Zusammenfassung

Das Ziel der vorliegenden Arbeit war die Implementierung und Auswertung einer bildbasierten Produktsuche im Shopsystem SAP Hybris Commerce. Die Verwendung der bildbasierten Suche im Bereich des E-Commerce ist einer der am häufigsten erwähnten Anwendungsfälle für diese Suchtechnologie. Dabei existieren bereits verschiedene Einbindungen dieser Art der Suche in Onlineshops, deren Einsatz allerdings noch sehr schwach verbreitet ist. Auf welchen Grundlagen Systeme dieser Art aufbauen, was für Probleme sie in sich bergen, die einer verbreiteten Verwendung eventuell im Weg stehen, und wie diese gelöst werden können, sollte in dieser Arbeit sowohl praktisch als auch theoretisch erläutert werden. Die Umsetzung dieses prototypischen Content-based Visual Information Retrieval Systems (CBVIR) wurde in Zusammenarbeit mit der Jenaer Digitalagentur dotSource GmbH durchgeführt. Dabei sollten sowohl die Chancen, Möglichkeiten und Probleme dieser Art der Suche, als auch die Grundlagen eines solchen Systems und dessen Erstellung erforscht und ausgewertet werden. Mit Hilfe des Shopsystems SAP Hybris Commerce, der integrierten Suchmaschine Apache Solr und dem zugehörigen Plugin lire-solr wurde eine funktionale Implementierung verwirklicht. Die Auswertung erfolgte in einer quantitativen Art mit der Hilfe etablierter Auswertungsmetriken und dem DeepFashion Bilddatensatz für Kleidungsstücke. Als Resultat steht eine funktionstüchtige Umsetzung einer bildbasierten Produktsuche zur Verfügung, welche auf Grundlage der Auswertung und der flexiblen Struktur weiter verbessert werden kann. Der schriftliche Teil dieser Ausarbeitung bietet einen Überblick über die wichtigsten Grundlagen von Systemen dieser Art, dokumentiert und erklärt alle Entscheidungen und Vorgänge zu deren Erstellung und Auswertung und führt in verschiedene Problemstellungen, Lösungen und zukünftige Aspekte ein. Das Ergebnis dieser Arbeit ist in diesem Sinne die sowohl praktische als auch theoretische Beantwortung der Frage, wie ein System für eine bildbasierte Suche in einem Onlineshop implementiert, getestet und verbessert werden kann, wodurch sowohl eine Wissensgrundlage für künftige Projekte dieser Art als auch eine technologische Grundlage für eine einfache Einbindung dieser Suche in SAP Hybris Commerce geschaffen wurde.

Abstract

The goal of this bachelor thesis was the implementation and evaluation of an image-based search for products in the shop system SAP Hybris Commerce. The usage of such an image-based search is one of the most discussed use cases for this method of searching. Although multiple integrations already exist within online shops, the usage of this search method is still poorly spread. In this written work the foundations of these systems, the problems they inherit, which may be responsible for the weak spread, and approaches for their solution should be practically and theoretically explained. The realization of this prototypical content-based visual information retrieval system (CBVIR) has been performed in collaboration with the digital agency dotSource GmbH located in Jena. In the process the chances, possibilities and problems as well as the foundations of such systems and their creation have been researched and evaluated. With the help of the shop system SAP Hybris Commerce, the integrated search engine Apache Solr and the corresponding plugin Lire-solr a functional implementation was accomplished. The evaluation has been performed with the help of quantitative methods involving established evaluation measures and the DeepFashion image library for fashion items. The result is a functional implementation of an image-based product search, which can be easily reused or improved on, because of its flexible structure and the executed evaluation. The theoretical and written part of this thesis allows an overview of the most important foundations of such systems, documents and explains the decisions and processes for their creation and also introduces important problems, solutions and future aspects of this field of research. In conclusion this thesis is the practical and theoretical answer to the question, how an image-based product search can be implemented into an online shop system as well as what the evaluation and possible improvements can look like. This results in a foundation of knowledge for future projects of this kind and also in a technological foundation for the simple implementation of this solution with SAP Hybris Commerce and its ecosystem.

Inhaltsverzeichnis

1 Einleitung

1.1 Thema und Problemstellung

Digitale Bilder gewannen in der nahen Vergangenheit eine immer größere Bedeutung für die Menschen, was sich unter anderem an den stetig weiter steigenden Datenmengen an Bildern in sozialen Netzwerken beobachten lässt. Facebook muss beispielsweise ca. 350 Millionen hochgeladene Bilder pro Tag speichern, verarbeiten und auch wieder abrufen können [22]. Der Anstieg der Datenmenge hängt vor allem mit dem gesunkenen Aufwand und Kosten für eine Einzelperson zusammen, um ein Bild zu produzieren, zu speichern, zu veröffentlichen und auch zu bearbeiten. Werkzeuge für diese Tätigkeiten werden den Benutzern meist unentgeltlich von den jeweiligen Plattformen zur Verfügung gestellt. Entscheidend ist dabei ebenfalls die hohe Verbreitung und Nutzung von Smartphones und den darin integrierten Kameras, die es jedem ermöglichen, ein digitales Bild aufzunehmen. In Diskrepanz zu der Einfachheit der Vorgänge zur Produktion und Verbreitung eines Bildes steht dabei die Komplexität der Weiterverarbeitung und Organisation der gesammelten Daten, wie in der Abbildung 1.1 dargestellt wird. Einzelpersonen könnten dieses Problem von umfangreichen privaten Bildersammlungen kennen. Für etablierte Anbieter mit hohem Bilddatenverkehr, wie zum Beispiel große soziale Netzwerke, wird das Organisieren, die Beschreibung, das Finden und die Rückgabe von Bildern mit steigenden Datenmengen eine immer größere Herausforderung im Bezug auf Aufwand und Kosten.[20]

Um mit den schieren Mengen an Informationen umgehen zu können, müssen effiziente Verfahren zur Speicherung, Verarbeitung und auch zur Abfrage/Suche entwickelt werden. Das Themengebiet der Beschreibung und Auffindung von relevanten Informationen, in Form von Dokumenten, in Bezug auf eine Suchanfrage bezeichnet man dabei als *Information Retrieval (IR)*. Historisch bedingt handelt es sich bei solchen Dokumenten meist um Arten von Textdokumenten, die mittels einer textbasierten Suchanfrage sortiert und zurückgegeben werden. Bei Anwendungen, in denen es sich um Informationen in Form von Bildern oder Videos handelt, spricht man deswegen spezieller von *Visual Information Retrieval* (VIR)[20]. IR und speziell VIR Systeme bieten dabei nicht nur geeignete Lösungen für die vorher beschriebene Problematik an, sondern ermöglichen es, durch die Menge der Daten, beziehungsweise den daraus

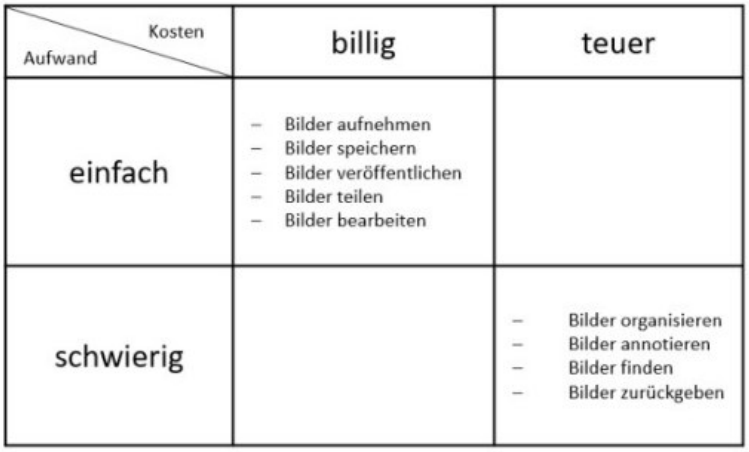

Abbildung 1.1: Diskrepanz zwischen Tätigkeiten der Bildproduktion und -verarbeitung

In dieser Abbildung soll die Diskrepanz zwischen Bildproduktion und -verbreitung zu der Bildorganisation und -suche dargestellt werden, indem Kosten und Aufwand verschiedener Tätigkeiten eingeordnet werden. Nach Vorbild von [20].

ableitbaren Erkenntnissen, und gestiegener technischer Leistungsfähigkeit, auch neue Verfahrensweisen zur Verarbeitung und Abfrage von Informationen zu entwickeln. Im Bereich der VIR Systeme gibt es beispielsweise Entwicklungen, um eine Sammlung visueller Informationen statt mit einer textbasierten Suche mit einer bildbasierten Suchanfrage durchzuführen. Im Anwendungsfall würde ein Nutzer zum Finden relevanter Bilder (oder Videos) ein Beispielbild als Suchanfrage übergeben und das System als Antwort visuell oder semantisch ähnliche Ergebnisse zurückliefern. Konzepte für Anwendungen dieser Art existieren zwar schon seit einigen Jahrzehnten, diese wurden jedoch erst vereinzelt und in naher Vergangenheit in der Praxis erprobt und umgesetzt [1]. Einer der am häufigst diskutierten Anwendungsfälle betrifft die Produktsuche im E-Commerce Bereich, bei der es in einem Online-Shop möglich sein soll, ein Bild des gesuchten Produktes an den Shop zu übergeben, welcher daraufhin Produkte mit ähnlichem Aussehen zurückliefert. Trotz des groß eingeschätzten Nutzens und der Tatsache, dass es schon einige Produkte oder Services gibt, die eine solche Einbindung einer Bildersuche anbieten, findet diese Methode in der Praxis derzeit nur wenig Verbreitung [17]. Welche Gründe es dafür gibt, welche Probleme damit zusammenhängen und wie es möglich ist, ein solches System zu implementieren, will ich in dieser Bachelorarbeit diskutieren. **Als Ziel soll dabei die Umsetzung einer bildbasierten Produktsuche im Shopsystem SAP Hybris Commerce, die Auswertung der Leistung dieses Ansatzes und die Analyse aufgetretener Probleme und deren Lösungen stehen**. Besonders die letzten beiden Aspekte sollen durch die Auswertung theoretischer und praktischer Erkenntnisse einen wissenschaftlichen Mehrwert schaffen und Erkenntnisse zur Weiterentwicklung und Verbreitung des Ansatzes im E-Commerce liefern. Bei dieser Arbeit unterstützt mich die Digitalagentur *dotSource GmbH* sowohl durch Material zur Umsetzung als auch durch einen externen Betreuer.

1.2 Über die Firma dotSource

Die *dotSource GmbH* wurde im Jahre 2006 als Agentur zur Beratung und Umsetzung von Projekten im Bereich des E-Commerce gemeinsam von Christian Grötsch und Christian Malik gegründet. Bis heute ist die Mitarbeiterzahl des Jenaer Unternehmens auf über 200 gewachsen und auch die Vielfalt des Produkt- und Leistungsangebots hat sich weiterentwickelt. Die *dotSource GmbH* bezeichnet sich selbst als Digitalagentur, die ihre Kunden beim Herstellen und Pflegen digitaler Kundenbeziehungen unterstützt. Die dotSource will helfen, Marketing, Vertrieb und Service ihrer Kunden zu digitalisieren. Dies wird vor allem im Umfang der angebotenen Leistungen deutlich, welche die Beratung der Kunden und die Konzeption, Umsetzung und Betreuung der erarbeiteten Lösungen umfasst. Auch die Produktpalette ist vielfältig und reicht von E-Commerce- über Kundenbetreuungs- bis hin zu Marketinglösungen [33]. Vor allem die Innovationsbereitschaft und die Einbindung neuer Ideen wird in der dotSource unterstützt, was durch die Veröffentlichungen eigener Whitepaper [8] und die Unterstützung vieler Studenten und Auszubildender deutlich wird. Auch die Ausarbeitung dieser Bachelorarbeit findet im Rahmen dieser Innovationsbereitschaft statt, um die Möglichkeiten, die eine bildbasierte Produktsuche im Bereich E-Commerce bietet, zu erforschen und eventuelle Problemstellungen aufzudecken. Bei der Erstellung unterstützte mich die dotSource durch die Bereitstellung von Bildungsmaterial und vor allem durch einen externen Betreuer, um mich bei fachlichen Fragen zu beraten.

1.3 Stand der Technik

Zuallererst soll hier erwähnt werden, dass das Forschungsfeld des Visual Information Retrieval ein sehr interdisziplinär geprägter Bereich ist, der beispielsweise durch Einflüsse aus der Bildverarbeitung, des Information Retrieval, des maschinellen Lernens und vielen mehr bestimmt wird [4]. Auch die Anzahl der möglichen Einsatzgebiete ist groß, beispielsweise sind Anwendungen im Bereich der Architektur, der Biochemie, im E-Commerce oder auch für die Bildung zu nennen. Diese Arbeit soll ihren Fokus hauptsächlich auf den Bereich des E-Commerce legen, wobei zu erwähnen ist, dass Entwicklungen in anderen Bereichen und Forschungsgebieten ebenfalls großen Einfluss auf den Bereich des E-Commerce haben können, was auch vice versa gültig ist [17].
Die Suche von Dokumenten basiert in den allermeisten Fällen auf Suchanfragen, welche aus einem Wort oder Textabschnitt bestehen. Vor allem in Webanwendungen, beispielsweise im E-Commerce, ist diese Suchart weit verbreitet, basiert allerdings immer noch auf den Grundprinzipien, welche schon in früheren Dokumentensammlungen, wie Bibliotheken, zur Sortierung und Suche der Texte und Abbildungen verwendet wurden. Auf diesen Erkenntnissen bauen auch

die Bemühungen auf anstatt von Text ein Bild als Suchanfrage zu verwenden, um visuell oder semantisch ähnliche Dokumente zu finden. Die Grundlagen für eine solche Suche wurden vor circa 20 Jahren gelegt und öffneten den Weg für weitere Entwicklungen und Forschungen in diesem Gebiet [4]. Der Vorteil einer bildbasierten Suche besteht in der Möglichkeit, von Bildern sehr komplexe Zusammenhänge in einer einfachen Form darzustellen, was mit reinem Text fast unmöglich ist. Dies zieht jedoch auch nach sich, dass eine Suche auf Grundlage von Bildern und deren visuellen oder semantischen Inhalten einen hohen Schwierigkeitsgrad aufweist [20]. Vor allem die Frage der mathematischen Beschreibung von Bildinhalten ,der Art der Berechnung von Ähnlichkeiten und der Vergleich von Bildinhalten sind Probleme, welche die frühe Zeit der Forschung dieses Feldes prägten, allerdings auch heute noch von großer Relevanz sind. In den frühen Jahren der Forschung beschäftigte man sich, aufgrund der geringeren Komplexität, vor allem mit der Analyse und dem Vergleich visueller Eigenschaften und Ähnlichkeit. Umsetzungen und Entwicklungen bezogen sich auf sehr kleine Anwendungsbereiche und es wurden Grundlagen für weitere Entwicklungen geschaffen [4].

Ein damals wie heute prominentes Problem ist die sogenannte *Semantic Gap*, welche im Lauf der Arbeit noch genauer erläutert wird. Kurz gesagt beschreibt sie die Diskrepanz zwischen automatisch extrahierbaren Bildeigenschaften und der semantischen Bedeutung dieser Inhalte für einen Betrachter [23]. Neuere Entwicklungen im Feld von VIR Systemen beschäftigen sich deshalb sowohl mit den Möglichkeiten der Verbreiterung des Anwendungsbereiches als auch mit den Chancen, welche eine semantische Bildanalyse für derartige Anwendungen birgt. Diese Betrachtungen sind aktuell Forschungsgegenstand und werden durch neue Entwicklungen immer weiter vorangetrieben. Da es sich bei dem Forschungsfeld rund um VIR Systeme um ein sehr interdisziplinäres Gebiet handelt, können auch zunächst fachfernere Themen einen großen Einfluss nehmen. Vor allem maschinelles Lernen und künstliche Intelligenz bieten für die semantische Analyse und Verbreiterung des Anwendungsbereiches ein unglaubliches Potential.

Große Digitalkonzerne wie Google [11], Ebay [9] oder Alibaba [35] bieten bereits VIR Lösungen mit einer bildbasierten Suche an. Aufgrund der immensen Mengen an Beispieldaten sowie personenbezogenen Informationen drängen diese Anbieter zuerst in die Lücke, welche eine bildbasierte Suche im Bereich des E-Commerce aufgetan hat. Die Beschreibung und Suche von komplexen Inhalten verspricht dabei eine effektivere und effizientere Suche, wovon sich die Unternehmen logischerweise einen höheren Umsatz und Gewinn erhoffen. Unter anderem aus diesem Grund wird die Verwendung einer bildbasierten Suche im E-Commerce Bereich immer wieder erwähnt. Aus welchen Gründen derartige Ansätze allerdings noch nicht die versprochene Verbreitung gefunden haben, auf welchen grundlegenden Problemen diese basieren und wie diese eventuell gelöst werden können, soll in dieser Arbeit diskutiert werden. Dabei wird im Verlauf der Arbeit hauptsächlich auf etablierte Grundlagen eingegangen, welche die Grundprinzipien derartiger Anwendungen und deren Probleme verdeutlichen. Bei passen-

den Gegebenheiten werden allerdings auch immer wieder Ausblicke in künftige Entwicklungen gewagt.

1.4 Ablauf und Vorgehensweise

Das Thema dieser Arbeit umfasst die Implementierung einer bildbasierten Produktsuche im Shopsystem SAP Hybris Commerce, sowie die Auswertung der Leistung dieser Umsetzung und der Analyse der aufgetretenen Probleme und deren Lösungen. Dieser Abschnitt soll den Ablauf des Projektes strukturiert darstellen, wobei tiefergehende Inhalte und Erläuterungen in den entsprechenden Kapiteln zu finden sind. Zuerst erfolgte eine längere Phase des Wissensaufbaus, um grundlegende Zusammenhänge des Forschungsfeldes rund um VIR Anwendungen zu verstehen und überblicken zu können. Auf dieser Grundlage konnte die Konzeption und Planung der Anwendung beginnen, wobei sich diesbezüglich vor allem an den Designeckpunkten, welche in [23] aufgestellt wurden, und den Strukturerklärungen aus [20] orientiert wurde. Auf der Basis eines grundlegenden Konzeptes und einer groben Aufbau- und Funktionsweise konnte eine Auswahl der Systemkomponenten getroffen werden. Nach dieser Entscheidung sollten zur weiteren Orientierung und Zielsetzung für das Projekt die Designfragen aus [23] verwendet werden. Die Beantwortung lieferte eine genauere Ausrichtung für die Umsetzung der Anwendung und nahm bereits Einfluss auf die praktische Implementierung. Diese praktische Umsetzung war der zweite größere Abschnitt der Arbeit und wurde mit Hilfe von Herangehensweisen in Anlehnung an ein agiles Projektvorgehen bearbeitet. So wurden zuerst die grundlegenden Funktionen umgesetzt und die Anwendung schrittweise verbessert, um die gesetzten Ziele zu erfüllen. Die dabei aufgetretenen Probleme, welche nicht in der Vorbetrachtung erkannt wurden, ließen sich meist durch einfachere Anpassungen lösen und wurden für die Auswertung aufgezeichnet. Nach der fertiggestellten Implementierung einer bildbasierten Produktsuche folgte eben jene Phase der Auswertung. Diese war bereits in der Konzeptphase bedacht worden und beeinflusste damit von Beginn an das System. Sowohl die genauere Planung der Bewertung der realisierten Implementation als auch die Umsetzung in einer möglichst automatisch funktionierenden Art und Weise wurden in diesem Arbeitsschritt verwirklicht. Im letzten Teil der Arbeit wurden die gewonnenen Informationen der Auswertung zur Bewertung des realisierten Systems verwendet und die aufgetretenen Probleme analysiert sowie Lösungen beziehungsweise Lösungsvorschläge vorgestellt.

Diese schriftliche Ausarbeitung dient dabei mehreren Zwecken. Zum einem stellt sie die notwendigen Wissensgrundlagen bereit, um derartige Systeme zu verstehen und die umgesetzte Anwendung analysieren und erweitern zu können. Sie soll ebenfalls als Grundlage zur Auswertung verwendet werden können, indem die erstellten Daten und die Art ihrer Gewinnung aufgezeichnet wurden. Nicht zuletzt bietet sie allerdings auch eine Art Anleitung zum Vorge-

hen bei der Erstellung eines ähnlichen Systems, da die einzelnen Arbeitsschritte ausführlich erklärt werden. Die schriftliche Ausarbeitung kann auch als eine Art Dokumentation für die umgesetzte Anwendung dienen, um deren Verwendung und Einbindung in andere Projekte so einfach wie möglich zu gestalten. Schlussendlich sollte sie allerdings den Hauptzweck erfüllen, die Vorgehensweise, Umsetzung, Auswertung und die Ergebnisse dieses Projektes darzustellen und zu erläutern, um einen wissenschaftlichen Mehrwert in verschiedenen Formen zu liefern.

2 Theoretische Grundlagen eines CBVIR Systems

Das folgende Kapitel soll den grundsätzlichen Aufbau eines IR beziehungsweise eines CBVIR Systems zunächst einmal verständlich und grundlegend erklären, bevor in den anschließenden Abschnitten noch spezieller auf andere Konzepte wie das der *Merkmale* oder *Distanzmetriken* eingegangen wird. Diese bedeutenden Komponenten werden in diesem Zug genauer beschrieben und es werden verschiedene Konzepte und Implementationen vorgestellt. Um eine Einordnung zu ermöglichen, möchte ich vorher noch einmal die genauen Definitionen und Spezifikationen der hier benutzten Systembegriffe erläutern und in Beziehung setzen.

Information Retrieval System (IR):

Ein Information Retrieval System ist, wie bereits genannt, ein System, welches aus einer Sammlung von Dokumenten die relevantesten im Bezug auf eine Anfrage sucht und zurückliefert [20]. Der Name gilt als Oberbegriff von Systemen die eine solche Funktion erfüllen.

Visual Information Retrieval System (VIR):

Visual Information Retrieval Systeme sind Systeme, bei denen die Sammlung von Dokumenten aus Bild- oder Videodokumenten besteht[14]. VIR Systeme sind damit eine Untergruppe der IR Systeme.

Content-Based Visual Information Retrieval System (CBVIR):

Content-Based Visual Information Retrieval Systeme sind wiederum eine Untergruppe der VIR Systeme, bei denen zur Suche und dem Vergleich von Dokumenten keine Metadaten oder Annotationen, sondern der visuelle Inhalt der Bilder oder Videos verwendet wird [23].

Als Anmerkung sollte noch ergänzt werden, dass der folgende Aufbau für IR Systeme und damit auch für die genannten Untergruppen gültig ist. Da es sich bei der hier geplanten Lösung um ein CBVIR System handelt, werde ich in den nachfolgenden Abschnitten jedoch hauptsächlich über für diese Systeme relevante Informationen schreiben, da weitere Ausführungen den Umfang der Arbeit übersteigen würden.

2.1 Struktur und Funktionsweise

Die Grundlagen von IR Systemen reichen bis in die Zeiten der ersten Sammlungen von Dokumenten und Informationen zurück, zum Beispiel in Form von Bibliotheken. Eine solche Sammlung ist in einer bestimmten Art und Weise organisiert, was sowohl eine Einordnung neuer Dokumente als auch die Suche nach bestimmten Informationen erlaubt, wobei diese Operationen früher ausschließlich manuell ausgeführt wurden. Auf dem gleichen Prinzip basieren auch moderne / automatische IR und VIR Systeme.[10]. In diesem Kapitel sollen die Grundlagen des Aufbaues und der Arbeitsweise dieser automatischen Systeme betrachtet werden.

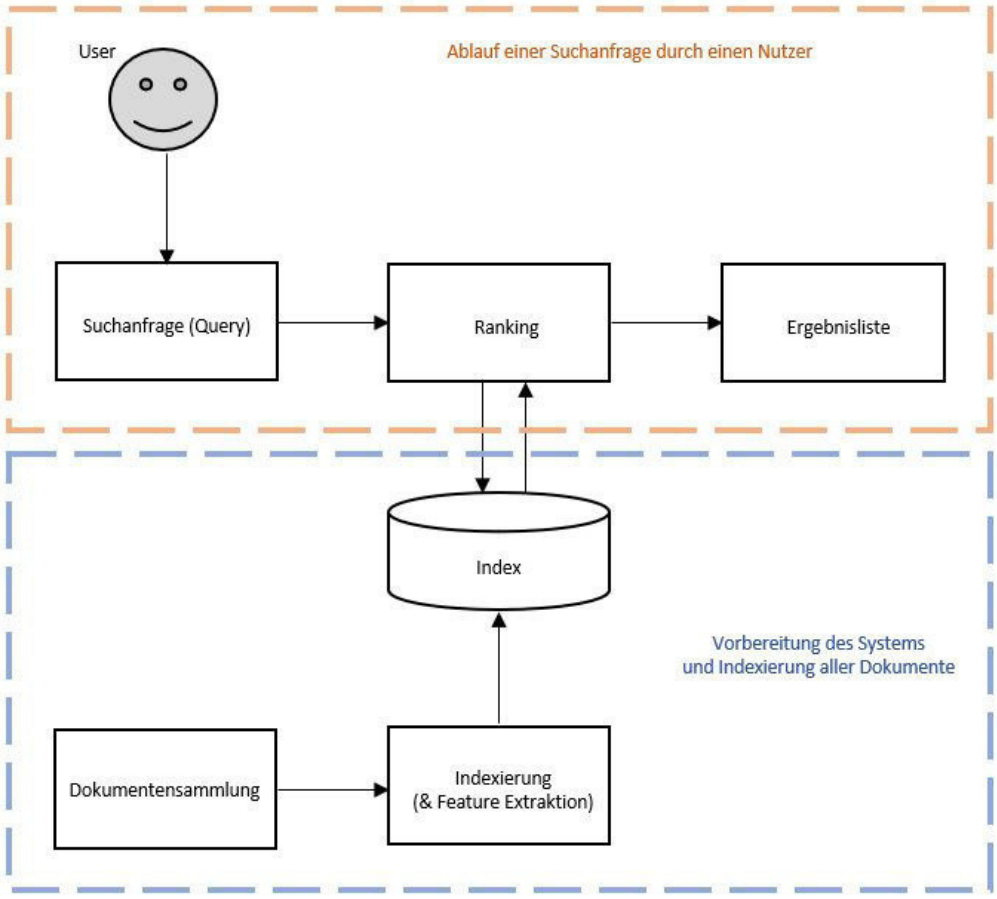

Abbildung 2.1: Aufbau eines IR Systems

Diese Abbildung zeigt den grundlegenden Aufbau und die Funktionsweise eines IR Systems, die auch für CBVIR Anwendungen gültig sind. Der orangefarbene Bereich umfasst dabei den Prozess, der bei der Suchanfrage eines Nutzer durchlaufen wird, während der blaufarbene Bereich die Prozesse zur Vorbereitung des System und die Indexierung der Dokumente darstellt. Nach Vorbild von [20].

