Hochschule **RheinMain**
Fachbereich Design Informatik Medien
Studiengang Medieninformatik

Abschlussarbeit

zur Erlangung des akademischen Grades

Master of Science

Data Mining komplexer Datenstrukturen aus PDF-Dokumenten

Vorgelegt von	Deniz Aydar
am	25. Juli 2022
Referent	Prof. Dr. Dirk Krechel
Korreferent	Prof. Dr. Philipp Schaible

Erklärung gemäß ABPO

Ich erkläre hiermit, dass ich

- die vorliegende Abschlussarbeit selbstständig angefertigt,
- keine anderen als die angegebenen Quellen benutzt,
- die wörtlich oder dem Inhalt nach aus fremden Arbeiten entnommenen Stellen, bildlichen Darstellungen und dergleichen als solche genau kenntlich gemacht und
- keine unerlaubte fremde Hilfe in Anspruch genommen habe.

Wiesbaden, 25. Juli 2022 _____

Deniz Aydar

Erklärung zur Verwendung der Masterthesis

Hiermit erkläre ich mein Einverständnis mit den im folgenden aufgeführten Verbreitungsformen dieser Abschlussarbeit:

Verbreitungsform	**Ja**	**Nein**
Einstellung der Arbeit in die Hochschulbibliothek mit Datenträger		×
Einstellung der Arbeit in die Hochschulbibliothek ohne Datenträger		×
Veröffentlichung des Titels der Arbeit im Internet	×	
Veröffentlichung der Arbeit im Internet		×

Wiesbaden, 25. Juli 2022 _____

Deniz Aydar

Zusammenfassung

PDF-Dokumente besitzen viele Informationen, aus denen sich neue Daten generieren lassen können. Doch das Extrahieren von solchen Daten ist heutzutage immer noch mit Hürden verbunden. Dies gilt auch für die Dokumente, die in den Prozessen von Autohäusern und deren Kfz-Werkstätten verwendet und anschließend gelagert werden. Um die Frage zu beantworten, inwiefern neue Daten aus dieser Art von Dokumenten verarbeitet werden können, wird im Rahmen dieser Arbeit ein System konzipiert und entwickelt, welches es ermöglichen soll, weiteres Wissen zu gestalten beziehungsweise zu generieren. Dieses System will dabei Ansätze aus der künstlichen Intelligenz nutzen und eine grafische Schnittstelle für die Möglichkeit der menschlichen Nutzung anbieten.

Inhaltsverzeichnis

Kapitel 1

Einleitung

„Digitalisierung im Alltag voranbringen“ – Das war einer der Wahlslogans während der Bundestagswahl 2021. Gemeint war damit die Forderung nach einer zunehmende Digitalisierung im privaten Alltag vieler Bürger:innen, aber auch in der Wirtschaft machte sich wachsend der Wunsch nach mehr digitalen Alternativen breit (vgl. Bundesregierung 2021). Dieser Wunsch beinhaltete vor allem einen Wechsel von gängigen Papierformen verschiedenster Dokumente hin zu denselben in digitaler Ausprägung.

Genau jenes Bedürfnis nach Digitalisierung betrifft auch die vielen Arbeitnehmer:innen und Händler:innen in Autohäusern und deren Kfz-Werkstätten, die durch ihre beruflichen Tätigkeiten mit einer Vielzahl an Unterlagen, Dokumenten oder Belegen arbeiten müssen. Unter dieser Vielzahl fallen Dokumente wie Werkstatt-, Kauf- und Mietverträge sowie Rechnungen oder Diagnoseberichte.

Jene Beschäftigte in einigen Autohäusern und deren Kfz-Werkstätten sind Kund:innen bei ilexius GmbH. Ilexius bietet Enterprise-Resource-Planning (ERP) Systeme an, mit Hilfe derer Arbeitsaufträge und Arbeitsschritte innerhalb des täglichen Arbeitsprozesses in Autohäusern und deren Werkstätten durch Digitalisierung vereinfacht werden. In Kooperation von ilexius GmbH werde ich daher die Problematik meiner Thesis, die ich im folgenden noch genauer beschreiben werde, lösen und in die Tat umsetzen.

Durch eine Digitalisierung jener Dokumente eröffnen sich Vorteile wie bessere Zugänglichkeit, größere Langlebigkeit und vor allem eine angepasste und leichtere Nutzung, die zu einer höheren Effektivität innerhalb täglicher Arbeitsschritte führt. Durch bisherige erste Schritte der Digitalisierung sind diese benötigten Dokumente bereits elektronisch aufgearbeitet und den Beschäftigten in Autohäusern und deren Kfz-Werkstätten zur Verfügung gestellt worden.

Die Folge dessen ist, dass alle Dokumente standardisiert sind und dadurch die in den Dokumenten beinhalteten Informationen neu verarbeitet werden können. Eine Extraktion der Daten der einzelnen Dokumente ist jedoch nur eingeschränkt möglich, da die Struktur im gängigen Gebrauch im Dateiformat Portable Document Format (PDF) festgelegt ist.

Jene Limitierung der Datenextraktion wirkt sich gleichermaßen auf die Dealer-Management-Systeme (DMS), mit solchen die Mitarbeiter:innen ihre Prozesse abwickeln, aus. Der daraus resultierende Arbeitsaufwand, welcher dabei entsteht, um die Daten wiederverwendbar zu konstruieren, ist derzeit enorm.

1.1 Zielsetzung

Aus diesem Grund möchte ich innerhalb dieser Master-Thesis ein System entwickeln, das genau jene Problematik erleichtert und löst. Innerhalb üblicher Methoden werden heutzutage die Daten zunächst über eine grafische Anzeige mithilfe einer bereits existierenden Benutzeranwendung via gängiges Kopieren und Einfügen entnommen. Dieser Schritt funktioniert zwar im Regelfall, ist jedoch oftmals stark fehleranfällig und kann lückenhaft sein.

In so einem Fall muss das Einfügen und Kopieren manuell durchgeführt werden, was bei einer riesigen Menge an Dokumente zu einer monotonen Arbeit für die Händler der Autohäuser führt. Andernfalls können die Daten eigenhändig abgeschrieben werden – jener Vorgang benötigt jedoch viele Ressourcen. Eher geeignet ist es, einen Datenkonverter für die PDF Dateien zu nutzen, um so die Dokumente beispielsweise in Bilder umzuwandeln, wodurch die Daten zugänglicher sind, aber immer noch verarbeitet werden müssen. Die letztgenannte Möglichkeit wirkt allerdings eher wie eine Übergangslösung als eine professionelle endgültige Durchführung.

Aufgrund dieser bisherigen teils aufwendigen und mit Fehlern verbundenen Möglichkeiten möchte ich mich in meiner Thesis von diesen Optionen abwenden und das Data-Mining nutzen. Das Data-Mining soll eine Automatisierung unterstützen, mit der Daten aus dem PDF-Dokument extrahiert werden. Allein über das Data-Mining ist es möglich, Inhalte zu extrahieren und für andere Systeme bereit zu stellen.

Der Unterschied zwischen meiner Verwendung des Data-Minings und zum gängigen Data-Mining liegt dabei in den Einschränkungen des Dateiformats: der Quelltext einer solchen Datei stellt erstens keine klare Hierarchie der Daten dar und zweitens besitzt es durch das Fehlen von Markierungen keine Informationen darüber, was die Daten an sich darstellen sollen [17].

Mit Hilfe dieser Technik des Data-Minings soll es ermöglicht werden, aktuelle Prozesse detaillierter zu steuern und zu überwachen. Des Weiteren können unter anderem Abgleiche von Rechnungen mit dem System durchgeführt werden und weitere Datenanalysen und Reporting kreiert werden. Mit dem derzeitigen Stand der Datenextraktion können jedoch lediglich Agierende aus dem technischen Bereich arbeiten, da allein sie das dafür technisch notwendige Know-how besitzen. Die Technologien hierfür benutzen unterschiedliche Ansätze und müssen daher zunächst für diesen Fall ausgewählt werden.

Damit jedoch auch für Agierende innerhalb des technischen Bereichs die Nutzung nicht zu abstrakt bleibt, soll durch die Entwicklung eines Systems der Zugriff greifbarer gestaltet werden, welches wiederum allein durch eine benutzerfreundliche Anwendung ermöglicht wird. Die Benutzer sollen eine Möglichkeit bekommen und über eine Eingabe der Steuerung bestimmen können, welche Informationen extrahiert werden sollen.

Darüber hinaus muss eine Automatisierung vorhanden sein, um auch eine große Datenmenge verarbeiten zu können; zudem sollen die Vorgänge eine niedrigere Fehlerquote aufzeigen. Die Prozesse müssen außerdem während des gesamten Ablaufs über eine Steuerung und Regelung kontrollierbar sein. Solche Prozesse können zum Beispiel über Regeln festgelegt beziehungsweise gesteuert werden, die hingehen bei einer Erfüllung weitere Aktionen oder Regeln auslösen können.

