

Intelligent Analysis for Multi-Level Data-Driven Prediction



Zhenpeng Li


Supervisor: **Dr. Changjing Shang (Joint Primary)**
Prof. Qiang Shen (Joint Primary)

Aberystwyth University

This dissertation is submitted for the degree of
Doctor of Philosophy

June 2019



Mandatory Layout of Declaration/Statements

Word Count of thesis: DECLARATION	55916
This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.	
Candidate name	Zhenpeng Li
Signature:	
Date	27/06/2019

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated. Where ***correction services** have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signature:	
Date	

[*this refers to the extent to which the text has been corrected by others]

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signature:	
Date	

NB: *Candidates on whose behalf a bar on access (hard copy) has been approved by the University should use the following version of Statement 2:*

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loans after expiry of a bar on access approved by Aberystwyth University.

Signature:	
Date	

Acknowledgements

I would like to express my sincere gratitude to my supervisors: Dr. Changjing Shang and Prof. Qiang Shen, for their guidance and motivation.

I am extremely grateful to my family: My dear wife Linxin Li, my parents Xiangqian Li and Xiaomin Liu, and my parents-in-law Guangxian Li and Xiaohong Li. The completion of this Ph.D. work would not have been possible without their warm support and encouragement.

I would like to thank all my fellow researchers in the Advanced Reasoning Group, for the stimulating discussions and helpful advice.

I would like to thank to Suresh Kumar and Asif Khan, for the warm inter-discipline discussion and cross-cultural communication.

I would also like to express my deepest appreciation for the Department of Computer Science and Springer Publishing, for their generous financial support.

Many thanks to all of the academic, administrative, technical staffs with the Department of Computer Science, Aberystwyth University, for their kind assistance throughout my study.

My gratitudes also go to the anonymous reviewers, journal editors, conference organisers with my submitted works, for their valuable input in refining my ideas.

Abstract

Prediction is one of the typical applications in the research fields of machine learning and data mining. Traditional predictive analytics focuses on estimating class membership or numeric value within a specific domain or region, which results in the development of classification models and regression models. However, the restriction on isolated information and the progress of computation technology jointly require the collection and connection of fragmented data for further investigation. Predictive analytics is nowadays expected to be extended its research area to seeking for relationships amongst separated data, which is widely known as link prediction or link analysis.

In this thesis, a novel predicting system performing the tasks of instance based missing information estimation, feature variable identification, and variable group pattern recognition has been presented. Specifically, an advanced regression model embedded with both hard and soft clustering techniques is novelly proposed to forecast missing feature values for objects of interest. Also, a link based model creatively employing the concept of connected-triple, and implemented with fuzzy logic, is invented to measure correlations between domain feature variables from various information sources. The resulting link based model has been further utilised as a foundation to construct an adaptative hierarchical knowledge base for describing feature variables under consideration, facilitating both dynamic updating and immediate query.

The adaptability and flexibility of the proposed work, together with its remarkable initial performance in sample applications are illustrated by experimental evaluation via datasets collected from both real-world domains and artificial production. The outcomes of comparative studies demonstrate the efficacy of the present work and its great potential for future use. Further suggestions on development and refinement of this research are provided to stimulate inspiration on improving the current work.

Table of Contents

List of Figures	vii
List of Tables	ix
List of Algorithms	x
1 Introduction	1
1.1 Regression Analysis for Prediction	3
1.2 Clustering Analysis for Predication	4
1.3 Link Analysis for Prediction	5
1.4 Challenges and Contributions	6
1.5 Structure of Thesis	7
2 Background	11
2.1 Clustering Metrics	11
2.2 Link Prediction Metrics	16
2.2.1 Vertex-based Metrics	17
2.2.2 Topology-based Metrics	18
2.2.3 Probabilistic Metrics	30
2.3 Fuzzy Link Prediction	34
2.3.1 Fuzzy Inference System	34
2.3.2 Fuzzy Inference for Link Prediction	36
2.4 Summary	41
3 Clustering Embedded Linear Regression for Prediction	42
3.1 Conceptual Framework of Predicting System	43
3.2 Implementation of Predicting System	44
3.2.1 Partition Subsystem	44
3.2.2 Classification Subsystem	45

3.2.3	Regression Subsystem	47
3.2.4	Estimation Subsystem	48
3.3	Case Study: Prediction on Student Academic Performance	48
3.3.1	Data Preparation	50
3.3.2	Experimental Setup	53
3.3.3	Experimental Result	56
3.4	Summary	65
4	Fuzzy Embedded Clustering Linear Regression for Prediction of Student Academic Performance	66
4.1	Structure of Predicting System	67
4.2	Implementation of Predicting System	69
4.2.1	Partition Subsystem	69
4.2.2	Regression Subsystem	70
4.2.3	Offset Value Generating Subsystem	72
4.2.4	Estimation Subsystem	74
4.3	Experimental Evaluation	74
4.3.1	Experimental Setup	75
4.3.2	Results and Discussions	76
4.4	Summary	83
5	Fuzzy Connected-Triple for Prediction of Inter-Variable Correlation	84
5.1	Predicting System	86
5.1.1	Conceptual Framework	86
5.1.2	Connected-Triple Extraction	87
5.1.3	Link Analysis	88
5.1.4	Fuzzy Inference Model	96
5.1.5	Illustrative Link Strength Prediction	98
5.2	Empirical Evaluation	100
5.2.1	Datasets	100
5.2.2	Methods for Comparison	102
5.2.3	Experimental Setup	102
5.2.4	Experimental Results	103
5.2.5	Complexity Analysis	109
5.3	Summary	110

6	Intelligent System for Variable Cluster Pattern Recognition	111
6.1	Conceptual Framework of Intelligent System	112
6.2	Implementation of Intelligent System	114
6.2.1	Variable Clustering Subsystem	114
6.2.2	Cluster Recognition Subsystem	119
6.3	Experimental Evaluation	125
6.3.1	Data Preparation	125
6.3.2	Experimental Setup and Results	126
6.4	Summary	136
7	Feature Variable Identification and Hierarchical Knowledge Reorganisation	137
7.1	Feature Variable Identification	138
7.2	Hierarchical Knowledge Reorganisation	140
7.3	Experimental Evaluation	149
7.3.1	Experimentation for Singleton Variable Identification	149
7.3.2	Experimentation on Hierarchical Structure Reorganisation	151
7.4	Intelligent System for Variable Detection and Hierarchical Knowledge Reorganisation	155
7.4.1	Generic Framework of Intelligent System	155
7.4.2	Empirical Study of Intelligent System	157
7.5	Summary	163
8	Conclusion	164
8.1	Summary of Thesis	164
8.2	Future Work	166
8.2.1	On Cluster Embedded Regression Analysis	167
8.2.2	On Student Academic Performance	167
8.2.3	On Fuzzy Connected-Triple for Inference	168
8.2.4	On Variable Cluster Pattern Recognition	168
8.2.5	On Hierarchical Knowledge Representation	169
Appendix A	Publications Arising from Thesis	171
A.1	Journal Articles	171
A.2	Conference Papers	171
Appendix B	Datasets Employed in Thesis	172
Appendix C	List of Acronyms	174

List of Figures

1.1	Process of knowledge discovery	2
1.2	Selection of Real-World Applications of Regression Predicting Model . . .	4
1.3	Structure of Thesis	8
2.1	Example of Hierarchical Structures for a Link Network	31
2.2	Example of Stochastic Block Model	33
2.3	General Architecture of FIS	35
3.1	Architecture of CELR Predicting System	43
3.2	Distribution of Assignment Scores	51
3.3	Distribution of Test Scores	51
3.4	Distribution of Final Exam Scores	52
3.5	Process of CELR for Predicting Student Academic Performance	53
4.1	Architecture of Final Period Grade Predicting System	68
5.1	Predicting Framework	86
5.2	Sample Datasets	87
5.3	Connected-Triples Extracted from Sample Datasets	88
5.4	Transitivity of Connected Triple	89
5.5	Example of Elbow Method	95
5.6	Fuzzy Membership Values of Link Weight with Respect to Different Mea- sures	97
5.7	Two Simple Datasets Used for Illustration	98
5.8	Prediction Accuracy for Real-world Categorical Data	105
5.9	Prediction Accuracy for Real-world Continuous Numeric Data by APCC . .	105
5.10	Prediction Accuracy for Real-world Continuous Numeric Data by MIC . .	106
5.11	Prediction Accuracy for Real-world Numeric-Categorical Mixed-Typed Data	106
5.12	Prediction Accuracy for Synthetic Data	108

6.1	Conceptual Structure of Feature Variable Cluster Recognition Model	114
6.2	A Sampled Feature Vector for Numeric Variable	120
6.3	Example of Feature Vector for a Singleton Variable	120
6.4	Example of Swapping Row Vectors in Target Table	123
7.1	Updating Process for Scenario-1	143
7.2	Updating Process for Scenario-2	146
7.3	Updating Process for Scenario-3	148
7.4	Conceptual Framework of Cluster Based Variable Detection and Hierarchy Reorganisation	156

List of Tables

2.1	Comparison of Neighbour-Based Metrics for Link Prediction	21
3.1	Selected Attributes for Student Academic Performance	52
3.2	Assignment Scores with Their Associated Level of Achievement and Grades (class)	54
3.3	Class Test Scores with Their Associated Level of Achievement and Grades (class)	54
3.4	Exam Scores with Their Associated Level of Achievement and Grades (class)	54
3.5	ECTS Grades with Corresponding Portugal/France Grades	55
3.6	Comparison of Statistical Indicators in Predicated Exam Scores Obtained by CELR and SMLR on SAP50A	57
3.7	Comparison of Statistical Indicators in Predicated Exam Scores Obtained by CELR and SMLR on SAP50B	59
3.8	Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Maths(GP)	61
3.9	Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Maths(MS)	62
3.10	Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Portuguese(GP)	63
3.11	Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Portuguese(MS)	64
4.1	Preprocessed Student Academic Performance Related Attributes	75
4.2	Comparison of Approaches in Terms of Mean Absolute Error for Prediction	77
4.3	Comparison of Approaches in Terms of Predicting Accuracy (%)	78
4.4	Comparison of Approaches in Terms of Predicting Result within 1-grade Error (%)	79
4.5	Comparison of Classification Accuracy (%)	82

5.1	Summary of Datasets: Link Prediction	101
5.2	Analysis of Time Complexity	109
6.1	Summary of Datasets: Variable Cluster Recognition	125
6.2	Comparison of ASC for Different Clustering Models on Data Corpus	128
6.3	Comparison of ASC for Different Clustering Models on Testing Dataset . .	129
6.4	Comparison on Accuracy of Successful Matching by Different Algorithms with Numeric Data Collections	133
6.5	Comparison on Accuracy of Successful Matching by Different Algorithms with Categorical Data Collections	134
6.6	Comparison on Accuracy of Successful Matching by Different Algorithms with Mixed-Type Data Collections	135
7.1	Summary of Datasets: Singleton Variable Identification	150
7.2	Comparison on Accuracy of Singleton Variable Identification	151
7.3	Comparison on NMI of Different Updating Process	154
7.4	Summary of Datasets: General Predicting Model	158
7.5	Comparison on Accuracy of Singleton Variable Identification	161
7.6	Comparison on NMI for Hierarchical Structure Update	162
B.1	Summary of Data Used in Thesis	173

List of Algorithms

1	Squeezer	15
2	PropFlow Algorithm for Prediction	29
3	FCC for Link Prediction	38
4	<i>k</i> -means Clustering Algorithm	45
5	<i>KNN</i> Classification Algorithm	46
6	Fuzzy C-means Clustering of Students	70
7	Efficient Agglomerative Clustering Algorithm for Feature Variables	116
8	Efficient Divisive Clustering Algorithm for Feature Variables	118
9	Evolutionary Algorithm to Shuffle Variable Term Distribution Vectors	124
10	Algorithm to Identify Uninformed Variable	139

Chapter 1

Introduction

It is certain that we are living in a data explosion era, demonstrated by the reality that tremendous volumes of information data have been being consistently generated at unprecedented and ever increasing pace. Massive data are collected and investigated in various domains, including commercial activities, engineering science, biomolecular research, social networks, and security monitoring [1]. Nowadays, data are being generated at every moment, and are collected and gathered at separate times, in diverse places, by different individuals or organisations. Importantly, data can, and in reality does exist in various sources, including mobile devices, public and private clouds, subscription-based services such as file sync and share, and virtual machines, to name just a few.

Over the past two decades, as the amount of data grows, it is undoubted that they play an increasingly significant role in different fields of daily life. It assists with decision making in a variety of application fields tremendously [2, 3]. It helps reveal essential laws and patterns of the objects existing in the world [4, 5]. It also provides evidence and experience for designing and creating novel solutions to various real world problems [6–9]. However, with the rapid growth of the available data, the success of managing and analysing the information embedded in the data becomes ever more practically important, whilst becoming ever more difficult in the meantime. Obviously, traditional manual knowledge discovery process is no longer an optimal option since it is increasingly expensive in computing and time-consuming [10]. Additionally, the conventional manner for data analysis relies significantly on the opinions of domain experts, who should have a precise and considerable understanding of the problem at hand. However, such opinions are often subjective [11] or inconsistent between distinct individuals [12, 13], which may lead to opposition or controversy. More importantly, data in its presenting format may involves a large quantities of instances and descriptive features, which are impractical for human beings to perform manual analysis in

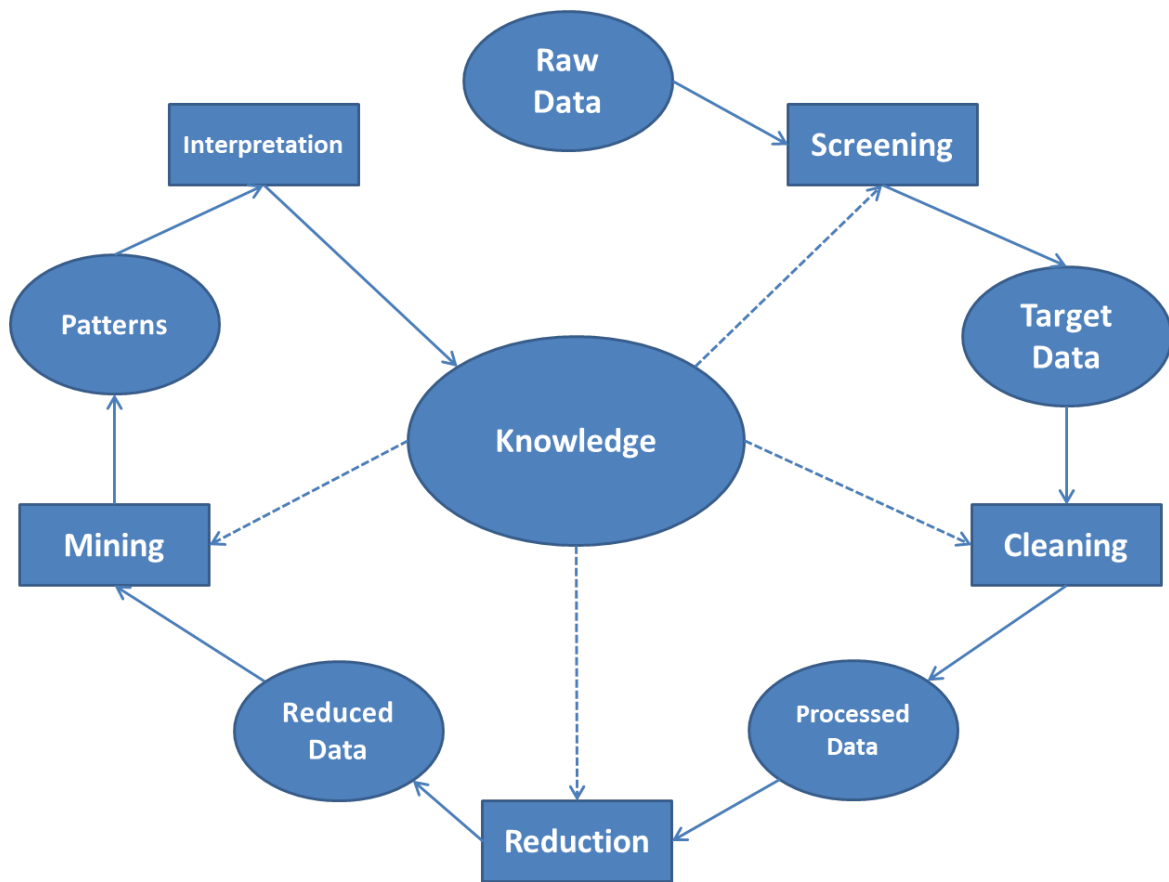


Fig. 1.1 Process of knowledge discovery

most circumstances.

Hence, automatic extraction of knowledge from data based on machine learning, widely regarded as knowledge discovery, has attracted great attention in recent years. It takes the advantages of overcoming the above mentioned drawbacks while providing a new insight for exploring and processing the data. Ideally, the resulting knowledge collected from such advanced technique needs to be both human-understandable and machine-interpretable and must represent knowledge in a manner that facilitates inferencing [14]. A general process for knowledge discovery is shown in Fig. 1.1 [15]. Simultaneously, a new term, emerging with the scenario of data deluge, named “data science”, has been proposed. Fundamentally, data science is a cross-disciplinary field that uses scientific approaches, procedures, algorithms and systems to obtain knowledge and insights from data in a wide variety of formats, either structured or unstructured [16].

The emphasis on predictive analytics is particularly strong in machine learning and knowledge discovery in databases. Predictive analytics encapsulate a large volumes of statistical techniques applied on the historical and current facts to forecast future outcomes or unknown events [17, 18]. Traditional predictive analytics focuses on estimating class membership or numeric value within a specific domain or region, which results in two different types of models: regression model and classification model. A general description of regression model will be concluded in Section 1.1. Classification model is big topic beyond the scope of this thesis. However, provided that no explicit information about class labels available, which is often the case in the real world, an automatic approach to grouping data under consideration into different categories is of great importance for investigation. Such unsupervised analytic technique is termed as clustering. A brief introduction of clustering analysis will be provided in Section 1.2. Nowadays, as pieces of data are scattering everywhere in the world, and they are frequently organised or structured in a random manner, collecting and connecting those fragmented data to search their potential relation for further research is worth investigating. And predictive analytics also extends its antenna to the area of finding relationships amongst the separated data, which is known as link prediction or link analysis. This will be discussed in Section 1.3.

1.1 Regression Analysis for Prediction

Regression analysis is a popular statistical process for estimating the relationships among variables, with the coefficients in the regression equation define the correlation between each of the independent variables and the outcome dependent variable. Generally, the regression models are divided into linear structure and non-linear counterpart, according to their respective combination format of model parameters. Regression analysis for prediction has been widely applied in various fields [19–22]. A few example areas are illustrated in Fig. 1.2. The best-known types of regression model for prediction are the following: Linear Regression for description of linear relationship, Logistic Regression for estimation of the probability of belonging to groups or categories, Cox Regression for modelling of survival data, and Poisson Regression for depiction of counting processes [23].

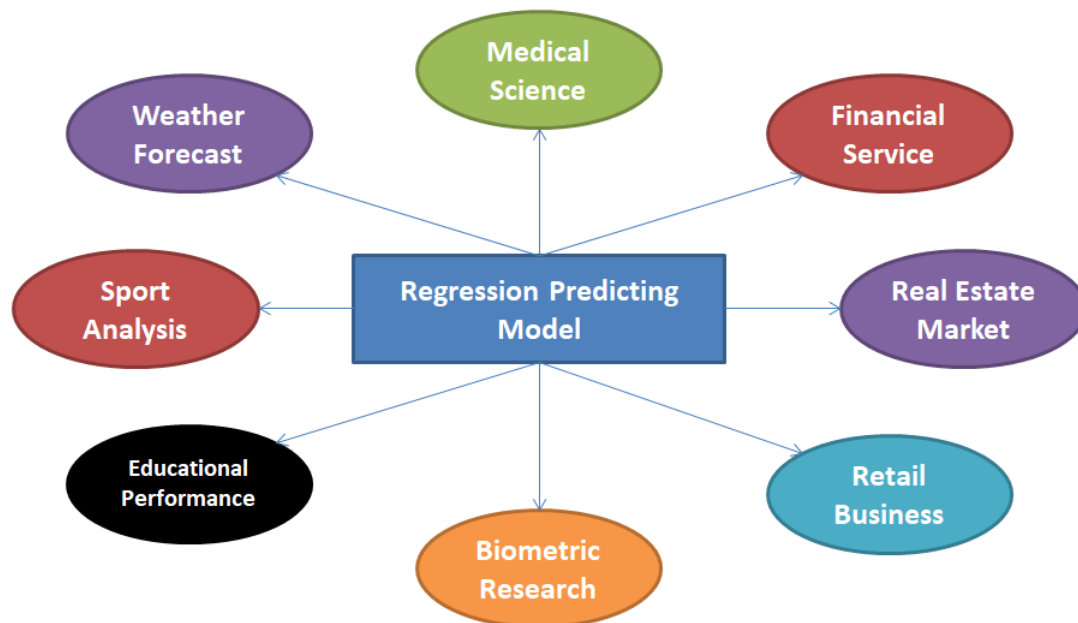


Fig. 1.2 Selection of Real-World Applications of Regression Predicting Model

Advanced Regression methods for prediction are assisted with feature selection step [24] [25] and segmentation of regression space [26, 27]. Outlier analysis [28, 29] and missing data treatment by imputation [30, 31] are also applied to enhance the performance of the regression model.

1.2 Clustering Analysis for Predication

Cluster analysis is commonly applied when exploring and predicting the structure of the data. It has always been employed in the first stage of understanding the raw information, especially for challenging problems where prior knowledge is insufficient. The need to acquire knowledge from excessive amount of data has been a significant driven force for clustering analysis. Conceptually, the spirit of cluster analysis is its involving process which divides data objects into groups in a way that the objects in the same group are more similar to each other than to those categorised in different groups [32]. Objects or entities under cluster analysis are usually described in terms of attribute (feature) values and relative proximity. Opposite to the supervised learning in which a category label is needed and often manually tagged by domain experts, clustering is unsupervised since it does not require such label information. Clustering has been applied to a range of real-world predicting

problems, including information retrieval [33, 34], pattern recognition [35, 36], imaging processing [37, 38], genetic analysis [39, 40], recommender systems [41, 42], etc. Given its great potential in application, a vast number of research studies focusing on various aspects of cluster analysis are carried out, such as determination of optimal cluster numbers [43], automatic detection of initial cluster centroid [44], similarity metrics for clustering procedure [45, 46], Aggregation of cluster ensembles [47, 48].

1.3 Link Analysis for Prediction

A wide variety of domains in the real world are relational in nature and richly structured, accommodating a set of objects related to each other in complex ways [49]. Such data propose new challenges of predicting potential relationships amongst these objects and determining the types or grades of the aforementioned relationships. Typically, a graph model or a network structure is capable of reflecting common patterns of interactions between the objects in the domain. Taking these patterns into consideration could help to provide a better prediction.

Social Network Analysis (SNA), is originally set up to present the social structures including objects (actors) and relationships amongst them [50]. These networks can be conveniently represented by employing vertices and links. The links show types of relationship amongst the vertices including kinship, friendship, collaborations, and any other interactions between the people, namely the vertices in such a network [51]. In particular, it is widely applied in recommendation systems for information retrieval, helping search for new friends [52] and potential business collaborators [53–55], finding domain experts or co-authors in academic fields [56]. Obviously, the concept of SNA models can be generalised. They are not only restricted to the use in networks concerning human beings, but also can be utilised to depict and analyse the structures in a wide variety of problem domains.

Link, a term used in SNA or network analysis (NA) to describe the connection between two discussed objects in domain, could contain valuable information with regard to the research interest. Link prediction is aimed to predict future possible links in the network. It can also be used to predict missing links due to incomplete data. In SNA, link prediction is one of the most salient tasks, including the discovery of missing or developing links in a certain network [57]. Recently, link prediction has become an important and effective technique in the study of biology, economy, and other cross-disciplines [58]. For instance, in bioinformatics research, link prediction is adjusted to present gene expression networks [59],

describing protein-protein interactions [60]. In politics research, link prediction is adopted to conceptualise a policy-making process as a network of political actors [61]. In public health care, link prediction is employed to assess factors contributing to the service, the care process and the patient outcome [62]. In project management, link prediction is utilised to measure the correlations amongst stakeholders, process-related values and outcome-related values [63]. In E-commerce, link prediction is equipped for providing interesting items in online shopping [53]. Last but not the least, in the field of national defence and public security, link prediction is assembled for terrorism and insurgency detection [64–67], money laundering prevention [68] and abnormal telecommunication surveillance [69].

1.4 Challenges and Contributions

In general, prediction is one of the most important and challenging applications in the research fields of machine learning and data mining. Traditional predictive analytics focuses on estimating class membership or numeric value within a specific domain or region, leading to the development of classification models and regression models. Although it has been widely demonstrated that no existing predictive model works perfectly well for every real world problem [70, 71], improving predicting accuracy has always been expected and pursued in the research field. In this thesis, a novel predicting system aimed at enhancing the performance of instance based missing information estimation has been presented. Particularly, an advanced regression model embedded with both hard and soft clustering techniques is novelly proposed to forecast missing feature values for objects of interest.

However, more significantly, most of the current studies work on single dataset, unable to discover hidden information from different data sources sufficiently. The restriction of research on isolated information, together with the progress of computation technology, requires the collection and connection of fragmented data for further investigation. Predictive analytics is nowadays expected to extend its antenna to seeking for relationships amongst separated data, which is envisaged to be implemented through the approaches of link analysis. Yet, majority of the existing link based studies concentrate on predicting tasks of individual items and records, which basically manipulate data at entity or instance level [72, 73], leaving predictive analytics at higher platform (variable level) blank. Unlike previous research that focused on identifying links between objects or entities in a specific region, in this thesis, a novel link based model, creatively employing the concept of connected-triple, and implemented with fuzzy logic, is invented to measure correlations between domain feature

variables from various information sources. The proposed model is primarily data-driven, offers a potentially effective mechanism for dealing with the problem of link prediction, particularly when any given information contents are obtained from different data sources where parts of the information overlap. Such link prediction problems are obviously of general interest in many data mining applications.

In addition, although it becomes more convenient to obtain data from various sources, thanks to the development of information technology, part of the acquired data may lack description to themselves for a variety of reasons, such as security consideration, equipment malfunction, ignorance from information collector, and inappropriate storage. Such deficiencies impede understanding and making use of information seriously, leading to considerable restrictions in data mining and knowledge extraction. The resulting link based model can be utilised as a foundation to construct an adaptative hierarchical knowledge base for describing feature variables under consideration, facilitating both dynamic updating and immediate query. The adaptability and flexibility of the proposed work, together with its remarkable initial performance in sample applications are illustrated by experimental evaluation via datasets collected from both real-world domains and artificial production. The outcomes of comparative studies demonstrate the general efficacy of the present work and its great potential for future use.

1.5 Structure of Thesis

This section outlines the structure of the remainder of this thesis. A flowchart presenting the thesis structure is shown in Fig. 1.3. Generally, based on background knowledge provided in Chapter 2, the thesis focuses on prediction of data at different levels: instance level, variable level, and variable cluster level. In particular, Chapter 3 and Chapter 4 aim at prediction on instance level, which can be summarised into a group, as presented in a dashed box. Chapter 5, chapter 6 and Chapter 7 are dedicated to prediction at variable level and variable cluster level, which can be generalised into another group. A list of publications arising from the work of this thesis is provided in Appendix A.

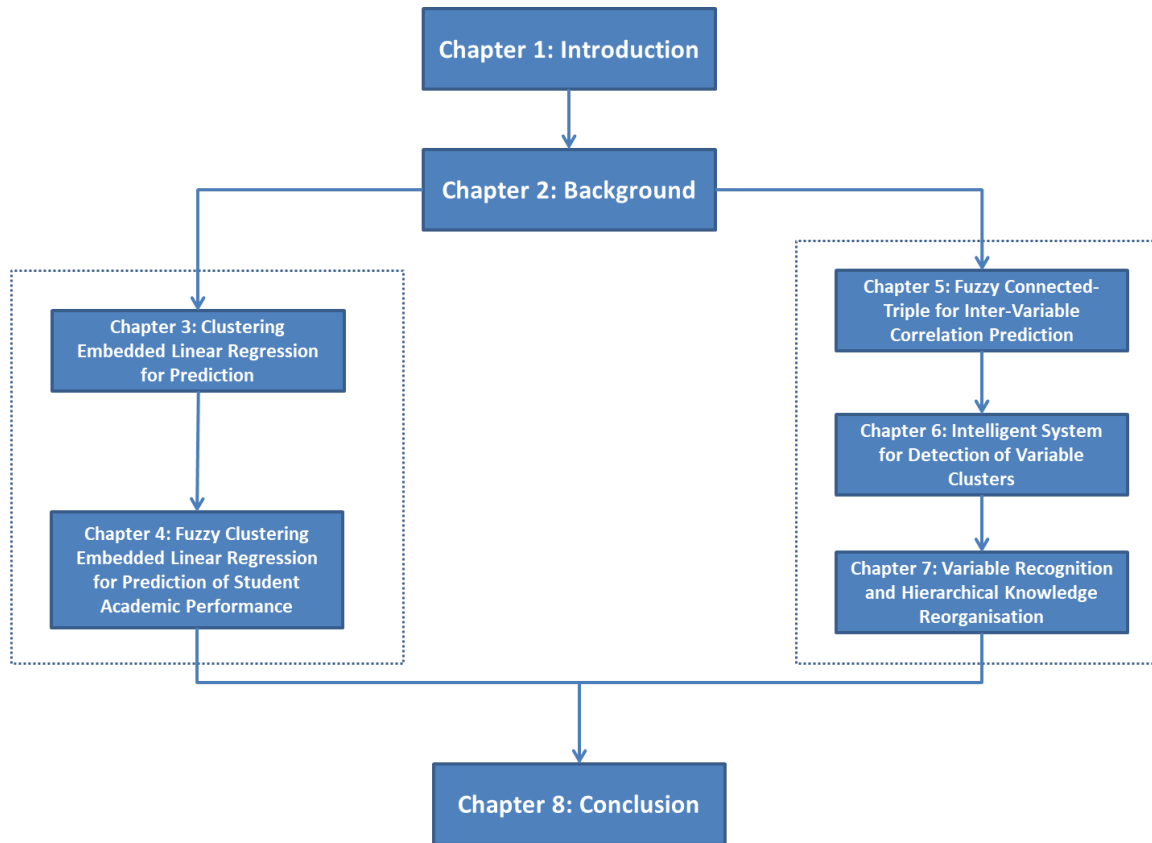


Fig. 1.3 Structure of Thesis

Chapter 2: Background. This chapter provides the knowledge premise for the proposed work discussed in this thesis. Primarily, an overview of popular clustering techniques which enable identifying underlying patterns of data is conducted, with a description of their fundamental concepts, implementation procedures and time complexity analysis. This part of knowledge sets up as a basis for this PhD project. A systematic study of existing link prediction techniques is followed, including vertex-based metrics, topological-based metrics, probabilistic-based metrics, with their general framework and technical essential depicted. This emerging research field attracting great attention, provides inspiration and creativity for the proposed work. Fuzzy logic allows reasoning with imprecision and uncertainty, which has a natural appeal for the task of prediction. Additionally, the basic model of fuzzy inference process with the state-of-the-art fuzzy inference metrics for link prediction are outlined and discussed.

