



Research Repository UCD

Title	Navigating Academia – Recommender Systems for Module Exploration
Authors(s)	Hagemann, Nina
Publication date	2022
Publication information	Hagemann, Nina. “Navigating Academia – Recommender Systems for Module Exploration.” University College Dublin. School of Computer Science, 2022.
Publisher	University College Dublin. School of Computer Science
Item record/more information	http://hdl.handle.net/10197/13360

Downloaded 2025-08-24 19:29:30

The UCD community has made this article openly available. Please share how this access benefits you. Your story matters! (@ucd_oa)



© Some rights reserved. For more information



Navigating Academia - Recommender Systems for Module Exploration

by

Nina Hagemann

This thesis is submitted to University College Dublin in fulfilment of
the requirements for the degree of Doctor of Philosophy

School of Computer Science

Head of School: Prof. Chris Bleakely

Principal Supervisor: Prof. Barry Smyth

Second Supervisor: Dr. Michael P. O'Mahony

Doctoral Studies Panel Members:

Assoc. Prof. Neil Hurley

Assis. Prof. Aonghus Lawlor

December 2021

CONTENTS

Acknowledgements	x
List of Publications & Awards	xi
1 Introduction	1
1.1 Higher Education	2
1.2 Module Choices: UCD Case Study	3
1.3 Recommender Systems	4
1.4 Hypotheses	8
1.4.1 Hypothesis 1: Textual module descriptors can be used to build rich content-based recommender systems, and the introduction of diversity improves the discoverability of long-tail options. . .	8
1.4.2 Hypothesis 2: Latent factors can be used to detect connections between modules that allow us to build sequential visual module exploration models.	9
1.4.3 Hypothesis 3: Content-based module similarities can be used to create models that help students' module exploration, elective module recommendations and improve student knowledge about their module space.	9
1.5 Key Contributions	10
1.6 Thesis Structure	11
2 Data-Driven Learning & Education	13
2.1 Educational Data Mining, Learning Analytics & Personalised E-Learning	14
2.2 Stakeholder: Institutions	18
2.2.1 Curriculum Analysis	19
2.2.2 Ressource Allocation & Timetabling	20
2.2.3 Drop-Out & Student Retention	21
2.3 Stakeholder: Instructors	23
2.3.1 Maintaining Learning Content	23
2.3.2 Assessment & Plagiarism Detection	24
2.3.3 Monitor Student Progress	26
2.4 Stakeholder: Students	27
2.4.1 Choosing Programmes of Study	27
2.4.2 Navigating Learning Material	29

2.4.3	Long Term Academic Planning	30
2.5	Discussion	32
3	Recommender Systems and Applications in Educational Data Mining/Learning Analytics	34
3.1	Types of Recommender Systems	35
3.1.1	Collaborative Filtering	35
3.1.2	Content-Based	39
3.1.3	Hybrid Approaches	40
3.2	Evaluating Recommender Systems	41
3.2.1	Offline Evaluation: Classic Evaluation Metrics	41
3.2.2	Beyond Accuracy	42
3.2.3	Online Evaluations	44
3.3	Recommender Systems for Academic Advising	45
3.3.1	Collaborative Filtering Approaches	45
3.3.2	Content-Based Approaches	47
3.3.3	Hybrid & Other Approaches	48
3.4	Ethical Implications of Recommender Systems for Academic Module Recommendations	49
3.5	Conclusion	50
4	Hybrid Content-Based Elective Module Recommender System	52
4.1	Motivation	54
4.1.1	Decreasing Allocations of General Elective Modules	55
4.1.2	Unsuccessful Allocation Ratios	56
4.1.3	Interim Conclusion	58
4.2	On Recommendation Diversity, Novelty, and Serendipity	58
4.3	Module Descriptor Creation	59
4.4	Module Recommendation Approaches	62
4.4.1	Hybrid Recommender	62
4.4.2	Collaborative Recommender	68
4.5	Evaluation	68
4.5.1	Dataset, Methodology & Metrics	70
4.5.2	Results	70
4.5.3	Discussion	71
4.6	Initial Prototype Design	72
4.6.1	Recommender System	72
4.6.2	Search Engine	74
4.7	Conclusion	78
5	Module Dependency Detection using Non-Negative Matrix Factorisation	81
5.1	Motivation	83
5.2	Topic Modelling for Module Dependency Detection	86
5.2.1	Non-Negative Matrix Factorisation	86
5.2.2	NMF for Module Dependencies Discovery	88
5.3	Evaluation	90
5.3.1	Dataset & Methods	91

5.3.2	Results	93
5.3.3	Discussion	94
5.4	Use Cases for Module Recommendations	96
5.4.1	Personal Curriculum Path	96
5.4.2	Exploration of the Module Space	97
5.4.3	Recommending Alternative Modules	99
5.5	Conclusion	99
6	Live User Study of Visual Module Explorer and Advisory System	101
6.1	Motivation	102
6.2	Visual Module Explorer & Advisory System	104
6.2.1	System Overview	106
6.2.2	Generating Module Representations	112
6.2.3	Visualisation & Recommendation	113
6.3	Live User Evaluation	119
6.3.1	Study Design	119
6.3.2	Research Questions	124
6.3.3	Participants	124
6.3.4	Pre-Study Survey	127
6.3.5	Analysis: Programme Structure	132
6.3.6	Analysis: Elective Modules	136
6.4	Conclusion	138
7	Conclusion	140
7.1	Summary of Contributions	140
7.1.1	Module Exploration	140
7.1.2	Elective Module Recommendations	142
7.2	Future Directions, Limitations and Open Questions	143
7.2.1	Optimising Recommendations	143
7.2.2	Textual Descriptor Enhancement	144
7.2.3	Explaining Recommendations	144

LIST OF TABLES

4.1	Module Descriptor Sections, Content and Form	61
4.2	Overlap of the Recommended Module Sets by the Different Approaches.	71
4.3	Evaluation Results for the Different Approaches.	71
5.1	Sentences and Words in Module Descriptions Before and After Pre-processing (PP).	89
5.2	Example of Document-Topic Matrix.	90
5.3	Top Words for Each of the Nine Topics.	90
5.4	Dependencies stated by the Different Experts.	91
5.5	Possible and Detected Dependencies by Year of Study.	92
5.6	Confusion Matrix.	93
5.7	Weighted Precision and Recall Scores for Dependencies by Level. . . .	94
6.1	Questions in Pre-Study Survey.	122
6.2	Questions in Post-Study Survey.	123
6.3	User Study: Participants Completion Statistics.	125
6.4	Spearman Correlation between Questions <i>Pre 3.3/Pre 3.4 (How satisfied overall were you with your chosen elective modules in the past? /How well informed do you feel about the available elective modules?)</i> and Questions <i>Post 3.1/Post 3.3/Post 3.4 (How would you rate the quality of the recommended elective modules? /How likely is it that you would consider one of the modules as your elective module? /How likely is it that you would use a system like this to find elective modules in the future?)</i>	136

LIST OF FIGURES

1.1	UCD Horizons Structure.	4
1.2	Example Threads of the UCD Subreddit.	5
1.3	Example of Content-Based Recommendations on Netflix.	6
1.4	Example of Collaborative Filtering Recommendations on Amazon: Other Items Customers Bought.	7
1.5	Example of Collaborative Filtering Recommendations on Amazon: Items Frequently Bought Together.	7
1.6	Example of Personalised Recommendations on Amazon.	7
2.1	Stakeholders and their Tasks in the Educational Environment.	15
2.2	Related Areas in Educational Data Mining/Learning Analytics.	16
2.3	Knowledge Discovery Cycle in Educational Data Mining/Learning Analytics.	17
3.1	User-Item Matrix for Collaborative Filtering Recommender System.	36
3.2	User-Item Matrix Example for Collaborative Filtering.	37
3.3	Term Frequency Matrix Example.	40
4.1	Decrease in General Elective (GE) Module Allocations.	55
4.2	The Long Tail of Elective Module Allocations : Top 50 Elective Mod- ules (2007 - 2015).	56
4.3	Top 50 Most Popular General Elective Modules (2007 - 2015).	57
4.4	UCD Module Catalogue: Module Descriptor.	63
4.5	Content-based Similarity Example on Textual Descriptor.	65
4.6	Taxonomy of Colleges, Programmes and Modules in University Col- lege Dublin.	66
4.7	Hybrid Recommender System Architecture.	69
4.8	Screenshot of the Recommender System Prototype.	73
4.9	Recommender System Prototype: Results with High Discovery.	75
4.10	UCD Module Search Engine (2017).	76
4.11	Screenshot of the Recommender System Prototype - Search Engine: Results for the Search Query <i>java</i>	77
4.12	Screenshot of the Recommender System Prototype - Search Engine: Module Information and Similar Module Recommendations.	79

5.1	Example of Explicit and Implicit Connections between Modules. . . .	82
5.2	Module Descriptor for Module COMP47590 <i>Advanced Machine Learning</i> with Learning Requirements and Learning Recommendations (2020).	85
5.3	Matrix Decomposition in NMF.	87
5.4	Example of <i>Irish Times</i> News Dataset.	87
5.5	Example of News Dataset NMF Decomposition.	88
5.6	Distribution of Core and Optional Modules per Level in 2018/2019 Term.	89
5.7	Number of Dependencies by Dependency Expert Score.	92
5.8	Weighted Precision and Recall Results for Increasing Dependency Expert Score Threshold.	94
5.9	Precision and Recall Results for Increasing Cosine Similarity Threshold.	95
5.10	Personal Curriculum Path Visualisation.	96
5.11	Mockup Visualisation of Module Recommendation with High Connectivity.	98
6.1	Visual Recommender & Advisory System: System Architecture Overview.	105
6.2	Distribution of Number of Sentences in Description and Learning Outcomes.	107
6.3	Distribution of Number of Words in Description and Learning Outcomes.	107
6.4	Module Descriptor Example Before and After Preprocessing.	108
6.5	Distribution of Number of Words in Description and Learning Outcome Combined Before and After Preprocessing (PP).	108
6.6	Graphical Representation of LDA.	110
6.7	Mockup of Visual Recommender.	111
6.8	Technical Details of Generating Module Representations.	112
6.9	Interactive Visualisation before Interaction.	114
6.10	Interactive Visualisation with Active Nodes after Interaction.	115
6.11	Module Recommender and Additional Information Panel after Interaction.	116
6.12	Interactive Visualisation with Active <i>Topic Bubble</i>	120
6.13	User Study: General Participation Distribution by Year.	125
6.14	User Study: Completion Rate by Year.	126
6.15	User Study: General Participation Distribution by Enrolled Programme.	126
6.16	User Study: Answers to Question Pre 1.3 (<i>What was your biggest motivation to choose the Bachelor programme you choose?</i>).	127
6.17	User Study: Answers to Question Pre 1.4 (<i>Which statement best describes your ideas about career goals and specialisations ?</i>).	128
6.18	User Study: Answers to Question Pre 2.1 (<i>How familiar are you with your programme structure (i.e. modules in upcoming terms, pre-/co-requisites)?</i>).	128
6.19	User Study: Answers to Question Pre 2.2 (<i>If you are unsure about the details of a module (e.g. availability, pre- and co-requisites), where would you go to find additional information?</i>).	129

6.20	User Study: Answers to Question Pre 2.3 (<i>How familiar are you with the different streams offered within Computer Science?</i>).	129
6.21	User Study: Answers to Question Pre 3.1 (<i>What is your greatest motivation when choosing elective modules?</i>).	130
6.22	User Study: Answers to Question Pre 3.2 (<i>How did you find the elective modules you ended up taking?</i>).	130
6.23	User Study: Answers to Question Pre 3.3 (<i>How satisfied overall were you with your chosen elective modules in the past?</i>).	131
6.24	User Study: Answers to Question Pre 3.4 (<i>How well informed do you feel about the available elective modules?</i>).	131
6.25	User Study: Answers to Question Post 2.4 (<i>Would you say your knowledge about the overall programme structure has improved?</i>).	132
6.26	User Study: Question Post 1.1 (<i>Please rate your overall experience using the programme visualisation.</i>).	133
6.27	User Study: Question Post 2.5 (<i>How likely is it that you would use a system like this to gain knowledge about upcoming modules and module paths?</i>).	133
6.28	User Study: Cohort Defintion.	135
6.29	User Study: RBS Values for Key Features for Cohort 1 (a), Cohort 3 (b), and Cohort 4 (c).	135
6.30	User Study: Answers to Question Post 3.1(a) (<i>How would you rate the quality of the recommended elective modules?</i>), Question Post 3.3(b) (<i>How likely is it that you would consider one of the recommended modules as your elective module?</i>), and Question Post 3.4(c) (<i>How likely is it that you would use a system this this to find elective modules in the future?</i>) . .	137

ABSTRACT

Personalised recommendations feature prominently in many aspects of our lives, from the movies we watch to the news we read and even the people we date. However, one area that is still relatively underdeveloped is the educational sector, where recommender systems have the potential to help students in a variety of ways, supporting their decision making when choosing a suitable university programme, finding the right study material, and making informed choices about their learning pathways. This work focuses on recommender systems for academic advising, helping students find the most suitable modules. Today's students enjoy various options regarding the availability of courses and modules, encouraging students to broaden their horizons, explore their interests and strengths, and develop new skills. One such opportunity offered in many universities is the possibility to freely choose elective modules from outside a student's primary area of study. Taking such elective modules is often a requirement and can significantly impact students' academic experience and overall performance.

In this thesis, we explore how recommender systems, and content-based approaches, in particular, can be used to support students in finding suitable modules, shape their academic and career paths, as well as gain knowledge and make more informed decisions. Our approach is based on the textual descriptors that are freely available on universities module catalogues to match students with modules based on their learned interests and preferences. In contrast to the majority of related work in the field, our approaches work independently of students' demographic, personal, and performance data. We show how the module descriptors can be used to extract module similarities and latent topics that allow for rich visualisation options and personalised module recommender systems.

We evaluate our approach using offline and online studies. In a live user study, we show that our approach can improve student knowledge about their subject and elective module options. Furthermore, the results show that the participating students largely enjoy interacting with the system and show a high likeliness of reusing the system again in the future.

Statement of Authorship

I hereby certify that the submitted work is my own work, was completed while registered as a candidate for the degree stated on the Title Page, and I have not obtained a degree elsewhere on the basis of the research presented in this submitted work.

Signature _____

ACKNOWLEDGEMENTS

This work would not have been possible without the help of many. First and foremost, I would like to thank my supervisors, Michael P. O'Mahony and Barry Smyth, for their guidance and support throughout my PhD. Their encouragement and knowledge have allowed me to grow and learn beyond what I thought was possible. I am deeply thankful for their time and commitment that has made this work possible.

I am grateful to the UCD Insight Centre for Data Analytics for hosting and providing me with the resources needed to conduct this work. Furthermore, I am thankful for all my fellow PhD students, researchers and Insight staff that have inspired me in many ways and have become friends along the way. I especially want to thank Khalil Muhammad for inspiring me to start this journey and who has been a great mentor throughout my PhD.

I am deeply thankful for my friends in Dublin, Berlin, and worldwide, who have supported me in so many ways. From late-night phone calls to care packages to spontaneous trips, you have all been a great source of fun, motivation, and love. Thank you for constantly reminding me of what is most important. I also want to thank my family, who always supported me in all my endeavours throughout this journey. Thank you, in particular, to my father for being my academic idol and always believing in me.

If anyone knows the challenges of a PhD secondhand, it is my fiancé Brian, and I can not be more thankful for having you by my side through all of this. Thank you for your endless optimism, love and support. This work would probably have been possible without you, but it would have only been half so much fun.

My PhD studies were funded by Science Foundation Ireland (SFI) under grant number 12/RC/2289.

LIST OF PUBLICATIONS & AWARDS

- (In preparation) N. Hagemann, M. P. O'Mahony, and B. Smyth. **Visual Module Exploration: A Live-User Evaluation**. KI-Künstliche Intelligenz 2022.
- (Published & Awarded *Best Student Paper in the application stream*) N. Hagemann, M. P. O'Mahony, and B. Smyth. **A Live-User Evaluation of a Visual Module Recommender and Advisory System for Undergraduate Students**. International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer, Cham, 2021, pp. 299-312
- (Published) N. Hagemann, M. P. O'Mahony, and B. Smyth. **Visualising Module Dependencies in Academic Recommendations**. Proceedings of the 24th International Conference on Intelligent User Interfaces: Companion, 2019, pages 77–78
- (Published) N. Hagemann, M. P. O'Mahony, and B. Smyth. **Module Advisor: a Hybrid Recommender System for Elective Module Exploration**. Proceedings of the 12th ACM Conference on Recommender Systems, 2019, pages 498–499
- (Published) N. Hagemann, M. P. O'Mahony, and B. Smyth. **Module Advisor: Guiding Students with Recommendations**. International Conference on Intelligent Tutoring Systems, 2018, pages 319-325

INTRODUCTION

Perhaps at no other time has education been more important than right now. The ever-changing economic world is demanding highly educated and skilled employees. However, with more specialised education comes an increase in choices for learners. Universities and other higher education institutions offer an extensive range of courses, allowing learners to broaden their horizons and specialise in increasingly niche and new fields. Additionally, a dizzying array of opportunities are available to learners around the world in online e-learning courses. There has never been a more accessible and extensive choice in learning opportunities than at the current time. Nevertheless, this freedom comes with many challenges. How can learners find suitable modules at the right time at the right institution?

Not only has the number of offered programmes at universities increased drastically, once enrolled in a programme, today's students are also faced with a wide variety of options, often making it difficult to choose the right modules. Universities have become multi-faceted institutions where students are encouraged to broaden their horizons, explore their interests and strengths, and develop new skills. Most universities offer students options to personalise their curriculum in one way or another by choosing additional minors, elective modules, or specialised streams.

In the following sections, we present the importance of higher education and the impact of academic advising for students' academic success. Next, we present a motivating case study from University College Dublin (UCD), showcasing the opportunities UCD offers and the issues that arise. Finally, we introduce recommender systems and how they can aid with these challenges.

1.1 Higher Education

Higher education has never been more critical than today and has increased massively in the last 50 years. The gross enrolment ratio in tertiary education in the EU has increased significantly, from only 17% in 1970 to over 65% in 2014.¹ In the changing economy, a completed higher degree is vital. The Organisation for Economic Co-operation and Development (OECD) show that employment rates rise to an average of 84% with any third level education compared to 57% with no upper second level [129]. In Ireland, the employment rate for people with an upper second level education is 69%, which rises to 83% for Bachelor's and 86% for Master's degrees in 2016 (25 to 64-year-olds).

In their study in 2015, the European Commission states one of their goals for their *Europe 2020* strategy is to have at least 40% of the 30 to 34-year olds complete higher education, with three-quarters of the European countries rating the importance of this goal as high or very high [194]. While there is no European-wide definition, three main points are used to define study success: (i) *completion* – the successful completion of the students' studies, (ii) *time-to-degree* – the completion of studies in a reasonable time period, and (iii) *retention* – the re-enrolment of students and the reduction in drop out rates.

Traditionally, Computer Science has one of the highest attrition rates, with drop out rates as high as 40% [20]. This, in part, is explained by poor advice given before and during college. Academic advising has long been standard practice in most higher level education institutions, and its impact has been researched and proven time and time again [49]. Advisors help students navigate the university's educational space, fulfil their programme requirements and find suitable modules, amongst other things [19]. Traditionally done fully offline, in-person academic advising is a significant task for universities and providing enough advisors for the growing number of students can be a challenge. Often, advisors can be overloaded and unable to provide the detailed personal support students need to succeed in the academic world.

By using recommender system approaches to provide support for academic advising, we can aid three different stakeholders: (i) *students* can benefit from additional information that is presented in a personalised matter, allowing them to gain an understanding of available options and to tailor their learning experiences accordingly; (ii) by expanding and diversifying the modules chosen by students we can support the *university* in allocating the right amount of resources and decreasing organisational issues such as module overcrowding and timetable clashes; (iii) by facilitating students to find their most suited elective modules we argue that *instructors* can benefit from

¹<https://datacatalog.worldbank.org/dataset/education-statistics>

an increased interest in the subject, rather than students who chose the module out of ulterior motives. With the explosion of Artificial Intelligence (AI) and the increasing amount of educational data available, the opportunity has never been better to create recommender systems for educational purposes.

1.2 Module Choices: UCD Case Study

University College Dublin (UCD) is Ireland's largest university, with over thirty thousand students in 2020. UCD encompasses 7 Colleges that host 38 Schools. From humanities to Veterinary to Engineering and Computer Science, UCD offers a wide variety of programme options. 60% of the student body are enrolled across 120 undergraduate programmes. In 2005 the UCD *Horizons* programme was introduced.² This programme restructured all undergraduate programmes to a fully modular and credit-based curriculum, which provides opportunities for undergraduate students to broaden their horizons by taking two elective modules each year, see Figure 1.1. These elective modules can be chosen from any programme and any school across the university, offering students a chance to explore their interests beyond their enrolled programme. Their main objective is to provide an opportunity to the students to explore new subjects or deepen their knowledge in a specific field. Additionally, some programmes offer a wide variety of optional modules and specialisation streams.

Unfortunately, in practice, student choices are often limited by discoverability challenges and overcrowded modules. As a result, many students follow the crowd or their peers' recommendations when selecting elective modules. Moreover, students seem to care less about the actual content of the module and its connection to their interests and more about how easy it is to get a good grade.³ We can see this develop in social media platforms such as the UCD page on Reddit⁴, where students every year ask for recommendations for *easy* modules, some anonymised examples are shown in Figure 1.2. In an internal report in UCD, it was also noted that "*dropouts had felt pressure and uncertainty about an overwhelming number of module choices*"⁵. While UCD *Horizons* offers a great opportunity for students, which many are enjoying greatly⁶, it seems as if finding the right modules poses a significant challenge for students.

²<https://www.myucd.ie/applying-to-ucd/ucd-horizons/>

³<https://www.universityobserver.ie/head-to-head-expanding-your-horizons/>

⁴<https://www.reddit.com/r/UCD>

⁵<https://www.irishtimes.com/news/education/the-sweet-and-sour-of-the-pick-n-mix-degree-1.2619237>

⁶<http://www.myucdblog.com/exploring-horizons-ucd/>

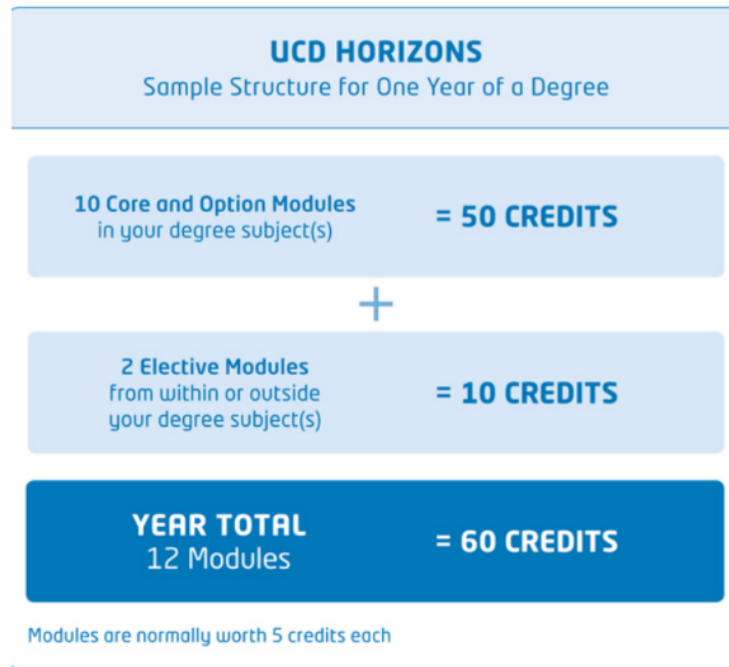


Figure 1.1: UCD Horizons Structure.

Source: <http://www.myucdblog.com/main-course-side-ucd-electives/>

Recommender systems have shown to be able to significantly support decision making for multiple different applications and use cases. Choosing the most suitable elective modules lends itself to an interesting recommender system challenge.

1.3 Recommender Systems

Recommender systems are omnipresent in our everyday life. With the ever-growing amount of data available online, they have become indispensable in many applications. From the movies we watch to the products we buy to the people we date, recommender systems help us navigate the vast space of available information. Recommender systems extend the classic information filtering systems and have since then developed into a rich research area itself [3]. However, compared to classic information filtering, recommender systems emphasise personalisation. Recommender systems are usually classified into two categories, content-based and collaborative filtering approaches. While content-based recommender systems recommend items similar to the users' past tastes, collaborative filtering recommends items based on similar users' preferences. Famously Netflix⁷ has one of the most advanced recommender system en-

⁷<https://www.netflix.com>



Figure 1.2: Example Threads of the UCD Subreddit.

gines today. After introducing the *Netflix Prize* in 2006, a machine learning competition, which would award the best recommender system approach that would enhance their current RMSE by 10% with 1 million dollars, the online movie streaming platform has evolved to encompass a wide variety of techniques [67]. One of the approaches is based on content-based filtering, the *"Because you watched ..."* video-video similarity ranker, see Figure 1.3 provides similar movies/tv-shows based on a previously watched item.

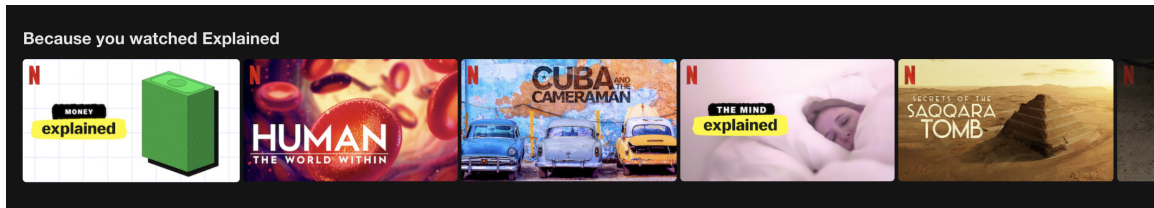


Figure 1.3: Example of Content-Based Recommendations on Netflix.

The largest e-commerce company in the world Amazon⁸ famously started with a user-based collaborative filtering approach in the 1990s [174], finding similar users and recommending items based on the notion of "people like you also bought ...".[96] Since then Amazon's recommender system has evolved greatly to incorporate multiple different approaches while still staying close to their collaborative filtering approach [174]. Nowadays, Amazon offers multiple different recommendations during the users' shopping experience, such as *"What other items do customers buy after viewing this item?"*, see Figure 1.4, or items that are frequently bought together, see Figure 1.5. These are examples of non-personalised recommendations on Amazon, which means different users would receive the same list of items. Amazon further provides personalised recommendations, such as *"Items that you might like"* based on the specific user's interaction with the system; for example, in Figure 1.6 similar books are recommended based on the user's past purchase or viewing history. While many recommender system approaches can be based on collaborative filtering or content filtering, current research has developed much further than these two classic approaches, combining them into hybrid systems and extending them with powerful new approaches [31].

Recommender system research for educational purposes has seen an increase in interest over the last ten years [48]. Starting with building recommender systems for online learning platforms and e-learning applications, research has recently started to explore the benefits of recommender systems and machine learning techniques in the academic environment [54, 81]. These techniques can support academia by predicting students performances and help identify students who might be at risk [1], to be able to offer support and decrease student dropout rates. Other work has focused on recommending majors [138] or support students in finding scholarships [53]. With the

⁸<https://www.amazon.com>

What other items do customers buy after viewing this item?

Page 1 of 4



Figure 1.4: Example of Collaborative Filtering Recommendations on Amazon: Other Items Customers Bought.

Frequently bought together



Figure 1.5: Example of Collaborative Filtering Recommendations on Amazon: Items Frequently Bought Together.

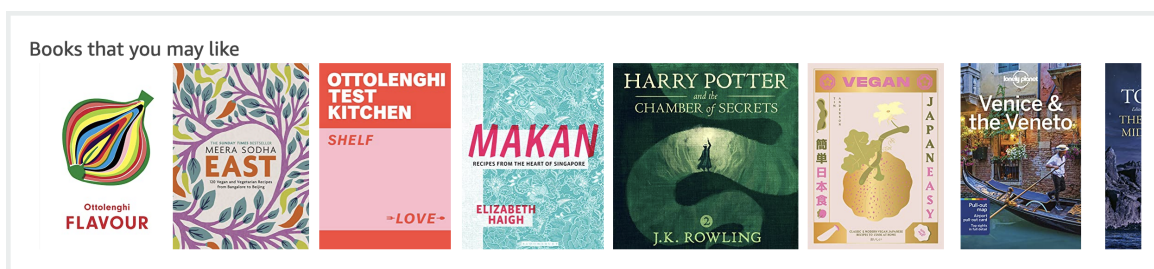


Figure 1.6: Example of Personalised Recommendations on Amazon.

growing amount of educational data available online new research areas such as *Educational Data Mining*, *Learning Analytics* and *Precision Education* have emerged, giving way to a broad spectrum of research opportunities.

In this work, we aim to contribute to this growing research field by exploring content-based approaches to help students navigate their academic choices, gain knowledge about their programme structure which ultimately allows them to make informed decisions in their academic journey.

1.4 Hypotheses

We formulate three main hypotheses that will guide us through our work with the ultimate aim of creating a recommender system for academic advising. We mainly focus on content-based approaches to amplify the rich information it holds, which allows us to build recommender systems that enable students to learn about their options and make well-informed decisions, rather than following their peers' recommendations or choosing *easy* modules.

1.4.1 Hypothesis 1: Textual module descriptors can be used to build rich content-based recommender systems, and the introduction of diversity improves the discoverability of long-tail options.

While collaborative filtering approaches are used prevalently in recommender systems for educational purposes, we argue that we can improve recommendation quality by focusing on content-based approaches. Using module descriptors, we are able to harvest information about modules that allow us to detect similarities between modules, independently of their popularity and performance prediction. While the possible grade is an important factor for students, we argue that modules that are uniquely suited to the student can increase the student's motivation which in turn can lead to a positive experience and good performances. Finally, we explore the effect of diversity in recommender systems for academic advising. We investigate the impact of diversity on recommending a diverse set of modules that allow us to find modules from a broader array of topics that uniquely fit students' needs and interests and help solve the long-tail problem in module recommendations.

1.4.2 Hypothesis 2: Latent factors can be used to detect connections between modules that allow us to build sequential visual module exploration models.

Higher education programmes are often of sequential nature, in the sense that modules are based on skills obtained in previous modules. Frequently, programmes have explicit pre- and co-requirements for modules and dictate an order in which modules must be taken. By focusing on the textual descriptors of modules, we can conduct latent variable analysis and use a topic modelling approach to identify unobserved patterns and abstract topics hidden in the module descriptors. These can be used to model overlap or dependencies between the modules in the module space, which provides additional information for advanced recommendation techniques and sophisticated visualisation of relationships between modules that enables students to understand and navigate complex programme structures. Furthermore, by harvesting module dependencies based on latent features, we can improve recommendations in the context of their connectivity to other modules and their location in the module space. This, in turn, provides rich information to build visualisations that can aid students in gaining knowledge about their module space and possible academic and career paths.

1.4.3 Hypothesis 3: Content-based module similarities can be used to create models that help students' module exploration, elective module recommendations and improve student knowledge about their module space.

Ultimately, content-based approaches are able to build rich online advising tools and recommender systems for elective modules. By helping students learn about their module space, the module connections and possible career paths, students are enabled to make more informed decisions when choosing their own academic curriculum. We argue that students, as internet natives, are very receptive to online tools and would value additional academic advising in the form of an online tool. Using online academic advising tools will ultimately increase students' knowledge and satisfaction when choosing their academic path.

We will outline our main contributions in regards to the established research questions in the following section.

1.5 Key Contributions

In this thesis, we focus on supporting students in finding suitable modules for their academic pathway. We aim to build recommender systems based on module content and the dependencies between modules rather than performance data. Furthermore, we aim to increase the discoverability of long-tail options in elective modules by introducing diversity and creating awareness in students. For that, we begin our research into hybrid content-based recommender systems that include a notion of diversity to allow students to discover modules from a more comprehensive array of options. Furthermore, due to the sequential nature of modules and the lack of explicit requirements, we focus on representing dependencies between modules in subsequent years to define module paths better. Lastly, we combine our approaches and findings to build an interactive system to conduct a live user study to test the proposed approaches.

We summarise our contributions as follows:

1. Initial Hybrid Recommender System (presented in [72, 71]) (Chapter 4)

- We propose a hybrid recommender system that uses textual module descriptors and university taxonomy, addressing **Hypothesis 1**. We combine techniques from data mining and natural language processing with the structural nature of universities to recommend modules that cover a wide variety of subjects while still being close to students' interests.
- Using state-of-the-art web development techniques, we implement a prototype of the *UCD Module Advisor*, that includes a personalised recommender system, as well as additional functionalities, such as an intelligent search engine.
- We evaluate our approach using an offline evaluation and a custom *simTo-Core* metric. We show that introducing diversity through taxonomy can increase the discoverability of modules while still recommending meaningful electives.

2. Module Dependency Detection (presented in [73]) (Chapter 5)

- We propose a matrix factorisation approach to detect dependencies between modules in subsequent years. Using non-negative matrix factorisation and module textual descriptors, we implement a dependency model that detects inter-year relationships between modules.
- We present multiple use cases and visualisations that would benefit from the presented system, such as a personal path representation.

- We create an expert-based ground truth using University College Dublin BSc Computer Science alumni. The established ground truth depicts perceived dependencies between modules within the BSc Computer Science programme structure.
 - We validate the aforementioned approaches using the created ground truth and show that we can sufficiently detect important dependencies between modules, as well as additional connections (**Hypothesis 2**).
3. **Interactive Module Advisory System and Recommender System (accepted to AI-2021 Forty-first SGAI International Conference on Artificial Intelligence 2021) (Chapter 6)**
- Combining methods and findings from the above systems, we present an interactive visual recommender system and advisory system that uses content-based recommender system techniques and matrix factorisation approaches to build a rich visualisation for module exploration as well as elective module recommendations, in regards to **Hypothesis 3**.
 - We implement a web-based framework for the presented approach and design an online user study.
 - We conduct an online user study with UCD BSc Computer Science students and collect qualitative and quantitative data regarding their prior knowledge of module dependencies and elective module options. The user study collects data about the participants usage of the system and survey how the proposed system facilitates a better understanding of programme structures and a more-informed approach to elective module selection.
 - We evaluate the collected qualitative and quantitative data and show that students are receptive to automated online academic advising. Our results show that the system can increase knowledge in the majority of participating students regardless of their year of study or previous knowledge.

1.6 Thesis Structure

In the following chapters, we motivate, present, evaluate and discuss each of the above contributions. The remainder of the thesis is divided into two main parts. Firstly, the first part covers the related work in the fields of Educational Data Mining and Learning Analytics (Chapter 2) and Recommender Systems (Chapter 3) as they relate to our research. Secondly, we then present three chapters to address each of the main

hypotheses. Chapter 4 introduces our research of hybrid content-based recommender system approaches. We discuss our motivation, use of diversity and offline evaluation. A web-based prototype is presented before we conclude the chapter. Following that, in Chapter 5, we present a matrix factorisation approach to detect module dependencies. We present our technical details and give an overview of use cases. Before we evaluate the approach, we present the implementation and statistical details of an expert-based ground truth. We combine our findings from Chapters 4 and 5 to implement a final web-based academic advising tool and recommender system presented in Chapter 6. We present our technical approach in detail before discussing the visualisation and user interface framework. Next, we present the setup and results of an online live user study and evaluate our findings. Finally, we conclude this work with a discussion of the work carried out and showcase an in-depth summary of our core contributions, as well as a discussion of possible future work in Chapter 7.

DATA-DRIVEN LEARNING & EDUCATION

Academia in the 21st century, from a student's perspective, is characterised by an increasing degree of choice as many institutions aim to provide their students with options to personalise their educational experience. Traditional classroom-based face-to-face education has increasingly introduced online opportunities and has merged into *blended learning* experiences. Additionally, e-learning courses provide a staggering amount of education online. Blended learning and e-learning platforms provide flexibility and freedom to traditional face-to-face education. Systems such as *Learning Management Systems* (LMS) (e.g. Moodle¹, Blackboard²) and *Massive Open Online Courses* (MOOC) (e.g. edX³ and Coursera⁴) open up a whole new world of opportunities for instructors and learners alike. These systems collect a new type of educational data. Their demographical and performance data no longer define students, but more fine-grained information, such as interaction patterns, time spent with specific subjects, and search patterns, to name a few, can be harvested, analysed, and provide useful insights.

This rapid development has changed the way education is delivered, even before the disruptions of the 2020/2021 coronavirus pandemic. The expectations of the various stakeholders and the data generated by a more integrated educational environment have highlighted exciting new opportunities as well as critical new challenges for institutions, instructors, and students alike.

In this chapter, we aim to summarise these opportunities and challenges that serve to provide a context for our work in the remainder of this thesis. In the next section, we summarise some of the ways that data is being used in universities worldwide, focus-

¹<https://moodle.org/>

²<https://www.blackboard.com/>

³<https://www.edx.org/>

⁴<https://www.coursera.org/>

ing on educational data mining and learning analytics. Following this, we highlight several specific and essential tasks within this ecosystem that relate, particularly to institutions, instructors, and students, summarising a selection of case studies in this regard.

2.1 Educational Data Mining, Learning Analytics & Personalised E-Learning

With the increasing learning opportunities, available resources and associated educational data, new objectives and challenges arise. Three main stakeholders are actively involved in creating, analysing and benefitting from educational data. Each of those stakeholders comes with their own specific set of tasks and objectives; see Figure 2.1. We identify the three stakeholders as follows: (i) *students* benefit from the additional opportunities as it gives them increasing freedom to learn what, how and when they want—allowing them to shape their academic path specifically to their strengths and needs and improve their experience and performance. However, the choices can be overwhelming, and students require additional support in finding the right opportunities and make informed decisions. (ii) *Instructors* are given vast options to improve their teaching through blended learning. Instructors can increase students’ motivations, learning experiences, and satisfaction by offering new technology-based learning opportunities. Instructors, however, must learn to understand the best way of integrating these techniques into their classic teaching methods. Further, the created data can significantly benefit instructors in understanding the students’ learning behaviours better if they have the means to harvest valuable information from the data. Lastly, (iii) *institutions* are responsible for creating a positive learning environment for students and instructors alike. Allocating the right resources for students and staff can create challenges, especially when new technology-driven options are introduced. By harvesting educational data, institutions have great potential to improve the efficiency of these tasks.

One of the biggest challenges for educational institutions concerns the collection, analysis, and utilisation of educational data to produce new insights that can improve academic life for students, instructors, and institutions [16]. While traditional student data, such as enrolment, demographic, and performance data, has been historically studied, with the new development in blended learning and its utilisation of online frameworks, additional data, such as interaction with content, student interaction, and forum threads to name a few, can now be collected. This new type of behavioural and

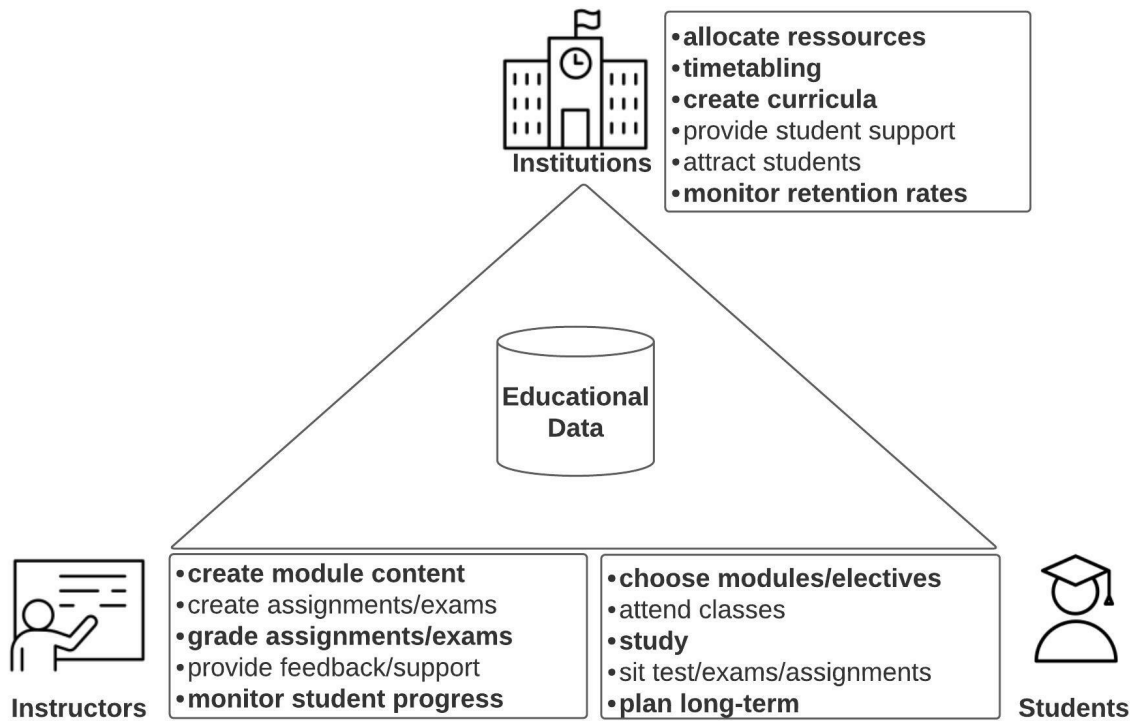


Figure 2.1: Stakeholders and their Tasks in the Educational Environment.

semantic data allows for increasingly advanced analysis techniques. Over the last 20 years, a large community of research has developed that aims to improve the educational sector using AI and machine learning techniques. Two different communities have been at the forefront of this research, *Educational Data Mining* (EDM) and *Learning Analytics* (LA).

- **Educational Data Mining** focuses on the discovery and exploration of knowledge in the unique and increasingly large-scale data harvested from educational applications [16].
- **Learning Analytics** is the measurement, collection, analysis, and reporting of data about learners and their contexts for the purposes of understanding and optimising learning and the environments in which it occurs [172].

While both communities have their own focus: LA focuses on the educational challenge, and EDM is focused on the technological challenges [157], both share common interests, techniques and objectives. Both communities are interdisciplinary and include a wide variety of areas such as information retrieval, data visualisation, network analysis, cognitive psychology, to name a few. Three main areas can be determined

that make up the foundation of both communities: (i) Computer Science, (ii) Statistics, and (iii) Education, see Figure 2.2 [157].

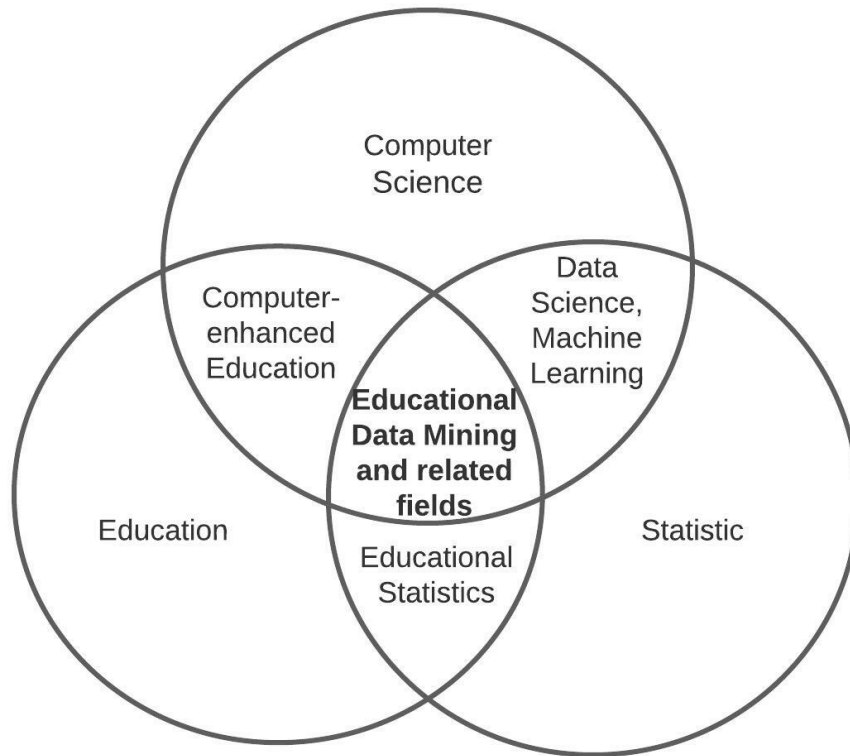


Figure 2.2: Related Areas in Educational Data Mining/Learning Analytics.

