Improving Students’ Academic Performance with AI and Semantic Technologies

Yixin Cheng

COMP 8755 Individual Computing Project
Supervised by: Dr. Bernardo Pereira Nunes
The Australian National University

November 2021
© Yixin Cheng 2021
Except where otherwise indicated, this thesis is my own original work.

Yixin Cheng
19 November 2021
Acknowledgments

Firstly, I must express my sincere appreciation to my supervisor, Dr. Bernardo Pereira Nunes, for his heuristic guidance, perceptive comments, and personal support. It is an honor as well as a luck to be his student. His enthusiasm and meticulousness towards academics motivate me along the way. He has monitored my progress and offered advice and encouragement throughout. Without his constant help, I would not have completed this project on schedule.

Thanks also to Felipe Cueto-Ramirez, for providing support and advice for my project.

Special thanks to Jinxiao Xie, who has always encouraged me to challenge myself and made time to help and support me.

Last but not least, I owe a debt of gratitude to my parents and sisters, who provided me with the opportunity to study abroad, and endless love.
Abstract

Artificial intelligence and semantic technologies are evolving and have been applied in various research areas, including the education domain. Higher Education institutions (HEIs) strive to improve students’ academic performance. Early intervention to at-risk students and a reasonable curriculum is vital for students’ success. Prior research opted for deploying traditional machine learning models to predict students’ performance. However, the existing research on applying deep learning models on prediction is very limited. In terms of curriculum semantic analysis, after conducting a comprehensive systematic review regarding the use of semantic technologies in the context of Computer Science curriculum, a major finding of the study is that technologies used to measure similarity have limitations in terms of accuracy and ambiguity in the representation of concepts, courses, etc. To fill these gaps, in this study, three implementations were developed, that is, to predict students’ performance using marks from the previous semester, to model a course representation in a semantic way and compute the similarity, and to identify the prerequisite between two similar courses. Regarding performance prediction, we used the combination of Genetic Algorithm and Long-Short Term Memory (LSTM) on a dataset from a Brazilian university containing 248730 records. As for similarity measurement between courses, we deployed Bidirectional Encoder Representation with Transformers (BERT) to encode the sentence in the course description from the Australian National University (ANU). We then used cosine similarity to obtain the distance between courses. With respect to prerequisite identification, TextRazor was applied to extract concepts from course descriptions, followed by employing SemRefD to measure the degree of prerequisite between two concepts. The outcomes of this study can be summarized as: (i) a breakthrough result improves Manrique’s work by 2.5% in terms of accuracy in dropout prediction; (ii) uncover the similarity between courses based on course description; (iii) identify the prerequisite over three compulsory courses of School of Computing at ANU. In the future, these technologies could potentially be used to identify at-risk students, analyze the curriculum of university programs, aid student advisors, and create recommendation systems.

Keywords: Dropout Prediction, Curriculum Semantic Analysis, Similarity Measurement, Prerequisite Identification, Genetic Algorithm, Long-Short Term Memory, Bidirectional Encoder Representation with Transformers, SemRefD
Contents

Acknowledgments iii

Abstract v

1 Introduction 1
  1.1 Problem Statement and Motivations 11
  1.2 Objectives 2
  1.3 Contributions 2
  1.4 Thesis Outline 2

2 Background and Related Work 3
  2.1 Background 3
  2.2 Related Work 4
    2.2.1 Methodology 4
    2.2.2 Tools/Frameworks 6
    2.2.3 Datasets 6
    2.2.4 Languages, Classes and Vocabulary 7
      2.2.4.1 Curriculum Design Analysis and Approaches 8

3 Approaches 11
  3.1 Dropout Prediction 11
    3.1.1 Procedure Overview 11
    3.1.2 Data Pre-processing 11
    3.1.3 Feature Selection 15
    3.1.4 Training and Test 17
  3.2 Curriculum Semantic Analysis 22
    3.2.1 Procedure Overview 22
    3.2.2 Similarity Measurement 23
    3.2.3 Prerequisite Identification 24

4 Results and Discussion 27
  4.1 Experimental Environment 27
  4.2 Dropout Prediction 27
    4.2.1 Evaluation Metrics 27
    4.2.2 Experimental Results an Discussions 28
      4.2.2.1 Feature Selection 28
      4.2.2.2 Dropout Prediction 29
List of Figures

2.1 Paper Selection Process ........................................... 5
3.1 The workflow of dropout prediction ................................. 12
3.2 Dataset snippet ....................................................... 12
3.3 Distribution by year .................................................. 13
3.4 Student distribution by degree ....................................... 13
3.5 Dropout distribution by year ......................................... 13
3.6 Procedure of GA ....................................................... 18
3.7 Uniform Crossover ..................................................... 18
3.8 The workflow of GA+SVM feature selection ......................... 19
3.9 The internal architecture of LSTM cell [Sanjeevi 2018] .......... 20
3.10 The workflow of LSTM + FCs ........................................ 22
3.11 The brief workflow of the Similarity Measurement ................. 23
3.12 Prerequisite Identification workflow ............................... 23
3.13 BERT Base .............................................................. 24
3.14 The prerequisite of concept A ($c_A$) and Concept B ($c_B$) .... 26
4.1 Confusion Matrix ......................................................... 28
4.2 Accuracy of LSTM+FC on ADM ...................................... 30
4.3 Loss of LSTM+FC on ADM ........................................... 30
4.4 Accuracy of LSTM+FC on ARQ ....................................... 30
4.5 Loss of LSTM+FC on ARQ ........................................... 30
4.6 Accuracy of LSTM+FC on CSI ....................................... 30
4.7 Loss of LSTM+FC on CSI ............................................. 30
4.8 All level courses similarity ............................................ 32
4.9 1000-level courses similarity ........................................ 33
4.10 2000-level courses similarity ....................................... 33
4.11 3000-level courses similarity ....................................... 33
4.12 4000-level courses similarity ....................................... 33
4.13 All level courses similarity .......................................... 34
5.1 Contract page 1 ......................................................... 46
5.2 Contract page 2 ........................................................ 47
5.3 README page 1 ......................................................... 52
5.4 README page 2 ......................................................... 53
5.5 README page 3 ......................................................... 54
5.6 README page 4 ........................................................................... 55
# List of Tables

| Table | Description                                                                 | Page |
|------|-----------------------------------------------------------------------------|------|
| 3.1  | Data wrangling techniques used in pre-processing                           | 14   |
| 3.2  | Genetic Algorithm techniques, descriptions, goals, and specific method used in this study | 17   |
| 3.3  | The components of a LSTM cell                                              | 20   |
| 4.1  | Experimental Environment                                                   | 27   |
| 4.2  | Results of ADM, ARQ, and CSI by SVM                                        | 29   |
| 4.3  | Results of ADM, ARQ, and CSI                                               | 31   |
Introduction

1.1 Problem Statement and Motivations

Students’ academic performance is not only important for themselves, but also important for higher education institutions (HEIs), who use it as a basis for measuring the success of their educational programs. A variety of metrics can be used to measure student performance. Dropout rates are notable among these metrics. In other words, the reduction of dropout rates could indicate the improvement of student’s academic performance. However, according to the published figures, each year, one-fifth of first-year undergraduates drop out of their degrees across Australia [Shipley and Ian, 2019]. In Brazil, it is estimated that only 62.4% of university enrolments succeed in obtaining an undergraduate degree [Sales et al., 2016]. These are concerning statistics for a country’s development and can affect students’ lives financially, mentally, and professionally. As a consequence, HEIs and researchers have shown an increasing interest in predicting systems to identify students at risk of dropping out [Manrique et al., 2019a]. For example, Ruben et al. created multiple feature sets from student data to predict whether a student would drop out by using machine learning algorithms [Manrique et al., 2019a]. Also, Mujica et al. pointed out that path analysis was useful for predicting student dropouts based on affective and cognitive variables [Diaz-Mujica et al., 2019].

According to Vergel et al., the curriculum design may affect the students’ performance and retention. A careful design is required to "understand the role of curriculum design in dropout rates and unmask how the political dynamics embedded in the curriculum influence students’ retention" [Vergel et al., 2018]. Moreover, Drennan, Rohde, Johnson, and Kuennen claimed that students’ academic performance in a course is related to their performance in their prerequisite courses [Drennan and Rohde, 2002], [Johnson and Kuennen, 2006]. It can be seen that curriculum plays a crucial role in student performance as well as decreasing dropout rates.

The main motivation for this research is to improve students’ academic performance based on the analysis of curriculum and student academic performance in different courses. Identifying prerequisites for a course is key to enhancing the student experience, improving curriculum design, and increasing graduation rates.
1.2 Objectives

The main objective of this study is to apply Artificial Intelligence and Semantic technologies to analyse and allow changes that may improve student academic performance.

The goal of this research project is threefold:

- Predict students’ performance in university courses using grades from previous semesters;
- Model a course semantic representation and calculate the similarity among courses; and, finally,
- Identify the sequence between two similar courses.

To accomplish these goals, a systematic review was carried out to identify what technologies are being used in CS curriculum.

1.3 Contributions

The main contributions of this thesis can be divided into:

- An approach for student dropout prediction using Genetic Algorithm (GA) and Long Short-Term Memory (LSTM);
- A comprehensive systematic review to understand how Semantic Web and Natural Language Processing technologies have been applied to curriculum analysis and design in Computer Science; and, finally,
- An analysis of a Computer Science curriculum using the SemRefD distance proposed by Manrique et al. [Manrique et al. 2019a] and BERT proposed by Devlin et al. [Devlin et al. 2018].

1.4 Thesis Outline

Section 2 presents the background information on dropout prediction and text analysis. Section 3 covers the methodology used in this work including a systematic review and general techniques. Section 4 presents the experiments, results and discussion and, finally, section 5 concludes this thesis with future work directions.
Chapter 2

Background and Related Work

The chapter is divided into two sections: background and related work. The following section briefly introduces the basic concepts unitized in this research. In related work, we conduct a systematic review regarding curriculum analysis by using Artificial Intelligence and Semantic Web technologies.

2.1 Background

Artificial Intelligence refers to the development of machines or computers that simulate or emulate the functions of the human brain. The functions of a computer differ according to the area of study. Prolog, for example, is a programming language that aims to understand human logic. It also applies mathematics in order to create systems that can discern relevant conclusions from a set of statements. Intelligent Agents is another example. Unlike traditional agents, Intelligent Agents are designed to take actions that are optimized to achieve a specific goal. Based on their perception of the environment and internal rules, intelligent agents make decisions. The study explores four growing fields: Machine Learning, Deep Learning, Natural Language Processing, and Semantic Web.

Machine Learning is a field of study that uses algorithms to detect patterns in data and to predict or make useful decisions. Different machine learning algorithms can be implemented in an innumerable number of scenarios. Machine learning algorithms are optimized for specific data sets; There are a variety of algorithms used to build models to suit different use cases. Using machine learning algorithms, we predict students’ performance in university courses based on their performance in previous courses.

Deep Learning is a neural network with multiple layers of perceptrons. The neural networks try to imitate human brain activities, albeit far from matching its capabilities, which enables it to study from high volume of data. Additional hidden layers can help to refine and optimize a neural network for accuracy, even when it has a single layer. Recurrent Neural Networks (RNNs) are one special type of neural networks that combines the information from previous time steps to generate updated outputs.

The Natural Language Processing (NLP) process is a psychological procedure that consists of analyzing successful people’s strategies and applying them to reaching a
Background and Related Work

personal goal. The process links cognitions, language, and patterns of action that are learned through experience to particular outcomes.