Im Schaubild 2.1 wird die grundsätzliche Struktur eines IR Systems und auch die verschiedenen Einflüsse auf ein solches System gezeigt. Als Grundlage dient dabei eine Sammlung aus Dokumenten, auch *Corpus* genannt. Die enthaltenen Dokumente können, je nach An-

wendung, verschiedenste Formate besitzen. In den meisten Fällen handelt es sich hierbei um Textdokumente, was vor allem an der Menge als auch an den einfachen Merkmalen dieser Art von Informationen liegt. Mit dem Aufkommen anderer Formate, zum Beispiel Bild oder Video, wurde es notwendig, IR Systeme auch für Anwendung auf solche Arten von Informationen anzupassen beziehungsweise zu entwickeln [31]. Grundsätzlich funktionieren IR und VIR Systeme trotz allem nach den gleichen Grundprinzipien, die im folgenden einmal erläutert werden sollen.

Um einen Vergleich und damit eine Ordnung der Dokumente einer Sammlung zu erlauben, müssen Merkmale festgelegt werden, welche die Dokumente und die enthaltenen Informationen in einer angemessenen Art und Weise beschreiben. Diese Merkmale, auch als *Features* bezeichnet, können relativ einfach, zum Beispiel die Wortanzahl eines Textes, oder auch komplexer, beispielsweise eine Kombination mehrerer simpler Features, sein. Je nach Art der Informationen müssen verschiedene Features verwendet werden, um eine möglichst genaue Beschreibung und Unterscheidung der Dokumente zu ermöglichen. Die dabei notwendige Auswahl oder auch die Entwicklung neuer Merkmale ist ein immens wichtiger Bestandteil eines IR Systems und wird in dieser Arbeit noch einmal genauer diskutiert. Grundlegend analysiert ein Feature den Inhalt eines Dokumentes und repräsentiert dies durch einen diskreten Wert, wie zum Beispiel die absolute Anzahl von Wörtern in einem Textdokument. Dieser Vorgang wird auch als *Feature Extraction* bezeichnet.

Die bei der Feature Extraction erhaltenen Werte werden in den meisten Fällen zusammen mit den Dokumenten in einem Feld einer Datenbank abgelegt. Um zu verhindern, dass bei einer Suche jedes Feld dieser Datenbank durchlaufen werden muss, wird ein Index der Einträge für einen schnelleren und effektiveren Zugriff verwendet. Ein Index beschreibt eine Datenstruktur, die eine effiziente Suche nach den in ihm enthaltenen Dokumenten ermöglicht. Die Implementierung ist dabei in den verschiedensten Formen möglich, die beispielsweise Listen oder auch Baumstrukturen umfassen. Die verwendete Form des Index ist stark von der Art der Daten und der Verwendung eines Systems abhängig. Die meisten Datenbanken beinhalten bereits die Funktionalitäten zur Erstellung eines Indexes für ihre Inhalte, wobei auch häufig eine von der Datenbank getrennte Umsetzung des Index verwirklicht wird. Der Index ist die Grundlage zur Suche und zum Vergleich der Dokumente und sollte deswegen in der Designphase gut durchdacht und angepasst werden. Mit dem erstellten Index ist die Grundlage zur Suche nach Dokumenten geschaffen [23].

Eine Suche wird in einem IR System durch eine Suchanfrage, auch *Query* genannt, eines Benutzers initiiert. Dieser Nutzer will durch die Suche ein bestimmtes Informationsbedürfnis stillen, dessen genaue Ausprägung von verschiedenen Faktoren abhängt. Der Einfluss des Benutzers, sowie seiner Einstellungen und Erwartungen auf ein IR System werden in dieser Arbeit noch einmal gesondert diskutiert. Die Suchanfrage kann, wie auch die Dokumente, verschie-

denste Formate vorweisen, beispielsweise Text oder Bild. Für das Verständnis eines IR Systems ist es wichtig zu verinnerlichen, dass es sich bei der Suchanfrage um ein eigenständiges Dokument handelt. Selbst wenn die Anfrage nur ein einzelnes Wort beinhaltet, kann diese als Textdokument betrachtet werden.

Wie schon erwähnt, ist es das Ziel eines IR Systems, aus einer Sammlung von Dokumenten die relevantesten Einträge im Bezug auf eine Suchanfrage zu finden und zurückzuliefern. Zum Zeitpunkt der Suche wurden die bisher erläuterten Schritte im Vorfeld durchgeführt, wodurch die beschreibenden Merkmale der Dokumente in einem Index abgelegt sind. Der erste Schritt der Suche besteht aus der Durchführung der Feature Extraction auf das Dokument der Suchanfrage, welches dadurch in der gleichen Weise beschrieben ist, wie die gesamte Sammlung an Dokumenten. Im zweiten Schritt werden die berechneten Merkmale der Suchanfrage mit den im Index gespeicherten Werten verglichen. Dafür existieren verschiedenste sogenannte Distanzmetriken, die einem jeden Dokument der Sammlung einen Ähnlichkeitswert, auch *Score* genannt, zuweisen. Mit diesem Score kann daraufhin eine Rangfolge, auch als *Ranking* bezeichnet, erstellt werden, welche die ähnlichsten Dokumente der Sammlung in Bezug auf die Suchanfrage enthalten sollte. Auch den dabei immens wichtigen Distanzmetriken wird hier noch einmal ein gesondertes Kapitel gewidmet [20].

2.2 Merkmale und Merkmalsextraktion

Um in einem VIR System eine möglichst effektive und effiziente Suche zu ermöglichen, ist es entscheidend in welcher Form die indexierten Dokumente beschrieben werden. Dazu dienen Merkmale, auch *Features* genannt, welche die Farb- und Helligkeitswerte einzelner Pixel sowie deren Koordinaten benutzen, um sowohl einfache als auch komplexere Beschreibungen eines Bildes zu erstellen. Ein nachvollziehbares Beispiel hierfür ist das eher simple Merkmal bestehend aus einem Histogramm der Helligkeitswerte eines Schwarz-Weiß-Bildes, welches die Häufigkeit der verschiedenen Helligkeiten darstellt (siehe Abbildung 2.2). Dieses Feature beschreibt das jeweilige Bild zwar in gewisser Hinsicht, kann jedoch nicht wirklich zur vollständigen Identifikation benutzt werden, da es keine Informationen über die örtliche Verteilung der Pixelwerte wiedergibt (siehe Abbildung 2.3).

Zur umfänglichen Charakterisierung eines Dokumentes können daher so viele Features verschiedener Komplexitäten verwendet werden, wie nötig sind. Jedes zusätzlich in die Betrachtung eingehende Merkmal zieht dabei natürlich auch eine, der Komplexität entsprechende, höhere Rechenlast und Rechendauer nach sich. Neben diesen Betrachtungen sollten die benutzten Merkmale ebenfalls eine möglichst große Diskrimination der Bilder erlauben und keine Redundanzen erzeugen. Einfacher formuliert sollte ein Merkmal also eine möglichst große Aussage-

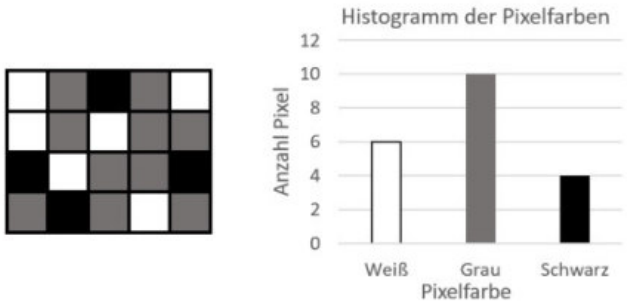

Abbildung 2.2: Beispiel eines Farbhistogrammes

Diese Darstellung zeigt auf der linken Seite ein 4x5 Pixel großes Beispielbild mit drei verschiedenen Pixelfarben. Das Histogramm auf der rechten Seite stellt die absolute Häufigkeit des Auftretens einer jeden Pixelfarbe dar und kann als ein sehr einfaches Merkmal angesehen werden.

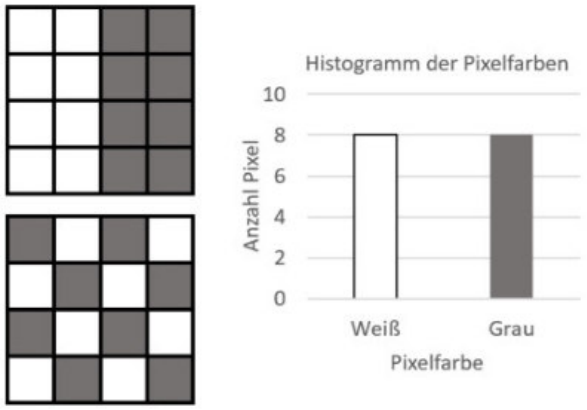

Abbildung 2.3: Problem der örtlichen Beziehungen in Histogrammen

Diese Abbildung zeigt auf der linken Seite zwei 4x4 Pixel große Beispielbilder mit zwei verschiedenen Pixelfarben. Wie zu erkennen ist, unterscheidet sich deren örtliche Verteilung extrem. Dies spiegelt sich allerdings nicht in dem Histogramm auf der rechten Seite wieder, welches für beide Bilder gültig ist. Daraus lässt sich ableiten, dass ein solches Histogramm nicht zur vollständigen Beschreibung ausreicht, da keine ortsbezogenen Informationen verarbeitet werden. Nach Vorbild von [20].

kraft bezüglich der Beschreibung und Relevanz des Dokumentes liefern [23] und sich nicht mit anderen Features in ihren Aussagen überschneiden. Ebenfalls entscheidend ist die Stabilität eines Merkmales, welche beschreibt, wie gut es auf Veränderungen eines Bildes im Gegensatz zu dessen Originalform, zum Beispiel durch Rotation oder Skalierung, reagiert und äquivalente Ergebnisse liefert. Die Funktionalität eines jeden Merkmales ist dabei logischerweise an die jeweilige Problemstellung beziehungsweise Anwendung gekoppelt. In der Designphase eines VIR Systemes ist deshalb der Abschnitt der sogenannten *Feature Selection* besonders wichtig. Bei diesem Prozess sollen die besten Features in Bezug auf eine Problemstellung gefunden und ausgewählt werden, um die bereits benannten Auswahlkriterien bestmöglich zu erfüllen [17]. Die resultierende Auswahl enthält natürlicherweise trotzdem noch Merkmale mit unter-

schiedlicher Aussagekraft und eventuellen Redundanzen, weshalb einigen Features eine stärkere Aussagekraft zugemessen werden sollte als anderen. Dies wird über ein sogenanntes Gewichtungsschema geregelt, in dem einem jeden Merkmal ein Gewichtungsfaktor entsprechend der Relevanz für die Beschreibung eines Bildes zugeordnet wird. Dieser Faktor bestimmt in der mathematischen Beschreibung des Bildes den Einfluss des jeweiligen Features auf das Endergebnis. Sowohl der Prozess der Feature Selection als auch die Erstellungen des Gewichtungsschemas können manuell oder auch automatisch durchgeführt werden. Vor allem durch neuere Entwicklungen in den Bereichen der künstlichen Intelligenz und des maschinellen Lernens entstehen immer besser konfigurierte und optimierte Systeme zur Lösung bestimmter Problemstellungen.

Bei dem Prozess der Feature Selection kann man aus einer nahezu unendlichen Menge von Merkmalen wählen, da man sowohl auf bereits erstellte als auch für die jeweilige Problemstellung neu konstruierte Features zurückgreifen kann. All diese Merkmale können anhand ihres Abstraktionsgrades in eine von drei Gruppen eingeordnet werden, welche hier genauer erläutert werden [23]:

Niedriger Abstraktionsgrad:

Die Merkmale dieser Kategorie basieren auf den *rohen* Pixelwerten, also Farb- und Helligkeitswerten, welche jedoch nur noch in einem geringen Maße verarbeitet und damit fast direkt zur Beschreibung eines Bildes verwendet werden. Features dieser Art haben zwar nur eine beschränkte Möglichkeit der Charakterisierung eines Bildes, sind jedoch im Gegenzug einfach und schnell zu berechnen.

Mittlerer Abstraktionsgrad:

In dieser Kategorie werden die Informationen der Pixel genutzt, um das Bild in einzelne Regionen, auch *Blobs* bezeichnet, zu unterteilen. Diese Bildbereiche zeichnen sich meist durch ähnliche visuelle Eigenschaften, beispielsweise Farbe oder Textur, aus und tragen deshalb meist schon eine Art semantische Bedeutung in sich, die jedoch noch nicht weiter beachtet wird. Diese etwas komplexere Form der Merkmale kann Bilder schon in einer (visuell) umfänglichen Art und Weise beschreiben, ist dafür allerdings auch schwerer und langsamer zu berechnen.

Hoher Abstraktionsgrad:

Features dieser Kategorie versuchen dem Bild im Gesamten, als auch einzelnen Teilgebieten oder -objekten eine semantische Information, also eine Art Bedeutung zuzuordnen. Die Analyse der Semantik eines Bildes ist eine der größten Herausforderungen im Bereich der VIR Systeme, auf die ich in dieser Arbeit noch spezieller, beispielsweise bezüglich der *Semantic Gap*, eingehe. Eine erfolgreiche Umsetzung eines Merkmales dieser Art kann nicht nur das Bild visuell, sondern auch inhaltlich beschreiben, ist jedoch nur sehr schwer

und langwierig zu berechnen. Zudem sind Implementierungen dieser Form bisher meist nur auf sehr spezielle Einsatzfälle ausgerichtet und damit kaum allgemeiner einsetzbar.

In den meisten Systemen bilden Merkmale mit niedrigem Abstraktionsgrad die Grundlage, während Features mittlerer Abstraktion je nach Fall für bessere Bewertung hinzugefügt werden [23]. Merkmale der dritten Kategorie kommen aus den genannten Gründen höchst selten zum Einsatz, auch wenn sich ihr Auftreten durch den Einsatz von künstlichen Intelligenzen und maschinellem Lernen in den letzten Jahren verstärkt hat, welche die Optimierung solcher komplexen Features deutlich vereinfacht.

Im Fall von VIR oder spezieller CBVIR Systemen werden meist drei Arten von Bildeigenschaften zur Beschreibung mittels der jeweiligen Merkmale herangezogen. Diese Eigenschaften basieren auf den Pixelwerten des Bildes und deren örtlichen Zusammenhängen, was in der folgenden Erläuterung genauer erklärt wird. Ein Feature eines VIR Systems kann dabei Kombinationen dieser Eigenschaften abdecken oder auch nur sehr schwer einer Kategorie zurechenbar sein. Die erwähnten Kategorien finden in Theorie und Praxis dieses Forschungsfeldes immer wieder Beachtung und sollten deshalb hier einmal beschrieben werden [17].

Farbbasierte Merkmale:

Merkmale dieser Kategorie basieren auf den Farben, welche in einem Bild enthalten sind. Farbeigenschaften sind für Menschen sehr einfach wahrzunehmen und bilden eine starke Grundlage zur Identifikation von Objekten und semantischen Zusammenhängen. Für die Anwendung in VIR Systemen eignen sich entsprechende Merkmale vor allem, da sie einfach aus den Pixelwerten abgeleitet werden können und eine hohe Aussagekraft bezüglich einfacher als auch komplexerer Bildeigenschaften besitzen. Zur Darstellungen der Farb- und auch Helligkeitsinformationen wurden verschiedenste Farbmodelle entwickelt. Ein jedes Modell spezifiziert ein Koordinatensystem zur Einordnung von Farbe, den sogenannten Farbraum, in dem jede enthaltene Farbe durch einen Datenpunkt dargestellt werden kann. Jedes Farbmodell wurde zu einem bestimmten Zweck konstruiert, welcher mit dem jeweiligen Einsatzgebiet zusammenhängt. Daraus folgend birgt ein jedes dieser Modelle auch bestimmte Stärken und Schwächen in Bezug auf eine Verwendung in einem VIR System, welche bei der Planung klar abgewogen werden müssen. Das wohl prominenteste Beispiel ist das *RGB Farbsystem*, welches aus einem kartesischem Koordinatensystem mit den Achsen Rot, Grün und Blau, also den primären Farben der menschlichen Farbwahrnehmung, besteht und auf einen Wertebereich von [0,1] normalisiert ist. Ein weiteres Modell ist das *HSI Farbsystem* und verschiedene Abwandlungen (HSV,HSB,...), welches auf der Trennung von Intensitäts- und Farbinformationen beruht und die Achsen Farbton (Hue), Sättigung (Saturation) und Intensität (Intensity) besitzt. Eine solche Trennung der Informationen setzt auch das *YCbCr Modell* um, dessen Achsen aus der Helligkeit und den Farbdifferenzen des Rot- und Blauanteils bestehen.

Tiefergehende Erläuterung zum Aufbau sowie den Vor- und Nachteilen verschiedener Farbsysteme bezüglich VIR Anwendungen können in [17] nachgelesen werden, da eine solche Erklärung den Umfang dieser Arbeit überschreiten würde. Mittels der gewonnenen Farbinformationen, die mit Hilfe eines Farbmodells dargestellt werden, können Merkmale zur Beschreibung eines Bildes erstellt werden. Als Ergebnis stehen sowohl sehr einfache Features, zum Beispiel das der dominantesten Farbe eines Bildes gemessen an deren Häufigkeit, etwas komplexere Merkmale, beispielsweise ein Farbhistogramm, welches die Häufigkeit aller Farben darstellt, oder auch sehr komplexe Beschreibungen, welche die örtliche oder stochastische Farbverteilung eines Bildes einbeziehen. Sowohl Farbinformationen als auch Farbmerkmale können zur Kombination und Verwendung mit Merkmalen basierend auf anderen Bildeigenschaften benutzt werden [20].

Texturbasierte Merkmale:

Die zweite Art von Merkmalen verwendet die Texturen in Bildern zu deren Beschreibung. Textur ist dabei ein Konzept, dessen Bedeutung den meisten Menschen intuitiv bekannt ist, aber das nur schwer genauer beschrieben werden kann. Eine oft verwendete Definition ist, dass es sich bei Textur um eine bestimmte, meist musterhafte, örtliche Anordnung von Helligkeiten, Farben oder besser gesagt Intensitäten in einem Bild handelt [20]. Es existieren auch andere Erklärungen dieses Begriffes, jedoch stimmen alle in bestimmten Punkten überein. Diese Punkte sagen aus, dass bei Texturen eine gewisse Varianz der Intensitätswerte der Nachbarpixel auftritt und dass es sich um eine homogene Eigenschaft handelt, die sich auf eine Größeneinheit bezieht, welche größer als die jeweilige Bildauflösung ist [17]. Eine Textur ist eine Eigenschaft, welche sich meist nur auf einen bestimmten Bereich eines Bildes beschränkt, welcher dadurch als eine Einheit mit ähnlichen Eigenschaften angesehen wird und auch eine klare Abgrenzung zu anderen Bereichen besitzt. Durch die Diversität der Ausprägung dieser Bildeigenschaft sind Texturen stark diskriminierend, lassen sich im Gegenzug allerdings schwer beschreiben und noch schwerer vergleichen. Um trotzdem eine Basis zur Darstellung verschiedener Texturen zu haben, existieren Modelle zu deren Beschreibung. Eine mögliche Art der Darstellungen findet mittels statistischer Methoden statt, welche mit Hilfe von Histogramminformationen eine Beschreibung und Vorhersage der Texturen erlaubt. Verwendung finden dabei unter anderem die statistischen Momente der Intensitäten sowie deren Varianz. Ein anderes Modell verwendet die Analyse von Ortsfrequenzen im Bild, um eine Beschreibung im Frequenzbereich zu liefern und die daraus resultierenden Koeffizienten zur Beschreibung und zum Vergleich von Texturen zu verwenden. Das letzte Modell versucht, Texturen als Ansammlung primitiver Texels [1] darzustellen. Diese Texels werden wiederum als eine Art kleines Muster in einer Textur beschrieben, welches sich in einer bestimmten Weise

[1] Eine Wortkombination aus Pixels und Texture.

örtlich wiederholt. Eine solche Beschreibung kann vor allem auf künstlich erstellte Texturen sehr gut angewandt werden. Mit Hilfe der genannten Modelle können Merkmale zur Beschreibung von Texturen erstellt werden. Ein solches Merkmal ist beispielsweise das Tamura Feature, welches eine Ansammlung von sechs einfachen Texturmerkmalen beinhaltet: Coarseness (Grobheit), Contrast (Kontrast), Directionality (Direktionalität), Line-likeness (Linienhaftigkeit), Regularity (Regularität) und Roughness (Rauheit). Ein weiteres verbreitetes Feature ist ein Kantenhistogramm oder auch *Edge Histogram* genannt. Dieses Feature ist beispielsweise Teil des MPEG-7 Standards und beschreibt die Häufigkeit des Auftretens bestimmter Kanten in einem Bild mit Hilfe eines Histogrammes. Das Bild wird dafür in gleich große Bereiche eingeteilt und für jeden dieser Blöcke wird eine dominante Kantenrichtung ermittelt, welche im folgenden Schritt der jeweiligen Gruppe im Histogramm zugeordnet wird[2] [20]. Die genannten und auch die vielen weiteren Merkmale basierend auf den Textureigenschaften des Bildes können natürlich mit anderen Features kombiniert werden.

Formbasierte Merkmale:

Die letzte Kategorie beschäftigt sich mit Merkmalen, welche auf den Formen, die in einem Bild zu erkennen sind, basieren. Eine Form wird als ein Abschnitt des Bildes beschrieben, welcher einheitliche Eigenschaften, wie Farbe oder Textur, besitzt und einen bestimmten und klar abgegrenzten Bereich an Bildpunkten umspannt. Jeder dieser Bereiche kann durch seine Farbe, seine Ausdehnungen, seine Fläche und seine Zentralität beschrieben und charakterisiert werden. Ein Bild kann abstrakt als eine Ansammlung und Zusammensetzung vieler Formen gesehen werden [14]. Diese Kategorie von Features birgt eine große Herausforderung bezüglich der Beschreibung bestimmter Formen und den darauf basierenden Vergleichen, da auch hier gilt, dass Form zwar ein intuitiv erfassbares Konzept darstellt, aber nur schwer, sowohl allgemein als auch im spezielleren gültig, definiert werden kann. Die verwendeten Modelle zur Beschreibung von Formen sind deshalb komplexer als die bisher genannten für Farbe und Textur, bieten im Gegenzug allerdings eine sehr große Aussagekraft, da Formen stark diskriminierend sind und auch eine hohe semantische Aussagekraft in sich tragen. Ein mögliches Vorgehen zur Erkennung von Formen in Bildern ist die Transformation in den Frequenzbereich um Intensitätsübergänge zu finden, welche meist an Formgrenzen auftreten. Ein Beispiel für ein derartiges Vorgehen sind Wavelet Transformationen. Um ein Bild in dieser Art möglichst kompakt zu beschreiben, können kleinere Koeffizientenwerte zusätzlich ignoriert beziehungsweise gestrichen werden. Ein weiteres Modell basiert auf der Darstellung von Formen mittels verschiedener Attribute oder durch einige aussagekräftige Bildpunkte der Form. Die Beschreibung und der Vergleich der Formen wird mit Hilfe von Trans-

[2] Im Falle des MPEG-7 Kantenhistogrammes existieren fünf Gruppen im Histogramm: vertikale Kanten, horizontale Kanten, 35 Grad Kanten, 135 Grad Kanten und nicht direktionale Kanten

formationen vorgenommen, die notwendig sind, um eine Form in eine andere umzuwandeln. Dies entspricht einer gewissen Veränderung der Formattribute oder -punkte. Mit Hilfe eines Wahlverfahrens, auch *Voting scheme* genannt, werden verschiedene Transformationsmöglichkeiten durchgegangen und die wahrscheinlichste beziehungsweise am häufigsten aufgetretene Transformation als Beschreibungsmittel verwendet. Als Beispiel eines Wahlverfahrens kann hier die *Hough Transformation* genannt werden, wobei für alle Vorgehen dieser Art gilt, dass eine große Anzahl an Attributen oder Punkten eine starke Steigerung des Rechenaufwands nach sich zieht. Es existieren noch verschiedene andere Verfahren zur Formerkennung, deren Erläuterung allerdings den Umfang dieser Arbeit überschreiten würde, jedoch in [17] nachgelesen werden können. Um eine erkannte Form zu beschreiben, werden verschiedene Methoden der computergestützen Geometrie verwendet, welche hier schon als Feature angesehen werden können, da mit ihrer Hilfe auch der Vergleich verschiedener Formen und Bilder durchgeführt wird. Zum einen kann eine Form durch eine endliche Anzahl von Punkten beschrieben werden, welche wiederum als Basis für den Vergleich genutzt werden können. Es kann aber auch der Ansatz über eine Beschreibung mittels Kurven, also durch Polynome, gewählt werden, welche wiederum andere Möglichkeiten des Vergleiches und der Auswertung bieten. Auch bei formbasierten Merkmalen gilt, dass man sie zur Konstruktion von neuen Features oder zur Kombination mit bereits bestehenden Merkmalen verwenden kann [17].