A variety of other terms and adjacent research communities have also emerged. For example, *Technology Enhanced Learning* (TEL) [48] covers several application areas and technologies, from learning analytics to improve teaching quality to recommending personalised learning paths to students and is sometimes defined as the super-category of EDM and LA. *Precision Education* (PE) is concerned with the task to provide a personalised learning experience for each individual [202]. Many techniques from data mining and artificial intelligence are used to achieve this goal [202]. *Big Data in Education* [197] and *Educational Data Science* [30] are closely related to EDM and focus on using data science methods on (big) educational data to provide new insights. On the other hand, terms like *Academic Analytics* and *Teaching Analytics* are concerned with the collection and analysis of educational data with a focus on educational challenges from the institutions and instructors points of view [34].

Since the publication of the first EDM/LA focused book in 2006, *Data Mining in E-Learning* [155], this research area has grown immensely. With two annual dedicated

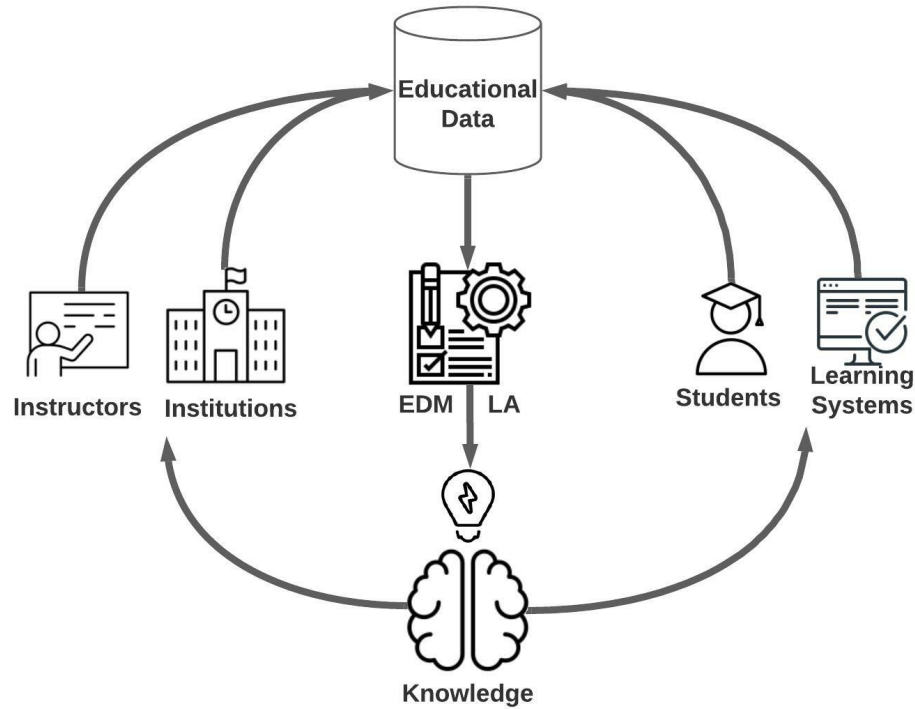


Figure 2.3: Knowledge Discovery Cycle in Educational Data Mining/Learning Analytics.

(EDM⁵ and LAK⁶) and many more related conferences, the community has provided large amounts of high-quality research. The number of papers published has increased from less than 1,000 in 2012 to 6,000 papers (LA) and over 3,000 papers (EDM) in 2018 [157]. In addition, more than 15 books have since been published covering Educational data mining and related topics, as well as the launch of two journals, *Journal of Educational Data Mining*⁷ and *Journal of Learning Analytics*⁸, dedicated to the topic, show the growing interest and importance of EDM/LA.

Overall, this emerging discipline has the ultimate objective to support academic stakeholders by discovering knowledge in educational data. This process draws from the *Knowledge Discovery in Databases* (KDD) process prominently used in data mining [58]. The adapted knowledge discovery process is depicted in Figure 2.3. There are four contributing sources of data, namely the learning systems, the institutions, instructors as well as the students themselves. This data can be gathered and used to generate new knowledge using techniques from EDM and LA. The gained knowledge is then fed back to the stakeholders, namely students, instructors and institutions. Further,

⁵<https://educationaldatamining.org/>

⁶<https://www.solaresearch.org/events/lak/>

⁷<https://jedm.educationaldatamining.org/index.php/JEDM>

⁸<https://learning-analytics.info/index.php/JLA>

the new knowledge is used to improve learning systems. EDM and LA can impact this cycle in many different ways and at different stages.

Due to the interdisciplinary nature of the field and the wide variety of application areas, summarising the work done in EDM/LA can be challenging. A wide range of technical approaches, often borrowed from well-known areas, such as data mining, recommender system and information retrieval, are used to solve a vast taxonomy of interlinked but individual objectives. Text mining approaches are used to analyse content from educational information such as forums, web pages, and documents [50]; further, clustering and classification approaches are used to mine similarities between learning materials, student behaviour, and learning pattern, to name a few [104]. Prediction approaches are used to model student behaviour [57], predict future performance [52], and allow for outlier detection [6], finding students who might show signs of weak performance or are on the verge of dropping out [201]. Deep learning approaches [200], and other advanced machine learning models are deployed to support the educational stakeholders recently, alongside a focus on explaining recommendations and visualising results [102].

To provide a structured overview of the relevant work in the field, we will present a selection of related work from the perspective of the different stakeholders. Then, we select a subset of the tasks/objectives detected in Figure 2.1 for each of the three main stakeholders and present relevant work before concluding this chapter with a discussion.

2.2 Stakeholder: Institutions

The institution comprises a variety of individual actors, such as universities, learning providers, and administrators. Typically an institutional stakeholder is responsible for the management, delivery, and support of educational resources. Therefore, the objectives and tasks are multifaceted. On the administration side, the main objectives can be described as improving efficiency in organising institutional resources (material and human) and utilising available resources to their full potential. Further, higher-level education institutions aim to enhance their programmes offer, validate the effectiveness of the blended learning techniques, and optimise technology usage. Finding cost-effective ways of improving retention and grades, as well as select the most qualified applicants is important [157].

To achieve these objectives, institutions face many organisational and administrative tasks. Some of which EDM/LA can provide valuable insights for and improve the

efficiency of higher level institutions. Following, we present a selection of related work that explicitly or implicitly aims to support the institutions in their educational tasks and objectives.

2.2.1 Curriculum Analysis

Curriculum Analysis [116], or Curriculum Mining [141], is concerned with the collection, analysis and visualisation of (administrative) curricular data – such as enrolment information, curriculum coherency, requisites – to inform and support students, instructors and institutions. Two of the main objectives in Curriculum Mining are: (i) the construction of an academic curriculum model and (ii) the analysis of curriculum conformance. Creating curricula is traditionally a manual task that falls on instructors and institutions. It can be highly time-consuming and requires ongoing supervision and optimisation. As a result, automated or intelligent curriculum analysis and evaluation approaches have gained interest in the EDM field. While often aimed to benefit instructors and institutions, curriculum analysis can also provide meaningful insights for students and help them explore their academic space.

To provide the best possible academic experience to students and instructors, institutions spend a lot of time and effort creating curricula and adapting and optimising content resources. Curriculum analysis and curriculum mining can support institutions to create well-rounded programmes that provide a competitive and interesting learning environment for students and instructors. Different techniques and objectives can be considered. For example, in [116], Mendez et al. introduce factor analysis for curriculum coherence detection. Undergraduate curricula have to follow many guidelines and provide students with coverage of the crucial foundations of their chosen programme. The coherence analysis in [116] tested if the current BSc Computer Science curriculum aligns with *ACM Body of Knowledge*⁹ and found that some modules show low coherency. The work proposes an in-depth analysis of the programme structure to identify the underlying issues and address the importance of the low-coherence modules within the CS programme.

Curriculum analysis can benefit all three main stakeholders, for example supporting students to develop self-regulation and improve their learning behaviour which in turn can lead to lower dropout and higher retention rates for institutions [145]. Early approaches relied heavily on manually collected opinions from experts, making it a time-consuming task [166]. Recently text mining and data-driven approaches have

⁹<https://dl.acm.org/cms/attachment/632213cd-cec3-48ca-9100-27b2e9f4c365/cs-csbok.html>

been proposed. For example, Kawintiranon et al. present an automatic text-based curriculum analysis using text mining, keyword extraction, and TF-IDF [87]. The analysis is based on keywords from course materials matching keywords from online documents. A new measurement is proposed to quantify associations between course materials and online documents using matching keywords. A Comparison-Matrix is constructed using keywords from the materials and keywords from the dict-matrix. A preliminary study comparing the computed results against student-based ground truth yielded favourable results.

Ajanovski presents a curriculum visualisation study that includes multiple alternative interfaces that provides insights to different stakeholders in the educational environment [5]. On the university management side, a visualisation that depicts prerequisites and dependencies between modules. This visualisation can help in investigating inter-dependencies and constraints between programmes. The insights gained through the visualisation can be beneficial when creating curricula by avoiding unattainable or too restrictive dependencies.

The related work presented in this section shows a strong interest in curriculum analysis/curriculum mining. While some approaches target to support institutions, most presented results can benefit multiple stakeholders. We detect a lack of evaluation on the impact of curriculum analysis on the institutions' objectives and aims, such as efficiently allocating resources and creating timetables. We present these tasks and related work in the following section.

2.2.2 Ressource Allocation & Timetabling

The educational process at higher level education institutions is a complex process that involves many aspects. The growing number of students, programmes, and individual freedom in students' academic careers makes academic planning even more complex [186]. To provide the optimal learning experience for students and a supportive environment for instructors and staff, planning the educational process is one of the significant tasks institutions have to face. Different stages are needed to implement the educational process, such as the creation of an academic calendar, select elective modules, create curricula and syllabus, and workload assignment [82]. To support the institutions in this mammoth task, research in EDM/LA has presented different approaches to aid in implementing this process.

Timetabling is a classic resource allocation challenge faced by every higher-level institution. Educational timetabling has long been a well-researched area, with first definitions dating back to the 1950s [199]. Timetabling can be defined as "*Timetabling is the*

allocation, subject to constraints, of given resources to objects being placed in space-time, in such a way as to satisfy as nearly as possible a set of desirable objectives." [199]. Educational timetabling, which can be further separated into course and exam-timetabling, is an organisational problem with many soft and hard constraints. *Hard constraints* are constraints that need to be strongly enforced, such as the constraint that neither student nor staff can be at two locations at the same time or all scheduled events require a suitable location and other resources (such as staff and equipment), whereas *soft constraints* can be time between events, event coherence, and staff and student preferences [31].

Research in EDM/LA can support the process of timetabling in active and passive ways. Using EDM approaches, such as linear programming approach presented in [93], subproblems of timetabling can be optimised. In [93] Kristiansen et al. show how partial Dantzig-Wolfe decomposition can help solve the *Elective Course Planning Problem* at a danish high school. Other research has focused on soft constraint optimisation [124], and student sectioning [123].

On the other hand, EDM/LA can passively support the academic institution by providing valuable insights into the effectiveness and effect of their resource allocation process. For example, in [13] Armatas and Spratt show that by using learning analytic techniques, curriculum reviews can be improved drastically. Actionable insight can be presented in a precise and insightful way to institutions, allowing them to gain knowledge that will allow them to improve their educational process. Other tools, such as automated academic advising [9], can further provide help to the institutions by lightening the amount of advising staff needed.

While resource allocation and timetabling present critical tasks of higher education institutions within the EDM/LA community, the focus is often on other objectives. The timetabling problem is a complex issue and has been often shown to be NP-complete or NP-hard, depending on the included constraints [144]. However, we have seen promising approaches in the EDM/LA community to support institutions. In the following section, we present research conducted to aid in predicting student dropouts to help improve student retention rates.

2.2.3 Drop-Out & Student Retention

Student retention refers to the graduation rates and aims to decrease the number of students that either drop out or transfer to another school. Student retention is closely related to the revenue and reputation of the institution. A low retention rate can cause financial losses and a decrease in university reputation for the institution [107]. Therefore student retention is one of the most critical challenges for institutional stake-

holders. In 2019 the Higher Education Authority (HEA) published a report showing that students in Ireland enrolled in computing and engineering programmes show the highest dropout rates up to 45% [143]. Students' reasonings for not progressing in their enrolled programmes can be multifaceted and is not further reported. In order to reduce the number of students leaving their enrolled programmes, EDM/LA techniques can be used to create early warning systems to predict students who might be at risk and allow institutions to implement support systems to help those students.

Predicting students' dropouts is a task that has received much attention in the EDM/LA community [11]. The task lends itself to several classification techniques, such as support vector machines [206], neural networks [99, 44], and decision trees [47] to mention a few. Pradeep et al. present a study of multiple classification and decision tree algorithms to predict students pass/fail performance in a module [148]. The study concludes that most classification approaches are well suited to predict if students pass or fail a module given their previous performances. Further, it was shown that feature selection can enhance performance and detect student features that are specifically important for performance prediction. More progressive and novel approaches have been presented in recent years; for example, in [26] Iam-On and Boongoen present a novel dimension reduction approach, link-based cluster ensemble, that allows a more accurate drop out prediction than baseline models. Iam-On and Boongoen compare different classification models to predict early dropouts. It was shown that the novel approach based on link-based cluster ensembles can outperform several other dimensionality reduction techniques.

As the majority of dropouts occur in the early stages of students' academic path [143], researchers have focused their efforts on detecting students at risk as early as possible. The earlier students at risk of dropping out are identified, the higher the chance for timely and effective intervention. For example, Dekker et al. show that using pre-university data in combination with students first-semester performance information can provide adequate predictions [43]. In [107] Marquez-Vera et al. focus on predicting high school dropout based on data collected at different stages of the academic year. The results show that the model could accurately predict dropouts using data from the first six weeks of the course.

Drop out prediction and student retention analysis has been shown to be an essential part of the EDM/LA research. A wide variety of approaches and data sources can be considered. By analysing student data before, during and after each term, prediction tools allow higher level institutions to gain insight and provide support to students who might be on the verge of dropping out [206].

2.3 Stakeholder: Instructors

The second main stakeholder in the educational environment is the instructor. Instructors refer to people who are teaching, such as educators, lecturers, teachers, and tutors. They are responsible for providing the best possible teaching experience to the students. Instructors face different organisational and teaching tasks, such as creating appropriate module content, choosing the proper blended learning techniques, and creating challenging but appropriate assignments. Further, instructors are tasked to monitor students' learning behaviour, performance progress and intervene and provide feedback and support to students in need. Lastly, instructors are tasked with grading and providing feedback on in-class assignments, homework, tests and final exams.

These tasks are highly time-consuming and show great potential for data-driven solutions. We will present some approaches presented in the EDM/LA community to support instructors.

2.3.1 Maintaining Learning Content

With the increasing use of blended learning, the tools available to instructors is growing steadily. Instructors face the time-consuming task of preparing course material and choosing suitable media. Learning management systems, such as Moodle and Blackboard, provide various tools to the instructors. Research in the EDM/LA community has approached this task from different perspectives, such as creating courseware [156], personalising the material [28, 40], and analysing students behaviour and interactions [106].

Garcia et al. present a hybrid recommender system that uses interaction data from LMS to create rules and recommendations to support instructors to maintain their learning materials online [62]. The approach discovers rules that allow instructors to evaluate and improve various features; for example, the system alerts the instructors about exercises that took students a long time, but the overall score was low, indicating that the wording in the exercise might be ambiguous or unclear or that the exercise is too strenuous. Other approaches supporting instructors in preparing and monitoring courseware include automated FAQ creation [39], creating adaptive textbooks [28], and finding optimal sequences in courseware [159]

Another approach supporting instructors in monitoring and creating content was adopted in [97]. Liu et al. implemented a bottom-up approach that highly focused on the instructors' needs. The data collected is processed and can be analysed by the instructor using an analysis engine. Amongst other insights, instructors can derive

meaning from their course data by selecting different contexts. The proposed system is part of an ongoing evaluation covering multiple courses and is under a constant improvement and adaptation cycle. The Curriculum Analytics Tool (CAT) developed in [68] allows instructors insight into their curriculum by analysing competencies and providing recommendations about improvements to the instructors. The presented tool utilises Natural Language Processing (NLP) techniques and similarity matching to recommend unique competency verbs for new modules, supporting instructors to list competencies and map the competencies to learning outcomes. Among other things, an evaluation of the accuracy of these recommendations is conducted and shows an overall accuracy of 74.69%. The tool can provide further insight into modules' learning contents and allow instructors to manage and adapt their modules according to the competencies.

Learning Analytics has the potential to improve how instructors create and maintain learning content, and in turn, significantly improve students learning experience by providing personalised recommendations, tutoring and supplemental resources [145]. By providing insightful analysis to instructors and allowing research to develop instructors-focused application educational data mining and learning, analytics can provide higher education facilities *"from simply understanding various data points [...], to using them to create actionable intelligence"* [146].

2.3.2 Assessment & Plagiarism Detection

Grading students' work is a crucial aspect of instructors work. Many modules in modern higher level education programmes have transitioned from an end-of-term final examination to continuous assessments throughout the term. From classic in-class tests and homework to projects and reports, a wide variety of possible assessments are available. While continuous assessment offers many advantages, such as monitoring students progress, higher engagement and practical experience, it also comes with an increased workload for instructors. Students must receive timely and detailed feedback on their continuous assessments to allow them to improve during a teaching term. However, depending on the class size and the type of assessment, the workload can be overwhelming for instructors. Further, additional challenges such as the risk of plagiarism can arise and require further attention.

As there are many different ways assessments can be conducted, there is a large variety of research conducted in the EDM/LA community to assist instructors in grading and providing feedback. While many Learning Management Systems already provide tools to create and automatically grade assessments, such as multiple choice and one-

line answer tests, the technical approaches in EDM/LA can provide advanced support for more detailed assignments. The assessment of free text, for example, in open-ended questions or reports, can benefit from text mining approaches. Supervised learning algorithms, active learning [55], and clustering approaches have all been shown to be beneficial. Escudeiro et al. present an active learning and automatic text classification approach in [55] that allows classification of free-text answers in exams. The approach was evaluated on real-world data from software engineering students. The overall accuracy of 68% was deemed reasonable, and it was concluded that a more diverse set of topics are needed to improve the system. Other approaches have focused on automated grading of programming assignments [38]. Orr and Russel present a neural network approach to automated grading and feedback generation for Python programming assignments [131]. An abstract syntax tree is used to represent the program. Personalised feedback is generated based on the features of the individual programs. The evaluation shows that the model can predict design scores with up to 94% accuracy and that students who followed the personalised feedback improved their program by over 19%. Recently, research has tackled the task of automated grading for presentations; for example, Haider et al. propose an automatic scoring system for presentation delivery skills [74].

Continuous assessments, while providing many advantages, also increases the workload for students drastically. Plagiarism is a growing problem, as students copy work from different sources, such as books, websites, forums and their peers, while failing to provide the appropriate citation. When grading students' assignments, there is not always ample time or resources available to the instructors to check for plagiarism. Plagiarism detection systems have been developed for many years [122], with more sophisticated and specialised approaches being implemented for new challenges, such as programming code plagiarism [41]. For example, Gehringer et al. present an interesting overview of previously presented techniques for plagiarism detection, such as Levenshtein and Smith-Waterman distance, or n-gram similarity. In [64] Gehringer et al. show how these approaches work differently depending on the type of exam question. It was concluded that it is clear that some techniques are more suitable for certain types of exams and that the technique used for plagiarism detection needs to be selected carefully to be beneficial.

Automated assessment has the potential to not only decrease the time commitment for instructors but also to standardise the assessment criteria and decrease the grading bias that can occur when multiple instructors are involved with the grading. In addition, timely feedback on students assignments allows students to improve their interaction with the module and can lead to overall higher performances.

2.3.3 Monitor Student Progress

Monitoring students' progress throughout the course can provide valuable insights for instructors. Identifying students who might be at risk of being left behind and being able to provide support and intervention is a primary objective for instructors. In traditional classroom education, instructors can easily assess students progress by their interaction with the material, while in blended education, some of this interaction is hidden online. Data mining techniques allow insights into students performance, interaction with the learning material and assessment performance, giving instructors valuable knowledge about the progress of the student cohort. Additionally, research in EDM/LA has provided work that focuses on computer-supported behavioural analytics (CSBA) that encompass the topics above as well as knowledge from other data sources, combined with computer-supported visualisation analytics (CSVA) these can provide vital insights for instructors into students' progress and learning experience [11].

The data collected through Learning Management Systems can act as a rich information source and has been widely used to predict students progress, and performance [34, 33, 117, 195]. For example, in [111] McCuaig and Baldwin show that the interaction data of students with the LMS can be harvested to predict the students' success in a module. Using decision trees, McCuaig and Baldwin show that using (semi-) passively collected student data from interaction with the LMS, students' success rate can be predicted. While a thorough evaluation is left for future work, the work concludes that two pieces of data seem particularly useful for success prediction, namely active days and self-check problem scores. Similarly, Macfadyen and Dawson present work in which they were able to show that data from LMS can be used to predict students at-risk in a timely manner allowing for early intervention, but also that by analysing students' forums interactions, a rich student communication network can be generated that yields further insight in students progresses [103].

There is valuable knowledge hidden in student interaction data. For instructors to learn and react to the gained knowledge, the information needs to be presented clearly, precisely, and timely. Some research has recently focused on visualising student models, and some tools for visualising certain aspects are available, such as MOCLog [109], and CourseVis [110]. These tools highly focus on visualising the level of participation and social interactions on the LMS. In [7] Al-Ashmoery et al. present a tool that incorporates advanced text analysis, such as semantic analysis. The tool uses an adapted LMS plugin that collects additional interaction data from the LMS website and creates multiple interactive visualisations for instructors to learn and adapt their teaching.

While there is no shortage of related work proving the importance of educational data for instructors, providing the implemented data mining solutions seems to be less explored. Researchers seem to agree that implementing suitable monitoring technologies can support instructors in analysing the impact of their online learning materials on the overall student learning and experience [103, 79].

2.4 Stakeholder: Students

Lastly, students and learners represent the third stakeholder in the educational environment. Unequivocally students will benefit from improvements and implementation of EDM/LA applications on instructor and institution level. Additionally, students provide their very own objectives and tasks. Students require large amounts of information, accurately and well presented to them to make informed decisions on essential factors of their academic journey, such as selecting the right programme/major, courses/modules and electives, and extracurricular credits. Students are presented with vast amounts of new knowledge and topics that they are often unfamiliar with; finding suitable learning materials to prepare for lectures and exams adequately can pose a significant challenge to many students.

While the research in Educational Data Mining/Learning Analytics is distinctly focused on supporting students, the majority of which is indirect. However, some work has put the students in the foreground, most prominently course recommendations (discussed in detail in Chapter 3.3), as well as programme/major recommendations and long term academic planning, which we will describe in the following sections.

2.4.1 Choosing Programmes of Study

Before students enter third-level education, they have to make the first significant decision in their academic life: choosing the right university and programme. Additionally, in some universities, especially in the United States, students must choose their majors and minors. Most students start university right after leaving their secondary education, at a very young age. Navigating the vast number of different courses, universities, and all the requirements of these choices can be daunting. While most research in EDM/LA has focused on support students once they have entered tertiary education, some research has been conducted providing help for this first important step.

When recommending programmes/majors to students, a careful selection of input data is required as no specific data is available to hint at students' preferences for a particular academic career. While some high school and graduation performance data can give some clues, for example, knowledge in Mathematics or English, the options in tertiary education go far beyond those taught in secondary education. Therefore research in recommending programmes/majors sometimes collect additional explicit information from the students. In [14] a rule-based system was implemented that uses information collected from English language and intelligence tests additionally to their academic record and demographic information. The decision support system was not evaluated, and it was concluded that the system would require more students for advanced testing. Deorah et al. allowed students to explicitly state programmes they are interested in, and those they are not, additionally to different high school performance data and personal details derived from a psychological questionnaire [45]. The approach uses case-based and rule-based reasoning to recommend the most suitable majors. Candidate majors are ranked based on the probability of successful completion. The evaluation showed that the system could make strong or mild recommendations for 68% of the test users. A focus on using secondary education data was proposed in the recommender system presented in [59] by Fong and Biuk-Aghai. The three-tier system architecture presents analyses, classifies and visualises student data from secondary school records, such as test grades and student profiles. A C4.5 decision tree model is used to generate rules which are then visualised for user interaction. Using historical student data and human user feedback, the evaluation focused on predicting university admission chances in different locations. F1 scores were used to compare different classifier approaches and concluded that the hybrid system provides slightly better results.

A case-based reasoning approach was presented in [121], in which courses are represented by their weighted concepts. Students are then modelled as cases based on the courses they have taken in the past. Recommendations are made based on the case similarity. The presented system was not evaluated, but a survey was proposed to test the system regarding usability, efficiency, maintainability, and portability. Another student-similarity based approach was implemented and tested in [114]. Meller et al. use students' academic histories to match it to programme structures using a nearest-neighbour algorithm and compare it to Naive Bayes and J48 approaches. While the newly presented approach outperformed the baselines in some cases, it was stated that a more detailed academic history modelling is needed in other cases. Finally, in [139], Park proposes a collaborative filtering recommender system based on a previous personalised grade prediction system, presented in [138], to recommend major, minor, and concentrations to students. The system can provide grade predictions for selected

major/minor/concentration alongside a list of predicted grades for modules and electives within the chosen course. Further, the proposed system can rank a list of courses based on the overall grade prediction. The system was not evaluated, but thorough testing is mentioned as the next step in this research.

Choosing the right programme and majors is the first vital step for students' successful academic journey. The related work presented is only a snippet of the current efforts to help young people in this significant decision. However, most research focuses on guiding students toward programmes and majors that they predict will ultimately lead to a high graduation grade. Other important factors, such as career choices and personal interests, are rarely highlighted. Further, in most work presented, students are faced with a black box solution. Only recently, more interest has been shown to provide visualisation and explanations for recommendations that can help students receive recommendations and ultimately allow them to gain knowledge and make informed decisions themselves; we will present related work in Section 3.2.

2.4.2 Navigating Learning Material

We previously presented EDM/LA approaches aimed at Learning Management Systems and Massive Open Online Courses from the perspective of the institutions and instructors. While these stakeholders are the predominant creators of the content, the students are on the receiving end. With the increasing amount of blended learning and online learning materials, students face the challenge of navigating the growing amount of information. While there is some research focusing on analysing students' behaviour and interaction on these websites, only a handful of work has been directed to actively support students in finding suitable materials at the right time.

Research in learning material recommendation often focuses on MOOCs as these can be more self-regulated than learning material in traditional academic settings. In MOOCs, students navigate various forms of learning material, such as videos, documents, quizzes and exams. Users and concepts of MOOCs can be represented using a Heterogeneous Information Network [142]. The network can be harvested to predict concepts for users of a MOOC to extend their current learning path. Frequently users of MOOCs use multiple platforms to learn about their topic of choice. Recent work has been conducted to represent the best sequences of topics in online courses [159]. Rudian and Pinkwart use Search Engine Result Pages of Google (SERPs) to rank 20 topics related to Artificial Intelligence to find the optimal order to study these topics. Three different algorithms are tested and evaluated against an expert ground truth constructed by four AI instructors. The results show that, while commercial popularity

extracted from SERPs are not a good indicator for topic sequences, a pair-wise comparison might be used to order topics within online courses. Other approaches to helping students in online learning scenarios include recommendations of remedial readings [184]. Thaker et al. present a domain-specific recommender system that is able to recommend documents based on students knowledge of concepts. Static Remedial Recommendation (StatRemRec) and Dynamic Remedial Recommendation (DynRemRec) were tested on an online reading platform dataset comprising quiz interactions. Different approaches were evaluated using Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) metrics, based on the ranked recommendations compared to the ground truth. The results show that using advanced approaches that incorporates domain knowledge significantly increases recommendation performance.

Adaptive learning is an emerging research area that aims to provide a unique learning experience based on students comprehension and preferences [108]. In 2016, Gartner ranked adaptive learning as the number one strategic technology in higher education¹⁰ and defined it as follows: *"Adaptive learning dynamically adjusts the way instructional content is presented to students based on their responses or preferences"*. Research in adaptive learning can provide helpful insight in three different parts of the adaptive learning framework [108]: (i) learner models can provide helpful insight into students attributes, preferences, and motivation[189]; (ii) other work focuses on the content model, that involves concepts to create learning maps and the delivery of the content [127]; (iii) the third part of the adaptive learning framework is the instructional model, research focused on this model aims to support instructors in providing guidance in sequencing and pacing of content [94].

While a considerable amount of research is focused on improving and analysing learning material, most of it does not seem to be directly targeted at the students. EDM/LA state *actionable intelligence* as one of their main objectives, therefore making the knowledge extracted from analysing educational material directly available to the student stakeholders poses an immense opportunity that so far seems under-explored.

2.4.3 Long Term Academic Planning

Once students are enrolled in their programme of choice, they are faced with even more choices to make. Most undergraduate programmes allow for the personalisation of the curriculum. Students can choose optional modules, decide on specific streams and have to make elective module choices. While some of those decisions are only made

¹⁰<https://www.gartner.com/en/newsroom/press-releases/2016-02-25-gartner-highlights-top-10-strategic-technologies-for-higher-education-in-2016>

on a semester to semester basis, they can ultimately have implications and effects on their long term academic path. These effects can be challenging for students to realise, especially in their first few semesters at the university. Therefore, higher level educations offer the help of academic advisors to the students to help build personalised academic paths. Academic advising, however, is a time-consuming task, and especially in online or blended learning, not always accessible to all students. The majority of work in EDM/LA has therefore concentrated on improving academic advising, module and module sequence recommendations, and automated academic advising. While we present related work in module recommender systems in detail in Chapter 3.3, in this section, we will briefly outline related work specifically dedicated to long term academic planning.

Multiple factors play a critical role when creating their personal academic paths, making this process particularly complicated and time-consuming. Firstly, personal interests, strengths and future career plans influence students choices. Secondly, constraints are given by the programme and university, such as pre- and co-requisites, incompatibilities and graduation requirements. Other factors for students include timetable constraints, personal preferences in lecturer, lecture and assessment style, and module difficulty. Most research in automated academic advising, therefore, concentrates on a subset of these factors. For example, in [196] Werghi and Kamoun present a Decision Support System (DSS) that implements a decision tree that can be traversed to find the optimal module sequence. The approach mainly considers three requirements, namely pre-requisites, the minimum amount of time to graduation, and academic recommendations. Additionally, student constraints, e.g. the maximum number of modules they would like to take per semester, are considered. The proposed system then conducts a search to determine the optimal student academic plan. While a prototype user interface was implemented, no testing or evaluation was presented. Recently, a similar DSS approach was presented by Shakhshi-Niaei and Abeuei-Mehrizi in [167]. The proposed system is based on an optimisation model that allows for a higher personalisation through direct students preferences. Students actively decide their preferences for factors, such as interest in different elective courses, preferred complexity, and availability in summer semesters. A linear multi-objective model is implemented using Microsoft Access and evaluated using students from different years of study and validated by the authors.

Academic advising systems are one of the most researched areas within the EDM/LA community [81, 17]. The complexity of requirements and hard and soft constraints of the university paired with students' personal preferences makes for an attractive problem area that lends itself to a variety of different machine learning, text mining,

and analytics techniques. Recommender systems provide a robust approach to the problem area; recommending modules, sequences, and electives can support academic advising systems. The following chapter will introduce the technical details of recommender systems techniques, evaluation, and advanced techniques before presenting related research for academic advising.

2.5 Discussion

Educational Data Mining, Learning Analytics and adjacent research communities are relatively new fields of study. The changing environment of education provides a new quality of educational data and allows researchers from vastly different backgrounds to apply their knowledge. Technical approaches from backgrounds such as Data Mining, Artificial Intelligence, Information Retrieval, and Network Analysis, paired with the substantial information provided by long-standing research in Pedagogy and Psychology, make for an exciting research field. We have presented an excerpt of research conducted that focus on different stakeholders and their tasks and objectives within the educational process.

Throughout the related work, we have seen that much work tackles the task of choosing suitable items. For example, this might be the proper order in timetables, the most suitable modules and courses for students or the best content for a specific module. Further, we have seen that more often than not, related work focuses on performance data, such as grades, to predict those items. However, while substantial work is being conducted, evaluation is not always done or not sufficiently targeted at the stakeholders. We argue that there is a strong focus in the research community on performance data as input/output and that offline evaluation using accuracy-related metrics lack the ability to evaluate usefulness in many parts. Further, we detect a lack of work with a strong focus on one or more stakeholders, failing to include the appropriate actors in creating and evaluating the implemented solutions.

We have seen the importance of recommender systems in research done in the EDM/LA community. Recommender systems lend themselves to support the task of finding the most suitable items and can be adapted to many different objectives. Furthermore, recommender systems often allow us to serve multiple stakeholders simultaneously. Therefore, it is not surprising that numerous work in EDM/LA focuses on recommender systems, especially recommender systems for module recommendations and academic advising. The following chapter presents an overview of recommender

systems techniques and evaluation metrics before introducing related work targeted at recommender systems for academic advising.

RECOMMENDER SYSTEMS AND APPLICATIONS IN EDUCATIONAL DATA MINING/LEARNING ANALYTICS

In the most general way, recommender systems can be defined as software tools and techniques that aim to recommend items to users [153]. However, in our modern world, where the number of options for any given choice can be overwhelming due to the explosive growth of available information online, filtering, prioritising, and ultimately finding the right items and information is a significant challenge. Therefore, recommender systems have evolved to be an indispensable component of the internet; by personalising and prioritising information, recommender systems play a vital part in users' decision making processes. Due to their importance in our modern day-to-day life, recommender systems have steadily evolved into one of the most extensive research fields in computer science, as well as other research areas such as economics, sociology, and psychology [98]. As a result, the number of algorithmic approaches and application areas are vast and rapidly expanding.

This chapter introduces the basic recommender system techniques, collaborative filtering and content-based approaches, as well as a brief introduction of hybrid recommender system techniques. We further briefly introduce how recommender systems can be evaluated using classic evaluation metrics and approaches that go beyond, by introducing novelty, diversity and serendipity. Finally, we present an overview of related work showcasing recommender system approaches for academic advising and module recommendations.

3.1 Types of Recommender Systems

Recommender systems have been an important research area since their first mention in the 1990s [3]. Since then, there has been much work done, developing new techniques and approaches. Both academia and industry have shown great interest and have helped the area develop rapidly in the last 30 years. The most prominent examples of early recommender systems are Amazon [174], MovieLens [75], and Netflix [67], helping users find and make informed decisions about their choices in movies, books and music; we have presented some examples in Section 1. Since then, recommender systems have evolved to be a part of most application areas online, either explicit or implicit. Nowadays, recommender systems aid users in finding the right hotels [160], partners [125], and jobs [46], just to name a few.

Recommender systems have been formally defined throughout the years; a general definition that is widely used was presented by Adomavicius and Tuzhilin in 2005 [3]:

More formally, the recommendation problem can be formulated as follows: Let C be the set of all users and let S be the set of all possible items that can be recommended. Let u be a utility function that measures the usefulness of item s to user c , that is, $u : C \times S \Rightarrow R$, where R is a totally ordered set (for example, nonnegative integers or real numbers within a certain range). Then, for each user $c \in C$, we want to choose such item $s \in S$ that maximizes the user's utility.

Even though one could argue that this definition is not general enough to include the most recent developments, it captures the foundation of recommender systems sufficiently. Although recommender systems have seen such a dramatic increase in research interest and application area, most approaches can be categorised in either *collaborative filtering*, *content-based* or *hybrid* approaches. We will present the three approaches in the following sections.

3.1.1 Collaborative Filtering

By far, the most widely used recommender system technique is based on collaborative filtering. This basic approach can be explained in layman's terms as "a system that will recommend items to users that similar users have liked in the past". The most well-known examples include Amazon's "*People like you also bought these items*", Reddit and YouTube¹.

¹<http://www.youtube.com>

For collaborative filtering approaches a user-item matrix is often created where each cell of the matrix $[U_n, I_m]$ represents the rating $r(n, m)$ of a User U_n for an item I_m , see Figure 3.1. Ratings can either be collected explicitly by asking users their feedback for a specific item they have purchased/consumed. These ratings can have multiple formats, such as Likert scales (e.g. 1-5 stars) or binary (like/dislike). Further implicit feedback can be collected by harvesting the users' interaction. For example, implicit feedback can be the number of listens to a song or time spend reading a news article.

	I_1	...	I_i	I_m
U_1						
U_2						
...						
U_j			$r(i,j)$			
...						
U_n						

Figure 3.1: User-Item Matrix for Collaborative Filtering Recommender System.

There are two general ways a collaborative filtering approach can be used to make predictions for users: (i) in a *user-based* approach, the user-item matrix is used to calculate the most similar users to the target user, based on those similar users, a rating for an unseen item for the target user is predicted; (ii) *item-based* approaches calculate the pairwise similarity of items that the target user has rated to the target item based on other users' ratings [86, 164]. The calculated similarities can then be used to either predict a rating for the target item to the target user or present a list of recommendations in a ranked top-n list.

Another widely used technique that has received much attention since it was used in the winning solution of the Netflix Prize challenge [92] is matrix factorisation, a collaborative filtering approach that is based on the assumption of latent features that can be used to capture user preferences [132]. Matrix factorisation involves the decomposition of the user-item matrix into two lower dimensionality matrices, namely the user-feature and item-feature matrices. Once the decomposition has been completed,

predictions for a given user-item pair is obtained by taking the dot product of the corresponding user and item feature vectors.

	Star Wars	Enchanted	Titanic	Robocop	Pretty Woman
Maria	1	3	5		5
Noel	3	1	2	4	2
Emiliy		5	4	1	5
Alison	4	3	4	2	1
Brian	3	?	1	5	2

Figure 3.2: User-Item Matrix Example for Collaborative Filtering.

We will explain a user-based collaborative filtering approach with a simple movie recommendation example. In Figure 3.2 we present example preference data (e.g. star ratings) for five users and five movies. To predict if the target user *Brian* would like or dislike the movie *Enchanted*, we first compute the similarity between the target user and all other users in the system. To achieve this, different approaches can be considered, such as Mean Squared Difference, Pearson Correlation, or Cosine Similarity [27]. For this example, we will use the Mean Squared Difference (MSD) metric [169] and compute the MSD between user a and user i as follows:

$$MSD_{a,i} = \frac{\sum_{j \in I_a \cap I_i} (r_{a,j} - r_{i,j})^2}{|\{j : j \in I_a \cap I_i\}|} \quad (3.1)$$

where I_a is the set of item user a has rated and $r_{a,j}$ denotes the rating for item j from user a . We then convert the difference into a similarity metric, using the minimum (r_{min}) and maximum ratings (r_{max}) as follows:

$$sim_{a,i} = 1 - \frac{MSD_{a,i}}{(r_{max} - r_{min})^2} \quad (3.2)$$

We can compute the similarities between the target user and the other users in our toy example. User-based collaborative filtering would then select a subset of similar users, *neighbours*, to use to calculate predictions. In our example we choose the two most similar users to our target user *Brian*, that is user *Noel* ($sim_{Brian,Noel} = 0.96$) and *Alison* ($sim_{Brian,Alison} = 0.91$). More advanced approaches for neighbourhood selection, such as similarity thresholding [76] can be used and can have implications on algorithm

performance. The generated neighbourhood can then be used to compute predictions for specific items or generate ranked lists of recommendations. In our example, we want to make a prediction for the target item *Enchanted* to the target user *Brian*. To calculate the prediction, multiple approaches can be used. The simplest option is to compute the mean of the neighbours' ratings for the target item, as follows:

$$pred_{a,j} = \frac{\sum_{i=1}^n r_{i,j}}{n} \quad (3.3)$$

where n is the number of neighbours. In our example this would mean we compute a predicted rating $pred_{\text{Brian}, \text{Enchanted}} = 2$. More advanced approaches such as weighted averages and Deviation from Mean [77] can be used to improve prediction accuracy.

In our small example, our user-item matrix is filled nearly completely; however, such a matrix would be very sparse in the real world, as users can only see and rate a small fraction of all available items. Further, user-based collaborative filtering suffers from scalability problems as the computation grows with the number of users and the number of items. One way of dealing with this bottleneck is an item-based collaborative filtering approach. For this, the pairwise similarity between all items in the data set is computed over the set of users who have rated both items. Similarity can be calculated using the same metrics (albeit adjusted for items) as user-based CF. From these similarities, item neighbourhoods can be created consisting of the most similar items. Using the neighbourhood, a prediction for an item for a target user can be computed using different approaches, such as weighted average, where item similarity acts as the weight.

Due to their independence of domain knowledge, collaborative filtering approaches are often used over more content-based approaches. In addition, user data is often available in abundance as it can be created implicitly, whereas content-based representations of items often have to be created (semi-) manually and need to be maintained due to its changing behaviour.

On the other hand, there are multiple challenges known for pure collaborative recommender systems. The *Cold Start Problem* describes the issues that a CF approach needs to require a critical amount of knowledge before recommendations can be calculated [165]. This cold start problem reoccurs for every new item and every new user to the system. Further *data sparsity* has been considered an issue in CF approaches. User-item matrices will have potentially millions of users and often even more items; however, every user will only have rated a small fraction of the available items. It was shown that if, on average, over 99.5% of users of a system have rated less than 1 – 2% of available items, the accuracy of recommendations decreases drastically [90, 115].

3.1.2 Content-Based

In comparison to collaborative filtering based recommender systems, content-based approaches are mainly focused on the characteristics of the items [140]. In content-based recommender systems, items are recommended to a user based on the similarities between target items and other items in the database. In layman's terms, a content-based recommender system can be described as, *recommending items to a user that are similar to items he/she has liked in the past*. To calculate the item-item similarity, an item profile needs to be created. The implementation of this item profile can require vastly different techniques depending on the item and its characteristics. Content-based approaches have their roots in information retrieval [15]. Oftentimes creating the items' features includes a textual analysis.

We can consider our example from Section 3.1.1 again, but instead of focusing on user ratings, we scrape the movies' information from a website. This information could include actors, genres, directors, and plots. To represent the items, we vectorise the documents by defining the terms in the corpus and calculate a score for each document and each term. The two most popular approaches for vectorisation are Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF), explained in detail in Section 4.4. In this introductory example, we will use the more straightforward BOW approach to vectorise the documents in our example.

The BOW approach counts the occurrences of each term in each document, also called the term frequency. After cleaning and preprocessing, such as removal of stop words, term stemming and converting to lower case, the textual descriptors of the movies, we can create a simple term-frequency (TF) matrix, see Figure 3.3, where we denote the raw count of each word in each document. In the BOW approach, the order of the words has no effect; therefore, sometimes, an n -gram model is used in which n terms are considered as one term. For example, in our example, the term *ship* often occurs in *Star Wars* as well as *Titanic*; if we considered bi-grams, the terms *assault ship* and *passenger ship* would be beneficial to distinguish between these two movies.

In this simple example we can use the term-frequency matrix to calculate the similarity between two documents/movies d_i and d_j as the angle between their vectors \vec{V} , as follows:

$$\text{sim}(d_i, d_j) = \frac{\vec{V}(d_i) \cdot \vec{V}(d_j)}{|\vec{V}(d_i)| |\vec{V}(d_j)|} \quad (3.4)$$

To make a recommendation to a user, we can then create a user profile by either collecting explicit information (e.g. asking the user about their favourite movies) or gather

	Star Wars	Enchanted	Titanic	Robocop	Pretty Woman
action	5	1	0	15	0
love	7	9	13	0	8
ship	8	0	15	0	0
roberts	0	0	0	0	6
...