Diese im vorherigen genannten Absatz beschriebene Eigenschaft kann durch eine symbolische künstlichen Intelligenz (KI) abgedeckt werden, welche eine vorgegebene Verarbeitung zulässt. Hiermit bezeichnet man einen altmodischen Ansatz der KI, bei der über das Festlegen von Symbolen menschliches Wissen in einer Logik gehalten wird und diese benutzt wird um weiteres Wissen sodann zu generieren [16]. Durch eine Parametrisierung einer solchen KI kann dann die Unschärfe der ausgewählten Daten festgelegt werden, damit eine Feineinstellung erfolgen kann. Das heißt, die Benutzer:innen eines solchen Systems soll sich beginnend von groben Definitionen für die Verarbeitung zu einer detaillierten und angepassteren Definition über Feedback des Systems hinarbeiten.

Mit Hilfe dieser Annäherung an Definitionen und Regeln kann so für eine bessere Erfolgsquote gesorgt werden. Zudem kann der Ansatz des Machine Learnings (ML), mit jenem erfolgreiche Verarbeitungen einem System antrainiert werden, weitere Dokumente aus Daten extrahieren. Hierbei ist noch offen, welcher Typ des Machine Learnings sich für diesen Fall am optimalsten eignet.

Beide Ansätze – der der symbolischen KI und der des Machine Learnings – bieten als Option die Entwicklung einer grafischen interaktiven Entwicklungsumgebung (IDE) an. Die IDE soll das Festlegen von Definitionen und Regeln

erlauben, mit welchen die Dokumente verarbeitet werden können. Außerdem soll diese Oberfläche verschiedene Funktionalitäten offerieren und auch Feedback sowohl bei einem problemlosen Lauf, als auch bei einem fehlerhaften Lauf, zurückgeben.

Des Weiteren soll es das System es ermöglichen, Änderungen der Definitionen anzumerken und Unterschiede zu erstellen. Mit dieser Funktion sollen auch bisherige Ergebnisse angezeigt werden. Die Festlegung von Definitionen und Regeln soll außerdem durch eine Ansicht der Dokumente unterstützt werden. Insgemein wird das System die zwei Ansätze als Subsysteme aufteilen um die Umsetzung zu modularisieren und einen Vergleich zu ermöglichen.

1.2 Ablauf

Um genau dieses optimale System für das beschriebene Szenario entwickeln zu können, werde ich in dieser Arbeit wie folgt vorgehen:

Für eine vollständige Auseinandersetzung soll eine weitere Analyse stattfinden, damit die vorhandenen Grundlagen für den Anwendungsfall, um den es in der Arbeit geht, evaluiert werden können. Das heißt, es werden mit Hilfe der Funktionalitäten der Bibliotheken Ergebnisse auf Basis der Datensätze erzeugt, um die Erfolgsquoten zu überprüfen. Diese werden dann verglichen und ausgewertet. Parallel dazu soll die Umgebung für die Entwicklung eingerichtet werden, mit der die Implementierung der Systeme stattfinden soll.

Danach widme ich mich der Konzeption und der Entwicklung des Systems. Diese beginnt mit der symbolischen KI als ein Subsystem des gesamtem Projekts, durch selbige ein direkterer Ansatz ermöglicht wird. Die Entwicklung hierbei wird im Wasserfall-Ansatz, einem Ansatz bei der phasenweise die Eigenschaften einer solchen KI umgesetzt werden, um eine Grundlage für die nächste Komponente zur Verfügung zu stellen.

Bei dieser Komponente handelt es sich um die interaktive Entwicklungsumgebung, die den Zugang für den Benutzer zum System darstellt. Deswegen folgt im nächsten Schritt die Konzeptionierung und Entwicklung der interaktiven Entwicklungsumgebung, welche sich während der Entwicklung der symbolischen KI parallelisieren lässt. Sobald das Gerüst der Benutzeroberfläche fertiggestellt ist, möchte ich dies mit der symbolischen KI verbinden.

Als Gegenstück wird sich dann mit dem Machine Learning Ansatz und dem dazugehörigen Subsystem, bei der gleichermaßen eine Konzeptionierung und Entwicklung stattfindet, gewidmet. Hier wird das bisherige Gesamtkonstrukt

in einem iterativen Ansatz umgesetzt, sodass der Hauptteil so früh wie möglich verfügbar ist, damit das Training des ML-Systems durch die Dokumente auch zeitnah starten kann. Auf das Auswerten dieses Subsystems folgt eine Anpassung dessen. Die Komponenten der symbolischen KI und des Machine Learnings erlauben damit einen Vergleich zwischen zwei verschiedene Ansätze aus der Künstlichen Intelligenz.

Danach wird das Software Testing für die beiden Subsysteme eingeführt. Mit Unit Testing, Integration Testing und Functional Testing wird das erfolgreiche Ausführen gewisser Eigenschaften abgedeckt. Dies dient wiederum als Grundlage dafür, die Continuous Integration (CI) für das System einzuführen, mit dieser eine sichere und konsistente Entwicklung nach dem Prototyping angeboten wird.

Wenn hierauf die Systeme gegeben sind, kann das grafische Tool weiter angepasst werden, welches sich dann mit den Systemen verbinden soll, um infolgedessen die Ausführbarkeit der Funktionalität überprüfen zu können. Um ein gemeinsames System zu ermöglichen, welches dann gegebenenfalls weitere Testschritte bedarf, werden anschließend die Komponenten zusammengeführt. Sobald jenes eine erfolgreiche Form annimmt, wird der Teil der Continuous Integration mit einbezogen, mit jener das System einen Zustand der Instandhaltung annehmen kann. Über Testing sollen so mögliche Fehlerquellen entdeckt werden, bevor diese überhaupt auftreten können.

Unter die Zielsetzung für meine beiden eben genannten Ansätze fallen die Kundendaten mit den Eigenschaften wie Name, Firma, Adresse und Kundennummer. Zu Eigenschaften der Fahrzeugdaten zählen Angaben der Fahrgestellnummer, Modell, Farbe, Motor, Erstzulassung, Letzter/nächster Service und TÜV sowie die Abrechnungsdaten wie die Jobs mit den dazugehörigen Arbeitspositionen, der Reparaturteile, den Preisen etc.

Dazu stehen als weitere Grundlage mehr als 10.000 annotierte unterschiedliche Dokumente mit genau den Daten, die extrahiert werden sollen, als Datenbank zur Verfügung. In Zukunft sollen für das Mengengerüst mehr als 100.000 Dokumente pro Jahr bearbeitet werden. Hierbei ist es aber auch möglich, aus den bereits bestehenden Daten einen Pool für den Lernprozess bzw. für die Verifikation zu bilden.

Zum Schluss werde ich die Ergebnisse des gesamten Systems betrachten und einen Ausblick zu der Thematik geben. Aus Datenschutzgründen werde ich in meiner Arbeit persönliche Informationen in den gezeigten Dokumenten anonymisieren und verpixeln.

Kapitel 2

Hintergrund

Um die Problematik detaillierter darzustellen und um eine Übersicht zu ermöglichen, wird in diesem Kapitel der Rahmen der Problematik aus technischer Sicht genauer dargestellt. Hierbei werden die möglichen Technologien abgewogen und evaluiert. Zunächst jedoch wird der Kontext ausführlicher erklärt.

2.1 Kontext

Im Automobilbereich wird der Verkauf von Kraftfahrzeugen in zwei Segmente aufgeteilt: dem Sales-Bereich, bei dem es sich um den Verkauf von Neu- und Gebrauchtwagen handelt und dem After-Sales-Bereich, bei jenem eine Bindung zum Kunden über den Verkauf von Verschleiß- und Ersatzteilen geschaffen wird. [32] Diese Aufteilung der Segmente wird ebenfalls von den Autohäusern angewendet, zugleich werden für diese Aufteilung jeweils gängige Abläufe als Prozesse benutzt.

Für den After-Sales-Bereich bietet ilexius GmbH zwei Arten von ERP-Systeme an. Die erste Art ist das von der Automobilmarke Jaguar Land Rover finanzierte ‘vTab’, welches eine Plattform für elektronische Fahrzeugkontrollen und ein Backend für Jaguar Land Rover Mitarbeiter:innen, mit dem die Autohäuser bewertet werden können, anbietet. Die zweite Art mit dem Namen ‘ATT’ ist eine Plattform. Mit derjenigen können Autohäuser ihre Arbeitsprozesse in Schrift, Sprache und Bild dokumentieren. Die Dokumentation erfolgt mittels mobiler Endgeräte und werden an das ERP-System weitergeleitet, sodass die Dokumentation archiviert werden kann.

Die ERP-Systeme, die bereits im Kapitel 1.1 beschrieben worden sind, unterstützen damit die Kunden:innen von ilexius GmbH in den Autohäusern und

deren Werkstätten bei der Planung, der Steuerung sowie der Verwaltung von verschiedenen Aufgaben in ebenjenem After-Sales-Bereich.

Eine Art der Aufgaben sind Arbeitsaufträge, bei der es sich in den meisten Fällen um die Handhabung und Abfertigung eines Autos inklusive Kundendaten handelt. Der Auftrag besteht hierbei aus einer Liste an Arbeitsabläufen, die als Jobs bezeichnet werden. Die Jobs wiederum unterteilen sich in einzelne Arbeitsschritte, die Joblines genannt werden, welche die konkreten Verrichtungen der Arbeit darstellen und somit den gesamten Arbeitsauftrag in einzelne Teilschritte vervollständigen.

