Spectral Clustering*-Algorithmus verwendet, die im Unterabschnitt 5.3.7 beschrieben und im Abschnitt 6.4 evaluiert werden. Die je nach Algorithmus erzeugten Cluster müssen zuletzt noch einer Internetdienstkategorie zugeordnet werden.

4.6. Kategoriezuordnung der Cluster

Die Zuordnung der Cluster basiert auf der bereits bekannten Kategoriezuordnung für die Internetdienste der Trainingsmenge. Anhand dieser bekannten Zuordnung wird innerhalb eines Clusters die Kategorie mit den meisten Internetdiensten bestimmt und der Cluster dieser Kategorie zugeordnet. In der Abbildung 4.7 wird dieses Verfahren an einem Beispiel illustriert. Der linke Cluster wird der Kategorie $\diamondsuit$ zugeordnet, da in diesem Cluster Internetdienste dieser Kategorie am häufigsten vorkommen. Diese Zuordnung hat eine Ausnahme und zwar, wenn ein Cluster nur aus Internetdiensten der Testmenge besteht. Für diesen Fall existiert eine zusätzliche Kategorie, der dieses Cluster zugeordnet werden.

4.7. Kategorien von Internetdiensten

Im Zusammenspiel zwischen den gefundenen Internetdiensten des Abschnitts 4.1 und den herausgearbeiteten Merkmalsmustern der Tabelle 4.1 haben sich die folgenden Internetdienstkategorien herauskristallisiert. Die Tabelle 4.4 führt alle geclusterten Internetdienstkategorien auf.

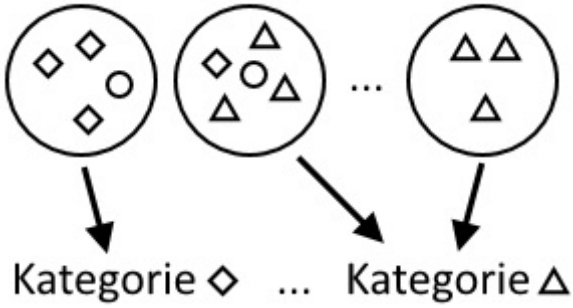


Abbildung 4.7.: Clusterzuordnung: Die Zuordnung eines Clusters zu einer Internetdienstkategorie basiert auf der Häufigkeit.

Kategorie	Seite
Login	37
Registrierung	38
Kontaktformular	39
Fahrplanauskunft	41
Flugauskunft	41
Autovermietung	44
Unterkunftssuche	45
Newsletterabonnierung	45
Wettervorhersage	48

Tabelle 4.4.: Internetdienstkategorien: Eine Liste aller Internetdienstkategorien und die dazugehörige Seitenreferenzierung für deren Beschreibung.

4.7.1. Login

Eines der häufigsten Internetdienste ist der Logindienst. Beim Login meldet sich der Benutzer in einem System an, um auf sein personalisiertes Benutzerkonto zugreifen zu können. Das Login dient der Authentifizierung des Nutzers und besteht in den meisten Fällen aus dem Nutzernamen und einem Passwort. Der Nutzername kann eine Abwandlung des Geburtsnamens, eine E-Mailadresse oder vom Nutzer frei erfunden sein. Diese Information allein genügt dem System jedoch nicht zur Gewährleistung einer sicheren Authentifizierung des Nutzers. Erst durch die Eingabe eines Passworts, das der Nutzer selbst gewählt hat, kann in der übereinstimmenden Kombination zwischen Nutzername und Passwort der Zugang zu dem Benutzerkonto gewährt werden.

Die Abbildung 4.8 zeigt einen exemplarischen Logindienst, wie er im Browser dargestellt wird. Der Dienst enthält je nach Art des Benutzernamens ein Text- bzw. E-Maileingabefeld, ein Passworteingabefeld, einen „Passwort vergessen?“-Link und eine Schaltfläche zur Ausführung des Logins. Zusätzlich wird dem Nutzer von einigen Login-Dienstanbietern über ein Auswahlkästchen das optionale Feature zur Verfügung gestellt, mit dem er unter der Verwendung von Cookies angemeldet bleibt.

Auf Grundlage des Clusteringergebnisses werden einige Internetdienste des Loginclusters miteinander verglichen. Dazu werden in der Tabelle 4.5 aus den ausgewählten Internetdiensten alle gefundenen Merkmale aufgeführt und in Beziehung zu den jeweiligen Internetdiensten gesetzt. Durch die eingeschränkte Variabilität bei der Implementierung dieses Diensts bildet die Anzahl an Eingabeelementen ein Indiz für diesen Cluster. Darauf aufbauend enthält jeder untersuchte Internetdienst noch das semantisch identische Element für die Passworteingabe. Zusätzlich kann aus der Tabelle entnommen werden, dass die

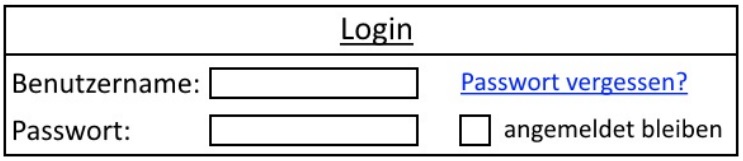


Abbildung 4.8.: Logindienst: Authentifizierung eines Nutzers über den Logindienst.

Merkmal: / Dienstanbieter:		kvv-shop.de	kvv.de	sparkassen-shop.de	bahn.de	aohostels.com	mvv-energie.de/de	hirschreisen.de/.../login	facebook.com/login
Merkmalsmuster	**Semantik**								
# Eingabeelemente		2	2	2	2	2	2	2	2
# Formularelemente		2	2	2	2	2	2	2	3
# Navigationselemente		2	1	2	2	0	1	0	2
Webseitenbeschreibung	Login	-	-	-	-	-	-	●	●
Einzeiliges Texteingabefeld	Benutzername	●	●	●	●	-	●	●	●
Spezielle Eingabefelder	E-Mail	-	-	-	-	●	-	-	-
Passworteingabe		●	●	●	●	●	●	●	●
Link	Passwort vergessen	●	●	●	●	-	●	-	●
Link	Neuregistrierung	●	-	●	●	-	-	-	●
Einfache Auswahl	angemeldet bleiben	-	-	-	-	-	-	-	●

Tabelle 4.5.: Vergleich einiger Logindienste: Auswertung der Merkmale für die jeweiligen Logindienste. Die rot eingerahmten Merkmale treten bei allen Logindiensten auf.

Internetdienste entweder ein spezielles Emaileingabefeld oder Texteingabefeld für den Benutzernamen haben. Durch die Möglichkeit, Texteingabefelder auf spezielle Eingabefelder gemäß des Unterkapitels 4.3 abzubilden, können je nach Datensatz diese beiden Merkmale vereint werden und bilden in diesem Fall dann ein weiteres Indiz. Die anderen Merkmale sind kein Indiz für diesen Cluster, da sie je nach Logindienst realisiert werden.

4.7.2. Registrierung

Mit der Registrierung wird das Anlegen eines Kontos in der Benutzerverwaltung übernommen. Bei entsprechend angebotener Schnittstelle kann der Nutzer auf dieses Konto über den zuvor beschriebenen Logindienst zugreifen. Die Registrierung eines Nutzers kann einerseits administrativ von der Verwaltung angelegt werden oder andererseits über einen Dienst erfolgen. Im folgenden wird die Registrierung eines Benutzerkontos mittels eines Internetdiensts betrachtet.

Für das Anlegen eines Benutzerkontos müssen mindestens Daten für eine spätere Authentifizierung des Nutzers abgefragt werden. Zusätzlich wird häufig um eine wiederholte Eingabe des Passworts gebeten, da der Nutzer durch die sofortige Verschlüsselung keine Kontrollmöglichkeit über die korrekte Eingabe hat. Je nach Registrierung können noch zusätzlich personenbezogene Daten abgefragt werden, um einerseits den Nutzer zu identifizieren und andererseits weitere Dienste nutzen zu können. Bei Einkaufsportalen wird

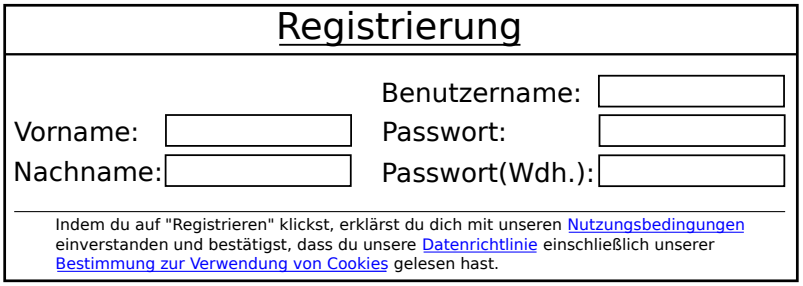


Abbildung 4.9.: Registrierungsdienst: Registrierung eines neuen Nutzers mittels des Registrierungsdiensts.

beispielsweise bei der Registrierung direkt nach der Zahlungsmethode und den Bankkontodaten gefragt, denn diese Daten werden für den Einkauf benötigt.

Die Abfrage nach Daten des Nutzers bzw. von personenbezogenen Daten verpflichtet den Dienstbetreiber zur Erfüllung von gesetzlichen Formalien. Der Dienstbetreiber ist dazu verpflichtet, den Nutzer bei der Registrierung über den Umgang mit seinen Daten zu informieren. Zusätzlich müssen dem Nutzer noch die Nutzungsbedingungen meistens in Form der allgemeinen Geschäftsbedingungen zugänglich gemacht werden.

Die Abbildung 4.9 zeigt beispielhaft einen Registrierungsdienst. Dabei werden personenbezogene Daten über standardmäßige Texteingabefelder oder im besonderen Fällen wie beispielsweise der Telefonnummer über die speziellen Eingabefelder abgefragt. Die Authentifizierungsdaten werden gemäß der im Unterabschnitt 4.7.1 erläuterten Formularfelder ermittelt. Die gesetzlich vorgeschriebenen Dokumente zur Datenschutzrichtlinie, den Nutzungsbedingungen und der Verwendung von Cookies werden jeweils per Link dem Nutzer zugänglich gemacht.

Die Tabelle 4.6 führt einen Teil der gefundenen Merkmale auf, die von den sieben aufgelisteten Internetdiensten extrahiert wurden. Es kristallisiert sich bei der Auswertung der Dienste heraus, dass erneut die Passworteingabe sowie zusätzlich ein spezielles E-Maileingabefeld von jedem der Dienste abgefragt wird. Die restlichen Merkmale werden alle je nach Anwendungsfall realisiert und bilden kein Indiz für diesen Cluster.

4.7.3. Kontaktformular

Viele Firmen bieten ihren Kunden die Dienstleistung an, sich mit ihnen auszutauschen. Diese Dienstleistung kann je nach Firmenbranche verschieden umgesetzt werden. Für Firmen, die eine sofortige Unterstützung bei ihren Produkten anbieten, werden Hotlines eingerichtet und bei nicht so dringlichen Problemen kann auf der Onlinepräsenz Kontakt mit der Firma aufgenommen werden. Die Abbildung 4.10 stellt exemplarisch ein Kontaktformular dar.

Dieses Kontaktformular muss als Mindestanforderung ein Textfeld enthalten, in dem der Kunde seine Kritik äußern kann. Dieses Textfeld wird typischerweise durch ein mehrzeiliges Texteingabefeld realisiert, denn durch den Einsatz von Zeilenumbrüchen kann die Leserlichkeit erheblich gesteigert werden. Zusätzlich sind häufige Abfragewerte der Betreff der Kontaktaufnahme, die elektronische Adresse sowie die Postanschrift, damit die Organisation auf die Äußerung antworten kann.

Die Tabelle 4.7 führt für sieben Internetdienste des Kontaktformulars einen Ausschnitt an abgeleiteten Merkmalen auf. Dabei wird ersichtlich, dass bei dieser Dienstkategorie keine Navigationselemente verwendet werden. Außerdem ist ein Indiz für diesen Cluster die Verwendung eines mehrzeiligen Texteingabefeldes. Zusätzlich wird bei allen Diensten jeweils

Merkmal: / **Dienstanbieter:**		xing.com	linkedin.com	ubuntuusers.de/register	facebook.com	kvv-abo.de/register	shutterstock.com	otto.de/user/register
Merkmalsmuster	**Semantik**							
# Eingabeelemente		4	4	5	5	7	3	6
# Formularelemente		5	4	6	10	12	5	8
# Navigationselemente		2	3	6	9	2	3	0
Webseitenbeschreibung	Registrierung	-	-	●	-	-	-	●
Mehrfache Auswahl	Anrede	-	-	-	●	-	-	●
Einzeiliges Texteingabefeld	Vorname	●	●	-	●	-	-	●
Einzeiliges Texteingabefeld	Nachname	●	●	-	●	-	-	●
Spezielle Eingabefelder	E-Mail	●	●	●	●	●	●	●
Passworteingabe		●	●	●	●	●	●	●
Passworteingabe (Wdh.)		-	-	●	●	●	●	●
Link	Nutzungsbedingung	●	●	●	●	●	●	-
Link	Datenschutzlinie	●	●	●	●	●	●	-
Link	Cookies	-	●	-	●	-	-	-
...	...							

Tabelle 4.6.: Vergleich einiger Registrierungsdienste: Auswertung der Merkmale für die jeweiligen Registrierungsdienste. Die rot eingerahmten Merkmale treten bei allen Registrierungsdiensten auf.

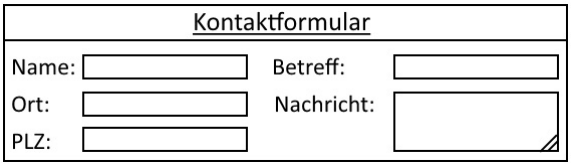


Abbildung 4.10.: Kontaktformulardienst: Kontaktaufnahme zu einer Organisation über den Kontaktformulardienst.

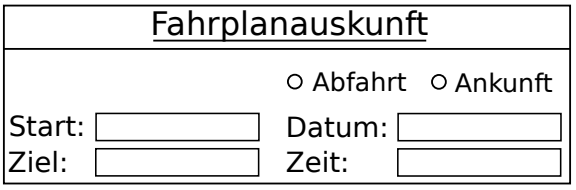


Abbildung 4.11.: Fahrplanauskunftsdienst: Abfrage des Fahrplans für den Personennahverkehr über den Fahrplanauskunftsdienst.

der Name und die E-Mailadresse abgefragt, damit auf die Mitteilung eine persönliche Antwort per Mail erfolgen kann. Häufig werden noch weitere personenbezogene Daten so wie der Betreff der Kontaktaufnahme ermittelt.

4.7.4. Fahrplanauskunft

Die Internetdienstkategorie der Fahrplanauskunft bezieht sich auf eine Reiseauskunft mit dem öffentlichen Personennahverkehr. Im Bereich des öffentlichen Personenverkehrs spielt die Fahrplanauskunft eine wichtige Rolle, denn laut der Publikation von [Sta] nutzten allein im Jahre 2014 ca. 11 Mrd. Menschen den öffentlichen Personennahverkehr. In der Abbildung 4.11 wird eine mögliche Realisierung dieser Dienstkategorie illustriert. Zur Erteilung der Fahrplanauskunft wird dabei der Start- und Zielort abgefragt sowie je nach Auswahl das gewünschte Abfahrts- oder Ankunftsdatum plus Uhrzeit ermittelt. Anhand dieser Daten wird ein Fahrplan erstellt, der dem Nutzer auflistet, wann und wo er mit welchem Verkehrsmittel fahren muss, damit die angegebenen Angaben erfüllt sind.

Beim Vergleich verschiedener als Fahrplanauskunft geclusterter Internetdienste haben sich bei ihrer Realisierung einige Merkmale wiederholt. Die Tabelle 4.8 präsentiert einen Teil der Merkmale, die aus den aufgeführten sieben Internetdiensten extrahiert wurden. Alle sieben Internetdienste haben den Reisestart, das Reiseziel, das Reisedatum und die Reiseuhrzeit abgefragt. Somit bilden diese vier Merkmale ein Indiz für diese Kategorie.

4.7.5. Flugauskunft

Der Luftverkehr hatte im Jahre 2014 ca. 208 Mio. Passagiere auf deutschen Flughäfen. Um diese Kundschaft zufrieden zu stellen, werden online verschiedenste Dienste angeboten. Beim Clustering wurden eine Vielzahl dieser Dienste zum Flugauskunftsdienst zusammengeführt.

Die Flugauskunftsdienste helfen den Kunden einen passenden Flug zu finden. In der Abbildung 4.12 ist ein möglicher Flugauskunftsdienst modelliert, mit dem der Kunde nach Flügen suchen kann. Dafür muss der Kunde folgende Angaben machen und zwar zunächst seinen Abflugort sowie sein Reiseziel benennen. Zusätzlich wird nach seinem Flugmodus gefragt, bei dem zwischen Hinflug mit oder ohne Rückflug gewählt werden kann. Abhängig von der Wahl des Flugmodus wird das Hinflug- plus das Rückflugdatum benötigt. Mit diesen Angaben kann nun ein Flug gesucht werden, jedoch müssen für die Flugbuchung noch weitere Angaben gemacht werden, die auch die Flugsuche beeinflussen können. Für

Merkmal: / **Dienstanbieter:**		berlin-airport.de/de/global/kontakt	kvv.de/service/kontakt	s-bahn-berlin.de/kontakt/kunden	berliner-volksbank.de/service/formulare	taxfreegermany.de/kontakt	fotofix.de/kontakt	diekombiloesung.de/footer/kontakt.html	stadtmobil.de/kontakt/
Merkmalsmuster	**Semantik**								
# Eingabeelemente		4	9	14	11	3	6	8	3
# Formularelemente		6	9	15	12	3	7	8	5
# Navigationselemente		0	0	0	0	0	0	0	0
Webseitenbeschreibung	Kontakt	●	-	●	-	●	●	●	●
Mehrfache Auswahl	Anrede	●	-	●	●	-	-	-	●
Einzeiliges Texteingabefeld	Vorname	●	●	●	●	-	●	●	-
Einzeiliges Texteingabefeld	Nachname	●	●	●	●	●	●	●	●
Spezielle Eingabefelder	E-Mail	●	●	●	●	●	●	●	●
Einzeiliges Texteingabefeld	Betreff	●	-	-	●	-	●	-	-
Mehrzeiliges Texteingabefeld		●	●	●	●	●	●	●	●
...	...								

Tabelle 4.7.: Vergleich einiger Kontaktformulardienste: Auswertung der Merkmale für die jeweiligen Kontaktformulardienste. Die rot eingerahmten Merkmale treten bei allen Kontaktformulardiensten auf.

Merkmal: / Dienstanbieter:		kvv.de	bahn.de/p/view	Ryanair[1]	bvg.de/de/	deutschebahn.com	s-bahn-berlin.de	mvv-muenchen.de	dvb.de/de-de/
Merkmalsmuster	**Semantik**								
# Eingabeelemente		5	4	7	4	4	4	4	4
# Formularelemente		6	14	11	6	5	5	5	5
# Navigationselemente		1	1	0	1	1	2	1	1
Webseitenbeschreibung	Reise/Fahrplan	●	●	-	-	-	●	●	●
Einzeiliges Texteingabefeld	Reisestart	●	●	●	●	●	●	●	●
Einzeiliges Texteingabefeld	Reiseziel	●	●	●	●	●	●	●	●
Spezielle Eingabefelder	Reisedatum	●	●	●	●	●	●	●	●
Spezielle Eingabefelder	Reisezeit	●	●	●	●	●	●	●	●
Mehrfache Auswahl	Abfahrt/Ankunft	●	●	-	●	●	●	●	●
Mehrfache Auswahl	Hin- +(Rückreise)	-	●	●	●	-	-	-	-
Link	Erweiterte Optionen	●	●	-	●	●	●	●	●
...	...								

[1] https://airporttransfers.ryanair.com/de

Tabelle 4.8.: Vergleich einiger Fahrplanauskunftsdienste: Auswertung der Merkmale für die jeweiligen Fahrplanauskunftsdienste. Die rot eingerahmten Merkmale treten bei allen Fahrplanauskunftsdiensten auf.

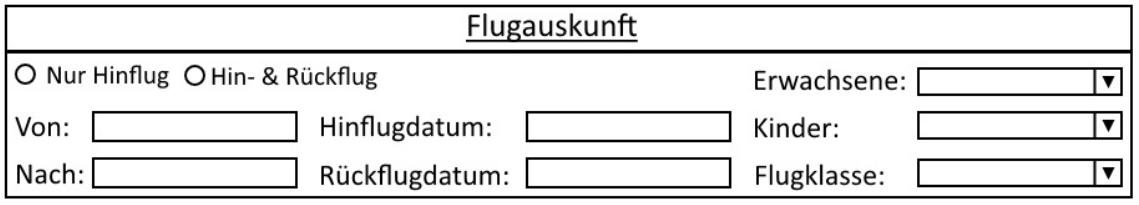


Abbildung 4.12.: Flugauskunftsdienst: Abfrage des Fahrplans für den Luftverkehr über den Flugauskunftsdienst.

die Buchung wird noch die Anzahl und der Typ an Passagieren benötigt. Mit dem Typ ist die Unterscheidung zwischen Erwachsenem, Kind und Baby gemeint. Desweiteren wird bei Flugzeugen der Komfort während des Fluges unterschieden, welche in Flugklassen wie Business, Economy und Premium eingeteilt sind. Mittels dieser Angaben listet der Dienst passende Flüge auf, von denen der Kunde einen wählen kann.

Die Tabelle 4.9 vergleicht verschiedene Internetdienste des Flugauskunftsclusters anhand ihrer extrahierten Merkmale. Die Tabelle zeigt nur einen Ausschnitt der Merkmale, die aus den sieben Diensten extrahiert wurden. Bei allen Internetdiensten wurde die Abfrage des Reisestarts und -ziels realisiert. Diese beiden Indizien teilen sich die Kategorie der Flugauskunft mit der Fahrplanauskunft. Dies spiegelt die Ähnlichkeit beider Kategorien wider, da sie beide Auskunft über unterschiedliche Reisemöglichkeiten geben. Zur Unterscheidung beider Kategorien konnten bei der Flugauskunft noch weitere Merkmale gefunden werden, die von allen Internetdiensten verwendet werden. Weitere Indizien sind das Abreise- und Rückreisedatum. Im Gegensatz zur Fahrplanauskunft wird selten nach einer Uhrzeit gefragt, da die Anzahl an Flugzeugen, die an einem bestimmten Datum vom Start- zum Zielort fliegen, sehr begrenzt ist. Weitere häufige Merkmale sind die Abfrage

Merkmal: / Dienstanbieter:		travelfusion.com/Flight	condor.com/de/	britishairways.com	germanwings.com/de.html	airberlin.com	wow-air.de/	ryanair.com	fly.de/Flugauskunft.html
Merkmalsmuster	**Semantik**								
# Eingabeelemente		4	4	3	4	4	2	4	4
# Formularelemente		11	9	11	8	9	10	5	15
# Navigationselemente		0	1	5	0	1	0	1	0
Webseitenbeschreibung	Flug	•	•	•	•	•	•	•	•
Einzeiliges Texteingabefeld	Reisestart	•	•	•	•	•	•	•	•
Einzeiliges Texteingabefeld	Reiseziel	•	•	•	•	•	•	•	•
Spezielle Eingabefelder	Abreisedatum	•	•	•	•	•	•	•	•
Spezielle Eingabefelder	Rückreisedatum	•	•	•	•	•	•	•	•
Spezielle Eingabefelder	Abreisezeit	•	-	-	-	-	-	-	-
Spezielle Eingabefelder	Rückreisezeit	•	-	-	-	-	-	-	-
Einzeiliges Texteingabefeld	Personen	•	•	•	•	•	•	-	•
Mehrfache Auswahl	Klasse	-	•	•	-	-	-	-	•
Einfache Auswahl	Nur Hinflug	-	•	•	•	•	•	•	•
Link	Erweiterte Optionen	-	•	-	-	•	-	-	-
...	...								

Tabelle 4.9.: Vergleich einiger Flugauskunftsdienste: Auswertung der Merkmale für die jeweiligen Flugauskunftsdienste. Die rot eingerahmten Merkmale treten bei allen Flugauskunftsdiensten auf.

der Personenzahl und die Auswahl zwischen Hinflug und Hinflug mit Rückflug.

4.7.6. Autovermietung

Carsharing bezeichnet das gemeinsame Nutzen von Automobilen. Die Anbieter stellen auf ihrer Online-Plattform einen Dienst zur Verfügung, über den ein Auto gebucht werden kann. Für eine Buchung gibt der Kunde die Nutzungsdauer und eine Station für die Abholung und die Rückgabe an, wobei sich diese beiden Stationen unterscheiden können. Die Abrechnung erfolgt bei Abgabe des Autos anhand der Nutzungsdauer und den gefahrenen Kilometern. Zusätzlich kann bei einigen Carsharing-Unternehmen noch die Autoklasse ausgewählt sowie einige Features dazu gebucht werden, wie beispielsweise ein Kindersitz.

Ein exemplarischer Autovermietungsdienst für das beschriebene Online-Buchungsverfahren wird in der Abbildung 4.13 dargestellt. Darin gibt der Nutzer den Abhol- und Rückgabeort und zusätzlich den genauen Zeitraum der Nutzung mit jeweils Datum und Uhrzeit an.

Beim Vergleich von acht Internetdiensten der Autovermietungskategorie in der Tabelle 4.10 traten drei Merkmale bei allen Diensten auf. Dabei handelt es sich um den Abholort, das Abhol- und Rückgabedatum. Weitere häufig auftretende Merkmale waren der Rückgabeort, die Abhol- und Rückgabezeit. Die anderen auftretenden Merkmale sind zum größten

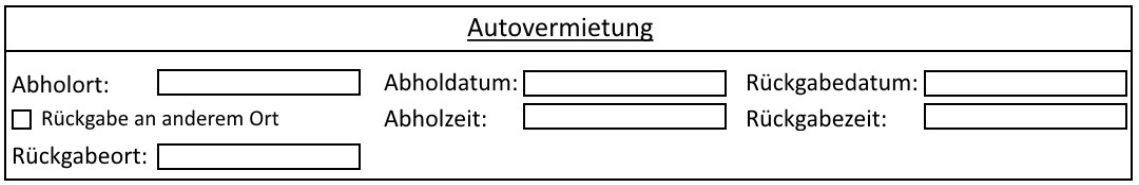


Abbildung 4.13.: Autovermietungsdienst: Suchen und Mieten von Autos über den Autovermietungsdienst.