Figure 3.3: Term Frequency Matrix Example.

user preferences by implicit behaviour (e.g. previously seen movies, clicked movies). Using similarity metrics, we can compute the similarities between the target user's profile and available movies. In our example, the target user *Brian* has rated the movie *Robocop* the highest; we could therefore recommend a ranked list of movies in order of descending similarity to this movie. In the small dataset in our example, the movie *Star Wars* would rank the highest.

Content-based filtering approaches have the advantage that they do not require a large amount of user data. On the other hand, domain knowledge is needed to feature engineer the item characteristics. Content-based approaches also suffer from the *new user* problem, while not to the same extent as collaborative filtering, which describes that there cannot be any recommendations calculated for a new user until some user interaction/preference data is gathered to create a user profile. To counteract this issue and other problems described by the two presented approaches, most recommender systems today use a hybrid approach which includes both and/or other recommender systems techniques. We will present some hybrid approaches in the following section.

3.1.3 Hybrid Approaches

Hybrid recommender systems can be implemented in various ways. Adomavicius and Tuzhilin [3] classify four different general approaches: (i) combining separate recommender systems, (ii) adding content-based characteristics to collaborative modules, (iii) incorporating collaborative filtering techniques to content-based recommenders, and (iv) combine both techniques to a new unifying model. In [32] Burke identifies seven different types of strategies for hybrid recommender systems: (i) weighted, (ii) switching, (iii) mixed, (iv) feature combination, (v) feature augmentation, (vi) cascade,

and (vii) meta-level. Burke presents the advantages and disadvantages of the different approaches and gives several examples of different prototypes.

Recently Cano and Morisio conducted a systematic literature review of 76 hybrid recommender research papers [35]. The work presented the hybrid recommender system approaches clustered by the seven classes defined by Burke [32]. The most frequent approaches are shown to be *weighted* hybrid systems, that is, systems that combine the score of different recommendation components numerically to a final recommendation score. Interestingly the survey also identified *education* as the third-highest application area of hybrid recommender systems (after *domain independent* and *movie* recommender systems applications). Overall, increased research interest in hybrid approaches has been shown, making for an exciting and fast progressing area.

Hybrid recommender systems have been shown to help overcome traditional recommender systems issues, such as the cold start or grey sheep problem [31, 66]. We will present some examples of hybrid recommender systems for academic purposes in Chapter 3.3.

3.2 Evaluating Recommender Systems

Generally, a recommender system can be evaluated in two ways: (i) online and (ii) offline. An *online study* where a body of users is allowed to interact with the system and potentially give explicit feedback about the recommendations can give vital insight into the usefulness of the system. However, online user studies can be costly – and are not always feasible; hence recommender systems are often validated using an *offline evaluation*.

3.2.1 Offline Evaluation: Classic Evaluation Metrics

Offline evaluations are widely performed due to their low costs and easily accessible data. Using historic or synthetic datasets, the effectiveness of the recommender system can be evaluated. Offline evaluations are attractive because they are reproducible and do not require user interaction. Offline evaluations simulate the recommendation process by using techniques such as cross-validation [158], the dataset is split into training and test set. The first is used to train the algorithm, whereas the test set is then used to validate the performance of the recommender system.

For top-n recommendation problems two metrics have been established as the standard: (i) *Precision* to depict the fraction of recommended relevant items, see Equation 3.5

and (ii) *Recall* represents the probability that a relevant item will be recommended, see Equation 3.6 [76].

For example, if we recommend 15 movies (the *retrieved items*) to a user (with a set of ten *relevant items*); if five out of the 15 retrieved items are within the set of *relevant items*, we calculate a precision of 33% and a recall of 50%.

$$Precision = \frac{|\text{relevantItems} \cap \text{retrievedItems}|}{|\text{retrievedItems}|} \quad (3.5)$$

$$Recall = \frac{|\text{relevantItems} \cap \text{retrievedItems}|}{|\text{relevantItems}|} \quad (3.6)$$

However, if the user has a relevant item set of size 100, the recall would decrease. Therefore precision alone is often considered to be biased. Likewise, it is possible to achieve a 100% recall by recommending all items. Both metrics can be combined to the *F1 metric*, see Equation 3.7, to reduce the risk of bias between precision and recall [188].

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (3.7)$$

The conventional accuracy evaluation measures are deficient in some key aspects. Where recommender systems focus solely on increasing accuracy, an implicit bias might be introduced towards popular items in the dataset. This leads to an exacerbation of the *long-tail* [83]. By inadvertently narrowing the items that are being recommended, the importance of novelty, diversity and serendipity in recommendations is neglected. In the following section, we will present research related to these important aspects of recommender system evaluation.

3.2.2 Beyond Accuracy

Traditionally recommender systems have been focused on predicting items as accurately as possible. However, with the growing application area and advanced research, it became apparent that accuracy is not the only important factor when developing recommender systems [113, 85]. For example, an eCommerce recommender system that would only recommend TVs after a user recently bought a TV would be of very little use. Likewise, a music recommender system would fail to impress its users if it only played music from artists the user had previously liked. While these are extreme examples, it accentuates the importance of building recommender systems that look beyond accuracy. Research has shown that introducing diversity, novelty, and serendipity into

the recommender system process can improve user satisfaction, and trust in the system [203, 173]. In the following sections, we will briefly outline diversity, novelty, and serendipity in recommender systems.

3.2.2.1 Diversity

Diversity is generally applied to a set of items and measures how different each item in a list of recommendations is to each other whilst still being relevant [4]. For example, we can imagine the recommendation of a list of all *Star Wars* movies to be less valuable to a user than a list of a more diverse set of *Science Fiction* movies. The most commonly used metric for measuring diversity for a set of recommendations R is defined by Smyth and McClave [175] as the pairwise distance between the items as follows:

$$Diversity(R) = \frac{\sum_{i \in R} \sum_{j \in R \setminus \{i\}} dist(i, j)}{|R|(|R| - 1)} \quad (3.8)$$

In the simplest form $dist$ can be defined as the inverse similarity of two items $1 - sim(i, j)$, where $sim(i, j)$ is the similarity between two items.

This definition has been widely accepted and used within recommender systems research. In addition, other definitions of diversity have been generated for application-specific problems [154, 173] and methods for increasing diversity within a ranked list were explored [80].

3.2.2.2 Novelty & Serendipity

Novelty can be described as a measure of how different an item is to a specific user's profile without disregarding accuracy [205]. There are multiple metrics proposed to measure novelty within a list of recommendations R . Commonly novelty can be defined as the complement of an items popularity $1 - p(i)$ where $p(i) = \frac{|\{u \in U, r_{ui} \neq \emptyset\}|}{|U|}$ depicts the percentage of users U who rated item i , defined as follows [85]:

$$Novelty(R) = \frac{\sum_{i \in R} -\log_2 p(i)}{|R|} \quad (3.9)$$

On the other hand, serendipity, defined as the "*phenomenon of finding valuable or agreeable things not sought for*"², is harder to formalise. While there is no generally accepted definition of serendipity in the recommender systems community, it is widely agreed

²<https://www.merriam-webster.com/dictionary/serendipity>

that it consists of two factors – surprise and relevance [78]. One approach of formalising serendipity, based on these two components, was presented by Ge et al. in [63] for a set of recommendations R for user u as follows:

$$\text{Serendipity}(R, u) = \frac{|R_{\text{unexp}} \cap R_{\text{useful}}|}{|R|} \quad (3.10)$$

where R_{unexp} denotes the subset of unexpected items, and R_{useful} are the useful items. While the usefulness of an item can be calculated using traditional similarity metrics, measuring unexpectedness proves to be more challenging but is often defined as the distance from a set of expected items [2].

Introducing any of these "beyond accuracy"-objectives into a recommender system allows for a potential decrease in accuracy, however often at the gain of user satisfaction. We will show how we implement a sense of novelty, diversity, and serendipity into our hybrid recommender system in Chapter 4.

3.2.3 Online Evaluations

Online evaluation, while still being less common than their offline counterparts, are becoming increasingly more popular [54]. Online evaluations can be performed either as a user study, where a small set of users trial a system and answers related questions or as large scale experiments that evaluate the performance of recommender systems in more detail [168]. A user study is conducted by observing user behaviour with the trialled recommender system and potentially collecting additional qualitative information in the form of surveys before, during and after the interaction. More extensive online evaluations can be conducted by performing A/B testing, in which two or more different versions of the system are presented to different sets of participants. This allows for a reliable and in-depth evaluation of the performance of the recommender system.

In comparison to offline evaluations, online evaluations have the advantage that different system goals can be tested, such as the user experience and satisfaction with the system. Additionally to classic evaluation metrics, the qualitative data collected in online evaluations allow for a more detailed and realistic evaluation of the system's performance and the users' perception of the system. However, due to the high costs and limited access to sufficient participants, online evaluations need to be carefully designed and considered [54].

3.3 Recommender Systems for Academic Advising

Recommender systems are a vital part of EMD and LA research, with various techniques and objectives. In Chapter 2 we presented how recommender systems are used in EDM/LA to support different objectives and tasks for the stakeholders involved. In this section, we focus on the task of recommending courses/modules to students. While there are different definitions of the term *course* and *module* around the world, in this work, we consider them to be equivalent and describe a single class taught within a programme of study, such as the course/module *Programming I* taught in the programme *BSc Computer Science*. Module selection plays a significant role in students' academic lives, and multiple factors are vital in making this decision. These factors can be grouped into two sections, (i) personal factors, such as career goals, interests, and (ii) organisational factors, such as timetable constraints and pre- and co-requisites [100]. Many important factors to consider, paired with the overwhelming importance of making the right decision, can lead to a significant challenge for students. Recommender systems have shown to be a vital part of people's decision-making process online, and their makeup lends itself to the application area of module recommendation, where it has the great potential to support students in making the right decisions [88]. In what follows, we will present the state-of-the-art research in the area of module recommendations and discuss possible shortcomings and future directions.

3.3.1 Collaborative Filtering Approaches

Collaborative filtering is still one of the most widely used recommender systems techniques, as presented previously. Its basic assumption that users will like what similar users have liked in the past can easily be adapted in the educational space. Further, collaborative filtering offers a wide variety of different techniques for prediction, such as classification, pattern mining and matrix factorisation, as well as descriptive methods, such as clustering and association rule mining [60].

Collaborative filtering approaches rely on the availability of user rating information, such as star ratings or likes/dislikes. This poses a problem in CF approaches for module recommendations, as ratings usually are not available for students' past modules. Many approaches have therefore used performance data, such as grades, to represent students' ratings of modules [8, 25]. For example, in [152], students' past grades are fed into an item- and user-based collaborative filtering approach to predict possible grades for elective modules. The modules with the best-predicted grades are presented in a ranked list. The Mean Absolute Error (MAE) values, between predicted and ac-

tual grades, are calculated for varying neighbourhood sizes. The results vary between 0.33 and 0.38. While there seems to be no statistically significant difference between approaches or neighbourhood sizes, the work concludes with a positive future work recommendation that includes an advanced approach that can take students' interests and learning objectives into account.

While grades might be readily available, their comparableness to preference data is disputable. Some research has, therefore, focused on binary data (module taken/module not taken) [22] or include other data, such as social or demographic data [61]. For example, in [12] module rating data was available, whereas in [37] teacher popularity was calculated and used in the collaborative filtering process. In addition, explicit student ratings for modules can be collected by asking the student to rate the modules, teachers, or skills directly [179].

While it is questionable if grades are equal to preferences, the fact that grade data is highly sparse can not be disputed; every student will only take a fraction of all available modules within the university. On average, an undergraduate student takes 5 to 10 modules each year. This means that the dataset available to create user profiles is minimal, especially for students in the initial stage of their academic career who are especially in need of support. Elbadrawy et al. approach this topic in their work in [51], by defining multi-granularity student and modules groups based on the enrolment pattern. These groups can be incorporated in different recommendation approaches, such as collaborative filtering, and have been shown to increase prediction accuracy.

Another often overlooked detail of module recommendations is the inherently sequential order of the curriculum. In most higher level institutions, undergraduate programmes are highly structured into years and levels. Most modules have pre- and co-requisites that mandate when a student can take specific modules. One approach to focusing on modules sequences was presented by Khorasani et al. in [89], where a Markov Chain model was integrated with a collaborative filtering recommender system. While the outcome was not evaluated against a baseline or ground truth, Khorasani et al. hypothesise that students who would follow their recommendation will have greater success than those who do not. A continuing study and evaluation of this result were stated as a future work once the data is available. Morsy and Karypis showed recently that sequences of modules could be used in representation learning to improve modules recommender systems [119].

Recent approaches have explored new data sources, such as information from social media networks, such as Twitter³ and Facebook⁴, to determine a students productivity

³<https://www.twitter.com>

⁴<https://www.facebook.com>

and motivation [105] or social tags to improve recommendations [91]. An increased interest in ontology-driven approaches can be detected in the past years. Ontologies can be used to incorporate learner characteristics such as learning style, study level and skill level into a recommender system [183, 65].

Even though collaborative filtering approaches to solve the module recommendation problem are prevalent, we argue that they have some critical limitations. The majority of the presented recommender systems in this section focus on students' past performance data as the input and the deciding factor for recommended modules. They also serve as the primary evaluation component. While we agree that students performance is an integral part of choosing the suitable modules, we argue that not only do grades not necessarily reflect personal preferences, but further should not be the deciding factor in students' academic choices. We argue that modules that are intrinsically suited to a specific student can increase their possibility of achieving high grades while strengthening their knowledge in their specific interest. Further, pure collaborative filtering approaches could be prone to the *lemming effect*, i.e. a phenomenon where large crowds of people follow the same behaviour for no other reason than the majority of their peers do so [90]. Hypothesising that students' choices in modules are highly impacted by their peers' choices already leads to the long tail phenomenon. As collaborative filtering intrinsically harvests the strength of numbers, we suspect that these techniques could increase the problem of overpopulating already popular modules and missing other, more diverse and potentially more personally suited modules.

3.3.2 Content-Based Approaches

Content-based approaches and their hybrid combinations are less represented in the field of recommender systems for course recommendations. Even though the problem seems to lend itself to rich textual analysis of module content, there seem to be very few approaches harnessing this rich source of information. However, related studies have shown that text mining approaches can harvest helpful information and can be used to analyse and understand module data [87].

There seems to be a shift in research direction in recent years with more approaches, at least including textual module data. For example, Gulzar et al. present a knowledge-based approach that uses the content of modules to build an information retrieval based search engine that extracts modules based on keywords. An additional ontological system then identifies modules for recommendation related to the previously identified modules [69].

A mainly content-based approach, and the only one of the sort, was recently presented by Morsomme and Alferez [118]. The course data is used to build a Latent Dirichlet Allocation model [24], which, combined with the student data, allows a predictive model to predict grades and courses based on students explicit input in the form of keywords. Manual expert validation was used to test the presented approach, but no quantitative results were presented.

While recently more approaches utilise textual components of educational data, content-focused approaches to the module recommendation problem are sparse, and the majority of work in the EDM/LA community can be classified as hybrid approaches. In the following section, we present an excerpt of those approaches.

3.3.3 Hybrid & Other Approaches

A large amount of related work does not fall directly into either of the presented categories. Instead, most approaches combine different techniques to a hybrid recommender system. This section presents an excerpt of related work that uses multiple techniques and advanced approaches to recommend modules.

Sundari et al. present a rule-based approach in [177] taking students' performances and previous elective choices into consideration. Each module is manually assigned to one of four categories (programming, conceptual, logical, and theoretical). According to a rule-based classifier, recommendations are then made using past grades of modules with the same category tag as the modules explicitly chosen by the student. The approach is tested with historical student data against other (undefined) approaches and has been shown to produce better results than random allocation. Another rule-based approach was presented in [171] and later in detail presented in [170], where Association Rule Mining is used to predict suitable modules. The approach proceeds as follows: (i) using the Apriori algorithm and students historical performance data association rules are created, (ii) then for each student courses are suggested based on the created rules. An offline evaluation of 100 students' historical data shows a mean precision and recall ≤ 0.5 in predicted modules.

A system using a semantic web and ontology-based approach is used to model higher education institutions in [128]. The system uses student profiles and explicit information (provided through surveys) combined with an ontology that captures higher education and associated employment outcomes to build a model that can recommend majors and universities to high school students.

Course recommendation is presented as an optimisation problem and a solution using Ant Colony Optimisation (ACO) described in [176]. ACO can be used to find the best path in oriented acyclic graphs. Students' grades are predicted using historical student data. In an offline experiment comparing against content-based and collaborative-filtering approaches, the ACO approach showed promising results regarding Mean Absolute Error. Interesting work on the complexity of constraints within higher education institutions is presented in [134].

Instead of predicting a specific grade for a module, Morsy and Karypis focus on recommending sequences of modules that will improve the students' overall GPA as well as optimise their programme completion time [119]. Modules are defined as *good/bad subsequent modules* where the student's grade is higher/lower than their average previous grade. Using this definition, two models are trained to predict sequences of courses: (i) Singular Value Decomposition and (ii) Course2Vec, a neural network-based log-linear model. An extensive offline experiment showed that the proposed approaches outperformed the baseline in terms of precision and recall.

3.4 Ethical Implications of Recommender Systems for Academic Module Recommendations

With the omnipresence and widespread application of recommender systems today it is important to consider the ethical implications of the systems. Recommender systems have been shown to be beneficial in recommending a great variety of products and services. And while these systems are designed to aid the users, recommender systems are often predominantly designed to benefit the seller [136]. Items recommended can be categorised by their risks. Items such as e-commerce, music, and movie recommendation, have a comparably low risk. If the user watches a movie that they dislike, they might lose two hours of their life. Other items, such as holidays, high price items (i.e. cars), and services, such as dating recommendations, generally are considered a higher risk, either monetarily or otherwise. Building recommender systems for those high-risk items/services requires particular consideration of their ethical correctness [182].

Recommender systems for academic purposes, such as module recommender systems are considered high-risk systems, as incorrect recommendations can have a substantial impact on the users' life. Therefore, we carefully consider the ethical concerns regarding such a system. We can identify two main concerns. Firstly, there is the issue of incorrect recommendations, which are recommendations that are made from

incorrect or incomplete data. For example, a module may be substantially revised but this change is not captured by the recommender system. While revised modules can be readily accommodated by content-based recommenders, these revisions represent a particular challenge for non-content-based approaches, such as collaborative filtering algorithms. Secondly, there is the issue of unfulfilled promises. Recommender systems can be presented as a black box, that takes in preference data and provides recommendation output, without any explanation of how or why these items are recommended [136]. Oftentimes recommended items are ranked by a predicted rating, such as stars. In module recommender systems, especially when performance data is used, recommendations can promise a certain good grade, or an estimate of how likely it is the student will pass a module. Due to the aforementioned high cost, those unfulfilled promises can be problematic and should be avoided so as to not cause any harm to students, as well as maintain the students' trust in the system.

To address these ethical concerns, we define three main guidelines for the research conducted in this thesis. Firstly, we aim to build an open box approach, that facilitates students understanding of how and why modules are recommended to them. Secondly, we focus on recommendations based on content similarity and refrain from using students' performance data such as grades for module recommendations. Thirdly, we aim to build a system that not only provides recommendations but one that also allows students to actively explore the module space. This facilitates students to gain knowledge about their options and the connections between modules and skills. This in turn ultimately allows students to make their own informed decisions. These three guidelines will allow us to be mindful of potential ethical issues and will stand as a reminder to focus on the needs and benefits of the users.

3.5 Conclusion

This chapter shows an overview of recommender system technology, different approaches, their advantages and disadvantages. Then, we presented classic recommender system evaluation techniques and introduced metrics beyond the traditional accuracy measure. We presented related work in Educational Data Mining and Learning Analytics that aim to build recommender systems for (elective) modules. Finally, we briefly discussed potential ethical implications for module recommender systems.

The related work shows a strong focus in the EDM/LA community on module recommendations, with the large majority of work utilising collaborative or hybrid approaches. Pure content-based or content-based-focused approaches seem to be under-

utilised. Further, we detect a stark focus on grade and performance data used as input and output data in the presented recommender systems. While we agree about the importance of module performance outcomes, we argue that content-based recommender systems can improve module recommendations by diverting focus away from performance towards recommendations that more specifically target students' strengths and interests. We detect a general lack of content-based approaches in recommender systems for (elective) module recommendations.

Further, the module recommendation problem presents particular challenges; for example, modules covering the same topic can be taught by different lecturers, with consequences for delivery modality, assessment strategy, and content scope. These differences can impact whether a module should be recommended to a student; in contrast to a content-based approach, a grade-based collaborative filtering approach will struggle to detect this change due to the cold start problem. Those changes would need to be handled manually and could cause a high maintenance effort. The cost of an incorrect recommendation can also affect students' overall perception and acceptance of online advising tools. If a student follows a module recommendation somewhat blindly or expects a good performance but then fails to achieve the predicted grade, students may fault the recommender system and would potentially not use the system again.

Our main takeaway from the related work survey is that content-based approaches have been under-utilised in the research area of online academic advising, even though content-based approaches are competent in recommender problems in which rich content-based representations for items exist. We further argue that due to the high-cost nature of the recommendation, a *black box* approach is not desirable. Students need to understand why and how the recommendations are presented to them. The recommender system should provide adequate information to increase students' knowledge to ultimately allow them to make informed decisions for themselves rather than following a recommendation blindly.

HYBRID CONTENT-BASED ELECTIVE MODULE RECOMMENDER SYSTEM

The key aim of this research is to support students in making the right module decisions in their academic careers. Our first step in this is to aid the students in choosing suitable elective modules. In UCD, as in many other universities around Europe and worldwide, students can choose a number of elective modules each year in addition to their core modules. However, in previous research conducted in 2007 [130] and in more detail in 2016 [70] we determined that there is a lack of diversity in modules chosen as electives by the UCD undergraduate students. To address this problem, we introduce a hybrid content-based recommender system that aims to provide students with personalised recommendations from a diverse set of elective modules.

In Chapter 3 we concluded that most recommender systems in the area of academic advising make use of collaborative filtering. Recommendations are made based on the similarity of students, using their grades as key features. We propose to focus on content-based similarities between modules instead. We argue that despite the importance of students' grades, the suitable module for the right student will ultimately provide a better learning experience than a module a student only takes to "get an easy pass". Furthermore, as modules can change from semester to semester, e.g. due to a change in lecturer or topics covered, recommending modules solely based on grades can lead to a bad experience for students if the "promised" good grade fails to come true. Further, we argue that using grades can lead to ethical and privacy issues. Classic content-based recommender systems use a "more-like-this" approach to make recommendations, which are recommendations made based on similarities between item content descriptions rather than ratings and recommends items that are similar to those the user has liked in the past. Traditional content-based approaches work with unstructured content data, such as articles and web pages [140].

We use module descriptors found on the official UCD module catalogue website¹ as the item content in this study. First, we clean the textual description of the module content, learning outcomes, and other information using traditional Natural Language Processing (NLP) techniques [84]. Next, we use the cleaned textual data to build a Vector Space Model (VSM) to represent the items. We can then compute the similarities using a similarity metric. Instead of using the entire student history data as input for the user profile, we give the students the opportunity to log their preferences explicitly. This allows us to not use the grades as an assumption of interest in a module.

To address the lack of diversity in students' elective module choices, we use the taxonomy of the university's college/school/programme structure to recommend a more diverse set of modules while still keeping the recommendations relevant to the students' interests.

We conduct an offline evaluation on historical student data from University College Dublin and introduce a custom *simToCore* metric to determine the quality of our recommendations. We implement a web-based *UCD Module Advisor* prototype to showcase our hybrid recommender system, as well as additional functionalities, such as an intelligent search engine.

The key contributions of this chapter are the following:

- We implement a hybrid recommender system that utilises module descriptors and harvests the natural structure of the university taxonomy to introduce diversity into our recommendations.
- We conduct an offline evaluation using historical UCD student data and introduce a custom *simToCore* metric to determine the quality of the recommendations.
- We implement a web-based student advisor that includes an elective module recommender system and an intelligent search engine.

The chapter is organised as follows. In Section 4.1 we present our motivation and preliminary exploratory data analysis results and introduce the notion of diversity in Section 4.2. Our recommender system approach is presented in detail in Section 4.4. In Section 4.5 we present the details and results of our offline evaluation. The web-based *UCD Module Advisor* prototype is described in Section 4.6 before concluding the chapter in Section 4.7.

¹https://hub.ucd.ie/usis/!W_HU_MENU.P_PUBLISH?p_tag=MODSEARCHALL

4.1 Motivation

University programme structures vary significantly around the world. Some universities have a strict curriculum that the students have to follow with little or no choices, whereas other universities might give the students more freedom when choosing modules. In general, in most third-level education institutions, students have some choice in their modules. This might be only a few modules each year that are freely chosen as elective modules, but not only do these modules count as much towards the students' overall grade, but more importantly, these modules offer a valuable opportunity for the students to explore other interests outside of their main area of study. In a society where young people have to face the decision of their college programme earlier than ever, it is advisable to allow them to explore the entirety of their academic options and help them to find their personal path. However, oftentimes students lack the knowledge to make informed decisions about their module choices, which can lead to students picking popular modules instead of modules that suit their individual interests and strengths.

In 2016 we conducted an exploratory data analysis on the historical student data set of the Computer Science undergraduate students in UCD between 2007 and 2015 [70]. This study was based on previous research conducted at UCD [130] shortly after the introduction of the *UCD Horizons* programme (see Chapter 1). In this earlier study, the need and requirements for a recommender system for elective modules were established, a first system was developed, and an evaluation was conducted. The results were favourable, and future work included the plans for an advanced approach and further study of students' requirements.

In UCD, most students can take at least two elective modules each year. To give students the chance to broaden their horizons, elective modules can be chosen freely, independent of the students' enrolled programme across all modules offered in UCD, called *General Elective* (GE) modules. Students in UCD have a wide variety of modules to choose from, as UCD comprises six different Colleges, from Arts and Humanities to Social Sciences and Law, to Engineering and Architecture. Every college contains several schools from among the 25 schools in UCD, with each school offering multiple different Bachelors and Masters programmes. Overall, UCD offers over 1000 modules each year. Finding the most suitable module in this large amount of possibilities poses a challenge for many students, as they have to navigate the module space and ascertain that they meet the module requirements. Some UCD Schools offer elective modules that are targeted at students enrolled in programmes at the same school. These elective modules are designed to support students in deepening their programme-specific

knowledge in a certain aspect of their area of study. These elective modules are called *In-Programme Elective (IPE)* modules.

In the exploratory data analysis [70], we established the need for a recommender system for elective modules as we detected several trends in students' elective module allocations, especially regarding the distribution between GE and IPE module allocations. We concluded this study with two main findings: (i) students choose increasingly more IPE modules over time, and (ii) GE modules have a high rate of unsuccessful allocations. In the following, we will briefly outline the conclusions from the exploratory analysis.

4.1.1 Decreasing Allocations of General Elective Modules

The exploratory data analysis was conducted on the dataset of UCD BSc Computer Science students between the years of 2007 and 2015, right after the introduction of the *UCD Horizons* programme. This programme restructured the undergraduate programmes to include two mandatory elective modules each year with the goal of allowing students to explore topics beyond their main area of study. In Figure 4.1 we present the percentage of students allocated to General Elective (GE) and In-Programme Elective (IPE) modules each year. We can see that while a large majority of students chose to broaden their horizons by taking GE modules in the first years after the introduction of the programme, in the later years, a decline is visible. In 2015 nearly 60% of BSc Computer Science students chose elective modules offered as In-Programme Elective by the School of Computer Science.

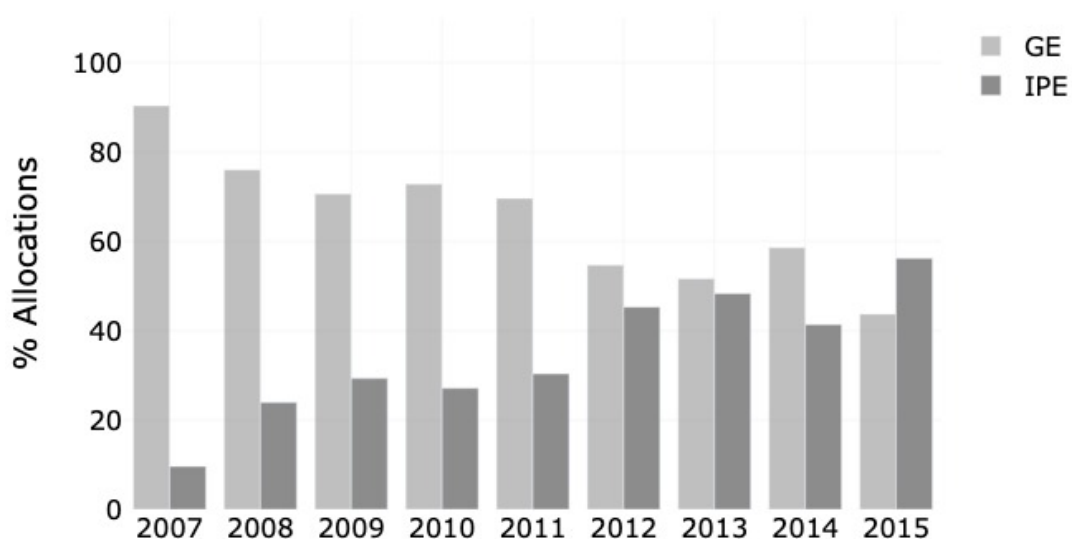


Figure 4.1: Decrease in General Elective (GE) Module Allocations.

Further, if we look at the overall number of allocations of students per elective module, we can see that a fraction of available modules receives a high number of allocations, creating a long tail of module allocation distribution seen in Figure 4.2. In this figure, we present the top 50 allocated elective modules over all years. We can see that the top 6 modules are all IPE modules (depicted in dark grey). Overall, more than 400 different modules were allocated in the nine years in our dataset, with the majority receiving less than ten allocations, further increasing this long tail.

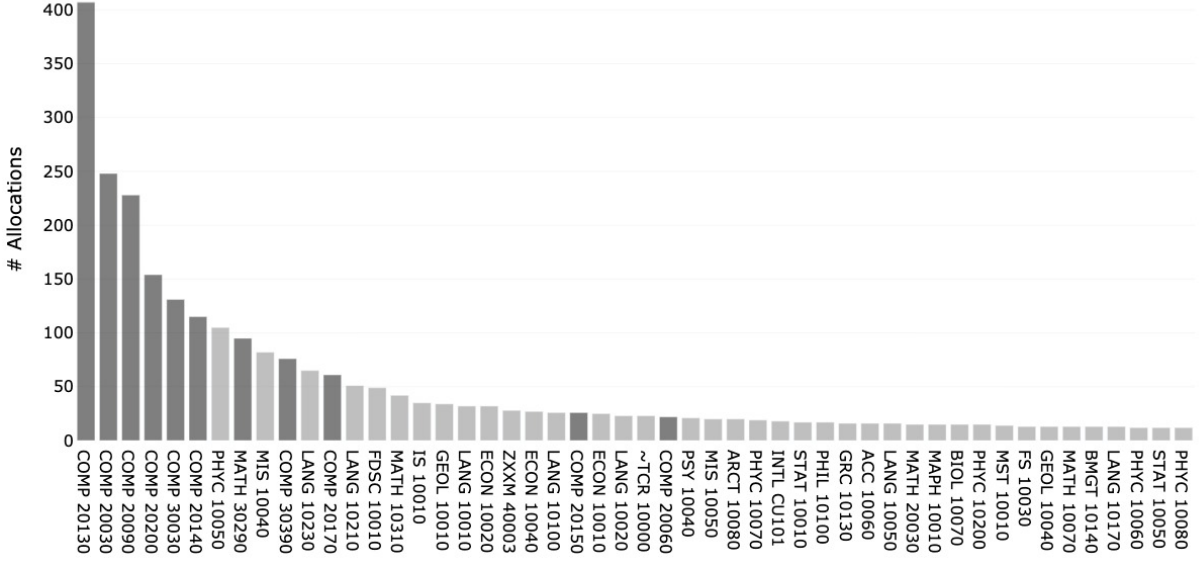


Figure 4.2: The Long Tail of Elective Module Allocations : Top 50 Elective Modules (2007 - 2015).

4.1.2 Unsuccessful Allocation Ratios

IPE modules are becoming increasingly more popular for Computer Science students, with a corresponding decrease in the popularity of GE Modules. Furthermore, GE modules tend to limit available spaces for elective students, while IPE modules do not. Therefore, we can take an in-depth look at the allocation numbers of GE modules. We calculate the *Unsuccessful Allocation Ratios* (UAR) for a GE module m in a term t as follows:

$$UAR(m, t) = \frac{\text{\#unsuccessful applications for module } m \text{ in term } t}{\text{\#total applications for module } m \text{ in term } t} \quad (4.1)$$

The UAR tell us the ratio of Computer Science students who applied for but were not allocated to their choice of elective module; for example, a module that 50 students applied to and 20 students were not allocated to would have a UAR of 0.4. The UAR

calculated takes only Computer Science students into account, as we do not have the allocation data of students enrolled in other programmes. Therefore, the UAR values only represent unsuccessful allocations for Computer Science students and do not represent the overall allocation percentages. Overall, we calculate a mean UAR over all years of 0.45 with a standard deviation of 0.18 and a median of 0.5. For example, the terms with the lowest and highest UARs occurred in 2012 and 2009, with scores of 0.41 and 0.53, respectively.

We normalise the allocation numbers by the total numbers of allocations in each year to a value between 0 and 1. We then combine the mean normalised allocation numbers $allo_n$ and the calculated UARs for each module to express an overall popularity pop of a module m as follows:

$$pop(m) = \frac{allo_n(m) + UAR(m)}{2} \quad (4.2)$$

In Figure 4.3 we present the 50 most popular GE modules ranked by the calculated popularity. We can see that modules that rate high in number of allocations, depicted in light grey, that were previously in the top 50 of allocated modules (see Figure 4.2), rate lower on the popularity scale. This shows us again that the popular GE modules receive more applications than there is availability, leading to many students having to choose their second or third elective module option.

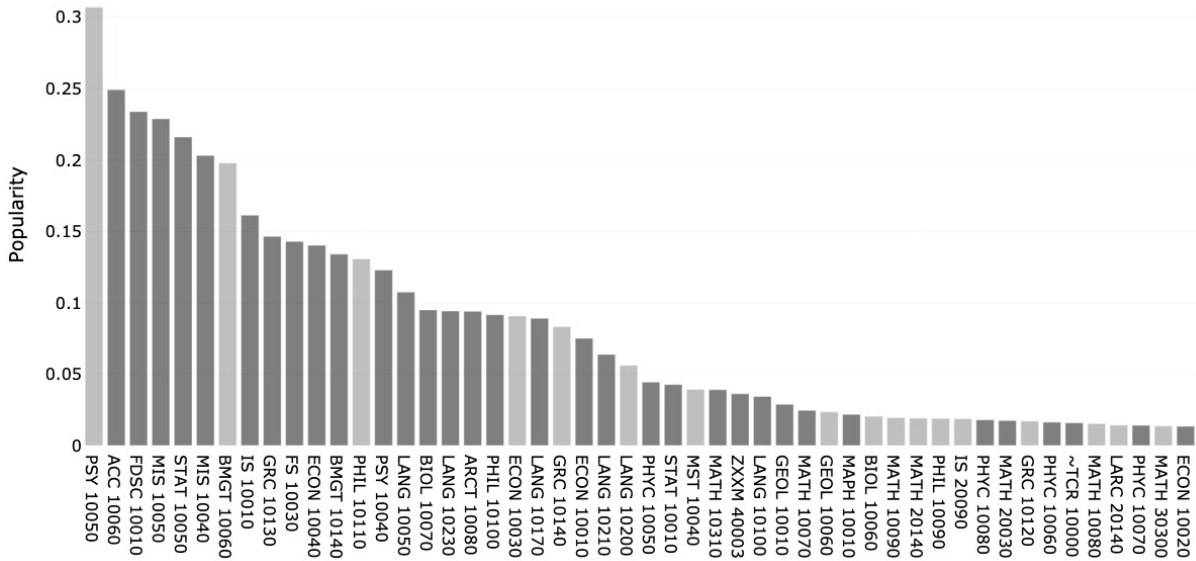


Figure 4.3: Top 50 Most Popular General Elective Modules (2007 - 2015).

4.1.3 Interim Conclusion

From these findings, we can hypothesise multiple connected problems. Firstly, we detect a decrease in students choosing to broaden their horizons by taking increasingly fewer modules outside their main area of study. We ascribe this trend in part to the low discoverability of these modules. Module recommendations seem to be spread mainly by word of mouth or through advertisement of the school². Finding modules outside of this scope is only possible through a non-personalised search engine. Furthermore, there is no advanced information available to students about choosing modules as electives, such as available spots for elective students or suitability for students from other schools. This, in turn, leads to the second problem, which is the availability of modules. The fact that GE modules have a limited amount of available spaces for elective students and the peer recommendations lead to a high possibility for students to be rejected from their first or second choice in elective modules. This further facilitates the increasing popularity of IPE modules, as these generally do not have a limit on available spaces. As a possible solution to these issues, we propose focusing on a content-based recommender system that has the ability to help students find elective modules that uniquely fit their interests. We further introduce diversity in the recommendation process to allow a more varied set of elective modules to be recommended. The following section presents how diversity has been used in the broader field of recommender systems to enhance recommendation quality.

4.2 On Recommendation Diversity, Novelty, and Serendipity

The importance of novelty, serendipity, and diversity in recommender systems has been long acknowledged in the research field [175, 190, 191, 113, 80]. In many recommender systems application areas, the idea of novelty, serendipity and diversity has been shown to produce an overall better recommendation experience for the user. However, in the field of recommender systems for educational purposes, most research has been focused on increasing recommendation accuracy. While it is undoubtedly vital to recommend modules that fit students' interests and strengths, we have seen that it is easy to create a *filter bubble*, an issue that can arise when there is a lack of diversity and serendipity in the recommendations that narrows the content of recommendations for a user over time. Especially collaborative filtering approaches have been shown to suffer from this problem [126].

²<https://www.universityobserver.ie/head-to-head-expanding-your-horizons/>

Research in different application areas has shown that introducing novelty, serendipity, and diversity into the recommender system, while minimising the impact on accuracy can improve the recommender system experience [112, 151]. For example, in [207] Zhang et al. introduce novelty into a music recommender system, inspired by principles of *serendipitous discovery*, by using a hybrid recommender approach which uses different Latent Dirichlet Allocation models. Serendipity was measured using a *Unserendipity* metric, based on distance-based novelty metrics, that measures the similarity between a user's historic data and the recommendations. The user study with 21 participants showed that recommendations with a higher rate of serendipity, albeit with slightly decreased accuracy, achieves higher satisfaction scores.

Recently, some work has been conducted introducing serendipity into course recommender systems. In [137] Pardos and Jiang present a study in which a content-based bag of words (BOW) recommender system is trialled against an RNN-based approach based on historic enrolment data. An offline and online user study showed that while the RNN models perform better in classic accuracy-based evaluation, the online evaluation results concluded that the simple BOW model was perceived as more serendipitous. It was concluded that more diverse recommendations might allow students to explore their options, but some students who focus on degree completion might prefer a more targeted approach. A hybrid model of the two was stated as a viable future work direction.

In this part of our work, our objective is twofold: (i) we aim to show that content-based recommender systems can produce suitable recommendations and (ii) introducing diversity allows us to recommend a more serendipitous set of elective modules. We introduce a hybrid module recommender system, which is aimed at helping students discover relevant elective modules while allowing them to broaden their horizons outside their main area of study. Furthermore, we allow the student to control the degree to which diversity is included in the recommendation process, aiming to better facilitate an exploration of the module space. We present our technical approach in the following section.

4.3 Module Descriptor Creation

This thesis focuses on content-based approaches to the module recommendation problem. As the main input data, we are utilising the module descriptors that are available to the students in the online module catalogue³. In this section, we briefly present

³https://hub.ucd.ie/usis/!W_HU_MENU.P_PUBLISH?p_tag=MODSEARCHALL

the creation and structure of the module descriptors and the impact on content and suitability for recommender systems.

To utilise natural language textual data for the purpose of recommendation, the text needs to be processed using text mining techniques. Text mining is a set of techniques that aims to discover interesting patterns and knowledge from text documents [181]. Text mining is widely applied in many application areas and has been shown to be beneficial using a variety of textual data [180]. However, there are significant challenges considering textual and specifically natural language documents, such as multilingualism, and specific domain knowledge [180]. Therefore, we need to consider the suitability of module descriptors for text mining approaches.

The module descriptor is grouped into eight sections with additional subsections, see Table 4.1 and Figure 4.4. The sections in the descriptor cover details from learning outcomes to assessment details. However, only the description and the learning outcome section are free text, with all other sections providing information in lists or tables. The listed information, while suitable for other tasks, such as constraint satisfaction problems, are less applicable for text mining approaches, as they are very short (1 to 2 words per item) and are sourced from a set of options (e.g. in-class test, homework, presentation for assessment strategies). Therefore we focus on the free texts available in the description and learning outcomes sections.

The description section aims to provide an overview of the module contents and is often written as a short paragraph, while the learning outcomes section lists skills and knowledge a student will acquire from taking this module. This section is often prefaced with the sentence *"On completion of this module students should"* and is followed by a bullet point style listing.

There are some implications of using module descriptors for a module recommender system. The module descriptors are provided by the lecturer of a given module. While lecturers are instructed to provide module descriptors that are complete, detailed and up-to-date, we can see a substantial difference in the quality of the module descriptors. Further, module descriptions might have a large overlap of repeating words, that are used to describe non-descriptive elements, such as *students*, *tutorial*, *learn*, *module*. As those words do not describe the content of a module a thorough cleaning and pre-processing of the descriptors is needed before applying text mining approaches. In Chapter 6.2.1.1 we present an analysis of the average length and quality of the module descriptors.

Section	Subsection	Content	Format
Description	-	Overall Description of module contents	free text
What will I learn?	Learning Outcomes	Defines module contents in detail and learning outcomes	free text
How will I learn?	Student Effort Hours; Approaches to Teaching and Learning	Breakdown of Student Effort Type (i.e. Lectures, Small Group, Tutorials)	table
Am I eligible to take this module?	Requirements, Exclusions and Recommendations; Module Requisites and Incompatibles	List modules that are required, incompatible, equivalent	list
How will I be assessed?	Assessment Strategy	Lists all assessments	list
What happens if I fail?	-	Lists resit options	list
Assessment feedback	Feedback strategies; How will my Feedback be Delivered?	Lists Feedback strategies	list
When is this module offered?	-	Lists timetable information	list

Table 4.1: Module Descriptor Sections, Content and Form

4.4 Module Recommendation Approaches

In this section, we describe the proposed hybrid content-based and baseline collaborative approaches to elective module recommendation. To begin, the following notation is introduced. Let S and M denote the set of students and modules, respectively. Each student, $s_i \in S$, is profiled by a subset of their previously taken modules. For example, these modules may either be core to the student's programme of study, optional modules from within their programme, elective modules, or combinations of these. Let P_i denote the profile of student s_i , where $P_i = \{m_1, m_2, \dots, m_l\}$ and $m_j \in M$ denotes a particular module taken by student s_i . Based on the modules in the profile, a ranked list of top- N elective module recommendations is generated for a student s_i by the hybrid recommender system as described below.

4.4.1 Hybrid Recommender

The proposed hybrid recommender consists of two components to rank candidate elective module recommendations as follows. The first component prioritises candidate modules that are similar in content to those in the student's profile (*deepen*); for this purpose, a traditional content-based recommender is used. The second component prioritises candidates from outside the student's programme area (*broaden*); in this case, hierarchical taxonomy of the available programmes of study and associated modules is created, and candidates who are least related to those in the student's profile are recommended. These components and how they can be combined to produce a single ranked list of recommendations are described in the following sections.

4.4.1.1 Content-Based Recommender

In UCD, each module has an accompanying module descriptor which provides a textual description of the aims of the module, the topics covered, and the learning outcomes, see Figure 4.4. Thus, modules can be viewed as documents made up of the set of terms contained in their descriptors. Using the Vector Space Model (VSM), [162], a vector represents each module in an n -dimensional space, where each dimension corresponds to a term from the overall set of terms in the module collection. Standard document preprocessing is performed, such as converting all text to lowercase, tokenisation, stop-word removal, and stemming [147].

