

**Automatisierte Zerlegung von Vorlesungsaufzeichnungen zur Realisierung eines Empfehlungssystem für Lerneinheit 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**

1. [Einleitung](#_bookmark0) 3
   1. [Problemstellung](#_bookmark1) 3
   2. [Zielsetzung und Einordnung der Arbeit](#_bookmark2) 3
   3. [Aufbau der Arbeit](#_bookmark3) 4
2. [Literaturrecherche und verwandte Arbeit](#_bookmark4) 6
   1. [Methode der Literaturrecherche](#_bookmark5) 6
   2. [Vorgehensweise](#_bookmark6) 6
   3. [Ergebnis der Literaturrecherche](#_bookmark9) 9
   4. [Verwandte Arbeiten](#_bookmark11) 11
      1. [Empfehlungssystem für E-learning](#_bookmark12) 11
      2. [Videozusammenfassung](#_bookmark13) 12
3. [Videozusammenfassung und Audiovisuelle Szeneexploration](#_bookmark14) 13
   1. [Videozusammenfassung](#_bookmark15) 13
   2. [Audiovisuelle Szeneexploration](#_bookmark17) 14
4. [Empfehlungssysteme](#_bookmark18) 16
   1. [Kollaboratives Filtern](#_bookmark20) 18
      1. [Objekt-basiertes kollaboratives Filtern](#_bookmark21) 18
      2. [Benutzer-basiertes kollaboratives Filtern](#_bookmark24) 20
      3. [Nachteile von Kollaboratives Filtern](#_bookmark26) 21
      4. [Implizit und explizit Feedback](#_bookmark27) 21
   2. [Andere Empfehlungssysteme](#_bookmark28) 22
   3. [Evaluierung der Empfehlungssysteme](#_bookmark29) 23
      1. [Genauigkeitsmetriken](#_bookmark30) 23
      2. [Klassifikationsmetriken](#_bookmark31) 24
5. [Empfehlungssystem für die Online-Vorlesung](#_bookmark36) 26
   1. [Empfehlungssystem im Kontext Vorlesung](#_bookmark37) 26

|  |  |  |
| --- | --- | --- |
|  | [5.2 Online-Vorlesung](#_bookmark38) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 28 |
| [**6**](#_bookmark39) | [**Konzeption**](#_bookmark39) | **29** |
|  | [6.1 Systemüberblick](#_bookmark40) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 29 |
|  | [6.2 Systemarchitektur](#_bookmark42) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 30 |
|  | [6.3 MP4 Video Input](#_bookmark44) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 31 |
|  | [6.4 Visuell-Features](#_bookmark45) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 31 |
|  | [6.5 Audio-Features](#_bookmark46) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 32 |
|  | [6.6 Erkennung von interessante Szeneänderungen](#_bookmark47) . . . . . . . . . . . . . . . . | 32 |
|  | [6.7 Szeneklassifizierung](#_bookmark48) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 32 |
|  | [6.8 Scheiden des Videos](#_bookmark49) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 33 |
|  | [6.9 Interaktionsdaten vom LMS](#_bookmark50) . . . . . . . . . . . . . . . . . . . . . . . . . . | 33 |
|  | [6.10 Kollaboratives Filtern](#_bookmark51) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 34 |
|  | [6.10.1 Data Understanding](#_bookmark52) . . . . . . . . . . . . . . . . . . . . . . . . . . | 34 |
|  | [6.10.2 Data preparation](#_bookmark54) . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 35 |
|  | [6.10.3 Modeling](#_bookmark55) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 35 |
|  | [6.10.3.1 Benutzer-basierte kollaboratives Filtern](#_bookmark56) . . . . . . . . . . | 36 |
|  | [6.10.3.2 Item-Based](#_bookmark58) . . . . . . . . . . . . . . . . . . . . . . . . . . | 37 |
|  | [6.10.4 Evaluation](#_bookmark60) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 37 |
| [**7**](#_bookmark61) | [**Implementierung**](#_bookmark61) | **39** |
|  | [7.1 Technologien](#_bookmark62) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 39 |
|  | [7.1.1 Python](#_bookmark63) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 39 |
|  | [7.1.2 Jupyter Notebook](#_bookmark64) . . . . . . . . . . . . . . . . . . . . . . . . . . . | 39 |
|  | [7.1.3 Pycharm](#_bookmark65) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 40 |
|  | [7.1.4 OpenCV](#_bookmark66) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 40 |
|  | [7.1.5 Surprise](#_bookmark67) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 40 |
|  | [7.1.6 Pandas](#_bookmark68) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 40 |
|  | [7.1.7 NumPy und SciPy](#_bookmark69) . . . . . . . . . . . . . . . . . . . . . . . . . . . | 41 |
|  | [7.1.8 Seaborn und Matplotlib](#_bookmark70) . . . . . . . . . . . . . . . . . . . . . . . . | 41 |
|  | [7.1.9 Scikit-learn](#_bookmark71) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 41 |
|  | [7.1.10 Moviepy](#_bookmark72) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 41 |
|  | [7.1.11 CSV](#_bookmark73) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
|  | [7.1.12 Keras](#_bookmark74) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
|  | [7.1.13 FFmpeg](#_bookmark75) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
|  | [7.2 Daten](#_bookmark76) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
|  | [7.3 Struktur des Pakets](#_bookmark77) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |

* 1. [Visuell Feature-Extraktion](#_bookmark79) 43
  2. [Audio Feature-Extraktion](#_bookmark80) 46
  3. [Erkennung von interessante SzeneÃ€nderung](#_bookmark81) 47
  4. [Szeneklassifizierung](#_bookmark82) 48
  5. [Schneiden der Videos](#_bookmark83) 48
  6. [Interaktionsdaten vom LMS](#_bookmark84) 49
  7. [Kollaboratives Filtern](#_bookmark85) 50
     1. [Data Unterstanding](#_bookmark86) 50
     2. [Data Preparation](#_bookmark92) 53
     3. [Modeling](#_bookmark94) 55
        1. [User-Based](#_bookmark95) 55
        2. [Item-Based](#_bookmark102) 60
        3. [Modell-based](#_bookmark105) 61
     4. [Evaluation](#_bookmark106) 63

##### [Evaluation](#_bookmark107) 65

##### [Fazit und Ausblick](#_bookmark108) 66

ASR Automatic Speech Recognition CBR Case Based Reasoning

CNN Convolutional Neural Network

CRISP Cross-Industry Standard Process for data mining CSV Comma-separated Values

DCT Discrete Cosine Transform HSV Hue Saturation Value

KNN K-Nearest Neighbors-Algorithm 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

Die Anzahl der Online-Vorlesungen ist durch die Corona-Pandemie sehr stark gestiegen. Viele Hochschulen haben die klassische Präsenzlehre auf digitales Format umgestellt. Dadurch lässt sich für den Studierenden ein zeitlicher und räumlicher Vorteil gewinnen. Die Online Lehre bringt zwar neu Impulse, erfordert jedoch aufwand: bei Lehrende kos- tet die Erzeugung von Vorlesungsvideos viel Zeit durch das Schneiden und Überarbeiten der Aufzeichnungen. Da die meisten Vorlesungsvideo eine längere Übertragungsdauer aufweisen, erschwert dies den studierenden relevanten Inhalten zu finden oder durchsu- chen. Es ist auch der Fall, wenn die Vorlesungsaufzeichnung in viele kleine Lernvideo zerlegt wird.

In dieser Arbeit werden Ansätze für die automatische Zerlegung von Vorlesungsaufzeich- nungen und Empfehlung von Lerninhalten untersucht. Das Ziel ist die Minimierung der Aufwand bei der Erstellung von Vorlesungsvideo für Lehrende und die Erleichterung für Studierende relevante Inhalte zu finden.

Die automatische Zerlegung der Aufzeichnungen besteht aus vier Schritte. Im ersten Schritt findet die Feature Extraktion statt. In diesem Schritt werden audiovisuelle Fea- tures wie silence und Fragehistogramm extrahiert. Im zweiten Schritt werden Szene- transitionen detektiert und die Score Funktion berechnet. Im dritten Schritt werden die Szenen mit Hilfe neuronalen Netzwerk klassifiziert. Im vierten Schritt erfolgt die Auf- schneidung der Szenen mittels ffmpeg und die Generierung einer Vorlesungsskript, die als Eingang für das Empfehlungssystem benötig wird.

Das Empfehlungssystem wird mittels Collaborative Filtering realisiert. Für den Test werden Benutzer mit implizit und explizit Feedback Daten generiert.

## Problemstellung

Die Anzahl der Online-Vorlesungen ist durch die Corona-Pandemie stark gestiegen [[12](#_bookmark119)]. Die Studierenden gewinnen dadurch eine hohe Flexibilität. Sie können sich die Lerninhalte 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 aufgezeichnet. Dann wird die Aufzeichnung manuell aufgeschnitten und überarbeitet, bevor die resultierenden Lerninhalte 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 automatische 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. Die LMS mit integrierten Empfehlungssystem werden dafür als Lösung angesehen [[54](#_bookmark152)]. Diese Systeme versuchen den Lernenden eine individuelle Bildung anzubieten, indem sie diesen auf Grundlage ihrer Ziele, Fähigkeiten und Präferenz Lerninhalte 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 [[54](#_bookmark152)].

## Zielsetzung und Einordnung der Arbeit

Das Ziel dieser Arbeit ist es, Vorlesungsaufzeichnungen automatisch 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 wer- den:

* 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 Empfehlungssystem für die Lerneinheiten der Online-Lehre?
* Wie können Benutzer mit unterschiedlichen Profilen für den Test von kollaboratives Filtern erzeugt werden?
* Welche Methoden und Techniken aus dem Bereich Statistik oder Maschinelles eignen sich für die Evaluierung eines Empfehlungssystems?

In Rahmen dieser Arbeit wird kein Softwareprodukt entwickelt, sondern es wird ein System experimentiert. Außerdem ist eine inhaltliche Analyse der Vorlesungsaufzeichnung ist nicht Teil dieser Arbeit. Des Weiteren basiert sich das Empfehlungssystem in dieser Arbeit auf kollaboratives Filtern.

Die vorliegende Thesis wird in zwei wissenschaftliche Forschungsgebiete eingeordnet. Aufgrund des engen Zusammenhangs mit *Videozusammenfassung* ordnet sich thema- tisch der Teilprozess der automatisierten Zerlegung der Aufzeichnungen im Gebiet der *Videozusammenfassung* ein. Was der Teilprozess des Empfehlungssystem angeht, reiht sich dieser Thesis im Forschungsfeld des *Empfehlungssystems im Kontext E-Learning* ein.

## 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 und dieser Arbeit thematisch in entsprechenden wissenschaftlichen Kontexten eingeordnet. Im Kapitel 2 wird eine systematische Literaturrecherche durchgeführt. Dies beinhaltet die Methode der Literaturrecherche der Forschungsfragen und die Vorgehensweise. Im selben Kapitel werden aktuelle wissenschaftliche Recherche zu den genannten Forschungsgebieten eingegangen. Dort werden die wesentlichen Konzepte in die 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 Empfehlungssysteme beschrieben. Dabei wird konkret auf das kollaboratives Filtern eingegangen, die eingesetzte Methoden und Annahmen, die dabei getroffen sind und die Evaluationsmethoden. Im Kapitel 5 wird versucht die vorliegende Arbeit thematisch genauerer in Gebiet E-Learning einzuordnen. Es werden einige Annahmen und Methoden aus dem vor Kapitel betrachtet und im Kontext der Online-Vorlesung adaptiert. Der Kapitel 6 beschäftigt sich mit dem Systementwurf. Dies beinhaltet die Systeminfrastruktur, Architektur. Ebenfalls wird das Konzept zu den hervorgehobenen Problemen in dieser Thesis vorgestellt. Das Kapitel 7 stellt die eigentliche Implementierung dar. Hierzu gehört die Vorstellung der verwendeten Frameworks, die Umsetzung der Aufzeichnung Segmentierung und des Empfehlungssystems. In Kapitel 8 wird die Evaluation für der entwickelte Ansatz durchgeführt. Hier wird insbesondere das Leistungsverhalten geprüft. Schließlich wird im Kapitel 9 die Arbeit kurz zusammengefasst, das Ergebnis vorgestellt und ein Ausblick aufgezeigt.

# Literaturrecherche und verwandte Arbeit

In diesem Kapitel wird eine systematische Literaturrecherche durchgeführt. Dies bein- haltet die Methode der Literaturrecherche und die Vorgehensweise. Dabei werden die wissenschaftlichen Beiträge in der Themengebiete hervorgehoben.

## Methode der Literaturrecherche

Zur Bestimmung der aktuelle Stand der Technik wird am Anfang eines wissenschaftlichen Arbeitsprozesses eine systematische Literaturrecherche durchgeführt [[16](#_bookmark122)]. 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](#_bookmark7)). 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.

## Vorgehensweise

Wie im vorherigen Abschnitt erwähnt, wird in der ersten Phase der Literaturrecherche Forschungsfragen definiert. In der Einleitung erwähnt wurde in jedem Forschungsgebiet Forschungsfragen definiert. Im Gebiet Videozusammenfassung wurden drei Fragen definiert:

* Welche Methoden und Techniken aus dem Bereich maschinelles Lernen sind für die audiovisuelle Szeneexploration eines Vorlesungsvideos geeignet?

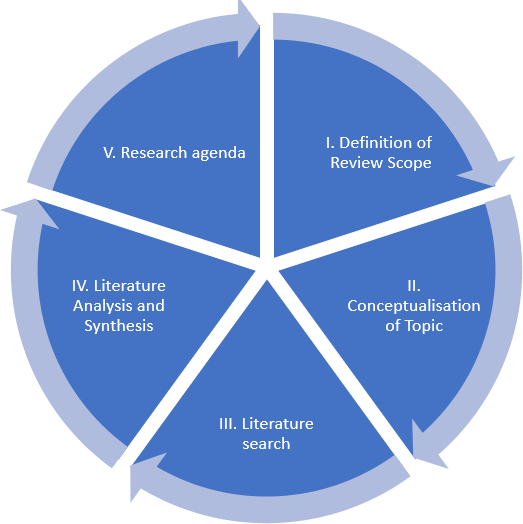


Abbildung 2.1: Methode der Literaturrecherche (Quelle: In Anlehnung an [[16](#_bookmark122)])

Diese Frage fokussiert sich auf die audiovisuelle Analyse der Aufzeichnung mit Hilfe von maschinelles Lernen Verfahren.

* Wie kann die Interessantheit einer Szeneänderung in der audiovisuelle Szeneanalyse definiert und bewertet werden?

In diese Frage liegt der Schwerpunkt auf die Erkennung von Szeneänderung in einer Vorlesungsaufzeichnung und die Bewertung der Interessantheit.

* Welche Tools sind für die Videosegmentierung geeignet?

Diese Frage befasst sich mit Tools, mithilfe deren erkannte Schnittpunkte in Video aufgeschnitten werden können.

Was den Gebiet Empfehlungssystem betrifft, werden ebenfalls drei Fragen erarbeitet:

* Welche Methoden und Techniken aus dem Bereich Statistik oder maschinelles eignen sich für die Realisierung eines Empfehlungssystem für E-Learning?

Die Frage fokussiert sich auf Techniken für die Implementierung eines Empfehlungssystem für der Kontext Online-Vorlesung.

* Wie können Benutzer mit unterschiedlichen Profilen für den Test von kollaboratives Filtern simuliert werden?

In dieser Farge geht es um die Generierung von Datensätze um die Qualität des implementierten Empfehlungssystem evaluieren zu können.

* Welche Methoden und Techniken aus dem Bereich Statistik oder Maschinelles eignen sich für die Evaluierung eines Empfehlungssystems?

Hier geht es um die Messung der Qualität Empfehlungssystems.

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

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

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 genannte Datenbanken. Bei der ersten Suche mit deutschen Schlüsselworten wird schnell aufgrund der wenigen Trefferquote festgestellt, dass die Sprache für die Recherche über die Themen nicht geeignet ist. Für die weitere Suche wird Englisch verwendet. Dafür werden englische Schlüsselwörter von Forschungsfragen abgeleitet. Diese suche ergibt eine sehr hohe Trefferquote (Abbildung [2.1](#_bookmark8)). Es lässt da- durch begründen, dass viele wissenschaftliche Arbeiten in Englisch publiziert sind. Die Trefferquote in IEEE Xplore ist deutlich wenig verglichen mit den anderen beiden Datenbanken. Eine Erklärung dafür könnte sein, dass IEEE Xplore sich 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 Literaturen 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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Suchb  numme | egSriuffcsh-e  r | 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

Nach Einsatz des oben beschriebenen Filters sinkt die Trefferliste signifikant ab. In der vierten Phase wird die verbleibende Trefferliste nach Relevanz geprüft. Diese erfolgt zunächst durch das Lesen der Abstrakte. Papers mit relevanten Abstracts werden voll- ständig weitergelesen. In der fünften Phase werden interessante Papers in der Agenda über- tragen.