Die Merkmale der genannten Kategorien können Bilder auf verschiedene Art beschreiben. Sogenannte *globale Features* verwenden Beschreibungen, welche die Gesamtheit eines Bildes einbezieht. Schon erwähnte Beispiele sind Farb- oder auch Kantenhistogramme. Der Vorteil global erstellter Merkmale ist, dass sie sich recht einfach und unkompliziert berechnen lassen und einen einfachen Vergleich ermöglichen. Ihr Nachteil besteht darin, dass auch irrelevante Bildbereiche in die Berechnung einfließen können, was das Endergebnis wiederum verfälscht und abschwächt. Im Gegensatz dazu werden bei sogenannten *lokalen Features* einige wenige Bildpunkte oder kleinere Bildbereiche verwendet, um eine Beschreibung des Bildes zu erstellen. Diese *Points of Interest* sind Bildauschnitte mit besonders hoher Aussagekraft und Diskriminalität, zum Beispiel markante Farb- oder Kantenübergänge. Die Suche geeigneter Points of Interest benötigt eine Vorverarbeitung, auf deren Basis die lokalen Merkmale berechnet werden können. So wird meist in einer definierten Umgebung um einen solchen Punkt ein relevantes Merkmal ausgewertet. Als Beispiel kann hier die Scale-Invariant Feature Transform, auch als SIFT Feature bekannt, benannt werden, welche in einem Umkreis von 16x16 Pixel um den jeweiligen Interessenspunkt ein Histogramm der Richtungen der Intensitätsverläufe (Histogramm of gradient orientation) erstellt. Eine Bildbeschreibung durch ein lokales Feature, also mittels der Darstellung mehrerer Points of Interest, besitzt eine sehr große Aussagekraft und kann vor allem für weitere Schritte bezüglich einer Objekterkennung verwendet werden. Ihr Nach-

teil besteht in ihrer Komplexität, welche sowohl die Berechnung als auch den Vergleich zu anderen Bildern erschwert. Auch hier muss im Entwicklungsprozess eine Abwägung zwischen lokalen und globalen Merkmalen vorgenommen werden, um die bestmögliche Umsetzung einer Anwendung zu gewährleisten [20].

Als Resultat der Auswertung eines Bildes mit der Hilfe verschiedener Merkmale steht eine mathematische Repräsentation der verwendeten Features, welche wiederum das Bild genau beschreiben. Um einen Vergleich zwischen zwei Bildern und damit auch eine Relevanzbestimmung zur ermöglichen, müssen die Endergebnisse der Merkmalsauswertung in einer Form gespeichert werden, die eine Ähnlichkeitsbestimmung zulässt. Hierfür gibt es den Ansatz der Definition eines sogenannten *Merkmalsraumes*. Dieser Merkmalsraum ist eine Art Koordinatensystem basierend auf allen verwendeten Features, wodurch die beschriebenen Dokumente beziehungsweise Bilder durch einen Punkt in diesem Raum beschrieben werden können. Durch geeignete Distanzmetriken kann, unter der Voraussetzung, dass der Abstand mit der Ähnlichkeit zweier Bilder gleichgesetzt werden kann, eine Ähnlichkeits- und Relevanzbestimmung erfolgen. Eine stark etablierte Implementation ist die eines Merkmalsvektors, in dem jedes Element des Vektors das Ergebnis eines verwendeten Features darstellt und damit jeder Vektor ein Bild innerhalb des Raumes beschreibt. Distanzmetriken zwischen Vektoren, zum Beispiel die Euklidische Distanz, sind etablierte Metriken und einfach zu berechnen [17]. Auf andere Formen von Distanzmaßen und deren Verwendung zur Auswertung und Erstellung einer Ergebnisliste, sowie auf verschiedene Möglichkeiten zur Suche der relevantesten Dokumente will ich im nächsten Abschnitt genauer eingehen.

Wie hier dargestellt wurde, ist die Thematik der Merkmale ein sehr tiefgehender Bestandteil von VIR Systemen und deren Erstellung, welcher in dem Umfang dieses Kapitels nur angeschnitten werden konnte. Für tiefere Einblicke empfehle ich einen Blick in die Literatur, welche ich in diesem Kapitel angeführt habe.

2.3 Suchalgorithmen und Distanzmetriken

Nachdem die Dokumente in einen passenden Merkmalsraum eingeordnet wurden, kann auf dieser Grundlage eine Distanz- und somit auch eine Ähnlichkeitsbestimmung zwischen zwei Bildern in diesem Raum durchgeführt werden. Dies kommt allerdings erst beim tatsächlichen Suchvorgang mit Hilfe eines VIR Systems zum Einsatz, da hier das Beispielbild als Suchanfrage benutzt und mit den Bildern im System verglichen wird. Um den Suchprozess so effizient wie möglich zu gestalten, existieren verschiedene Vorgehensweisen zum Durchsuchen des Indexes nach Kandidaten zur Auswertung. Einige Vertreter, die auch in der praktischen Umsetzung Verwendung finden, sollen hier einmal vorgestellt werden [20]. Es ist allerdings wichtig zu

erwähnen, dass je nach verwendeter Suchstrategie auch der Index anders gestaltet und an die Suche angepasst werden muss.

Lineare Suche:

Die lineare Suche ist das einfachste Verfahren zum Durchlaufen eines Indexes. Wie es der Name schon vermuten lässt, wird hier der Index Eintrag für Eintrag durchlaufen und die Bewertung für jeden dieser Einträge durchgeführt. Diese Vorgehensweise ist vor allem wegen ihrer Einfachheit beliebt, führt allerdings bei sehr großen Datensätzen zu einer großen Rechendauer.

k-nächste-Nachbarn Suche:

Dieses Verfahren beruht auf dem Aufbau eines räumlichen Indexes, zum Beispiel mit Hilfe eines *R-Tree*, der eine räumliche Repräsentation der Indexeinträge in dem Merkmalsraum und damit auch deren örtliche Zusammenhänge beschreibt. Durch ein solches Verfahren kann die Suche nach relevanten Dokumenten auf die Berechnung der Lage des Beispieldokumentes im Merkmalsraum und darauf folgend das Finden der naheliegensten Indexeinträge in dessen Umgebung heruntergebrochen werden. Diese Suche nach den k-nächsten Nachbarn, wobei k für eine natürliche Zahl steht, ist durch den Aufbau des räumlichen Indexes effizienter und schneller als eine lineare Suche, allerdings ist sie auch komplexer umzusetzen.

Suche mittels Hashing:

Eine sogenannte Hash-Funktion hat die Aufgabe, eine große Datenmenge zu einem einzelnen Wert zu reduzieren. Für Sicherheitsanwendungen ist es dabei wichtig, dass die Ursprungsdaten nicht oder nur sehr schwierig aus dem resultierenden Hashwert abgeleitet werden können. Diese Eigenschaft der Hashfunktion spielt für die Anwendung zur Verbesserung der Suche allerdings keine große Rolle. Besonders wichtig für eine solche Verwendung ist das Vorkommen von Kollisionen. Als solche bezeichnet man das Auftreten des gleichen Hashwertes als Ergebnis zweier unterschiedlicher Eingabedaten. In den meisten Anwendungsgebieten von Hashverfahren sind Kollisionen ungewollt und sollen möglichst verhindert werden. Im Falle der Verwendung zur Suche in einem Index wird die Hashfunktion allerdings so konstruiert, dass es zu einer Kollision kommt, falls ein gewisses Maß der Ähnlichkeit zwischen zwei Eingabedokumenten besteht. Durch dieses Verfahren kann eine Vorauswahl der Indexeinträge ermittelt werden, welche dann mit Hilfe anderer Suchverfahren ausgewertet werden kann.

Die genannten Suchverfahren sollen dabei nur als Beispiel dienen und wurden vor allem ausgewählt, da sie in der Implementation verwendet werden. Es existieren verschiedene Prozesse für die Suche nach Dokumenten, welche nach entsprechenden Anpassungen der Indexstruktur auch für CBVIR Anwendungen eingesetzt werden können. Die Entwicklung weiterer und immer

effizienterer Suchverfahren ist ein eigenes Forschungsfeld und eine intensivere Auseinandersetzung würde den Rahmen dieser Arbeit übersteigen.

Um die mittels des Suchverfahrens gefunden Dokumente in einer nach Relevanz geordneten Liste zurückgeben zu können und dabei ein ideales Ergebnis zu erhalten, werden Distanzmetriken zur Bewertung der Ähnlichkeit benutzt, wobei die verwendeten Distanzmetriken dem menschlichen visuellen Ähnlichkeitsempfinden entsprechen und möglichst gleiche Ergebnisse liefern sollten. In der Realität kann die menschliche Wahrnehmung nur schwer durch einen solchen Zusammenhang beschrieben werden, weshalb dieses Themengebiet immer noch Forschungsgegenstand ist und immer weitere Optimierungen erfolgen [17]. Abgesehen von dieser Betrachtung ist es weiterhin unbedingt nötig, dass die verwendeten Distanzfunktionen bestimmte Eigenschaften erfüllen. Die benutzte Metrik sollte keine negativen Werte produzieren können, die Identität des Eingangswertes beibehalten, symmetrisch sein und die Dreickecksungleichung erfüllen [20].

Die Distanzfunktion $d(x, y)$ berechnet den Abstand zwischen den Dokumenten x und y. Für sie sollte gelten:

Nicht negativ: $d(x, y) \geq 0$

Identität bewahren: $d(x, x) = 0$

Symmetrie: $d(x, y) = d(y, x)$

Dreiecksungleichung: $d(x, z) \leq d(x, y) + d(y, z)$

Neben diesen wichtigen Voraussetzungen sollte darauf geachtet werden, dass die Dokumente in einer normalisierten Form vorliegen, um die Ergebnisse der Distanzberechnung nicht zu verfälschen. Die Metrik zur Berechnung kann nun auch auf die schon im vorherigen Kapitel genannten Gewichtungsschemen zurückgreifen, um eventuell wichtigere Eigenschaften eines Bildes stärker einzubeziehen als unwichtige Features [17].

Die zu verwendenden Abstandsmetriken hängen von den benutzten Features und ihrer Darstellung sowie der Beschreibung im Merkmalsraum ab. Sollte eine Distanzmetrik den Anforderungen einer Implementation nicht genügen, gilt auch hier, dass ein geeignetes Abstandsmaß ebenfalls neu konstruiert werden kann, solange die genannten Voraussetzungen erfüllt werden. Für viele Einsatzfälle reichen jedoch simple Distanzmaße für Vektoren aus, da in den meisten Fällen Merkmalsvektoren verwendet werden. Für Vektoren gibt es aufgrund der verbreiteten Einsatzgebiete schon viele etablierte Entfernungsmaße, von denen hier einige einfache Vertreter genannt werden sollen, um die Funktionsweise zu verdeutlichen [20].

Manhattan Distanz

Die Manhattan Distanz zweier Vektoren a und b berechnet sich aus der Summe der absoluten Differenzen ihrer Einzelelemente a_i und b_i.

$$d(a,b) = \sum_{i=1}^{n} |a_i - b_i|$$

Euklidische Distanz

Die Euklidische Distanz zwischen zwei Vektoren wird berechnet, indem zuerst die Quadrate der Differenzen aller Einzelelemente gebildet werden, diese zusammenaddiert werden und schließlich aus dem Ergebnis die Quadratwurzel gezogen wird.

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + ... + (a_n - b_n)^2} = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Kosinus Koeffizient

Der Kosinus Koeffizient von zwei Vektoren berechnet ihre Ähnlichkeit in Bezug auf den Winkel zwischen ihnen. Der Wertebereich des Ergebnisses beträgt hierbei $[-1;1]$. In diesem Zusammenhang bedeutet 1 Gleichheit der Vektoren oder besser gesagt ihrer Richtungen, 0 Unabhängigkeit zwischen ihnen und -1 Ungleicheit beziehungsweise Entgegengesetztheit.

$$d(a,b) = \frac{a * b}{|a| * |b|} = \frac{\sum_{i=1}^{n} a_i * b_i}{\sqrt{\sum_{i=1}^{n} {a_i}^2} * \sqrt{\sum_{i=1}^{n} {b_i}^2}}$$

Die dargestellten Beispiele für Distanzmetriken richten sich an Anwendungsfälle mit Merkmalsvektoren. Aber auch für andere Beschreibungsformen wie Histogramme oder Hashergebnisse gibt es entsprechende Abstandsmaße, die für eine Ähnlichkeitsbestimmungen verwendet werden können. Eine tiefergehende Ausführung würde den Rahmen dieser Arbeit überschreiten, weshalb ich hier auf die entsprechende Literatur, zum Beispiel [20] oder [23] verweisen möchte.

Im Falle einer Suchanfrage an ein CBVIR System wird das eingegebene Suchdokument zur Suche mit allen schon analysierten Bildern im Index verglichen. Im Index ist eine Repräsentation aller Dokumente in Form der jeweiligen Merkmale und damit eine Beschreibung ihrer Lage im definierten Merkmalsraum abgelegt. Zum Suchzeitpunkt wird das Beispielbild, mit dessen Hilfe gesucht wird, auf die gleiche Weise analysiert wie alle Bilder im Index auch. Als Ergebnis liegt eine Repräsentation des Bildes der Anfrage im Merkmalsraum vor. Auf deren Grundlage können nun alle Dokumente des Systems mit der Suchanfrage verglichen werden, indem eine Distanz- und damit Ähnlichkeitsbemessung mit Hilfe der oben benannten Distanzmetriken

durchgeführt wird. Als Resultat dieses Vergleiches kann eine Ergebnisliste sortiert nach der visuellen Ähnlichkeit oder auch der Relevanz erstellt werden.

2.4 Probleme und Designherausforderungen

In dem vorherigen Kapitel wurde beschrieben wie ein IR oder noch spezieller ein CBVIR System aufgebaut ist und die Funktionsweise einiger Elemente genauer erläutert. Zum einen sollte dadurch ein gewisses Verständnis für die Vorgänge in einem solchen System vermittelt, zum anderen aber auch die Grundlage geschaffen werden, um sich mit den Herausforderungen und Problemen für Anwendungen dieser Art beschäftigen zu können. In dem vorherigen Abschnitt dieser Arbeit wurde oft erwähnt, dass bei vielen Fragestellung eine Abwägung nötig ist und eine Lösung auf das jeweilige Problem angepasst werden muss. Um die Anwendbarkeit und Grenzen von CBVIR Systemen besser zu verstehen, werden folgend die schwerwiegendsten Herausforderungen aufgezählt und genauer erläutert.

Als erste Einschränkung für CBVIR Systeme ist zunächst einmal die Breite des Anwendungsbereiches, auch *Domain* genannt, zu benennen. Der Anwendungsbereich beschreibt in diesem Zusammenhang den Umfang, den ein CBVIR System umfassen soll. Genauer werden in dieser Betrachtung der Größenmaßstab[3], die Varianz der Bilder, mögliche semantische Inhalte, die Art der Produktion und viele weitere Eigenschaften bewertet [20].

Breiter Anwendungsbereich (Broad Domain):

Im diesem Fall handelt sich sich meist um Anwendungen, die auf einer **sehr großen Bildersammlung** arbeiten, deren Bilder zusätzlich eine **große Varianz** an visuellen Inhalten aufweisen. Ebenfalls zeichnen sie sich durch eine **Vielzahl verschiedener semantischer Inhalte** aus und sind zudem meist unter **sehr unterschiedlichen Produktionsbedingungen** entstanden[4]. Eine so große Anzahl an Variablen macht es sehr schwer, ein CBVIR System auf eine Problemstellung auf diesen Grundlagen einzustellen beziehungsweise zu optimieren, da zudem meist auch kein Expertenwissen oder weitere Metadaten zur Einordnung zur Verfügung stehen. Ein Beispiel eines solchen Anwendungsumfanges stellen die Bildersuchen von Internetsuchmaschinen dar, welche allerdings durchaus Metadaten und Annotationen verwenden, um die Leistung ihres Systems zu verbessern [11].

[3] Dieser ist meist mit der Menge der im Index enthaltenen Bilder gleichzusetzen.

[4] Dies bedeutet, dass Bilder zum Beispiel verschiedenartige Hintergründe besitzen oder wechselnde Lichtbedingungen bei der Produktion benutzt wurden.

Schmaler Anwendungsbereich (Narrow Domain):

Bei Anwendungen dieser Art handelt es sich meist um Systeme, die auf **kleine bis mittelgroße Bildersammlungen** ausgelegt sind und deren **visuelle Varianz sehr klein** ist. Die dargestellten Inhalte haben zudem meist **einheitliche semantische Bedeutungen** und wurden in einer **kontrollierten Umgebung** aufgenommen. Außerdem stehen bei der Entwicklung meist noch Expertenwissen oder weitere Metadaten zur Verfügung, um das System einzustellen und zu optimieren. Diese Konfiguration ist aufgrund der geringen Variabilität einfacher vorzunehmen als bei einem breiteren Anwendungsumfang. Ein Beispiel für ein System dieser Art ist eine medizinische Anwendung zur Erkennung von Tumoren auf schwarz-weiß Bildern von Untersuchungen durch beispielsweise Computertomographie [20].

Die beiden dargestellten Szenarien sind natürlich nur Extrema eines Spektrums vieler verschiedener Anwendungsfälle, allerdings zeigen sie auch, wie viele unterschiedliche Faktoren bei dem Design einer CBVIR Lösung beachtet werden müssen. Durch die Vielfalt eventueller Problemstellung lässt sich deswegen hier nur schwer eine allgemeine Anleitung zu deren Lösung präsentieren, weshalb ich damit mehr eine Auflistung relevanter Einflüsse liefern und keine Lösungsvorschläge anbringen will. Diese sollten besser durch Abwägung und Nachforschung in entsprechender Literatur erarbeitet werden.

Eine weitere große Problemstellung für alle IR Systeme stellt die Variabilität, die der Nutzer selbst in das System einbringt, dar. Der Benutzer einer Anwendung geht mit bestimmten Intentionen und Bedürfnissen an die Verwendung eines solchen Systems heran. Je nach Ausprägung dieser Eigenschaften erwartet ein Nutzer unterschiedliche Ergebnisse, da für ihn andere Relevanzkriterien gelten als für andere Nutzer [4]. Sowohl die Intention als auch die Bedürfnisse sind individuell und durch die Erfahrungen, die derzeitigen Emotionen und die Erwartungen des Benutzers an das System, sowie von bisherigen Nutzungserlebnissen und vielen anderen Einflüssen abhängig. Um dem einzelnen Nutzer trotz der Varianz an möglichen Ausprägungen der Einflüsse eine zielführende, effektive und effiziente Suche zu ermöglichen, ist es von hoher Bedeutung, dies in den Designprozess von CBVIR Systemen und dessen Abwägungen mit einzubeziehen. Dies kann beispielsweise durch vorherige Benutzerforschung sowie Untersuchung der Benutzungssituation und genaue Aufgabenanalyse durchgeführt werden, um das System möglichst genau auf die Anforderungen einstellen zu können. Eine weitere Möglichkeit besteht darin, den Nutzer selbst vor der Suche weitere Angaben bezüglich seiner Bedürfnisse machen zu lassen oder ihm die Möglichkeit zu geben, auf Suchergebnisse zur reagieren und diese schrittweise zu verfeinern. Ein solches Vorgehen wird auch als *Relevance Feedback* bezeichnet. Der Nachteil dieser Herangehensweise ist, dass der Nutzer schon gewisse Erfahrungen im Umgang mit dem System benötigt, da die Konfigurationsmöglichkeiten ihn auch verwirren könnten. Das Gebiet der Nutzer- und Bedürfnisanalyse ist nur ein Teil der Erstellung eines CBVIR Systems,

bildet jedoch ein eigenes Forschungsgebiet in dem immer weitere Fortschritte erzielt werden. Für tiefere Einblicke empfehle ich einen Blick in die entsprechende weiterführende Literatur, zum Beispiel [19].

Auch die Definition der Ähnlichkeit von Bildern an sich kann schon zu Problemen führen. In den meisten Anwendungsfällen für VIR Systeme soll der einfachere Fall der visuellen Ähnlichkeit umgesetzt werden. Diese basiert auf den Pixelwerten eines Bildes und kann meist problemlos automatisch festgestellt werden. Der Nutzer eines Systems bringt jedoch auch eine semantische Interpretation eines Bildes in den Vergleichsprozess ein. Daraufhin wird nicht der rein visuelle Inhalt, sondern auch dessen semantische Bedeutung und Zusammenhänge im Bild beurteilt. Ein solcher semantischer Ähnlichkeitsvergleich ist allerdings um ein Vielfaches komplexer umzusetzen. Das wirkliche Problem ist allerdings, dass sich die beiden Komponenten in einem Anwendungsfall nie wirklich sauber trennen lassen. Das System an sich könnte zwar beispielsweise nur auf visueller Ähnlichkeit basieren, allerdings würde der Benutzer unweigerlich eine semantische Komponente, durch seine Erwartungshaltung bezüglich der Suchergebnisse, einbringen [20]. Eine ähnliche Problemstellung wird auch durch die, immer wieder in der Literatur erwähnte, *Semantic Gap* betrachtet. Genauer gesagt beschreibt diese die Ungleichheit zwischen den visuellen Informationen, die aus einem Bild extrahiert werden können, und deren Interpretation oder Bedeutung für einen Nutzer. Durch technische Neuerungen und Erkenntnisse, beispielsweise maschinelles Lernen, wird allerdings immer häufiger auch eine semantische Komponente in die Betrachtung einbezogen, wodurch die Semantic Gap immer weiter schrumpft. Diese zusätzliche Komponente bietet viele Möglichkeiten für bessere Ergebnisse, erfordert allerdings auch immer weitere Forschung in den entsprechenden Themengebieten [1].

Neben den genannten Problemstellungen existieren noch viele weitere kleinere als auch größere Herausforderungen für CBVIR Systeme. Die hier erläuterten Probleme betreffen allerdings fast alle Anwendungen, während viele andere nur für spezielle Einsatzformen relevant sind. Bei Interesse möchte ich hier auf die Auflistung verschiedener weiterer *Gaps* in [7] verweisen, welche diese Herausforderungen in abstrakter Weise beschreiben.

3 Praktische Implementation eines CBVIR Systems

Wie das Thema der Arbeit schon darstellt, soll eine Implementierung einer bildbasierten Produktsuche im Shopsystem SAP Hybris Commerce umgesetzt werden. Eine solche Realisierung ist der Definition nach ein CBVIR System, welches ein Beispielbild als Suchanfrage verwendet, es mit den hinterlegten Produktbildern vergleicht und die relevantesten Ergebnisse zurückliefert, um dem Kunden einen Kauf des jeweiligen Produktes zu ermöglichen. Logischerweise treffen somit auch alle bisher erläuterten Konzepte, Probleme und Lösungen auf diese Implementierung zu. In diesem Kapitel soll erklärt werden, wie das System aufgebaut ist, wie entscheidende Designfragen beantwortet wurden, wie das System schlussendlich genau funktioniert, welche Probleme es bei der Umsetzung gab und wie sich diese teilweise lösen ließen. Als Inspiration für die Implementierung diente folgender Blogeintrag [26], der sich mit den Grundlagen einer Umsetzung in SAP Hybris Commerce beschäftige. In diesem Zusammenhang konnte die Thematik allerdings nur oberflächlich angekratzt werden, weshalb ich mit dieser Arbeit eine tiefergehende Auseinandersetzung mit dem Thema und eine wissenschaftliche Auswertung der Ergebnisse erreichen will. Eine äußerst wichtige Ergänzung ist, dass die umgesetzte Realisierung **lediglich visuelle Ähnlichkeiten** vergleichen sollte und semantische Interpretationen somit in der Ergebniszusammenstellung keine Rolle spielen. Aus diesem Grund ergibt es auch nur Sinn, Produkte in diese Suche einzubeziehen, welche vor allem durch ihre visuellen Eigenschaften für Kunden interessant sind. In dem Falle dieser Realisierung werden deshalb Bilder von Kleidungstücken als Testdaten verwendet, da diese die genannten Voraussetzungen sehr gut erfüllen.

3.1 Systemkomponenten und Voraussetzungen

In diesem Abschnitt der Arbeit sollen die Komponenten des technischen Umfeldes der umgesetzten CBVIR Lösung erläutert werden. Diese waren teilweise durch Vorgaben vorherbestimmt, aber auch das Resultat mehrerer Abwägungen bezüglich der besten Umsetzung. Die Abbildung 3.1 zeigt die hier erklärten Bestandteile noch einmal in einer kompakten Form.