Chapter 3: Clustering Embedded Linear Regression for Prediction. This chapter presents a novel intelligent system aiming at predicting a missing feature value (in the numeric form) for an object being followed with interest. It gives a retrospective review on the deficiencies

of the existing popular predicting approaches of the similar kind at first, and defines the application scenario for the proposed model. Then, it concludes the conceptual framework of the predicting system, with the implementation for each of its components clearly described in detail. The time complexity for each of the subsystems is examined to show the efficiency of the proposed model. A case study of its application on student academic performance reveals its efficacy and illustrates that the proposed work has great potential for future use. The contents of this chapter has been published in [74].

Chapter 4: Fuzzy Clustering Embedded Linear Regression for Student Academic Prediction. This chapter is an extension of Chapter 3, focusing on the study of predicting student academic performance. It refines the predicting model described in Chapter 3 by systematically considering the influence factors which have significant effect on student academic performance. Fuzzy clustering, a soft computing technique which enables accommodating both local and global information, provides an informative basis for reasoning. A new schema encapsulating previous academic records and student normal study behaviour is assisted to perform the estimating process. A paper proposing this refined technique has been published in [75].

Chapter 5: Fuzzy Connected-Triple for Prediction of Inter-Variable Correlation. Identification of hidden relationships between domain attributes from different data sources is of great practical significance and forms an emerging field in data mining. However, currently there seldom exist any systematic methods that can effectively handle this problem, especially when dealing with imprecisely described associations. In this chapter, a novel data-driven approach for inter-variable correlation prediction is proposed by exploiting the concept of connected-triples. The work is implemented with the use of fuzzy logic. Through the exploitation of link strength measurements and fuzzy inference, the job of detecting similar or related variables can be accomplished via examining link relation patterns within and across different data sources. Empirical evaluation results are discussed, revealing the potential of the proposed work in predicting interesting attribute relations, while involving simple computation mechanisms. The initial concept of this part has been published initially in [76], and winning the best student paper award in the 17th UK Workshop on Computational Intelligence, with a further and more in-depth version in [77].

Chapter 6: Intelligent System for Detection of Variable Clusters. This chapter is composed on the basis of Chapter 5 by further solving the real-world problem of assigning description to dataset with uninformed feature variables. Such investigation is carried out by

examining the underlying group patterns of such uninformed variables (within the dataset) against its identified counterpart in existing informed sources. A reliable structure for both of the informed and uninformed variables is required before commencing the examining process. In particular, three types of data corpus: continuous numeric, categorical, and mixed type of the two are investigated respectively. This chapter also works as a premise for Chapter 7.

Chapter 7: Variable Recognition and Hierarchical Knowledge Reorganisation. This chapter focuses on two main tasks: variable identification and variable structure update, according to different criteria. The job of variable identification is conducted when an uninformed variable cluster is detected similar to one of its counterparts in an informed source, whereas the task of variable structure update is performed when there exists no similar counterpart to the examined uninformed cluster. Such update manipulation can be regarded as a reorganisation procedure to the existing hierarchical knowledge structure. An iterative searching process and a flexible updating scheme are given in this chapter for each of the tasks.

Chapter 8: Conclusion. This chapter concludes the major contributions provided by the thesis, accompanied with a discussion on directions which set up the basis for future research.

Appendices Appendix A provides the publications arising from the work presented in the thesis, including both published papers in academic journals and conferences. Appendix B summaries information about the benchmark datasets employed in the thesis. Appendix C lists the acronyms used throughout the thesis.

Chapter 2

Background

This chapter first presents the fundamental concept of data clustering, including a number of benchmark algorithms that have been employed for various real-world problem. The second part of this chapter includes a large-scale review of link prediction metrics existing in the literature. A selection of modern link prediction techniques embedded with fuzzy logic for qualitative reasoning is presented in the third part. These components jointly provide the backgrounds and premises towards the proposed approaches that are designed to undertake the challenging tasks discussed in this thesis, with the last section providing a summary to this chapter.

2.1 Clustering Metrics

Data clustering is one of the basic and effective tools for understanding the structure of information or data. It plays a foremost and crucial role in data mining, pattern recognition, information retrieval and machine learning. Clustering aims at partitioning data into groups such that the data in the same group are more similar to each other than to those in different groups. Clustering is labelled an unsupervised learning technique as the measurement of similarity or proximity is conducted without knowledge of category assignments [78]. Due to its characteristic of knowledge free, research on algorithm selection, similarity measure, parameter initialisation, criterion setting to assist the performance of the clustering technique are widely conducted. However, in general, there exists no universal clustering algorithm that performs satisfactorily on data from different sources in various application fields. There are quantities of clustering algorithms developed in the literature. In this section, a selection of well-known example techniques are presented.

- ***k*-means:** It is one of the best known clustering algorithm that partitions data entities into groups. It originates from the concept of representing each of the k clusters by 'centroid' (mean of the members in the cluster). k -means is an iterative approach which exploits the square error (i.e., the total distance or dissimilarity between each data entity and the cluster centre) as a criterion function [79]. Basically, it starts with initialising centroids randomly, followed by assign data entities to clusters in order to minimise such square error. The criterion function works well with both compact and separated clusters. Mathematically, given a dataset with x representing its data instances involved, the square error e^2 for a clustering $\pi = C_1, \dots, C_k$ with k clusters is determined as:

$$e^2(\pi) = \sum_{p=1}^k \sum_{\forall x \in C_p} \|x - c_p\|^2 \quad (2.1)$$

where $\|\cdot\|$ denotes the Euclidean distance norm and c_p depicts the centroid of the p th cluster.

A general concept of the k -means metric is as follows:

- Select k data entities randomly as initial cluster centroids.
- Repeat:
 - * Assign each data entity to the cluster with the closest centroid (measured by aforementioned distance measure or other alternatives).
 - * Update the centroid of each cluster by the mean (average) of all the current data entities involved in that cluster.

Until the termination criterion is met.

Note that the common termination criteria includes: (1) no changes are made to the centroids; (2) no improvement for the criteria function; (3) the maximum number of iterations (refinements) is reached. k -means algorithm is applied in a variety of studies due to its efficiency, with the time complexity of $O(nkl)$, with n , k , l representing the number of data entities under discussion, number of clusters needed, and number of iterations, respectively. The drawback of k -means metric is its sensitivity to the choice of initial cluster centroids. In real-world application, the algorithm needs to run multiple times with different initial setups to obtain the ultimate solution which best satisfies the selection criteria.

- **Partitioning Around Medoids (PAM):** PAM is a variant of k -means, which provides more robustness to handle noisy data or outliers [80]. Specifically, for PAM, the medoid (centre) of a cluster is represented by one of the data instances within it, which can be less affected by extreme values than a mean value obtained by k -means. Primarily, it randomly selects k data entities as initial medoids to the clusters. Each of the remaining data entities is assigned to the cluster whose medoids is most similar to it. Then, in each of the iterations, the updated medoids for each cluster is determined by searching for the data entity with the minimum total distance to all others in the cluster, and all the data entities are reassigned to clusters based on their distance to the new set of medoids. The termination criteria for PAM is similar to its execution on k -means, However, it is worth mentioning that the computational cost of PAM is $O(n^2kl)$, which is more expensive compared with k -means, where n, k, l represents the same indicator as above stated.
- **Hierarchical Agglomerative Clustering (HAC):** HAC begins by considering each data entity as a singleton cluster, and then iteratively merges analogous clusters until forming a vast group. Such iterative process is capable of creating a hierarchical tree or dendrogram, which can be cut at any level to obtain the desired data partitions. In practice, the execution of HAC is guided by different definitions of distance between clusters:

- single-linkage (SL): It sets the dissimilarity degree between two clusters to be the minimum distance between all pairs of data entities, where each data entity in the pair is taken from distinct clusters. Formally, let C_p and C_q be two different clusters, the SL distance $SLD(C_p, C_q)$ between them can be calculated as:

$$SLD(C_p, C_q) = \min_{\forall x \in C_p, y \in C_q} d(x, y) \quad (2.2)$$

where $d(x, y)$ is the distance between data entities $x, y \in X$. Conventionally, $d(x, y)$ are measured by Manhattan distance, Euclidean distance, etc.

- Complete-Linkage (CL): It defines the dissimilarity degree between two clusters by the greatest distance between data entities in the clusters. Similar to SLD , its mathematical definition can be formulated as follows:

$$CLD(C_p, C_q) = \max_{\forall x \in C_p, y \in C_q} d(x, y) \quad (2.3)$$

- Average-Linkage (AL): It explores the average value of all pair-wise distance amongst data entities in two clusters as the cluster dissimilarity measure. Specifically, the average linkage based distance $ALD(C_p, C_q)$ between two clusters C_p and C_q can be computed as follows:

$$ALD(C_p, C_q) = \frac{1}{n_p n_q} \sum_{\forall x \in C_p} \sum_{\forall y \in C_q} d(x, y) \quad (2.4)$$

where n_p and n_q each represents the count of data entities in C_p and C_q .

HAC provides an intuitive visualisation for the hierarchical structure of the data entities, which can assist to perform data analysis in a systematic manner. Nevertheless, HAC has a disadvantage in its computational complexity of cubic order, which may limit its application on large datasets.

- **ROCK ROCK** (RObust Clustering using linKs) is a hierarchical clustering technique designed for categorical data [81]. This method is based on Jaccard Coefficient [82] to measure the similarity between a pair of data entities. Mathematically, for a pair of data entities $x, y \in X$, their similarity degree $sim(x, y)$ can be measured as:

$$sim(x, y) = \frac{|A_x \cap A_y|}{|A_x \cup A_y|} \quad (2.5)$$

where A_x and A_y are sets of all attribute values for x and y , respectively. Therefore, the similarity between x and y is measured by the proportion of their shared attribute values. Particularly, x and y are regarded as ‘neighbours’ if their similarity degree reaches or exceeds a specified threshold θ , i.e., $sim(x, y) \geq \theta$. Following that, the number of ‘links’ between any pair of data entities is determined by the number of their common ‘neighbours’. Based on the definitions of ‘neighbour’ and ‘link’, an agglomerative clustering method is applied to create the dendrogram. Initially, each data entity is considered as a singleton cluster, then the clusters are gradually combined together according to their pairwise closeness degree. Fundamentally, the closeness degree between pairs of clusters is described as the sum of the count of the links between all pairs of data entities within the examined clusters. Similar to HAC, ROCK algorithm also exhibits the time complexity of $O(n^3)$, where n denotes the number of entities the dataset.

- **Squeezer**: Squeezer is a single-pass metric which considers a single data entity at a time [83]. In Squeezer, each data entity is either assigned to one of the existing clusters

if its similarity to that cluster is above a pre-set threshold θ' , or allocated to a new established cluster. The basic process of Squeezer is as shown in Algorithm 1:

Algorithm 1: Squeezer

Input:

S_C : A set of empty clusters: set as an empty set

```

1 repeat
2   | Select a data entity  $x \in X$  randomly ;
3   | if  $S_C$  is empty then
4     |   CREATE a cluster  $C \in S_C$  and set  $x$  as its member ;
5     | end
6     | else
7       | Find the cluster  $C_i \in S_C$  that is closest to  $x$  ;
8       | if  $\text{sim}(C_i, x) \geq \theta'$  then
9         |   SET  $x$  as a member of cluster  $C_i$  ;
10      | end
11      | else
12        |   CREATE a new cluster  $C' \in S_C$  and SET  $x$  as its member ;
13        | end
14      | end
15 until Until no data entity remaining in X;

```

Output: S_C : An updated set of clusters

Squeezer does not require the number of clusters as an initial setup, whereas it asks for a pre-defined threshold for the similarity measure. It possesses an advantage of acceptable computational complexity with $O(nkd)$, where n represents the number of data entities in discussion, k denotes the number of clusters and d is termed as the number of attribute values for each of the data entities. Yet, it is sensitive to the order of the data entities selected each time. Hence, disparate sequences of data input may lead to different clustering consequences.

- **Spectral Clustering (SC):** SC derives from graph partitioning [84, 85]. It has become increasingly popular due to its promising performance in graph-based clustering [86]. Given a graph whose vertex represent data entities, and each edge is weighted in accordance with the pairwise relation between its involving vertices, the task of looking for a good clustering of the underlying data can be transferred into finding a nice partition of the graph. In general, this metric reduces the dimensionality of data via the spectrum of similarity matrix at the beginning, then performs a simple

clustering algorithm such as k -means on the transformed data. The process of SC is described as follows:

1. Create a normalised Laplacian matrix from similarity matrix obtained from the original dataset.
2. Perform an eigenvalue decomposition process for the Laplacian matrix resulted from Step 1.
3. Select k eigenvectors with respect to the k largest eigenvalues to create a new matrix U with k dimensions.
4. Perform a simple clustering algorithm to U and acquire the final clustering result.

SC requires no assumptions on shapes of clusters, and can even handle unusual form of interwind spirals [87]. However, it is widely accepted that the performance of SC relies heavily on the characteristics of the similarity matrix or the graph structure [88]. It is computationally expensive (with the time complexity of $O(n^3)$) unless the graph generated from data source is sparse [89].

2.2 Link Prediction Metrics

As described in Section 1.3, link prediction performs a significant role for the emerging research field of social network analysis (SNA), which fundamentally outlines structures including actors and relationships amongst them [50]. These networks can be conveniently represented by employing vertices and links. The links show types of relationship amongst the actor (represented by vertices) in such a network [51]. Obviously, the concept of SNA models can be generalised and widely applied in various real-world applications. They are not only restricted to the use in networks concerning human beings, but also can be utilised to depict and analyse the structures in a wide variety of problem domains.

Consider a network, represented by a graph $G(V, E)$ at a particular time t_0 , where V and E each denotes a set of vertices and edges in G , respectively. The task of link prediction aims at searching for potential links or unobserved links between vertices for a specific future moment t_1 ($t_1 > t_0$) in the current network. There exists many generic, simple and basic link prediction metrics, which use information of vertices, topology and social theory to calculate the similarities of vertex pairs. Moreover, learning-based link prediction methods are more complex, but they are established on features provided by the basic metrics and external

information. In this section, a systematic review of link prediction metrics is presented.

2.2.1 Vertex-based Metrics

Computing the similarity between a pair of vertices in a network is an intuitive scheme for solving the link prediction problem. It is natural to believe that the more similar the pair is, the more likely that there exists a link connecting them, and vice versa. This is consistent with the fact that people are prone to establish relationships with others who possess the common religious belief, share similar interests, or hold the same educational background. Such likelihood can be measured by similarity metrics. Particularly, a pair of vertices (x, y) in G is assigned a score representing the similarity degree between x and y . A higher score indicates that x and y are likely to be linked sometime in future, whereas a lower score implies that x and y are less probable to be connected. Therefore, the task of predicting disappearing or unobserved links in a network is accomplished by ranking the similarity scores amongst vertex pairs.

In a real-world network, its involving vertices usually possess intrinsic attributes or useful information, e.g., the personal profile in online social networks, mail name and address in email networks, and publication record in academic networks. Such information can be directly captured for calculating the similarity degree between the referencing vertices. As these attribute values are in the textual format in most cases, the text-based and string-based similarity metrics are naturally performed for the scenario [90–92]. Similarly, common interests or behaviours shared by actors (represented by vertices in the network) can also be utilised to measure the similarity amongst them. These features are frequently characterised by the actions they take and usually represented as a vector. Thus the similarity between two vertices is measured by matching their respective action vectors [93]. [94] proposed a complement to link prediction by inferring a portion of the unobserved values to the networks with missing information before executing similarity computation. This concept is adaptable to various real-world networks.

In short, vertex-based metrics are mainly applied to the circumstances under which the attributes and behaviours of the included vertices are available.

2.2.2 Topology-based Metrics

In a simple network where vertex attribute information is unavailable, topological structure is widely explored to search the association pattern amongst its involving vertices. An initial systematic review on topology-based metrics is presented in [57]. And a number of metrics of the same category were proposed since then [95]. These metrics can be categorised into neighbour-based metrics, path-based metrics, and random-walk-based metrics, according to their intrinsic characteristics.

2.2.2.1 Neighbour-based Metrics

In a graph network G , a neighbour vertex of a vertex x is a vertex that is connected to x with an explicit edge. The neighbourhood of a vertex x in G is the subgraph of G induced by all vertices connected to x , denoted as $\Gamma(x)$. Inspired by the notion that the neighbours may share close relationships between each other and the information acquired from the neighbourhood may provide strong evidence for prediction, researchers have designed a number of neighbour-based metrics with their improved variations for link prediction.

- **Common Neighbours (CN)** The CN predictor captures the idea that two strangers who have a common friend may be introduced by that friend. It is one of the most popular measurements applied in link prediction problems owing to its low computational complexity [96]. For two vertices, x and y , CN is directly defined by the number of common neighbours that both x and y have. A considerable number of the common neighbours indicates the great possibility to establish a link between x and y . This metric is defined in the following formula:

$$CN(x,y) = |\Gamma(x) \cap \Gamma(y)| \quad (2.6)$$

- **Jaccard Coefficient (JC)** JC is originally proposed as a similarity metric for information retrieval [97]. The concept of this metric has been transferred into task of link prediction in recent studies. Mathematically, it normalises the CN score by considering the amount of neighbours that either x or y has. The JC score is defined as:

$$JC(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \quad (2.7)$$

- **Weighted Jaccard Coefficient (WJC)** This metric is an improved version of JC by considering the available information amongst pairs of vertices in the neighbourhood,

with the information presented by edge weights. Basically, for the edges which are connected to common neighbours from target vertex-pair, WJC performs a normalisation step to their edge weights [98]. It assigns greater values for vertex-pairs which share a higher sum of weights over common neighbours relative to the total weights of all the neighbours they have. For two distinct vertices x and y , this measure is defined by

$$WJC(x,y) = \frac{\sum_{z \in \Gamma(x) \cap \Gamma(y)} w(x,z) + w(y,z)}{\sum_{z' \in \Gamma(x) \cup \Gamma(y)} w(x,z') + w(y,z')} \quad (2.8)$$

where $w(x,z)$, $w(y,z)$, $w(x,z')$, $w(y,z')$ each denotes the edge weights between vertex-pair (x,z) , (y,z) , (x,z') , (y,z') , respectively.

- **Adamic-Adar Coefficient (AAC)** The AAC metric was initially proposed for computing the similarity between two distinct webpages [99]. Nowadays, this metric attracts more attention in social network analysis. Unlike JC and its variation forms which only consider the direct neighbours to the target vertex, the AAC metric also takes neighbour vertices of neighbours into account. The mathematical definition of AAC is articulated as:

$$AAC(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log |\Gamma(z)|} \quad (2.9)$$

It is apparent that in AAC, common neighbours of vertex-pair (x,y) which have fewer number of neighbours, are weighted more heavily.

- **Resource Allocation (RA)** RA metric is inspired by the physical process of resource allocation [100]. Actually, it is a variant of AAC, with heavier punishment to the contribution of the high-degree common neighbours in the network. The calculation formula of RA is as follows:

$$RA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{|\Gamma(z)|} \quad (2.10)$$

- **Weighted Resource Allocation (WRA)** Similar to WJC, WRA extract available information amongst pairs of vertices in the network, presented by weights. Formally,

WRA is defined as:

$$WRA(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{w(x, z) + w(y, z)}{s(z)} \quad (2.11)$$

where, $w(x, z)$ and $w(y, z)$ denotes the edge weights of vertex-pair (x, z) and (y, z) , respectively, and $s(z)$ depicts the sum of weights for the vertex z associated with all of its existing edges, such that

$$s(z) = \sum_{a \in \Gamma(z)} w(z, a) \quad (2.12)$$

- **Preferential Attachment (PA)** PA has received great attention as a metric to measure the growth of networks [96, 101]. The PA metric suggests that new links are more possibly to be constructed between higher-degree vertices rather than lower ones. It is mathematically defined as:

$$PA(x, y) = |\Gamma x| \cdot |\Gamma y| \quad (2.13)$$

In Table 2.1, a comparison of the popular neighbour-based metrics for link prediction is illustrated, with respect to normalization step, time complexity analysis and characteristic description. It is clear that four metrics: CN, AAC, RA and PA, do not involve normalisation steps, which implies that the similarity degrees for vertex-pairs calculated through these metrics only have the ranking meaning. Time complexity is a significant factor when selecting metrics, especially for large scale networks. Assume that a network contains m vertices, each vertex have n neighbours on average, and the network is stored in the format of adjacent matrix, the time complexity of finding all neighbours of a vertex is $O(m)$, and the time complexity of calculating the intersection or union of two sets is $O(n^2)$. All the approaches within discussion include the step of searching for neighbours of two involving vertices, which results in time complexity of $O(2m)$. However, the overall time expenses of calculating the similarity between a pair of vertices with respective metrics may vary from each other. Obviously, PA is the most efficient metric amongst all the discussed methods, since it only needs the above mentioned step. CN, containing one more step of computing the intersection of two neighbour sets, leads to the total time cost of $O(2m + n^2)$. The time complexity of JC and WJC is $O(2m + n^2)$, as it encapsulate steps of calculating both intersection and union of two sets. For AAC, RA and WRA, besides the aforementioned steps, they also need to search for the neighbours for each of the common neighbour vertices to the target vertex-pair, which lead to an additional time cost of $O(mn)$. Therefore, their

time complexities are $O((n+2)m+n^2)$.

Table 2.1 Comparison of Neighbour-Based Metrics for Link Prediction

Metric	Normalisation	Time Complexity	Characteristic Description
CN	No	$O(2m+n^2)$	(1)
JC	Yes	$O(2m+2n^2)$	(1)(2)
WJC	Yes	$O(2m+2n^2)$	(3)
AAC	No	$O((n+2)m+n^2)$	(4)(5)
RA	No	$O((n+2)m+n^2)$	(4)(6)
WRA	Yes	$O((n+2)m+n^2)$	(3)(4)
PA	No	$O(2m)$	(1)(7)

- (1) Simple and intuitive
- (2) Relative to total number of neighbours
- (3) Rich in edge information
- (4) Consider neighbour of common neighbours
- (5) Common neighbours with fewer neighbours weighted more heavily
- (6) Punish high-degree common neighbours more heavily
- (7) Prefer high-degree vertices

It should be noted that although there are great many neighbour-based metrics with their variations available, proper selection of metrics according to the characteristics of networks is necessary and essential, since quantities of experimental evaluations have demonstrated that there exists no absolute dominating metric for various practical applications [57].

2.2.2.2 Path-based Metrics

Apart from the vertex and neighbour based metric, paths connecting two vertices in the network can also be explored to measure the similarities between vertex-pairs. This yield a different category of similarity measure, named as path-based metrics.

- **Katz** For a pair of vertices (x,y) , Katz metric [102] considers the ensemble of all paths existing between them, and it straightforwardly sums up all the weights over the considering paths. However, it penalise the contribution of longer paths in the

similarity computation by setting up a damp factor corresponding to the path length. The formal equation to compute the Katz value is as follow:

$$katz(x,y) = \sum_{l=1}^{\infty} \beta^l \cdot |paths_{x,y}^l| \quad (2.14)$$

where l denotes the path length, β^l indicates the damp factor and $paths_{x,y}^l$ represents the set of all paths from x to y with length l . β is usually set to a small value greater than zero. A tiny value of β may lead Katz measure considering only the shorter paths heavily, resulting in this path-based metric working in a similar way to the neighbour-based counterparts. One problem with Katz metric is its cubic computational complexity, which could be infeasible for large networks [103].

- **Local Path (LP)** LP metric [104] can be regarded as a special case of Katz. Unlike the metrics that only use the information of the nearest neighbours (be they adjacent or otherwise), it makes use of further information from local paths with a length value of 2 and 3. Let A denote the adjacent matrix of all vertices in the discussed network, and A^2 and A^3 represent the adjacent matrices based on A with a length of 2 and 3, respectively. Here, A^2 contains two vertices in the network connected through a path with length of 2. Similarly, each element in A^3 denotes two vertices in the network connected through a path with length of 3. LP is then defined as follows:

$$LP = A^2 + \alpha A^3 \quad (2.15)$$

where α is a small number close to zero, which is being used to penalise the weight of the paths with greater length. In the experiment, α is set to 0.01 (as with the default value typically used when running this metric). An extended version of LP is named as Local Weighted Path (LWP), which makes use of further information from local paths [105]. LP metric has the cubical time complexity since it involves matrix multiplication.

- **Relation Strength Similarity (RSS)** This metric was originally introduced as an asymmetric measure for weighted social networks [106]. It may also be adopted as a symmetric measure for the various real-world problems [107] [108]. Suppose that there are T simple paths (with no circles in paths) shorter than a path length of e from the vertex x to y in the network, and a path with length of u ($u \leq e$) from x and y is formed with Z vertices $x_1, x_2, \dots, x_{Z-1}, x_Z$, where x_1 represents x and x_Z represents y .

Then, the RSS metric from x to y is defined by

$$RSS(x,y) = \sum_{t=1}^T R_u^*(x,y) \quad (2.16)$$

with

$$R_u^*(x,y) = \begin{cases} \prod_{z=1}^{Z-1} R(x_z, x_{z+1}) & Z \leq e+1 \\ 0 & otherwise \end{cases} \quad (2.17)$$

where $R(x_z, x_{z+1})$ denotes the link strength of the two adjacent vertices x_z and x_{z+1} within a particular path connecting x and y . The time complexity for RSS is significantly affected by the length of paths, and can be extraordinarily high in extreme cases.

- **Connected-Path (CP)** The CP metric further includes the uniqueness property of the link patterns to refine the similarity estimation [109]. In particular, for a link network $G = (V, W)$, a path established between two vertices $x, y \in V$, named as $path(x, y)$ or p , is a sequence of unique vertices x, x_1, \dots, x_n, y with edges $w_{x, x_1}, w_{x_1, x_2}, \dots, w_{x_{n-1}, x_n}, w_{x_n, y} \in W$. The length of path $length(p)$ equals $|p| - 1$, where $|p|$ denotes the number of vertices in path p . Additionally, $PATH(x, y, r)$ represents the set of all paths between x and y whose length meets the criterion of $2 \leq length(p) \leq r$. Thus, the similarity degree between x and y is defined by the accumulated uniqueness measure acquired from all paths in $PATH(x, y, r)$. This metric can be formally described as follows:

$$CP(x,y) = \sum_{p \in PATH(x,y,r)} \frac{U(p)}{length(p)} \quad (2.18)$$

where $U(p)$ denotes the uniqueness of path p , which can be calculated through the following equation:

$$U(path(x,y)) = \prod_{z \in path(x,y), z \neq x,y} UQ(z) \quad (2.19)$$

In Eq. 2.19, $UQ(z)$ represents the uniqueness score of vertex $z \in path(x, y)$, which is computed by Eq. 2.20.

$$UQ(z) = \frac{w_{z,z-1} + w_{z,z+1}}{\sum_{\forall g \in V} w_{z,g}} \quad (2.20)$$

where $w_{z,g}$ denotes the weight of an edge between $z \in path(x,y)$ and any other vertex $g \in V$, $w_{z,z-1}$ and $w_{z,z+1}$ represent the weights of the edges from z to its adjacent vertices in $path(x,y)$, respectively.

Note that in Eq. 2.18, the uniqueness degree of a path is normalised by its length, which shows that the longer paths are considered to be less informative than the shorter ones. Such insight is consistent with human logical thinking intuitively. In order to set the CP score into a unified scale, a final step of normalisation is needed, with CP_{max} (maximum estimate value between any two vertices in G) being the normalisation factor. Hence, the similarity degree $sim_{CP}(x,y)$ between x and y by CP metric can be determined as:

$$sim_{CP}(x,y) = \frac{CP(x,y)}{CP_{max}} \quad (2.21)$$

Similar to RSS metric, the time complexity for CP metric is also sensitive to the size of the network. Assume that a link network involves m vertices, and each of them is linked to n others on average, the time complexity for CP metric to generate all pairwise similarity score is $O(m^2n^r)$.

2.2.2.3 Random Walk-based Metric

Social interactions between vertices in linked graph networks can also be modelled via random walk, which investigates transition probabilities from a single vertex to its neighbours and propagates such probabilities to calculate the likelihood of a destination travelled by a random walker from the source vertex. There exists a number of link prediction metrics which compute similarity degrees between vertices based on the concept of random walk.

- **SimRank (SR)** The well-known SimRank algorithm [110] was proposed on the basis of the intuition that two vertices within a graph are similar if they are connected to similar vertices in the graph. This can obviously be adapted for use in various studies. For a pair of vertices x and y , their SR score is computed by

$$SR(x,y) = \frac{\gamma \sum_{x' \in \Gamma(x)} \sum_{y' \in \Gamma(y)} SR(x',y')}{|\Gamma(x)||\Gamma(y)|} \quad (2.22)$$

where $\gamma \in [0, 1]$ is a decay factor that represents the confidence level of accepting two non-identical vertices as similar.

The SR algorithm performs through an iterative process, with $SR_k(x, y)$ denoting the k th iterative computation for the similarity between x and y . At initial stage, $SR_0(x, y)$ is set to:

$$SR_0(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases} \quad (2.23)$$

It has been shown that the value of $SR_k(x, y)$ is not decreasing as k increases, and it eventually converges to a stable limit [110]. In practical applications, k is usually set to a small number to control the computational complexity.