Sind diese Aufträge abgeschlossen, benutzen die Mitarbeiter:innen der Autohäuser ihre Dealer-Management-Systeme, um eine Rechnung zu erstellen und diese im ERP-System zu archivieren. Das ERP bietet des Weiteren eine Schnittstelle für diese Dokumente, wie z.B die Rechnungen oder die Garantieaufträge an, die allein über Scanner die Dokumente übertragen. Dort werden die Dokumente verarbeitet und mit Hilfe eines OCR Scanners digitalisiert.

Derzeit ist allerdings die Digitalisierung nicht in der Lage, einen Kontext beziehungsweise (bzw.) eine Semantik für diese Dokumente zu erstellen. Idealerweise kann dies genutzt werden, um aus den Dokumenten Arbeitsaufträge zu generieren und somit zu digitalisieren. Demzufolge ergibt sich hier an der Stelle die Möglichkeit, einen neuen Ansatz zu testen, infolgedessen die Informationen aus diesen Dokumenten als Daten erhaltbar sind.

2.2 Analyse

Das Extrahieren von Daten ist eine gängige Methode, um Informationen aus verschiedenen Systemen wie Datenbanken oder Software as a Service (SaaS) Plattformen zu erhalten. Durch diese Beschaffung können die Daten weiter verarbeitet werden, wodurch neue Lösungsmöglichkeiten für das eigene System bereitzustellen. Die Arten des Extrahierens unterscheiden sich hierbei im Zeitverlauf und der Herangehensweise. So können sie extrahiert werden, sobald Änderungen der Daten beobachtet oder mitgeteilt worden sind. Durch diese Herangehensweise werden so die Daten schrittweise herangeschafft [28].

Des Weiteren gibt es die Option, eine komplette Extraktion durchzuführen. Dieser Ansatz kann sich in vielen Fällen als nachteilig erweisen, da die Daten jedes mal bei Anpassungen der ursprünglichen Information neu verarbeitet werden müssen. Aufgrund des Bezugs der Problematik im Rahmen der Thesis auf Dokumente, die bereits im ERP-System archiviert wurden, trifft dieser Fall hier nicht zu.

2.2.1 Extraktionsbibliotheken

Für das Data Mining von PDF Dateien existieren bereits Lösungen, die von großen Unternehmen wie Amazon Web Services (AWS) oder Adobe angeboten werden. So nutzen Produkte wie Textractor [11] von AWS oder Adobes Extractor API [15] künstliche Intelligenz, Machine Learning und Optical Character Recognition (OCR), um die Extraktion der Daten zu ermöglichen. Unternehmen wie ABBYY [10] haben sich bereits auf diesem Gebiet spezialisiert und sind erfolgreich im Umgang mit der Datenextraktion. Auch existieren bereits Open-Source Bibliotheken, die sich mit dem Entnehmen der Daten aus PDF Dokumenten auseinandersetzen.

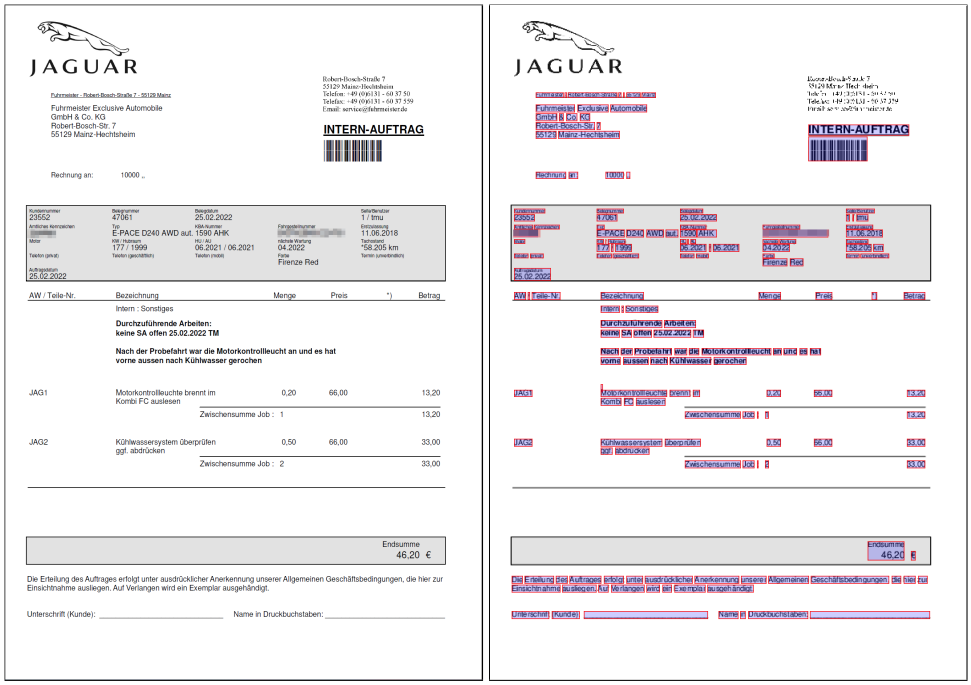
Die meisten der Open-Source Bibliotheken lassen sich hinsichtlich der Aufteilung der Aufgaben gruppieren. So erlauben unter anderem PDFMiner [33] oder Apache Tika [12], welches vor der Extraktion der Texte und Metadaten noch die Klassifizierung und Identifizierung des Medientyps von Dateien anhand einer umfangreichen Hierarchie besitzt, Texte aus PDF Dokumenten zu gewinnen [6]. Funktionalitäten für das Extrahieren von Daten aus Tabellen bieten Projekte wie Tabula [30] oder Camelot [14] an. Diese Projekten decken ferner einen weiteren Aspekt durch die Nutzung der Funktionalitäten ab.

Bei den beiden zuletzt genannten Bibliotheken weisen sich niedrige Fehlerquoten [13] und die Nutzung von einer Fuzzy Logik auf. Camelot ermöglicht überdies ein Web Interface, mit dessen Hilfe eine simple Extraktion mittels einen Browser lokal stattfinden kann. Weitere Bibliotheken, wie Textricator [20] sind umfangreicher gestaltet und ergeben für die Anfragen spezifischer Inhalte geeignete Schnittstellen.

Doch auch wenn dies mit textbasierten Dokumenten funktioniert und der Export flexibel ist, können zunächst weder nicht-textbasierte noch komplexere PDF Dokumente verarbeitet werden. Des Weiteren sind die Verarbeitungen nicht automatisierbar und die Nutzer:innengruppe wird durch eine fehlende grafische Komponente eingeschränkt. Aus dieser Problematik resultierend, zeigt sich, dass community-angetriebene Projekte wie PdfMiner.six [22], das eine Abspaltung des bereits erwähnten PdfMiner ist, und PdfPlumber [27], welches wiederum auf PdfMiner.six basiert, geeignete Alternativen darstellen.

PdfPlumber stellt im Vergleich zu seinem Ausgangsprojekt überdies eine Tabellenextraktion bereit und bietet für beide Aspekte der Extraktion eine Schnittstelle an. Es stellt eine Kommandozeile, die Möglichkeit, visuelles Debugging zu nutzen, und Funktionalitäten wie das Entfernen von Duplikaten in Texten, zur Verfügung.

Das visuelle Debugging, welches in Abbildung 2.1 zu sehen ist, ermöglicht es unter anderem, die Texte des Dokuments hervorzuheben, was wiederum die Extraktion greifbarer darlegt und die Entwicklung mit den Funktionalitäten unterstützt. Auch die Funktionalität des Entfernens von Duplikaten kann dann zum Einsatz kommen, wenn Dokumente bei ihrer Erstellung oder Verarbeitung versehentlich die selben Einträge nochmal im Quellcode des PDFs besitzen. In dieser Abbildung sind die persönlichen Daten des Auftrags anonymisiert und deshalb verpixelt.



(a) ohne visuellem Debugging (b) mit visuellem Debugging

Abbildung 2.1: visuelles Debugging bei einem Dokument

Für die Auswahl einer Extraktionsbibliothek ist auch die dazugehörige Programmiersprache für das System der Datenextraktion, in der die Bibliothek umgesetzt worden ist, relevant. Darunter fallen die bereits genannten Bibliotheken, welche sich in zwei Sprachen gruppieren lassen. Projekte wie Tika, Tabula und Textricator sind mit Hilfe von Java implementiert, während die restlichen, aufgezählten Bibliotheken in Python geschrieben sind. Für Tabula gilt hier hingegen die Ausnahme, dass es für das Projekt unterschiedliche Wrapper für weitere Programmiersprachen wie Python gibt. Hieraus ist zu beobachten, dass die Nutzung von Python viele mögliche Extraktionswerkzeuge erlaubt, welches sich für die spätere Umsetzung als noch bedeutsam erweist.

Die Hürden für eine komplett automatische Extraktion bestehen jedoch nach wie vor. So ist nämlich nicht ersichtlich, wie das zu erwartende Verhalten der Logik der Extraktion für den generellen Fall auszusehen hat. Eine differenzierte Extraktion für weitere Eigenschaften eines Dokuments, wie zum Beispiel Zeilenumbrüche von Absätzen, die Seitennummern, die Kopf- und Fußzeilen, die Formatierung et cetera (etc.) ist daher nötig. Die Frage, die sich dabei stellt, ist, welche Eigenschaften eines Dokuments konkret zusätzlich extrahiert werden sollen. Um eine Auswahl zu treffen, müssen die Arten der Dokumente, die in dem ERP-System auftreten, präziser durchleuchtet werden.