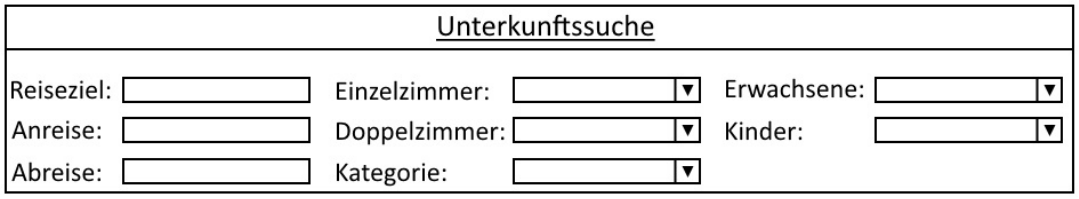


Abbildung 4.14.: Unterkunftssuchdienst: Suche nach freien Unterkünften anhand des Unterkunftssuchdiensts.

Teil nicht in der Tabelle aufgeführt, da sie jeweils nur bei wenigen Internetdiensten entdeckt wurden. Meist handelt es sich dabei um Auswahlkriterien für das Fahrzeug oder die Versicherung.

4.7.7. Unterkunftssuche

Der Unterkunftssuchdienst unterstützt den Nutzer bei der Suche einer für ihn passenden Unterkunft zu einem beliebigen Zeitraum an einem beliebigen Ort. Zusätzlich können häufig über den Dienst noch verschiedenste Kriterien gewählt werden, die den Komfort des Aufenthalts betreffen.

Die Unterkunftssuche basiert auf der Anforderung aus einer riesigen Menge an Unterkünften die passenden herauszufiltern. Ein möglicher Dienst, der diese Aufgabe übernimmt, wird in der Abbildung 4.14 präsentiert. In der Abbildung wird der Dienst aus Sicht des Nutzers dargestellt sowie die Implementierung der einzelnen Formularelemente in HTML aufgezeigt. Für eine erfolgreiche Suche muss der Nutzer zuerst einmal angeben in welchem Ort er eine Unterkunft sucht und zu welchem Zeitraum. Dadurch werden alle Unterkünfte herausgefiltert, die in der Nähe des Ortes liegen und zu diesem Zeitpunkt noch freie Zimmer haben. Weitere Angaben, die die Auswahl einschränken, sind die Anzahl der Zimmer sowie die Kategorie der Unterkunft. Zusätzlich wird für eine mögliche spätere Buchung die Anzahl der Erwachsenen und Kinder abgefragt.

Die Tabelle 4.11 zeigt einen Ausschnitt der gefundenen Merkmale bei den acht aufgelisteten Internetdiensten der Unterkunftssuchekategorie. Die große Anzahl der Formularelemente für einen Dienst weist darauf hin, dass Dienste dieser Kategorie sehr unterschiedlich implementiert werden. Außerdem entstehen sehr viele Merkmale, da verschiedenste Kriterien abgefragt werden können, wie beispielsweise ob die Unterkunft einen Pool oder WLAN haben soll. Allerdings konnte bei dieser Kategorie eine Schnittmenge von drei Merkmalen gefunden werden, die bei allen Diensten auftauchen. Sie sind in der Tabelle rot umrahmt, dabei handelt es sich um das Reiseziel, das Anreise- und Abreisedatum.

4.7.8. Newsletterabonnierung

Der Newsletter ist Vereinen, Verbänden oder anderen Organisationen ein beliebtes Mittel, um ihre Abonnenten über Neuigkeiten zu informieren. Auf die Kommunikation bezogen ist

Merkmal: / **Dienstanbieter:**		12mietwagen.de	rentalcars.com	tui.com/cars/ibe	swoodoo.com/cars	bahn.de	goeuro.de/cars/index	delmundo.de/mietwagen	buchbinder.de
Merkmalsmuster	**Semantik**								
# Eingabeelemente		4	2	3	4	6	4	3	6
# Formularelemente		16	12	12	6	6	6	5	8
# Navigationselemente		0	0	0	0	0	0	0	0
Webseitenbeschreibung	Autovermietung	●	●	●	●	-	●	●	-
Einzeiliges Texteingabefeld	Abholort	●	●	●	●	●	●	●	●
Einzeiliges Texteingabefeld	Rückgabeort	●	●	-	●	●	●	-	●
Einfache Auswahl	selber Rückgabeort	●	●	-	-	-	●	-	-
Spezielle Eingabefelder	Abholdatum	●	●	●	●	●	●	●	●
Spezielle Eingabefelder	Abholzeit	●	●	●	●	●	-	●	●
Spezielle Eingabefelder	Rückgabedatum	●	●	●	●	●	●	●	●
Spezielle Eingabefelder	Rückgabezeit	●	●	●	●	●	-	●	●
Einfache Auswahl	Nur Flughäfen	●	-	-	-	-	-	-	-
Einfache Auswahl	Fahrer: 25-70?	-	●	-	-	-	●	-	-
...									

Tabelle 4.10.: Vergleich einiger Autovermietungsdienste: Auswertung der Merkmale für die jeweiligen Autovermietungsdienste. Die rot eingerahmten Merkmale treten bei allen Autovermietungsdiensten auf.

Merkmal: / **Dienstanbieter:**		ostsee-strandurlaub.net	hotelscombined.com	travel24-hotels.de	hotel.de/de/	reiseland-brandenburg.de	hrs.com	bookinghotel.elal.com/
Merkmalsmuster	**Semantik**							
# Eingabeelemente		8	1	3	3	0	7	1
# Formularelemente		14	26	6	5	5	10	8
# Navigationselemente		0	0	0	0	1	0	0
Webseitenbeschreibung	Unterkunft	●	●	●	●	-	●	●
Einzeiliges Texteingabefeld	Reiseziel	●	●	●	●	●	●	●
Spezielle Eingabefelder	Anreisedatum	●	●	●	●	●	●	●
Spezielle Eingabefelder	Abreisedatum	●	●	●	●	●	●	●
Mehrfache Auswahl	Zimmertyp	●	-	-	-	-	-	●
Mehrfache Auswahl	Zimmeranzahl	●	●	●	●	-	●	●
Mehrfache Auswahl	Personen	-	-	●	●	●	●	●
Mehrfache Auswahl	Kinder	-	-	●	-	-	●	●
Einfache Auswahl	WLAN	-	-	-	-	-	-	●
...								

Tabelle 4.11.: Vergleich einiger Unterkunftssuchdienste: Auswertung der Merkmale für die jeweiligen Unterkunftssuchdienste. Die rot eingerahmten Merkmale treten bei allen Unterkunftssuchtdiensten auf.

Abbildung 4.15.: Newsletterabonnierungsdienst: Anmeldung beim Verteiler eines Newsletters über den Newsletterabonnierungsdienst.

somit der Newsletter das Gegenstück zum Kontaktformular, bei dem der Kunde der Organisation seine Nachricht übermittelt. Um den jeweiligen Newsletter zu erhalten, muss der Kunde sich auf dem Verteiler anmelden und den gewünschten Newsletter abonnieren. Diese Anmeldung wird häufig online über einen Dienst auf der Internetseite der Organisation angeboten.

Die Abbildung 4.15 zeigt eine mögliche HTML-Codierung eines Dienstes zur Newsletterabonnierung und die dazu passende Darstellung der Formularelemente. Im folgenden wird für die Verteilung des Newsletters die elektronische Übermittlung per E-Mail angenommen. Aufgrund der Annahme muss bei der Anmeldung die E-Mailadresse angegeben werden. Diese Information alleine würde für die Verteilung des Newsletters schon ausreichen, trotzdem wird häufig der Name des Abonnenten und dessen Anrede abgefragt, um persönliche Newsletter versenden zu können. Die Abfrage der persönlichen Daten hat zur Folge, dass ähnlich wie bei der Registrierung dem Abonnenten die Datenschutzrichtlinien über einen Link zugänglich gemacht werden müssen.

Beim Vergleich einiger Internetdienste dieser Kategorie konnten eine Vielzahl an verschiedenen Merkmalen erkannt werden. Die Tabelle 4.12 führt nur einen Teil dieser Merkmale auf. Angesichts der Notwendigkeit der E-Mailangabe ist dies das einzige Merkmal, das in allen Internetdiensten existiert. Ansonsten beziehen sich viele Merkmale auf die Auswahl verschiedener Newsletterarten und sind daher pro Internetdienst einmalig.

4.7.9. Wettervorhersage

Der Wettervorhersagedienst gibt dem Nutzer eine ortsabhängige Vorhersage des Tageswetters und des Trends für die kommenden Tage. Für die Berechnung der örtlichen Wettervorhersage ist die Angabe des Ortes oder der Postleitzahl von Nöten. Ein solches Formular wird in der Abbildung 4.16 dargestellt. Die Abfrage des Ortes oder der Postleitzahl wird dabei von demselben Texteingabefeld übernommen. Intern wird die Eingabe dann in geografische Koordinaten umgewandelt und anhand dieser Koordinaten das Wetter vorhergesagt.

Beim Vergleich einiger Wetterauskunftsdienste haben sich die in der Tabelle 4.13 aufgeführten Merkmale erkennen lassen. Viele der Dienste sind auf eigenen Internetseiten implementiert und daher wird bereits in der Internetseitenbeschreibung mit Stichworten, wie Wettervorhersage oder Wetterbericht auf diesen Dienst verwiesen. Außerdem wird bei diesen abgebildeten Diensten für die Abfrage der Position stets ein Texteingabefeld ver-

Merkmal: / **Dienstanbieter:**		kvv.de/newsletter	bahn.de	hm.com/de/newsletter	Computerwoche[1]	Hinkel360 [2]	init-ka [3]	KVB-Köln [4]
Merkmalsmuster	**Semantik**							
# Eingabeelemente		3	3	1	1	3	10	3
# Formularelemente		4	4	2	2	4	15	5
# Navigationselemente		0	1	1	0	0	0	0
Webseitenbeschreibung	Newsletter	•	-	•	•	•	•	•
Mehrfache Auswahl	Anrede	•	•	-	-	•	•	•
Einzeiliges Texteingabefeld	Vorname	•	•	-	-	•	•	•
Einzeiliges Texteingabefeld	Nachname	•	•	-	-	•	•	•
Spezielle Eingabefelder	E-Mail	•	•	•	•	•	•	•
Link	Erweiterte Optionen	-	•	-	-	-	-	-
Einfache Auswahl	Datenschutzlinie	-	-	•	-	-	-	•
...								

[1] http://www.computerwoche.de/p/newsletter,272
[2] http://www.hinkel360.de/headnav/newsletter/e-newsletter.html
[3] http://www.init-ka.de/de/kontakt/newsletter-abonnieren.php
[4] kvb-koeln.de/german/newsletter

Tabelle 4.12.: Vergleich einiger Newsletterdienste: Auswertung der Merkmale für die jeweiligen Newsletterdienste. Die rot eingerahmten Merkmale treten bei allen Diensten dieser Kategorie auf.

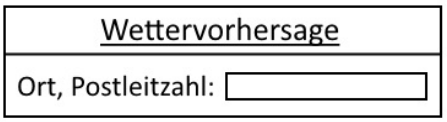


Abbildung 4.16.: Wettervorhersagedienst: Auskunft über die Wettervorhersage für einen Ort anhand des Wettervorhersagediensts.

Merkmal: / Dienstanbieter:		wetter.biz	allewetter.de	wetter24.de	wetter.info	wetter.com	wetteronline.de	t-online.de/wetter	donnerwetter.de
Merkmalsmuster	**Semantik**								
# Eingabeelemente		1	1	1	1	1	1	1	1
# Formularelemente		1	1	1	1	1	1	1	1
# Navigationselemente		0	0	0	0	0	0	0	0
Webseitenbeschreibung	Wetter	•	•	•	•	•	•	•	•
Einzeiliges Texteingabefeld	Ort/Postleitzahl	•	•	•	•	•	•	•	•

Tabelle 4.13.: Vergleich einiger Wettervorhersagedienste: Auswertung der Merkmale für die jeweiligen Wettervorhersagedienste. Die rot eingerahmten Merkmale treten bei allen Wettervorhersagediensten auf.

wendet. Durch die Einheitlichkeit der Dienste tritt jedes Merkmal stets bei allen Diensten auf.

4.8. Konstruktionsplan: Ableiten von aktiven Ontologien

Der Konstruktionsplan gibt an, wie für eine Kategorie von Internetdiensten eine aktive Ontologie abgeleitet wird. Dafür wird zuerst die Abbildung der einzelnen Formularelemente auf Elemente einer aktiven Ontologie erläutert. Darauf aufbauend wird im zweiten Schritt beschrieben, wie abstrahiert betrachtet aus einer Internetdienstkategorie eine aktive Ontologie abgeleitet wird. Zur Veranschaulichung der Ableitung werden noch zwei konkrete Beispiele präsentiert.

4.8.1. Ableiten von Sensorknoten aus Formularelementen

Die aktive Ontologie verarbeitet die Anfrage des Nutzers und führt daraufhin den angefragten Dienst aus. Dazu muss sie für einen Dienst die jeweils benötigten Informationen aus der Anfrage herausfiltern. Die benötigten Informationen werden durch die Formularelemente innerhalb des Dienstformulars abgefragt und daher muss jedes Formularelement innerhalb der aktiven Ontologie abgebildet sein. Nachfolgend werden die Abbildungen der einzelnen Formularelemente auf Konzeptknoten beschrieben. Zusätzlich gilt im allgemeinen für alle Formularelemente, wenn sie das *pattern*-Attribute verwenden, dass die folgenden Angaben nicht mehr gelten und dieses Element dann stets durch einen Sensorknoten abgebildet wird, der in der Anfrage nach Begriffen sucht, die den regulären Ausdruck des pattern-Attributes erfüllen.

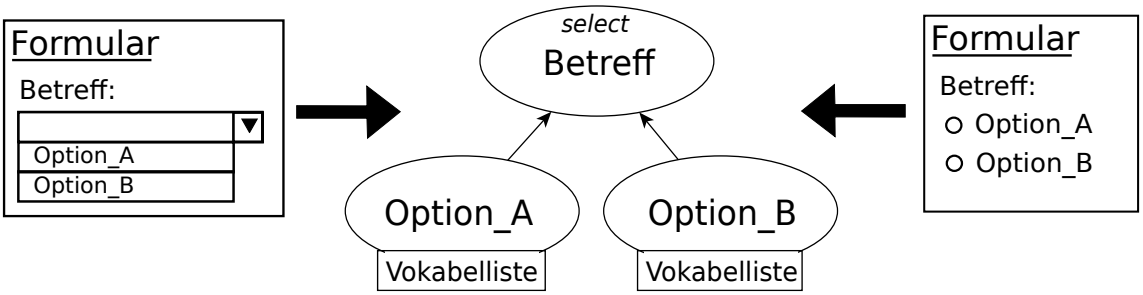


Abbildung 4.17.: Formularelement → aktive Ontologie: Transformation eines einfachen Auswahlelements in die Konzepte der aktiven Ontologie.

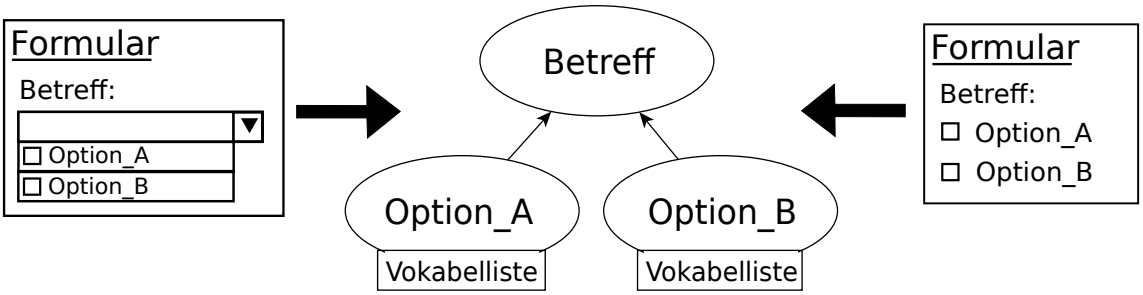


Abbildung 4.18.: Formularelement → aktive Ontologie: Transformation der Auswahlelemente in die Konzepte der aktiven Ontologie.

Einfache Auswahl

Eine einfache Auswahl kann mit verschiedenen Elementen implementiert werden, jedoch gelten gemeinsame Regeln für die Abbildung. Für jede Auswahlmöglichkeit wird ein Sensorknoten mit einer Vokabelliste erzeugt. Die Vokabelliste enthält jeweils die Auswahlmöglichkeit, um eine Wahl dieser Möglichkeit zu erkennen. Zur Ableitung der Zusammengehörigkeit und zur Einschränkung der Auswahl auf ausschließlich eine Option werden die erzeugten Sensorknoten über einen Selektionsknoten vereint. In der Abbildung 4.17 ist exemplarisch die Ableitung beider Implementierungen einer einfachen Auswahl veranschaulicht.

Mehrfache Auswahl

Die mehrfache Auswahl unterscheidet sich bei der Erkennung der Optionen nicht von der einfachen Auswahl. Einziger Unterschied ist, dass durch die Wahl mehrerer Optionen der Selektionsknoten durch einen Kombinationsknoten ersetzt werden muss. Die Abbildung 4.18 skizziert die Ableitungen der beiden Varianten, um eine mehrfache Auswahl zu implementieren.

Passworteingabe

Die Passworteingabe ist ein eingabe-sensibles Feld, deren Wert geheim gehalten werden sollte. Daher ist eine Eingabe über die natürliche Sprache nicht empfehlenswert. Sollten die Voraussetzungen für eine abhörsichere Spracheingabe des Passworts gegeben sein, dann kann das Passwort nur über einen Präfix- oder Postfix-Sensorknoten erkannt werden.

Spezielle Eingabefelder

Die speziellen Eingabefelder werden je nach Typ anders abgebildet. Nachfolgend wird die Abbildung für die einzelnen Typen beschrieben.

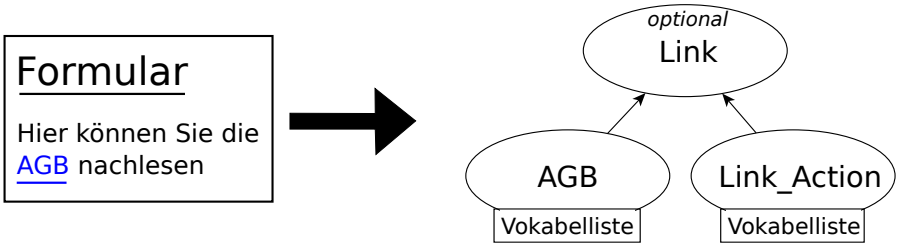


Abbildung 4.19.: Formularelement → aktive Ontologie: Transformation eines Linkelements in die Konzepte der aktiven Ontologie.

email,url,number,tel: Diese Eingabefelder akzeptieren nur Eingaben, die bestimmte reguläre Ausdrücke erfüllen. Analog wird somit für dieses Eingabefeld ein Sensorknoten erzeugt, der nach Informationen in der Anfrage sucht, die den regulären Ausdruck erfüllen.

date, datetime, datetime-local, time: Für diese Eingabefelder müssen mehrere Sensorknoten erzeugt werden, die über einen Selektionsknoten verbunden sind. Beispielsweise wird für die Erkennung der Uhrzeit ein Präfix-Sensorknoten mit dem Präfix *um*, ein Postfix-Sensorknoten mit dem Postfix *Uhr*, ein Sensorknoten, der auf den regulären Ausdruck $* : *$ reagiert und ein spezieller Sensorknoten, der Begriffe wie morgens in eine Zeit transformiert, benötigt. Sollte in der Anfrage eine Uhrzeit übermittelt werden, dann erkennt dies einer der Sensorknoten und über den Selektionsknoten wird dieser Knoten ausgewählt, wie dies in der Abbildung 4.20 skizziert ist.

color: Die Informationen dieses Eingabefelds können entweder über einen Sensorknoten mit einer Vokabelliste, die beispielsweise alle Farben enthält, oder über einen Präfix-Sensorknoten erkannt werden. Die beiden Sensorknoten werden erneut über einen Selektionsknoten vereint.

file: Dieses Eingabefeld wird durch einen Postfix-Sensorknoten abgebildet, der beispielsweise auf den Postfix „Datei “ reagiert.

search: Mit diesem Feld kann ähnlich zum Texteingabefeld jede Art von Information abgefragt werden, daher muss je nach Kontext ein Sensorknoten erzeugt werden.

Texteingabefeld

Das einzeilige und mehrzeilige Texteingabefeld sind sehr variable Felder. Jedes der speziellen Felder kann mit diesen beiden Standardfeldern ersetzt werden, daher ist die Herleitung der Semantik dieser Felder sehr schwer. Je nach Semantik muss die Information dieses Elements durch eine andere Art von Sensorknoten erkannt werden. Allerdings wird der Präfix- bzw. Postfix-Sensorknoten die häufigste Wahl zur Erkennung der Information sein.

Link

Im Allgemeinen wird ein Link des Formulars auf drei Knoten der aktiven Ontologie abgebildet. Die Abbildung 4.19 illustriert passend dazu an einem Beispiel die Ableitungsregeln. Die beiden Sensorknoten erkennen einerseits den Text bzw. den Namen des Links mittels einer Vokabelliste und andererseits die Aktion, die für den Link ausgeführt werden soll. Als Aktion könnten beispielsweise die Begriffe *öffnen* und *lesen* über eine Vokabelliste erkannt werden. Die beiden Knoten werden über einen optionalen Kombinationsknoten vereint. Dieser Kombinationsknoten ist optional, da der Link nur zusätzliche Informationen zur Verfügung stellt, die aber zur Ausführung des Formulars nicht notwendig sind.

4.8.2. Ableiten einer aktiven Ontologie anhand eines Formulars

Im vorherigen Unterkapitel wurde beschrieben wie die einzelnen Formularelemente in die Repräsentation der aktiven Ontologie abgebildet werden. In diesem Kapitel wird auf einer Abstraktionsebene höher die Abbildung eines Formulars auf eine aktive Ontologie erläutert.

Für die Abbildung eines Formulars auf eine aktive Ontologie müssen jeweils die einzelnen Formularelemente abgeleitet und direkt mit dem Wurzelknoten in Beziehung gesetzt werden. Im nächsten Schritt gilt es, die Beziehung als obligatorisch oder optional zu klassifizieren. Diese Information kann entweder in der Syntax oder Bezeichnung des Formularelements kodiert sein. Um ein obligatorisches Formularelement handelt es sich, wenn das *required*-Attribute im Formularelement spezifiziert ist oder das Label des Formularelements mit einer „verpflichtend"-Fußnote gekennzeichnet ist. Die Fußnote kann je nach Entwurf natürlich auch die optionalen Eingabefelder kennzeichnen. Zusätzlich wird häufig noch die Sprache *JavaScript* verwendet, die die einzelnen Eingabefelder validiert und anhand der Validierung kann somit auch die Unterscheidung zwischen obligatorisch und optional vorgenommen werden.

Beispiel: Transformation eines Formulars in eine aktive Ontologie

Die Abbildung 4.20 skizziert die Ableitung der einzelnen Formularelemente innerhalb des Bahnauskunftsformulars auf eine aktive Ontologie. Im Formular wird nach dem Startbahnhof, dem Zielbahnhof, dem Datum, der Uhrzeit, ob es sich um die Abfahrts- oder die Ankunftszeit handelt und der Personenanzahl gefragt. Allerdings ist die Angabe der Personen optional, was durch die Fußnote gekennzeichnet ist. Bei der Ableitung dieses Formulars wird jedes zusammengehörende Formularelement für sich abgeleitet und das Resultat mit dem Wurzelknoten verbunden.

Die beiden Elemente zur Abfrage des Start- und Zielbahnhofs sind einzeilige Texteingabefelder. Aufgrund ihrer Semantik kann gefolgert werden, dass sie nur Bahnhöfe einer Bahnhofsliste annehmen. Zur Erkennung des Datums und der Uhrzeit werden diese Elemente auf mehrere Sensorknoten abgebildet, da diese Daten jeweils in verschiedener Formatierung übermittelt werden können. Die einzelnen Sensorknoten werden mit einem Selektionsknoten vereint, da nur eine Formatierung auftreten kann. Die Auswahl zwischen Abfahrts- und Ankunftszeit kann über Sensorknoten mit den jeweiligen Begriffen in der Vokabelliste erkannt werden. Da dies eine sich gegenseitig ausschließende Auswahl ist, werden die beiden Sensorknoten über einen Selektionsknoten vereint. Für die Ableitung der letzten Angabe wird für jede mögliche Anzahl an Personen ein Sensorknoten erzeugt, der auf die jeweilige Zahl reagiert.

An diesem Beispiel wird deutlich, dass durch häufig auftretende *1-zu-n* Beziehungen, die Ableitung selbst für kleine Formulare komplex werden kann. Zudem ist die Ableitung der einzelnen Elemente sehr von der Semantik der Elemente abhängig und diese ist bei vielen Implementierungen nicht eindeutig.