- **PageSim (PS)** PS [111] was developed to capture similar web pages based on associations implied by their hyperlinks. Fundamentally, the similarity degree between a pair of vertices x and y is dictated by the coherence of ranking scores propagated to them from any other vertices in V . It is noteworthy that ranking scores are explicitly generated using the page rank scheme of the well known Google search engine [112]. Specifically, for a network $G = (V, E)$, let $PR(x)$ denote the PageRank score of a vertex $x \in V$. $PR(x)$ can be estimated from the following iterative procedure (i.e., $PR(x) = \lim_{k \rightarrow \infty} PR_k(x)$):

$$PR_k(x) = (1 - \beta) + \beta \sum_{z \in \Gamma(x)} \frac{PR_{k-1}(z)}{|\Gamma(z)|} \quad (2.24)$$

where k is the number of iterations, and β is a decaying factor ranging between $[0, 1]$ and usually set to 0.85 [112]. $PR_0(x)$, the initial state of PR score for vertex x in the iteration process is set to 1 in most of the cases [113]. Having achieved this, the PR score of x propagated from x to $y \in V$ can be calculated as follows:

$$PG(x, y) = \begin{cases} \sum_{p \in PATH(x, y, r)} \frac{d \cdot PR(x)}{\prod_{z \in p, z \neq y} |\Gamma(z)|} & x \neq y \\ PR(x) & x = y \end{cases} \quad (2.25)$$

where d is a damping factor, $PATH(x, y, r)$ represents a set of paths from x to y with maximum length of r . Assume that $|V| = m$ ($|V|$ denotes the number of vertices in V), let $PS(x, y)$ be the PS score of vertex-pair (x, y) , $PS(x, y)$ is defined as:

$$PS(x, y) = \sum_{i=1}^m \frac{\min(PG(x_i, x), PG(x_i, y))^2}{\max(PG(x_i, x), PG(x_i, y))} \quad (2.26)$$

Note that PS is a symmetric method with $PS(x, y) = PS(y, x)$ and $PS(x, y)$ is always in a range of $[0, 1]$. Moreover, each vertex in the network is most similar to itself, i.e., $PS(x, x) = \max_{z \in V} PS(x, z)$.

The time complexity of PS is huge. Assisted with the prune technique to limit the length of propagation, the time complexity of PS metric can be reduced to $O(m^2 n^{2r})$ (n denotes the average number of neighbours for each vertex in the network), which still makes it infeasible to handle large sized networks.

- **Hitting Time (HT)** The basic concept of HT derives from random walks on a graph [114]. For a pair of vertices x and y in a graph network, let $HT(x, y)$ defines the expected number of steps required for a random walk starting at x to reach y . The smaller value of HT denotes that the vertices are similar to each other, thus they have a higher opportunity of creating a link between them. Similar to RSS discussed in 2.2.2.2, the original HT is designed for directed graph network. However, a variant of HT, named as commute time (CT), has been proposed to handle with the bi-directed graph network by considering both $H(x, y)$ and $H(y, x)$. Specifically,

$$CT(x, y) = HT(x, y) + HT(y, x) \quad (2.27)$$

The HT metric can benefit from its simple computational complexity, but its predicted value may possibly lead to high variance [57]. Particularly, one difficulty with HT metric is that $HT(x, y)$ could be quite a small value when y is a vertex with a large stationary probability $\pi(y)$, regardless of the identity of x . To overcome this drawback, normalised versions of HT, denoted by NHT, is proposed by defining $NHT(x, y) = HT(x, y) \cdot \pi(y)$. Likewise, normalised version of CT metric, named as NCT, can also be depicted by $NCT(x, y) = CT(x, y) \cdot \pi(y) + CT(y, x) \cdot \pi(x)$.

Another issue with HT metric is its sensitivity to parts of the graph which are far from the target vertex-pair. For instance, $HT(x, y)$ (even when x and y are connected by very short paths in the graph network) can be seriously affected by another connected vertex z in the graph network, where z is far away from x and y with high stationary probability

$\pi(z)$. It could be difficult for a random walk to escape from the neighbourhood of z . A feasible method to tackle this problem is to allow the random walk from x to y to periodically reset, returning to x with a fixed probability α at each step. Based on this method, vertices within a certain range to the target vertex-pair is considered more heavily, while distant parts to the target vertex-pair in the graph network will be rarely explored.

- **Rooted Pagerank (RP)** It has been demonstrated that the PageRank measures [112] used for webpage ranking has inherent relationship with the HT analysis. Hence, the pagerank score can also be treated as an indicator for link prediction. However, since the pagerank score itself is an index for a single vertex, it requires to be modified such that it can be utilised to represent the similarity degree between a pair of vertices. The original concept of pagerank is designed based on a webpage network under the following assumption: for some fixed probability α , a surfer at a webpage jumps to a random webpage with the probability of α and goes to another webpage directed by a hyperlink with the probability of $1 - \alpha$. The importance degree of a webpage web_a is expected to be the sum of the importance degree of all the webpages that link to web_a . The term importance degree used in pagerank scheme can be transferred into stationary distribution under the random walk strategy for networks. For the task of link prediction, the similarity degree between a pair of vertices x and y can be measured as the stationary probability of y in a random walk that returns to x with the probability of $1 - \alpha$ in each step, and moves to a random neighbour with the probability of α . Similar to HT, the RP metric is defined to be asymmetric originally and can be modified into a symmetric version by summing with the counterpart in which the roles of x and y are exchanged. Mathematically, let A be an adjacent matrix for the discussed graph network G , I be an Identity matrix in the same size of A , D be the diagonal degree matrix with $D[i, i] = \sum_j A[i, j]$ and $N = D^{-1}A$ be the adjacent matrix with row summation normalised to 1, the RP score for all vertex-pairs in G can be calculated as follows:

$$RP = (1 - \alpha)(I - \alpha N)^{-1} \quad (2.28)$$

- **PropFlow (PF)** PF metric [115] is a similarity measure based on information flow [116], and can also be regarded as localised version of RP. It has the advantage of being insensitive to the topological noise far from the target vertices. Unlike RP metric, the process of performing PF does not require a walk restart or convergence, but simply

employs a searching strategy with limited length l . Therefore, it is more efficient than RP and SR metrics in computational complexity. For a pair of vertices (x, y) in a graph network G , their PF score is proportional to the probability that a random walk starting from x and ending at y within l steps. Equation 2.29 shows the formula of computing PF score for (x, y) when they are directly linked.

$$PF(x, y) = PF(a, x) \frac{w_{xy}}{\sum_{k \in \Gamma(x)} w_{xk}} \quad (2.29)$$

where, w_{xy} denotes the weight of the link between vertices x and y , k represents x 's neighbours whose length to the starting vertex is greater than that of x , a depicts the previous vertex of x on a random walk path. Note that when x is the starting vertex, $PF(a, x)$ is set to be 1 by default. If x and y are not directly connected, PF metric sums up the PF score of the all the shortest paths from vertex x to y . A pseudo code of the PF predictor for estimating x to all the other vertices within maximum length l in a graph network is shown in Algorithm 2.

Algorithm 2: PropFlow Algorithm for Prediction**Input:** $G = (V, W)$: A graph network x : Source vertex in G l : Maximum length $Found$: A set to store available vertices $NewSearch$: A set to store vertices $OldSearch$: A set to store vertices S : A set to store a vertex with its flow score

```

1 INSERT  $x$  into  $Found$  ;
2 INSERT  $x$  into  $NewSearch$  ;
3 INSERT  $(x, 1)$  into  $S$  ;
4 for  $CurrentDegree \leftarrow 0$  to  $l$  do
5    $OldSearch \leftarrow NewSearch$  ;
6   EMPTY  $NewSearch$  ;
7   while  $OldSearch$  is not empty do
8     REMOVE  $x_i$  from  $OldSearch$  ;
9     FIND  $VertexInput$  using  $x_i$  in  $S$  ;
10     $SumOutput \leftarrow 0$  ;
11    foreach  $x_j$  in neighbours of  $x_i$  do
12      | ADD  $w_{ij}$  to  $SumOutput$  ;
13    end
14     $Flow \leftarrow 0$  ;
15    foreach  $x_j$  in neighbours of  $x_i$  do
16      |  $Flow \leftarrow VertexInput \times \frac{w_{ij}}{SumOutput}$  ;
17      | INSERT or SUM  $(x_i, Flow)$  into  $S$  ;
18    end
19    if  $x_i$  is not in  $Found$  then
20      | INSERT  $x_i$  into  $Found$  ;
21      | INSERT  $x_i$  into  $NewSearch$  ;
22    end
23  end
24 end

```

Output: $PF(x, y)$ for all neighbours y of x under neighbourhood range within length of l

2.2.3 Probabilistic Metrics

The probabilistic metrics for link prediction are considered supervised models and are mostly based on Bayesian Theory. The general concept for probabilistic metrics is to grasp the posterior probability which denotes the chance of co-occurrence for the vertex-pairs in the discussed domain. Generally, probabilistic metrics are time consuming and only capable of handling networks involving thousands of entities(vertices). However, they provide valuable insights into the network organization and modularisation, which can not be fully grasped by the similarity-based metrics [95, 117].

2.2.3.1 Hierarchical Probabilistic Graph Metric

Studies have suggested that real-world networks may exhibit hierarchical structure, where vertices divided into groups can be further clustered into groups of groups, and so forth over multiple scales [50, 118, 119]. This metric infers hierarchical structure from network data and can be used for prediction of missing links. Basically, it is proposed as a probabilistic model for hierarchical random graphs. The learning task is to use the observed network data to fit the most likely hierarchical structure through statistical inference: a combination of the maximum likelihood approach and a Monte Carlo sampling algorithm.

The conceptual framework for this hierarchical probabilistic model (HPM) can be detailed as follows: Let $G = (V, W)$ be a graph network with m vertices. A dendrogram D is a binary tree with m leaves corresponding to the vertices of G . Each of the $m - 1$ internal nodes of D corresponds to the group of vertices that are descended from it. A probability p_r is associated with each internal node r . Then, given two vertices $x, y \in V$, the probability p_{xy} that they are connected by an edge is $p_{xy} = p_r$, where r is the lowest common ancestor in D . Thus, the likelihood of the hierarchical random graph can be presented as a combination of $(D, \{p_r\})$ which consists of the dendrogram D and the set of probabilities $\{p_r\}$.

The learning task is to find the hierarchical random graph or graphs that best fits the observed network data. Assume that all hierarchical graphs are *a priori* equally likely, by Bayesian theorem, the probability that a given model $(D, \{p_r\})$, is the correct explanation of the data is proportional to the posterior probability or likelihood with which the model generates the observed network. The goal is to maximize such posterior probability or likelihood.

Let E_r denote the count of edges in G whose endpoints have r as their lowest common ancestor in D , and let L_r and R_r , represent the numbers of leaves in the left and right subtrees

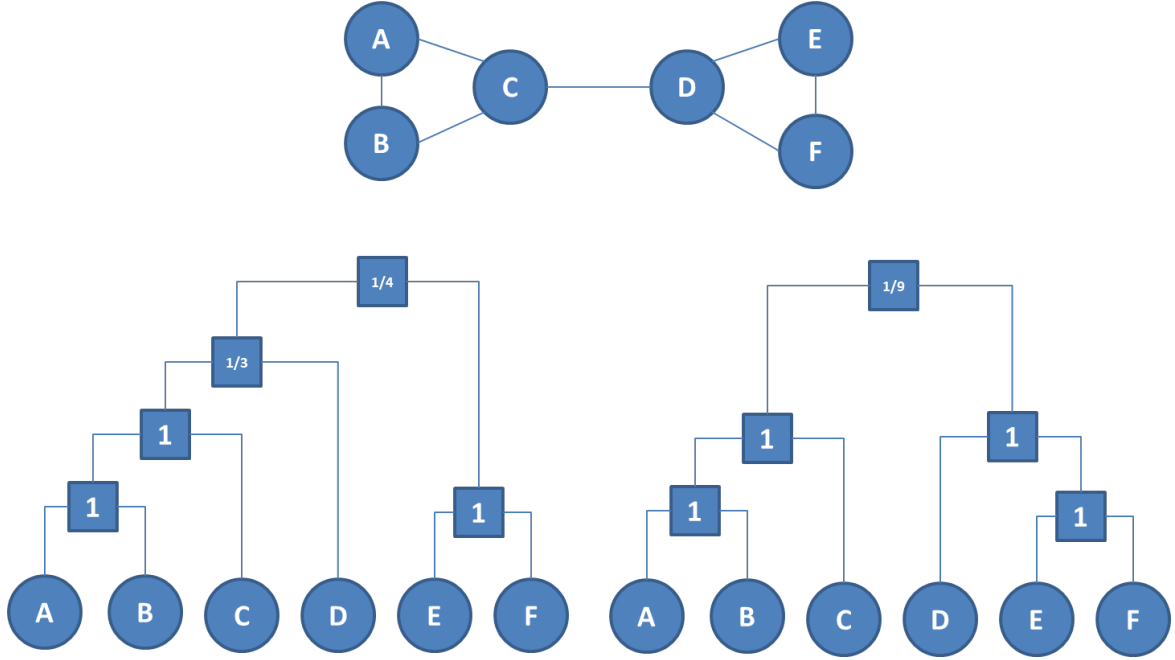


Fig. 2.1 Example of Hierarchical Structures for a Link Network

rooted at r , respectively. Then, the likelihood \mathcal{L} of the hierarchical random graph consisting D and $\{p_r\}$ can be calculated as:

$$\mathcal{L}(D, \{p_r\}) = \prod_{r \in D} p_r^{E_r} (1 - p_r)^{L_r R_r - E_r} \quad (2.30)$$

Such likelihood measure can be maximised by a set of probabilities $\{\bar{p}_r\}$, with \bar{p}_r calculated as :

$$\bar{p}_r = \frac{E_r}{L_r R_r} \quad (2.31)$$

An illustrated example for the hierarchical structure of a link network is presented in Fig. 2.1. As is shown, the example network G consists of six vertices, followed by the likelihood of two possible dendrograms presented. The internal nodes r of each dendrogram (represented by box) are labelled with the maximum likelihood probability \bar{p}_r . According to Eq.2.30, the likelihoods of the two dendrograms are $\mathcal{L}(D_1) = (1/3)(2/3)^2 \cdot (1/4)^2(3/4)^6 = 0.00165$ and $\mathcal{L}(D_2) = (1/9)(8/9)^8 = 0.0433$, respectively. It is obvious that the second dendrogram reveals the hierarchical structure of the example network in a more acceptable manner (which is also demonstrated by $\mathcal{L}(D_2) > \mathcal{L}(D_1)$), since it correctly divides the network into two significantly connected sub-graphs at the first stage.

The choice amongst the dendrograms are performed by a Markov Chain Monte Carlo (MCMC) sampling approach with probabilities proportional to their likelihood. To create the Markov chain, the approach first generates a set of transitions between possible dendrograms through rearrangement or reconstruction. In the rearranging step, the approach chooses an internal node of a dendrogram and then selects among various configuration of the subtree at that vertex uniformly. Once the transition criteria is clear, the sampling process creates a random walk. Whether a new rearrangement being accepted or not is determined by the Metropolis-Hastings sampling rule, i.e., for a transition from a the original dendrogram D to a reconstructed dendrogram D' , the transition is adopted if $\Delta \log \mathcal{L} = \log \mathcal{L}(D') - \log \mathcal{L}(D)$ is non-negative, otherwise it is adopted with a probability of $\mathcal{L}(D')/\mathcal{L}(D)$.

For link prediction, a set of sample dendrograms are created at regular intervals once the MCMC random walk reaches an equilibrium. Then, for the pair of vertices x and y with no connection exists currently, the model computes a mean probability $p(xy)$ of they are being connected by averaging over the corresponding probability $p(xy)$ for all of the sampled dendrograms. For binary decision, a threshold can be specified. The unique nature of the hierarchical random graph model is that it enables presenting a general view for the network. Simultaneously, it allows sampling over the set of hierarchical structures to obtain a consensus probability. The drawback of this metric is that it may not be that accurate unless the MCMC sampling process converges to the stationary distribution in a limited or reasonable number of steps. Additionally, the entire process could be very costly for networks in large size.

2.2.3.2 Stochastic Block Metric

Stochastic block model (SBM) is one of the most general network models in which the vertices are partitioned into groups [120, 121]. The probability for the connection of two vertices in the network depends completely on their membership to those groups. The stochastic block model is capable of capturing the community structure [122], role-to-role connections [123], and group based interactions [124]. Conceptually, for a link network G , a block model $M = (P, Q)$ consists of a partition P dividing all the vertices in G into distinct groups and a matrix Q involving probabilities of linkage between pairs of groups. Arbitrarily selecting two different groups from P , named as V_α and V_β respectively, denoting their connecting probability by $Q_{\alpha\beta}$, then, the likelihood of G based on a pair of P and Q can be defined as follows:

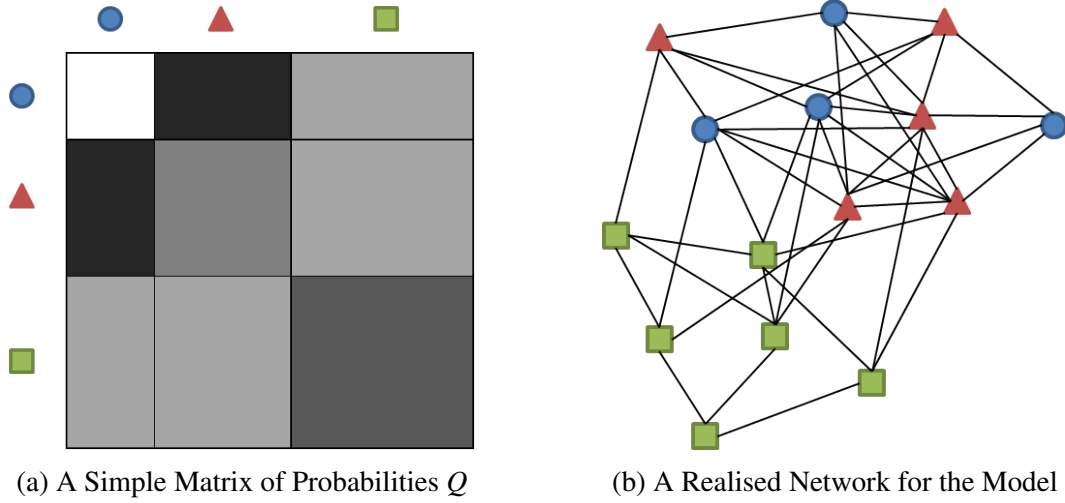


Fig. 2.2 Example of Stochastic Block Model

$$\mathcal{L}(G|P, Q) = \prod_{\alpha \leq \beta} Q_{\alpha\beta}^{l_{\alpha\beta}} (1 - Q_{\alpha\beta})^{r_{\alpha\beta} - l_{\alpha\beta}} \quad (2.32)$$

where $l_{\alpha\beta}$ denotes the number of existing edges inter-connecting vertices in groups V_α and V_β , and $r_{\alpha\beta}$ denotes the number of vertex-pairs (x, y) with $x \in V_\alpha$ and $y \in V_\beta$ respectively. Thus, $Q_{\alpha\beta}^*$, the optimal value of $Q_{\alpha\beta}$ which maximises $\mathcal{L}(G|P, Q)$ is:

$$Q_{\alpha\beta}^* = \frac{l_{\alpha\beta}}{r_{\alpha\beta}} \quad (2.33)$$

An example of SBM is illustrated in Figure 2.2. A matrix Q in Fig. 2.2a presents the vertices in discussion are divided into three distinct groups, each of which contains 4, 5, 6 vertices, respectively, and are represented as circles, triangles and squares according to their membership to different groups. The value of each matrix entry $Q_{\alpha\beta}$ is represented by the shade degree of grey. For instance, a circle does not link to any other circles. However, circles link to squares with tiny probability, and link to triangles with high probability. Fig. 2.2b is a realised network of Q . It is apparent to observe that the number of links between the circle and the triangle is much more than that between the circle and the square, which is consistent with the contents revealed by Q .

Let Ω be the set of all possible partitions on G , by the Bayesian Theorem [125], the reliability of an individual link between vertices x and y can be calculated as:

$$\mathcal{L}(G_{xy} = 1|G) = \frac{1}{Z} \sum_{P \in \Omega} \int_N \mathcal{L}(G_{xy} = 1|P, Q) \mathcal{L}(G|P, Q) p(P, Q) dQ \quad (2.34)$$

where Z is a normalising factor, N denotes the number of distinct group pairs, $\mathcal{L}(G_{xy} = 1|P, Q)$ is determined as $Q_{\alpha\beta}^*$, and $p(P, Q)$ is set to a constant value since the equation is formulated under the assumption that no prior knowledge about the models is provided.

Note that the number of different partitions is incredibly huge, which makes it infeasible to sum over the results of the all possible partitions in real practice. Thus, similar to HPM, a Metropolis sampling procedure [126] is applied to assist the estimation of the link reliability. However, the whole process still leads to expensive time cost which is only suitable for handling small to medium sized networks [117].

In spite of its ordinary performance in time efficiency, SBM is widely applied in various fields [127–130], as it is not only capable of undertaking the task of predicting missing links in the network, but also provides an approach to identifying possible spurious links (the existent links with the low reliabilities) [131].

2.3 Fuzzy Link Prediction

Fuzzy set theory enables processing with approximation, uncertainty and imprecision, where a number of real-world problems cannot be successfully tackled by binary encoding to model [132]. Fuzzy logic which tolerates partial truth is the premise to perform approximate reasoning, which aims at producing conclusions from inexact assumptions while being analogous to human logical thinking. Fuzzy systems, based on fuzzy logic, encapsulate both the natural observation and information from domain experts' descriptions in terms of natural language, and generate interpretable knowledge in linguistic expressions for the transparent insights into the behaviour of a complex system.

2.3.1 Fuzzy Inference System

Fuzzy inference system (FIS) [133–136] is a type of fuzzy systems designed to formulate human knowledge for reasoning in a systematic manner, assisted with the information coming from sensory measurements and fuzzy mathematical models. Such transformation introduced by FIS vividly map human knowledge onto mathematical formulas for systematic analysis. FIS provide a general basis which allows a wide range of applications, including decision making, pattern discovery, event prediction, system control etc [137]. The conceptual architecture for a typical FIS [138] is illustrated in Fig. 2.3.

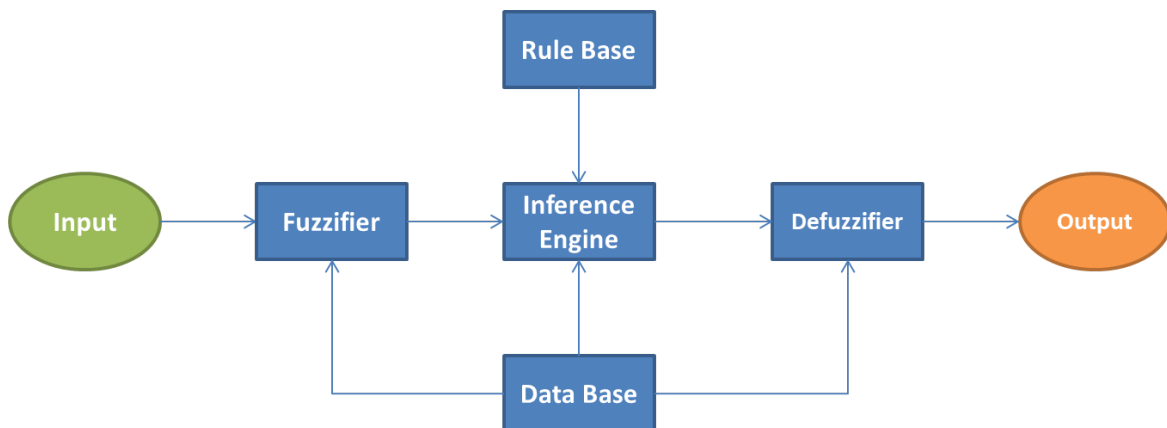


Fig. 2.3 General Architecture of FIS

As is shown in Fig. 2.3, the general architecture for a typical FIS consists of five components, including: a fuzzifier, an inference engine, a defuzzifier, a data base and a rule base. The functionality for each of these components are described as follows:

- 1 Fuzzifier: it provides an interface for transforming crisp inputs obtained from natural data sources into fuzzy presentations. The fuzzifier enables representing crisp numerical values in the form of linguistic terms in natural language with certain membership degree. The transformation procedure is conducted by searching through a collection of semantic mappings that relates the input crisp value with a group of predefined fuzzy sets.
- 2 Inference Engine: it performs operations on the fuzzy values passed from the fuzzifier. These fuzzy values of individual features attempt to fire rules stored in the rule base. The firing strength with respect to each fuzzy rule is computed by applying fuzzy logical operators on membership values of the existing conditional antecedents. In the case of multiple fuzzy rules being fired simultaneously, all the outputs generated by the fired rules are required to be combined into an aggregated fuzzy set before inputting into the defuzzifier.
- 3 Defuzzifier: it conducts a reverse mapping process on the aggregated fuzzy set obtained from the inference engine, to generate a crisp value for subsequent operation. In many real-world applications, the final output of an engineering system is required to be in the format of real-value. In order to achieve this goal, finding a representative point for such aggregated fuzzy set is necessary. The process to perform such task is termed as defuzzification. And usually, centroid, bisector, maximum are adopted to be the defuzzified value.

- 4 Data Base: it stores the definitions of crisp values with respect to a number of overlapped concepts defined as fuzzy membership functions for each of the features variables. In other words, it provides the evidence or regulations for converting crisp digits into fuzzy values and the reverse procedure. It also preserves a collection of linguistic labels which are both transparent to human users and essentially needed for reasoning.
- 5 Rule Base: it contains a set of linguistic rules. Generally, two ways are available to construct a fuzzy rule base for reasoning: (1) explicit translation of domain experts' knowledge into conditional statement; (2) generalise knowledge from observed entries represented by feature-consequence pairs. The former approach often offers high semantic level and satisfactory generalisation capability, whereas the latter is a data-driven approach which fits the situation well when there is not sufficient expertise available.

2.3.2 Fuzzy Inference for Link Prediction

Fuzzy logic and FIS have been applied in a wide variety of research and industrial fields [139–143], due to their Intrinsic ability to incorporate human expert knowledge and granular computing [144]. Its successful application in control systems and expert systems has been widely revealed in corpus of literatures [145–147]. Nowadays, the empirical concept has also been transferred into network analysis (NA), which is an emerging research field in the recent years. However, in the study of NA, most of the works based on fuzzy logic have been focused on network community detection [148–151]. leaving the investigation on fuzzy link or fuzzy logic based link prediction at rather a coarse level. In this section, existing metrics in a few number based on fuzzy model for link prediction is summarised.

2.3.2.1 Fuzzy Link Prediction Based on Clustering Analysis

- **Fuzzy Link Prediction Based on Local Clustering Coefficient (FCC):**

FCC is a soft metric for link prediction by exploring the concept of clique. In graph theory, a clique represents a subset of vertices in which every pair of distinct vertices in it are connected through a link, inducing that a subgraph formed by a clique is a complete graph [152]. Local clustering coefficient (LCC) [153] is selected to measure how neighbours of a vertex tend to form a clique. In SNA, it is considered that vertices with greater LCC are likely to assist generating links between its neighbours (currently not linked together). FCC metric [154] extends the traditional concept of LCC which simply focused on triplets related to the target vertex by performing

granular computation to take broader clusters into account. Basically, a softer definition for the concept of clique has been set up as a premise to support the FCC metric, which also provides consistency with human perception. Assume that S represents a clique, it requires the satisfaction of the following criteria:

C_1 : Most of the elements in S are closely related.

C_2 : None of the elements in S are too far from any of the others.

C_3 : Every element in S is better connected to the members of S than any element in S .

For the above mentioned criteria, basic linguistic labels (i.e., most, close, far) are defined as fuzzy terms, and the mathematical specification for each of the criteria are formulated in [155]. Let (x, y) be a pair of vertices in the network under discussion, $FCC(x, y)$, the score for a predicted link connecting them is calculated by the sum of $FCC(x) + FCC(y)$, where $FCC(x)$ and $FCC(y)$ denotes the updated definition of LCC for vertex x and y , respectively. Note that LCC for any vertex z in the network stands for its minimum satisfaction of the criteria, it can be interpreted that:

$$FCC(z) = \min_j [FCC_j(z)] \quad (2.35)$$

where j ($1 \leq j \leq 3$) is the index for each of the selecting criteria. The pseudocode of FCC metric for link prediction is shown in Algorithm 3. Taking both the existing small-world phenomenon in real network [156] and computational complexity into account, the search space for the paths is restrained to a limited area where the length of the paths is set to a small value less than or equal to 4 in applications.

Algorithm 3: FCC for Link Prediction**Input:** S_{Link} : A set of potential Links

```

1 foreach  $link \in S_{Link}$  do
2   foreach vertex  $x$  connected by  $link$  do
3     FIND all paths  $P$  with length  $l \leq K$  starting from  $x$  ;
4     COMPUTE for  $C_1$  ;
5     foreach vertex  $z$  in  $P$  do
6       FIND all paths  $P'$  with length  $l' \leq K$  starting from  $z$  ;
7       COMPUTE for  $C_2$  ;
8     end
9     foreach vertex  $r \notin P$  do
10      FIND all paths  $P''$  with length  $l'' \leq K$  starting from  $r$  ;
11      COMPUTE for  $C_3$  ;
12    end
13  end
14  SET  $C(x)$  to the minimum of  $C_1$  to  $C_3$  ;
15  COMPUTE  $FCC_{link}$  ;
16 end

```

Output: FCC_{link} for each $link \in S_{link}$

- **Fuzzy Link Prediction Based on Cluster Overlapping (FCO):**

FCO considers the overlapping degree of two clusters with regard to the target vertex-pair [154]. Formally, for a vertex x in the network, its corresponding cluster $C^l(x)$ is represented as a set of vertices which can be reached from x within l steps. In particular, for a pair of vertices (x, y) in the network, let $C^l(x)$ and $C^l(y)$ be their respective cluster. The likelihood of link existed between them can be measured by the overlapping degree of their respective clusters. The mathematical formula for calculating such overlapping degree $OD(x, y)$ is stated as follows:

$$OD(x, y) = \frac{\sum_{p=1}^n close(x, y)}{|\sum_{s, t \in C^l(x)} w(s, t)| + |\sum_{s', t' \in C^l(y)} w(s', t')|} \quad (2.36)$$

where n is termed as the number of possible paths connecting x and y , $w(s, t)$ represents the weight of an edge between two vertices s and t in $C^l(x)$. Similarly, $w(s', t')$

represents the weight of an edge between two vertices s' and t' in $C^l(y)$. In Eqn. 2.36, $close(x,y)$ is defined as:

$$close(x,y) = \begin{cases} 1 & l(x,y) < 2 \\ \frac{w(x,z)+w(z,y)}{2*10^{l(x,y)-2}} & l(x,y) = 2 \\ \frac{w(x,z)+w(z,e)+w(e,y)}{2*10^{l(x,y)-2}} & l(x,y) = 3 \\ 0 & l(x,y) > 3 \end{cases} \quad (2.37)$$

where $l(x,y)$ denotes the length of the path between vertex x and y , with z, e each depicts a vertex in path. It is worth mentioning that when the cluster for a vertex only contains its adjacent neighbours, this metric can be simplified into a variant of JC.