W3C (World Wide Web Consortium) standards enable the Semantic Web to be extended from the World Wide Web. W3C standardizes the process of making information on the World Wide Web machine-readable. In order to accomplish this goal, a series of communication standards were created that enable developers to describe concepts and entities, as well as their classification and relationships. The Resource Description Framework (RDF) and Web Ontology Language (OWL) have enabled developers to create systems that store and use complex knowledge data bases known as knowledge graphs.

2.2 Related Work

This section discusses tools, frameworks, datasets, semantic technologies, and approaches used for curriculum design in Computer Science. Note that we present the methodology used to carry out a systematic review, but for brevity and adequacy we only present the most relevant works and sections. The complete systematic review will be submitted to a conference in Computer Science in Education as an outcome of this thesis.

2.2.1 Methodology

This systematic review was conducted following the methodology defined by Kitchenham and Charters ([BA and Charters, 2007]. The method is composed of three main steps: planning, conducting and reporting.

The planning stage helps to identify existing research in the area of interest as well as build the research question. Our research question was created based on the PICOC method ([BA and Charters, 2007] and used to identify keywords and corresponding synonyms to build the search string to find relevant related works. The resulting search string is given below.

("computer science" OR "computer engineering" OR "informatics") AND ("curriculum" OR "course description" OR "learning outcomes" OR "curricula" OR "learning objects") AND ("semantic" OR "ontology" OR "linked data" OR "linked open data")

To include a paper in this systematic review, we defined the following inclusion criteria: (1) papers must have been written in English; (2) papers must be exclusively related to semantic technologies, computer science and curriculum; (3) papers must have 4 or more pages (i.e., full research papers); (4) papers must be accessible online; and, finally, (5) papers must be scientifically sound, present a clear methodology and conduct a proper evaluation for the proposed method, tool or model.

Figure [2.1] shows the paper selection process in detail. Initially, 4,510 papers were retrieved in total. We used the ACM Digital Library\(^1\), IEEE Xplore Digital Library\(^2\),

---

1. https://dl.acm.org/
2. https://ieeexplore.ieee.org/
§2.2 Related Work

Springer\(^3\), Scopus\(^4\), ScienceDirect\(^5\) and Web of Science\(^6\) as digital libraries. The Springer digital library returned a total of 3,549 studies. The large number of papers returned by the Springer search mechanism led us to develop a simple crawling tool to help us in the process of including/rejecting papers in our systematic review. All the information returned by Springer was collected and stored in a relational database. After that, we were able to correctly query the Springer database and select the relevant papers for this systematic review.

We also applied the forward and backward snowballing method to this systematic review to identify relevant papers that were not retrieved by our query string. The backward snowballing method was used to collect relevant papers from the references of the final list of papers in Phase 1, whereas the forward snowballing method was used to collect the papers that cited these papers [Wohlin 2014]. Google Scholar\(^7\) was used in the forward snowballing method. In total, 37 studies were identified as relevant; most of the studies were published in the last few years, which shows the increasing relevancy of this topic.

In the following sections we present the most relevant works that inspired the proposed approaches reported in this thesis.

---

\(^3\)https://www.springer.com/
\(^4\)https://www.scopus.com/
\(^5\)https://www.sciencedirect.com/
\(^6\)https://www.webofscience.com/
\(^7\)https://scholar.google.com/
2.2.2 Tools/Frameworks

Protégé\cite{Imran2016} is a popular open-source editor and framework for ontology construction, which several researchers have adopted to design curricula in Computer Science \cite{Tang2013, Wang2019, Nuntawong2017, Karunananda2012, Hedayati2016, Saquicela2018, Maffei2016, Vaquero2009}. Despite its wide adoption, Protégé still presents limitations in terms of the manipulation of ontological knowledge \cite{Tang2013}. As an attempt to overcome this shortcoming, Asoke et al. \cite{Karunananda2012} developed a curriculum design plug-in called OntoCD. OntoCD allows curriculum designers to customise curricula by loading a skeleton curriculum and a benchmark domain ontology. The evaluation of OntoCD is performed by developing a Computer Science degree program using benchmark domain ontologies developed in accordance with the guidelines provided by IEEE and ACM. Moreover, Adelina and Jason \cite{Tang2013} used Protégé to develop SUCO (Sunway University Computing Ontology), an ontology-specific Application Programming Interface (API) for curricula management system. They claim that, in response to the shortcoming with using the Protégé platform, SUCO shows a higher level of ability to manipulate and extract knowledge and will function effectively if the ontology is processed as an eXtensible Markup Language (XML) format document.

Other specific ontology-based tools in curriculum management have also been developed. CDIO \cite{Liang2012} is an example of such tool. CDIO was created to automatically adapt a given curriculum according to teaching objectives and teaching content based on the designed constructed ontology. A similar approach was used by Maffei et al. \cite{Maffei2016} to model the semantics behind functional alignments in order to design, synthesize, and evaluate functional alignment activities. In Mandić’s study, the author presented a software platform\cite{Mandic2018} for comparing chosen Curriculum for information technology teachers \cite{Mandic2018}. In Hedayati’s work, the authors used the curriculum Capability Maturity Model (CMM), which is a taxonomical model used for describing the organization’s level of capability in the domain of software engineering \cite{Paulk1993}. An ontology-driven model for analyzing the development process of the vocational ICT curriculum in the context of the culturally sensitive curriculum in Afghanistan is used as a reference model \cite{Hedayati2016}.

2.2.3 Datasets

There are several datasets used in curriculum design studies in Computer Science. The open-access CS2013\cite{CS2013} dataset is the result of the joint development of a computing curriculum sponsored by the ACM and IEEE Computer Society \cite{Piedra2018}. The CS2013 dataset has been used in several studies \cite{Piedra2018, Aeiad2016, Nuntawong2016, Nuntawong2017}.
2.2.2 Related Work

[Karunananda et al., 2012], [Hedayati and Mart, 2016], and [Fiallos, 2018] to develop ontologies or as a benchmark curriculum in similarity comparison between computer science curricula.

Similar to CS2013, The Thailand Qualification Framework for Higher Education (TQF: HEd) was developed by the Office of the Thailand Higher Education Commission to be used by all higher education institutions (HEIs) in Thailand as a framework to enhance the quality of course curricula, including the Computer Science curriculum. TQF: HEd was used for the guidelines in terms of ontology development in the following studies [Nuntawong et al., 2017], [Nuntawong et al., 2016], [Hao et al., 2008], and [Nuntawong et al., 2015].

Other studies use self-created datasets (e.g., [Wang et al., 2019], [Maffei et al., 2016], [Hedayati and Mart, 2016], and [Fiallos, 2018]). Specifically, in Wang’s work, the Ontology System for the Computer Course Architecture (OSCCA) was proposed based on a dataset created using course catalogs from top universities in China as well as network education websites [Wang et al., 2019]. In Maffei’s study, the authors experimented and evaluate the proposal based on the Engineering program at KTH Royal Institute of Technology in Stockholm, Sweden [Maffei et al., 2016]. In Gubervic et al.’s work, the dataset used in comparing courses comes from Faculty of Electrical Engineering and Computing compared to all universities from United States of America [Guberovic et al., 2018]. In Fiallos’s study, not only did the author adopt CS2013 for domain ontologies modeling, but also the core courses from Escuela Superior Politécnica del Litoral (ESPOL 11) Computational Sciences were collected for semantic similarity comparison [Fiallos, 2018].

2.2.4 Languages, Classes and Vocabulary

RDF is used as the design standard for data interchange in the following studies [Piedra and Caro, 2018], [Nuntawong et al., 2017], and [Saquicela et al., 2018]. In particular, Saquicela et al. [Saquicela et al., 2018] generated curriculum data in the RDF format, creating and storing data in a repository when the ontological model has been defined and created.

OWL is an extension of RDF that adds additional vocabulary and semantics to the classic framework [McGuinness and van Harmelen, 2004]. OWL is used in many studies [Piedra and Caro, 2018], [Barb and Kilicay-Ergin, 2020], [Mandic, 2018], [Wang et al., 2019], [Maffei et al., 2016], and [Vaquero et al., 2009] for representing and sharing knowledge on the Web 12. Apart from OWL, only two studies used XML 13 due to implementation requirements of research ([Tang and Hoh, 2013] and [Hao et al., 2008]).

Body of Knowledge (BoK), a subclass of OWL, is a complete set of concepts, terms and activities, which can represent the accepted ontology for a professional domain [Piedra and Caro, 2018]. BoK has become a common development method in many

---

11 https://www.espol.edu.ec/
12 https://www.w3.org/OWL/
13 https://www.w3.org/standards/xml/
Background and Related Work

studies [Piedra and Caro, 2018], [Nuntawong et al., 2017], [Nuntawong et al., 2016], [Hao et al., 2008], [Karunananda et al., 2012], [Tapia-Leon et al., 2018], [Chung and Kim, 2014], and [Nuntawong et al., 2015]. In Piedra’s study [Piedra and Caro, 2018], the BoK defined in CS2013 ontology was viewed as a to-be-covered description of the content and a curriculum to implement this information. Similarly, Nuntawong et al. [Nuntawong et al., 2017] applied the BoK, which is based on the ontology of TQF: HEd, to conduct the ontology mapping.

The Library of Congress Subject Headings (LCSH) is a managed vocabulary upheld by the Library of Congress. LCSH terminology is a BoK, which contains more than 240,000 topical subject headings. The equivalence, hierarchical, and associative types of relationships between headings can be offered. In Adrian’s study, the authors create an ontology based on LCSH and the Faceted Application of Subject Terminology (a completely enumerative faceted subject terminology schema originated from LCSH), to assess the consistency of an academic curriculum and apply it to an Information Science curriculum [Barb and Kilicay-Ergin, 2020].

Knowledge Area (KA), also a subclass of OWL, is an area of specialization such as Operating Systems and Algorithm. The relationship between BoK and KA is built in various ways in the studies. For example, in Piedra’s study [Piedra and Caro, 2018], each BoK class contains a set of KAs. In contrast, Nuntawong et al. [Nuntawong et al., 2017] considered KA as the superclass of BoK. KA classification was proposed in Orellana et al.’s study [Orellana et al., 2018]. In that paper, the curricula are classified in the KA defined by UNESCO [https://whc.unesco.org/en/] which defines 9 main areas and 24 subareas of knowledge. To do this, they convert the curricula to the vector space and then process with using traditional supervised approaches such as the support vector machines and k-nearest neighbors [Orellana et al., 2018]. By classifying the KA, the similarity measurement can be applied more easily.

2.2.4.1 Curriculum Design Analysis and Approaches

One development approach found is the extraction and interrelationship analysis. NLTK [https://www.nltk.org/] is deployed to segment raw data into terms, using various algorithms to extract and analyse the interrelationship of items, then construct an ontology by using a certain framework such as Protégé as in [Piedra and Caro, 2018], [Wang et al., 2019], and [Tapia-Leon et al., 2018].

Text Mining, also known as text analysis methods, have been used to find the interesting patterns in HEIs’ curriculum [Orellana et al., 2018]. With using Text Mining approaches, keywords can be extracted from both found documents and course materials for the further comparison and analysis [Kawintiranon et al., 2016]. In the next section, the application of Text Mining approaches in Computer Science curriculum will be elaborated.
In the context of curriculum similarity measurement, Gomaa and Fahmy [H. Gomaa and A. Fahmy, 2013] define string-based, corpus-based, knowledge-based and hybrid similarity measures. String-based measures the distance between string (words) that can be compared by characters or terms. Corpus-based approach measures the semantic meaning of terms and phrases which are provided in the corpus. Knowledge-based uses synsets-formed word networks such as WordNet to compare cognitive meaning between each other. String-based similarity was conducted by Corpus-based was done in [Orellana et al., 2018], [Pawar and Mago, 2018], [Aeiad and Meziane, 2016] and [Fiallos, 2018]. Knowledge-based similarity approach is proposed in [Nuntawong et al., 2015].

