Hochschule Reutlingen
Reutlingen University

Automatisierte Zerlegung von Vorlesungsaufzeichnungen zur Realisierung eines Empfehlungssystems für die Lerneinheiten der Online-Lehre

Master-Thesis
im Studiengang Digital Business Engineering
Herman-Hollerith-Zentrum, Fakultät Informatik
Hochschule Reutlingen
Alteburgstraße 150
72762 Reutlingen

Eingereicht von: Arnaud Zobel Kamlo Ngako
Matrikel Nr. 766608
E-Mail: Arnaud_Zobel.Kamlo_ngako
@Student.Reutlingen-University.DE

Betreuer/Prüfer: Prof. Dr. Christian Decker
Prof. Dr. Jürgen Münch

Tag der Abgabe: 31. Dezember 2021

Inhaltsverzeichnis

Nomenklatur

ASR	Automatic Speech Recognition
C.a.	Circa
CBR	Case Based Reasoning
CNN	Convolutional Neural Network
CSV	Comma-separated Values
D.h.	Das heißt
DCT	Discrete Cosine Transform
HSV	Hue Saturation Value
LMS	Learning Management System
MAE	Mean Absolute Error
NLP	Natural Language Processing
OCR	Automatic Speech Recognition
RGB	Red Green Blue
RMSE	Root Mean Square
ROI	Region Of Interest
STFT	Short-time Fourier transform
SVD	Single Value Decomposition
Z.B.	Zum Beispiel

Abstrakt (Deutsch)

Die Anzahl der Online-Vorlesungen ist durch die Corona-Pandemie sehr stark gestiegen. Viele Hochschulen haben die klassische Präsenzlehre auf digitales Format umgestellt. Dadurch ergibt sich für den Studierenden ein zeitlicher und räumlicher Vorteil. Die Onlinelehre bringt zwar neu Impulse, erfordert jedoch mehr zeitlichen aufwand. Eine Vorlesung didaktisch für das Online-Format aufzubereiten ist Aufwendig. Außerdem wird das individuelle Lernen nicht unterstützt.

In dieser Arbeit werden Ansätze zur automatisierte Zerlegung der Vorlesungsaufzeichnungen untersucht, um passende Lerneinheiten für die Online-Lehre zu erstellen. Des Weiteren wird auf Basis der erstellten Lerneinheiten ein Empfehlungssystem für die Lerneinheiten der Online-Lehre realisiert.

Der Lösungsprozess für die automatisierte Zerlegung der Aufzeichnungen besteht wird in 4 Hauptschritten unterteilt. Im ersten Schritt werden audiovisuelle Features wie die Fragehistogramm und die akustische Salienz extrahiert. Im zweiten Schritt werden diese Features analysiert, um Szeneänderungen und interessante Szeneänderungen zu erkenne. Im dritten Schritt werden Szenen mit dem CNN-Model klassifiziert. Im letzten Schritt wird das Eingangvideo mit dem FFmpeg-Programm aufgeschnitten. Der Vorteile von dieser Lösung ist, dass sie für die Zerlegung kein Wissen über den Inhalt des Videos benötigt.

Für die Entwicklung des Empfehlungssystems, werden die klassische Lösungsschritte für ein maschinelles Lernen Problem nach CRISP verwendet. Das Empfehlungssystem basiert auf das kollaborative Filtern. Unterschiedliche Ansätze des kollaborativen Filterns werden experimentiert und der geeigneter ausgewählt. Das Empfehlungssystem wird mit der Kreuzvalidierung evaluiert. Dazu werden Datensätze generiert, die aus Benutzerinteraktionsdaten besteht.

Abstrakt (Englisch)

The number of online lectures has risen very sharply as a result of the Corona pandemic. Many universities have converted traditional face-to-face teaching to digital format. This gives students an advantage in terms of time and space. Online teaching brings new impulses, but requires more effort. Preparing a lecture didactically for the online format is time-consuming. In addition, individual learning is not supported.

In this thesis, approaches for automated segmentation of lecture recordings are investigated in order to create suitable learning units for online teaching. Furthermore, a recommendation system for the learning units of online teaching is implemented based on the created learning units.

The solution process for the automated decomposition of the recordings is divided into 4 main steps. In the first step, audiovisual features such as color histogram and acoustic saliency are extracted. In the second step, these features are analyzed to detect scene changes and interesting scene changes. In the third step, scenes are classified using the CNN model. In the last step, the input video is sliced using the FFmpeg program. The advantage of this solution is that it does not require any knowledge about the content of the video for the segmentation.

For the development of the recommender system, the classical solution steps for a machine learning problem according to CRISP are used. The recommender system is based on collaborative filtering. Different approaches of collaborative filtering are experimented and the more suitable one is selected. The recommender system is evaluated using cross validation. For this purpose, datasets are generated which consists of user interaction data.

1 Einleitung

1.1 Problemstellung

Die Anzahl der Online-Vorlesungen ist durch die Corona-Pandemie stark gestiegen [1]. Die Studierenden gewinnen dadurch eine hohe Flexibilität. Sie können sich die Lerneinheiten beliebig oft und unabhängig von Vorlesungszeiten und Hörsäle ansehen. Eine Vorlesung didaktisch für ein Online-Format aufzubereiten, wird jedoch zu einer aufwendigen Aufgabe. Die Vorlesung wird zunächst vom Professor aufgezeichnet. Dann wird die Aufzeichnung manuell aufgeschnitten und überarbeitet, bevor die daraus resultierenden Lernabschnitte in ein Learning Management System (LMS) hochgeladen werden. Diese manuellen Aufgaben werden oft nicht korrekt durchgeführt. Infolgedessen erfüllen viele Online-Vorlesungen nicht die Kriterien für das Online-Format. Die meistens weisen eine lange Übertragungsdauer auf, was die Suche nach relevanten Inhalten für Studierende schwer macht. Eine automatisierte Zerlegung der Aufzeichnungen in das Online-Format könnte diese Aufwände reduzieren.

Des Weiteren ist bei einer großen Anzahl an Lernvideos für Studierenden schwierig zu finden was sie als nächstes lernen sollen. Außerdem sind viele Online-Vorlesungen auf statische Lernmaterialien basiert, die die Vielfalt der Lernenden nicht berücksichtigt. Das LMS mit integriertem Empfehlungssystem wird dafür als Lösung angesehen [2]. Diese Systeme versuchen den Lernenden eine individuelle Bildung anzubieten, indem sie diesen auf Grundlage ihrer Ziele, Fähigkeiten und Präferenz Lernabschnitte empfehlen. Allgemeine Data-Mining-Tools, wie *Weka* und *Intelligent Miner* werden heutzutage angesetzt, um Empfehlungsprobleme zu lösen. Diese Tools sind jedoch nicht speziell für pädagogische Zwecke konzipiert und werden auch nicht dafür gepflegt [2].

1.2 Zielsetzung und Einordnung der Arbeit

Das Ziel dieser Arbeit ist es, Vorlesungsaufzeichnungen automatisiert zu zerlegen, um passende Lerneinheiten für die Online-Lehre zu erzeugen. Des Weiteren soll auf Basis

erzeugten Lerneinheiten ein Empfehlungssystem realisiert werden. Lerneinheiten sollen so klassifiziert werden, dass individuelle Lernpfade möglich sind. Individuelle Pfade sind das Resultat von Empfehlungen an einer Studierende, was er als nächstes lernen soll. Man kann es sich analog zum bekannten Amazon oder Netflix Empfehlungssystem vorstellen: *„Studierenden, die diese Lerneinheit/Abschnitt gesehen haben, haben als nächstes auch diesen/jenen gesehen.*“

Im Rahmen dieser Arbeit soll konkret auf folgende Forschungsfragen eingegangen werden:

- Welche Methoden und Techniken aus dem Bereich maschinelles Lernen eignet sich für eine audiovisuelle Szeneexploration?
- Wie kann die Interessantheit einer Szeneänderung in ein Vorlesungsvideo definiert und bewertet werden?
- Welche Tools sind für die Videosegmentierung geeignet?
- Welche Methoden und Techniken aus dem Bereich Statistik oder Maschinelles eignen sich für die Realisierung eines Empfehlungssystems für die Lerneinheiten der Online-Lehre?
- Wie können Benutzer mit unterschiedlichen Profilen für den Test von kollaboratives Filtern erzeugt werden?
- Wie lässt sich ein Empfehlungssystem evaluieren?

In Rahmen dieser Arbeit wird kein Softwareprodukt entwickelt, sondern es werden Experimente durchgeführt. Eine inhaltliche Analyse der Vorlesungsvideos ist nicht Teil dieser Arbeit. Des Weiteren basiert das Empfehlungssystem in dieser Arbeit auf das kollaborative Filtern. In dieser Arbeit werden Lerneinheiten und Lernabschnitte als Synonym verwendet.

Die vorliegende Thesis wird in zwei wissenschaftliche Forschungsgebiete eingeordnet. Aufgrund des engen Zusammenhangs mit der Videozusammenfassung ordnet sich thematisch der Prozess der automatisierten Zerlegung der Aufzeichnungen in dieses Gebiet ein. Was der Prozess des Empfehlungssystem angeht, reiht sich dieser Thesis im Forschungsfeld des *Empfehlungssystems.* Aufgrund des Bezugs zum Thema Vorlesung wird eher von *Empfehlungssystems im Kontext E-Learning* gesprochen.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in 9 Kapitel ein. Das Kapitel 1 beschäftigt sich mit den grundlegenden Überlegungen der Problemstellung sowie die Zielsetzung dieser Arbeit. Dabei werden konkrete Forschungsfragen definiert, dann diese Arbeit thematisch in entsprechende wissenschaftliche Kontexte eingeordnet. Im Kapitel 2 wird eine systematische Literaturrecherche durchgeführt. Dies beinhaltet die Methode der Literaturrecherche der Forschungsfragen und die Vorgehensweise. Im selben Kapitel werden auf aktuelle wissenschaftliche Recherchen zu den genannten Forschungsgebieten eingegangen. Dort werden die wesentlichen Konzepte der Forschungsgebiete dieser Thesis dargestellt. Das Kapitel 3 beschäftigt sich mit der theoretischen Grundlage zum Forschungsbereich Videozusammenfassung und audiovisuelle Szeneanalyse. In diesem Kapitel werden die wesentlichen Konzepte und Methoden zur Videozusammenfassung dargestellt. Im Kapitel 4 wird eine detaillierte Grundlage zu Empfehlungssystemen beschrieben. Dabei wird konkret auf das kollaborative Filtern eingegangen, die eingesetzten Methoden und Annahmen, die dabei getroffen sind und die Evaluationsmethoden. Im Kapitel 5 wird versucht die vorliegende Arbeit thematisch genauer in das Gebiet E-Learning einzuordnen. Es werden einige Annahmen und Methoden aus den vorherigen Kapiteln betrachtet und im Kontext der Online-Vorlesung adaptiert. Das Kapitel 6 beschäftigt sich mit dem Systementwurf. Dies beinhaltet die Systeminfrastruktur und -architektur. Ebenfalls wird das Lösungsonzept der hervorgehobenen Problemen in dieser Thesis vorgestellt. Das Kapitel 7 stellt die eigentliche Implementierung dar. Hierzu gehört die Vorstellung des verwendeten Frameworks, die Umsetzung der automatischen Segmentierung der Aufzeichnung und des Empfehlungssystems. In Kapitel 8 wird die Evaluation des entwickelten Ansatzes durchgeführt. Hier wird insbesondere das Leistungsverhalten und Genauigkeit geprüft. Schließlich wird im Kapitel 9 die Arbeit kurz zusammengefasst, das Ergebnis vorgestellt und ein Ausblick aufgezeigt.

2 Literaturrecherche und verwandte Arbeit

In diesem Kapitel wird eine systematische Literaturrecherche durchgeführt. Dies beinhaltet die Methode der Literaturrecherche und die Vorgehensweise. Dabei werden die wissenschaftlichen Beiträge in den Themengebieten hervorgehoben.

2.1 Methode der Literaturrecherche

Zur Bestimmung des aktuellen Stands der Technik, wird am Anfang eines wissenschaftlichen Arbeitsprozesses eine systematische Literaturrecherche durchgeführt [3]. Die Autoren beschreiben die Vorgehensweise zur Literaturrecherche in fünf Phasen und legen den Schwerpunkt in der Literaturprüfung. Im Rahmen dieser Arbeit wurde die Literaturrecherche nach diesem Vorgehen durchgeführt (Abbildung 2.1). In der ersten Phase werden Forschungsfragen festgelegt. In der zweiten Phase findet die Organisation der Analyse statt. Hier werden Literaturdatenbanken definiert. In der dritten Phase wird die eigentliche Literatursuche durchgeführt. Hierbei kommen die Methoden *Forward Search* und *Backward Search* zum Ansatz. Die vierte Phase dient zur Analyse der Literaturrecherche. Schließlich wird in der fünften Phase das Ergebnis der Recherche dokumentiert.

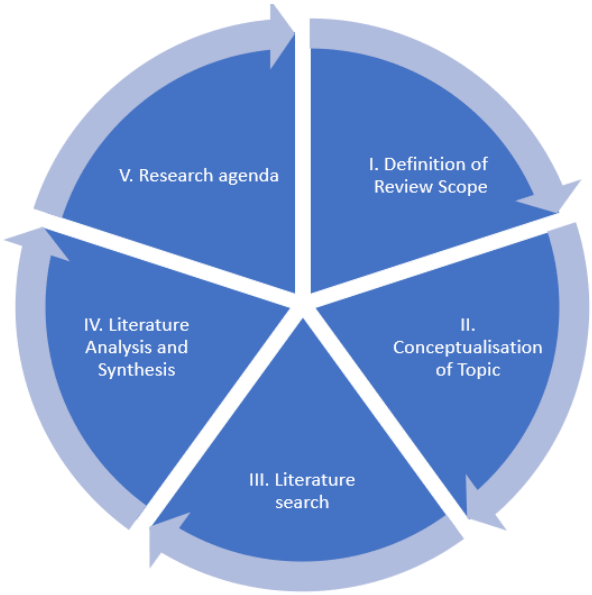


Abbildung 2.1: Methode der Literaturrecherche (Quelle: In Anlehnung an [3])

2.2 Vorgehensweise

Wie bereits erwähnt, werden in der ersten Phase der Literaturrecherche Forschungsfragen definiert. In der Einleitung wurden in jedem Gebiet drei Fragen definiert. Auf diese werden hier nicht mehr weiter-eingegangen.

Nachdem die Forschungsfragen definiert sind, werden in der zweiten Phasen der Literaturrecherche Datenbanken für die Recherche festgelegt. Da die Hochschule die Lizenz zu folgenden Online-Datenbanken hat, wurden diese für die Recherche ausgewählt.

- ScienceDirect: ist eine digitale Forschungsdatenbank mit wissenschaftlichen und medizinischen Veröffentlichungen.
- Springer Link: ist eine große digitale Bibliothek aus den Bereichen Engineering, Biomedizinisch, Sozialwissenschaft, etc.
- IEEE Xplore: ist eine digitale Forschungsdatenbank für die Suche nach Zeitschriften, Artikeln, Konferenzberichten und technischen Normen der Bereiche Informatik, Elektrotechnik und Elektronik.

Suchbegriff	Stiffing	Springer Link	IEEE Xplore	ScienceDirect
1	Lecture video segmentation	17964	70	1283
2	Video summarization/lecture	2874	48	85389
3	Scene change detection	73799	2530	57817
4	Audio-visual scene analysis	5380	128	1975
5	Lecture recommender system	8727	14	56449
7	Recommender system evaluation	20197	1455	1115897
8	video segmentation tool	26512	579	15363

Tabelle 2.1: Trefferquote der ersten Suchstrings

In der dritten Phase der Recherche findet die eigentliche Literatursuche statt. Dazu wird zunächst klargemacht, welche Suchbegriffe eine wichtige Rolle spielen. Da Forschungsfragen in der ersten Phase definiert wurden, werden davon Schlüsselwörter herangezogen. Damit startet die Suche in oben genannten Datenbanken. Bei der ersten Suche mit deutschen Schlüsselwörtern wird schnell aufgrund der niedrigen Trefferquote festgestellt, dass die Sprache für die Recherche dieser Themen nicht geeignet ist. Für die weitere Suche wird Englisch verwendet. Dafür werden englische Schlüsselwörter von Forschungsfragen abgeleitet. Diese Suche ergibt eine sehr hohe Anzahl an Treffern (Abbildung 2.1). Dies lässt sich dadurch begründen, dass viele wissenschaftliche Arbeiten in Englisch publiziert sind. Die Trefferquote in IEEE Xplore ist im Vergleich zu den anderen beiden Datenbanken deutlich geringer. Eine Erklärung dafür könnte sein, dass sich IEEE Xplore auf wissenschaftliche Publikationen beschränkt. Was Springer Link betrifft, ist die Trefferquote hingegen sehr hoch, jedoch ist der Anteil an wissenschaftliche Artikel gering. Um die relevante Literatur zu extrahieren, werden die Schlüsselwörter mit logischen Operatoren „AND“ und „OR“ verknüpft. Des Weiteren werden nachfolgende Filter verwendet:

- Zeitraum der Publikation: Letzte 5 Jahren
- Themengebiet: Artifiziell Intelligenz
- Publikationsform: Paper
- Schlüsselwort im Titel enthalten: ja

Nach Einsatz des oben beschriebenen Filters, sinkt die Trefferliste signifikant ab. In der vierten Phase wird die verbleibende Trefferliste nach Relevanz geprüft. Dies erfolgt zunächst anhand der Abstracts. Papers mit relevanten Abstracts werden vollständig weitergelesen. In der fünften Phase werden interessante Papers in die Agenda übertragen.

2.3 Ergebnis der Literaturrecherche

Die Tabelle 2.2 stellt das Ergebnis der Recherche dar. Es ist festzustellen, dass am Anfang der Suche die Trefferquote sehr hoch ist. Nach Anwendung der beschriebenen Filter reduziert sich die Trefferliste kontinuierlich. Das Ergebnis der Suche ergibt pro Datenbank und Suchstring 0 bis maximal zwei relevante Papers. Insgesamt ergibt sich für das Themengebiet *Videozusammenfassung* 11 und für das Empfehlungssysteme 7 relevante Papers. Diese Papers werden als Grundlage für die Bearbeitung der vorliegenden Thesis angesetzt. Diese wurden während der Bearbeitung dieser Thesis durch *forward* und *backward search* erweitert.

Datentank	Suchbegriff	Anzahl Treffer	Publikations-form Paper	Zeitraum 2017-2021	Themengebiet	Schlüsselwort im Titel enthalten	Relevant nach Lesen des Abstracts	Relevant
ScienceDirect	1	1283	749	40	6	0	0	1
	2	85389	55898	1430	100	10	3	1
	3	57817	4695	2257	84	14	1	1
	4	100	80	50	10	8	2	1
	5	56449	27181	1016	45	3	2	1
	7	1116176	767682	43559	439	6	4	1
	8	1010	500	100	14	0	0	0
IEEE Xplore	1	70	66	18	5	-	3	1
	2	48	39	19	10	8	2	1
	3	2530	2007	483	176	5	1	0
	4	128	98	30	9	5	0	0
	5	14	12	6	5	-	3	1
	7	1455	1224	524	111	10	5	2
	8	15	11	7	4	0	0	0
Springer Link	1	17964	13886	3003	2150	10	3	1
	2	2874	1200	792	60	30	4	2
	3	73799	20834	13187	5		2	1
	4	5380	1063	275	60	6	5	1
	5	8727	7234	2507	89	7	5	2
	6	20197	1897	501	40	2	2	0
	7	50	40	14	10	7	5	2
	8	80	20	10	8	0	0	0

Tabelle 2.2: Ergebnis der Literaturrecherche

2.4 Verwandte Arbeiten

Im Bereich Empfehlungssystem und Videozusammenfassung wurde in der letzten Dekade eine Vielzahl an Artikel publiziert. Der nachfolgende Abschnitt thematisiert aktuelle Ansätze in diesen Bereichen und erörtert, ob bereits Ansätze existieren, welche die definierte Forschungsfrage beantworten. Die Publikationen setzten dabei auf eine Vielzahl von unterschiedlichen Herangehensweisen.

Empfehlungssystem im Kontext E-learning

Dan Fu et al. präsentieren einen Ansatz zur Analyse von Vorlesungsvideos basierend auf dem Inhalt. Die Autoren verwenden Schlüsselbildextraktion um das Input-Video zu segmentieren. Die Frames werden in bestimmten Zeitintervallen gelesen. Mithilfe der OCR-Technologie, die auf Schlüsselbilder angewendet wird, werden Schlüsselwörter extrahiert. Die Autoren verwenden ASR-Techniken, um textuelle Features aus der Audiospur zu extrahieren. Auf Grundlage der gefundenen Schlüsselwörter, werden Weblinks, Bildlinks und YouTube-Links bereitgestellt. Benutzer können über die bereitgestellten Links auf die entsprechenden Videos zugreifen und eine Bewertung für das angesehene Video abgeben. Auf Basis der Bewertung werden Empfehlungen generiert. Das Empfehlungssystem wird mithilfe der Pearson-Korrelation und Cosinus-Ähnlichkeitsmaß umgesetzt [4].