2.3.2.2 Fuzzy Link Prediction Based on Order-of-Magnitude Metric (FOM)

A common disadvantage of the numerical measures previously discussed is their inability to achieve coherent and natural interpretation via existing seemingly fine-grained scales. Exploring a linked network with crisp valued criteria is generally considered inflexible comparing to the use of linguistic descriptors. Particularly, a misinterpretation of a link measure may happen if there exists an unduly high property value within a linked network. A more accurate and natural measure is to exploit qualitative labels.

To tackle such a crucial drawback, an advanced approach has been proposed in which a link measure is gauged in accordance with its specific FOM spaces [157, 158], an extension of the crisp-interval Order-of-Magnitude space [159]. Let $FOM(\pi) = (L(\pi), F(\pi))$ be an FOM space of the link measure π , and $L(\pi)$ and $F(\pi)$ each represents a set of qualitative labels and a set of fuzzy sets which define such labels. Note that the development of a qualitative reasoner usually involves aggregating a number of different link measures, with each link measure presenting a particular feature of the discussed link. And these features values generated by various link measure are often represented with qualitative labels of different granularity, defined on dissimilar universes of discourse. Therefore, prior to the aggregation process, the homogenization process performed on the traditional OM model is similarly required to map the fuzzy-set based feature variables onto the unified scale $U = [0, 1]$.

Assume that m distinct measures are adopted to perform link analysis, each carries a qualitative variable, denoted as V_i ($1 \leq i \leq m$), with a corresponding weight, denoted as W_i

($1 \leq i \leq m$). Following the pre-processing procedure, the aggregated outcome ϕ (also a fuzzy set ranging in $U = [0, 1]$) can be estimated through the weighted average function φ :

$$\phi = \varphi(V_1, \dots, V_m, W_1, \dots, W(m)) = \frac{V_1 W_1 + \dots + V_m W_m}{W_1 + \dots + W_m} \quad (2.38)$$

The membership function of ϕ is represented by $\mu_\phi(t)$, where t is an ordinary weighted average which can be computed as:

$$t = \varphi(x_1, \dots, x_m, w_1, \dots, w_m) = \frac{x_1 w_1 + \dots + x_m w_m}{w_1 + \dots + w_m} \quad (2.39)$$

where $x_i \in V_i$ and $w_i \in W_i$ ($1 \leq i \leq m$). According to the extension principle, the membership function of ϕ is:

$$\mu_\phi(t) = \sup(\min(\mu_{V_1}(x_1)), \mu_{W_1}(w_1)), \dots, \min(\mu_{V_m}(x_m)), \mu_{W_m}(w_m)) \quad (2.40)$$

It is worth mentioning that calculating the exact membership function $\mu_\phi(t)$ is computationally expensive. Thus, a discrete approximation by α -cut fuzzy arithmetic is used to perform the fuzzy set aggregation step [160]. Specifically, the α -cut of a variable V_i and its corresponding weight W_i ($1 \leq i \leq m$) are denoted as:

$$(V_i)_\alpha = \{(x_i, \mu_{V_i}(x_i)) | x_i \in V_i, \mu_{V_i}(x_i) \geq \alpha\} \quad (2.41)$$

$$(W_i)_\alpha = \{(w_i, \mu_{W_i}(w_i)) | w_i \in W_i, \mu_{W_i}(w_i) \geq \alpha\} \quad (2.42)$$

Note that the α -cuts are crisp intervals which can be expressed in continuous closed form:

$$(V_i)_\alpha = [(a_i)_\alpha, (b_i)_\alpha] = \left[\min\{x_i \in V_i | \mu_{V_i}(x_i) \geq \alpha\}, \max\{x_i \in V_i | \mu_{V_i}(x_i) \geq \alpha\} \right] \quad (2.43)$$

$$(W_i)_\alpha = [(c_i)_\alpha, (d_i)_\alpha] = \left[\min\{w_i \in W_i | \mu_{W_i}(w_i) \geq \alpha\}, \max\{w_i \in W_i | \mu_{W_i}(w_i) \geq \alpha\} \right] \quad (2.44)$$

where $(a_i)_\alpha$ and $(c_i)_\alpha$ denote the left endpoints of $(V_i)_\alpha$ and $(W_i)_\alpha$, respectively. Similarly, $(b_i)_\alpha$ and $(d_i)_\alpha$ denote the right endpoints of $(V_i)_\alpha$ and $(W_i)_\alpha$, respectively.

Thus, the α -cut of ϕ , denoted as $(\phi)_\alpha$ can be acquired in a way that

$$(\phi)_\alpha = \left[\min \phi(x_1, \dots, x_m, w_1, \dots, w_m), \max \phi(x_1, \dots, x_m, w_1, \dots, w_m) \right] \quad (2.45)$$

where $\forall \alpha \in (0, 1]$, $(a_i)_\alpha \leq x_i \leq (b_i)_\alpha$, $(c_i)_\alpha \leq w_i \leq (d_i)_\alpha$, $a_i, x_i, b_i \in V_i$ and $c_i, w_i, d_i \in W_i$ ($1 \leq i \leq m$). Owing to the monotonicity of ϕ , Eq. 2.45 can be simplified as:

$$(\phi)_\alpha = \left[\min_{w_i \in \{(c_i)_\alpha, (d_i)_\alpha\}} f_L(w_1, \dots, w_m), \max_{w_i \in \{(c_i)_\alpha, (d_i)_\alpha\}} f_R(w_1, \dots, w_m) \right] \quad (2.46)$$

where $f_L(w_1, \dots, w_m)$ and $f_R(w_1, \dots, w_m)$ representing the left and right point of the value interval are respectively defined as:

$$f_L(w_1, \dots, w_m) = f((a_1)_\alpha, \dots, (a_m)_\alpha, w_1, \dots, w_m) = \frac{(a_1)_\alpha w_1 + \dots + (a_m)_\alpha w_m}{w_1 + \dots + w_m} \quad (2.47)$$

$$f_R(w_1, \dots, w_m) = f((b_1)_\alpha, \dots, (b_m)_\alpha, w_1, \dots, w_m) = \frac{(b_1)_\alpha w_1 + \dots + (b_m)_\alpha w_m}{w_1 + \dots + w_m} \quad (2.48)$$

It has been demonstrated that the qualitative approach consistently outperforms traditional numerical methods over datasets in the field of terrorism detection and author collaboration [109]. However, a current restraint for the order-of-magnitude model is its initial setup requires expert directed specification of qualitative variables (i.e., link property measures and their relative weights for aggregation). Despite the fact that the performance of this predicting model is generally robust to different parameter settings [157], the automatic determination of the underlying qualitative definition through a data-driven process is worth investigating, when there are sufficient training data available.

2.4 Summary

This chapter has introduced a selection of clustering metrics in both forms of general partition and hierarchical configuration at first, followed by a systematic review on link prediction techniques. The underlying inspirations for link prediction span a wide range of areas, including intrinsic features, topological structure, probability analytics and fuzzy logic. In general, the contents summarised in this chapter provide the foundation for the subsequent theoretical framework.

Chapter 3

Clustering Embedded Linear Regression for Prediction

Prediction is one of the most typical tasks in the research fields of machine learning and data mining. Predicting a continuous numeric feature is generally known as regression among related fields. Currently, regression analysis is also regarded as a mathematical approach to differentiating influential variables from the ordinary counterparts and sorting out the manner in which those influential variables interact with each other. Although there exists a variety of types of regression analysing metrics, with each having its own importance on a specific condition where they are best suited to be applied, at their core they all focus on examining the influence of one or more independent variables on a single dependent variable. Simple multi-variable linear regression (SMLR), one of the conventional approaches for regression problem which requires no prior knowledge about the dependencies on feature variables, has demonstrated its effectiveness in various application [20, 161]. However, it has two significant deficiencies: (1) It is not capable of generating a satisfactory structure which can model randomly distributed data or data organised in multiple regular forms. (2) It is sensitive to noisy entities or outliers in the examined dataset. Having noticed the drawbacks hidden behind this well-known approach, a novel predicting framework which can better handle the above mentioned problems is proposed and discussed in this chapter.

The rest of this chapter is arranged as follows. Section 3.1 introduces the concept of the proposed model, with a brief description of its components. Section 3.2 specifies the implementation of the predicting model in detail. Section 3.3 shows a case study of applying the predicting model to a real-world scenario of estimating student academic performance. Section 3.4 summarises this chapter.

3.1 Conceptual Framework of Predicting System

The structure of the predicting system is shown in Fig. 3.1. The system comprises four distinct component subsystems, each of which implements the following functionalities, respectively: partition, classification, regression and estimation. These activities are integrated together to form the overall predicting model, whose implementation involves a 4-step computational algorithm:

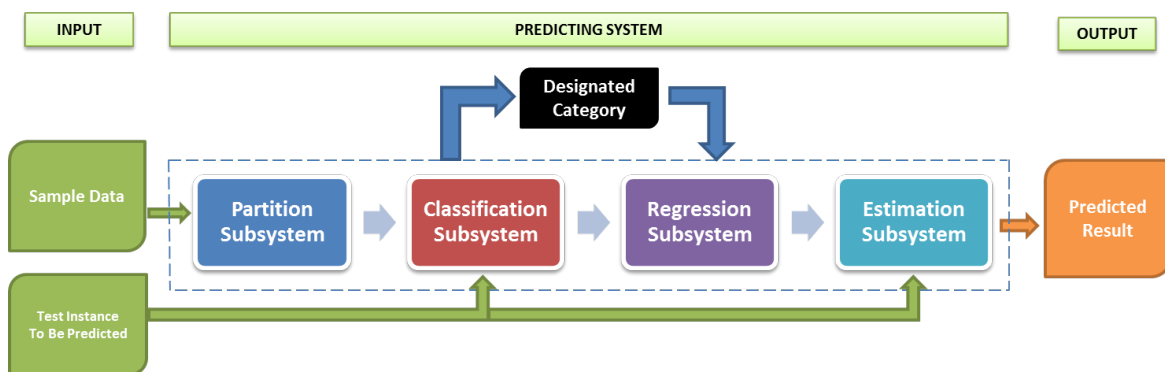


Fig. 3.1 Architecture of CELR Predicting System

- 1) **Partition Subsystem:** Partition sample data into different categories based on the similarities of their respective attribute (feature) values. This step results in sample data divided into a pre-defined number of groups, with data instances in the same group being deemed more similar to those in other groups.
- 2) **Classification Subsystem:** Identify the category to which a testing instance belongs according to the similarity of its attribute values to those of the sample data. This step assigns the testing instance to one of the categories (groups) generated by Step 1.
- 3) **Regression Subsystem:** For the designated category obtained by Step 2, determine the relationship between a dependent feature variable of interest and the independent attributes based on the sample data within the target category.
- 4) **Estimation Subsystem:** For a target testing instance, calculate the predicted result on its missing feature value according to the observed independent feature values and the relationship formula determined in Step 3.

3.2 Implementation of Predicting System

This section describes the implementation of the proposed framework for prediction. Note that the conceptual framework proposed in Section 3.1 can be implemented via various combinations of approaches for each of the component subsystems. However, in this section, one of the most efficient implementations is demonstrated.

3.2.1 Partition Subsystem

The aim of the partition subsystem is to divide the instances in training dataset into different categories according to their respective attribute (feature) vector. Those training instances with similar attributes values are partitioned into the same group. To implement such a partitioning task, k -means clustering [79], one of the simplest unsupervised learning algorithms [162] is employed. It works by grouping a set of instances in a way such that instances in the same cluster are more similar to each other than to those in other clusters. The parameter k ($k \leq n$) here, representing the number of partitions, together with the initial centroid of each cluster, can be either given by domain experts, or generated through automatic steps [43, 163].

In the process of clustering, the Euclidean distance between each instance and the centroid of each cluster is calculated to determine which cluster the instance belongs to. The pseudo-code for the implementation of this simple k -means clustering is shown in Algorithm 4.

Algorithm 4: *k*-means Clustering Algorithm**Input:***k*: Number of clusters $C = \{c_1, c_2, \dots, c_k\}$: *k* centroids of *k* clusters*n*: Number of instances in the training dataset $S = \{s_1, s_2, \dots, s_n\}$: A training dataset of *n* instances

```

1 repeat
2   foreach  $s_i (1 \leq i \leq n) \in S$  do
3     foreach  $c_j (1 \leq j \leq k) \in C$  do
4       COMPUTE Euclidean distance  $d(s_i, c_j)$  between  $s_i$  and  $c_j$ ;
5     end
6      $s_i \rightarrow \{Cluster_j | \arg \min_{c_j \in C} d(s_i, c_j)\}$ ;
7     foreach  $c_j (1 \leq j \leq k) \in C$  do
8        $c_j \leftarrow$  average of  $\{s_i | s_i \in Cluster_j\}$ ;
9     end
10  end
11 until none of  $c_j$  in  $C$  changed;
Output: A set of Clusters

```

Note that if different weights are imposed upon each of the attribute variables, then the above algorithm will need to take the weighting scheme into consideration. This is however, straightforward due to the fact that only linear rescaling of the component contributions need to be calculated.

The time complexity of the *k*-means algorithm here is $O(nkl)$, where *n* is the number of instances in the training dataset, *k* is the number of clusters, and *l* is the number of iterations taken by the algorithm to converge. Usually, *k* and *l* are fixed in advance. So the algorithm has a linear time complexity with respect to the size of the training dataset $O(n)$ [162].

3.2.2 Classification Subsystem

The general objective of classification is, through supervised learning, to identify which class a new or unseen observation belongs to, on the basis of a training dataset containing instances whose class labels are known [164]. This is also the case for the present subsystem. In particular, for predicting a feature value for an object, when a sample of (other) instances are available, the task can be guided by determining the position of this object with regard to

any clusters obtained by the above clustering method.

It is natural to believe that a group of objects with similar features values may belong to the same category. From this observation, the K -Nearest-Neighbours (KNN) classifier [165] is employed to perform the required classification job. KNN is chosen since it is one of the simplest classification methods and works well when there is little or no prior knowledge about the distribution of the domain data. For the ease of description, the clusters generated by the K -means clustering method are termed as classes here.

K -Nearest-Neighbours (KNN) classifier [165] is employed to perform the required classification. KNN is chosen since it is one of the simplest classification methods and works well when there is little or no prior knowledge about the distribution of the domain data. For the ease of description, the clusters generated by the k -means clustering method are termed as classes here.

To classify an unlabelled instance, the K -nearest instances of it are selected to vote for which class it belongs to. The K -nearest instances refer to those classified records whose distances to the unlabelled instance are measured to be the K shortest ones (though these distances do not have to be equal). The simplest way to find the nearest neighbours of an instance is to compute its Euclidean distance to all the training examples. The pseudo-code of kNN classification is shown in Algorithm 5.

Algorithm 5: KNN Classification Algorithm

Input:

K : Number of nearest neighbours

n : Number of instances in the training dataset

$S = \{s_1, s_2, \dots, s_n\}$: A training dataset of n instances

$s_p (s_p \notin S)$: Instance to be predicted

1 **foreach** $s_i (1 \leq i \leq n) \in S$ **do**

2 | COMPUTE Euclidean distance $d(s_p, s_i)$ between s_p and s_i ;

3 **end**

4 **ORDER** $d(s_p, s_i), (1 \leq i \leq n)$ from smallest to largest and **CREATE** a list L ;

5 **ASSIGN** s_p to the most frequent class which the first K instances in L belong to;

Output: A cluster which s_p belongs to

The implementation of this KNN classification method is straightforward, with a test time complexity of $O(n + Kn)$ [166], where n is the size of the training dataset, and K is the number of selected nearest neighbours.

3.2.3 Regression Subsystem

Regression analysis is a popular statistical process for estimating the relationships among continuous numeric variables, which has been widely applied [20]. Assume that there is no prior knowledge about the variable dependency available (which is usually the case when exploring a new research field), multi-variable linear regression, a highly flexible mechanism for examining the relationship of a collection of independent variables to a single dependent variable [167], is an appropriate choice to perform the final prediction.

The basic idea of this linear regression model is: given a dataset $\{y_i, x_{i1}, x_{i2}, \dots, x_{im}\}$, ($i = 1, \dots, n$) of n instances, the relationship between the dependent variable y_i and the m independent variables x_{ij} ($1 \leq j \leq m$) is assumed to be linear. That is, the underlying relationship amongst all the variables takes the form of

$$y_i = \alpha_0 + \alpha_1 x_{i1} + \alpha_2 x_{i2} + \dots + \alpha_m x_{im} \quad (3.1)$$

where $\alpha = [\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_m]$ is the so-called regression coefficient vector.

Let $\hat{1}$ denote the unit value vector (of an n dimensionality) and

$$X = \begin{bmatrix} \hat{1} & x_1 & x_2 & \dots & x_m \end{bmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1m} \\ 1 & x_{21} & x_{22} & \dots & x_{2m} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nm} \end{pmatrix} \quad (3.2)$$

Then, the multi-variable linear regression to estimate the predicted dependent feature variable is of the following equation form:

$$y = X\alpha \quad (3.3)$$

where $y = [y_1, y_2, \dots, y_n]^T$ is a vector of dependent feature values for the training sample data which is already known.

In order to determine the value of α_j ($0 \leq j \leq m$), conventional least squares (LS) estimator is adopted here owing to its computational simplicity. The LS method minimises

the sum of squared residuals, and leads to a closed-form expression for estimating the unknown vector α :

$$\alpha = (X^T X)^{-1} X^T y \quad (3.4)$$

The complexity of the LS algorithm for multi-variable linear regression is $O(m^2n)$, where m is the number of independent variables and n is the number of instances in the training dataset. Since m is usually fixed and known in advance, if m is much smaller than n , the asymptotic time complexity is $O(n)$.

3.2.4 Estimation Subsystem

This subsystem implements a straightforward, and final step of the entire computation process of the integrated system. From the resulting parameter estimation for α , given the independent feature vector F for a testing instance t , where $F^t = [f_1^t, f_2^t, \dots, f_m^t]$, the dependent feature value e_p for prediction can be calculated as

$$e_p = \alpha_0 + f_1^t \alpha_1 + f_2^t \alpha_2 + \dots + f_m^t \alpha_m \quad (3.5)$$

Thus, the predicted feature value of the target instance is obtained. Likewise, for cases where predicting dependant feature value for a number of instances, the task can also be implemented through this method. Simply, the time complexity of the estimation method is $O(1)$ for one predicted instance, and $O(n)$ for n instances.

3.3 Case Study: Prediction on Student Academic Performance

The prediction of students academic performance plays an important role in educational institutions, especially for higher educational institutions such as universities and colleges. For example, when reviewing applications from prospective students, the prediction may help universities to find candidates who are eligible for a particular academic program and identify those applicants who are likely to perform well in the university [168]. The prediction of academic performance can also help students and their referees to estimate what kind of higher educational institutions they may be qualified for and what type of academic major may be suitable for them.

3.3 Case Study: Prediction on Student Academic Performance

Furthermore, the results obtained from such prediction for students already at a university may be used for classifying students, thereby enabling the educational institution to provide them with additional support such as customised personal assistance and tutoring resources. The results of prediction can also be used by lecturers as well to specify the most appropriate teaching materials and actions for each group of students to meet their needs. Thus, developing a prediction tool is very important for educational institutions.

Predicting student exam scores is a section of estimating student overall academic performance, and is of great value to study. In general, for a typical academic module, students' performance is usually assessed by assignments, class tests and a final exam. The integrated overall evaluation for a student's performance on a module is typically based on a certain combination of the outcomes of these aspects.

Unfortunately, there are occasions that a student fails to attend an exam due to inescapable reasons, such as physical injury or other medical situations. There may even be cases where a number of students in the same module fail to attend the exam because of unavoidable weather conditions or a natural disaster. Thus, if something happens like this, how to evaluate their performance in the exam becomes a problem. Perhaps rearranging another exam could be a way forward, but this may carry significant disadvantages, including but not being limited to the following: (1) Unfairness - The questions in the rearranged exam may not be precisely in line with the previous one in difficulty or complexity level, since the questions in the two exams cannot be exactly the same, causing unfairness when ranking student in their academic performance. (2) Costs - Once another exam needs to be organised, new exam paper needs to be set and vetted, the calendar and the exam rooms need to be rescheduled, and the invigilators need to be arranged and trained, increasing the workload and overheads significantly. (3) Disruption - The rearranged exam might produce contradiction to the designated teaching plan, especially when decisions must be made on the basis of the exam outcomes in order for normal teaching activities to be carried on continuously. For any of these cases, successfully predicting student exam scores and using them as evidence to evaluate student academic performance offers great potential benefits.

Over the past decades, methods for data mining and machine learning have been applied to commercial business analysis and prediction successfully. However, their applications in education field is still at a coarse level. Recently, Oladokun [169] employed artificial neural networks to predicting student academic performance in an engineering course. Chen [168] introduced an alternative for training neural networks in an effort to predict student academic

performance. Hidayah [170] developed a neuro fuzzy approach for classifying students through academic performance prediction in a conventional classroom context. Fire [171] made an attempt to predict student exam scores by analysing social network data between students.

Previous methods for prediction have to an extent neglected intuitive results achieved by students in given academic modules, such as assignment marks and class test marks. This section presents a novel approach to predicting student exam scores based on CELR, and using basic attributes which are directly related to the teaching of the academic module concerned.

3.3.1 Data Preparation

Originally, two small datasets (SAP50A and SAP50B [172]) are used as examples and in the form of numerical crisp scores between 0 and 100, which is one of the most popular ways to measure student academic performance. The objective of experiment with SAP50A is to provide evidence that the proposed method can produce acceptable result on an ideal and representative dataset. However, the reason to select two types of training dataset for experiment is that the students performance in an assessment component does not always distribute normally, thus SAP50B, a dataset in which data are distributed rather randomly, is selected to conduct another experiment. The distribution scores of the assignment, test and final exam in SAP50B are shown in Fig.3.2, Fig. 3.3 and fig.3.4 [172]. It is worth mentioning that each of the datasets consists of 50 instances, involving three conditional attributes: assignment score, test score and final exam score, accompanied with five possible classification outcomes: “Unsatisfactory”, “Satisfactory”, “Average”, “Good” and “Excellent”. Since partitioning 50 instances into five classes may possibly result in a class containing very few instances, which might probably cause heavy bias to make prediction, based on experts opinion, the five classes are merged into three, with “Satisfactory” and “Average” combined to “new average”, and “Good” and “Excellent” combined to “new good”. For simplicity, the grades of “Unsatisfactory”, “new average” and “new good” are denoted as C, B, A respectively.

3.3 Case Study: Prediction on Student Academic Performance

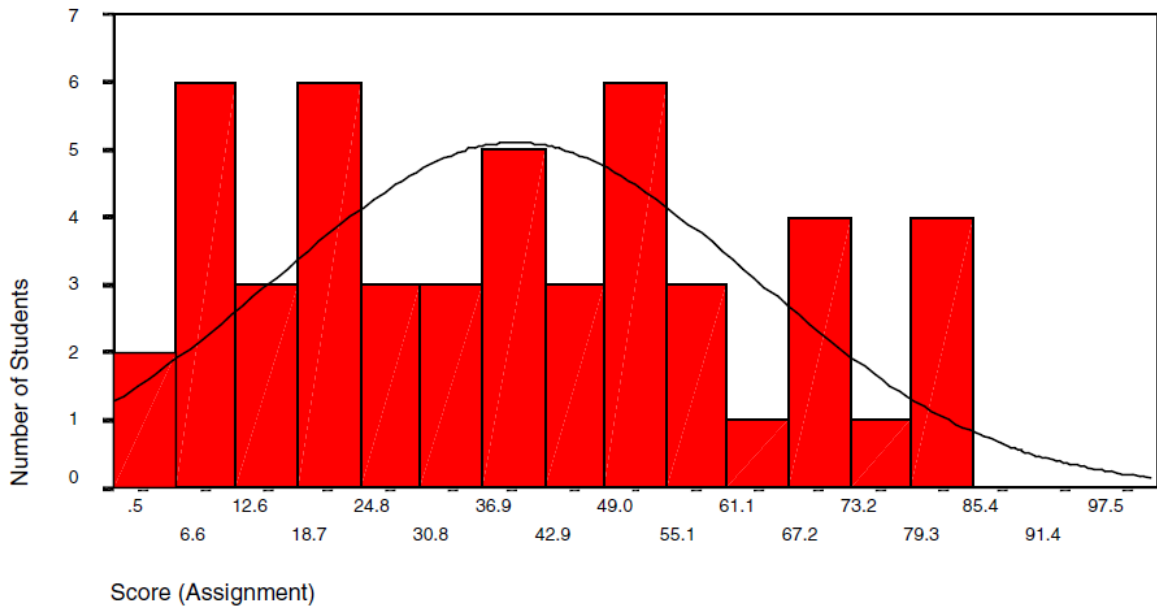


Fig. 3.2 Distribution of Assignment Scores

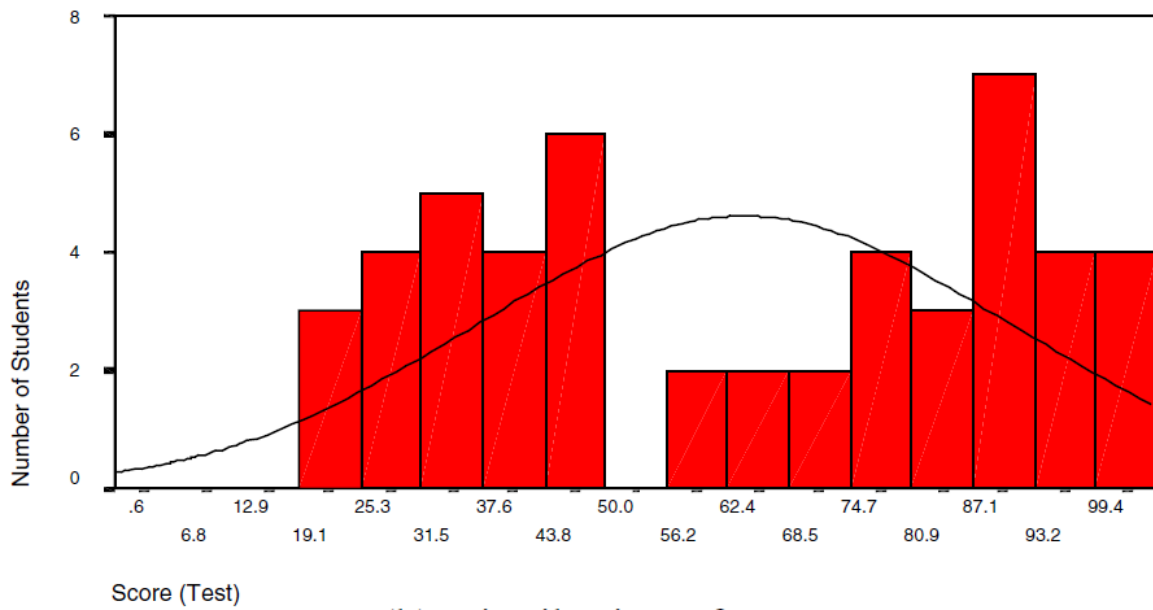


Fig. 3.3 Distribution of Test Scores

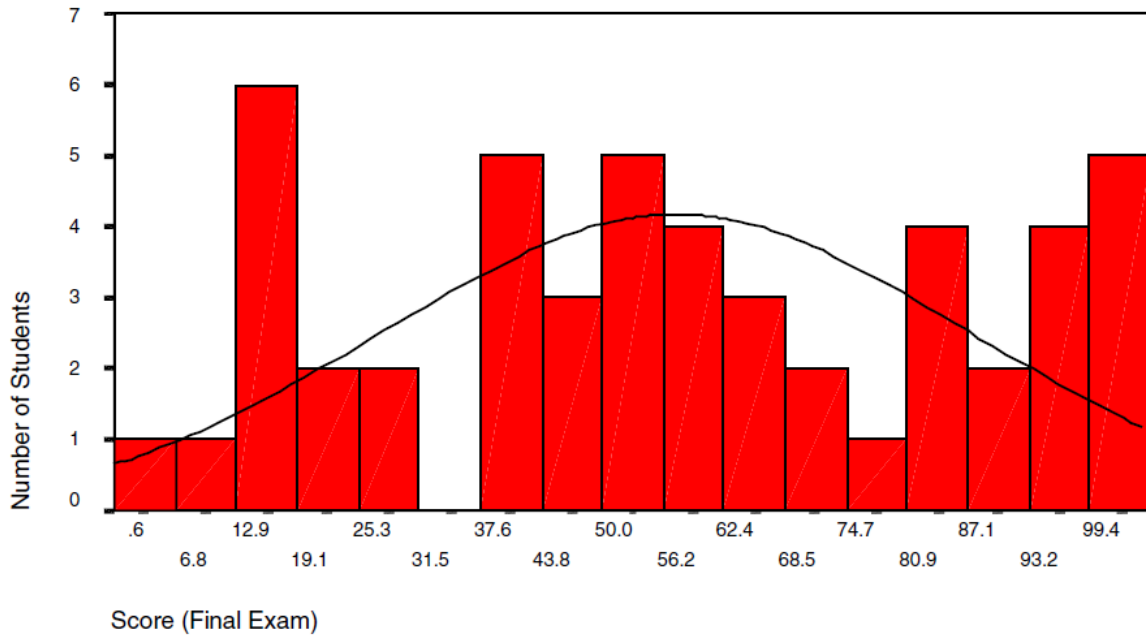


Fig. 3.4 Distribution of Final Exam Scores

Another four datasets (grouped and named as SAP-PLUS), each containing hundreds of instances collected from two Portuguese schools (GP and MS) about their students in Maths and Portuguese language study, are used to test the proposed predicting system as well [173]. As with many European countries (e.g. France, Portugal), a 20-point grading scale is used to evaluate student academic performance, where 0 is the lowest grade and 20 is the perfect score. During the school year, students are evaluated in three periods and the last evaluation (G3 in Table 3.1) corresponds to the final period grade. In the complete dataset, more than 30 attributes with related data to each attribute are collected. For simplicity and clarity, continuous numeric attributes which are closely related to the student academic performance and the study behaviour based on expert’s opinion are selected to conduct the experiment. The selected attributes are shown in Table 3.1.