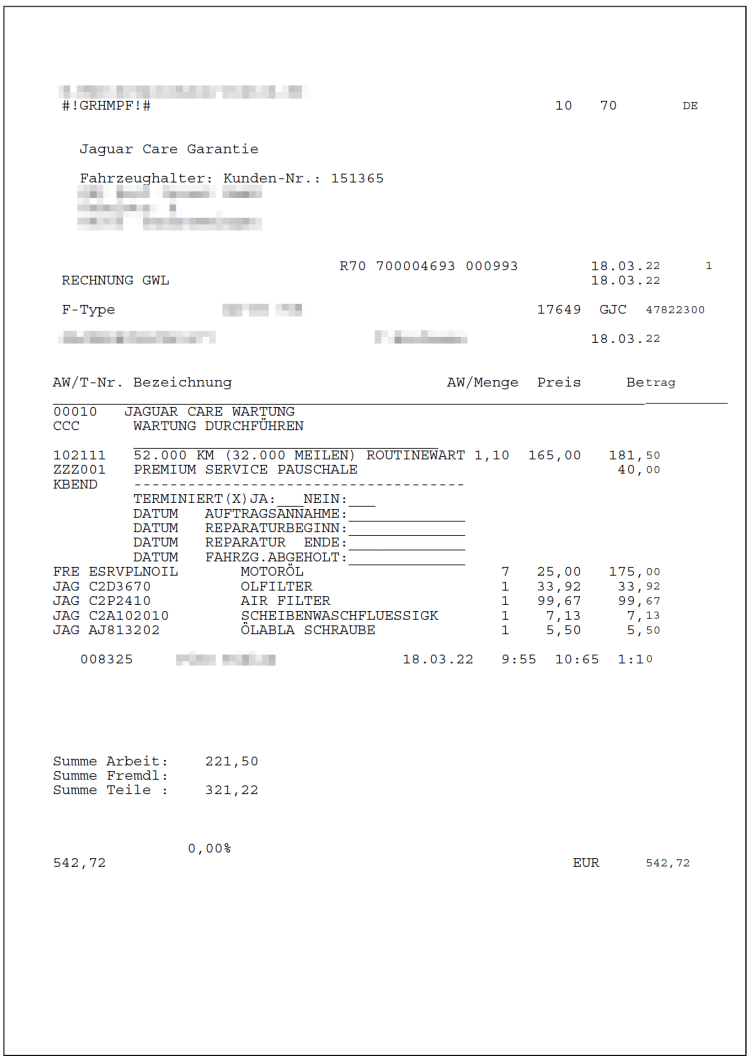
2.2.2 Dokumentarten

Bei den Dokumenten, die in das ERP-System eingescannt werden, handelt es sich in der Mehrheit um Garantie- und Werkstattaufträge und Rechnungen. Jedes Autohaus benutzt hierbei ein individuelles Muster bzw. Format für die jeweilige Art des Dokuments. Dabei ist jedoch der Aufbau dieser Dokumente zu Teilen indirekt durch die von den Autohaus genutzten Dealer-Management-Systeme vorgegeben.

Ebendaher passiert es, dass es Autohäuser gibt, welche das gleiche DMS verwenden und zudem darüber hinaus auch einen detailliert angepassten Dokumentenaufbau für ihre Aufträge benutzen. Dem Umfang dieser Thesis geschuldet werde ich mich auf drei Dokumententypen von verschiedenen Autohäusern mit ihren jeweils genutzten DMS beschränken und anhand jener drei Typen mein Vorgehen exemplarisch demonstrieren. Die Abbildungen der Dokumente in diesem Abschnitt zeigen in einigen Fällen persönliche Daten und wurden deshalb, wie in der Einleitung beschrieben, anonymisiert.

In Abbildung 2.2 wird die erste Art eines Auftrags dargestellt. Bei diesem digitalen Schriftstück handelt es sich zunächst von der Form her um eine Garantierechnung des Dealer-Management-Systems Care. Das Dokument besitzt drei Bereiche, die relevante Informationen für einen Arbeitsauftrag enthalten. In der oberen Hälfte befindet sich zunächst das Adressfeld, welches die Kundeninformationen im genormten Adressformat enthält. Unter dem Adressfeld befindet sich ein weiterer Abschnitt, in dem die Fahrzeugdaten festgehalten werden.

Hierbei ist zu beobachten, dass dieser Abschnitt sich mit einer Tabelle vergleichen lässt. Die einzelnen Einträge werden mit jeweils unterschiedlich großen Abständen verteilt dargestellt, gleichzeitig erfüllen sie eine Gesamthöhe pro Spalte beziehungsweise eine Gesamtbreite pro Höhe einer Tabelle. Gleichwohl fällt auf, dass sich - im Gegensatz zur Anzahl der Zeileneinträge - die Anzahl der Spalteneinträge pro Zeile unterscheidet.



#!GRHMPF!# 10 70 DE

Jaguar Care Garantie

Fahrzeughalter: Kunden-Nr.: 151365

R70 700004693 000993 18.03.22 1
RECHNUNG GWL 18.03.22

F-Type 17649 GJC 47822300

18.03.22

AW/T-Nr.	Bezeichnung	AW/Menge	Preis	Betrag
00010	JAGUAR CARE WARTUNG			
CCC	WARTUNG DURCHFÜHREN			
102111	52.000 KM (32.000 MEILEN) ROUTINEWART	1,10	165,00	181,50
ZZZ001	PREMIUM SERVICE PAUSCHALE			40,00
KBEND				
	TERMINIERT(X)JA:___NEIN:___			
	DATUM AUFTRAGSANNAHME:_____			
	DATUM REPARATURBEGINN:_____			
	DATUM REPARATUR ENDE:_____			
	DATUM FAHRZG.ABGEHOLT:_____			
FRE ESRVPLNOIL	MOTORÖL	7	25,00	175,00
JAG C2D3670	OLFILTER	1	33,92	33,92
JAG C2P2410	AIR FILTER	1	99,67	99,67
JAG C2A102010	SCHEIBENWASCHFLUESSIGK	1	7,13	7,13
JAG AJ813202	ÖLABLA SCHRAUBE	1	5,50	5,50

008325 18.03.22 9:55 10:65 1:10

Summe Arbeit: 221,50
Summe Fremdl:
Summe Teile : 321,22

0,00%
542,72 EUR 542,72

Abbildung 2.2: Exemplar eines Garantieauftrags

Zudem fehlen jegliche Einrahmungen beziehungsweise jegliche Trennlinien für Zeilen und Spalten. Bei den Einträgen in diesem Abschnitt geht es um die Abrechnungsnummer, das Datum der Reparatur, den Fahrzeugmodellnamen, das Kfz-Kennzeichen, den zuletzt abgelesen Kilometerzählerstand, die Arbeitsauftragsnummer, der Fahrzeug-Identifikationsnummer beziehungsweise der 'Vehicle Identification Number' (VIN), die/die zuständige/n Autohausmitarbeiter:in und das Datum der Erstellung des Arbeitsauftrags. Verpixelt und damit anonymisiert wurden die Felder des Kfz-Kennzeichens, der VIN und der/des zuständigen Mitarbeiter:in.

Im unteren Bereich wird eine Tabelle durch die Aufzählung der einzelnen Jobs und Joblines dargestellt. Die Tabelle ist an der Kopfzeile zu erkennen, die die

Jobs und Joblines in die Arbeitsnummer beziehungsweise in die Teilenummer, in die Bezeichnung, in die Menge des Arbeitszeitwertes, in den Preis und in den Betrag übersichtlich aufteilt. Die Teilenummer beziehen sich hierbei auf die Identifikationsnummer der Reparaturteile. Der Arbeitszeitwert ist dabei eine Einheit für die genaue Abrechnung der Arbeitszeiten, um eine Aufschlüsslung der Stunden zu ermöglichen. [9] Die Menge wird hierbei als Reparaturzeit oder als Anzahl der Elemente angegeben. Daneben existiert bei dieser Tabelle nur für die Kopfzeile eine untere Trennlinie. Auch hier sind Probleme bezüglich (bzgl.) der Form der Tabelle zu erkennen.