4.8.3. Ableiten einer aktiven Ontologie anhand mehrerer Formulare einer Kategorie

Die Ableitung einer aktiven Ontologie für ein Formular wird auf mehrere Formulare erweitert. Dafür müssen die Formulare miteinander verglichen werden, um für die Menge der Formularelemente mit derselben Semantik eine Ableitung auf die aktive Ontologie zu erzeugen. Aus einer anderen Sicht betrachtet werden durch diese Vorgehensweise alle Formulare zu einem großen Formular vereint, wobei doppelte Formularelemente, die nach

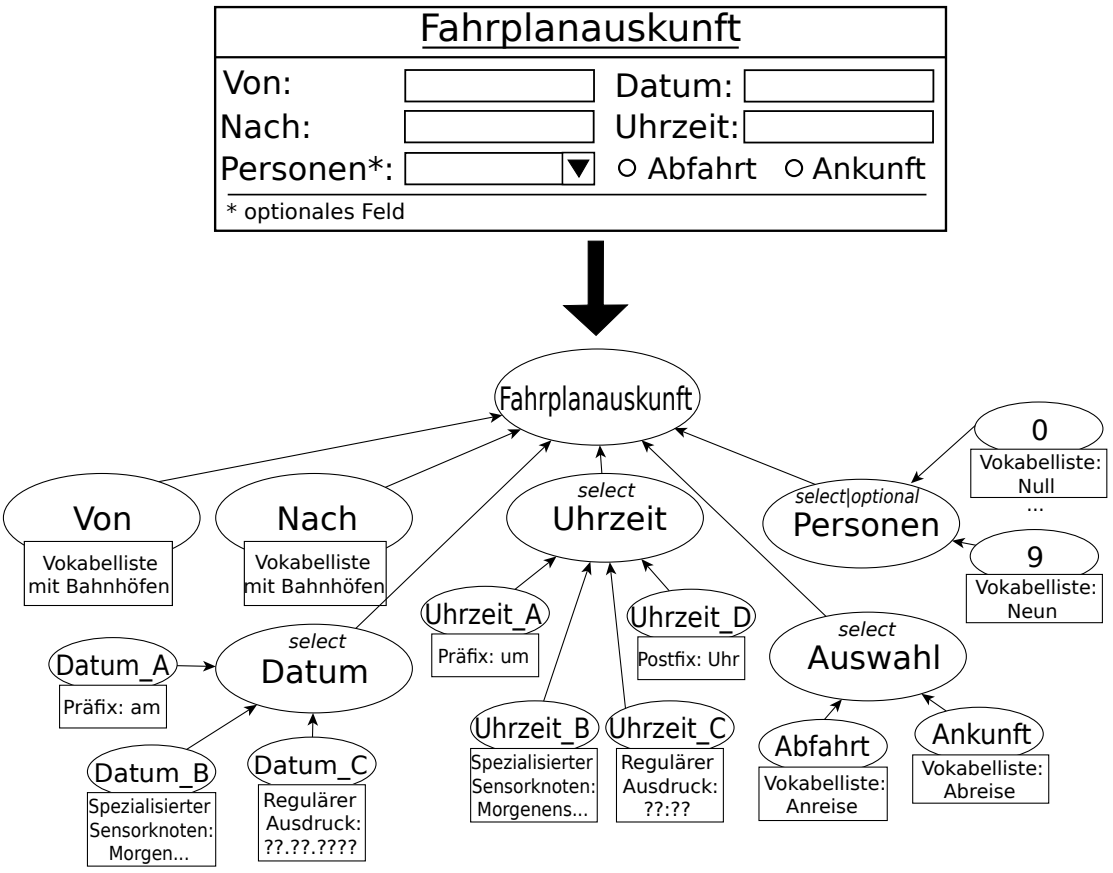


Abbildung 4.20.: Formular → aktive Ontologie: Ableitung eines Formulars zu einer aktiven Ontologie.

derselben Information fragen, entfernt werden. Daraufhin werden die einzelnen verbleibenden Formularelemente abgeleitet.

Das Verfahren für die Bestimmung der Beziehung zwischen dem Wurzelknoten und den abgeleiteten Knoten muss nun auch von einem auf mehrere Formulare erweitert werden. Dazu wird jeweils die Menge der semantisch gleichen Formularelemente betrachtet. Befindet sich in dieser Menge ein Formularelement von jedem Formular der Kategorie und sind zusätzlich alle diese Formularelemente innerhalb ihres Formulars obligatorisch, dann ist auch die Beziehung zum Wurzelknoten obligatorisch. In allen anderen Fällen handelt es sich um eine optionale Beziehung.

Beispiel: Transformation der Formulare einer Kategorie zu einer aktiven Ontologie

In der Abbildung 4.21 sind drei verschiedene Formulare der selben Kategorie abgebildet. Durch einen Abgleich der drei Formulare wird ersichtlich, dass trotz unterschiedlichem Label die Formulare A und B jeweils nach einem Ort, der Ankunfts- und Abfahrtszeit fragen. Das Formular hingegen benötigt nur einen Ort und eine Ankunftszeit. Aus dem Vergleich aller Formulare ergibt sich, dass der Ort und die Ankunftszeit von allen und im Gegensatz dazu die Abreisezeit nur von zwei Formularen benötigt werden. Somit werden Ort und Ankunftszeit gemäß der zuvor hergeleiteten Regel obligatorische Knoten der aktiven Ontologie und die Ankunftszeit ein optionaler Knoten.

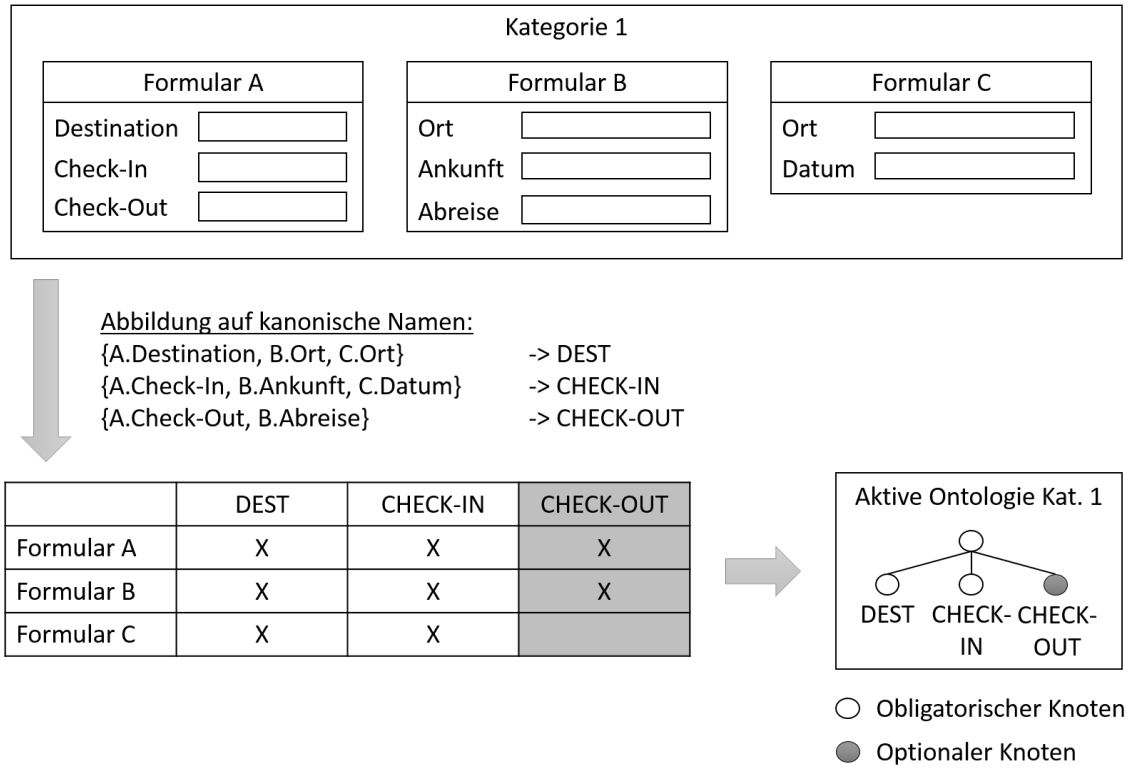


Abbildung 4.21.: Kategorie → aktive Ontologie: Ableitung einer aktiven Ontologie aus einer Kategorie mit drei Formularen. Über den Vergleich der Formulare wird entschieden welcher Knoten der abgeleiteten Formularelemente in der aktiven Ontologie obligatorisch bzw. optional ist.

4.8.4. Ableiten einer aktiven Ontologie für eine Internetdienstkategorie

Die Ableitung einer aktiven Ontologie für eine Internetdienstkategorie ist die konkrete Anwendung des im Unterabschnitt 4.8.3 beschriebenen Verfahrens. Der erste Schritt wäre somit, die einzelnen Formulare innerhalb einer Kategorie miteinander zu vergleichen und eine Menge an semantisch unterschiedlichen Formularelementen zu gewinnen. Dieser Vergleich wurde bei der Analyse der einzelnen Kategorien bereits vorgenommen, um die Merkmale einer Kategorie herauszuarbeiten. Bei der Reiseauskunftskategorie handelt es sich dabei beispielsweise um die Tabelle 4.8. Allerdings dürfen in der Abbildung nur die Merkmale betrachtet werden, die sich auf ein Formularelement beziehen. Dies führt dazu, dass die Merkmalsmuster für die Navigationselemente, die Internetseitenbeschreibung und die Anzahl an Formular- und Navigationselementen nicht betrachtet werden.

Im zweiten Schritt wird jedes der semantisch unterschiedlichen Formularelemente abgeleitet. Zur Vervollständigung der Ableitung muss die Beziehung zum Wurzelknoten bestimmt werden. Diese Information kann jedoch wiederum aus der Tabelle 4.8 entnommen werden, denn bei der Analyse der Internetdienstkategorien wurden die Merkmale herausgefiltert, die in jedem Formular auftraten. Diese Merkmale bilden nun eine obligatorische Beziehung zum Wurzelknoten und die anderen eine optionale. Somit kann anhand den Analysewerten der Internetdienstkategorien und Ableitungsregeln aus einer Internetdienstkategorie eine aktive Ontologie entworfen werden.

4.9. Zusammenfassung

In der Analyse lag zu Beginn der Fokus auf dem Finden von Internetdienstkategorien. Dazu wurde über einen Webcrawler eine Internetdienstsammlung aufgebaut. Anhand dieser

Internetdienste wurden elf Merkmalsmuster abgeleitet, die sich auf die Semantik der Formularelemente, die Webseitenbeschreibung oder die Häufigkeit bestimmter Elemente beziehen. Zur Bestimmung der Semantik wurden aus bestimmten Attributwerten der jeweiligen Elemente die Nomen extrahiert und in einer Liste gespeichert. Anschließend wurden aufgrund der Varianz zwischen verschiedenen Implementierungen von Internetdiensten Regeln für die Vergleichbarkeit zweier Merkmale bestimmt. Dies führte dazu, dass vergleichbare Merkmale bei einer Semantikübereinstimmung vereint wurden. Der nächste Schritt zur Findung von Internetdienstkategorien war die Selektion der Merkmale. Dabei war das Ziel irrelevante und redundante Merkmale zu entfernen. Schlussendlich konnten neun Internetdienstkategorien abgeleitet werden. Für jeden dieser Dienste wurde die Funktionalität beschrieben, das Erscheinungsbild skizziert sowie mögliche auftretende obligatorische und optionale Merkmale diskutiert.

Neben der Kategorisierung der Internetdienste und der gewonnenen Information über die Merkmale der jeweiligen Internetdienstkategorien musste noch ein Plan zur Abbildung der Nutzeranfrage auf die Formularelemente entworfen werden. Dafür bildet die aktive Ontologie die Schnittstelle zur Erkennung der natürlichen Sprache. Somit wurden Regeln für das Ableiten der einzelnen Formularelemente, eines Internetdienstes und einer ganzen Internetdienstkategorie erstellt.

5. Entwurf und Implementierung

In diesem Kapitel werden der Entwurf und die Implementierung des Webcrawlers und des Clusterers erläutert. Beide Programme wurden bei der Analyse zur Erfüllung der Zielsetzung dieser Arbeit verwendet. Der Webcrawler hat die Aufgabe, Internetdienste im Internet zu finden und der Clusterer soll neu gefundene Internetdienste klassifizieren.

5.1. Entwurf des Webcrawlers

Der Webcrawler hat die Aufgabe, anhand der Webseiten im Internet Internetdienste zu finden. Probleme bei der Umsetzung entstehen durch zyklische Verlinkungen der einzelnen Dokumente. Um für eine URL entscheiden zu können, ob sie schon einmal gefunden wurde, müssen alle bereits gefundenen URLs gespeichert werden. Diese Speicherung führt dazu, dass der Webcrawler aufgrund der riesigen Speichermenge nicht mehr skaliert. Ein weiteres Problem ist die Erkennung identischer Dokumente von unterschiedlichen URLs. Denn der Abgleich und die Speicherung aller Dokumente beeinträchtigt ebenfalls die Skalierbarkeit und Geschwindigkeit. Daher werden im Kontext der konkreten Aufgabenstellung die folgenden Annahmen getroffen, die den Entwurf und die Implementierung vereinfachen.

Annahme 1: Ein Internetdienst ist von der Startseite aus nach spätestens einem Link erreichbar.

Annahme 2: Der selbe Internetdienst kann nur innerhalb des selben Hosts vorkommen.

Die erste Annahme beruht auf der Tatsache, dass sich bei einer seriös strukturierten Webseite ein Großteil der Internetdienste entweder sofort auf der Startseite oder eine hierarchische Ebene darunter befindet. Der Begriff hierarchische Ebene bezieht sich auf einen gerichteten Linkgraph, der im Unterkapitel 2.3.6 beschrieben wurde. Durch diese Annahmen können einige Probleme des Webcrawlers gelöst werden.

5.1.1. Optimierung der Skalierbarkeit

Die Grundidee zur Verbesserung der Skalierbarkeit besteht darin, nicht jede URL zu speichern sondern jeden gefundenen Host. Somit entsteht der Host-gesehen-Test, der Zyklen auf Hostebene unterdrückt. Diese Zyklenfreiheit führt dazu, dass jeder Host nur einmal untersucht wird. Daher muss bei der Umsetzung sichergestellt werden, dass der aktuelle Host vollständig analysiert wurde bevor der nächste Host betrachtet wird.

Die vollständige Untersuchung eines Hosts kann im allgemeinen nicht erreicht werden, weshalb jede URL gespeichert wird. Allerdings kann in diesem Fall aufgrund der ersten Annahme der Host auf eine wohldefinierte Menge an URLs eingeschränkt werden. Diese Menge wird vollständig untersucht, indem von der Startseite beginnend die Dokumente analysiert werden, auf die die Startseite verweist. Damit in diesem Schritt eine URL nicht zweimal untersucht wird, werden alle bereits gefundenen Links der Startseite gespeichert und Duplikate entfernt. Dies wird im folgenden als URL-gesehen-Test bezeichnet. Außerdem werden auch auf den Seiten, die von der Startseite verlinkt sind, die Links extrahiert, aber nur um URLs mit neuen Hosts zu finden.

Somit entsteht ein Host-Speicher, der alle gefundenen Hosts und ein URL-Speicher, der temporär für den aktuell untersuchten Host alle darin gefundenen URLs puffert. Daraus kann folgender Ablauf abgeleitet werden. Zuerst wird ein Host aus dem Host-Speicher ausgelesen und auf dessen Startseite nach Internetdiensten gesucht. Anschließend werden die Links der Startseite extrahiert. Die gefundenen Links werden entweder in den URL-Speicher geschrieben, wenn es sich um interne Links handelt oder andernfalls dem Host-Speicher zugewiesen. Als intern wird ein Link bezeichnet, der auf ein Dokument innerhalb des selben Hosts verweist. Daraufhin wird iterativ der URL-Speicher ausgelesen und jeweils nach neuen Internetdiensten und Hostadressen gesucht. Sobald der URL-Speicher geleert ist, wird der nächste Host aus dem Host-Speicher geladen und der Ablauf beginnt von vorne.

5.1.2. Erkennung von Internetdienstduplikaten

Neben der Erkennung von Zyklen im Linkgraph sollte der Webcrawler duplizierte Internetdienste abfangen. Diese können entstehen, wenn beispielsweise ein Logindienst ins Menü integriert ist und dieses Menü innerhalb des Hosts für jedes Dokument verwendet wird. Zur Erkennung eines duplizierten Internetdienstes werden die Formularelemente auf Äquivalenz verglichen. Bei Übereinstimmung der Implementierung der Elemente wird von einem Internetdienstduplikat ausgegangen und dieser Internetdienst nicht ausgegeben.

Für die Überprüfung müssen alle Formularelemente als String gespeichert werden. Durch die zweite Annahme wird dies umgangen, indem die Strings nur solange gespeichert werden bis der aktuelle Host als vollständig betrachtet angenommen wird.

5.1.3. Einschränkungen der URL

Durch die Suche von Internetdiensten kann die URL auf die ausschließliche Verwendung von *HTTP*- oder *HTTPS*-Protokoll eingeschränkt werden. Dies hat den Hintergrund, dass andere Protokolle, wie beispielsweise *FTP*, dem Nutzer keine Internetdienste auf Grundlage eines HTML-Formulars anbieten. Eine weitere Einschränkung der URL beruht auf dem Ansatz, ausschließlich englisch- oder deutschsprachige Webseiten anhand der Top-Level-Domäne zu filtern. Der Webcrawler soll somit nur URLs der Top-Level-Domäne *.de*, *.com*, *.org*, *.net* oder *.edu* untersuchen. Diese Einschränkung wird aufgrund der Semantikerkennung beim Clusterer vorgenommen. Jedoch könnte durch den Einsatz von Übersetzungswerkzeugen der Clusterer auf die Erkennung jeder Sprache erweitert werden, wodurch die Einschränkung hinfällig wird.

5.1.4. Modulaufbau

Vor der Extraktion der Informationen auf einer Webseite muss diese vom Webcrawler erstmals ausgewählt und analysiert werden. Der grobe Entwurf des Webcrawlers ist in der Abbildung 5.1 illustriert. Das *Seed*-Modul lädt in das *Frontier*-Modul die initialen Webseiten, bei denen der Webcrawler seine Suche beginnt. Daraufhin wird im *Frontier*-Modul

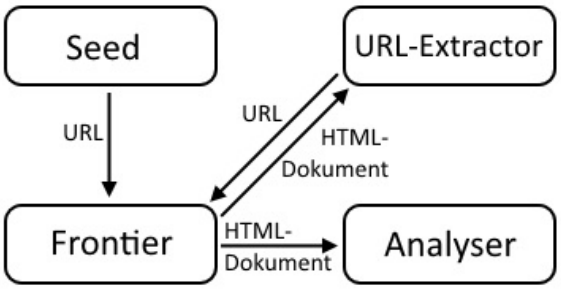


Abbildung 5.1.: Modulübersicht des Webcrawlers: Diagramm mit allen Modulen des Webcrawlers und ihre Abhängigkeiten voneinander.

eine URL ausgewählt und das Dokument der URL heruntergeladen. Dieses Dokument wird sowohl an das *Parser*-Modul als auch an das *Analyser*-Modul übergeben. Im *Parser*-Modul werden alle im Dokument implementierten HTML-Links gefiltert und die jeweiligen URLs dem *Frontier*-Modul übergeben und von diesem gegebenenfalls gespeichert. Das *Analyser*-Modul erhält ebenfalls das heruntergeladene Dokument und sucht darin nun nach Internetdiensten. Diese Schritte werden iterativ mit der Wahl der nächsten URL im *Frontier*-Modul fortgeführt. Im folgenden wird der Aufbau der einzelnen Module erläutert.

Seed-Modul

Das Seed-Modul wird ausschließlich beim Start des Werbcrawlers aufgerufen. Es hat die Aufgabe den leeren URL-Speicher mit initialen URLs zu füllen. Dazu wird eine Liste von URLs dem *Frontier*-Modul übergeben.

Frontier-Modul

Das Frontier-Modul ist für die Speicherung der URLs sowie für die Wahl der nächsten URL zuständig. Der Ablauf beim Aufruf dieser beiden Methoden ist in der Abbildung 5.2 skizziert.

Für das Speichern einer URL muss der Methode die zu speichernde URL und das Dokument übergeben werden, von dem es extrahiert wurde. Nun werden die einzelnen Fallunterscheidungen getätigt. Dazu wird zuerst geprüft, ob die neue URL vom selben Host ist wie ihre Elterndokumente. Bei Gleichheit wird nun noch geprüft ob, die URL von der Startseite extrahiert wurde und falls ja, ob sie bereits schon einmal gespeichert wurde. Diese beiden Abfragen sind notwendig, um einerseits den Linkgraph von der Startseite nur höchstens bis zur ersten Abstraktionstiefe abzusuchen und andererseits um zyklischen Verlinkungen entgegen zu wirken. Sollten die Hosts der neuen URL und des Elterndokuments ungleich sein, dann ist die Abstraktionstiefe uninteressant und es wird nur noch ausschließlich geschaut, ob der Host bereits gefunden wurde. Je nach Fallunterscheidung wird die neue URL verworfen oder entweder im URL-Speicher oder Host-Speicher gepuffert.

Beim Laden der nächsten URL wird stets der Speicher mit neuen URLs ausgelesen. Solte dieser leer sein, dann wird der Host-Speicher angesprochen. Die erhaltene URL wird daraufhin heruntergeladen und das Dokument zurückgeliefert.

Dieses Modul enthält noch zwei weitere Klassen, die für die Verwaltung der beiden Speicher und Auswertung des Host- bzw. URL-gesehen-Tests zuständig sind.

URL-Extractor

Das *URL-Extractor*-Modul erhält vom *Frontier*-Modul ein Dokument und sucht darin nach allen Links. Aus jedem Link wird die absolute URL extrahiert und diese bei Erfüllung

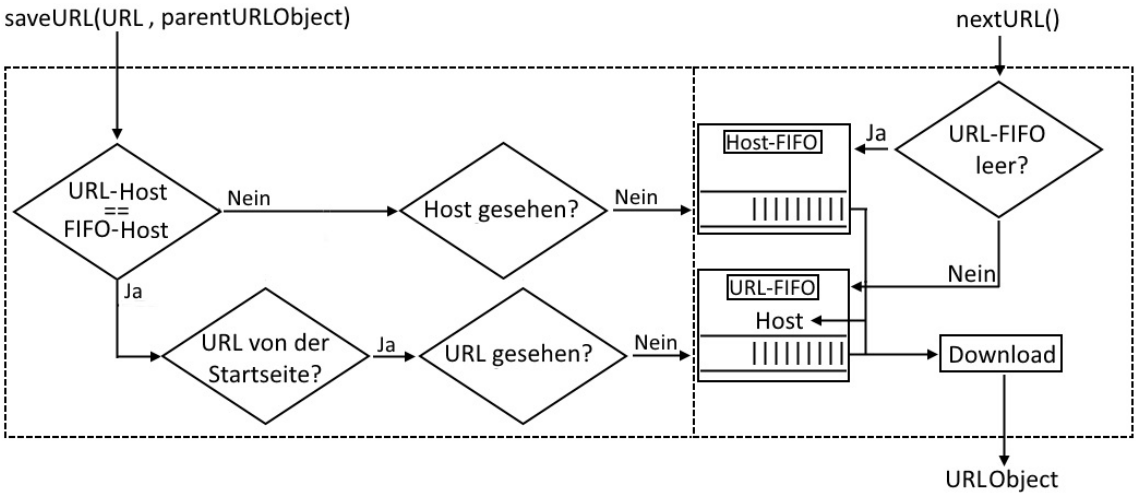


Abbildung 5.2.: *Frontier*-Modul: Ablaufdiagramm für die beiden Methoden zur Speicherung einer URL sowie zur Wahl der nächsten URL.

der URL-Einschränkungen (Unterabschnitt 5.1.3) zum Speichern an das *Frontier*-Modul übergeben.

Analyser

In diesem Modul wird innerhalb des überlieferten Dokuments nach Internetdiensten gesucht. Bei erfolgreicher Suche wird das HTML-Formular zusammen mit der Webseitenbeschreibung und der URL des Dokuments ausgegeben.

5.2. Implementierung des Webcrawlers

Für die Implementierung des Webcrawlers ist die Programmiersprache Python[1] verwendet worden. Ausschlaggebend dafür war die Größe des Projekts, der flexible Umgang mit Zeichenketten und die Effizienz beim Vergleich von Strings. Zusätzlich kann über die Bibliotheken urlparse[2], urllib[3] und BeautifulSoup[4] einfach die URL transformiert sowie innerhalb des HTML-Dokuments navigiert werden.

5.2.1. URL- und Host-gesehen-Test

Die beiden URL- und Host-gesehen-Tests werden ähnlich implementiert. Im folgenden wird exemplarisch der URL-gesehen-Test betrachtet. Die Buchstaben der URL, für die der Test ausgeführt wird, müssen in einem Vorverarbeitungsschritt in Kleinbuchstaben transformiert werden. Zusätzlich werden alle Sonderzeichen am Ende des Strings entfernt, denn fügt man einer URL am Ende des Strings einen Slash hinzu, dann handelt es sich weiterhin um die selbe URL. Danach wird diese URL mit allen anderen bereits gesehenen und gespeicherten URLs verglichen.

5.2.2. Speicher

Beim Entwurf des Webcrawlers konnten zwei verschiedene Speicherarten abgeleitet werden. Zum einen die zwei Speicher für die nächste URL und zum anderen zwei Speicher,

[1] *Python 2.7.9*. Zugriff: 2015.04.24. URL: `https://www.python.org/`
[2] *urlparse*. Zugriff: 2015.04.24. URL: `https://docs.python.org/2/library/urlparse.html`
[3] *urllib*. Zugriff: 2015.04.24. URL: `https://docs.python.org/2/library/urllib.html`
[4] *Beautiful Soup 4*. Zugriff: 2015.04.24. URL: `http://www.crummy.com/software/BeautifulSoup/`

```
<internetdienst>
<alink>URL DER WEBSEITE</alink>
<headcontent><head>WEBSEITENBESCHREIBUNG</head></headcontent>
<form> FORMULAR </form>
</internetdienst>
```

Abbildung 5.3.: Format der Ausgabe eines Internetdienstes

die jeweils für die Prüfung des Host- bzw. URL-gesehen-Tests benötigt werden. Bei den Speichern für die nächste URL handelt es sich um eine Warteschlange (engl. *first in first out queue*), die in der Reihenfolge des Eintreffens die URLs wieder ausgibt. Diese Art von Speicher wird verwendet, da innerhalb des Speichers keine Priorisierung der URLs nötig ist, da dies bereits durch die Speicherzuteilung in Abhängigkeit vom aktuellen Host getätigt wurde. Für die gesehen-Tests werden Hashtabellen verwendet, weil bei dieser Datenstruktur in konstanter Zeit überprüft werden kann, ob eine URL bereits gespeichert wurde.

5.2.3. Ausgabe

Für die Ausgabe des Webcrawlers wird eine externe Datei *output.xml* erzeugt. In diese Datei werden alle gefundenen Internetdienste eingeschrieben. Für die Speicherung eines einzelnen Internetdienstes wird das Format der Abbildung 5.3 verwendet. Die Ausgabe eines Internetdienstes enthält den URL der Webseite, die Webseitenbeschreibung und das Formular. Sollten auf einer Webseite mehrere Internetdienste angeboten werden, unterscheidet sich die Ausgabe nur anhand des Formulars.

5.3. Entwurf des Clusterers

Diese Software hat die Aufgabe, neue Internetdienste zu klassifizieren. Zur Umsetzung wird der Ansatz verfolgt, ein maschinell lernendes System zu entwerfen. Dieses System generiert Wissen aus seinen Erfahrungen. Abgeleitet auf den Entwurf des Systems bedeutet dies, dass der Clusterer anhand bereits bekannter Klassifikationen (=Erfahrungen) die neuen Internetdienste klassifizieren soll. Für ein maschinell lernendes System werden zwei Eingabemengen von Internetdiensten benötigt. Zum einen die Trainingsmenge, die den Internetdienst und dessen Klassifikation enthält und zum anderen die Testmenge, deren Internetdienste klassifiziert werden sollen.