Table 3.1 Selected Attributes for Student Academic Performance

Attribute	Description
study-time	weekly study time (numeric)
failure-count	number of failures in the past for this academic module (numeric: integer)
absence	days of school absence (numeric)
G1	first period grade (numeric: from 0 to 20)
G2	second period grade (numeric: from 0 to 20)
G3	final period grade (numeric: from 0 to 20)

Note that the values of the selected attributes are almost within the same range (in the same order of magnitude), thus no requirement about data preprocessing is needed prior to performing the proposed predicting model.

3.3.2 Experimental Setup

Taking Dataset SAP50A and SAP50B as example, for the task of predicting student academic performance, its involving procedure is shown in Fig. 3.5.

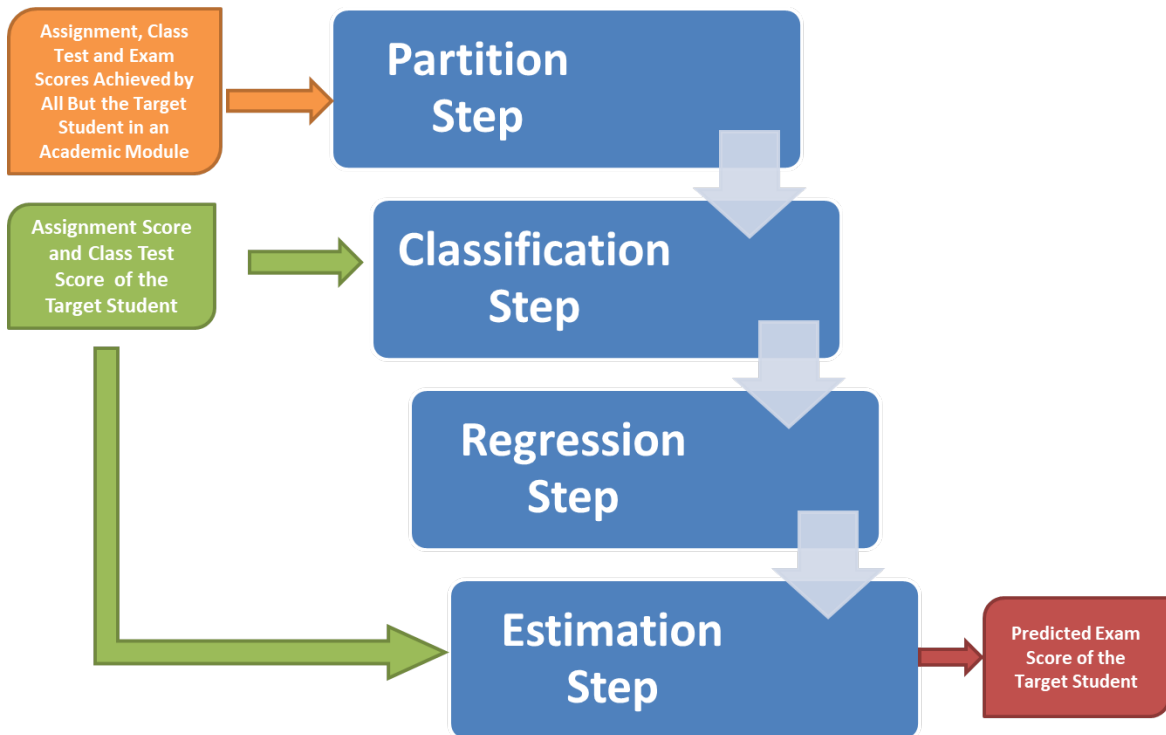


Fig. 3.5 Process of CELR for Predicting Student Academic Performance

For experiments on datasets SAP50A and SAP50B, each dataset is split into subsets for 3-fold cross validation (3-FCV)[174]. The reported results are based on an average of 10 times of the 3-FCV. For partition step, in the process of K -means clustering, two different ways to determine initial centroids are given to show the results. One is based on experts opinion, and the other is by statistical computation, which will be discussed in Section 3.3.2.1. For experiments on SAP-PLUS, a alternatively, an automatic approach through statistical computation is applied to generate the initial centroids the partition step. And 10 runs of 10-fold cross validation (10-FCV) are conducted to present the experimental result, since the size of the examined datasets are much larger than SAP50A and SAP50B. In the step of classification, the number of nearest neighbours K is odd and set from 3 to 7 over

different runs. In reality, it has a natural appeal to assume that a group of students with similar academic records on a module may have a similar ability amongst each other in knowledge comprehension and application and hence, achieve similar result in the final exam. This provides an intuitive approach to estimating the likely exam score of a student (who missed the exam due to good reasons) using corresponding features of his/her peers of the same intellectual ability. To validate the significance of the experimental result, paired *t*-tests are carried out. The baseline for comparison is the result of running SMLR.

3.3.2.1 Methods for Partition

- **Partition based on experts opinion:** The classification of the grades in the experiment for SAP50A and SAP50B is based on an interval that refers to the level of performance given by experts as shown in Table 3.2, Table 3.3 and Table 3.4.

Table 3.2 Assignment Scores with Their Associated Level of Achievement and Grades (class)

Assignment Score	Level of achievement	Grade (Class)
0-15	Unsatisfactory	C
16-45	average	B
46-100	good	A

Table 3.3 Class Test Scores with Their Associated Level of Achievement and Grades (class)

Class test Score	Level of achievement	Grade (Class)
0-35	Unsatisfactory	C
36-65	average	B
66-100	good	A

Table 3.4 Exam Scores with Their Associated Level of Achievement and Grades (class)

Exam Score	Level of achievement	Grade (Class)
0-25	Unsatisfactory	C
26-45	average	B
46-100	good	A

From Table 3.2 to Table 3.4, it is not difficult to identify the mid-grade point for each kind of examination method by $[Score_{(min)}+Score_{(max)}]/2$. For instance, the mid-grade point of grade C for student assignment is calculated as $(0 + 15)/2 = 7.5$, the mid-grade

point of grade B for student assignment is calculated as $(16 + 45)/2 = 30.5$, and the mid-grade point of grade A for student assignment is calculated as $(46 + 100)/2 = 73$. Similarly, the mid-grade point of each grade for class test and exam can be calculated. Thus, a matrix M containing mid-grade points of different grades for assignment, class test and exam is obtained, where

$$M = \begin{bmatrix} 7.5 & 30.5 & 73 \\ 17.5 & 50.5 & 83 \\ 12.5 & 35.5 & 73 \end{bmatrix} \quad (3.6)$$

Each column vector of M can be employed as a centroid for clustering in the partition subsystem.

For data corpus SAP-PLUS, the original records of student academic grades can be transformed into European Credit Transfer System (ECTS) Grades, according to Erasmus grade conversion system – a European programme that enables students exchange in 31 countries [173], The details of the transforming rule are listed in Table 3.5.

Table 3.5 ECTS Grades with Corresponding Portugal/France Grades

ECTS Grades	Portugal/France Grades
Excellent (A)	16-20
Good (B)	14-15
Satisfactory (C)	12-13
Sufficient (D)	10-11
Fail (E)	0-10

Similarly, the expertise based centroid determination method can be applied to the datasets involved in SAP-PLUS.

- **Partition based on computation:** Different from applying initial centroids provided by lecturers or domain experts for clustering in the partition subsystem, another method to determine initial centroids based on computation is described as follows (taking dataset SAP50A and SAP50B as example):

- 1) Calculate the arithmetic average avg_i ($1 \leq i \leq n$) of the assignment score a_i , class test score t_i and exam score e_i for each instance in the training dataset.
- 2) Sort the instances in the training dataset according to their arithmetic averages.
- 3) Divide all the instances into K clusters evenly based on their arithmetic averages after sorting, and name the results $Cluster_j$ ($1 \leq j \leq K$).

- 4) Calculate the arithmetic averages of assignment scores $avg_j^{(a)}$, class test scores $avg_j^{(t)}$ and exam scores $avg_j^{(e)}$ within each cluster $Cluster_j$ ($1 \leq j \leq K$), and assign the results to be the initial centroid of each cluster.

Analogously, for dataset corpus SAP-PLUS, such initialising steps are applied to the previous numeric academic records. However, in Step 4), average value of features including study behaviour indicators also needs to be calculated to form the initial centroid.

For the experiment, in order to be consistent with the expertise based approach, the number of clusters generated from such automatic method is determined to be equal to the manual defined counterpart. Advanced techniques capable of determining the number of clusters intelligently will be discussed in Section 8.

3.3.3 Experimental Result

The experimental result are illustrated by three different indicators:

1. Mean absolute error of prediction, which provides a general view for the estimating precision.
2. Average percentage of predicted scores closest to ground truth, which comparatively reveals the predicting accuracy amongst different methods.
3. Average percentage of estimation error within a certain numeric range, focusing on the tolerance of an approach for its error space.

These indicators each presenting a particular feature of the metrics and are jointly considered to disclose the general efficacy of the predicting models under examination.

In the partition subsystem of the predicting framework, basic sorting algorithm and statistical computation are used to determine the initial centroids of each cluster. In the process of implementing cross validation, the initial centroids for clustering are changed each time with the alteration of training and testing dataset, which makes difference to the fixed initial centroids given by lecturers or domain experts. For the classification subsystem, in same with the previous experiment, odd number of nearest neighbours K is set from 3 to 7 over different runs. The above mentioned statistical indicators obtained from experimental results are listed in Table 3.6.

Table 3.6 Comparison of Statistical Indicators in Predicated Exam Scores Obtained by CELR and SMLR on SAP50A

Indicators	CELR(E)			CELR(A)			SMLR
	$K = 3$	$K = 5$	$K = 7$	$K = 3$	$K = 5$	$K = 7$	
Mean absolute error of prediction	7.82 ± 1.12 (v)	8.74 ± 1.03 (v)	8.92 ± 1.36	7.59 ± 1.24 (v)	8.51 ± 0.97 (v)	8.68 ± 1.25	9.31 ± 1.74
Average percentage of predicted scores closest to ground truth	25.69 ± 3.53 (v)	13.73 ± 3.99 (v)	6.43 ± 2.14	29.25 ± 3.93 (v)	13.13 ± 3.72 (v)	6.13 ± 2.21	5.66 ± 1.08
Average percentage of estimation error within 10 points (%)	82.35 ± 4.89 (v)	62.94 ± 6.58	61.96 ± 8.56	81.57 ± 5.06	63.53 ± 6.31	61.76 ± 8.14	60.12 ± 9.32
Average percentage of estimation error within 5 points (%)	47.05 ± 6.51 (v)	38.62 ± 7.29	36.67 ± 7.83	48.62 ± 6.17	39.41 ± 7.10	36.25 ± 7.62	36.02 ± 8.59

¹. (E): Expert determination of initial centroids.

². (A): Automatic determination of initial centroids.

³. K : Number of nearest neighbours selected.

⁴. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

It is obvious that in Table 3.6, the CELR with 3 nearest neighbours for classification performs best among all the listed approaches, with the smallest absolute mean error of prediction 7.59 and 7.82 for CELR(A) and CELR(E) respectively, and the largest numbers of predicted scores which are closest to the ground truths. The proposed method with 5 and 7 nearest neighbours for classification performs poorer than the previous one but better than SMLR. The reason for their worse performance to 3NN classification is due to the vote for some instances which share similarities to both classes. Similar results are also illustrated for dataset SAP50B, which are shown in Table 3.7.

Table 3.7 Comparison of Statistical Indicators in Predicated Exam Scores Obtained by CELR and SMLR on SAP50B

Indicators	CELR(E)			CELR(A)			SMLR
	$K = 3$	$K = 5$	$K = 7$	$K = 3$	$K = 5$	$K = 7$	
Mean absolute error of prediction	10.29 ± 2.14 (v)	12.27 ± 2.03	12.92 ± 2.36	10.02 ± 1.94 (v)	11.71 ± 2.37 (v)	13.01 ± 2.25	13.61 ± 2.74
Average percentage of predicted scores closest to ground truth	18.76 ± 3.42 (v)	13.73 ± 4.24	11.43 ± 3.88	20.82 ± 4.83 (v)	11.13 ± 3.67	12.24 ± 3.34	12.38 ± 4.09
Average percentage of estimation error within 10 points (%)	71.23 ± 5.29 (v)	60.91 ± 7.58	62.24 ± 7.21	69.45 ± 7.28 (v)	63.89 ± 8.11	62.54 ± 6.16	62.82 ± 7.17
Average percentage of estimation error within 5 points (%)	32.45 ± 5.48 (v)	29.44 ± 5.92	28.62 ± 6.25	34.62 ± 7.12	30.05 ± 8.19	30.89 ± 7.24	28.78 ± 8.23

¹. (E): Expert determination of initial centroids.

². (A): Automatic determination of initial centroids.

³. K : Number of nearest neighbours selected.

⁴. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

Note that SAP50B is a different dataset from SAP50A in terms of data distribution. This is to reflect the fact that students performance in an assessment component does not always distribute normally. Intuitively, the performance of the proposed method to dataset SAP50B seems not as good as it performed to dataset SAP50A. This is due to the fact that some of the instances in SAP50B are too extreme to be fitted by regression equation, which leads to heavy bias in estimation. However, the overall performance of the proposed method is still better than SMLR, which demonstrate that the pre-handling of data before applying regression model is significantly valuable.

The same training and testing steps are carried out to dataset corpus SAP-PLUS. As mentioned in Section 3.3.1, SAP-PLUS contains four datasets from two schools, denoted as Maths(GP), Portuguese(GP), Maths(MS) and Portuguese(MS), respectively. Note that for the current circumstance, the predicted results are within a range of $[0, 20]$, thus, the indicator of average percentage of estimation error within a specific region needs to be adjusted accordingly. Here, 3 and 1 are adopted respectively to replace the previous testing setup of 10 and 5 for experimentation on SAP50A and SAP50B. The experimental results on these four datasets are presented in Table 3.8 to 3.11.

Unsurprisingly, CELR again outperforms SMLR significantly with regard to most of the statistical indicators and generates predicting results which are closer to the ground truths. Moreover, in terms of some statistical indicators, such as Mean absolute error of prediction and average number of estimation error within a particular range, the proposed approach with computational method to determine initial centroids for clustering even performs better than the method with initial centroids given by lecturers or domain experts, which shows the advantage of computational intelligence in the proposed framework.

Table 3.8 Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Maths(GP)

Indicators	CELR(E)			CELR(A)			SMLR
	$K = 3$	$K = 5$	$K = 7$	$K = 3$	$K = 5$	$K = 7$	
Mean absolute error of prediction	1.21 ± 0.17	1.28 ± 0.19	1.36 ± 0.16	1.17 ± 0.19 (v)	1.21 ± 0.27	1.31 ± 0.25	1.35 ± 0.21
Average percentage of predicted scores closest to ground truth	17.73 ± 3.21 (v)	14.93 ± 4.84 (v)	13.41 ± 5.08	18.25 ± 5.25 (v)	12.13 ± 4.72	12.12 ± 2.78	12.43 ± 4.08
Average percentage of estimation error within 3 points (%)	92.42 ± 3.31	92.76 ± 3.14	92.57 ± 3.15	93.87 ± 3.23	93.48 ± 3.49	92.77 ± 3.56	92.19 ± 3.26
Average percentage of estimation error within 1 points (%)	50.11 ± 1.16	51.22 ± 1.10	50.25 ± 1.12	52.05 ± 1.18 (v)	51.41 ± 1.25	51.66 ± 1.29 (v)	49.32 ± 1.18

¹. (E): Expert determination of initial centroids.

². (A): Automatic determination of initial centroids.

³. K : Number of nearest neighbours selected.

⁴. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

Table 3.9 Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Maths(MS)

Indicators	CELR(E)			CELR(A)			SMLR
	$K = 3$	$K = 5$	$K = 7$	$K = 3$	$K = 5$	$K = 7$	
Mean absolute error of prediction	1.16 ± 0.26	1.20 ± 0.21	1.21 ± 0.19	1.16 ± 0.22	1.16 ± 0.21	1.18 ± 0.23	1.21 ± 0.22
Average percentage of predicted scores closest to ground truth	16.25 ± 2.72 (v)	15.72 ± 5.44 (v)	11.71 ± 6.12 (*)	16.07 ± 5.16 (v)	14.13 ± 3.11	12.02 ± 4.78	14.04 ± 3.21
Average percentage of estimation error within 3 points (%)	94.12 ± 3.23	93.86 ± 3.11	93.01 ± 4.42	94.34 ± 4.01 (v)	94.01 ± 3.57	93.26 ± 3.64	92.19 ± 3.53
Average percentage of estimation error within 1 points (%)	68.02 ± 1.21	67.52 ± 1.62	67.40 ± 1.45	68.11 ± 1.16	67.22 ± 1.10	68.16 ± 1.32	67.38 ± 1.14

1. (E): Expert determination of initial centroids.

2. (A): Automatic determination of initial centroids.

3. K : Number of nearest neighbours selected.

4. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

Table 3.10 Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Portuguese(GP)

Indicators	CELR(E)			CELR(A)			SMLR
	K = 3	K = 5	K = 7	K = 3	K = 5	K = 7	
Mean absolute error of prediction	1.82 ± 0.19 (v)	1.86 ± 0.19	1.89 ± 0.18	1.82 ± 0.20 (v)	1.85 ± 0.21	1.88 ± 0.19	1.93 ± 0.20
Average percentage of predicted scores closest to ground truth	14.25 ± 3.43	8.28 ± 1.72 (*)	7.97 ± 1.78 (*)	22.63 ± 4.17 (v)	22.11 ± 4.84 (v)	12.41 ± 2.06	14.43 ± 4.08
Average percentage of estimation error within 3 points (%)	90.43 ± 1.23 (v)	90.11 ± 1.49	90.01 ± 1.56	90.57 ± 1.15 (v)	90.12 ± 1.74	89.78 ± 1.14	88.42 ± 1.26
Average percentage of estimation error within 1 points (%)	55.11 ± 1.36	55.26 ± 1.38	54.87 ± 1.41	56.98 ± 1.39 (v)	56.41 ± 1.41 (v)	56.72 ± 1.33 (v)	54.67 ± 1.36

¹. (E): Expert determination of initial centroids.

². (A): Automatic determination of initial centroids.

³. K: Number of nearest neighbours selected.

⁴. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

Table 3.11 Comparison of Statistical Indicators in Predicated Final Grade Obtained by CELR and SMLR on Portuguese(MS)

Indicators	CELR(E)			CELR(A)			SMLR
	$K = 3$	$K = 5$	$K = 7$	$K = 3$	$K = 5$	$K = 7$	
Mean absolute error of prediction	1.67 ± 0.15	1.66 ± 0.15	1.67 ± 0.16	1.64 ± 0.15 (v)	1.66 ± 0.16	1.68 ± 0.15	1.76 ± 0.16
Average percentage of predicted scores closest to ground truth	17.73 ± 1.84 (v)	17.26 ± 2.20 (v)	14.45 ± 1.86 (v)	18.23 ± 2.43 (v)	11.27 ± 1.72	10.78 ± 1.52	10.29 ± 2.08
Average percentage of estimation error within 3 points (%)	88.26 ± 1.76 (v)	88.02 ± 1.80 (v)	87.82 ± 1.76 (v)	88.87 ± 1.75 (v)	88.12 ± 1.74 (v)	88.03 ± 1.81 (v)	86.78 ± 1.96
Average percentage of estimation error within 1 points (%)	60.87 ± 1.46 (v)	60.21 ± 1.58 (v)	59.80 ± 1.62	62.12 ± 1.34 (v)	61.41 ± 1.55 (v)	60.29 ± 1.43 (v)	59.43 ± 1.61

1. (E): Expert determination of initial centroids.

2. (A): Automatic determination of initial centroids.

3. K : Number of nearest neighbours selected.

4. The best results are highlighted in boldface, with sign (*) / (v) indicates that the corresponding result is significantly worse / better than that of SMLR of 95% confidence interval.

3.4 Summary

This chapter has presented a new framework to predict continuous feature value which is missing. For the typical task of predicting continuous feature values with no prior knowledge about the predictor dependency available, linear regression analysis is universally regarded as one of the most effective methods. CELR is an eager learning approach (constructing models based on training instances to interpret the underlying data) [175], which improves the performance of conventional linear regression by partitioning training data into subspaces. It works on the assumption that all the feature variables examined as predictors are independent from each other. The proposed work performs the predicting task by employing simple clustering, classification and regression mechanisms within an integrated framework, which is convenient to implement. Time complexity for each of the involved in components of the predicting system is examined, revealing the efficiency for applying the model to various scenarios. A case study of comparative experimental investigations on prediction of student academic performance is carried out, demonstrating the potential of the proposed work in producing more intuitive and interpretable results. More in-depth discussions about the refinement of the predicting model are given in Chapter 8.

Chapter 4

Fuzzy Embedded Clustering Linear Regression for Prediction of Student Academic Performance

As described in Section 3.3, the prediction of student academic performance is important to both educational institutions and students themselves for a variety of reasons. However, previous techniques often consider only past numeric data for prediction, whereas others overuse different types of indicative attribute, leading to the creation of complicated predicting methods whose results are difficult to interpret. In a more broad sense of addressing the problem of predicting student academic performance, a number of proposed methods work based on the use of large quantities of previous exam results. For example, student performance in prior academic courses is used to predict their performance in a subsequent course [176]. It has been shown that previous success in high school mathematics and science has a positive correlation with the study of computer science at universities [177]. Also, high school performance and background in mathematics is utilised to predict final exam grades in an introductory computer science course [178]. Apart from previous academic records, different types of other attributes, including age and gender [179], educational level of the parents [180], emotional factors [181], social relationships [171], and even the complexity measure of lecture notes [182] have been taken into consideration in the existing work.

Whilst researching into relationships between student academic performance and a wide variety of individual attributes is meaningful and worthwhile, the overuse of different types of indicative attribute has led to the creation of complicated score predicting methods which may be hard to implement and whose results may be difficult to interpret. Certain types of attribute may not be easy to obtain during the normal teaching process. Moreover, previous methods

for prediction may be excessively focused on the relationship between student academic performance and a particular type of attribute, ignoring the fact that such performance is a synthesised consequence of many reasons. Having taken notice of this, a novel approach to predicting student academic performance proposed will be discussed in this chapter, based on the synthesis of just basic attributes that are related to the academic course and the students' normal study behaviour.

The remainder of this chapter is organised as follows. Section 4.1 outlines the proposed architecture for building the intelligent predicting system to predict student academic performance. Section 4.2 describes the functionality of each component within the system, and analyses their complexity. Section 4.3 shows the experimental evaluation, supported by comparative studies with the real grades and other methods of prediction. Finally, Section 4.4 concludes the chapter.

4.1 Structure of Predicting System

This section presents the proposed general framework to predict student academic performance, represented as final period grade. The structure of the proposed predicting system is shown in Fig. 4.1. Analogous to the predicting model discussed in Chapter 3, the conceptual framework for the novel predicting approach proposed here also comprises four distinct component subsystems, each of which implements the following functionalities, respectively: partition, regression, offset value generation and estimation. However, this advanced technique performs soft computation to categorise student records and make better use of global knowledge for prediction, while taking more necessary factors (student study behaviours) into account. These activities aforementioned are integrated together to form the overall student final period grade predicting mechanism, whose implementation involves a 4-step computational algorithm:

- 1) **Partition Subsystem:** Partition data of sample students (typically from previous years on the same course, whose final period grades are available) into different categories based on the similarities of their existing academic records (excluding their final period grade), and obtain the membership values for each of the sample students with regard to the partitions.
- 2) **Regression Subsystem:** Determine for each partition (category), the relationship between the final period grade and the previous records in the academic module concerned. Such a determination process works on each of the partitions obtained by

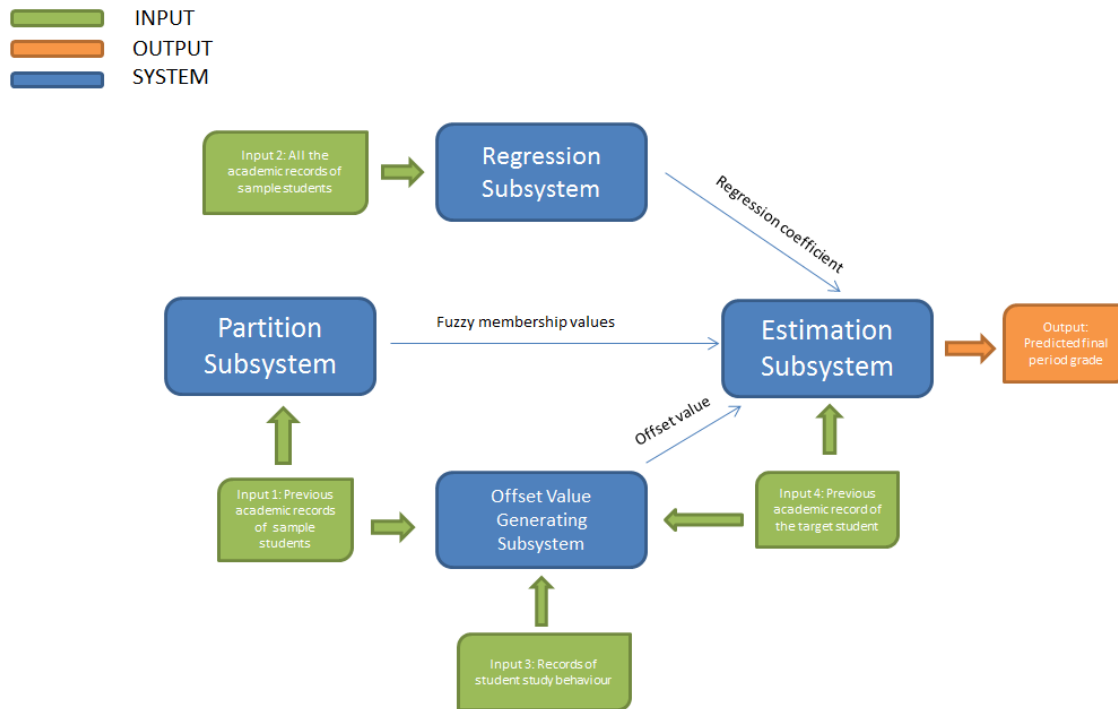


Fig. 4.1 Architecture of Final Period Grade Predicting System

Step 1, resulting in a formula which measures the correlation amongst the final period grade and the previous records of sample students for each of the partitions.

- 3) **Offset Value Generating Subsystem:** Generate an offset value of the predicted final period grade for the target student according to the similarity of the student's own normal study behaviour and the behaviour of other students with the same or similar previous academic records. This step performs the task of adjusting the predicted results by considering the record of student study behaviour.
- 4) **Estimation Subsystem:** Estimate the predicted final period grade of the target student based on the membership values obtained in Step 1, the relationship amongst final period grade and previous academic records determined by Step 2, and the offset value acquired from Step 3. Such an estimating process systematically considers the factors involved in previous steps, thus providing a comprehensive predicted result.

The working details of these subsystems are explained in Section 4.2.

4.2 Implementation of Predicting System

This section describes the implementation steps for the conceptual framework outlined in Section 4.1, with computational analysis for each of the subsystems being provided.

4.2.1 Partition Subsystem

Let an academic record list of n students be the input of the partition subsystem. In general, each record in the list, describing a student with their previous grade and final period grade (the grade values are in numeric form) for a given academic module, forms an instance of the training dataset.

The aim of the partition subsystem is to divide the instances regarding a certain module into different categories according to a formulaic synthesis of students' academic records over previous periods. Those students with similar previous academic records are partitioned into the same group. Hence, the outputs of the partition subsystem are denoted as groups of instances with similar academic records. However, in real situation, it may be difficult to distinguish exact groups to which the instances may belong in accordance to their formulaic synthesis of previous academic records, due to their relevance to different groups. Thus, fuzzy c-means clustering [183, 184], which has natural appeal to handle such uncertainty, is used to implement this subsystem. The resulting membership values for an instance to each group will play an important role in later steps.

The initial centroid of each cluster can be preset by lecturers (as domain experts), or generated through the following steps from the training samples:

- 1) Calculate the arithmetic average avg_i ($1 \leq i \leq n$) of the previous grades for each instance in the training dataset.
- 2) Sort the instances in the training dataset according to their arithmetic averages.
- 3) Divide all the instances into K clusters evenly based on their arithmetic averages after sorting, and name the results C_j ($1 \leq j \leq K$).
- 4) Calculate for each cluster C_j , the arithmetic average of the academic grades over each of the previous period, and assign the results as the initial centroid of the corresponding cluster.

In the process of clustering, the Euclidean distance between a newly given instance and the centroid of each cluster is calculated in order to determine which cluster the instance

belongs to. The pseudo code for the implementation of fuzzy c-means student clustering algorithm is shown in Algorithm 6.