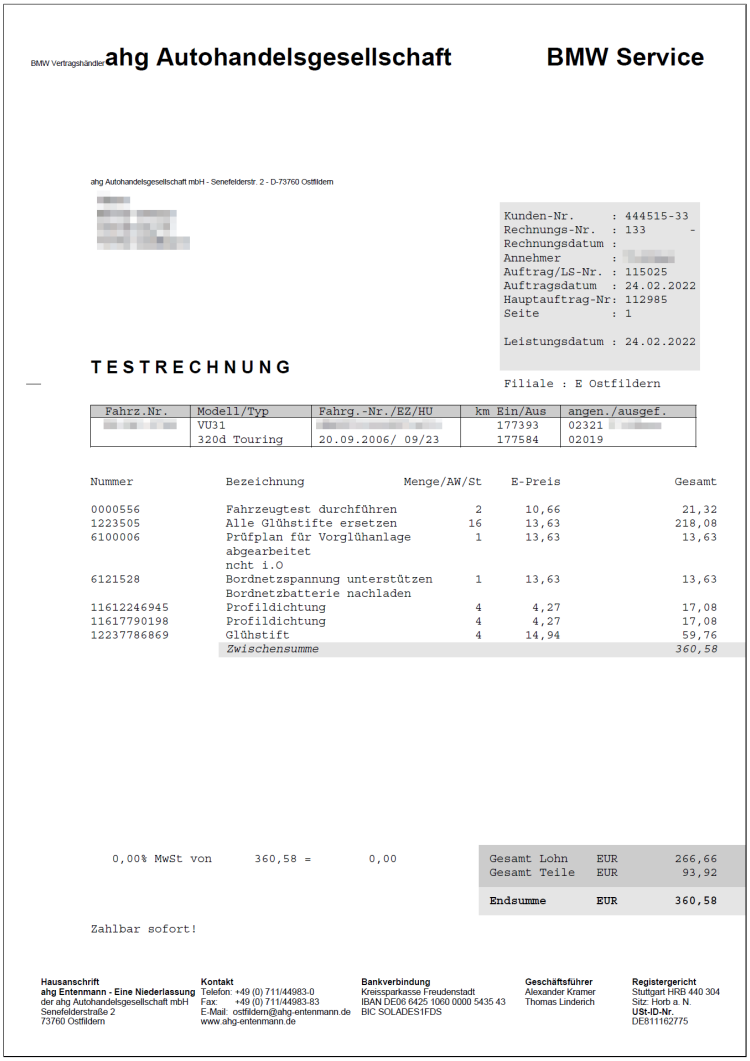
Während die ersten vier Zeilen noch dezent erscheinen, besitzt die fünfte Zeile im Abschnitt mit der Bezeichnung einen längeren Text, sodass der restliche Text in weiteren Zeilen in der gleichen Spalte darunter geschrieben steht. Die Abstände zwischen den Spalten insgesamt scheinen konsistenter zu sein, sind jedoch in der Abbildung 2.2 in mehreren Zeileneinträgen Überläufe bei den Arbeitsnummern bzw. den Teilenummern zu sehen. In der letzte Zeile ist darüber hinaus zu erkennen, dass die Einträge zu Teilen auch verschoben dargestellt werden, was eine eindeutige Erkennung der Positionierung noch weiter erschwert. Inhaltlich werden Jobs mit den dazugehörigen Joblines darunter angereiht dargestellt.

Dies ist durch die Codierung der Arbeitsnummern bzw. der Teilenummer zu erkennen, die Hierarchie ist durch das jeweilige Dealer-Management-System vorgegeben. In der ersten Spalte können daher Codes von Jobs, Reparaturzeiten, Ersatzteilen, Fremdleistungen, Stempelzeiten und Kundenbeschwerden auftreten.

Für die Codierung ist keine standardisierte Spezifikation vorhanden, weshalb das für jedes Autohaus und seine Nutzung der DMS individuell definiert ist. So sind die Codes der Jobs als eine fünfstellige Zeichenkette aus Ziffern bestimmt, die mit den Ziffern ‘00’ angefangen. Die Reparaturzeiten dagegen besitzen eine sechsstellige Zeichenkette aus Ziffern. Unterhalb der Tabelle werden am Ende des Dokuments schließlich die gesamten Kosten aus den Betragseinträgen summiert dargestellt. Diese Summe lässt sich wiederum in Kosten der Arbeit und der Teile aufteilen.

In Abbildung 2.3 wird eine Rechnung der Formel 1 DMS dargestellt. Auch hier sind sämtliche personenbezogenen Daten verpixelt und anonymisiert worden. Im Vergleich zur vorherigen Abbildung 2.2 ist ein ähnlicher Aufbau zu sehen. Während das Adressfeld geläufig aussieht, sind jedoch die Fahrzeug- und Kundendaten diesmal in zwei unterschiedlichen Tabellen aufgeteilt.

Die erste Tabelle befindet sich im oberen Teil der rechten Seite des Dokuments, wo die einzelnen Kundeninformationen aufgelistet sind. Die Tabelle für die



BMW Vertragshändler **ahg Autohandelsgesellschaft** **BMW Service**

ahg Autohandelsgesellschaft mbH - Senefelderstr. 2 - D-73760 Ostfildern

[redacted]

Kunden-Nr. : 444515-33
Rechnungs-Nr. : 133 -
Rechnungsdatum :
Annehmer : [redacted]
Auftrag/LS-Nr. : 115025
Auftragsdatum : 24.02.2022
Hauptauftrag-Nr: 112985
Seite : 1

Leistungsdatum : 24.02.2022

TESTRECHNUNG

Filiale : E Ostfildern

Fahrz.Nr.	Modell/Typ	Fahrg.-Nr./EZ/HU	km Ein/Aus	angen./ausgef.
[redacted]	VU31	[redacted]	177393	02321 [redacted]
	320d Touring	20.09.2006/ 09/23	177584	02019

Nummer	Bezeichnung	Menge/AW/St	E-Preis	Gesamt
0000556	Fahrzeugtest durchführen	2	10,66	21,32
1223505	Alle Glühstifte ersetzen	16	13,63	218,08
6100006	Prüfplan für Vorglühanlage abgearbeitet ncht i.O	1	13,63	13,63
6121528	Bordnetzspannung unterstützen Bordnetzbatterie nachladen	1	13,63	13,63
11612246945	Profildichtung	4	4,27	17,08
11617790198	Profildichtung	4	4,27	17,08
12237786869	Glühstift	4	14,94	59,76
	Zwischensumme			*360,58*

0,00% MwSt von 360,58 = 0,00

Gesamt Lohn	EUR	266,66
Gesamt Teile	EUR	93,92
Endsumme	**EUR**	**360,58**

Zahlbar sofort!

Hausanschrift
ahg Entenmann - Eine Niederlassung
der ahg Autohandelsgesellschaft mbH
Senefelderstraße 2
73760 Ostfildern

Kontakt
Telefon: +49 (0) 711/44983-0
Fax: +49 (0) 711/44983-83
E-Mail: ostfildern@ahg-entenmann.de
www.ahg-entenmann.de

Bankverbindung
Kreissparkasse Freudenstadt
IBAN DE06 6425 1060 0000 5435 43
BIC SOLADES1FDS

Geschäftsführer
Alexander Kramer
Thomas Linderich

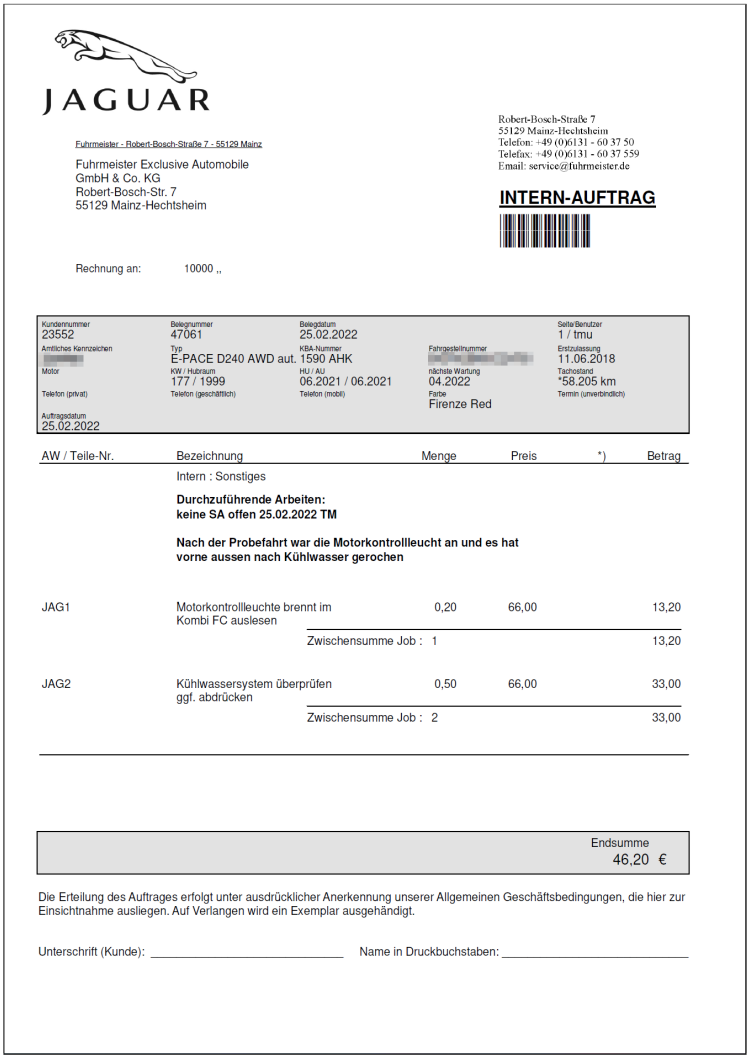
Registergericht
Stuttgart HRB 440 304
Sitz: Horb a. N.
USt-ID-Nr.
DE811162775

Abbildung 2.3: Exemplar einer Rechnung

Fahrzeuginformationen befindet sich im unteren Bereich der oberen Hälfte des Dokuments und ist aufgrund der vorhandenen Trennlinien eine klassische Tabelle. Im letzten Teil der Formel 1 Rechnung befindet sich die Liste der Job und Joblines.