Die darin dargestellte Komponente der Hybris Commerce Erweiterung mit dem Namen *imageSearch* dient zur Verbindung der folgend genannten Bestandteile und regelt dabei ebenfalls deren Konfiguration und die gesamte Funktionsweise des Systems. Sie ist der Hauptbestandteil dieser praktischen Implementation einer CBVIR Lösung und wird deshalb noch genauer im Abschnitt der Systemvorstellung diskutiert. Da sie unter starkem Einfluss der anderen Systemkomponenten steht, wird die Erweiterung bereits hier erwähnt und in der Abbildung dargestellt.

Als erste Systemkomponente stand, wie schon am Thema zu erahnen, das Shopsystem *SAP Hybris Commerce* (1) fest. Das System ist eine E-Commerce Plattform, über die sowohl im B2C (Business to Customer) als auch im B2B (Business to Business) Bereich der Onlinemarkt erschlossen werden kann. Neben einigen anderen bekannten Systemen wie Intershop oder auch Magento hat sich SAP Hybris Commerce immer weiter verbreitet und ist ein etablierter Bestandteil der E-Commerce Landschaft geworden. Dass die Implementierung eines CBVIR Systems auf Grundlage dieser Plattform vorgenommen wird, hat weniger technische Gründe, als die einfache Tatsache, dass sich das Entwicklerteam der dotSource, dem ich angehöre, auf Lösungen für SAP Hybris Commerce spezialisiert hat und ich deshalb schon Erfahrung im Umgang sammeln konnte. Außerdem wird dadurch auch der größte Mehrwert für das Entwicklerteam geleistet, da Erkenntnisse oder Umsetzungen eventuell in andere Projekte übernommen werden könnten. Die Auswahl dieser Plattform als Grundlage für die Umsetzung einer bildbasierten Produktsuche in einem Shopsystem zieht logischerweise allerdings andere Betrachtungen bezüglich der Auswahl von Systemkomponenten nach sich. Rein technisch ist SAP Hybris Commerce ein recht komplexes System, das durch verschiedene Schichten mit unterschiedlichen Aufgaben strukturiert ist. Es kann durch das Hinzufügen eigener Erweiterung, auch *Extensions* genannt, in einer systemkonformen Art und Weise erweitert werden, was bei der Realisierung dieses Projektes auch so geschehen ist. Weiterhin ist es wichtig zu erwähnen, dass SAP Hybris Commerce auf der Programmiersprache *Java* basiert, um eine Auswahl weiterer Komponenten zu treffen. Als Basis dieses Projektes wurde ein sogenannter *Accelerator* von SAP Hybris Commerce verwendet, was eine Art Demoinstanz mit allen nötigen Grundeinstellungen darstellt und als Startpunkt der Entwicklung verwendet werden kann. Auf die Details der Implementierung werde ich im Abschnitt der Systemvorstellung noch einmal genauer eingehen.

Eine Suchfunktion, welche meist auf textbasierten Suchanfragen beruht, ist in den meisten Shopsystemen standardmäßig implementiert. So auch in SAP Hybris Commerce, was in seinem Accelerator auf die Verwendung von Apache Solr (2) setzt. Diese textbasierte Suchmaschine wird sowohl für die Produktsuche durch den Kunden als auch für systemadministrative interne Suchvorgänge verwendet. SAP Hybris Commerce startet, falls gewollt, beim Hochfahren der Plattform die benötigten Solr-Instanzen als eigene Server. Eine Instanz kann mehrere Indexe

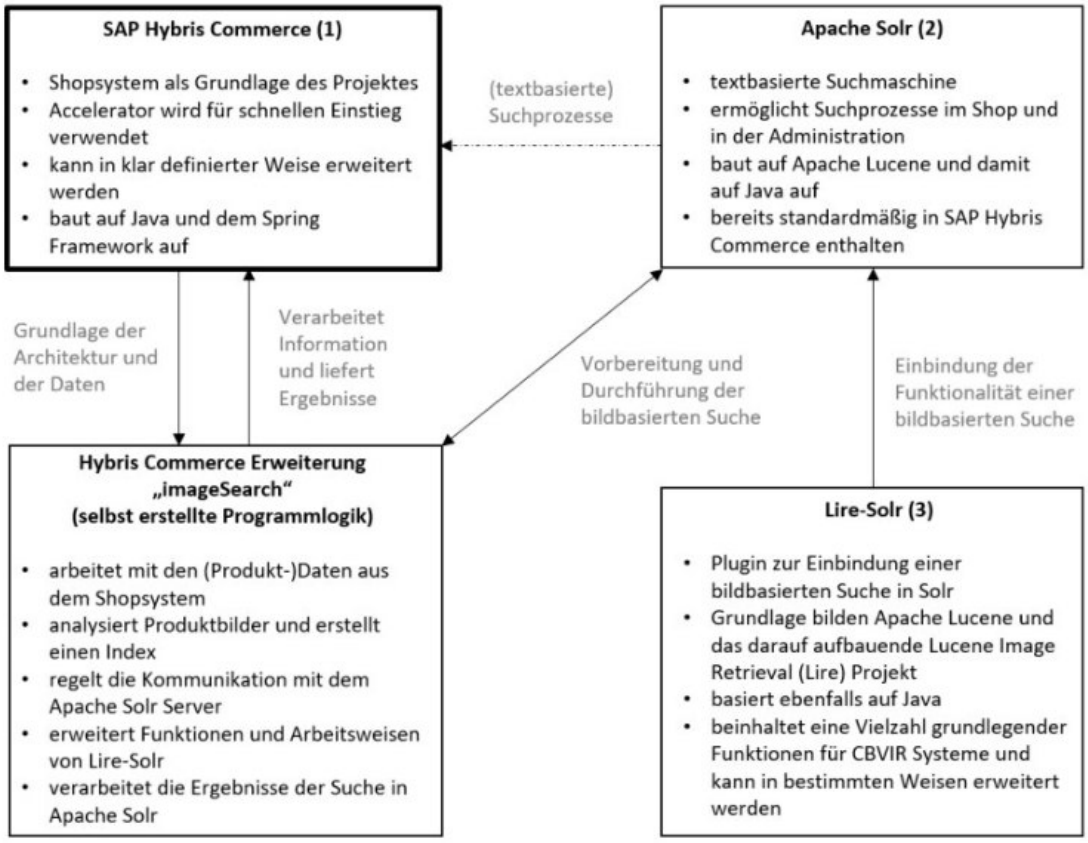

Abbildung 3.1: Komponenten um das CBVIR System

Diese Abbildung stellt die Komponenten dar, welche die Erstellung und Arbeitsweise des umgesetzten CBVIR Systems in direkter Weise beeinflussen. Diese technischen Umgebungsvoraussetzungen und -entscheidungen haben die realisierte Implementation entscheidend geprägt.

zur Verwendung durch verschiedene Suchfunktionen enthalten. Der Solr-Server und somit auch ein jeder Index kann über einfache protokollbasierte Anfragen an den Server angesprochen werden. Verwendung finden hierfür beispielsweise das HTTP oder auch das HTTPS Protokoll, die es ermöglichen, Suchanfragen zu stellen, neue Einträge in einen Index einzufügen und vieles weitere mehr. Um die Anbindung einfacher zu gestalten, verwendet SAP Hybris Commerce intern eine Javabibliothek namens *SolrJ*, um die Kommunikation zu den Solr-Instanzen aufzubauen. SolrJ verwendet ebenfalls die eben genannten Protokolle, bietet allerdings einfachere Konfigurationsmöglichkeiten. Die Entscheidung Apache Solr als Suchmaschine für die CBVIR Realisierung zu verwenden, basiert darauf, dass es bereits sehr gut in SAP Hybris Commerce eingebunden ist. Ein Problem ist jedoch, dass Solr eine textbasierte Suchmaschine ist und nicht wirklich mit Bildern umgehen kann. Jedoch erlauben es Plugins, eigene Logik in Apache Solr einzubringen und damit die Verwendungsmöglichkeiten zu erweitern. Da Solr ebenfalls in Java geschrieben ist, kann auch hier eine einfache Anbindung an Hybris Commerce erfolgen.

Die letzte und sehr wichtige Komponente bei der Realisierung der CBVIR Anwendung ist ein solches Plugin für Apache Solr. Solr basiert auf *Apache Lucene*, was eine Open-Source-

Bibliothek zur Erstellung von Information Retrieval Lösungen ist und ebenfalls auf Java basiert. Auch hier ist die Anwendung nur auf textbasierte Systeme ausgelegt. Um auch bildbasierte Suchanwendungen zu ermöglichen, existiert seit 2006 das Open-Source-Projekt *Lucene Image Retrieval*, oder kurz *LIRE*, welches die Grundlagen von Apache Lucene nutzt, um die Erstellung von Information Retrieval Systemen für Bilder- und Videodokumente zu ermöglichen. LIRE bietet dabei sowohl ein Grundgerüst zur Erstellung von VIR Systemen an, kann jedoch auf Grund vieler vorimplementierter Funktionen auch als vollumfängliche VIR Lösung genutzt werden. Der Vorteil für die Implementation in SAP Hybris Commerce ist, dass ein Plugin für Apache Solr existiert, welches einen Großteil der Funktionalitäten von LIRE in die Suchmaschine einbringt. Das verwendete Plugin trägt den Namen *lire-solr* (3) und ist, wie alle schon genannten Systeme, ebenfalls ein Open-Source-Projekt und basiert auf Java. Der Vorteil der Tatsache, dass alle hier genannten Technologien, bis auf SAP Hybris Commerce, Open-Source-Projekte sind, besteht in der Offenheit des gesamten Quellcodes, immer fortschreitender Updates und der einfachen Erweiterbarkeit der Anwendung. lire-solr bildet bei der Umsetzung dieser CBVIR Lösung in einem Shopsystem das Herzstück der Anwendung, da es die Funktionalität zur Auswertung von Bildern durch Merkmale, deren Einordnung in einen Merkmalsraum und nicht zuletzt das Vorgehen zur Ähnlichkeitsbestimmung enthält. Bei der Auswahl der Systemkomponenten spielte außerdem eine Rolle, dass sich durch lire-solr in einfacher Art und Weise eigene Features, Distanzmetriken und viele weitere Ergänzungen in das System einbringen lassen. In der praktischen Umsetzung kam dies allerdings leider nicht zum Tragen, da das Hinzufügen eigener Logik einen hohen Arbeitsaufwand bedeutet und den Umfang dieser Arbeit überstiegen hätte.

Kurz noch einmal zusammengefasst, beeinflussten drei wichtige Komponenten die Umsetzung des Systems. SAP Hybris Commerce bildet die Grundlage für die Implementation und ist die Plattform, auf der das Shopsystem läuft. Apache Solr ist eine in SAP Hybris Commerce integrierte textbasierte Suchmaschine, welche in dieser Anwendung hauptsächlich für die Erstellungen eines Indexes und dessen Verwaltung zuständig ist. Das Plugin lire-solr fügt die benötigten VIR Funktionalitäten zu Apache Solr hinzu und ist vor allem für die Bildverarbeitung, -einordnung und -auswertung verantwortlich.

Natürlich standen zur Implementation auch andere Softwarelösung im Bereich von VIR Anwendungen zur Auswahl, beispielsweise *imgSeek [15], FIRE [12] oder Windsurf [34]*. Neben den oben schon erläuterten Vorteilen der Umsetzung mittels Solr und lire-solr, haben vor allem Lizenzfragen, die Verwendbarkeit von Metadaten, die Erweiterbarkeit des Systems und die technologische Ähnlichkeit zur Auswahl der beschriebenen Komponenten geführt. Im Falle eines anderen Shopsystems als Grundlage für die Anwendung wäre die Auswahl deshalb eventuell anders ausgefallen.

3.2 Designfragen

Bevor im nächsten Abschnitt der Aufbau und die Funktionsweise der Implementierung des CBVIR Systems aufgezeigt werden soll, will ich in diesem Segment der Arbeit auf wichtige Designentscheidungen eingehen. Dafür werde ich die in [23] gestellten Designfragen für die umgesetzte Realisierung beantworten. Ich hoffe damit einen guten Einblick in wichtige Entscheidungen geben zu können.

Welche Merkmale sollen verwendet werden und wie werden diese dargestellt?

Lire bietet eine Vielzahl an lokalen und globalen Features bereits vorimplementiert zum Einsatz in einer Anwendung an. In lire-solr sind leider nicht all diese Features eingebaut und vor allem die Verwendung lokaler Merkmale würde einen hohen Aufwand für deren Implementation und Benutzung bedeuten. Außerdem ist die Verwendung lokaler Features vor allem im Zusammenhang mit einer Objekterkennung sinnvoll. Da es sich bei der Realisierung allerdings eher um einen Prototypen handeln sollte, war eine solche Objekterkennung ein optionaler und kein integraler Bestandteil der Umsetzung. Aus diesem Grund entschied ich mich für die Benutzung der schon implementierten, einfacher zu verwendenden und simpler auszuwertenden globalen Features. In lire-solr wird zur Beschreibung eines Bildes meist nur ein Feature verwendet. Dies kann mittels eines Rerankings der Ergebnisse durch ein anderes Merkmal oder die Konstruktion eines eigenen komplexeren Merkmales umgangen werden. In der hier umgesetzten Implementation wurde dies allerdings nicht gemacht, um einen einfachen nachvollziehbaren Aufbau, eine einfache Erweiterbarkeit und eine gute und vergleichbare Auswertung zu ermöglichen. Die schlussendlich verwendeten Merkmale sind das MPEG-7 Color Layout (CL), das Pyramid Histogram of oriented Gradients (PHOG), das MPEG-7 Edge Histogram (EH), das Auto Color Correlogram (AC) und das Fuzzy Color and Texture Histogram (FCTH). Mit dieser Auswahl hoffe ich, die Breite der angesprochen Merkmale und der verwendeten Bildeigenschaften gut abgebildet zu haben. Leider sind in dieser Zusammenstellung keine rein formbasierten Merkmale vertreten, da diese in den meisten Fällen in die Kategorie der lokalen Features fallen. Allerdings sind Farb- und Texturmerkmale in verschiedenen Formen und sogar deren Kombination, auch mit Formeigenschaften, in Merkmalen enthalten. Bezüglich der Auswertung sind, aufgrund des begrenzten Umfangs der Arbeit, allerdings nur zwei Merkmale relevant, welche deswegen genauer vorgestellt werden sollen.

Das MPEG-7 Color Layout ist ein farbbasiertes Feature, welches allerdings auch eine ortsbezogenene Komponente in die Bildbeschreibung einbringt. Zunächst sollte erwähnt werden, dass das Bild als Beschreibung im YCbCr Farbraum vorliegt und die folgenden Operationen auf jede einzelne Komponente angewandt werden. Das Bild, beziehungsweise die jeweilige Komponente, wird zunächst gleichmäßig in 8x8, also insgesamt 64, Blöcke eingeteilt, für welche jeweils die dominante Farbe berechnet wird (siehe Abbildung 3.2). Diese Beschreibung des Bildes wird im nächsten Schritt mit Hilfe der Diskreten Cosinus Transformation (DCT) in kompakter Form dargestellt und nachfolgend durch den Vorgang des Zig-Zag-Scans quantisiert. Als Ergebnis liegt eine kompakte Bildbeschreibung in Form von quantisierten DCT-Koeffizienten vor [16].

Abbildung 3.2: MPEG-7 Color Layout Aufteilung und Farbanalyse

Diese Abbildung stellt die Aufteilung des Bildes in 64 Blöcke und die anschließend durchgeführte Berechnung der dominanten Farbe für jeden Block dar [27].

Das Pyramid Histogram of oriented Gradients ist ein teilweise formbasiertes Merkmal, welches allerdings auch einige Einflüsse von Texturen in sich trägt. Dazu wird zuerst eine Kantenerkennung, beispielsweise durch den Canny Algorithmus, auf das Bild durchgeführt. Auf das Resultat wird darauf folgend ein Filter zur Erkennung der Kantenorientierung angewandt, beispielsweise ein 3x3 Sobel-Filter. Die Ergebnisse werden danach in ein Histogramm eingetragen, welches verschiedene Klassen von Kantenorientierungen darstellt. Das Bild wird danach in 2x2 Zellen eingeteilt, für welche diese Vorgehen wiederholt wird. Je nach Tiefe des PHOG kann dieser Vorgang noch auf weitere Unterteilungen der Zellen angewandt werden. Nach Beendigung der Histogrammerstellungen werden die Histogramme aller Zellen mit ihren jeweiligen Eltern-Zellen kombiniert, was ebenfalls je nach Tiefe mehrmals durchgeführt werden muss (siehe Abbildung 3.3). Schlussendlich steht eine Bildbeschreibung in Form eines zusammengefassten Histogrammes zur Verfügung, welches aus Kantenorientierungen und deren ortsbezogenen Daten besteht [2].

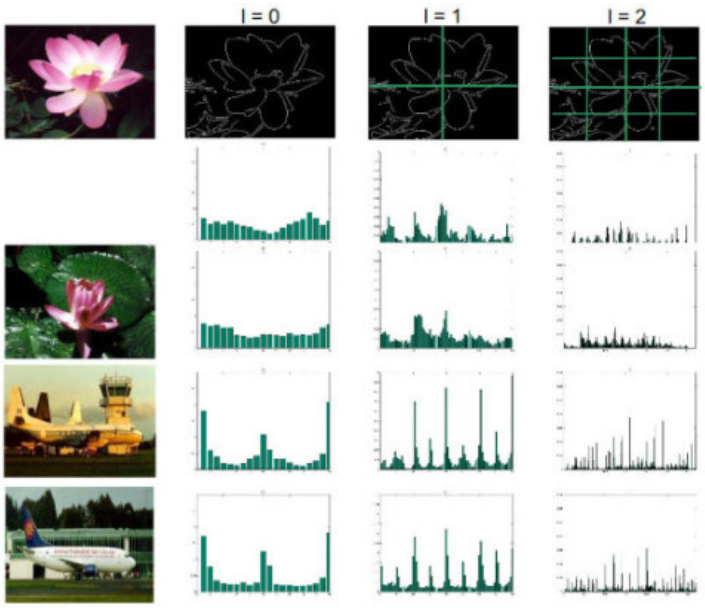

Abbildung 3.3: PHOG Aufteilung und Histogrammberechnung

Diese Abbildung stellt die schon erwähnte Aufteilung eines Bildes in mehrere Zellen und Unterzellen sowie die Berechnung und das Zusammenstellen der jeweiligen Kantenhistogramme dar. Mit Hilfe der Unterteilung in Zellen wird gewährleistet, dass auch Informationen bezügliches des Ortes beziehungsweise der Lage des Pixels in die Bewertung einfließen. Je feiner die umgesetzte Einteilung, desto größer ist die örtliche Auflösung und desto genauer sind auch die gewonnenen Informationen [2].

Die genannten Features werden in der Folge als Merkmalsvektor gespeichert, welcher im Index in einem *Base64* encodierten String hinterlegt wird. Dieses textbasierte Format erlaubt die Indexierung von den gewonnen Bilddaten in der Suchmaschine Solr.

Welches Ähnlichkeitsmaß soll verwendet werden?

Die verwendeten Ähnlichkeitsmaße hängen stark von den implementierten Merkmalen ab, weshalb die Definition dieser Metriken ein integraler Bestandteil der Umsetzung eines Merkmales in lire-solr ist. Die schon vorher implementierten und hier verwendeten Features greifen auf die Manhattan Distanz zurück. Die Realisierung hängt dabei stark von den Komponenten des jeweiligen Features ab, weshalb die Umsetzung von Merkmal zu Merkmal immer etwas unterschiedlich ausfällt. Die Wahl der Entwickler von lire-solr fiel vor allem auf Grund der Einfachheit auf dieses Maß, von dem, wie bei vielen anderen auch, davon ausgegangen wird, dass es gut genug mit der menschlichen Wahrnehmung übereinstimmt.

Welche Formen von Suchanfragen sollen verwendet werden?

Ein Ziel dieser Arbeit war die Umsetzung einer bildbasierten Produktsuche. Ein solches System für Suchanfragen wird als *Query by example* bezeichnet, was bedeutet, dass die Suchanfrage aus einem Beispieldokument desselben Formats besteht, wie auch die Dokumente im Index. In der speziellen Umsetzung für Onlineshops hielten mein Betreuer

und ich zwei Arten, das Bild an das System zu übergeben, für sinnvoll. Ein Nutzer sollte, durch die Interaktion mit einem Interface, in der Lage sein, ein Bild von seinem lokalen Gerät hochladen und für die Suche verwenden zu können. Ebenfalls unterstützt wird das Bereitstellen mit Hilfe einer URL, welche auf ein Bild verweist.

Wie wird dem System mitgeteilt welche Merkmale verwendet werden sollen?

Da es sich bei dem System mehr um eine Art Prototyp als um eine vollumfänglich einsatzbereite Realisierung handelt, trifft der Nutzer die Entscheidung, auf Grundlage welchen Merkmals die Suche ausgeführt wird, selbst. Ihm stehen dabei alle schon benannten Merkmale zur Verfügung. Um die Verwendung einfacher zu gestalten, wurden die technischen Merkmalsbezeichnungen durch eine kurze Beschreibung der verwendeten und relevanten Bildeigenschaften ergänzt. Trotz dieser Maßnahme setzt dieses Auswahlverfahren ein grundlegendes Verständnis für die Funktionsweise von Plattformen dieser Art voraus.

Wo werden die Bilddateien gespeichert?

Die umgesetzte Anwendung basiert auf der Onlineshop-Plattform SAP Hybris Commerce. Die zu indexierenden Bilder sind als Produktbilder in dem System gespeichert, müssen allerdings vorher durch manuellen Import oder mit Hilfe sogenannter *ImpEx Dateien* hinzugefügt werden. Welche Bilder später tatsächlich indexiert werden, hängt davon ab, ob das Produkt überhaupt in der jeweiligen Shopinstanz vertreten und auffindbar sein soll.

Wie kann der Benutzer dem System Feedback geben und wie wird dieses genutzt?

Bei dem Verfahren des Relevanz-Feedbacks wird dem Nutzer nach einem Suchdurchlauf die Möglichkeit gegeben, eine Rückmeldung an das System zu senden, um bestimmte Relevanzkriterien und Suchparameter anzupassen. Ein solches Feedbacksystem zählt zu den eher fortgeschritteneren Verfahren in einer CBVIR Anwendung, weshalb es in der Realisierung nicht umgesetzt wurde. Allerdings würde die Einbindung eine Verbesserung der Suchergebnisse und des Nutzungserlebnisses liefern und sollte deshalb in die Betrachtung bezüglich eventueller Verbesserungen und Erweiterungen aufgenommen werden.

Die hier beantworteten Fragen sollten einen grundlegenden Eindruck bezüglich der Designentscheidungen und deren Gründe betreffend dieser Umsetzung vermittelt haben. In dem folgenden Abschnitt soll der genaue Systemaufbau und Prozessablauf vorgestellt werden, um ein tiefes Verständnis für die Umsetzung zu vermitteln.

3.3 Vorstellung des Systems

Wie schon im Abschnitt über die Systemkomponenten erwähnt, besteht die hier umgesetzte Implementation eines CBVIR Systems aus dem Shopsystem SAP Hybris Commerce, der darin integrierten Suchmaschine Apache Solr und einem Plugin für diese Suchmaschine, um die Auswertung und Indexierung von Bildern zu erlauben. Um die nötigen Prozesse und deren Programmlogik umzusetzen, wurde eine Erweiterung für SAP Hybris Commerce erstellt, welche passenderweise *ImageSearch* benannt wurde. Sie enthält sowohl den Quellcode und notwendige Konfigurationsdateien als auch weitere benötigte Ressourcen. Die Struktur der Verzeichnisse und die Umsetzung der Prozesse in Programmlogik orientiert sich dabei streng an den für SAP Hybris Commerce gegebenen *Best Practices* [29]. Für ein besseres Verständnis ist zu erwähnen, dass SAP Hybris Commerce auf dem Java-Framework *Spring* basiert, um interne Prozesse zu vereinfachen. Das Spring-Framework dient vor allem der Organisation von Abhängigkeiten zwischen Programmteilen sowie der Umsetzung verschiedener Softwarearchitekturen innerhalb des Systems. Dies wird beispielsweise durch die Verwendung der Model-View-Controller Architektur, kurz *MVC*, deutlich. Diese dient der Aufgabenteilung, indem die Prozesse in die Abschnitte der Datenhaltung (Model), der visuellen Ausgabe (View) und deren Verbindung durch Prozesslogik (Controller) getrennt werden. Neben dieser Aufteilung sollten auch der weitere Systemaufbau und die ablaufenden Prozesse im Schaubild 3.4 verdeutlicht werden. Diese Abbildung spiegelt die Struktur eines IR Systems, wie sie in Abbildung 2.1 gezeigt wird, wider. Im weiteren Verlauf dieses Abschnittes will ich auf die enthaltenen Bestandteile der Implementierung genauer eingehen und deren Aufbau sowie die Funktionsweise erklären.