Fatiha Bousbahi und Henda Chorfi schlagen ein MOCC-Empfehlungssystem vor, welches mit dem CBR-Ansatz und Informationssuchtechniken realisiert ist. Auf Basis der Lernprofile, Bedürfnisse und des Wissensstandes der Lernenden, erstellt das System die passende Empfehlung. CBR ist eine Art inhaltsbasiertes Empfehlungssystem, welches als Basis für die Beschreibung des Empfehlungsobjekts verwendet wird [5].

Harshit Jain und Anika schlagen eine Methode zur Entwicklung eines MOOC-Empfehlungssystem vor, unter Anwendung von Data Mining Techniken wie Random Forest, Classification Tree und K-Nearest Neighbors. Die Autoren klassifizieren zunächst die Benutzer anhand ihrer Aktivitätsprotokolle in zwei Kategorien: aktive und passive Lernende. MOOCs sind ein neuer Trend. Damit können Hochschullehrer ihre Vorlesungsvideos kostenfrei oder kostengünstig einer großen Anzahl an Studierenden weltweit im Internet zugänglich machen [6]. Nach Anwendung von Data Mining Ansätzen wird die Kursempfehlung separat für jede Kategorie erstellt [7].

Cristóbal Romero et al. wenden das Data Mining Tool Weka auf das LMS Moodle an, um Empfehlungen für Lehrende und Lernende zu erstellen. Mit diesem Ansatz werden

Aktivitätsprotokolle der Benutzer heruntergeladen und extern mit dem Open Source Tool Weka analysiert. Mithilfe der Visualisierung- und Klassifikationsfunktion könnten interessante Informationen identifiziert werden [8].

Zameer Gulzar et al. schlagen eine Hybride Methodologie zusammen mit Ontologie vor, um personalisierte Kursempfehlungen basierend auf Benutzerpräferenzen abzugeben [9]. Die Ontologie ist eine Möglichkeit, die Eigenschaften von Objekten und ihre Beziehungen zueinander darzustellen, indem eine Reihe von Konzepten und Kategorien definiert wird, die das Objekt repräsentieren.

Videozusammenfassung

In [10] wird ein Ansatz zur Segmentierung von Lernvideo und Verwendung der natürlichen Sprachverarbeitung (NLP) gezeigt, um wichtige linguistische Features aus dem Video zu extrahieren. Die Autoren nutzen die visuellen, Audio- und textuellen Features, um zeitliche Feature-Vektoren zur erstellen, welche der verbesserten Segmentierung dienen. Anschließend wird NLP-Kosinusähnlichkeit auf die Cluster angewendet, um verschiedene Themen in Videos zu identifizieren. Für die textuelle und Audio Feature-Extraktion setzten die Autoren ASR und OCR ein.

In seinem Beitrag, schlägt [11] eine Methode für die Zusammenfassung von Whiteboard-Video vor, indem Feature-Repräsentationen von erkannten handschriftlichen Inhaltsregionen extrahiert werden, um die eindeutigen Inhalte zu bestimmen. Dabei wird das Histogramm von Gradienten verwendet, um die erkannten Regionen darzustellen.

In [12] wird eine Methode zur Segmentierung des Videos auf der Grundlage des zuvor notierten Beginn und Ende jedes Themas beschrieben. Anschießend werden die einzelnen Themen auf Basis der Audio-Features zusammengefasst.

In [13]wird eine Methode zur Extraktion von Inhalten eines Vorlesungsvideos unter Verwendung der Sprecher-Aktionen gezeigt. Jedes Vorlesungsvideo wird in kleine zeitliche Einheiten unterteilt, die als Aktionssegmente bezeichnet werden. Zur Analyse der Pause des Sprechers werden Körper- und Handskelett-Daten extrahiert, um daraus bewegungsbasierte Merkmale für jedes Segment zu berechnen. Danach wird die dominante Sprecher-Aktion jedes Segmente mit Random Forests und dem bewegungsbasierten Merkmal klassifiziert.

3 Videozusammenfassung und Audiovisuelle Szeneexploration

Dieses Kapitel beschäftigt sich mit der theoretischen Grundlage zum Forschungsbereich Videozusammenfassung und audiovisuelle Szeneanalyse. Hier werden die wesentlichen Konzepte und Methoden dieser Bereiche dargestellt.

3.1 Videozusammenfassung

Bevor das Empfehlungssystem für Lerneinheiten der Online-Lehre realisiert werden kann, werden zunächst aus Vorlesungsaufzeichnungen passende Lerneinheiten erzeugt. Dazu werden einige Videozusammenfassungstechniken angewendet. In diesem Abschnitt werden gängige Methoden aus der Literatur erfasst.

Ein Video ist eine Sammlung von Bildern in einer festgelegten Reihenfolge, die mit sehr hoher Geschwindigkeit ablaufen [14]. Die Videozusammenfassung ist eine Technik, mit der eine kurze Zusammenfassung des Inhalts eines längeren Videos erstellt wird, indem die informativsten oder interessantesten Bereiche des Videos für potenzielle Nutzer ausgewählt werden. Der Output der Videozusammenfassung besteht in der Regel aus einer Reihe von Schlüsselbildern(Frames) oder einem kurzen Video, das vom originalen Video extrahiert ist. Die Videozusammenfassung zielt darauf ab, dass Durchsuchen einer großen Sammlung von Videodaten zu beschleunigen und eine effiziente Darstellung des Videoinhalts zu erreichen [15]. Im allgemein, werden drei Schritte während des Videozusammenfassungsprozesses identifiziert. Im ersten Schritt wird das Video analysiert, um audiovisuelle Informationen herauszufinden. Im zweiten Schritt werden Bilder ausgewählt, die den Inhalt des Videos repräsentieren. Zum Schluss wird das Ergebnis organisiert [16]. In der Literatur wird zwischen zwei Arten von Videozusammenfassung unterschieden: die statische Zusammenfassung, auch *key frame summarization* genannt und der dynamischen Zusammenfassung, auch *video skimming summarization* genannt [16, 14].

Bei der statischen Zusammenfassung besteht das Ergebnis der Zusammenfassung aus einzelnen Schlüsselbildern mit höchster Priorität. Diese werden im ersten Schritt extrahiert, indem Bilder gleichmäßig übersprungen oder zufällig ausgewählt werden. Im nächsten Schritt können die Schlüsselbilder durch kollaborative Darstellung der benachbarten Bilder bestimmt werden. Mit dieser Methode werden Bilder mit minimalem Rekonstruktionsfehler ausgewählt [16]. Zur Bestimmung der Schlüsselbilder wird am häufigsten die Farbbasierte Technik eingesetzt. Dabei wird das Farbhistogramm aus RGB oder HSV der einzelnen Bilder berechnet. Das Ergebnis ist die Farbverteilung der Bilder anhand dessen zwei aufeinander folgende Bilder verglichen werden können. Dabei wird angenommen, dass Bilder eines Videos in derselben Aufnahme ähnliche Werte haben[16].

Das Ergebnis der dynamischen Zusammenfassung ist ein kurzes Video, in dem die interessantesten Szenen aus dem Eingangsvideo in Form eines Abstrakt wiedergeben werden. Hierzu werden Techniken wie das SVD, Bewegungsmodell und die semantische Analyse angewendet [16, 14]. Die Tabelle 3.1 stellt den Unterschied zwischen statischer und dynamischer Zusammenfassung dar.

Key frame based summarization	Video skimming
Produce set of key frames as summary	Produce small video as summary
Contains no motion information	Contains motion information
Limited user viewing experience	High User viewing experience
No time bounds and synchronization needed	Time restricted and synchronization is necessary
Only video frames considered	Video, audio and text data are considered
Helps in video indexing and searching	Helps in video indexing and searching but performance is less then key frame-based summarization

Tabelle 3.1: Unterschied zwischen statischer und dynamischer Zusammenfassung (Quelle: [16])

3.2 Audiovisuelle Szeneexploration

Der Vorteil einer Videotechnik besteht darin, dass sie sowohl Audio als auch visuelle Daten erlaubt, die während einer Szeneexploration untersucht werden können. Bei der audiovisuellen Exploration ist die Salienz ein wichtiger Faktor für die Bestimmung von wichtigen Ereignissen in dem Video. In der Literatur wird zwischen akustischer Salienz, welche ein hörbares Ereignis darstellt, und der visueller Salienz, welche den sichtbaren Anteil beschreibt, unterschieden. Durch Salienz wird die Aufmerksamkeit erzeugt und

den Fokus auf das Ereignis gelenkt. Die akustische Salienz kann z.B. durch eine starke Änderung im Audio-Spektrum festgestellt werden. Mit diesem Faktor können Ereignisse wie das Fallen eines Objekts oder Konversationen in Videos dargestellt werden. Die akustische Salienz wird auf Basis der Kurzzeit-Fouriertransformation (STFT) bestimmt. Bei visuelle Salienz wird eine Salienzkarte generiert, indem an jeder Bildstelle die Merkmaldifferenzen berechnet, normiert und zusammenaddiert werden. Die visuelle Salienz kann verwendet werden, um die spontane Rotation der Aufmerksamkeit und des Blicks zu erklären. Zur Bestimmung der visuellen Salienz wird die diskrete Cosisnus-Transformation (DCT) auf Grauwert-Bildern anwendet[17, 18]. Außer Salienz, werden typischerweise Merkmale wie Farbe, Schattierung, Form, Bewegung, Textur und Kontext im Prozess der audiovisuellen Analyse genutzt. In vielen Fällen können Objekte erfolgreich anhand der Form erkannt werden.

4 Empfehlungssysteme

In diesem Kapitel werden theoretische Grundlagen zu Empfehlungssysteme detailliert beschrieben. Dies beinhaltet die Definition, die wesentliche Typen, die eingesetzte Methoden und Annahmen und die Evaluationsmethoden. Speziell wird auf das kollaborative Filtern eingegangen, welches ein Hauptbestandteil dieser Arbeit ist.

Das Empfehlungssystem hat seinen Ursprung in Forschungsfelder des Data Mining, Information Retrieval, Statistik und maschinelles Lernen [19]. Es war ursprünglich als System definiert, indem Menschen Empfehlungen als Input eingeben, welche das System aggregiert und an geeigneten Empfängern weiterleitet. Heutzutage hat diesen Begriff eine breitere Konnotation. Es wird als ein System beschrieben, das den Effekt hat, Benutzer auf eine personalisierte Art und Weise bei der Auswahl von nützlichen Objekte aus einer großen Menge zu unterstützen [20]. Empfehlungssysteme beeinflussen die Entscheidungen von Menschen im alltäglichen Leben, ohne dass sie ein Bewusstsein darstellen [21]. Z.B. beim Kauf eines Produkts beim Online-Händler Amazon.com, eine der bekanntesten Applikation mit integrierte Empfehlungssystem, stehen die Benutzer vor enorm Auswahl an potenziell interessanten Produkten[21]. Das im Shop integrierte Empfehlungssystem spielt hier die Rolle des Assistenzsystems und schlägt dem Benutzer vor, welche Artikel interessant seien könnten. Man kann sich hier an dem bekannten „Amazone-Spruch“ erinnert: „*Kunden, die diesen Artikel gekauft haben, haben auch diesen gekauft*“. Empfehlungssysteme werden heute in viele Bereiche eingesetzt. Die Abbildung 4.1 stellt eine Übersicht über der Realwelt Empfehlungssysteme.

Das Empfehlungsproblem wird oft entweder als *prediction Version of problem* oder *ranking version of problem* formuliert [22].

- Prediction version of problem

  In diesem Ansatz wird die Bewertung eines Benutzers vorhergesagt, mit der Annahme, dass Bewertungsverhalten vorhanden sind, anhand deren die Präferenzen des Benutzers für ein Objekt ermittelt werden kann. Dieses Problem lässt sich auch als „*Problem der Matrixvervollständigung*“ beschreiben, da es eine ursprünglich unvollständige Matrix n X m existiert, deren fehlende Werte anhand Data-Mining

Empfehlungssysteme	Empfohlene Produkte
Amazon.com	Bücher oder andere Produkte
Netflix	DVDs, Videostreaming
Jester	Witze
GroupLens	Nachrichten
MovieLens	Filmen
ast.fm	Musik
Google News	Nachrichten
Google Search	Werbeanzeigen
Facebook	Freunde, Werbeanzeige
Pandora	Musik
YouTube	Online Videos
Tripadvisor	Reiseprodukte
IMDb	Filmen

Tabelle 4.1: Beispiele Empfehlungssysteme aus dem Alltag (Quelle:In Anlehnung an [22])

Algorithmen vorhergesagt werden sollen.

- Ranking version of problem

  In der Praxis ist es nicht notwendig die Bewertung eines Benutzers vorherzusagen, sondern die Top-N Objekte zu finden, die diesem interessieren könnten. Dieser Ansatz, noch als *Top-N Empfehlungsproblem* bezeichnet, ist die weitverbreitete Methode [22].

Das Empfehlungsproblem lässt sich wie folgt formal beschreiben [22]:

Sei U die Menge aller Benutzer im System, I die Menge aller Objekte im System, R eine vorher festgelegte Menge mit reeller Werte zur Beschreibung der Nutzbarkeit. Sei f_u die Funktion zur Ermittlung der Nutzbarkeit eines Objekts i für einen Benutzer u (Gleichung 4.1).

$$f_u U * I -> R \tag{4.1}$$

Um dem Benutzer $u \in U$ Objekte zu empfehlen, werden Objekte $i \in I$ gesucht, welche die Nützlichkeit für den Benutzer u maximieren (Gleichung 4.2).

$$\forall u \in U i_{u=} argmax R(i, u) \tag{4.2}$$

In [22] werden vier Ziele für Empfehlungssysteme beschrieben:

- **Relevance**: es soll Objekte empfohlen werden, die dem Nutzer relevant sind.
- **Novelty**: empfohlene Objekte sollen dem Benutzer zuvor unbekannt sein.
- **Serendipity**: empfohlene Objekt ist von Nutzer unerwartet.
- **Increasing recommendation diversity**: empfohlene Objekte sollen vielfältig sein damit die Chance groß ist, dass dem Benutzer einige interessieren.

Aus unterschiedliche Anwendungskontext ergibt sich unterschiedliche Ansätze von Empfehlungssysteme [22]. In folgenden Abschnitten werden diese eingegangen.

4.1 Kollaboratives Filtern

Das kollaborative Filtern (*collaborative filtering*) ist eine der erfolgreichsten und weit verbreitetsten Technologien für das Empfehlungssystem[23]. Diese Erfolg lässt sich dadurch begründen, dass diese Technologie kein Wissen über das Einsatzgebiet erfordert. Das kollaborative Filtern wird in Memory-basiert (memory-based) und Model-basiert (model-based) unterschieden [20]. Beim Memory-basierten Ansatz, erfolgt die Vorhersage der Bewertung eines Objekts durch einen Benutzer anhand der historischen Bewertungsverhalten anderer Benutzer. Beim Model-basierten im Gegensatz, erfolgt diese anhand maschinelles Lernen Verfahren, die versuchen in Bewertungen von verschiedenen Benutzer Muster zu finden und daraus ein Modell zu lernen, auf Basis deren neuen Objekten vorgeschlagen werden.

4.1.1 Memory-basiertes kollaboratives Filtern

Das Memory-basierte kollaborative Filtern wird in Benutzer-basiert (user-based) und Objekt-basiert (item-based) unterschieden[24].

4.1.1.1 Objekt-basiertes kollaboratives Filtern

Dieses Verfahren verwendet die Ähnlichkeit von Objekten basierend auf Benutzerbewertungsverhalten, um die Empfehlung zu erstellen. Unterschiedliche Objekte die von unterschiedliche Benutzer die ähnliche Bewertung bekommen, werden als gleich angesehen unter der Annahme, dass Benutzer für die gleiche Objekte dieselben Präferenz haben. Die Abbildung 4.1 beschreibt der Ablauf des Algorithmus.

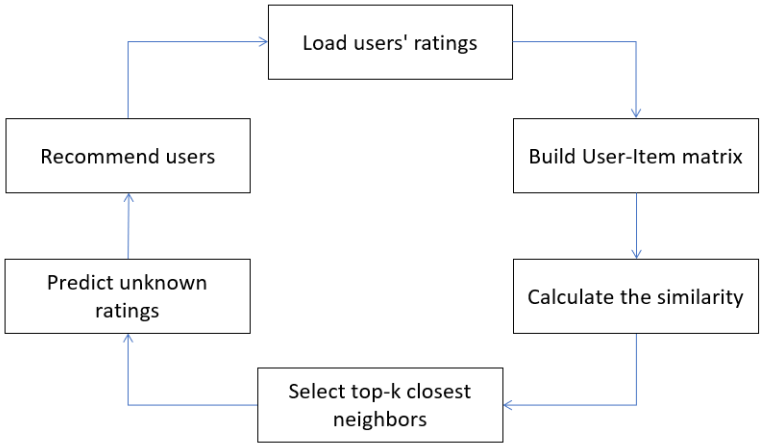


Abbildung 4.1: Ablauf des Algorithmus für das Kollaboratives Filtern (In Anlehnung an [24])

Im ersten Schritt des Algorithmus werden alle bewertete Objekte I_u eines Benutzers herangezogen. Im zweiten Schritt wird auf Basis dieser Daten die Zuordnungsmatrix Benutzer-Objekte erstellt (Abbildung 4.2). In dieser Matrix repräsentieren die Zeilen $u_1 ... u_n$ die Benutzer ($u \in U$) und die Spalten $i_1 ... i_m$ die Objekte ($i \in I$). In Zellen sind die einzelne Benutzerbewertungen ($R_{(u,i)}$) für Objekte. Unbekannte bzw. unbewertete Objekte sind mit Symbol $\oslash$ gekennzeichnet.

	i_1	i_2	i_3	...	i_j	...	i_m
u_1			R	$\oslash$	R		
u_2			R	$\oslash$	$\oslash$		
u_3			R	$\oslash$	R		
...			...	...	...		
u_j			...	...	...		
...			...	...	...		
u_n			R	$\oslash$	R		

Abbildung 4.2: User-Item-Matrix

Im dritten Schritt des Algorithmus wird die Ähnlichkeitsberechnung zu einem Zielobjekt $i_u \in I$ ermittelt. Dabei kommt häufig die Kosinusähnlichkeit als Metrik zum Einsatz (Gleichung 4.3) [25].

$$sim(i,j) = cos(i,j) = \frac{i.j}{\parallel i \parallel \parallel j \parallel} = \frac{\sum_{u \in U} r_{u,i}.r_{u,j}}{\sqrt{\sum_{u \in U} r_{u,i}^2}\sqrt{\sum_{u \in U} r_{u,j}^2}} \tag{4.3}$$

Wie in der vorherige Gleichung dargestellt, die Messung der Ähnlichkeit zwischen zwei Objekte i und j besteht darin beide Objekte als Vektors darzustellen und die Kosinusähnlichkeit der Vektoren zu berechnen, wobei $r_{u,i}$ repräsentiert die Bewertung des Benutzer u für ein Objekt i.

Im vierten Schritt des Algorithmus wird die Bewertung $R_{(u,i)}$ des Benutzers vorhergesagt. Dazu werden die k nächste Nachbarn aus der vorherigen Berechnung herangezogen. Als Nachbarn werden Objekte bezeichnet, die eine hohe Ähnlichkeit zu dem Zielobjekt Objekt i aufweisen. Die folgende Formel wird zu Berechnung der Vorhersage verwendet. Dabei enthält die Menge N_k die nächste k ähnlichste Objekte zu i und $r_{u,j}$ ihre Bewertung.

$$r_{u,i} = \frac{\sum_{j \in N_k} sim(i,j).r_{u,j}}{\sum_{j \in N_k} sim(i,j)} \tag{4.4}$$

Im letzten Schritt des Algorithmus schlägt das System die Objekte mit größten geschätzte Bewertungen vor [19]

4.1.1.2 Benutzer-basiertes kollaboratives Filtern

Im Gegensatz zum Objekt-basiertes Verfahren, ist das Ziel des Benutzer-basierten kollaboratives Filterns, Benutzer zu finden, die ähnliche Bewertungsverhalten wie der aktive Benutzer haben, dann ihre Bewertungen für die Vorhersage der Bewertung des aktiven Benutzers verwenden. Die Idee beruht sich auf die Annahme, dass Benutzer mit ähnlicher Bewertung unterschiedlicher Objekte dieselben Interessen hätten[25]. Um ein aktiver Benutzer zu empfehlen, findet das System die k ähnliche Nutzer im System, um die Objekte dem aktiven Benutzer mithilfe ihrer Bewertungen zu empfehlen[25].