COMP40320 Recommender Systems

Academic Year 2016/2017

Recommendation technologies have become an important part of our online experiences, helping us to discover books, movies, and music that are relevant to our likes and preferences. So much so, in fact, that recommender systems are now a fundamental component of most ecommerce platforms, streaming services, and other content sites. At their core recommender systems operate by learning about the likes and dislikes of individuals and groups of users so that they may proactively tailor content for these users.

In this course we will cover the fundamentals of recommender systems technologies including the main approaches to building and evaluating recommender systems (content-based vs collaborative filtering vs hybrid approaches) as well as a variety of more advanced topics, from generating diverse and novel recommendations to explaining recommendations to coping with malicious users.

This module will be assessed by continuous assessment only which will take the form of a number of practical projects and reports related to the development of recommender systems technologies.

Please note that proficiency in the Java Programming Language is required.


Show/hide content

+ Open All

Curricular information is subject to change

What will I learn?	+
How will I learn?	+
Am I eligible to take this module?	+
How will I be assessed?	+
What happens if I fail?	+

Recommender Systems (COMP40320)

SUBJECT:	Computer Science
COLLEGE:	Science
SCHOOL:	Computer Science
LEVEL:	4 (Masters)
CREDITS:	10.0
SEMESTER:	Semester Two
MODULE COORDINATOR:	Dr Michael O'Mahony
MODE OF DELIVERY:	N/A
HOW WILL I BE GRADED?	40% 

 Print Page

(Google Chrome is recommended when printing this page)

Figure 4.4: UCD Module Catalogue: Module Descriptor.

Let $T = \{t_1, t_2, \dots, t_n\}$ denote the set of terms in the collection. Formally, each module $m_j \in M$ is represented as a vector of term weights, where each weight indicates the degree of association between the module and the corresponding term:

$$m_j = \{w_1^j, w_2^j, \dots, w_n^j\}, \quad (4.3)$$

where w_k^j is the weight of term t_k for module m_j .

For term weighting, we employ Term Frequency-Inverse Document Frequency (TF-IDF) [161], a commonly used scheme in information retrieval. This statistical method creates a weight for every word in a given document and is based on its frequency in the given document based on its importance in the whole set of documents [161]. TF-IDF relies on two hypotheses: (i) words that frequently occur in a document are descriptive of that document's content (TF), and (ii) words that occur in many documents are less important to a specific document (IDF). Other techniques such as part-of-speech-Tagging [150], the process of marking words corresponding to their part of speech (such as adjective or noun), can be used alternatively or additionally.

We calculate the normalised term frequency for a term t_i in a document d_j as follows:

$$nTF(t_i, d_j) = \frac{f(t_j, d_j)}{\max\{f(w, d_j) : w \in d_j\}} \quad (4.4)$$

where $f(t_i, d_j)$ denotes the raw count of the term t_j in the document d_j and $\max\{f(w, d_j) : w \in d_j\}$ denotes the maximum term frequency in d_j . By normalising the term frequency we can offset the bias of longer documents. We then calculate the inverse document frequency (IDF) to reduce the weights of each term t_i that appear in many documents, as follows:

$$IDF(t_i, D) = \log\left(\frac{|D|}{|\{d \in D : t_i \in d\}|}\right) \quad (4.5)$$

where d denotes a document in the set of documents D and $|D|$ denotes the total number of documents in a document.

Given the vector space representation of modules, the similarity between two modules, m_i and m_j , is computed using cosine similarity [161], see Figure 4.5. The rank score of a candidate elective module, m_c , for student s_i is calculated as the mean cosine similarity between m_c and each of the modules in the student's profile, P_i , as follows:

$$\text{score}_{CB}(s_i, m_c) = \frac{1}{|P_i|} \sum_{m_j \in P_i} \text{sim}(m_c, m_j) \quad (4.6)$$

Candidates with higher scores are ranked higher in the recommendation list. Thus, the content-based recommender facilitates students to deepen their learning by suggesting elective modules similar to those in the student's profile.

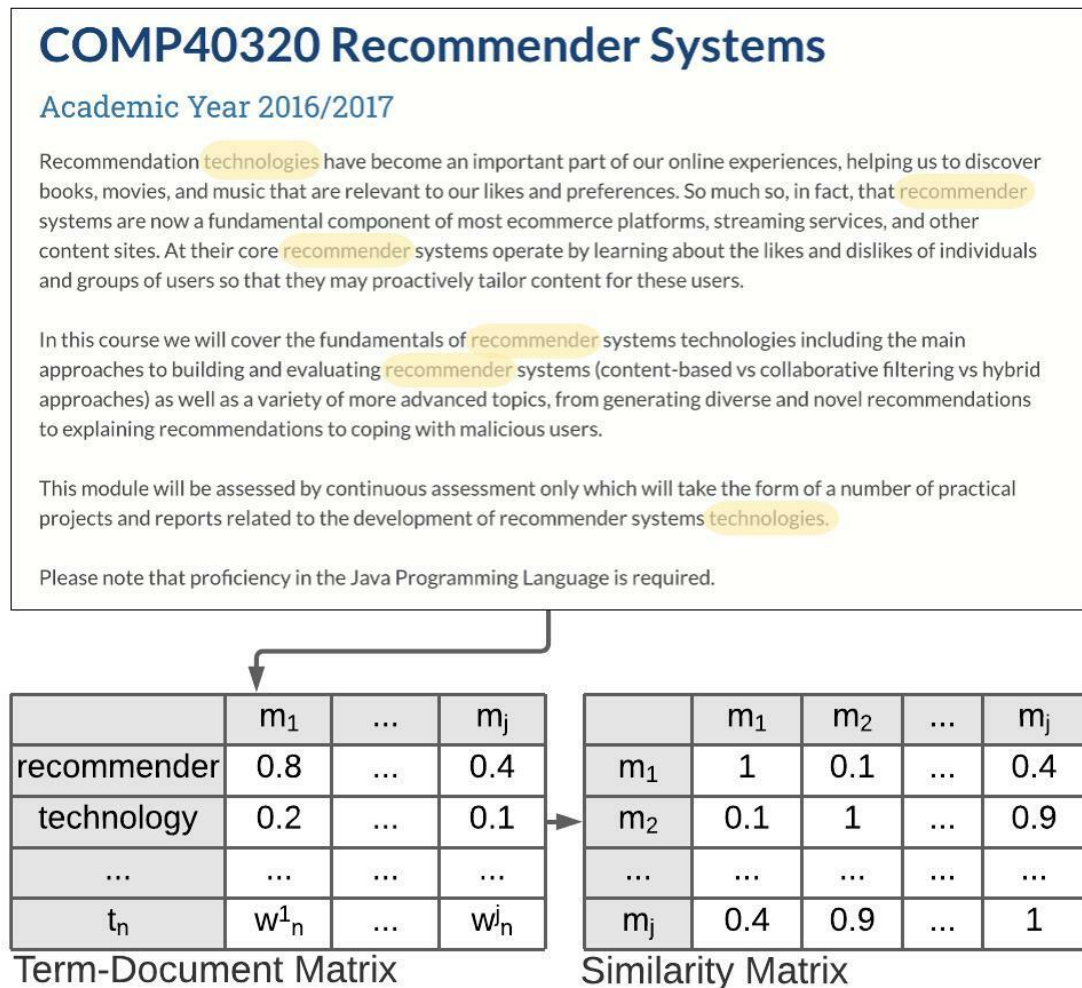


Figure 4.5: Content-based Similarity Example on Textual Descriptor.

4.4.1.2 Taxonomy-Based Recommender

In order to recommend modules to students from outside their programme of study, an approach based on a hierarchical taxonomy of the academic structure of UCD is used. Briefly, there are six Colleges in UCD, each with several constituent Schools. Each School offers many programmes of study, and each module is associated with one or more of these programmes; see Figure 4.6.

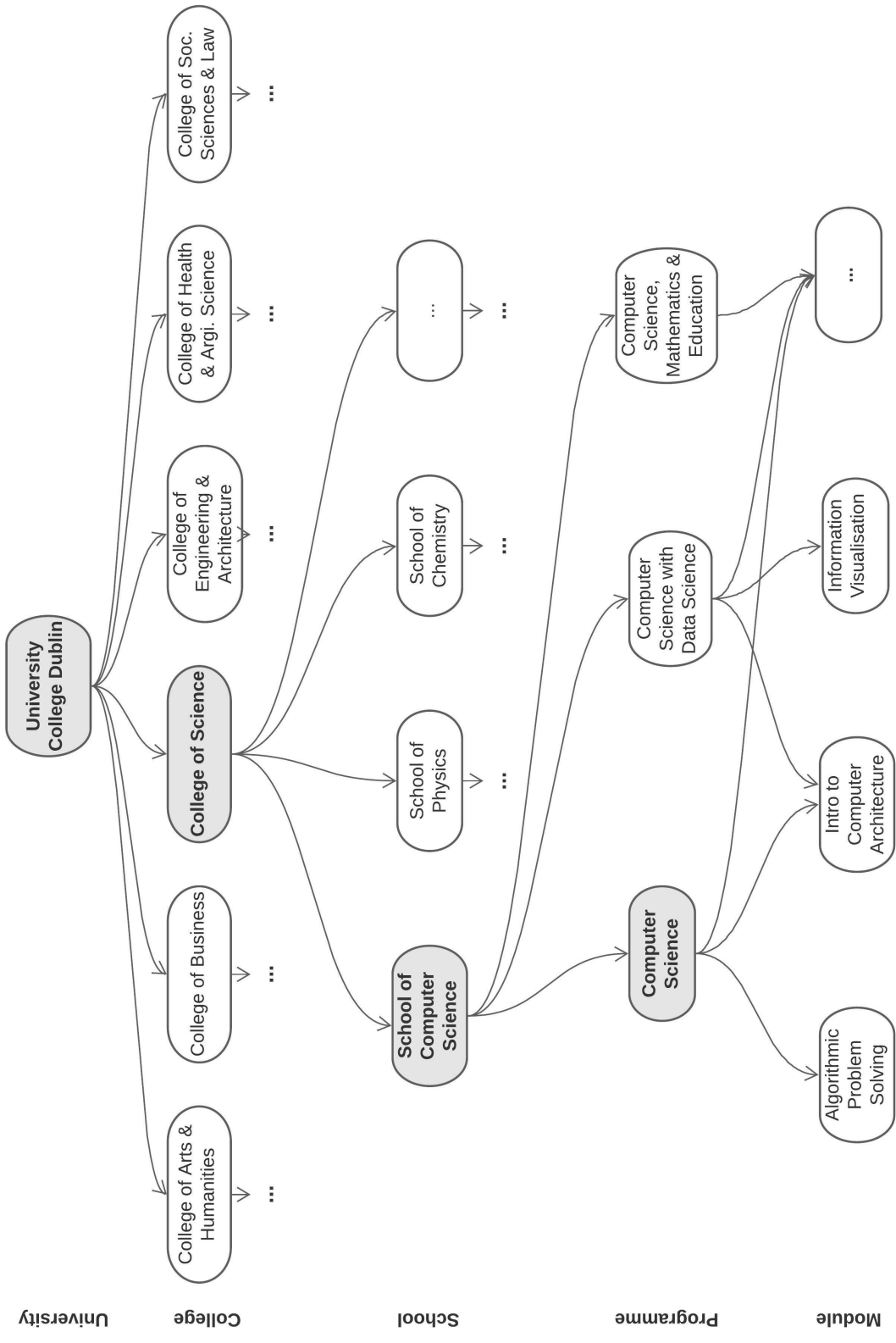


Figure 4.6: Taxonomy of Colleges, Programmes and Modules in University College Dublin.

While more sophisticated approaches are possible, here we make the general assumption that modules from the same programme are more closely related than those from different programmes. The following approach is used to calculate the rank score of a candidate elective module m_c for a given student s_i with profile P_i :

$$\text{score}_{TB}(s_i, m_c) = \frac{1}{|P_i|} \sum_{m_j \in P_i} \text{rel}(m_c, m_j) , \quad (4.7)$$

where $\text{rel}(m_c, m_j)$ is 0 if both modules belong to the same programme; 0.33 if the modules are from different programmes offered by the same School; 0.66 if the modules are offered by different Schools in the same College; and 1 if the modules are from programmes offered by Schools in different Colleges, see Equation 4.8. Using this approach, higher scores are assigned to candidate elective module recommendations which are less related (based on academic structure) to those in the student's profile, thereby facilitating the student to choose elective modules to broaden their learning experience. Thus, for example, the score for the module *Algorithmic Problem Solving* and *Intro to Computer Architecture* would be 0 as they are both core modules in the BSc Computer Science, whereas the score of *Algorithmic Problem Solving* and *Information Visualisation* would be 0.33 as they are core modules from different programmes within the School of Computer Science, see Figure 4.6.

$$\text{rel}(m_c, m_j) = \begin{cases} 0 & \text{if both modules in the same programme} \\ 0.33 & \text{if modules from different programmes but same School} \\ 0.66 & \text{if modules offered by different Schools in the same College} \\ 1 & \text{if modules from programmes in different Colleges} \end{cases} \quad (4.8)$$

4.4.1.3 Hybrid Recommendation Ranking

The above provides two alternatives with which to recommend elective modules, with candidates ranked in descending order of their content-based and taxonomy-based scores. The former prioritises candidates who are similar to a student's profile, while the latter prioritises candidates who are least related to a student's core programme of study. These approaches can be combined to allow students to better explore the wide range of elective module choices available from across the university's diverse subject

offerings. An overall score for a candidate elective module m_j is calculated for student s_i as follows:

$$\text{score}(s_i, m_c) = \alpha \text{score}_{CB}(s_i, m_c) + (1 - \alpha) \text{score}_{TB}(s_i, m_c) , \quad (4.9)$$

where the parameter α can be varied to influence the recommendation of elective modules depending on whether a student wishes to deepen or broaden their learning experience. By increasing the diversity we can enhance the novelty and serendipity of the recommended elective modules.

The overall system architecture of our hybrid recommender approach is visualised in Figure 4.7.

4.4.2 Collaborative Recommender

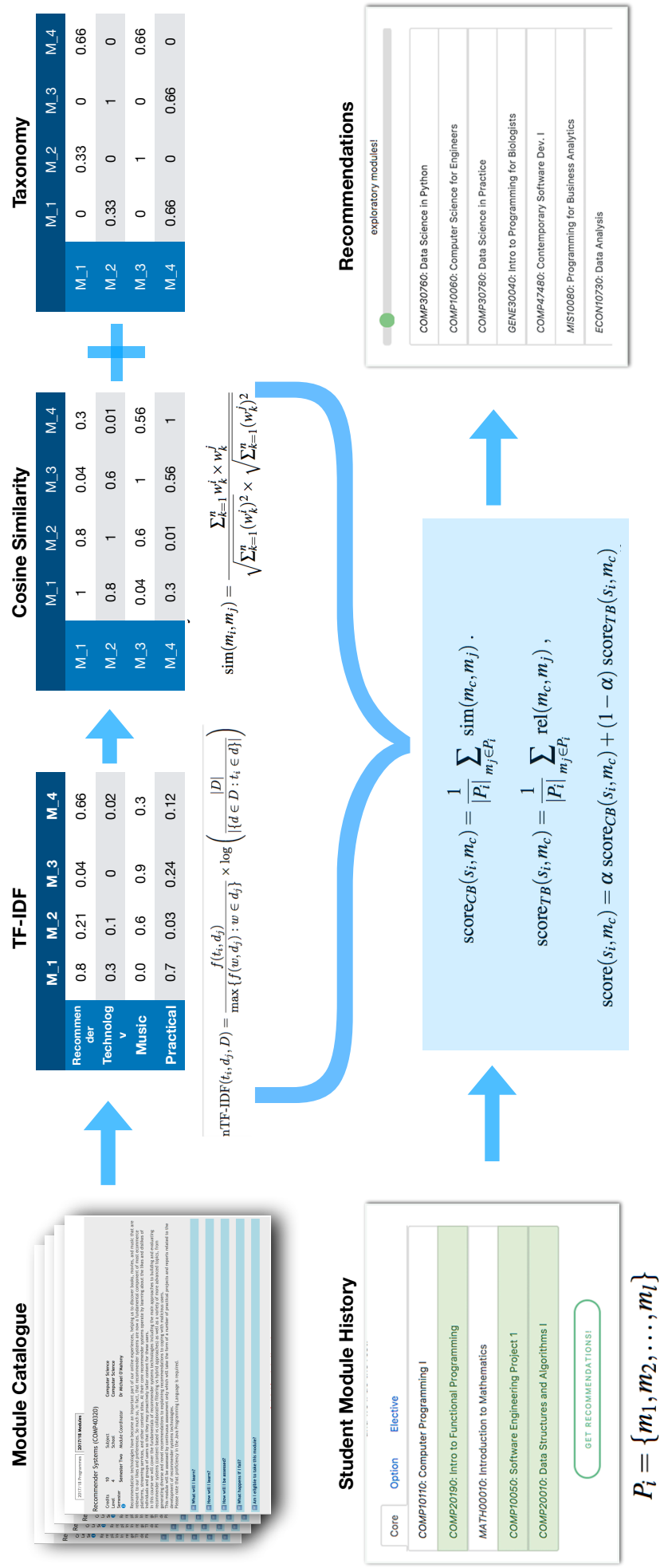
Separately, we also consider a traditional neighbourhood-based collaborative filtering approach to elective module recommendation [169]. As before, each student s_i is profiled by a subset of the modules, P_i , which they have previously taken. Then, the neighbourhood for a given student s_i is determined based on profile similarity, where the similarity between two profiles, P_i and P_j , is calculated using the overlap coefficient [193]:

$$\text{sim}(P_i, P_j) = \frac{|P_i \cap P_j|}{\min(|P_i|, |P_j|)} . \quad (4.10)$$

Once the k most similar students (neighbours) to student s_i are identified, a top- N list of elective module recommendations, ranked by their frequency of occurrence in neighbour profiles, is returned to the student. Using this approach, the elective modules popular among students with similar profiles are recommended, depending on the distribution of elective modules in neighbour profiles; these can be from within or outside the core programme area of student s_i .

4.5 Evaluation

This part of our research aims to establish if content-based recommender system approaches can produce suitable elective module recommendations. We are interested in how a content-based approach compares to a collaborative filtering approach concerning diversity. Further, we evaluate it using a taxonomy of university programme structures as a means to introduce diversity increases the scope of recommended electives.



4.5.1 Dataset, Methodology & Metrics

We randomly selected 100 Computer Science students that have completed their second year as our test set. An average of 20 core modules represents each student. We select 3 of these modules randomly as the input for our recommender system to simulate a student's input in the web application.

We conduct a leave-one-out test [163] and generate a top-10 recommendation set for each student for each recommendation approach: a pure content-based approach ($\alpha = 1$), three hybrid approaches ($\alpha = [0.25, 0.5, 0.75]$), and the collaborative filtering method (*CF*).

To evaluate the offline results, we are not using a classic accuracy score as we hypothesise that our ground truth, that is, the set of elective modules taken by students, is skewed due to the reasons explained above (i.e. students primarily following peer recommendations or simply choosing popular modules). One of our main objectives is to broaden the range of modules that students are aware of. Hence, we evaluate our results by comparing the number of distinct modules recommended overall users and the number of distinct subjects covered by these recommendations. To evaluate relevance, we use *sim-to-core* (*StC*), a metric that determines the average similarity of the most similar module in the student's profile, P_i , to each module in the recommendation set, R_i , as shown in Equation 4.11:

$$StC(P_i, R_i) = \frac{\sum_{m_k \in R_i} sim_{max}(m_k, P_i)}{|R_i|}, \quad (4.11)$$

where $R_i = \{m_1, \dots, m_r\}$ is the set of elective module recommendations and $sim_{max}(m_k, P_i)$ returns the maximum similarity between the recommended elective module m_k and the modules in the student's profile.

4.5.2 Results

Firstly, we consider the overlap coefficient [193] of the recommendation sets produced by the various approaches (Table 4.2). We calculate the overlap over the top-10 recommendations. As expected, as more diversity is introduced into the recommendation process (i.e. as α is decreased), a decrease in the overlap between the recommended sets is observed. For example, an overlap of 76.5% in recommended sets is seen between the pure content-based recommender ($\alpha = 1$) and the hybrid approach with $\alpha = 0.5$. Comparing the recommendations made by the collaborative filtering approach, we see approximately 3% of the same modules being recommended; since this

approach operates over the limited set of largely popular modules actually selected by students, this result is expected.

α	1	0.75	0.5	0.25	CF
1	1.000	0.901	0.765	0.626	0.031
0.75		1.000	0.862	0.724	0.032
0.5			1.000	0.860	0.033
0.25				1.000	0.034
CF					1.000

Table 4.2: Overlap of the Recommended Module Sets by the Different Approaches.

Table 4.3 shows that there is a gradual increase in both the number of distinct modules (D. Mod.) recommended and the number of distinct subjects covered (D. Sub.) as diversity is introduced (i.e. as α decreases). While the number of distinct modules varies and only increases marginally, we can detect an increase of distinct subjects with decreasing *alpha* value, as the approach recommends increasingly more diverse sets of modules. Moreover, the percentage of in-programme (Computer Science) modules (% IPE) recommended also reduces, while the reduction in the *sim-to-core* (*StC*) metric is less pronounced.

α	D. Mod.	D. Sub.	% IPE	<i>StC</i>
1	149	31	24.1	0.012
0.75	156	34	21.1	0.011
0.5	157	37	17.9	0.009
0.25	154	44	12.3	0.008
CF	60	28	26.7	0.002

Table 4.3: Evaluation Results for the Different Approaches.

The results also show that the collaborative filtering approach produces recommendations with the lowest number of distinct modules and subjects covered, while the percentage of IPE modules recommended is the highest. Thus, it can be seen that the hybrid approach can successfully improve recommendation diversity without significantly compromising relevance, while the collaborative filtering approach recommends from a relatively small set of modules.

4.5.3 Discussion

The results show that the content-based approach performs better than the collaborative filtering technique based on our metrics. Using a content-based approach, we can

significantly increase the number of modules and subjects recommended to the students. After introducing diversity into the recommender system, we can increase the number of recommended general electives without significantly decreasing the similarity to the student profile. The collaborative filtering approach performs as expected and tends to recommend from a relatively small set of popular modules that are not very similar to the students' selected modules in their profile.

4.6 Initial Prototype Design

We implemented a *UCD Module Advisor* prototype that demonstrates the presented hybrid recommender system approach. The web application offers two main functionalities. Firstly, a personalised recommender system where students can choose modules from their module history and receive elective module recommendations based on their choices. Secondly, we build a search engine that allows the students to search for modules by module code, title or keyword and allows for the addition of conversational constraints, such as module-level or day of the lecture. The students further receive information about the modules and are offered similar modules as alternatives.

The prototype was implemented using `Flask`⁴.

4.6.1 Recommender System

The personalised recommender system where students can choose modules from their module history and receive elective module recommendations based on their choice is presented in Figure 4.8. On the left-hand side, the students are presented with their module history. From this list, the students can select any number of modules (core, option or elective) that they have previously enjoyed or are interested in general. The selected modules will appear in the middle section of the screen. The modules can then be removed again by clicking on them in the middle section. After every interaction (adding or removing a module), the system will present elective module recommendations on the right-hand side of the screen, based on the current selection of modules. The recommendations are calculated using the hybrid content-based approach as described in Section 4.4.

Below the left-hand column, the students are presented with a *Discovery* slider. By changing the slider's value, the elective module recommendations below will automatically recalculate based on the set value. The slider determines the α value in our

⁴<https://flask.palletsprojects.com/en/1.1.x/>

Search
Search for modules

Recommender System
Get module recommendations

Info
More about this project

Welcome to the Module Advisor

Select Modules

Please select your favourite modules!
Can't find a module? Click here to search

COMP10040 : INTRODUCTION TO COMPUTER ARCHITECTURE

COMP10070 : FORMAL FOUNDATIONS

COMP10110 : COMPUTER PROGRAMMING I

COMP10130 : COMPUTER SCIENCE IN PRACTICE

COMP10050 : SOFTWARE ENGINEERING PROJECT 1

COMP10120 : COMPUTER PROGRAMMING II

MATH10210 : FOUNDATIONS OF MATHEMATICS

Feeling adventurous? Use the *Discovery* slider to see more exploratory modules!

Chosen Modules

These are the modules you have chosen so far.
Click to delete!

COMP30490 : COLLECTIVE INTELLIGENCE

COMP30030 : INTRODUCTION TO ARTIFICIAL INTELLIGENCE

COMP47490 : MACHINE LEARNING

Recommended Electives

Intrigued?
Click for more information or save to favourites!

COMP30120 : INTRODUCTION TO MACHINE LEARNING

PSY30340 : HUMAN INTELLIGENCE AND PERSONALITY

COMP30230 : CONNECTIONIST COMPUTING

COMP30260 : AI FOR GAMES AND PUZZLES

COMP20040 : DATA STRUCTURES AND ALGORITHMS II

COMP40580 : NATURAL COMPUTING

COMP40000 : APPLICATIONS FOR ROBOTICS

Figure 4.8: Screenshot of the Recommender System Prototype.

system as described in Section 4.4.1.3. This allows the students to gradually explore modules outside of their field of study and broaden their horizons about available modules in different areas. The slider allows the students to dictate the amount of diversity and acts as a natural explanation for the recommended modules.

On the right-hand side of the recommender system prototype, we present the students with the ranked elective module recommendations. By clicking on a recommended module, the student is brought to the module information page, which contains additional information about the module selected, such as the description, learning outcomes, lecturer and ten similar modules are presented (calculated as described in Section 4.4.1.1). A link is also provided to the official UCD module catalogue for further information. In previous studies, we included a variety of information, such as a *popularity* ranking and an indicator of the difficulty of a given module (e.g. fail/pass rates). However, we decided in discussion with the School of Computer Science not to include such information based on module-specific data and historical data due to privacy concerns. We, therefore, focus on information that is freely available through the UCD website when presenting recommended modules.

In the example shown in Figure 4.8, the student has chosen three computer science modules related to Artificial Intelligence and Machine Learning. The *Discovery* slider is set to approximately 0.3, thereby introducing relatively low diversity into the recommendation process. We can see that most modules in the recommendation list are Computer Science modules closely related to the selected modules. By increasing the discovery value, the student can introduce a higher diversity into the recommendation process. With a lower alpha/higher discovery value, modules from diverse subjects such as Psychology, Engineering and even Forestry, which are still related to the selected modules, can be recommended; see Example in Figure 4.9.

4.6.2 Search Engine

When it comes to finding elective modules, students at UCD mainly have to rely on the UCD website to find interesting modules. The website provides a search functionality, presented in Figure 4.10. The module search functionality provided by UCD allows the students to search by subject, School, or keyword and filter by their level. While the design has been changed in current years, the functionalities seem to have been unchanged.

However, the current UCD module search engine, as implemented, only takes module titles into account. We can imagine a Computer Science student in their second year looking for an elective module. This particular student might be interested in improv-

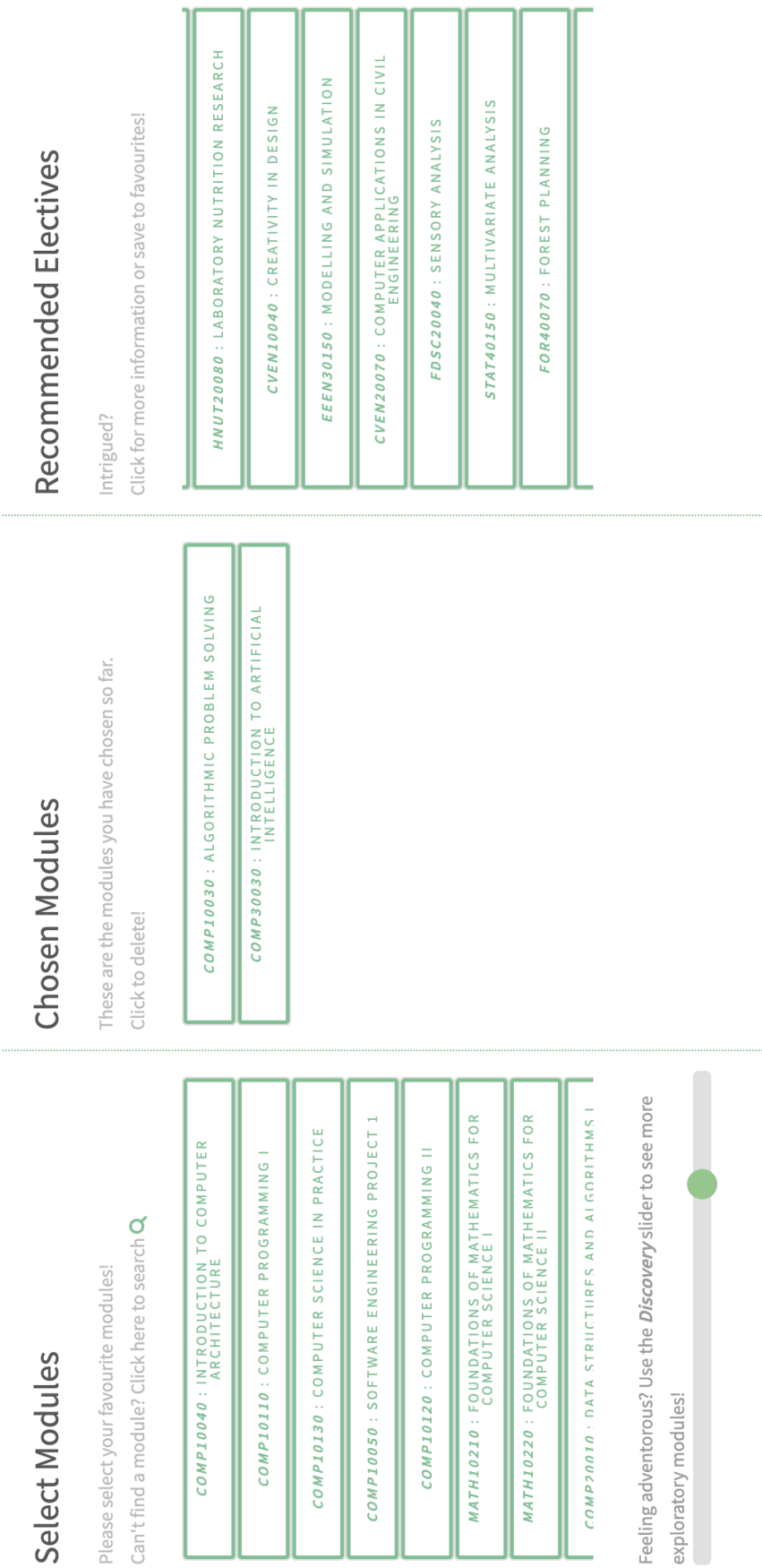


Figure 4.9: Recommender System Prototype: Results with High Discovery.

Figure 4.10: UCD Module Search Engine (2017).

ing their Java programming skills. A search on the UCD module search engine for the keyword *Java* yields two results, i.e. *COMP20250 - Introduction to Java* and *COMP41200 - Professional Java Programming*. This result will not provide enough information for the student, as *COMP20250 - Introduction to Java* is a core module that the student will have to take, and the student is not eligible to take the level 4 *COMP41200 - Professional Java Programming*. Moreover, if the student were to search for *java programming* or *java language* the UCD module search engine would have yielded no results.

In our search engine, we combine multiple advanced search functions. Firstly, we include all textual descriptor parts in our search index, such as the module description, learning outcomes and assessment details. This allows the student to generate more detailed searches, e.g. *java level 2* to find all modules in level 2 that are related to Java, or *java level 2 project* which will find modules related to Java in level 2 that use a project as assessment. We can see the outcome of the search for *java* in our search engine prototype in Figure 4.11. We can see that our search engine can find nine modules in total.

Secondly, additionally to our direct search, we implement a fuzzy string matching approach using the *Levenshtein Distance* [204]. This metric measures the distance between two sequences of words by calculating the minimum number of edits (i.e. insertions, deletions, or substitutions) needed to change a sequence into the other. The Levenshtein Distance is defined as follows:

$$lev_{a,b}(i, j) = \begin{cases} \max(i, j) & \text{if } \min(i, j) = 0, \\ \min \begin{cases} lev_{a,b}(i-1, j) + 1 \\ lev_{a,b}(i, j-1) + 1 \\ lev_{a,b}(i-1, j-1) + 1_{a_i \neq b_j} \end{cases} & \text{otherwise.} \end{cases} \quad (4.12)$$



Here's what we found...

- COMP20250 : INTRODUCTION TO JAVA
- COMP41200 : PROFESSIONAL JAVA PROGRAMMING
 - COMP20030 : WEB DESIGN
- COMP30330 : COMPILER CONSTRUCTION
- COMP30490 : COLLECTIVE INTELLIGENCE
- COMP47480 : CONTEMPORARY SOFTWARE DEVELOPMENT
- COMP30300 : OBJECT ORIENTED PROGRAMMING
- COMP30310 : WEB-BASED INFORMATION SYSTEMS
- COMP30160 : OBJECT-ORIENTED DESIGN

Figure 4.11: Screenshot of the Recommender System Prototype - Search Engine: Results for the Search Query *java*.

Where a and b are two strings, $1_{a_i \neq b_j}$ denotes 0 if $a = b$ and 1 otherwise. This allows us to approximately match strings based on their distance, measured by the number of edits needed to make them equal. The benefit of this approach is that we are able to deal appropriately with any query containing typographical errors.

Suppose the students click on a module in the search results. In that case, we provide the student with the module description, the link to the UCD module catalogue for further information about the module, as well as the ten most similar modules, calculated based on the content-based recommender system, without any diversity, presented in Section 4.4.1.3, see Figure 4.12.

In our example, the student might be interested in the Collective Intelligence module, as it is a level three module and would allow him to use his Java skills in a new application area.

4.7 Conclusion

Helping students find suitable elective modules from many available choices is a task precisely suited to a recommender system. This prototype represents the first step in a solution to support students in making informed decisions in their academic careers.

We conclude that there is a legitimate need for a recommender system in the area of elective module recommendations. We have shown that module descriptions can be used to make meaningful recommendations. In contrast, while collaborative filtering approaches will give accurate results in a traditional sense, it will not help the problem of discoverability of modules as it promotes primarily already popular modules. We have shown that the proposed hybrid recommender system can add diversity to the set of recommendations. While the taxonomy-based recommender represents the first step, nonetheless, it is capable of facilitating the discoverability of modules outside of the students' core areas of study.

We developed a prototype of a web-based *UCD Module Advisor* prototype that comprises multiple functionalities to support students in gaining knowledge about their modules and elective module opportunities. The proposed search engine provides greater functionality than that currently in place. The elective recommender system of the prototype implements a user interface for the hybrid content-based approach we presented in this study. The students can choose the degree of diversity themselves by changing the *Discovery* slider. This allows the students to explore the module space in a self-directed way and act as a natural explanation for the recommended elective modules.



COMP30490 - Collective Intelligence

Module Description

While there has been significant progress in the area of Artificial Intelligence in recent times, the development of fully automated and robust algorithms to solve a range of complex problems remains a challenge. Collective intelligence provides an alternative approach to conventional AI techniques, where the combined efforts of humans and machines can be leveraged using the power of the Web to solve problems that are currently beyond the reach of modern AI. This module will explore this science of collective intelligence, providing concrete examples of how some of the most complex problems can be successfully solved using collective intelligence. The module will begin by considering recommender systems, which discover relevant content by learning the particular preferences of individuals and groups of users. A variety of recommender system approaches will be considered – ranging from collaborative techniques that leverage collective intelligence to more traditional approaches that are based on content descriptions. The module will also examine a variety of other problems that are amenable to collective intelligence – for example, understanding images, reading text, translating speech, recognising relevant information, answering questions, predicting future events, etc. The module will also consider how the business of the Web has adapted to take advantage of collective intelligence with special attention paid to some of the emerging business models that have developed as a result. This module will be assessed by continuous assessment only which will take the form of a number of practical projects and reports related to recommender systems and collective intelligence. Please note that proficiency in the Java Programming Language is required

[MORE INFORMATION!](#)

These Modules could be interesting for you!

ARCT30040 : ARCHITECTURAL DESIGN VI

PSY30340 : HUMAN INTELLIGENCE AND PERSONALITY

MIS30040 : ANALYTICS MODELLING

LAW30460 : INDUSTRIAL RELATIONS LAW

COMP10020 : INTRODUCTION TO PROGRAMMING II

COMP30030 : INTRODUCTION TO ARTIFICIAL INTELLIGENCE

COMP20250 : INTRODUCTION TO JAVA

COMP30260 : AI FOR GAMES AND PUZZLES

MATH20210 : FUNDAMENTALS OF ACTUARIAL AND FINANCIAL MATHEMATICS II

PHYC40430 : NANOMECHANICS - FROM SINGLE MOLECULES TO SINGLE CELLS

Figure 4.12: Screenshot of the Recommender System Prototype - Search Engine: Module Information and Similar Module Recommendations.

The results presented in this chapter support Hypothesis 1 in the sense that we have shown that a content-based recommender system is able to provide suitable elective module recommendations while also increasing diversity compared to a collaborative filtering approach. In addition, the implemented prototype provided helpful insight regarding the space of available modules from which elective modules are selected. We conclude that while our prototype allows exploration by actively choosing modules and adjusting the *Discovery* slider, students might benefit from additional guidance and explanations regarding the recommendations being presented to them. For example, the relationships and dependencies between modules in subsequent years of degree programmes are certainly a key consideration for students when choosing elective or optional modules at any given point in their studies. For this, we explore how we can detect and visualise connections and dependencies between modules. The following chapter focuses on this sequential nature of modules, discovering module connections using matrix factorisation and exploring different use cases. Finally, we refine our prototype by presenting visualisation options that allow students to explore their module space and the connections between modules interactively.

MODULE DEPENDENCY DETECTION USING NON-NEGATIVE MATRIX FACTORISATION

This part of the research focuses on the often sequential nature of undergraduate degrees and the relations and dependencies that exist between modules. Undergraduate programmes are structured into levels (or years). While explicit pre- and co-requisites are typically defined for certain modules in programmes, these formal requirements often do not capture all the dependencies that exist between modules, especially optional and elective modules that may be taken during a programme of study. Looking at the modules within the UCD BSc Computer Science programme, we can see some clear connections. For example, it is evident that *Computer Programming I* will lay the foundations for *Computer Programming II*. However, the optional second-year module *UNIX Programming* does not have any explicit requirements stated; however, as it is taught using the programming language C, a connection to *Computer Programming I* (which introduces the foundation of programming in C) should be noted, see Figure 5.1. While this dependency is obvious if one researches modules in detail, students can easily miss these implicit dependencies when planning their curriculum. This is only exacerbated when choosing elective modules outside of the student's core area of study, as the students will be less familiar with the various explicit requirements or implicit dependencies of other programmes. Therefore, it is challenging to determine if a module is suitable and, aside from any formally stated requirements, whether specific prior knowledge would be useful in taking a module.

In this part of our work, we focus on dependencies between modules in undergraduate programmes that go beyond explicit pre- and co-requisites. We use matrix factorisation to determine latent factors and use similarity to present dependencies between modules in subsequent levels. Our approach uses non-negative matrix factorisation

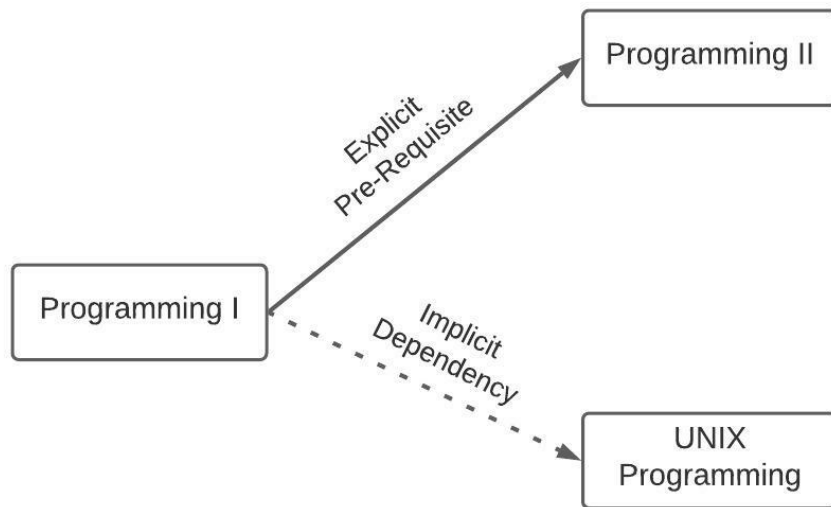


Figure 5.1: Example of Explicit and Implicit Connections between Modules.

(NMF) [95] to detect latent topics within the textual module descriptors. This allows us to build a representative clustering of module affiliations and build a rich, connected model of the module space, where modules can be grouped into sets of modules with shared topics. Furthermore, this information can be used to detect implicit dependencies between modules in addition to those captured in formal pre- and co-requisites, as well as additional information that allows us to create graph-based visualisations of dependencies in the module space. We can harvest this information for multiple use cases, from guiding students to select their most suitable personal curriculum to help students better understand the importance of modules in the broader context of their area of study.

To evaluate our approach, we surveyed UCD Computer Science alumni to create a module dependency ground truth, which allows us to test the accuracy of our system in discovering such module dependencies. The results were favourable, as it detects the majority of module dependencies. Furthermore, we determined that this approach can be leveraged to help with various use cases.

The key contributions of this chapter are the following:

- We build a non-negative matrix factorisation model that allows us to detect implicit dependencies between modules in subsequent years. By harvesting this information and the module similarity, we can build connected graph visualisations that can be used to serve different use cases. We will present three possible uses cases and visualisation prototypes.

- We conducted a survey to create an expert-based ground truth. We asked four UCD BSc Computer Science alumni to rate all possible connections between the BSc Computer Science programme modules and utilise the answers to create the ground truth.
- We conducted an offline evaluation using the expert ground truth to evaluate the approach. We show that the approach can detect the majority of stated connections between modules.

The rest of the chapter is organised as follows. In Section 5.1 we motivate our approach and present related work in the area. The technical details of our approach are presented in Section 5.2, alongside a short introduction to NMF. We present our evaluation, and ground truth creation in Section 5.3. In Section 5.4 we present three use cases that utilise the results of the dependency detection approach. Finally, we conclude this chapter in Section 5.5.

5.1 Motivation

The work presented in this section focuses on the sequential nature of undergraduate degrees and the relations and dependencies between modules. In universities, students are often offered opportunities to personalise their curriculum. Often modules are related to each other and are restricted by pre- and co- requisites. However, these dependencies are not always formally defined. It can be an additional challenge for students who are trying to find suitable modules to make an informed decision if requisite definitions are missing or incomplete. This can lead to students being declined their module choices because they do not satisfy the formal requirements or students being disadvantaged by insufficient prior knowledge. Additionally, choosing suitable modules in the context of the sequence of their academic path can positively impact students' overall performance, satisfaction, and graduation time.

The natural sequential nature of academic programmes lends itself to a sequence-based approach. In [120] the relationship between course sequences and students' time to graduate and graduation Grade Point Average (GPA) was explored. The work shows that there is a strong correlation between the order in which students take the modules and their *time to degree* (TTD). The proper sequence of modules seems to impact the students' ability to graduate on time substantially. While there seems to be less of an impact of the sequencing regarding the overall GPA, the work provides solid empirical evidence of the importance of module order.

Other efforts have been made to recommend sequences of modules to students or taking the modules' order into account. In [135] a precedence mining model is presented that uses students' ratings to find patterns in sequential courses. Different recommendation algorithms, such as the *Popularity Algorithm* and the *Single Item Max-Confidence Algorithm* are then used to score and rank course recommendations. An offline and online study is conducted and evaluated based on historical student data and the results of a user study. It is concluded that algorithms based on precedence mining information can provide higher predictability and coverage than traditional collaborative filtering approaches. In their later work [134], Parameswaran et al. describe how complex constraints and pre-requisites can be implemented in a course recommender system. It is argued that including constraints and requirements into a recommender system is vital to be able to offer optimal personalised recommendations to students. In [42] six different ranking methods have been explored to discover optimal course sequences. From historical data, course sequences of high and low performing students were compared and concluded that hidden pre-requisites might exist, making some course sequences more valuable than others. Overall, it was concluded that obvious and latent dependencies between courses could be detected using an appropriate network model.