Algorithm 6: Fuzzy C-means Clustering of Students

Input:

K : Number of clusters

$C = \{c_1, c_2, \dots, c_K\}$: Centroids of K clusters

m : Fuzzy partition matrix exponent, $m > 1$

n : Number of instances in the training dataset

$S = \{s_1, s_2, \dots, s_n\}$: training dataset of n student records

μ_{ij} : Degree of membership of s_i ($1 \leq i \leq n$) in j^{th} cluster ($1 \leq j \leq K$)

$J = \sum_{i=1}^n \sum_{j=1}^K \mu_{ij}^m \|s_i - c_j\|^2$: Objective function

ϵ : Specified minimum threshold between iterations

1 COMPUTE initial cluster membership values μ_{ij} by $\mu_{ij} = \frac{1}{\sum_{l=1}^K \left(\frac{\|s_i - c_j\|}{\|s_i - c_l\|}\right)^{\frac{2}{m-1}}}$, where

$\|*\|$ stands for Euclidean distance ;

2 **repeat**

3 COMPUTE cluster centers: $c_j = \frac{\sum_{i=1}^n \mu_{ij}^m s_i}{\sum_{i=1}^n \mu_{ij}^m}$;

4 UPDATE μ_{ij} according to: $\mu_{ij} = \frac{1}{\sum_{l=1}^K \left(\frac{\|s_i - c_j\|}{\|s_i - c_l\|}\right)^{\frac{2}{m-1}}}$;

5 COMPUTE objective function J ;

6 **until** J improves by less than ϵ ;

The time complexity of this algorithm is $O(ndKl)$ [185], where n represents the number of instances in the training dataset, d represents the dimension of the instance in the training dataset, K represents the number of clusters, and l represents the number of iterations taken by the algorithm to converge. Typically, K and l are fixed in advance and are usually not too large. Therefore, the algorithm has the time complexity of the size of the training dataset times the dimensionality of each training instance, namely $O(nd)$.

4.2.2 Regression Subsystem

Regression analysis is a popular statistical process for estimating the relationships among variables, which has been widely applied [20]. It is utilised here as a segment to predict student final period grade. In particular, multi-variable linear regression, a highly flexible

mechanism for examining the relationship of a collection of independent variables with a single dependent variable [167], is an appropriate choice to perform the prediction. This is because different intuitive academic attributes are required to be taken into consideration to form the required regression model, with each of them being regarded as an independent factor in the evaluation of student academic performance.

The basic idea of the linear regression model is: given a dataset $\{y_i, x_{i1}, x_{i2}, \dots, x_{im}\}$, ($i = 1, \dots, n$) of n instances, the relationship between the dependent variable y_i and the m independent variables x_{ij} ($1 \leq j \leq m$) is assumed to be linear. That is, the underlying relationship amongst all the variables takes the form of

$$y_i = \alpha_0 + \alpha_1 x_{i1} + \alpha_2 x_{i2} + \dots + \alpha_m x_{im} \quad (4.1)$$

where $\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_m$ are termed regression coefficients.

To implement the algorithm of multi-variable linear regression for predicting student's final period grade, assume that there are m independent variables G_1, G_2, \dots, G_m : their values as given in the training samples are denoted as vectors. Denote these vectors as $V_{G_1} = [G_1^1, G_1^2, \dots, G_1^n]^T$, $V_{G_2} = [G_2^1, G_2^2, \dots, G_2^n]^T, \dots, V_{G_m} = [G_m^1, G_m^2, \dots, G_m^n]^T$, respectively, where n stands for the number of instances in the training dataset. Denote the dependent variable by V_{G_p} , with its value set encoded as the vector $V_{G_p} = [G_p^1, G_p^2, \dots, G_p^n]^T$.

Let $\hat{1}$ denote the unit value vector (of an n dimensionality) and

$$X = \begin{bmatrix} \hat{1} & V_{G_1} & V_{G_2} & \dots & V_{G_m} \end{bmatrix} = \begin{pmatrix} 1 & G_1^1 & G_2^1 & \dots & G_m^1 \\ 1 & G_1^2 & G_2^2 & \dots & G_m^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & G_1^n & G_2^n & \dots & G_m^n \end{pmatrix} \quad (4.2)$$

Then, the multi-variable linear regression to estimate the exam marks is of the following form:

$$y = X\beta \quad (4.3)$$

where β is an $m + 1$ dimensional regression coefficients vector, containing $\beta_0, \beta_1, \beta_2, \dots, \beta_m$ as sequenced elements.

In order to determine the element values of β , conventional least squares (LS) estimator is adopted here owing to its computational simplicity. The LS method minimises the sum of squared residuals, and leads to a closed-form expression for estimating the unknown vector β :

$$\beta = (X^T X)^{-1} X^T y \quad (4.4)$$

Note that for each cluster generated by the partition subsystem, a calculation on the corresponding regression coefficient vector $\beta^i (1 \leq i \leq K)$ is required, where K is the number of clusters generated by the partition subsystem. Thus, the complexity of the LS algorithm for multi-variable linear regression is $O(m^2n)$, where m is the number of independent variables and n is the number of instances in the training dataset. Since m is usually fixed and known in advance, and m is typically much smaller than n for the present application, the asymptotic time complexity of the regression subsystem can be approximated by $O(n)$.

4.2.3 Offset Value Generating Subsystem

In practice, predicting student academic performance with only consideration of previous records is not always enough. It is not surprising that students may achieve quite different results in their final period even if they have had the same or similar achievements at previous stages. This reality makes the task of reaching highly accurate prediction a challenge. Having taken notice of this, aspects other than just the student previous academic records need to be taken into account in order to generate better predicting results. Nowadays, it is commonly recognised that the study behaviour has a significant impact upon student academic achievement [186–188], making it an interesting factor worth investigating.

The present subsystem is developed in an effort to optimise predicted final period grade, by generating an offset value to the interim predicted final period grade for a given (target) student. Suppose that there are m previous academic records available for the target (student) instance. The computational process of this subsystem involves the following steps:

- 1) Calculate the Euclidean distances $ed_i (1 \leq i \leq n)$ between the target instance and every instance in the training dataset by:

$$ed_i = \sqrt{\sum_{j=1}^m (G_j^t - G_j^i)^2} \quad (4.5)$$

where n is the number of instances in the training dataset and G_j^t is the academic record of the target instance regarding the j^{th} period.

- 2) Find the nearest instances to the target by sorting the Euclidean distances returned by step 1, and put them into a vector (of a varying dimensionality), named $V_{nearest}$.
- 3) Calculate the differences in the final period grades between each pair of the instances in $V_{nearest}$, and find the maximum difference, denoting it by MAX_d . If $V_{nearest}$ contains only one element then the difference is set to 0.
- 4) If the Euclidean distance between the target and a certain instance in $V_{nearest}$ equals 0, copy the fuzzy membership values associated with that instance in the $V_{nearest}$ as the corresponding membership values of the target. If the Euclidean distance between the target instance and an instance in $V_{nearest}$ does not equal to 0, calculate its fuzzy membership values to each cluster in the same way as done by the fuzzy c-means method.
- 5) Preprocess the given data in response to student normal study behaviour as follows. For a datum presented in boolean form, transform it into “0” or “1”, where “0” represents “NO” and “1” represents “YES”. For a datum given in numeric form, normalise it to fall within the interval of $[0, 1]$.
- 6) Without losing generality, suppose that there are P attributes reflecting certain aspects of student normal study behaviour in the dataset, denoted by A_l ($1 \leq l \leq P$), and that there are N instances in $V_{nearest}$. For each instance in $V_{nearest}$, calculate its difference $Diff_k$ ($1 \leq k \leq N$) to the target instance by the following:

$$Diff_k = \sum_{l=1}^p (A_l^t - A_l^s) \quad (4.6)$$

where A_l^s represents the l^{th} attribute of the study behaviour involved in the instances within $V_{nearest}$, and A_l^t represents the l^{th} attribute of the study behaviour concerning the target instance.

- 7) For each instance in $V_{nearest}$, calculate its similarity to the target instance by the following [189]:

$$Sim_k = \frac{\sum_{i=1}^P (1 - |A_i^s - A_i^t|)}{P} \quad (4.7)$$

- 8) Calculate the offset value of the predicted final period grade for the target instance by:

$$offset_value = MAX_d \cdot \sum_{k=1}^N (Diff_k \cdot Sim_k) \quad (4.8)$$

Note that although the implementation of the offset value generating subsystem includes many steps, its time complexity is acceptable, with $O(n^2)$ to form the $V_{nearest}$, $O(N)$ to find the MAX_d , $O(P)$ to calculate $Diff_k$, $O(P)$ to compute Sim_k , and $O(N)$ to calculate the $offset_value$. Hence, the total time complexity is $O(n^2 + 2N + 2P)$. Since N and P are usually not large numbers for the present problem, the time complexity can be approximated by $O(n^2)$.

4.2.4 Estimation Subsystem

Given the regression coefficient vectors $\beta^1, \beta^2, \dots, \beta^K$ (where K is the number of clusters given by the partition subsystem), and a set of fuzzy membership values $V_\mu = [\mu_1, \mu_2, \dots, \mu_K]$ for a target student, the estimation subsystem implements a straightforward and final step of the entire computation process. Suppose that the vector of previous grades is denoted by V_{G_p} and that the $offset_value$ of the target instance has been obtained (see the preceding sub-section), the predicted final period grade $G_{predicted}$ can be calculated as

$$G_{predicted} = \left(\sum_{i=1}^K \mu_i V_{G_p}^T \beta^i \right) + offset_value \quad (4.9)$$

where $V_{G_p}^T$ denotes the transpose of V_{G_p} .

Note that the task of predicting final grade for a number of students can be implemented through the recursive application of this method. The predicted results of target students together with information regarding their previous academic grades and records of their study behaviours can be used to construct new training instances to enlarge the training dataset. Simply, the time complexity of the estimation method is $O(K)$ for one student, and $O(Kn)$ for n students.

4.3 Experimental Evaluation

This section presents experimental studies of the proposed approach. The work both illustrates the implemented system in action and demonstrates its efficacy.

4.3.1 Experimental Setup

4.3.1.1 Dataset preparation

A data corpus (SAP-PLUS) of four datasets, as stated in Section 3.3, are used as examples to conduct the experimentation. It is worth mentioning again that in the complete dataset, more than 30 attributes with related data to each attribute are collected. For simplicity and clarity, attributes which are closely related to the student academic performance and the study behaviour based on expert’s opinion are selected to conduct the experiment. The selected attributes are shown in Table 4.1.

Table 4.1 Preprocessed Student Academic Performance Related Attributes

Attribute	Description
study-time	weekly study time (numeric: 1: less than 2 hours, 2: 2 to 5 hours, 3: 5 to 10 hours, or 4: more than 10 hours)
failure-count	number of failures in the past for this academic module (numeric: integer)
support	extra support from educational school or family or other sources (binary: 1 for “yes”, 0 for “no”)
study-aim	whether or not to take higher education (binary: 1 for “yes”, 0 for “no”)
activities	extra curricular activities (binary: 1 for “yes”, 0 for “no”)
absence	days of school absence (numeric: from 1: few to 3: many)
health	(numeric: from 1: bad to 3: good)
G1	first period grade (numeric: from 0 to 20)
G2	second period grade (numeric: from 0 to 20)
G3	final period grade (numeric: from 0 to 20)

Before implementing the system, data preprocessing is carried out. In particular, attributes related to the normal student study behaviour, such as “study-time”, “failure-count” and “absence” are normalised into the range between 0 and 1. For instance, for a sample dataset with n instances, the normalised value of the data with regard to the attribute “study-time” for the i^{th} ($1 \leq i \leq n$) instance, denoted by $std_study-time_i$, is defined as follows:

$$std_study-time_i = \frac{study-time_i - study-time_{min}}{study-time_{max} - study-time_{min}} \quad (4.10)$$

where $study-time_{min}$ and $study-time_{max}$ are the maximum and minimum value of the attribute “study-time”, respectively.

4.3.1.2 Experimental method

In the experiments, each dataset is split into subsets for 10-FCV. The reported results are based on an average of 10 times of the 10-FCV. Since the ground truth of the students final period grades are in the form of integer, whereas the predicted final period grade are in the

form of floating-point number, the predicted data need to be transformed back to integers to support interpretability. Without losing fairness, when conducting the experiments and comparing the proposed work with other techniques, in addition to the empirical study on original estimated results, truncation, rounding, and rounding-off metrics are also investigated according to the statistical requirement, mapping the resulting predicted data onto an integer.

Despite the fact that the main use of the proposed system is to predict the numeric grade (in terms of integer scores) for a given target student, the method can also be applied as a classification model to categorise a target into a specific class based on the predicted numeric grade. According to the Erasmus grade conversion system (discussed in Section 3.3), the grades can be transformed into European Credit Transfer System (ECTS) Grades. The details of the transforming rule are listed in Table 3.5.

4.3.2 Results and Discussions

4.3.2.1 Prediction of numeric grades

For the analysis of the proposed approach, the predicted final period grades are compared with the corresponding underlying ground truth. The work is also compared with the standard multi-variable linear regression method (SMLR) as the baseline metric (specifically, best performance of original / truncation / rounding / rounding-off results based on SMLR are adopted as benchmark) and the clustering embedded linear-regression method (CELR) discussed in Chapter 3. Both of these methods can be widely adopted in the field of numeric prediction, especially when there is little knowledge of the non-linear relations between the end result and the attributes, as it is the case for the present application. In performing the step of partition, the number of clusters K is set to 5 in accordance with the Erasmus grade conversion system of classifying students. For the partition subsystem of the proposed approach (named as fuzzy clustering embedded linear clustering or FCELR), m is set to 2 and ε is set to 10^{-6} . For CELR, the initialisation of centroids for the partition subsystem is determined by the statistical computation approach, and the number of nearest neighbour for classification subsystem is set to 3, as it performs best in the experimentation presented in Section 3.3. All 4 available datasets are used here for training and testing. The resulting statistical indicators are listed in Table 4.2, 4.3 and 4.4.

Table 4.2 Comparison of Approaches in Terms of Mean Absolute Error for Prediction

Dataset	FCELR		SMLR		CELR	
	Original	Rounding-off	Original	Rounding-off	Original	Rounding-off
Maths (GP)	1.16 ± 0.17 (v)	1.18 ± 0.30 (v)	<u>1.35</u> ± 0.21	1.37 ± 0.22	1.21 ± 0.17 (v)	1.20 ± 0.20 (v)
Portuguese (GP)	1.77 ± 0.25 (v)	1.77 ± 0.23 (v)	<u>1.93</u> ± 0.20	1.94 ± 0.21	1.82 ± 0.20 (v)	1.84 ± 0.22 (v)
Maths (MS)	1.18 ± 0.21	1.17 ± 0.23	1.21 ± 0.22	<u>1.20</u> ± 0.23	1.16 ± 0.22	1.18 ± 0.21
Portuguese (MS)	1.63 ± 0.18 (v)	1.64 ± 0.18 (v)	<u>1.77</u> ± 0.16	1.79 ± 0.19	1.64 ± 0.15 (v)	1.66 ± 0.17 (v)

1. The benchmarks are presented with underlines.

2. The best results are highlighted in boldface.

3. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than benchmark of 95% confidence interval.

Table 4.3 Comparison of Approaches in Terms of Predicting Accuracy (%)

Dataset	FCELR			SMLR			CELR		
	Truncation	Rounding	Rounding-off	Truncation	Rounding	Rounding-off	Truncation	Rounding	Rounding-off
Maths (GP)	51.4 ± 1.4 (v)	51.8 ± 1.2 (v)	52.3 ± 1.2 (v)	48.8 ± 1.3 (*)	49.2 ± 1.1 (*)	<u>49.8 ± 1.2</u>	49.8 ± 1.0	49.4 ± 1.1 (*)	50.1 ± 1.1
Portuguese (GP)	47.6 ± 1.2 (v)	48.2 ± 1.1 (v)	48.4 ± 1.2 (v)	43.2 ± 1.2 (*)	43.6 ± 1.1	<u>43.8 ± 1.2</u>	47.2 ± 1.1 (v)	47.0 ± 1.0 (v)	47.3 ± 1.2 (v)
Maths (MS)	58.2 ± 1.0 (v)	57.7 ± 0.9 (v)	58.1 ± 1.0 (v)	55.2 ± 1.1 (*)	55.7 ± 1.2 (*)	<u>56.3 ± 1.2</u>	58.4 ± 1.2 (v)	57.8 ± 1.1 (v)	58.6 ± 1.1 (v)
Portuguese (MS)	48.6 ± 1.2 (v)	48.9 ± 1.1 (v)	49.2 ± 1.1 (v)	44.3 ± 1.3	43.6 ± 1.1 (*)	<u>44.6 ± 1.1</u>	48.1 ± 1.2 (v)	47.8 ± 1.3 (v)	48.3 ± 1.2 (v)

1. The benchmarks are presented with underlines.

2. The best results are highlighted in boldface.

3. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than benchmark of 95% confidence interval.

From Table 4.2, it can be observed that the approximating metric for mapping a real value into an integer (i.e., rounding-off) may lead to little variance to the original predicted result and avoid distorting the conclusion dramatically. FCELR performs best for most of the cases by considering the statistical factor of absolute mean error, which reveals its remarkable advantage over the baseline approach. The experimental results given in both Table 4.3 and Table 4.4 show that the FCELR outperforms SMLR and CELR, regarding predicting accuracy and within 1-grade error. These results jointly demonstrate that predicted final period grades are closer to the ground truth, demonstrating the significant potential of the proposed framework.

It is worth noting that FCELR generally performs even better than CELR, because for a large number of sample students, their academic grades are distributed normally and continuously. It is relatively more difficult for SCLR to find clear boundaries amongst the clusters. The proposed system has fuzzy-clustering embedded, avoiding the need of stating exactly to which category a target may belong. Instead, the academic records of each student that are considered belonging to a certain category are associated with membership values. Such membership values are used in computing the weight of each regression model contributing to the predicted grade of the target student.

Another reason for the proposed to outperform both SMLR and SCLR is that it makes better use of the attributes about the student normal study behaviour. Although these attributes are also taken into account by the other two predicting models, their values on the 0-1 scale are too small to make significant contributions in these models.

Also, FCELR possesses an interesting ability thanks to the introduction of an offset value. From a list of sample students with the same or similar previous academic records, given their final period grades, the proposed system can generate a predicted result exceeding the limits of these sample students. This may have helped further improve its performance.

4.3.2.2 Prediction of 5-level grades

To further analyse the results achievable FCELR, advanced universal classification techniques such as Naive Bayes method, K -NN, neural network, support vector machines (SVM), decision trees and random forest are also employed to classify the predicted final period grades. Particularly, the NaiveBayes metric released with Weka software [190] is set as a baseline model, IBk, MultilayerPerceptron, SMO, J48 and RandomForest algorithms released with the same software package are used to represent other listed classification approaches

respectively, with K being set to 3 for K -NN, and the polynomial kernel selected to implement SMO.

Table 4.5 Comparison of Classification Accuracy (%)

Dataset	FCELR	IBk	MultilayerPerceptron	SMO	J48	RandomForest	NaiveBayes
Maths (GP)	78.3 ± 6.4 (v)	69.5 ± 6.5	70.6 ± 6.7	78.5 ± 6.5 (v)	78.4 ± 6.4 (v)	71.7 ± 6.3	72.2 ± 6.5
Portuguese (GP)	77.2 ± 5.1 (v)	65.5 ± 5.8 (*)	71.8 ± 5.7	71.5 ± 4.9	74.3 ± 5.2	68.3 ± 6.3	71.3 ± 5.3
Maths (MS)	81.2 ± 5.2 (v)	73.4 ± 6.7	77.5 ± 5.5	80.1 ± 5.8 (v)	80.8 ± 6.7 (v)	73.7 ± 5.6	75.7 ± 5.7
Portuguese (MS)	75.7 ± 5.4 (v)	65.1 ± 5.2 (*)	74.4 ± 5.5	72.5 ± 5.7	73.1 ± 4.3	71.2 ± 4.7	72.2 ± 5.5

¹. The best two results are highlighted in boldface.

². Sign (*) / (v) indicates that the corresponding result is significantly worse / better than benchmark of 95% confidence interval.

The resultant accuracies are shown in Table 4.5. For comparison, the two highest classification accuracy are denoted in boldface. Clearly, FCELR is in general amongst the group with the best predicting accuracy. Although ranked 3rd for its performance on dataset Maths(GP), it yields similar predicting accuracy to SMO and J48 (ranked 1st and 2nd respectively), which is still significantly better than the given benchmark. The empirical study against these popular classification models demonstrates the great prospect of FCELR on prediction of student academic performance.

4.4 Summary

This chapter has proposed a novel approach to predicting student performance in academic courses. Unlike simple clustering regression analysis which takes part of the precise sample data into consideration, the proposed approach processes the universal data with an embedded step of fuzzy clustering. This has an intuitive appeal in handling a large number of student academic records which are typically normally and continuously distributed. The work makes use of attributes that are related to observed student study behaviour, by introducing an offset value in the predicting model. The implementation of the embedded fuzzy clustering approach, supported by the offset value mechanism, generates better results than the existing methods. With fuzzy representation, the approach synthesises the use of intuitive attributes from an academic course and from student normal study behaviour. This helps make the predicted results more readily interpretable, while involving simple computation.

Chapter 5

Fuzzy Connected-Triple for Prediction of Inter-Variable Correlation

Chapter 3 and Chapter 4 have presented intelligent systems for predicting uninformed feature values at object or instance level. From this chapter, the horizon of prediction has been moved up to the attribute (feature) or variable level. In order to perform investigation on attribute variables, their related feature data requires comprehensive examination. Over the past decade, such data has been growing at an astonishing pace, with the term ‘Big Data’ becoming a hot topic in both industry and academia. In particular, data mining, a process to discover patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems [191], has been under vibrant investigation. As the amount of available data grows, the success of managing and analysing the information embedded in the data becomes ever more practically significant, whilst becoming ever more difficult in the meantime.

Fortunately, the increasing growth of computational capability has to a certain extent, enabled the handling of such large amount of data through a range of approaches, including the method of social network analysis (SNA) that has been increasingly gaining popularity. In SNA, link prediction is one of the most salient and challenging tasks. It is particularly difficult to perform the discovery of missing or developing links in a certain network of interest [57]. However, link prediction is very useful to help: infer the underlying complete network (from partially observed structures) [192, 193], understand the evolution of networks [194, 195], and predict hyper-links in heterogeneous social networks [196]. Traditionally, most of the approaches for detecting unobserved links are based on topological information, including neighbour-based metrics, path-based metrics and random walk-based metrics [95]. Recent studies have extended such classical metrics by adding weights to the existing links

within a topological graph in response to the information obtained from explicitly related sources [197]. Besides, other collections of approaches, including probabilistic methods and algorithmic methods, have been proposed to handle different types of link prediction problem [198]. Nevertheless, typical existing approaches (including all discussed above) are set for a specific problem within a local scope, dealing with the information coming from a single data source.

Addressing the task of link prediction, the use of connected-triples has an intuitive appeal. A connected-triple is a graph representation formed by three vertices and two undirected edges, with each edge connecting two distinct vertices out of the three via the remaining vertex. A network constructed with such connected triples offers a potentially effective mechanism for link prediction, particularly when any given information content is obtained from different data sources where parts of the information overlap. Inspired by this observation, unlike previous research that focussed on identifying links between objects or entities in a specific region, this chapter presents an innovative piece of work that is driven by the interests in searching for links between variables extracted from different data/information sources, through the introduction and exploitation of fuzzy connected-triple.

The potential underlying links between variables or entities collected from different sources are usually hidden, not obvious or even difficult to be discovered, making the task of link prediction from such data sources a challenge. Traditionally, this type of work has generally been handled by human experts. Thus, designing and implementing a predicting method which learns from human logical reasoning will be helpful to automate such prediction processes, especially when facing large and diverse data sources. Practically, when describing a link or a set of links, linguistic terms such as “Strong”, “Medium” and “Weak” are natural adjectives to depict the link strength rather than crisp numerical values (that are typically utilised in conventional connected-triple models). In addition, common knowledge such as “if A has a strong link to B , and B has a strong link to C , then A may have a strong link to C ” perfectly matches human logical thinking. It is to reflect such intuitions, fuzzy logic is adopted in the present work to serve as the basis upon which to develop a multi-source link prediction model. Such link prediction problems are obviously of general interest in many data mining applications.

Overall, this chapter presents two major contributions to knowledge: (1) It proposes a novel approach to determining the correlation between attribute variables from distinct datasets with different entity references. (2) It proposes a fuzzy link prediction model which

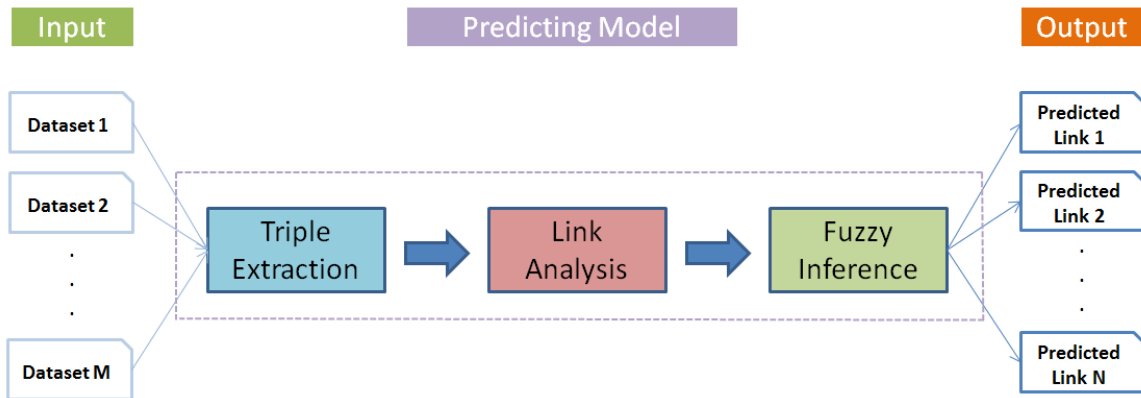


Fig. 5.1 Predicting Framework

radically departs from conventional crisp representation of connected-triple-based link detection, resulting in models that resemble human inference and facilitate interpretability.

The rest of this chapter is arranged as follows. Section 5.1 introduces the proposed architecture for the development of a fuzzy connected-triple system for link prediction, describing details on model construction, link measures, and inference procedures. Section 5.2 exhibits the results of empirical evaluation, supported by comparative studies with alternative predicting methods. Section 5.3 concludes the chapter with outlook for further development.

5.1 Predicting System

This section presents the proposed general framework for developing a system that predicts link strengths with data from multiple sources. It describes the system's components and their associated time complexity analyses.

5.1.1 Conceptual Framework

The structure of the predicting system is shown in Fig. 5.1. As can be seen, it comprises three distinct component subsystems, each of which implements the functionality of: triple extraction, link analysis, and fuzzy inference, respectively. These activities are integrated to construct a required predicting model, whose implementation steps are detailed below.

Dataset 1					Dataset 2			
	V_A	V_B	V_C	V_D		V_C	V_D	V_E
x_1					y_1			
x_2					y_2			
...					...			
x_r					y_s			

Fig. 5.2 Sample Datasets

5.1.2 Connected-Triple Extraction

5.1.2.1 Concept of Connected-Triple

Connected-triple modelling, first introduced to analyse global clustering coefficient [199], is also referred to as a method for measuring network transitivity. For instance, it may be applied to measure the extent to which a friend of someone's friend is also the friend of that person. Formally, a connected-triple, $Triple = \{V_{Triple}, W_{Triple}\}$, is a subgraph of $G(V, W)$, where V represents the set of vertices in the graph and W represents the set of edges connecting related pairs of vertices, containing three vertices $V_{Triple} = \{v_i, v_j, v_k\} \subset V$ and two edges $W_{Triple} = \{w_{ij}, w_{jk}\} \subset W$, with w_{ik} being unknown as there is no direct edge connecting v_i and v_k . The vertex v_j connecting the other two vertices is called the centre of the triple, and v_i or v_k is called an end of the triple (there being two ends per triple, of course).

5.1.2.2 Extracting Connected-Triples from Datasets

Extracting connected-triples from (the same or different) original datasets plays a fundamental role in the present work. An example of two distinct datasets is shown in Fig. 5.2, where the variables v_C and v_D co-occur in both datasets (encircled in red), whilst the variables v_A and v_B only appear in Dataset 1, and v_E only appears in Dataset 2. Importantly, an obvious but crucial point is that although there exist variables co-occurring in more than one dataset, these datasets cannot be easily merged into one since the instances in the datasets can be totally distinct, and so can the numbers of instances in the datasets. For example, the instances x_1, x_2, \dots, x_r in Dataset 1 and the instances y_1, y_2, \dots, y_s in Dataset 2 are completely different from each other, although they share the two aforementioned common variables. Also, Dataset 1 has r instances but Dataset 2 contains s instances, while $r \neq s$.

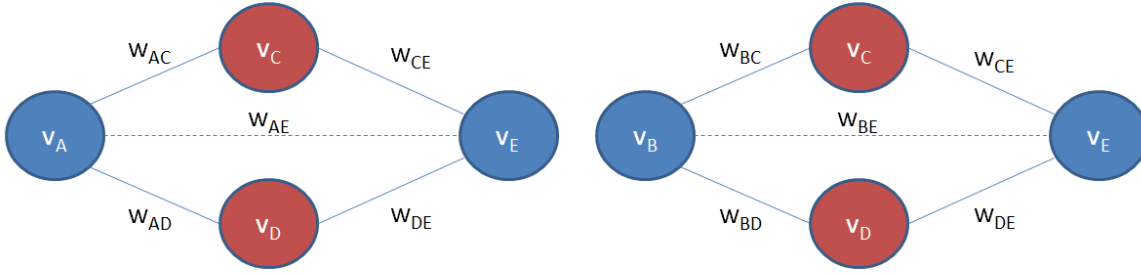


Fig. 5.3 Connected-Triples Extracted from Sample Datasets

An example of extracting connected-triples from original datasets is shown in Fig. 5.3, with each vertex representing a variable in the sample datasets given in Fig. 5.2. For instance, v_A in Fig. 5.3 denotes the (same) variable v_A in Dataset 1 of Fig. 5.2. A link (represented in a solid line) between two distinct variables denotes that these variables are co-occurring in at least one of the sample datasets, and therefore, indicates that they are to a certain extent related to each other. In Fig. 5.3, four triples, $Triple_i, i = 1, 2, 3, 4$, are formed from Datasets 1 and 2 in Fig. 5.2, where $V_{Triple_1} = \{v_A, v_C, v_E\}$, $V_{Triple_2} = \{v_A, v_D, v_E\}$, $V_{Triple_3} = \{v_B, v_C, v_E\}$, and $V_{Triple_4} = \{v_B, v_D, v_E\}$. The centres of these four connected-triples are v_C and v_D , respectively. The dash line between v_A and v_E and that between v_B and v_E represent the potential links between pairs of the variables v_A and v_E and those of v_B and v_E , respectively, which do not exist in the originally provided datasets.