Der im vorherigen Abschnitt beschriebene Ablaufalgorithmus gilt auch für die Benutzerbasierte Methode mit dem Unterschied, dass die Berechnung der Ähnlichkeit (Schritt 3) auf Basis der Benutzerähnlichkeit stattfindet. Für diese Berechnung, wird häufig Pearson-Korrelation als Metrik verwendet (Gleichung 4.5).

$$sim(x,y) = \frac{\sum i_{i=1}^{n}(x_i - \overline{x})(yi - \overline{y})}{\sum i_{i=1}^{n}\sqrt{(x_i - \overline{x})^2} \sum i_{i=1}^{n}\sqrt{(y_i - \overline{y})^2}} \tag{4.5}$$

Nach Ermittlung der k nächste Nachbarn, wird die Nutzbarkeit berechnet (Gleichung 4.6)

$$r(u,i) = \overline{r}_u + \frac{\sum_{v \in N_k} sim(u,v)(r_{v,i}\overline{r}_v)}{\sum_{v \in N_k} sim(u,v)} \tag{4.6}$$

4.1.2 Model-basiertes kollaboratives Filtern

Wie bereits erwähnt wird bei diesem Ansatz maschinelles Lernen Models verwendet, um Muster in Bewertungen von Benutzer zu lernen, um auf Basis deren neue Objekte vorzuschlagen. In diesem Bereich hat sich Latent Faktor Models mit der Matrix-Faktorisierung durch eine sehr hohe Genauigkeit der Empfehlung etabliert. In seiner Grundform, charakterisiert die Matrix-Faktorisierung Benutzer und Objekte durch einen Vektor von Einflussfaktoren, die aus Bewertungsinformationen abgeleitet werden [26]. Die Matrix-Faktorisierung im Kontext von Empfehlungssystem oft Singular Value Decomposition (SVD) genannt, wird formal als beschrieben als, eine Matrix A von Bewertungen, welche in ein Produkt von Matrizen $A = U \sum V^T$ zerlegt wird, wobei $\sum V^T$ die latente Faktoren für die Objekte darstellen und U die Benutzer (Abbildung 4.3) [27]. Die Fragezeichen in dieser Abbildung repräsentieren die zu berechnete Features, um die ursprüngliche Matrix A zu wiederherstellen.

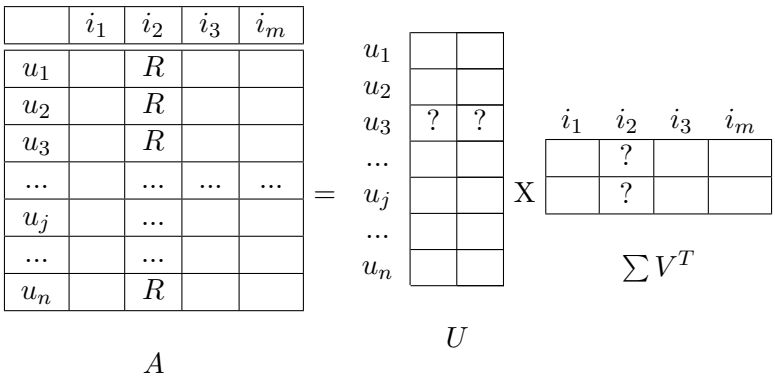


Abbildung 4.3: Matrix-Faktorisierung

4.1.3 Nachteile von Kollaboratives Filtern

- **New Item Problem**

  Neue Objekte können bei der Empfehlung nicht berücksichtigt werden, da das System dazu keine Bewertungsdaten hat. Eine mögliche Lösung hierfür wäre die Bereitstellung von Empfehlungen auf Basis systemübergreifende Informationen[21].

- **New User Problem**

  Ähnlich wie das New Item Problem hat das System keine Information über neue Benutzer im System. Diese müssen erst eine Anzahl an Objekten bewerten, um vom System berücksichtigt zu werden. Eine mögliche Lösung hier wäre die Erstellung eines initialen Benutzerprofils [21].

- **Sparsity**

  Unter diesem Begriff wird eine geringe Abdeckung von Bewertung eines Benutzers zu Objekten verstanden. Dieses Problem tritt oft auf, wenn es eine große Anzahl an Benutzer oder Objekt im System gibt. Eine Lösung dafür wäre bei der Berechnung von Ähnlichkeit von Benutzer nicht nur die Benutzer-Objekt-Matrix zu betrachten, sondern auch Benutzer Abhängige Faktoren wie demografische Daten einbeziehen[21].

4.1.4 Implizites und explizites Feedback

Die Information, die das Empfehlungssystem von den Benutzern zur Berechnung der Nutzbarkeit benötigt, sind Feedback genannt. Es gibt zwei Arten von Feedbacks: die explizite und die implizite. Bei explizit Feedback gibt den Benutzer explizit eine Bewertung in einer Skala, welche die Höhe seines Interesses/seiner Zufriedenheit an einem Objekt entspricht [22]. Bei einer 5-Punkte-Skala, kann der Wert aus der Menge $\{-2, -1, 0, 1, 2\}$ gezogen werden, indem der Wert -2 eine starke Unzufriedenheit bedeutet und 2 eine starke Zufriedenheit. Die Anzahl der Bewertungspunkte kann je nach System variieren. Häufig werden 5-Punkte, 7-Punkte und 10-Punkte verwendet.

Bei implizitem Feedback hingegen, werden Benutzerpräferenzen aus Aktivitäten der Benutzer abgeleitet. In E-Commerce z.B. der Kaufverhalten von Kunden kann als implizite Bewertung verwendet werden. Wenn der Kunde ein Produkt kauft, ist es wahrscheinlich, dass Interesse an das Produkt hat. Im Bereich Streaming, wenn ein Nutzer einen Film anschaut, ist es wahrscheinlich, dass er daran interessiert ist. Aber die Tatsache, dass ein Kunde ein Produkt aus einer großen Menge nicht kauft, bedeutet nicht unbedingt, dass er keine Präferenz dafür hat. Es besteht die möglich, dass er von deren Existenz nicht bewusst ist [22]..

4.2 Andere Empfehlungssysteme

Das inhaltsbasierte Filtern (Content-based filtering) ist andere Art des Empfehlungssystems, welche im Gegensatz zu kollaboratives Filtern, die Beschreibungen von Objekten nutzt, um die Empfehlung zu erstellen. Die Annahme hierbei ist dass, ein Nutzer, der positiv auf Objekte mit bestimmten Eigenschaften reagiert hat, kann sich auch für ähnliche Objekte interessierteren, die mit denselben Eigenschaften beschrieben sind [22].

Das hybride Filtern ist auch eine andere Art des Empfehlungssystems, die kollaborative und inhaltsbasierte Methoden kombiniert. Es ermöglicht bestimmte Einschränkungen dieser beide Methoden durch die Kombinierung umzugehen. In [19] werden verschiedene Möglichkeiten dieser Kombinierung beschrieben:

- **Combining Separate Recommenders**: bei diesem Ansatz werden kollaborative und inhaltsbasierte Methoden unabhängig voneinander implementiert und die Ergebnisse kombiniert.
- **Adding Content-Based Characteristics to Collaborative Models**: bei diesem Ansatz, basiert das Empfehlungssysteme auf die kollaborative Techniken und verwendet die inhaltsbasierte Methode sofern keine Information für kollaboratives Verfahren vorliegen.
- **Adding Collaborative Characteristics to Content-Based Models**: bei diesem Ansatz werden kollaborative Methoden in ein inhaltsbasiertes Verfahren integriert. Falls keine Informationen für inhaltsbasierte Verfahren existieren, kann die Empfehlung auf Grundlage der kollaborative Methode herleitet werden.
- **Developing a Single Unifying Recommendation Model**: Bei diesem Verfahren wird ein Modell entwickelt, indem Bewertungen und Objektattribute kombiniert werden.

4.3 Evaluation der Empfehlungssysteme

Die Evaluierung von Empfehlungssysteme und ihre Algorithmen wird zu einer herausfordernde Aufgabe[23]. Ein Algorithmus kann für ein Datensatz gut und ein anderer schlecht funktionieren. Außerdem wird die Empfehlung für unterschiedlich Zwecke verwendet. Für Systeme wo die Empfehlung zur Unterstützung von Entscheidungen eingesetzt werden, hatten Forscher in der Vergangenheit die Genauigkeit als Metrik verwendet, um die Qualität des Empfehlungssystem zu messen. Heutzutage liegt den Fokus unteranderem auf

die Benutzerzufriedenheit, indem gemessen wird, wie oft das System seine Nutzer zu einer falschen Entscheidungen geführt hat[23]. Diese beide Evaluationsmethoden werden noch Online- und Offline-Methoden genannt.

Bei der Online-Methode wird die Reaktion des Benutzers im Livesystem auf die präsentierten Empfehlungen gemessen. Bei dieser Methode sind die Klassifikation-Metriken die wichtigsten Faktoren. Da die Online-Evaluierung eine aktive Beteiligung der Nutzer erfordern, ist diese oft für Benchmarking und Forschung ungeeignet. Es ist wichtig das System mit mehrere Datensätze zu testen, um sicher zu stellen, dass der Algorithmus unter einer Vielzahl von Bedingungen gut funktioniert. In diese Fälle ist die Offline-Evaluierung geeignet, wobei historische Daten der Benutzer verwendet werden [22]. Diese Methode wird in der Literatur viel mehr behandelt, obwohl die Online-Methode zum Vervollständigen der Evaluierung notwendig ist. Im Folgenden werden die Evaluationsmetriken beschrieben.

4.3.1 Genauigkeitsmetriken

Die Genauigkeitsmetriken messen, wie nah die vom Empfehlungssystem vorhergesagten Bewertungen an den tatsächlichen Benutzerbewertungen sind[22]. Dieser Metriken werden von der Offline-Methode verwendet. Die Genauiketsmetriken sind der mittlere absolute Fehler (MAE) und der mittlere quadratische Fehler (RMSE)[22].

Die **MAE** misst die durchschnittliche absolute Abweichung zwischen einer vorhergesagten Bewertung und der tatsächlichen Bewertung des Nutzers. Zur Berechnung wird der Datensatz in Training- und Testdatensatz aufgeteilt. Dann wird das Modell mit Trainingsdatensatz angelernt und mit Testdatensatz verglichen. Dabei werden die tatsächliche Benutzerbewertungen $R_{u,i}$ geheim gehalten und das MAE mit der vorhergesagte Bewertungen berechnet.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |r_i - R_i| \tag{4.7}$$

Die **RMSE** ist eine mit dem MAE zusammenhängende Maß, die die Fehlerquadrate summiert. Größere Fehler sind durch RMSE schneller sichtbar als kleinere. Aus diesem Grund ist RMSE am häufigsten zur Evaluierung von Empfehlungssystemen eingesetzt [23].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i - R_i)^2} \tag{4.8}$$

4.3.2 Klassifikationsmetriken

Klassifizierungsmetriken messen, wie häufig das Empfehlungssystem den Benutzer zur richtige oder falsche Entscheidungen aufgrund der Vorhersage führt. Diese Metrik wird oft bei der Online-Methode verwendet. Die Evaluation dieser Systeme entwickelt sich zu einem klassischen Klassifikationsproblem mit vier unterschiedliche Klassen: True-Positive, False-Positive, False-Negative, True-Negative (Abbildung 4.2).

	Empfohlen (System)	Nicht empfohlen (System)
Genutzt (Benutzer)	True-Positive	False-Negative
Ungenutzt (Benutzer)	False-Positive	True-Negative

Tabelle 4.2: Klassifikationstabelle

Die Evaluationsmetriken bei diesem Ansatz sind *Precison* und *Recall*. Das Precision (Gleichung 4.9) ist der Ratio von guten klassifizierten Objekten auf die gesamte Anzahl an vorgeschlagene Objekte. Diese gibt die Wahrscheinlichkeit an, in wie fern vorgeschlagenes Objekt für den Benutzer relevant ist[23]

$$Precision = \frac{TP}{TP + FP} \tag{4.9}$$

Das Recall (Gleichung 4.10) lässt sich definieren als das Verhältnis der ausgewählten relevanten Objekte zur Gesamtzahl Anzahl der verfügbaren relevanten Objekte. Es gibt die Wahrscheinlichkeit an, dass ein relevantes Objekt ausgewählt wird[23]..

$$Recall = \frac{TP}{TP + TN} \tag{4.10}$$

Es gibt verschiedene Ansätze, um Precision und Recall zu kombinieren. Ein davon ist F1-Metrik (Gleichung 4.11), die beide Metriken zu einer Zahl kombinieren. Es dient dazu das harmonische Mittel beider Werte zu berechnen.

$$F1 = 2.\frac{Precision.Recall}{Precision + Recall} \tag{4.11}$$

5 Empfehlungssystem für die Online-Vorlesung

Das vorherige Kapitel hat eine ausführliche Grundlage über Empfehlungssysteme dargestellt, welche als Grundlage für jede Anwendungsgebiet dient. Es wurden dabei verschiedene Methoden vorgestellt, wobei Annahmen getroffen wurden. Ein der Anwendungsgebiet könnte das E-Learning bzw. die Online-Vorlesung sein. In diesem Kapitel wird versucht diese Arbeit thematisch genauerer in diesem Gebiet einzuordnen. Es werden einige Annahmen und Methoden aus dem Grundlagen-Kapitel betrachtet und in diesem Kontext adaptiert. Insbesondere wird der Bereich E-Commerce mit E-Learning verglichen. Außerdem wird in diesem Kapitel die Struktur einer Online-vorlesung beschrieben, die auch als Basis für die automatisierte Zerlegung der Aufzeichnungen dient.

5.1 Empfehlungssystem im Kontext Vorlesung

Das Empfehlungssystem wurde bereit im Bereich E-Commerce erfolgreich eingesetzt. Obwohl die eingesetzten Methoden ähnlich mit E-learning sind, werden einige Unterschiede in [2] notiert:

- Ziel: Das Ziel von Empfehlungssysteme in E-Commerce ist es, den Umsatz zu steigern. Dieses Ziel ist messbar z.B. in Form von Geldbeträge. Im Bereich E-learning ist das Ziel von Empfehlungssysteme die Verbesserung des Lernprozesses, welcher eher subjektiv und schwierige zu messen ist.
- Nutzungszweck: In E-Commerce ist der Zweck des Empfehlungssystems die Kunde beim Kauf zu unterstützen während in E-Learning die Lernende beim Lernen unterstützt werden.
- Technik: Die Anwendung von Empfehlungssysteme in E-Learning stellt besondere Herausforderungen, die in anderen Bereichen nicht zu finden sind, insbesondere die Notwendigkeit, die pädagogische Aspekte des Lernenden und des Systems zu

berücksichtigen. Manche traditionellen Techniken aus der Grundlage können in E-Learning angesetzt werde, manche nicht.

In Kapitel 4 wurde das Empfehlungssystem definiert als ein Softwaresystem, welches nützliche Objekte den Benutzer auf Basis ihres Bewertungsverhalten empfiehlt. Dabei wurden Objekte als E-Commerce Produkte gesehen. Für der Kontext E-Learning sind die Objekte kontextbezogene Elemente wie Lernmaterialien oder Lernabschnitte [2].

Ebenso wurden bisher Benutzer eines Empfehlungssysteme in eindimensional betrachtet. Im Bereich Online-Vorlesung wird zwischen zwei Benutzergruppen unterschieden: die Professoren und Studierende. Die Professoren sind für die Planung und Bereitstellung der Lernressourcen zuständig. Sie können das Empfehlungssystem nutzen, um sich ein Überblick über Studierendenverhalten oder Leistung zu verschaffen. Dieser Ansatz von Empfehlungssystem ist nicht Teil dieser Arbeit. Die zweite Benutzergruppe sind Studierende, die eigentliche Endnutzer des Empfehlungssystem sind. Sie sind auch gleichzeitig einen zentralen Bestandteil des Systems aufgrund ihrer aktiven Rolle.

In [2] werden spezifische Ziele für Empfehlungssysteme im Kontext E-Learning definiert:

- Find Novel Items: Neuen oder passenden Lernmaterialien/-inhalte an Studierende empfehlen.
- Find Peers: Lernpartnern empfehlen, die gleiche Interessen oder Lerneigenschaften aufweisen.
- Find Good Pathways: Alternativen Lernpfade empfehlen.

Bisher wurde das explizite Feedback als Bewertung von Benutzer in Form von Skala definiert. In Bereiche wie E-Commerce, Entertainment wo der Unterhaltungsfaktor im Vordergrund steht, reicht diese Form von Information aus, da ein Benutzer ein Produkt entweder gut oder schlecht finden kann. Im Bereich E-Learning ist es nicht ausreichend Aufgrund der Komplexität von Lernressourcen [28, 29]. Ein Studierender kann ein Lernabschnitt schlecht bewerten, weil der Inhalt ihm nicht detailliert fällt. Er kann auch eine geringere Bewertung geben, weil ihm den Inhalt zu schwer oder zu leicht empfindet. Diese Bewertungen basierend auf verschiedenen Interpretationen können vom Empfehlungssystem nicht berücksichtigt werden. In diesem Zusammenhang hat sich den Begriff implizit Feedback in vielen Bereiche etabliert [30]. Im Bereich E-Learning, wird vor allem den Umgang von Studierende mit Lernressourcen analysiert und die Bewertung davon abgeleitet. Ein guter Kandidat dafür wäre die Dauer oder die Häufigkeit der Nutzung von Lernabschnitte [31].

5.2 Die Online-Vorlesung

Unter dem Begriff Vorlesung wird eine Lehrveranstaltung verstanden, welche im Kern darin besteht, Studierende mittels eines Vortrags zu unterrichten [32]. In traditionelle Vorlesungen, der Dozent steht vorne im Hörsaal, spricht und zeigt Folien, während die Studierende zuhören und sich Notizen machen [33]. Das Ziel der Vorlesung ist es, wissen an die Studierende zu vermittelt. Bei der Vorlesung nutzen Dozenten unterschiedliche Lehrmethoden, die alle dieselbe Ziel haben, dass Studierende das dargestellte Thema verstehen. Für die Verständigkeit einer Lehrveranstaltung, beginnen Pädagogen meistens mit einer Einleitung und erklären die Kernideen beim weiteren Verlauf. Dabei werden Hilfsmittel wie Textes, Bilder oder Diagramme verwendet. Eine gute Strukturierung der Vorlesung führt zu einem besseren Verständnis. Um zu diesem Ziel zu kommen, nutzten Pädagogen eigene Wege. Trotzdem werden in viele Vorlesungen einige Gemeinsamkeit gefunden: sie enthalten eine zu Beginn eine Einleitung, in der meistens das Thema aus der früheren Vorlesung wiederholt wird. Danach folgt den eigentlichen Inhalt mit Beispielen, Praktische Übungen, Diskussion etc. Am Ende eines Vorlesungsabschnitts werden Fragen oft gestellt.

Während der Vorlesung kann der Informationsfluss durch ein Direktgespräch erfolgen oder der Dozent kann Hilfsmittel verwenden. Beim Direktgespräch steht der Professor im Vollbild und die Informationsquelle ist die Audiospur. Je nach Inhalt der Vorlesung können bestimmte Hilfsmittel wie PowerPoint Folien und Whiteboard verwendet werden. Andere Hilfsmittel wie externe Programme kommen auch oft zum Einsatz.

Die Online-Vorlesung ist die multimediabasierte Form der Vorlesung. Es fängt meistens mit einer Aufzeichnung der Vorlesung an, anhand dessen Lerneinheiten/abschnitte erzeugt werden. Lerneinheiten/abschnitte sind thematische Unterteilung der Vorlesungsaufzeichnung, welche als Grundlage für die Online-Vorlesung dienen. Zur Aufzeichnung wird oft ein Screen Recording verwendet. Beim Screen Recording wird der Bildschirminhalt des Präsentationsrechner aufgezeichnet. Ein Nachteil dieses Verfahrens ist, dass die Skalierung deren Pixelrepräsentation nur unter Qualitätsverlust erfolgt[34]. Für die Aufzeichnung einer Vorlesung bietet sich unter Umständen auch das Format Videokonferenz, wobei Kameras und Mikrophones zu Einsatz kommen.

6 Konzeption

Bisher wurde die hervorgehobenen Probleme in dieser Arbeit beschrieben und die theoretischen Grundlagen dafür erarbeitet. In diesem Kapitel wird eine Lösung konzipiert. Die vorgeschlagene Lösung wird anhand der Systemarchitektur und des Systemablaufs dargestellt.