In our approach, we use matrix factorisation techniques to model dependencies between modules. Non-negative matrix factorisation (NMF) is a well-known state of the art feature extraction algorithm, especially useful in situations where there are many attributes with weak predictability [133]. NMF uses techniques from multivariate analysis and linear algebra to produce meaningful patterns. Matrix factorisation approaches have been shown to show valuable results in the domain of grade prediction. For example, in [51] matrix factorisation is used to predict grades and rank courses. This work adapts the matrix factorisation techniques for context-aware recommendation presented previously [18] by adding student- and course-side contexts. The approach was evaluated against different approaches and has shown promising results. Furthermore, Bhumichitr presents a low-rank matrix factorisation approach called *Alternating Least Squares* (ALS) algorithm, first proposed by [21], to recommend elective courses to students [22]. The experiments presented compare the ALS approach and collaborative filtering based approaches on historical course enrolment data. The results show that the matrix factorisation approach can outperform classic collaborative filtering approaches, showing up to 86% accuracy results.

In UCD, each module descriptor in the module catalogue provides an *Am I eligible to take this module?* section with two categories *Requirements, Exclusions and Recommendations* and *Module Requisites and Incompatibles*, to provide students with the information about explicit requisites and recommendations for the module, see Figure 5.2. How-

COMP47590 Advanced Machine Learning

Academic Year 2021/2022

COMP47590 is an advanced module on Machine Learning that builds on the core concepts covered in COMP47490 or COMP47460. Either COMP47490 or COMP47460 is a prerequisite for this module. This module covers advanced, state of the art topics in machine learning in areas such as deep learning, ensemble methods, semi-supervised learning, human-in-the-loop machine learning, unsupervised machine learning, reinforcement learning, and social network analysis. Significant prior programming experience is essential (in either Java, Python or C/C++).

Show/hide content

+ Open All

Curricular information is subject to change

What will I learn?	+
How will I learn?	+
Am I eligible to take this module?	-
<p>Requirements, Exclusions and Recommendations</p> <p>Learning Requirements: Either COMP47490 or COMP47460 is a prerequisite for this module. To complete the continuous assessment, this module requires significant prior programming experience in a language such as Java, Python, Ruby or C/C++.</p> <p>Learning Recommendations: Students should have strong mathematical ability, as some of the algorithms require some understanding of linear algebra and statistical concepts.</p> <p>Module Requisites and Incompatibles Not applicable to this module.</p>	

/hub.ucd.ie/jusis/IW_HU_MENU.P_PUBLISH7p_tag=MODULE&MODULE=COMP47590#collapseCB100-60

Figure 5.2: Module Descriptor for Module COMP47590 *Advanced Machine Learning* with Learning Requirements and Learning Recommendations (2020).

ever, our research shows that out of the 52 core and optional modules offered in the BSc Computer Science programme in the 2018/2019 term, only four module descriptors defined any pre-requisites, and only two modules defined co-requisites. Furthermore, no other module in this set defined any further requirements, such as exclusions, recommendations or incompatibilities.

With the sparsity of the available information, we argue that students are not sufficiently aware of the explicit requirements and implicit dependencies between their modules. This can lead to multiple issues when selecting optional or elective modules and curating their personal academic plan. For example, students might take modules that have explicit pre-requisites or implicit dependencies that they are unaware of, leaving the student to either not being eligible to take the module and have to choose another module at the last minute, or students might lack the knowledge that is required to excel in the module, leading to students having to put in extra work to keep up or face being left behind. Using matrix factorisation, we aim to detect explicit requirements and implicit dependencies between modules in consecutive years of academic programmes, which allows us to cater to several use cases and ultimately help students understand the sequential nature of their academic journey and the options along the way.

5.2 Topic Modelling for Module Dependency Detection

In this part of our work, we use non-negative matrix factorisation to extract features from the module descriptors. NMF is a widely used feature extraction algorithm that is especially useful for text mining in high-dimensional and sparse data sets that facilitates interpretable output. In what follows, we describe non-negative matrix factorisation in detail before presenting our system architecture.

5.2.1 Non-Negative Matrix Factorisation

Non-negative matrix factorisation is a technique derived from multivariate analysis and linear algebra that has been applied in many different areas such as astronomy [56], bioinformatics [192], and computer vision [29]. In recommender systems and adjacent fields, it is predominantly used for topic modelling, clustering, feature extraction, and rating prediction. NMF is particularly useful in applications where many

of the attributes present are ambiguous or hard to predict, as NMF combines these attributes to produce meaningful patterns and topics.

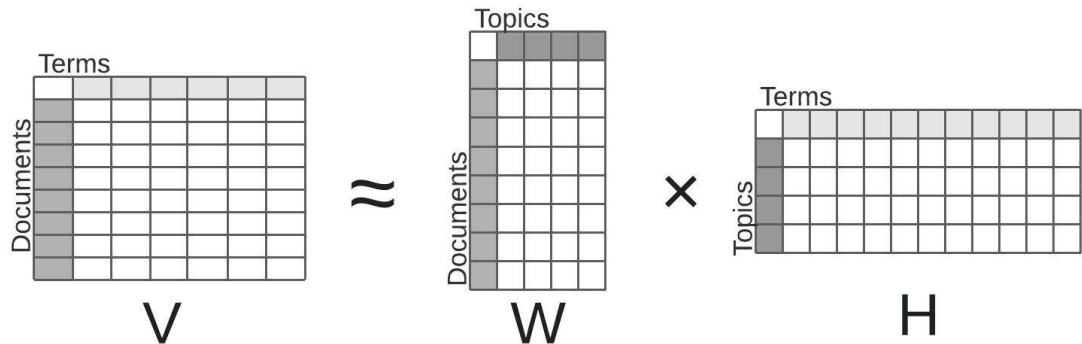


Figure 5.3: Matrix Decomposition in NMF.

NMF decomposes a document-term matrix (V) into two smaller matrices, namely the document-topic matrix (W) and topic-term matrix (H), see Figure 5.3. The document-term matrix V describes the visible variables; it comprises the term frequencies of each document. The values in V can be approximated through matrix multiplication of the sub-matrices. The feature matrix W contains the topic distribution of each document, whereas matrix H represents the word frequencies in each topic or hidden variables [95].

To create the two lower-ranking matrices, the NMF algorithm assumes two non-negative matrices and iteratively modifies them, minimising a loss function. This way, the structure of the original data can be preserved, and both matrices are non-negative. Once an approximation error or number of iterations is reached, the algorithm terminates.

Document	Header	Article
d_1	Ireland now has one of the fastest vaccination rates, data shows	London-based not-for-profit organisation Our World in Data states that ...
d_2	Gavin Coombes represents a new Munster that can end barren run	Munster possess the personnel to win silverware this season, the key is ...
d_3	Unemployment in Irish economy falls to pandemic low of 10%	The headline rate of unemployment in the Irish economy has fallen to 10 ...
...

Figure 5.4: Example of *Irish Times* News Dataset.

We can consider a simple example, in which we want to categorise news articles to the topics they discuss in order to present users with articles they might be interested in. For this we consider a dataset where each document contains a news header and article

taken from the *Irish Times* website¹ in the year 2021, see Figure 5.4. We can create the document-term matrix and, using the describe NMF approach decompose them into the lower ranking matrices W and H see Figure 5.5. We can see in this minimal example that document 1 d_1 shows the strongest weights to topic 1 and topic 3, which looking at the topics-term matrix might describe articles related to the vaccination and the Covid-19 pandemic. Topic 2 shows the highest weights for terms related to *munster* and might describe sports-related news; we can see document 2 d_2 showing the highest weights for this topic.

	ireland	vaccination	munster	economy	...
d_1	3	7	0	1	...
d_2	1	0	6	0	...
d_3	4	2	0	6	...
...

 \approx

	Topic 1	Topic 2	Topic 3	...
d_1	0.81	0.01	0.34	...
d_2	0.02	0.65	0.11	...
d_3	0.29	0.00	0.67	...
...

 \times

	ireland	vaccination	munster	economy	...
Topic 1	0.051	0.083	0.001	0.005	...
Topic 2	0.033	0.019	0.081	0.002	...
Topic 3	0.043	0.031	0.007	0.088	...
...

Figure 5.5: Example of News Dataset NMF Decomposition.

NMF is often used for text mining as it has many advantages compared to other topic modelling approaches. Using non-negative numbers allows for the topics to be easily interpretable. Further, due to the breakdown of the features, NMF introduces a notion of context in the sense that the same word can be interpreted for different meanings in different topics.

5.2.2 NMF for Module Dependencies Discovery

To implement the NMF approach for module dependency discovery, we utilise a scraped dataset of textual descriptors of the UCD BSc Computer Science core and optional modules as offered in the 2018/2019 term and presented on the UCD module catalogue website. There are 52 modules in total, with 31 modules being core modules and 18 optional modules. The distribution of core and optional modules per level can be seen in Figure 5.6. While there is only one optional module offered in year 1 and no options in year 2, the distribution shifts to 11 optional modules in year 3, and in year 4, nine out of the ten modules offered are optional.

Using the approach as described in Section 4.4, modules are viewed as documents made up of the set of terms contained in their descriptors. We use the Vector Space Model to represent each module as a vector in an n -dimensional space, where each dimension corresponds to a term from the overall set of terms. We apply standard text

¹<https://www.irishtimes.com/>

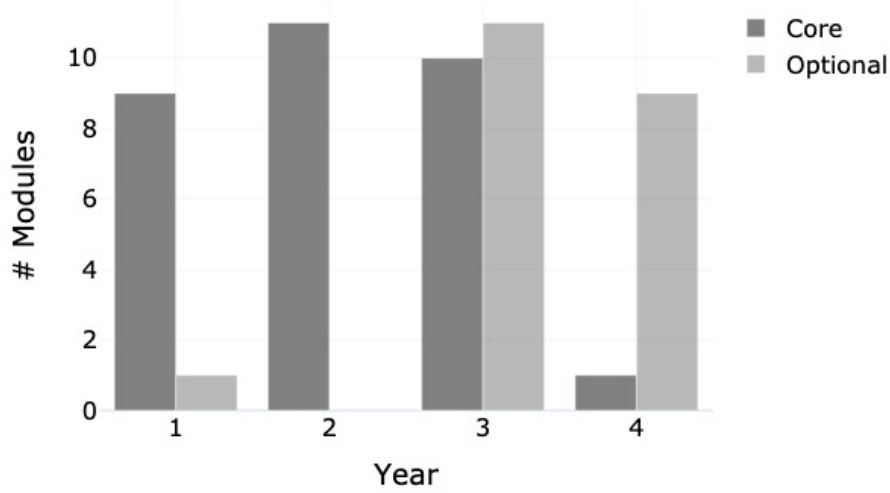


Figure 5.6: Distribution of Core and Optional Modules per Level in 2018/2019 Term.

preprocessing, such as tokenisation, stop-word removal, and stemming, before applying TF-IDF for term weighting, with an n-gram range of 2, allowing for bi-grams. In Table 5.1 we present the maximum, minimum, mean, and standard deviation of terms in the module descriptors before and after the preprocessing (PP) steps. We can see that while there seem to be some outliers, the average module descriptor comprises seven sentences and about 100 words. After the preprocessing steps, in which we remove common stop words and the top 10 most common words in our set, the average number of words decreases to approximately 60 words. Thus, an average decrease of approximately 55% (with a standard deviation of 7.4) per module descriptor is measured.

	Sentences					Words				
	Min	Max	Mean	Median	STD	Min	Max	Mean	Median	STD
Before PP	2.0	29.0	7.0	6.0	4.26	32.0	237.0	109.0	95.5	50.4
After PP	-	-	-	-	-	18.0	168.0	60.7	50.0	29.8

Table 5.1: Sentences and Words in Module Descriptions Before and After Preprocessing (PP).

After the preprocessing steps and term weighting, we apply NMF to the created term-document matrix. NMF is a soft clustering algorithm in which each item can be assigned to several clusters rather than definitively assigned to just one. NMF allows us to identify clusters of items that share latent features. The calculated coefficient matrix W presents the membership weights for every module relative to each topic; an example is presented in Table 5.2. To determine the optimal number of topics, we take guidance from the ACM Computing Classification System [36], that lists 13 overarch-

ing topics within the computing field. We train the NMF model with increasing k from three to 13 and compare the distribution of modules per topic, as well as the average document-topic values. We conclude that nine topics provide the best results. The created topics and their respective top words are presented in Table 5.3. We can see that the top words for each topic seem to describe it sufficiently. We manually name each topic for easier description. For example, we named Topic 5 *Machine Learning & AI* as its main words are *machine learning*, *game*, *intelligence*, *learning*. In Table 5.2 we can also see that the Module *Intro to AI* shows the highest topic weight for this topic, as well as Topic 1 *Programming* and a smaller weight for Topic 8 *Advanced Programming*.

Topic/Module	0	1	2	3	4	5	6	7	8
Algo. Prob. Solving	0.0	0.08	0.14	0.0	0.0	0.19	0.0	0.0	0.0
Programming I	0.0	0.27	0.02	0.01	0.12	0.0	0.0	0.0	0.0
Data Structures	0.02	0.06	0.17	0.0	0.09	0.0	0.0	0.0	0.0
Intro to AI	0.0	0.15	0.0	0.0	0.0	0.86	0.0	0.0	0.02

Table 5.2: Example of Document-Topic Matrix.

We apply Cosine similarity on the coefficient matrix to calculate the similarity between modules. Finally, we use these similarity scores to present dependencies between modules in subsequent levels.

Topic	Top Words
0 - "Computer Architecture"	parallel, processor, architecture, performance, cycle
1 - "Programming"	programming, language, java, object oriented
2 - "Theoretical Fundaments"	proof, discrete mathematics, sets, principles
3 - "Linear Algebra"	linear, matrix, algebra, equation, vector
4 - "Software Engineering"	software, development, design, tools, testing
5 - "Machine Learning & AI"	machine learning, game, intelligence, learning
6 - "Databases"	information, database, data, information systems
7 - "Technical Fundaments"	circuits, logic, architecture, digital, systems
8 - "Advanced Programming"	mobile, current, networks, technologies, applications

Table 5.3: Top Words for Each of the Nine Topics.

5.3 Evaluation

We evaluate our approach presented above using an offline evaluation. However, we can not depend on the ground truth of actual module pre-requisite data, as there are nearly no explicitly defined requirements or implicitly inferred dependencies within

the UCD module catalogue. We, therefore, aim to create a ground truth based on expert knowledge.

We present the ground truth implementation and our evaluation results in the following sections.

5.3.1 Dataset & Methods

To evaluate implicit dependencies between modules in consecutive years, we require a ground truth representing those relationships. For this task, we asked four UCD BSc Computer Science alumni, that are now also teaching modules in the UCD BSc Computer Science programme, to define the ground truth. The four experts provide insight into the programme structure and module dependencies from two perspectives: (i) from a student view and (ii) from an instructor perspective. We presented each expert with every core and optional module in the programme space and asked them to rate every module in the subsequent year on a scale of 0 (*no dependency*) to 3 (*high dependency*), that is "How high is the dependency of module y to module x?". We combined the answers of the four experts E and calculated the mean dependency expert score $ExScore$ for every module m_i to each candidate module m_c as follows,

$$ExScore(m_i, m_c) = \frac{\sum_{i \in E} expertScore_i}{12} \quad (5.1)$$

where a score of 1 means a strong dependency between the modules (all four experts rated the dependency as *high* = 3).

Overall we asked the experts to rate 572 possible dependency connections. In Table 5.4 we show the differences in dependencies stated by the four experts. Each expert noted between 73 and 218 out of the 572 possible dependencies. Further, we can detect a difference in rated dependency strength between the experts. While experts 1, 2 and 4 mainly noted weak and medium dependencies, expert 3 shows an overall higher mean dependency expert score (2.2) by rating mainly medium and strong dependencies.

	# Dep (% Dep)	Weak Dep (%)	Medium Dep (%)	Strong Dep (%)	Mean (Median) Dep Score
Expert 1	206 (36.0%)	92 (16.1%)	69 (12.1%)	45 (7.9%)	1.77 (2.0)
Expert 2	73 (12.8%)	35 (6.1%)	24 (4.2%)	14 (2.4%)	1.71 (2.0)
Expert 3	218 (38.1%)	42 (7.3%)	89 (15.6%)	87 (15.2%)	2.20 (2.0)
Expert 4	113 (19.8%)	51 (8.9%)	58 (10.1%)	4 (0.7%)	1.58 (2.0)

Table 5.4: Dependencies stated by the Different Experts.

The union of all experts' noted dependencies, without taking the strength of the dependency into account, shows 323 or approximately 56% of all possible connections. If we only consider dependencies where two out of the four experts agree, this number decreases to 174 or approximately 30% of all possible dependencies. Further, in Figure 5.7 we present the union of dependencies as defined by the experts and the calculated dependency expert score. We can see that the majority of dependencies score below 0.5.

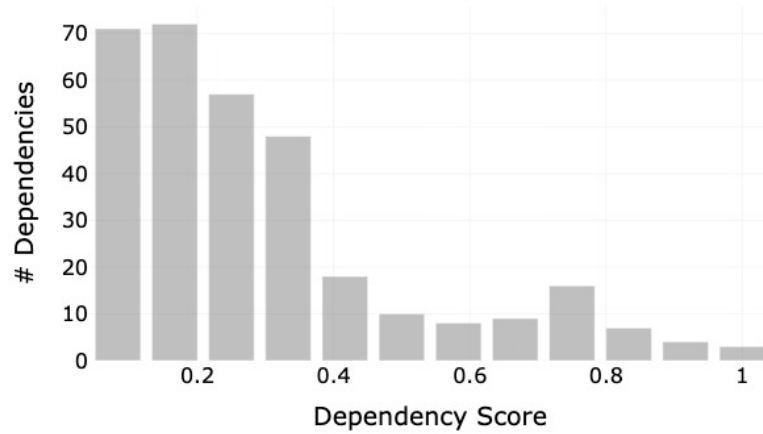


Figure 5.7: Number of Dependencies by Dependency Expert Score.

Further, we group the dependencies by their level/year of study. The majority of possible dependencies, 326 dependencies (63.29%), can be found from year 3 to year 4, with 131 (22.9%) dependencies between year 1 and 2 and 115 (21.1%) possible dependencies between year 2 and 3. In Table 5.5 we present the number and percentages of dependencies stated by the union of experts as well as the overlap of 2 out of 4 experts. While the most potential dependencies can be found in levels 3 to 4, the experts annotated only 40.8% (18.71% with 50% agreement) of them as a dependency. Out of the 115 possible connections in years 2 to 3, the experts rated 59.13% (37.39%) as dependencies. The experts found more than 93% (53.44%) of the 131 possible connections in levels 1 to 2 to be valid dependencies.

	# Poss. Dep. (%)	# Union (%)	# 50% Agreement (%)
Level 1 - 2	131 (22.90%)	122 (93.13%)	70 (53.44%)
Level 2 - 3	115 (21.10%)	68 (59.13%)	43 (37.39%)
Level 3 - 4	326 (63.29%)	133 (40.80%)	61 (18.71%)

Table 5.5: Possible and Detected Dependencies by Year of Study.

For the following evaluation, we consider the ground truth, the set of dependencies where at least two out of the four experts noted any dependency between the modules.

5.3.2 Results

We evaluate our approach, as presented in Section 5.2 using the expert ground truth. Firstly, we present the confusion matrix in Table 5.6 that shows the True Positives, False Positives, True Negatives, False Negatives between the predicted dependencies and the expert stated dependencies. Using the NMF approach, a dependency is assumed to exist if the cosine similarity between modules (using the coefficient matrix) is greater than 0. From the 174 dependencies stated in our ground truth, the approach detects 117 and misses 57. While the ground truth indicated no dependencies for 398 of the possible dependencies, 190 dependencies were predicted by the NMF approach (false positives). However, there is considerable overlap in connections that were stated as no dependencies by the experts as well as the NMF approach, as depicted by both the numbers of true positives (117) and true negatives (208). Finally, we calculate the precision and recall scores, as defined in Section 3.2, and weight them by the number of instances in each class (dependency/no dependency), which gives us a weighted precision of 0.66 and a weighted recall of 0.57.

	Predicted Dependency	Predicted No Dependency
Expert Dependency	117	57
Expert No Dependency	190	208

Table 5.6: Confusion Matrix.

We also evaluate the precision and recall for the dependencies for each level separately. In Table 5.7 we present the achieved precision and recall scores. While the differences are small, we can see that we can achieve the highest recall and second-highest precision for the dependencies in level 1 to level 2. Interestingly, these are the dependencies that show the highest percentage of valid dependencies and the highest agreement between the experts, see Table 5.5. Further, the mean dependency expert score for dependencies in levels 1 to 2 is the highest overall. This allows the conclusion that the approach is specifically suitable to predict dependencies where the experts detect a strong dependency.

Further, we can visualise the precision and recall depending on the dependency expert score. In Figure 5.8 we present the values for precision and recall with increasing dependency expert score threshold. With an increasing expert score threshold, the set of dependencies that are predicted decreases. The increasing precision with the slightly

	Weighted Precision	Weighted Recall
Level 1 - 2	0.70	0.70
Level 2 - 3	0.63	0.61
Level 3 - 4	0.71	0.50

Table 5.7: Weighted Precision and Recall Scores for Dependencies by Level.

decreasing recall tells us that our approach can accurately predict module dependencies with a high dependency expert score.

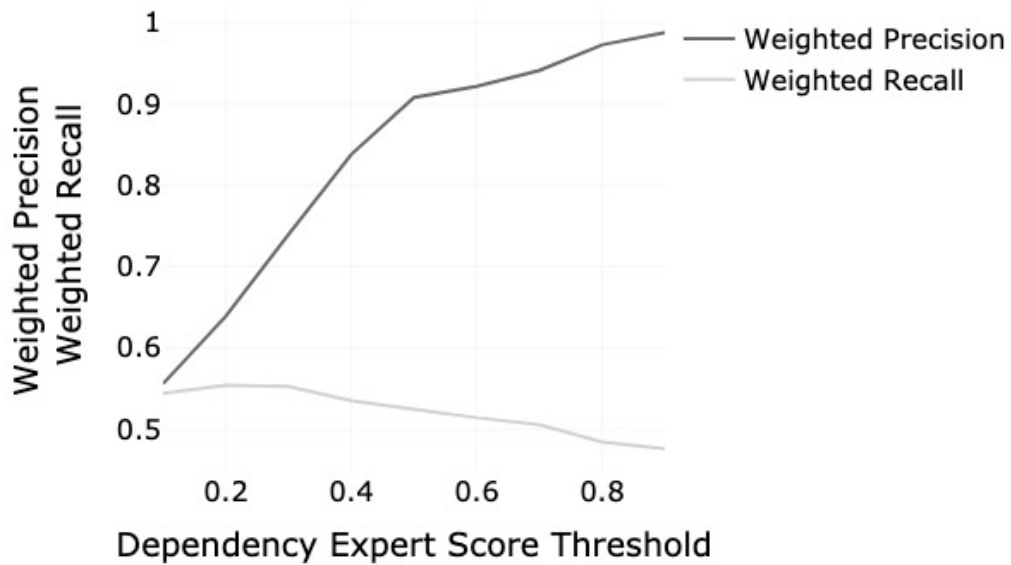


Figure 5.8: Weighted Precision and Recall Results for Increasing Dependency Expert Score Threshold.

We also present the precision and recall results with increasing cosine similarity threshold (i.e. the similarity above which the NMF approach predicts a dependency between modules) and dependency expert score threshold of 0 in Figure 5.9. Again, we can see increased precision in higher cosine scores with the concomitant decrease in recall as the NMF approach predicts fewer dependencies.

5.3.3 Discussion

Our results in Figure 5.8 show that with an increasing threshold in dependency expert score, the weighted precision increases. That shows that we can detect dependencies with high dependency expert scores better than those with lower dependency expert scores. The weighted recall is seen to decrease, as expected, since fewer de-

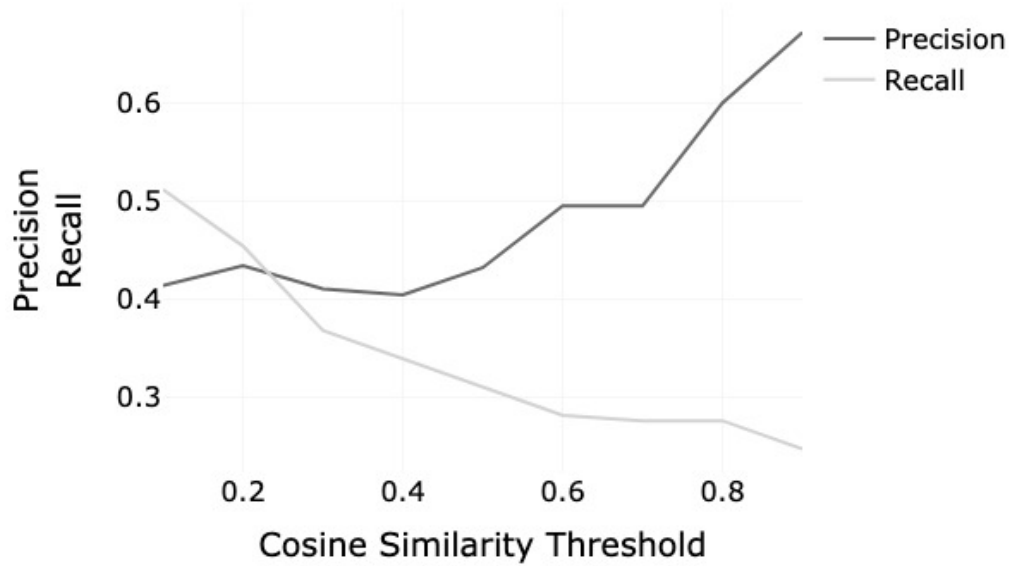


Figure 5.9: Precision and Recall Results for Increasing Cosine Similarity Threshold.

dependencies are predicted while the set of relevant (ground truth) dependencies remain unchanged. Likewise, the increase in precision we detect with increasing cosine similarity threshold, see Figure 5.9, is an indicator that high cosine similarity scores are a stronger indicator of module dependencies as defined by our expert ground truth.

Further, by taking a closer look at the confusion matrix, we can see a high amount of *false positives*. That means our approach detects dependency connections where our experts do not. We suspect that the expert might miss dependencies due to different reasons, such as the experts not remembering sufficiently, changes in curriculum, or experts not having taken a specific module at all. This is supported by the fact that 137 of the 190 false positives are in levels 3 to 4; as discussed earlier, there is a more significant number of modules and possible module dependencies in the last year of the UCD BSc Computer Science Programme to facilitate students with specialisation options. A brief manual analysis of the false positives substantiates our suspicion; we can detect some dependencies, for example, between the modules *Cloud Computing* and *Contemporary Software Development*, which was not stated by the experts but received a very high cosine similarity score and arguably a dependency connection is present in this case. Notwithstanding the above and other such examples, we conclude that the high rate of false positives seen requires additional analysis and further examination of the ground truth creation (e.g. by involving a greater number of human annotators); these matters are left to future work.

5.4 Use Cases for Module Recommendations

We can leverage the output of the above approach in multiple ways, serving different purposes and use cases, three of which we will present in the following sections.

5.4.1 Personal Curriculum Path

In this scenario, a student might be interested in a specific area of their programme and dependencies between modules that are essential steps on the way to this goal can be highlighted. For example, Figure 5.10 shows the personal path for a first-year student interested in the area of machine learning.

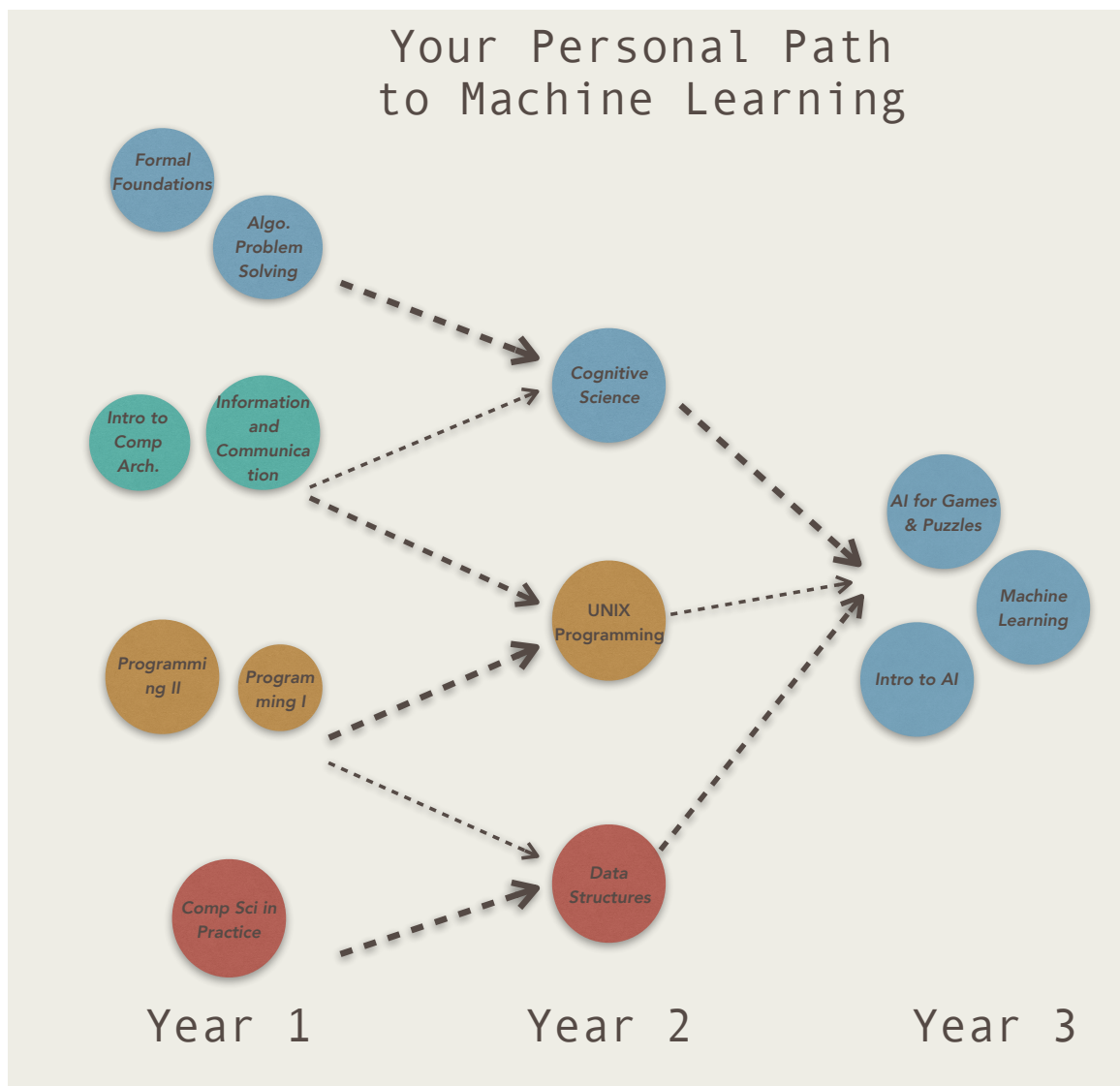


Figure 5.10: Personal Curriculum Path Visualisation.

Modules with the highest dependencies to this area for each year of the student's undergraduate programme are shown. Modules in this visualisation are clustered by their highest topic affiliation (colours of nodes) and year of study. The sizes of the nodes (modules) and arrows depict the importance of modules and relationships for the specified area of interest. In the example shown, modules belonging to four out of nine different clusters, as described in Section 5.2, are associated with the learning pathway.

In the first year, the diversity of topics is the highest, covering important foundational topics in Computer Science, while topics in later years are more specific to Machine Learning. *Topic 5* (depicted in blue) clusters modules related to Artificial Intelligence and Data Science. Unsurprisingly, these make up the majority of the key dependencies, but also modules from *Topic 2* (Theoretical Foundations, depicted in red) show strong dependencies to Machine Learning. While these connections seem obvious to Machine Learning professionals, we note that none of the modules presented in this example are official pre-requisites.

We argue that the visualisation proposed in this use case has great potential to support students in understanding the module space. By visualising the relationships between core, optional, and elective modules, the students can increase their knowledge about how modules are connected and how the skills acquired will benefit their learning path. Overall, this can help facilitate students in making more informed decisions when choosing modules. We discuss and consider this use case further in detail in Chapter 6.

5.4.2 Exploration of the Module Space

In a second use case, we can imagine a contrasting scenario to the use case above. This is a student who might not have a clear idea of their career or academic goals. It is easy to imagine students at the beginning of their academic life without knowing where that path might lead them. However, these students also have to make early decisions in their academic path about optional modules, elective modules or specific streams. What can we recommend to students without explicit preferences or goals?

Using the calculated dependencies as vertices in a directed graph, we can borrow ideas from graph theory to identify nodes/modules that satisfy the requirements (both explicit and implicit) of a wide variety of modules in the years to come. For example, in a directed graph, the indegree and outdegree can be calculated. We could harvest this information to find nodes that have a high outdegree and additionally target modules in a diverse set of topics. This allows the students to pick modules that will give them a

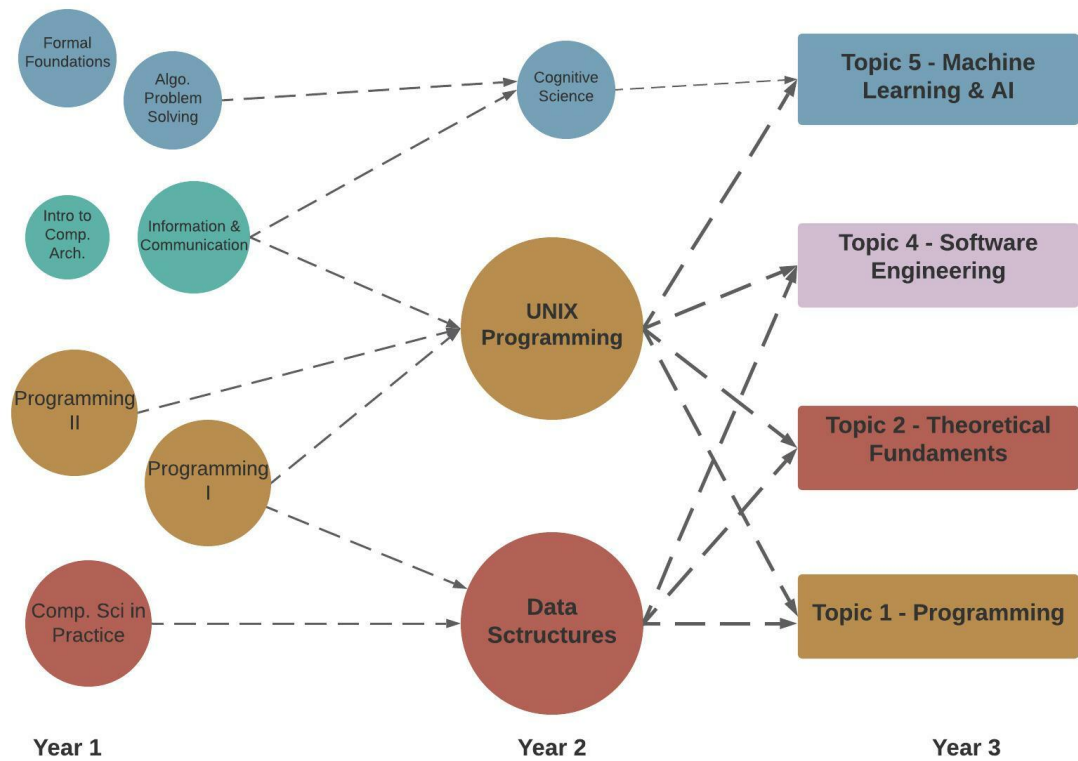


Figure 5.11: Mockup Visualisation of Module Recommendation with High Connectivity.

solid foundation of knowledge that is useful in multiple specialised paths. We present a visualisation mockup of this use case in Figure 5.11. Again, we consider a first-year student, but this student is unsure about their possible career or specialisation options in the future. In the visualisation, we can highlight modules that show dependencies to a diverse set of topics in year 3. In this example, the module *UNIX programming* provides skills that have dependencies to modules in topics 1, 2, 4, and 5. Likewise, the module *Data Structures* shows a higher degree of diverse connections and can therefore be recommended to a student who wants to build a solid foundation and explore different paths in the future. As in the previous use case, the presentation of the modules can increase students understanding of how modules are connected, which can support their knowledge about the overall programme structure and its connection to specialisations and career path options. We further discuss and consider this use case in our user study presented in Chapter 6.

5.4.3 Recommending Alternative Modules

Other factors can heavily affect students decisions when choosing modules, such as availability or timetable clashes. As we have shown previously in Section 4.1, a high percentage of students each year will not get a place in their first or even second choice of elective module. Especially for students who want to acquire specific skills for a particular topic or career path, finding a suitable substitute module that will provide them with the right skills and possible pre-requisites for modules in subsequent years can be challenging.

Using the module dependencies, we can detect replacement modules for modules unavailable to the student by finding similar modules or providing similarly strong dependencies to a specific set of subsequent modules, topics, or career paths. This use case is not further evaluated in this work, and we will leave this together with other organisational and constraint problems, such as timetabling, for future work.

5.5 Conclusion

This chapter presented our approach to detect dependencies (both explicit and implicit) between modules in consecutive years of a degree programme. We use NMF to find the latent factors within the academic module space that allow us to cluster and define module-module dependencies. To evaluate our approach, we created an expert ground truth by conducting a survey of university alumni. The information provided

by the four experts allowed us to calculate a dependency score between modules in the undergraduate Computer Science programme. Then, using precision and recall evaluation metrics, we conducted an offline evaluation using this expert ground truth.

The results showed that the NMF approach can detect the dependencies noted by the expert and identifies additional dependencies between modules that the experts did not identify. Thus, we can conclude that using non-negative matrix factorisation is feasible to depict module dependencies satisfactorily. This gives us a vital step in our journey to create a visual recommender system for students to explore their module space interactively.

We presented three potential use cases that showcase how the calculated dependencies can support module space exploration and student module recommendations. We visualised an initial graphical representation for two use cases, presenting possible user interfaces that can support students in gaining knowledge and understanding of their module space, the relationships between modules and their topics, and their potential specialisations and career paths. These two use cases provide the groundwork for our final online advising system, which we will present and evaluate using a live user study in the following chapter.

LIVE USER STUDY OF VISUAL MODULE EXPLORER AND ADVISORY SYSTEM

The main objective of this work is to develop approaches that help students make more informed decisions in their academic lifecycle. For this, we focused on building a content-based approach in Chapter 4, showing that this technique allows us to recommend diverse and suitable elective modules to students based on their explicit interests. We further explored matrix factorisation for the presentation of dependencies and connections between modules in consecutive years in Chapter 5. The technique allows us to present modules in a visual graph network, using module connections as links. In our previous work, we presented prototypes of an online web-based advisory system, see Chapter 4, as well as visualisation approaches in Chapter 5. In this concluding study of our work, we present an improved system based on the previously presented research findings.

The visual recommender and advisory system, presented in this chapter, draws on the ideas of module dependency and module space visualisation, using only the textual module descriptors. We combine those ideas into an interactive module exploration and recommender system. The system architecture relies on our findings from Chapter 4, where we use a content-based approach to generate module similarities. In Chapter 5 we concluded that a topic modelling approach of latent factors in the module space is feasible. We include these findings in our visual advisory system by using LDA to depict module affiliation. Finally, we implement an improved visualisation based on our use case prototypes based on the use cases presented in Chapter 5 and include interactive functionalities for explicit data collection and student-led module exploration.

While we were able to evaluate previous approaches using offline evaluation and an expert ground truth, an in-depth live user evaluation is needed to gain further insight into the effect of our approaches. In this part of our research, we incorporate our sys-

tem into a web-based framework which allows us to conduct a live user study. The framework enables us to monitor user interaction and collect qualitative and quantitative data from the UCD BSc Computer Science programme participants. The results from over 100 participants show that the trialled system improves students' knowledge about their programme structure and evaluates the need and acceptance of online academic advising.

The key contributions of this chapter are the following:

- We present an academic advising system which is comprised of two main components: Using matrix factorisation techniques and content-based recommender system approaches, we are able to build (i) a rich *visualisation* of the module space which allows students to self-explore the space and their options and (ii) we present students with suitable *elective module recommendations* based on their explicitly stated interests, and possible module paths.
- We built a browser-based framework and conducted a live user study. Using state-of-the-art techniques, we embed our visual module explorer and recommender system in a modern web framework. We further designed two questionnaires to record participants' initial knowledge and aspirations as well as their experiences and opinions after using the system.
- We evaluate our system and hypotheses using the collected qualitative and quantitative data. Our results show two major takeaways: (i) the majority of participants stated a knowledge gain after using the system, independently of their initial knowledge, and (ii) the majority of participants enjoyed using the presented system and would use it again.

The rest of the chapter is organised as follows. In Section 6.1 we present the system architecture and technical details of our approach. The design and research questions of our user study are presented in Section 6.2. The results of the user study are presented in three parts in Section 6.3 before we conclude this chapter with a conclusion in Section 6.4.

6.1 Motivation

From the previously presented work and related research shown, it is clear that recommender systems have the ability to support students in finding the right modules greatly. We have shown in Chapter 4 that, by focusing on the content, we can

build rich module representations and recommend a wide variety of modules. Providing students with additional information about the module space, such as connections and dependencies between modules, can be beneficial in optimising students' understanding of their programme space. An important factor to consider when building recommender system is *explanation* [10, 187]. Explanations are essential in helping users trust the system, which is particularly important when recommending high-cost items. Throughout this research, we have visualised our approaches and argued that the visualisation of dependencies between modules could act as a form of a natural explanation for students to better understand the recommendations presented.

Explanations in recommender systems have long been an essential part of recommender systems research. Many studies have shown that recommender systems can achieve higher acceptance in users when accompanied by explanations [185, 149, 77, 178]. In collaborative filtering approaches, the explanation often references other users preferences, such as "People who bought X are also interested in Y". In contrast, content-based approaches might use item features to explain recommendations, e.g. "You might enjoy book X because it is written by the same author as book Y that you have previously purchased". In [23] it was shown that content-based explanations are significantly more effective, as they enable the users to make more informed decisions.

Additionally, visualisations can help gain trust and understanding in recommender systems. In [10] it was shown that simple graphs and descriptions can outperform other explanation types in collaborative filtering approaches. Especially personalised visualisation options could improve users' satisfaction in recommender systems. Recently, Tsai and Brusilovsky have surveyed different approaches to visualising explanations for recommender systems [187]. It was shown that overall, users preferred visual explanations over textual ones.

Gaining students' trust and acceptance of module recommendations play a vital role when implementing such systems. Visualising recommendations and connections between modules can help explain the recommendations and increase the students' ability to self-explore their module space.

In [102], the *CourseQ* system, a web-based interactive recommender system, is presented. The system uses course information and creates a 2D layout using Linear Discriminant Analysis and T-Distributed Stochastic Neighbour Embedding to reduce the dimensionality of the vector module representation. The model is presented as a cluster of all modules (represented as nodes) coloured in the colour according to their calculated topic. The user interface also includes additional information such as keywords. Students can interact with the visualisation by selecting keywords, zooming into the visualisation, and applying filters (such as time of day of the module). Based on this

input, the system will highlight modules that fit the students' needs in the visualisation and present the topic distribution of a selected module. A user study was conducted in which a baseline user interface was presented to half of the 32 participants, which had the same functionalities as *CourseQ* but lacked the visualisation element. The participants were asked to freely interact with the system before answering a questionnaire. The results show that the system that included the visualisation showed significantly better ratings for four properties: perceived accuracy, information sufficiency, explanation and transparency, and confidence and trust. The work concludes that visualisation in academic recommender systems can increase these aspects and notes that future work could include structure-related topics, such as prerequisites. While the work in [101] and in more detail presented in [102] provides some interesting results that tie closely with our approaches, we advance the modalities of interactivity by allowing students to interact with the visualisation beyond zooming and panning.