Die Abbildung 5.4 skizziert den Ablaufplan des Clusterers. Im ersten Schritt werden anhand der Trainingsmenge für jeden Internetdienst die Merkmale erkannt und erzeugt. Dies wird genauso für die Testmenge getan. Anschließend werden die erzeugten Merkmale der Trainingsmenge vereinigt und selektiert. Im nächsten Schritt möchte man nun für jeden Internetdienst der Testmenge bestimmen, welches der selektierten Merkmale bei ihm auftritt. Dafür werden dieses mal für die Testmenge Merkmale erkannt und erzeugt. Es wird nun versucht die erzeugten Merkmale einem selektierten Merkmal zuzuordnen. Sollte dies für ein Merkmal nicht möglich sein, wird es verworfen. Durch diesen Schritt bleiben die selektierten Merkmale übrig, jedoch ist sowohl für die Trainings- als auch Testmenge bekannt, welche dieser Merkmale ein Internetdienst enthält. Auf dieser Grundlage wird die Trainings- und Testmenge geclustert und anhand der Klassifikation der Trainingsmenge, die einzelnen Cluster einer Internetdienstkategorie zugeordnet. Zum Schluss werden die Klassifikationen für die einzelnen Internetdienste der Testmenge ausgegeben.

Im folgenden wird zuerst der Grobentwurf mittels der grafischen Benutzeroberfläche und der Paketstruktur dargestellt und anschließend der Feinentwurf der einzelnen Verfahren erläutert.

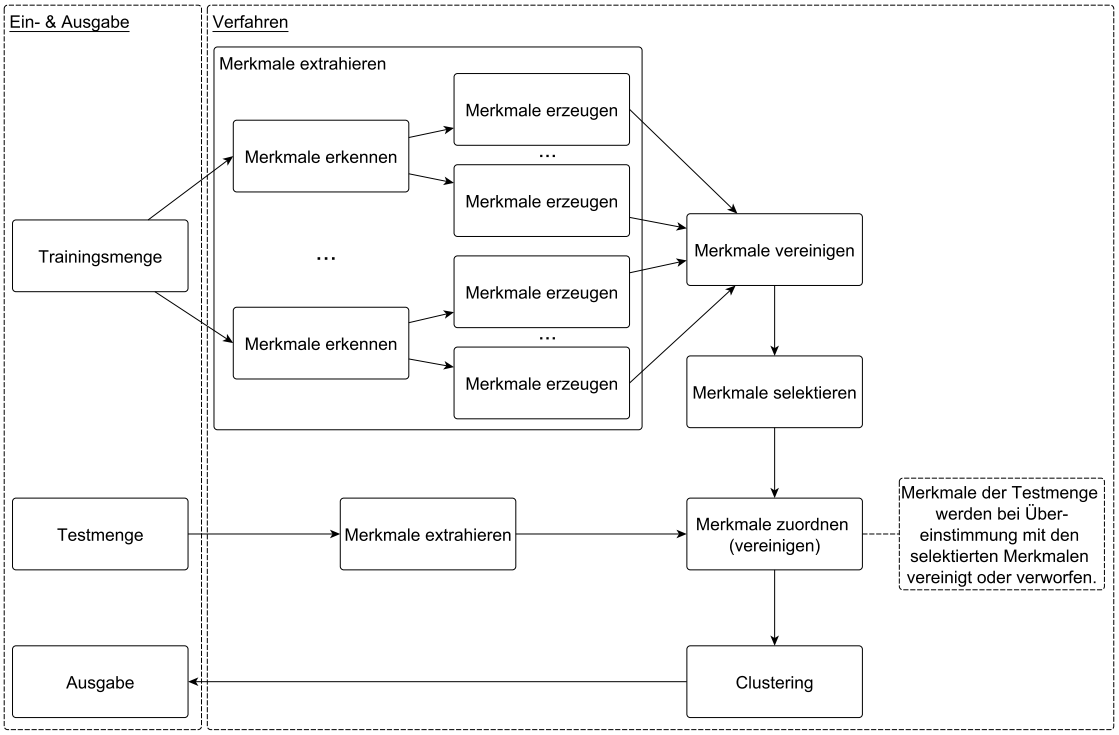


Abbildung 5.4.: Ablaufplan des Clusterers: Darstellung der Abhängigkeiten zwischen den einzelnen Verfahren sowie der Ein- und Ausgabe.

5.3.1. Grafische Benutzeroberfläche

Die grafische Benutzeroberfläche des Clusterers ist in der Abbildung 5.5 dargestellt. Der Aufbau der Oberfläche entspricht fünf unterschiedlichen Bereichen. Im ersten Bereich werden die Dateien für die Ein- und Ausgaben spezifiziert. Über einen Laden- bzw. Speichern-Dialog werden die Dateien für die Trainings-, Testmenge und das Resultat bestimmt. Der zweite Bereich stellt eine Liste an Internetdiensten dar, aus der ein Internetdienst gewählt werden kann. Für den gewählten Internetdienst wird daraufhin im vierten Bereich dessen Formular und im fünften Bereich dessen Merkmale und Klassifikation, jeweils falls bereits berechnet, präsentiert. Im dritten Bereich kann der Nutzer entscheiden welche der beiden Mengen in der Liste des zweiten Bereiches angezeigt werden sollen. Zusätzlich enthält dieser Bereich noch drei Schaltflächen, mit denen der Nutzer den Ablauf der Abbildung 5.4 steuern kann. Über den *Load Sets*-Button werden die Trainings- und Testmenge aus den angegebenen Dateien geladen und in der Liste des zweiten Bereiches dargestellt. Mit dem *Extract Features*-Button werden aus den geladenen Mengen gemäß der Abbildung die Merkmale extrahiert, vereinigt und selektiert. Diese resultierenden Merkmale werden daraufhin im fünften Bereich dargestellt. Neben den Merkmalen werden in diesem Bereich auch noch die Clusterergebnisse durch die beiden Textfelder *Cluster(correct)* und *Cluster(predict)* dargestellt. Das erste Feld gibt -falls vorhanden- die manuell zugeordnete und das zweite Feld die vom Clusteringalgorithmus klassifizierte Internetdienstkategorie an. Damit der Clusteringalgorithmus ausgeführt und somit eine Internetdienstkategorie klassifiziert wird, muss zuerst der *Cluster Webservices*-Button aufgerufen werden. Dieser Button führt anhand der Merkmale das Clustering durch und gibt die Klassifikationen aus.

5.3.2. Paketstruktur

Die Paketstruktur des Clusterers wird in der Abbildung 5.6 modelliert. Es spiegelt die Abhängigkeiten zwischen den einzelnen Paketen wider. Das *Controller*-Paket ist die Zentrale,

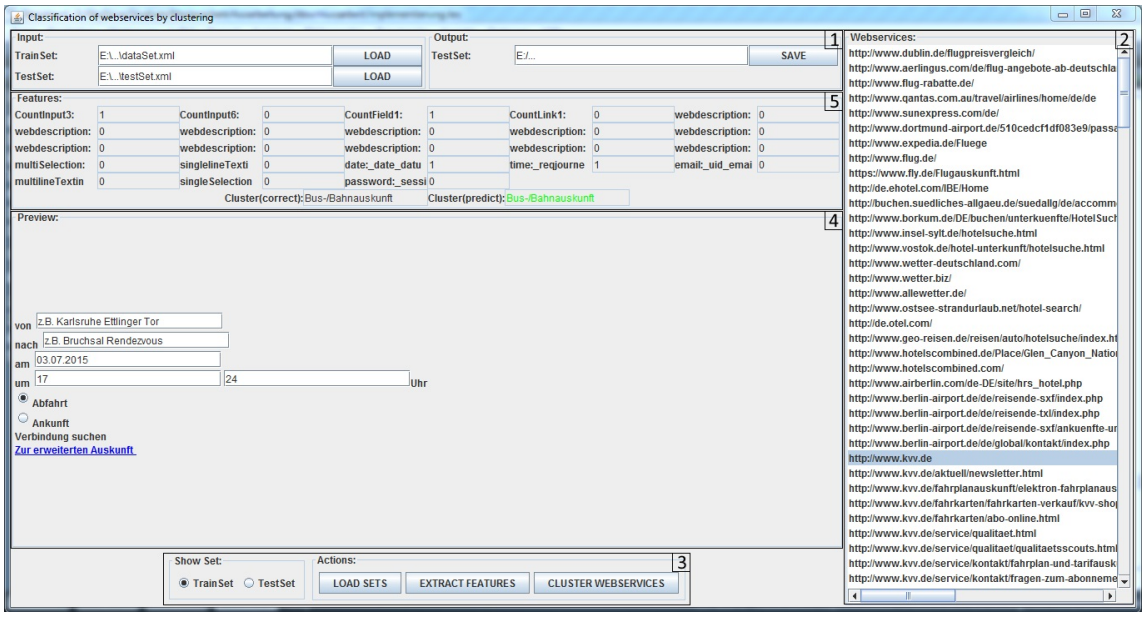


Abbildung 5.5.: Grafische Benutzeroberfläche des Clusterers: Unterteilung der GUI in fünf unterschiedliche Bereiche.

die die Nutzereingaben abarbeitet und den Ablauf steuert. Basierend auf dem *Model-View-Controller*-Muster führt das *Controller*-Paket eine Aufgabe aus und speichert dies im *Model*-Paket. Über diese Änderung im Model-Paket wird das *View*-Paket mittels des Beobachtermusters benachrichtigt und passt daraufhin die Darstellung auf die Änderung an. Die beiden anderen Abhängigkeiten vom *Controller*-Paket zum *Webservice*-Paket bzw. *FeatureContainer*-Paket sind Aufrufe zur Erfüllung einer Aufgabe. Beispielsweise beim Laden der Internetdienste greift ein Controller auf das *Webservice*-Paket zu und erzeugt für jeden Internetdienst eine Instanz einer *Webservice*-Klasse. Die erzeugten Instanzen werden daraufhin im *Model*-Paket gespeichert und folglich in der *View* dargestellt. Der Aufruf des *FeatureContainer*-Pakets bildet den Ablauf für die Erzeugung, Vereinigung und Selektion der Merkmale ab. Dafür greift das *FeatureContainer*-Paket auf die Implementierungen der Merkmalsmuster im *FeaturePattern*-Paket zu. Das *FeaturePattern*-Paket nimmt wiederum die Funktionalität des *Util*-Pakets wahr, um die Semantik der Elemente zu erkennen.

Controller-Paket

Das *Controller*-Paket steuert die Aktionen des Nutzers. Über die Klassen des *controller.buttonlistener* Unterpakets wird der Aufruf der Schaltflächen im dritten Bereich der grafischen Benutzeroberfläche erkannt. Je nach Button wird daraufhin einer der Controller *CLoad*, *CExtract* oder *CClustering* aufgerufen. Diese Controller führen den Algorithmus für die jeweilige Aufgabe aus, indem sie beispielsweise Methoden anderer Klassen aufrufen.

Zusätzlich findet sich in diesem Paket noch der Controller *CMain*, der die *main*-Methode implementiert, die beim Start des Programms aufgerufen wird. Somit werden in dieser Methode die initialen Instanziierungen des Systems vorgenommen.

View-Paket

Das *View*-Paket in der Abbildung 5.8 ist für die grafische Darstellung zuständig. Das Paket enthält eine Oberklasse *View* und fünf Unterklassen, die von der *View*-Klasse erzeugt werden. Jede dieser fünf Klassen entspricht einem der fünf Bereiche, die im Unterabschnitt 5.3.1 für die grafische Oberfläche beschrieben wurden. Die *View*-Klasse enthält einen Konstruktor, der die grafische Oberfläche mit all den Bereichen erzeugt und eine

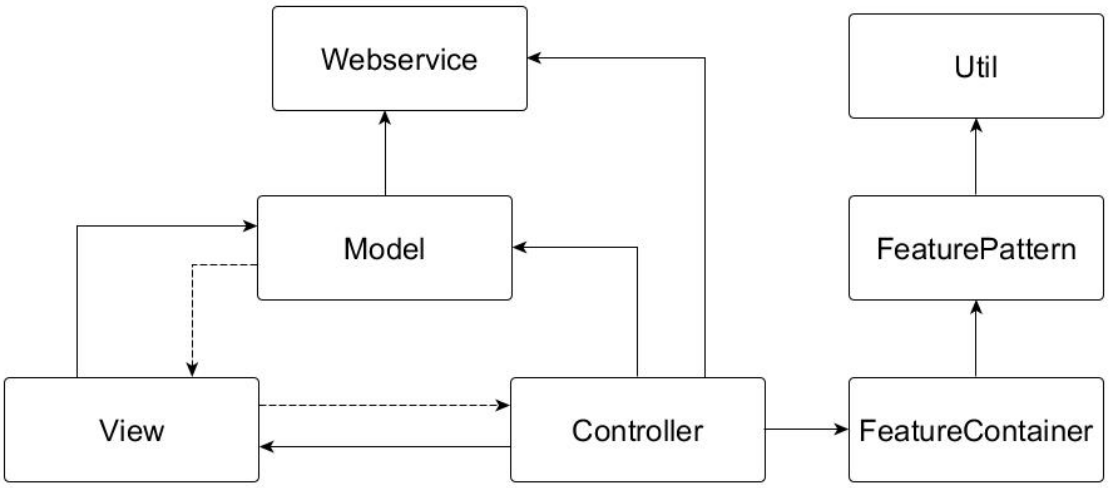


Abbildung 5.6.: Paketstruktur des Clusterers: Darstellung der Abhängigkeiten zwischen den einzelnen Paketen.

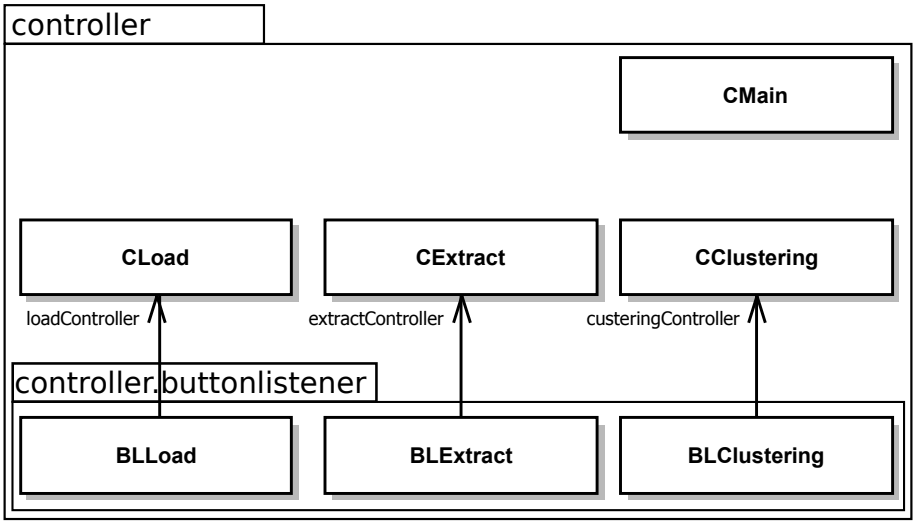


Abbildung 5.7.: *Controller*-Paket: Diagramm aller Klassen und Unterpakete des *Controller*-Pakets.

update-Methode, die bei bestimmten Änderungen gemäß des Beobachter-Entwurfsmusters aufgerufen wird. Diese Änderungen werden daraufhin in der grafischen Oberfläche dargestellt.

Model-Paket

Das in der Abbildung 5.9 dargestellte *Model*-Paket ist für die Speicherung der Daten zuständig. Das Paket teilt sich auf in eine Oberklasse *Model* und zwei Unterklassen *FeatureModel* und *InternetdienstModel*. Die Klasse *FeatureModel* verwaltet die Merkmale. Im Gegensatz dazu werden in der Klasse *InternetdienstModel* alle Internetdienste gespeichert. Die Oberklasse *Model* vereinigt die beiden Unterklassen und liefert über die Methoden *getTrainSetInstances* und *getTestSetInstances* eine Abbildung, in der für alle Internetdienste der jeweiligen Kategorie deren Werte für alle Merkmale aufgeführt sind. Zusätzlich kann über die Oberklasse noch auf die Instanz der Unterklassen zugegriffen werden.

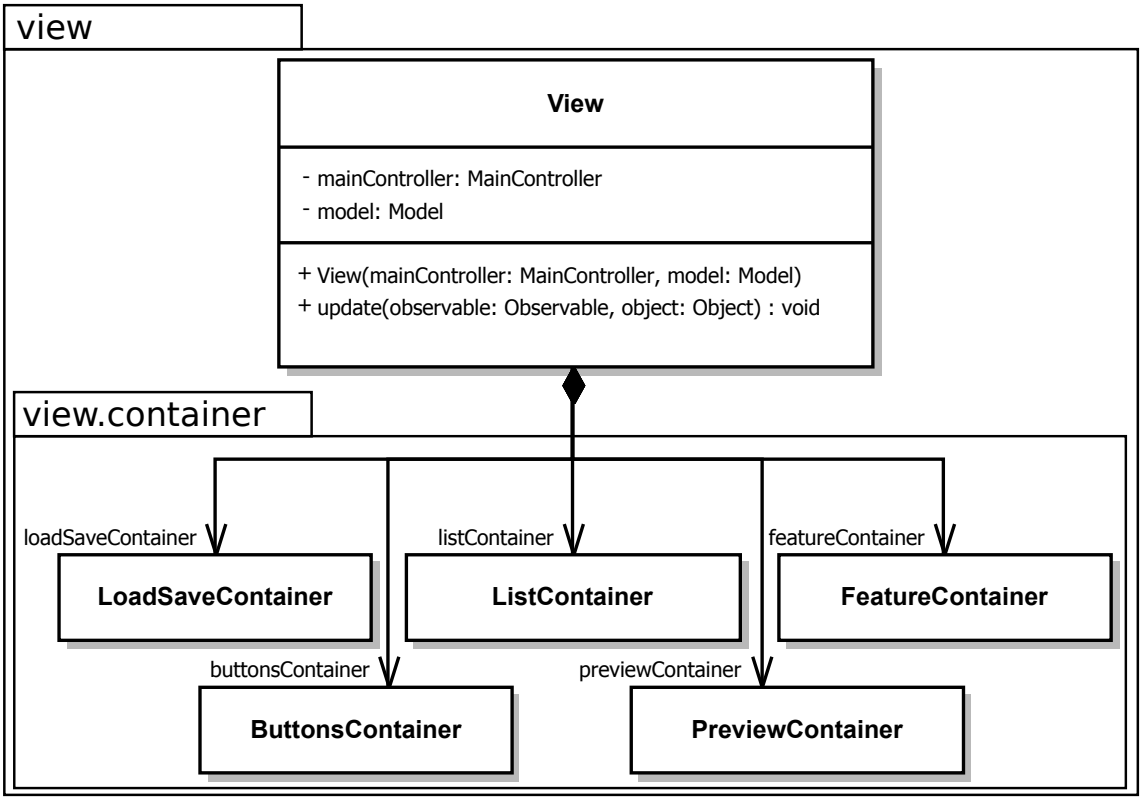


Abbildung 5.8.: *View*-Paket: Diagramm aller Klassen und Unterpakete des *View*-Pakets.

FeaturePattern-Paket

Das *FeaturePattern*-Paket bildet die im Abschnitt 4.2 analysierten Merkmalsmuster ab. Dazu wird für jedes Merkmalsmuster eine Klasse erzeugt, die von der abstrakten Klasse *FeaturePattern* erbt. Die Abbildung 5.10 skizziert passend dazu den Aufbau des *FeaturePattern*-Pakets. Die abstrakte Klasse definiert einen String für den Namen des Merkmalsmusters, einen Satz an Strings für die Speicherung der Semantik anhand von Begriffen und eine Abbildung, die für einen Internetdienst den Wert des Merkmals liefert. Zusätzlich implementiert diese abstrakte Klasse noch einige Methoden, die den Zugriff der jeweiligen Variablen handhaben. Die Klassen, die von der abstrakten Klasse *FeaturePattern* erben, erzeugen bei ihrer Konstruktion durch die Angabe des Internetdienstes das jeweilige Merkmal. Dafür muss im Konstruktor spezifiziert sein wie das Merkmalsmuster heißt und aus welchen Attributen oder Elementen die Begriffe für die Semantik gefiltert werden. Zusätzlich wird der Internetdienst, von dem das Merkmal abgeleitet wurde, noch in der Variable *webservices* gespeichert, um nachher Kenntnis darüber zu haben, welcher Internetdienst dieses Merkmal besitzt. Aufgrund der Vielzahl an Merkmalsmustern werden die Klassen analog zu der Einteilung der Merkmalsmuster in die Unterpakete *count*, *head* und *form* unterteilt.

FeatureContainer-Paket

Das *FeatureContainer*-Paket erzeugt, vereint und speichert alle Merkmale. Die Abbildung 5.11 spiegelt die interne Klassenstruktur wider. Die Ausgangssituation dieses Pakets bilden die drei abstrakten Klassen *AFeatureContainer*, *AFeatureContainerNode* und *AFeatureContainerLeaf*. Mit der konkreten Implementierung dieser Klassen kann eine Baumstruktur aufgebaut werden. Die abstrakte Klasse *AFeatureContainerNode* enthält Methoden für die inneren Knoten und *AFeatureContainerLeaf* für die Blattknoten. Zusätzlich

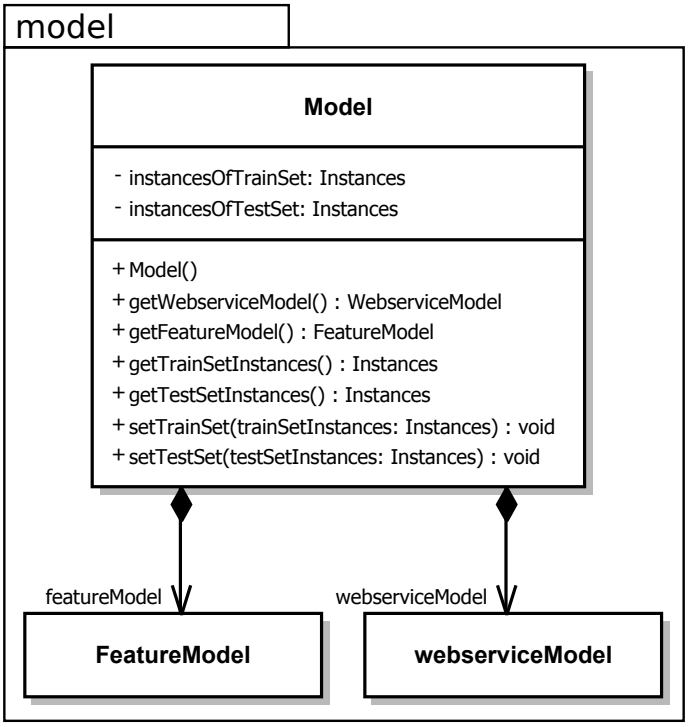


Abbildung 5.9.: *Model*-Paket: Diagramm aller Klassen des *Model*-Pakets.

werden in der übergeordneten abstrakten Klasse *AFeatureContainer* gemeinsame Methoden definiert und teilweise auch implementiert.

Um nun für jedes der implementierten Merkmalsmuster im *FeaturePattern*-Paket konkrete Merkmale zu erzeugen, wird im *FeatureContainer*-Paket für jedes Merkmalsmuster eine Klasse erzeugt, die von *AFeatureContainerLeaf* erbt. Jede dieser Klassen erzeugt und verwaltet alle Merkmale des dazugehörigen Merkmalsmusters. Für die Erzeugung wird jeweils auf die passende Klasse im *FeaturePattern*-Paket zugegriffen, da darin die Erzeugung des Merkmals spezifiziert ist. In den Blattknoten wird neben der Erzeugung auch noch die Vereinigung von zwei Merkmalen des gleichen Merkmalsmusters übernommen, da diese Merkmale von der selben Klasse verwaltet werden.

Die Vereinigung von Merkmalen mit unterschiedlichen Merkmalsmustern wird von den inneren Knoten übernommen. Diese Funktionalität wird dadurch erreicht, dass die inneren Knoten die Merkmale ihrer Kindknoten kennen und somit gemäß der Tabelle 4.3 Merkmale von unterschiedlichen Merkmalsmustern vereinen können. Aus diesem Sachverhalt ergibt sich, dass der Wurzelknoten, also die *Container*-Klasse, alle Merkmale kennt.

Aufgrund der Übersichtlichkeit wurde in der Abbildung 5.11 die interne Paketstruktur nicht abgebildet. Allerdings enthält das Paket *featurecontainer* die drei abstrakten Klassen und die Unterpakete *featurecontainer.node* und *featurecontainer.leaf* bilden die jeweiligen Blatt- bzw. inneren Knoten ab.

Util-Paket

Das *Util*-Paket ist in der Abbildung 5.12 dargestellt. Es ist ein Hilfspaket, das Methoden für die Semantikerkennung anbietet. Die *PoSTagger*-Klasse ist nach dem Singleton-Entwurfsmuster entworfen und liefert über die beiden abgebildeten Methoden die Instanz des jeweiligen Taggers. Die *StringFormatter*-Klasse ist eine Hilfsklasse, die statische Methoden zur Formatierung von Strings anbietet. Die Aufgaben und die Implementierung der beiden Klassen wird im Unterabschnitt 5.4.3 erläutert.