6.1 Systemarchitektur

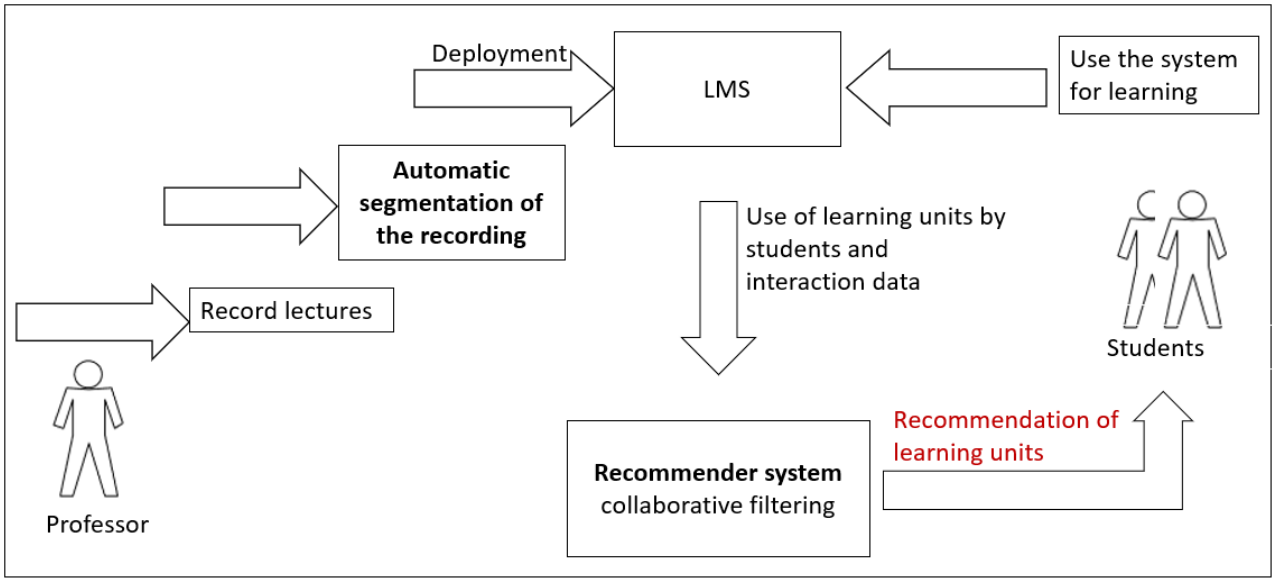


Abbildung 6.1: Systemarchitektur

Die Abbildung 6.1 stellt die Systemarchitektur dar. Zu Beginn des Prozesses, zeichnet der Dozent die Vorlesungen auf. Dann werden diese mit dem System zerlegt, um passende Lerneinheiten zu erstellen. Diese Lerneinheiten werden in das LMS zur Verfügung gestellt. Im Rahmen ihrer Lernprozesse greifen Studierenden darauf zu, um sich das dargestellte Wissen anzueignen. Die Nutzungsdaten der Lerneinheiten durch letzteren

werden vom LMS System protokolliert. Diese fließen später in das in LMS integriertes Empfehlungssystem ein. Nach Anwendung von kollaborativen Algorithmen auf diese Daten, wird der Studierenden Lerneinheiten empfohlen. Die Herausforderung in dieser Arbeit liegt im Teil der automatisierten Zerlegung der Aufzeichnungen und Realisierung des Empfehlungssystem. Eine detaillierte Beschreibung der beiden Komponente erfolgt im Systemablauf.

6.2 Systemablauf

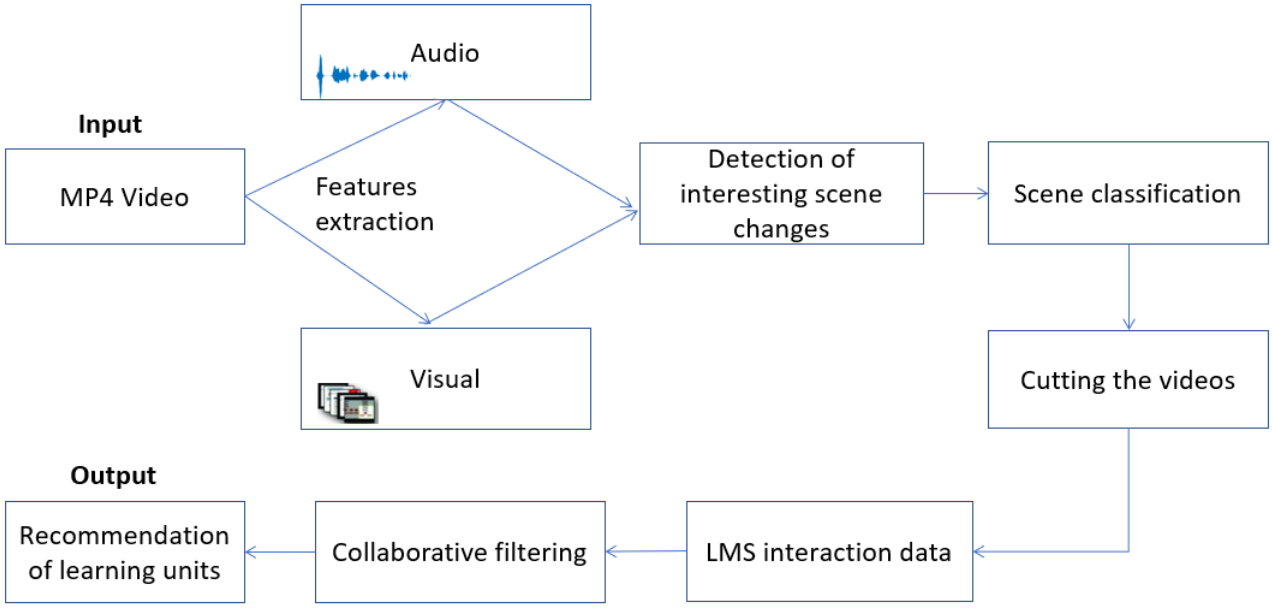


Abbildung 6.2: Systemablauf

Die Abbildung 6.2 stellt den Ablauf des zu entwickelten System dar. Die dazugehörige Komponente können in zwei Prozesse sortiert werden. Von *MP4 Video* bis hin zu *Cutting the videos* gehören die Komponente zum Prozess der *automatisierten Zerlegung der Aufzeichnungen*. Die Komponente *Collaborative Filtering* und *Recommendation of learning units* gehören zum Prozess des *Empfehlungssystems für Lerneinheiten*. Was der Komponente *LMS interaction data* betrifft, ist es die Schnittstelle zwischen beiden Prozesse. Im Folgenden werden die einzelne Komponente eingegangen.

6.3 Input MP4 & Video Visual Features Extraction

Das Input des Systems ist eine Vorlesungsaufzeichnung-Videodatei in MP4-Format. Dieses Format wird häufig zum Speichern von Video-, Text- und Audiodaten verwendet wird. Nachdem diese Datei ausgelesen ist, startet die visuelle Feature-Extraktion. Hierfür wird die beschriebene Schlüsselbildextraktion-Technik aus Kapitel 3 verwendet. Das Video wird Bild pro Bild ausgelesen, dann die Farbhistogramme aus RGB der einzelnen Bildern berechnet. Diese gibt die Anzahl der Pixel des Bildes für einen bestimmten Helligkeitswert von 0 bis 256 an [35]. In der Literatur wurde nachgewiesen, dass eine signifikante Unähnlichkeit der Histogramme zwischen den Einzelbildern eines Video auf einen schnellen Szenenwechsel hinweist. Auf Basis der Farbhistogramme wird der Euklidische Abstand zwischen zwei aufeinanderfolgende Bilder berechnet. Anhand dieses Abstands entschieden wird, ob die Bilder signifikante Unähnlichkeiten aufweisen. In dieser Arbeit, wenn der Euklidische Abstand eines Bildes den Durchschnitt aller Distanzen übersteigt, wird das Bild als Schlüsselbild betrachtet. Für die Optimierung der Szeneänderungerkennung werden Audio-Features benötigt.

6.4 Audio Features Extraction

In Kapitel 3 wurde beschrieben, dass die Salienz ein wichtiger Faktor für die audiovisuelle Szeneanalyse ist. Dabei wurde zwischen akustische und visuelle Salienz unterschieden. In dieser Arbeit wird die akustische Salienz angewendet, um die Pause in das Videos zu erkennen. Dazu wird die Audio-Spur vom Eingangsvideos getrennt und separat analysiert. Mithilfe von Fouriertransformation-Algorithmen wird die Audio in Amplituden umgewandelt. Schließlich werden die Amplutuden analysiert, um Stillstände in das zu finden. In dieser Arbeit, werden Amplituden zwischen Werte -6 und 6 als Stillstände betrachtet. Diese Werte wurden ermittelt. Um betrachtet zu sein sollte der Stillstand mindestens 1 Sekunde sein. Als Pause werden Stillstände von mindestens 3 Sekunden betrachtet.

6.5 Detection of interesting scene changes

In den vorherigen Abschnitten wurde beschrieben, wie audiovisuelle Features extrahiert werden. Dabei wurde bei visuelle Features das Wechsel von Schlüsselbildern als Anfang einer neuen Szene betrachtet. Der Ansatz von Audio-Feature-Extraktion diente

dazu Pausen in das Video zu erkennen. In diesem Schritt werden diese Features kombiniert, um Interessante Szeneänderungen zu erkennen. Dazu werden die Zeitliche Features-Vektoren untersucht. Features-Positionen mit gleichzeitigen Änderungen in Audio und visuelle Features werden als Interessante betrachtet. Diese Positionen bekommen eine Höhe Priorität. Es wurde mehrfach beobachtet, dass vor einem Themawechsel der Professor Frage stellt, wartet auf die Antwort und danach die Powerpoint-Folie wechselt oder zuerst die Powerpoint-Folie wechselt, Frage stellt und auf Antwort wartet. Auch eine solche Ereignisse in das Video werden als interessante Änderungen betrachtet. Außerdem wird eine minimale konfigurierbare Dauer von 2 Minuten für Interessante Szenen definiert. Interessante Szenen dürfen keine Fehlerszene sein, welche durch Aufnahmefehler entstehen. In Anbetracht dessen können interessante Szenen aus einer Sequenz von Szenen bestehen.

6.6 Scene Classification & Cutting the videos

Nach Erkennung von interessante Szenen, werden diese klassifiziert. Hierfür werden 5 Klassen definiert:

- **Slide**: Szene mit PowerPoint-Folie im Vollbild
- **Whiteboard**: Szene mit Skizze, Handschiff
- **Professor**: Szenen mit dem Professor im Vollbild
- **Fault:** Aufnahmefehler (Schwarz Bild).
- **Other**: Externe Programme, Text, und sonstiges

Die Abbildung 6.3 veranschaulicht die verschiedene Szenenklassen.

Als Eingangs für den Klassifikator werden pro Szene drei Bilder übergeben: das Start-, End- und Mittelbild. Anhand eines trainierten Modell weist den Klassifikator der Bilder eine Klasse zu. Um Fehler bei der Klassenerkennung umzugehen, wird die meinst zugewiesen Klasse der drei Bilder als Szeneklasse verwendet. Das zu verwendete vortrainierte Modell besteht aus verschiedene Vorlesungsbilder der genannte Klassen. Diese Model, welches der CNN Klassifikator verwendet, stammt aus einer Vor-Arbei zum vorliegenden Thema. CNN ist eine Art künstliches neuronales Netzwerk, das häufig in der Bilderkennung und Verarbeitung angewendet wird.

Das Ergebnis der Klassifizierung, wird verwendet um die Liste der interessante Szenen zu verbessert, indem *Fault* Szenen (Fault) gefiltert werden. Des Weiteren wird die Szeneklasseninformationen bei der Generierung der LMS Daten verwendet, um die Vielfalt der

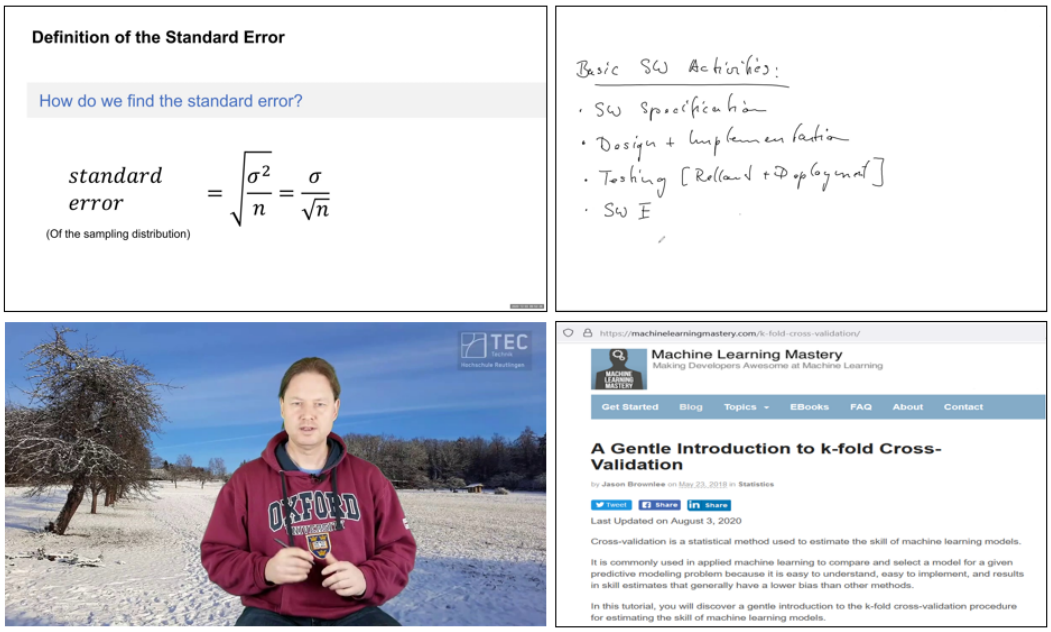


Abbildung 6.3: Unterschiedlische Arten der Szenen einer Vorlesung: Silde, Whiteboard, Professor, Other

generierte Daten zu gewährleisten. Zum Schluss der Szeneklassifizierung wird eine CSV-Datei der interessante Szenen, ihre Start- und Endpositionen, ihre Dauer und Szeneklasse erstellt. Diese Datei wird im weiteren Verlauf Vorlesungsskript genannt. Des Weiteren wird Anhand der Start- und Endpositionen jeder interessante Szene der Videoteil vom Eingangvideo mit dem FFmpeg-Tool geschnitten.

6.7 LMS interaction data

Wie erwähnt, sind die Interaktionsdaten vom LMS die Schnittstelle zwischen der *automatisierten Zerlegung der Aufzeichnungen* und dem *Empfehlungssystem für Lerneinheiten*. Die aus der Zerlegung resultierende Lernabschnitte werden verwendet, um LMS Interaktionsdaten zu generieren, welche für die Implementierung und Evaluierung des Empfehlungssystems benötigt werden. Im Kapitel 5 wurden Studierenden als Endbenutzer des Empfehlungssystems für Lerneinheiten beschrieben. Zunächst werden die Studentenprofile definiert.

- *learningDuration in %*: es handelt sich um die Nutzungsdauer der Lerneinheit durch den Studierenden. Es wird angenommen, dass bei interessanten Lernabschnitten die Studierende mehr Zeit verbringen. Diese Dauer wird in Prozentsatz

gegeben.Wenn eine Studierende ein Abschnitt vollständig angeschaut hat, dann ist die Dauer der Nutzung 100%.

- *numberOfPostsAboutScene*: es ist die Anzahl am Nachrichten, die die Studierende über die Lerneinheit in Forum des LMS geschickt hat. Dabei wird angenommen, dass bei mehr Nachrichten der Lerneinheit Interessant ist.
- *userId*: es ist eine eindeutige Nummer, die die Studierende identifiziert.
- *sceneId*: es ist eine eindeutige Nummer, die die Lerneinheit identifiziert.

Anhand von Profils wird der Vorlesungsskripts herangezogen, um die Benutzer und ihre Profildaten zu generiert. Vier Benutzergruppen werden generiert : Faule Studierenden, Mittel-faul, Mittel-, und gute Studierenden. Der Unterschied zwischen diese Gruppen liegt an dem Grad der Interaktion mit den Lerneinheiten. Z.B. gute Studierende interagieren viel mit Lerneinheiten als faule Studierende. Außerdem variieren diese Profildaten je nach Art (Klasse) der Lerneinheit . Meisten werden Whiteboards in der Vorlesung für Übungen und Demonstrationen verwendet. Faule Studierende Interessieren sich nicht für Übungen.

6.8 Collaborative filtering & Recommendation of learning units

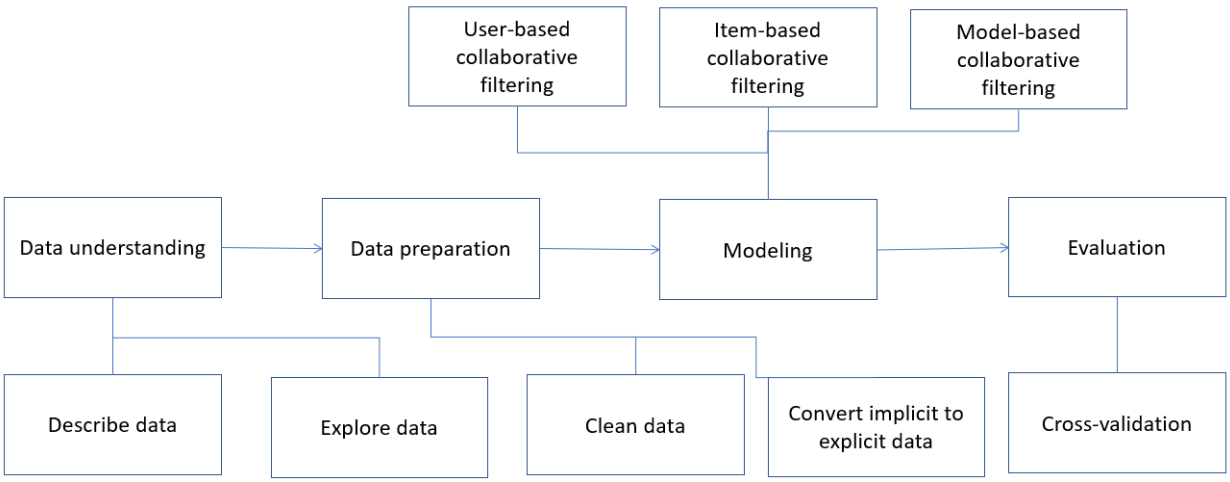


Abbildung 6.4: Lösungsprozess des Empfehlungsproblem nach CRISP-Modell

Die Implementierung des Empfehlungssystems folgt die klassische Lösungsschritte für ein maschinelles Lernen Problem. Die Abbildung 6.4 veranschaulisch den Lösungsprozess,

welchen vom CRISP-Modell abgeleitet ist. Im Folgenden werden die einzelne Komponente des Prozesses beschrieben.

6.8.1 Data understanding & preparation

In diesem Schritt werden die Interaktionsdaten vom LMS und der Vorlesungsskript analysiert. Danach werden die Interaktionsdaten bereinigt, indem mögliche redundante Daten entfernt werden. Des Weiteren werden die implizite Bewertungsdaten in explizite umgewandelt, um klassische kollaborative Algorithmen auf dem Datensatz anwenden zu können. Diese Algorithmen erwarten Daten in Form eines Tupels (*user, object, rating*). Die Umwandlung wird durch die Normierung und Gewichtung der Features *learningDuration in %* und *numberOfPosts* erfolgen. Nachdem beide Features in expliziten Bewertungen umgewandelt sind, wird bei der Modellierung das Feature *learningDuration in %* als *rating* verwendet. Während der Evaluation wird es mit dem Feature *numberOfPosts* kombiniert, um den Einfluss von neuen Features in das Empfehlungssystem zu bewerten. Eine Möglichkeit dieser Kombination ist die Addition.

6.8.2 Modeling

Bei der Modellierung wird auf dem vorbereitete Datensatz kollaborative Algorithmen angewendet. Im Kapitel 4 wurden Memory-basiertes und Model-basiertes kollaboratives Filtern beschrieben. Bei der Model-basierten Verfahrensart, wurde die Matrix-Faktorisierung mit SVD als das populärste Modell vorgestellt. Beim Memory-basierten Verfahren wurde zwischen Benutzer-basierten und Objekt-basierten Ansatz unterschieden. Im Rahmen dieser Arbeit werden die genannte Verfahren experimentiert, um das geeignete für das System auswählen. Im Folgenden werden Memory-basierte Ansätze noch mit konkrete Beispiele eingegangen.

6.8.2.1 Benutzer-basiertes kollaboratives Filtern

Die Abbildung 6.5 stellt ein Beispiel für den Benutzer-basierten Ansatz dar. Wie in dieser Abbildung dargestellt, hat die Studentin die Lerneinheiten 1, 2 und 3 angeschaut und der Student die Lerneinheiten 1 und 2. Da dieser Student die Lerneinheit 3 noch nicht angeschaut hat, wird das System ihm es empfehlen, wenn er ähnlich mit der Studentin ist.