Hier fallen ähnliche Problematiken, wie die unterschiedlichen Abstände zwischen den Spalten der Einträge im Vergleich zu dem Care Garantieauftrag auf. Auch existieren Überläufe der Einträge, allerdings treten diese nur in den Feldern der Bezeichnung auf. Die Gesamtkosten werden dann am Ende des Dokuments in den Teilsummen der Beträge bekannt gegeben.

Bei der dritten und damit letzten Dokumentart (Abbildung 2.4) handelt es sich



JAGUAR

Fuhrmeister - Robert-Bosch-Straße 7 - 55129 Mainz

Fuhrmeister Exclusive Automobile
GmbH & Co. KG
Robert-Bosch-Str. 7
55129 Mainz-Hechtsheim

Robert-Bosch-Straße 7
55129 Mainz-Hechtsheim
Telefon: +49 (0)6131 - 60 37 50
Telefax: +49 (0)6131 - 60 37 559
Email: service@fuhrmeister.de

INTERN-AUFTRAG

Rechnung an: 10000 ,,

Kundennummer 23552	Belegnummer 47061	Belegdatum 25.02.2022		Seite/Benutzer 1 / tmu
Amtliches Kennzeichen	Typ E-PACE D240 AWD aut. 1590 AHK	KBA-Nummer	Fahrgestellnummer	Erstzulassung 11.06.2018
Motor	KW / Hubraum 177 / 1999	HU / AU 06.2021 / 06.2021	nächste Wartung 04.2022	Tachostand *58.205 km
Telefon (privat)	Telefon (geschäftlich)	Telefon (mobil)	Farbe Firenze Red	Termin (unverbindlich)
Auftragsdatum 25.02.2022				

AW / Teile-Nr.	Bezeichnung	Menge	Preis	*)	Betrag
	Intern : Sonstiges				
	Durchzuführende Arbeiten: **keine SA offen 25.02.2022 TM**				
	Nach der Probefahrt war die Motorkontrollleucht an und es hat vorne aussen nach Kühlwasser gerochen				
JAG1	Motorkontrollleuchte brennt im Kombi FC auslesen	0,20	66,00		13,20
	Zwischensumme Job : 1				13,20
JAG2	Kühlwassersystem überprüfen ggf. abdrücken	0,50	66,00		33,00
	Zwischensumme Job : 2				33,00

Endsumme 46,20 €

Die Erteilung des Auftrages erfolgt unter ausdrücklicher Anerkennung unserer Allgemeinen Geschäftsbedingungen, die hier zur Einsichtnahme ausliegen. Auf Verlangen wird ein Exemplar ausgehändigt.

Unterschrift (Kunde): _____ Name in Druckbuchstaben: _____

Abbildung 2.4: Exemplar eines Werkstattauftrags

um einen Werkstattauftrag von dem Dealer-Management-System KFZ3000. Auch dieses hier abgebildete Dokument wurde aus Datenschutzgründen anonymisiert. Dieser hier gezeigte Auftrag lässt sich in Bezug auf die Aufteilung der einzelnen Segmente mit dem Care-Warranty-Dokument vergleichen. Während auch das Adressfeld erneut normiert abgebildet ist, werden Kunden- und Fahrzeugdaten über Schlüssel-Wert Anordnungen in einem größeren Feld dargestellt.

Die untere Hälfte des Werkstattauftrags besitzt gleicherweise wie die bereits anderen genannten Dokumente eine tabellenähnliche Auflistung der Job und Joblines. Ebenfalls sind hier Überläufe und unterschiedliche Abstände in dem Bereich zu erkennen. Hinzu kommt, dass, im Gegensatz zu den anderen Dokumentarten, zwischen den Zeilen ein Art Summenstrich hinzugefügt wird,

der die Anzahl der Jobs anzeigen soll. Außerdem wird im unteren Bereich des Dokuments lediglich die gesamte Betragssumme gezeigt.

Insgesamt lässt sich feststellen, dass alle hier vorgestellten Dokumentarten durch eine gemeinsame Problematik mehrerer Hürden miteinander verbunden sind. Die Darstellung der Kunden- und Fahrzeuginformationen wird entweder in einer Tabellenform gemeinsam (siehe Abbildung 2.2) oder in zwei verschiedenen Tabellen (siehe Abbildung 2.3) abgebildet. Diese Tabellenform besitzt außerdem selten Trennlinien, welche geeignet sind, um eine derartige Tabelle als solche definieren zu können. Für die Liste der Job und Joblines tritt das gleiche Problem auf.

Diese Auflistung wird allein durch ihre Anordnung der Einträge und deren Abstände dazwischen als solche erkenntlich, jedoch bricht sie in den prototypischen Dokumenten aber mindestens (mind.) einmal die Form durch Überläufe oder Verschiebungen der Texte. Es ist festzustellen, dass verschiedene Tabellenstrukturen für unterschiedlichen Dokumentarten existieren. Dieses Angelegenheit besteht, da bisher für Tabellen keine universelle Standardisierung definiert ist.

Damit diese Formen durch eine Logik des Extrahierens abgedeckt werden, besteht die Möglichkeit der Fallunterscheidung. Bei dieser Unterscheidung sollen die Dokumente anhand ihrer Arten konzipiert und definiert werden, sodass die Entnahme detaillierter stattfinden kann. Für andere Bereiche eines Dokuments wie dem Adressfeld werden gewöhnliche Funktionalitäten der Extraktion genutzt und zur Verfügung gestellt. Diese können dann von allen weiteren Dokumentarten geteilt und genutzt werden, sodass eine Verallgemeinerung verschiedener Dokumententypen ermöglicht wird.

2.2.3 Frameworks

Damit das System für den Prozess der Extraktion als solches sich zeitnah im Rahmen einer Thesis umsetzen lässt, können Frameworks dem ganzen Zeitfenster entgegenkommen. Im Zuge dessen werden für die Teilsysteme verschiedene Technologien betrachtet und in Erwägung gezogen. So wird zunächst die symbolische künstliche Intelligenz dem ersten Teilsystem zugeordnet und in seiner Art unterschieden. Zudem müssen die Realisierungsmöglichkeiten der symbolischen KI betrachtet und aus den verschiedenen Arten eines wissensbasiertes Systems muss ein Modell ausgewählt werden.

Ein solches wissensbasiertes System ist hierbei der Oberbegriff für Programme, die mit Hilfe einer Wissensbasis Mechanismen der Schlussfolgerung nutzen, um Lösungen für Probleme zu finden [2]. Es existieren verschiedene Modelle eines

wissensbasiertes Systems, die sich, je nach Anwendungsfall, als passend erweisen. So gibt es den Ansatz des regelbasierten Systems bzw. Produktionssystems, bei dem die Schlussfolgerung über Regeln und Fakten, die von einem Benutzer definiert werden, gestaltet wird.

Bekannt für die Erstellung eines solchen Systems, ist das in der Programmiersprache C geschriebene Werkzeug CLIPS[1], welches über eine eigene Syntax für die Regeln verfügt. Diese Umsetzung wird auch mit dem Wrapper 'clipspy' für die Programmiersprache Python zur Nutzung bereitgestellt. Andere Umsetzungen konzentrieren sich im Vergleich eher auf die Inferenzmaschine, die auch als Regelinterpreter gilt. Projekte wie 'Pyke' [19], 'Durable Rules' [26] oder 'Experta' [23], dem Nachfolger von einem Expertensystem Framework namens 'pyKnow', erlauben es eine Regelmenge zu definieren und die einzelnen Regeln mit selbstdefinierten Funktionen zu koppeln.

Für die Entwicklungsumgebung, die den Zugang zum System darstellen soll, eignen sich sowohl Benutzerschnittstellen (UI) Bibliotheken für den Desktop als auch für den Web. Die Priorität liegt hierbei bei der Erstellung der einzelnen Komponenten der graphische Benutzeroberflächen (GUI). So können Projekte wie Qt [25] und ImGui [21], welches in C++ geschrieben ist, eine Bibliothek bereitstellen, die auf einer nativen Ebene funktionieren und damit in verschiedenen Anwendungsfällen genutzt werden. Letzter genanntes benutzt intern einen Zustandsautomat um die Render-Pipeline darstellen, was etwas mehr Erfahrung mit der Grafikprogrammierung erfordert, um dies verwenden zu können [4].

Dem gegenüber stehen abstraktere UI-Lösungen wie das Benutzen eines Web Frameworks, welche im Front-End Bereich verwendet werden. Bekannte Bibliotheken wie Vue und React ermöglichen eine schnelle und gängige Umsetzung einer Oberfläche, die auch die Zwecke einer Entwicklungsumgebung erfüllen können. Weitere Kandidaten wie Svelte [29] versuchen durch das Nutzen eines Complier, die Arbeit von dem Browser zu der Build Zeit umzulagern.