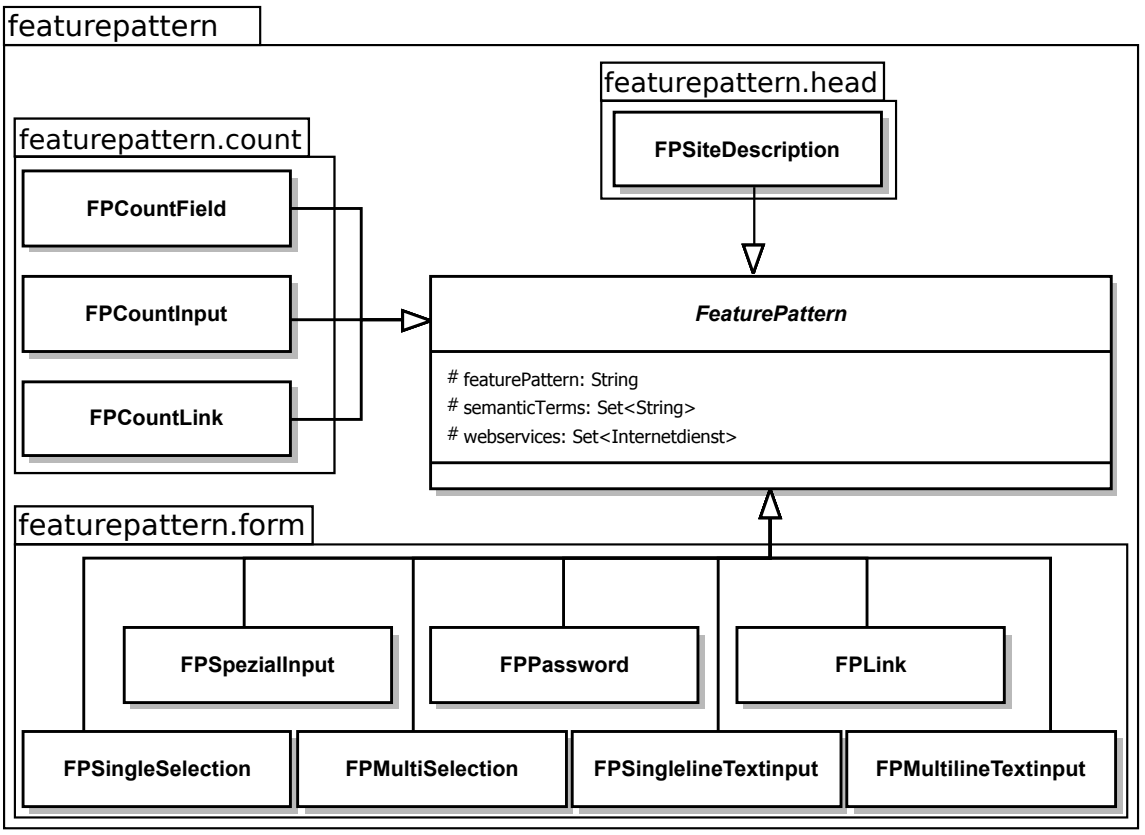


Abbildung 5.10.: *Feature*-Paket: Diagramm aller Klassen und Unterpakete des *Feature*-Pakets.

Webservice-Paket

In der Abbildung 5.13 ist das *Webservice*-Paket mit der Klasse *Webservice* und seiner Hilfsklasse *ClusterRenamer* dargestellt. Die *Webservice*-Klasse modelliert einen Internetdienst. Somit wird für jeden in den Eingabedateien spezifizierten Internetdienst ein Objekt dieser Klasse erzeugt. Jeder Internetdienst enthält bei dieser Modellierung eine URL der Webseite, ein Formular, eine Webseitenbeschreibung, falls vorhanden eine manuell bestimmte Klassifizierung und nach dem Clustering eine vorhergesagte Klassifizierung. Alle diese Variablen werden aus der XML-Datei, in der die Internetdienste spezifiziert sind, abgeleitet.

Auf all diese Variablen kann jeweils über *Getter*-Methoden zugegriffen werden. Eine *Set*-Methode existiert nur für das Setzen des Clusters, das der Clusteringalgorithmus für diesen Internetdienst vorhergesagt hat. Außerdem existieren noch zwei *Getter*-Methoden, die nicht die Nummer sondern den Namen der Internetdienstkategorie zurückgeben. Zur Umsetzung dieser Funktionalität benötigen die beiden Methoden die Hilfsklasse *ClusterRenamer*. Diese Klasse hat die Zuordnung einer Clusternummer zu einer Internetdienstkategorie fest kodiert und führt die Abbildung über die Methode *rename* durch.

5.3.3. Laden der Trainings- und Testmenge

Der Clusterer erhält als Eingabe zwei Dateien. Eine Datei enthält die Menge aller Internetdienste für die Trainingsmenge und die andere Datei alle Internetdienste der Testmenge. Die Aufgabe beim Laden der Eingabedateien ist nun die darin spezifizierten Internetdienste im *Model* zu speichern. Dafür wird für jeden Internetdienst ein *Webservice*-Objekt erzeugt und je nach Eingabedatei der Trainings- oder Testmenge des *Models* hinzugefügt.

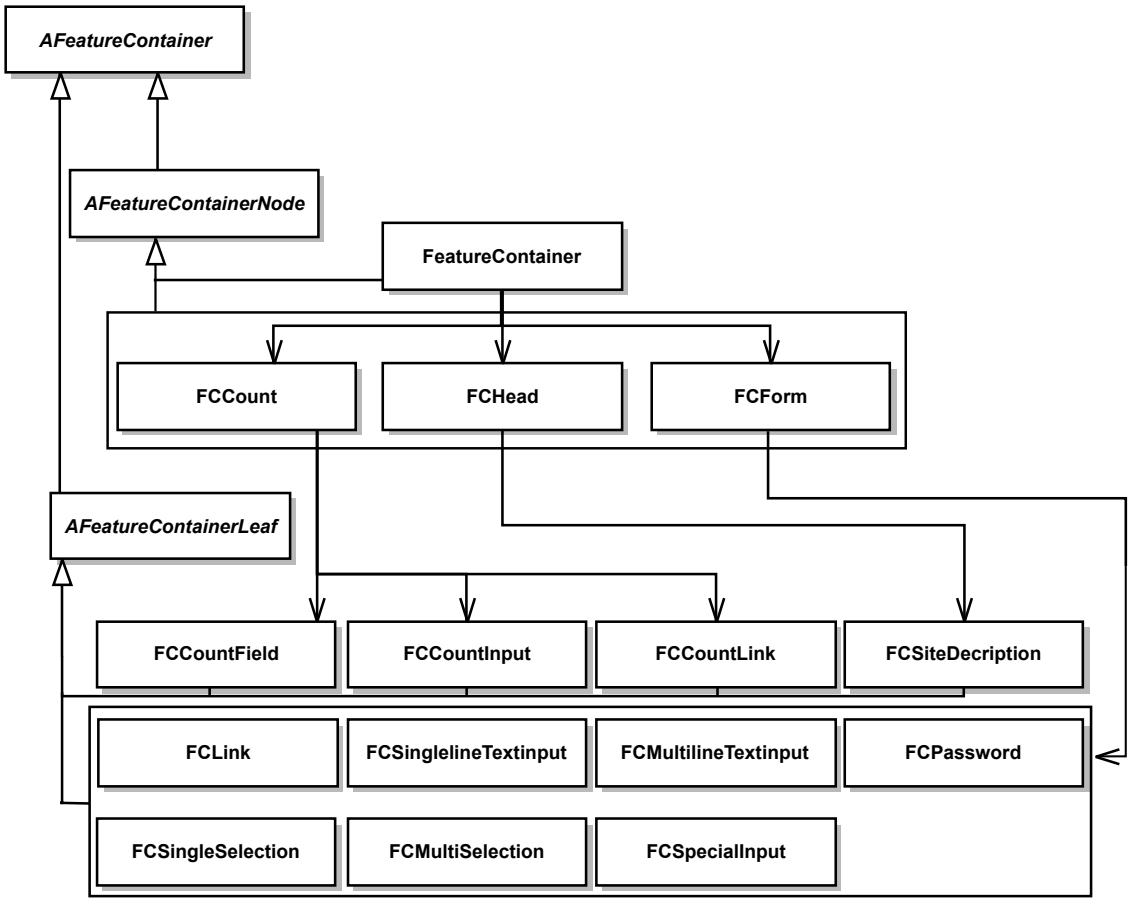


Abbildung 5.11.: *FeatureContainer*-Paket: Diagramm aller Klassen und Unterpakete des *FeatureContainer*-Pakets.

Die beiden Eingabedateien sind im XML-Format codiert und enthalten eine Liste mit allen Internetdiensten der jeweiligen Menge. Dabei ist ein Internetdienst gemäß der Ausgabe des Webcrawlers, die in der Abbildung 5.3 illustriert ist, spezifiziert. Zusätzlich kann der Spezifikation noch über die Umgebung

$$< \text{cluster} > \text{ZAHL} < /\text{cluster} >$$

die manuelle Klassifikation des Internetdienstes hinzugefügt werden. Diese Variable wird verwendet, um die Ergebnisse des Clusterers nachher autonom zu evaluieren.

5.3.4. Merkmalserkennung

Für die Merkmalsextraktion und damit verbunden die Erzeugung und Vereinung von Merkmalen ist das *FeatureContainer*-Paket zuständig. Die Klassen innerhalb des Pakets sind in einer Baumstruktur angeordnet, wobei die Klasse *FeatureContainer* dem Wurzelknoten entspricht. Die Abbildung 5.14 skizziert einen Teil Klassen und ihre Beziehungen.

Die Grundidee dieser Struktur ist, dass der Aufruf einer Methode vom Wurzelknoten an die Blattknoten delegiert wird und die Ergebnisse in umgekehrter Richtung wieder aufgesammelt werden. Nach diesem Prinzip funktioniert auch die Erzeugung der Merkmale. Dafür wird wie in der Abbildung dargestellt am Wurzelknoten die *createFeature*-Methode aufgerufen und zu allen Blattknoten delegiert. Diese Blattknoten erzeugen daraufhin die Merkmale und speichern sie. Aufgrund der Struktur werden in jeder Klasse Merkmale des selben Merkmalsmusters gespeichert, daher bietet es sich an, vor der Speicherung eines

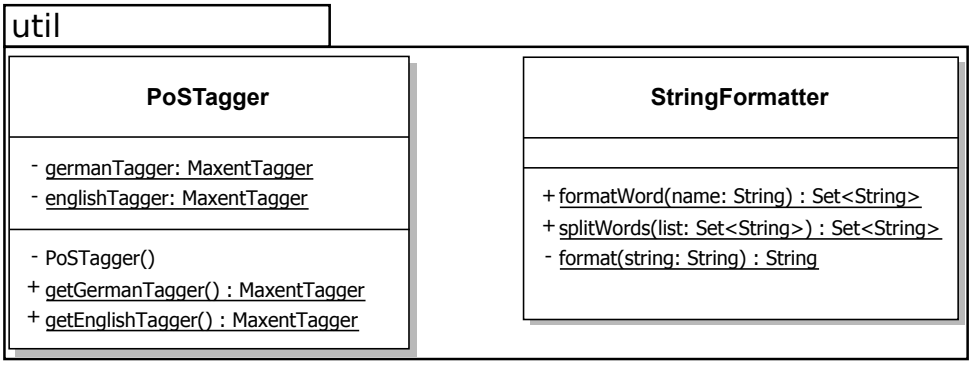


Abbildung 5.12.: *Util*-Paket: Diagramm der beiden Hilfsklassen zur Erzeugung eines Merkmals.

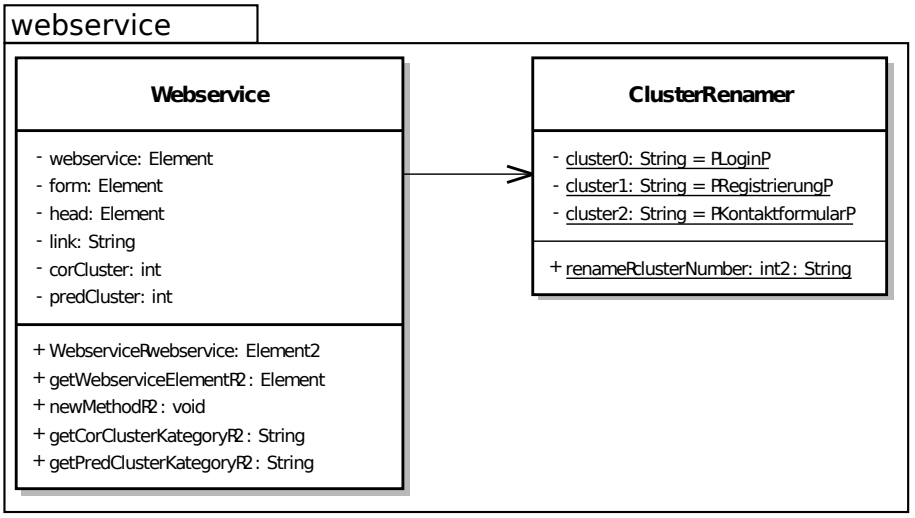


Abbildung 5.13.: *Webservice*-Paket: Diagramm aller Klassen des *Webservice*-Pakets.

neuen Merkmals zu überprüfen, ob dieses Merkmal nicht mit einem bereits in der Klasse vorhandenen Merkmal vereint werden kann. Dadurch werden bereits bei der Erzeugung der Merkmale alle übereinstimmenden Merkmale mit gleichen Merkmalsmustern vereint. Gemäß der Tabelle 4.3 werden somit alle auf der Diagonalen befindlichen Vereinigungen bereits bei der Erzeugung vorgenommen.

Die übrigen Vereinigungen werden anschließend durch den Aufruf der *merge*-Methode abgearbeitet. Im Gegensatz zur *createFeature*-Methode wird der Aufruf der Methode am Wurzelknoten nicht bis zu den Blattknoten delegiert sondern, nur bis zu einem inneren Knoten der Zugriff auf die Merkmale zweier unterschiedlicher aber vereinbarer Merkmalsmuster hat. Im Bezug auf die Abbildung wird von der Klasse *FCForm* die Vereinigung der Merkmale mit dem *FCSinglelineTextinput*- und dem *FCSpecialInput*-Merkmalsmuster implementiert.

Nach Ausführung beider Methoden gibt der *getFeatures*-Methodenaufruf am Wurzelknoten alle extrahierten Merkmale zurück. Dabei werden wiederum von den Blattknoten über die inneren Knoten hin zum Wurzelknoten alle erzeugten Merkmale aufgesammelt.

5.3.5. Merkmalserzeugung und -vereinigung

Bei der Merkmalserzeugung wird je nach Merkmalsmuster nach Elementen innerhalb des Formulars oder der Webseitenbeschreibung gesucht. Im Falle eines Treffers wird ein Merk-

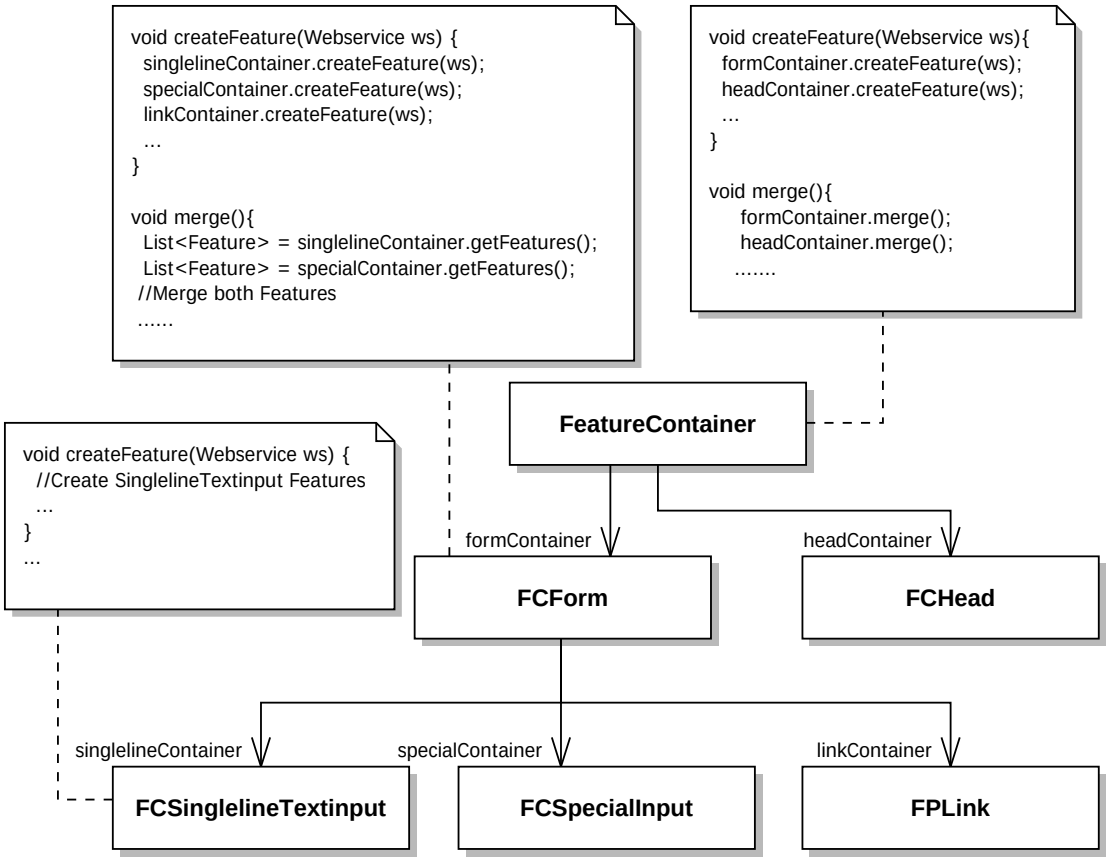


Abbildung 5.14.: Merkmalserkennung: Darstellung eines Ausschnitts des *FeatureContainer*-Pakets. Anhand der Baumstruktur werden die Methodenaufrufe *createFeature* und *merge* an die entsprechenden Klassen delegiert.

mal erzeugt, dass als Typ den Namen des Merkmalsmusters und je nach Muster gegebenenfalls noch die Semantik des Elements speichert.

Die Semantik eines Formularelements wird über eine Liste an Begriffen gespeichert. Diese Liste entspricht den Begriffen bestimmter Attributwerte des Formularelements sowie dessen Label. Die wichtigste Bedingung für die Extraktion ist die Aussagekraft des Begriffs für das zugrundeliegende Formularelement. Denn es sollen Worte, die in jedem Element auftreten können, wie beispielsweise ein Artikel, nicht in die Liste aufgenommen werden. Daher werden die folgenden drei Annahmen über die Aussagekraft getroffen:

Annahme 1: Ein Attributwert oder Label mit nur einem Wort gibt Auskunft über die Semantik des Elements.

Annahme 2: Für Attributwerte oder Labels mit mehreren Wörtern sind die darin enthaltenen Nomen von hoher Aussagekraft für das Formular.

Annahme 3: Sollte ein nach Annahme 1 und 2 aussagekräftig klassifizierter Begriff mindestens nochmal bei zwei anderen Elementen innerhalb des Formulars auftauchen, dann handelt es sich um einen übergeordneten Begriff, der für das spezielle Formularelement keine Aussagekraft hat.

Die Annahmen 1 und 2 basieren auf der Grundlage, irrelevante Worte zu entfernen. Im Gegensatz dazu wird durch die Annahme 3 versucht, redundante Begriffe zu entfernen, da sie beispielsweise Auskunft über den Internetdienst geben, aber nicht über die Semantik dieses bestimmten Elementes. Ein Beispiel dafür sind Implementierungen von Internet-

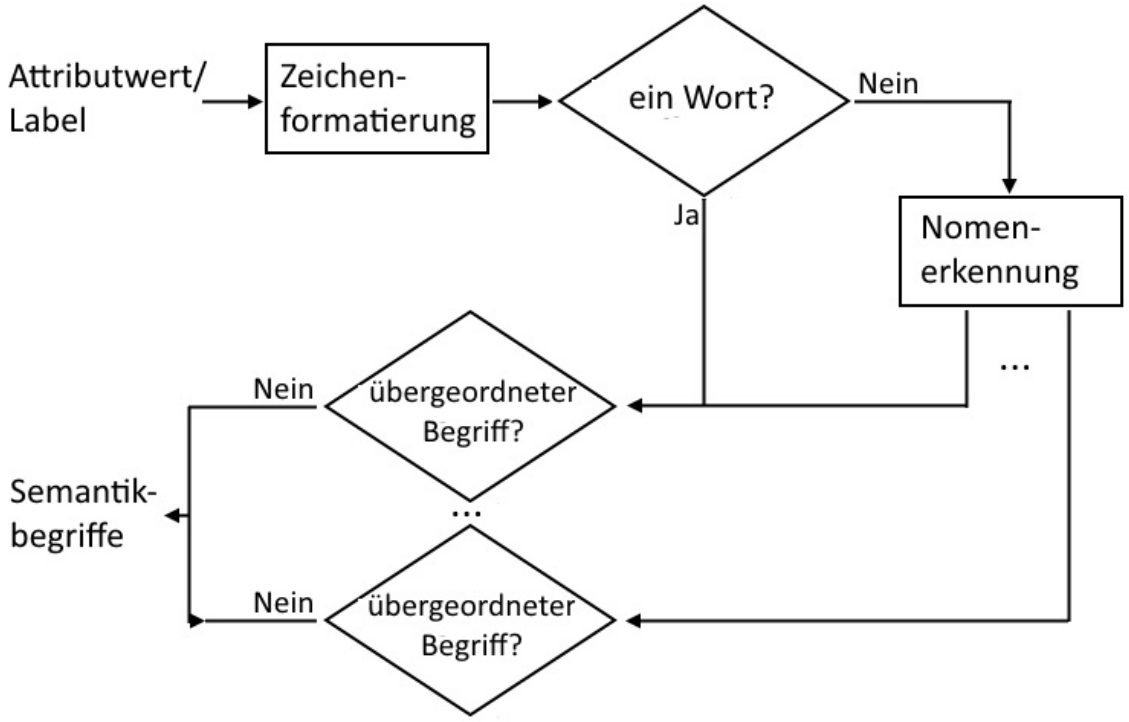


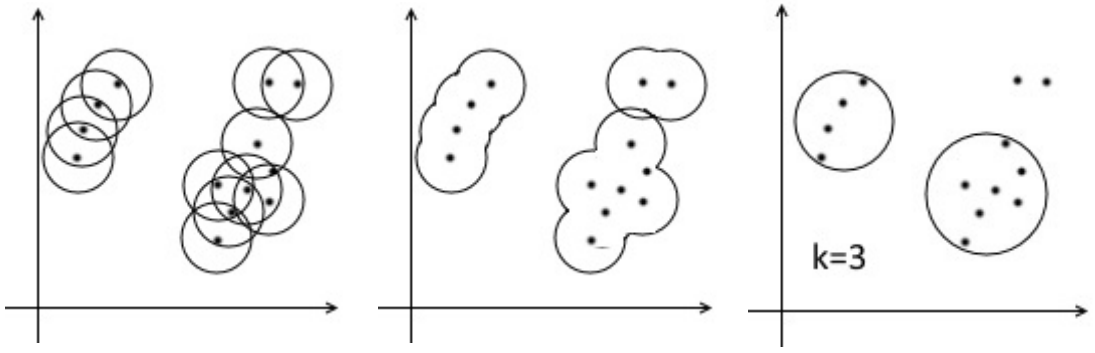
Abbildung 5.15.: Merkmalserzeugung: Darstellung der Aktivitäten bei der Extraktion von Sematikbegriffen

diensten, die jedem *name*-Attribut den Namen des Internetdienstes übergeben. Auf Basis der Annahmen illustriert die Abbildung 5.15 den Ablauf der Extraktion von Begriffen aus einem Attribut oder dem Label. Zusätzlich zu der Fallunterscheidung und der Nomensextraktion kommt die Vorverarbeitung des Attributwertes bzw. Labels. Bei diesem Vorverarbeitungsschritt wird der String formatiert, da bei der Angabe von Attributwerten meist auf das Leerzeichen verzichtet wird und somit die Eingabe von mehreren Wörtern durch verschiedenste Zeichen oder durch die Groß- und Kleinschreibung getrennt werden. Diese Trennung wird in der Formatierung versucht rückgängig zu machen und somit die Strings zu vereinheitlichen. Eine detailliertere Beschreibung der Umsetzung von Stringformatierung und Nomenerkennung findet man im Abschnitt 5.4.

5.3.6. Merkmalsselektion

Bei der Merkmalsselektion werden redundante und irrelevante Merkmale verworfen. Für die Klassifikation dieser Merkmale wird das Verfahren nach [Hal99] verwendet, dass die Korrelation und Merkmalsanzahl betrachtet. Dabei wird für eine Menge an Merkmalen ein Index berechnet und anschließend die Indexwerte verschiedener Teilmengen verglichen. Die Merkmale der Teilmenge mit dem höchsten Index werden als relevant klassifiziert und bilden die Ergebnismenge der Selektion. Für die Berechnung des Index einer Teilmenge werden die Merkmalsanzahl, die (Intra-)Korrelation zwischen den Merkmalen und die (Inter-)Korrelation zwischen dem Merkmal und der Kategorie des Internetdienstes in Beziehung gesetzt. Dabei erhöht sich der Index bei der Hinzunahme von Merkmalen oder bei Erhöhung der Intra-Korrelation und im Gegensatz dazu sinkt der Index, wenn die Inter-Korrelation sich erhöht. Diese Aussage gilt nur unter der Voraussetzung, dass sich die anderen beiden Variablen jeweils nicht ändern. Dies ist jedoch meist nicht gegeben, da beispielsweise beide Korrelationen von den Merkmalen abhängen.

Zur vollständigen Umsetzung der Selektion müssen nun noch die Teilmengen gebildet werden. Da eine vollständige Untersuchung aller möglichen Teilmengen zu zeitaufwendig



(a) Aufspannen des Radius r (b) Vereinigung (c) Clustering

Abbildung 5.16.: Clustering: Darstellung der Vorgehensweise des *DBScan*-Algorithmus

ist, wird die Bestensuche in Vorwärtsrichtung eingesetzt. Den Ursprung bildet die leere Menge und dazu wird jedes der vorhandenen Merkmale hinzugefügt und anschließend der Index dieser Mengen verglichen. Die Teilmenge mit dem höchsten Index bildet nun den Ursprung und es wird erneut jeweils eins der restlichen Merkmale zur Ursprungsmenge hinzugefügt und anschließend evaluiert. Dies wird iterativ fortgesetzt bis der Index der Ursprungsmenge höher ist als alle Indizes der Ursprungsmenge plus eines der restlichen Merkmale.

5.3.7. Clustering

In der Analysephase wurden zwei verschiedene Clusteringalgorithmen verwendet. Nachfolgend wird für beide Algorithmen der Entwurf beschrieben, sowie im Abschnitt 6.4 eine Evaluierung durchgeführt.

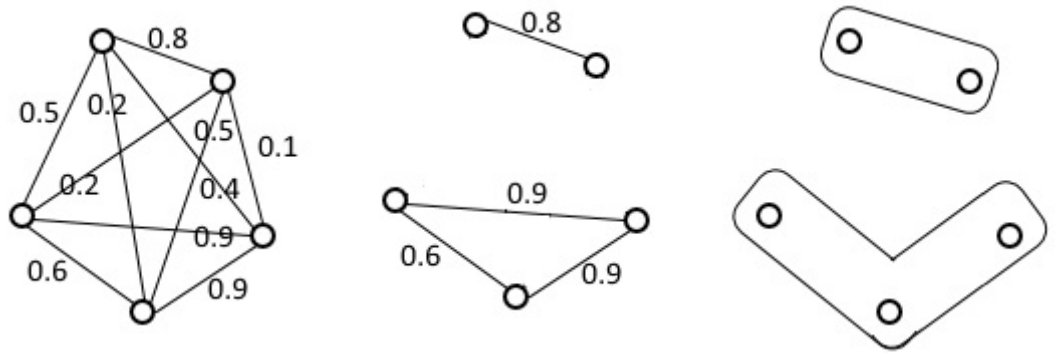
Dichtebasiertes Spatial Clustering - *DBScan*

Der *DBScan* nach [Est+96] überführt die Internetdienste in einen Merkmalsvektorraum, indem er die Merkmale auf einen Merkmalsvektor abbildet. Dabei entspricht jeder der Achsen des Raumes einem Merkmal. Basierend auf dieser Ausgangsposition kann nun das Clustering anhand zweier Parameter reguliert werden:

Radius r: Der Radius r gibt an, in welchem Umkreis um einen Datenpunkt nach anderen Datenpunkten gesucht wird.