Die Erläuterung der Funktionsweise beginnt mit den Prozessen des Systems, welche vor der Durchführung eines Suchvorgangs abgeschlossen werden müssen und sich in dem Schaubild in dem blau gekennzeichneten Bereich befinden. Diese Vorgänge werden *offline* ausgeführt, also bevor das System wirklich angewendet werden kann. Die erste Komponente ist die **Dokumentensammlung** (1). Wie schon erwähnt wurde, sollte die realisierte Anwendung auf Basis von Bildern von Kleidungsstücken arbeiten. Der Grund dafür war zum einem die große Bedeutung von visuellen Eigenschaften für diese Produktkategorie, als auch die Verfügbarkeit verschiedener Testdatensätze für einen solchen Anwendungsfall. Nach ausgiebiger Recherche entschied ich mich für die Verwendung des *DeepFashion* Datensatzes [18] für *inShop* Bilder. Bei diesem handelt es sich um 7982 Produkte, welche durch insgesamt 52712 Bilder dargestellt werden. Weitere Betrachtungen bezüglich der verwendeten Bilder werden im Abschnitt zur Auswertung der Anwendung erläutert. Um die Umsetzung so anwendungsbezogen wie möglich zu halten, sollten nur Bilder von Produkten, welche im Shopsystem SAP Hybris Commerce hinterlegt sind, in das CBVIR System einbezogen werden. Aus diesem Grund mussten mithilfe des Bilddatensatzes und den verfügbaren Metadaten zuerst die 7982 Produkte in SAP Hybris

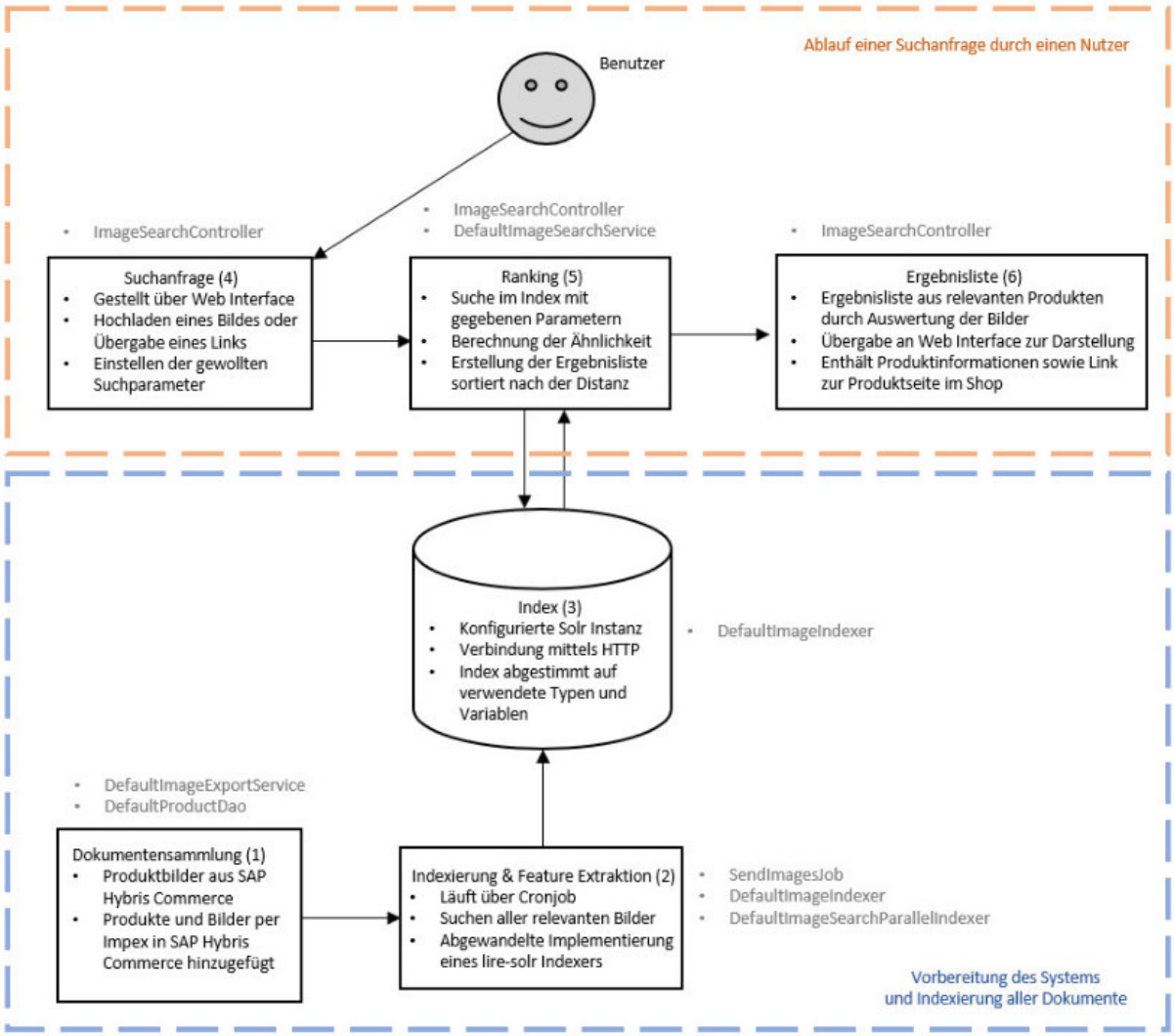

Abbildung 3.4: Aufbau und Funktionsweise der umgesetzten Anwendung

Dieses Schaubild orientiert sich am bereits vorgestellten Aufbau eines IR Systems und wurde auf die realisierte Umsetzung angepasst. Die Komponenten werden durch Aufzählung der wichtigsten Prozesse genauer beschrieben. Die grau gehaltenen Ergänzungen neben den Bestandteilen geben die jeweiligen Java Klassen an, welche eine Rolle in deren Implementation spielen. Nach Vorbild von [20].

Commerce importiert und diesen danach die jeweiligen Produktbilder zugeordnet werden. Dies geschah durch die Verwendung von, für SAP Hybris Commerce üblichen, *Impex*[1] Dateien, die mit Hilfe einiger Helferklassen automatisch aus den verfügbaren Daten erstellt wurden. Nach deren Import in das Shopsystem steht die benötigte Dokumentensammlung zur Verfügung.

Der zweite Bestandteil der umgesetzten Anwendung ist verantwortlich für die **Indexierung und Feature Extraktion** (2). Dabei sollen die relevanten Produktbilder aus dem Shopsystem herausgesucht, diese analysiert und deren Beschreibung in Form von Merkmalen in dem Index gespeichert werden. Da dieser Vorgang je nach Anwendung oder besser gesagt der Häufigkeit

[1] Wortkombination aus Import und Export.

der Updates der Produkte und Bilder mehr als nur einmal ausgeführt werden können muss, wird der Prozess durch einen sogenannten *Cronjob* initiiert. Ein Cronjob ist Quellcode, welcher automatisch von einem Server in bestimmten Zeitabständen ausgeführt wird, was in diesem Fall die automatische Aktualisierung des Index ermöglicht. Die Java-Klasse zur Umsetzung dieses Cronjobs trägt den Namen *SendImagesJob*. Der darin enthaltene Quellcode veranlasst alle notwendigen Schritte zur Indexierung und Merkmalsextraktion. Im ersten Schritt werden alle Produkte herausgesucht, welche für die Anwendung relevant sind. Dies wurde bereits bei dem Import der Produkte und der dabei geschehenen Zuordnung zu einem Produktkatalog[2] vorbereitet, da dieser nun als Relevanzkriterium dient. Die Suche und Rückgabe der Produktdaten unterliegt dabei den Systemgrundlagen von SAP Hybris Commerce. Da diese Plattform, wie schon erwähnt, auf der MVC Architektur aufbaut, werden in dem System Modellobjekte benutzt, um Datenbankeinträge darzustellen. Diese Modellobjekte für die Produkte werden mit Hilfe von weiteren Klassen, wie dem *DefaultProductDao* oder dem *DefaultImageExportService*, den anderen Systemkomponenten zur Verfügung gestellt. Die auf diese Art und Weise bereitgestellten Objekte namens *ProductModel* beinhalten alle wichtigen Informationen zu dem dargestellten Produkt sowie Funktionen für die weitere Benutzung. Die Informationen eines solchen Objektes beinhalten ebenfalls Verweise auf die verwendeten Medienobjekte, wodurch man Zugriff auf die Produktbilder bekommt. Im zweiten Schritt dieses Systembestandteiles werden diese Produktbilder mit Hilfe von *Lire* ausgewertet beziehungsweise die vorher bestimmten Merkmale extrahiert. Dies geschieht mit Hilfe des von Lire bereitgestellten und für diese Anwendung leicht abgewandelten *DefaultImageSearchParallelIndexer*. Die Analyse der Bilder findet dabei mit der Hilfe der *Threading* Technologie statt, wodurch verschiedene Analyseprozesse parallel durchgeführt werden können. Bei der Merkmalsextraktion an sich wird dafür die, in der Definition der Features in deren jeweiligen Klassen eingeführte, Methode zur Berechnung der Merkmale verwendet. Die daraus gewonnenen Daten, sowie einige Informationen über die Bilder an sich, werden als Resultat der Merkmalsextraktion in einer XML-Datei abgelegt, um eine einfache Speicherung der gewonnenen Daten zu ermöglichen. Diese Datei kann in der vorliegenden Form als Anfrage an die vorher eingerichtete Solr Instanz geschickt werden, woraufhin die enthaltenen Daten als Dokumente in den ebenfalls vorher erstellten Index aufgenommen werden. Die Klasse *DefaultImageIndexer* dient dabei zur Einstellung der richtigen Verbindungsparameter zu der jeweiligen Solr-Instanz und bietet weitere Konfigurationsmöglichkeiten bezüglich des *DefaultImageSearchParallelIndexer* und der Anfrage an den Solr-Server.

Die dritte Komponente des Systems ist der **Index** (3), welcher mit der Hilfe von Solr und dem lire-solr Plugin erstellt wird. Um eine möglichst einfache Betrachtung zu ermöglichen und

[2] Ein Produktkatalog dient in SAP Hybris Commerce der Organisation von Produkten und weiteren systeminternen Zusammenhängen.

spezielle Konfigurationen zu erlauben, wurde, neben der bereits existierenden Solr-Instanz für die klassische attributbasierte Produktsuche, noch eine Weitere für die Erstellung des Indexes dieser CBVIR Anwendung angelegt. Bei beiden Instanzen handelt es sich um Solr-Server, die mit der SAP Hybris Commerce Plattform integriert arbeiten und somit bei dem Systemstart der Plattform ebenfalls hochgefahren werden. Der Index wird, im Falle von Solr, auch als *Core* bezeichnet und muss vor der Verwendung auf die benutzten Datentypen und Feldbezeichnungen seiner zukünftigen Einträge konfiguriert werden. Diese Einstellungen sind vor allem in Zusammenhang mit der Installation des lire-solr Plugins immens wichtig. Die vorzunehmenden Schritte werden in [5] und [21] sehr gut erläutert und sollen deshalb, auch im Hinblick auf den Umfang dieser Ausführung, nicht noch einmal erwähnt werden. Auch in SAP Hybris Commerce können bezüglich der integrierten Solr-Instanzen noch verschiedene Parameter angepasst werden. Zu diesen zählt auch, dass die Kommunikation zu dem Solr-Server der CBVIR Anwendung nur mittels HTTP verläuft, da es ein einfaches Verfahren ist und somit weniger Komplikationen auftreten konnten. Zur Kommunikation der Ergebnisse der Merkmalsextraktion wird der *DefaultImageIndexer* verwendet, der dazu auf viele Einstellungsparameter Zugriff haben muss. Aus diesem Grund wird diese Klasse noch einmal im Zusammenhang mit dem Index im Schaubild dargestellt. Die bisher vorgestellten Komponenten aus dem Schaubild hatten vor allem die Aufgabe, die Dokumente aufzubereiten, zu analysieren und daraus einen Index zu konstruieren, also zusammengefasst das System für eine wirkliche Suchanwendung vorzubereiten. Die folgenden Bestandsteile gehören thematisch zu dem orangefarbenen Bereich des Schaubildes, welcher den Ablauf einer Suchanfrage eines Nutzer und die damit in Verbindung stehenden Komponenten beschreibt. Der Benutzer selbst soll in diesem Zusammenhang allerdings nicht näher betrachtet werden, da es sich bei dieser Realisierung eines CBVIR Systems mehr um eine Art Prototyp handelt als um eine realistische Endanwendung und somit eine klare Nutzerdefinition nicht entscheidend ist. Für eine produktive Umsetzung eines solchen Systems sollte dieser Aspekt allerdings genauer betrachtet werden, wie in dem Kapitel über Probleme eines CBVIR Systems dargestellt wurde.

Der nächste Bestandteil des Schaubildes ist die **Suchanfrage** (4), welche von einem Benutzer initiiert wurde. Da es sich bei dieser Umsetzung um eine Integration in ein Onlineshopsystem handelt, wird diese Suchanfrage über ein Web-Interface in einem Browser gestartet. In der, in SAP Hybris Commerce verwendeten, MVC-Architektur stellt dieser Bestandteil des Systems den *View* dar, also eine visuelle Darstellung mit deren Hilfe mit dem System interagiert werden kann und welche die Ergebnisse einer Suchanfrage nach Beendigung anzeigt. Die Webseite ist durch den Zusatz */imageSearch/search* an die IP-Adresse oder den Domainnamen der Instanz des SAP Hybris Commerce Webservers zu erreichen. Das Interface zur Übergabe einer Suchanfrage an das System wird in der Abbildung 3.5 dargestellt.

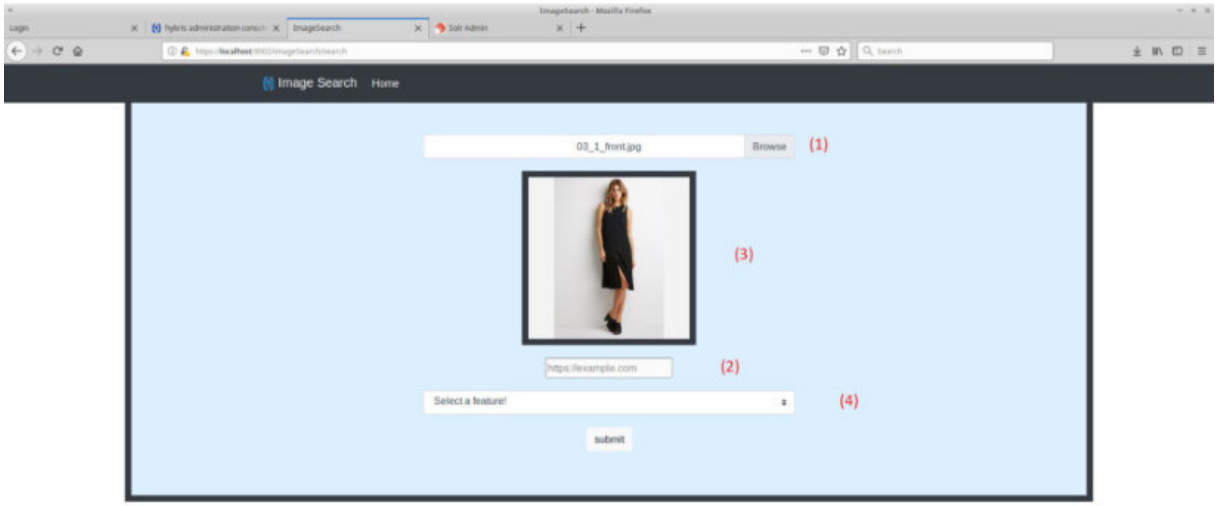

Abbildung 3.5: Interface zum Stellen einer Suchanfrage

Diese Abbildung zeigt das Benutzer-Interface zur Eingabe einer Suchanfrage an das realisierte CBVIR System. Es ist als Webseite realisiert worden, da auf diese Weise eine einfache Umsetzung der MVC-Architektur möglich war. Sowohl die Übergabe eines Bildes durch Hochladen als auch die Angabe einer URL sind möglich.

Im Bild markiert, erkennt man die zwei Eingabemöglichkeiten für ein Beispielbild durch Hochladen eines Bildes von seinem lokalen Gerät (1) oder durch Angabe einer URL, welche auf ein Bild verweist (2). In beiden Fällen wird versucht, dem Nutzer zu erlauben, nur geeignete Dateien an das System weiterzugeben. Falls eine legitime Datei hochgeladen oder verlinkt wurde, wird das entsprechende Bild im Interface angezeigt (3). Ebenfalls enthält das Interface Eingabemöglichkeiten für das zu verwendende Merkmal, welches aus einer Liste ausgewählt werden kann (4). Die Realisierung des Benutzer Interfaces wird mit Hilfe des schon einmal erwähnten *ImageSearchController* umgesetzt, welcher dem Namen entsprechend die Controller Komponente in der MVC-Architektur darstellt. Dieser verknüpft im Stile des verwendeten Spring Frameworks die verwendeten URL's mit einer Funktion, welche zur Ausführung von Programmlogik verwendet werden kann und schlussendlich auch zur Darstellung der JSP-Dateien als Website benutzt wird. Diese Java Server Pages (JSP) ermöglichen es dynamisch XML und HTML Inhalte durch Einbindung von Java-Quellcode zu erstellen und sind deshalb für die Verwendung in dieser Umsetzung bestens geeignet. Der Controller ermöglicht ebenfalls die Verarbeitung von Informationen, welche in Verbindung mit einer Anfrage an eine URL geschickt werden, wie es beispielsweise bei dem gezeigten Benutzer Interface beim Start einer Suche geschieht. Ebenfalls erlaubt es der Controller, dass Parameter an die zu generierende Website übergeben werden, um dynamisch Inhalte erstellen zu können, was in einem anderen Bestandteil noch einmal von Bedeutung ist. Die Suchanfrage und alle angegebenen Parameter werden an die Methode des Controllers mit der verknüpften URL */imageSearch/result* weitergegeben und dort verarbeitet.

Mit dieser Verarbeitung beschäftigt sich die nächste Komponente, das **Ranking** (5). Der erste Schritt im Ablauf dieser Komponente ist die Vorbereitung und Durchführung der Suche im Index mit Hilfe des angegebenen Beispielbildes. Dafür verwendet der Controller die Klasse *DefaultImageSearchService*, welche eine Vorverarbeitung des Bildes und die Merkmalsextraktion durchführt, wie sie auch bei allen indexierten Bildern verrichtet wurde, und danach eine Suchanfrage an die Solr-Instanz und den entsprechenden Index erstellt und abschickt. Zur Durchführung der Suche im Index wird auf die Grundlagen von lire-solr und den darin integrierten Prozessen zurückgegriffen, da eine von Grund auf neue Implementierung den Umfang und Aufwand der Arbeit weit überstiegen hätte. Für das Verständnis komplexerer Zusammenhänge und Vorgänge möchte ich deshalb auf die jeweiligen Dokumentationen von solr [32] und lire-solr [5] verweisen. Die gesendete Suchanfrage wird aufgrund der angegebenen Parameter mit dem *LireRequestHandler* verarbeitet. Dieser ist Bestandteil des lire-solr Plugins und für die Verarbeitung der Suchanfragen für die erstellte Anwendung verantwortlich. Eine vollumfängliche Erläuterung der internen Prozesse von Solr in Zusammenarbeit mit dem verwendeten Plugin übersteigt den Umfang dieser Arbeit, weshalb nur kurz erwähnt werden soll, dass hier die Suche und der Vergleich der Bilder durchgeführt wird. Dafür können, je nach Einstellung des Index und der Parameter der Suchanfrage, die verschiedenen Suchalgorithmen wie lineare Suche oder Hashing verwendet werden, welche in vorherigen Abschnitten bereits erwähnt wurden. Die Berechnung der Ähnlichkeit auf Basis der verwendeten Merkmale wird auf Grundlage der Distanzdefiniton durchgeführt, welche in der Klasse eines jeden Features enthalten sein muss. Diese Definition verwendet in den meisten Fällen die Manhattan Distanz, um die Ähnlichkeit als Zahlenwert darstellen zu können. Da es sich um ein Distanzmaß handelt, gilt, dass je größer der erhaltene Wert, desto unähnlicher sind sich die verglichenen Bilder. Die berechnete Distanz, oder auch *Score* bezeichnet, ist für die einzelnen Merkmale nicht normalisiert, da es für diese einfache Anwendung nicht notwendig war. Sollte eine komplexere Umsetzung realisiert werden, in der Merkmale beziehungsweise die berechneten Distanzen kombiniert betrachtet werden, so sollte eine Normalisierung durchgeführt werden. Nach der Berechnung und Erstellung der Liste erhält der DefaultImageSearchService eine Antwort des angesprochenen Solr-Servers, welche eben jene Ergebnisliste der relevantesten Produktbilder enthält. Diese Liste ist logischerweise nach dem berechneten Score sortiert, sodass ähnlichere Bilder zuerst genannt werden. Die Ergebnisliste wird zur weiteren Verarbeitung wieder an den Controller zurückgegeben, was uns schlussendlich zu dem letzten Bestandteil des umgesetzten Systems und auch des Schaubildes führt.

Die Komponente der **Ergebnisliste** (6) stellt das Endresultat einer Suchanfrage dar. Die Liste, welche von Solr zurückgeliefert wurde, enthält Angaben zu dem jeweils indexierten Bild und Verweise auf das dargestellte Produkt. Um eine einfachere Verknüpfung mit SAP Hybris Commerce zu erlauben, wird im, hier ebenfalls verwendeten, *ImageSearchController* deshalb aus dieser Liste eine Aufzählung der jeweiligen Produkte in gleicher Reihenfolge konstruiert.

Diese Liste wird zusammen mit einigen weiteren Objekten und Parametern an eine JSP-Datei weitergeleitet, welche verwendet wird, um eine dynamische Ergebnisseite zu präsentieren. Diese entspricht innerhalb der MVC-Architektur erneut der View Komponente und wird in der Abbildung 3.6 dargestellt.

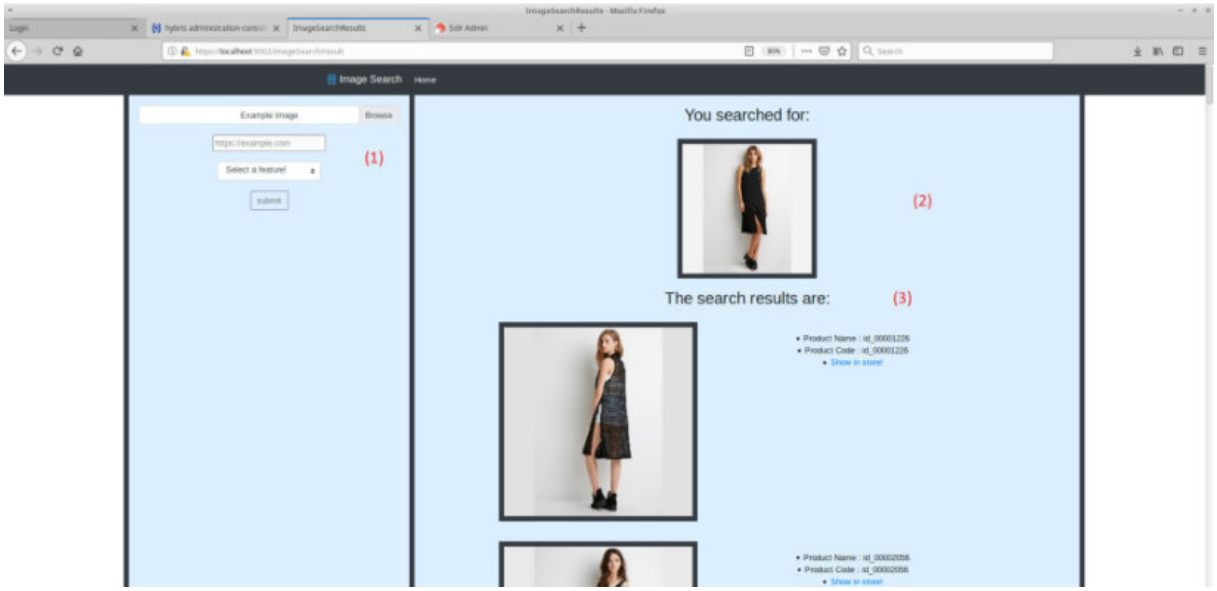

Abbildung 3.6: Interface zur Darstellung der Ergebnisse einer Suche

Diese Abbildung zeigt die Darstellung des Ergebnisses einer Suche mittels des CBVIR Systems. In der rechten Spalte wird das Beispielbild sowie die Ergebnisliste präsentiert, während in der linken Spalte erneut eine Suchanfrage an das System gestellt werden kann.

Dabei dient die linke Spalte der Website zur Eingabe einer weiteren Suchanfrage, falls diese notwendig ist (1). In der rechten Spalte wird zuerst das Bild angezeigt, welches vom Benutzer als Beispielbild zur Suche angegeben wurde (2). Im Anschluss daran folgen die Darstellungen der Produkte mit dem jeweiligen Produktbild (3), welches laut der Distanzberechnung am relevantesten bezüglich der Anfrage war. Diese Angaben beinhalten dabei verschiedene Informationen bezüglich des Produktes, sowie auch einen Link zu der entsprechenden Produktseite im Onlineshop, um sofort einen Kauf zu ermöglichen. Auf diese Art und Weise wird die Suchanfrage eines Nutzers durch das System beantwortet und schließt damit den vorgestellten Prozess ab.

Die im Schaubild 3.4 abgebildeten Prozesse und Umsetzungen wurden in diesem Kapitel einmal genauer erläutert und Zusammenhänge sowie auch Hintergründe der Realisierung beleuchtet. Das umgesetzte System orientiert sich stark an der, in der Abbildung 2.1 dargestellten, Architektur eines IR Systemes, wodurch ein einfaches Verständnis und auch eine simple Erweiterbarkeit erreicht werden soll. Bei der Umsetzung dieser CBVIR Implementierung in das Shopsystem SAP Hybris Commerce ergaben sich allerdings auch einige Probleme, welche im nächsten Abschnitt betrachtet werden sollen.