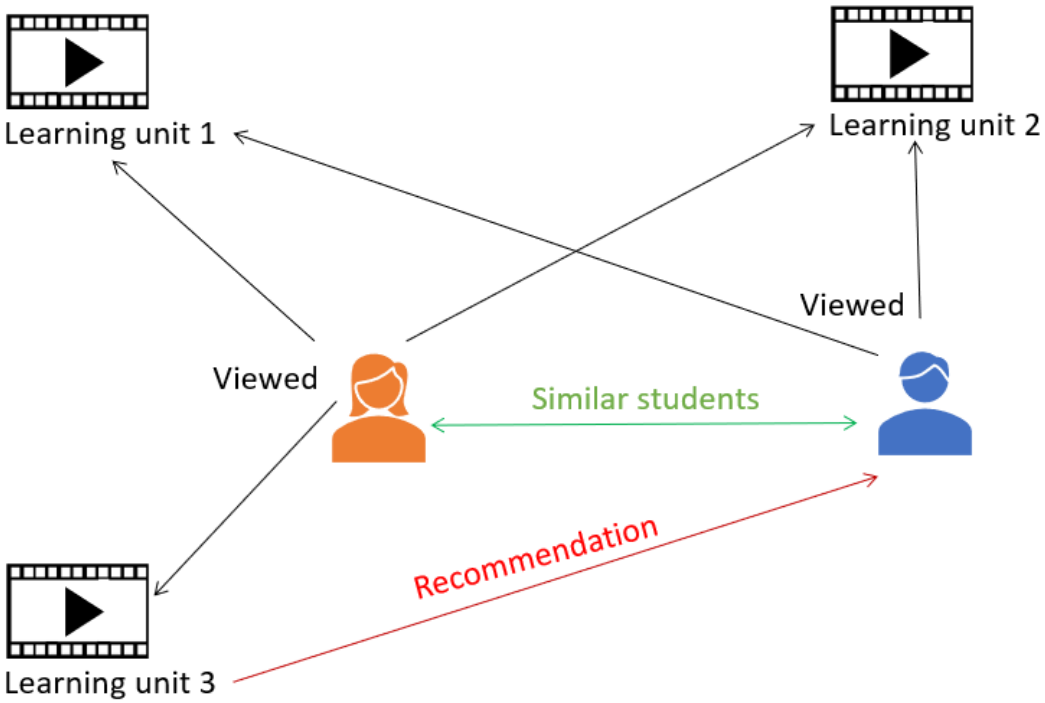


Abbildung 6.5: Beispiel Benutzer-basiertes kollaboratives Filtern

6.8.2.2 Objekt-basiertes kollaboratives Filtern

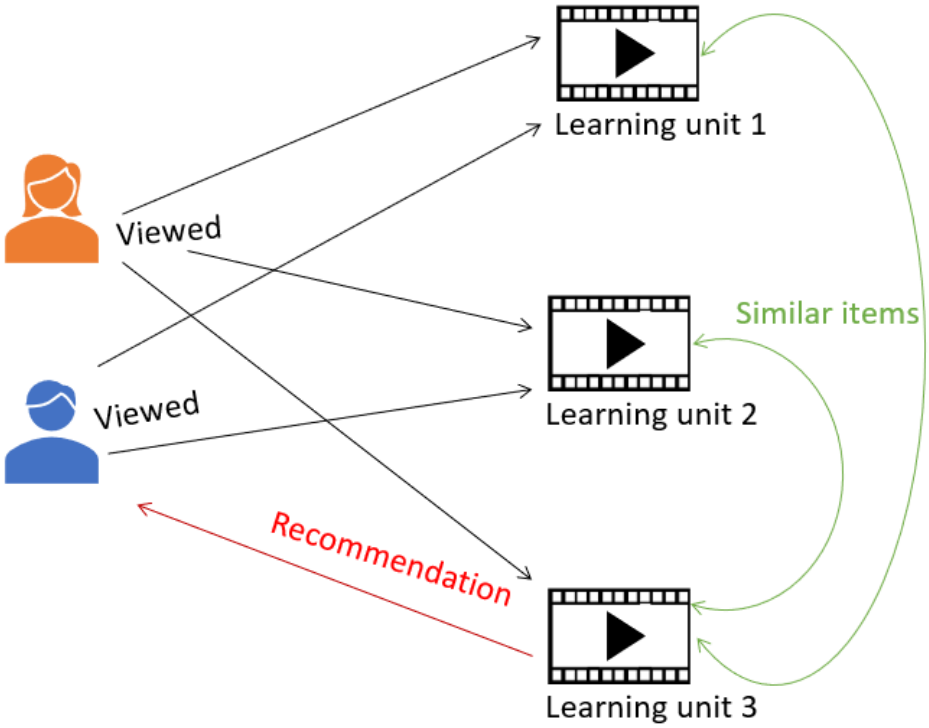


Abbildung 6.6: Beispiel Objekt-basiertes kollaboratives Filtern

Die Abbildung 6.6 veranschaulicht ein Beispiel der Objekt-basierte Methode dar. In diesem Beispiel hat die Studentin alle Lerneinheiten angeschaut und der Student nur die Lerneinheiten 1 und 2. Da dieser Student die Lerneinheit 3 noch nicht angeschaut hat, wird es ihm vom System empfehlen, wenn mindestens eine der von ihm angeschaute Lerneinheiten (1,2) ähnlich zu der Lerneinheiten 3 ist.

6.8.3 Evaluation

Im Kapitel 4 wurde die Offline-Methode als geeignete Methode für die Forschung und Benchmarking beschrieben. Da in dieser Arbeit es um Experimente geht, wird diese Methode verwendet. Die generierte Datensätze werden auf die entwickelte kollaborative Algorithmen angewendet, um ihre Genauigkeiten zu prüfen. Dazu wird das Framework k-fold Kreuzvalidierung (k-fold Cross Validation) verwendet. Es ist eine statistische Methode zur Einschätzung der Fähigkeiten von maschinellem Lernen Modelle, welche auch

bei Empfehlungssystem angewandt wird[36]. Dieses Framework teilt den Datensatz in k gleich große Partitionen. Jede einzelne Partition wird als einständiger Datensatz für die Evaluation verwendet. Die Modelle des Verfahren werden auf die restliche k-1 Partitionen trainiert und mit dem Testpartition die Vorhersage für die Benutzer evaluiert [36]. In dieser Arbeit wird das beschriebene Verfahren mit dem Parameter$k = 5$ evaluiert.

7 Implementierung

In diesem Kapitel werden die Implementierungsdetails zum beschriebene Konzept in Kapitel 6 veranschaulischt. Für die Implementierung werden Vorlesungsaufzeichnungen vom Forschungsinstitut Herman-Hollerith-Zentrum zur Verfügung gestellt, die sich aus verschiedenen Studienfächern der Bachelor- und Masterprogramme zusammensetzten. Diese Online-Vorlesungen wurden von verschiedenen Dozenten gehalten und mit dem Videokommunikationsprogramm Zoom aufgezeichnet. Für alle Programmierungsaktivitäten wird Python verwendet. Python ist ein High-Level und Allzweckprogrammiersprache, die aufgrund seiner hochentwickelten Datenstrukturen im Bereich des maschinellen Lernen weit verbreitet ist [37]. Ausgehend von der verwendeten Technologien, werden im Weiteren alle wesentliche Aktivitäten beschrieben.

7.1 Technologien

Aufgrund der explorativen Datenanalyse, findet die Entwicklung in zwei unterschiedliche Umgebungen statt. Die Entwicklung des *Empfehlungssystems für Lerneinheiten* findet in Jupiter Notebook statt. Jupiter Notebook ist eine Open-Source-Webanwendung, die ermöglicht die Erstellung von Dokumenten, welche Python Code, Visualisierungen und Kommentare enthalten [38]. Die Entwicklungsaktivitäten für die *automatisierte Zerlegung der Aufzeichnungen* finden in Pycharm statt. Pycharm ist eine integrierte Entwicklungsumgebung, die die Entwicklung von Data Science mit Anaconda unterstützt [39]. Verschiedenen Frameworks werden für die Implementierung des Systems eingesetzt. Die Abbildung 7.1 veranschaulicht diese und die Teilprozesse, wo diese eingesetzt sind.

Framework	Beschreibung	Teilprozess
OpenCV	Open Computer Vision (OpenCV) ist eine Open-Source-Software-Bibliothek für die Bild- und Videoanalyse, welche in C/C++ implementiert ist[40]. Die Version 4.5.1.48 wurde in dieser Arbeit Verwendet.	Visual features extraction
Surprise	Es ist eine Open-Source-Python-Bibliothek für die Entwicklung von Empfehlungssysteme mit expliziten Bewertungsdaten. Das Framework bietet vorgefertigte Vorhersage-Algorithmen wie KNN, SVD und NMF [41]. Die Version 1.1.1 wurde angewandt.	Collaborative filtering - Modeling & Evaluation
Pandas	Es ist eine Python-Bibliothek, die High-Level-Datenstrukturen für Daten-bezogene Aufgaben zur Verfügung stellt. Dieses Werkzeug wird oft als beliebteste Werkzeug zur Datenanalyse und -manipulation in Python angesehen[42].	Collaborative filtering - Data understanding & preparation
NumPy	Es ist eine Python-Bibliothek für wissenschaftliche Berechnungen. Die Bibliothek bietet Implementierungen von mehrdimensionalen Arrays und Matrizen.	Collaborative filtering - Data understanding & preparation Audiovisual Features Extraction
SciPy	Es ist eine Sammlung mathematischer Algorithmen und gängiger numerischer Routinen in Wissenschaft und Technik [43].	Visual features extraction
Matplotlib	Matplotlib ist eine Bibliothek zur Erstellung statischer, animierter und interaktiver Visualisierungen in Python[44].	Audiovisual Features Extraction
Moviepy	Es ist ein Python-Modul für die Videobearbeitung, das für grundlegende Operationen Audio-Extraktion anbietet [45].	Audio Features Extraction
CSV	Es ist ein Python-Modul zum Lesen und Erzeugen von tabellarischen Daten im CSV-Format [46].	LMS interaction daten Cutting the videos
Keras	Es ist eine Open-Source-Bibliothek, die eine Python-Schnittstelle für künstliche neuronale Netze bietet. Diese Bibliothek enthält Implementierungen von häufig verwendeten Bausteinen für neuronale Netze wie TensorFlow [47].	Scene classification
FFmpeg	Es ist ein Modul, das aus Bibliotheken und Programmen zur Verarbeitung von Video- und Audiodaten besteht [48].	Cutting the videos

Tabelle 7.1: Technologien

7.2 Klassen und Notebooks

Die Abbildung 7.1 veranschaulicht die verwendeten Klassen und Notebooks sowie ihre Funktionen.

Klasse/Notebook	Beschreibung
main.py	Es ist das Startprogramm, welche die nötigen Klassen aufruft, um die Auszeichnungen zu zerlegen.
sceneDetector.py	Es ist die Klasse, die für die Szenedetektion. Es verwendet Hilfsklassen *audioFeaturesUtils.py* und *VisualFeaturesUtils.py.*
audioFeaturesUtils.py	Es stellt Methoden für die Audio Feature-Extraktion und -analyse zur Verfügung.
VisualFeaturesUtils.py	Es stellt Methoden für die visuelle Feature-Extraktion und -analyse zur Verfügung.
sceneClassificator.py	Diese Klasse wird verwendet um die erkannte Szenen zu klassifizieren.
cutVideo.py	Diese Klasse ist für das Schneiden der Videos zuständig.
generateLMSinteractiondata.py	Es wird für die Generierung der Interaktionsdaten vom LMS verwendet.
utils.py	Es bietet allgemeine Hilfsfunktionen für das Programm
recommenderSystem_data_ Preparation.ipynb	Dieses Notebook wird für die Vorbereitung des Datensatzes verwendet.
recommenderSystem_item-based-collaborative-filtering.ipynb	Dieses Notebook wird für die Umsetzung des Objekt-basierten kollaborativen Filtern verwendet.
recommenderSystem_user-based-collaborative-filtering.ipynb	Dieses Notebook wird für die Umsetzung des Benutzer-basierten kollaborativen Filtern angewandt.
recommenderSystem_model-based-collaborative-filtering.ipynb	Das Notebook bietet eine Implementierung für das Model-basierte kollaborative Filtern

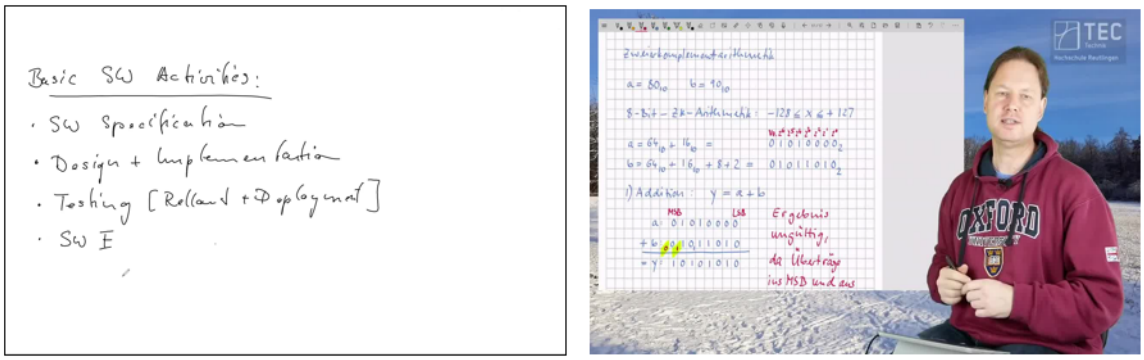
Abbildung 7.1: Struktur der Implementierung

7.3 Visuell Feature-Extraktion

Nachdem alle erforderliche Bibliotheken in PyCharm importiert sind, wird im ersten Schritt des Prozesses des *automatisierten Zerlegung der Vorlesungsaufzeichnungen* visuelle Features extrahiert. Dazu wird das Video-Verzeichnis vom Programmargumente extrahiert. In diesem Verzeichnis werden MP4-Dateien durchsucht und einzeln prozessiert. Die Datei wird mit der Methode *videoCapture* der OpenCV-Bibliothek ausgelesen und in ein Objekt *videoCap* gespeichert. Von diesem Objekt, werden die Video-Eigenschaften wie die Bildfrequenz (*fps*) und Anzahl der Gesamte Bilder des Videos (*totalFrames*) ausgelesen. Diese Eigenschaften werden während der gesamte Videoanalyse gebraucht.

Als nächstes findet die visuelle Feature-Extraktion in einer While-Schleife statt. Das Videoobjekt wird Bild pro Bild mit der Methode *read* der OpenCV-Bibliothek ausgelesen. Um die Dauer der Analyse zu reduzieren, wird nach dem Lesen das zu 50% verkleinert. Im Weiteren Verlauf wird zwischen zwei Arten von Vorlesungsvideo unterschieden. Die erste Arte ist eine Video, die *Whiteboard*-Szenen enthält mit Professor und das Whiteboard im Vollbild.

Für diese Art von Videos wird eine Vorprozessierung durchgeführt, wobei die Bildregion mit dem Whiteboard (ROI) extrahiert wird. Die restliche Region wird ignoriert. Das Ziel der Vorprozessierung ist es störende Informationen aus dem Bild zu entfernen. Zur Extraktion dieser Region wird die Methode *getRegionOfInterest* verwendet, welche die Methode *findContours* der OpenCV-Bibliothek anwendet. Des Weiteren wird von der extrahierte Region die Hintergrundinformationen subtrahiert. Dafür wird die Methode *preprocess* verwendet, welche verschiedene Algorithmen *GaussianBlur, Canny, dilate, erode* der OpenCV-Bibliothek auf das Bild anwendet. Die zweite Art der Vorlesungsvideo, ist Video mit mit *Whitebaord*-Szene ohne Professor im Vollbild. Für diese Art von Video wird keine Vor-Prozessierung durchgeführt. Die Abbildung 7.2 veranschaulisch die beiden Arten von Whiteboard.



Whiteboard ohne Professor im Bild Whiteboard mit Professor im Vollbild

Abbildung 7.2: Unterschiedliche Typen von Bilder

Im weiteren Schritt, wird die Farbhistogramm des Bildes berechnet. Dafür wird die Methode *calcHist* der OpenCV-Bibliothek genutzt. Anhand des Farbhistogramms wird der Euklidische Abstand zwischen des aktuellen Bildes und den vorherigen berechnet. Dazu wird die Farbmomente dieser Bilder berechnet. Dazu gehören die Farbdurchschnitt und die Standardabweichung. Die Euklidische Distanz ergibt sich aus der Summe der Differenzen der Farbmomente des aktuellen Bildes und des vorherigen Bildes. Hierzu wird die Methode *getEuclideanDistance(currentColorMoments, previousColorMoments)* verwendet.

Neben dem Euklidischen Abstand wird die Entropie und dominante Farbe des aktuellen Bildes. Die Entropie repräsentiert die durchschnittliche Information des Bildes. Hierzu kommt die logarithmische Funktion der Math-Bibliothek zum Ansatz. Auf Basis der Entropie wird die Entropie-Differenz des aktuellen und vorherigen Bildes berechnet. Die oben erwähnte While-Schleife wird verlassen sobald alle Bilder prozessiert sind. Die Abbildung veranschaulischt das Ergebnis dieses Vorgangs für ein kurzes Video von ca. 2 Minuten.

7.4 Audio Feature-Extraktion

Als erster Schritt der Audio Feature-Extraktion wird die Audiospur vom Eingangsvideo getrennt. Dazu wird das Video mit der Methode *VideoFileClip* der Moviepy-Bibliothek ausgelesen, dann seine Audio-Spur mit der Methode *write_audiofile* in einer WAV-Datei gespeichert. Im nächsten Schritt wird diese WAV-Datei mit der Methode *wavfile.read* der

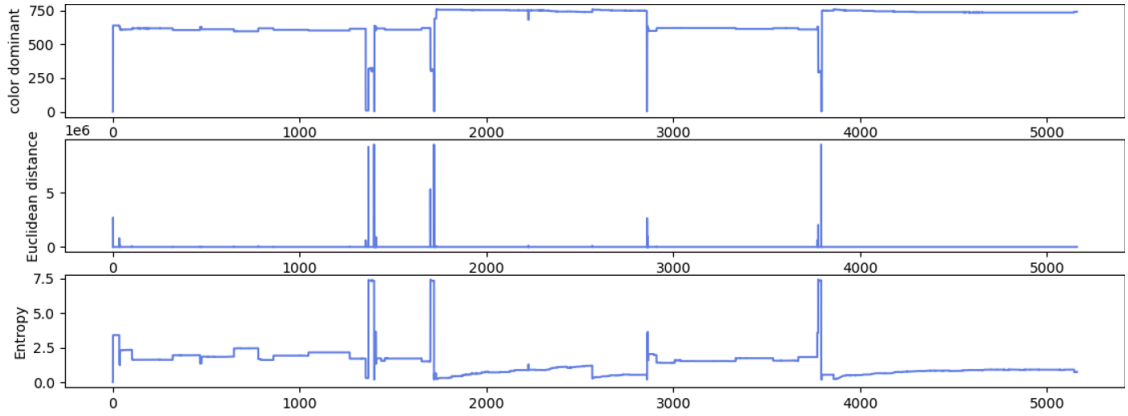


Abbildung 7.3: Ergebnis der Visuell-Feature-Extraktion

Scipy-Bibliothek ausgelesen und als numpy-Array von Audio-Frame in einer neuen Variable gespeichert. Die Audio-Frames repräsentieren die Amplituden des Audio-Signals. Die Abbildung 7.4 veranschaulisch die gelesene Amplituden für das in Abschnitt 7.3 erwähnte Video.

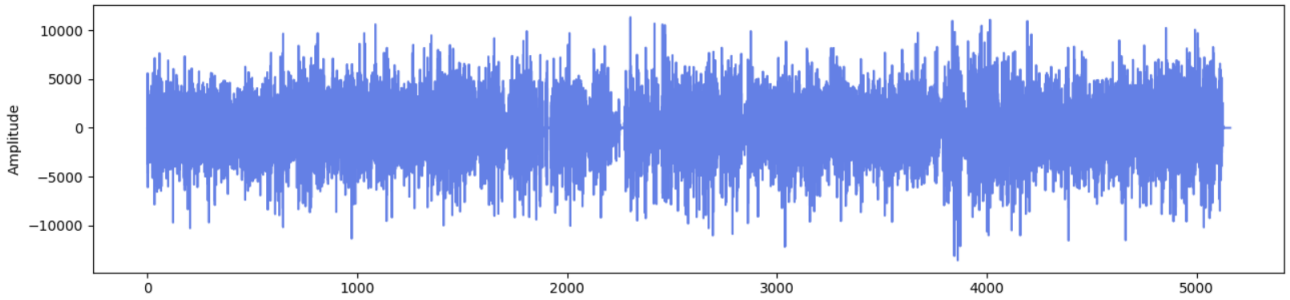


Abbildung 7.4: Amplituden des Audio-Signals

Im nächsten Schritt wird anhand dieser Amplitude-Daten Ruhezeiten in das Video ermittelt. Vorher wird versucht Audio- und Videoframes zu synchronisieren, indem bei Audio-Frames jedes k Frame betrachtet, mit *k=audio_rate/video_rate.* Das Ziel der Synchronisierung ist, identifizieren zu können auf welches Videoframe die Ruhezeit festgestellt ist. Schließlich werden Ruhezeiten in der Audioframeliste durchsucht. Wie in der Konzeption erwähnt, werden Ruhezeiten Bereiche mit Amplitude zwischen -6 und +6 betrachtet. Das Ergebnis dieser Teilprozess ist eine zeitliche Liste mit 0 und 1 wobei 0 an Positionen ohne Ruhezeiten steht und 1 an Positionen mit Ruhezeiten Pause.