Anzahl k: Der Parameter k gibt die Mindestanzahl an Datenpunkten an, die zusammen ein Cluster bilden können.

Mit Hilfe der Abbildung 5.16 werden im folgenden die einzelnen Schritte des *DBScan*-Algorithmus beschrieben. Zu Beginn wird mit der Angabe dieser beiden Parameter für jeden Datenpunkt im Merkmalsvektorraum im Radius r nach anderen Datenpunkten gesucht (Abbildung 5.16a). Befindet sich in diesem Radius ein anderer Datenpunkt, dann gehören diese Datenpunkte zusammen (Abbildung 5.16b). Nachdem dies für jeden Datenpunkt getan wurde, bilden die Vereinigungen, die mindestens k Datenpunkte enthalten, einen Cluster und die übrigen Datenpunkte werden als Ausreißer bezeichnet (Abbildung 5.16c). Die Ausreißer können nicht klassifiziert werden, da sie keinem Cluster zugeordnet werden können.



(a) Grapherzeugung (b) Kanten entfernen ($\sigma = 0.55$) (c) Clustering

Abbildung 5.17.: Clustering: Darstellung der Vorgehensweise des *Spectral Clustering*-Algorithmus.

Spectral Clustering

Beim *Spectral Clustering* basiert das Clustering auf der Graphentheorie. Aus den Datenpunkten wird ein voll verbundener Graph erzeugt, dessen Knoten den Datenpunkten entsprechen. Die Abbildung 5.17a zeigt einen solchen Graphen, bei dem jeder Knoten eine Kante zu jedem anderen Knoten besitzt. Das Kantengewicht zweier Knoten berechnet sich aus ihren Merkmalen und zwar anhand einer Ähnlichkeitsfunktion. Sind die Merkmale zweier Datenpunkte identisch, dann liefert die Ähnlichkeitsfunktion den Wert 1 und je unterschiedlicher die Merkmale werden, desto mehr konvergiert der Wert der Ähnlichkeitsfuntion gegen 0. Auf diesem Graphen wird nun das Clustering berechnet, wobei der folgende Parameter die Clusterbildung reguliert:

Parameter σ: Der Parameter σ gibt den Mindestwert der Ähnlichkeitsfunktion vor.

Mit der Angabe des Parameters σ werden aus dem Graphen alle Kanten mit kleinerem Ähnlichkeitswert entfernt (Abbildung 5.17b). Anhand dieses Graphen werden nun die Cluster gebildet, indem einfach alle Knoten bzw. Datenpunkte, die eine Kante zueinander haben im selben Cluster landen(Abbildung 5.17c).

5.3.8. Ausgabe

Nach dem Clustering werden die erzeugten Merkmale gemäß dem Abschnitt 4.6 einer Kategorie zugeordnet und anschließend ausgegeben. Wobei die Ausgabe aus der Klassifikation und den Merkmalen des jeweiligen Internetdiensts der Testmenge besteht.

5.4. Implementierung des Clusterers

Der Clusterer ist in der Programmiersprache Java geschrieben. Nachfolgend wird die Implementierung der einzelnen entworfenen Verfahren erläutert.

5.4.1. Grafische Benutzeroberfläche

Die grafische Benutzeroberfläche ist mit der Grafikbibliothek *swing* implementiert. Sie ist Bestandteil der Java-Runtime und bietet einen komfortablen Umgang zur Gestaltung von Oberflächen. Die Benutzeroberfläche des Clusterers ist über das *BoxLayout* in y-Richtung in einen linken und rechten Bereich eingeteilt. Innerhalb des linken Teils werden die einzelnen Bereiche über das *BoxLayout* in x-Richtung gesetzt.

5.4.2. Laden der Trainings- und Testmenge

Zum Einlesen der Dateien in einen String wird die *Apache Commons IO*-Bibliothek [5] verwendet. Dieser String wird über den HTML/XML-Parser *jsoup*[6] gesplittet und somit für jeden Internetdienst und dessen Spezifikationen eine Instanz der *Webservice*-Klasse erzeugt. Zum Schluss wird jedes dieser Objekte im *Model* je nach Zugehörigkeit in einer Liste für die Trainings- oder die Testmenge gespeichert.

5.4.3. Merkmalserkennung

Die einzelnen Blattknoten des *FeatureContainer*-Pakets spezifizieren die Extraktion eines Merkmals je nach Muster. Innerhalb der Musterklassen des *FeaturePattern*-Pakets wird daraufhin noch der Typ und gegebenenfalls die Attribute bzw. das Label für die Semantikerkennung definiert. Allerdings haben alle Merkmalsmuster für die Extraktion die Gemeinsamkeit, dass nach dem Auftritt von einem oder mehreren bestimmten Elementen gesucht wird. Diese Suche innerhalb des Formulars oder der Webseitenbeschreibung wird erneut vom HTML-Parser *jsoup* durchgeführt.

Beispiel: Merkmalsmuster für eine Passworteingabe

Die Extraktion dieses Merkmalsmusters ist in der Klasse *FCPassword* spezifiziert. Die Spezifikation sieht vor, dass für jedes Passworteingabefeld innerhalb des Formulars ein Merkmal erzeugt wird. Der folgende Quellausschnitt illustriert wie diese Spezifikation mittels des HTML-Parser *jsoup* implementiert wird:

```
Element form = webservice.getForm();
for (Element passwordElement : form.select("input[type=password]"){
  FPSpecialInput passwordFeature = new FPSpecialInput(...);
  ...
}
```

5.4.4. Merkmalserzeugung und -vereinigung

Bei der Erzeugung eines Merkmals muss häufig auch die Semantik erkannt werden, wie dies im Unterabschnitt 5.3.5 erläutert ist. Nachfolgend sind die Implementierung der Stringformatierung und Nomenerkennung erläutert.

Stringformatierung

Die Stringformatierung setzt sich aus zwei Teilen zusammen. Im ersten Schritt wird versucht, zusammengesetzte Worte zu splitten und beim zweiten Teil wird jedes einzelne Wort vereinheitlicht.

Die Trennung zusammengesetzter Wörter wird über die Erkennung verschiedener regulärer Ausdrücke realisiert. Ein Indiz für zusammengesetzte Worte bilden besondere Symbole, wie beispielsweise Klammern, der Schräg- oder Bindestrich. Ein weiteres Indiz ist, wenn ein Großbuchstabe direkt auf einen Kleinbuchstaben folgt. Anhand der Erkennung dieser Indizien wird der String gesplittet. Für jedes dieser gesplitteten Worte wird daraufhin der zweite Teil der Stringformatierung angewandt.

Bei der Vereinheitlichung jedes Wortes werden alle Buchstaben in Kleinbuchstaben formatiert, die Umlaute *ä*, *ö*, *ü* durch *ae*, *oe*, *ue* ersetzt und zuletzt alle Zeichen entfernt, die nicht den Buchstaben von *a* bis *z* entsprechen.

[5] *Apache Commons IO 2.4*. Zugriff: 2015.09.05. URL: `https://commons.apache.org/proper/commons-io/`
[6] *JSoup 1.8.1*. Zugriff: 2015.09.05. URL: `http://jsoup.org/download`

Nomenerkennung

Zur Bestimmung der Nomen verwendet der Clusterer einen *Part-of-speech tagger*[7] (*PoSTagger*), der alle Worte eines Satzes je einer Wortgruppe, wie beispielsweise der Gruppe von Nomen, zuteilt. Da diese Zuteilung abhängig von der Sprache ist, wird ein deutschsprachiger und ein englischsprachiger PoSTagger erzeugt. Sollte der deutschsprache PoSTagger bei der Zuteilung ein Fremdwort erkennen, dann wird die Zuteilung nochmals mit dem englischsprachigen PoSTagger vorgenommen.

5.4.5. Merkmalsselektion

Zur Selektion der Merkmale wird die WEKA-Bibliothek [8] verwendet. Sie implementiert einen *AttributeSelection*-Filter, der unter der Angabe eines Evaluierers und eines Suchalgorithmus die Menge an Merkmalen reduziert. Allerdings werden von Weka auch Implementierungen dieser beiden Parameter geliefert. So kann die Bestensuche über die *BestFirst*-Klasse und die Indexierung der Teilmengen gemäß des Entwurfs über den *CfsSubsetEval*-Evaluierer gewählt werden.

5.4.6. Clustering

Eine Implementierung der beiden eingeführten Clusteringalgorithmen *Spectral Clustering* und *DBScan* liefert erneut die WEKA-Bibliothek. In der Bibliothek sind beide Algorithmen über eine gemeinsame Schnittstelle definiert und können somit bis auf die Parametereinstellung identisch behandelt werden.

Nach dem Erzeugen einer Instanz des Algorithmus und der Parametersetzung werden über die Methode *buildClusterer* die Cluster gebildet. Als einzige Eingabe erhält die Methode den Datensatz in Form eines Objekts der *Instances*-Klasse, die alle zu clusternden Internetdienste und dessen Merkmale enthält. Mit dem Aufruf werden nun je nach Clusteringalgorithmus die Cluster gebildet. Zum Auslesen der Ergebnisse wird der *AddCluster*-Filter angewandt, der dem Datensatz ein Merkmal hinzufügt, in dem die berechnete Clusterzuteilung des Algorithmus spezifiziert ist. Anschließend kann innerhalb des Datensatzes für jeden Internetdienst dieses Merkmal ausgelesen werden.

5.4.7. Ausgabe

Für die Ausgabe wird lediglich ein String erzeugt, der die Klassifizierungen und Merkmale der jeweiligen Internetdienste der Testmenge enthält. Die Formatierung der Ausgabe wird im folgenden spezifiziert:

$$
\begin{aligned}
Ausgabe &:= INTERNETDIENST; INTERNETDIENST; \ldots \\
INTERNETDIENST &:= Klassifikation : MERKMALE \\
MERKMALE &:= [MERKMAL], [MERKMAL], \ldots \\
MERKMAL &:= Merkmalsmuster : Semantikbegriff, \\
&\qquad Semantikbegriff, \ldots
\end{aligned}
$$

Die Ausgabe setzt sich also zusammen aus mehreren Internetdiensten, die jeweils durch ein Semikolon getrennt sind. Für jeden dieser Internetdienste wird die Klassifikation gefolgt von seinen Merkmalen ausgegeben, wobei diese beiden Teile über den Doppelpunkt getrennt sind. Die einzelnen Merkmale werden von eckigen Klammern umgeben und jeweils durch ein Komma getrennt. Innerhalb eines Merkmals wird zuerst das Merkmalsmuster gefolgt von einem Doppelpunkt und anschließend einer Liste aller Sematikbegriffe des Merkmals. Diese Begriffe sind erneut durch ein Komma getrennt.

[7] *Part-Of-Speech Tagger 3.5.2.* Zugriff: 2015.09.05. URL: `http://nlp.stanford.edu/software/tagger.shtml#Download`

[8] *Weka 3.6.* Zugriff: 2015.09.05. URL: `http://www.cs.waikato.ac.nz/ml/weka/downloading.html`

5.4.8. Clusterer im Betrieb

Der erläuterte Ablauf des Clusterers ist für eine große Trainingsmenge sehr zeitaufwendig, daher wird versucht im Betrieb nur die wirklich notwendigen Schritte auszuführen. Im Allgemeinen wird es der Fall sein, dass die Trainingsmenge sich nur selten ändert und jeweils nur über die Testmenge neue Internetdienste klassifiziert werden sollen. Dies führt dazu, dass die ständig äquivalente Merkmalsextraktion und -selektion für die Trainingsmenge redundant wäre. Daher bietet das System die Möglichkeit den Zustand nach einmaliger Merkmalsextraktion und -selektion zu speichern. Diesbezüglich werden die nach der Selektion resultierenden Merkmale im *arff*-Format gespeichert. Bei einer erneuten Ausführung des Clusterers mit der selben Trainingsmenge kann anhand dieser Datei der Zustand nach der Merkmalsselektion wieder hergestellt werden. Somit werden in den folgenden Schritten nur noch die zu klassifizierenden Internetdienste über die Testmenge eingelesen, die Merkmale dieser Dienste extrahiert, das Clustering mit der Zuordnung der Kategorien zu einem Cluster durchgeführt und das Ergebnis ausgegeben.

5.5. Zusammenfassung

In diesem Kapitel wurde ein Webcrawler zur autonomen Extraktion und ein Clusterer zur Klassifikation der Internetdienste entworfen. Der in Python implementierte Webcrawler untersucht dabei die Startseite eines Hosts nach Internetdiensten. Zusätzlich zur Startseite werden noch alle darauf verlinkten Dokumente durchsucht.

Der in Java implementierte Clusterer ist ein maschinell lernendes System und kann über eine grafische Benutzeroberfläche bedient werden. Außerdem ist der Clusterer abstrakt betrachtet im *Model-View-Controller*-Architekturmuster entworfen. Über die Auswahl der Trainingsmenge werden anhand von autonom erzeugten Merkmalen Cluster gebildet. Abhängig von den Merkmalen der zu klassifizierenden Internetdienste der Testmenge werden diese entweder einem dieser erzeugten Cluster zugeordnet oder bilden einen neuen Cluster. Sollten sie einen neuen Cluster bilden, dann hat das System noch keine Erfahrungen zu diesem Internetdienst und vermerkt diesen als *unbekannt*. Im Fall, dass der Internetdienst einem bestehenden Cluster zugeordnet wird, erhält dieser die Klassifizierung des Clusters.

6. Evaluation

Die Bewertung meiner Masterarbeit teilt sich in eine interne und externe Evaluierung des entwickelten Clusterers auf. Bezüglich der internen Evaluierung wird abhängig vom Clusteringalgorithmus und seiner Parametrisierung die interne Struktur der gebildeten Cluster untersucht. Darauf aufbauend wird in der externen Evaluierung der Betrieb der Software simuliert und somit die Klassifikation von unbekannten Internetdiensten bewertet. Zum Schluss werden die Resultate der einzelnen Clusteringalgorithmen verglichen und in Relation gesetzt.

6.1. Aufbau

Zur Evaluation des gesamten Clusterers muss die interne und externe Qualität bestimmt werden. Sie geben Auskunft über das Verhalten des Clusters in der Trainingsphase und in der Betriebsphase. Voraussetzung für die Verhaltensanalyse ist die Eingabe eines Datensatzes an Internetdiensten. Daher werden im folgenden der Datensatz, sowie der Aufbau der internen und externen Evaluation dargelegt.

6.1.1. Datensatz

Zur Erzeugung der Eingabemenge wurde der entwickelte Webcrawler verwendet. Unter der Angabe bestimmter initialer Webseiten wurden 292 der extrahierten Internetdienste als Datensatz zusammengefasst. Die Wahl der initialen Webseiten war willkürlich und basierte auf dem Ansatz, für jede der Internetdienstkategorien des Analysekapitels mindestens eine initiale Webseite zu spezifizieren. Eine Auflistung aller initialen Webseiten und die Kategorien der Internetdienste, die sich auf den jeweiligen Seiten befinden, kann der Tabelle 6.1 entnommen werden.

Die 292 enthaltenen Internetdienste verteilen sich gemäß der Tabelle 6.2 auf die einzelnen Internetdienstkategorien. Detaillierter betrachtet wird in der Tabelle 6.2a die Verteilung auf die analysierten Internetdienstkategorien aufgeführt. Zu diesen 260 Internetdiensten sind im Datensatz noch weitere Internetdienste enthalten, die anhand ihrer Dienstleistung manuell gruppiert wurden. Die Tabelle 6.2b führt diese Kategorien auf. Bei genauerer Betrachtung sind in den zusätzlichen Kategorien sehr spezielle Internetdienste enthalten. Die zusammengefassten Internetdienste agieren zwar auf einer abstrakten semantischen Ebene ähnlich, jedoch sind ihre Implementierungen häufig sehr unterschiedlich. Ihre spezielle Funktionalität führte auch dazu, dass diese Dienste nicht im Fokus der Arbeit lagen.

Initiale Webseiten	Internetdienstkategorien
www.bahn.de	Fahrplanauskunft
http://www.kvv.de	Fahrplanauskunft
Berlin-Airport[1]	Flugauskunft
www.billiger-mietwagen.de	Autovermietung
http://www.hrs.de/	Unterkunftssuche
www.aohotels.com	Unterkunftssuche
http://www.wetter.com	Wettervorhersage

[1] http://www.berlin-airport.de/de/reisende-sxf/ankuenfte-und-abfluege/fluggesellschaften/

Tabelle 6.1.: Initiale Webseiten des Webcrawlers: Diese Seiten wurden als Eingabe dem Webcrawler übergeben.

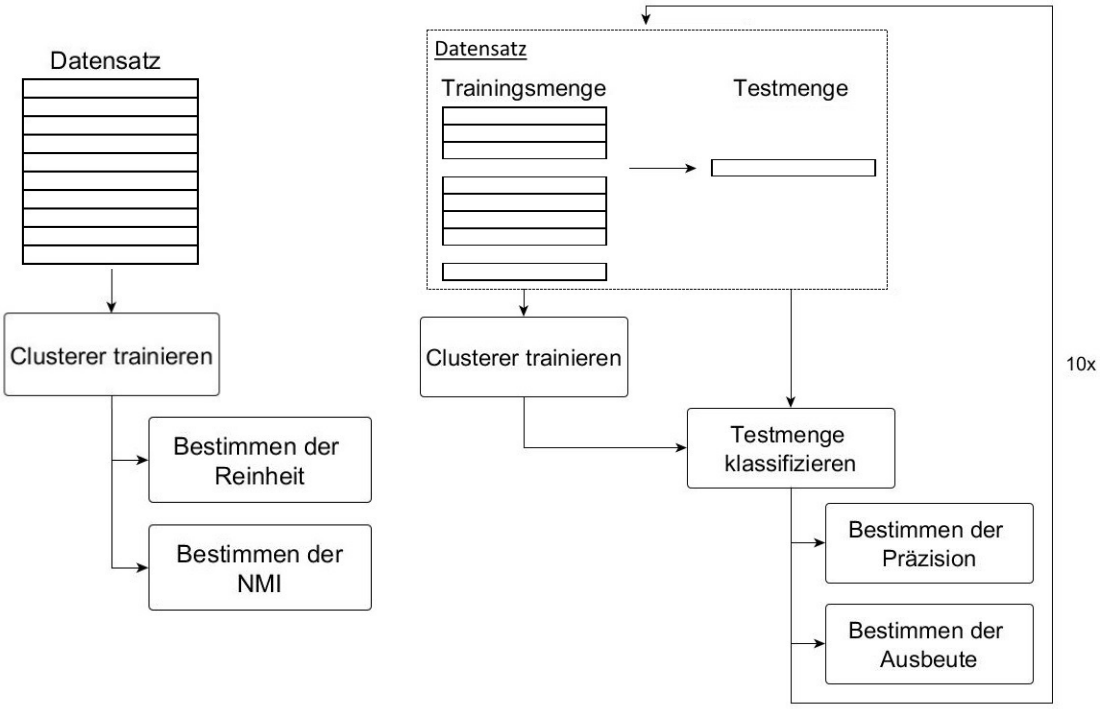
Internetdienstkategorien	Anzahl
Login	36
Registrierung	12
Kontaktformular	39
Fahrplanauskunft	41
Flugauskunft	23
Autovermietung	19
Unterkunftssuche	41
Newsletterabonnierung	24
Wettervorhersage	25
Gesamt	**260**

(a) analysierte Internetdienstkategorien

Internetdienstkategorien	Anzahl
Sprachauswahl	13
Mehrfache Auswahlen	3
Veranstaltungsfilter	1
Parkplatzverfügbarkeit	1
Buchung verwalten	2
Filterung von Artikeln	4
Ticketrechner	3
Feedbackformular	1
Suche	4
Gesamt	**32**

(b) nicht analysierte Internetdienstkategorien

Tabelle 6.2.: Datensatz: Die Verteilung der 292 Internetdienste auf die einzelnen Kategorien



(a) Evaluierung des Custeringalgorithmus

(b) Evaluierung der Klassifikation

Abbildung 6.1.: Evaluation: Ablauf beider Evaluationsmodi

6.1.2. Evaluierung des Clusteringalgorithmus

Bei der Evaluation des Clusteringalgorithmus wird der Umgang mit Wissen(=Erfahrungen) des maschinell lernenden Systems bewertet. Das Wissen des Clusterers basiert auf der Trainingsmenge. Folglich werden die anhand der Trainingsmenge erzeugten Cluster evaluiert.

Die Abbildung 6.1a skizziert den Ablauf dieses Modus. Als Trainingsmenge erhält der Clusterer den gesamten Datensatz. Für diesen Datensatz wird die Analyse ausgeführt und Cluster, mittels der erzeugten Merkmale, gebildet. Diese Cluster werden über die Metriken im Abschnitt 6.2 ausgewertet und somit die Qualität des Clusterers bestimmt.

6.1.3. Evaluation des Klassifikators

Bei der Evaluation des Klassifikators wird der Betrieb des Clusterers simuliert. Die Bewertung basiert dabei auf den Metriken des Abschnitts 6.3, die für jede Internetdienstkategorie angeben mit welcher Präzision und Ausbeute unbekannte Internetdienste erkannt werden.

Zur Simulation dieser Betriebsphase wird eine 10-fache stratifizierte Kreuzvalidierung angewandt, deren Ablauf in der Abbildung 6.1b skizziert ist. Zur Bestimmung von unbekannten Internetdiensten muss der Datensatz in eine Trainings- und Testmenge aufgeteilt werden. Bei der 10-fachen stratifizierten Kreuzvalidierung wird dazu der Datensatz in zehn gleichgroße disjunkte Teilmengen aufgespalten und jede der Teilmengen enthält eine annähernd gleiche Verteilung der Kategorien. In jedem Durchlauf bildet eine andere Teilmenge

die Testmenge und die übrigen die Trainingsmenge. Damit jede der Teilmengen einmal die Testmenge bildet, werden zehn Durchläufe vollzogen. In jedem Durchlauf werden die Internetdienste der Testmenge klassifiziert und anschließend anhand verschiedener Metriken bewertet.

6.2. Metriken für die Evaluation des Clusteringalgorithmus

Diese Metriken geben Auskunft über die innere Struktur der gebildeten Cluster. Diese kann je nach Parametrisierung des Clusteringalgorithmus unterschiedlich ausfallen und muss daher bewertet werden. Typische Problematiken sind die Über- bzw. Unteranpassung des Systems. Bei umgekehrter Betrachtung kann wiederum aus den Evaluationswerten eine geeignete Parametrisierung des Algorithmus für die Evaluation des Klassifikators gewählt werden. Im folgenden werden dazu zwei Metriken vorgestellt und anhand von Beispielen die Aussage ihrer Werte veranschaulicht.

6.2.1. Reinheit

Das Maß der Reinheit $r \in (0, 1]$ gibt Auskunft über die innere Struktur der Cluster. Ein Reinheitswert von 1 besagt, dass in jedem Cluster nur Datenpunkte der selben Kategorie sind. Mit sinkendem Reinheitswert erhöht sich somit die Anzahl an Datenpunkten mit verschiedenen Kategorien. Konkret berechnet sich die Reinheit durch die Aufsummierung aller Datenpunkte, deren Kategorie im jeweiligen Cluster die Mehrheit bildet und anschließender Normierung über alle Datenpunkte.

Problematisch an dieser Metrik ist, dass die Anzahl an Cluster nicht betrachtet wird. Folglich kann die Reinheit erhöht werden, wenn das Clustering überangepasst wird. Im Extremfall wird somit für jeden Datenpunkt ein Cluster erzeugt und dies führt automatisch zum optimalen Reinheitswert von 1. Zur Verhinderung dieses Sachverhalts wird die normalisierte Transinformationsmetrik betrachtet.

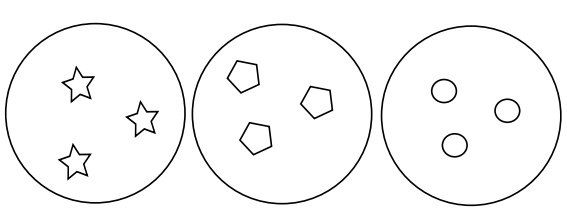
6.2.2. Normalisierte Transinformation

Die normalisierte Transinformationsmetrik (engl. *normalized mutual information - NMI*) bewertet einen Clusteringalgorithmus anhand der Wahrscheinlichkeit, dass eine bestimmte Kategorie in einem bestimmten Cluster ist, sowie der Anzahl an Clustern und Kategorien. Durch die Normalisierung ergibt sich ein Wertebereich zwischen inklusive 0 und 1 für diese Metrik. Aus einem hohen Wert kann abgeleitet werden, dass die Anzahl an Cluster in etwa der Anzahl an Kategorien entspricht und außerdem die Reinheit in den Clustern hoch ist. Im Gegensatz dazu ist bei einem niedrigen Transinformationswert der Clusterer über- bzw. untergepasst oder der Reinheitswert sehr niedrig.

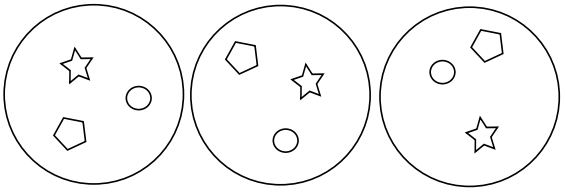
Aus der zuvor beschriebenen Interpretation der Metrikenwerte wird ersichtlich, dass die Reinheit und die normalisierte Transinformationsmetrik abhängig voneinander sind. Jedoch kann durch die Betrachtung beider Werte die innere Struktur der Cluster nachvollzogen und somit das Clustering evaluiert werden.