String-based similarity between terms was measured in many studies [Orellana et al., 2018], [Pawar and Mago, 2018], [Barb and Kilicay-Ergin, 2020], [Seidel et al., 2020], [Wang et al., 2019], and [Saquicela et al., 2018]. Orellana et al. [Orellana et al., 2018] used cosine similarity between terms to acquire the level of similarity between two course descriptions. Adrian and Wang used the same approach to measure the similarity [Barb and Kilicay-Ergin, 2020], [Wang et al., 2019].

Pawar and Mago utilized the Bloom’s taxonomy to measure the similarity of sentence pairs in the Learning outcomes [Pawar and Mago, 2018]. In another paper, Saquicela et al. used the K-means, an unsupervised clustering algorithm to calculate the similarity among courses content [Saquicela et al., 2018].

Corpus-Based similarity measurement was proposed in these studies [Orellana et al., 2018], [Pawar and Mago, 2018], [Barb and Kilicay-Ergin, 2020], and [Fiallos, 2018]. In Orellana et al.’s study, the topics are extracted and processed by LSA. Through the process, the terms and documents are located within their context to get the most relevant documents (Wikipedia articles and curricula) by adding the similarity threshold.

Knowledge-based (Semantic) similarity measurement was proposed in [Nuntawong et al., 2016] and [Pawar and Mago, 2018]. In Nuntawong’s paper [Nuntawong et al., 2017], the authors designed curriculum ontology and defined the ontology mapping rules by semantic relationships that can be established between curricula. After completing the steps above, an ontological system was built by converting input curriculum data to ontology. Retrieve BoK in KA from TQF: HEd and course descriptions which were compared to WordNet. In the end, calculate the semantic similarity values with using extended Wu & P’s algorithm [Wu and Palmer, 1994]. To calculate the semantic similarity between words, Pawar and Mago used Synsets from WordNet. The method simulates supervised learning through the use of corpora. Additionally, they used the NLTK-implemented max similarity algorithm to determine the sense of the words [Pawar and Mago, 2018].

One important component in curricula is the Learning Outcomes (LOs). It defines what students are expected to learn by taking the course. There are six layers in a hierarchical structure in Bloom’s Taxonomy (Remembering, Understanding, Applying, Analysing, Evaluating, and Creating) [Lasley, 2013]. Semantic technologies are used along with the Bloom Taxonomy to calculate the similarity of learning outcomes between courses. Pawar and Mago [Pawar and Mago, 2018] propose a semantic
similarity metric using WordNet\textsuperscript{18} to generate a score to compare LOs based on the Bloom taxonomy. Similarly, Mandić [Mandić, 2018] proposed the taxonomic structural similarity between curricula by applying the revised Bloom’s taxonomy which has the adjustment in cognitive levels of learning.

This section partially presented some of the works found in the literature that used semantic technologies to help university stakeholders design curricula for Computer Science. The following chapters present the approaches and analysis carried out using previous works as inspiration.
Chapter 3

Approaches

This chapter contains two sections. The first section introduces the dropout prediction task and a series of techniques to predict students’ performance. The second part presents semantic techniques that can be used to analyze the relationship between Computer Science courses.

3.1 Dropout Prediction

This section presents an approach to predicting dropout. We illustrate the entire dropout prediction workflow followed by the description of the dataset used and the corresponding pre-processing measurements. We then present an SVM-based genetic algorithm (GA) for feature selection and our LSTM approach for dropout prediction.

3.1.1 Procedure Overview

As illustrated in Figure 3.1, the entire dropout prediction workflow is composed of four steps. Details of each step will be explained in the following sections.

Briefly, the first step of dropout prediction is responsible for data pre-prepocessing including obtaining datasets. This step uses data wrangling and machine learning (ML) techniques. The second step is responsible for the feature selection. It uses the pre-processed data outputted in the first step. Steps 3 and 4 are merged together and start with training and testing the Long Short-term Memory (LSTM) and Fully Connected (FC) neural network.

3.1.2 Data Pre-processing

Before introducing the data pre-processing, we introduce the data set used for this experiment. Figure 3.2 presents a few instances of the dataset used.
**Approaches**

Figure 3.1: The workflow of dropout prediction

Figure 3.2: Dataset snippet
§3.1 Dropout Prediction

(a) Student distribution by year

(b) Record distribution by year

Figure 3.3: Distribution by year

(a) Student distribution by degree

(b) Dropout distribution by year

Figure 3.4: Student distribution by degree Figure 3.5: Dropout distribution by year

The dataset used in this thesis has been used in previous research [Manrique et al. 2019b] and is provided by a Brazilian university. It contains 248,730 academic records of 5,582 students enrolled in six distinct degrees from 2001 to 2009. The dataset is in Portuguese and we will translate the main terms for better understanding.

The dataset contains 32 attributes in total: cod_curso (course code); nome_curso (course name); cod_hab (degree code); nome_hab (degree name); cod_enfase (emphasis code); nome_enfase (emphasis name); ano_curriculo (curriculum year); cod_curriculo (curriculum code); matricula (student identifier); mat_ano (student enrolment year); mat_sem (semester of the enrolment); periodo (term); ano (year); semestre (semester); grupos (group); disciplina (discipline/course); semestre_recomendado (recommended semester); semestre_do_aluno (student semester); no_creditos (number of credits); turma (class); grau (grades); sit_final (final status (pass/fail)); sit_vinculo_atual (current status); nome_professor (professor name); cep (zip code); pontos_enem (national exam marks); diff (difference between semesters and students performance); tentativas (number of attempts in a course); cant (previous course); count (count);
identificador (identifier); nome_disciplina (course name).

Most of the data is related to the status of a student in a give course and degree. The "semestre" attribute is related to the semester a student took a course and has been previously used to create a time series of students Manrique et al. [2019a]. The attribute "sit_vinculo_atual" indicates that there are 12 status of enrolment where three of them ("DESLIGADO","MATRICULA EM ABANDONO","JUBILADO") represent the dropout status. Grades is scaled from 0 to 10 inclusive where 10 is the maximum grade. The dataset is anonymous and the identifiers do not allow re-identification. The students' ids are encoded with dummy alphanumeric code such as "aluno1010" ("student1010"). The dataset was profiled by year and degree and is presented in Figure 3.3 - 3.5.

To pre-process the data, we first split and reorganised the data set. We split the dataset into three parts: CSI ("Information Systems"), ADM ("Management"), and ARQ ("Architecture") datasets. After that we removed duplicate data, resolved data inconsistencies and removed outliers. For example, in the attribute "grades", a few marks were greater than 10, an invalid instance given the university marking scheme, and therefore removed. Attributes were also removed such as "grupos" as it was irrelevant and basically a copy of another attribute ("disciplinas"). Table 3.1 presents all information of the data-preprocessing step.

Another technique used was data validation where inappropriate data is identified and removed. As part of the data pre-processing step, we also converted categorical to numerical data. For this step we used a tool named LabelEncoder which is an extensive library of Sklearn.

For the dropout status attribute ("sit_vinculo_atual"), the instances were replaced by 0 (dropout) and 1 (enrolled). With respect to data imputation, the Random Forest (RF), which is a popular Machine Learning algorithm for constructing multiple decision trees in training, was employed to implement. As for normality or non-normality, linearity or non-linearity type of data, RF has a better performance than other algorithms in terms of handling imputation [Pantanowitz and Marwala 2009], [Shah et al., 2014], and [Hong and Lynn 2020]. To be specific, the sequence of imputation is from the least column to the ones with more missing data, filling the missing values with

---

Table 3.1: Data wrangling techniques used in pre-processing.

| Technique         | Operation                      |
|-------------------|--------------------------------|
| Data Cleaning     | outliers/duplicates correction |
| Data Validation   | inappropriate data identification |
| Data Enrichment   | Data enhancement               |
| Data Imputation   | missing values filling         |
| Data Normalisation| data re-scale                  |

---

[1] https://scikit-learn.org/
0 in other columns, and then running the RF and performing iterations until each missing value will be handled. The procedure of imputation is shown in Algorithm 1. Next comes to group the data by "course" and "semester" attributes.

The final step normalizes all the input data except the "sit_vinculo_atual" and "semestre" with the z-score method as z-score can improve the model performance than other techniques [Imron and Prasetyo, 2020], and [Cheadle et al., 2003]. The z-score method is computed based on a set of values $S = \{x_1, x_2, ..., x_n\}$, the mean of the set ($\mu$), and the standard deviation ($\sigma$). The formula is presented in what follows.

$$z = \frac{x - \mu}{\sigma} \quad (3.1)$$

The last pre-processing step applied to the dataset is the data enrichment step. As Table 3.1 shows, the number of instances in the CSI dataset is very small and not sufficient for training/testing purposes. For this, we used the Synthetic Minority Oversampling Technique (SMOTE) to increase the number of instances of the CSI dataset. Briefly, SMOTE generates new instances based on real ones. SMOTE considers the minorities to generate and balance the dataset with new instances.

Algorithm 1 Procedure of Imputation

| Input: Dataset with missing values |
|-----------------------------------|
| Output: Complete dataset with predicted values |

1. procedure
2. Get the order of imputation column by the number of missing values
3. do
4. Fill NULL with 0 in all NULL except the column which has least number of missing values
5. Train RF to predict the missing values
6. while Number of missing values > 0 ➔ Perform Iteration again

3.1.3 Feature Selection

After the data is pre-processed, to reduce the time consumption and increase the computational efficiency in training, it is necessary to apply feature selection to remove irrelevant features. Consequently, Support Vector Machine(SVM)-based Genetic Algorithm (GA) will be introduced in this section.

In GA, optimization strategies are implemented based on the simulation of evolution by natural selection for a species. Table /reftab:ga lists popular GA operators for creating optimal solutions and solving search problems based on biologically inspired principles such as mutation, crossover, and selection. Moreover, as its nature mentioned previously, the application of GA for feature selection is popular and was proven to improve the model performance in [Babatunde et al., 2014], [Huang et al., 2007], and [Leardi 2000]. The whole procedure is displayed in Figure 3.6.