3.4 Probleme bei der Umsetzung und deren Lösung

Wie schon in den bisherigen Abschnitten dieser Arbeit erwähnt wurde, existieren bei der Umsetzung eines CBVIR Systems einige Herausforderungen und Stolpersteine. In diesem Abschnitt soll auf diejenigen Probleme eingegangen werden, welche bei dieser Implementation aufgetreten sind. Dabei handelt es sich sowohl um konzeptionelle Hindernisse als auch um eher kleine Komplikationen.

Zuerst soll die Problemstellung der **Planung und Konzeption** behandelt werden. Da es sich bei einem solchen Projekt um ein sehr komplexes Unterfangen handelt, sollte im Vorhinein eine ausgiebige Phase zur Planung der Anwendung, Festlegung des genauen Anwendungsfalles und der genauen Konzeption der internen Abläufe anberaumt werden. Ebenfalls sollte eine gewisse Grundexpertise bezüglich der verwendeten Technologien vorhanden sein, welche sowohl bei der Planung als auch bei der Umsetzung von Vorteil ist. Im Falle dieser Anwendung lag diese grundlegende Expertise im Vorhinein leider noch nicht gänzlich bei mir vor. Das nötige Wissen und die technischen Gegebenheiten erschlossen sich deshalb, auch trotz der kompetenten Beratung durch meine Betreuer, erst im Verlauf der Umsetzung im vollen Maße. Die Konzeption der realisierten Implementation wurde zwar gewissenhaft durchgeführt, allerdings hätten mit einem größeren Erfahrungsschatz einige grundlegende Architekturentscheidungen früher und besser auf die Technik angepasst getroffen werden können. Dies äußert sich zwar nur an wenigen Stellen der fertigen Umsetzung, sollte allerdings in zukünftigen Implementationen Beachtung finden.

Eine, direkt mit der ersten Herausforderung zusammenhängende, Problemstellung ist die *Codestruktur* des umgesetzten Projektes. Da auch der Quellcode, aufgrund fehlender Expertise vor Beginn der Umsetzung, dynamisch an die jeweiligen Problemstellungen und Ziele angepasst wurde, schlichen sich kleinere Mängel ein. Dies betraf sowohl die Durchsetzung der Best Practices bezüglich SAP Hybris Commerce als auch die Grundlagen des Schreibens von sauberem Code. Von Anfang an, sowohl bei der Planung als auch bei der Umsetzung, hätte man sich stärker auf Konventionen für klare Codestruktur einstellen müssen. Für tiefere Einblicke in dieses Thema empfiehlt sich ein Blick in weiterführende Literatur, beispielsweise [24]. Die Lösung dieses Problems in Form eines ausführlichen *Refactorings* würde leider den Zeitrahmen der Bearbeitung für diese Arbeit übersteigen und wird deshalb wohl erst nach Abgabe durchgeführt werden. Dies betrifft allerdings in keinster Weise die beschriebenen Funktionsweisen der Umsetzung und deren Architektur, sondern nur den Aufbau und die Umsetzung des Quellcodes.

Die bisher erwähnten Hindernisse bei der Implementation sind vor allem von konzeptioneller Form und beinhalten auch persönliche Einflüsse bezüglich gemachter Fehler und daraus

gezogenen Schlüssen. Die nachfolgenden Problemstellungen sind eher technischer Natur und beziehen sich direkter auf die umgesetzte Realisierung des CBVIR Systems.

Ein Punkt, bei dem es immer wieder zu kleineren Komplikationen kam, war der **Umgang mit Apache Solr**. Diese textbasierte Suchmaschine bietet eine Vielzahl von Möglichkeiten zur Konfiguration und ebenfalls eine sehr große Anzahl verschiedener Funktionen zur Verwendung in einem Projekt. Da es sich bei der hier umgesetzten Implementation mehr um eine Art Prototyp handelt und Probleme weitestgehend vermieden werden sollten, wurde die Integration in Solr ebenfalls recht einfach gehalten. Lediglich das lire-solr Plugin musste installiert und zusammen mit der entsprechenden Solr-Instanz an die Gegebenheiten angepasst werden. Trotz dieser simplen Implementierung kam es aufgrund von Designentscheidungen zu einigen Komplikationen, von denen hier zwei genauer erläutert werden sollen. Das erste Problem basierte auf der Tatsache, dass zu Beginn eine Kombination aus Dateiname und Verzeichnis als einzigartige Erkennungsnummer für ein Bild im Index hinterlegt wurde. Diese Erkennungsnummer war zwar eindeutig und einzigartig, allerdings führte eine zu große Übereinstimmung zwischen zwei Einträgen und deren Erkennungsnummern dazu, dass Solr diese als Duplikate ausmachte und nicht in den Index übernahm. Um dies zu umgehen, wurde daraufhin stattdessen eine zufällig generierte Identifikationsnummer verwendet und für den Namen und das Verzeichnis ein weiteres Feld in jedem Eintrag des Indexes erstellt. Ein weiterer Problemfall trat beim Erstellen der Ergebnisliste zu Tage. Da jedes Produktbild einzeln indexiert wurde, konnte es vorkommen, dass in der Ergebnisliste mehrere Bilder des selben Produktes angezeigt wurden. Da es sich um eine Produktsuche und nicht um eine reine Bildersuche handelt, war dieses Verhalten nicht erwünscht. Solr bietet für diese Art von Problemen Funktionen zur Lösung an. Sowohl das *Collapsing* [3] als auch das *Grouping* [28] sind Erweiterungen von Solr, welche es erlauben, Einträge in der Ergebnisliste zusammenzufassen, falls diese in einem bestimmten Feld übereinstimmen. Auf diese Art und Weise hätte man die Produkt-Identifikationsnummer als Feld angeben müssen, sodass für jedes Produkt nur ein Bild in die Liste gelangen könnte. Genauer gesagt wäre das Bild mit dem geringsten Distanzwert ausgewählt worden. Leider konnten beide dieser Funktionen aufgrund der veränderten Indexstruktur, welche durch die Verwendung des lire-solr Plugins bedingt wurde, nicht einwandfrei eingesetzt werden. Aus diesem Grund wurde eine Lösung umgesetzt, welche eine Neuanordnung durch Aussortieren von Duplikaten nach Beendigung der Suche durchführt. Diese Lösung ist zwar funktional und ausreichend, allerdings nicht wirklich elegant und sollte sowohl bei einem Refactoring noch einmal betrachtet als auch bei einer erneuten Umsetzung von Beginn an anderes bedacht werden. Auch weitere Einstellungen bezüglich der Solr Instanz können zur Lösung dieses Problems verwendet werden, wurden allerdings aufgrund der Komplexität und des Zeitaufwandes in dieser Situation nicht näher betrachtet.

Die letzte Gruppe von Komplikationen trat vor allem aufgrund der **angestrebten Implementierung des CBVIR Systems** auf. Hierbei offenbarten sich einige, teilweise auch schon erwähnte, Probleme solcher Anwendungen. Die erste Herausforderung dieser Art trat mit der Auswahl der zu verwendenden Merkmale auf. Auf der Basis von Lire existierten schon sowohl globale als auch lokale Features, von denen allerdings nur die globalen vollständig und einfach anwendbar in lire-solr implementiert wurden. Aus diesem Grund, aber auch wegen der guten Leistungen solcher recht einfachen Merkmale, wurden diese für die Realisierung der Anwendung verwendet. Globale Features bergen allerdings auch einige Nachteile, welche vor allem in der realitätsnahen Verwendung der umgesetzten Lösung auffallen. Globale Merkmale beziehen Eigenschaften des gesamten Bildes zu dessen Beschreibung ein, weshalb auch unwichtige Bestandteile, wie beispielsweise der Hintergrund, an Bedeutung gewinnen. Die als Beispiel indexierten Bilder sind ausnahmslos in bestimmten Posen und vor einem weißen Hintergrund aufgenommen, wodurch sich diese Bilder untereinander meist nur in wenigen Abschnitten unterscheiden. Bei der Suche eines Bildes mit einem chaotischerem Hintergrund oder anderen Bildaufteilungen kommt es daher zu ungenaueren Ergebnissen, welche zwar visuelle Ähnlichkeiten zum Beispielbild besitzen, allerdings keinen Schwerpunkt auf das Objekt von Interesse setzen. Für dieses Problem gibt es mehrere Lösungsstrategien, welche allerdings auf Grund des beschränkten Umfangs und Aufwands für diese Arbeit leider nicht umgesetzt werden konnten. Eine Möglichkeit wäre die Verwendung von lokalen Merkmalen, entweder zur Beschreibung der Bilder oder besser gesagt einzelnen Interessenspunkten, welche eine genauere Differenzierung ermöglichen. Eine zweite Variante wäre eine Vorverarbeitung der Bilder mit Hilfe einer Objekterkennung, sodass beispielsweise ein Bild zur Suche auf die Ausmaße beschnitten wird, welche das Objekt von Interesse in diesem Bild einnimmt. Beide Verfahrensweisen erlauben es den Einfluss von irrelevanten Bildabschnitten zu verringern, besitzen allerdings auch eine höhere Komplexität und bedeuten damit größeren Rechenaufwand und längere Rechenzeit. Eben jene Rechenzeit und -last führt zu einem weiteren Hindernis bei der Verwendung dieser CBVIR Anwendung. Der Indexierungsvorgang benötigt relativ viel Zeit und skaliert schlecht mit einer steigenden Menge von Bildern. Die durchschnittliche Zeit zur Indexierung eines Bildes beträgt 30 Millisekunden, was bei der bisherigen Menge von 52512 Bildern eine Zeit von circa 27 Minuten bedeutet. Diese kann zwar durch die parallele Verfahrensweise der Indexierung mittels gegebener Hardware weiter gesenkt werden, stößt jedoch aufgrund der schlechten Skalierung bei größeren Bildmengen trotzdem an gewisse Grenzen. Lösungen für dieses Problem wäre die Reduzierung der indexierten Informationen auf ein notwendiges Minimum, eine verbesserte Verfahrensweise zur Indexierung der Bilder mit weniger Zwischenschritten und auch die Verwendung eines besseren und schnelleren Codierungsverfahrens für die Merkmalsinformationen. Eine solche Verbesserung des Systems ist vor allem bei einem Einsatz in einer produktiven Umgebung umzusetzen. Schlussendlich ergab sich noch ein weiteres Problem bezüglich der **Auswertung der Leistung** der umgesetzten CBVIR Lösung. Darauf werde ich spezieller im

nächsten Kapitel der Arbeit zur *Auswertung* eingehen, allerdings möchte ich die Komplikationen hier schon einmal kurz erwähnen. Die Auswertung einer CBVIR Anwendung basiert auf der Verwendung bestimmter Qualitätsmaße zur Beurteilung der Ergebnisliste bezüglich einer bestimmten Suchanfrage. Für eine genaue Auswertung ist es allerdings notwendig, Metadaten sowohl über die indexierten Bilder als auch über die Bilder zur Suche zu benutzen. Ohne diese Metadaten kann keine Aussage über das Maß der Genauigkeit einer vorgeschlagenen Ergebnisliste getroffen werden. Aus diesem Grund war die einzig sinnvolle Vorgehensweise zur Auswertung eine Aufteilung der verwendeten Produktbilder in einen Teil zur Indexierung und einen Teil zur Suche, da die Struktur der Metadaten übereinstimmt und damit ein Maß für die Richtigkeit der Ergebnisse erstellt werden kann. Das Problem der Auswertung von CBVIR Lösungen ist allerdings in diesem Forschungsfeld weit verbreitet, da jedes System eigene Anforderung und Besonderheiten besitzt und auch weil die Zusammenstellung eines Datensatzes von Testbildern mit den jeweiligen Metadaten und Auswertungshilfen eine sehr aufwendige und teils auch komplexe Aufgabe darstellt. Aus diesem Grund fällt es auch schwer die Leistung von CBVIR Systemen untereinander zu vergleichen, da meist keine gemeinsame Grundlage für diesen Vergleich existiert [23]. Dieses Hindernis ist regelmäßig Gegenstand neuer Entwicklungen und Bemühungen für eine verbesserte Auswertung wie auch einen verbesserten Vergleich von CBIR Systemen, allerdings bleibt die Problemstellung trotzdem bestehen. Wie dieses Problem für die umgesetzte Implementation angegangen wurde und auch wie die Auswertung an sich ablief, wird im nächsten Abschnitt dieser Arbeit noch einmal genauer beleuchtet.

4 Auswertung der Leistung des Systems

Um die Leistung eines CBVIR Systems besser einschätzen zu können und darauf aufbauende Verbesserungen zu konstruieren, ist es von immenser Bedeutung die Ergebnisse einer Implementation auswerten und bewerten zu können. In welcher Art und Weise dies durchgeführt werden kann, wie es zur Auswertung der realisierten Umsetzung genutzt wurde und was aus den Ergebnissen für Schlüsse gezogen werden können, wird in diesem Abschnitt der Arbeit näher diskutiert.

4.1 Grundlagen und Planung

Bevor die Konzeption der Auswertung für die realisierte Implementation eines CBVIR Systems erläutert wird, soll vorher näher auf die Grundlagen eingegangen werden, auf denen eine solche Auswertung beruht. Diese Ausführungen basieren auf den Grundlagen von [20], allerdings empfiehlt sich für weiterführende und tiefergehende Einblicke auch der Blick in beispielsweise [1] oder [23]. Zunächst ist zu erwähnen, dass verschiedene Arten der Auswertung von Systemen existieren, von denen vor allem die heuristische und die quantitative Auswertung in dem Feld von CBVIR angewandt werden. Die heuristische Bewertung der Leistung eines Systems findet mit Hilfe von ausgewählten Experten des jeweiligen Gebiets statt, die durch die Auswertung der Ergebnisse bezüglich einiger repräsentativer Suchanfragen die Leistung des Systems bewerten. Dieses Vorgehen kann zwar gute Ergebnisse liefern und ist relativ einfach durchzuführen, allerdings birgt es auch einige Nachteile. So ist dieser Prozess beispielsweise sehr subjektiv, nur schlecht skalierbar und auch nur schwer vergleichbar. Die zweite Vorgehensweise ist die quantitative Auswertung, bei der mittels der Arbeit mit Modellen und Gütezahlen eine automatische Auswertung der Leistung eines Systems erbracht werden soll. Die dabei verwendeten Vorgehensweisen hängen stark von dem Aufbau und der Funktionsweise des jeweiligen CBVIR Systems ab, sodass zur Auswahl der richtigen Auswertungsmethoden eventuell ebenfalls Expertenwissen benötigt wird. Die quantitativen Auswertungsverfahren arbeiten dabei mit sogenannten *Gütezahlen*, auch als *Figures of Merit* oder *Ergebnismetriken* bezeichnet, welche verwendet werden, um die Leistung eines Systems bezüglich einer bestimmten Aufgabe durch einen Zahlenwert auszudrücken. Das Erstellen immer neuerer und besserer Ergebnisme-

triken ist dabei schon ein Forschungsfeld an sich und kann aus diesem Grund leider in diesem Abschnitt nur oberflächlich angeschnitten werden. Ein CBVIR System liefert in den meisten Fällen, wie in der umgesetzten Implementation, eine Ergebnisliste, sortiert nach Relevanz, oder eine Ergebnismenge, deren Elemente alle relevant für die Suchanfrage sind, ohne eine Abstufung vorzunehmen. Die Einträge der Liste oder Menge können dabei in verschiedene Gruppen eingeordnet werden[1] was mit Hilfe der Abbildung 4.1 veranschaulicht werden soll.

wahr positiv (true positive):

Das jeweilige Dokument d ist relevant und wurde als Ergebnis einer Suche zurückgeliefert, was $d \in RA$ entspricht.

wahr negativ (true negative):

Das jeweilige Dokument d ist nicht relevant und wurde nicht als Ergebnis einer Suche zurückgeliefert, was $d \in D \setminus (R \cup A)$ entspricht.

falsch positiv (false positive):

Das jeweilige Dokument d ist nicht relevant, aber wurde als Ergebnis einer Suche zurückgeliefert, was $d \in A \setminus R$ entspricht.

falsch negativ (false negative):

Das jeweilige Dokument d ist relevant, aber wurde nicht als Ergebnis einer Suche zurückgeliefert, was $d \in R \setminus A$ entspricht.

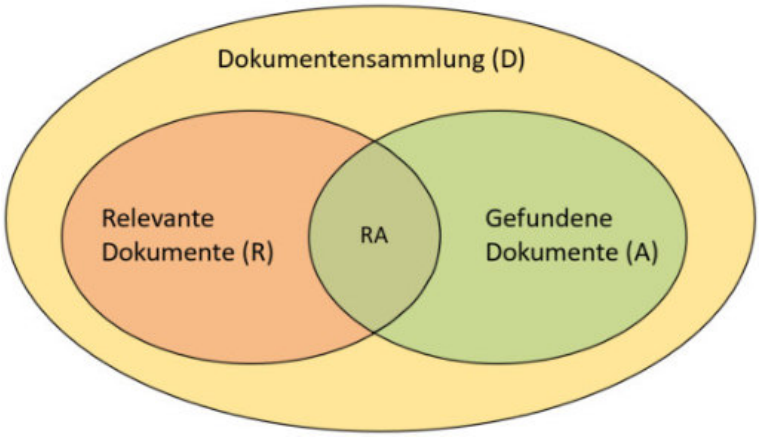

Abbildung 4.1: Venn Diagramm zur Darstellung der Dokumentenkategorien

Dieses Venn Diagramm zeigt die Menge der gesamten Dokumentensammlung und die Untermengen der relevanten Dokumente und der gefundenen Dokumente in Bezug auf eine einzelne Suchanfrage. Die Überschneidung der beiden Untermengen ist damit die Menge der Dokumente, welche relevant sind und auch gefunden wurden. Diese Überschneidung sollte logischerweise so groß wie nur möglich sein. Nach Vorbild von [20].

Falls es sich bei dem, vom CBVIR System zurückgegebenen, Ergebnis lediglich um eine unsortierte Menge an Dokumenten handelt, so sollte das Ziel sein, dass diese Ergebnismenge einen möglichst großen Anteil relevanter Dokumente enthält. Die Auswertung einer solchen Anwendung ist vergleichsweise einfach, da es sich nur um eine binäre Relevanzentscheidung handelt,

welche einfacherer zu bewerten ist. Für Fälle, in denen sortierte Ergebnislisten, eventuell sogar mit einem Score, zurückgeliefert werden, kann zu einer genaueren Auswertung ebenfalls der jeweilige Rang eines Dokumentes in der Liste und auch dessen Score verwendet werden. Diese Bewertung liefert zwar genauere Einblicke, ist allerdings auch wesentlich komplexer. Für beide Arten von Ergebnissen existieren verschiedenste Auswertungsmetriken, von denen hier einmal einige grundlegende vorgestellt werden sollen, da sie auch in der Auswertung der umgesetzten Implementation verwendet werden. Allerdings gilt auch hier, wie es schon bei den Merkmalen erwähnt wurde, das immer wieder neue Metriken konstruiert und verwendet werden können.

Zwei grundlegende Metriken sind die Präzision (Precision) p und die Trefferquote (Recall) r. Diese beiden Auswertungsmaße beziehen sich auf die Darstellung der Ergebnisse als unsortierte Menge, sodass eine Veranschaulichung mittels des Venn-Diagramms in Abbildung 4.1 möglich ist. Die Präzision bezeichnet den Anteil der relevanten und gefundenen Dokumente im Vergleich zu der Menge aller Elemente des Ergebnis.

$$p = \frac{|RA|}{|A|}$$

Die Trefferquote definiert sich als Verhältnis von der Anzahl der relevanten und gefundenen Dokumente zu der gesamten Menge an relevanten Dokumenten.

$$r = \frac{|RA|}{|R|}$$

Beide Metriken besitzen eine hohe Aussagekraft über die Ergebnisse einer Anwendung, sind allerdings auch untereinander verbunden, weshalb meist ein Kompromiss zur Optimierung einer dieser Werte eingegangen werden muss. Beispielsweise würde eine größere Ergebnismenge meist auch eine größere Trefferquote bedeuten, allerdings verschlechtert sich in den meisten Fällen zugleich die Präzision. Aus diesem Grund existieren auch Maße, welche beide Metriken miteinander kombinieren, worauf an dieser Stelle allerdings nicht weiter eingegangen werden soll. Die erwähnten Metriken beziehen sich allerdings nur auf eine unsortierte Ergebnismenge. Für eine weiterführende Auswertung empfiehlt es sich deshalb, auch den Rang eines jeden Ergebnisses in die Betrachtung einzubeziehen. Eine solche Bewertung ermöglicht die sogenannte *Precision at k* ($p_{\mathsf{at}}(k)$), also die Bestimmung der Präzision an einer Stelle k in der Ergebnisliste. Dies eignet sich vor allem für Webanwendungen, wie die hier umgesetzte Implementierung, da in diesem Umfeld meist nur eine bestimmte Anzahl von Ergebniselementen angezeigt wird. Da eine Festlegung von k als natürliche Zahl allerdings nur sehr willkürlich möglich wäre, hat sich der Einsatz der *durchschnittlichen Präzision* (Average precision) durchgesetzt [23]. Diese berechnet die Summe aller Präzisionswerte bei den Positionen der relevanten Dokumenten, welche im Anschluss durch die Menge der relevanten Dokumente geteilt wird.

$$ap = \frac{1}{|R|} * \sum_{k=1}^{N} p_{\mathsf{at}}(k) * rel(k)$$

$$rel(k) = \begin{cases} 1 \text{ , falls an der Position k ein relevantes Element ist.} \\ 0 \text{ , falls an der Position k kein relevantes Element ist.} \end{cases}$$

Diese Auswertungsmetrik beinhaltet sowohl Betrachtungen bezüglich der reinen Ergebnismenge als auch Informationen über die Ränge der jeweiligen Elemente, wodurch dieses Maß eine hohe Aussagekraft besitzt, welche allerdings etwas komplexer zu interpretieren ist. Eine hohe durchschnittliche Präzision kann zum einen durch eine hohe Präzision berechnet an der Stelle k erreicht werden, wird allerdings auch davon beeinflusst an welcher Stelle die relevanten Dokumente in der Ergebnisliste stehen. Je höher die Nummer des Dokumentes in der Ergebnisliste ist, desto weniger Einfluss hat dieses Element auf das Auswertungsergebnis. Zusätzlich soll hier erwähnt werden, dass verschiedene Variationen der Formel zur Berechnung der durchschnittlichen Präzision existieren, welche speziell auf einige Anwendungsgebiete zugeschnitten sind. Eine solche Variation wird auch zur Auswertung der umgesetzten CBVIR Lösung verwendet. Auf Grund der Einfachheit der Anwendung und der besser nachvollziehbaren Auswertung wurde die Größe der Ergebnisliste auf einen festen Wert beschränkt. Andernfalls hätte beispielsweise ein Grenzwert bezüglich des Scores oder ein bestimmter Prozentsatz relevanter Dokumente verwendetet werden können, um die Größe der Ergebnisliste auf eine dynamischere Art und Weise zu begrenzen. Die Entscheidung für eine feste Ergebnisanzahl wurde ebenfalls davon untermauert, dass ein solches Vorgehen im Bereich des E-Commerce sehr häufig anzutreffen ist. Da bei dieser Realisierung die Ergebnisliste eine feste Größe besitzt, welche meist kleiner sein wird als die Anzahl potenziell relevanter Dokumente, wird zur Normalisierung das Minimum aus entweder der Anzahl der relevanten Dokumente oder der Länge der Ergebnisliste verwendet.

$$ap = \frac{1}{|min(R, A)|} * \sum_{k=1}^{N} p_{\mathsf{at}}(k) * rel(k)$$

Da allerdings bei beiden Betrachtungen immer nur die Kombination einer Suchanfrage und den jeweiligen Ergebnissen, zusammen auch als *Topic* (T) bezeichnet, verwendet werden, sollte zur Bewertung der Leistung eines Systems der Mittelwert über viele dieser Topics gebildet werden, um eventuelle Fehlereinflüsse ausgleichen zu können. In dieser Art ist die *Mean average precision* (Map) definiert, welche, entgegen ihres Namens, den Mittelwert vieler Topics einer Average precision Berechnung ermittelt.

$$Map = \frac{1}{|T|} * \sum_{t_{\mathsf{i}} \in T} ap(t_{\mathsf{i}})$$

Diese Auswertungsmethode bietet einen guten Einblick in die Leistung eines Systems und kann ebenfalls für einen Vergleich verwendet werden, falls ähnliche Grundvoraussetzungen zwischen zwei Anwendung bestehen. Neben den hier beschriebenen Maßen zur Bewertung eines CBVIR Systems existieren noch verschiedene andere Metriken, welche beispielsweise in [1] genauer erläutert werden, allerdings an dieser Stelle aufgrund des begrenzten Umfangs der Arbeit nicht tiefer erklärt werden sollen. Auch auf die Möglichkeiten, die eine Einbindung des berechneten Scores der Ergebnisdokumente für eine Auswertung darstellt, kann hier nicht weiter eingegangen werden. Grundsätzlich wäre eine solche Auswertung aufwendiger, allerdings im Gegenzug auch genauer und aussagekräftiger. In dem Fall dieser Implementation können die berechneten Werte allerdings nicht mit Hilfe von Metadaten oder Vergleichswerten bestätigt werden, weshalb deren Verwendung für die Auswertung leider nicht möglich ist. In der durchgeführten Auswertung wurde weiterhin nicht auf die benötigte Rechendauer und die resultierende Rechenlast eingegangen. Beide Aspekte sind für Systeme in einer produktiven Umgebung zu beachten und können gegebenenfalls auch optimiert werden, allerdings waren für die prototypische Realisierung andere Gesichtspunkte von größerer Bedeutung. Da sich sowohl die Rechenlast als auch die Rechendauer in annehmbaren und akzeptablen Einheiten bewegten, wurde keine größere Aufmerksamkeit mehr auf diese gelegt, wobei auch eine Auswertung durch eine höhere Skalierung schwer umzusetzen gewesen wäre. Für eine Umsetzung mit anderen Spezifikationen sollte eine genaue Betrachtung der verwendbaren Metriken und Schwerpunkte für die Auswertung durchgeführt werden, um möglichst aussagekräftige Ergebnisse zu ermöglichen. Nachdem die Grundlagen der Auswertungsmetriken und deren Funktionsweise hier beispielhaft erläutert wurden, bezieht sich der nächste Abschnitt der Arbeit vor allem auf die Konzeption der Auswertung bezüglich der umgesetzten Implementation.