6.2.3. Interpretation der Metriken

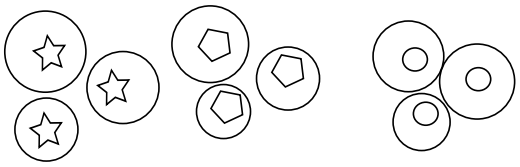
In der Abbildung 6.2 ist viermal der selbe Datensatz unterschiedlich geclustert worden. Beim optimalen Clustering in der Abbildung 6.2a nehmen beide Metriken den Höhstwert an. Im Gegensatz dazu wird in der Abbildung 6.2b einer der schlechtesten Fälle veranschaulicht, indem jede Kategorie in jedem Cluster einmal vorhanden ist. Dies spiegelt sich jedoch auch in beiden Metriken durch niedrige Werte wider. Die anderen beiden Abbildungen 6.2c und 6.2d stellen eine extreme Über- bzw. Unteranpassung beim Clustern dar. Dies kann allerdings nur durch die Betrachtung beider Metriken erkannt werden. Denn beispielsweise bei der Überanpassung ist die Reinheit optimal und erst durch den niedrigen NMI-Wert wird deutlich, dass der Clusterer überangepasst ist.



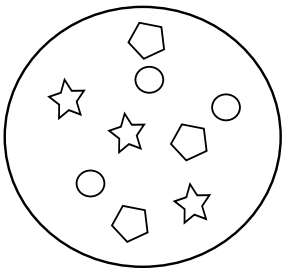
(a) Reinheit = $\frac{3+3+3}{9} = 1$; NMI = 1



(b) Reinheit = $\frac{1+1+1}{9} = 0.33$; NMI = 0



(c) Reinheit = $\frac{1+1+1+1+1+1+1+1+1}{9} = 1$; NMI = 0.5



(d) Reinheit = $\frac{1}{9} = 0.11$; NMI = 0.5

Abbildung 6.2.: Metriken des Clusteringalgorithmus: Analyse der Abhängigkeiten zwischen Reinheit und der normalisierten Transinformationsmetrik.

6.3. Metriken für die Evaluation des Klassifikators

Bei der Evaluation des Klassifikators werden nicht die gebildeten Cluster, sondern die resultierende Klassifikation betrachtet. Im Falle der Kreuzvalidierung führt dies dazu, dass in jedem Durchlauf und für jede Kategorie die beiden nachfolgenden Metriken berechnet werden. Anschließend werden die jeweiligen Werte pro Durchlauf für eine Gesamtauswertung gemittelt. Die Mittelung wird gewichtet vorgenommen, je nachdem wie viele Internetdienste einer Kategorie in einem Durchlauf geclustert wurden.

Für die Berechnung der beiden Metriken werden die Datenpunkte in vier Mengen eingeteilt. Ausgangspunkt der exemplarischen Mengenbeschreibung ist ein binärer Klassifikator, der zwischen Kategorie A und B unterscheidet. Möchte man nun die beiden Metriken für die Kategorie A bestimmen, dann ergeben sich folgende vier Mengen:

Richtig positiv(RP): Alle Datenpunkte der Kategorie A, die auch in die Kategorie A klassifiziert wurden.

Falsch positiv(FP): Alle Datenpunkte der Kategorie B, die fälschlicherweise der Kategorie A zugeordnet wurden.

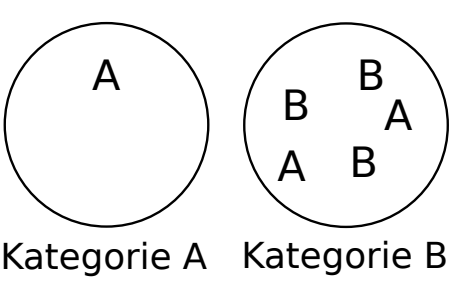
Richtig negativ(RN): Alle Datenpunkte der Kategorie B, die auch in diese Kategorie klassifiziert wurden.

Falsch negativ(FN): Alle Datenpunkte der Kategorie A, die fälschlicherweise der Kategorie B zugeordnet wurden.

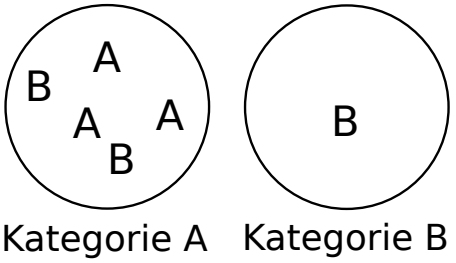
6.3.1. Präzision

Die Präzision gibt Auskunft darüber, wie viel Prozent der zur gleichen Kategorie klassifizierten Datenpunkte wirklich zu dieser Kategorie gehören.

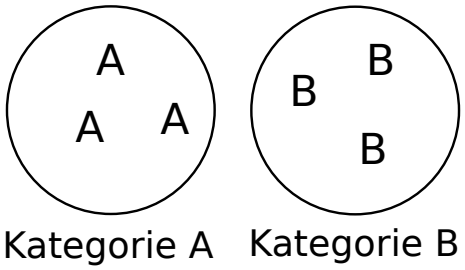
$$P = \frac{RP}{RP + FP} \tag{6.1}$$



(a) Präzision $= \frac{1}{1} = 1$; Ausbeute $= \frac{1}{1+2} = 0.33$



(b) Präzision $= \frac{3}{5} = 0.66$; Ausbeute $= 1$



(c) Präzision $= 1$; Ausbeute $= 1$

Abbildung 6.3.: Metriken des Klassifikators: Analyse der Abhängigkeiten zwischen Präzision und Ausbeute (Betrachtung der Kategorie A).

6.3.2. Ausbeute

Die Ausbeute bestimmt wie viel Prozent einer Kategorie erkannt wurden.

$$R = \frac{RP}{RP + FN} \tag{6.2}$$

6.3.3. Interpretation der Metriken

Betrachtet man ausschließlich eine der Metriken, gibt dies keine Auskunft über die Qualität des Klassifikators. Dies illustrieren die Beispiele der Abbildung 6.3. Die Abbildung 6.3a veranschaulicht beispielsweise eine qualitativ schlechte Klassifikation, trotzdem ist die Präzision für die Kategorie A bei 100%. Analog dazu kann in der Abbildung 6.3c erst durch die Betrachtung der Präzision die schlechte Qualität des Clusterings erkannt werden. Daraus folgt, dass erst bei hohen Werten beider Metriken auch das Clustering qualitativ höherwertig ist.

6.4. Evaluation des Clusterers

Für den *Spectral Clustering*- und *DBScan*-Algorithmus werden im Folgenden die Evaluation des Clusteringalgorithmus zur Bestimmung der Parametrisierung und anschließend des Klassifikators betrachtet. Dabei werden jeweils die Evaluationen anhand der Metriken bewertet und zum Schluss die Evaluationswerte beider Algorithmen miteinander verglichen.

6.4.1. DBScan

Der *DBScan*-Algorithmus clustert den Datensatz mittels der Dichte. Die Parameter für den Radius r und die Mindestanzahl an Datenpunkten pro Cluster k bestimmen die Zusammengehörigkeit eines Clusters. Je nach Parametrisierung werden somit unterschiedlich viele Cluster gebildet.

Evaluation des Clusteringalgorithmus

Die Abbildung 6.4 stellt für die beiden unabhängigen Parameter r und k die Auswertung der Metriken für die interne Evaluation dar. Diese beiden Parameter bilden zusammen die x-Achse. Da die drei Kurven unterschiedliche Skalen haben, ist auf der linken y-Achse die Skala von null bis eins für die beiden Metriken Reinheit und NMI und auf der rechten y-Achse die Skala für die Anzahl an Clustern dargestellt. Bei Betrachtung der Kurven ist zu erkennen, dass es sich im Gegensatz zum *Spectral Clustering*-Algorithmus jeweils um diskrete Funktionen handelt. Diese Diskretisierung entsteht durch die Anwendung des euklidischen Abstandsmaß in Verbindung mit den binären Merkmalen.

In der Abbildung ist zu erkennen, dass der Wert aller drei Kurven bei gleichem Radius und ansteigender Mindestanzahl k abnimmt. Für die Clusteranzahl ist dieses Verhalten erwünscht, jedoch nicht für die anderen beiden Metriken. Da die NMI-Metrik die Clusteranzahl und die Reinheit mit berücksichtigt, wird nach einer Parametrisierung gesucht, für die der NMI-Wert maximal wird. Dieses Maximum wird bei $k = 1$ und $r = 1$ erreicht. Somit erhält man einen NMI-Index von 0.66, der zu 74 Clustern mit einer Reinheit von 66% führt.

Betrachtet man den NMI-Index für verschiedene k, so ergibt sich jeweils für $r = 1$ das lokale Maximum. Dieses Maximum sinkt mit zunehmendem k, sowie auch die Werte für Reinheit und Clusteranzahl. Daraus folgt, dass zwar Cluster reduziert werden, aber wohl eher die Cluster mit einer hohen internen Reinheit entfernt werden. Der Vergleich der Kurven für $k = 1$ und $k = 2$ bei jeweils einem Radius von $r = 1$ bestätigt diese Annahme. Denn bei $k = 2$ entstehen 59 Cluster weniger als bei $k = 1$. Somit sind diese 59 Cluster alle jeweils Cluster mit nur einem Datenpunkt und deren Reinheit liegt folglich jeweils bei 100%.

Im Hinblick auf die Bewertung des Clusteringalgorithmus für den maximalen NMI-Index von 0.66 ist die Bildung von 74 Clustern im Grenzbereich für eine Überanpassung des Systems. Jedoch entsteht durch den zweithöchsten NMI-Index bereits ein unterangepasstes System, dass nur 15 Cluster erzeugt. Somit kann durch die Diskretisierung kein Kompromiss zwischen Über- und Unteranpassung gefunden werden. Dies spiegelt sich durch den relativ niedrigen maximalen NMI-Index von 0.66 wider.

Evaluation des Klassifikators

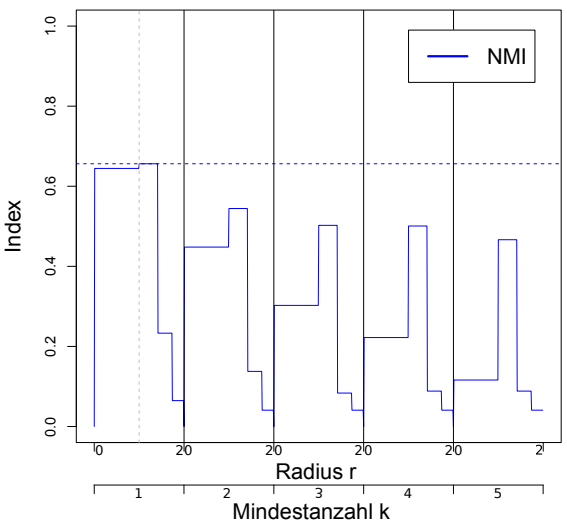
Bei der externen Evaluation werden nun über die 10-fache Kreuzvalidierung die Genauigkeit, Präzision und Ausbeute bei der Erkennung von unbekannten Internetdiensten bewertet. Da sich in jedem der 10 Durchläufe die Trainingsmenge ändert, wird die Parametrisierung des Algorithmus jeweils über eine interne Evaluation vorgenommen. Dies bedeutet, dass das Parameterpaar aus Radius r und Clusteranzahl k mit dem höhsten NMI-Index bestimmt wird.

Die Tabelle 6.3 führt die komprimierte Auswertung der Metriken für die gesamte Kreuzvalidierung auf. Die Metriken sind dabei für jede Kategorie bestimmt worden, sowie zusätzlich noch für die gesamte Klassifikation. Die detaillierte Auswertung der Metriken für jeden Durchlauf kann dem Abschnitt A entnommen werden.

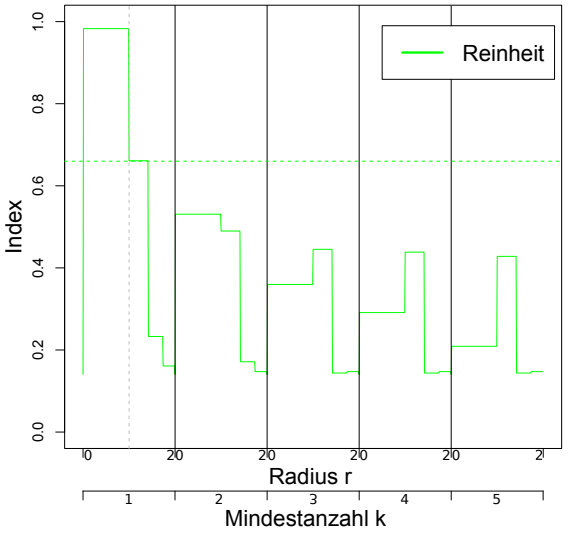
Die Auswertungen der Präzision und Ausbeute für die Kategorien des Analyse-Kapitels sind in der Tabelle 6.3 aufgeführt. Die Gesamtpräzision von 51% und die Gesamtausbeute von 45% sind für eine Klassifikation relativ niedrig und führen nur zu einer korrekten Klassifikation jedes zweiten Internetdienstes.

6.4.2. Spectral Clustering

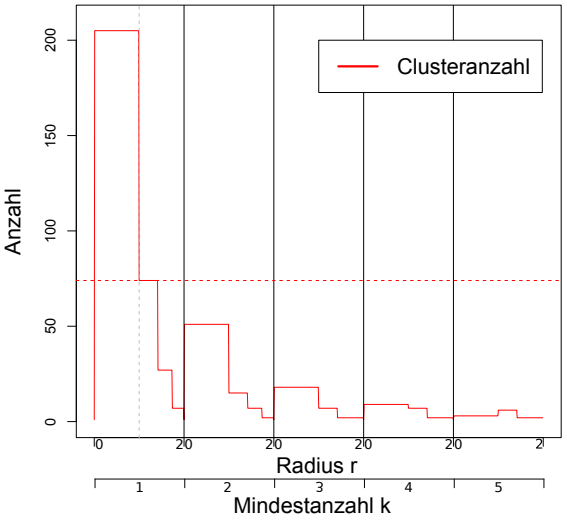
Der im Abschnitt 5.3.7 beschriebe *Spectral Clustering*-Algorithmus kann über den Parameter σ variiert werden.



(a) NMI



(b) Reinheit



(c) Anzahl an Clustern

Abbildung 6.4.: *DBScan*: Darstellung der Metriken für die interne Evaluation. Bei $k = 1, r = 1$ wird der NMI-Index maximal (NMI = 0.66). Dabei werden 74 Cluster gebildet, die zu 66% rein sind.

Internetdienstkategorien	**Anzahl**	**Präzision**	**Ausbeute**
Login	36	91,00%	78,00%
Registrierung	12	50,00%	50,00%
Kontaktformular	39	79,00%	64,00%
Fahrplanauskunft	41	56,00%	54,00%
Flugauskunft	23	39,00%	26,00%
Autovermietung	19	0,00%	0,00%
Unterkunftssuche	41	36,00%	27,00%
Newsletterabonnierung	24	29,00%	13,00%
Wettervorhersage	25	77,00%	88,00%
Gesamtauswertung	260	54,80%	47,44%

(a) Auswertung der analysierten Internetdienstkategorien

Internetdienstkategorien	**Anzahl**	**Präzision**	**Ausbeute**
Sprachauswahl	13	31,00%	23,00%
Mehrfache Auswahl	3	0,00%	0,00%
Buchung verwalten	2	0,00%	0,00%
Filterung	4	50,00%	50,00%
Ticketrechner	3	0,00%	0,00%
Suche	4	0,00%	0,00%
unbekannt	3	8,00%	66,00%
Gesamtauswertung	32	19,59%	21,78%

(b) Auswertung der analysierten Internetdienstkategorien

Tabelle 6.3.: Evaluation des Klassifikators: Auswertung der Präzision und Ausbeute für den *DBScan*-Algorithmus mittels der 10-fach Kreuzvalidierung.

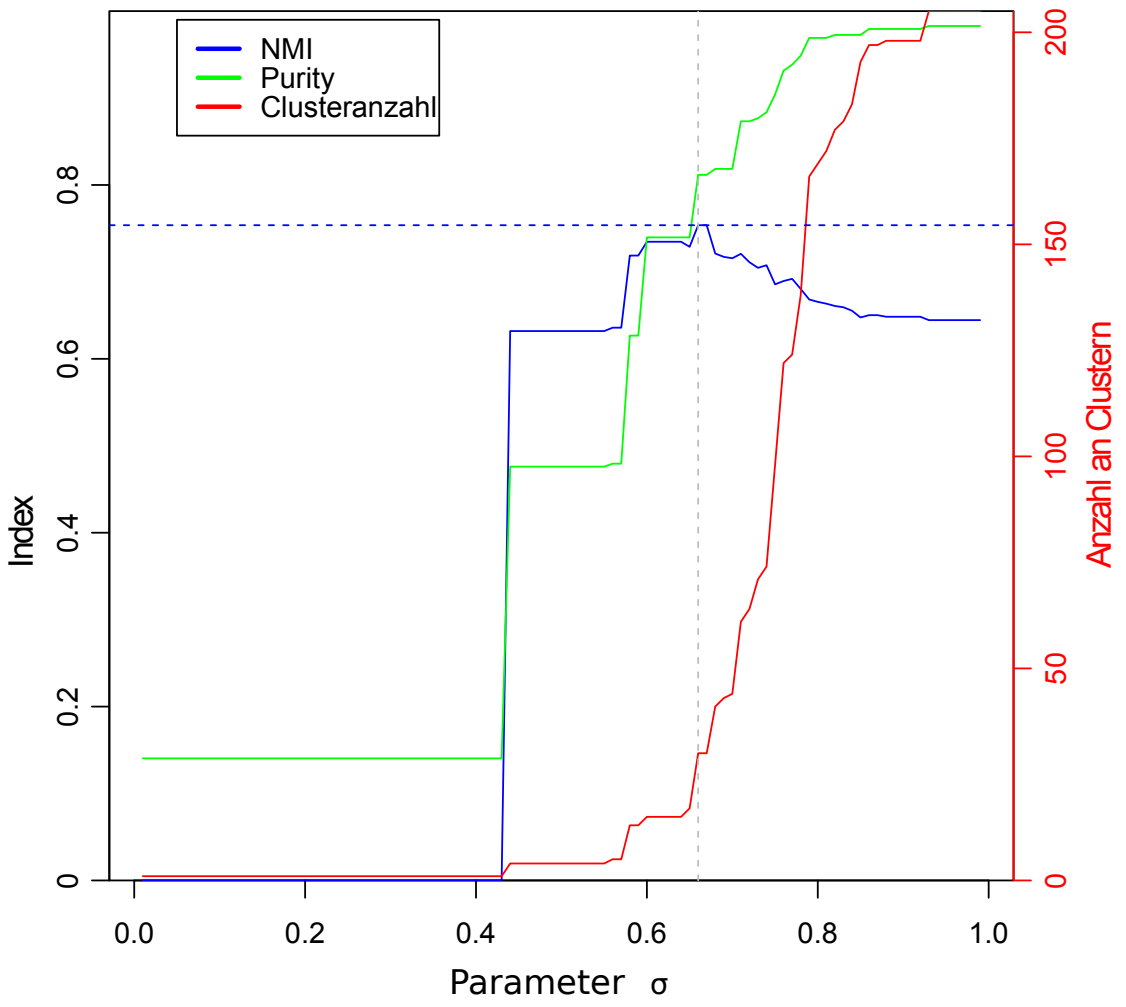


Abbildung 6.5.: *Spectral Clustering*: Darstellung der Metriken für die interne Evaluation. Bei $\sigma = 0.66$ wird der NMI maximal (NMI = 0.76) und es werden 30 Cluster gebildet, die zu 81% rein sind.

Evaluation des Clusteringalgorithmus

In der Abbildung 6.5 werden in Abhängigkeit der Parametrisierung, die Reinheit, der NMI, sowie die resultierende Clusteranzahl dargestellt. Beim Wert $\sigma = 0.66$ wird der NMI-Index maximal. Für höhere σ-Werte neigt das System zur Überanpassung, da die Clusteranzahl rasant steigt und für niedrigere Werte gehen durch die Vereinigung von Clustern Information verloren, was zu einer Unteranpassung führt. Für Parameter $\sigma = 0.66$ erhält man einen NMI-Index von 0.76 und 30 Cluster die zu 81% rein sind.

Bei einer Gesamtzahl von 18 verschiedenen Internetdienstkategorien, die der Clusterer anhand des Datensatzes kennt, ist die Bildung von 30 Clustern noch weit entfernt von einer Überanpassung. Zudem sprechen die beiden anderen Metriken mit Werten über 75% für eine gute interne Qualität.

Evaluation des Klassifikators

Für die externe Evaluation wird der *Spectral Clustering*-Algorithmus im Betrieb getestet. Die Betriebsphase wird durch die 10-fach Kreuzvalidierung simuliert. Um den Freiheitsgrad der Evaluation zu minimieren, wird der Parameter σ in jedem Durchlauf festgesetzt. Die

Internetdienstkategorien	Anzahl	Präzision	Ausbeute
Login	36	89,00%	94,00%
Registrierung	12	83,00%	75,00%
Kontaktformular	39	73,00%	87,00%
Fahrplanauskunft	41	81,00%	73,00%
Flugauskunft	23	84,00%	39,00%
Autovermietung	19	11,00%	5,00%
Unterkunftssuche	41	42,00%	66,00%
Newsletterabonnierung	24	68,00%	54,00%
Wettervorhersage	25	87,00%	100,00%
Gesamtauswertung	260	69,38%	69,86%

(a) Auswertung der analysierten Internetdienstkategorien

Internetdienstkategorien	Anzahl	Präzision	Ausbeute
Sprachauswahl	13	58,00%	62,00%
Mehrfache Auswahl	3	0,00%	0,00%
Buchung verwalten	2	0,00%	0,00%
Filterung	4	25,00%	25,00%
Ticketrechner	3	0,00%	0,00%
Suche	4	0,00%	0,00%
unbekannt	3	0,00%	0,00%
Gesamtauswertung	32	26,69%	28,31%

(b) Auswertung der nicht analysierten Internetdienstkategorien

Tabelle 6.4.: Evaluation des Klassifikators: Auswertung der Präzision und Ausbeute für den *Spectral Clustering*-Algorithmus mittels der 10-fachen Kreuzvalidierung.

Bestimmung des σ wird über die Auswertung des maximalen NMI-Index für die gegebene Trainingsmenge vorgenommen.

Die Tabelle 6.4 zeigt die Resultate der 10-fachen Kreuzvalidierung. Dabei werden bereits für die einzelnen Kategorien die über jeden Durchlauf gemittelte Präzision und Ausbeute aufgeführt. Zusätzlich wird die Gesamtpräzision bzw. -ausbeute anhand des Mittelwerts über die einzelnen Metrikenwerte bestimmt, wobei jeder Wert nach der Anzahl an Internetdiensten pro Kategorie gewichtet wird.

In dieser Tabelle sind die Kategorien, die nur einen Internetdienst enthalten, unter der Kategorie *unbekannt* zusammengefasst. Dies hat den Hintergrund, dass bei der Kreuzvalidierung dieser eine Internetdienst der Testmenge zugeteilt wird und somit dem Clusterer beim Training unbekannt ist. Dies führt dazu, dass der Clusterer diesen Dienst nicht erkennt. Somit gelten die Internetdienste, die eine eigene Kategorie bilden, als richtig klassifiziert, falls das System sie als unbekannt bzw. Ausreißer klassifiziert. Die detaillierte Auswertung der Metriken für jeden Durchlauf der Kreuzvalidierung können dem Abschnitt A entnommen werden.

Die Auswertung der Metriken der analysierten Internetdienstkategorien sind in der Tabelle 6.4a aufgeführt. Bei diesen Kategorien werden eine Präzision und Ausbeute von fast 70% erreicht. Daraus folgt, dass im Schnitt sieben von zehn Internetdiensten dieser Kategorien korrekt klassifiziert werden. Außerdem spricht die Ausgeglichenheit zwischen Präzision und Ausbeute für eine gute Parameterwahl des Clusterers. Für die übrigen Kategorien der Tabelle 6.4b ist die Erkennung sehr gering. Einzige Ausnahme ist die Sprachauswahl, da einerseits von dieser Kategorie zumindest 13 Internetdienste im Datensatz enthalten sind

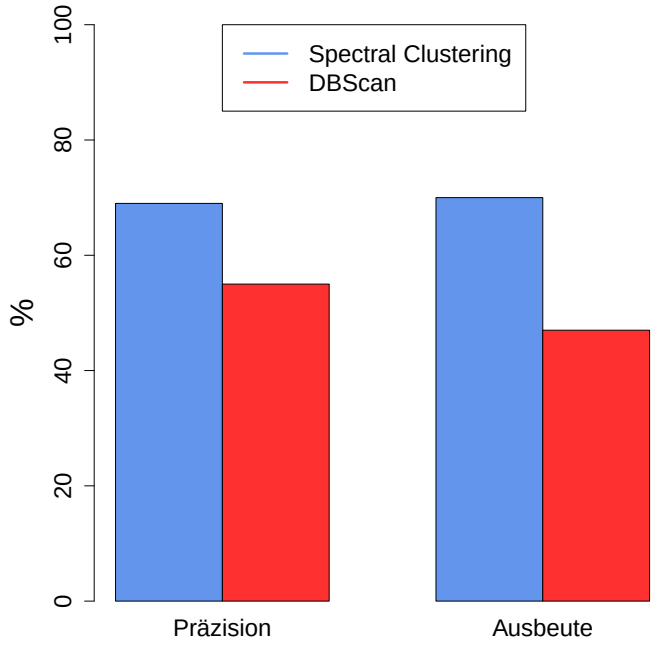


Abbildung 6.6.: Ergenis der Evaluation: Vergleich der Gesamtpräzision bzw. -ausbeute zwischen dem *Spectral Clustering*- und *DBScan*-Algorithmus.

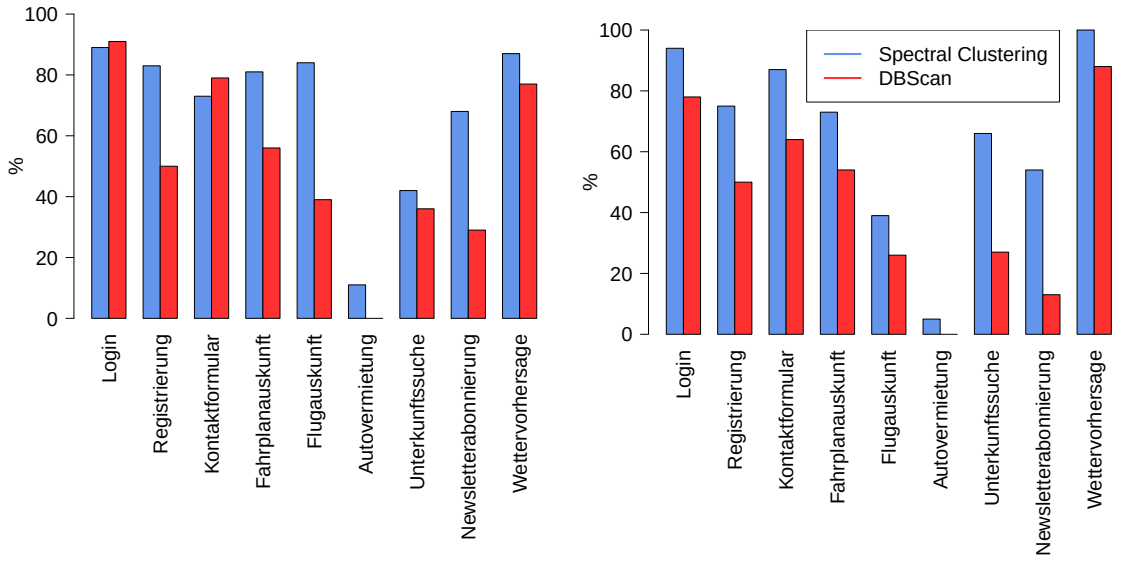
und andererseits die Dienste intern sich sehr ähnlich sind. Für die übrigen Kategorien gilt dies nicht. Die Internetdienste der Kategorien werden zwar aufgrund ihrer Semantik gruppiert jedoch unterscheiden sich die Implementierungen sehr stark, was dazu führt, dass die jeweiligen Internetdienste nicht erkannt werden.