Für das Teilsystems, welches den Ansatz des maschinellen Lernens nutzen soll, scheinen auch hier ML-Frameworks eine geeignete Lösung zu sein. Kandidaten sind hierbei die zurzeit konventionellen Frameworks wie TensorFlow [31], pyTorch [24] und spaCy [18], welche für die Anwendungsgebiete unter anderem in der Bilderkennung, der Computer Vision und des Neuro-Linguistische Programmieren (NLP) eine automatisierte Lösung anbieten. Weitere neue Projekte wie Fonduer [7] fokussieren sich mit Hilfe von schwachem überwachten Lernen (supervised learning) auf die Extraktion von umfangreichen formatierten Daten, worunter auch PDF Dokumente fallen. Dieser Lernansatz findet auch bei nicht vollständig beschrifteten Trainingsdaten einen Anwendungszweck [8],

weswegen das für einfache Dokumente sich als praktisch erweisen kann.

Kapitel 3

Konzeption und Umsetzung

Für den Sprung zur Umsetzung bedarf es als Nächstes die Konzeption des technischen Rahmens. Zunächst wird der Aufbau des Projekts vorgestellt, durch welches die beschriebene Problematik gelöst werden soll. Danach wird der Verlauf der Entwicklung über die Aufteilung des Systems in seine Komponente dargestellt, welche im Detail ihre Eigenschaften und Zusammenhänge beschreiben.

3.1 Aufbau

Für die Struktur des Systems ist vorerst die Aufteilung des Projekts relevant. Dies kann je nach einem ausgewählten Architekturmuster unterschiedlich gestaltet werden. Da die Teilsysteme wiederum als eigene Systeme dargestellt werden können und für die Anwendungsfälle kommunizieren beziehungsweise interagieren müssen, scheint hier der Ansatz der verteilten Systeme geeignet zu sein. Die Kategorie der verteilte Systeme ist in weitere unterschiedliche Architekturmustern gegliedert. Eines dieser Mustern ist die Client-Server Architektur, die es erlaubt über ein Netzwerk die Kommunikation bereitzustellen und die Teilsysteme und ihre Aufgaben aufzuteilen.

So gibt es drei Komponenten mit unterschiedlichen Aufgaben, die die in vorherigen beschriebenen Teilsysteme abgebildet werden: Die erste Komponente kümmert sich um die Extraktion über ein regelbasierte Logik, womit die Daten von mehreren Dokumenten eines Dokumententyps automatisiert erhalten werden. Die zweite soll die Entwicklungsumgebung umsetzen und damit auch jegliche Erstellung und Bearbeitung von Regeln über Textdateien ermöglichen. Die dritte und letzte Komponente soll für die Extraktion der Dokumente den Lösungsansatz des Machine Learnings umsetzen.

Mit der beschriebenen Architekturmuster fungieren die erste und dritte Komponente als Server, die die Logik der Extraktion ausführen sollen. Die Entwicklungsumgebung als zweite Komponente ist im Gegensatz dazu als Client zu betrachten, der den Benutzer:innen Änderungen der Regeldefinitionen zulässt. Damit dieser Netzwerkansatz für die Architektur umgesetzt werden kann, müssen die Komponenten Schnittstellen auf der Netzwerkebene bereitstellen.

Für die jeweiligen Komponenten, die sich mit der Extraktionslogik beschäftigen, passt hier der Lösungsansatz der Benutzung eines Webservers, welcher die Anfragen verarbeiten und die verschiedenen Zustände zurückgibt. Bei der Komponente der Entwicklungsumgebung eignet sich der Ansatz einer Webanwendung, womit weitere Technologien aus dem Webbereich genutzt werden können. Somit wird jegliche Bearbeitung der Definitionen durch die IDE als Anfrage verschickt und die daraus folgenden Antworten als Ergebnisse präsentiert.

Für die praktische Umsetzung beziehungsweise für die Entwicklung wird der Lösungsentwurf als ein Softwareprojekt in Form eines Monoliths konstruiert, welches das ganze System darstellt. Das Projekt trägt den Namen ‘Smart Document Extractor’ und wird mit dem gängigen Versionsverwaltungssystem Git aufgesetzt. Die darin gegliederten Komponenten werden als ‘Rule System’, ‘IDE’ und ‘ML’ genannt.

Somit wird das Projekt als lokales Repository und als Remote Repository auf dem GitLab Server des Unternehmens festgehalten. Im Wurzelverzeichnis wird zudem mit der Container Virtualisierung von Docker eine Betriebssystemagnostische Umgebung für die Teilsysteme definiert. Dies ist nicht nur der Softwareentwicklung, sondern auch der Continuous Integration und deren Testschritte dienlich und ermöglicht einen Ausbau der Skalierung der Teilsysteme.

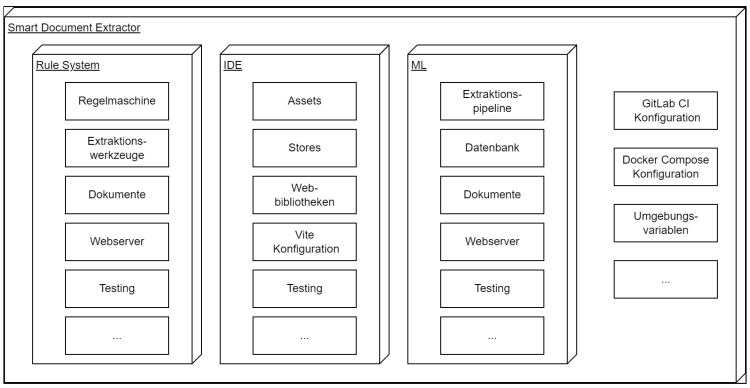


Abbildung 3.1: Struktur des Softwareprojekts

Die einzelnen Teilsysteme werden mit Hilfe einer Docker Compose Datei dann

orchestriert, sodass die Container gemeinsam gestartet werden und ihre Kommunikation über ein gemeinsames Netzwerk führen. Die zusätzliche Konfiguration von Umgebungsvariablen in dem Projekt erlaubt es außerdem, sensible Daten wie die Anmeldeinformationen der Zugänge der Teilsysteme oder weiteren Dienste für den Arbeitsbereich dynamisch anzupassen. Die Konfiguration wird hierbei von dem Versionsverwaltungssystem ignoriert.

Abbildung 3.1 zeigt eine Übersicht des Projekts und seine Eigenschaften, auf die im weiteren Verlauf detaillierter eingegangen wird. Dort sind neben den beschriebenen Konfigurationsdateien die Teilsysteme und deren Module als Komponenten dargestellt.

3.2 Symbolische KI

Für das Teilsystem der symbolischen künstlichen Intelligenz wurde der Ansatz beziehungsweise die Nutzung eines wissensbasiertes Systems ausgewählt. Dies ermöglicht die Steuerung der Extraktion über definierte Fakten und Regeln. In Kapitel 2.2.1 wurde ersichtlich, dass es viele Frameworks gibt, die ein solches System bereitstellen können. Doch zeigt sich auch bei jenen, dass die Nutzung eines dieser Frameworks nicht für den Fall passend ist, da mehr Funktionalitäten als benötigt abgedeckt werden.

Ein regelbasiertes System zeichnet sich durch eine Menge von Regeln, den Produktionsregeln, eine Menge von Fakten, welche sich während des Inferenzsvorgangs verändert, und durch den Regelinterpreter aus. Der Interpreter kann mit Hilfe dieser Menge die Schlussfolgerungen in Form einer Vorwärtsverkettung oder Rückwärtsverkettung durchführen [5].

Für den Kontext der Dokumentenextraktion bedarf es nur die Ausführung von Extraktionen und die Generierung neuer Fakten, sobald die benutzerdefinierten Regeln mit den bereits vorhandenen Fakten erfüllt wurden, als Funktionalitäten. Die vorgestellten Bibliotheken bringen einen größeren Aufwand mit sich, welche weitere Abhängigkeiten mit sich bringen und den Rahmen des Projekts übersteigen. Daher wird hier der Lösungsweg über ein eigen implementiertes Regelsystem verfolgt. Der Regelinterpreter wird speziell für den Zweck der Datenextraktion entwickelt und passt sich damit an Nutzung des gesamten Projekts an.

3.2.1 Extraktionsvorgang

Bevor sich der Regelmaschine gewidmet werden kann, muss zunächst die konkrete Extraktion betrachtet werden. Im Abschnitt 2.2.2 wurde gezeigt, dass die Hauptproblematik der Dokumente in deren Extraktion von Tabellen liegt. Hierfür definieren wir Tabellen, die für jede Zeile und Spalte Trennlinien besitzen, als konventionelle Tabellen. Ein Beispiel hierfür ist die Tabelle, die in Abbildung 2.3 die Fahrzeugdaten bereit hält. Tabellen, die keine oder nur zu Teilen Trennlinien für die Zeilen und Spalten nutzen, werden als unkonventionelle Tabellen definiert. Diese Art der Tabelle tritt in den drei vorgestellten Dokumentarten in Form der Auflistung von den Jobs und Joblines auf. Andere Tabellen, die eine beliebige Form und somit die restlichen Tabellenarten darstellen, werden als benutzerdefinierte Tabelle definiert.