In this thesis, to perform a GA-based feature selection, a set of techniques was deployed. First, a population was randomly generated as the initial solution whose
size is 1,000, followed by binary encoding which determines the chosen feature as 1, and 0 represents the column that will not be chosen. The next step is to calculate the fitness of each individual in the population by performing SVM in the three datasets (ADM, CSI, and ARQ). SVM is a supervised linear machine learning technique that is most commonly employed for classification purposes, and it has good performance in the fitness calculation in feature selection [Tao et al., 2019]. It is defined as a linear classifier with a maximum interval on the feature space (when a linear kernel is used), which is basically an interval maximization strategy that results in a convex quadratic programming problem. Given a training dataset, \( D = \{(x_{11}, x_{12}, \ldots, x_{1n}, y_1), (x_{21}, x_{22}, \ldots, x_{2n}, y_2), \ldots, (x_{m1}, x_{m2}, \ldots, x_{mn}, y_n)\}, y_n \in \{-1, +1\} \), where \( m \) is the number of samples in the dataset, and \( n \) is the number of features. In this experiment \( n = 27 \). The target is to get a decision boundary so as to separate the samples into different areas. The separator in two-dimensional space is a straight line and can be written as formula 3.2:

\[
mx + c = 0 \tag{3.2}
\]

Once mapping the separator to the \( n \)-dimensional space, it will become the hyperplane separator \( H \). It can be written as formula 3.3:

\[
H : w^T x + b = 0 \tag{3.3}
\]

where \( w = (w_1, w_2, \ldots, w_n) \) where \( w \) is the vector determining the hyperplane direction, The hyperplane is located at the origin after a displacement term \( b \) is applied. By determining the vector \( w \) and the bias \( b \), the division hyperplane is denoted as \((w, b)\). In the sample space, a given point vector \( \varnothing(x_o) \) is the distance from the hyperplane.

\[
d_H(\varnothing(x_o)) = \frac{|w^T \varnothing(x_o) + b|}{\|w\|_2} \tag{3.4}
\]

where \( \|w\|_2 \) is the 2-norm which can be defined by \( w \) as in formula 3.5:

\[
\|w\|_2 = \sqrt{w_1^2 + w_2^2 + \ldots + w_n^2} \tag{3.5}
\]

Suppose hyperplane \((w, b)\) can classify the training dataset, so as to \((x_o, y_o) \in D\), the following corollary can be obtained as below:

\[
\begin{cases}
  w^T \varnothing(x_o) + b \geq 1, & y_m = +1 \quad \text{if correct} \\
  w^T \varnothing(x_o) + b \leq 1, & y_m = -1 \quad \text{if incorrect}
\end{cases} \tag{3.6}
\]

The vectors close to the hyperplane make the formula 3.6 hold are named "support vectors". The sum of the distance of 2 vectors which are from different areas to the hyperplane is called "margin", which is defined in the formula 3.7:

\[
\gamma = \frac{2}{\|w\|_2} \tag{3.7}
\]
Table 3.2: Genetic Algorithm techniques, descriptions, goals, and specific method used in this study

| Operator      | Description                                      | Goal                                      | Method deployed                          |
|---------------|--------------------------------------------------|-------------------------------------------|------------------------------------------|
| Initialization| initialization of population acquisition         | to generate a set of solutions            | Randomly generated initialization         |
| Encoding      | representation of the individuals               | to convey the necessary information       | Binary Encoding                          |
| Fitness       | the degree of health of individuals              | to evaluate the optimality of a solution  | Support Vector Machine (SVM)             |
| Crossover     | parents are chosen for production                | to determine which solutions are to be preserved | Uniform Crossover                        |
| Selection     | solutions are selected for production           | to select the individuals with better adaptability | Proportional Selection                   |
| Mutation      | a gene is deliberately changed                   | to maintain diversity in the population set | mutation rate setting-up                 |
| Termination   | a process stops the evolution                    | to terminate and output the optimal outcomes | Maximum generations                      |

To obtain a partitioned hyperplane with maximum margin, that is, to find parameters $w$ and $b$ that comply with the constraint in equation 3.6 such that the optimal hyperplane classifies all the points in $D$ correctly, namely:

$$w^* = \arg \max_w \gamma, \text{s.t.} \min_{y \in D} \left[ w^T \varnothing(x_o) + b = 1 \right]$$ (3.8)

Followed by the fitness computation with SVM in 5-fold cross-validation, which is a technique to prevent overfitting, the proportional selection was conducted to get the individuals, the process is similar to the roulette wheel, that is, individuals with higher fitness ratings will have the greater chance to be selected, the formula of this process is shown in equation 3.9:

$$q_s(x_i(t)) = \frac{f_s(x_i(t))}{\sum_{j=1}^{n_s} f_s(x_j(t))}$$ (3.9)

where $n_s$ is a population’s total number of chromosomes whose initial number is 1,000, $q_s(x_i)$ is the possibility of $x_i$ being selected, and $f_s(x_i)$ is the fitness of $x_i$ to yield float type of value greater than 0.

After completing the selection, the uniform crossover was performed to get the offspring between a pair of selected parents. Specifically, each gene is selected at random from one of the corresponding genes on every parent chromosome as shown in Figure 3.7. Note that this method will only yield one offspring and the cross rate is 0.75. Reproduction is followed by mutation which certain gene(s) will be changed randomly by the setting rate, in this thesis, is 0.002. The final step is the terminal criterion, when the maximum generation is reached, the population has evolved 100 generations, and an optimal subset will be generated as the output. The entire workflow is illustrated in Figure 3.8.

3.1.4 Training and Test

After feature selection, 3 subsets corresponding to the 3 used datasets in this study have been generated with optimal features. In this step, it’s necessary to train the model with processed data so as to predict dropout.

To begin with, Long short-term memory (LSTM) is used for the training model in this study, LSTM is a special type of recurrent neural networks (RNNs). Unlike tradi-
Approaches

Figure 3.6: Procedure of GA

Figure 3.7: Uniform Crossover
Figure 3.8: The workflow of GA+SVM feature selection
approaches

Table 3.3: The components of a LSTM cell

| Component       | Description                  | Denote |
|-----------------|------------------------------|--------|
| Forget Gate     | activation vector with sigmoid | $f$    |
| Candidate Layer | activation vector with Tanh   | $\tilde{C}$ |
| Input Gate      | activation vector with sigmoid | $I$    |
| Output Gate     | activation vector with sigmoid | $O$    |
| Hidden state    | hidden state vector          | $H$    |
| Memory state    | cell state vector            | $C$    |
| Current Input   | vector                       | $X$    |

Figure 3.9: The internal architecture of LSTM cell [Sanjeevi, 2018]

As illustrated in Figure 3.9, LSTM consists of the following components which has been listed in Table 3.3. The inputs at each time sequence are comprised of 3 elements, that is, $X_t$, $H_{t-1}$, and $C_{t-1}$. With regards to outputs, $H_t$ and $C_t$ are exported by the LSTM. Note that there are 8 weight $W$ parameters in total, 4 associated with hidden state and others linked with the input. Moreover, 4 bias $b$ will be used in one unit. All the $W$ and $b$ will be initialized randomly in the first place, and will be adjusted by back-propagation. To prevent gradients exploring when gradients reach the designed threshold, which is 1.01, clipping the gradients.

Once execution in LSTM during one sequence $t$ with using the dataset from section 3.1.3. The first move is to determine which information will be forgotten as...
formula 3.10 illustrates. This step is executed by the forget gate, $H_{t-1}$ and $X_t$ pass the gate to get an output between 0 and 1.

$$f_t = \sigma(W_f \cdot [H_{t-1}, x_t] + b_f)$$ (3.10)

where $t$ is between 0 to 31 inclusive which indicates there are 32-time steps per student representing the information across 8 semesters and 4 courses each semester. Regarding the students who dropped out midway or have not finished all the semesters, padding time series sequences to make sure the total time step is the same. The $x_t$ is the input, whose size initial input ($x_0$) equals the number of features in the input dataset. Furthermore, there are two hidden layers and the hidden size is 50 which determines the size of the $H_0$ and $C_0$.

The second step is to decide what new messages will be used for storing in the cell state. First, input gate will get the update values $i_t$, followed by a $\text{tanh}$ layer creates a vector of new candidate values $\tilde{C}_t$, these values will be combined and the state updated. The formula for this is shown in 3.11 and 3.12:

$$i_t = \sigma(W_i \cdot [H_{t-1}, x_t] + b_i)$$ (3.11)

$$\tilde{C}_t = \text{tanh}(W_C \cdot [H_{t-1}, x_t] + b_C)$$ (3.12)

Next step to update the $C_{t-1}$, and create a new cell state $C_t$. According to the formula 3.13 combining the previous outputs $f_t, i_t$, and $\tilde{C}_t$, by a series of arithmetic operations, to get the final cell state $C_t$, which will be the input in the next step.

$$C_t = f_t \ast C_{t-1} + i_t \ast \tilde{C}_t$$ (3.13)

The final step is to output the hidden state. As shown in formula 3.14 and 3.15 similar to step 1 and 2, carrying $H_{t-1}$ and $X_t$ as inputs, to pass output gate with sigmoid which will make a decision regarding the parts of outputs. Then output is transited to a $\text{tanh}$ layer to acquire the hidden state $H_t$, which will be used as the input in the time step or another neural network.

$$O_t = \sigma(W_o \cdot [H_{t-1}, x_t] + b_o)$$ (3.14)

$$H_t = O_t \ast \text{tanh}(C_t)$$ (3.15)

In this study, after a series of hyper-parameter tuning, the final output hidden state will be regarded as the input of a 3-layer fully connected (FC) neural network to predict the dropout. The hidden size in FC is 128, in the first layer, the sigmoid is deployed. Likewise, ReLU is applied in the output layer. As for loss measurement, as illustrated in formula 3.16 Mean Squared Error (MSE) is used to compute the loss:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$ (3.16)

where $n$ is the number of samples, $Y_i$ is the observed value, and $\hat{Y}_i$ is the predicted
After computing the loss, the back-propagation will be executed to adjust the weights by using Adam as the optimizer. Furthermore, by introducing a dropout layer on outputs of each LSTM layer apart from the last layer, with a dropout possibility equal to 0.7, to prevent overfitting. Figure 3.10 visualizes the process of once training.

### 3.2 Curriculum Semantic Analysis

Identifying dropout is an important step to understand the reason why students fail and what can be done to increase graduation rates. We argue that the sequence of courses taken by students may influence student attrition. This section presents an approach to create a sequence of courses based on its descriptions. The idea is that in the future be able to correlate dropout rates, the sequence of courses taken by students and the course content/description.

We divide this section into two parts. The first part introduces methods to measure the similarity between course pairs using the Bidirectional Encoder Representations from Transformers (the so-called BERT language model). The second part is responsible for ordering course pairs by employing Semi-Reference Distance (SemRefD) [Manrique et al., 2019b].

#### 3.2.1 Procedure Overview

As Figure 3.11 and 5.2 show, the first move is to unify BERT to conduct similarity measurement. It involves three steps, that is, dataset acquirement, sentence embedding, and similarity measurement. Likewise, the second step uses the same dataset, the first process is to extract entities, followed by concept comparison and the sequence will be identified between courses in the end.
3.2.2 Similarity Measurement

To compare the courses description, it is necessary to encode the contextual data into vectors so as to be comparable in a semantic way. To achieve this, we make use of BERT for sentence embedding.

BERT was proposed by Google [Devlin et al., 2018], which is a revolutionary pre-training model that uses multi-head attention-based transformer to learn contextual relations between words or sentences in the context. Transformers were also proposed by Google [Vaswani et al., 2017]. It consists of an encoder, which is a bidirectional RNN, and a decoder. There is a scaled dot-product attention layer and a feedforward neural network layer in the encoder. With regards to self-attention, it unifies the matrix representation to calculate scoring and final output embedding in one step as formula 3.17 illustrates:

$$Attention(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$  \hspace{1cm} (3.17)

where $K$ and $V$ create a key-value pair of input of dimension $d_k$, $Q$ stands for the query. The output is a weighted sum of values.

Compare with computing the attention one time, the multi-head mechanism which is used in BERT, goes through the scaled dot-product attention several times at the same time and creates separate embedding matrices that are combined into one final output embedding as formula 3.18 shown:

$$MultiHead(Q, K, V) = \text{Concat}(head_1, ..., head_h)W^O$$ \hspace{1cm} (3.18)

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$ and $W^Q$, $W^K$, $W^V$, and $W^O$ are separate weight matrices.
BERT contains 12 stacked encoders in the base version, 24 stacked encoders in a larger version. Transformer encoders read each sequence of words at once instead of the sequential reading of text input as in directional models. With this characteristic, a model is able to determine a word’s context based on all of its surrounding information as shown in Figure 3.13.