## Ergebnis der Literaturrecherche

Die Tabelle [2.2](#_bookmark10) stellt das Ergebnis der Recherche dar. Es ist festzustellen, dass am An- fang der Suche die Trefferquote sehr hoch ist. Nach Anwendung von beschriebenen Filters 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 relevante Papers und für das Empfehlungssysteme 7. Diese Papers werden als Grundstein für die Bearbeitung der Vorliegende Thesis angesetzt. Diese wurden Während der Bearbeitung dieser Thesis durch *Forward search* und *backward* erweitert*.*

*2*

10

Tabelle 2.2: Ergebnis der Literaturrecherche

0

0

0

8

10

20

80

8

2

5

7

10

14

40

50

7

0

2

2

40

501

1897

20197

6

2

5

7

89

2507

7234

8727

5

1

5

6

60

275

1063

5380

4

1

2

5

13187

20834

73799

3

2

4

30

60

792

1200

2874

2

1

3

10

2150

3003

13886

17964

1

Springer

Link

0

0

0

4

7

11

15

8

2

5

10

111

524

1224

1455

7

1

3

-

5

6

12

14

5

0

0

5

9

30

98

128

4

0

1

5

176

483

2007

2530

3

1

2

8

10

19

39

48

2

1

3

-

5

18

66

70

1

IEEE Xplore

0

0

0

14

100

500

1010

8

1

4

6

439

43559

767682

1116176

7

1

2

3

45

1016

27181

56449

5

1

2

8

10

50

80

100

4

1

1

14

84

2257

4695

57817

3

1

3

10

100

1430

55898

85389

2

1

0

0

6

40

749

1283

1

ScienceDirect

Relevant

Relevant

nach Lesen des Abstrakts

Schlüsselwort

im Titel enthalten

Themengebiet

Zeitraum

2017-2021

Publikations-

form Paper

Anzahl

Treffer

Suchbegriffs-

nummer

Datentank

*Literaturrecherche und verwandte Arbeit*

## Verwandte Arbeiten

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

### Empfehlungssystem für E-learning

Dan Fu et al. präsentieren ein Ansatz zur Analyse der Vorlesungsvideo basierend auf dem Inhalt. Die Autoren verwenden Schlüsselbildextraktion, um das Input-Video zu segmentieren. Die Frames werden in bestimmte Zeitintervall gelesen. Mithilfe der OCR- Technologie, die auf Schlüsselbilder angewendet wird, werden Schlüsselwörter zu extrahieren. Die Autoren verwenden ASR-Techniken, um textuelle Features aus der Audiospur zu extrahieren. Auf Grundlage der gefundenen Schlüsselwörter werden Weblinks, Bild- links 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 [[24](#_bookmark130)].

Fatiha Bousbahi und Henda Chorfi schlagen ein MOCC-Empfehlungssystem vor, welches mit CBR-Ansatz und Informationssuchtechniken realisiert ist. Auf Basis der Lernprofile, Bedürfnisse und Wissensstandes der Lernenden, erstellt das System die passende Empfehlung. CBR ist eine Art Inhaltsbasiertes Empfehlungssystem, die als Basis für die Beschreibung des Empfehlungsobjekts verwendet [[15](#_bookmark123)].

Harshit Jain und Anika schlagen eine Methode zur Entwicklung eines MOOC-Empfehlungssystem unter Anwendung von Data Mining Techniken wie Random Forest, Classification Tree

und K-Nearest Neighbors. Die Autoren klassifizieren zunächst die Benutzer in zwei Kategorien Anhang ihrer Aktivitätsprotokolle: aktive und passive Lernende. MOOCs sind eine neue Tendenz für Hochschullehrer ihre Vorlesungsvideos kostenfrei oder kostengünstig einer großen Anzahl an Studierende weltweit im Internet zugänglich zu machen [[25](#_bookmark131)]. Nach Anwendung von Data Mining Ansätze geben wird die Kursempfehlung separat für jede Kategorie gegeben [[33](#_bookmark138)].

Cristóbal Romero et al. verwenden das Data Mining Tool Weka auf das LMS Moodle 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 [[55](#_bookmark154)].

Zameer Gulzar et al. schlagen eine Hybride Methodologie zusammen mit Ontologie vor, um personalisierte Kursempfehlung basierend auf Benutzerpräferenz abzugeben [[26](#_bookmark132)]. Die Ontologie ist eine Möglichkeit, die Eigenschaften eines Objekts und ihre Beziehungen zueinander darzustellen, indem eine Reihe von Konzepten und Kategorien definiert wird, die das Objekt repräsentieren.

### Videozusammenfassung

In [[10](#_bookmark118)] wird einen 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 für eine verbesserte Segmentierung dienen. Anschließend wird NLP-Kosinusähnlichkeit auf die Cluster angewendet, um verschiedene Themen in Video zu identifizieren. Für die textuelle und Audio Feature- Extraktion setzten die Autoren ASR und OCR ein.

In seinem Beitrag, [[39](#_bookmark142)] schlagt eine Methode für die Zusammenfassung von Whiteboard- Video vor, indem Feature-Repräsentationen von erkannten handschriftlichen Inhaltre- gionen extrahiert werden, um die eindeutigen Inhalte zu bestimmen. Histogramms von Gradienten wird dabei verwendet, um die erkannten Regionen darzustellen.

In [[58](#_bookmark156)] wird eine Methode zur Segmentierung des Videos auf der Grundlage der zuvor notierte Beginn und Ende jedes Themas beschrieben. Anschießen werden die einzelnen Themen auf Basis der Audio-Features wird zusammengefasst.

In [[61](#_bookmark158)]wird eine Methode zur Extraktion von Inhalten aus Vorlesungsvideos unter Verwendung der Sprecher-Aktionen gezeigt. Jedes Vorlesungsvideo wird in kleine zeitliche Einheiten unterteilt, die als Aktionssegmente bezeichnet wird. Unter Verwendung von Pause, werden Körper- und Handskelett-Daten extrahiert und verwendet, um bewegungsbasierte Merkmale für jedes Segment zu berechnen. Danach wird die dominante Sprecher-Aktion jedes Segment mit Random Forests und dem bewegungsbasierten Merk mal klassifiziert.

# 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 zur Videozusammenfassung und audiovisuelle Szeneanalyse dargestellt.

## Videozusammenfassung

Bevor das Empfehlungssystem für Lerneinheiten der Online-Lehre realisiert werden kann, wird zunächst aus Vorlesungsaufzeichnungen passende Lerneinheiten erzeugt. Dazu wer- den 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 sich mit sehr hoher Geschwindigkeit bewegen [[14](#_bookmark121)]. Die Videozusammenfassung ist eine Technik, mit der eine kurze Zusammenfassung des Inhalts eins längeren Videos erstellt wird, in- dem die informativsten oder interessantesten Bereiche des Videos für potenzielle Nutzer ausgewählt werden. Der Output der Videozusammenfassung besteht in der Regel aus einer Reihe von Schlüsselbilder (Frames) oder ein kurzes Video, das vom originalen Video extrahiert ist. Die Videozusammenfassung zielt darauf, das Durchsuchen einer großen Sammlung von Videodaten zu beschleunigen und eine effiziente Darstellung des Videoinhalts zu erreichen [[47](#_bookmark147)]. 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 [[37](#_bookmark141)]. In der Literatur wird zwischen Arten von Videozusammenfassung unterschieden: die statische Zusammenfassung, auch *key frame summarization* genannt und dynamische Zusammenfassung, auch *video skimming summarization* genannt [[37](#_bookmark141), [14](#_bookmark121)].

Bei der statischen Zusammenfassung besteht das Ergebnis der Zusammenfassung aus einzelne Schlüsselbilder mit höchster Priorität. Diese werden im ersten Schritt extrahiert, indem Bilder gleichmäßig übersprungen oder zufällige ausgewählt werden. Im nächsten Schritt können die Schlüsselbilder durch kollaborativen Darstellung der benachbarten Bilder bestimmt werden. Mit dieser Methode werden Bilder mit minimalem Rekonstruktionsfehler ausgewählt [[37](#_bookmark141)]. Zur Bestimmung der Schlüsselbilder wird am häufigsten die Farbbasierte Technik eingesetzt. Dabei wird die Farbhistogramm aus RGB oder HSV der einzelnen Bilder berechnet. Das Ergebnis ist die Farbverteilung der Bilder Anhang dessen zwei aufeinander folgende Bilder verglichen werden können. Dabei wird angenommen, dass Bilder in derselben Aufnahme ähnliche Werte haben [[37](#_bookmark141)].

Das Ergebnis der dynamischen Zusammenfassung ist ein kurzes Video, in der die Inter- essanteste Szene aus Eingangsvideo in Form eines Abstrakt wiedergeben werden. Hierzu werden Techniken wie SVD, Bewegungsmodell und semantische Analyse angewendet [[37](#_bookmark141), [14](#_bookmark121)]. Die Tabelle [3.1](#_bookmark16) stellt den Unterschied zwischen statische und dynamische Zusam- menfassung 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 statische und dynamische Zusammenfassung (Quelle:

[[37](#_bookmark141)])

## Audiovisuelle Szeneexploration

Der besondere Reiz der 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 das Video. In der Literatur wird zwischen akustische Salienz, welche ein hörbares Ereignis darstellt, und visuelle Salienz, welche den Sichtbaren Anteil beschreibt. Durch Salienz wird die Aufmerksamkeit erzeugt und den Fokus auf

das Ereignis gelenkt. Die akustische Salienz kann z.B. durch eine Starke Änderung in Audio-Spektrum festgestellt werden. Mit diesem Faktor können Ereignisse wie das Fallen eines Objekts oder Konversationen in Video dargestellt werden. Die akustische Salienz wird auf Basis der Kurzzeit-Fouriertransformation (STFT) bestimmt. Bei visuelle Salienz wird eine Salienzkarte generiert, indem an jede 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 Cosinus-Transformation (DCT) auf Grauwert-Bildern anwendet [[30](#_bookmark135), [40](#_bookmark143)]. Außer Salienz, werden typischerweise Merkmale wie Farbe, Schattierung, Form, Bewegung, Textur und Kontext im Prozess der audiovisuellen Analyse genutzt. In vielen Fällen ermöglicht die Form eine erfolgreiche Erkennung von Objekten durchzuführen.

# Empfehlungssysteme

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

Das Empfehlungssystem hat seinen Ursprung in Forschungsfelder des Data Mining, Information Retrieval, Statistik und maschinelles Lernen [[8](#_bookmark116)]. 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 Objekten aus einer großen Menge zu unterstützen [[17](#_bookmark124)]. Empfehlungssysteme beeinflussen heute die Entscheidungen von Menschen im Alltag, ohne dass sie ein Bewusstsein darstellen [[22](#_bookmark129)]. Z.B. beim Kauf eines Produkts beim Online-Händler Amazon.com, eine der bekanntesten Applikation mit integrierte Empfehlungssystem [[22](#_bookmark129)], stehen die Benutzer dieses Shops vor enorm Auswahl an potentiell interessanten Produkten, die meistens von anderen Kunden durch den Kauf, bewertet wurden. Das im Shop integrierte Empfehlungssystem spielt die Rolle des Assistenzsystems und schlägt dem Benutzer vor, welche Artikel interessant seien könnten („*Kunden, die diesen Artikel gekauft haben, haben auch diesen gekauf*t“). Empfehlungssysteme werden heute in viele Bereiche eingesetzt. Die Abbildung [4.1](#_bookmark19) stellt eine Übersicht über der Realwelt Empfehlungssysteme.

Das Empfehlungsprobleme wird entweder als *prediction Version of problem* oder *ranking version of problem* formuliert [[9](#_bookmark117)].

* Prediction version of problem

In diesem Ansatz wird die Bewertung eines Benutzers vorhergesagt, mit der Annahme, dass Bewertungsverhalten vorhanden sind, Anhang dessen die Präferenzen des Benutzers für ein Objekt ermittelt werden kann. Dieses Problem lässt sich auch

|  |  |
| --- | --- |
| 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 Realwelt Empfehlungssysteme(Quelle:In Anlehnung an [[9](#_bookmark117)])

als „*Problem der Matrixvervollständigung*“ beschreiben, da es eine ursprünglich un- vollständige Matrix *n X m* existiert, deren fehlende Werte Anhang Data-Mining Algorithmen vorhergesagt werden sollen, um die Matrix zu vervollständigt.

* Ranking version of problem

In der Praxis ist es nicht notwendig die Bewertung eines Benutzers vorherzusagen, sondern die Top-k Objekte finden, die diesem Benutzer Interessieren könnten oder die Top-N Nutzer finden, die für ein Objekt angesprochen werden können. Dieser Ansatz, noch als *Top-N Empfehlungsproblem* bezeichnet, ist die an der weitverbreiteten Methode [[9](#_bookmark117)].

Allerdings sind die beide Problemformulierungen ähnlich, da die Lösung des *Prediction version of problem* ist die Abfanglösung für *Ranking version of problem*.

In [[9](#_bookmark117)] 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 das dem Benutzer einige interessieren.

Das Empfehlungsproblem wie folgt mathematisch beschreiben[[9](#_bookmark117)]:

Sei *U* die Menge aller Benutzer im System, *I* die Menge aller Objekte im System, *R* eine vorher festgelegte Menge mit reeller Werter zur Beschreibung der Nutzbarkeit. Sei *fu* die Funktion zur Ermittlung der Nutzbarkeit eins Objekts *i* für einen Benutzer *u*.

*fuU* ∗ *I*− *> R* (4.1)

Um dem Benutzer *u* ∈ *U* ein Objekt zu empfehlen, werden Items *i* ∈ *I* gesucht, welche die Nützlichkeit für den Benutzer *u* maximieren:

∀*u* ∈ *Uiu*=*argmaxR*(*i, u*) (4.2)

Aus unterschiedliche Anwendungskontext von Empfehlungssysteme ergibt sich unter- schiedliche Ansätze [[9](#_bookmark117)]. In folgenden Abschnitten werden diese beschrieben.

## Kollaboratives Filtern

Das kollaborative Filtern (*collaborative filtering*) ist der erfolgreichsten und weit verbreitetsten Technologien für das Empfehlungssystem [[28](#_bookmark134)]. Es lässt sich dadurch begründen, dass diese Technologie kein Wissen über das Einsatzgebiet erfordert. Die Empfehlung wird auf Grundlage der Benutzerverhalten erstellt. Das System empfiehlt den Benutzer Objekte auf Basis der Ähnlichkeitsberechnung dieser Benutzer oder Objekte.

Das Kollaborative Filtern wird in Speicher-basiert (memory-based) und Model basiert (model- based) unterteilt [[17](#_bookmark124)]. Der wesentliche Unterscheide in beide Verfahren ist, dass modelbasierte Ansätze maschinelles Lernen Verfahren für die Vorhersage der Benutzerbewertung eingesetzt, während Speicher-basierte Ansatz die statistischen Methoden verwendet. In der Literatur wurde nachgewiesen, dass durch die Kombination beider Methoden die Qualität der Empfehlung erhöht wird. Das Speicher-basierte Empfehlungssysteme werden auch in Benutzer-basiert(user-based) und Objekt-basiert(item-based) unterschieden [[27](#_bookmark133)].

Im nachfolgenden Abschnitt werden diese beide Verfahren näher untersucht.

### Objekt-basiertes kollaboratives Filtern

Dieses Verfahren verwendet die Ähnlichkeit von Objekten basierend auf Benutzerbewer tungsverhalten um die Empfehlung zu erstellen. Unterschiedliche Objekte die von unter- schiedliche 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](#_bookmark22) beschreibt der Ablauf des Algorithmus.

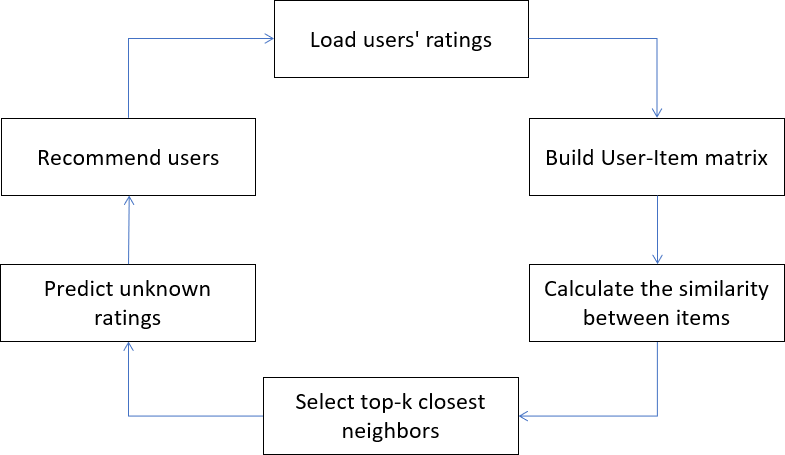


Abbildung 4.1: Ablauf des Algorithmus für das Kollaboratives Filtern (In Anlehnung an [[27](#_bookmark133)])

Im ersten Schritt des Algorithmus werden alle bewertete Objekte *Iu* eines Benutzers herangezogen. Im zweiten Schritt wird auf Basis dieser Daten die Zuordnungsmatrix Benutzer-Objekte (Abbildung [4.2](#_bookmark23)) erstellt. In der Matrix repräsentieren Zeilen *u*1*...un* die Benutzer (*u* ∈ *U*) und die Spalten *i*1*...in den Objekten* (*i* ∈ *I*). In Zellen sind die einzelne Benutzerbewertungen ( *R*(*u,i*)) für Objekte. Unbekannte bzw. unbewertete Objekte sind

mit Symbol gekennzeichnet.

ø

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | *i*1 | *i*2 | *i*3 | ... | *ij* | ... | *im* |
| *u*1 |  |  | *R* |  | *R* |  |  |
| *u*2 |  |  | *R* |  | ø |  |  |
| *u*3 |  |  | *R* |  | *R* |  |  |
| ... |  |  | ... |  | ... |  |  |
| *uj* |  |  | ø |  | *R* |  |  |
| ... |  |  | ... |  | ... |  |  |
| *un* |  |  | *R* |  | *R* |  |  |

Abbildung 4.2: User-Item-Matrix

Im dritten Schritt des Algorithmus wird die Ähnlichkeitsberechnung zu einem Zielobjekt

*iu* ∈ *I* ermittelt. Dabei kommt häufig die Kosinus Ähnlichkeit zum Einsatz [[53](#_bookmark153)].