6.5. Diskussion

Für der Bewertung der beiden Clusteringalgorithmen ist das Ergebnis des Clusterers entscheidend, daher werden die Werte der Metriken für die Evaluation des Klassifikators verglichen. In der Abbildung 6.6 sind in einem Säulendiagramm die Gesamtwerte der analysierten Kategorien im Bezug auf Präzision und Ausbeute je nach Algorithmus aufgeführt. Dabei ist deutlich zu erkennen, dass der *Spectral Clusteing* signifikant höhere Werte in beiden Metriken aufweist. Zudem bestätigt die Differenz zwischen Präzision und Ausbeute die Evaluation des *DBScan*-Algorithmus, denn die höhere Präzision ist ein Indiz für eine Überanpassung des Clusterers.

Betrachtet man nun die vorgenannte Zusammenstellung der Gesamtwerte detaillierter, dann werden große Unterschiede zwischen den einzelnen Kategorien deutlich. Die Abbildung 6.7 illustriert dazu die Metrikenwerte beider Algorithmen sortiert nach den einzelnen analysierten Kategorien. Dabei sind die beiden Diagramme, der Abbildung 6.7a für die Präzision und Abbildung 6.7b für die Ausbeute auf den ersten Blick ziemlich ähnlich. Ein signifikanter Unterschied zeigt sich im Wert des *Spectral Clustering*-Algorithmus für die Flugauskunfskategorie, der bei der Präzision einen viel höheren Wert als bei der Ausbeute annimmt. Dies hängt mit der Ähnlichkeit zwischen den Kategorien Flugauskunft und Autovermietung zusammen. Durch die Ähnlichkeit werden viele Internetdienste der Autovermietungskategorie von diesem Algorithmus als Flugauskunft klassifiziert. Dies führt zum einen dazu, dass die Ausbeute der Flugauskunftskategorie sinkt und zum anderen, dass für die Autovermietungskategorie sowohl die Ausbeute als auch die Präzision gegen null geht.

Abgesehen von den tendenziell niedrigeren Werten für den *DBScan*-Algorithmus, was bereits in den Gesamtwerten erarbeitet wurde, unterscheiden sich die Algorithmen erheb-



(a) Präzision

(b) Ausbeute

Abbildung 6.7.: Ergebnis der Evaluation: Vergleich der Metrikwerte für die Evaluation des Klassifikators zwischen dem *Spectral Clustering*- und *DBScan*-Algorithmus.

lich bei der Newsletterabonnierung. Ein Grund dafür ist die Ähnlichkeit zur Kategorie des Kontaktformulars. Der *DBScan*-Algorithmus neigt in diesem Fall dazu, Internetdienste der Newsletterabonnierungskategorie der Kategorie des Kontaktformulars zuzuordnen. Dies spiegelt sich erneut in der hohen Präzision für das Kontaktformular und der niedrigen Präzision und Ausbeute der Newsletterabonnierung wider.

Ein weiterer Faktor für die Senkung der Metrikenwerte ist, neben der Ähnlichkeit zweier Kategorien, die individuelle Implementierung eines Internetdiensts. Durch die Individualität entstehen viele zwar syntaktisch korrekte, aber semantisch nicht deutbare Internetdienste. Eine unsaubere Implementierung führt beim Clusterer dazu, dass die Semantik eines Feldes nicht richtig erkannt werden kann, was sich auf die Klassifikation auswirkt. Häufig tritt dieser Faktor bei autonom erzeugten HTML-Seiten auf..

Jedoch zeigt diese Auswertung auch, dass für manche Kategorien sehr hohe Erkennungsraten erzielt werden. Beispielsweise können für die Login- oder die Wettervorhersagekategorie Werte über 85% erreicht werden. Gründe dafür sind die Implementierungsähnlichkeit aller Internetdienste einer Kategorie, sowie die eindeutige Verwendung spezieller Formularelemente. Ein Beispiel dafür ist das Element zur Passworteingabe für einen Logindienst. Eine weitere Kategorie mit hohen Metrikwerten für den *Spectral Clustering*-Algorithmus ist die Fahrplanauskunft. Mit Werten über 80% für die Präzision und Ausbeute werden im Schnitt vier von fünf Internetdiensten dieser Kategorie korrekt klassifiziert.

Zusammenfassend betrachtet, liefert der *Spectral Clustering*-Algorithmus bis auf zwei kleine Ausnahmen für jede Kategorie in beiden Metriken stets höhere Werte.

6.6. Zusammenfassung

Bei der Evaluation der Clusteringalgorithmen wurde deutlich, dass der *Spectral Clustering*-Algorithmus bessere Werte für die Metriken liefert. Er klassifiziert im Schnitt 70% der Internetdienste korrekt, wohingegen der *DBScan*-Algorithmus der im Schnitt nur 50% korrekt klassifiziert. Diese Werte wurden über die externe Evaluation, die den Betrieb des

Clusterers simulieren soll, gemessen. Die dafür verwendeten Metriken sind die Präzision und die Ausbeute bei der Kassifizierung eines Internetdienstes zu einer Kategorie. Zusätzlich wurde noch die interne Qualität der Cluster über die interne Evaluation betrachtet. Anhand der Reinheit, dem NMI und der Clusteranzahl wurde für den *Spectral Clustering*-Algorithmus eine höhere interne Qualität ermittelt, die sich dann bei der externen Evaluation bestätigte. Außerdem diente die interne Evaluation zur Parametrisierung der Algorithmen, da sie das Verhalten des Clusterers in der Trainingsphase simuliert.

Die Evaluation offenbarte, dass die Klassifizierungsrate je nach Kategorie sehr schwankt. Ein Grund für die niedrige Erkennungsrate ist die Ähnlichkeit zwischen mehreren Kategorien, was zu einer falschen Zuordnung der Internetdienste führt. Mit der Weiterentwicklung des Semantik Web könnte diese Schwäche durch eine zuverlässigere Semantikerkennung bei der Merkmalserzeugung verbessert werden. Diese und weitere Ausblicke werden im nachfolgenden Kapitel betrachtet.

7. Zusammenfassung und Ausblick

Die Software-Ergonomie ist ein Teilgebiet der Informatik und arbeitet auf leicht verständliche und schnell benutzbare Softwaresysteme hin. Die Zielsetzung dieser Arbeit optimiert aus ergonomischer Sicht die Verwendung von Internetdiensten, indem der Nutzer durch die Formulierung seiner Anfrage den Dienst in Anspruch nimmt. Dies hat für den Nutzer den Vorteil, dass er ohne lästiges Einarbeiten in die Funktionsweise des bestehenden Softwareprogramms den Dienst über die natürliche Sprache bedienen und ausführen kann. Zu diesem Zweck wurden in dieser Arbeit die existierenden Internetdienste analysiert und ein System zur autonomen Klassifizierung der Dienste entwickelt. Durch die Klassifizierung und die damit verbundene Gruppierung von Internetdiensten kann für alle Internetdienste einer Kategorie eine gemeinsame aktive Ontologie entwickelt werden. Dazu wurde in dieser Arbeit ein Konstruktionsplan entworfen, der Regeln für das Ableiten der einzelnen Formularelemente, eines Internetdienstes und einer ganzen Internetdienstkategorie enthält. Somit kann auf Basis des Konstruktionsplans und der Analyseergebnisse der Internetdienstkategorien bereits manuell eine aktive Ontologie modelliert werden.

Zur Sammlung der Dienste wurde in dieser Arbeit ein Webcrawler entworfen, der auf der Startseite und den darin enthaltenen verlinkten Seiten nach Internetdiensten sucht. Auf der Grundlage dieser Sammlung wurden Merkmale zur Unterscheidung der Internetdienste erarbeitet, woraufhin folgende neun am häufigsten vorkommenden Kategorien von Internetdiensten ermittelt werden konnten:

1. Login
2. Registrierung
3. Kontaktformular
4. Fahrplanauskunft
5. Flugauskunft
6. Autovermietung
7. Unterkunftssuche
8. Newsletterabonnierung
9. Wettervorhersage

Zur Klassifikation der Internetdienste wurde ein Clusterer als maschinell lernendes System entworfen. Die Merkmale für die Unterscheidung der Kategorien gewinnt er autonom aus der Semantik der einzelnen Formularelemente, der Webseitenbeschreibung und der Häufigkeit bestimmter Elemente. Die Semantik wird dabei anhand bestimmter Attributwerte des jeweiligen Elements erlangt und in Form von Begriffslisten gespeichert. Bei speziellen Elementen wird auf die Speicherung dieser Begriffslisten verzichtet, da durch ihre spezielle Typisierung die Semantik bereits bekannt ist.

Für die Klassifikation der Internetdienste werden diese geclustert und anschließend jedes Cluster einer Kategorie zugeordnet. Für die Bildung der Cluster wurden der *Spectral*

Clustering- und *DBScan*-Algorithmus verwendet. Bei der Evaluation dieser Algorithmen stellte sich heraus, dass der *Spectral Clustering*-Algorithmus die analysierten Kategorien mit einer Präzision und Ausbeute von 69% erkennt. Bei detaillierter Betrachtung der Evaluationswerte hatten einige Kategorien, wie beispielsweise das Login und die Wettervorhersage, qualitativ hochwertige Erkennungsraten von fast 90%. Im Gegensatz dazu bereitete beispielsweise die Autovermietung aufgrund ihrer Ähnlichkeit zum Flugauskunftsdienst große Probleme bei der Klassifikation. Neben der Ähnlichkeit mancher Kategorien stellte die Individualität der Implementierungen die Semantikerkennung vor Schwierigkeiten, was wiederum zu einer sinkenden Erkennungsrate des Clusterers führte.

Zusätzlich wurden noch die Merkmale der gleich kategorisierten Internetdienste miteinander verglichen. Die daraus abgebildete Schnittmenge an Merkmalen ermöglicht in Verbindung mit den Ableitungsregeln des Konstruktionsplans die Erzeugung der kategoriebezogenen aktiven Ontologie.

7.1. Ausblick: Autonome Abbildung von Formularelementen auf Konzepte der aktiven Ontologie

Für die im Konstruktionsplan entworfene Abbildung der Formularelemente auf Konzepte der aktiven Ontologie kann ein Programm entwickelt werden, dass diese autonom erzeugt. Somit würde im Zusammenspiel mit dem in dieser Arbeit entwickelten Webcrawler und Clusterer eine autonome Aktivitätskette vom Sammeln der Internetdienste bis hin zur Erzeugung der jeweiligen aktiven Ontologie ausgeführt werden. Zur vollständigen Anwendbarkeit müsste noch ein Vermittler entwickelt werden, der die Schnittstelle für den Nutzer stellt und zwischen Nutzeranfrage und Internetdienstauswahl vermittelt.

7.2. Ausblick: Semantik Web

Die Entwicklung des Semantik Web wird immer stärker vorangetrieben. Die Grundidee ist dabei, dass Daten zusätzlich zu ihrer Information noch die Semantik der Information speichern. Ein Beispiel dafür ist der Begriff „Bank“. Der Mensch kann aus dem Kontext, in dem der Begriff fällt, erkennen, ob es sich um einen Gegenstand handelt auf dem man sitzen kann oder ob damit ein Geldinstitut gemeint ist. Das Computersystem hat jedoch schlechte kognitive Fähigkeiten und kann daher die Semantik nur schwer unterscheiden. Das Semantik Web liefert diese Unterscheidung und würde somit für den Clusterer zu einer deutlich leistungsstärkeren und zuverlässigeren Semantikerkennung führen.

Literatur

[AHS12] Yuan An, Xiaohua Hu und Il-Yeol Song. "Learning to discover complex mappings from web forms to ontologies". In: *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM. 2012, S. 1253–1262.

[BLMM+94] Tim Berners-Lee, Larry Masinter, Mark McCahill u. a. "Uniform resource locators (URL)". In: (1994).

[Gal+05] Avigdor Gal u. a. "Automatic ontology matching using application semantics". In: *AI magazine* 26.1 (2005), S. 21.

[GMJ04] Avigdor Gal, Giovanni Modica und Hasan Jamil. "Ontobuilder: Fully automatic extraction and consolidation of ontologies from web sources". In: *Data Engineering, 2004. Proceedings. 20th International Conference on*. IEEE. 2004, S. 853.

[Guz08] Didier Guzzoni. "Active: a unified platform for building intelligent applications". bibtex: Guzzoni2008. Diss. École Polytechnique Fédérale De Lausanne, 28. Jan. 2008. URL: `http://biblion.epfl.ch/EPFL/theses/2008/3990/3990_abs.pdf` (besucht am 22. 10. 2014).

[HN99] Allan Heydon und Marc Najork. "Mercator: A scalable, extensible web crawler". In: *World Wide Web* 2.4 (1999), S. 219–229.

[JBL08] Jae-Yoon Jung, Joonsoo Bae und Ling Liu. "Hierarchical business process clustering". In: *Services Computing, 2008. SCC'08. IEEE International Conference on*. Bd. 2. IEEE. 2008, S. 613–616.

[Lia+09] Qianhui Liang u. a. "Clustering web services for automatic categorization". In: *Services Computing, 2009. SCC'09. IEEE International Conference on*. IEEE. 2009, S. 380–387.

[Pag+99] Lawrence Page u. a. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report 1999-66. Previous number = SIDL-WP-1999-0120. Stanford InfoLab, Nov. 1999. URL: `http://ilpubs.stanford.edu:8090/422/`.

[Pin94] Brian Pinkerton. "Finding what people want: Experiences with the WebCrawler". In: *Proceedings of the Second International World Wide Web Conference*. Bd. 94. Chicago. 1994, S. 17–20.

[RD12] P Ravinder Reddy und A Damodaram. "Web services discovery based on semantic similarity clustering". In: *Software Engineering (CONSEG), 2012 CSI Sixth International Conference on*. IEEE. 2012, S. 1–7.

[SS02] Vladislav Shkapenyuk und Torsten Suel. "Design and implementation of a high-performance distributed web crawler". In: *Data Engineering, 2002. Proceedings. 18th International Conference on*. IEEE. 2002, S. 357–368.

[Zha+09] Xizhe Zhang u. a. “Web service community discovery based on spectrum clustering”. In: *Computational Intelligence and Security, 2009. CIS'09. International Conference on*. Bd. 2. IEEE. 2009, S. 187–191.

Anhang

A. Ergebnisse der 10-fachen Kreuzvalidierung

In den Tabellen A.1 und A.2 sind die detaillierten Ergebnisse der 10-fachen Kreuzvalidierung des *Spectral Clsutering*-Algorithmus. Für jeden Durchlauf und jede Kategorie sind der Wert der jeweiligen Metrik und die Anzahl der zu erkennenden Internetdienste aufgeführt. Zusätzlich wird der Gesamtwert der Metrik des jeweiligen Durchlaufs in der letzten Spalte und der jeweiligen Kategorie in der letzten Zeile aufgeführt. Der gleiche Aufbau gilt für die Tabellen A.3 und A.4, die die Ergebnisse der 10-fachen Kreuzvalidierung des *DBScan*-Algorithmus präsentieren.

Durchlauf	Präzision																Gesamt
	Login	Registrierung	Fahrplanauskunft	Wettervorhersage	Kontaktformular	Flugauskunft	Unterkunftssuche	Newsletterabonnierung	Autovermietung	Sprachauswahl	Mehrfache Auswahl	Buchung verwalten	Filterung	Ticketrechner	Suche	unbekannt	
1	60%	0%	80%	100%	60%	0%	100%	100%	0%	100%	-	-	-	-	0%	0%	**65 %**
Anzahl	3	1	4	3	4	2	4	3	2	2					1	1	**30**
2	100%	100%	100%	100%	100%	0%	60%	100%	0%	50%	-	-	-	-	-	0%	**73 %**
Anzahl	3	2	4	3	4	2	5	2	2	2						1	**30**
3	100%	100%	80%	43%	100%	50%	67%	33%	0%	0%	-	0%	-	-	-	-	**65 %**
Anzahl	4	2	4	3	4	2	4	2	1	2		1					**29**
4	67%	0%	75%	50%	25%	0%	36%	0%	0%	0%	0%	0%	0%	-	-	-	**31 %**
Anzahl	4	1	4	2	3	3	4	2	2	1	1	1	1				**29**
5	100%	100%	100%	67%	100%	0%	33%	100%	0%	100%	0%	-	0%	-	-	-	**64 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
6	100%	100%	75%	100%	100%	100%	80%	100%	100%	0%	0%	-	100%	-	-	-	**87 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
7	100%	100%	75%	100%	100%	100%	33%	100%	0%	100%	-	-	0%	0%	-	-	**73 %**
Anzahl	4	1	5	2	4	2	4	2	2	1			1	1			**29**
8	100%	100%	50%	100%	100%	50%	75%	100%	0%	100%	-	-	-	0%	0%	-	**72 %**
Anzahl	4	1	4	2	4	2	4	3	2	1				1	1		**29**
9	75%	100%	75%	100%	57%	100%	100%	0%	0%	100%	-	-	-	0%	0%	-	**64 %**
Anzahl	3	1	4	3	4	2	4	3	2	1				1	1		**29**
10	75%	100%	100%	100%	80%	33%	100%	100%	0%	50%	-	-	-	-	0%	0%	**75 %**
Anzahl	3	1	4	3	4	2	4	3	2	1					1	1	**29**
Gesamt:	89%	83%	81%	87%	79%	84%	42%	68%	11%	58%	0%	0%	25%	0%	0%	0%	**65 %**
	36	12	41	25	39	23	41	24	19	13	3	2	4	3	4	3	**292**

Tabelle A.1.: *Spectral Clustering*: Werte für die Präzision der 10-fachen Kreuzvalidierung.

Durchlauf	Ausbeute																Gesamt
	Login	Registrierung	Fahrplanauskunft	Wettervorhersage	Kontaktformular	Flugauskunft	Unterkunftssuche	Newsletterabonnierung	Autovermietung	Sprachauswahl	Mehrfache Auswahl	Buchung verwalten	Filterung	Ticketrechner	Suche	unbekannt	
1	100%	0%	100%	100%	75%	0%	50%	67%	0%	100%	-	-	-	-	0%	0%	**63 %**
Anzahl	3	1	4	3	4	2	4	3	2	2					1	1	**30**
2	100%	50%	100%	100%	100%	0%	60%	50%	0%	50%	-	-	-	-	-	0%	**67 %**
Anzahl	3	2	4	3	4	2	5	2	2	2						1	**30**
3	75%	100%	100%	100%	75%	50%	50%	50%	0%	0%	-	0%	-	-	-	-	**66 %**
Anzahl	4	2	4	3	4	2	4	2	1	2		1					**29**
4	100%	0%	75%	100%	33%	0%	100%	0%	0%	0%	0%	0%	0%	-	-	-	**48 %**
Anzahl	4	1	4	2	3	3	4	2	2	1	1	1	1				**29**
5	100%	100%	50%	100%	100%	0%	100%	100%	0%	100%	0%	-	0%	-	-	-	**69 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
6	100%	100%	75%	100%	75%	33%	100%	50%	50%	0%	0%	-	100%	-	-	-	**72 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
7	100%	100%	60%	100%	100%	50%	25%	100%	0%	100%	-	-	0%	0%	-	-	**66 %**
Anzahl	4	1	5	2	4	2	4	2	2	1			1	1			**29**
8	75%	100%	25%	100%	100%	100%	75%	67%	0%	100%	-	-	-	0%	0%	-	**66 %**
Anzahl	4	1	4	2	4	2	4	3	2	1				1	1		**29**
9	100%	100%	75%	100%	100%	100%	50%	0%	0%	100%	-	-	-	0%	0%	-	**66 %**
Anzahl	3	1	4	3	4	2	4	3	2	1				1	1		**29**
10	100%	100%	75%	100%	100%	100%	50%	67%	0%	100%	-	-	-	-	0%	0%	**72 %**
Anzahl	3	1	4	3	4	2	4	3	2	1					1	1	**29**
Gesamt:	94%	75%	73%	100%	87%	39%	66%	54%	5%	62%	0%	0%	25%	0%	0%	0%	**65 %**
	36	12	41	25	39	23	41	24	19	13	3	2	4	3	4	3	**292**

Tabelle A.2.: *Spectral Clustering*: Werte für die Ausbeute der 10-fachen Kreuzvalidierung.

Durchlauf	Präzision																Gesamt
	Login	Registrierung	Fahrplanauskunft	Wettervorhersage	Kontaktformular	Flugauskunft	Unterkunftssuche	Newsletterabonnierung	Autovermietung	Sprachauswahl	Mehrfache Auswahl	Buchung verwalten	Filterung	Ticketrechner	Suche	unbekannt	
1	100%	0%	75%	60%	57%	50%	100%	0%	0%	0%	-	-	-	-	0%	14%	**51 %**
Anzahl	3	1	4	3	4	2	4	3	2	2					1	1	**30**
2	60%	0%	0%	75%	75%	0%	25%	0%	0%	0%	-	-	-	-	-	11%	**28 %**
Anzahl	3	2	4	3	4	2	5	2	2	2						1	**30**
3	100%	100%	60%	60%	80%	0%	50%	0%	0%	100%	-	0%	-	-	-	-	**60 %**
Anzahl	4	2	4	3	4	2	4	2	1	2		1					**29**
4	100%	0%	67%	100%	100%	0%	0%	0%	0%	100%	0%	0%	100%	-	-	-	**47 %**
Anzahl	4	1	4	2	3	3	4	2	2	1	1	1	1				**29**
5	100%	100%	100%	100%	100%	100%	100%	0%	0%	0%	0%	-	100%	-	-	-	**79 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
6	100%	100%	100%	100%	100%	0%	100%	100%	0%	0%	0%	-	0%	-	-	-	**72 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
7	100%	100%	75%	100%	100%	100%	0%	100%	0%	100%	-	-	0%	0%	-	-	**68 %**
Anzahl	4	1	5	2	4	2	4	2	2	1			1	1			**29**
8	100%	100%	0%	100%	100%	50%	67%	100%	0%	0%	-	-	-	0%	0%	-	**61 %**
Anzahl	4	1	4	2	4	2	4	3	2	1				1	1		**29**
9	75%	0%	38%	75%	40%	100%	0%	0%	0%	0%	-	-	-	0%	0%	-	**33 %**
Anzahl	3	1	4	3	4	2	4	3	2	1				1	1		**29**
10	60%	0%	44%	33%	40%	0%	100%	0%	0%	0%	-	-	-	-	0%	0%	**35 %**
Anzahl	3	1	4	3	4	2	4	3	2	1					1	1	**29**
Gesamt:	91%	50%	56%	77%	79%	39%	36%	29%	0%	31%	0%	0%	50%	0%	0%	8%	**51 %**
	36	12	41	25	39	23	41	24	19	13	3	2	4	3	4	3	**292**

Tabelle A.3.: *DBScan*: Werte für die Präzision der 10-fachen Kreuzvalidierung.

Durchlauf	Ausbeute																Gesamt
	Login	Registrierung	Fahrplanauskunft	Wettervorhersage	Kontaktformular	Flugauskunft	Unterkunftssuche	Newsletterabonnierung	Autovermietung	Sprachauswahl	Mehrfache Auswahl	Buchung verwalten	Filterung	Ticketrechner	Suche	unbekannt	
1	100%	0%	75%	100%	100%	50%	50%	0%	0%	0%	-	-	-	-	0%	100%	**57 %**
Anzahl	3	1	4	3	4	2	4	3	2	2					1	1	**30**
2	100%	0%	0%	100%	75%	0%	40%	0%	0%	0%	-	-	-	-	-	100%	**40 %**
Anzahl	3	2	4	3	4	2	5	2	2	2						1	**30**
3	75%	100%	75%	100%	100%	0%	25%	0%	0%	50%	-	0%	-	-	-	-	**59 %**
Anzahl	4	2	4	3	4	2	4	2	1	2		1					**29**
4	50%	0%	50%	100%	33%	0%	0%	0%	0%	100%	0%	0%	100%	-	-	-	**31 %**
Anzahl	4	1	4	2	3	3	4	2	2	1	1	1	1				**29**
5	50%	100%	25%	100%	75%	33%	50%	0%	0%	0%	0%	-	100%	-	-	-	**45 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
6	75%	100%	75%	100%	25%	0%	25%	50%	0%	0%	0%	-	0%	-	-	-	**41 %**
Anzahl	4	1	4	2	4	3	4	2	2	1	1		1				**29**
7	100%	100%	60%	100%	50%	50%	0%	50%	0%	100%	-	-	0%	0%	-	-	**52 %**
Anzahl	4	1	5	2	4	2	4	2	2	1			1	1			**29**
8	50%	100%	0%	50%	75%	100%	50%	33%	0%	0%	-	-	-	0%	0%	-	**41 %**
Anzahl	4	1	4	2	4	2	4	3	2	1				1	1		**29**
9	100%	0%	75%	100%	50%	50%	0%	0%	0%	0%	-	-	-	0%	0%	-	**41 %**
Anzahl	3	1	4	3	4	2	4	3	2	1				1	1		**29**
10	100%	0%	100%	33%	50%	0%	25%	0%	0%	0%	-	-	-	-	0%	0%	**38 %**
Anzahl	3	1	4	3	4	2	4	3	2	1					1	1	**29**
Gesamt:	78%	50%	54%	88%	64%	26%	27%	13%	0%	23%	0%	0%	50%	0%	0%	66%	**45 %**
	36	12	41	25	39	23	41	24	19	13	3	2	4	3	4	3	**292**

Tabelle A.4.: *DBScan*: Werte für die Ausbeute der 10-fachen Kreuzvalidierung.