To start with, the datasets this study uses ANU Course & Program Website[^2]. Furthermore, this study takes Computer Science courses into consideration, hence, course code beginning with "COMP" will be considered. After using a tokenizer to segment the full text into sentences and applying BERT on sentences as such for encoding, the vectors of sentences are constructed. Next, the vectors will be computed by cosine similarity measurement which aims to get the distance, the opposite of similarity as equation 3.19 specifies:

\[
\cos(t, e) = \frac{t \cdot e}{\|t\| \|e\|} = \frac{\sum_{i=1}^{n} t_i e_i}{\sqrt{\sum_{i=1}^{n} (t_i)^2} \sqrt{\sum_{i=1}^{n} (e_i)^2}}
\]  

(3.19)

where \(t\) and \(e\) are sentence vectors which were generated by BERT, the output ranges within \([-1, 1]\) depicting the degree of contextual similarity.

To get the entire contextual similarity, the average similarity among sentences will be calculated. After the processes above, the course similarity is obtained.

### 3.2.3 Prerequisite Identification

Despite the similarity between courses having been computed, the sequence between highly similar courses remains uncertain, hence, identifying the prerequisite depen-
dency (PD) between two highly similar courses becomes necessary.

Regarding PD, it is a relation between two concepts where, in education, the prerequisite concept should be taught first. For instance, *Binary Tree* and *Red-black Tree* are two concepts belonging to the data structure field in Computer Science, the latter should be introduced after the former. By measuring the prerequisite relationship between courses, the curriculum will be analyzed as a whole.

To begin with, similar to the similarity measurement, the same dataset, and pre-processing techniques were used in this study. Subsequently, the entities behind the text will be extracted, and the tool named *TextRazor* was employed to complete this task. *TextRazor* is an NLP-based API developed to segment text and capture conceptual terms. Then a technique called Semi-Reference Distance (SemRefD) was conducted to measure semantic PD between the entities of two courses in DBpedia\(^3\) Knowledge Graph (KG).

With respect to KG, it is known as a semantic network, which stands for a real-world entities network, in this study, they refer to concepts, and illustrate the relationship between them. DBpedia is one of the main KGs on the Semantic Web and provides a wide variety of topics that can be used to encompass courses from various fields of study. Moreover, it is also very tolerant and inclusive of many different semantic properties, which empower the liberal connection to multiple types of concepts. Using the given concept as a query in DBpedia, there will be two lists storing the candidate concepts, that is, the direct list and the neighbor list. As for a direct list, in which a list of concepts sharing a category with the given concept will be returned. For the latter list, in which the candidate list is expanded by adding concepts linked to the target through non-hierarchical pathways up to \(m\) hops [Manrique et al., 2019a]. The path length parameter \(m\) decides the maximum length of the path between the target concept and the farthest candidate concept to consider, in this study, \(m = 1\). Soon after acquiring the candidate lists, the SemRefD will be performed to compute the degree of prerequisite in the next step.

SemRefD was presented by Manrique et al. [Manrique et al., 2019a] based on Reference Distance (RefD) which was proposed by Chen et al. [Chen et al., 2018] as defined in formula (3.20) by inputting two concepts which are denoted \(c_A\) and \(c_B\) as below:

\[
RefD(c_A, c_B) = \sum_{j=1}^{k} \frac{i(c_j, c_A)s(c_j, c_B)}{s(c_j, c_B)} - \sum_{j=1}^{k} \frac{i(c_j, c_A)s(c_j, c_B)}{s(c_j, c_B)}
\]  

(3.20)

An indicator function \(i(c_j, c_A)\) indicates that there is a relationship between \(c_j\) and \(c_A\), and a weighting function \(s(c_j, c_A)\) indicates whether there is a relationship. The values of \(RefD(c_A, c_B)\) range from -1 to 1. According to Figure 3.14, \(c_B\) is more likely to be a prerequisite for \(c_A\) the if it is closer to 1.

RefD does not take into account the semantic properties of DBpedia to determine whether two concepts have a prerequisite dependency. SemRefD does. In the weighting function \(1(c_j, c_A)\), the common neighbors’ concepts in the KG KG hierarchy

\(^3\)https://www.dbpedia.org/
are considered, while in the indicator function $1(c_j, c_A)$, the property paths between concept target and related concepts are considered [Manrique et al., 2019a].

As a result, all concepts from the two courses will be compared and summed up to reveal their order of them, that is, whether A is a prerequisite of B or B is a prerequisite of A.
Results and Discussion

In this chapter, the results will be presented in 2 sections from experiments described in Chapter 3, namely, dropout prediction and curriculum semantic analysis along with discussions in detail. Notably, the results yielded by methods in the systematic review have been indicated in Section 2.2.1. Additionally, the experimental environment will be introduced initially.

4.1 Experimental Environment

The experiments are conducted on the following devices and corresponding hardware as shown in Table 4.1. Specifically, the paper retrieving process in systematic review is completed on the Apple Macbook Pro. The rest of the experiments are conducted on the server.

4.2 Dropout Prediction

This section contains two subsections. The evaluation criteria will be presented firstly, followed by the results from feature selection and dropout prediction.

4.2.1 Evaluation Metrics

Dropout prediction is evaluated by accuracy, precision, recall, and F1 score based on the confusion matrix as shown in Figure 4.1. Specifically, the \( TP, \) \( TN, \) \( FP, \) and \( FN \) are defined as below:

- True Positive (\( TP \)): Models predicted positive outcomes and are accurate.
- False Positive (\( FP \)): Models predicted positive outcomes and are inaccurate.
- True Negative (\( TN \)): Models predicted negative outcomes and are accurate.
- False Negative (\( FN \)): Models predicted negative outcomes and are inaccurate.

Table 4.1: Experimental Environment

| Model               | CPU                          | GPU                          | Memory | Hard disk size | System                |
|---------------------|------------------------------|------------------------------|--------|----------------|-----------------------|
| Server              | Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz | Tesla V100-DGXS-32GB*4  | 251GB  | 10TB           | Ubuntu 18.04.6       |
| Apple Macbook Pro    | M1 chip                      | 8-core GPU                   | 16GB   | 512GB          | macOS Monterey 12.0.1 |

27
Results and Discussion

| Actual Class | Predicted Class |
|--------------|----------------|
| 0            | 1              |
| FP           | TP             |
| TN           | FN             |

**Figure 4.1:** Confusion Matrix

False Negative (FN): Models predicted negative outcomes and are inaccurate.

To be specific, accuracy refers to the proportion of correctly predicted observations in whole samples. The precision measures the correctly predicted TP samples to the total positive ones. Recall depicts the sensitivity of the model by measuring the correctly predicted positive observations to all the observations. F1 score measure the overall performance of model The equations are as formula 4.1 - 4.4 shown below:

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.1}
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}
\]

\[
F1 = \frac{2 \times (\text{Recall} \times \text{Precision})}{\text{Recall} + \text{precision}} = \frac{TP}{TP + \frac{TP + TN}{2}} \tag{4.4}
\]

4.2.2 Experimental Results and Discussions

As mentioned in Section 3.1.2, ADM, ARQ, and CSI datasets will be used to evaluate the proposed models in terms of feature selection and dropout prediction. The first section is regarding feature selection, which was proposed in Section 3.1.3, followed by dropout prediction, as mentioned in section 3.1.4.

4.2.2.1 Feature Selection

The results of feature selection as listed in Table 4.2. After 100 generations of evolution in a population comprised of 1000 individuals, as for the performance of final optimal individuals among these datasets, ARQ has the best performance in terms of accuracy (90.61), F1 score (95.03) with the least number of dropped features, indicating that this individual fits the environment most than the others. By comparison, CSI has the least performance in terms of measurement metrics. Observing the lower score in precision than recall across used datasets, indicates that the model has its bias in terms of the prediction preference in positive instead of negative, and emphasises the importance of the balance of datasets. Furthermore, by acquiring the ratio of negative in CSI (24%), ADM (13%), and ARQ (10%), also proves this inference. Thus, employing a model on a balanced dataset may improve its performance. As for ADM, it has the nearly same recall as ARQ’s and the best recall score in this experiment. Overall, the results turn out this model align with this experiment.
Table 4.2: Results of ADM, ARQ, and CSI by SVM

| Dataset | Accuracy(%) | Precision(%) | Recall(%) | F1 score(%) | No. dropped features |
|---------|-------------|--------------|-----------|-------------|---------------------|
| ADM     | 88.42       | 88.75        | 99.52     | 93.83       | 10                  |
| ARQ     | 90.61       | 91.14        | 99.26     | 95.03       | 6                   |
| CSI     | 81.23       | 82.28        | 97.84     | 89.36       | 12                  |

4.2.2.2 Dropout Prediction

Likewise, this experiment used the same datasets on the dropout prediction. As dropout prediction is one of the main objectives of this project, the performance of the proposed model is important to this project. As illustrated in 4.2 - 4.7 these outcomes on training reveal the capacity of the model, whose end of the abscissa is the \( \text{epoch}(\text{every } 10 \text{epochs}) \times \text{timestep} \). Furthermore, the performance of the model on test further validates the suitability of the model on datasets as shown in Table 4.3.

According to the accuracy and loss during training, we can observe that the model always converges after 10 epochs. As for the accuracy in ADM and ARQ datasets, which starts from below 10% initially, then fall to the lower, and rises to close to the top afterwards. For ARQ, the curve is similar to some extent. After investigation, multiple reasons are identified, which leads to the abnormal curve. For instance, the dropout rate is high, which shows that the model drops the well-functional neurons, or the batch size is large. In this study, the time step is fixed as mentioned in section 3.1.4 which indicates that the batch size is uncontrollable. In addition, after hyper-parameter tuning, high dropout rate (0.7) enables the best results of the model, which shows that selecting the current dropout rate is a trade-off decision. Thus, this study chose to keep a better performance by using the current dropout rate.

In terms of loss, after reaching the convergence, it will repeat the loss curve along with the rest of the time step. Apart from the repetition of loss, we also observed the unstable curve during one epoch training, which is named multiple descent \cite{Chen2020}. The reason behind the anomaly may vary. For example, we assumed there are minima \( \theta_1 \) and \( \theta_2 \), when the distance between two minima \( d = \| \theta_1 - \theta_2 \| \) is very small, and the learning rate is not small enough, which leads to cross a local \( \theta_1 \) and arrives in \( \theta_2 \) eventually. Also, this phenomenon can be caused by datasets \cite{Chen2020}.

With respect to the performance of the proposed model in the test, the model performs well in ADM and ARQ, whose best accuracy reaches 92.83% and 97.65% respectively. Notably, the accuracy of ARQ improves the result in Manrique’s previous work (95.2%) by 2.45% \cite{Manrique2019}. Finally, we captured a potential improvement for the model by adopting dynamic time steps instead of fixed steps to make full use of the dataset. Overall, the model proposed by this study is suitable for the current datasets.
Results and Discussion

Figure 4.2: Accuracy of LSTM+FC on ADM

Figure 4.3: Loss of LSTM+FC on ADM

Figure 4.4: Accuracy of LSTM+FC on ARQ

Figure 4.5: Loss of LSTM+FC on ARQ

Figure 4.6: Accuracy of LSTM+FC on CSI

Figure 4.7: Loss of LSTM+FC on CSI
Table 4.3: Results of ADM, ARQ, and CSI

| Dataset | Avg. acc (10 iter.)(%) | Avg. acc (100 iter.)(%) | Avg. acc (200 iter.)(%) | Top. acc(%) |
|---------|------------------------|------------------------|------------------------|-------------|
| ADM     | 88.76                  | 88.24                  | 90.17                  | 92.83       |
| ARQ     | 94.84                  | 95.76                  | 95.19                  | 97.65       |
| CSI     | 74.98                  | 75.84                  | 76.31                  | 81.52       |

4.3 Curriculum Semantic Analysis

This section contains two subsections, that is, similarity measurement, as mentioned in Section 3.2.2, and prerequisite identification as mentioned in Section 3.2.3.