(*i, j*) = *cos*(*i, j*) = *i.j* = q Σ*u*∈*U ru,i.ru,j*

Σ

*r*

*r*

*sim*

∥ *i* ∥∥ *j* ∥

2

*u,i*

qΣ*u*∈*U*

2

*u,j*

*u*∈*U*

(4.3)

Wie in der Gleichung zu sehen ist, die Messung der Ähnlichkeit zwischen zwei Objekte *i* und *j* besteht darin beide Objekte als Vektors darzustellen und die Ähnlichkeitkosinus der Vektoren zu berechnen, wobei *ru,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, die eine hohe Ähnlichkeit zu dem Zielobjekt Objekt *i* auf- weisen, bezeichnet. Die folgende Formel wird zu Berechnung der Vorhersage verwendet. Dabei enthält die Menge *Nk* die nächste *k* ähnlichste Objekte zu *i* und *ru,j* ihre Bewer- tung.

*ru,i*

ΣΣ=

*j*∈*Nk sim*(*i, j*)*.ru,j j*∈*Nk sim*(*i, i*)

(4.4)

Nach Berechnung der Nutzbarkeit der unbekannten Objekte, schlägt das System die mit größten geschätzte Bewertungen Objekte vor [[8](#_bookmark116)]

### Benutzer-basiertes kollaboratives Filtern

Im Gegensatz zum Objekt-basiertes Verfahren, das Ziel des *User-based Collaborative filtering* ist es, Benutzer zu finden, die ähnliche Bewertungsverhalten wie der aktive Benutzer haben und ihre Bewertungen für die Vorhersage der Bewertung des aktiven Benutzers verwenden. Die Idee dahinter beruht sich auf die Annahme, dass Benutzer mit ähnlicher Bewertung unterschiedlicher Objekte dieselben Interessen hätten [[53](#_bookmark153)]. Um ein aktiver Benutzer zu empfehlen, findet das System die *k* ähnliche Nutzer im System, um die Objekte mit Hilfe deren Bewertungen zu empfehlen [[53](#_bookmark153)].

Der im vorherigen Abschnitt beschriebene Ablaufalgorithmus gilt auch für die Benutzer- basierte Methode mit einem kleinen Unterschied. Im dritten Schritt, basiert die Ähn- lichkeitsberechnung nicht auf Objektähnlichkeit, sonder auf Benutzerähnlichkeit. Dazu wird häufig Pearson Korrelation verwendet (Gleichung [4.1.2](#_bookmark25) ).

( ) = Σ *in*

*i*=1

*i*

(*xi* − *x*)(*yi* − *y*)

*i*

(4.5)

*i*

*sim x, y*

*i*

Σ *in*=1√(*x*

− *x*)2 Σ *in*=1√(*y*

− *y*)2

Nach Ermittlung der *k* nächste Nachbar, wird die Nutzbarkeit wie folgt berechnen:

*r*(*u, i*) = *ru*

*v*∈*Nk sim*(*u, v*)(*rv,irv*)

*v*∈*Nk sim*(*u, v*)

Σ Σ+

(4.6)

### 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, wie Objekte, die über die gesamte Benutzer Gruppe am beliebtesten sind [[22](#_bookmark129)].

##### New User Problem

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

##### 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 und 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 [[22](#_bookmark129)].

### Implizit und explizit 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/Zufriedenheit an einem Objekt entspricht [[9](#_bookmark117)]. 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 große 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 er ein Interesse über das Produkt hat. Im Bereich Streaming, wenn ein Nutzer einen Film anschaut, ist es wahrscheinlich, dass er daran interessiert ist. In diese Fälle ist die Bewertung binär (0 für keine Interesse und 1 für Interesse). Aber die Tatsache, dass ein Kunde ein Produkt aus einer großen Menge nicht kauft, oder dass ein Benutzer einen Film nicht abspielt, bedeutet ab nicht unbedingt, dass er keine Präferenz für diese hat. Es besteht die möglich, dass er von deren Existenz nicht bewusst ist [[9](#_bookmark117)].

## Andere Empfehlungssysteme

Das inhaltsbasierte Filtern(Content-based filtering) ist andere Art des Empfehlungssystem , 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[[9](#_bookmark117)].

Das hybride Filtern ist eine Art des Empfehlungssystem, die kollaborative und inhalts- basierte Methoden kombiniert. Es ermöglicht bestimmte Einschränkungen dieser beide Methoden zu kombinieren. In [[8](#_bookmark116)] wird verschiedene Möglichkeiten für die 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 die- sem 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 die- sem Ansatz werden kollaborative Methoden in ein inhaltsbasiertes Verfahren in- tegriert. Im Fall es 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.

## Evaluierung der Empfehlungssysteme

Die Evaluierung von Empfehlungssysteme und ihre Algorithmen ist eine herausfordernde Aufgabe. Ein Algorithmus kann für ein Datensatz gut und ein anderer schlecht funktio- nieren. Ein anderes Problem ist, dass die Empfehlung für unterschiedlich Zwecke ver- wendet wird. Für Systeme wo die Empfehlung zur Unterstützung von Entscheidungen eingesetzt werden, hatten Forscher in der Vergangenheit nur die Genauigkeit als Metrik verwendet, um die Qualität des Empfehlungssystem zu messen. Heute liegt den Fokus insbesondere auf die Benutzerzufriedenheit, indem gemessen wird, wie oft das System seine Nutzer zu einer falschen Entscheidungen geführt hat[[28](#_bookmark134)].

Zur Evaluierung des Empfehlungssystem kann die Online- oder Offline-Methode wer- den. Bei der Online-Methode wird die Reaktion des Benutzers im Livesystem auf die präsentierten Empfehlungen gemessen. Bei der Online-Evaluation sind die Klassifikationsmetriken die wichtigsten Faktoren. Da die Online-Evaluierung eine aktive Beteiligung der Nutzer erfordern, ist diese oft für Benchmarking und Forschung ungeeignet. Es ist besonders 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 dafür geeignet, wobei historische Daten der Benutzer verwendet werden. Diese Methode wird in der Literatur viel mehr behandelt, obwohl die Online-Methode zum Vervollständigen der Evaluierung zwingend notwendig ist [[9](#_bookmark117)]. Im Folgenden werden die Evaluations Metriken eingegangen.

### Genauigkeitsmetriken

Die Genauigkeitsmetriken messen, wie nah die vom Empfehlungssystem vorhergesagten Bewertungen an den tatsächlichen Benutzerbewertungen sind. Dieser Metrik ist besonders wichtig für die Evaluation von Systemen, wo die vorhergesagte Bewertung dem Nutzer angezeigt wird. Da Anhand der vorhergesagte Bewertungswerte, die Empfehlungsliste sortiert werden kann, ist dieser Metrik auch für das Top-N Problem gut geeignet. Die Genauiketsmetriken In sind der mittlere absolute Fehler(MAE) und der mittlere quadratische Fehler verwendet(RMSE)[[9](#_bookmark117)].

**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 *Ru,i* geheim gehalten und das MAE mit der vorhergesagte Bewertungen berechnet.

##### RMSE

*n*

*MAE* = *n* |*ri* − *Ri*| (4.7)

1 Σ

*i*=1

Eine mit dem MAE zusammenhängende Maße ist RMSE, der die Fehlerquadrate summiert. Größere Fehler sind durch RMSE schneller sichtbar als kleinere. Auf diesem Grund ist RMSE am häufigsten zur Evaluierung der Empfehlungssysteme eingesetzt[[28](#_bookmark134)].

*n*

‚. Σ1

*RMSE* = , *n*

*i*=1

(*ri* − *Ri*)2

(4.8)

### Klassifikationsmetriken

Klassifizierungsmetriken messen, wie häufig das Empfehlungssystem den Benutzer zur richtige oder falsche Entscheidungen aufgrund der Vorhersage führt. Diese Metrik ist für Systeme wo Bewertungswerte vorhergesagt werden nicht geeignet, sondern für Systeme wo Objekte dem Nutzer vorgeschlagen werden. Die Evaluation dieser Systemen entwickelt sich zu einem klassischen Klassifikationsproblem mit vier unterschiedliche Klassen: True-Positive, False-Positive, False-Negative, True-Negative ([4.2](#_bookmark32)).

|  |  |  |
| --- | --- | --- |
|  | 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 [4.3.2](#_bookmark33)) ist der Ratio von guten klassifizierten Objekten auf die gesamte Anzahl an vorgeschlagene Objekte. Es gibt die Wahrscheinlichkeit das ein vorgeschlagenes Objekt relevant ist oder nicht [[28](#_bookmark134)]

*Precision* =

*TP TP* + *FP*

(4.9)

Das Recall (Gleichung [4.3.2](#_bookmark34)) 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 [[28](#_bookmark134)].

*Recall* =

*TP TP* + *TN*

(4.10)

Es gibt verschiedene Ansätze, um Precision und Recall zu kombinieren. Eins davon ist F1-Metrik(Gleichung [4.3.2](#_bookmark35)), die beide Metriken zu einer Zahl kombinieren. Es dient dazu das harmonische Mittel beider Werte zu berechnen.

*Precision.Recall*

*F* 1 = 2*.*

*Precision* + *Recall*

(4.11)

# Empfehlungssystem für die Online-Vorlesung

Das vorherige Kapitel hat eine ausführliche Grundlage über Empfehlungssysteme dargestellt, welche als Grundlage für jede Anwendungsgebiet dient. Es wurde dabei verschiede- ne Methoden vorgestellt wobei Annahmen getroffen wurden. Ein von diese Anwendungs- gebiet 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 der Grundlage 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 als Basis für die automatische Zerlegung der Aufzeichnungen dient.

## 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 Unter- schiede notiert [[54](#_bookmark152)]:

* Ziel: Das Ziel von Empfehlungssysteme in E-Commerce ist es den Umsatz zu steigern. Dieses Ziel kann in Form von Geldbeträge oder Kundenanzahl gemessen wer- den. 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 Empfehlungssystem die Kunde beim Kauf zu unterstützen während in E-Learning Lernende beim Lernen unter- stützt 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, andere 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 Lerninhalte [[54](#_bookmark152)].

Ebenso wurde bislang 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 Studenten-verhalten oder Leistung zu verschaffen. Dieser Ansatz von Empfehlungssystem ist nicht Teil der vorliegenden Arbeit und deswegen nicht weiter be- handelt. Die zweite Benutzergruppe sind Studierende, die die eigentliche Endnutzer des Empfehlungssystem. Sie sind auch gleichzeitig einen zentralen Bestandteil des Systems aufgrund ihrer aktiven Rolle.

In [[54](#_bookmark152)] 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

Bislang wurde explizites 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, weil ein Benutzer ein Produkt entweder gut oder schlecht finden kann. Im Bereich E-Learning ist es nicht ausreichend Aufgrund der Komplexität von Lernressourcen [[46](#_bookmark146), [42](#_bookmark144)]. Ein Student kann z.B. ein Lerninhalt schlecht bewerten, weil den Inhalt ihm nicht detailliert fällt. Er kann auch eine geringere Bewertung geben, weil ihm den Inhalt zu schwer oder leicht empfiehlt. Diese Bewertung aufgrund von verschiedenen Interpretationen kann vom Empfehlungssystem berücksichtigt werden. In diesem Zusammenhang hat sich den Begriff implizit Feedback in vielen Bereiche etabliert [[49](#_bookmark148)]. 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 Lerninhalte[[34](#_bookmark139)].

## Online-Vorlesung

Unter dem Begriff Vorlesung wird eine Lehrveranstaltung verstanden, welche der Kern darin besteht, Studierende mittels eines Vortrags zu unterrichten [[13](#_bookmark120)]. In traditionelle Vorlesungen, der Dozent steht vorne im Hörsaal, spricht und zeigt Folien, während die Studierende zuhören und sich Notizen machen [[56](#_bookmark155)]. Das Ziel der Vorlesung ist es, wissen den studierende zu vermittelt. Bei der Vorlesung nutzen Dozenten unterschiedliche Lehrmethode, 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 Weiterverlauf. 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 Professoren eigene Wege. Trotzdem werden in viele Vorlesungen einige Gemeinsamkeit gefunden: sie enthalten eine zu Beginn Einleitung. wobei meistens das Thema aus der früheren Vorlesung wiederholt wird, um Studierende an das Thema zu erinnern. Danach folgt den eigentlichen Inhalt mit Beispielen, Praktische Übungen, Diskussion etc. Am Ende eines Vorlesungsabschnitts werden Fragen gestellt und beantwortet.

Während des Vorlesungsvortrags 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, Excel und Webseiten kommen auch oft zum Einsatz.

Die Online-Vorlesung ist die multimediabasierte Form der Vorlesung. Es fängt meistens mit einer Aufzeichnung der Vorlesung an, Anhang dessen Lerninhalte/abschnitte erzeugt werden. Lerninhalte/abschnitte sind thematische Unterteilung der Vorlesungsaufzeichnung, welche als Grundlage für die Online-Vorlesung dienen. Zur Aufzeichnung wird oft das 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 [[41](#_bookmark145)]. Für die Aufzeichnung eine Vorlesung bietet sich unter Umständen auch das Format Videokonferenz, wobei Kameras und Mikrophones zu Einsatz kommen.

1. **Konzeption**

Bislang wurde die hervorgehobenen Probleme in der vorliegenden Arbeit beschrieben und die theoretischen Grundlagen in der jeweiligen Themengebiete geschafft. In diesem Kapitel wird eine Lösung konzipiert. Die vorgeschlagene Lösung wird Anhang von Infrastruktur- und Architekturdiagramm dargestellt. Die beteiligte Komponente wird ausführlich beschrieben.

* 1. **Systemüberblick**

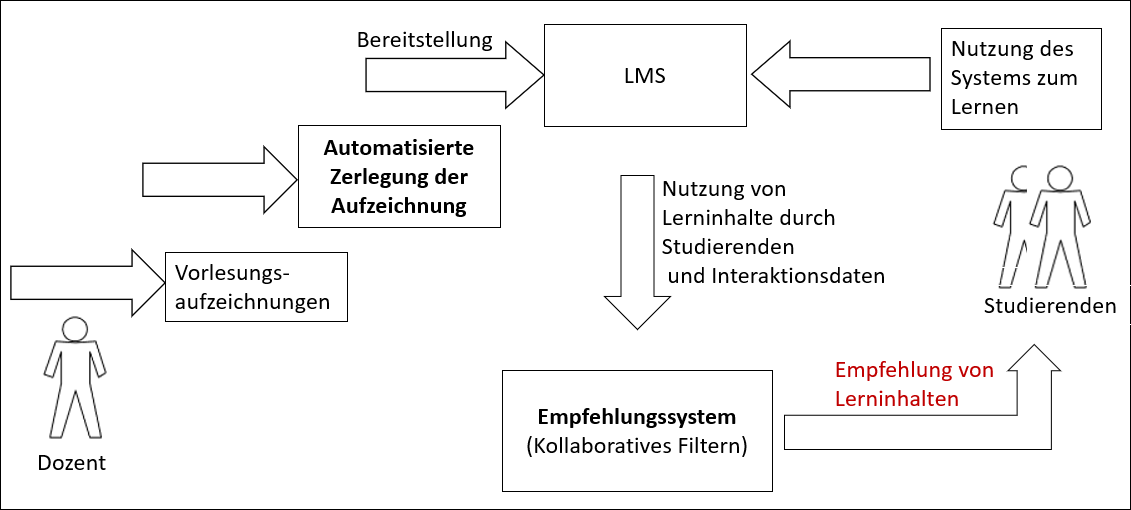


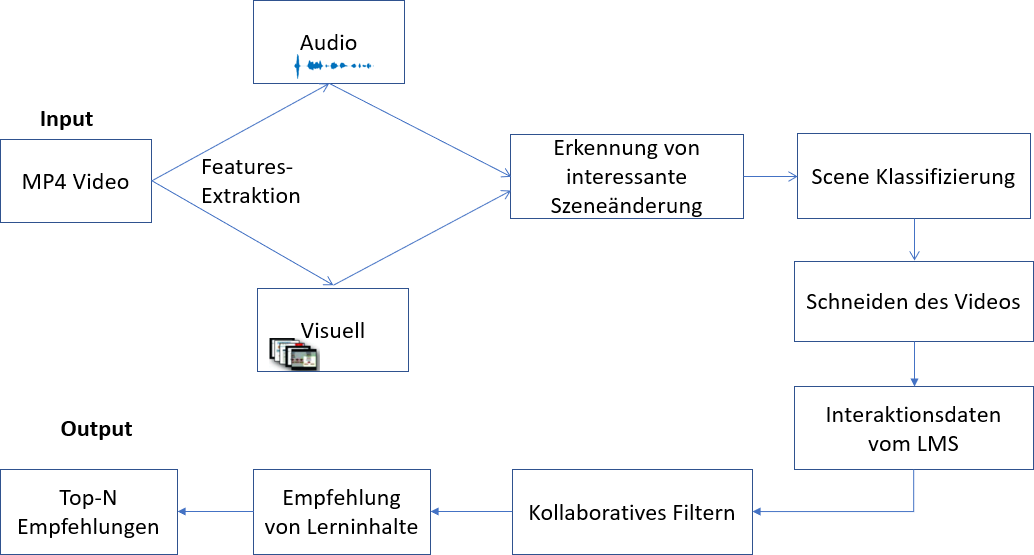
Abbildung 6.1: Systeminfrastruktur

Die Abbildung [6.1](#_bookmark41) stellt die Systeminfrastruktur dar. Im System sind zwei Akteure beteiligt: der Dozent und die Studierende. Zu Beginn des Prozesses, zeichnet der Dozent die Vorlesungen auf. Nachdem diese mit dem automatisierten Zerlegung-Tools zerlegt sind, werden die daraus resultierende Lerninhalte (Lernvideos) in das LMS System hoch- geladen. Im Rahmen ihrer Lernprozesse greifen Studierenden auf das System zu, um sich

das dargestellte Wissen anzueignen. Die Nutzungsdaten der Lerninhalte durch letzteren werden vom LMS System protokolliert. Diese Daten fließen in das integriertes Empfehlungssystem ein. Nach Ansatz von kollaborative Algorithmen auf die Nutzungsdaten, werden den Studierenden Lerninhalte empfohlen. Genauerer gesagt, schlägt das System der Studierenden vor, was sie als nächstes lernen sollen.