5.1.2.3 Transitivity Property of Connected-Triple

An interesting but important characteristic of connected-triple is its transitivity property. According to this property, two independent connected-triples can form a third connected-triple. For instance, as shown in Fig. 5.4, from $Triple_1 = \{\{v_A, v_B, v_C\}, \{w_{AB}, w_{BC}\}\}$, a new link w_{AC} connecting v_A and v_C may be generated. Likewise, from $Triple_2 = \{\{v_C, v_D, v_E\}, \{w_{CD}, w_{DE}\}\}$, another new link w_{CE} connecting v_C and v_E may also be obtained. Based on the variables v_A, v_C, v_E , and the links w_{AC}, w_{CE} , an extended connected-triple $Triple_3 = \{\{v_A, v_C, v_E\}, \{w_{AC}, w_{CE}\}\}$ (depicted with dash lines) can be computationally produced.

5.1.3 Link Analysis

Having identified a new connected-triple from the source datasets, the task of determining correlation between a pair of variables that belong to two different datasets becomes to

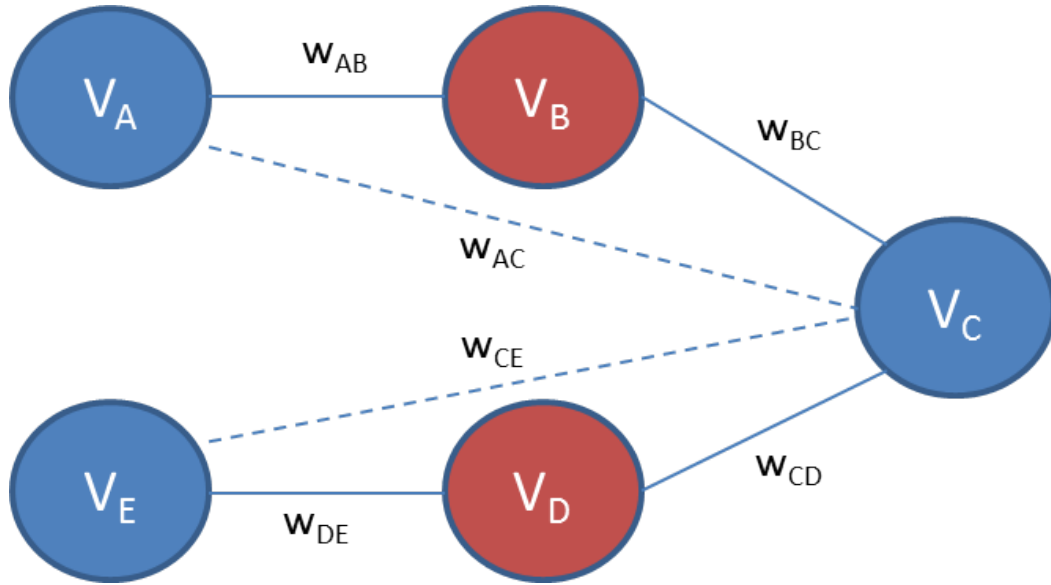


Fig. 5.4 Transitivity of Connected Triple

predict whether there exists a (hidden) link between the two end vertices. If so, a further question is what may be the strength on such a link. To address these issues, prerequisites including the properties of any known links between pairs of vertices in the triple need to be obtained in advance.

In practice, the link property is generally described by its weight, which may correspond to a wide variety of aspects depending on the underlying application problem. For each connection between a given pair of distinct variables, different mechanisms may therefore be devised for estimating the strength of that connection. For instance, in a route graph or map, the weight of a link may indicate the route distance between two linked venues. In a graph of co-authorship, the weight of a link may denote the number of papers two authors collaborated to publish. In a graph of webpage linkages, the weight on a link may represent the popularity of people stepping from one to another. In the current study, a link between two vertices signifies a certain relationship between those variables in the datasets. Thus, the weight of a link is utilised to capture and reflect the closeness or correlation of the corresponding variables.

5.1.3.1 Categorical Data

For a pair of variables in a dataset filled with discrete or nominal values, their relationship can be described by the co-occurrence frequency of the variables taking on a common value. For such data, two indices to measure link strengths can readily be adopted, namely

Normalised Mutual Information (NMI) and Frequency of Most Popular Term-Pair (FMTP). These strengths are detailed below, which can themselves be combined to form fused link properties.

- 1) **Normalised Mutual Information (NMI)** Generally speaking, mutual information is a symmetric measure to quantify the statistical information shared between two distributions [200]. The use of this measure in the present research provides a sound indication of the shared information between a given pair of variables. In particular, for two discrete random variables v_A and v_B , the mutual information between them can be denoted as $MI(v_A, v_B)$ and computed by

$$MI(v_A, v_B) = \sum_{v_b \in D_B} \sum_{v_a \in D_A} p(v_a, v_b) \log\left(\frac{p(v_a, v_b)}{p(v_a)p(v_b)}\right) \quad (5.1)$$

where $p(v_a, v_b)$ is the joint probability distribution function of v_A and v_B , and $p(v_a)$ and $p(v_b)$ are the marginal probability distribution functions of v_A and v_B , with v_A and v_B defined over the domains D_A and D_B , respectively. Note that there is no upper bound for $MI(v_A, v_B)$. Thus, for better facilitating interpretation and comparison, a normalised version of $MI(v_A, v_B)$ that ranges from 0 to 1 is desirable while describing the relationship strength between v_A and v_B .

Let $H(v_A)$ denote the entropy of v_A [201], which is defined by

$$H(v_A) = - \sum_{v_a \in D_A} p(v_a) \log p(v_a) \quad (5.2)$$

From this, the normalised mutual information between v_A and v_B [48], denoted by $NMI(V_A, V_B)$, can be computed such that

$$NMI(v_A, v_B) = \frac{MI(v_A, v_B)}{\sqrt{H(v_A)H(v_B)}} \quad (5.3)$$

The time complexity of computing NMI is $O(mnd)$, where d denotes the number of instances in the dataset, and m and n represent the cardinalities of variable domains of v_A and v_B , respectively. Typically, m and n are fixed to a small or medium number in advance. From psychological viewpoint, to ensure model interpretability, the cardinalities are normally set to a maximum value of 9. Therefore, this measurement has the linear time complexity proportional to the size of the dataset, namely $O(d)$.

- 2) **Frequency of Most Popular Term-Pair (FMPT)** NMI may be a simple measurement computationally. However, only taking it into consideration when modelling the link strengths between distinct variables may not be sufficiently effective. In particular, the frequency of occurrence of different terms with regard to a certain variable within a given dataset can be rather different. This is because datasets may be rather skewed; certain terms may have a very high occurrence frequency but one or more of the others may have a very low frequency. This is rather common a phenomenon in real-world problems. For example, more than 90% of the primary school pupils are guarded by their parents and they are much less likely to be guarded by other relatives. The statistics of blood type distribution in the UK also shows that 44% of the population have blood type *O*, and only 10% have blood type *B* [202].

When considering any link relationship between two variables v_A and v_B of such skewed datasets, suppose that V_A^1 and V_B^1 are the most popular terms taken by the variables v_A and v_B , respectively. Then, even if most of the instances have the term V_A^1 for v_A and V_B^1 for v_B simultaneously, the NMI score of the link between v_A and v_B may still be low. This is because the NMI score is significantly affected by the number of other term-pairs and their proportion. In this case, judging the link strength between these two distinct variables by only calculating the NMI score may seriously distort the result, misinterpreting the closeness of the relationship between the two. This calls for the development of the so-called frequency of the most popular term-pair measure (FMPT).

Without losing generality, assume that a given dataset includes a total of d instances, and that v_A and v_B are two discrete variables describing the instances in the dataset, each containing m and n terms, respectively. Let V_A^i ($1 \leq i \leq m$) and V_B^j ($1 \leq j \leq n$) be the terms possibly taken by v_A and v_B , and $S_{V_A^i}$ and $S_{V_B^j}$ ($1 \leq j \leq n$) be the set of instances which has the term V_A^i for v_A and V_B^j for v_B . The FMPT score or weight on the link between the variable v_A and v_B is defined by

$$FMPT(v_A, v_B) = \frac{\max_{1 \leq i \leq m, 1 \leq j \leq n} d(S_{V_A^i} \cap S_{V_B^j})}{d} \quad (5.4)$$

where $d(S_{V_A^i} \cap S_{V_B^j})$ denotes the number of instances which have the term V_A^i for the variable v_A and V_B^j for v_B simultaneously.

Note that the FMPT score is also ranged from $[0,1]$. The time complexity of computing FMPT is also $O(mnd)$, where m, n, d are of the same meanings as previously defined.

- 3) **Fusion of Link Properties** As indicated above, both NMI and FMPT take values from the same range $[0,1]$. It is therefore convenient to aggregate the results if both are applied. The fusion of these two measurements is useful because they capture different underlying relationship properties of the datasets in general and the variables' terms in particular. For a certain link between two distinct discrete variables v_A and v_B , given the NMI and FMPT scores, the combined weight of the link $SYN(v_A, v_B)$ can be calculated in a straightforward manner such that

$$SYN(v_A, v_B) = \max(NMI(v_A, v_B), FMPT(v_A, v_B)) \quad (5.5)$$

Obviously, the combined link weight has the same real value range as either of the component weights, i.e., between 0 and 1. The complexity of this fusion step is extremely simple, being $O(2)$. This may be linearly generalised if there are more than 2 such base link strengths. The benefit of adopting the maximum operator is that it takes into consideration the most salient feature of the data while being simple in computation.

Note that the strength fusion does not have to be implemented as above, but can be done in various alternative ways, e.g., by finding the arithmetic average of the component strengths, if preferred. However, this does not affect the approach taken, rather than adding a small amount of extra computational expense and so is regarded as being beyond the scope of the current investigation.

5.1.3.2 Continuous Numeric Data

- 1) **Absolute Pearson Correlation Coefficient (APCC)** For a pair of variables with continuous data as their entities, the aforementioned measurements may not work. Instead, statistical means to measure the bivariate correlation may be a fitted alternative. Specifically, Pearson Correlation Coefficient (PCC) [203], a measure of the linear correlation between two continuous variables, is adopted here. A simple but important factor needs to be noted is that for traditional use of PCC, it has a value range between $[-1, 1]$. Considering the main concern here is whether two variables have strong correlation and if so, how strong such a relationship may hold, whether the two variables have a negative or positive correlation is beyond the current scope. Hence, only the absolute

value of Pearson Correlation Coefficient, APCC, is herein employed to measure the link strength between two continuous variables. Formally, the APCC between two variables v_A and v_B can be written as follows:

$$APCC(v_A, v_B) = \frac{|\sum_{g=1}^d (V_a^g - \bar{V}_A)(V_b^g - \bar{V}_B)|}{d\sigma_{V_A}\sigma_{V_B}} \quad (5.6)$$

where V_a^g and V_b^g represents the value of v_A and that of v_B for the g -th instance in the dataset, respectively; \bar{V}_A and \bar{V}_B stand for the average value of all the instances with regard to v_A and that to v_B ; and σ_{V_A} and σ_{V_B} denote the standard deviation of v_A and that of v_B within the discussed dataset. The time complexity of computing APCC for any variable pair is $O(d)$, where d denotes the number of instances in the dataset.

- 2) **Maximal Information Coefficient (MIC)** Recently, a novel method named Maximal Information Coefficient (MIC) to measure the correlation between two continuous numeric variables was proposed [204]. Similar to NMI, MIC also takes value between zero and one, and it has two main properties: Generality and equitability. Generality means that with a sufficiently large sample size, it can capture a wide range of associations, including linear, parabolic, exponential, and periodic. More sophisticatedly, it is capable of handling non-functional correlations such as round and eclipse. Equitability means that MIC gives similar scores to equally noisy relationships regardless the type of relationships. The formula for computing MIC between v_A and v_B is defined as follows:

$$MIC(v_A, v_B) = \max_{n_{v_A}, n_{v_B} < B(n)} \left\{ \frac{I(v_A, v_B)}{\log \min\{n_{v_A}, n_{v_B}\}} \right\} \quad (5.7)$$

where

$$\begin{aligned} I(v_A, v_B) &= H(v_A) + H(v_B) - H(v_A, v_B) \\ &= -\sum_{i=1}^{n_{v_A}} p(v_A^i) \log p(v_A^i) - \sum_{j=1}^{n_{v_B}} p(v_B^j) \log p(v_B^j) \\ &\quad + \sum_{i=1}^{n_{v_A}} \sum_{j=1}^{n_{v_B}} p(v_A^i, v_B^j) \log p(v_A^i, v_B^j) \end{aligned} \quad (5.8)$$

Note that n_{v_A} and n_{v_B} are the number of bins for partitioning v_A and that for v_B , respectively, and they are set to satisfying the criterion $n_{v_A} \cdot n_{v_B} < B(n)$. Usually, $B(n)$ is set to $B(n) = n^{0.6}$ in real world applications, where n is the number of instances in the sample data [204].

5.1.3.3 Numeric-Categorical Mixed-Type Variable Pair

In real-world applications, datasets are not always arranged with same type of variables. It is common to see datasets with mixed types of attribute value. For instance, a data table with regard to personal information may contain a nominal value to represent gender and a numeric value to denote age. Unfortunately, determining the correlation between different types of attribute variable seems difficult. Here, a heuristic is proposed to assist approximating the relationship between attribute variables of different types (numeric-categorical mixed type).

Basically, this method works by transforming the numeric variable in a mixed-type variable pair into its categorical counterpart at first, and then performing corresponding correlation measuring metric on this transformed variable pair. In order to execute such a transformation process, a specific number of class labels for the numeric attribute variable is required to be determined in advance. This task is carried out by performing a series of unsupervised clustering algorithm with different a number of clusters selected each time, and searching for the number which best segments the numeric values into categories.

The elbow method, due to its effectiveness and efficiency, is adopted to help such transformation. It works on the assumption that adding another cluster does not generate a remarkable improvement under a provided objective function. The objective function $f_{improve}$ representing performance improvement, is defined as:

$$f_{improve}(k) = \frac{SSE(k-1) - SSE(k)}{SSE(k-1)} \quad (k = 2, 3, \dots, 9) \quad (5.9)$$

where $SSE(k)$ denotes the total sum of square errors determined by selecting k as the number of clusters. A demonstrating example of the elbow method is shown in Fig. 5.5. The number of clusters in this case is determined as 4, which is the obvious elbow point or turning point. For the present study, the number of clusters selected ranges between 2 to 9, which considers both computational efficiency and consistency with human common sense. Note that the term “mixed type feature variable pair” used in the thesis represents “numeric-categorical mixed-type feature variable pair”, unless otherwise stated .

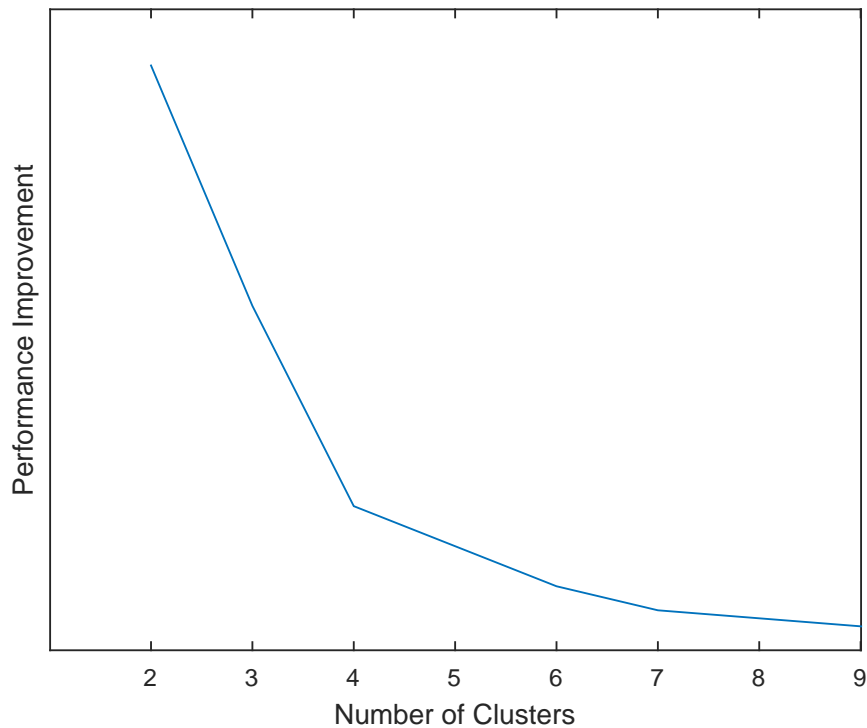


Fig. 5.5 Example of Elbow Method

In particular, the simple k -means clustering method is applied to partition the numeric data into various groups, with random initial centroids provided. The performance evaluation is based on an average of 20 runs of the algorithm. Having determined the number of clusters, a refined PAM approach embedded with quantile based determination of initial medoids is ultimately conducted to categorise continuous values of a numeric variable and assign each of them with a class label. Specifically, quantiles employed here represent a set of values for a numeric variable which divides the examined dataset into equal sized subsets. The initial medoids for PAM are determined as the average value of two adjacent quantiles. The minimum and maximum value of the variable in the dataset are also adopted to help determine the initial medoids for PAM. For instance, provided that four clusters of instances in a dataset are required to be generated, the initial medoids for PAM to create such groups are respectively set to the median value of the minimum and 1st quartile, and that of 1st and 2nd quartile, 2nd and 3rd quartile, 3rd quartile and the maximum value, with regard to the discussed variable.

5.1.4 Fuzzy Inference Model

Having determined the weights over given links within a connected-triple model, the predicting system reaches its final step: logic deduction. A fuzzy inference model is employed to implement this task, providing a flexible means to perform human-interpretable reasoning by the use of linguistic terms rather than numeric values (although the linguistic terms still have their underlying numerical interpretation). For the problem of link prediction, linguistic labels such as “Strong”, “Medium” and “Weak” are natural words that are commonly used to describe link strengths. The present work follows this practical observation, and attempts to mine the underlying logic hidden beneath the connected-triple:

$$\begin{aligned} &\mathbf{IF} \textit{link}_1 \mathbf{IS} (\textit{strong}\backslash\textit{medium}\backslash\textit{weak}) \\ &\mathbf{AND} \textit{link}_2 \mathbf{IS} (\textit{strong}\backslash\textit{medium}\backslash\textit{weak}) \\ &\mathbf{THEN} \textit{link}_3 \mathbf{IS} (\textit{strong}\backslash\textit{medium}\backslash\textit{weak}) \end{aligned}$$

where \textit{link}_1 and \textit{link}_2 represent the two known links in a certain triple, each of which connects the triple centre to one of the two ends, and \textit{link}_3 represents the link to be established with a (predicted) link strength score. Such a fuzzy system involves two key procedures as detailed below.

5.1.4.1 Link Weight Fuzzification

To enable the capture and representation of imprecisely described link weights, and to support the derivation of the required fuzzy inference model through data-driven learning, fuzzification of the link strengths for each identified connected-triple is necessary. Without losing generality, to ensure interpretability of the resulting model, a set of membership functions used to depict link strengths is presumed to have been prescribed by domain experts. However, for applications where there is a sufficient amount of historical data, a clustering method may be employed to derive the required set of (potentially more objective) linguistic terms. In this work, especially for the experimental evaluation to be presented in the next section, the linguistic terms used are predefined by the domain experts (with prescribed asymmetrical membership functions used to partition the underlying problem domains), without any optimisation and are shown in Fig. 5.6.

5.1.4.2 Fuzzy Inference

In the process of performing fuzzy inference for link prediction, as with other applications of fuzzy systems, t -norm and t -conorm operators are adopted to interpret logic connectives over connected-triples, aggregating fuzzy values [205]. In general, for each pair of end vertices,

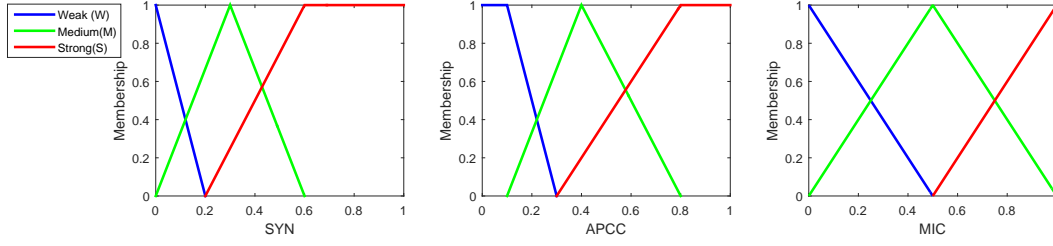


Fig. 5.6 Fuzzy Membership Values of Link Weight with Respect to Different Measures

there may exist several distinct centres connecting them to form different connected-triples. As such, each connection will lead to an intermediate inference outcome regarding the link strength, indicating the level that that triple may contribute towards the final prediction result. Thus, a t -conorm operator is needed to aggregate all the intermediate predicted outcomes together.

Given a connected-triple CT , let $f_{link_1}^L$ and $f_{link_2}^L$ be the fuzzy membership values of the link strengths, or link weights on the links $link_1$ and $link_2$, where linguistic terms $L \in \mathcal{L}$, with \mathcal{L} representing a collection of all fuzzy sets used to express the linguistic labels (namely, the terms “Strong”, “Medium” and “Weak” as given in the previous example). The predicted fuzzy value of a single connected-triple can then be described as a membership function:

$$F_{P_{CT}} = [\nabla(f_{link_1}^{L_1}, f_{link_2}^{L_1}), \nabla(f_{link_1}^{L_2}, f_{link_2}^{L_2}), \dots, \nabla(f_{link_1}^{L_M}, f_{link_2}^{L_M})] \quad (5.10)$$

where ∇ denotes a certain predefined t -norm, and M represents the number of the terms possibly used to describe the linguistic link strength.

Suppose that there are N connected-triples formed by a specific pair of end vertices with a common corresponding centre, the predicted fuzzy value for the link strength of P_{link} can be logically interpreted as the following:

$$\begin{aligned} F_{P_{link}} = & [\Delta(f_{P_{CT}^1}^{L_1}, f_{P_{CT}^2}^{L_1}, \dots, f_{P_{CT}^N}^{L_1}), \\ & \Delta(f_{P_{CT}^1}^{L_2}, f_{P_{CT}^2}^{L_2}, \dots, f_{P_{CT}^N}^{L_2}), \\ & \vdots \\ & \Delta(f_{P_{CT}^1}^{L_M}, f_{P_{CT}^2}^{L_M}, \dots, f_{P_{CT}^N}^{L_M})] \end{aligned} \quad (5.11)$$

where Δ represents an extended version of a certain t -conorm which can take a finite number of arguments. It aggregates those fuzzy membership values obtained from each connected-

Sample data 1

No	Family support	1 st semester grade	2 nd semester grade
1	yes	B	B
2	yes	A	B
3	yes	B	A
4	no	C	C
5	yes	B	A
6	no	B	C
7	yes	A	B
8	yes	B	B
9	no	C	B
10	no	B	B
11	yes	B	B
12	yes	C	C
13	yes	B	A
14	no	B	B

Sample data 2

No	Family size	1 st semester grade	2 nd semester grade
1	large	B	C
2	medium	B	B
3	small	A	B
4	small	A	A
5	medium	B	B
6	large	C	C
7	large	C	B
8	small	A	A
9	small	B	B
10	medium	B	B

Fig. 5.7 Two Simple Datasets Used for Illustration

triple corresponding to a pair of variables and generates a new fuzzy membership value for the predicted link between those two variables. As the final result, what is returned is a fuzzy value regarding that to what extent a detected link is of a certain strength with respect to each individual predefined link weights (whose definition has been provided by the domain experts). If however, it is desirable to provide a numerical number for the predicted link strength an additional computational step is to defuzzify the resulting fuzzy membership value.

5.1.5 Illustrative Link Strength Prediction

Consider two simple datasets regarding student academic performance, as shown in Fig. 5.7. The two datasets contains 14 and 10 distinct instances, respectively, with the attributes “1st semester grade” and “2nd semester grade” shared by both. This illustrative example is to demonstrate that the proposed approach can predict the correlation between the variable “Family support” in dataset 1 and the variable “Family size” in dataset 2, with an intuitively appealing measured link strength.

For shorthand, denote the variables “Family support”, “1st semester grade”, “2nd semester grade” and “Family size” as *f_{sup}*, “1_{sg}”, “2_{sg}” and *f_{zise}*, respectively. Then, from the given datasets, the following two connected-triples can be directly extracted from these

datasets, one from each:

$$Triple_1 = \{\{v_{f_{sup}}, v_{1sg}, v_{f_{size}}\}, \{w_{f_{sup}-1sg}, w_{1sg-f_{size}}\}\}$$

$$Triple_2 = \{\{v_{f_{sup}}, v_{2sg}, v_{f_{size}}\}, \{w_{f_{sup}-2sg}, w_{2sg-f_{size}}\}\}$$

From these, according to Eqn. (5.1), (5.2) and (5.3) it can be computed that:

$$NMI(V_{f_{sup}}, V_{1sg}) = 0.139, NMI(V_{f_{sup}}, V_{2sg}) = 0.172$$

$$NMI(V_{f_{size}}, V_{1sg}) = 0.580, NMI(V_{f_{size}}, V_{2sg}) = 0.474$$

Similarly, through Eqn. (5.4) it can be computed that:

$$FMPT(V_{f_{sup}}, V_{1sg}) = 0.429, FMPT(V_{f_{sup}}, V_{2sg}) = 0.357$$

$$FMPT(V_{f_{size}}, V_{1sg}) = 0.300, FMPT(V_{f_{size}}, V_{2sg}) = 0.300$$

Thus, the weights on these links can be computed by Eqn. (5.5), such that

$$SYN(V_{f_{sup}}, V_{1sg}) = \max(0.139, 0.429) = 0.429$$

$$SYN(V_{f_{sup}}, V_{2sg}) = \max(0.172, 0.357) = 0.357$$

$$SYN(V_{f_{size}}, V_{1sg}) = \max(0.580, 0.300) = 0.580$$

$$SYN(V_{f_{size}}, V_{2sg}) = \max(0.474, 0.300) = 0.474$$

Having acquired the weights for the existing links (within individual datasets), the next step is to conduct fuzzy inference. Suppose that the fuzzy membership functions of a synthesised link strength is provided by the domain experts in linguistic terms as specified in Fig. 5.6. In this simple illustration, assume that the Max-Min aggregation method is taken to compute the *SYN* weights. Then, for *Triple_1*, according to Eqn. (5.10), its weight F_{PCT} can be calculated such that

$$\begin{aligned} & [\min(f^W(0.429), f^W(0.58)), \min(f^M(0.429), f^M(0.58)), \min(f^S(0.429), f^S(0.58))] \\ & = [0, 0.067, 0.5725] \end{aligned}$$

where f^W , f^M , f^S denote the fuzzification results of the link weights with respect to the linguistic terms “Weak”, “Medium” and “Strong”, respectively. What this fuzzy result stands

for is that the detected link is not “Weak” (as it is of a zero membership value with regard to this strength label), a tiny membership for the fuzzy concept “Medium”, and a significant membership value for the given linguistic term “Strong”. Following the same calculating procedure, for *Triple_2*, its F_{PCT} score is $[0, 0.42, 0.3925]$. Hence, with respect to Eqn. (5.11), the predicted fuzzy value representing the strength of the link between variables “Family support” and “Family size” is:

$$[\max(0, 0), \max(0.067, 0.42), \max(0.5725, 0.3925)] = [0, 0.42, 0.5725]$$

Finally, if a numerical strength score between the two variables is desirable (as opposite to a fuzzy value), then by employing the centre of gravity (COG) method for defuzzification, the predicted link score of 0.5804 can be obtained in a straightforward manner. Note that the above illustrative example is carried out on categorical data sets. However, this method can also be applied to numeric data sets, although the strategy to measure link strengths needs to be adjusted accordingly, as previously stated.

5.2 Empirical Evaluation

This section presents experimental studies of the proposed approach, with comparison against other popular link prediction techniques on different types of datasets demonstrated. Complexity analysis is also conducted and presented to check the efficiency of the proposed work.

5.2.1 Datasets

The experimental evaluation is conducted on both real world data from UCI benchmark data sets [206] and on a collection of synthetic datasets. Since there is hardly any corpora of datasets designed particularly for the current study, the data sets from UCI benchmark data sets are split into several subsets with overlapped variables according to human knowledge. To test the performance of the proposed approach on larger sized groups of datasets involving more variables, six different corpora of synthetic data sets are also generated to conduct the experiment. Table 5.1 shows a summary of the characteristics of all datasets employed.

Table 5.1 Summary of Datasets: Link Prediction

Dataset Collection	Type	NDC	ANIED	ANVED	ANVCTD
Bank	C	7	6459	4	2
Mushroom	C	9	912	5	2
Salary	C	6	5028	6	1
Student-Por	C	4	163	8	3
Student-Mat	C	4	101	8	2
Connect-4	C	6	11034	7	2
Wine	N	4	1256	3	1
Twitter	N	7	82038	13	4
Facebook	N	5	104	5	2
Urban	N	13	203	12	3
News	N	12	3048	7	2
Music	N	11	211	11	4
Automobile	M	3	106	10	2
HeartDisease	M	5	100	16	2
Arrhythmia	M	10	121	30	3
Internet	M	7	1264	10	4
Insurance	M	10	900	10	2
Covertime	M	5	116345	13	3
Synthetic-1	C	22	15491	18	4
Synthetic-2	C	30	24823	20	3
Synthetic-3	N	25	21990	20	2
Synthetic-4	N	35	19926	18	4
Synthetic-5	M	15	12020	15	2
Synthetic-6	M	20	10050	20	3

¹. Type: C = Categorical Data; N = Numeric Data; M = Mixed-Type Data

². NDC: Number of Datasets in Collection

³. ANIED: Average Number of Instances in Each Dataset

⁴. ANVED: Average Number of Variables in Each Dataset

⁵. ANVCTD: Average Number of Variables Co-existing in Two Datasets

5.2.2 Methods for Comparison

Predicting link strengths among variables observed in different data sources is a brand new topic. As such, it is impractical to directly compare this work with any existing work with respect to this novel problem. Instead, a set of existing link prediction methods based on graph topology are implemented for comparison. In each of the compared methods, each variable is regarded as a vertex in the graph, and the similarity score between two variables is interpreted as the weight of the assumed link for the corresponding pair of vertices. In particular, neighbour-based metrics including WJC, WRA, path-based metrics including LWP and RSS, random walk-based metric SR are selected to perform the comparison. The concept and formulation of these methods are described in Section 2.2.2. Specifically, for LWP metric, the decaying factor α is set to 0.01, as with the default value typically used when running this metric. For RSS metric, in order to guarantee that each pair of variables have at least one path connecting them, the path length for search is set to $k - 1$, where k denotes the number of datasets in the corpus. For SR, the decay factor γ is set to 0.8, as being widely used in various applications. Additionally, the number of iterations for SR is set to 20 in the present experimental evaluation.