4.3.1 Similarity Measurement

After the encoding between sentences captured by BERT and the average similarity between courses computed hereby, the results have been acquired and visualized in heat map format from Figure 4.8 ranging from 0.8265 to 1.0, which lists the result of all the course comparison, to Figure 4.9 - Figure 4.12 as below.

In 1000-level courses comparison, we can see that COMP1110 (Structured Programming) has the closest average distance to the rest of courses. In contrast, the COMP1600 (Foundations of Computing) has the farthest from the others. From course content to interpretation, similarity among COMP1100 (Programming as Problem Solving) to COMP1100 is one of the most fundamental programming courses among 1000-level courses, and COMP 1600 focuses on the mathematical perspective.

With respect to the 2000-level course comparison, COMP2100 (Software Design Methodologies) has the closest average distance to the rest of the courses. On the contrary, the COMP2560 (Studies in Advanced Computing R & D) has the farthest to others. To sum up, the overall similarity between 2000-level courses becomes lower than 1000-level’s, which indicates that curriculum differentiation has appeared.

As regards 3000-level and 4000-level courses, it can be seen that this trend has become more obvious and grown, which also aligns with real situations, for example, students’ knowledge grows, the course content deepens and divided by specifications, such as Data Science, Machine Learning, etc. In the meantime, students also have an interest to dive in, which aids students to make use of strengths in the region of interest and improve their academic performance in return. Finally, we identified an obstacle in automatic evaluation, which is a common limitation in studies in systematic review [Piedra and Caro, 2018], [Pata et al., 2013], [Yu et al., 2007], [Saquicela et al., 2018] and [Tapia-Leon et al., 2018].
Figure 4.8: All level courses similarity
4.3.2 Prerequisite Identification

As Section 4.3.1 presented, the similarity course has been computed, thus it is necessary to identify the prerequisite between two similar courses.

For this study, we selected 3 courses, that is, COMP1100, COMP1110, and COMP2100 to conduct this experiment as they have high similarity scores in the previous stage and they are all programming-oriented courses. The result is shown in Figure 4.13.

Between COMP1100 and COMP2100, whose similarity is 0.9479, the prerequisite score is 13.17. Based on the rule of thumb in the experiment, the score is very high,
**Results and Discussion**

**Figure 4.13:** All level courses similarity

| Concept Extracted | Similarity | Score (Course 1 -> Course 2) |
|-------------------|------------|-----------------------------|
| COMP 1100         | 117        | 0.9479                      | 13.17               |
| COMP 2100         | 124        |                             |                     |
| COMP 1110         | 118        | 0.9566                      | 12.06               |
| COMP 2100         | 124        |                             |                     |
| COMP 1100         | 117        | 0.9543                      | 4.4                 |
| COMP 1110         | 118        |                             |                     |

indicating that there is a strong relation in prerequisite in these two courses.

Similarly, COMP1100 and COMP2100 still have a very high score, which is 12.06, indicating that COMP1110 is also one of the prerequisites of COMP2100.

Regarding COMP1100 and COMP1110, the similarity is 0.9543 and the score is 4.4, which reveals that although there is a strong bond between these two courses, the prerequisite relationship is not as strong as the relationship with COMP2100 respectively.
Conclusion and Future Work

This study was undertaken to improve students’ academic performance by using AI and semantic technologies. To achieve this goal, we came up with three objectives, along with three implementations, that is, to predict students’ performance using grades from the previous semester, to model a course representation in a semantic way and compute the similarity, and to identify the sequence between two similar courses. For highlighting the research gaps in the Computer Science curriculum semantic analysis and contributing to the growing field of research, we conducted a systematic review regarding what semantic technologies are currently being used. A major finding of the study is that technologies used to measure similarity have limitations in terms of accuracy and ambiguity in the representation of concepts, courses, or curriculum. Our research fills this gap. Furthermore, this review also inspires us to think further regarding identifying the sequence between similar courses.

Regarding students’ academic prediction, we conducted a dropout prediction experiment on a dataset from a Brazilian university. Three clean datasets were generated after pre-processing. Then, LSTM was performed in order to predict dropouts based on a SVM-based GA feature selection. Taken together, the results from the experiment illustrate that LSTM has strong adaptability in predicting dropout as there is breakthrough progress in terms of accuracy, which improves the best accuracy by 2.45% over Manrique’s work \cite{Manrique2019} on the ARQ dataset. Due to unbalanced datasets, we also observed the limitations of this study. The model has a bias in feature selection, abnormal accuracy decline during training, and multiple descents in loss, which emphasizes the importance of the balance of datasets.

With respect to course similarity measurement, we deployed BERT, which has the strong power of input embedding, to encode the sentence in the course description from the Australian National University. We then used cosine similarity to obtain the distance between courses. As a result, we found that COMP1110 has the closest average similarity to the rest of the 1000-level courses. In 2000-level courses, we identified COMP2100 has the closest relationship with the rest. In terms of courses at the 3000 and 4000 level, since specialization has formed, comparing them is pointless. The results also align with this viewpoint. As regards limitations, at this stage we cannot evaluate the result.

The final step in this project is measuring the sequence between two similar courses. We employed the textRazor to extract entities from course description and
then used SemRefD, which was proposed and evaluated by [Manrique et al. 2019a], to measure the degree of prerequisite between two concepts. By deploying the model on COMP1100, COMP1110, and COMP2100 respectively, we established the relationship between these three courses. The results show that COMP1100 and COMP1110 have a strong prerequisite relationship with COMP2100, while the relationships between the two courses are inclined to be on the same level.

In terms of the future work, these technologies could potentially be used to analyse the curriculum of university programs, to aid student advisors, and to create recommendation systems that combine semantic and deep learning technologies. Furthermore, this study suggested that using GA+LSTM model could provide institutions with early detection for dealing with problems and retaining students. As for the experiment’s future refinement, first, in dropout prediction, dynamic time steps can be introduced in LSTM instead of using fixed steps to make full use of the dataset. Moreover, a more balanced dataset could be used in the experiment. We will continue to investigate and develop appropriate models to improve the results in the future.
Aeiad, E. and Meziane, F., 2016. Validating learning outcomes of an E-Learning system using NLP. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9612 (2016), 292–300. doi:10.1007/978-3-319-41754-7_27 (cited on pages 6 and 9)

BA, K. and Charters, S., 2007. Guidelines for performing systematic literature reviews in software engineering. 2 (01 2007). (cited on page 4)

Babatunde, O.; Armstrong, L.; Leng, J.; and Diepeveen, D., 2014. A genetic algorithm-based feature selection. International Journal of Electronics Communication and Computer Engineering, 5 (07 2014), 889–905. (cited on page 15)

Barb, A. S. and Kilicay-Ergin, N., 2020. Applications of natural language techniques to enhance curricular coherence. Procedia Computer Science, 168 (2020), 88–96. doi:https://doi.org/10.1016/j.procs.2020.02.263 https://www.sciencedirect.com/science/article/pii/S1877050920304026 "Complex Adaptive Systems“Malvern, PennsylvaniaNovember 13-15, 2019. (cited on pages 7, 8, and 9)

Cheadle, C.; Vawter, M. P.; Freed, W. J.; and Becker, K. G., 2003. Analysis of microarray data using z score transformation. The Journal of Molecular Diagnostics, 5, 2 (2003), 73–81. doi:https://doi.org/10.1016/S1525-1578(10)60455-2 https://www.sciencedirect.com/science/article/pii/S1525157810604552 (cited on page 15)

Chen, L.; Jianbo, Y.; Shuting, W.; Bart, P.; and Giles, C., 2018. Investigating active learning for concept prerequisite learning. In 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, 7913–7919. AAAI press. Funding Information: We gratefully acknowledge partial support from the Pennsylvania State University Center for Online Innovation in Learning. Publisher Copyright: Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.; 32nd AAAI Conference on Artificial Intelligence, AAAI 2018 ; Conference date: 02-02-2018 Through 07-02-2018. (cited on page 25)

Chen, L.; Min, Y.; Belkin, M.; and Karbasi, A., 2020. Multiple descent: Design your own generalization curve. CoRR, abs/2008.01036 (2020). https://arxiv.org/abs/2008.01036. (cited on page 29)

Chung, H. S. and Kim, J. M., 2014. Semantic model of syllabus and learning ontology for intelligent learning system. Lecture Notes in Computer Science (including subseries
Bibliography

Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, 8733 (2014), 175–183. doi:10.1007/978-3-319-11289-3_18 (cited on page 3)

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K., 2018. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805 (2018). http://arxiv.org/abs/1810.04805 (cited on pages 2 and 23)

Díaz-Mujica, A.; Pérez, M.; Bernardo, A.; Cervero, A.; and González-Pienda, J., 2019. Affective and cognitive variables involved in structural prediction of university dropout. Psicothema, 31 (10 2019), 429–436. doi:10.7334/psicothema2019.124 (cited on page 1)

Drennan, L. and Rohde, F., 2002. Determinants of performance in advanced undergraduate management accounting: An empirical investigation. Accounting and Finance, 42 (02 2002), 27–40. doi:10.1111/1467-629X.00065 (cited on page 1)

Fiallos, A., 2018. Assisted curricula design based on generation of domain ontologies and the use of NLP techniques. 2017 IEEE 2nd Ecuador Technical Chapters Meeting, ETCM 2017, 2017-Janua (2018), 1–6. doi:10.1109/ETCM.2017.8247474 (cited on pages 7 and 9)

Guberovic, E.; Turcincovic, F.; Relja, Z.; and Bosnic, I., 2018. In search of a syllabus: Comparing computer science courses. 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings, (2018), 588–592. doi:10.23919/MIPRO.2018.8400111 (cited on page 7)

Hao, X.; Meng, X.; and Cui, X., 2008. Knowledge Point Based Curriculum Developing and Learning Object Reusing. Knowledge Creation Diffusion Utilization, (2008), 126–137. (cited on pages 7 and 8)

Hedayati, M. H. and Mart, L., 2016. Ontology-driven modeling for the culturally-sensitive curriculum development: A case study in the context of vocational ICT education in Afghanistan. Proceedings of the 10th INDIACom; 2016 3rd International Conference on Computing for Sustainable Global Development, INDIACom 2016, , Ontology 101 (2016), 928–932. (cited on pages 6 and 7)

H.Gomaa, W. and A. Fahmy, A., 2013. A Survey of Text Similarity Approaches. International Journal of Computer Applications, 68, 13 (2013), 13–18. doi:10.5120/11638-7118 (cited on page 9)

Hong, S. and Lynn, H. S., 2020. Accuracy of random-forest-based imputation of missing data in the presence of non-normality, non-linearity, and interaction. BMC Medical Research Methodology, 20 (2020). (cited on page 14)

Huang, J.; Cai, Y.; and Xu, X., 2007. A hybrid genetic algorithm for feature selection wrapper based on mutual information. Pattern Recognition Letters, 28, 13 (2007), 1825–1844. doi:https://doi.org/10.1016/j.patrec.2007.05.011 https://www.sciencedirect.com/science/article/pii/S0167865507001754 (cited on page 15)
IMRAN, M. AND YOUNG, R. I., 2016. Reference ontologies for interoperability across multiple assembly systems. International Journal of Production Research, 54, 18 (2016), 5381–5403. doi:10.1080/00207543.2015.1087654 (cited on page 6)