Die Herausforderung dieser Arbeit liegt in die hervorgehobene Komponente in der Abbildung: das Zerlegen der Aufzeichnungen und die Realisierung des Empfehlungssystem. Eine detaillierte Beschreibung dieser Komponente erfolgt in der Systemarchitektur.

* 1. **Systemarchitektur**

Abbildung 6.2: Systemarchitektur

Die Abbildung [6.1](#_bookmark41) stellt die Architektur des zu entwickelten Systems dar. Die dazugehörige Komponente können in zwei Themengebiete sortiert werden. Vom *MP4 Input Video* bis zum *Schneiden des Videos* gehören die Komponente zum Thema automatische Zerlegung der Vorlesungsaufzeichnungen. Von *kollaboratives Filtern* bis hin zur *Top-N Empfehlungen*, gehören die Komponente zum Thema Empfehlungssystems. Was der

Komponente *Interaktionsdaten vom LMS* betrifft, ist es die Schnittstelle zwischen der beiden Themengebiete. In folgenden Abschnitten werden, die Die einzelnen Komponente detailliert beschrieben.

* 1. **MP4 Video Input**

Der Eingang des Systems ist die Vorlesungsaufzeichnung, die der Professor realisiert und in ein MP4-Format vorlegt hat. MP4 ist ein digitales Multimedia-Containerformat, das am häufigsten zum Speichern von Video-, Text- und Audiodaten verwendet wird. Im Kapitel 3 wurden Techniken zu Videozusammenfassung und audiovisuell Szeneexploration beschrieben, wobei zwischen statische und dynamische Zusammenfassung unterschieden war. Da die Vorlesungsaufzeichnung eher ein statisches Material ist, werden die Techniken zur statischen Videozusammenfassung angewendet, um visuelle Features zu extrahieren. Was Audiofeatures betrifft, werden Techniken zur Audioszeneanalyse angewendet. Im Weiteren werden diese beide Feature-Extraktionstechniken erläutert. zu extrahieren.

* 1. **Visuell-Features**

Die Schlüsselbildextraktion-Technik wird zur visuellen Feature-Extraktion angewendet. Das Input-Video wird Bild pro Bild ausgelesen, dann die Farbhistogramme aus RGB der einzelnen Bilder berechnet. Die Farbhistogramme stellen die Tonwertverteilung eines Bildes. Diese gibt die Anzahl der Pixel des Bildes für einen bestimmten Helligkeitswert von 0 bis 256 an [[32](#_bookmark137)]. Auf Basis der Farbhistogramme wird der Euklidische Abstand zwischen zwei aufeinander folgende Bilder berechnet. Anhand des Abstands wird entschieden, ob die Frames signifikante Unähnlichkeiten aufweisen oder nicht. Es wird ausgegangen, dass eine signifikante Unähnlichkeit der Histogramme zwischen den Einzelbildern auf einen schnellen Szenenwechsel im Video hinweist, der möglicherweise interessante Komponenten enthalten könnte [[32](#_bookmark137)]. In dieser Arbeit, wenn der Euklidische Abstand von zwei aufeinanderfolgenden Bildern den Durchschnitt der Distanzen über- steigt, wird das Bild als Schlüsselbild betrachtet. Seine Position in das originale Video wird in gespeichert, mithilfe dessen das Bild bei Bedarf später extrahiert werden kann. Das Ergebnis der Schlüsselbildextraktion ist eine Liste mit allen Positionen der Schlüsselbilder in das Eingangsvideo. Für eine bessere Segmentierung der Aufzeichnung werden auch die Audio-Features benötigt.

* 1. **Audio-Features**

In Kapitel 3 wurde beschrieben, dass 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 Pausen in Vorlesungsvideo zu erkennen. Dazu wird die Audio-Spur vom Eingangsvideos extrahiert, dann mit Hilfe der Fouriertransformation in Audio-Frames umgewandelt. Als nächstes werden die Frames analysiert. Frames, die sich an den Wert 0 annähern werden als Stillstand betrachtet. Sollte der Stillstand länger als eine Sekunde sein, dann wird die Position des Frames einer Liste gespeichert. Dabei werden die Positionen des Audios und Video-Frames synchronisiert. Das Ergebnis der Audio-Features-Extraktion ist eine Zeitliche Liste mit Werte 0 und 1, wobei 0 die Stellen ohne Stillstand bedeuten und 1 Stellen mit Stillstand.

* 1. **Erkennung von interessante Szeneänderungen**

In den vorherigen Abschnitten wurde erklärt, wie die Audio- und Visuelle-Features extrahiert werden. Dabei wurde bei Visuelle-Features der Schlüsselbildwechsel als Anfang einer neuen Szene betrachtet. Der Ansatz von Audio-Features diente zur Pausenerkennung, auf Basis dessen das Video auch segmentiert werden kann. Hier werden diese beiden Features kombiniert, um Interessante Szenen zu finden. Dazu werden beide Zeitliche Features parallel dargestellt. Positionen mit gleichzeitigen Überschneidungen in Audio- und Visuelle-Features bekommen ein hohes Scoring. Z.B. Wenn gleichzeitig ein Szenenwechsel stattfindet und eine Pause festgestellt wird, passiert wahrscheinlich etwas interessantes an der Stelle. Es wurde mehrfach beobachtet, dass vor einem Themawechsel der Professor Frage stellt, wartet auf die Antwort und danach die Folie wechselt oder zuerst die Folie wechselt, dann Frage stellt und auf Antwort wartet. Positionen wo ein solches Verhalten in Video festgestellt sind bekommt auch ein hohes Scoring. Dabei wird als Pause Ruhezeit von mindestens drei Sekunden betrachtet.

* 1. **Szeneklassifizierung**

Nachdem Interessante Szenen erkannt sind, werden diese klassifiziert. Das Ziel der Klassifizierung ist es, jede erkannte Szene eine Klasse zuzuweisen. Dadurch können Fehlerszenen identifiziert werden und beim Schneiden des Videos zu entfernen oder ignorieren.

Als Fehlerszenen werden Szenen mit Aufnahmefehler bezeichnet(z.B. Schwarz Bild). Au- ßerdem wird die Szeneklassen bei der Generierung der LMS Daten benötigt. Es werden vier verschiedene Szeneklassen unterschieden:

* **Slide**: Szenen mit PowerPoint-Folie im Vollbild
* **Whiteboard**: Szenen mit Skizze, Handschiff
* **Professor**: Szenen mit dem Professor im Vollbild
* **Fault:** Schwarz Bild.
* **Other**: Programmcode, Text, und sonstiges

Als Eingangs für den Klassifikator werden pro Szene drei Bilder extrahiert: das Start-, End- und Mittelbild. Auf Basis des gesamten Bildinhaltes weist der Klassifikator dem Bild bzw. Szene eine Klasse zu. Das CNN Model wird als Klassifikator verwendet. Es ist eine Art künstliches neuronales Netzwerk, das häufig in der Bilderkennung und - Verarbeitung angewendet wird.

* 1. **Scheiden des Videos**

Nach der Szeneklassifizierung wird einer CSV-Datei mit allen erkannten Szenen, die Start- und Endpositionen, die Längen und Szeneklassen generiert. Diese Datei wird im weiteren Verlauf als Vorlesungsskript genannt. Anhang der Start- und Endpositionen der erkannten Szenen wird das Eingangsvideo geschnitten. Dazu wird das FFmpeg-Tool angesetzt.

* 1. **Interaktionsdaten vom LMS**

Wie schon erwähnt, ist die Datensatzgenerierung die Schnittstelle zwischen der Segmentierung der Aufzeichnung und dem Empfehlungssystem. Der Datensatz wird benötigt, um das Empfehlungssystem zu testen und evaluieren. Im Kapitel 4(Abschnitt Evaluierung der Empfehlungssysteme) wurde die Offline-Evaluierung als geeignete Methode für die Forschung vorgestellt. Diese Methode, die historische Daten der Benutzer verwendet, wird im Rahmen dieser Arbeit für die Evaluierung angesetzt. Ebenso wurde in Kapitel 5 (Empfehlungssystem für die Online-Vorlesung) Studierenden als Endbenutzer des Empfehlungssystems für Lehreinheiten der Online-Vorlesung beschrieben. Hier werden Studenten mit ihrer Interaktionsdaten mit Lerninhalten generiert. Es werden für jede Benutzer implizite Feedback-Informationen generiert. Zunächst werden mögliche relevante

Profilinformationen definiert, die als Features für das Empfehlungssystem verwendet werden.

* *learningDuration*: es handelt sich um die Nutzungslänge der Lerneinheit von der Studierende. Genauerer gesagt, es ist wie lange der Student mit einem Video gelernt hat. Es wird angenommen, dass bei Interessante Lehrinhalte die Studierende mehr Zeit mit dem Video verbringt. Um die Berechnung zu vereinfachen, wird diese Länge in Prozentzahl definiert gehalten. Bei 100% wurde das Video komplett angeschaut.
* *numberOfPostsAboutScene*: es ist die Anzahl am Post, die die Studierende in Forum über die Lerneinheit geschickt hat. Dabei wird angenommen, dass bei mehr Posts ist die Lerneinheit Interessant.
* *userId*: es ist eine eindeutige Nummer, die die Studierende identifiziert.
* *sceneId*: es ist eine eindeutige Nummer, die die Lerneinheit identifiziert.

Anhang von definierten Features wird der Datensatz als CSV-Datei mit Hilfe der Vorlesungsskript generiert. Dazu werden Studierende mit unterschiedlichen Profilen generiert: Faule Studierende, Mittel-faul, Mittel, und gute Studierende. Der Unterschied zwischen den Profilen liegt in dem Grad der Interaktion mit Lerneinheiten. Z.B. gute Studierende interagieren sehr viel mit Lerninhalten. Auf dem generierten Datensatz werden kollaborative Filtern-Algorithmen angewendet, um die Empfehlungen zu erstellen.

* 1. **Kollaboratives Filtern**

Die Implementierung des kollaborativen Empfehlungssystems in dieser Arbeit folgt die klassisch Lösungsschritte für ein maschinelles Lernen Problem. Das erstellte Modell (Abbildung [6.3](#_bookmark53)) wird vom CRISP-Modell abgeleitet (Abbildung [6.3](#_bookmark53) ). Im Folgenden werden die einzelne Schritte des Modells beschrieben.

* + 1. **Data Understanding**

In diesem Schritt wird der generierter Datensatz aus dem vorherigen Abschnitt und der Vorlesungsskript in Jupiter Notebook geladen und analysiert. Es werden Hilfemittels wie deskriptiver Statistik und Verteilungsfunktionen angesetzt, um die Daten zu explorieren.

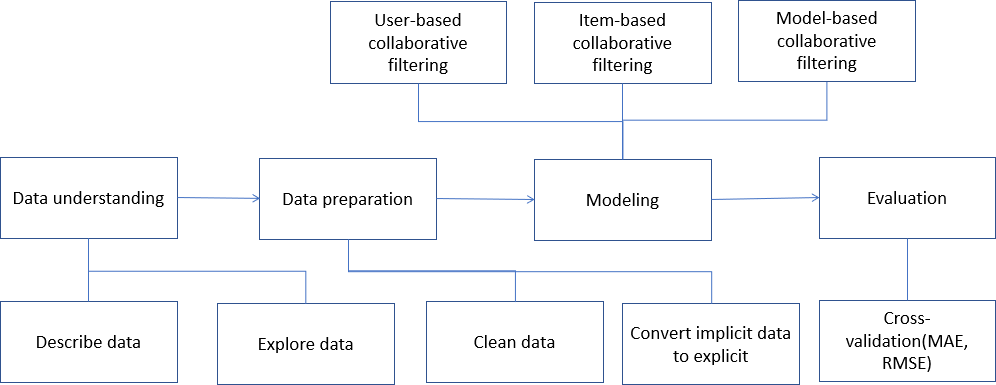


Abbildung 6.3: Architektur des Empfehlungssystem

* + 1. **Data preparation**

In diesem Schritt wird der Datensatz bereinigt indem mögliche Duplikate gesucht und entfernt werden. Außerdem ist ein wichtiger Schritt hier die impliziten Daten in explizit zu konvertieren, um klassische kollaborative Algorithmen auf dem Datensatz anwenden zu können. Diese erfolgt erstens durch die Normierung der Features *learningDuration*, *numberOfPosts und zweitens* durch die Gewichtung der genannten Features*.* Während der Normierung wird jedes Feature in Wertebereich [0,1] umgewandelt. Dann werden die Features durch mit definierten Gewichten multipliziert. Das Gewicht gibt an, wie wichtig das Feature für die Ermittlung des Interesses des Benutzers für die Lerneinheit ist. Wenn eine Studierende eine Lerneinheit komplett anschaut, ist es sehr wahrscheinlich, dass diese das Video mag. Ebenso wenn er das Video in der ersten Minute abbricht, ist es sehr wahrscheinlich, dass diese keine Interesse daran hat. Auf diesem Grund ist dieses Feature viel wichtiger gegenüber *numberOfPosts.* Im Rahmen dieser Arbeit bekommt das Feature *learningDuration* das Gewicht von 5 und *numberOfPosts* das Gewicht 2. Somit ergibt sich eine 7-Punke Bewertungssystem ([0,1] x 5 + [0,1] x 2). Das Ergebnis der Konvertierung ist ein Tupel (userId, sceneId, rating), die für die Modellierung verwendet wird.

* + 1. **Modeling**

Im Kapitel 4 wurden die Algorithmen für das kollaborative Filtern ausführlich beschrieben. Dabei wurde modellbasierte und Speicherbasierte Ansätze vorgestellt. Beim

Speicherbasierten Ansatz wurde noch zwischen Benutzer-basierte und Objekt-basierte Methoden unterschieden. Alles genannte Ansätze werden im Rahmen dieser Arbeit experimentiert. Im Folgenden werden Anhang zwei konkrete Beispiele das zu implementierende System vorgestellt.

##### Benutzer-basierte kollaboratives Filtern

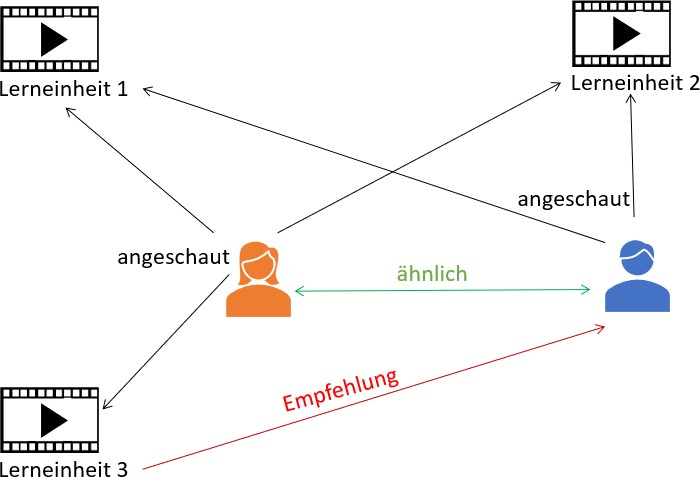


Abbildung 6.4: Beispiel Benutzer-basiertes kollaboratives Filtern

Die Abbildung [6.4](#_bookmark57) stellt ein Beispiel für die Benutzerbasierte Methode dar. Die Studentin hat die Lerneinheiten 1, 2 und 3 angeschaut und der Student die Lerneinheiten 1 und

2. Es wird angenommen, dass beide Studierende die angeschaute Lehreinheiten mögen. Da der Student die Lerneinheit 3 noch nicht angeschaut hat, wird das System ihm es empfehlen, wenn er ähnlich mit zu der Studentin ist. Wie schon beschrieben ist Ähnlich- keit in der Hinsicht auf das Bewertungsverhalten. Zur Umsetzung wird der Algorithmus [4.1](#_bookmark22) herangezogen.