7.5 Erkennung von interessante Szeneänderung

7.5.1 Erkennung von Szeneänderung

Nachdem audiovisuelle Features extrahiert sind, wird auf deren Basis Szeneänderung erkannt. Als Szene wird ist eine Sequenzielle Abfolge von gleichmäßige Bilder betrachtet. Bevor die Szenen erkannt werden wird die Durchschnitt Euklidische Distanz (*meanEuclideanDistance*) zwischen alle Videobilder berechnet. Der berechnete Wert wird als Schwellenwert für die Erkennung der Bildänderung verwendet. Dafür wird die Liste der euklidische Distanzen Sekundenweise durchgegangen. Innerhalb jede Sekunde werden die Abstände der mit dem genannten Schwellenwert verglichen. Sollte der Abstand den Schwellenwert übersteigen, dann wird die Distanz des entsprechende Bild zu dem nächst Bild verglichen. Sollte diese den definierte Schwellenwert von 1 übersteigen, dann wird das Bild als Schlüsselbild betrachtet. In der entsprechende Position in Eingangsvideo wird gespeichert. Die Abbildung 7.5) stellt ein Beispiel der Postionen von Schlüsselbilder in das Video dar. Die Features Color Dominant und Entropy werden hier nicht weiter betrachtet, da diese das selbe Ergebnis wie die euklidische Distanzen liefern.

```
[938.0, 1177.0, 2803.0, 2852.0, 3023.0]
```

Abbildung 7.5: Indexes von Schlüsselbilder

7.5.2 Interessante Szeneänderungen

Bisher wurden auf Basis audiovisuelle Szeneänderungen und Pausen in das Video erkannt. Nun werden diese Features kombiniert, um Interessante Szeneänderungen zu erkennen. Dazu werden die beide zeitliche Feature-Vektoren durchgegangen. Wie in Konzept erklärt, Positionen, wo Ereignisse in beiden Features gleichzeitig oder fast gleichzeitig passieren werden in einer neuen Liste gespeichert. In diesem Vektor, Positionen mit Ereignisse enthalten den Wert 1. Ebenso wir die Dauer der einzelne Szenen überprüft. Sollte eine Szene kürzer als 2 Minuten sein, wird diese nicht ein einzelne Szene betrachtet.

7.6 Szeneklassifizierung

In diesem Schritt werden erkannte Szenen aus dem vorherigen Abschnitt klassifiziert. Pro Szene werden die 3 genannte Bilder extrahiert, dann dem CNN-Klassifikator übergeben. Mit der Keras-Bibliothek werden die Bilder geladen, dann als Bild-Array konvertiert. Auf Basis dieses Arrays Vergleich der Klassifikator das Bild mit Modell erwähnten Model. Schließlich wird dem Bild eine Klasse mit der Vorhersage Funktion des CNN-Algorithmus einer Klasse zugeordnet. Da pro Szene drei Bilder übergeben wurden, wird die meist zugeordnete Klasse als finale Szeneklase behalten. Die Abbildung 7.6 stellt ein Beispielergebnis der Klassifizierung dar. Auf Basis der Ergebnisse der Klassifizierung, wird die Liste der interessante Szenen verbessert, indem Szenen der Klasse Fault entfernt werden.

```
{1: 'Slide', 2: 'Slide', 3: 'Professor', 4: 'Fault',
5: 'Whiteboard',6: 'Whiteboard'}
```

Abbildung 7.6: Ergebnis der Klassifikation

7.7 Schneiden der Videos

In diesem Teilprozess wird die Liste der interessante Szeneänderungen verwendet, welche aus Indexe der erste Bilder jeder interessante Szene besteht. Diese Bilderindexe werden als nächstes in Sekunde umgerechnet. Um die Videos zu schneiden, wird das Kommandozeile-Programm *ffmpeg* verwendet. Dem Programm wird die Start- und Einpositionen in Sekunde für jede Szene übergeben wird. Nach Ausführung des Programms werden die Videos in Distributionsverzeichnis erstellt, welches vom Videoverzeichnis abgeleitet wird. Ebenso wird der beschriebene Vorlesungsskript erstellt. Diese Datei wird im Abschnitt (7.9) näher eingegangen.

7.8 Interaktionsdaten vom LMS

Die zu generierende Daten wurden im Kapitel 6.9 beschrieben. Die Herausforderung in diesem Teilprozess ist es, ähnliche Benutzer zu generieren, um das Ergebnis der Benutzerbasierten kollaborativen Filtern prüfen zu können. Dazu wurde 4 Benutzerprofile definiert. Für jedes Profil werden arbiträre Wertebereiche definiert. Jeder Wertebereich besteht aus einer Unter- und Obergrenze.

```
very_lazy_students = [[[80, 100], [0, 0], [0, 10]], [0, 0], [5]]
middle_lazy_students = [[[60, 80], [0, 10], [10, 30]], [0, 1], [5]]
medium_students = [[[40, 60], [50, 60], [50, 80]], [1, 2], [5]]
good_students = [[[10, 20], [0, 100], [20, 100]], [1, 2], [5]]
```

Abbildung 7.7: Unterschiedliche Benutzergruppen

Das erste Listenelement ist wieder eine Liste von Wertebereichen für *learningDuration in %*, da für diese Profil der Wertebereiche abhängig von der Szeneklassen. Das erste Element dieser Liste steht für den Wertbereich der Szenen der Art *Professor*, das zweite für *Whiteboard* und das dritte für *Slide*. Z.B. Studierende des Profils Mittel-Studierende (medium_students) schauen 40 bis 60% der Szenen der Klasse *Professor*, 50 bis 60% der Szenen der Klasse *Whiteboard* und 50 bis 80% der Szenen der Klasse *Slide*. Das zweite Listenelement ist der Wertebereiche für die Profilinformation *numberOfPosts* und das letzte die Anzahl an der zu generierende Benutzer für diese Gruppe Nachdem die Wertebereiche für Daten definiert sind, wird pro Gruppe Zufallsdaten. Damit Benutzer innerhalb jedes Profil stark korrelieren bzw. ähnlich sind, wird pro Profil ein Benutzer generiert und auf Basis diesem die restliche Benutzer durch eine Art Klone generiert.

7.9 Kollaboratives Filtern

Dieser Abschnitt veranschaulicht die Implementierungsdetails zum kollaborativen Empfehlungssystem, welches in Kapitel 5 ausführlich beschrieben wurde. Nach Import der nötigen Bibliotheken in das Jupiter Notebook, wird im ersten Schritt dieses Prozesses versucht die Daten zu verstehen.

7.9.1 Data Unterstanding

Die LMS Interaktionsdaten aus Abschnitt 7.8 und der Vorlesungsskript aus Abschnitt 7.7 werden in Jupiter Notebook geladen und in jeweilige *panda* Dataframes gespeichert. Einfache Datenanalysemethoden werden durchgeführt, um nützliche über die Daten Informationen zu erhalten. Als erstes werden die Spalten der Datensätze untersucht. Die Tabelle 7.8 stellt eine Überblick über die Spalten des Vorlesungsskripts dar.

Um die erste 5 Daten dieser Tabelle anzuzeigen (Abbildung 7.9) wird die Methode *head* der Panda-Bibliothek verwendet .

Spalte	Beschreibung
sceneId	Es ist eine eindeutige ID, die die Szene identifiziert. Eine Szene repräsentiert eine hier eine Lerneinheit.
start frame	Index des ersten Bildes der Szene in das original Video.
end frame	Index des letzten Bildes der Szene in das originale Video.
duration in s	Dauer der Szene in Sekunde
scene class	Die bei der Klassifikation zugeordnete Klasse der Szene.

Abbildung 7.8: Spalten des Vorlesungsskript

	sceneId	start frame	end frame	duration in s	scene class
0	1	0.0	350.0	350.08	Slides
1	2	350.0	950.0	600.12	Professor
2	3	950.0	1830.0	880.32	Slides
3	4	1830.0	2863.0	1032.96	Slides
4	5	2863.0	3178.0	314.52	Slides

Abbildung 7.9: Erste 5 Einträge des Vorlesungsskripts

Mit der Methode *describe* der genannte Bibliothek, wird die beschreibende Statistik des Datensatzes ausgegeben (Abbildung 7.10).

	sceneId	title	start frame	end frame	duration in s	scene type
count	12.000000	12	12.000000	12.000000	12.000000	12
unique	NaN	12	NaN	NaN	NaN	2
top	NaN	movie_2	NaN	NaN	NaN	Slides
freq	NaN	1	NaN	NaN	NaN	10
mean	6.500000	NaN	3193.433333	3648.280000	454.776667	NaN
std	3.605551	NaN	1976.001373	1793.854356	368.932838	NaN
min	1.000000	NaN	0.000000	350.000000	6.600000	NaN
25%	3.750000	NaN	1610.000000	2604.750000	172.900000	NaN
50%	6.500000	NaN	3737.500000	4373.920000	332.300000	NaN
75%	9.250000	NaN	4788.400000	5012.290000	670.170000	NaN
max	12.000000	NaN	5450.720000	5457.320000	1119.560000	NaN

Abbildung 7.10: Deskriptive Statistik des Vorlesungsskripts

Der Statistik ist zu entnehmen, dass der Datensatz aus 12 Szenen besteht. Die Durchschnittliche Dauer einer Szenen ist 7.5 Minuten. Die längste Szene dauert 19 Minuten. Die Mehrheit der Szenen sind von der Klasse *Slides*.

Nachdem eine Überblick über die Szenen bzw. Lerneinheiten des Systems verschafft ist, werden die Interaktionsdaten vom LMS analysiert. Die Spalten dieses Datensatzes wurden schon im Kapitel 6.7 beschrieben. Die Abbildung 7.11 stellt die erste 5 Daten dieses Datensatzes dar.

	userId	sceneId	learningDuration in %	numberOfPosts
0	1	1	55	1
1	1	2	39	2
2	1	3	23	2
3	1	4	40	1
4	1	5	42	2

Abbildung 7.11: LMS Interaktionsdaten

Mit der beschreibenden Statistik, werden mehr Informationen über den Datensatz ent-

halten (Abbildung 7.12).

Abbildung 7.12: Beschreibende Statistik der LMS Interaktionsdaten

	userId	sceneId	learningDuration in %	numberOfPosts
count	240.000000	240.000000	240.000000	240.000000
mean	10.500000	6.500000	38.316667	1.337500
std	5.778332	3.459267	26.332913	1.022063
min	1.000000	1.000000	0.000000	0.000000
25%	5.750000	3.750000	17.000000	1.000000
50%	10.500000	6.500000	33.000000	1.000000
75%	15.250000	9.250000	57.250000	2.000000
max	20.000000	12.000000	100.000000	4.000000

Der Statistik ist zu entnehmen, dass die Studierenden im Durchschnitt 38% jeder Szene anschauen. Eine Vielzahl an Szenen wurden noch nicht angeschaut.

7.9.2 Data Preparation

Nachdem die Datensätze verstanden sind, wird der Datensatz mit LMS Interaktionsdaten vorbereitet. Zuerst wird geprüft, ob der Datensatz Duplikate enthält. Diese Prüfung ergibt keine Duplikate. Des Weiteren werden implizites zu explizites Feedback durch Normierung umgewandelt. Die betroffene Features sind *learningDuration in %* und *numberOfPosts*. Bei der Normierung werden beide Features in Wertebereich [0...1] gebracht. Da die Informationen für *learningDuration in %* in Prozentsatz vorliegen, werden diese durch 100 geteilt, um zum genannten Wertebereich zu kommen. Dadurch bekommt jede vollständig angeschaut Szene 1-Bewertungspunkt. Wenn eine Szene z.B. 10% angeschaut wurde, bekommt diese 0.1-Bewertungspunkt. Was das Feature *numberOfPosts* angeht wird dieses logarithmisch normiert. Diese erfolgt durch die Anwendung der unterstehende Gleichung.

$$log(\frac{1 + numberOfPosts}{1 + MAX(numberOfPosts)}) \tag{7.1}$$

In dieser Gleichung ist $MAX(numberOfPosts)$ die Maximale Anzahl an Posts im Datensatz. Als nächstes werden die genannten Features mit ihren Gewichten multipliziert.

	userId	sceneId	rating
0	1	1	2
1	1	2	1
2	1	3	1
3	1	4	2
4	1	5	2

Abbildung 7.13: Ergebnis der Konvertierung von implizite zu explizite Daten

Da das Feature *learningDuration in %* eine bessere Aussage über die Präferenz des Benutzers gibt, wird für diesem das Gewicht 5 zugewiesen und dem Feature *numberOfPosts* das Gewicht 2. Somit liegt die Bewertung mit *learningDuration in %* im Wertebereich [0...5] und von *numberOfPosts* im Bereich [0...2]. Nach der Umwandlung wird wie in der Konzeption erwähnt für die Modellierung das Feature *learningDuration in %* als Benutzerbewertung verwendet *(rating= learningDuration in %) und.* während der Evaluation beide Feature durch die Addition kombiniert (*rating=learningDuration in %+numberOfPosts*). Schließlich *learningDuration in %* und *numberOfPosts* gelöscht, da diese schon in das neue Feature *rating* umgewandelt sind. Die Abbildung 7.13 stellt das Ergebnis der Vorbereitung in der Form (*user, item rating*). Dieses Ergebnis wird in einer separaten CSV-Datei gespeichert.

Die Abbildung 7.14 veranschaulicht die Verteilung der Bewertungen im vorbereiteten Datensatz.

Abbildung 7.14: Explizite Bewertungsdaten

7.9.3 Modeling

In diesem Teilprozess findet die tatsächliche Umsetzung des kollaborativen Filtern statt. Nachfolgend werden die Implementierungsdetails eingegangen. Zur Umsetzung des Model-basierten und Objekt-basierten Ansatzes wird die Surprise-Bibliothek verwendet, welche vorgefertigte Algorithmen dafür bereitstellt. Für der Benutzer-basierter Ansatz wird das Ablaufsalgoritmus Schritt für Schritt umgesetzt, da die Umsetzung. Im Rahmen dieser Arbeit wird das Ranking Version of Problem als geeignet Methode für die Empfehlung verwendet. D.h. Nachdem die Bewertungen der unbewerteten Objekte vorhergesagt sind, werden die Top-N groß geschätzte Objekte als Ergebnis der Empfehlung verwendet.

7.9.3.1 Benutzer-basiertes kollaboratives Filtern

Die Abbildung 7.15 stellt die konkrete Implementierung des Algorithmus dar.

Im ersten Schritt wird der vorbereitete Datensatz im neuen Jupiter Notebook geladen, dann in ein *Panda* Dataframe gespeichert. Danach wird die Benutzer-Objekt-Matrix gebildet (Abbildung 7.16)

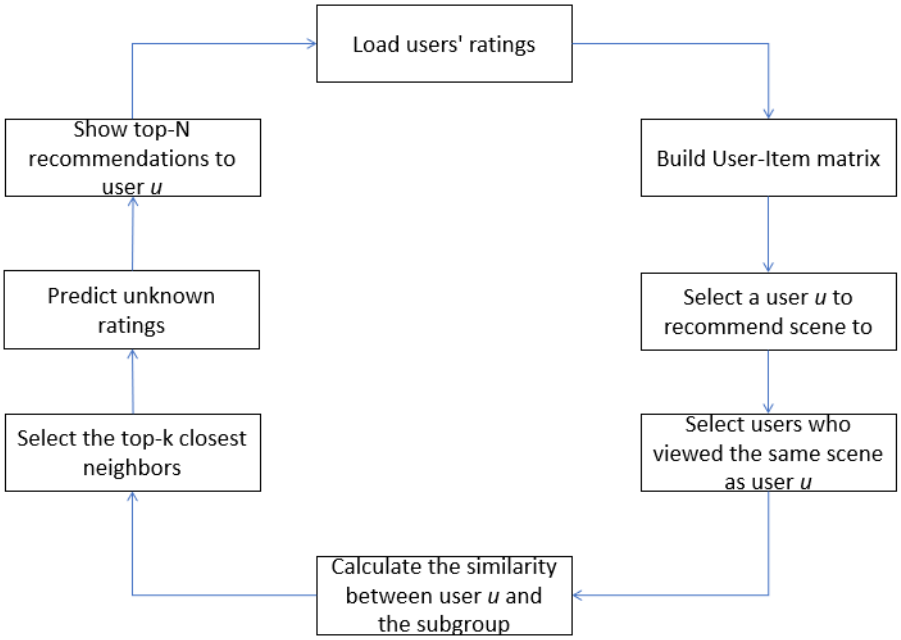


Abbildung 7.15: Benutzer-basiertes kollaboratives Filtern Algorithmus

sceneId	1	2	3	4	5	6	7	8	9	10	11	12
userId												
1	2	1	1	2	2	1	2	2	3	4	4	4
2	3	1	0	1	2	0	3	1	3	4	5	3
3	3	1	1	1	2	1	2	2	3	3	4	4
4	2	2	1	1	1	0	2	2	4	4	4	5
5	2	2	1	1	1	1	2	2	3	4	4	4

Abbildung 7.16: Benutzer-Objekt-Matrix

Aus dieser Matrix sind Szenen mit dem Bewertungswert 0 die unbekannte Szene für den Benutzer, deren Bewertungswerte vorhergesagt werden sollen. Als erstes wird ein aktiver Benutzer u ausgewählt, deren Bewertungswerte vorhergesagt werden. Um die Implementierung zu illustrieren, wird der Benutzer $(u = 2)$ ausgewählt. In sein Bewertungsliste, hat diesen Benutzer noch zwei unbewertete Szenen. Dieser Benutzer gehört zur Gruppe *Gute-Studierende* zusammen mit Benutzer 1 ,3 ,4 ,5. Im weiteren Schritt werden alle

Benutzer ausgewählt, die dieselbe Szenen gesehen haben, wie der Benutzer u. Dann wird mit dieser Sub-Gruppe von Benutzer die Ähnlichkeitsberechnung durchgeführt. Dafür wird Pearson Korrelation als Metrik verwendet. Das Ergebnis dieser Berechnung wird sortiert und die Top-k ähnliche Benutzer ausgegeben (Abbildung 7.17). In dieser Arbeit wurde $k = 5$ ausgewählt.

	similarityIndex	userId
1	1.000000	2
2	0.881358	3
0	0.860474	1
4	0.825340	5
3	0.785409	4

Abbildung 7.17: Top-5 ähnliche Benutzer

Wie in dieser Abbildung zu sehen ist, sind die Top-5 ähnliche Benutzer zu Benutzer u die andere Benutzer des selben Profil. Je nah der Korrelationswert (in Abbildung similarityIndex) an 1 ist, desto ähnlicher ist der Benutzer zu Benutzer u. In der gleiche Abbildung ist festzustellen, dass der aktive Benutzer u mit sich selbst zu 100% korreliert. Im Weiteren wird dieser Benutzer von der Liste ähnlicher Benutzer entfernt, um Bias in das Ergebnisempfehlung zu vermeiden.

Nachdem die Top-k ähnliche Benutzer ermittelt sind, wird auf Basis deren Bewertungen die Bewertungen des Benutzers u vorhergesagt. Zunächst werden die Bewertungen der Top-k ähnlichen Benutzer herangezogen, dann mit jeweiligen Ähnlichkeitsmaß multipliziert. Dadurch entsteht das Feature *weightedRating* (Abbildung 7.18).

	similarityIndex	userId	sceneId	rating	weightedRating
0	0.881358	3	1	3	2.644073
1	0.881358	3	2	1	0.881358
2	0.881358	3	3	1	0.881358
3	0.881358	3	4	1	0.881358
4	0.881358	3	5	2	1.762715

Abbildung 7.18: Feature *weightedRating*

Danach wird das *weightedRating* summieren, nachdem diese pro Szene gruppiert ist. Die Vorhersage-Bewertungswerte des Benutzers u ergeben sich aus der Formel *weightedRating*/sum_similarity. Die Abbildung 7.19 veranschaulich die Ergebnisse dieser Berechnung. Die Spalte *prediction score* repräsentiert die vorhergesagte Bewertungswerte. In dieser Abbildung ist es festzustellen, dass die beliebte Szenen der Gruppe an der ersten Stellen stehen. Außerdem nähren die Vorhersageergebnisse die Echte Bewertungswerte des Benutzers u annähern.

prediction score	sceneId
4.234270	12
4.000000	11
3.737111	10
3.234270	9
2.262889	1

Abbildung 7.19: Vorhersage für alle Objekte

Da das Ergebnis der Empfehlung ausschließlich aus unbekannte Objekte bestehen soll, wird im nächsten Schritt unbekannte Objekte ermittel und deren Bewertungen von der Vorhersageergebnisse entnommen. Zur Ermittlung dieser Objekte wird der Datensatz durchgegangen wird und alle Szenen des Benutzers u mit Bewertungswert 0 zurückgegeben. Die finale Liste mit der Top-N Empfehlungen wird zurückgegeben (Abbildung 7.20). In dieser Arbeit wurde $N = 10$ ausgewählt. Da der Benutzer u nur noch zwei unbekannte Objekte hatte, besteht das Ergebnis der Empfehlung aus dieser Objekte.

prediction score	sceneId
1.00000	3
0.76573	6

Abbildung 7.20: Ergebnis der Empfehlung

7.9.3.2 Objekt-basiertes kollaboratives Filtern

Die Modellierung dieses Ansatzes findet in einem separaten Jupiter Notebook statt. Wie erwähnt, ist der Ablauf des Algorithmus ähnlich zu dem Objekt-basierten Ansatz mit dem Unterschied, dass die Berechnung der Ähnlichkeiten Objekt-basiert stattfindet. Die Implementierung dieses Ansatzes erfolgt mit der Surprise-Bibiliothek. Im ersten Schritt wird der vorbereitete Datensatz aus Abschnitt 7.9.2 in das Notebook geladen und in ein *Panda* Dataframe gespeichert. Dann wird dieses Dataframe in ein *Surprise* Dataframe umgewandelt. Dafür wird der Wertebereich der Bewertungen des Datensatzes zu konfigurieren. Da die Bewertungen des Datensatzes im Bereich [1...5] liegen, werden diese Werte konfiguriert. Für die Berechnung der Ähnlichkeit zwischen Objekten, wird das KNN-Algorithmus der genannte Bibliothek verwendet. Bei der Instanziierung dieses Algorithmus wird der Parameter *sim_options* des Types *Dictionary* übergeben, welcher die Konfigurationsparameter *name* und *user_based* enthält. Beim Parameter *name* wird der zu verwendetet Metrik für die Ähnlichkeitsberechnung konfiguriert. Hier wurde die Kosinusähnlichkeit ausgewählt. Der Parameter user_based legt fest, ob die Berechnung der Ähnlichkeit Benutzer-basiert oder Objekt-basiert ist. Dieser Parameter wird auf *False* gesetzt, um die Berechnung Objekt-basiert zu machen. Nach dieser Konfigurationen wird der KNN-Algorithmus auf dem Datensatz angelernt, nachdem diesen in ein Surprise-Trainset-Objekt umgewandelt ist (Abbildung 7.21).

```
algo.fit(data_df.build_full_trainset())
```

```
Computing the cosine similarity matrix...
Done computing similarity matrix.

<surprise.prediction_algorithms.knns.KNNBasic at 0x1ff0de3c7f0>
```

Abbildung 7.21: Berechnung der Ähnlichkeit mit KNN

Nachdem das Modell träniert ist, können die nächste k ähnlichsten Objekten zu einem Ziel-Objekt mit der Methode *get_neighbors* des Algorithmus ausgegeben werden. Z.B. Für eine gegebene Szene $i = 1$ werden $k = 10$ ähnlichsten Szenen in der Abbildung 7.22 sortiert ausgegeben.