IMRON, M. A. AND PRASETYO, B., 2020. Improving algorithm accuracy k-nearest neighbor using z-score normalization and particle swarm optimization to predict customer churn. (cited on page 15)

JOHNSON, M. AND KUENNEN, E., 2006. Basic math skills and performance in an introductory statistics course. Journal of Statistics Education, 14, 2 (2006), null. doi:10.1080/10691898.2006.11910581 https://doi.org/10.1080/10691898.2006.11910581 (cited on page 1)

KARUNANANDA, A. S.; RAJAKARUNA, G. M.; AND JAYALAL, S., 2012. OntoCD - Ontological solution for curriculum development. International Conference on Advances in ICT for Emerging Regions, ICTer 2012 - Conference Proceedings, (2012), 137–144. doi:10.1109/ICTer.2012.6421412 (cited on pages 6, 7 and 8)

KAWINTIRANON, K.; VATEEKUL, P.; SUCHATO, A.; AND PUNYABUKKANA, P., 2016. Understanding knowledge areas in curriculum through text mining from course materials. In 2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), 161–168. doi:10.1109/TALE.2016.7851788 (cited on page 8)

LASLEY, T. J., 2013. Bloom’s Taxonomy. doi:10.4135/9781412957403.n51 (cited on page 9)

LEARDI, R., 2000. Application of genetic algorithm–pls for feature selection in spectral data sets. Journal of Chemometrics - J CHEMOMETR, 14 (09 2000), 643–655. doi:10.1002/1099-128X(200009/12)14:5/63.0.CO;2-E (cited on page 15)

LIANG, Y. AND MA, X., 2012. Teaching reform of software engineering course. ICCSE 2012 - Proceedings of 2012 7th International Conference on Computer Science and Education, 1936–1939. doi:10.1109/ICCSE.2012.6295452 (cited on page 6)

MAFFEI, A.; DAGHINI, L.; ARCHENTI, A.; AND LOHSE, N., 2016. CONALI Ontology. A Framework for Design and Evaluation of Constructively Aligned Courses in Higher Education: Putting in Focus the Educational Goal Verbs. Procedia CIRP, 50 (2016), 765–772. doi:10.1016/j.procir.2016.06.004 http://dx.doi.org/10.1016/j.procir.2016.06.004 (cited on pages 6 and 7)

MANDIĆ, M., 2018. Semantic web based software platform for curriculum harmonization *. ACM International Conference Proceeding Series, (2018). doi:10.1145/3227609.3227654 (cited on pages 6, 7 and 10)

MANRIQUE, R.; PEREIRA Nunes, B.; AND MARINO, O., 2019a. Exploring knowledge graphs for the identification of concept prerequisites. Smart Learning Environments, 6 (12 2019), 21. doi:10.1186/s40561-019-0104-3 (cited on pages 1, 2, 14, 25, 26, 29, 35 and 36)
**Manrique, R.; Pereira Nunes, B.; Marino, O.; Casanova, M.; and Nurmikko-Fuller, T., 2019b.** An analysis of student representation, representative features and classification algorithms to predict degree dropout. 401–410. doi:10.1145/3303772.3303800 (cited on pages 13 and 22)

**McGuinness, D. L. and van Harmelen, F., 2004.** OWL Web Ontology Language Overview. W3C recommendation, 10, February (2004). http://www.w3.org/TR/owl-features/ (cited on page 7)

**Nuntawong, C.; Namahoot, C. S.; and Brückner, M., 2016.** A web based cooperation tool for evaluating standardized curricula using ontology mapping. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9929 LNCS (2016), 172–180. doi:10.1007/978-3-319-46771-9_23 (cited on pages 6, 7, 8 and 9)

**Nuntawong, C.; Namahoot, C. S.; and Brückner, M., 2017.** HOME: Hybrid ontology mapping evaluation tool for computer science curricula. Journal of Telecommunication, Electronic and Computer Engineering, 9, 2-3 (2017), 61–65. (cited on pages 6, 7, 8 and 9)

**Nuntawong, C.; Snae, C.; and Brückner, M., 2015.** A semantic similarity assessment tool for computer science subjects using extended wu & palmer’s algorithm and ontology. Lecture Notes in Electrical Engineering, 339 (02 2015), 989–996. doi:10.1007/978-3-662-46578-3_118 (cited on pages 7, 8 and 9)

**Orellana, G.; Orellana, M.; Saquicela, V.; Baculima, F.; and Piedra, N., 2018.** A text mining methodology to discover syllabi similarities among higher education institutions. Proceedings - 3rd International Conference on Information Systems and Computer Science, INCISCOS 2018, 2018-Decem (2018), 261–268. doi:10.1109/INCISCOS.2018.00045 (cited on pages 8 and 9)

**Pantanowitz, A. and Marwala, T., 2009.** Missing data imputation through the use of the random forest algorithm. Advances in Computational Intelligence, 116 (01 2009). doi:10.1007/978-3-642-03156-4_6 (cited on page 14)

**Pata, K.; Tammets, K.; Laanpere, M.; and Tomberg, V., 2013.** Design principles for competence management in curriculum development. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8095 LNCS (2013), 260–273. doi:10.1007/978-3-642-40814-4_21 (cited on page 31)

**Paulk, M.; Curtis, B.; Chrissis, M.; and Weber, C., 1993.** Capability Maturity Model for Software, Version 1.1. (cited on page 6)

**Pawar, A. and Mago, V., 2018.** Similarity between learning outcomes from course objectives using semantic analysis, bloom’s taxonomy and corpus statistics. arXiv, (2018). (cited on page 9)
PIECH, C.; SPENCER, J.; HUANG, J.; GANGULI, S.; SAHAMI, M.; GUIBAS, L. J.; AND SOHL-DICKSTEIN, J., 2015. Deep knowledge tracing. CoRR, abs/1506.05908 (2015). http://arxiv.org/abs/1506.05908 (cited on page 20)

PIEDRA, N. AND CARO, E. T., 2018. LOD-CS2013: Multilearning through a semantic representation of IEEE computer science curricula. IEEE Global Engineering Education Conference, EDUCON, 2018-April (2018), 1939–1948. doi:10.1109/EDUCON.2018.8363473 (cited on pages 6, 7, 8, and 31)

SALES, A. R. P.; BALBY, L.; AND CAJUEIRO, A., 2016. Exploiting academic records for predicting student drop out: a case study in brazilian higher education. J. Inf. Data Manag., 7 (2016), 166–180. (cited on page 1)

SANJEEVI, M., 2018. Chapter 10.1: Deepnlp - lstm (long short term memory) networks with math. https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235 (cited on pages 5 and 20)

SAQUICELA, V.; BACULIMA, F.; ORELLANA, G.; PIEDRA, N.; ORELLANA, M.; AND ESPINOZA, M., 2018. Similarity detection among academic contents through semantic technologies and text mining. In IWSW. (cited on pages 6, 7, 9, and 31)

SEIDEL, N.; RIEGER, M.; AND WALLE, T., 2020. Semantic textual similarity of course materials at a distance-learning university. CEUR Workshop Proceedings, 2734 (2020). (cited on page 9)

SHAH, A. D.; BARTLETT, J. W.; CARPENTER, J. R.; NICHOLAS, O.; AND HEMINGWAY, H., 2014. Comparison of random forest and parametric imputation models for imputing missing data using mice: A caliber study. American Journal of Epidemiology, 179 (2014), 764 – 774. (cited on page 14)

SHIPLEY, B. AND IAN, W., 2019. Here comes the drop: university drop out rates and increasing student retention through education. https://www.voced.edu.au/content/ngv:84995 (cited on page 1)

TANG, A. AND HOH, J., 2013. Ontology-specific API for a curricula management system. 2013 2nd International Conference on E-Learning and E-Technologies in Education, ICEEE 2013, (2013), 294–297. doi:10.1109/ICLeTe.2013.6644391 (cited on pages 6 and 7)

TAO, Z.; HUILING, L.; WENWEN, W.; AND XIA, Y., 2019. Ga-svm based feature selection and parameter optimization in hospitalization expense modeling. Applied Soft Computing, 75 (2019), 323–332. doi:https://doi.org/10.1016/j.asoc.2018.11.001 https://www.sciencedirect.com/science/article/pii/S1568494618306264 (cited on page 16)

TAPIA-LEON, M.; RIVERA, A. C.; CHICAIZA, J.; AND LUJÁN-MORA, S., 2018. Application of ontologies in higher education: A systematic mapping study. In 2018 IEEE Global Engineering Education Conference (EDUCON), 1344–1353. doi:10.1109/EDUCON.2018.8363385 (cited on pages 8 and 31)
Vaquero, J.; Toro, C.; Martín, J.; and Aregita, A., 2009. Semantic enhancement of the course curriculum design process. In Proceedings of the 13th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems: Part I, KES ’09 (Santiago, Chile, 2009), 269–276. Springer-Verlag, Berlin, Heidelberg. doi:10.1007/978-3-642-04595-0_33 [cited on pages 6 and 7].

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I., 2017. Attention is all you need. CoRR, abs/1706.03762 (2017). http://arxiv.org/abs/1706.03762. (cited on page 23).

Vergel, J.; Quintero, G. A.; Isaza-Restrepo, A.; Ortiz-Fonseca, M.; Latorre-Santos, C.; and Pardo-Oviedo, J. M., 2018. The influence of different curriculum designs on students’ dropout rate: a case study. Medical Education Online, 23, 1 (2018), 1432963. doi:10.1080/10872981.2018.1432963 [cited on page 1].

Wang, Y.; Wang, Z.; Hu, X.; Bai, T.; Yang, S.; and Huang, L., 2019. A Courses Ontology System for Computer Science Education. 2019 IEEE International Conference on Computer Science and Educational Informatization, CSEI 2019, 3 (2019), 251–254. doi:10.1109/CSEI47661.2019.8938930 [cited on pages 6, 7, 8, and 9].

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. ACM International Conference Proceeding Series, (2014). doi:10.1145/2601248.2601268 [cited on page 5].

Wu, Z. and Palmer, M., 1994. Verbs semantics and lexical selection. In Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, ACL ’94 (Las Cruces, New Mexico, 1994), 133–138. Association for Computational Linguistics, USA. doi:10.3115/981732.981751 [cited on page 9].

Xia, K.; Huang, J.; and Wang, H., 2020. Lstm-cnn architecture for human activity recognition. IEEE Access, 8 (2020), 56855–56866. doi:10.1109/ACCESS.2020.2982225 [cited on page 20].

Yu, X.; Tungare, M.; Fan, W.; Pérez-Quinones, M.; Fox, E. A.; Cameron, W.; and Cassel, L., 2007. Using automatic metadata extraction to build a structured syllabus repository. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4822 LNCS (2007), 337–346. doi:10.1007/978-3-540-77094-7_43 [cited on page 31].
Final Project Description

This project aims to improve students’ performance by predicting students’ dropouts and curriculum semantic analysis.

As for dropout prediction, this project uses SVM-based Genetic Algorithm to perform a feature selection; followed by using LSTM to predict the dropout. The details is regarding conducted a dropout prediction experiment on a dataset from a Brazilian university. Three clean datasets were generated after pre-processing. Then, LSTM was performed in order to predict dropouts based on a SVM-based GA feature selection.

With respect to course similarity measurement, we deployed BERT, which has the strong power of input embedding, to encode the sentence in the course description from the Australian National University. We then used cosine similarity to obtain the distance between courses.