##### Item-Based

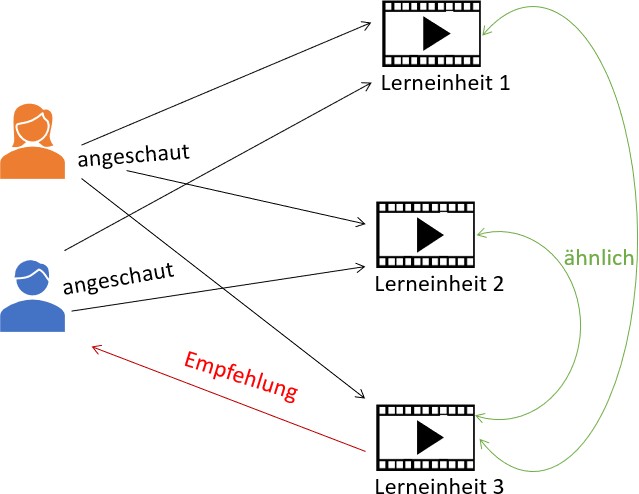


Abbildung 6.5: Beispiel Objekt-basiertes kollaboratives Filtern

Die Abbildung [6.5](#_bookmark59) stellt ein Beispiel für die Objektbasierte Methode dar. Die Studentin hat alle Lerneinheiten angeschaut und der Student nur die Lerneinheiten 1 und 2. Wenn die Lerneinheiten 1 und 3 ähnlich sind, oder die Lerneinheiten 2 und 3, dann wird dem Studenten die Lerneinheit 3 empfohlen. Zur Umsetzung wird ebenso der Algorithmus [4.1](#_bookmark22) herangezogen.

* + 1. **Evaluation**

Wie schon erwähnt, wird die Offline-Evaluierungsmethode in dieser Arbeit angesetzt. Ein dazu passendes Framework ist die Kreuzvalidierung (Cross-Validation), welche eine statistische Methode zur Einschätzung der Fähigkeiten von maschinellen Lernen Modelle ist. Sie wird häufig in maschinelles Lernen verwendet, um ein Modell für ein bestimmtes Vorhersagemodellierungsproblem zu vergleichen. Dieses Verfahren, noch k-fold Kreuzvalidierung genannt, teilt einen gegebene Datensatz in k Gruppen. Jede einzelne Gruppe,

wird als Testdatensatz verwendet und die restliche als Trainingsdatensatz. Das Modell wird auf Trainingsdatensatz trainiert und auf Testdatensatz getestet. Das Ergebnis der Gruppen-Evaluation (RMSE, MAE) wird behalten und mit dem nächsten Modell fortgesetzt. Zum Schluss wird das Ergebnis der Evaluation zusammengefasst [[3](#_bookmark111)].

1. **Implementierung**

In diesem Kapitel wird das beschriebene System im Kapitel 6 technisch umgesetzt. Es werden zunächst die verwendeten Technologien und Daten vorgestellt, dann startet die Implementierung mit der Segmentierung der Aufzeichnungen, gefolgt vom Empfehlungssystem.

* 1. **Technologien**

In diesem Abschnitt werden die erforderlichen und wichtigsten Technologien für die Implementierung beschrieben.

* + 1. **Python**

Alle Programmierungsaktivitäten in dieser Arbeit wurden in Python geschrieben. Python ist ein High-Level und Allzweckprogrammiersprache, die im Bereich des maschinellen Lernens weit verbreitet ist. Diese Programmiersprache ist aufgrund seiner hochentwickelten Datenstrukturen für die Entwicklung von Algorithmen und die Durchführung explorativer Datenanalysen geeignet [[50](#_bookmark149)]. Die letzte stabile Python Version (3.8) wurde in dieser Arbeit verwendet.

* + 1. **Jupyter Notebook**

Jupyter Notebook ist eine Open-Source-Webanwendung, die ermöglicht die Erstellung von Dokumenten, die Python Code, Gleichungen, Visualisierungen und Kommentare enthalten. Zu den Anwendungsbereichen gehören: unter anderem Datenbereinigung und Umwandlung, numerische Simulation, statistische Modellierung, Datenvisualisierung und maschinelles Lernen [[51](#_bookmark150)]. Das Jupyter Notebook wurde in dieser Arbeit verwendet, um den Prozess zum Experimentieren und Aufbau des Empfehlungssystems auf Jupyter Notebook durchgeführt. Das Jupiter Notebook Distributionspaket von Anaconda 3 wurde verwendet.

* + 1. **Pycharm**

*7 Implementierung*

Pycharm ist eine integrierte Entwicklungsumgebung für die Computerprogrammierung, insbesondere für die Python Sprache. Pycharm bietet Code-Analyse, einen grafischen Debugger, ein integriertes Unit-Test-Framework sowie die Unterstützung der Entwicklung Data Science mit Anaconda [[35](#_bookmark140)]. Die Programmierungsaktivitäten für die Zerlegung der Vorlesungsaufzeichnungen fand in dieser Entwicklungsumgebung statt.

* + 1. **OpenCV**

Open Computer Vision (OpenCV) ist eine Open-Source-Software-Bibliothek für die Bild- und Videoanalyse, welche in C/C++ implementiert ist. Diese Bibliothek wird weltweit für diesen Zweck angesetzt [[19](#_bookmark126)]. OpenCV wurde in dieser Arbeit verwendet für die visuelle Szeneanalyse Verwendet. Seine Version 4.5.1.48 wurde verwendet.

* + 1. **Surprise**

Surprise ist eine Open-Source-Python-Bibliothek für die Entwicklung von Empfehlungssysteme mit expliziten Bewertungsdaten. Das Framework wurde auf Basis von Scikit- learn-API entwickelt, welche den Benutzer mit Python maschinelles Lernen Ökosystem sehr familiär ist. Unter anderem bietet Surprise verschiedene vorgefertigte Vorhersage- Algorithmen wie KNN, SVD, PMF, SVD++ und NMF. Außerdem bietet das Framework Werkzeuge wie Kreuzvalidierung für die Evaluation der Leistung der Algorithmen [[31](#_bookmark136)]. In der Vorliegende Arbeit wurde dieses Framework in der Version 1.1.1 angewendet, um das Modellbasierte kollaborative Filtern zu experimentieren. Ebenso wurde es zur Evaluation der Performanz der Algorithmen angesetzt.

* + 1. **Pandas**

Pandas ist eine Python-Bibliothek, die High-Level-Datenstrukturen für Daten-bezogene Aufgaben zur Verfügung stellt. Pandas wird oft als das beliebteste Werkzeug zur Datenanalyse und -manipulation in Python angesehen. Das Framework wird häufig in den Bereichen Finanzen, Statistik und maschinelles Lernen angesetzt [[52](#_bookmark151)]. Panda ist ein zentrales Werkzeug in dieser Arbeit für die Entwicklung des Empfehlungssystem.

* + 1. **NumPy und SciPy**

*7 Implementierung*

NumPy ist eine wichtige Python-Bibliothek für wissenschaftliche Berechnungen. Das Paket bietet effektive Implementierungen von mehrdimensionalen Arrays und Matrizen sowie fortgeschrittene mathematische Funktionen, die mit diesen Arrays arbeiten [[20](#_bookmark127)].

SciPy ist eine Sammlung mathematischer Algorithmen und gängiger numerischer Routinen in Wissenschaft und Technik. Seine Funktionen basieren auf den Datenstrukturen von NumPy [[18](#_bookmark125)]. In dieser Arbeit wurde beide Bibliotheken für Mathematische Berechnungen angesetzt.

* + 1. **Seaborn und Matplotlib**

Matplotlib ist eine Bibliothek zur Erstellung statischer, animierter und interaktiver Vi- sualisierungen in Python [[21](#_bookmark128)].

Seaborn ist eine Python-Datenvisualisierung-Bibliothek, die auf Matplotlib basiert. Sie bietet eine High-Level-Schnittstelle zum Zeichnen attraktiver und informativer statistischer Grafiken. Die Plot-Funktionen arbeiten mit Daten-Frames und Arrays, die Datensätze enthalten, und führen intern die notwendige semantische Zuordnung und statistische Aggregation durch, um Diagramme zu erstellen [[60](#_bookmark157)]. Beide Bibliotheken wurden in dieser Arbeit für die explorative Analyse für das Empfehlungssystem verwendet.

* + 1. **Scikit-learn**

Scikit-learn ist eine Python-Bibliothek für maschinelles Lernen, die auf NumPy und SciPy basiert. Sie bietet fortgeschrittene Implementierungen beliebter Algorithmen für maschinelles Lernen [[6](#_bookmark114)]. In dieser Arbeit wurde Scikit-learn für die Umsetzung des Ob- jektbasierten Filtern verwendet.

* + 1. **Moviepy**

MoviePy ist ein Python-Modul für die Videobearbeitung, das für grundlegende Operationen auf Videos geeignet ist[[7](#_bookmark115)]. In dieser Arbeit wurde das Modul für die Audio- Feature-Extraktion angewendet.

* + 1. **CSV**

Das CSV ist ein Python-Modul zum Lesen und Schreiben von tabellarische Daten im CSV-Format[[2](#_bookmark109)]. In dieser Arbeit wurde es für das Lesen und Erzeugen der CSV-Dateien verwendet.

* + 1. **Keras**

Keras ist eine Open-Source-Bibliothek, die eine Python-Schnittstelle für künstliche neu- ronale Netze bietet. Keras enthält zahlreiche Implementierungen von häufig verwende- ten Bausteinen für neuronale Netze. Es unterstützt Backends wie TensorFlow, Microsoft Cognitive Toolkit und Theano [[5](#_bookmark113)]. In dieser Arbeit wird diese Bibliothek für Szeneklassifikation verwendet.

* + 1. **FFmpeg**

FFmpeg ist ein Modul, das aus Bibliotheken und Programmen zur Verarbeitung von Video-, Audio- und anderen Multimediadaten besteht. Es verfügt über ein Kommando- zeilenprogramm, das für die Verarbeitung von Video- und Audiodaten in Python auf- gerufen werden kann [[1](#_bookmark110)]. In dieser Arbeit, wurde FFmpeg für das Schneiden der Videos verwendet.

* 1. **Daten**

Für die Implementierung sind Daten erforderlich. Es wurde vom Forschungsinstitut Herman-Hollerith-Zentrum Vorlesungsaufzeichnungen zur Verfügung gestellt, die sich aus verschiedenen Studienfächern der Bachelor- und Masterprogramme zusammensetzen. Diese Online-Vorlesungen wurden von verschiedenen Dozenten gehalten und mit dem Videokommunikationsprogramm Zoom gezeichnet. Die Beschriebene Daten wurden für die Implementierung verwendet.

* 1. **Struktur des Pakets**

Die Abbildung [7.1](#_bookmark78) stellt die Struktur des Pakets für die Implementierung dar. Die Implementierung besteht aus 8 Python-Dateien und 4 Jupiter Notebooks. Die Datei *main.py*

ist das Startprogramm, für die Zerlegung der Aufzeichnung. Als Parameter erwartet diese Datei ein Verzeichnis wo die MP4-Vorlesungsaufzeichnungen zu finden sind. Im gleichen Verzeichnis wird das Ergebnis des Systems abgelegt. Die Datei *SceneDetector.py* ist das zentrale Skript, welcher für die Szenedetektion, die Szeneklassifikation und das Schneiden der Videos zuständig ist. Dieses Skript verwendet Hilfsdateien *audioFeature- sUtils.py* und *VisualFeaturesUtils.py*, die jeweils Funktionen für die Audio- und visuell Videoanalyse zur Verfügung stellen. Es verwendet die Datei *sceneClassificator.py* für die Szeneklassifikation und *cutVideo.py* für das Schneiden der Videos. Die Datei *utils.py* bietet allgemeinen Hilfsfunktionen für das Programm an. Die Datei *generateLMSin- teractiondata.py* ist für die Generierung der LMS Daten zuständig. Die Jupiter No- tebooks *recommenderSystem\_dataPreparation.ipynb*, *recommenderSystem\_item-based- collaborative-filtering.ipynb*, *recommenderSystem\_user-based-collaborative-filtering.ipynb* und *recommenderSystem\_modell-based-collaborative-filtering.ipynb* implementieren das Empfehlungssystems.

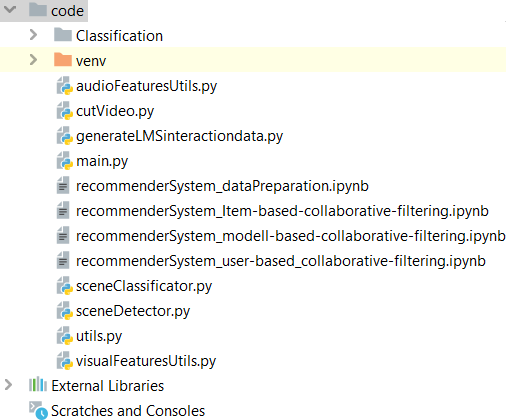


Abbildung 7.1: Struktur des Pakets

* 1. **Visuell Feature-Extraktion**

Nachdem alle erforderlichen Bibliotheken in PyCharm importiert sind, beginn die Implementierung mit der Feature-Extraktion. Die MP4-Vorlesungsaufzeichnungen aus dem eingegebenen Verzeichnis werden durchsucht und nacheinander prozessiert. Jede MP4-

Datei wird mit der Methode *videoCapture* der OpenCV-Bibliothek ausgelesen, dann als Objekt in einer Variable *videoCap* gespeichert.

videoCap = cv2 . VideoCapture ( v i d e o f i l e )

Die Metadaten des Videos wie Bildfrequenz(*fps*) und Anzahl Frames(*totalFrames*) wer- den von diesem Objekt ausgelesen.

f p s = videoCap . get ( cv2 .CAP\_PROP\_FPS)

total Frames = i n t ( videoCap . get ( cv2 .CAP\_PROP\_FRAME\_COUNT) )

In einer While-Schleife findet die Feature-Extraktion statt. Jedes Frame des Videos wird mit der Methode *read* der OpenCV-Bibliothek ausgelesen.

succe s s , image = videoCap . read ( )

Dann wird das Frame zu 50% verkleinert, um die Dauer der weiteren Analyse zu reduzie- ren.

s c a l e \_pe r c e nt = 50

width = i n t ( image . shape [ 1 ] ∗ s c a l e \_pe r c e nt / 100 ) he i g ht = i n t ( image . shape [ 0 ] ∗ s c a l e \_pe r c e nt / 100 ) dim = ( width , heig ht )

t o t a l P i x e l s = width ∗ he i g ht image = cv2 . r e s i z e ( image , dim )

Im Weiteren Verlauf wird überprüft, ob das Eingangsvideo vom Typ *videoType1* ist. Es handelt sich um Videos wo gleichzeitig der Professor und das Whiteboard im Vollbild sind. Für diese Art von Videos wird für eine bessere Analyse eine Vorverarbeitung durch- geführt. Dabei wird vom Bild nur das Region Of Interest (ROI) extrahiert. Das ROI ist der Bereich mit dem Whiteboard. Dabei wird angenommen, dass das Whiteboard, der Bereich mit der größten Kontur im Bild ist. Zur Umsetzung wurde die Methode *getRegio- nOfInterest* implementiert, welche die Methoden *findContours* der OpenCV-Bibliothek und die Funktion *max* der Builtins-Bibliothek anwendet.

def g e t Re g i o n Of In te r e s t ( image ) : lower = np . array ( [ 3 0 , 30 , 3 0 ] )

higher = np . array ( [ 2 5 0 , 250 , 2 5 0 ] )

mask = cv2 . inRange ( image , lower , higher )

contours , h i e r a r c h y = cv2 . f ind Contours ( mask , cv2 .RETR\_EXTERNAL, cv2 .CHAIN\_APPROX\_NONE)

img = cv2 . drawContours ( image , contours , −1, 255 , 3 ) c = max( contours , key=cv2 . contour Area )

x , y , h , w = cv2 . bounding Rect ( c )

image = cv2 . r e c t a n g l e ( img , ( x , y ) , ( x + w, y + h ) , ( 0 , 255 , 0 ) , 5 ) image = image [ y : y + h , x : x + w]

return image

Abbildung.

Nach Extraktion des ROI werden von diesem Bild Hintergrundinformationen entfernt und die Texte im Bild hervorgehoben. Hierfür wird die Methode *preprocess* verwen- det, welche die Methoden *cvtColor, GaussianBlur, Canny, dilate, erode* der OpenCV- Bibliothek nutzt.

def p r e p r o c e s s ( img ) :

imgGray=cv2 . cvt Color ( img , cv2 .COLOR\_BGR2GRAY) imgBlur=cv2 . Gaussian Blur ( imgGray , ( 5 , 5 ) , 1 ) imgCanny=cv2 . Canny( imgBlur , 2 0 0 , 2 0 0 )

k e r n e l=np . ones ( ( 5 , 5 ) )

img Dial=cv2 . d i l a t e ( imgCanny , kernel , i t e r a t i o n s =2) imgThres=cv2 . erode ( imgDial , kernel , i t e r a t i o n s =1) return imgThres

Abbildung.

Nach der Vorverarbeitung wird die Farbhistogramm des Bildes mit der Methode *calcHist*

der OpenCV-Bibliothek berechnet.

histogram = cv2 . c a l c H i s t ( [ image ] , [ 0 ] , None , [ 2 5 6 ] , [ 0 , 2 5 6 ] )

Anhang des Farbhistogramms wird der Euklidische Abstand zwischen dem Bild und den vorherigen berechnet. Dazu wird zunächst die Farbmoment des aktuellen Bildes berechnet. Farbmomente sind Maße, die die Farbverteilung im Bild charakterisieren. Dazu gehören die Farbdurchschnitt, die Standardabweichung und die Schräglage(skewness). Zur Berechnung dieser Informationen werden die Methode *pow* und *sqrt* der Math-Bibliothek verwendet.

f o r p i x e l s in histogram : sum Pixels += p i x e l s

color Mean = f l o a t ( sum Pixels / t o t a l P i x e l s ) f o r p i x e l s in histogram :

sum Pixels Cubes += math . pow( p i x e l s − colorMean , 3 ) sum Pixels Squares += math . pow( p i x e l s − colorMean , 2 )

va r i a n c e = f l o a t ( sum Pixels Squares / t o t a l P i x e l s ) s td Deviation = math . s q r t ( va r i a n c e )

avgSumOfCubes = f l o a t ( sum Pixels Cubes / t o t a l P i x e l s ) skewness = f l o a t ( avgSumOfCubes ∗∗ ( 1 . / 3 . ) )

Anschließen wird die Euklidische Abstand berechnet. Diese ergibt sich aus der Differenz der Farbmomente zwischen den aktuellen Bild und das vorherige. Hierzu wird die Me- thode *getEuclideanDistance(currentColorMoments, previousColorMoments)* verwendet.