Die umgesetzte Implementierung gibt auf eine Suchanfrage eine Ergebnisliste von begrenzter Größe zurück, welche in diesem Fall 30 Elemente umfasst. Die Größe kann dabei beliebig angepasst werden, hat allerdings Einfluss auf die Präzision und die Trefferquote, wie schon einmal erläutert wurde. Da es sich um eine geordnete Ergebnisliste handelt, fiel die Wahl der verwendeten Metriken auf die Precision at k, worauf die Average precision und die Mean average precision aufbauen. Da diese Metriken einfach verständlich sind und auch die Ranginformation in der Ergebnisliste einbeziehen, sind sie für die vorliegende Anwendung sehr gut geeignet. Die Entscheidung zur Verwendung fiel dabei relativ früh im Entwicklungsprozess und beeinflusste damit auch andere Gebiete der Anwendung. Ein Beispiel dafür ist die Auswahl des Bilddatensatzes und die Gründe für dessen Verwendung. Das Problem, unter dem das Konzept der Auswertung von IR Systemen immer wieder leidet, ist die Definition von Relevanz für einzelne Dokumente. Um die Ergebnisliste bezüglich einer Suchanfrage auswerten zu können, muss vorher bekannt sein, welche Dokumente im Zusammenhang mit der gegebenen Suchanfrage als relevant angesehen werden. Diese Auswahl beziehungsweise Relevanzbestimmung kann mit

modernen Methoden in einigen Fällen automatisch durchgeführt werden, bleibt allerdings in den meisten Fällen eine manuelle Tätigkeit. Auch dabei gibt es unterschiedliche Vorgehensweisen und Ergebnisse, wobei in den meisten Fällen eine ausführliche Sammlung an Metadaten zu jedem einzelnen Dokument eines Datensatzes erstellt wird. Dies war auch der ausschlaggebende Punkt für die Entscheidung zur Verwendung des DeepFashion Datensatzes, da dieser eine vergleichsweise hohe Anzahl an Metadaten enthielt. Wie diese weiter verwendet wurden und wie die Auswertung insgesamt ablief, soll dabei im nächsten Abschnitt der Arbeit diskutiert werden.

4.2 Ablauf der Durchführung

Wie im vorherigen Abschnitt beschrieben, wurde der DeepFashion Bilddatensatz zur Auswertung der hier umgesetzten Implementation verwendet. Für die Durchführung wurde der Datensatz deshalb geteilt, um einen Anteil normal zu indexieren und den zweiten Teil als Bilder für Suchanfragen zu verwenden. Um eine große Menge an Suchdurchläufen zu erreichen, deren Daten trotzdem noch gut überschaubar und aussagekräftig sind, wurde die Anzahl der Dokumente, welche als Beispielbilder für die Suchanfragen dienen, auf 200 festgelegt. Diese 200 Produktbilder wurden zuvor zufällig aus allen 52712 möglichen Bildern des Datensatzes ausgewählt. Ursächlich für die Verwendung von Bildern des gleichen Datensatzes für die Suchanfragen waren die einheitlichen Metadaten, die eine Auswertung im Gegensatz zu Bildern anderer Art erst ermöglichen. Diese liegen als .txt oder .json Dateien dem Datensatz bei und enthalten verschiedene Daten für die Bewertung. Es lassen sich ebenfalls Informationen aus dem Verzeichnispfad der Ordnerstruktur des Datensatzes generieren. Beide Informationsquellen werden in Abbildung 4.2 gezeigt.

Da bei der Umsetzung fünf verschiedene Merkmale zur Bildbeschreibung implementiert wurden, diese sich allerdings in ihren Beschreibung an manchen Stellen überschneiden, sollten für diese Auswertung lediglich zwei Merkmale kontrolliert und bewertet werden. Dadurch blieb auch die Bewertung an sich übersichtlicher und erlaubt einfachere Verbesserung in bestimmten Bereichen. Um ein möglichst großes Feld der Bildeigenschaften abzudecken, entschied ich mich für das MPEG-7 Color Layout als farbbasiertes Merkmal und für das Pyramid Histogram of oriented Gradients als textur- und formbasiertes Merkmal. Da beide Features auf unterschiedlichen Bildeigenschaften basieren, müssen auch andere Relevanzkriterien für die Auswertung verwendet werden. Für das farbbasierte Merkmal wurde dabei die Farbangabe in den Metadaten verwendet. Da es wenige bis gar keine Metadaten bezüglich textur - und formbasierter Merkmale gab, wurde für diesen Fall die Produktkategorie, welche aus der Ordnerstruktur abgeleitet werden kann, als Relevanzkriterium festgelegt. Da bei Kleidungsstücken die meisten Produktarten eine relativ einzigartige Form und Kontur besitzen, versprach ich mir von diesem

Abbildung 4.2: Metadaten aus dem DeepFashion Datensatz

Diese Abbildung zeigt die Struktur der vorliegenden Metadaten. Der obere Abschnitt ist ein Ausschnitt aus einer .json Datei, welche genauere Informationen zu Produkten beinhaltet. Im unteren Bildabschnitt wird die Ordnerstruktur des Bilddatensatzes, aus dem ebenfalls Informationen gewonnen werden können, schemenhaft dargestellt. [18]

Relevanzkriterium eine gute Leistung. Da die Varianz der Ausdrücke sowohl bei den Kategorien, aber vor allem bei den Farben sehr groß war, musste im Vorfeld eine Zuordnung von Ausdrücken zu Oberkategorien geschehen. Die Umsetzung ist dabei durchaus diskussionswürdig und hätte auch an einigen Stellen anders ausfallen können. Schlussendlich steht eine Zuordnung der Metadaten zu einigen Oberkategorien für die Auswertung bereit. Vor der Evaluierung wurden, mit Hilfe einiger Zeilen Java Code, beispielsweise im *ProductCategoryProvider* und dem *ProductColorProvider*, die jeweiligen Metadaten aus den .txt und .json Dateien extrahiert, die eben beschriebene Zuordnung zu den Oberkategorien umgesetzt und die Informationen mit der Bildidentifikationsnummer, bestehend aus der vorgegebenen Produkt-ID und dem Dateinamen des Bildes, verknüpft. Bei der Relevanzbestimmung während der Suche wurde dann nur verglichen, ob die hinterlegten Informationen beziehungsweise Kategorien zweier Bilder eine Übereinstimmung zeigen. Sobald dies zutraf, galten die Bilder als füreinander relevant. Für die Berechnung der jeweiligen Metriken wurde der *EvaluationService* erstellt. In diesem existierten zwei Methoden, für Farbe und für Kategorie, welche als Eingabewert eine Ergebnisliste und eine Kategorie oder mehrere Farben verwenden, welche zu dem Bild der Suchanfrage gehören, und einen Abgleich mit den Einträgen der Liste durchführen. Falls auf diese Weise ein Element der Liste als relevant angesehen wird, berechnet man, wie bereits erläutert, die Präzision an dieser Stelle k. Mit diesen Daten können schlussendlich die durchschnittliche Präzision eines Suchdurchlaufes berechnet und daraufhin auch die Mean average precision ermittelt werden.

Um dieses Vorgehen automatisch auf die 200 ausgewählten Testbilder anwenden zu können, wurde eine Methode im *ImageSearchController* erstellt, welche automatisch jedes dieser Bilder als Beispielbild für eine Suchanfrage verwendet. Die daraus resultierenden Ergebnislisten wurden mit Hilfe des EvaluationService ausgewertet und die Ergebnisse pro Testbild in einer .csv Datei gespeichert, welche zur weiteren Auswertung, beispielsweise in Microsoft Excel, verwendet werden kann. Eine solche .csv Datei enthält die Angaben über jeden Eintrag in der spezifischen Ergebnisliste. So wird der Rang, die Bildidentifikationsnummer, die Farben oder die Kategorie, die Präzision an dieser Stelle, die Trefferquote an dieser Stelle und auch die daraus berechnete durchschnittliche Präzision angegeben, wie in Abbildung 4.3 gezeigt wird. Um diese beispielhafte Darstellung der Auswertungsform noch etwas anwendungsnäher zu präsentieren, werden in Abbildung 4.4 die entsprechenden Produktbilder der Anfrage und Suche ebenfalls gezeigt.

Rank	code	colors	p(k)	r(k)	ap(k)
0	id_00000095/01_1_front.jpg	[black, gold]			
1	id_00007881/04_3_back.jpg	[black]	1	0.038461538	
2	id_00000762/02_7_additional.jpg	[black, natural]	1	0.076923077	
3	id_00007676/01_3_back.jpg	[black, silver]	1	0.115384615	
4	id_00005986/03_3_back.jpg	[black, white]	1	0.153846154	
5	id_00001463/05_3_back.jpg	[olive, black]	1	0.192307692	
6	id_00007073/01_3_back.jpg	[black]	1	0.230769231	
7	id_00004493/01_1_front.jpg	[black, white]	1	0.269230769	
8	id_00001792/02_3_back.jpg	[charcoal]	0.875	0.269230769	
9	id_00002517/01_1_front.jpg	[black]	0.888888889	0.307692308	
10	id_00001467/01_1_front.jpg	[black, turquoise]	0.9	0.346153846	
11	id_00000945/01_1_front.jpg	[black]	0.909090909	0.384615385	
12	id_00005908/01_7_additional.jpg	[black]	0.916666667	0.423076923	
13	id_00002446/01_1_front.jpg	[black]	0.923076923	0.461538462	
14	id_00003650/03_7_additional.jpg	[black]	0.928571429	0.5	
15	id_00001705/03_7_additional.jpg	[black]	0.933333333	0.538461538	
16	id_00001619/02_3_back.jpg	[charcoal, grey]	0.875	0.538461538	
17	id_00006441/05_1_front.jpg	[black, olive]	0.882352941	0.576923077	
18	id_00004503/02_7_additional.jpg	[black, cream]	0.888888889	0.615384615	
19	id_00001723/05_1_front.jpg	[black, yellow]	0.894736842	0.653846154	
20	id_00005152/10_1_front.jpg	[navy, white]	0.85	0.653846154	
21	id_00002817/04_3_back.jpg	[black]	0.857142857	0.692307692	
22	id_00005698/01_3_back.jpg	[black, white]	0.863636364	0.730769231	
23	id_00006062/04_1_front.jpg	[black]	0.869565217	0.769230769	
24	id_00002365/01_1_front.jpg	[black, taupe]	0.875	0.807692308	
25	id_00001048/02_3_back.jpg	[black, gold]	0.88	0.846153846	
26	id_00005048/02_7_additional.jpg	[black, cream]	0.884615385	0.884615385	
27	id_00005703/01_3_back.jpg	[black, gold]	0.888888889	0.923076923	
28	id_00000261/02_4_full.jpg	[black]	0.892857143	0.961538462	
29	id_00006785/03_2_side.jpg	[taupe, black]	0.896551724	1	
30	id_00006430/03_7_additional.jpg	[olive]	0.866666667	1	
					0.799128813

Abbildung 4.3: Ergebnis einer Suche dargestellt in einer .csv Datei

Die Abbildung zeigt den Inhalt einer .csv Datei, welche die Informationen zur Auswertung einer Suchanfrage darstellt. Der Eintrag mit Rang 0 ist jenes Bild, welches als Suchanfrage verwendet wurde und der letzte Eintrag gibt die durchschnittliche Präzision an. Einem jeden Ergebniselement ist eine oder mehrere Farben zugeordnet, welche in bestimmte Kategorien eingeteilt sind. Entspricht diese Kategorie jenen des Suchdokumentes, wird das Ergebniselement als relevant angesehen, was an der Erhöhung des Wertes der Trefferquote erkennbar ist. Das gleiche Vorgehen wird auch bei der Auswertung mit Hilfe der Produktkategorien verwendet.

Abbildung 4.4: Ergebnisse einer beispielhaften Suchanfrage

Diese Darstellung zeigt das Bild einer Suchanfrage sowie die auf dieser Grundlage gefundenen Produktbilder. Es handelt sich dabei um die selbe Suchanfrage, welche auch schon in der Abbildung 4.3 behandelt wurde. Das oberste Bild wurde zur Suche verwendet. Die folgenden Produktfotos sind die Ergebnisse der Bildersuche, wobei die Bilder an ihrem linken Rand durch den jeweiligen Rang in der Ergebnisliste und einem Symbol bezüglich ihrer Relevanz ergänzt wurden. Ein grünes Häkchen steht dabei für die Relevanz eines Elements, während das rote Kreuz bedeutet, dass ein Dokument irrelevant bezüglich der Suchanfrage ist.

Um den gesamten Ablauf der Evaluierung und Bewertung des umgesetzten CBVIR Systems noch einmal nachvollziehbarer und kompakter darzustellen, soll an dieser Stelle die Abbildung 4.5 eine Zusammenfassung aller Abläufe und Funktionen liefern. Schlussendlich soll noch erwähnt werden, dass das Beifügen der gesamten Auswertung in gedruckter Form aufgrund des Umfangs nicht möglich war. Allerdings sind die entsprechenden Auswertungstabellen und auch die Zuordnung der Farben und Kategorien der elektronischen Form dieser Arbeit beigelegt.

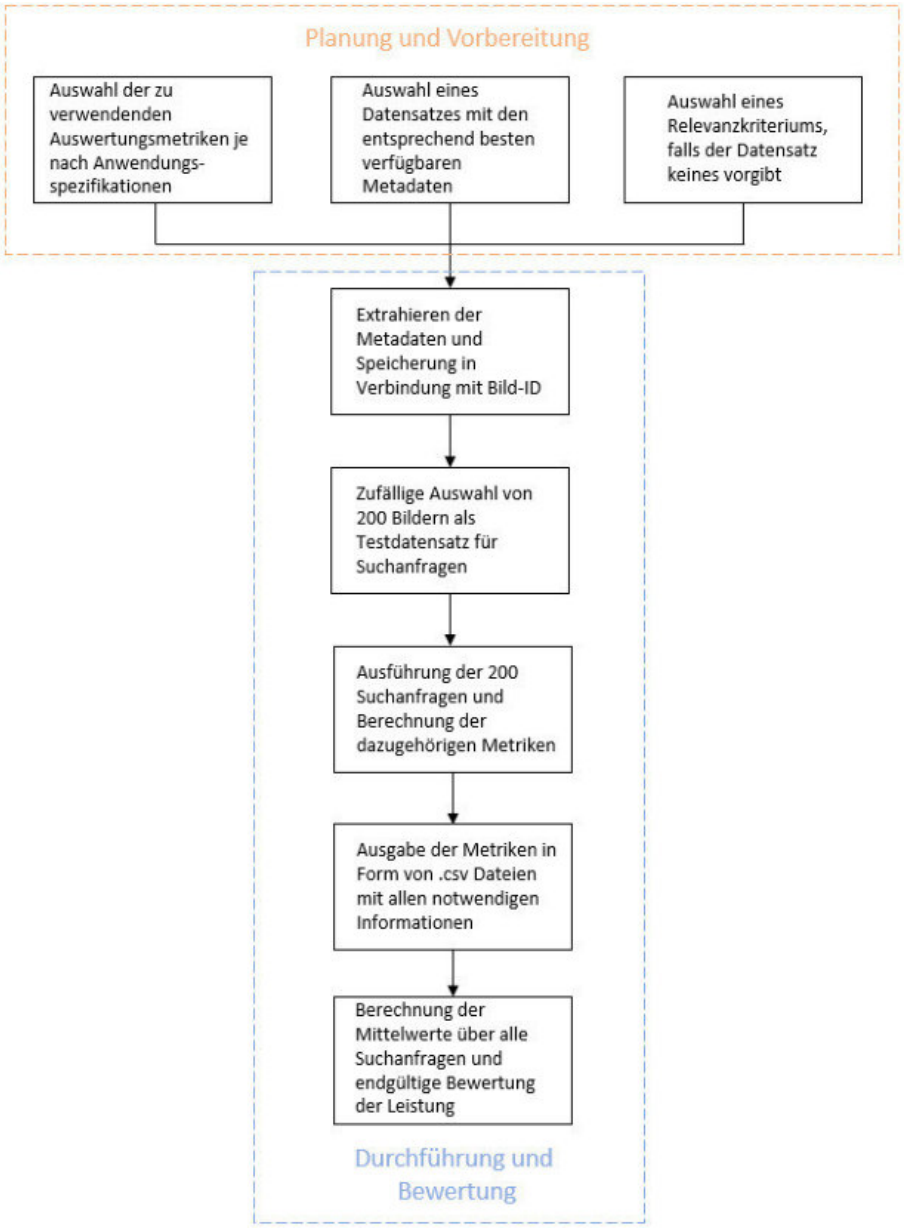

Abbildung 4.5: Ablauf der Auswertung

Diese Abbildung zeigt den Ablauf der Auswertung. Der orangefarbene Bereich bildet dabei die Prozesse der Planung und Vorbereitung ab, während in dem blaufarbenen Bereich der tatsächliche Ablauf bei der Evaluierung aufgezeigt wird.

4.3 Diskussion der Ergebnisse

Wie schon in dem vorherigen Abschnitt der Arbeit erwähnt wurde, liegen die Ergebnisse aller Suchdurchläufe der Auswertung in Form von .csv Dateien vor. Der ebenfalls darin berechnete Wert der durchschnittlichen Präzision einer jeden Ergebnisliste soll dabei als Grundlage zur Berechnung der Mean average precision dienen. Diese wird für die endgültige Auswertung der Leistung des umgesetzten CBVIR Systems benutzt. Für die Berechnung wurde die bereits genannte Formel verwendet, was bedeutet, dass der Mittelwert der durchschnittlichen Präzision über alle 200 Suchdurchläufe berechnet wurde. Dies wurde jeweils für das farbbasierte- und für das textur-/formbasierte Merkmal durchgeführt. Die Berechnung für beide Merkmale lieferte folgende Ergebnisse:

Map für farbbasiertes Merkmal : 0,26482377
Map für textur-/formbasiertes Merkmal : 0,249876541

An diesen Werten kann man zunächst einmal feststellen, dass die beiden ausgewählten Merkmale ungefähr gleich gute Ergebnisse liefern. Insgesamt sind diese Werte zwar relativ klein, was auf eine eher passable Leistung schließen lässt, allerdings sind sie für eine erste prototypische Implementierung eine gute Basis, um Verbesserung einzuführen. Um eine kurze Einordnung der Ergebnisse zu erlauben, soll hier erwähnt werden, dass sich die durchschnittlichen Leistungen von CBVIR Systemen beziehungsweise deren Merkmalen meist im Bereich einer Mean average precision von 0,3 bis 0,6 bewegen [20] [6]. Es ist anzumerken, dass globale Merkmale meist am schlechtesten abschneiden (Map: 0,2-0,4), lokale Features eine durchaus gute Leistung zeigen (Map: 0,3-0,6) und einige fortschrittliche Modelle mit Hilfe von maschinellem Lernen und künstlicher Intelligenz sogar sehr gute Ergebnisse liefern können (Map: 0,5-0,8) [25]. Diese Werte hängen allerdings stark von dem jeweils ausgewerteten Merkmal oder System und vor allem von dem verwendeten Bilddatensatz sowie dessen Metadaten ab. Ein Vergleich dieser Werte ist deshalb zwar durchaus möglich, allerdings nicht immer sehr sinnvoll, da ein besonderer Augenmerk auf die gemeinsamen und unterschiedlichen Grundlagen der Auswertung zu legen ist. Trotz alledem kann aus diesem Wertebereich geschlussfolgert werden, dass sich die Leistung des Systems eher im unteren Spektrum ansiedelt, was allerdings den verwendeten Merkmalen durchaus entspricht. Bevor eine weitere Interpretation dieser Werte erfolgt, soll allerdings eine Fehlerbetrachtung bezüglich der Auswertung und dem gegebenen System durchgeführt werden. Ein großes Fehlerpotential bei der Auswertung der Anwendung lieferte die Bestimmung der Relevanz aus den zugrundeliegenden Metadaten. Vor allem die Zuordnung zu Oberkategorien lässt einige Fehlerquellen entstehen, da hier eine Auswahl durch heuristische Methoden entstanden ist. Auf diese Art und Weise wurden eventuell für das System ähnliche

Bilder in verschiedene Kategorien eingeteilt. Auf der anderen Seite hätte eine zu große Anzahl unterschiedlicher Relevanzklassen beziehungsweise Kategorien wohl ebenfalls zu einer schlechteren Bewertung geführt, da viele ähnliche Produkte und Bilder mit unterschiedlichen Daten versehen waren. Für weitere Untersuchungen und Projekte dieser Art sollte daher unbedingt großen Wert auf die Grundlagen zur Auswertung gelegt werden. Falls dabei kein passender Datensatz mit beiliegenden Metadaten gefunden werden kann, sollten sogar Überlegungen angestellt werden, diesen selbst zu erstellen. Für zukünftige Projekte dieser Art mag diese Aufgabe durch den Einsatz einer automatischen Kategorisierung, beispielsweise mit Hilfe maschinellen Lernens, erleichtert werden, sie sollte allerdings trotzdem ein Hauptaugenmerk bei der Konzeption eines solchen Systems darstellen. Die zweite größere Fehlerquelle bildet die ausschließliche Verwendung von globalen Merkmalen, da auf diese Art das gesamte Bild für die Merkmalsextraktion verwendet wird. In den meisten Fällen einer bildbasierten Produktsuche ist allerdings nur der Bildausschnitt relevant, welcher das gesuchte Objekt zeigt. Die Auswirkungen dieser Fehlerquelle wurden vermutlich durch den einheitlichen visuellen Aufbau der indexierten Bilder relativ klein gehalten, da dadurch die meisten Bildunterschiede durch die Kleidungsstücke verursacht wurden. Falls allerdings mehrere Kleidungsstücke auf einem Bild zu sehen waren, wurden natürlich beide in die Suche einbezogen. Besonders auffällig wird dieses Problem bei der Verwendung eines realitätsnäheren Bildes für die Suchanfrage, welche beispielsweise eine komplette Person vor einem uneinheitlichen Hintergrund zeigt. Für den Benutzer mag nur ein Kleidungsstück auf diesem Bild von Relevanz sein, allerdings erlaubt das System in der jetzigen Form keine weitere Konkretisierung, sowohl durch Bearbeitung des Bildes selbst oder auch durch Hinzufügen weiterer Suchinformationen. Auf eine Suchanfrage dieser Art werden die visuell ähnlichsten Bilder zurückgegeben, allerdings ist nicht zu garantieren, dass eine Relevanz für den Benutzer besteht. Diese beiden beschriebenen Fehlerquellen hatten wohl den größten Einfluss auf die Auswertung des Systems, konnten allerdings aufgrund der Einfachheit der Anwendung selbst, der beschränkten Auswahl an Testdaten und nicht zuletzt dem begrenzten Umfang dieser Arbeit nicht korrigiert werden. Natürlich haben auch andere kleinere Probleme eine Rolle bei der Leistung des Systems gespielt, welche teilweise in vorherigen Kapiteln schon besprochen wurden, diese lassen sich allerdings schwerer nachvollziehen als die beiden bisher genannten. Die Verbesserung und Lösung aller genannten und nicht genannten Probleme dieser Anwendung und ihrer Auswertung sind dabei Aufgaben für künftige Arbeiten an dem umgesetzten System und sollen noch einmal in dem Abschnitt zukünftige Verbesserungen genauer betrachtet werden.

Die gewonnenen Ergebnisse bezüglich der Leistung des Systems, welche oben bereits aufgeführt wurden, sind eher in dem unteren Leistungsspektrum anzuordnen. Dies ist für diese eher prototypische Anwendung zwar nicht wünschenswert gewesen, lag allerdings im Rahmen der erwarteten Szenarien. Ein Vergleich mit anderen Systemen ist aufgrund der unterschiedlichen

Definition von Relevanz nur bedingt möglich, allerdings erlauben die Ergebnisse eine Kontrolle der Auswirkung von Veränderungen an dem umgesetzten System. Wie schon erwähnt, sind die Average precision und die darauf basierende Mean average precision zwar zwei sehr aussagekräftige und weit verbreitete Auswertungsmetriken, allerdings lassen sie sich nicht in jedem Fall einfach interpretieren. Die Gründe für die eher schlechten Ergebnisse können zum einem an einer geringen Anzahl relevanter gefundener Ergebnisse liegen oder auch durch deren Platzierung an höheren Rängen in der Ergebnisliste ausgelöst werden. Wahrscheinlich ist dabei eine Mischung dieser beiden Ereignisse, deren Auftreten seine Gründe vor allem in den schon genannten Fehlerquellen finden sollte. Auf der Basis des Endergebnisses der Auswertung wurde eine Grundlage für weitere Verbesserungen der umgesetzten Implementierung gelegt, auf die in Zukunft aufgebaut werden kann.