It seems clear that recommender systems in academic areas can greatly benefit from including explanation and exploration components. One option to realise this is visualisation and interactivity. We detect a lack of research being done in this particular area. The requirements for the new system are, therefore, threefold: (i) visualise module connections in the context of a four-year undergraduate programme, (ii) allow student-led exploration using interactivity, and (iii) provide elective module recommendations and additional information to students. The work done in this chapter aims to show the effect of an interactive visualisation on academic module exploration and recommender system and its benefit for students' acceptance of the system.

6.2 Visual Module Explorer & Advisory System

We draw on ideas from our previously designed systems in Chapter 4 and Chapter 5 to build a visual recommender system using module description data from University College Dublin. We aim to provide students with an interactive visualisation of the modules that are available to them given their programme of study and to offer students specific recommendations for modules that fit their experience and goals. This section presents an overview of the system and outlines the text mining techniques used to represent modules in a suitable way to visualise the resulting information space and generate recommendations.

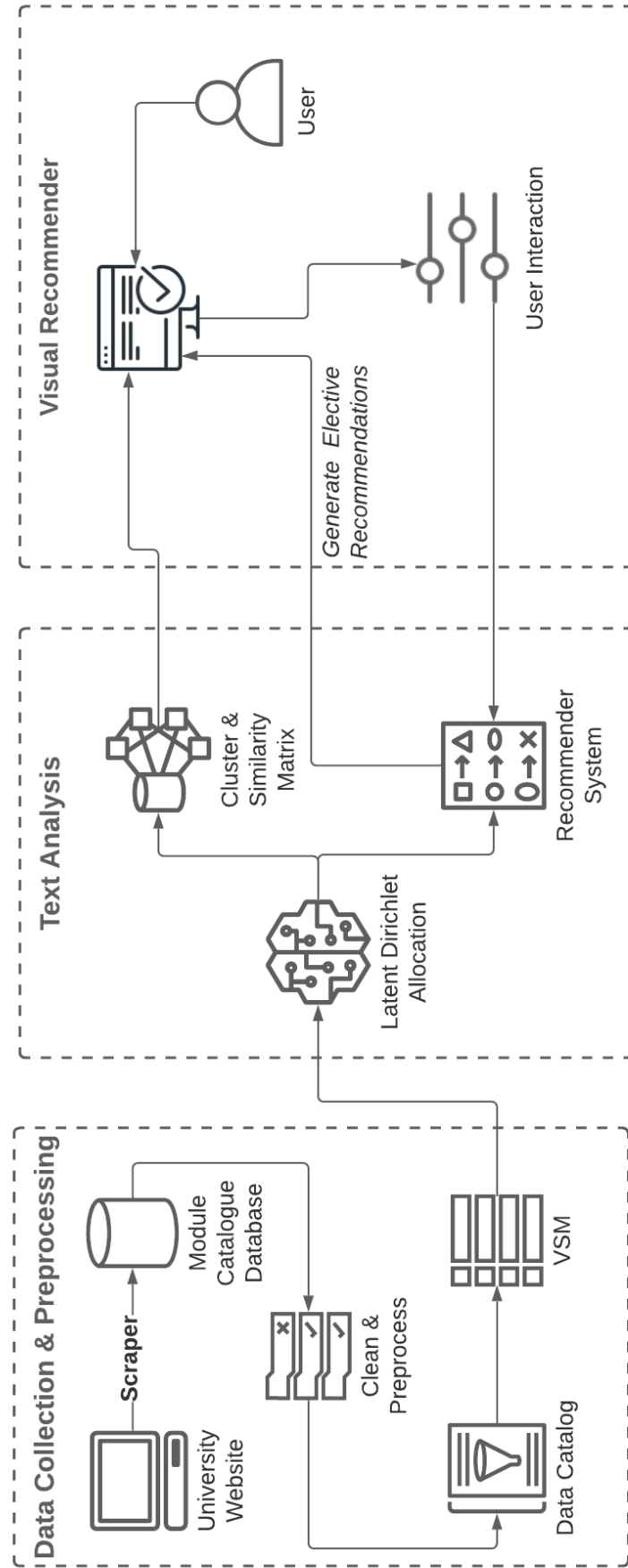


Figure 6.1: Visual Recommender & Advisory System: System Architecture Overview.

6.2.1 System Overview

Figure 6.1 presents the overall system architecture, comprising of three main components (i) Data Collection and Preprocessing, (ii) Text Analysis (iii) Visualisation and Recommendation. We will discuss each of the components in detail in what follows.

6.2.1.1 Data Collection and Preprocessing

The *data collection and processing* component is responsible for collecting and preparing the raw module description data. A web scraper collects this data from UCD's module catalog¹; for this study, we focused on the Bachelor of Science in Computer Science programme, which comprises 61 modules. Module descriptors are provided by the module coordinators and consist of various information such as title, trimester, stage, description, learning outcomes and assessment information. They vary significantly in quality and level of detail provided, making for a challenging text-based recommendation scenario.

We scraped the information of 1510 modules across all 39 programmes of UCD as offered in the academic term 2020/2021. We perform a brief analysis of the quality of the module descriptors across all scraped modules. The mean number of sentences in the description part of the module descriptor is 7.19 with a median of 6 sentences and a standard deviation of 4.35. This translates to approximately 138.13 (mean)/123 (median) and a standard deviation of 78.42 of the number of words per description. Secondly, we look at the textual descriptors of the *Learning Outcomes* section of the module descriptors. We can find a mean of 5.76, median of 5 sentences, and standard deviation of 5.32. Learning outcomes have a mean of 93.18 words (median 80 words) with a standard variation of 65.76. The distributions are presented in Figure 6.2 and Figure 6.3. We combine both textual descriptors, which leads to an average of approximately 12 sentences per module (median 11 and standard deviation of 7.76) with a mean of 229.02/median 228 words and a standard deviation of 117.17.

We perform standard preprocessing steps on all module descriptors (description and learning outcome): lower case conversion, tokenisation, stop-word removal, and removal of the top 10 most frequent words. We can see a significant reduction in the average word count per module after the preprocessing steps. In Figure 6.4 we can see an example of a module description for an introduction to psychology module before and after preprocessing. This example shows how generic module descriptors can impact the ability to accurately represent them in the recommender system. After

¹https://sisweb.ucd.ie/usis/!W_HU_MENU.P_PUBLISH?p_tag=MODSEARCHALL

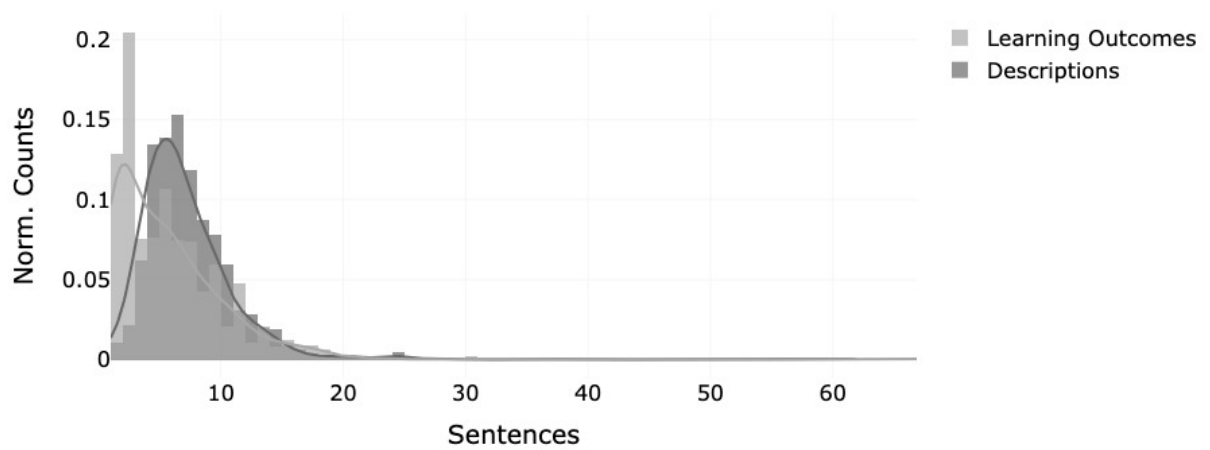


Figure 6.2: Distribution of Number of Sentences in Description and Learning Outcomes.

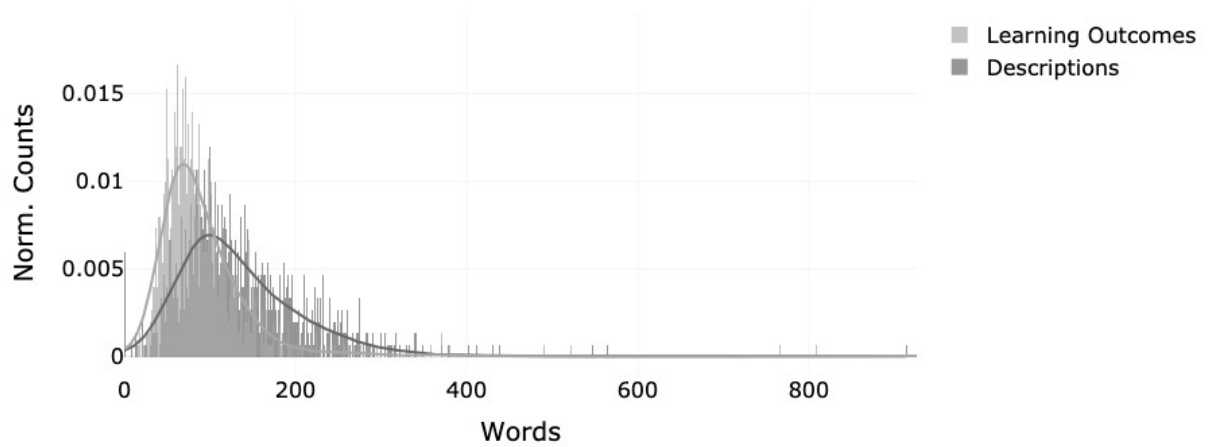


Figure 6.3: Distribution of Number of Words in Description and Learning Outcomes.

removing stop and common words, in this example, about 40% of the description is left.

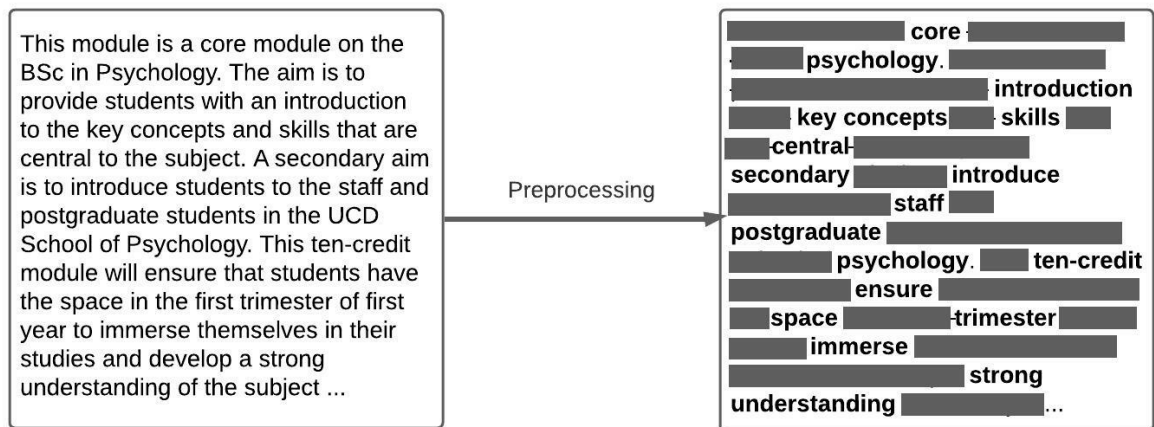


Figure 6.4: Module Descriptor Example Before and After Preprocessing.

Overall, the median number of words of description and learning outcome is reduced by 50% to 113 words per module description after all preprocessing steps, see Figure 6.5.

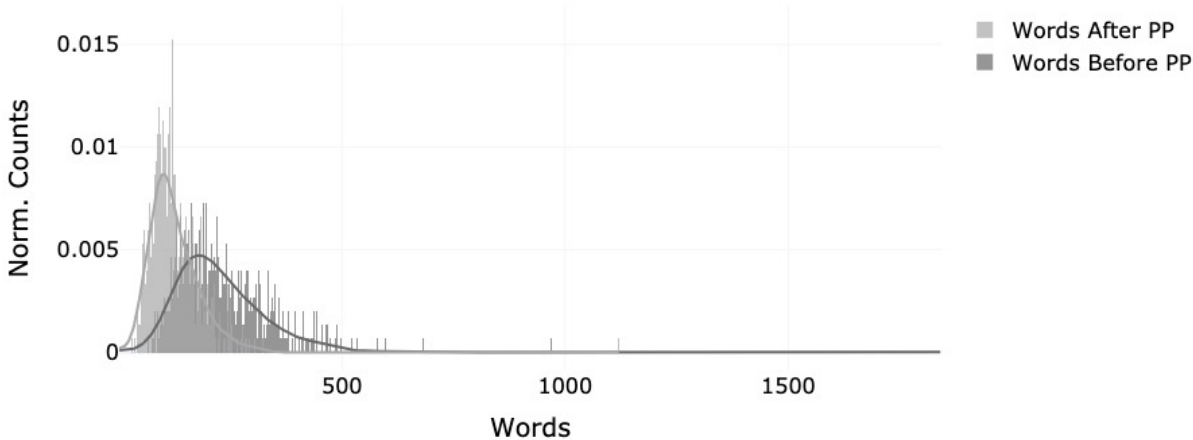


Figure 6.5: Distribution of Number of Words in Description and Learning Outcome Combined Before and After Preprocessing (PP).

The scraped, cleaned, and preprocessed module descriptors are then stored in a module data catalogue along with additional information, such as year and semester offered, lecturer, and level.

6.2.1.2 Text Analysis

The *text analysis* component is responsible for generating module representations that are suitable for recommendations. We use Latent Dirichlet Allocation [24] model as the topic modelling approach, in contrast to the matrix factorisation approach we utilised in Chapter 5. While we achieved favourable results in our previous work using NMF with nine topics, for this work, we determined more cohesive results using an LDA model with five topics. This decision was made based on the following experimental results and observations: (i) as we focus on visualising the module space, nine topics have shown to be too detailed to visualise coherently. While nine topics might cover slight differences in latent topics of the modules, overall, those details are hard for students to understand, especially when only conveyed through a graph visualisation. (ii) NMF has shown subpar results when clustering modules with lower amounts of topics, while LDA showed well distributed and cohesive results. Further we removed the diversity “*exploration*” component introduced in Chapter 5. This decision was made based on two main reasons. Firstly, this user study focuses on students module preferences and exploration ability. We aim to determine why students chose their elective modules and how familiar they are with their options. Secondly, as we determine in Chapter 5 we can achieve a diverse set of recommendations without introducing explicit diversity measures by using a content-based recommender system approach. While we acknowledge the importance of diversity in module recommender systems, we choose to omit this functionality for the sake of the clarity of the user study’s main objective.

Latent Dirichlet Allocation is a generative probabilistic model containing three layers of Bayesian probability models. In LDA text classification, the layers represent words, topics and texts. The general idea is that each document can be represented as a mixture of several underlying topics, and each of those topics is a mixture of several words. Dirichlet distribution is used to model the relation between documents and topics in the corpus. The corpus D is defined as a set of M documents $D = w_1, w_2, \dots, w_M$, where each document w is represented as a sequence of N words, $w = (w_1, w_2, \dots, w_N)$. The three-layer probabilistic model is presented in plate notation in Figure 6.6, α and β parameters are sampled in the process of generating the corpus. The corpus-level variables θ are sampled once per document. The variables z and w are generated for each word in each document in the word layer. K denotes the number of topics, and φ presents the vectors storing the Dirichlet-distributed topic-word distributions.

The joint distribution of a topic mixture θ is then calculated as follows:

$$p(\theta, z, w | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta) \quad (6.1)$$

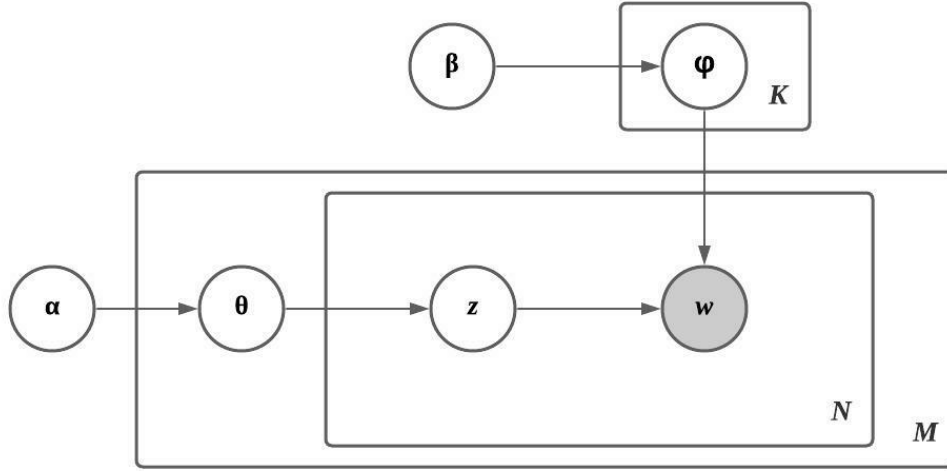


Figure 6.6: Graphical Representation of LDA.

To infer the topics, the generative process is described as follows [24]:

1. Choose distribution φ for each topic from Dirichlet distribution with parameter β
2. Choose distribution θ for each document from Dirichlet distribution with parameter α
3. For each word w in each document:
 - (a) Select a topic z from topic distribution θ
 - (b) Select a word w from topic z

The three-layer architecture of LDA allows us to sample the topic node repeatedly, which leads to documents being associated with multiple topics rather than one single topic.

6.2.1.3 Visual Module Explorer and Module Recommender

The *visual module explorer* component is responsible for the primary user interface, which presents the end-user with a network-based visualisation of relevant modules alongside a series of specific module recommendations based on their selections.

We based our visualisation on our research of related work and the use cases presented but not evaluated in Chapter 5. We present the students with a sequential module

map of their academic space and additional recommendations based on their interaction. This style of visualisation presents the academic space in two parts: (i) academic structure is inherently sequential, especially in European higher institutions, in which undergraduate studies are highly sequential, building from one year to others, (ii) modules are highly interlinked with another, with co- and pre-requisites not always clear to the students, modules often have an overlap in skills and learning outcomes that are not always apparent to students. The presented visualisation can help students understand these two critical factors of their academic module space better. In Figure 6.7 we present an early mockup of the visualisation approach. Modules are grouped by their year on the x-axis and by their dominant latent topic on the y-axis, as well as colour coded. Arrows denote connections between modules in consecutive years.

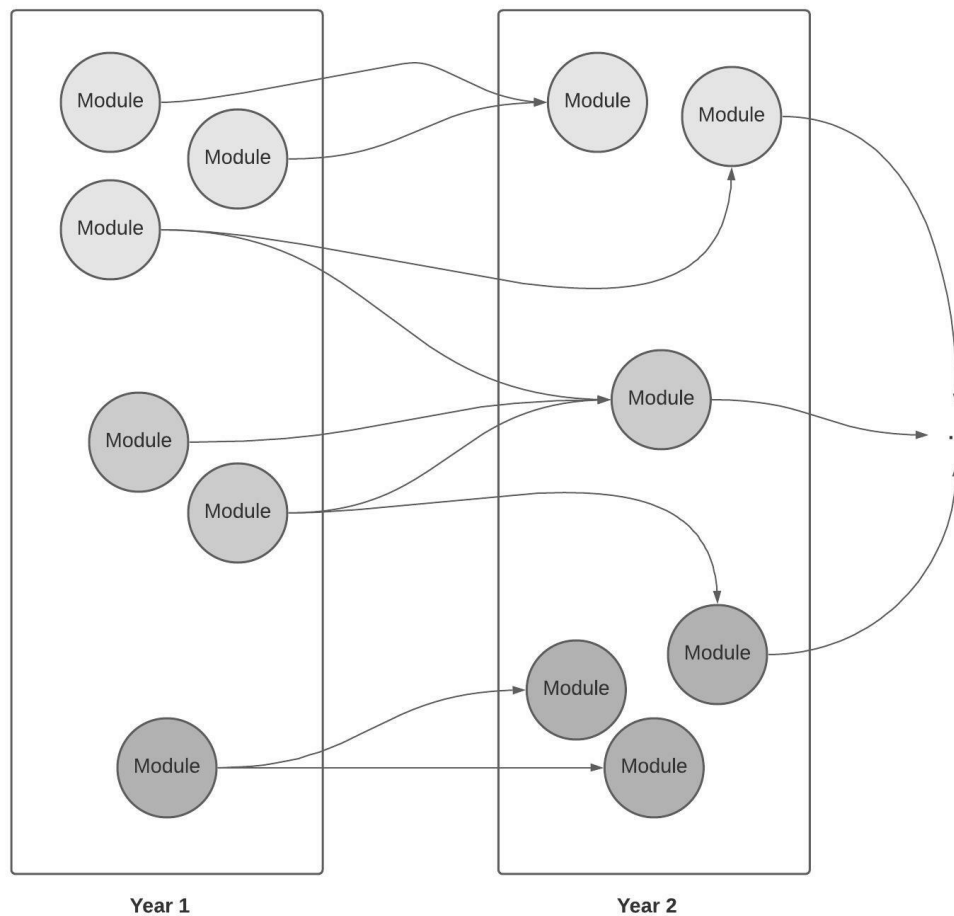


Figure 6.7: Mockup of Visual Recommender.

Further, we argue that the presented visualisation can act as a natural explanation for module recommendations. As discussed in the motivation (Section 6.1), explanations for module recommendation has been rarely addressed in previous approaches. However, in general, in recommender systems, the need for explanations has been proven

manifold. Explanations in recommender system become especially important when the costs for a *wrong* decision is high, such as booking an expensive hotel or buying a car. Arguably, choosing the wrong modules in a student's academic career can have unfavourable or even worse consequences (a low grade can lead to the loss of stipends or even a dropout). In many approaches presented in the past years, predicted grades were the main explanation used, and while the potential grade is arguably an important factor for students when choosing modules, it is also a high-risk explanation to get wrong.

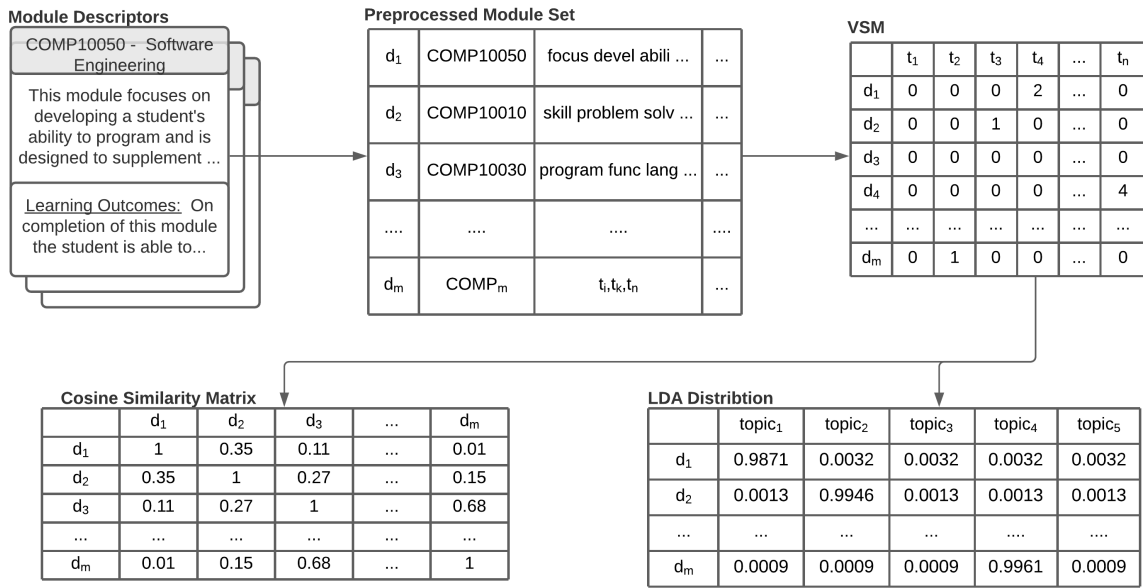


Figure 6.8: Technical Details of Generating Module Representations.

6.2.2 Generating Module Representations

To represent the module space adequately, we need to represent the modules based on their affiliation to a topic as well as the links between modules in subsequent years. To generate the representation for topic affiliation and module-module connectivity, we use two common text mining approaches, namely cosine similarity and Latent Dirichlet Allocation. We present the technical details in Figure 6.8. The basis for both techniques lies in the preparation of the scraped module descriptors. We use classic preprocessing steps (stop-word removal, tokenisation, lemmatisation) to clean the module descriptors. A Vector Space Model is created based on the frequency of each term t_1, \dots, t_n in each of the documents d_1, \dots, d_m . This sparse matrix is then used to create two components for the visualisation:

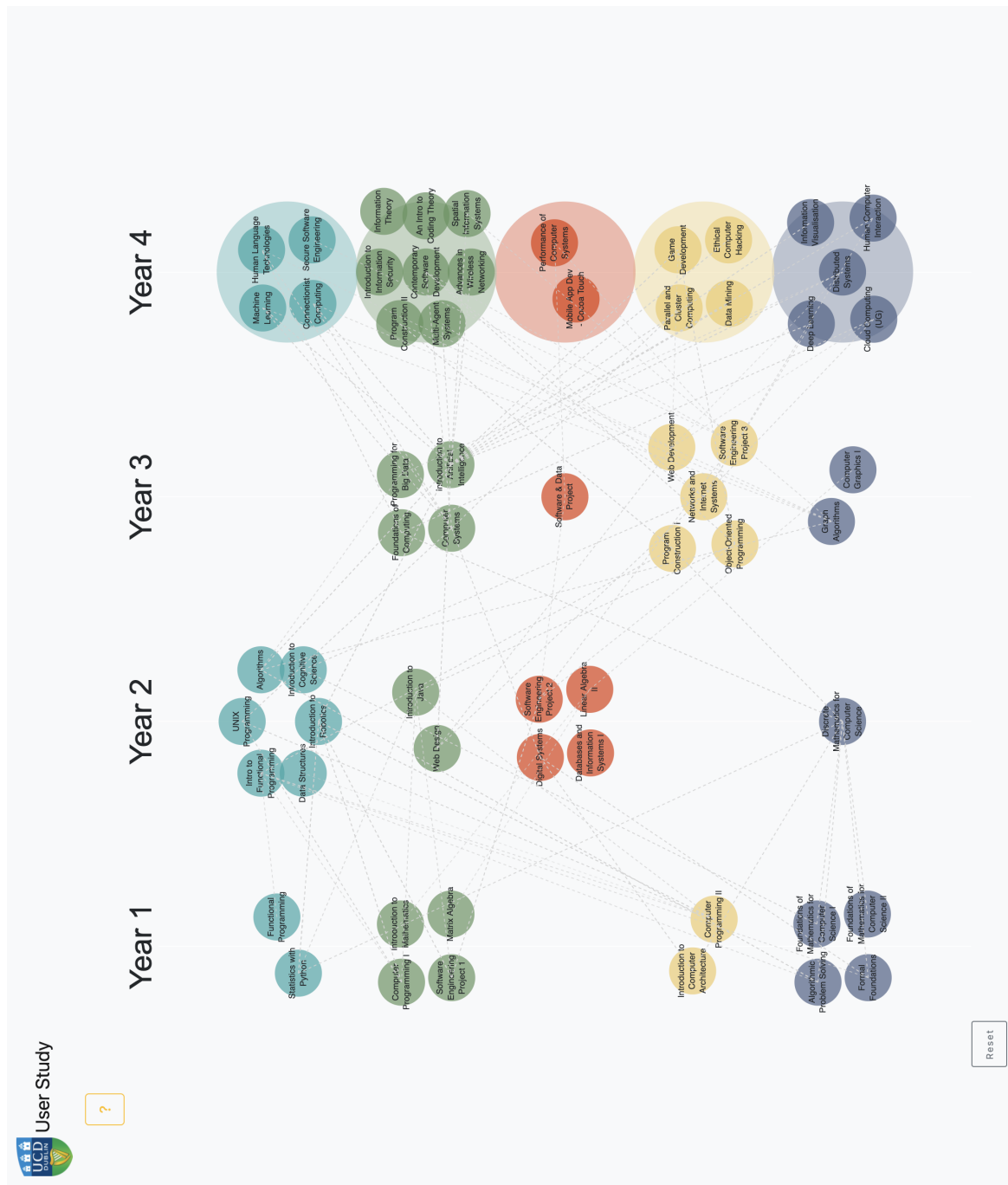
1. Using the vector representation of the module descriptors we calculate the pairwise *cosine similarity* between each module d_1, \dots, d_m . While there is considerable potential to use more sophisticated similarity metrics, we have found cosine similarity to perform satisfactorily. The cosine similarity matrix not only allows us to represent connectivity between the modules in the students' programme space but also establishes the basis for the elective recommender system.
2. We build a *Latent Dirichlet Allocation (LDA)* model, using the vector representation of the module descriptors. A commonly used technique in natural language processing, LDA is based on a generative statistical process that allows us to describe sets of documents by unobserved topics. These latent topics are based on sets of reoccurring terms presented in the vector space model. LDA allows us to calculate the distribution for each document d_1, \dots, d_m , that is each module in the programme space, to the generated latent topics $topic_1, \dots, topic_5$. We use these topics to cluster the modules into coherent groups of modules based on their dominant LDA distribution.

With these two module representations, we are able to visualise the programme space and calculate recommendations in our visualisation. We present the details of the visualisation and recommender system in the following section.

6.2.3 Visualisation & Recommendation

To visualise our system, we developed an interactive visualisation that shows the similarities between modules and the underlying structure of the programme space. The interaction with the visualisation allows the students to actively explore the space, learn about its structure and possible career path/specialisation options.

The interactive visualisation is a session-based system, which means we do not store any information about the user and do not require a login. For the user study, we present every student with the same initial *empty* visualisation, see Figure 6.9. We present each of the core and optional modules in the BSc Computer Science programme located by the year they are offered to the students on the x-axis. On the y-axis, we utilise the latent topics calculated by the LDA model to cluster the modules and colour code them. These clusters present the student with coherent groups of modules in each year as well as throughout their academic career. We further visualise interconnectivity between the module in subsequent years by connecting them based on their calculated cosine similarity. In the initial visualisation (Figure 6.9), we show all calculated similarities. Once the user interacts with the visualisation by clicking any module node,



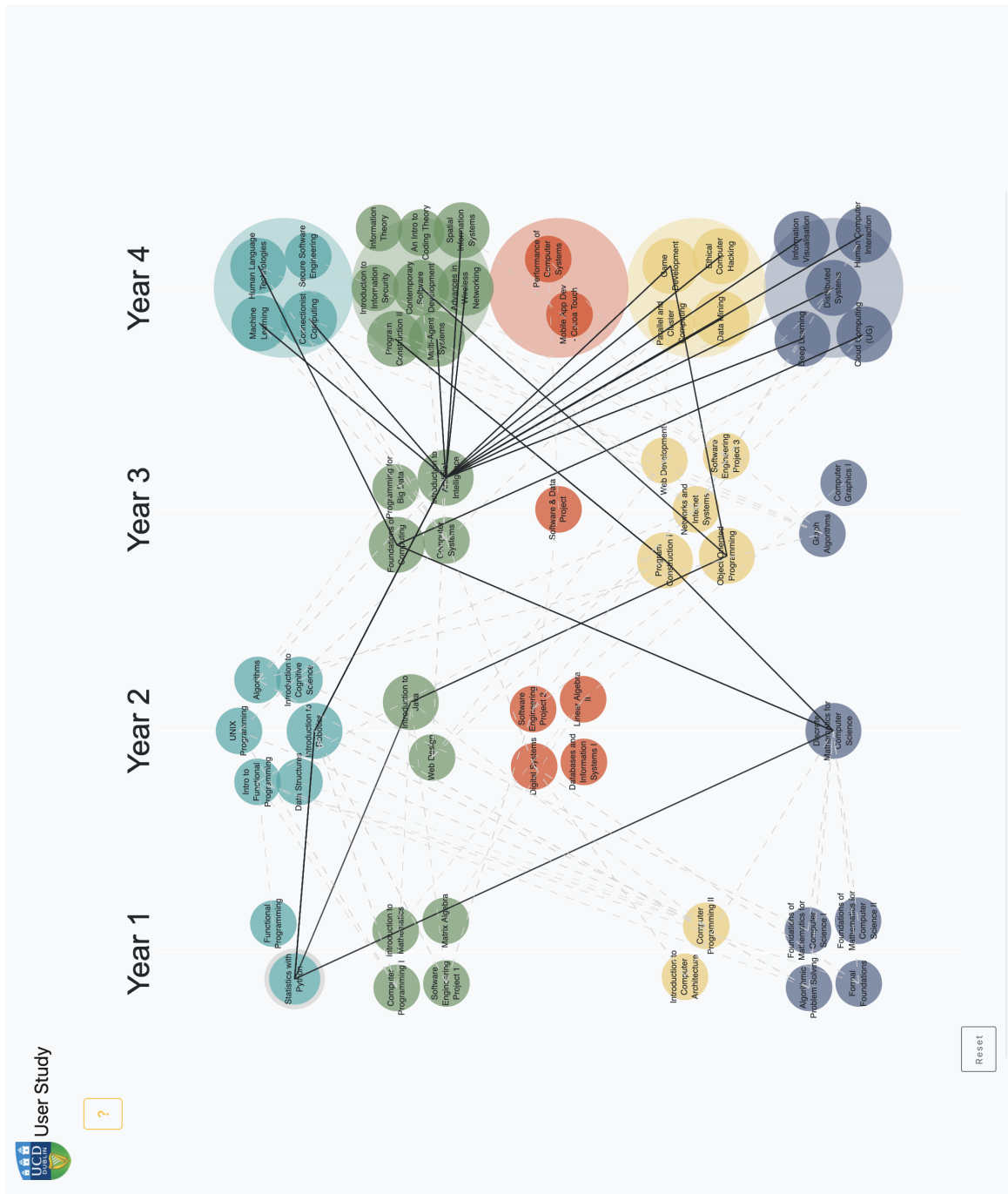


Figure 6.10: Interactive Visualisation with Active Nodes after Interaction.

Recommended Electives

Interested? Click the module to get more information!

PHYC20040 - Exploring the Solar System

PHYC10150 - Physics for Engineers I

ARCT40930 - Computational Design

FDSC40620 - Design Thinking for Food Packaging

COMP47680 - Human Computer Interaction

MIS10060 - Introduction to Business Analytics

CHEN10040 - Intro. to Eng. Computing

RDGY30090 - Radiography Research Project

EEEN10020 - Robotics Design Project

GEOG10140 - Mapping a Sustainable World

Additional Information

You choose Module(s)
Please click the modules for additional information

COMP10040 - Introduction to Computer Architecture


COMP10050 - Software Engineering Project 1

COMP10040 - Introduction to Computer Architecture

This module is part of **Topic 2**

Relevant keywords for this module:
architecture designs computer systems logic gates


Introduction to Computer Architecture is connected to 12 subsequent modules

The following modules are strongly connected: 

COMP30080 - Computer Systems

COMP20020 - Digital Systems

COMP30190 - Program Construction II

These modules have the highest degree of connectivity: 

COMP30080 - Computer Systems

Figure 6.11: Module Recommender and Additional Information Panel after Interaction.

the visualisation highlights the connected modules and paths throughout the years. In Figure 6.10, we can see the visualisation after the user has selected the first year module *Statistics with Python*. The visualisation shows three connections to modules in the subsequent year (*Introduction to Robotics*, *Introduction to Java*, and *Discrete Mathematics for Computer Science*). In year 3, the connections show strong connectivity to *Introduction to Artificial Intelligence* which opens up many options for the student in the fourth year. The participant can click as many or as few modules as they want, with the visualisation updating constantly, exploring different options along the way.

At the bottom of the visualisation, the student is presented with recommended elective modules, calculated on the current set of modules selected, as well as their connected modules in the path, see Figure 6.11. The list of recommended electives is updated after every interaction with the visualisation. Additionally, the bottom right of the visualisation offers additional information about the selected modules. In Figure 6.11 the student has currently selected two modules (*Introduction to Computer Architecture* and *Software Engineering Project 1*). For each of these modules, the visualisation offers additional information in the form of relevant keywords and modules that are strongly connected or have high connectivity to that module, which implements the notion presented in the use case in Section 5.4.2. This allows the student to further learn about the connections in the module. Strong connections depict a high similarity between the chosen and connected modules, whereas a module in the path with high connectivity has the potential to open up many options in subsequent years, as it shows a high similarity to many modules in the coming years. All information in the bottom half of the visualisation is clickable and will direct the student to additional information. Clicking modules will redirect the student back to the UCD module catalogue, whereas clicking on keywords will redirect to the relevant Wikipedia² article.

Algorithm 1 presents the pseudocode for the visualisation procedure and recommender system in detail. Initially, the user profile P is empty because the user has not indicated the modules that he/she is interested in. The default visualisation presents a summary visualisation of the modules that are available to them. When the student selects a module, m_i , the visualisation is updated to indicate the links between this module and related modules, M_c , and m_i and M_c are added to the user profile, P in line 5-7. If they are already included in the user's profile, then by reselecting the modules, the user is de-activating these modules in the visualisation, and they are removed from P ; see lines 9-10.

During each loop, the user is presented with a new set of recommendations based on the current state of their profile. In brief, we recommend the top-10 most similar

²www.wikipedia.com

modules based on their average similar to the modules in P , calculated using cosine similarity of the term-document matrix; this similarity scoring function is provided in line 14 and used in line 15 to identify the top-10 most similar modules which are then recommender to the user in line 16.

Algorithm 1: Update User Profile, Visualisation, and Recommendations

Data: M the set of available modules; s student id

Result: Updated User Profile, Visualisation, and Recommendation

```

/*  $P$  is the in-session user profile. */
1  $P = \emptyset$ ;
2 while session active do
    /* user remains active in the session by interacting with
       the visualisation/recommendations. */
3   if clicks( $s, m_i$ ) then
       /*  $M_c$  denotes the set of modules connected/related to
          module  $m_i$ . */
4      $M_c = \text{connected}(m_i)$ ;
5     if  $m_i$  not in  $P$  then
6        $P = P + \{m_i\}$ ;
7        $P = P + M_c$ ;
8     else
9        $P = P - m_i$ ;
10       $P = P - M_c$ ;
11    end
12    if  $P \neq \emptyset$  then
13      visualise( $P$ );
        /* Recommend the top- $n$  modules with the highest
           similarity scores with respect to  $P$ . */
14       $\text{score}(m) = \text{lambda } m : \frac{1}{|P|} \sum_{p \in P} \text{cosine}(m, p)$ ;
15       $R = \text{Top}_n(M, \text{score})$ ;
16      recommend( $R$ );
17    end
18  end
19 end

```

We added *topic bubbles* to the visualisation in year 4, which is drawn from the use case presented in Section 5.4.1. The aim of these is to depict a potential specialisation/career goal. Students in the BSc Computer Science programme at UCD have the highest choice of optional modules in their last year. We aim to support students who might have a clear career/specialisation goal in mind with this functionality. The larger *topic bubbles* enclose all modules in the final year 4 that belong to the same latent topic. The students can click on a *topic bubble* if they are interested in modules that are

specifically important on the path to this career/specialisation goal. In Figure 6.12 we present the visualisation after the *topic bubble* for Topic 3 was clicked. Topic 3 shows a strong relationship to the field of Software Engineering, with modules like *Software Engineering Project 1, 2, and 3*, as well as *Software & Data Project* in the path for this specification. The lower part of the visualisation, the recommender and additional information panel, will also give slightly different information to the student when a *topic bubble* is clicked, emphasising the important modules in the path to this specialisation and giving specific links to the students to gain further information about this career option.

6.3 Live User Evaluation

In this section, we describe the results of a live-user study based on the experiences of undergraduate students at University College Dublin who were invited to use the system to inform and reflect on their own module choices.

6.3.1 Study Design

The browser-based study was designed in three stages. In the first part, students were asked several questions about their background, year of study, career goals, and familiarity with their module options. After completing this initial questionnaire, participants then used the visual module explorer and recommender after a short introductory explanation of how it worked. Finally, students were then asked to complete a second questionnaire to ascertain their experience using the system and its perceived utility.

The primary aim of the study was to answer a number of research questions, including (i) the clarity of their career goals; (ii) their level of familiarity with the modules offered as part of their programme of study; (iii) the quality of their experience when using the module advisor system; (iv) their improvement in knowledge about their programme structure and module options after using the system; and (v) how likely it would be that they use a similar system in the future again?

6.3.1.1 Surveys

The surveys were designed to collect very few personal (year of study and programme enrolled) and no demographic data to allow participants to express their options freely.



Figure 6.12: Interactive Visualisation with Active Topic Bubble.

We further worked closely with the School of Computer Science to comply with all ethical requirements. We decided not to ask any questions regarding specific modules or collect data that could be connected to specific modules or lecturers.

In total, we asked 21 questions, 11 questions in the pre-study survey and ten questions in the post-survey question. We split the questions in each survey into three categories, namely "General" (Questions Pre 1.1 to Pre 1.4 and Question Post 1.1), "Programme Structure" (Questions Pre 2.1 to Pre 2.3 and Questions Post 2.1 to Post 2.5), and "Elective Modules" (Questions Pre 3.1 to Pre 3.4 and Questions Post 3.1 to Post 3.4). The questions and answers to the Pre-Study Survey and Post-Study Survey can be seen in Table 6.1 and Table 6.2 respectively.

6.3.1.2 Visualisation & Web Application Technical Details

The visualisation presented previously was embedded in a web application framework that allowed us to present the visualisation to the student and provide them with the interactivity implemented.

The web application was implemented using the `Flask`³ micro framework, which depends on the `Werkzeug`⁴ utility library, a Web Server Gateway Interface (WSGI), and the `Jinja`⁵ template engine. `Flask` allows us to build Python based websites and supports a wide variety of extensions. For example the surveys were created using the `FlaskWTF`⁶ and the `WTForms`⁷ extensions. We used `PostgreSQL`⁸ as our relational database management system and connected it to our `Flask` framework using `SQLAlchemy`⁹ and `Flask-SQLAlchemy`¹⁰. The visualisation was realised using `Plotly`¹¹ with `Plotly Dash`¹² for the interaction functionalities. Standard Web Development tools, that is HTML and JavaScript were used for the website functionalities and classic CSS as well as the `Flask-Bootstrap`¹³ extensions was used for styling. The user study was hosted by `Heroku`¹⁴, a cloud platform as a service provider.

³<https://flask.palletsprojects.com/en/2.0.x/>

⁴<https://werkzeug.palletsprojects.com/en/2.0.x/>

⁵<https://jinja.palletsprojects.com/en/3.0.x/>

⁶<https://flask-wtf.readthedocs.io/>

⁷<https://wtforms.readthedocs.io/>

⁸<https://www.postgresql.org/>

⁹<https://www.sqlalchemy.org/>

¹⁰<https://flask-sqlalchemy.palletsprojects.com/en/2.x/>

¹¹<https://plotly.com/python/>

¹²<https://plotly.com/dash/>

¹³<https://pythonhosted.org/Flask-Bootstrap/>

¹⁴<https://www.heroku.com/>

	Question	Answer
<i>Pre 1.1</i>	Which programme are you currently enrolled in?	BSc Computer Science, BSc Computer Science w/ Data Science, Other
<i>Pre 1.2</i>	What year are you currently in?	1, 2, 3, 4, 4+
<i>Pre 1.3</i>	What was your biggest motivation to choose the Bachelor programme you choose?	General Interest, Job Opportunities, CAO Points, Interest in specific CS stream, Other
<i>Pre 1.4</i>	Which statement best describes your ideas about career goals and specialisations ?	clear career goal, some idea, unsure, vague idea, no idea
<i>Pre 2.1</i>	How familiar are you with your programme structure (i.e. modules in upcoming terms, pre-/co-requisites)?	very familiar, familiar, unsure, unfamiliar, very unfamiliar
<i>Pre 2.2</i>	If you are unsure about the details of a module (e.g. availability, pre- and co-requisites), where would you go to find additional information?	School of Computer Science, Friends, UCD Website, Student Office, Module Catalogue, Other
<i>Pre 2.3</i>	How familiar are you with the different streams offered within Computer Science?	very familiar, familiar, unsure, unfamiliar, very unfamiliar
<i>Pre 3.1</i>	What is your greatest motivation when choosing elective modules?	personal interest, easy good grades, easy to pass, small time commitment, fits time schedule, preparation for upcoming modules, achieve specific goal, other
<i>Pre 3.2</i>	How did you find the elective modules you ended up taking?	School of Computer Science, Friends, UCD Website, Student Office, Module Catalogue, Other
<i>Pre 3.3</i>	How satisfied overall were you with your chosen elective modules in the past?	very happy, happy, unsure, unhappy, very unhappy
<i>Pre 3.4</i>	How well informed do you feel about the available elective modules?	very informed, informed, unsure, uninformed, very uninformed

Table 6.1: Questions in Pre-Study Survey.