Neben dem Euklidischen Abstand wird die Entropie des Bildes mit der Methode *ge- tEntropy(histogram, totalPixels)* berechnet. Die Entropie repräsentiert die durchschnitt- liche Information des Bildes. Hierzu kommt die logarithmische Funktion *log* der Math- Bibliothek zum Ansatz.

Nachdem alle Frames prozessiert wurden, werden die extrahierte Features verarbeitet. Zunächst wird die Durchschnitt Distanz(*meanEuclideanDistance*) zwischen alle Frames berechnet. Der berechnete Wert wird als Schwellenwert für die Erkennung der Bildänderung im Video angesetzt. Des Weiteren werden die Zeitliche Features Sekunde Weise untersucht, um mögliche Transitionen zu finden. Eine Sekunde repräsentiert ein Set von *x* Bilder, welches durch die Bildauflösung *fps* bestimmt ist. Diese Auflösung ist vom Aufnahmekamera abhängig. Zur Ermittlung der Transition wird die Liste der Euklidische Distanzen durchgegangen. Innerhalb jeder Sekunde wird der Abstand der Bilder mit dem zuvor berechneten Schwellenwert verglichen. Sollte der Abstand den Schwellenwert übersteigen, dann wird das Bild mit dem nachfolgenden Bild verglichen. Sollte der Abstand und die Entropie-Differenz der beiden Bilder den Schwellenwert von jeweils 1 und 2 übersteigen, dann wird das Bild Schlüsselbild betrachtet. Seine Position im Eingangsvideo wird als Transition-Position gespeichert. Diese Schwellwerte 1 für die Euklidische Distanz und 2 für die Entropie wurden ermittelt. Das Ergebnis dieses Schritts ist eine zeitliche Liste mit Werten 0 und 1, wobei 0 für keine Transition steht und 1 für Transition.

Abbildung list 0,1

* 1. **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 Bibliothek *moviepy*

ausgelesen, dann seine Audiospur mit *write\_audiofile* derselbe Bibliothek temporäre als WAV-Datei gespeichert.

video = moviepy . e d i t o r . Vi d e o Fi l e Cli p ( v i d e o Fi l e ) audiodata = video . audio

audiodata . w r i t e \_a u d i o f i l e ( " audio . wav " )

Dann wird die WAV-Datei mit der Methode *wavfile.read* der Scipy-Bibliothek ausgele- sen. Das Ergebnis des Auslesens ist ein numpy-Array von Audio-Frame und das Audio Sampling Rate. Die Audio-Frames repräsentieren die Amplituden des Audio-Signals.

audio\_rate , audio = s c i p y . i o . w a v f i l e . read ( " audio . wav " ) channel 1 = audio [ : , 0]# l e f t

Die Abbildung x ist ein Plot der Gelesene Daten.

Die Audiospur und das Video unterschiedliche Sampling rate haben, wird im weiteren Schritt versucht beide Spuren synchronisieren, um die zwei Features zeitliche parallel darstellen zu können. Dazu wird bei dem Audio nur jedes k Frame, mit *k=audio\_rate/video\_rate*

f o r i in range ( 0 , l e n ( channel 1 ) ) :

i f i % ( audio\_rate / f p s ) == 0 : channels . append ( channel 1 [ i ] )

Schließlich wird in der Audiospur nach Pausen gesucht. Als Pause werden Bereiche mit Ruhezeit von mindesten einer Sekunde betrachtet. Als Ruhezeit werden Amplitude zwi- schen -6 und +6 betrachtet.

Das Ergebnis von diesem Schritt ist eine Liste mit 0 und 1 wobei 0 für Pause Steht und 1 für keine Pause(Abbildung).

* 1. **Erkennung von interessante Szeneänderung**

Bisher wurden audiovisuelle Features extrahiert und auf Basis deren Szeneänderung und Pausen erkannt. In diesem Schritt werden beide Features kombiniert, um Interessante Szeneänderungen zu finden. Dazu werden zeitliche Features parallel repräsentiert. Positionen, wo Ereignisse gleichzeitig auf Audio und Visuelle Features festgestellt werden, bekommen ein hohes Scoring. Wenn z.B. ein Folienwechsel stattfindet und gleichzeitig, eine Pause zu sehen ist, bedeutet es wahrscheinlich, dass etwas Interessantes passiert. Es ist auch der Fall, wenn ein Folienwechsel stattfindet und direkt davor oder danach

eine kurze Pause festgestellt ist. Es ist öfter der Fall beim Themenänderung während einer Vorlesung. Dabei wird als Pause Ruhezeit von mindestens 3 Sekunden betrachtet.

f o r i in range ( 0 , l e n ( change Timeline ) ) : i f change Timeline [ i ] == 1 :

i f ( i >0 and ( breakTime Line [ i −1] == 1 ) ) or

( i <l e n ( breakTimeLine)−1

and breakTime Line [ i +1] ==1): interesting Changes Time Line . append ( 1 )

e l s e :

interesting Changes Time Line . append ( 0 )

e l s e :

interesting Changes Time Line . append ( 0 )

* 1. **Szeneklassifizierung**

Im vorherigen Abschnitt wurde interessante Szenen erkannt und die entsprechenden Positionen in Eingangsvideo in einer Liste gespeichert. In diesem Abschnitt werden diese Szenen klassifiziert. Dazu wird pro Szene 3 Bilder extrahiert: das erste, mittlere und letzte Bild. Diese Bilder werden dem Klassifikator übergeben. Mit der load\_img Funktion der Keras-Bibliothek, werden diese gespeicherte Bilder ausgelesen, dann in ein Bild- Array mit der img\_to\_array Methode derselbe Bibliothek umgewandelt. Der Klassifikator Vergleich die Bilder mit einer Datenbank, die aus Vorlesungsbilder besteht. Das verwendete vor-trainierte Modell stammt aus einer Vorarbeit zum vorliegenden Thema. Zum Abschluss des weist der Klassifikator dem Bild eine Klasse zu. Die definierte Klas- sen(Slide, Whiteboard, Professor und Others) in Kapitel 6.7 werden hierzu verwendet.

Abbildung(klass result)

Auf Basis des Ergebnisses der Klassifikation, werden die Szenen, die der Klasse *fault* zu- gewiesen wurden von der finale Liste der Szeneänderungen entfernt.

* 1. **Schneiden der Videos**

Zum Schneider der Videos wird die finale Liste der Szeneänderungen verwendet, welche einzelne Schnittpositionen enthält. Um zu Schneiden, wird das Kommandozeile- Programm *ffmpeg* angewendet, welcher als Parameter den Start und Endpositionen für

die Szenen verwendet. Die Start- und Endpositionen werden in Sekunden umgerechnet. Die Funktion *system* der Bibliothek *os* wird genutzt, um die Kommandozeile auszuführen.

s ta r t Po s = s t r ( segments [ x ] / f l o a t ( f p s ) )

endPos = s t r ( segments [ x + 1 ] / f l o a t ( f p s ) − segments [ x ] / f l o a t ( f p s ) ) cmd = " ffmpeg −i " + v i d e o f i l e + " −s s " + s ta r t Po s + " −t " +

endPos + " −c copy −map 0 " + output\_dir + scene\_name + "−%d " % index + " . mp4 "

os . system ( cmd)

Nachdem die Videos geschnitten sind, wird der Vorlesungsskript in Distributionspfad er- stellt.

Abbildung

* 1. **Interaktionsdaten vom LMS**

Die zu verwendete LMS Daten wurden im Kapitel 6.9 beschrieben. Ebenso wurde in genannten Abschnitt die 4 Benutzerprofile definiert (Faule Studierende, Mittel-faul, Mittel- Student, und gute Studierende). Auf Basis dieser Informationen und der Vorlesungsskript, wird in diesem Schritt Benutzer mit ihrer Interaktionsdaten generiert. Dazu wird für jedes Profil arbiträre Wertebereiche definiert. Für das Profil mittlere Studierende

z.B. schauen Studierende zwischen 40 bis 60% der gesamten Länge von Lerneinheit vom Typ Professor, 50 bis 60% für der Typ Whiteboard und 50 bis 80% für der Type Silde. Studierende dieses Profil schicken 1 bis 2 Nachrichten in Forum über die Lerneinheit. Anhang diese Wertebereiche, wird der Vorlesungsskript vom letztem Abschnitt ausgelesen und pro Profil Zufallsdaten für der erste Benutzer mithilfe der Methode *randint* der Random-Bibliothek generiert.

f o r x in range ( 0 , l e n ( s c e n e \_l i s t ) ) :

i f s c e n e \_l i s t [ x ] . get ( " type " ) == " Pr o f e s s o r " :

l e ar n i n g Du r ati o n = random . randint ( p r o f i l e s [ i ] [ 0 ] [ 0 ] [ 0 ] , p r o f i l e s [ i ] [ 0 ] [ 0 ] [ 1 ] )

e l i f s c e n e \_l i s t [ x ] . get ( " type " ) == " S l i d e s " :

l e ar n i n g Du r ati o n = random . randint ( p r o f i l e s [ i ] [ 0 ] [ 1 ] [ 0 ] , p r o f i l e s [ i ] [ 0 ] [ 1 ] [ 1 ] )

e l s e :

l e ar n i n g Du r a t i o n = random . randint ( p r o f i l e s [ i ] [ 0 ] [ 2 ] [ 0 ] , p r o f i l e s [ i ] [ 0 ] [ 2 ] [ 1 ] )

values\_learning Duration . append ( l e a r n i n g Du r ati o n )

numberOfPosts = random . randint ( p r o f i l e s [ i ] [ 1 ] [ 0 ] , p r o f i l e s [ i ] [ 1 ] [ 1 ] ) ;

Damit alle Benutzer jedes Profil stark korrelieren, werden die restliche Benutzer auf Basis des ersten generiert. Genauer gesagt, werden die andere Benutzer vom ersten geklont, sodass sie unterschiedliche Werte haben, blieben aber im selben Wertebereich. Dazu wird die Methode *randn* der *Mtrand*-Bibliothek verwendet.

err\_rec\_learning\_duration = 50 ∗ randn ( l e n ( s c e n e \_l i s t ) ) + 1 err\_rec\_rec\_post = 0 . 5 ∗ randn ( l e n ( s c e n e \_l i s t ) ) + 1 record\_learning\_duration = f i r s t\_r e c o r d \_l e a r n i n g \_d u r a t i o n + err\_rec\_learning\_duration

record\_number\_post = f irst\_record\_number\_posts + err\_rec\_rec\_post

Schließlich werden die generierte Daten in einer CSV-Datei gespeichert. f o r x in range ( 0 , l e n ( s c e n e \_l i s t ) ) :

w r i te r . writerow ({ f i e l d n a m e s [ 0 ] : user\_id ,

f i e l d n a m e s [ 1 ] : s c e n e \_l i s t [ x ] . get ( " scene Id " ) ,

f i e l d n a m e s [ 2 ] : i n t ( record\_learning\_duration [ x ] ) , f i e l d n a m e s [ 3 ] : i n t ( record\_number\_post [ x ] ) ,

* 1. **Kollaboratives Filtern**

Dieser Abschnitt veranschaulich die Implementierungsdetails zum kollaborativen Empfehlungssystem. Nachdem die nötigen Bibliotheken in das Jupiter Notebook importiert wurden, wird zuerst versucht die zu verarbeitende Daten zu verstehen.

* + 1. **Data Unterstanding**

Der Vorlesungsskript vom Abschnitt 7.8 und die generierte LMS Interaktionsdaten vom Abschnitt 7.9 werden im Notebook mit der Methode *read\_csv* der Panda Bibliothek geladen und in jeweiligen panda dataframes gespeichert.

movies\_df = pd . read\_csv ( ’ d i s t r i b u t i o n / movies . csv ’ )

rati ngs \_df = pd . read\_csv ( ’ d i s t r i b u t i o n / movie\_ratings . csv ’ )

Zur Datenvisualisierung der ersten Daten wird Methode *head* der Panda-Bibliothek verwendet.

movies\_df . head ( )

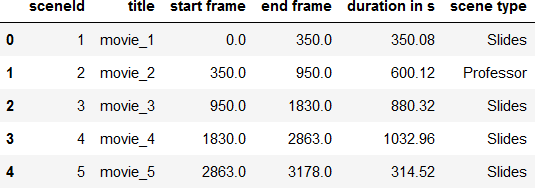


Abbildung 7.2: Vorlesungsskript

Es ist festzustellen, dass der Vorlesungsskript 6 Spalten hat: *sceneId, title, start frame, end frame, duration in s, scene type*.

rati ngs \_df . head ( )

Mit der Methode *describe* dieser Bibliothek kann die deskriptive Statistik dieser Daten ausgegeben werden(Abbildung [7.4](#_bookmark89)).

movies\_df . d e s c r i b e ( i n c l u d e =’ a l l ’ )

Es ist festzustellen, dass die zu analysierende Online-Vorlesung aus 12 Szenen besteht, die von 1 bis 12 nummeriert sind. Die Szenen sind die Lerneinheiten. Jede Lerneinheit dauert im Durchschnitt 454 Sekunden (7.5 Minuten). Die längste Lerneinheit dauert

c.a.19 Minuten. Die Mehrheit der Szenen sind PowerPoint-Folien *(Slide*).

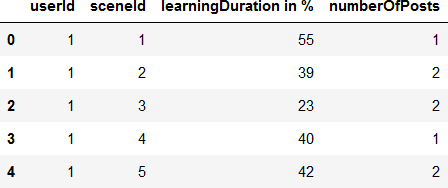


Abbildung 7.3: LMS Interaktionsdaten

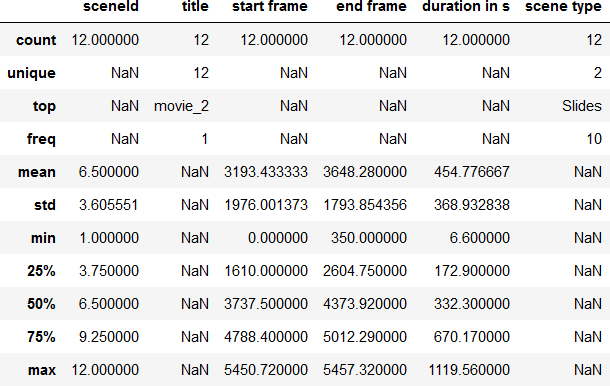


Abbildung 7.4: Deskriptive Statistik für der Vorlesungskript

Bei LMS Interaktionsdaten bestehen die Tabelle aus 5 Spalten: *userId, sceneId, lear- ningDuration in %, numberOfPosts*

Mit der beschreibenden Statistik kann die Tabelle weiter analysiert (Abbildung [7.6](#_bookmark91)). Es ist festzustellen, dass die Tabelle aus 240 Zeilen besteht. Es liegt daran, dass jede Studierende durch 12 Zeilen repräsentiert ist, wobei die einzelnen Zeilen sind die Bewertungen für jede Lerneinheit. Aus der Statistik ist es zu entnehmen, dass die Studierenden im Durchschnitt 38% jeder Lerneinheit anschaut. 75% der Studierenden schauen 57% jeder Lerneinheit an. Es gibt Lerneinheiten, die gar nicht angeschaut wurden. Diese werden

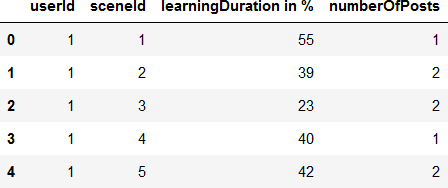
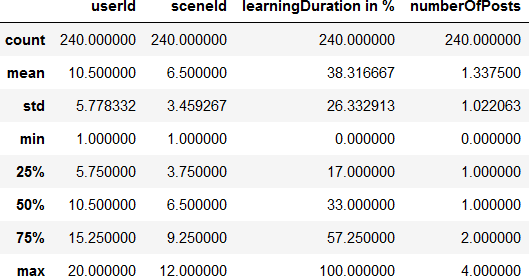


Abbildung 7.5: LMS Interaktionsdaten

im nächsten Verlauf las unbekannte Lerneinheiten betrachtet, welche vom System empfohlen werden sollen. was das Feature „numberOfPosts“ betrifft, ist festzustellen, dass maximal 4 Posts in Forum für jede Lerneinheit geschickt wurde. Im Durchschnitt schickt jede Studierende 1 Posts pro Lerneinheit.

Abbildung 7.6: Beschreibende Statistik der LMS Interaktionsdaten



* + 1. **Data Preparation**

Nachdem ein Überblick über die zu verarbeiten Daten verschlaft ist, werden einige Vor- bereitungsschritte auf die Daten durchgeführt. Dazu gehört die Prüfung nach redundanten Daten mit Funktion *duplicated* der Panda-Bibliothek. Außerdem werden die implizite Da ten in explizit konvertiert, um Standard kollaborative Algorithmen anwenden zu können. Wie schon in Kapitel 4(Empfehlungssysteme) beschrieben, erwarten diese Algorithmen Daten in Form eines Tupels (Benutzer, Objekt, Bewertung). Um dazu zu gelangen, werden die beide Spalten „learningDuration in %“ und „numberOfPosts“ in explizite Bewertung konvertiert und zusammengefügt. Dazu werden beide Features zuerst in Wertebereich [0,1] normiert, dann gewichtet. Zur Normierung wird beim *learningDuration in %* der Wert durch 100 geteilt. Das heißt jede Lerneinheit, die vollständig angeschaut ist, be- kommt 1-Bewertungspunkt. Was das Feature *numberOfPosts* angeht, wird es logarithmisch normiert(siehe unterstehende Gleichung).