Textfelder können im Vergleich zu den definierten Tabellen leichter extrahiert werden, weswegen die Funktionalität für alle Dokumentarten beziehungsweise Dokumententypen benutzt werden kann. Dafür wird eine gemeinsame Klasse namens 'DocumentExtractionType' definiert, von der die weiteren Klassen erben, die die jeweilige Dokumentart abbilden. So werden des Weiteren die Dokumentarten durch die Klassen 'CareWarranty', 'Formel1Invoice' und 'KFZ3000Invoice' dargestellt. Die gemeinsamen Funktionalitäten, die in der Elternklasse definiert sind, bestehen aus der Textextraktion von Textfeldern, der Textextraktion von konventionellen Tabellen, der Textextraktion von unkonventionellen Tabellen und der Textextraktion von benutzerdefinierten Tabellen.

3.2.2 Regelsyntax

Für die Speicherung der Regeln und Fakten kann dies über den persistenten Ansatz der Datenbanken ermöglicht werden. Jedoch erlaubt der Ansatz der Haltung der Regeln und Fakten über eine Datei im Dateisystem den schnellen Zugriff und die schnelle Bearbeitung jener. Des Weiteren tritt hier die Frage auf in welchem Dateiformat, die Regeln und Fakten festgehalten werden. Ein gängiges Serialisierungsformat wie JSON eignet sich durch ein festgelegtes Schema für die Speicherung, da die Daten strukturiert aufbewahrt werden. Jedoch existieren weitere Dateiformate, die eine geeignete Alternative darstellen.

Die Auszeichnungssprache YAML bildet die Obermenge zu JSON ab und ermöglicht neben den generischen Datentypen, die auch in JSON definiert sind, einen übersichtlicheren aber auch komplexeren Ansatz für ein Serialisierungsformat. Diese Sprache besitzt im Vergleich zu JSON eine bessere Lesbarkeit der Syntax, ist kompakter gestaltet und bietet eine hohe Kompatibilität zu

dem JSON Format. [3]. Die Syntax für Kommentare ist zudem auch verfügbar, wodurch sich diese Auszeichnungssprache für Konfigurationsdateien eignet, was wiederum für die Definitionen der Regeln und Fakten abgebildet werden kann. Die Serialisierungsformate unterscheidet sich letztendlich in der Zielsetzung: JSON wird für den leichten Datenaustausch für Webanwendungen aufgrund der leichten Serialisierung und Übersetzung mit dem Parser benutzt. YAML dagegen hat eine erhöhte Komplexität für die Serialisierung und Übersetzung, aber ist für Dateien gedacht, die von Menschen bearbeitet und damit genutzt werden sollen.

Mit Hilfe einer Bibliotheken für die Serialisierung und Übersetzung von YAML-Dateien werden somit die Regeln und Fakten während der Laufzeit geladen, bearbeitet und gespeichert. Hierfür pathenize

In einem eigenen Modul

Damit das System für die Datenextraktion von großen Dokumentenmengen ermöglicht wird, bedarf es hier zunächst die konkrete Benutzung der Funktionalitäten der Extraktion.

Wie im Kapitel 2.2.2 gezeigt wurde, existieren für die verschiedenen Dokumentarten sowohl Gemeinsamkeiten als auch Unterschiede in den Eigenschaften. Um dies abzubilden, wird für jede Dokumentart ein Modul definiert, welches die weiteren Bestandteile für den Extraktionsprozess besitzt.

3.3 Entwicklungsumgebung

Für die graphische Schnittstelle der Benutzer des Systems

3.4 Machine Learning

Während

Kapitel 4

Ergebnisse

Kapitel 5

Konklusion und Ausblick

Im Rahmen der Abschlussarbeit wurde ein System konzipiert und umgesetzt, welches eine Extraktion von Dokumenten über zwei Ansätze ermöglicht. Über die Teilsysteme wurden technische Funktionalitäten eingeführt werden um dem Kapitän die Webanwendung als PWA anzubieten.

Insgesamt eignen sich Webanwendungen um Ressourcen übersichtlicher darzustellen und mobil zu beobachten. Auch können Offline-Zustände dem Benutzer trotzdem erlauben Aspekte der Anwendung weiter zu benutzen. Aus den Konzepten und der Umsetzung ergeben sich viele weitere Ideen und Ansätze, die realisierbar sind und implementiert werden können.

Literaturverzeichnis

[1] clips. CLIPS: A Tool for Building Expert Systems. https://www.clipsrules.net/.

[2] David Embrey. The Skill, Rule and Knowledge Based Classification. *Human Error*, page 10.

[3] Malin Eriksson and Victor Hallberg. Comparison between json and yaml for data serialization. *The School of Computer Science and Engineering Royal Institute of Technology*, pages 1–25, 2011.

[4] Borko Furht, Esad Akar, and Whitney Angelica Andrews. *Creating User Interface*, pages 7–11. Springer International Publishing, Cham, 2018.

[5] Di Liu, Tao Gu, and Jiang-Ping Xue. Rule Engine based on improvement Rete algorithm. In *The 2010 International Conference on Apperceiving Computing and Intelligence Analysis Proceeding*, pages 346–349, 2010.

[6] Chris A. Mattmann and Jukka L. Zitting. *Tika in Action*. Manning Publications, Shelter Island, NY, 2012. Includes index.

[7] Sen Wu, Luke Hsiao, Xiao Cheng, Braden Hancock, Theodoros Rekatsinas, Philip Levis, and Christopher Ré. Fonduer: Knowledge base construction from richly formatted data. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1301–1316. ACM, 2018.

[8] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, August 2017.

Online-Quellen

[9] Bedeutung AW (Arbeitswert). https://wissen.autoixpert.de/hc/de/articles/360008973819-Bedeutung-AW-Arbeitswert-.

[10] abbyy. PDF Software: Open, Read & Edit PDFs. https://pdf.abbyy.com/. [letzter Zugriff: 4. Juni 2022].

[11] amazon. Intelligente Extraktion von Text und Daten mit OCR – Amazon Textract – Amazon Web Services. https://aws.amazon.com/de/textract/. [letzter Zugriff: 4. Juni 2022].

[12] apache. Apache Tika – Apache Tika. https://tika.apache.org/. [letzter Zugriff: 4. Juni 2022].

[13] atlanhq. Comparison with other PDF Table Extraction libraries and tools · atlanhq/camelot Wiki. https://github.com/atlanhq/camelot. [letzter Zugriff: 4. Juni 2022].

[14] camelot. Camelot: PDF Table Extraction for Humans. Atlan Technologies Pvt Ltd, July 2022. [letzter Zugriff: 4. Juni 2022].

[15] Adobe I/O-Adobe Developers. Extract Text from PDF | Extract Data from PDF | Visualizer - Adobe Developers. https://developer.adobe.com/document-services/apis/pdf-extract/. [letzter Zugriff: 4. Juni 2022].

[16] Ben Dickson. What is symbolic artificial intelligence? https://bdtechtalks.com/2019/11/18/what-is-symbolic-artificial-intelligence/, November 2019. [letzter Zugriff: 4. Juni 2022].

[17] Docsumo. PDF Scraper - Scrape data from pdf | PDF data extraction. https://docsumo.com/blog/extract-data-from-pdf, June 2022. [letzter Zugriff: 4. Juni 2022].

[18] explosion. spaCy · Industrial-strength Natural Language Processing in Python. https://spacy.io/.

[19] Bruce Frederiksen. Welcome to Pyke. http://pyke.sourceforge.net/index.html.

[20] measures. Measuresforjustice/textricator: Textricator is a tool to extract text from documents and generate structured data. https://github.com/measuresforjustice/textricator. [letzter Zugriff: 4. Juni 2022].

[21] omar. Dear ImGui, July 2022.

[22] pdfminer. Welcome to pdfminer.six's documentation! — pdfminer.six __VERSION__ documentation. https://pdfminersix.readthedocs.io/en/latest/.

[23] Roberto Abdelkader Martínez Pérez. Nilp0inter/experta, July 2022.

[24] pyTorch. PyTorch. https://www.pytorch.org.

[25] qt. Qt | Cross-platform software development for embedded & desktop. https://www.qt.io.

[26] Jesus Ruiz. Durable_rules, March 2022. [letzter Zugriff: 4. Juni 2022].

[27] Jeremy Singer-Vine. Pdfplumber, July 2022.

[28] stichdata. What is Data Extraction? [Tools & Techniques]. https://www.stitchdata.com/what-is-data-extraction/. [letzter Zugriff: 4. Juni 2022].

[29] svelte. Svelte • Cybernetically enhanced web apps. https://svelte.dev/.

[30] tabulapdf. Tabulapdf/tabula: Tabula is a tool for liberating data tables trapped inside PDF files. https://github.com/tabulapdf/tabula. [letzter Zugriff: 4. Juni 2022].

[31] tensorflow. TensorFlow. https://www.tensorflow.org/?hl=de.

[32] Laura Theuerer. Was macht das After Sales Management im Autohaus? | Karriere-Ratgeber - kfz-betrieb Jobs. https://jobs.kfz-betrieb.de/karriere-ratgeber/was-macht-das-after-sales-management-im-autohaus-51.html. [letzter Zugriff: 4. Juni 2022].

[33] unixuser. PDFMiner. https://www.unixuser.org/~euske/python/pdfminer/. [letzter Zugriff: 4. Juni 2022].