	Question	Answer
<i>Post 1.1</i>	Please rate your overall experience using the programme visualisation.	very good, good, unsure, bad, very bad
<i>Post 2.1</i>	Would you say the programme structure visualisation was useful/informative to you?	very useful, useful, unsure, rather not useful, not useful at all
<i>Post 2.2</i>	Would you say you discovered new/surprising connections between modules?	agree completely, agree, unsure, rather disagree, disagree completely
<i>Post 2.3</i>	Did the clustering of the modules make sense to you?	a lot of sense, some sense, unsure, not a lot of sense, no sense at all
<i>Post 2.4</i>	Would you say your knowledge about the overall programme structure has improved?	high improvement, some improvement, unsure, small improvement, no improvement
<i>Post 2.5</i>	How likely is it that you would use a system like this to gain knowledge about upcoming modules and module paths?	very likely, likely, unsure, unlikely, very unlikely
<i>Post 3.1</i>	How would you rate the quality of the recommended elective modules?	very good, good, unsure, bad, very bad
<i>Post 3.2</i>	Were you aware of the elective modules that were presented to you?	aware of all, aware of some, unsure, unaware of most, unaware of all
<i>Post 3.3</i>	How likely is it that you would consider one of the modules as your elective module?	very likely, likely, unsure, unlikely, very unlikely
<i>Post 3.4</i>	How likely is it that you would use a system like this to find elective modules in the future?	very likely, likely, unsure, unlikely, very unlikely

Table 6.2: Questions in Post-Study Survey.

6.3.2 Research Questions

Following, we will evaluate the results of the user study. For this, we consider two user study-hypotheses (UH), which together support the analysis of our thesis Hypothesis 3, as presented in Chapter 1.4:

- UH1: Students can benefit from online academic advising tools and are receptive to use such a system.
- UH2: Students who feel less informed about their programme structure or have lesser clarity of their career path options can benefit more from online academic advising.

We will evaluate in two parts: Firstly, we present overall participation results and answer some research questions we established earlier in Section 6.3.1. Secondly, we will discuss an in-depth analysis of the results regarding the two hypotheses.

To simplify the analysis of some questions, we convert the responses (of questions with a 5 Point Likert Scale answer range) to calculate a *Relative Benefit Score (RBS)*, as shown in Equation 6.2, which calculates the proportion of positive answers (< 3) for each question; e.g. $RBS = 0.8$ indicates that 80% of the answers were scored as either a 1 (*very high*) or a 2 (*high*) on the Likert scale.

$$RBS(Q_i, S_i) = \frac{|\{s : s < 3\}|}{|S_i|} \quad (6.2)$$

where $S_i = 1, 2, 3, 4, 5$, i.e. the set of possible responses to a question.

6.3.3 Participants

The user study was sent via email to all current 523 UCD Computer Science undergraduate students enrolled in the 2020/2021 term. We received participation of approximately 19% of the student body. However, due to the nature of the study consisting of three consecutive parts, we had a dropout rate of approximately 60% of the participants. That means that not every one of the 97 participants completed all three parts of the user study (pre-study survey, interactive visualisation, and post-study survey). Therefore, we define three different sets of participants according to their completion of the study: (i) participants who only answered the pre-study survey, (ii) participants who answered the pre-study survey and interacted at least once with the interaction, and (iii) participants who answered both surveys and interacted with the visualisation

at least once. In Table 6.3 we define the three sets of participants that we will use in the following evaluation.

Set	#	%	completed parts
1	97	18.55%	pre-study survey
2	62	11.85%	pre-study survey + >1 interaction w/ visualisation
3	45	8.6%	pre-study survey + >1 interaction w/ visualisation + post-study survey

Table 6.3: User Study: Participants Completion Statistics.

Half of our participants are in their first two years of study, approximately 27% in each year. The highest participation ($\sim 35\%$) was received from students in their third year. Unsurprisingly, we received the least participation from students in their final year ($\sim 11\%$), see Figure 6.13. In Figure 6.14 we see the completion rate (i.e. completed all three parts) per year. Participants in the first and final year are the most likely to complete the user study with a $\sim 54\%$ and $\sim 45\%$ completion rate. The majority of our participants (81%) are enrolled in the *BSc Computer Science* programme with the other 19% being enrolled in the *BSc Computer Science with Data Science* programme, see Figure 6.15.

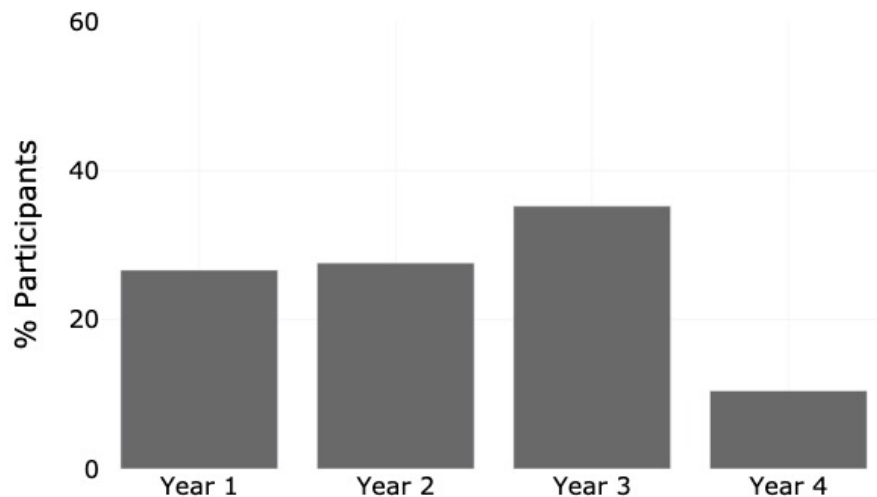


Figure 6.13: User Study: General Participation Distribution by Year.

The following section will examine the students' initial knowledge and experiences as stated in their pre-study survey.

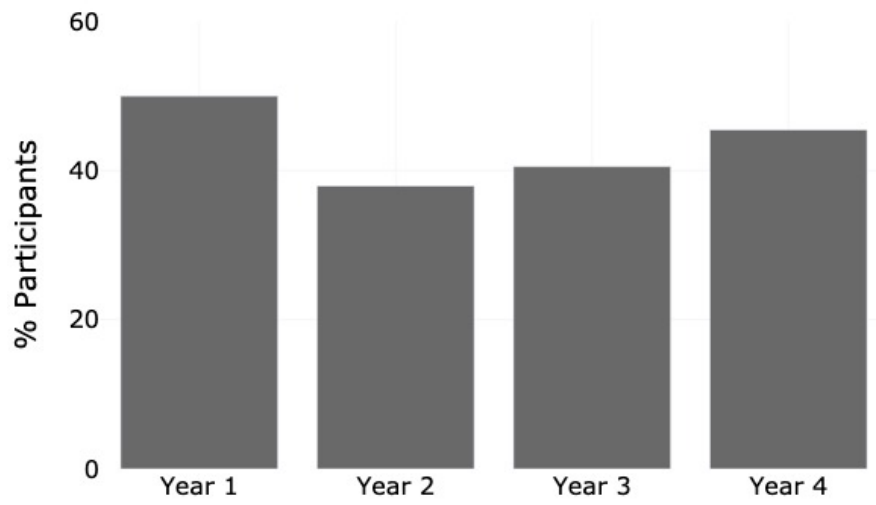


Figure 6.14: User Study: Completion Rate by Year.

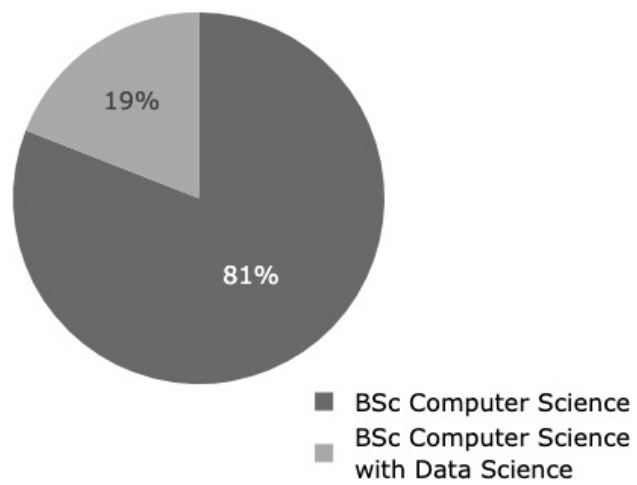


Figure 6.15: User Study: General Participation Distribution by Enrolled Programme.

6.3.4 Pre-Study Survey

One of the primary goals of this study was to evaluate the current state of students' knowledge regarding their programme structure and experience with their module choices. In this section, we will evaluate the participants' answers to the question in the pre-study survey.

We present the answers to Questions Pre 1.3 and Pre 1.4 in Figures 6.16 and 6.17 respectively. We can see that most participants ($\sim 57\%$) choose a degree in Computer Science out of General Interest, followed by $\sim 30\%$ of participants choosing it because of good job opportunities after graduation. In Figure 6.17 we can see that participants are split evenly when it comes to the clarity of their career goals. Approximately 50% of the participants stated that they have a *clear* or *general idea*, whereas the other half stated *unsure*, *vague idea*, or *no idea*.

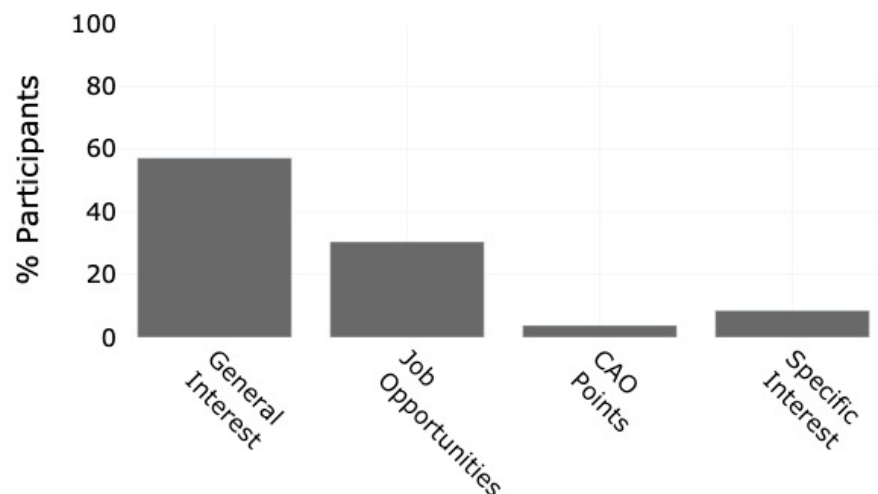


Figure 6.16: User Study: Answers to Question Pre 1.3 (*What was your biggest motivation to choose the Bachelor programme you choose?*).

Regarding participants familiarity with the programme structure, we were pleasantly surprised that the majority of students stated a high general familiarity with their programme structure as well as their stream options, see Figure 6.18 and 6.20. From Question Pre 2.2, we can conclude the importance of the online information provided by UCD, as we can see nearly 75% of the participants stating that their primary source of information about their programme is the UCD website as well as the UCD module catalogue. Approximately 20% of participants stated that they get their information mainly from their peers, see Figure 6.19.

The third section of the pre-study survey asked the participants about their past motivation and satisfaction in selecting elective modules. We can see a similar trend in posit-

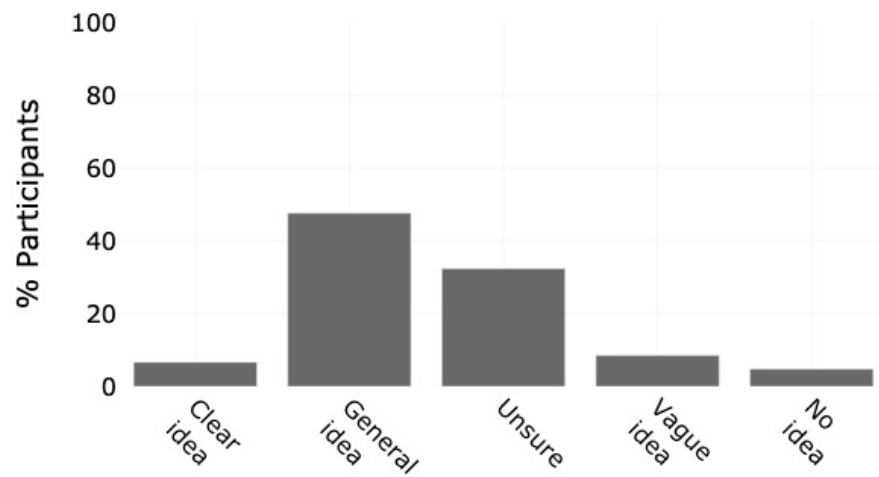


Figure 6.17: User Study: Answers to Question Pre 1.4 (Which statement best describes your ideas about career goals and specialisations ?).

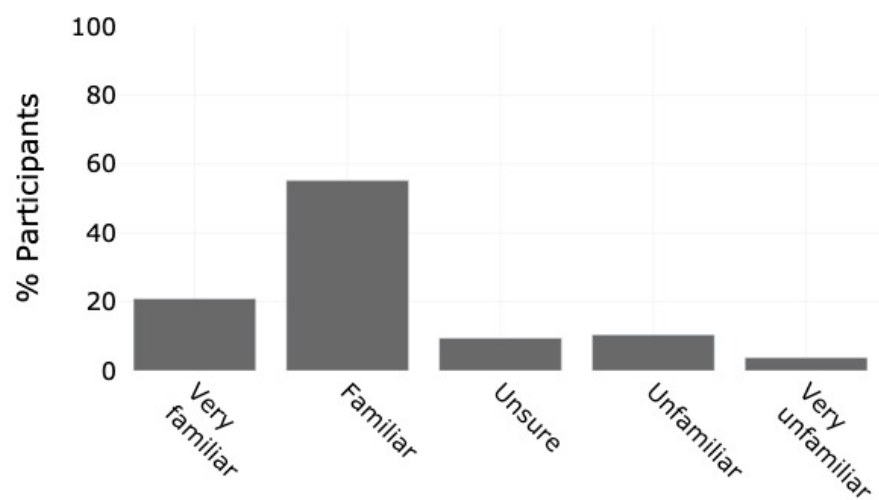


Figure 6.18: User Study: Answers to Question Pre 2.1 (How familiar are you with your programme structure (i.e. modules in upcoming terms, pre-/co- requisites)?).

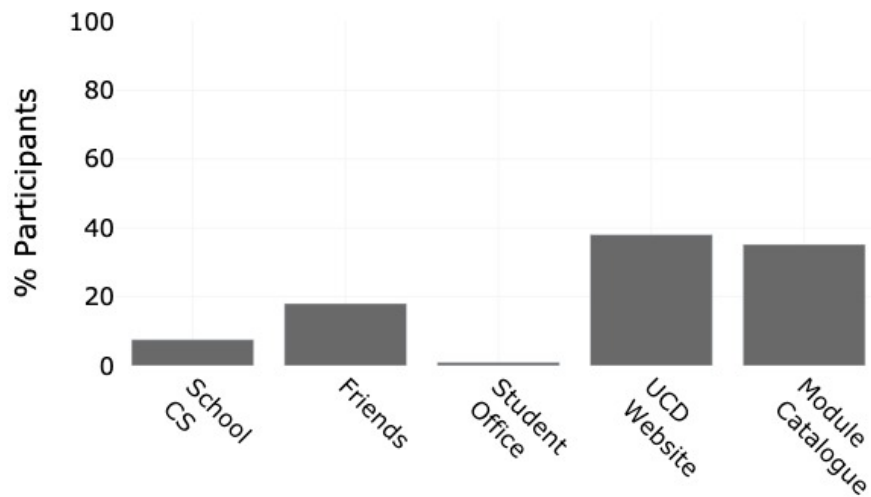


Figure 6.19: User Study: Answers to Question Pre 2.2 (*If you are unsure about the details of a module (e.g. availability, pre- and co-requisites), where would you go to find additional information?*).

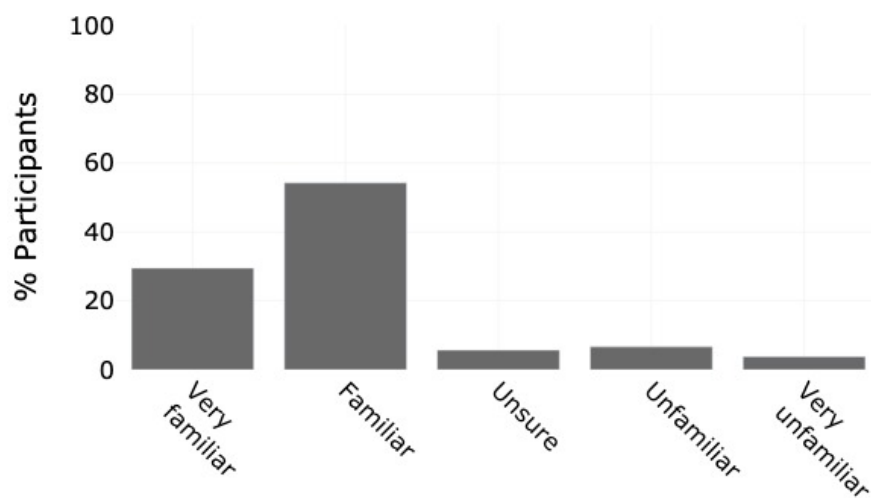


Figure 6.20: User Study: Answers to Question Pre 2.3 (*How familiar are you with the different streams offered within Computer Science?*).

ive answers. The majority of participants ($\sim 63\%$) stated they choose elective modules based on their personal interest (Figure 6.21), and were either *happy* or *very happy* with their choices, Figure 6.23. Again we can see the importance of the UCD website and the module catalogue specifically as nearly 85% stated that they found their information about elective modules there (Figure 6.22). Overall students feel *informed* (51.4%) or *very informed* (18.1%) about their module choices (Figure 6.24).

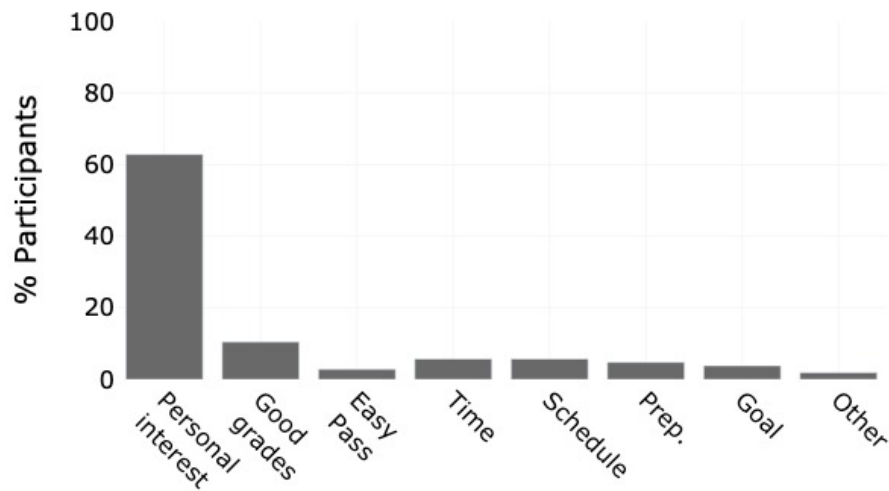


Figure 6.21: User Study: Answers to Question Pre 3.1 (*What is your greatest motivation when choosing elective modules?*).

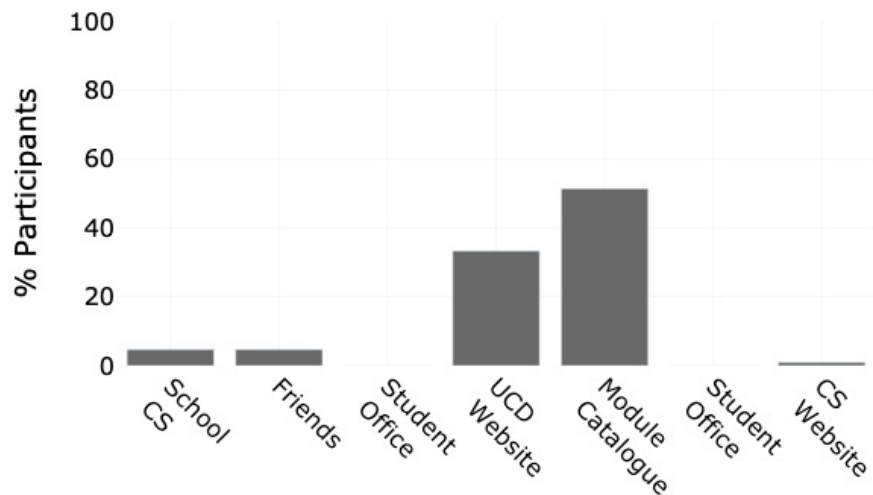


Figure 6.22: User Study: Answers to Question Pre 3.2 (*How did you find the elective modules you ended up taking?*).

From these results, we can conclude that the participating undergraduate Computer Science students overwhelmingly feel informed about their options as well as their overall programme structure. Furthermore, a surprisingly high amount of students

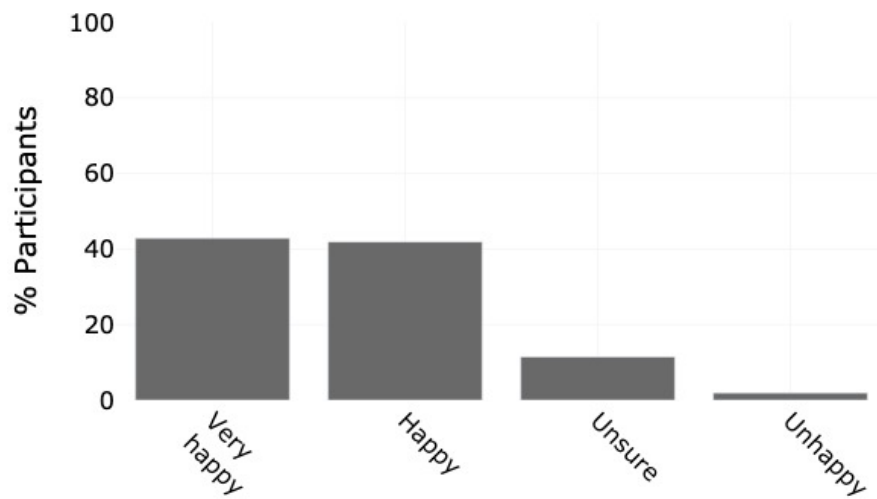


Figure 6.23: User Study: Answers to Question Pre 3.3 (*How satisfied overall were you with your chosen elective modules in the past?*).

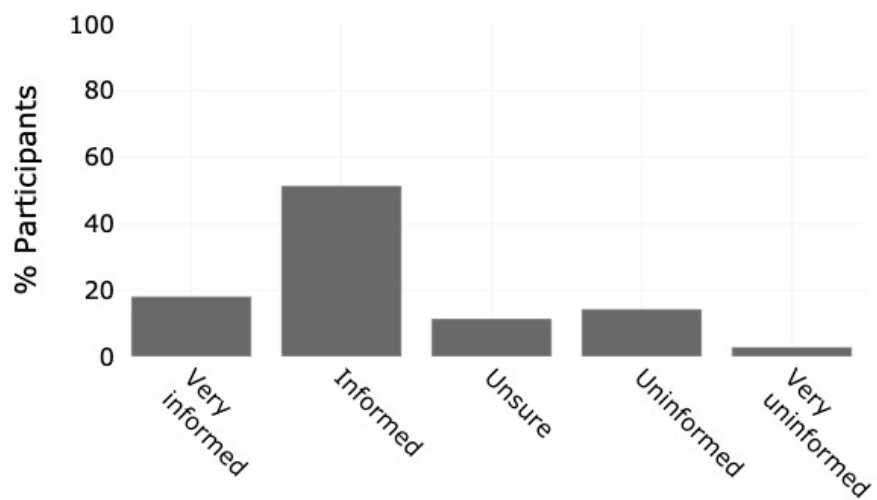


Figure 6.24: User Study: Answers to Question Pre 3.4 (*How well informed do you feel about the available elective modules?*).

choose modules out of personal interest, in contrast to our previous hypothesis, that students pick modules based on hearsay and "easy grades". In the following section, we will take an in-depth look at the results of the post-survey questionnaire, focusing on students' knowledge gain and overall experience with the presented system.

6.3.5 Analysis: Programme Structure

In this section, we will address the two main hypotheses of this evaluation, as described in Section 6.3.2. For this, we focus on three features of our post-survey questionnaire, namely the students' overall experience of the system (Post 1.1), their perceived knowledge gain after using the interactive visualisation system (Post 2.4), and their likeliness to reuse a system like this in the future (Post 2.5)

The relevant results from the post-questionnaire are presented in Figure 6.25, Figure 6.26, and Figure 6.27. They indicate: that most students (71.1%) show either a *high* or *some* level of improvement in their knowledge and understanding of the programme structure as a result of using the system (see Figure 6.25); that 82.2% of students found the overall experience to be *good* or *very good* (Figure 6.26); and that 66.7% of students indicated they would be *likely* or *very likely* to reuse the system again if offered (Figure 6.27).

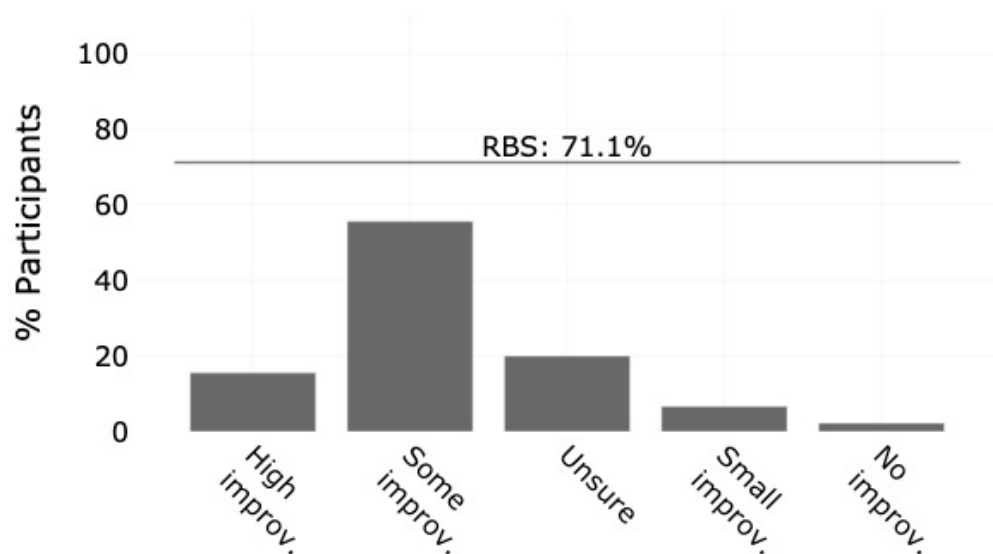


Figure 6.25: User Study: Answers to Question Post 2.4 (*Would you say your knowledge about the overall programme structure has improved?*).

While these results speak to the overall utility of the system from a student perspective — with a large majority of students positively disposed towards the system — they ig-

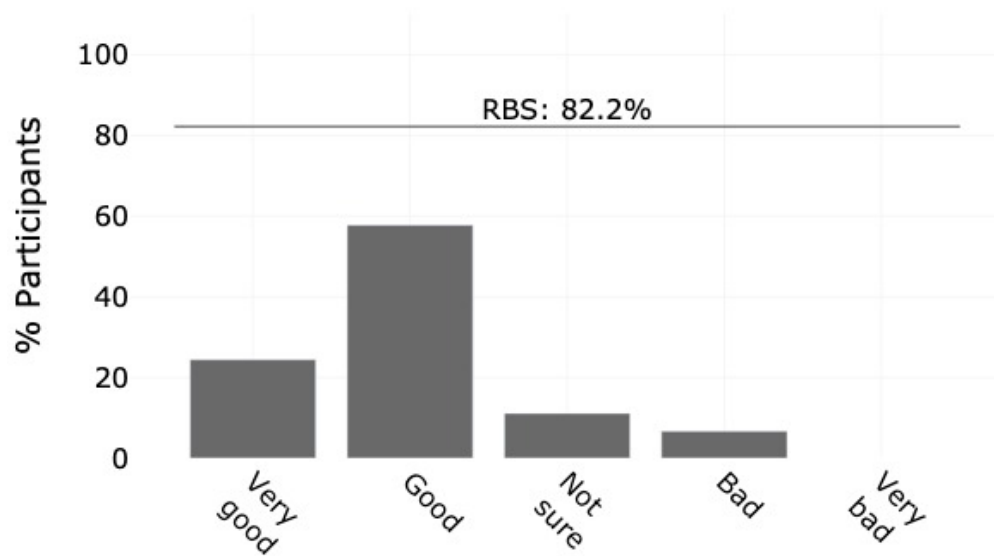


Figure 6.26: User Study: Question Post 1.1 (*Please rate your overall experience using the programme visualisation.*).

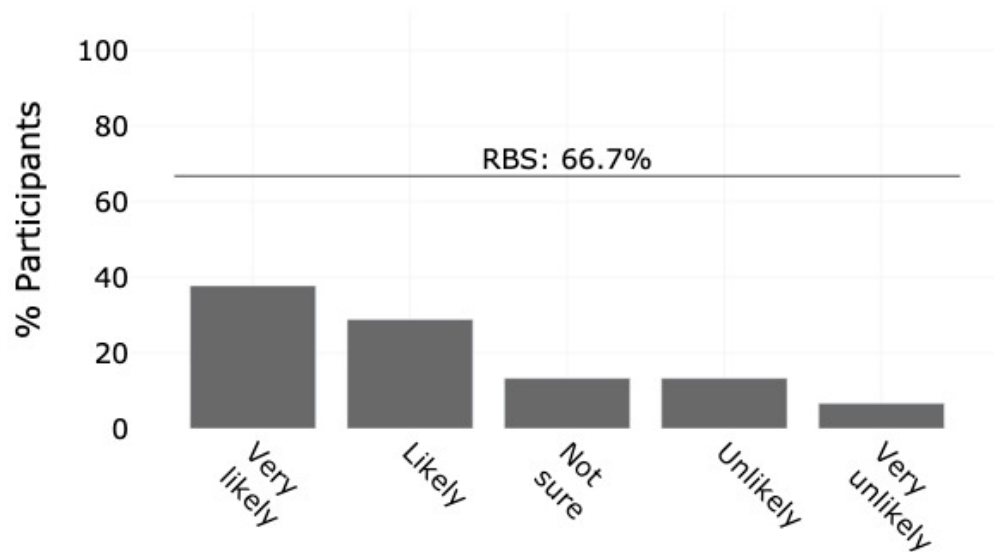


Figure 6.27: User Study: Question Post 2.5 (*How likely is it that you would use a system like this to gain knowledge about upcoming modules and module paths?*).

nore the different perspectives that students may have. For example, do students with different initial levels of knowledge regarding their programme structure and career path benefit differently? Conversely, are students with a high programme structure familiarity less likely to use an online academic advising system?

To answer such questions, we next look at a cohort-based analysis by dividing students into four distinct cohorts, as shown in Figure 6.28, based on their level of familiarity with the programme structure and the degree of clarity in their career goals. We can see that most students fall into one of three of these cohorts – Cohort 1 (*familiar/clear*) (20 students), Cohort 3 (*unfamiliar/unclear*) (10 students), or Cohort 4 (*familiar/unclear*) (13 students) – with only two students in Cohort 2 (*unfamiliar/clear*). This is perhaps not surprising as it indicates that students with clear career goals also have a good understanding of their programme structure and module choices. Due to the low numbers of participants in Cohort 2, we will omit those from further evaluation. The bar charts in Figure 6.29 report the RBS scores for the knowledge gain, user experience, and likelihood of reuse questions discussed in the previous section for Cohorts 1, 3, and 4.

In terms of knowledge gain, each of the cohorts shows a majority of students gaining some knowledge from using the system, even those in Cohorts 1 and 4, which reported as being familiar with programme structure, to begin with. The highest knowledge gain is associated with Cohort 3, which is perhaps not surprising since these students presented as being unfamiliar with programme structure and so had the most to gain from the system.

A similar trend can be seen in the RBS values for overall experience: Participants in Cohort 3 (*unfamiliar/unclear*) show the highest RBS of 90%, where as students in Cohort 1 (*familiar/clear*) are again slightly under the overall mean of 82.2% with an RBS of 80% and participants in Cohort 4 (*familiar/unclear*) showing an even lower score of 77%.

These results support our second hypothesis that students who are less informed or less clear about their career goals have more to gain from the advisory system than more informed/clear students. Somewhat surprisingly, Cohort 3 (*unfamiliar/unclear*) presents with a lower RBS (60%) for likeliness of reuse than Cohort 1 (*familiar/clear*) (75%). Cohort 4 (*familiar/unclear*) showing the lowest RBS value for this feature again, with still more than half (54%) of the students saying they would reuse the system. It seems as if students who have already have a clear career goal are especially interested in additional sources of information such as the presented system, while students who have a less clear goal might need additional directions and pointers to find the information they need.

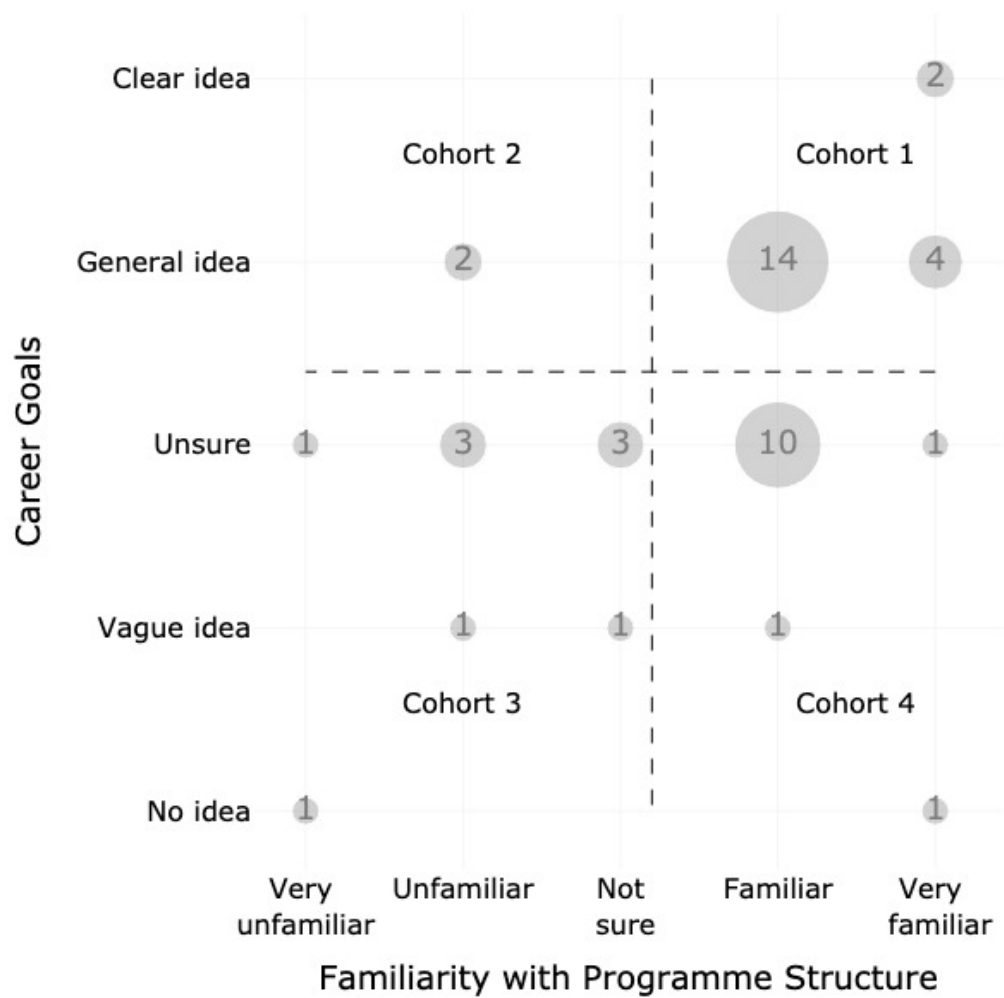


Figure 6.28: User Study: Cohort Definition.

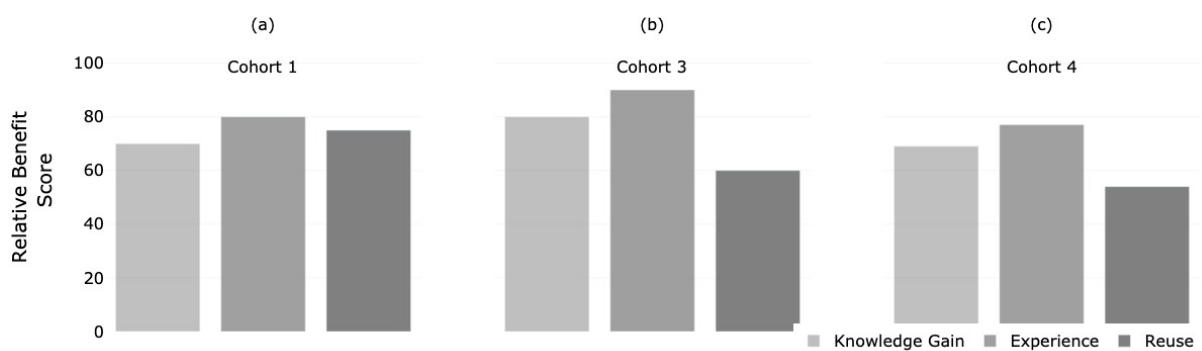


Figure 6.29: User Study: RBS Values for Key Features for Cohort 1 (a), Cohort 3 (b), and Cohort 4 (c).

6.3.6 Analysis: Elective Modules

This section of the chapter will take a detailed look at the participants concerning elective modules. In Section 6.3.4 we have concluded that the majority of students feels (very) informed about their elective module choices and were (very) happy about their electives in the past. Therefore, we hypothesised that students who feel less informed or are unhappy about their past choices will benefit from the recommender system for elective modules provided in the visualisation.

We present the results of participants' answers in the post-survey questionnaire about the quality of the recommended modules (Post 3.1), the likeliness of the participants choosing one of the recommendations as their elective module (Post 3.3), as well as the likeliness of participants to reuse a system like this to find elective modules (Post 3.4) are presented in Figure 6.30. Overall we can see a positive reaction of the participants to the presented elective modules. Over half of the participants rated the recommendations either good or very good, with nearly the rest of the participants being unsure about them. Even though students seemed unsure about some of the recommendations being made, a large number of participants (66.7%) stated that it is likely or very likely they would consider choosing one of the presented electives in a future year. A further 60% of participants stated that it would be (very) likely that they would reuse the system again in the future to find elective modules.

	<i>Post 3.1</i>	<i>Post 3.3</i>	<i>Post 3.4</i>
<i>Pre 3.3</i>	0.07	-0.03	0.20
<i>Pre 3.4</i>	0.06	0.09	-0.01

Table 6.4: Spearman Correlation between Questions *Pre 3.3/Pre 3.4* (*How satisfied overall were you with your chosen elective modules in the past?/How well informed do you feel about the available elective modules?*) and Questions *Post 3.1/Post 3.3/Post 3.4* (*How would you rate the quality of the recommended elective modules?/How likely is it that you would consider one of the modules as your elective module?/How likely is it that you would use a system like this to find elective modules in the future?*).

Our results show that students who stated a high familiarity with their elective module choices do not significantly correlate with their overall likeliness of choosing any of the recommended electives nor the likeliness of reuse. Neither do students who stated to be less satisfied with their elective modules in the path show any significantly increased interest in the elective recommender system compared to satisfied students. We present the Spearman correlation [198] results in Table 6.4.

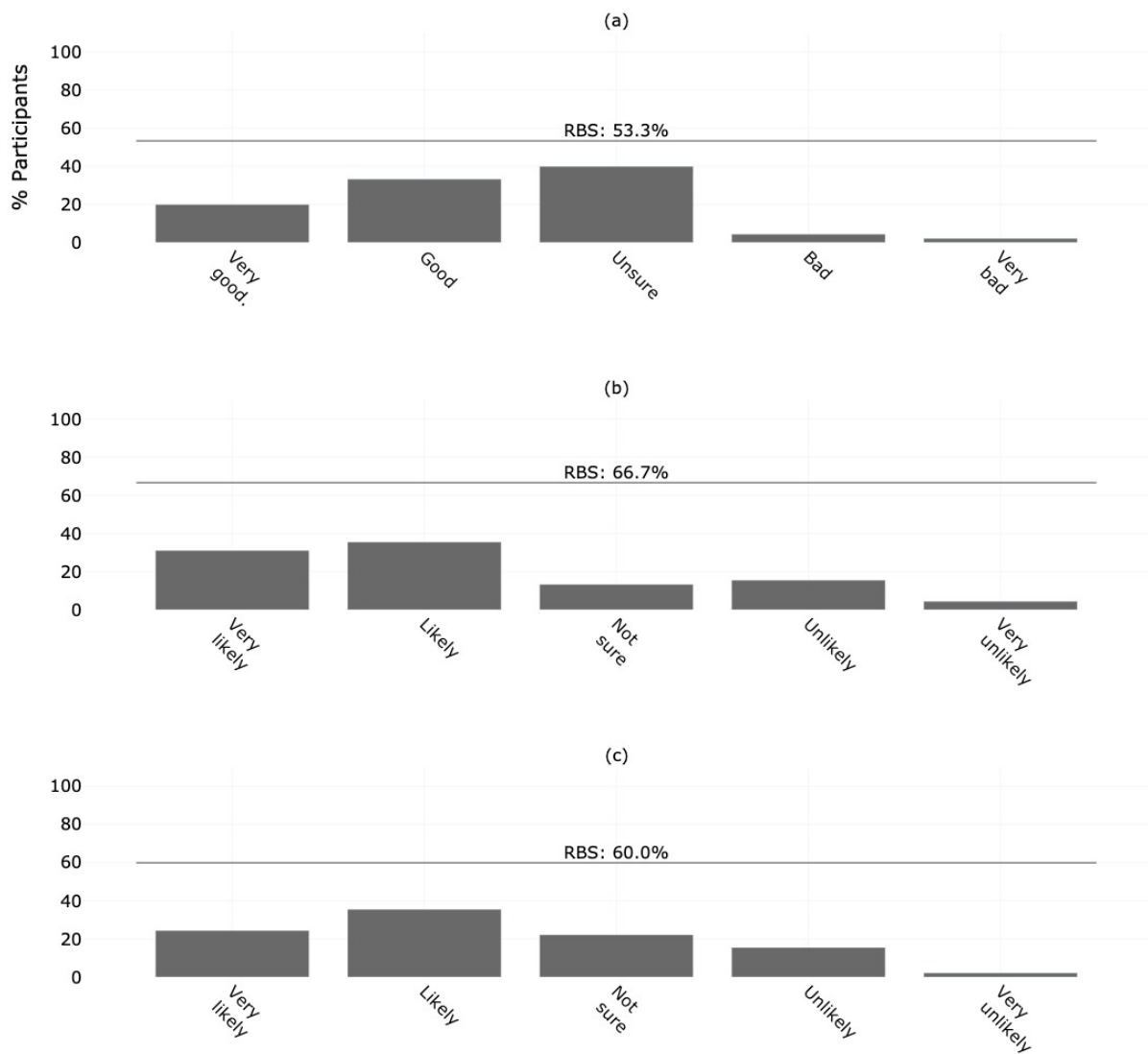


Figure 6.30: User Study: Answers to Question Post 3.1(a) (*How would you rate the quality of the recommended elective modules?*), Question Post 3.3(b) (*How likely is it that you would consider one of the recommended modules as your elective module?*), and Question Post 3.4(c) (*How likely is it that you would use a system this this to find elective modules in the future?*)

We can conclude that most participants enjoyed the elective module recommendations and that they would reuse the system to find suitable modules in the future, independently from their initial knowledge or satisfaction with their previous choices.