1 + *numberOf P osts*

( ) (7.1)

*log* 1 + *MAX*(*numberOf P osts*

Der Ansatz von *MAX*(*numberOf P osts*) in dieser Gleichung kommt, dadurch dass die

maximal Anzahl an Posts nicht beschränkt ist. Deswegen ist der Wert vom Datensatz abhängig. Nach der Normierung werden die genante Features Features mit entsprechen- de Gewichte multipliziert. Da *learningDuration* eine bessere Aussage über die Präferenz für eine Lerneinheit gegenüber *numberOfPosts* gibt, wird die Gewichtung 5 *learning- Duration* zugewiesen und *numberOfPosts* Gewichtung 2. Damit liegt *learningDuration* im Wertebereich [0,5] und *numberOfPosts* im Bereich [0,2]. Nachdem beide Features in einer expliziten Bewertung konvertiert sind, können sie einzeln als Bewertung betrachtet werden oder sie können kombiniert werden, um ein einziges Feature zu bilden. Dadurch entsteht das neue Feature *rating*. Im weiteren Verlauf wird zuerst nur das *learning- Duration* als Bewertung betrachtet werden, danach beide Features durch die Addition kombiniert, um den Einfluss vom neuen Feature in das System zu bewerten.

def convert To Explicit Rating ( ) : number\_post\_weight=2 learning\_duration\_weight=5

l earning\_duration\_vector=rati ng s\_df [ ’ l e ar n i n g Du r ati o n in % ’] l earning\_duration\_vector =( l earning\_duration\_vector / 100 ) ∗ learning\_duration\_ number\_post\_vector = r ating s\_df [ ’ numberOfPosts ’ ]

number\_post\_vector = ( np . l o g (1+number\_post\_vector )/ math . l o g (1+max( number\_post\_vector ) ) ) ∗ number\_post\_weight rati ng s \_df [ ’ rating ’ ] = rating\_vector

return r atings \_df

e xp l i c i t\_d a ta= convert To Explicit Rating ( )

Nach der Konvertierung werden unnötige Daten gelöscht.

e xp l i c i t\_d a ta = e xp l i c i t\_d a ta . drop ( ’ numberOfPosts ’ , 1 )

e xp l i c i t\_d a ta = e xp l i c i t\_d a ta . drop ( ’ l e ar n i n g Du r a t i o n in % ’ , 1 ) e xp l i c i t\_d a ta . to\_csv ( ’ data\_preprocessed . csv ’ )

Das neue Feature *rating* wird in Abbildung [7.7](#_bookmark93) mit Hilfe der Seaborn-Bibliothek darge- stellt werden.

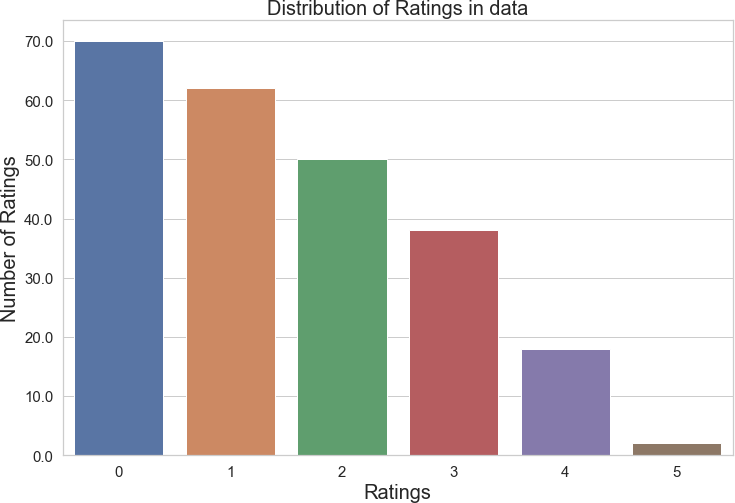


Abbildung 7.7: Ergebnis der Konvertierung

Zum Abschluss der Datenvorbereitungsphase, wird das Ergebnis in einer CSV-Datei mit der Funktion *to\_csv* der Panda-Bibliothek gespeichert.

e x p l i c i t\_d a ta . to\_csv ( ’ data\_preprocessed . csv ’ )

* + 1. **Modeling**

Bisher wurde die Daten vorbereitet. In diesem Schritt wird das eigentliche kollaborative Filtern umgesetzt.

##### User-Based

Die Abbildung [7.8](#_bookmark96) beschreibt die Implementierungsschritte. Es startet mit dem Laden des vorbereiteten Datensatz in ein Panda Dataframe.

prepared\_data = pd . read\_csv ( ’ data\_preprocessed . csv ’ )

Als nächstes wird ein Benutzer-Objekt-Matrix mit der *pivot* Funktion der Panda-Bibliothek gebildet.

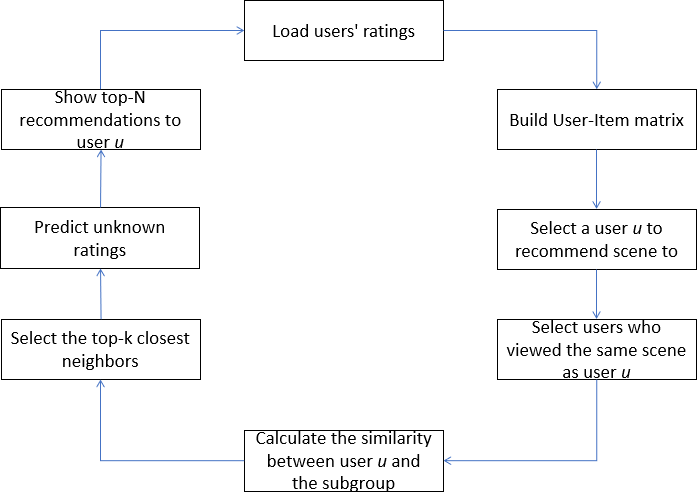


Abbildung 7.8: Benutzer-basiertes kollaboratives Filtern Algorithmus

movie\_matrix = data . pivot ( index =’ user Id ’ , columns=’ scene Id ’ , va l u e s =’ rating ’ ) . f i l l n a ( 0 ) . astype ( i n t )

Die Abbildung [7.9](#_bookmark97) stellt die ersten Einträge der Matrix dar. Wie schon erwähnt, sind Objekte mit 0-Bewertungswert, die unbekannten Objekte für der Benutzer, deren Bewertungswerte vorhergesagt werden.

Nachdem die Benutzer-Objekt-Matrix gebildet ist, wird ein aktiver Benutzer gewählt. Der Benutzer mit der ID 2 wurde als aktiver Benutzer ausgewählt. In seiner Bewertungsliste befindet sich zwei unbekannte Objekte. Dieser Benutzer gehört zum Profil *good\_students.* Wie in der Matrix zu sehen ist, sind Benutzer dieses Profil sehr aktiv beim Anschauen der Lerneinheiten*.* Als Erinnerung wurde im Abschnitt 7.9 pro Benutzerprofil 5 Benutzers generiert. Das heißt der aktive Benutzer ist im selben Profil mit Benutzer der erste 5 Benutzer.

s e l e c te d \_u s e r=2

input Movies=data . groupby ( ’ user Id ’ ) . get\_group ( s e l e c te d \_u s e r )

Als nächstes wird ein Sub-Set von Benutzer gebildet, die dieselbe Lerneinheiten wie der Aktive Benutzer, gesehen bzw. bewertet haben.

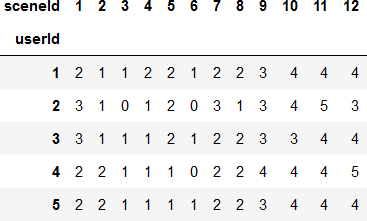


Abbildung 7.9: Student-Lerneinheit-Matrix

user Subset = data [ data [ ’ scene Id ’ ] . i s i n ( input Movies [ ’ scene Id ’ ] . t o l i s t ( ) ) ] user SubsetGroup = user Subset . groupby ( [ ’ user Id ’ ] )

user SubsetGroup = s o r te d ( user SubsetGroup , key=lambda x : l e n ( x [ 1 ] ) , r e v e r s e=T

Danach findet eine Ähnlichkeitsberechnung zwischen diesen Subset und der aktive Benutzer Anhang Pearson Korrelation angesetzt. Dazu wird die Funktion *corr\_with* der Panda Bibliothek verwenden.

Auf Basis dieser Berechnung wird das Ergebnis sortiert und die Top-k Elemente der Liste ausgegeben. Da der Wert von *k* einen Einfluss auf die Genauigkeit des Algorithmus hat, ist es einstellbar. Der wert 5 wurde für *k* ausgewählt.

top Users=pearsonDF . sort\_values ( by=’ s i m i l a r i ty I n d e x ’ , ascending=False ) [ 0 : top\_k ]

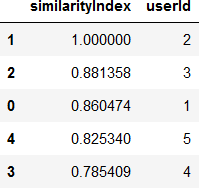


Abbildung 7.10: Ähnliche Benutzer zu Benutzer 2

Wie zu sehen ist, korreliert der Benutzer mit ID 2 stark mit anderen Benutzern desselben Profil. Er korreliert mit sich selbst zu 100%. Damit keine Bias in dem Ergebnis

der Empfehlung entstehen wird dieser Benutzer von der Liste der Ähnliche Benutzer entfernt.

top Users=pearsonDF . sort\_values ( by=’ s i m i l a r i ty I n d e x ’ , ascending=False ) [ 1 : top\_k\_users ]

Nachdem die Top-k Ähnliche Benutzer bekannt sind, wird auf Basis Ihrer Bewertun- gen die Bewertung des aktiven Nutzer für alle Objekte berechnet. Zunächst wird die Bewertungen dieser Benutzer herangezogen.

top Users Rating=top Users . merge

( data , l e f t\_on =’ user Id ’ , right\_on =’ user Id ’ , how=’ inner ’ )

Dann mit der Berechnete Ähnlichkeit multipliziert. Dadurch entsteht das Feature *weigh- tedRating*. Für den weiteren Verlauf wird nur Benutzer mit einer Positiven Korrelation betrachtet.

top Users Rating [ ’ weighted Rating ’ ] = top Users Rating [ ’ s i m i l a r i ty I n d e x ’ ]

∗ top Users Rating [ ’ rating ’ ]

top Users Rating=top Users Rating [ top Users Rating [ ’ s i m i l a r i ty I n d e x ’ ] > 0 ]

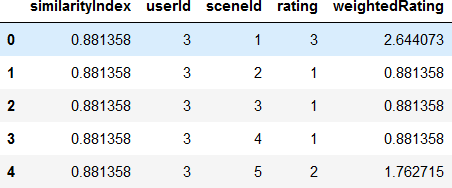


Abbildung 7.11: Gewichtete Bewertung

Danach wird die Sum-Funktion der Panda Bibliothek verwendet um das Ergebnis zu summierren, nachdem dieses pro Benutzer gruppiert war.

tempTopUsersRating = top Users Rating . groupby ( ’ scene Id ’ ) . sum ( ) [ [ ’ s i m i l a r i ty I n d e x

Als Bewertungsvorhersage wird den gewichteten Durchschnitt der Bewertungen verwendet. Das Ergebnis wird dann Pro vorhergesagte Bewertung sortiert.

recommendation\_df = pd . DataFrame ( )

recommendation\_df [ ’ p r e d i c t i o n score ’ ] = tempTopUsersRating [ ’ sum\_weightedRatin

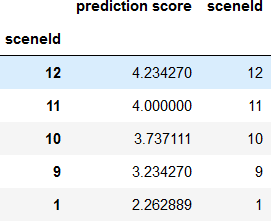


Abbildung 7.12: Vorhersage für alle Objekte

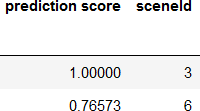


Abbildung 7.13: Ergebnis der Empfehlung

/ tempTopUsersRating [ ’ sum\_similarity Index ’ ]

Die Abbildung [7.12](#_bookmark100) wurden die Bewertungen des aktiven Nutzers für alle Objekte vorher- gesagt. Im nächsten Schritt werden die unbekannten Objekte gefiltert. Dazu werden diese zuerst ermittelt indem die Benutzer-Objekt-Matrix durchgegangen und alle Objekte mit dem Bewertungswert 0 für der aktive Benutzer herangezogen. Schlussendlich wird die finale Liste mit Top-N Empfehlungen ausgegeben, welche nur aus unbekannten Objekten besteht.

f inal\_recommendation . head ( top\_n\_items )

Wie in das Ergebnis zu sehen ist, wird dem aktiven Benutzer die bisher unbekannte Objekte empfohlen.

##### Item-Based

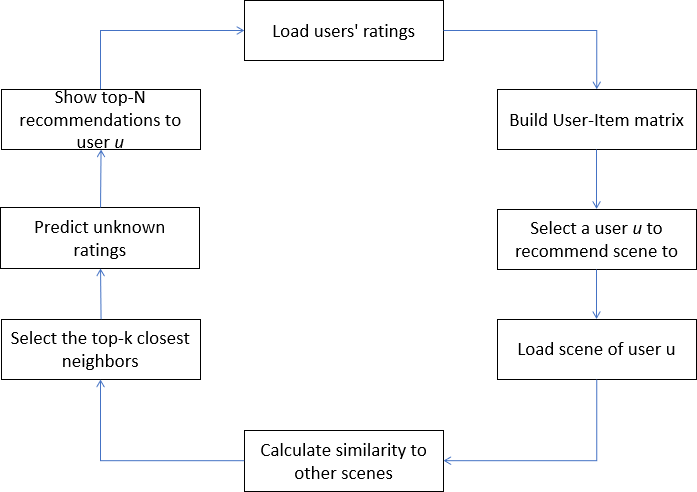


Abbildung 7.14: Objekt-basiertes kollaboratives Filtern

Die Implementierung dieser Methode erfolgt in einem separaten Jupiter Notebook. Die Abbildung [7.14](#_bookmark103) beschreibt die einzelnen Programmierungsschritte. Die Schritte 1 bis 4 wurden in vorherigen Abschnitt behandelt und gelten auch in dieser Methode. Im Schritt 5 findet die Ähnlichkeitsberechnung zwischen Objekte statt. Dafür wird die Klasse *Nea- restNeighbors* der Sklearn-Bibliothek verwendet, um das Algorithmus zu instanziieren. Als Metrik wird die Kosinusähnlichkeit und das Algorithmus *brute* verwendet.

k=5

cosine\_knn=Nearest Neighbors ( n\_neighbors=k , algorithm =’ brute ’ , metric =’ cosine ’

Des Weiteren wird das Modell auf dem Datensatz mit der Methode *fit* angelernt, dann mit der Methode *kneighbors* die nächste k Nachbarn von jedem Objekt zurückgegeben. Das Ergebnis ist ein Distanzvektor von jedem Objekt zu seinen Nachbarn, wobei die Distanzen zwischen 0 und 1 liegen. Je näher die Distanz zu 0, desto ähnlicher sind die Objekte.

item\_cosine\_knn\_fit=cosine\_knn . f i t ( movie\_matrix .T. va l u e s )

i tem\_distances , i tem\_indices=item\_cosine\_knn\_fit . kneighbors ( movie\_matrix .T. va l

Nach Ermittlung Ähnlicher Objekte werden die Top-N Empfehlungen mit der Methode get\_Top\_N\_recommendations berechnet.

def get\_Top\_N\_recommendations ( user Id , top\_n =10 ): p r i n t ( " Scenes viewed in the past : " )

p r i n t ( movies\_rated [ u s e r Id ] )

p r i n t ( " Recommendations : " . format ( u s e r Id ) ) f o r k , v in topRecommendations . i tems ( ) :

i f u s e r Id == k :

f o r i in v [ : top\_n ] :

p r i n t ( ’ scene {} with s i m i l a r i t y : { : . 4 f } ’ . format ( i [ 0 ] , 1 − i [ 1

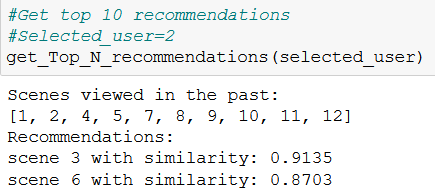


Abbildung 7.15: Ergebnis Objekt-basiert kollaboratives Filtern

##### Modell-based

Die Umsetzung des Modellbasierten Verfahren findet auch im Separaten Jupiter Notebook statt. Im ersten Schritt wird der vorbereitete Datensatz gelesen und in ein Pan da Dataframe *data* gespeichert. Dann wird das Dataframe mittels *load\_from\_df* der Surprise-Bibiliothek in ein *Surprise* Dataframe konvertiert*.* Im Näschen Schritt werden Algorithmen werden modellbasierte Algorithmen SVD und KNN der Surprise-Bibliothek auf den Datensatz angewendet.

reader = Reader ( r a t i n g \_s c a l e =(1 , 5 ) )

data\_df = Dataset . load\_from\_df ( data [ [ ’ user Id ’ , ’ scene Id ’ , ’ rating ’ ] ] , reader

##### SVD

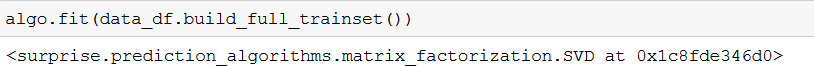
Singular Value Decomposition (SVD) ist ein gängiges Verfahren zur Dimensionalitätsreduktion in der Datenwissenschaft, welches auch für die Implementierung des Empfehlungssystems verwendet wird. Empfehlungssysteme die, dieses Verfahren verwenden,

folgen im Allgemeinen einem Muster, bei dem eine Matrix *A* von Bewertungen in ein Produkt von Matrizen(*A* = *U V T* ) zerlegt wird, die latente Faktoren für die Objekte( *V T* ) und die Benutzer(*U* ) darstellen[[4](#_bookmark112)]. Um das Algorithmus zu Instanziieren wird die Klasse *SVD* der Surprise-Bibliothek verwendet.