Im nächsten Schritt werden die bisher unbewertete Objekte durch den aktiven Benutzer ermittelt. Wie beim Benutzer-basierten Ansatz werden alle Objekte des aktiven Benutzer mit dem Bewertungswert 0 zurückgegeben. Danach werden die Bewertungswerte des

```
inner_id = algo.trainset.to_inner_iid(1)
neighbors = algo.get_neighbors(inner_id, k=10)
neighbors
```

```
[4, 7, 6, 8, 1, 3, 9, 2, 10, 11]
```

Abbildung 7.22: Ähnliche Objekte

aktiven Benutzers für unbekannte Objekte mit der Methode *predict* des Algorithmus vorhergesagt. Schließlich werden die Ergebnisse der Vorhersage sortiert und die Top-N Elemente ausgegeben. Die Abbildung stellt das Ergebnis der Empfehlung für den Benutzer $u = 2$ dar.

sceneId	prediction score
3	1.985293
6	1.938131

Abbildung 7.23: Ergebnis mit Objekt-basierten Ansatz

7.9.3.3 Model-basiertes kollaboratives Filtern

Ebenso wie bei Objekt-basierter Ansatz wird der vorbereitete Datensatz in ein *Surprise* Dataframe umgewandelt und das zu verwendete Algorithmus instanziiert. Hier wird das Algorithmus SVD der Surprise-Bibliothek verwendet, welche eine Implementierung der Matrix-Faktorisierung bereitstellt. Als nächstes wird das Modell auf dem Datensatz angelernt, um Muster in Bewertungen von Benutzer zu finden (Abbildung 7.24).

Nach diesem Schritt werden die bisher unbewertete Objekte durch den aktiven Benutzer ermittelt und deren Bewertungswerte vorhergesagt. Diese Vorhersage erfolgt mit der Methode *predict* des genannten Algorithmus, die als Parameter den Benutzer und das

```
algo.fit(data_df.build_full_trainset())
```

```
<surprise.prediction_algorithms.matrix_factorization.SVD at 0x1c8fde346d0>
```

Abbildung 7.24: Anlernen des Modells mit SVD

sceneId	prediction score
6	1.389786
3	1.283437

Abbildung 7.25: Ergebnis der Empfehlung mit SVD

vorherzusagende Objekts erwartet. Die Bewertungen von allen unbekannten Objekten werden vorhergesagt. Die Ergebnisse der Vorhersage aller unbewertete Objekte werden in einer Liste hinzugefügt. Anschließen wird die Liste nach Bewertungswert sortiert und die Top-N Elemente als Ergebnis der Empfehlung ausgegeben. Die Abbildung stelle das Ergebnis für den aktiven Benutzer $u = 2$

Die Evaluation der oben entwickelten Ansätze wird im nächsten Kapitel behandelt.

8 Evaluation

In diesem Kapitel wird das entwickelte System auf Performance und Genauigkeit evaluiert. Das ziel der Evaluation ist es, die Schwachstellen im System zu finden, um mögliche Optimierungsmaßnahmen zu definieren. Zunächst werden die verwendete Testdaten und Hardware beschrieben. Danach findet die Evaluation jeder Prozesses separat statt. Im Anschluss wird das Ergebnis diskutiert und Optimierungsmaßnahmen eingegangen.

Testdaten

Die Tests werden auf Online-Vorlesungen aus verschiedenen Studienfächern durchgeführt. Da diese Vorlesungen durch verschiedenen Professoren gehalten wurden, enthalten sie ihre einzelne Spezifizitäten bezüglich der Struktur der Vorlesung. Die Tabelle 8.1 stellt eine Übersicht der zu verwendete Online-Vorlesungen dar. Die erste drei Vorlesungseinheiten wurden von der Hochschule zur Verfügung gestellt. Diese Vorlesungen wurden mit dem Videokommunikationsprogramm Zoom aufgezeichnet. Die vierte Einheit wurde mit einem anderen unbekannten Programm aufgezeichnet und stammt aus Youtube.

Vorlesung	Einheit	Professor
Software Engineering	GMT20200327-125004_Software-E_2736x1700	A
Computernetzwerke	GMT20200327-125004_Software-E_2736x1700	A
Artificial Intelligence	Artificial Intelligence - VL - W21-Statistics-3	B
Digitaltechnik	MEB09 Digitaltechnik Übung - 2021-01-18	C

Tabelle 8.1: Testdaten

Hardware

Das unterstehende Hardware wurde für die Implementierung und Evaluation verwendet.

- **Hardware**: Lenovo-Laptop
- **Betriebssystem**: Windows 10
- **Prozessor**: Intel Core i7-8750H 2.20GHz
- **Arbeitsspeicher**: 8GB

8.1 Automatisierte Zerlegung der Vorlesungsaufzeichnung

In diesem Abschnitt wird die Performance der einzelnen Komponente dieses Prozesses gemessen. Des Weitteren wird die Erkennungsgenauikeit der Vorlesungsszenen überprüft und das Ergebnis der Zerlegung eingegangen.

8.1.1 Performance

Die Performance bezieht sich auf dem Datenspeicherverbrauch und die Laufzeit der Teilprozesse. Die Messungen werden in Tabellen 8.2 bis 8.5 veranschaulisch. Dabei sind die temporäre Artefakte, die zwischen Ergebnisse jedes Teilprozesse. Die Finale Artefakte repräsentieren die einzelnen Endergebnisse.

Lecture: Computernetzwerke Professor: Prof A Audio fps: 44100 Video fps: 25 Duration: 1:30:57	Temporal Artifacts (MB)	Final Artifacts (MB)	Running Time (sec)
Audio Features Extraction	WAV (918)	-	66
Visuell Features Extraction	-	-	2924
Detection of interesting scene changes	JPG (22), TxT, CSV	-	56
Scene Classification	-	-	12
Cutting the Videos	-	CSV, MP4 (286)	2
LMS Interaction Data	-	CSV	<1
Total	940	286	3062
Legend: - = Artifact not produced			

Tabelle 8.2: Performanz der Vorlesung Computernetzwerke

Vorlesung: Software Engineering Professor: Prof A Audio fps: 44100 Video fps: 25 Länge: 1:26:00	Temporal Artifacts (MB)	Final Artifacts (MB)	Running Time (sec)
Audio Feature Extraktion	WAV (864)	-	67
Visuell Feature Extraktion	-	-	2749
Detection of interesting Scene Changes	JPG (45,1), TxT	-	133
Scene Classification	-	-	26
Cutting the Video	-	CSV, MP4 (161)	3
LMS Interaktionsdaten	-	CSV	<1
Gesamt	891,1	161	2981
Legend: - = Artifact not produced			

Tabelle 8.3: Performanz der Vorlesung Software Engineering

Lecture: Artificial Intelligence Professor: Prof B Audio fps: 44100 Video fps: 30 Länge: 00:31:41	Temporal Artifacts (MB)	Final Artifacts (MB)	Running Time (sec)
Audio Feature-Extraktion	WAV (321)	-	
Visuell Feature-Extraktion	-	-	
Detection of interesting Scene Changes	JPG (69,1), TxT	-	
Scene Classification	-	-	
Cutting the Videos	-	CSV, MP4 (69,6)	
LMS Interaction Data	-	CSV	<1
Total	390,1	69,6	
Legend: - = Artifact not produced			

Tabelle 8.4: Performanz der Vorlesung Artificial Intelligence

Lecture: Digitaltechnik Professor: Prof B Audio fps: 44100 Video fps: 30 Duration: 01:12:24	Temporal Artifacts (MB)	Final Artifacts (MB)	Running Time (sec)
Audio Feature Extraction	WAV (730)	-	54
Visuell Feature Extraction	-	-	949
Detection of interesting Scene Changes	JPG (222), TxT	-	63
Scene Classification	-	-	124
Cutting the Videos	-	CSV, MP4 (702)	4
LMS Interaction Data	-	CSV	<1
Total	952	702	1196
Legend: - = Artifact not produced			

Tabelle 8.5: Performanz der Vorlesung Digitaltechnik

Datenspeicher

Aus der Messung ist zu entnehmen, dass der meiste Speicherplatz vom Teilprozess *Audio Feature Extraction* belegt wird. Die erzeugte Audio-Datei (WAV in temporäre Artefakte) ist bis zu 3 Mal größer als das Eingangsvideos. Diesen Unterschied kommt durch die unterschiedliche Kodierungsformate. Der nächste größte Speicherplatzverbraucher unterscheidet sich vom Aufzeichnungsprogramm. Bei Zoom-Aufzeichnungen ist der Teilprozess *Recognitions of interesting Scene Changes* durch die Erzeugung von Schlüsselbilder (JPG temporäre Artefakte), gefolgt vom Teilprozess *Cutting the Videos* durch die Erstellung von Lerneinheiten (MP4 in finale Artefakte) . Die Größe dieser Bilderdateien (JPG) ist abhängig an Anzahl ermittelten Schlüsselbilder. Was die gesamt Größe der Lerneinheiten betrifft, ist es fast gleich groß wie das Eingangsvideo, was zu erwarten war. Im Gegensatz zu Zoom-Vorlesungen, ist bei der Vorlesung Digitaltechnik der Teilprozess *Cutting the Videos* der nächst größte Speicherplatzverbraucher. Diesen Unterschied liegt wahrscheinlich darin, dass FFmpeg-Programm eine andere Kodierungstechnik als das Eingangsvideo verwendet. Die Artefakte CSV und TXT liegen in Byte bis Kilobyte-Bereich und können deshalb was die Speicherplatz Belegung angeht vernachlässigt werden.

Laufzeit

Auch hier es festzustellen, dass die Gesamtlaufzeit des Prozesses des *automatisierten Zerlegung der Aufzeichnungen* abhängig vom verwendete Aufzeichnungsprogramm ist.

Für eine Zoom-Vorlesung von 1 Stunde 30 Minuten, beträgt diese Zeit 51 Minuten. Das entspricht 57% der Dauert des Eingangsvideo. Bei der Vorlesung Digitaltechnik mit einem anderen Aufzeichnungsprogramm ist diese Laufzeit wesentlich kürzer. Diese Beträgt 27% der Dauer des Eingangsvideo. Insgesamt, wird die meiste Zeit im Teilprozess *Visuell-Feature Extraktion* verbraucht. Die verbrauchte Zeit beträgt entspricht mehr als 90% der Gesamtlaufzeit des Prozesses. Diese Größe dauert kommt dadurch, dass bei der Feature-Extraktion das Eingangsvideo Bild Pro Bild gelesen wird. Was die Szeneklassifikation angeht, ist die Laufzeit abhängig vom Anzahl an einzelne Vorlesungsszenen. Außerdem ist zu erwähnen, dass die Videoauflösung keinen Einfluss auf die Gesamtlaufzeit hat. Dieser Vorteil wurde durch die Redimentionierung der Bilder im Vor-Prozessierung Schritt gewonnen.

8.1.2 Erkennungsgenauikeit

In diesem Evaluierungsschritt wird geprüft wie genauerer das System die einzelnen Szenen, Pausen und interessante Szenen in der Vorlesungsaufzeichnung erkennt. Dazu werden die Aufzeichnungen manuell auf die Anzahl an vorkommenden Szenen, Pausen und Interessante Szenen analysiert und das Ergebnis gegen des Systemergebnis verglichen. Die Auswertung wird in der Tabelle 8.7 dargestellt. Als nächstes werden die Einträge dieser Tabelle in der Tabelle 8.7 beschrieben.

Measurement \ Lecture	Computer-netzwerke	Software Engineering	Artificial Intelligence	Digitaltechnik
Scenes	16	21	13	6
Breaks	13	30	1	32
Interesting Scenes	14	10	3	4
Recognized Scenes	23	46	36	250
Recognized Breaks	13	30	1	32
Class of Recognized Scenes	P(3), W(7), S(10), F(3)	P(3), W(7), S(30), F(6)	P(2), W(7), S(32), F(2)	P(250)
Recognized Interesting Scenes	11	13	4	4
Legend: P = Professor, F = Fault, W = Whiteboard, S = Slides,				

Tabelle 8.7: Ergebnis der Szeneerkennung

Metriken	Beschreibung
Single Scenes	Einzelne vorkommende Szenen im Eingangsvideo. Für eine Szene der Klasse Powerpoint, handelt es sich um den Videoteil mit dem selben Powerpoint-Folie. Für eine Szene der Klasse Whiteboard, ist es der Videoteil zwischen zwei leeres Whiteboards.
Breaks	Die vorkommenden Pausen im Eingangsvideos. Als Pause werden Ruhezeiten von mindestens 3s betrachtet.
Interesting Scenes	Lerneinheiten im Eingangsvideo. Diese ist eine Sequenz von Szenen.
Recognized Scenes	Vom System erkannten einzelnen Szenen.
Recognized Breaks	Vom System erkannten Pausen
Class of Recognized Scenes	Vom System zugeordnete Klasse der Szene
Recognized Interesting Scenes	vom System erkannte Lernabschnitte. Diese entsteht durch das Filtern von unnötige Szeneänderungen.

Tabelle 8.6: Beschreibung der Metriken

Vorlesung Software Engineering

- **Struktur**: Diese Vorlesungseinheit besteht aus 7 Szene aus unterschiedlicher Klassen. Die Vorlesung startet und endet mit einer Szene der Klasse *Slides*. Während der Vorlesung wird zwischen *Professor*-Szenen und *Whiteboard*-Szenen gewechselt oder zwischen *Professor*- und *Slide*-Szenen. Die *Slide-Szene* besteht aus einzelnen Szenen (Powerpoint-Folien). Ebenso besteht die *Whitebaord-Szene* aus einzelnen *Whiteboard*-Szenen.
- **Ergebnis der Erkennung**: Alle Szenen und einzelne Szenen werden vom System erkannt. Zusätschlisch werden *Fault-Szenen* erkannt. Es wurde festgestellt, dass die *Fault*-szenen oft durch die Transition von einer Szene zu einer andere von unterschiedlischen Klassen. Es werden 2 einzelne Szenen der Klasse Slide und Whiteboards 2 Mal erkannt.
- **Finales Ergebnis und Auswertung**: Aus der Analyse resultieren 13 Lerneinheiten (in der Tabelle Recognized Interesting Scenes). Die Dauert dieser Lerneinheiten liegen im Bereich von 2,5 bis 24 Minuten.

Vorlesung Computernetzwerke

- **Struktur**: Diese Einheit besteht aus 5 Szenen aus unterschiedlicher Klasse. Diese Vorlesung beginn und endet mit einer Szene der Klasse *Professor*. Während der Vorlesung wird zwischen Professor- und Whiteboard-Szenen geschaltet oder zwischen Professor- und Slides-Szenen.
- **Ergebnis der Erkennung:** Alle Szenen und einzelne Szenen werden vom System erkannt.
- **Finales Ergebnis und Auswertung**: Aus der Analyse entsteht 11 Lerneinheiten. Die Dauer dieser Einheiten liegt zwischen 2 und 16 Minuten.

Vorlesung Artificial Intelligence

- **Struktur**: Ebenso wie die Vorlesung Computernetzwerke, beginn und endet diese Vorlesung mit *Professor*-Szenen. Während der Vorlesung werden ausschließlich Powerpoint-Folien gezeigt. Diese Powerpoint-Folien enhalten viele Farben.
- **Ergebnis der Erkennung**: Alle Szenen und einzelne Szenen werden vom System erkannt. Es werden 3 einzelne **Sildes**-Szenen 2 Mal erkannt.
- **Finales Ergebnis und Auswertung**: Aus der Analyse resultieren 4 Lerneinheiten. die Dauer von dieser Einheiten liegt zwischen 4 und 11 Minuten.

Vorlesung Digitaltechnik

- **Struktur**: Diese Vorlesung beginn mit einer *Professor*-Szene. Während der gesamte Vorlesung wird ausschließlich auf Whiteboard gearbeitet. Hier geht es um das spezielle Whiteboard mit dem Professor und das Whiteboard im gleichzeitig Vollbild.
- **Ergebnis der Erkennung**: Die Professor Szene wird richtig erkannt. Nicht alle *Whiteboard*-Szenen werden erkannt.
- **Finales Ergebnis und Auswertung**: 4 Lerneinheiten resultieren aus der Analyse. Die Dauer von diesen liegt zwischen 4 und 35 Minuten. Die nicht Erkennung von manche *Whiteboard*-Szenen führt zur längere Dauer der mancher Lerneinheiten.

Bei alle Lerneinheiten wurde beobachtet, dass Außer der ersten Lerneinheit, die andere Lerneinheiten mit einem Schwarzbild beginnen. Dies kommt durch das Schneiden der

Videos. Wahrscheinlich verwendet dazu das FFmpeg-Programm nicht die richtige Kodierendung

8.2 Empfehlungssystem

In diesem Abschnitt werden die entwickelte kollaborative Verfahren auf die Vorhersagegenauigkeit der Benutzerbewertung und die Laufzeit geprüft. Am Ende wird das beste Modell für den Datensatz ausgewählt. Die Evaluierung findet gemäß Kapitel 6.8.3 statt. Die Algorithmen werden unter verschiedene Bedingungen getestet. Die Ansätze werden mit einem Datensatz getestet, wo die Anzahl an Benutzer wesentlich größer ist als die Anzahl an vorhandenen Objekten und umgekehrt. Für den Test werden die Lerneinheiten der vorherigen Abschnitts verwendet, um LMS Interaktionsdaten zu generieren. Das Kreuzvalidierung-Framework *cross_validate* der Surprise-Bibliothek wird für die Validierung verwendet, mit dem Parameter $kfolds = 5$.

8.2.1 Anzahl Benutzer größer als Anzahl Objekte

Zur Generierung des Datensatzes werden die Lerneinheiten der der Vorlesung Computernetzwerk verwendet (11 Lerneinheiten). Insgesamt werden 1000 Benutzer aus den 4 definierten Profile generiert. Die Abbildungen 8.1 bis 8.3 veranschaulischen das Ergebnis der Messung mit der Metriken MAE und RMSE.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	1.1839	1.1674	1.1691	1.1902	1.1830	1.1787	0.0089
MAE (testset)	0.9632	0.9317	0.9297	0.9365	0.9336	0.9389	0.0123
Fit time	0.97	1.01	0.92	0.96	0.94	0.96	0.03
Test time	1.05	0.92	0.86	0.92	0.88	0.93	0.07

10.77217149734497

Abbildung 8.1: Evaluation des Benutzer-basierten kollaborativen Filtern

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.9630	0.9358	0.9732	0.9486	0.9424	0.9526	0.0137
MAE (testset)	0.7471	0.7323	0.7512	0.7306	0.7372	0.7397	0.0082
Fit time	0.02	0.02	0.02	0.02	0.01	0.01	0.00
Test time	0.04	0.03	0.04	0.06	0.03	0.04	0.01

0.38523316383361816

Abbildung 8.2: Evaluation des Objekt-basierten kollaborativen Filtern

```
                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.7363  0.7482  0.7526  0.7567  0.7422  0.7472  0.0072
MAE (testset)     0.5859  0.5992  0.6040  0.6087  0.5870  0.5970  0.0091
Fit time          0.51    0.41    0.42    0.42    0.42    0.44    0.04
Test time         0.01    0.01    0.01    0.04    0.01    0.02    0.01
3.374509572982788
```

Abbildung 8.3: Evaluation des Model-basierten kollaborativen Filtern

Computernetzwerke, 1000 students, 11 learning units		MAE	RMSE	Running time(sec)
memory-based	User-based	0.93	1,17	10,7
	Item-based	0,73	0.95	0,3
model-based	SVD	0,59	0,74	3,3

Tabelle 8.8: Anzahl Benutzer größer als Objekte

Laufzeit

Der Messung ist zu entnehmen, dass der Benutzer-basierter Ansatz viel Zeit verwendet. Es wird dadurch begründet, dass bei diesem Ansatz die Berechnung der Ähnlichkeit Benutzer-basiert ist, und das System 1000 Benutzer enthält. Des Weiteren ist festzustellen, dass der Objekt-basierter Ansatz sehr wenig Zeit benötigt. Ein Grund dafür könnte sein, dass wenige Objekte (Lerneinheiten) im System vorhanden sind.