6.4 Conclusion

In this chapter, we presented an advisory system consisting of a visual module exploration tool and a module recommender system aimed to support students in exploring their academic module space. We implemented a web-based user study framework and conducted a live-user study involving undergraduate Computer Science students at University College Dublin.

Overall, we conclude that the results of this user study support both of our principal hypotheses. In general, students reported a benefit from using the system in terms of the knowledge they gained, the overall experience offered, and their likelihood to reuse such a system is rolled out. Moreover, students that were less familiar with their programme's structure and who had unclear career goals, gain more knowledge than students who were familiar with their programme's structure or who did have clear career goals.

However, it is worth noting that even students who were already familiar with their programme's structure or had clear career goals still benefited in terms of knowledge gain, user experience, and likelihood of reuse. Going into this study, we felt that these students might not benefit to any significant degree from the visual module explorer provided if they already knew about the modules on offer and their connections over the years. Nevertheless, the results indicate that these well-informed students also benefited from the recommendations provided.

Regarding the elective module recommender system, we were able to show a similar trend. The majority of participants enjoyed the recommender system functionality and would consider choosing one of the recommended electives in the future. The participants seemed receptive to the presented system and showed a high likeliness of reuse. We showed that students' reception of the system was independent of their satisfaction with previously chosen elective modules. We showed that there is no significant correlation between a student's initial knowledge about elective module options or past module satisfaction and their enjoyment of the system, likeliness to choose a recommended module and likeliness of reuse.

That being said, it is likely that the system could be fine-tuned or adapted to provide different types of students with different types of advisory feedback. For example, it

should be possible to prioritise or emphasise information about programme structure or career opportunities in order to optimise the recommendations and support that each student receives. Furthermore, we propose to trial the system on different programmes to investigate if students in programmes with more flexible structures might benefit differently.

CONCLUSION

The increase in online and blended learning has lead to a variety of new educational data readily available. This has allowed an exciting new research area to emerge. Educational Data Mining, Learning Analytics and related fields have shown how technology can support different stakeholders in the educational environment. In this work, we focused on contributing to the research area by exploring how content-based recommender system approaches can improve the module recommendation task, its capability for module exploration and its impact on recommendation diversity.

In the following, we will provide an outline of the key contributions presented in this research and conclude this work by giving an outlook for future research.

7.1 Summary of Contributions

Throughout this work, we aimed to build a recommender system that can support the students and the university alike. Consequently, we diverged from the majority of work done in the area by concentrating on module content rather than student and performance data. Our primary focus was to provide students with the knowledge and information needed to make their own informed decisions when choosing the right path in their academic life. Therefore the distinct directions of *Module Exploration* and personalised and diverse *Elective Module Recommendations* have driven this work.

7.1.1 Module Exploration

From early on in our work, we focused on content-driven module representations and recommendations, arguing that the modules' content has a high potential for generating useful information. At the same time, we highlighted the importance of diversity to support discoverability of long-tail options. Not only do students benefit from the ad-

ditional knowledge that allows them to find modules that are uniquely suited to their strengths and interest, but at the same time, we can ease the pressure on academic planning by spreading out the allocation applications more evenly. Additionally, this could lead to a lower percentage of students not being allocated to their first or second choices.

We started exploring module representations by building a hybrid model based on textual descriptors of modules. The recommender system computes similarities based on explicit students preferences. First, we introduce *exploration* by allowing the students to select and deselect their preference modules actively and iteratively, showing instant changes in the set of recommended electives. Secondly, the students have complete control over the diversity introduced into the recommendations by changing the *Discovery* slider. The implemented prototype presented how this approach can be visualised, giving students the highest possible amount of control over their preferences and choices, eliciting a sense of exploration and explanation for the recommended modules. In an offline evaluation, we showed that the content-based approach alone can outperform the collaborative filtering approach in terms of diversity of modules and topics as well as similarity to the students' interests. We can further improve the diversity by introducing the university's taxonomy into the recommendation process. We, therefore, conclude that these findings support our Hypothesis 1 (*Textual module descriptors can be used to build rich content-based recommender systems, and the introduction of diversity improves the discoverability of long-tail options.*).

Subsequently, we focused on the interconnectivity between modules. This part of our research was driven by the sequential nature of modules paths within university programs and their potential lack of explicit requirement statements. We were able to show that matrix factorisation models can sufficiently detect dependencies between modules in module paths. This finding allows us to model paths for module exploration in even more detail as well as incorporate additional use cases. In a visualisation prototype, we presented how these dependencies can be used to visualise connections between modules for multiple use cases. In support of Hypothesis 2 (*Unobserved variables can be used to detect connections between modules that allow us to build sequential visual module exploration models.*), we can conclude that our evaluation results using an expert ground truth showed favourable results regarding the detection of implicit dependencies. Further, we were able to derive different use cases and suitable visualisation options.

We combined the two approaches to build an academic advisor and recommender system in the final part of our work. The system allows for module exploration by presenting students with an interactive visualisation of their programme space. We

use dependencies between modules to depict connections and requirements between modules in consecutive years. Students can freely explore different paths through their programme and gain knowledge about possible career and specialisation options. The recommender system is designed to support specific module paths by recommending compatible elective module options. The online user study showed that we can improve students' knowledge about their programme structure and elective module options independently of students' incipient knowledge and goals. The user study results support our third hypotheses (*Content-based module similarities can be used to create models that help students' module exploration, elective module recommendations and improve student knowledge about their module space.*), showing that the participating students gained knowledge about their module space and elective module options. We were further able to support our hypotheses regarding students' willingness to adapt to online academic advising systems, showing that the majority of participants stated satisfaction with the presented system and a high likelihood of reuse.

7.1.2 Elective Module Recommendations

Our focus to create an elective module recommender system was instigated by the need for students to be able to find uniquely suitable modules as well as promoting lesser-known modules across the university to ease allocation issues. We motivated the requirements for a support system for students in finding elective modules by presenting results from a previous data analysis that showed the stark increase of students taking less diverse modules in recent years. While students stated that they were overwhelmingly satisfied with their previously chosen modules as presented in Section 6.3, the results of the user study showed that we were able to increase students' knowledge about their elective module options and that we were able to provide students with elective modules that they were not aware of. This supports our first hypothesis, showing us that content-based recommender system approaches are able to provide diverse and serendipitous recommendations.

There is no shortage of previous work done building elective module recommender systems. However, the majority of those are based on and predicting recommendations by their calculated grade. We were able to show that we can provide meaningful elective module recommendations by purely using textual module descriptors. Our initial research showed that we can improve the discoverability of long-tail options by introducing diversity using the university taxonomy. However, even without introducing diversity into our recommender system, we were able to show that pure content-based approaches are able to recommend a diverse set of recommendations.

Using matrix factorisation approaches, we were able to find meaningful dependencies and similarities between modules from the specific programme structure and a set of recommendations that spans multiple schools and institutions.

Overall, we conclude that by providing students with the right tools and information, we can improve their awareness and knowledge about their options when it comes to choosing elective modules. This additional knowledge allows students to make more informed decisions and help them carve their own personal path through their academic and further professional careers.

7.2 Future Directions, Limitations and Open Questions

Academic advising proves to be a vast area with a multitude of application options and approaches. We have shown that content-based approaches to module exploration and module recommendation can achieve great results. Nevertheless, there are still open issues and many directions for future work. For example, further, an improvement could be made in optimising recommendations for specific subsets of students and situations, improving technical approaches by expanding textual descriptors to enhance recommendations and improving explanations.

7.2.1 Optimising Recommendations

There are multiple ways we can optimise the recommendations. Firstly, we have seen that there are very different types of students; some might have a clear career goal, while others still explore different paths. Further, students in their first year might require different support than students in their later years of study. In future work, an in-depth analysis of different student needs might yield additional insight that would allow for a specified approach in module recommendations.

Secondly, when choosing modules, students should choose modules that uniquely fit their interests and strengths and support their academic path; however, organisational and preference constraints, such as timetable clashes, preferences in time, lecturer, and assessment type, play a critical role in creating their academic path. We can imagine including a personalised timetable planner that includes a recommender system as presented in our work, in addition to multiple optimisations and organisational options for students to create their most efficient timetable each year.

Lastly, in our work, we highly focused on students as our primary stakeholders. However, an interesting perspective for future work could be to adapt our approaches to

serving other stakeholders, such as instructors and institutions. For example, an attractive study could include the effect that module recommendations can have on the distribution of students in elective modules and the impact on organisational resource allocation.

7.2.2 Textual Descriptor Enhancement

Throughout our work, we have used the freely available module descriptors presented on the UCD Website. In multiple analyses, we have seen a significant difference in their quality; some descriptions might be very short or ambiguous. While our results show that the content-based approach works satisfactorily on these descriptors, there are multiple directions for future work to improve the technical approach, help instructors write better descriptors, and improve the overall quality of information and recommendations.

An interesting future directive could lie in enhancing module descriptors artificially. We detected that the average textual description of a module is relatively short and further declines after cleaning the text. Data augmentation has shown significant benefits in machine learning, especially in Computer Vision. Natural Language approaches in data augmentation have shown promising results and could prove beneficial in enhancing module descriptors to improve techniques presented here. Other enhancement options, such as identifying and including tags and keywords, are further future work options.

Further, it would be very interesting to analyse the quality of module descriptors in more detail. The results might present insights into what factors of a description allows a recommender system to match that module well. This knowledge might be used to help instructors to write more meaningful and impactful module descriptors that will help feed the recommender system engine to produce more accurate module dependencies and recommendations.

7.2.3 Explaining Recommendations

The work presented in this thesis has argued about the importance of explanations in recommender systems severalfold. While we included the notion of explanations through visualisation, interactivity and self-led discovery, we acknowledge an excellent potential for future work in this area.

Explanations have shown repeatedly that they improve the trust and satisfaction of users with the system. Especially in recommendation scenarios where the cost of a wrong decision is high, it is crucial to provide users with suitable explanations to soothe their reservations and allow them to make the best-informed decision possible. Therefore, an interesting future directive is to include more definitive explanations and analyse the difference in explanations forms. Especially the difference between textual and visual explanations for module recommendations could substantially impact the EDM/LA research community.

BIBLIOGRAPHY

- [1] A. Acharya and D. Sinha. Early Prediction of Students Performance using Machine Learning Techniques. *International Journal of Computer Applications*, 107(1):975–8887, 2014.
- [2] P. Adamopoulos and A. Tuzhilin. On unexpectedness in recommender systems: Or how to better expect the unexpected. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(4):1–32, 2014.
- [3] 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.
- [4] R. Agrawal, S. Gollapudi, A. Halverson, and S. Jeong. Diversifying search results. In *Proceedings of the second ACM international conference on web search and data mining*, pages 5–14, 2009.
- [5] V. V. Ajanovski. Guided exploration of the domain space of study programs. In *4th Joint Workshop on Interfaces and Human Decision Making for Recommender Systems (IntRS) 2017*, volume 43, 2017.
- [6] P. Ajith, M. S. S. Sai, and B. Tejaswi. Evaluation of student performance: an outlier detection perspective. *Int. J. Innov. Technol. Explor. Eng*, 2(2):40–44, 2013.
- [7] Y. Al-Ashmoery, R. Messoussi, and R. Touahni. Analytical tools for visualisation of interactions in online e-learning activities on lms and semantic similarity measures on text. *Journal of Theoretical & Applied Information Technology*, 73(1), 2015.
- [8] A. Al-Badarenah and J. Alsakran. An automated recommender system for course selection. *International Journal of Advanced Computer Science and Applications*, 7(3):166–175, 2016.
- [9] M. T. Al-Nory. Simple decision support tool for university academic advising. In *2012 International Symposium on Information Technologies in Medicine and Education*, volume 1, pages 53–57. IEEE, 2012.
- [10] M. Z. Al-Taie and S. Kadry. Visualization of explanations in recommender systems. *Journal of Advanced Management Science Vol*, 2(2):140–144, 2014.

- [11] H. Aldowah, H. Al-Samarraie, and W. M. Fauzy. Educational data mining and learning analytics for 21st century higher education: A review and synthesis. *Telematics and Informatics*, 37:13–49, 2019.
- [12] S. Anuvareepong, S. Phooim, N. Charoenprasoplar, and S. Vimonratana. Course recommender system for student enrollment using augmented reality. In *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*, pages 212–217. IEEE, 2017.
- [13] C. Armatas and C. F. Spratt. Applying learning analytics to program curriculum review. *The International Journal of Information and Learning Technology*, 2019.
- [14] M. Z. Aslam, Nasimullah, and A. R. Khan. A Proposed Decision Support System/Expert System for Guiding Fresh Students in Selecting a Faculty in Gomal University, Pakistan, 2012.
- [15] R. Baeza-Yates, B. Ribeiro-Neto, and Others. *Modern information retrieval*, volume 463. ACM press New York, 1999.
- [16] R. S. Baker and P. S. Inventado. Educational data mining and learning analytics. In *Learning analytics*, pages 61–75. Springer, 2014.
- [17] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel. Educational data mining applications and tasks: A survey of the last 10 years. *Education and Information Technologies*, 23(1):537–553, 2018.
- [18] L. Baltrunas, B. Ludwig, and F. Ricci. Matrix factorization techniques for context aware recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 301–304, 2011.
- [19] M. B. Baxter Magolda. Self-authorship: The foundation for twenty-first-century education. *New directions for teaching and learning*, 2007(109):69–83, 2007.
- [20] T. Beaubouef and J. Mason. Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin*, 37(2):103–106, 2005.
- [21] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Seventh IEEE international conference on data mining (ICDM 2007)*, pages 43–52. IEEE, 2007.
- [22] K. Bhumichitr, S. Channarukul, N. Saejiem, R. Jiamthapthaksin, and K. Nongpong. Recommender Systems for university elective course recommendation. In *2017 14th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 1–5. IEEE, 2017.
- [23] M. Bilgic and R. J. Mooney. Explaining recommendations: Satisfaction vs. promotion. In *Beyond personalization workshop, IUI*, volume 5, page 153, 2005.
- [24] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.

- [25] Q. E. Booker. A student program recommendation system prototype. *Issues in Information Systems*, pages 544–551, 2009.
- [26] T. Boongoen and N. Iam-on. Improved student dropout prediction in Thai University using ensemble of mixed-type data clusterings. *International Journal of Machine Learning and Cybernetics*, pages 497–510, 2017.
- [27] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *Conference on Uncertainty in Artificial Intelligence*, 2013.
- [28] P. Brusilovsky, J. Eklund, and E. Schwarz. Web-based education for all: a tool for development adaptive courseware. *Computer networks and ISDN systems*, 30(1-7):291–300, 1998.
- [29] I. Buciu. Non-negative matrix factorization, a new tool for feature extraction: theory and applications. *International Journal of Computers, Communications and Control*, 3(3):67–74, 2008.
- [30] S. Buckingham Shum, M. Hawksey, R. S. J. D. Baker, N. Jeffery, J. T. Behrens, and R. Pea. Educational data scientists: a scarce breed. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 278–281, 2013.
- [31] R. Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, pages 1–29, 2002.
- [32] R. Burke. *Hybrid Web Recommender Systems*, pages 377–408. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
- [33] J. P. Campbell. *Utilizing student data within the course management system to determine undergraduate student academic success: An exploratory study*. Purdue University, 2007.
- [34] J. P. Campbell, P. B. DeBlois, and D. G. Oblinger. Academic analytics: A new tool for a new era. *EDUCAUSE review*, 42(4):40, 2007.
- [35] E. Çano and M. Morisio. Hybrid recommender systems: A systematic literature review. *Intelligent Data Analysis*, 21(6):1487–1524, 2017.
- [36] L. N. Cassel, S. Palivela, S. Marepalli, A. Padyala, R. Deep, and S. Terala. The new acm ccs and a computing ontology. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL '13*, page 427–428, New York, NY, USA, 2013. Association for Computing Machinery.
- [37] P.-C. Chang, C.-H. Lin, and M.-H. Chen. A hybrid course recommendation system by integrating collaborative filtering and artificial immune systems. *Algorithms*, 9(3):47, 2016.
- [38] B. Cheang, A. Kurnia, A. Lim, and W.-C. Oon. On automated grading of programming assignments in an academic institution. *Computers & Education*, 41(2):121–131, 2003.

- [39] D.-Y. Chiu, Y.-C. Pan, and W.-C. Chang. Using rough set theory to construct e-learning faq retrieval infrastructure. In *2008 First IEEE International Conference on Ubi-Media Computing*, pages 547–552. IEEE, 2008.
- [40] O. Conlan, V. Wade, C. Bruen, and M. Gargan. Multi-model, metadata driven approach to adaptive hypermedia services for personalized elearning. In *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 100–111. Springer, 2002.
- [41] G. Cosma and M. Joy. An approach to source-code plagiarism detection and investigation using latent semantic analysis. *IEEE transactions on computers*, 61(3):379–394, 2011.
- [42] M. Cucuringu, C. Z. Marshak, D. Montag, and P. Rombach. Rank aggregation for course sequence discovery. In *International Conference on Complex Networks and their Applications*, pages 139–150. Springer, 2017.
- [43] G. W. Dekker, M. Pechenizkiy, and J. M. Vleeshouwers. Predicting Students Drop Out: A Case Study. *International Working Group on Educational Data Mining*, 2009.
- [44] D. Delen. A comparative analysis of machine learning techniques for student retention management. *Decision Support Systems*, 49(4):498–506, 2010.
- [45] S. Deorah, S. Sridharan, and S. Goel. SAES-expert system for advising academic major. In *2010 IEEE 2nd International Advance Computing Conference (IACC)*, pages 331–336. IEEE, 2010.
- [46] M. Diaby, E. Viennet, and T. Launay. Toward the next generation of recruitment tools: an online social network-based job recommender system. In *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, pages 821–828. IEEE, 2013.
- [47] A. Djulovic and D. Li. Towards Freshman Retention Prediction : A Comparative Study. *International Journal of Information and Education Technology*, 3(5), 2013.
- [48] H. Drachsler, K. Verbert, O. C. Santos, and N. Manouselis. Panorama of recommender systems to support learning. In *Recommender systems handbook*, pages 421–451. Springer, 2015.
- [49] J. K. Drake. The role of academic advising in student retention and persistence. *About Campus*, 16(3):8–12, 2011.
- [50] X. Du, J. Yang, J.-L. Hung, and B. Shelton. Educational data mining: a systematic review of research and emerging trends. *Information Discovery and Delivery*, 2020.
- [51] A. Elbadrawy and G. Karypis. Domain-aware grade prediction and top-n course recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 183–190, 2016.
- [52] A. Elbadrawy, A. Polyzou, Z. Ren, M. Sweeney, G. Karypis, and H. Rangwala. Predicting student performance using personalized analytics. *Computer*, 49(4):61–69, 2016.

- [53] G. Engin, B. Aksoyer, M. Avdagic, D. Bozanlı, U. Hanay, D. Maden, and G. Ertek. Rule-based expert systems for supporting university students. *Procedia Computer Science*, 31:22–31, 2014.
- [54] M. Erdt, A. Fernandez, and C. Rensing. Evaluating recommender systems for technology enhanced learning: a quantitative survey. *IEEE Transactions on Learning Technologies*, 8(4):326–344, 2015.
- [55] N. Escudeiro, P. Escudeiro, and A. Cruz. Semi-Automatic Grading of Students' Answers Written in Free Text. *Electronic Journal of e-Learning*, 9(1):15–22, 2011.
- [56] P. A. Estévez. Big Data Era Challenges and Opportunities in Astronomy-How SOM/LVQ and Related Learning Methods Can Contribute? In *WSOM*, page 267, 2016.
- [57] R. Farmanbar, S. Niknami, D. R. Lubans, and A. Hidarnia. Predicting exercise behaviour in Iranian college students: Utility of an integrated model of health behaviour based on the transtheoretical model and self-determination theory. *Health Education Journal*, 72(1):56–69, 2013.
- [58] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and Others. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *KDD*, volume 96, pages 82–88, 1996.
- [59] S. Fong and R. P. Biuk-Aghai. An automated university admission recommender system for secondary school students. In *The 6th International Conference on Information Technology and Applications*, page 42, 2009.
- [60] S. Francisco, M. A. Masethe, S. O. Ojo, and S. A. Odunaike. Taxonomy of Recommender Systems for Educational Data Mining (EDM) Techniques : A Systematic Review. *World Congress on Engineering and Computer Science*, 0958:7–12, 2019.
- [61] K. Ganeshan and X. Li. An Intelligent Student Advising System Using Collaborative Filtering. *IEEE Frontiers in Education Conference (FIE)*, pages 1–8, 2015.
- [62] E. García, C. Romero, S. Ventura, and C. De Castro. An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modeling and User-Adapted Interaction*, 19(1-2):99–132, 2009.
- [63] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 257–260, 2010.
- [64] E. F. Gehringer, X. Liu, A. Kariya, and G. Wang. Comparing and combining tests for plagiarism detection in online exams. In *International Conference on Educational Data Mining*, pages 2–6, 2020.
- [65] G. George and A. M. Lal. Review of ontology-based recommender systems in e-learning. *Computers & Education*, 142:103642, 2019.

- [66] M. Ghazanfar and A. Prugel-Bennett. Fulfilling the needs of gray-sheep users in recommender systems, a clustering solution. *International Conference on Information Systems and Computational Intelligence*, 2011.
- [67] C. A. Gomez-Urbe and N. Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.
- [68] S. Gottipati and V. Shankararaman. Competency analytics tool: Analyzing curriculum using course competencies. *Education and Information Technologies*, 23(1):41–60, 2018.
- [69] Z. Gulzar, A. A. Leema, and G. Deepak. Pcrs: Personalized course recommender system based on hybrid approach. *Procedia Computer Science*, 125:518–524, 2018.
- [70] N. Hagemann. Preliminary Studies for an Academic Module Recommender System : An Exploratory Data Analysis of UCD Student and Module Data, 2016.
- [71] N. Hagemann, M. P. O’Mahony, and B. Smyth. Module advisor: a hybrid recommender system for elective module exploration. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 498–499, 2018.
- [72] N. Hagemann, M. P. O’Mahony, and B. Smyth. Module advisor: Guiding students with recommendations. *Intelligent Tutoring Systems (ITS)*, 10858:319–325, 2018.
- [73] N. Hagemann, M. P. O’Mahony, and B. Smyth. Visualising module dependencies in academic recommendations. In *Proceedings of the 24th International Conference on Intelligent User Interfaces: Companion*, pages 77–78, 2019.
- [74] F. Haider, M. Koutsombogera, O. Conlan, C. Vogel, N. Campbell, and S. Luz. An active data representation of videos for automatic scoring of oral presentation delivery skills and feedback generation. *Frontiers in Computer Science*, 2:1, 2020.
- [75] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.
- [76] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, 1999.
- [77] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250, 2000.
- [78] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

- [79] I. Hilliger, C. Miranda, S. Celis, and M. Pérez-SanAgust\'. Evaluating Usage of an Analytics Tool to Support Continuous Curriculum Improvement. In *EC-TEL (Practitioner Proceedings)*, 2019.
- [80] N. Hurley and M. Zhang. Novelty and diversity in top-n recommendation-analysis and evaluation. *ACM Transactions on Internet Technology (TOIT)*, 10(4):1–30, 2011.
- [81] O. Iatrellis, A. Kameas, and P. Fitsilis. Academic advising systems: A systematic literature review of empirical evidence. *Education Sciences*, 7(4):90, 2017.
- [82] A. Jabbarova. Planning and Organization of the Educational Process in the Credit-Module System at Higher Education. *JSPI*, 2020.
- [83] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender systems: an introduction*. Cambridge University Press, 2010.
- [84] J. S. Justeson and S. M. Katz. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural language engineering*, 1(1):9–27, 1995.
- [85] M. Kaminskas, D. Bridge, and I. Centre. Diversity , Serendipity , Novelty , and Coverage : A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Transactions on Interactive Intelligent Systems*, 7(1):1–42, 2016.
- [86] G. Karypis. Evaluation of Item-Based Top-N Recommendation Algorithms. In *Proceedings of the Tenth International Conference on Information and Knowledge Management, CIKM '01*, pages 247–254, New York, NY, USA, 2001. Association for Computing Machinery.
- [87] K. Kawintiranon, P. Vateekul, A. Suchato, and P. Punyabukkana. Understanding knowledge areas in curriculum through text mining from course materials. In *2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pages 161–168, 2016.
- [88] A. Khalid, K. Lundqvist, and A. Yates. International Review of Research in Open and Distributed Learning. *Victoria*, 21(4), 2020.
- [89] E. S. Khorasani, Z. Zhenge, and J. Champaign. A markov chain collaborative filtering model for course enrollment recommendations. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 3484–3490. IEEE, 2016.
- [90] A. Klahold. *Empfehlungssysteme*. Springer, 2009.
- [91] A. Klačnja-Milićević, B. Vesin, and M. Ivanović. Social tagging strategy for enhancing e-learning experience. *Computers & Education*, 118:166–181, 2018.
- [92] Y. Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 81(2009):1–10, 2009.

- [93] S. Kristiansen, M. Sørensen, and T. R. Stidsen. Elective course planning. *European Journal of Operational Research*, 215(3):713–720, 2011.
- [94] S. R. Kumar and S. Hamid. Analysis of Learning Analytics in Higher Educational Institutions : A Review. *Advances in Visual Informatics*, 1:185–196, 2017.
- [95] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- [96] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.
- [97] D. Y. T. Liu, C. E. Taylor, A. J. Bridgeman, K. Bartimote-Aufflick, A. Pardo, and Others. Empowering Instructors Through Customizable Collection and Analyses of Actionable Information. In *PCLA@ LAK*, pages 3–9, 2016.
- [98] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou. Recommender systems. *Physics Reports*, 519(1):1–49, 2012.
- [99] I. Lykourantzou, I. Giannoukos, V. Nikolopoulos, G. Mpardis, and V. Loumos. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education*, 53(3):950–965, 2009.
- [100] N. D. Lynn and A. W. R. Emanuael. A review on Recommender Systems for course selection in higher education. *IOP Conference Series: Materials Science and Engineering*, 1098(Aasec 2020):1–7, 2021.
- [101] B. Ma, M. Lu, and Y. Taniguchi. Exploration and Explanation : An Interactive Course Recommendation System for University Environments. *ACM IUI 2021 Workshops*, 2021.
- [102] B. Ma, M. Lu, Y. Taniguchi, and S. Konomi. CourseQ: the impact of visual and interactive course recommendation in university environments. *Research and Practice in Technology Enhanced Learning*, 16(1):1–24, 2021.
- [103] L. P. Macfadyen and S. Dawson. Mining LMS data to develop an “early warning system” for educators: A proof of concept. *Computers & education*, 54(2):588–599, 2010.
- [104] S. Mackney and R. Shields. Learning analytics for student success at university: trends and dilemmas. In *The Educational Intelligent Economy: BIG DATA, Artificial Intelligence, Machine Learning and the Internet of Things in Education*. Emerald Publishing Limited, 2019.
- [105] Y. Madani, M. Erritali, J. Bengourram, and F. Sailhan. Social collaborative filtering approach for recommending courses in an E-learning platform. *Procedia Computer Science*, 151:1164–1169, 2019.
- [106] A. V. Manjarres, L. G. M. Sandoval, and M. S. Suárez. Data mining techniques applied in educational environments: Literature review. *Digital Education Review*, (33):235–266, 2018.

- [107] C. Márquez-Vera, A. Cano, C. Romero, A. Y. M. Noaman, H. Mousa Fardoun, and S. Ventura. Early dropout prediction using data mining: a case study with high school students. *Expert Systems*, 33(1):107–124, 2016.
- [108] F. Martin, T. Sun, and C. D. Westine. A systematic review of research on online teaching and learning from 2009 to 2018. *Computers and Education*, 159:1903–1929, 2020.
- [109] R. Mazza, M. Bettoni, M. Faré, and L. Mazzola. Moclog—monitoring online courses with log data. *Moodle ResearchConference*, 1, 2012.
- [110] R. Mazza and V. Dimitrova. CourseVis: A graphical student monitoring tool for supporting instructors in web-based distance courses. *International Journal of Human-Computer Studies*, 65(2):125–139, 2007.
- [111] J. McCuaig and J. Baldwin. Identifying Successful Learners from Interaction Behaviour. *International Educational Data Mining Society*, 2012.
- [112] C.-n. Z. S. M. Mcnee, J. A. Konstan, and G. Lausen. Improving Recommendation Lists Through Topic Diversification. *International conference on World Wide Web*, 14:22–32, 2005.
- [113] S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI’06 extended abstracts on Human factors in computing systems*, pages 1097–1101, 2006.
- [114] T. Meller, E. Wang, F. Lin, and C. Yang. New classification algorithms for developing online program recommendation systems. In *2009 International Conference on Mobile, Hybrid, and On-line Learning*, pages 67–72. IEEE, 2009.
- [115] P. Melville, R. J. Mooney, R. Nagarajan, and Others. Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai*, 23:187–192, 2002.
- [116] G. Méndez, V. Perimetral, and K. Chiluiza. Techniques for Data-Driven Curriculum Analysis. *LAK*, pages 148–157, 2014.
- [117] L. V. Morris, C. Finnegan, and S.-S. Wu. Tracking student behavior, persistence, and achievement in online courses. *The Internet and Higher Education*, 8(3):221–231, 2005.
- [118] R. Morsomme and S. V. Alferez. Content-Based Course Recommender System for Liberal Arts Education. *International Educational Data Mining Society*, 2019.
- [119] S. Morsy and G. Karypis. Learning Course Sequencing for Course Recommendation. 2018.
- [120] S. Morsy and G. Karypis. A study on curriculum planning and its relationship with graduation gpa and time to degree. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 26–35, 2019.

- [121] L. Mostafa, G. Oatley, N. Khalifa, and W. Rabie. A case based reasoning system for academic advising in egyptian educational institutions. In *2nd International Conference on Research in Science, Engineering and Technology (ICRSET'2014) March*, pages 21–22, 2014.
- [122] M. Mozgovoy, T. Kakkonen, and G. Cosma. Automatic student plagiarism detection: future perspectives. *Journal of Educational Computing Research*, 43(4):511–531, 2010.
- [123] T. Müller and K. Murray. Comprehensive approach to student sectioning. *Annals of Operations Research*, 181(1):249–269, 2010.
- [124] K. Murray, T. Müller, and H. Rudová. Modeling and solution of a complex university course timetabling problem. In *International Conference on the Practice and Theory of Automated Timetabling*, pages 189–209. Springer, 2006.
- [125] J. K. R. NAYEK and R. DAS. Evaluation of Famous Recommender Systems: A Comparative Analysis. *Evaluation*, 2021.
- [126] T. T. Nguyen, P.-M. Hui, F. M. Harper, L. Terveen, and J. A. Konstan. Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of the 23rd international conference on World wide web*, pages 677–686, 2014.
- [127] N. B. A. Normadhi, L. Shuib, H. N. M. Nasir, A. Bimba, N. Idris, and V. Balakrishnan. Identification of personal traits in adaptive learning environment: Systematic literature review. *Computers & Education*, 130:168–190, 2019.
- [128] C. Obeid, I. Lahoud, H. El Khoury, and P.-A. Champin. Ontology-based recommender system in higher education. In *Companion Proceedings of the The Web Conference 2018*, pages 1031–1034, 2018.
- [129] OECD. *Education at a Glance 2017*. 2021.
- [130] M. P. O'Mahony and B. Smyth. A recommender system for on-line course enrolment: an initial study. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 133–136, 2007.
- [131] J. W. Orr and N. Russell. Automatic Assessment of the Design Quality of Python Programs with Personalized Feedback. *arXiv preprint arXiv:2106.01399*, 2021.
- [132] P. Ott. *Incremental matrix factorization for collaborative filtering*. Citeseer, 2008.
- [133] P. Paatero. Least squares formulation of robust non-negative factor analysis. *Chemometrics and intelligent laboratory systems*, 37(1):23–35, 1997.
- [134] A. Parameswaran, P. Venetis, and H. Garcia-Molina. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Transactions on Information Systems (TOIS)*, 29(4):1–33, 2011.

- [135] A. G. Parameswaran, G. Koutrika, B. Bercovitz, and H. Garcia-Molina. Rec-splorer: recommendation algorithms based on precedence mining. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 87–98, 2010.
- [136] D. Paraschakis. Recommender systems from an industrial and ethical perspective. In *Proceedings of the 10th ACM conference on recommender systems*, pages 463–466, 2016.
- [137] Z. A. Pardos and W. Jiang. Designing for serendipity in a university course recommendation system. In *Proceedings of the tenth international conference on learning analytics & knowledge*, pages 350–359, 2020.
- [138] D. H. Park, H. K. Kim, I. Y. Choi, and J. K. Kim. A literature review and classification of recommender systems research. *Expert systems with applications*, 39(11):10059–10072, 2012.
- [139] Y. Park. A Recommender System for Personalized Exploration of Majors, Minors, and Concentrations. In *RecSys Posters*, 2017.
- [140] M. J. Pazzani, J. Muramatsu, D. S. Billsus, and Webert. Identifying interesting web sites. *Aaai*, pages 54–59, 1996.
- [141] M. Pechenizkiy, N. Trcka, P. De Bra, and P. Toledo. CurriM: curriculum mining. In *Educational Data Mining 2012*, 2012.
- [142] G. Piao. Recommending Knowledge Concepts on MOOC Platforms with Metapath-based Representation Learning. *Educational Data Mining*, 2021.
- [143] V. Pigott and D. Frawley. *An Analysis of Completion in Irish Higher Education 2007/2008 Entrants: A Report by the Higher Education Authority*. Higher Education Authority, 2019.
- [144] N. Pillay. A survey of school timetabling research. *Annals of Operations Research*, 218(1):261–293, 2014.
- [145] M. D. Pistilli and G. L. Heileman. Guiding early and often: Using curricular and learning analytics to shape teaching, learning, and student success in gateway courses. *New Directions for Higher Education*, 2017(180):21–30, 2017.
- [146] M. D. Pistilli, J. E. Willis, and J. P. Campbell. Analytics through an institutional lens: Definition, theory, design, and impact. In *Learning analytics*, pages 79–102. Springer, 2014.
- [147] M. F. Porter. An algorithm for suffix stripping. *Program*, 1980.
- [148] A. Pradeep. Students Dropout Factor Prediction Using EDM Techniques. *International Conference on Soft-Computing and Networks Security*, pages 1–7, 2015.
- [149] P. Pu and L. Chen. Trust-inspiring explanation interfaces for recommender systems. *Knowledge-Based Systems*, 20(6):542–556, 2007.

- [150] A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on empirical methods in natural language processing*, 1996.
- [151] R. K. Ratner, B. E. Kahn, D. Kahneman, B. E. Kahn, and D. Kahneman. Choosing Less-Preferred Experiences for the Sake of Variety. *Journal of Consumer Research*, 26(1):1–15, 1999.
- [152] S. Ray and A. Sharma. A collaborative filtering based approach for recommending elective courses. In *International Conference on Information Intelligence, Systems, Technology and Management*, pages 330–339. Springer, 2011.
- [153] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.
- [154] K. Robinson, D. Brown, and M. Schedl. User insights on diversity in music recommendation lists. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR 2020)*, 2020.
- [155] C. Romero and S. Ventura. *Data mining in e-learning*, volume 4. WIT press, 2006.
- [156] C. Romero and S. Ventura. Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6):601–618, 2010.
- [157] C. Romero and S. Ventura. Educational data mining and learning analytics : An updated survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(December 2019):1–21, 2020.
- [158] K. A. Ross, C. S. Jensen, R. Snodgrass, C. E. Dyreson, C. S. Jensen, R. Snodgrass, and L. Chen. Cross-validation. *encyclopedia of database systems*, 2009.
- [159] S. Rüdian and N. Pinkwart. Finding the optimal topic sequence for online courses using SERPs as a Proxy. *Educational Data Mining*, 2021.
- [160] R. Saga, Y. Hayashi, and H. Tsuji. Hotel recommender system based on user’s preference transition. In *2008 IEEE International Conference on Systems, Man and Cybernetics*, pages 2437–2442. IEEE, 2008.
- [161] G. Salton and M. J. McGill. *Introduction to modern information retrieval*. mcgraw-hill, 1983.
- [162] G. Salton, A. Wong, and C.-S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
- [163] C. Sammut and G. I. Webb, editors. *Leave-One-Out Cross-Validation*, pages 600–601. Springer US, Boston, MA, 2010.
- [164] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

- [165] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260, 2002.
- [166] T. Sekiya. Mapping Analysis of CS2013 by Supervised LDA and Isomap. *IEEE International Conference on Teaching, Assessment and Learning for Engineering*, (December):33–40, 2014.
- [167] M. Shakhshi-Niaei and H. Abuei-Mehrzi. An optimization-based decision support system for students’ personalized long-term course planning. *Computer Applications in Engineering Education*, 28(5):1247–1264, 2020.
- [168] G. Shani and A. Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer, 2011.
- [169] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating “word of mouth”. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, 1995.
- [170] R. Shatnawi, Q. Althebyan, B. Ghaleb, and M. Al-Maolegi. A Student Advising System Using Association Rule Mining. *International Journal of Web-Based Learning and Teaching Technologies (IJWLTT)*, 16(3):65–78, 2021.
- [171] R. Shatnawi, Q. Althebyan, B. Ghalib, and M. Al-Maolegi. Building a smart academic advising system using association rule mining. *arXiv preprint arXiv:1407.1807*, 2014.
- [172] G. Siemens. Learning Analytics : The Emergence of a Discipline. *American Behavioral Scientist*, 57:1380–1400, 2013.
- [173] M. Slaney and W. White. Measuring playlist diversity for recommendation systems. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, pages 77–82, 2006.
- [174] B. Smith and G. Linden. Two Decades of Recommender Systems at Amazon . com. *IEEE Internet Computing*, 21(3):12–18, 2017.
- [175] B. Smyth and P. McClave. Similarity vs. diversity. In *International conference on case-based reasoning*, pages 347–361. Springer, 2001.
- [176] J. Sobecki and J. M. Tomczak. Student courses recommendation using ant colony optimization. In *Asian Conference on Intelligent Information and Database Systems*, pages 124–133. Springer, 2010.
- [177] M. R. Sundari, G. Shreya, and T. Jawahar. Course Recommendation System. *International Journal of Computer Applications*, 175(29):13–16, 2020.
- [178] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Providing justifications in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 38(6):1262–1272, 2008.

- [179] K. Taha. Automatic academic advisor. In *8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pages 262–268. IEEE, 2012.
- [180] R. Talib, M. K. Hanif, S. Ayesha, and F. Fatima. Text mining: techniques, applications and issues. *International Journal of Advanced Computer Science and Applications*, 7(11):414–418, 2016.
- [181] A.-H. Tan et al. Text mining: The state of the art and the challenges. In *Proceedings of the pakdd 1999 workshop on knowledge discovery from advanced databases*, volume 8, pages 65–70. Citeseer, 1999.
- [182] T. Y. Tang and P. Winoto. I should not recommend it to you even if you will like it: the ethics of recommender systems. *New Review of Hypermedia and Multimedia*, 22(1-2):111–138, 2016.
- [183] J. K. Tarus, Z. Niu, and B. Khadidja. E-learning recommender system based on collaborative filtering and ontology. *International Journal of Computer and Information Engineering*, 11(2):256–261, 2017.
- [184] K. Thaker, L. Zhang, D. He, and P. Brusilovsky. Recommending Remedial Readings Using Student Knowledge State. *International Educational Data Mining Society*, 2020.
- [185] N. Tintarev and J. Masthoff. Evaluating the effectiveness of explanations for recommender systems. *User Modeling and User-Adapted Interaction*, 22(4-5):399–439, 2012.
- [186] N. Trcka and M. Pechenizkiy. From local patterns to global models: Towards domain driven educational process mining. In *2009 Ninth international conference on intelligent systems design and applications*, pages 1114–1119. IEEE, 2009.
- [187] C.-H. Tsai and P. Brusilovsky. Evaluating visual explanations for similarity-based recommendations: User perception and performance. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization*, pages 22–30, 2019.
- [188] C. Van Rijsbergen. Information retrieval: theory and practice. In *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar on Data Base Systems*, pages 1–14, 1979.
- [189] M. Vandewaetere, P. Desmet, and G. Clarebout. The contribution of learner characteristics in the development of computer-based adaptive learning environments. *Computers in Human Behavior*, 27(1):118–130, 2011.
- [190] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 109–116, 2011.
- [191] S. Vargas, P. Castells, and J. Wang. Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. *International Workshop on Diversity in Document Retrieval*, pages 109–116, 2011.

- [192] M. Venkatasubramanian, K. Chetal, D. J. Schnell, G. Atluri, and N. Salomonis. Resolving single-cell heterogeneity from hundreds of thousands of cells through sequential hybrid clustering and NMF. *Bioinformatics*, 36(12):3773–3780, 2020.
- [193] M. K. Vijaymeena and K. Kavitha. A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal*, 3(2):19–28, 2016.
- [194] H. Vossensteyn, A. Kottmann, B. Jongbloed, F. Kaiser, L. Cremonini, B. Stensaker, E. Hovdhaugen, and S. Wollscheid. *Drop-Out and Completion in Higher Education in Europe - Literature Review*. 2015.
- [195] A. Y. Wang, M. H. Newlin, and T. L. Tucker. A discourse analysis of online classroom chats: Predictors of cyber-student performance. *Teaching of Psychology*, 28(3):222–226, 2001.
- [196] N. Werghi and F. K. Kamoun. A decision-tree-based system for student academic advising and planning in information systems programmes. *International Journal of Business Information Systems*, 5(1):1–18, 2010.
- [197] B. Williamson. *Big data in education: The digital future of learning, policy and practice*. Sage, 2017.
- [198] C. Wissler. The Spearman correlation formula. *Science*, 22(558):309–311, 1905.
- [199] A. Wren. Scheduling, timetabling and rostering—a special relationship? In *International conference on the practice and theory of automated timetabling*, pages 46–75. Springer, 1995.
- [200] Z. Wu, T. He, C. Mao, and C. Huang. Exam paper generation based on performance prediction of student group. *Information Sciences*, 532:72–90, 2020.
- [201] W. F. W. Yaacob, N. M. Sobri, S. A. M. Nasir, N. D. Norshahidi, and W. Z. W. Husin. Predicting student drop-out in higher institution using data mining techniques. In *Journal of Physics: Conference Series*, volume 1496. IOP Publishing, 2020.
- [202] S. J. H. Yang. Guest Editorial: Precision Education-A New Challenge for AI in Education. *Journal of Educational Technology & Society*, 24(1), 2021.
- [203] T. Yu, J. Guo, W. Li, H. J. Wang, and L. Fan. Recommendation with diversity: An adaptive trust-aware model. *Decision Support Systems*, 123:113073, 2019.
- [204] L. Yujian and L. Bo. A normalized Levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.
- [205] L. Zhang. The Definition of Novelty in Recommendation System. *Journal of Engineering Science & Technology Review*, 6(3), 2013.
- [206] Y. Zhang, S. Oussena, T. Clark, and H. Kim. Use Data Mining to Improve Student Retention in Higher Education-A Case Study. In *ICEIS (1)*, pages 190–197, 2010.
- [207] Y. C. Zhang, D. Ó. Séaghdha, D. Quercia, and T. Jambor. Auralist: introducing serendipity into music recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 13–22, 2012.