Σ Σ

a l go = SVD( )

Nach der Instanziierung wird das Modell auf dem gesamten Datensatz mit der fit- Methode angelernt, nachdem der Datensatz mit der build\_full\_trainset-Methode in ein Surprise-Trainset-Objekt umgewandelt ist.



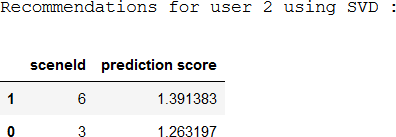
Dann werden die Bewertungen für zuvor ermittelte unbekannte Objekte für der aktive Nutzer mit der predict-Methode des Algorithmus vorhergesagt.

f o r movie\_id in movies\_to\_predict :

svd\_recomendations . append ( ( movie\_id , a l go . p r e d i c t ( uid=selected\_user , i i d=m svd\_rec=pd . DataFrame ( svd\_recomendations , columns =[ ’ scene Id ’ , ’ p r e d i c t i o n

Das Ergebnis wird sortiert dem Benutzer angezeigt.

p r i n t ( " Recommendations f o r user {0} using SVD : " . format ( s e l e c te d \_u s e r ) ) svd\_rec . sort\_values ( ’ p r e d i c t i o n score ’ , ascending=False ) . head ( top\_n\_items )



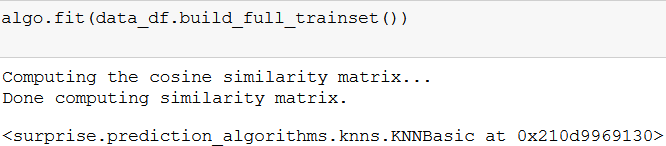
##### KNN

Dieses Algorithmus der Surprise-Bibliothek ist von grundlegendem Ansatz der nächsten Nachbarn (k-nearest neighbors) Algorithmus abgeleitet. Bei der Instanziierung des Algorithmus wird das sim\_options übergeben, welches Parameters für Konfiguration des Algorithmus enthält. Das Argument *name* legt fest welche Metrik das Algorithmus verwenden soll. Das Argument user\_based ist ein Boolean, welches festgelegt, ob die Benutzerbasierte oder die Objekt-basierte Methode verwendet wird.

sim\_options = {" name " : " c o s i n e " , " user\_based " : False } a l go = KNNBasic ( sim\_options=sim\_options )

Ebenso wie bei SVD, wird das Modell auf dem gesamten Datensatz mit der fit-Methode angelernt, nachdem den Datensatz mit der build\_full\_trainset-Methode in ein Surprise- Trainingset-Objekt umgewandelt ist.

a l go . f i t ( data\_df . b u i l d \_f u l l \_t r a i n s e t ( ) )



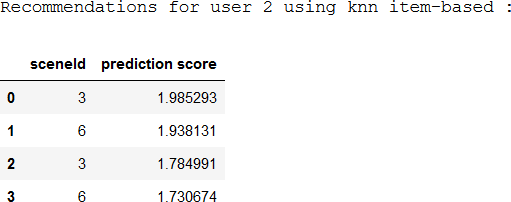
Die Bewertung der unbekannten Objekte für der aktive Nutzer wird mit der predict- Methode des Algorithmus vorhergesagt. Das Ergebnis wird sortiert ausgegeben.

f o r movie\_id in movies\_to\_predict : knn\_recommendations .

append ( ( movie\_id , a l go . p r e d i c t ( uid=selected\_user , i i d=movie\_id ) . e s t ) ) knn\_rec=pd . DataFrame ( knn\_recommendations , columns =[ scene Id , ’ p r e d i c t i o n s

p r i n t ( " Recommendations f o r user {0} using knn item−based : " . format ( s e l e c te d \_u

knn\_rec . sort\_values ( ’ p r e d i c t i o n score ’ , ascending=False ) . head ( top\_n\_items )



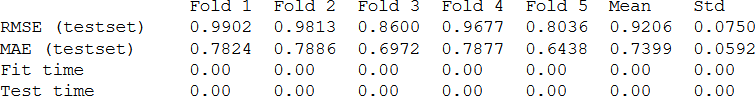
* + 1. **Evaluation**

Die Evaluation-Methode mit Kreuvalidierung wurde in Kapitel 6.10.4 beschrieben. Zur Umsetzung wird die Methode cross\_validate der Surprise-Bibliothek verwendet. Als

Parameter erwartet diese Methode das zu verwendete Algorithmus. Dies kann z.B. KNN oder SVD sein. Der zweite Parameter ist der zu evaluierte Datensatz. Als dritte Argument werden die Evaluationmetriken konfiguriert. Der vierte Parameter ist die Anzahl der k-folds. Als Wert für diesen Parameter wurde 5 ausgewählt.

c r o s s \_v a l i d a te ( a l go=algo , data=data\_df , measures =[ ’RMSE’ , ’MAE’ ] , cv=5 , verbos

Das Ergebnis der Evaluation wir tabellarisch ausgegeben. Die Abbildung stellt die Eva- luation mit KNN item-based Algorithmus dar. Eine detaillierte Evaluation folgt im nächsten Kapitel.



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 3. November 2021 [UNTERSCHRIFT]

[2.1 Methode der Literaturrecherche (Quelle: In Anlehnung an [](#_bookmark7)[16](#_bookmark122)[])](#_bookmark7) 7

* 1. [Ablauf des Algorithmus für das Kollaboratives Filtern(In Anlehnung an](#_bookmark22)

[[](#_bookmark22)[27](#_bookmark133)[])](#_bookmark22) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

* 1. [User-Item-Matrix](#_bookmark23) 19
  2. [Systeminfrastruktur](#_bookmark41) 29
  3. [Systemarchitektur](#_bookmark43) 30
  4. [Architektur des Empfehlungssystem](#_bookmark53) 35
  5. [Beispiel Benutzer-basiertes kollaboratives Filtern](#_bookmark57) 36
  6. [Beispiel Objekt-basiertes kollaboratives Filtern](#_bookmark59) 37
  7. [Struktur des Pakets](#_bookmark78) 43
  8. [Vorlesungskript](#_bookmark87) 51
  9. [LMS Interaktionsdaten](#_bookmark88) 51
  10. [Deskriptive Statistik für der Vorlesungskript](#_bookmark89) 52
  11. [LMS Interaktionsdaten](#_bookmark90) 52
  12. [Beschreibende Statistik der LMS Interaktionsdaten](#_bookmark91) 53
  13. [Ergebnis der Konvertierung](#_bookmark93) 55
  14. [Benutzer-basiertes kollaboratives Filtern Algorithmus](#_bookmark96) 56
  15. [Student-Lerneinheit-Matrix](#_bookmark97) 57
  16. [Ähnliche Benutzer zu Benutzer 2](#_bookmark98) 57
  17. [Gewichtete Bewertung](#_bookmark99) 58
  18. [Vorhersage für alle Objekte](#_bookmark100) 59
  19. [Ergebnis der Empfehlung](#_bookmark101) 59
  20. [Objekt-basiertes kollaboratives Filtern](#_bookmark103) 60
  21. [Ergebnis Objekt-basiert kollaboratives Filtern](#_bookmark104) 61
  22. [Trefferquote der ersten Suchstrings](#_bookmark8) 9
  23. [Ergebnis der Literaturrecherche](#_bookmark10) 10

[3.1 Unterschied zwischen statische und dynamische Zusammenfassung(Quelle:](#_bookmark16)

[[](#_bookmark16)[37](#_bookmark141)[] )](#_bookmark16) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

* 1. [Beispiele Realwelt Empfehlungssysteme(Quelle:In Anlehnung an [](#_bookmark19)[9](#_bookmark117)[])](#_bookmark19) 17
  2. [Klassifikationstabelle](#_bookmark32) 24

1. A complete, cross-platform solution to record, convert and stream audio and video.
2. Csv file reading and writing.
3. A gentle introduction to k-fold cross-validation.
4. How you can build simple recommender systems with surprise.
5. Keras: simple, flexibel, powerful.
6. scikit-learnmachine learning in python.
7. User guide.
8. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions* *on Knowledge and Data Engineering*, 17(6):734–749, 2005.
9. Charu C Aggarwal et al. *Recommender systems*, volume 1. Springer, 2016.
10. Mohannad AlMousa, Rachid Benlamri, and Richard Khoury. Nlp-enriched auto- matic video segmentation. In *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)*, pages 1–6, 2018.
11. Bitkom. Corona treibt digitalisierung voran – aber nicht alle unternehmen können mithalten, November 2020.
12. Katja Grashöfer Björn Bohnenkamp, Marcus Burkhardt. Online-lehre 2020 – eine medienwissenschaftliche perspektive. In *Hochschulforum Digitalisierung*, 2020.
13. Donald A Bligh. *What’s the Use of Lectures?* Intellect books, 1998.
14. Amit Bora and Shanu Sharma. A review on video summarization approcahes : Recent advances and directions. In *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, pages 601–606, 2018.
15. Fatiha Bousbahi and Henda Chorfi. Mooc-rec: A case based recommender system for moocs. *Procedia - Social and Behavioral Sciences*, 195:1813–1822, 2015. World Conference on Technology, Innovation and Entrepreneurship.
16. Jan vom Brocke, Alexander Simons, Bjoern Niehaves, Bjorn Niehaves, Kai Reimer, Ralf Plattfaut, and Anne Cleven. Reconstructing the giant: On the importance of rigour in documenting the literature search process. 2009.
17. Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling* *and user-adapted interaction*, 12(4):331–370, 2002.
18. The SciPy community. Scipy community.
19. Ivan Culjak, David Abram, Tomislav Pribanic, Hrvoje Dzapo, and Mario Cifrek. A brief introduction to opencv. In *2012 Proceedings of the 35th International Con-* *vention MIPRO*, pages 1725–1730, 2012.
20. NumPy developers. Numpy.
21. Matplotlib development team. Matplotlib: Visualization with python.
22. Benedikt Engelbert. *Maschinelle Generierung von Empfehlungen zur Lehr-*

*/Lernunterstützung im Hochschulkontext*. PhD thesis, Universität Osnabrück, 2016.

1. Python Software Foundation. The python tutorial.
2. Dan Fu, Qingtang Liu, Si Zhang, and Jianhu Wang. The undergraduate-oriented framework of moocs recommender system. In *2015 International Symposium on* *Educational Technology (ISET)*, pages 115–119, 2015.
3. David Gerhard, Paula Heidkamp, Alexandra Spinner, Bianca Sommer, Anika Sprick, Bianca A Simonsmeier, and Michael Schneider. Vorlesung. In *Gute Hoch-* *schullehre: Eine evidenzbasierte Orientierungshilfe*, pages 13–38. Springer, 2015.
4. Zameer Gulzar, A Anny Leema, and Gerard Deepak. Pcrs: Personalized cour- se recommender system based on hybrid approach. *Procedia Computer Science*, 125:518–524, 2018.
5. Yaqiong Guo, Mengxing Huang, and Tao Lou. A collaborative filtering algorithm of selecting neighbors based on user profiles and target item. In *2015 12th Web* *Information System and Application Conference (WISA)*, pages 9–14, 2015.
6. Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on In- formation Systems (TOIS)*, 22(1):5–53, 2004.
7. Donald D Hoffman and Manish Singh. Salience of visual parts. *Cognition*, 63(1):29– 78, 1997.
8. Xiaodi Hou, Jonathan Harel, and Christof Koch. Image signature: Highlighting sparse salient regions. *IEEE Transactions on Pattern Analysis and Machine Intel-* *ligence*, 34(1):194–201, 2012.
9. Nicolas Hug. Surprise: A python library for recommender systems. *Journal of Open* *Source Software*, 5(52):2174, 2020.
10. Shruti Jadon and Mahmood Jasim. Video summarization using keyframe extraction and video skimming. *arXiv preprint arXiv:1910.04792*, 2019.
11. Harshit Jain and Anika. Applying data mining techniques for generating moocs recommendations on the basis of learners online activity. In *2018 IEEE 6th Inter- national Conference on MOOCs, Innovation and Technology in Education (MITE)*, pages 6–13, 2018.
12. Gawesh Jawaheer, Martin Szomszor, and Patty Kostkova. Comparison of implicit and explicit feedback from an online music recommendation service. In *procee- dings of the 1st international workshop on information heterogeneity and fusion in* *recommender systems*, pages 47–51, 2010.
13. Jetbrains. The python ide for professional developers.
14. Jeevamol Joy and Renumol Vemballiveli Govinda Pillai. Review and classification of content recommenders in e-learning environment. *Journal of King Saud University-* *Computer and Information Sciences*, 2021.
15. Mahesh Kini M. and Karthik Pai. A survey on video summarization techniques. In *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*, volume 1, pages 1–5, 2019.
16. Stephan Kopf, Fleming L ampi, Thomas King, and Wolfgang E ffelsberg Malte P robst. Edl-editor: Eine anwendung zur a utomatischenaufbereitung v on vorle- sungsvideos. Lehrstuhlfür PraktischeInformatik IV.
17. Bhargava Urala Kota, Alexander Stone, Kenny Davila, Srirangaraj Setlur, and Venu Govindaraju. Automated whiteboard lecture video summarization by content region detection and representation. In *2020 25th International Conference on Pattern* *Recognition (ICPR)*, pages 10704–10711, 2021.
18. Benjamin Kühn. *Interessengetriebene audiovisuelle Szenenexploration*, volume 22. KIT Scientific Publishing, 2016.
19. Tobias Lauer and Stephan Trahasch. Begriffsbesprechung: Vorlesungsaufzeichnung:. 4(3):61–61, 2005.
20. Jie Lu. A personalized e-learning material recommender system. In *Internatio- nal Conference on Information Technology and Applications*. Macquarie Scientific Publishing, 2004.
21. M. Belz und J. Signerski-Krieger M. Besse, J. Wiltfang. Einführung digitaler leh- re im fach psychiatrie als reaktion auf covid-19: eine vergleichende evaluation zur präsenzlehre. *Nervenarzt*, 2021.
22. Olga Marino and Gilbert Paquette. A competency—driven advisor system for multi- actor learning environments. *Procedia Computer Science*, 1(2):2871–2876, 2010. Proceedings of the 1st Workshop on Recommender Systems for Technology Enhan- ced Learning (RecSysTEL 2010).
23. Ammar A. Neamah and Amer S. El-Ameer. Design and evaluation of a course re- commender system using content-based approach. In *2018 International Conference* *on Advanced Science and Engineering (ICOASE)*, pages 1–6, 2018.
24. Miguel Ángel Sicilia, Elena García-Barriocanal, Salvador Sánchez-Alonso, and Cris- tian Cechinel. Exploring user-based recommender results in large learning object repositories: the case of merlot. *Procedia Computer Science*, 1(2):2859–2864, 2010. Proceedings of the 1st Workshop on Recommender Systems for Technology Enhan- ced Learning (RecSysTEL 2010).
25. Chong-Wah Ngo and Feng Wang. *Video Summarization*, pages 3320–3324. Springer US, Boston, MA, 2009.
26. Jürg Niederhauser. *Die schriftliche Arbeit*. Dudenverl., 2011.
27. Douglas W Oard, Jinmook Kim, et al. Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems*, volume 83, pages 81–83. WoUongong, 1998.
28. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine* *Learning research*, 12:2825–2830, 2011.
29. Fernando Pérez and Brian Granger. The jupyter notebook.
30. The Pandas Project. Package overview.
31. Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.
32. Cristobal Romero and Sebastian Ventura. Educational data mining: A survey from 1995 to 2005. *Expert systems with applications*, 33(1):135–146, 2007.
33. Cristóbal Romero, Sebastián Ventura, and Enrique García. Data mining in course management systems: Moodle case study and tutorial. *Computers & Education*, 51(1):368–384, 2008.
34. Henk G Schmidt, Stephanie L Wagener, Guus ACM Smeets, Lianne M Keemink, and Henk T van der Molen. On the use and misuse of lectures in higher education. *Health Professions Education*, 1(1):12–18, 2015.
35. Dhruv Shah, Pratik Shah, and Asim Banerjee. Similarity based regularization for online matrix-factorization problem: An application to course recommender sys- tems. In *TENCON 2017 - 2017 IEEE Region 10 Conference*, pages 1874–1879, 2017.
36. Anusha Vimalaksha, Siddarth Vinay, Abhijit Prekash, and N. S. Kumar. Automated summarization of lecture videos. In *2018 IEEE Tenth International Conference on Technology for Education (T4E)*, pages 126–129, 2018.
37. Klaus Wannemacher. Organisation digitaler lehre in den deutschen hochschulen. In

*Hochschulforum Digitalisierung*, 2016.

1. Michael Waskom. An introduction to seaborn.
2. Fei Xu, Kenny Davila, Srirangaraj Setlur, and Venu Govindaraju. Content extrac- tion from lecture video via speaker action classification based on pose information. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1047–1054, 2019.