Regarding measuring the sequence between two similar courses. We employed the textRazor to extract entities from course description and then used SemRefD, which was proposed and evaluated by Manrique, to measure the degree of prerequisite between two concepts. By deploying the model on COMP1100, COMP1110, and COMP2100 respectively, we established the relationship between these three courses.

The outcomes of project are expected to use these technologies could potentially be used to analyse the curriculum of university programs, to aid student advisors, and to create recommendation systems that combine semantic and deep learning technologies. Furthermore, this study suggested that using GA+LSTM model could provide institutions with early detection for dealing with problems and retaining students.
Appendix 2

Contract
INDEPENDENT STUDY CONTRACT

Note: Enrolment is subject to approval by the Honours/projects co-ordinator

SECTION A (Students and Supervisors)

| UniID:     | a6809382 |
|------------|----------|
| FAMILY NAME: | Cheng    |
| PERSONAL NAME(S): | Yixin   |
| PROJECT SUPERVISOR (may be external): | Bernardo Pereira Nunes |
| COURSE SUPERVISOR (a RSCS academic): | Penny Kyburz |
| COURSE CODE, TITLE AND UNIT: | COMP8755 <Individual Computing Project> 12 units |

SEMESTER | ☑️ S1 YEAR: 2021 | ☑️ S2 YEAR: 2021 |

PROJECT TITLE: Improving Students’ Academic Performance through AI and Semantic Technologies

LEARNING OBJECTIVES:
- Understanding and ability to apply/implement semantic technologies, AI / Machine Learning models and Data Mining techniques;
- Understanding fundamental research to address a research question;
- Apply computing knowledge and implementation skills to the area of Computers in Education;
- Understanding of interdisciplinary research; and,
- Understand how to conduct a substantial research-based project Understand how to analyse data, synthesize and report research findings.

PROJECT DESCRIPTION:
- Write a literature survey
- Development of a model to improve students’ performance using AI/Semantic technologies
- Train and test the model on large datasets
- Tune parameter and inference
- Evaluation/validation of the proposed model
- Deploy the model
- Write thesis

Research School of Computer Science

Figure 5.1: Contract page 1
**ASSESSMENT** (as per course’s project rules web page, with the differences noted below):

| Assessed project components: | % of mark | Assessed project components: | % of mark |
|-----------------------------|-----------|-----------------------------|-----------|
| Thesis                      | (85%)     | Thesis (reviewer mark)      | 60%        |
| Presentation                | (10%)     | Artefact (supervisor project mark) | 30%        |
| Critical Feedback           | (5%)      | Presentation                | (10%)     |

**MEETING DATES (IF KNOWN):**
Weekly

**STUDENT DECLARATION: I agree to fulfil the above defined contract:**

| Signature | Date |
|-----------|------|
| Yixia Cheng | 17/02/2021 |

**SECTION B (Supervisor):**

I am willing to supervise and support this project. I have checked the student’s academic record and believe this student can complete the project.

| Signature | Date |
|-----------|------|
|           | 18/02/2021 |

**Reviewer:**

Name: Penny Kyburz | Signature:

**Reviewer 2:** (for Honours only)

Name: X | Signature:

**REQUIRED DEPARTMENT RESOURCES:**

**SECTION C (Honours / Projects coordinator approval)**

| Signature | Date |
|-----------|------|
|           | 18/02/2021 |

Research School of Computer Science  Form approved CDC 11-Jul-19

---

**Figure 5.2: Contract page 2**
Appendix 2
Appendix 3

Description of Software

**Dropout prediction**

preprocessing.py implements the pre-processing. subset_seperator(self) is separating raw dataset by degree; organise(self) is for data cleaning, data normalization and Data validation. impute_missing_value(self) is for data imputation.

feature_selector.py implements the feature selection by using GA+SVM. get_fitness (self,pop,path) is for getting fitness of each individual by SVM; select(self,pop, fitness), crossover(self,parent, pop), mutate(self,child), and evolution(self) are the steps of GA.

xx_dataloader.py, xx=ADM,ARQ, or CSI implements the training and test by LSTM.

Directly Run xx_dataloader.py, xx=ADM,ARQ, or CSI to get the result of preprocessing, feature selection and training and test.

**Similarity Measurement**

Installment of BERT as Service and runing The installment of BERT used in this project as the link above or following.

Install Bert as a Service by pip install -U bert-serving-client(instruction is available on https://github.com/hanxiao/bert-as-service)

Run the Bert as a Service pointing to the unzipped downloaded model using the following command: bert-serving-start -model_dir /your_directory/wwm_uncased_L-24_H-1024_A-16 -num_worker=4 -port XXXX -max_seq_len NONE

run python similarity.py to get the result of two sentences in two courses comparison. The result will be in /result/similarity_measurement/full_similarity
run python text_similarity.py to get the result of two courses comparison, which will be stored in /result/similarity_measurement/full_similarity/similarity_full.csv

run python diagram.py to visualize the result.

**Prerequisite Identification**

Install the external libraries by using conda or pip command. RefDSimple.py is for using RefD to retrieve and get potential candidates from DBpedia https://www.dbpedia.org/ which is a main knowledge graph in Semantic Web.

dependency_extractor.py is using TextRazor (https://www.textrazor.com/) to segment and extract entities from text.

cfg is the configuration file, you may need to change the proxy before using it.

run dependency_extractor.py to get the result which will be in /result/prerequisite_identification

**Systematic Review-Paper Crawling**

Run paper_crawler.py to acquire the paper from SPringer to local file named "dataset-byabstract", which includes title, abstract, and so on.
Appendix 4

README file
COMP8755 Project—Improving Students’ Academic Performance with AI and Semantic Technologies

This project contains two main parts, that is, *Dropouts_prediction* and *Curriculum_Analysis* in the separate corresponding folders. There is also a README file in each folder.

**Dropout_Prediction**

This sub-project is for predicting student dropout in 3 degree (ADM, ARQ, and CSI) at a Brazilian university. By using Genetic Algorithm (GA)+ Support Vector machine (SVM) for feature selection and Long short-term memory (LSTM) for Time Series (TS) data training and test.

**Dataset**

*historicsFunal.csv* is the raw data for this project, it contains 32 features including student info such as course, degree, grades, enrollment status and so on.

*ADM.csv, ARQ.csv, and CSI.csv* are three datasets are separated from raw data by degree

*xx_clean.csv, xx=ADM,ARQ, or CSI* is the clean dataset processed by *preprocessor.py*. Derive from the above three datasets.

*xx_final.csv, xx=ADM,ARQ, or CSI* is the final dataset selected by *?feature_selector.py* and ready for training. Derive from above three datasets.

**Imported Libraries**

The scripts of projects are all based in Python 3.8 and anaconda package manager. The external dependencies are as following:

- torch
- torch.nn
- torch.utils.data
- torch.autograd
- torch.utils.data
- matplotlib.pyplot

---

**Figure 5.3**: README page 1
• sklearn.svm
• sklearn.model_selection
• sklearn.preprocessing
• sklearn.impute
• sklearn.ensemble

*Note: Install all libraries by using `conda install` or `pip install`*

**Pre-processing**

`preprocessing.py` implements the pre-processing. `subset_seperator(self)` is separating raw dataset by degree; `organise(self)` is for data cleaning, data normalization and Data validation. `impute_missing_value(self)` is for data imputation.

**Feature Selection**

`feature_selector.py` implements the feature selection by using GA+SVM. `get_fitness(self, pop, path)` is for getting fitness of each individual by SVM; `select(self, pop, fitness)`, `crossover(self, parent, pop)`, `mutate(self, child)`, and `evaluation(self)` are the steps of GA.

**Training+Test**

`xx_dataloader.py`, `xx=ADM,ARQ, or CSI` implements the training and test by LSTM.

**Running project**

**Directly Run** `xx_dataloader.py`, `xx=ADM,ARQ, or CSI` to get the result of preprocessing, feature selection and training and test.

**Curriculum_Analysis**

This sub-project contains two parts, one is *similarity measurement*, which aims to get the similarity between two courses with using BERT; another one is *prerequisite identification*, which is about using SemFD to analysis semantic relation between two concepts and courses further.

**Dataset**

The dataset this project used is from ANU Course & Program [https://programsandcourses.anu.edu.au/](https://programsandcourses.anu.edu.au/), which is a

---

**Figure 5.4: README page 2**
website providing the course description and curriculum arrangement.

The raw_data stores each course in txt format; the raw_data_csv stores each course in csv format.

Imported Libraries

The scripts of projects are all based in Python 3.8 and anaconda package manager. The external dependencies are as following:

- bert_serving.client
- termcolor
- seaborn
- textrazor
- SPARQLWrapper

*Note: Install all libraries by using conda install or pip install*

Similarity Measurement

This part focuses on the similarity measurement relying on BERT to conduct sentence embedding. Note that this work is based on the @blindreviewrepo on Github. [https://github.com/blindreviewrepo/WebAndReligion#run-the-script-scriptsimilarity_fullpy-book_file_txt-book_file_2txt-output_matrix_filecsv](https://github.com/blindreviewrepo/WebAndReligion#run-the-script-scriptsimilarity_fullpy-book_file_txt-book_file_2txt-output_matrix_filecsv)

Installment of BERT as Service and running

The installment of BERT used in this project as the link above or following.

1. Install Bert as a Service by `pip install -U bert-serving-client` (instruction is available on [https://github.com/hanxiao/bert-as-service](https://github.com/hanxiao/bert-as-service))
2. Download large BERT model at [https://storage.googleapis.com/bert_models/2018_06_30/wwm_uncased_L-24_H-1024_A-16.zip](https://storage.googleapis.com/bert_models/2018_06_30/wwm_uncased_L-24_H-1024_A-16.zip)
3. Run the Bert as a Service pointing to the unzipped downloaded model using the following command: `bert-serving-start -model_dir /your_directory/wwm_uncased_L-24_H-1024_A-16 -num_worker 4 -port XXXX -max_seq_len NONE`

Sentence Embedding Generation and similarity acquirement

run `python similarity.py` to get the result of two sentences in two courses comparison. The result will be in `/result/similarity_measurement/full_similarity`
run `python text_similarity.py` to get the result of two courses comparison, which will be stored in
`/result/similarity_measurement/full_similarity/similarity_full.csv`

run `python diagram.py` to visualize the result.

**Prerequisite Identification**

This part focuses on the *prerequisite of concepts and courses* in further move.

1. Install the external libraries by using conda or pip command
2. `RefDSimple.py` is for using RefD to retrieve and get potential candidates from DBpedia
   [https://www.dbpedia.org](https://www.dbpedia.org) which is a main knowledge graph in Semantic Web.
3. `entity_extractor.py` is using TextRazor ([https://www.textrazor.com](https://www.textrazor.com)) to segment and extract entities from
   text.
4. `config.cfg` is the configuration file, you may need to change the proxy before using it.
5. `run entity_extractor.py` to get the result which will be in `/result/prerequisite_identification`

**Systematic Review-Paper Crawling**

This sub-project is regarding acquiring the paper from Springer [https://www.springer.com/gp](https://www.springer.com/gp) by using
beautifulsoup.

**Imported Libraries**

The scripts of projects are all based in Python 3.8 and anaconda package manager. The external dependencies are as following:

- bs4
- selenium
- webdriver_manager

*Note: Install all libraries by using `conda install` or `pip install`*

**Usage**

Run `paper_crawler.py` to acquire the paper from SPringer to local file named "datasetbyabstract", which
includes title, abstract, and so on.

---

**Figure 5.6: README page 4**