5 Fazit

5.1 Zusammenfassung der Ergebnisse

Die Zielstellung dieser Arbeit war die Implementierung einer bildbasierten Produktsuche im Shopsystem SAP Hybris Commerce sowie die Auswertung der Leistung einer solchen Suche und der Analyse der aufgetretenen Probleme und deren Lösung. Schlussendlich wurden alle diese Ziele umgesetzt. Die erstellte *imageSearch* Erweiterung für SAP Hybris Commerce erlaubt die Einbindung der Funktionsweisen eines CBVIR Systems mit Hilfe des integrierten Solr-Servers und dem darin installierten Plugin lire-solr. Es kann ein Beispielbild zur Suchanfrage durch das Hochladen einer lokalen Datei oder durch die Übergabe einer URL an das System weitergeleitet werden. Die Anwendung extrahiert die verwendeten Merkmale zur Bildbeschreibung und vergleicht diese mit allen schon indexierten Bilddaten. Diese Bilddaten basieren auf den Produktbildern aus dem Onlineshopsystem SAP Hybris Commerce. Als Ergebnis dieses Vergleiches steht eine Liste von Produktbildern, welche nach dem berechneten Score, welcher der visuellen Ähnlichkeit entsprechen sollte, sortiert ist. Die Darstellung des Interfaces zur Suche und der Ergebnisliste sind als eigenständige Webseiten umgesetzt. Die Elemente der Ergebnisliste beinhalten allerdings Produktinformationen und auch einen Link zur der jeweiligen Produktseite im Onlineshop, wo das entsprechende Produkt gekauft werden kann. Die bei der Umsetzung aufgetretenen Probleme basieren dabei teilweise auf Einschränkungen, die CBVIR Systemen meist generell anhängen, aber auch auf Herausforderungen, die spezieller auf Grund von Designentscheidungen und der Implementierung dieser Anwendung aufgetreten sind. Vor allem bei der Konzeption existiert dabei kein klares Problem und keine klare Lösung, sondern es handelt sich mehr um das Finden eines Kompromisses zwischen verschiedenen Faktoren, um die Spezifikationen einer Anwendung so gut wie möglich zu erfüllen. Für diese Fälle sollte diese Arbeit zumindest Grundlagen geliefert haben, um den Planungsvorgang einer solchen Implementierung einfacher und einheitlicher zu gestalten. Ebenfalls können anhand der umgesetzten Lösung die Auswirkung solcher Entscheidungen auf das System und seine Leistung beobachtet werden. Die darauf basierenden spezielleren Problemstellungen der realisierten Anwendung werden dahingehend genauer analysiert und gelöst. Da allerdings manche Herausforderungen zur Lösung einen tieferen Eingriff in das System erfordern, wurde für diese Fälle nur ein Lösungsvorschlag geliefert, dessen Umsetzung in die Kategorie der zukünftigen Verbesserungen

aufgenommen wurde. Die Basis für die Kontrolle der künftigen Verbesserung des Systems liefert dabei die Auswertung der Leistung der Anwendung. In dieser wurde auf etablierte Metriken für eine solche Bewertung zurückgegriffen. Auf Grund des verwendeten Datensatzes und der Definition der Relevanzkriterien besteht allerdings auch hier noch Verbesserungspotential. Die gelieferten Werte können dennoch zum Vergleich der Leistung des System vor und nach einer Verbesserung verwendet werden, da sich auch in der Auswertung die genannten Probleme widerspiegeln. Schlussendlich wurde das angestrebte Ziel dieser Arbeit allerdings in jedem Punkt umgesetzt. Es wurde eine prototypische Implementation eines CBVIR Systems in das Shopsystem SAP Hybris Commerce erstellt, die Anwendung wurde mit etablierten Metriken des Information Retrieval ausgewertet und die aufgetretenen und bestehenden Probleme wurden analysiert und, solang es der Umfang der Arbeit zugelassen hat, auch gelöst. Neben dieser Umsetzung bietet diese Arbeit aber vor allem den Aufbau einer Wissensbasis für die Implementierung von Systemen dieser Art und hat diesbezüglich die wichtigsten Punkte abgesteckt und erläutert.

5.2 Zukünftige Verbesserungen

Im Laufe der Ausführungen dieser Ausarbeitung wurden neben der Bau- und Funktionsweise von CBVIR Systemen ebenfalls die direkt und indirekt daraus folgenden Herausforderungen und Probleme erläutert und diskutiert. In diesem Abschnitt soll deshalb der Spielraum analysiert werden, welche jene Problemstellungen zur Verbesserung von Anwendungen dieser Art lassen. Diese Neuerungen konnten auf Grund des Umfang der Arbeit allerdings nicht direkt umgesetzt werden und sind deshalb, entsprechend der Kapitelüberschrift, für künftige Veränderungen oder neue Projekte vorgesehen. Zuerst soll dabei auf die hier umgesetzte Anwendung im speziellen eingegangen werden. Die Hauptprobleme wurden bereits in der Auswertung erläutert. Wie dort schon erwähnt wurde, verwendet die realisierte Implementierung nur globale Merkmale zur Beschreibung eines Bildes. Die daraus resultierenden Herausforderungen lassen sich dabei auf zwei Arten lösen. Die erste beinhaltet die Verwendung komplexerer Merkmale, die bessere Differenzierung zwischen den Bildern erlauben. Vor allem die Verwendung von lokalen Merkmalen, welche Interessenpunkte in Bildern feststellen und auch wiedererkennen sollen, könnten dabei eine durchaus wichtige Rolle einnehmen. Dadurch würde nicht mehr das gesamte Bild mit all seinen eventuell irrelevanten Bereichen in die Bewertung einfließen, sondern nur die relevanten Bildausschnitte das Ergebnis beeinflussen. Der Nachteil dieses Ansatzes ist die erhöhte Komplexität, welche eine höhere Rechenlast und Rechendauer nach sich zieht, weshalb in dieser prototypischen Anwendung vorerst auf deren Verwendung verzichtet wurde. Der zweite Lösungsvorschlag beinhaltet die Benutzung einer Objekterkennung, eventuell kombiniert mit einer Art Nutzerfeedback, um das Bild auf den relevanten Bereich zuzuschneiden. Allerdings

würde auch diese Vorverarbeitung eine höhere Rechenlast und -dauer bedeuten. Beide Ansätze beruhen darauf, den Anteil und Einfluss der relevanten Bildausschnitte in der Beschreibung der Bilder zu erhöhen. Neben diesen beiden grundsätzlichen Verbesserungsvorschlägen existieren weitere kleine Problemstellungen, welche für eine prototypische Anwendung zunächst einmal ignoriert wurden, allerdings in einer produktiven Umgebung anders gelöst werden könnten. Ein Beispiel dafür ist die Aktualisierung des Solr-Index falls neue Produkte und Bilder hinzugefügt oder gelöscht wurden. In der jetzigen Umsetzung erledigt dies ein Cronjob, der in bestimmten Zeitabständen ausgeführt wird. Eine sinnvollere Einbindung würde ein dynamisches Update des Index erlauben, also das Hinzufügen oder Löschen einzelner Bestandteile zur Laufzeit der Anwendung, ohne den gesamten Index zu beeinflussen. Auf diese Art und Weise würde die Administration der Inhalte dieses Systems sehr viel einfacher werden. Ein ebenfalls durchaus interessanter Verbesserungsansatz ist die Verwendung weiterer Parameter zur Spezifikation der Suche. So könnte beispielsweise durch die Angabe bestimmter Kategorien oder Farben eine entsprechende Vorauswahl der Bilder getroffen werden. Ebenfalls kann die Einbindung von Feedback durch den Nutzer eine sinnvolle Erweiterung sein, indem beispielsweise indexierte Produktbilder direkt als Suchanfrage verwendet werden können oder der Nutzer mehrere, für ihn relevante, Bilder zur Verfeinerung der Suche angeben oder auswählen kann. Neben den genannten Ansätzen existieren noch einige kleinere Verbesserungsmöglichkeiten in der prototypisch umgesetzten Anwendung, von denen die hier erläuterten allerdings die größten Verbesserungen versprechen sollten. Abseits der realisierten Umsetzung bietet allerdings auch eine Überarbeitung der Auswertung ein Verbesserungspotential. Dabei sollte vor allem auf den verwendeten Datensatz und die beiliegenden Metadaten geachtet werden auf deren Basis die Festlegung und Bestimmung der Relevanz erfolgt. Eine möglichst genaue Relevanzdefinition stellt eine solide Grundlage zur Auswertung eines Systems und den eingebundenen Verbesserungen dar. Nicht zuletzt ist zu erwähnen, dass auch die Verwendung von effektiveren und effizienteren Merkmalen eine Chance für die Zukunft dieses Projektes darstellt. Sowohl die Erstellung gänzlich neuer Features als auch die Kombination bereits existierender Merkmale bieten interessante Möglichkeiten für die Verbesserung dieses Systems und für die Leistung von künftigen Projekten.

Neben den erläuterten Verbesserungen für die umgesetzte Anwendung soll hier auch noch einmal auf den Fortschritt im gesamten Forschungsfeld der CBVIR Anwendungen eingegangen werden. Durch die steigende Anzahl an multimedialen Inhalten und der immer wieder prophezeiten verbreiteten Verwendung der bildbasierten Suche wird fortwährend an entsprechenden Lösungen entwickelt [17]. Sowohl die Konstruktion immer besserer und umfassenderer Merkmale zur Bildbeschreibung als auch die Entwicklungen neuer Möglichkeiten der Bildvorverarbeitung sind dabei naheliegende Schwerpunkte, in denen grundlegende Sachverhalte kontinuierlich erforscht und erweitert werden. Neben diesen sehr bildbasierten Inhalten

ist allerdings auch die Forschung bezüglich effektiver und effizienter Suchverfahren und Indexstrukturen von immenser Bedeutung, um bei umfangreichen Anwendungen eine möglichst geringe Rechenzeit zu ermöglichen. Ebenfalls interessant sind Möglichkeiten bezüglich der Berechnung der resultierenden Rankings in Form einer Ergebnisliste. Auch hier werden weiterhin intensive Forschungen betrieben, um ein möglichst nutzernahes Resultat zu erhalten, indem eventuell vorher gesammelte Nutzerdaten mit in die Auswertung einbezogen werden. Sowohl auf den Suchvorgang und die Indexstruktur als auch auf das Ranking konnte im Laufe dieser Arbeit nur kurz eingegangen werden, da diese Bereiche nicht ausschließlich für CBVIR Systeme bedeutend sind. Dennoch bieten sie ein großes Potential zur Verbesserung von Anwendungen dieser Art, wie sie bei anderen Themengebieten bereits gezeigt haben. Auch die Auswertung von CBVIR Anwendung ist und bleibt ein relevantes Themengebiet des Forschungsfeldes und wird unter anderem durch die Konstruktion von Auswertungsmetriken und vor allem der Erstellung neuer Datensätze und deren Annotierung immer weiter vorangetrieben. Schlussendlich soll dabei noch einmal auf das große Potential hingewiesen werden, welches maschinelles Lernen und künstliche Intelligenz bezüglich aller angesprochenen Forschungsbereiche birgt. Durch die automatische Erstellung und Auswahl von Merkmalen, intelligenter Indexstrukturen, Modelle für das Ergebnisranking und nicht zuletzt der Annotierung und Auswertung von Datensätzen bietet diese Technologie unglaubliche Möglichkeiten zur Verbesserung von CBVIR Systemen. Allerdings befindet sich auch diese Technologie noch in ihren Anfängen und wird vor allem durch die beschränkte Menge von Daten zum Lernen und der Breite des Themengebietes noch in ihrer Verwendung für CBVIR Lösungen gebremst. Mit weiteren Entwicklungen in beiden Themenfeldern rückt eine verbreitete Anwendung einer bildbasierten Suche allerdings immer näher.

5.3 Ausblick VIR im E-Commerce

Eine Verwendung innerhalb eines Onlineshops oder, besser gesagt, in Zusammenhang mit einer E-Commerce Anwendung ist ein sehr häufig genannter Anwendungsbereich für CBVIR Systeme. Wie schon in der Einleitung erwähnt wurde, ist die Verwendung dieser Art der Produktsuche allerdings noch nicht wirklich etabliert. Wie in dieser Arbeit beschrieben wurde, umfasst das Feld des Information Retrieval, vor allem bei visuellen Inhalten, einen großen Aufgaben- und Forschungsbereich. Die Bedeutung verschiedener Einflüsse zieht dabei einige Herausforderungen in der Umsetzung nach sich, die im Verlauf der Arbeit sowohl theoretisch als auch teilweise am praktischen Beispiel diskutiert wurden. Der größte Vorteil, den eine bildbasierte Suche verspricht, ist das Wegfallen einer komplexen, meist schriftlichen, Beschreibung eines Produktes und Bildinhaltes. Auf diese Art und Weise kann ein Kunde sehr viel zielgerichteter und effektiver suchen, was logischerweise eine Gewinnsteigerung des Unternehmens bedeuten

sollte. Auch aus diesem Grund wird dieser Anwendungsfall wohl immer wieder erwähnt. Ein großes Problem von E-Commerce Anwendungen ist allerdings die, schon einmal erwähnte, Breite ihres Anwendungsbereiches. Anstatt einer einfachen Kategorisierung einer kleinen Anzahl von Bildern mit geringer Varianz und einheitlichen Produktionsbedingungen handelt es sich in den meisten Anwendungsfällen um eine mehrstufige Relevanzbestimmung einer sehr großen Anzahl von Bildern mit großer inhaltlicher Varianz. Vor allem die Suchanfragen, welche die Nutzer in Alltagssituationen stellen, sind dabei durch uneinheitliche Produktionsumstände geprägt. Mit den einfachen Mitteln, die in der hier umgesetzten Anwendung verwendet wurden, ließe sich ein solch umfangreiches und skalierbares System nicht verwirklichen. Frühe Anwendungen scheiterten meist an diesem begrenzten Anwendungsbereich, da sie auf diese Weise nur spezielle Suchanfragen bezüglich eines Themengebietes sinnvoll beantworten konnten. Durch technische Fortschritte und Entwicklungen wurden vereinzelt immer wieder Versuche einer Umsetzung möglich, die jedoch nie zu einer weiten Verbreitung und Anwendung führten. Erst die Verwendung von maschinellem Lernen und künstlicher Intelligenz brachte einen großen Fortschritt bei der Verwirklichung von CBVIR Anwendungen im E-Commerce Bereich. Große Digitalunternehmen wie Google [11] oder Amazon [13] trieben die Forschungen in diesem Bereich voran, wobei auch etablierte Onlineshopanbieter wie Zalando [30] oder Ebay [9] eigene Umsetzungen in ihre Anwendung etablierten. Diese Entwicklung ist allerdings logisch und nachvollziehbar wenn bedacht wird, wie viele Trainingsdaten für die Erstellung einer CBVIR Anwendung mit Hilfe von maschinellem Lernen und künstlicher Intelligenz benötigt werden und wer Zugriff auf diese Daten hat. Aus diesem Grund werden wohl die ersten etablierten Anwendungen von eben jenen großen Digitalunternehmen umgesetzt, welche die notwendigen Ressourcen dafür bereitstellen können. Eine Implementierung wird dabei vermutlich zuerst in das eigene technologische Ökosystem erfolgen, bevor einige diese Software als eine Art Service zur Einbindung in andere Anwendungen anbieten. Ich schätze, dass eine sofort verwend- und konfigurierbare Umsetzung eines solchen Suchverfahrens in Shopsystemen wie SAP Hybris Commerce oder Intershop erst nach einer bestimmten Zeit zu erwarten ist, sollte sich die Verwendung dieser Suchmethode tatsächlich weiter verbreiten. Dies hängt allerdings nicht nur mit dem technologischen Fortschritt, sondern auch mit der Akzeptanz und Verwendung durch den Nutzer zusammen. Die bildbasierte Produktsuche sollte daher einen wahrnehmbaren Vorteil für den Benutzer bieten und eine möglichst gute Benutzungserfahrung erlauben. Mit der Kombination aus technischem Fortschritt und der Nutzungsbereitschaft der Kunden könnte die Verwendung von CBVIR Lösungen in Onlineshopsystemen in Zukunft eine weite Verbreitung finden und damit eventuell der oft erwähnten Bedeutung für die Produktsuche gerecht werden. Eine Vorhersage über den Verlauf der weiteren Entwicklungen in diesem Bereich lässt sich logischerweise nur schwer treffen, allerdings darf man auf die Forschungen und Neuerungen dieses Themengebiets sehr gespannt sein und erwarten, dass diese sich ebenfalls auf den E-Commerce Markt auswirken werden.

Literatur

[1] Jenny Benois-Pineau, Frederic Precioso und Matthieu Cord. *Visual indexing and retrieval*. SpringerBriefs in computer science. New York: Springer, 2012. ISBN: 9781280793257. DOI: 10.1007/978-1-4614-3588-4. URL: http://dx.doi.org/10.1007/978-1-4614-3588-4.

[2] Anna Bosch, Andrew Zisserman und Xavier Munoz. „Representing shape with a spatial pyramid kernel“. In: *Proceedings of the 6th ACM international conference on Image and video retrieval*. Hrsg. von Nicu Sebe und Marcel Worring. New York, NY: ACM, 2007, S. 401–408. ISBN: 9781595937339. DOI: 10.1145/1282280.1282340.

[3] *Collapse and Expand Results | Apache Solr Reference Guide 7.5*. 18.09.2018. URL: https://lucene.apache.org/solr/guide/7_5/collapse-and-expand-results.html (besucht am 22. 05. 2019).

[4] Ritendra Datta u. a. „Image retrieval: Ideas, influences, and trends of the new age“. In: *ACM Computing Surveys (CSUR)* 40.2 (2008), S. 5. ISSN: 0360-0300. DOI: 10.1145/1348246.1348248. URL: http://dl.acm.org/ft_gateway.cfm?id=1348248&type=pdf.

[5] *dermotte/liresolr*. URL: https://github.com/dermotte/liresolr (besucht am 22. 05. 2019).

[6] Thomas Deselaers, Daniel Keysers und Hermann Ney. „Features for image retrieval: an experimental comparison“. In: *Information Retrieval* 11.2 (2008), S. 77–107. ISSN: 1386-4564. DOI: 10.1007/s10791-007-9039-3.

[7] Thomas M. Deserno, Sameer Antani und Rodney Long. „Ontology of gaps in content-based image retrieval“. In: *Journal of digital imaging* 22.2 (2009), S. 202–215. DOI: 10.1007/s10278-007-9092-x.

[8] *Downloads Whitepaper*. 15.08.2018. URL: https://www.dotsource.de/e-commerce-whitepaper-downloads (besucht am 22. 05. 2019).

[9] *eBay Powers Searching and Shopping with Images on Mobile Devices - eBay Inc.* 2018. URL: https://www.ebayinc.com/stories/press-room/uk/ebay-powers-searching-and-shopping-with-images-on-mobile-devices/?utm_source=301Redirect&utm_medium=301Redirect&utm_campaign=301Redirect (besucht am 22. 05. 2019).

[10] Peter Enser. „The evolution of visual information retrieval“. In: *Journal of Information Science* 34.4 (2008), S. 531–546. ISSN: 0165-5515. DOI: 10.1177/0165551508091013.

[11] *Find related images with reverse image search - Android - Google Search Help*. URL: https://support.google.com/websearch/answer/1325808?co=GENIE.Platform%3DAndroid&hl=en (besucht am 22. 05. 2019).

[12] *Fire-cbir by deselaers*. 5.04.2015. URL: http://deselaers.github.io/fire-cbir/ (besucht am 22. 05. 2019).

[13] *Flow Powered by Amazon - Apps on Google Play*. URL: https://play.google.com/store/apps/details?id=com.a9.flow&hl=en (besucht am 22. 05. 2019).

[14] Amarnath Gupta und Ramesh Jain. „Visual information retrieval“. In: *Communications of the ACM* 40.5 (1997), S. 70–79. ISSN: 00010782. DOI: 10.1145/253769.253798.

[15] *imgSeek Github Repository*. URL: https://github.com/ricardocabral/iskdaemon (besucht am 22. 05. 2019).

[16] E. Kasutani und A. Yamada. „The MPEG-7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval“. In: *2001 International Conference on Image Processing*. Piscataway: I E E E, Aug. 2001, S. 674–677. ISBN: 0-7803-6725-1. DOI: 10.1109/ICIP.2001.959135.

[17] Michael S. Lew, Hrsg. *Principles of visual information retrieval*. Advances in pattern recognition. London: Springer, 2001. ISBN: 1852333812.

[18] Ziwei Liu u. a. „DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations“. In: *29th IEEE Conference on Computer Vision and Pattern Recognition*. Piscataway, NJ: IEEE, 2016, S. 1096–1104. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.124.

[19] Mathias Lux, Christoph Kofler und Oge Marques. „A classification scheme for user intentions in image search“. In: *CHI '10 Extended Abstracts on Human Factors in Computing Systems*. Hrsg. von Elizabeth Mynatt. New York, NY: ACM, 2010, S. 3913. ISBN: 9781605589305. DOI: 10.1145/1753846.1754078.

[20] Mathias Lux und Oge Marques. „Visual Information Retrieval using Java and LIRE“. In: *Synthesis Lectures on Information Concepts, Retrieval, and Services* 5.1 (2013), S. 1–112. ISSN: 1947-945X. DOI: 10.2200/S00468ED1V01Y201301ICR025.

[21] Mathias Lux u. a. „LireSolr“. In: *ICMR'17*. Hrsg. von Bogdan Ionescu u. a. New York, NY, USA: ACM Association for Computing Machinery, 2017, S. 466–469. ISBN: 9781450347013. DOI: 10.1145/3078971.3079014.

[22] Oge Marques. „Visual Information Retrieval: The State of the Art“. In: *IT Professional* 18.4 (2016), S. 7–9. ISSN: 1520-9202. DOI: 10.1109/MITP.2016.70.

[23] Oge Marques und Borivoje Furht. *Content-based image and video retrieval*. Bd. 21. Multimedia systems and applications series. Boston und London: Kluwer Academic Publishers, 2002. ISBN: 9781402070044.

[24] Robert C. Martin und Michael C. Feathers. *Clean Code: Refactoring, Patterns, Testen und Techniken für sauberen Code*. 1. Aufl. Heidelberg: mitp, 2009. ISBN: 9783826655487.

[25] Paolo Napoletano. „Visual descriptors for content-based retrieval of remote-sensing images“. In: *International Journal of Remote Sensing* 39.5 (2018), S. 1343–1376. ISSN: 0143-1161. DOI: 10.1080/01431161.2017.1399472.

[26] *Product Image Visual Search in SAP Commerce Cloud / Hybris Commerce*. URL: https://hybrismart.com/2018/08/26/product-image-visual-search-in-sap-commerce-cloud-hybris-commerce/ (besucht am 22. 05. 2019).

[27] *Representative color selection 2 - Color layout descriptor (Bildquelle)*. 14.04.2019. URL: https://upload.wikimedia.org/wikipedia/commons/f/f2/Representative_color_selection_2.jpg (besucht am 22. 05. 2019).

[28] *Result Grouping | Apache Solr Reference Guide 7.5*. 18.09.2018. URL: https://lucene.apache.org/solr/guide/7_5/result-grouping.html (besucht am 22. 05. 2019).

[29] *SAP Commerce - SAP Help Portal*. 15.05.2019. URL: https://help.sap.com/viewer/product/SAP_COMMERCE/6.7.0.0/en-US (besucht am 22. 05. 2019).

[30] *Shop the Look with Deep Learning – Zalando Tech Blog*. 29.04.2019. URL: https://jobs.zalando.com/tech/blog/shop-look-deep-learning/?gh_src=4n3gxh1 (besucht am 22. 05. 2019).

[31] A.W.M. Smeulders u. a. „Content-based image retrieval at the end of the early years“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.12 (2000), S. 1349–1380. ISSN: 01628828. DOI: 10.1109/34.895972.

[32] *Solr Tutorial | Apache Solr Reference Guide 7.5*. 18.09.2018. URL: https://lucene.apache.org/solr/guide/7_5/solr-tutorial.html (besucht am 22. 05. 2019).

[33] *Vision & Mission*. 16.08.2018. URL: https://www.dotsource.de/agentur/vision-mission/ (besucht am 22. 05. 2019).

[34] *Windsurf Home Page*. 3.06.2014. URL: http://www-db.deis.unibo.it/Windsurf/ (besucht am 22. 05. 2019).

[35] Yanhao Zhang u. a. „Visual Search at Alibaba“. In: *KDD2018*. Hrsg. von Yike Guo und Faisal Farooq. New York, NY: Association for Computing Machinery Inc. (ACM), 2018, S. 993–1001. ISBN: 9781450355520. DOI: 10.1145/3219819.3219820.

Abbildungsverzeichnis

Abkürzungsverzeichnis

AC	Auto Color Correlogram
B2B	Business to Business
B2C	Business to Customer
CBVIR	Content-based Visual Information Retrieval
CL	MPEG-7 Color Layout
CSV	Comma-seperated Values
DCT	Diskrete Cosinus Transformation
EH	MPEG-7 Edge Histogram
FCTH	Fuzzy Color and Texture Histogram
HSB	Hue Saturation Brightness
HSI	Hue Saturation Intensity
HSV	Hue Saturation Value
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IR	Information Retrieval
JSON	JavaScript Object Notation
JSP	Java Server Pages
LIRE	Lucene Image Retrieval
MAP	Mean Average Precision
MVC	Model View Controller
PHOG	Pyramid Histogram Of Oriented Gradients
RGB	Rot Grün Blau
SIFT	Scale-Invariant Feature Transform
TXT	Text
URL	Uniform Resource Locator
VIR	Visual Information Retrieval
XML	Extensible Markup Language