Genauigkeit

Aus der Messung ist zu entnehmen, dass der Model-basierter Ansatz eine bessere Vorhersagegenauigkeit im Vergleich zu den anderen Ansätze liefert. Des Weiteren ist festzustellen, dass der Objekt-basierter Ansatz ein besseres Ergebnis im Vergleich zum Benutzer-basierten Ansatz liefert.

In Tabelle 8.8 wird das Ergebnis der Messung noch zusammen.

8.2.2 Anzahl Objekte größer als Benutzer

Die Lerneinheiten aus der Zerlegung der beiden Vorlesungseinheiten von Professors A werden für die Generierung der Benutzer und LMS Daten verwendet. Der generierte Datensatz besteht aus 10 Benutzer und 24 Lerneinheiten. In der Abbildungen 8.1 bis 8.3 werden die Messung dargestellt.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	1.6259	1.3911	1.4799	1.6069	1.5770	1.5362	0.0882
MAE (testset)	1.4289	1.2181	1.2243	1.4536	1.3627	1.3375	0.0996
Fit time	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Test time	0.00	0.00	0.00	0.00	0.02	0.00	0.01

0.3690311908721924

Abbildung 8.4: Evaluation des Benutzer-basierten kollaborativen Filtern

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.9229	0.9367	0.8995	0.9344	0.9105	0.9208	0.0142
MAE (testset)	0.7666	0.7320	0.7446	0.7702	0.7576	0.7542	0.0142
Fit time	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Test time	0.00	0.01	0.00	0.00	0.00	0.00	0.00

0.09914278984069824

Abbildung 8.5: Evaluation des Objekt-basierten kollaborativen Filtern

Laufzeit

Hier ist festzustellen, dass die Laufzeit der Benutzer-basierter Ansatz deutlich gering ist, im Vergleich zum vorherigen Test. Es lässt sich dadurch begründen, dass im System nur wenige Benutzer vorhanden sind. Des Weiten ist festzustellen, dass der Objekt-basierten Ansatz auch in diesem Test die beste Laufzeit hat.

Genauigkeit

Die Ergebnisse der Vorhersagegenauigkeit sind ähnlich zum vorherigen Test. Des Weiteren ist hier festzustellen, dass der Objekt-basierter Ansatz ein besseres Ergebnis als im vorherigen Test hat. Sein Ergebnis ist ähnlich zu zum Model-basierten Ansatz, der immer der auch hier das beste Ergebnis liefert. Der Grund für ein besseres Ergebnis des Objekt-basierten Ansatz könnte sein dass das Algorithmus mir mehr Daten besser arbeitet.

Das Ergebnis dieses Tests wird noch in der Tabelle 8.9 zusammengefasst.

8.3 Diskussion

Die Evaluierung des Empfehlungssystems hat gezeigt, dass aufgrund der Größe Laufzeit und die unzureichende Vorhersagegenauigkeit beim Benutzer-basierten Ansatz, dass

```
                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.8935  0.9214  0.9383  0.9078  0.9222  0.9167  0.0151
MAE (testset)     0.7155  0.7925  0.7650  0.7373  0.7881  0.7597  0.0295
Fit time          0.02    0.02    0.02    0.02    0.01    0.01    0.00
Test time         0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.20740151405334473
```

Abbildung 8.6: Evaluation des Model-basierten kollaborativen Filtern

Computernetzwerke and Software Engineering, 15 students, 25 course units		MAE	RMSE	Running time(sec)
memory-based	User-based	1,33	1,53	0,3
	Item-based	0,75	0,92	0,09
model-based	SVD	0,75	0,91	0,2

Tabelle 8.9: Anzahl Objekte größer als Anzahl Benutzer

diesen Ansatz für der Datensatz nicht geeignet ist. Im Sinne der Laufzeit, war der Objekt-basierter Ansatz der schnellste. Was die Genauigkeit der Vorhersage betrifft, lieferte den Model-basierten Ansatz das beste Ergebnis. Bei Anzahl an Objekte größer als Benutzer lieferte auch der Objekt-basierter Ansatz ein vergleichbares Ergebnis. Betrachtet man die Performanz und die Vorhersagegenauigkeit, ist der Objekt-basierter Ansatz mehr für den verwendete Datensatz geeignet. Außerdem, sorgt dieser Ansatz für mehr Vielfalt in das Ergebnis der Empfehlung.

Während der Evaluierung wurde auch die Feature *learningDuration in %* und *numberOf-Posts* kombiniert, um der Einfluss von einem neuen Feature in das Empfehlungssystem zu bewerten. Diese Kombination hatte dazu geführt, dass die Genauigkeit der Vorhersage verschlechtert war. Ein Grund dafür könnte sein dass die Kombinierung dieser Features durch die Addition ungeeignet ist.

Bei der Evaluierung der automatisierten Zerlegung der Aufzeichnung wurde 3 Engpässe gefunden. Manche Szenen wurden nicht richtig erkannt und manche zwei oder drei Mal. Dies hatte zur Folge, dass einige aus der Analyse resultuerende Lerneinheiten einen längeren Übertragungsdauer aufweisen. Eine Möglichkeit für die Optimierung wäre die Erkennung der Szeneänderung innerhalb einer Sequenz vom Whiteboard-Szene mithilfe der Bildklassifikation durchzuführen. Dadurch könnte leeres Whiteboard erkannt werden und die Videoteile zwischen zwei leeres Whiteboards als ein einzelnen Szene betrachtet. Außerdem wurde bei der Performanzevaluierung festgestellt, dass die meiste Analysezeit im Teilprozess Visuell Feature Extraktion verbraucht wird. Eine Möglichkeit dieses Pro-

blem zu beseitigen wäre anstatt alle Videobilder zu lesen, nur Bilder in einer zeitliche Intervalle zu lesen. Zum Schluss wurde festgestellt, dass das Schneiden der Videos mit dem FFmpeg-Programm einen Schwarzbild am Anfang manche Szenen darstellte. Hier könnte noch anderen Konfigurationsparameters dieses Programm ausprobiert werden. Obwohl das Empfehlungssystem auf Basis der generierten LMS interaktionsdaten evaluiert werden konnte, waren die generierte Daten nicht vielfältig und manche Benutzer hatten mit Benutzer andere Profile korreliert. Der Grund dafür ist das die Korrelation eher ein Verhalten ist und es kann aufgrund von Zufallsdaten Benutzer generiert werden, die das selbe Verhalten mit Benutzer anderer Profile hat.

9 Fazit und Ausblick

Im Rahmen dieser Thesis wurden eine systematische Literaturrecherche in Forschungsgebiete der *Videozusammenfassung* und *Empfehlungssystem* durchgeführt. Dabei wurde festgestellt, dass es in beiden Gebieten eine Vielzahl an Publikationen gibt, jedoch nur wenige adressieren der Vorlesung- oder Leraning Kontext. Ebenso wurden Verwandte Arbeiten in genannten Forschungsfelder thematisiert. Dabei wurde festgestellt, dass es bereits Ansätze existieren, die manche der definierten Forschungsfrage versuchen eingehen. Auf Grundlage dieser Recherche wurde eine Lösung konzipiert, welche aus zwei Prozesse zusammensetzt.

Der erste Prozess war die *automatisierte Zerlegung der Vorlesungsaufzeichnungen*. Für diesen Prozess wurde die Techniken der statischen Videozusammenfassung und audiovisuelle Szeneanalyse eingesetzt um Szeneänderungen zu erkennen. Der wesentliche Vorteile dieser Lösung gegenüber bestehende Ansätze, ist dass sie kein Wissen über die verwendete Sprache in der Vorlesungssvideo benötigt.

Der zweite Prozess war das *Empfehlungssystem für Lerneinheiten*. Hierbei wurde unterschiedliche Ansätze des kollaborativen Filtern entwickelt. Dabei wurde die *Top-N Empfehlung* als Methode der Empfehlung von Lerneinheiten ausgewählt. Als Schnittstelle zwischen beide Prozesse wurde LMS Interaktionsdaten generiert, wobei Annahmen getroffen wurden. Das entwickelte System wurde im Kapitel 8 evaluiert. Dabei wurde die Performance der einzelnen Teilprozesse geprüft. Außerdem wurde beim ersten Prozess die Erkennungsgenauigkeit der Szenen geprüft und die resultierende eingegangen. Beim zweiten Prozess wurde die Vorhersagegenauigkeit der Benutzerbewertung der verschiedene kollaborative Ansätze geprüft und auf Basis deren Ergebnis der Objekt-basierter Ansatz als geeignete Modell ausgewählt.

Die Evaluierung hat gezeigt, dass das System einige Einschränkungen unterliegen und hat die im selben Kapitel erwähnt schwäche. Aufgrund der zeitlichen Rahmenbedingungen konnte das System nicht in größerem Umfang auf die Qualität und Mehrwert überprüft werden. Hierfür würde sich an der Hochschule eine Online-Evaluierung des Ergebnis des Objekt-basierten Ansatzes eignen, um die Benutzerzufriedenheit auf die

präsentierten Empfehlungen zu messen. Diese kann in Form einer Befragung der Studierende sein. Des Weiteren könnte die Studierende über die Gebrauchstauglichkeit der erstellte Lerneinheiten befragt werden. Um die Qualität der Zerlegung zu erhöhen, könnte die leere Whiteboards in Vorlesungsvideo durch die Bildklassifikation erkannt werden. Außerdem könnte eine maximale Dauer für die Lerneinheiten definiert werden. Als weiter Forschungsarbeit könnte die Integration des entwickelte Empfehlungssystem in das LMS der Hochschule sein.

Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie alle wörtlich oder sinngemäß übernommenen Stellen in der Arbeit gekennzeichnet habe. Die Arbeit wurde noch keiner Kommission zur Prüfung vorgelegt und verletzt in keiner Weise Rechte Dritter

Waiblingen, den 12. Dezember 2021 [UNTERSCHRIFT]

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Björn Bohnenkamp, Katja G. Marcus Burkhardt B. Marcus Burkhardt: Online-Lehre 2020 – Eine medienwissenschaftliche Perspektive. In: *Hochschulforum Digitalisierung*, 2020

[2] Romero, Cristobal ; Ventura, Sebastian: Educational data mining: A survey from 1995 to 2005. In: *Expert systems with applications* 33 (2007), Nr. 1, S. 135–146

[3] Brocke, Jan v. ; Simons, Alexander ; Niehaves, Bjoern ; Niehaves, Bjorn ; Reimer, Kai ; Plattfaut, Ralf ; Cleven, Anne: Reconstructing the giant: On the importance of rigour in documenting the literature search process. (2009)

[4] Fu, Dan ; Liu, Qingtang ; Zhang, Si ; Wang, Jianhu: The Undergraduate-Oriented Framework of MOOCs Recommender System. In: *2015 International Symposium on Educational Technology (ISET)*, 2015, S. 115–119

[5] Bousbahi, Fatiha ; Chorfi, Henda: MOOC-Rec: A Case Based Recommender System for MOOCs. In: *Procedia - Social and Behavioral Sciences* 195 (2015), 1813-1822. http://dx.doi.org/https://doi.org/10.1016/j.sbspro.2015.06.395. – DOI https://doi.org/10.1016/j.sbspro.2015.06.395. – ISSN 1877–0428. – World Conference on Technology, Innovation and Entrepreneurship

[6] Gerhard, David ; Heidkamp, Paula ; Spinner, Alexandra ; Sommer, Bianca ; Sprick, Anika ; Simonsmeier, Bianca A. ; Schneider, Michael: Vorlesung. In: *Gute Hochschullehre: Eine evidenzbasierte Orientierungshilfe*. Springer, 2015, S. 13–38

[7] Jain, Harshit ; Anika: Applying Data Mining Techniques for Generating MOOCs Recommendations on the Basis of Learners Online Activity. In: *2018 IEEE 6th International Conference on MOOCs, Innovation and Technology in Education (MITE)*, 2018, S. 6–13

[8] Romero, Cristóbal ; Ventura, Sebastián ; García, Enrique: Data mining in course management systems: Moodle case study and tutorial. In: *Computers & Education* 51 (2008), Nr. 1, S. 368–384

[9] Gulzar, Zameer ; Leema, A A. ; Deepak, Gerard: Pcrs: Personalized course recommender system based on hybrid approach. In: *Procedia Computer Science* 125 (2018), S. 518–524

[10] AlMousa, Mohannad ; Benlamri, Rachid ; Khoury, Richard: NLP-Enriched Automatic Video Segmentation. In: *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)*, 2018, S. 1–6

[11] Kota, Bhargava U. ; Stone, Alexander ; Davila, Kenny ; Setlur, Srirangaraj ; Govindaraju, Venu: Automated Whiteboard Lecture Video Summarization by Content Region Detection and Representation. In: *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, S. 10704–10711

[12] Vimalaksha, Anusha ; Vinay, Siddarth ; Prekash, Abhijit ; Kumar, N. S.: Automated Summarization of Lecture Videos. In: *2018 IEEE Tenth International Conference on Technology for Education (T4E)*, 2018, S. 126–129

[13] Xu, Fei ; Davila, Kenny ; Setlur, Srirangaraj ; Govindaraju, Venu: Content Extraction from Lecture Video via Speaker Action Classification Based on Pose Information. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, S. 1047–1054

[14] Bora, Amit ; Sharma, Shanu: A Review on Video Summarization Approcahes : Recent Advances and Directions. In: *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, 2018, S. 601–606

[15] In: Ngo, Chong-Wah ; Wang, Feng: *Video Summarization*. Boston, MA : Springer US, 2009. – ISBN 978–0–387–39940–9, 3320-3324

[16] Kini M., Mahesh ; Pai, Karthik: A Survey on Video Summarization Techniques. In: *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)* Bd. 1, 2019, S. 1–5

[17] Hou, Xiaodi ; Harel, Jonathan ; Koch, Christof: Image Signature: Highlighting Sparse Salient Regions. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (2012), Nr. 1, S. 194–201. http://dx.doi.org/10.1109/TPAMI.2011.146. – DOI 10.1109/TPAMI.2011.146

[18] Kühn, Benjamin: *Interessengetriebene audiovisuelle Szenenexploration*. Bd. 22. KIT Scientific Publishing, 2016

[19] Adomavicius, G. ; Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. In: *IEEE*

Transactions on Knowledge and Data Engineering 17 (2005), Nr. 6, S. 734–749. http://dx.doi.org/10.1109/TKDE.2005.99. – DOI 10.1109/TKDE.2005.99

[20] Burke, Robin: Hybrid recommender systems: Survey and experiments. In: *User modeling and user-adapted interaction* 12 (2002), Nr. 4, S. 331–370

[21] Engelbert, Benedikt: *Maschinelle Generierung von Empfehlungen zur Lehr-/Lernunterstützung im Hochschulkontext*, Universität Osnabrück, Diss., 2016

[22] Aggarwal, Charu C. u. a.: *Recommender systems*. Bd. 1. Springer, 2016

[23] Herlocker, Jonathan L. ; Konstan, Joseph A. ; Terveen, Loren G. ; Riedl, John T.: Evaluating collaborative filtering recommender systems. In: *ACM Transactions on Information Systems (TOIS)* 22 (2004), Nr. 1, S. 5–53

[24] Guo, Yaqiong ; Huang, Mengxing ; Lou, Tao: A Collaborative Filtering Algorithm of Selecting Neighbors Based on User Profiles and Target Item. In: *2015 12th Web Information System and Application Conference (WISA)*, 2015, S. 9–14

[25] Ricci, Francesco ; Rokach, Lior ; Shapira, Bracha: Introduction to recommender systems handbook. In: *Recommender systems handbook*. Springer, 2011, S. 1–35

[26] Koren, Yehuda ; Bell, Robert ; Volinsky, Chris: Matrix Factorization Techniques for Recommender Systems. In: *Computer* 42 (2009), Nr. 8, S. 30–37. http://dx.doi.org/10.1109/MC.2009.263. – DOI 10.1109/MC.2009.263

[27] *How you can build simple recommender systems with Surprise*. https://towardsdatascience.com/how-you-can-build-simple-recommender-systems-with-surprise-b0d32a8e4802. – [Online; Zugriff am 27. Oktober 2021]

[28] Sicilia, Miguel Ángel ; García-Barriocanal, Elena ; Sánchez-Alonso, Salvador ; Cechinel, Cristian: Exploring user-based recommender results in large learning object repositories: the case of MERLOT. In: *Procedia Computer Science* 1 (2010), Nr. 2, 2859-2864. http://dx.doi.org/https://doi.org/10.1016/j.procs.2010.08.011. – DOI https://doi.org/10.1016/j.procs.2010.08.011. – ISSN 1877–0509. – Proceedings of the 1st Workshop on Recommender Systems for Technology Enhanced Learning (RecSysTEL 2010)

[29] Lu, Jie: A personalized e-learning material recommender system. In: *International Conference on Information Technology and Applications* Macquarie Scientific Publishing, 2004

[30] Oard, Douglas W. ; Kim, Jinmook u. a.: Implicit feedback for recommender systems. In: *Proceedings of the AAAI workshop on recommender systems* Bd. 83 WoUongong, 1998, S. 81–83

[31] Jawaheer, Gawesh ; Szomszor, Martin ; Kostkova, Patty: Comparison of implicit and explicit feedback from an online music recommendation service. In: *proceedings of the 1st international workshop on information heterogeneity and fusion in recommender systems*, 2010, S. 47–51

[32] Bligh, Donald A.: *What's the Use of Lectures?* Intellect books, 1998

[33] Schmidt, Henk G. ; Wagener, Stephanie L. ; Smeets, Guus A. ; Keemink, Lianne M. ; Molen, Henk T. d.: On the use and misuse of lectures in higher education. In: *Health Professions Education* 1 (2015), Nr. 1, S. 12–18

[34] Lauer, Tobias ; Trahasch, Stephan: Begriffsbesprechung: Vorlesungsaufzeichnung:. 4 (2005), Nr. 3, 61. http://dx.doi.org/doi:10.1524/icom.2005.4.3.61. – DOI doi:10.1524/icom.2005.4.3.61

[35] Jadon, Shruti ; Jasim, Mahmood: Video summarization using keyframe extraction and video skimming. In: *arXiv preprint arXiv:1910.04792* (2019)

[36] *A Gentle Introduction to k-fold Cross-Validation*. https://machinelearningmastery.com/k-fold-cross-validation/. – [Online; Zugriff am 30. Oktober 2021]

[37] Pedregosa, Fabian ; Varoquaux, Gaël ; Gramfort, Alexandre ; Michel, Vincent ; Thirion, Bertrand ; Grisel, Olivier ; Blondel, Mathieu ; Prettenhofer, Peter ; Weiss, Ron ; Dubourg, Vincent u. a.: Scikit-learn: Machine learning in Python. In: *the Journal of machine Learning research* 12 (2011), S. 2825–2830

[38] Pérez, Fernando ; Granger, Brian: *The Jupyter Notebook*. https://jupyter.org/. – [Online; Zugriff am 13. Oktober 2021]

[39] Jetbrains: *The Python IDE for Professional Developers*. https://www.jetbrains.com/pycharm/. – [Online; Zugriff am 13. Oktober 2021]

[40] Culjak, Ivan ; Abram, David ; Pribanic, Tomislav ; Dzapo, Hrvoje ; Cifrek, Mario: A brief introduction to OpenCV. In: *2012 Proceedings of the 35th International Convention MIPRO*, 2012, S. 1725–1730

[41] Hug, Nicolas: Surprise: A Python library for recommender systems. In: *Journal of Open Source Software* 5 (2020), Nr. 52, 2174. http://dx.doi.org/10.21105/joss.02174. – DOI 10.21105/joss.02174

[42] PROJECT, The P.: *Package overview*. https://pandas.pydata.org/pandas-docs/stable/getting_started/overview.html. – [Online; Zugriff am 10. Oktober 2021]

[43] COMMUNITY, The S.: *SciPy community*. https://docs.scipy.org/doc/scipy/reference/tutorial/general.html. – [Online; Zugriff am 13. Oktober 2021]

[44] TEAM, Matplotlib development: *Matplotlib: Visualization with Python*. https://matplotlib.org/. – [Online; Zugriff am 13. Oktober 2021]

[45] *User Guide*. https://zulko.github.io/moviepy/. – [Online; Zugriff am 13. Oktober 2021]

[46] *CSV File Reading and Writing*. https://docs.python.org/3/library/csv.html. – [Online; Zugriff am 13. Oktober 2021]

[47] *Keras: simple, flexibel, powerful*. https://keras.io/. – [Online; Zugriff am 13. Oktober 2021]

[48] *A complete, cross-platform solution to record, convert and stream audio and video*. https://www.ffmpeg.org/. – [Online; Zugriff am 13. Oktober 2021]