5.2.3 Experimental Setup

In all experiments carried out, for each corpus of datasets, an n -fold cross validation [174] is performed, where n is the number of datasets in each corpus. In particular, the $n - 1$ folds of data form the training instances which are split by columns into subsets with selected variables being co-existed in two or more data subsets, whilst the testing dataset is a fold of data with all the variables under discussion included. The following reported results are based on an average of running 10 times n -fold cross validation.

It is important to mention that as with any real-world application, the ground truth of the link strengths between variables is not a natural existence in these datasets. Thus, in the following experiments, any “ground truth” is artificially computed by the testing data using the corresponding method as outlined in Section 5.1.3. At first glance, this may sound unintuitive but it provides a common ground for fair comparisons to be carried out.

Note that not all the approaches implemented for comparison may necessarily generate predicted results ranging between $[0, 1]$, and that normalising these results can only provide relative values for all the predicted links. Thus, simply obtaining normalised results may mis-

lead the interpretation of the computed link strength. A precision measure which calculates the percentage of correct predictions according to the portion of founded links is therefore, employed to articulate the predicting accuracy. In particular, for all unobserved links defined by the training datasets, their predicted link strengths are computed by each of the predicting algorithms and then ranked in descending order. Simultaneously, their corresponding "ground truth" (of the link strengths) are calculated through the testing datasets and ranked in descending order as well. The predicating accuracy is determined by comparing the number of the correct predictions against the scenario where a specific portion of unobserved links is assumed.

When conducting the experiments, for simplicity and clarity, *t-norm* and *t-conorm* are initially implemented with minimum and maximum operators, respectively. To reflect the flexibility of the proposed approach, and also to strengthen comparative studies, another type of operator combination, namely, algebraic product and bounded sum, are also applied to form the Bounded Sum-Algebraic Product (BSAP) interpretation. Additionally, the Centre of Gravity (COG) method is employed to perform in the defuzzification step.

5.2.4 Experimental Results

The experiment results are measured by predicting accuracy. That is, the number of correctly predicted results that are disclosed by each compared method, over that of retrieved variable pairs. In experimental running, all potential variable pairs are examined and ranked in descending order, and the top- K percent of the disclosed variable pairs are selected to compare against the "ground truth" (as indicated in Section 5.2.1). The predicting results revealed in this paper is simply based on the top-ranked variable pairs within the first 50% of them all. This is because the predicting accuracy generally retains an increasing trend in response to the increasing ratio of predicted links. When the number of predicted links reaches its maximum, meaning that all potential variable pairs are taken into account, the predicting accuracy will be 1. In practice, it is the highly ranked variable pairs that are generally more attractive and more useful.

5.2.4.1 Experimental Results for Real Data

Fig. 5.8 to Fig. 5.11 show the experimental results for the corpora of real-world datasets. These results jointly demonstrate that the proposed approach is generally very competitive under different circumstances. In particular, the proposed method consistently outperforms the neighbour-based metrics (WJC, WRA) and the random walk-based metric (SimRank)

across most corpora of datasets. Note that LWP and RSS metric can perform well for specific corpora of datasets. This has much to do with the distribution of the variables in each dataset within such a corpus. For instance, in the corpus of Salary datasets, most of the variables have an explicit relation with the variable 'salary', which is natural and makes it easy for the LWP metric to handle. This is also the case for the corpora of Student-Port, Student-Mat and Music datasets. In the corpus of Twitter datasets, almost each dataset is simply connected with only one another through a few number of overlapped variables, which is suitable for RSS, but may not fit for others such as neighbour-based metrics. However, the proposed method is still competitive according to the predicting accuracy, regardless of the variable distribution for each dataset corpus. This shows the robustness and adaptability of the present approach.

Another interesting finding is that for variable pairs with the “strongest” link strength, say, top 5% or top 10%, the proposed approach performs best among all compared methods to identify them. Although the predicting accuracy is not sufficiently high to meet human expectation, it is worth recognising that the task of identifying “strong” links is much more difficult than just finding whether there is a general link [109]. Such a detection is also of practical significance since in real-world applications, it is the identification of any variable pair that is associated with the most “Strong” link that is generally more attractive to the users.

Note that for the two distinct implementations of the proposed approach, using either Max-Min or BSAP interpretation, it is difficult to judge which one performs better. This may reflect the robustness of the underlying approach, but no theoretical proof for this hypothesis is done, which remains as active further research.

In comparison of APCC and MIC for measuring correlation between numeric variable pairs, MIC generally outperforms APCC for each of the listed predicting models, as the predicting accuracy based on MIC metric being higher than that of APCC, which demonstrates its advantage of capturing a variety of associations as relationship measurements.

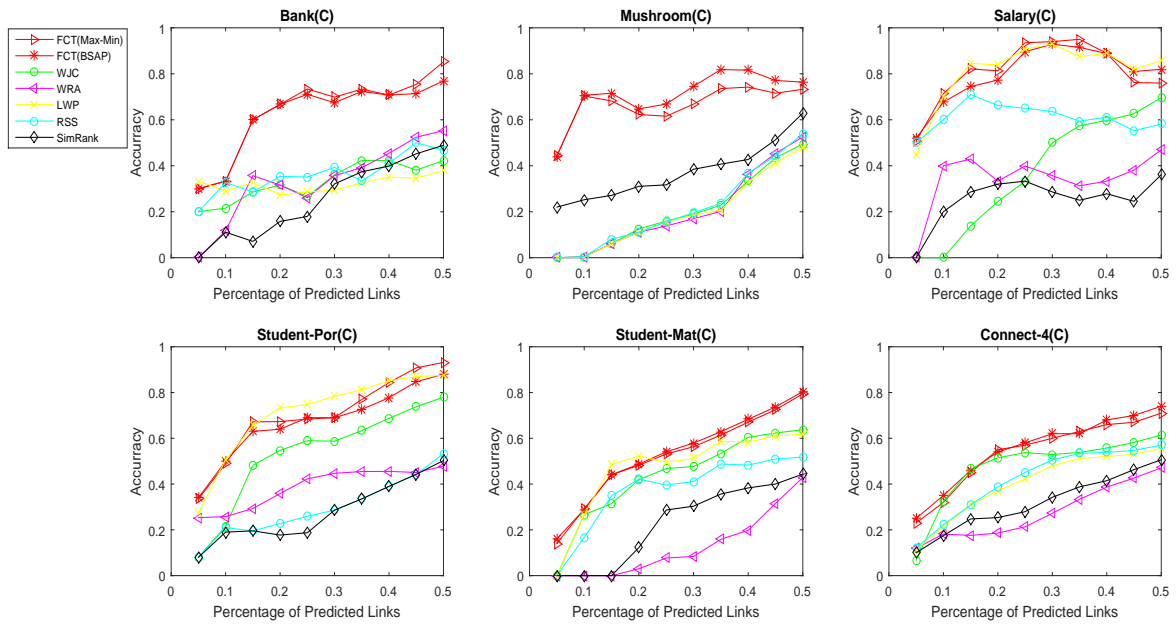


Fig. 5.8 Prediction Accuracy for Real-world Categorical Data

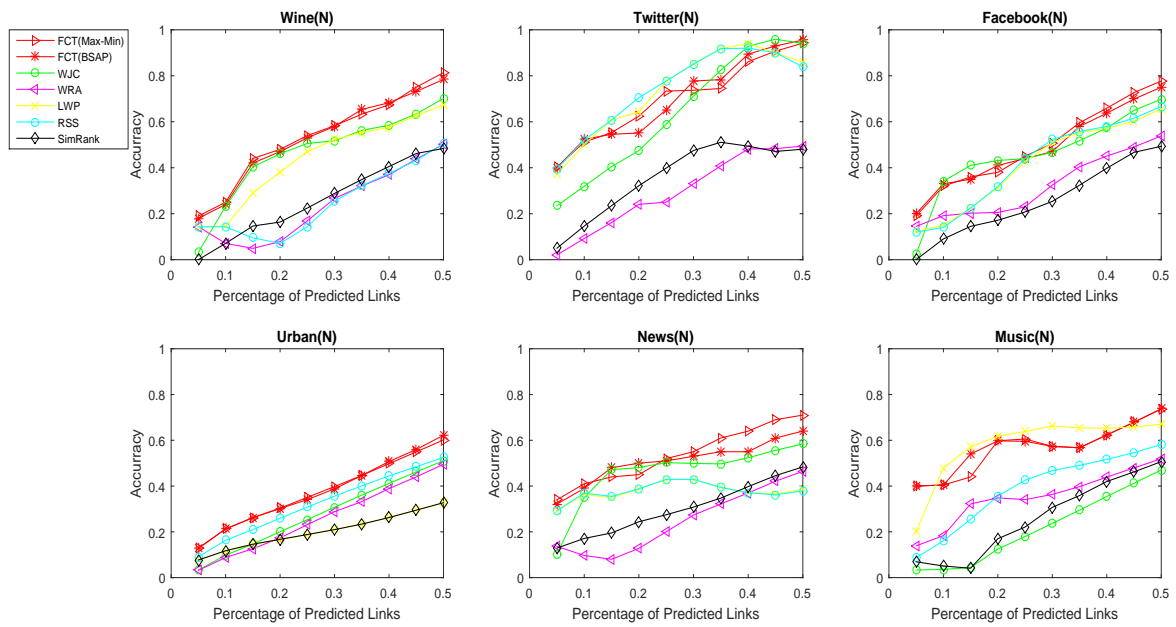


Fig. 5.9 Prediction Accuracy for Real-world Continuous Numeric Data by APCC

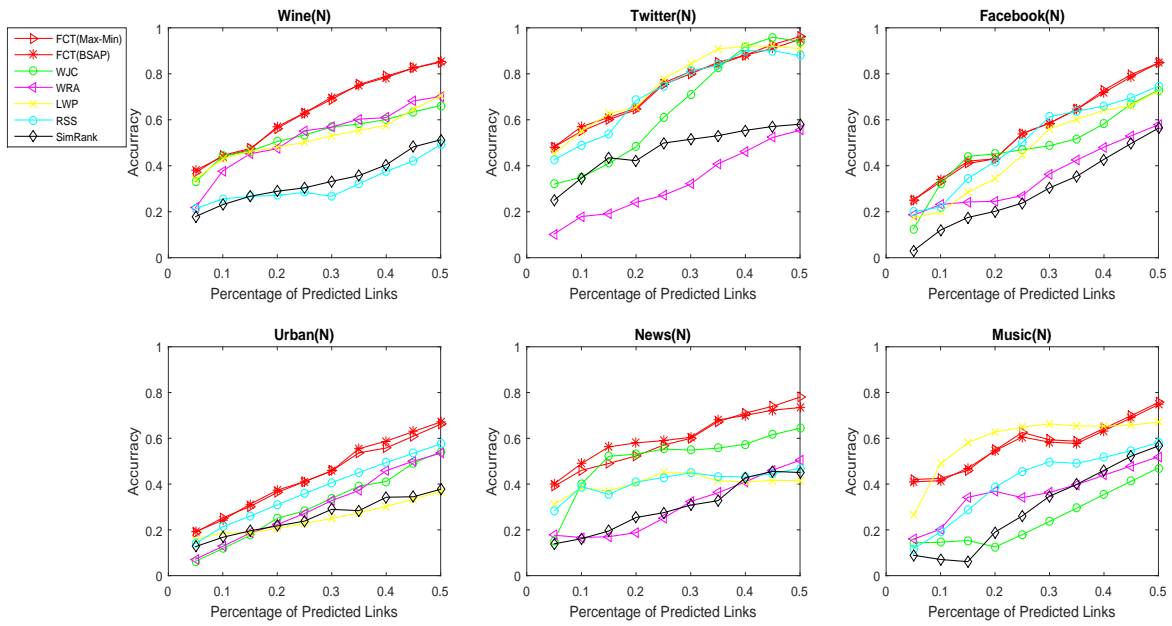


Fig. 5.10 Prediction Accuracy for Real-world Continuous Numeric Data by MIC

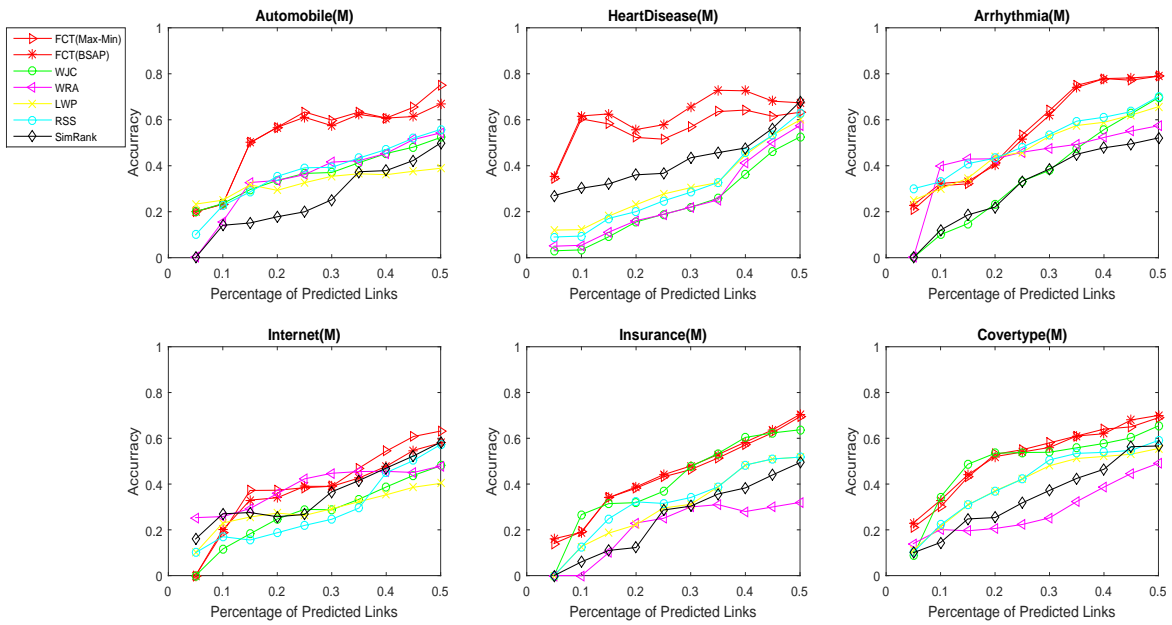


Fig. 5.11 Prediction Accuracy for Real-world Numeric-Categorical Mixed-Typed Data

5.2.4.2 Experimental Results for Synthetic Data

Another set of experimentations has been conducted on different corpora of synthetic datasets. The experimental results are shown in Fig. 5.12. It can be seen that the predicting accuracy for several of the compared methods declines to an extent with respect to the results shown earlier. Due to the increasing number of datasets in the corpus, certain incorporated datasets may not necessarily have common variables amongst them. This situation makes accurate prediction far more difficult.

The neighbour-based metrics suffer significantly from this condition. Both WJC and WRA lead to unsatisfactory results. This may be expected by the fact that neither of them is able to make integrated consideration of both nearest neighbour and non-nearest neighbour variables across different datasets. SR performs slightly better than neighbour-based metrics on larger corpus of datasets, as it does not take just common neighbour variables into account. The propagation of similarity scores amongst variable pairs in the entire dataset corpus could have a positive effect on predicting accuracy for larger sized dataset corpus. Interestingly, the performance of RSS has not been adversely much affected by the increased size of dataset collections, since it guarantees to find routes connecting a variable pair. Conversely, the performance of LWP drops dramatically, because it only takes paths of a length of 2 and 3 into consideration. Nevertheless, compared against all these, the proposed approach still performs better in most of the cases, illustrating once again the efficacy of utilising the transitivity property over the structure of connected-triples.

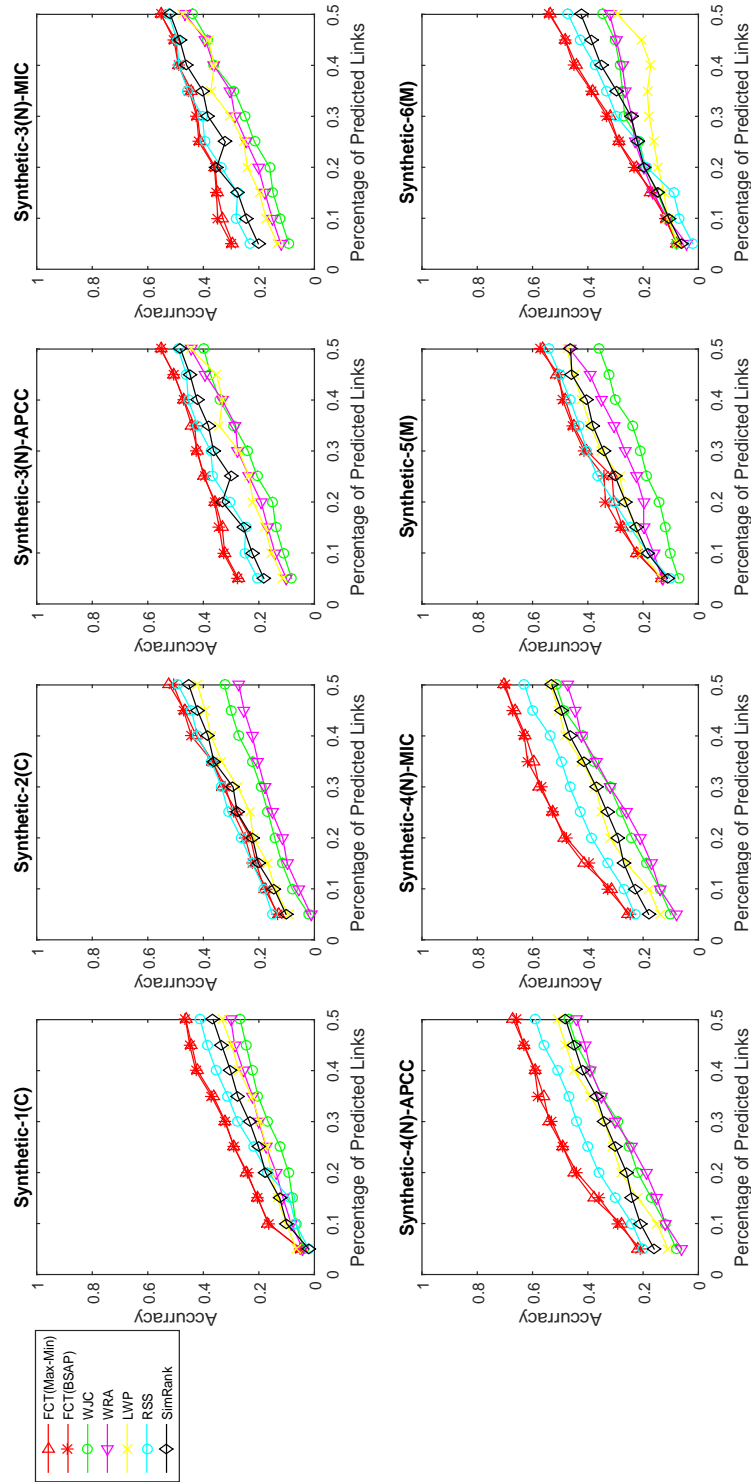


Fig. 5.12 Prediction Accuracy for Synthetic Data

Table 5.2 Analysis of Time Complexity

Method	Time Complexity
FCT	$O(k^2 p^2 q^2 l f)$
WJC	$O(k^2 p^2 q^2)$
WRA	$O(k^2 p^2 q^2)$
LWP	$O(k^3 p^3)$
RSS	$O(k^2 p^2 q^{e+1})$
SR	$O(k^2 p^2 q^2 r)$

^{1.} l : Number of linguistic terms to describe link strength

^{2.} f : Time for defuzzification

^{3.} e : Path length for RSS

^{4.} r : Number of iterations for SR

5.2.5 Complexity Analysis

In addition to evaluating these methods in terms of predicting accuracy, it is important to investigate the computational complexity that would determine their actual efficiency for real-world applications. Suppose that a particular corpus incorporates k datasets, that each dataset contains p variables on average, and that every two datasets share on average q identical variables. Table 5.2 shows the time complexity to find the correlation between all the potential variable pairs for each of the compared algorithms.

WJC and WRA are the most efficient metrics among the compared methods, SR is slightly more expensive than WJC and WRA, with required r iterations of refinement. Generally, the time complexity of path-based metrics is higher than that of neighbour-based methods. LWP has the cubical time complexity since it involves matrix multiplication. The time complexity for RSS is significantly affected by the length of paths, and can be extraordinarily high in extreme cases.

Although the proposed approach is not the most satisfactory in terms of time complexity due to the time expenses incurred by performing fuzzy inference, it is acceptable, especially when compared with the path-based metrics dealing with large corpus involving many datasets. Taking both predicting accuracy and time complexity into joint consideration, the proposed approach is considerably competitive upon most occasions.

5.3 Summary

This Chapter has presented a novel data-driven approach to predicting the connections between variables that are hidden in different datasets. Techniques for measuring correlation between domain variables of a certain corresponding type have been proposed. Assisted with the concept of fuzzy connected-triple, the relationships between distinct variables and their transitivity can be naturally captured, represented and reasoned through the link notation. The use of fuzzy inference supports the link prediction process to be more consistent with human reasoning, with the predicted results being readily interpretable. Experimental results on different corpora of datasets have shown that the proposed approach generates more accurate predicted outcomes, while involving relatively simple computation.

Chapter 6

Intelligent System for Variable Cluster Pattern Recognition

Pattern recognition, in the context of artificial intelligent, is defined as the automatic discovery of regularities in data through computational algorithms and making use of the recognised regularities to take actions on identifying the data into different categories [207]. Its process can be either “supervised”, where existing patterns can be obtained from a given data source, or “unsupervised”, where entire new patterns need to be discovered. Pattern recognition is widely applied in various fields, with frontier research domains including computer vision [208, 209], natural language processing [210, 211], activity detection [212, 213], bioinformatics [214, 215], security and privacy check [216, 217], automatic control [218, 219], medical diagnosis [220, 221], etc. Note that part of the above mentioned study areas may share overlaps with others.

However, till now, the study on pattern recognition mainly has focused on identifying a particular class for an observed data, or determining the underlying patterns for groups of items and activities, both of which are at entity or instance level. This results in the existing work not making full use of information provided by the data source, with little investigation into feature variables existing in the datasets. Yet, with the development of information technology, the real-world data grows rapidly, and it becomes more convenient to obtain data from various sources. Simultaneously, as indicated in Section 1.4, part of the acquired data may lack description to themselves, due to a variety of reasons, such as security consideration, equipment malfunction, ignorance from information collector, and inappropriate storage. For instance, when participating in projects with sensitive or confidential information (e.g., medical science, demographic and criminal research), a dataset is sometimes provided with its data description hidden on purpose for data security control. Another common scenario is

that the data dictionary with respect to a specific dataset may be missing during its life of storage, resulting in difficulty to handle those data. Such scenarios that involve no attribute name or label information hinder the understanding of data. For easy cross-referencing, the attribute (or feature) variables with the above mentioned characteristics are interchangeably termed as uninformed (unknown / unspecified) variables. These terms are alternatively used in latter sections and chapters of the thesis so as to avoid multiple times of repetition.

In this chapter, a brand new model working on feature variables is proposed on the basis of group analysis. As an initial attempt, it is aimed to learn knowledge of unknown feature variables from informed data sources. Essentially, to start with, for a specific research field of interest, the model performs the task of grouping informed feature variables from various given sources into a vast hierarchical structure, with each layer of the structure representing clusters of variables with analogous similarity or correlation degree. Likewise, for a dataset of the same or similar field with uninformed variables involved, the same operating procedure is conducted over the included data, producing in groups of uninformed variables organised in another hierarchical manner. Then, a group of uninformed variables within the constructed hierarchy is checked against the informed variable clusters from the previous established structure to detect its most similar counterpart. Such a modelling process can be regarded as a type of prediction or approximation as well. This predicting model also forms as a basis upon which to implement succeeded tasks of variable identification and knowledge reconstruction, which will be discussed in Chapter 7.

The remainder of this chapter is arranged as follows: Section 6.1 outlines the basic structure of the proposed model, with a brief description of its involving components. Section 6.2 describes the implementation of this novel predicting model, with distinct strategies to handle different data types specified in detail. Section 6.3 shows the experimental evaluation results, supported by comparative studies with other methods modified to suit the current task of cluster recognition. Finally, Section 6.4 concludes the chapter.

6.1 Conceptual Framework of Intelligent System

The conceptual framework of the proposed intelligent feature variable cluster predicting system is illustrated in Fig. 6.1, which contains three associative components: link extraction subsystem, variable clustering subsystem and cluster recognition subsystem. Given a corpus of M datasets from a source domain (grouped and highlighted with dashed circle) and a single testing dataset from the same or similar domain with uninformed feature variables, the

process of performing group estimation on the uninformed feature variables is accomplished by executing each of the subsystems consecutively. The fundamental functionalities for each of the subsystems is described as follows:

- **Link Extraction Subsystem:** Extract both existed and potential links between each pair of feature variables from the domain data corpus. Determine the similarity degree for a pair of feature variables by measuring its link weight. Appropriate link measurement is required to be adopted according to the type of the feature variables involved. Similar steps are also conducted to testing datasets with uninformed variables. The conceptual structure and concrete implementation of this subsystem has been detailed in Chapter 5.
- **Variable Clustering Subsystem:** For an informed data corpus, group all the feature variables into clusters according to their similarity degree acquired from link extraction subsystem, so that the feature variables with a higher link weight between them are gathered together, and the variable pair with a lower link weight are separated into different categories. Likewise, the same action is required to perform on the testing dataset to obtain another set of clusters with uninformed variables. Generally, the discussed subsystem generates two distinct and separate variable cluster structures, with pairs of the involving clusters coming from each of the cluster structure being examined in the next step.
- **Cluster Recognition Subsystem:** Select each of the uninformed variable clusters from the cluster set obtained from the previous clustering step. This subsystem is devised with an attempt to identify any uninformed cluster by comparing it to the existing informed variable clusters. This step may result in two opposite consequences: (1) the uninformed variable cluster is clearly or approximately identified; and (2) there exists no informed variable cluster matching this uninformed counterpart appropriately.

The contents of the variable clustering subsystem and cluster matching subsystem (highlighted in the red dashed box in Fig. 6.1) are within the scope of this chapter, with the implementation for each of the two subsystems being specified in Section 6.2. As described above, variable cluster pattern recognition may lead to different conclusions. Hence, corresponding means to handle respective circumstances are necessary. When facing scenario (1), it is insightful and understandable to perform a further search process for determining specific labels(names) for the uninformed variables within the cluster. However, when facing scenario (2), such an ambitious search may lead to unreliable result. Instead, it may be sensible to assume that the given unknown variable cluster is not presented in the current level of

understanding of the domain feature variables. Thus, an immediate update for the existing variable structure may offer a better option. Therefore, succeeding subsystems: variable recognition subsystem and knowledge reorganisation subsystem are proposed accordingly to deal with such scenarios, which will be discussed in Chapter 7.

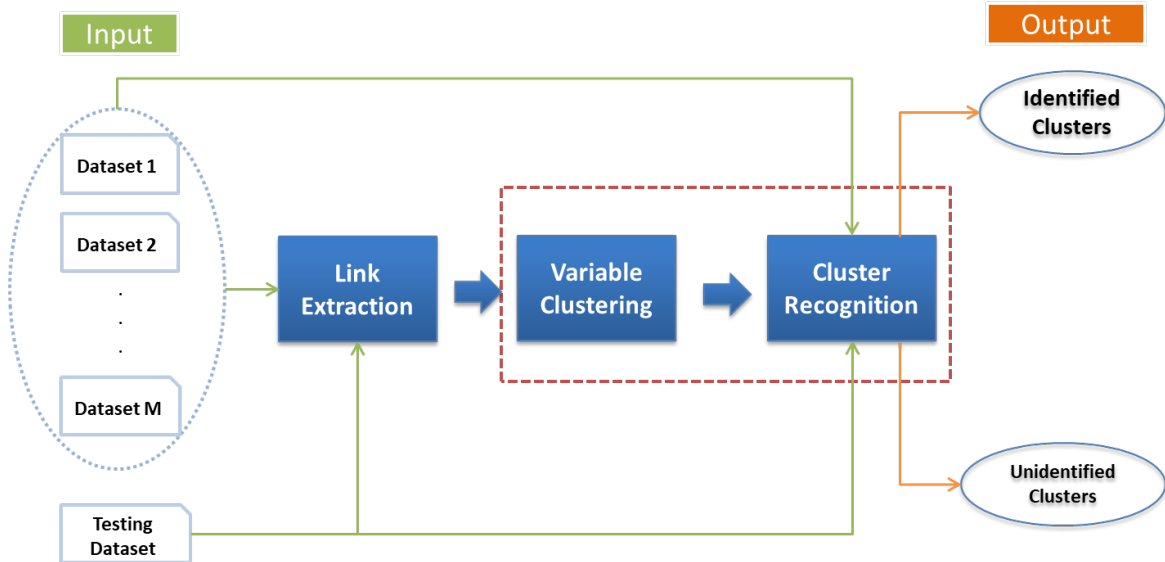


Fig. 6.1 Conceptual Structure of Feature Variable Cluster Recognition Model

6.2 Implementation of Intelligent System

This section describes the implementation of the proposed framework for uninformed variable cluster recognition. It specifies the system's components and their associated time complexity analyses.

6.2.1 Variable Clustering Subsystem

By performing the functionality of link extraction subsystem, the link pattern for a pair of feature variables coexisting in a single dataset can be measured in a straightforward manner. Also, the link pattern between a pair of variables which does not co-appear in a single dataset can be inferred automatically. Thus, the link strength for all pairs of variables within a specific corpus of datasets (each pair of variables can be either coexist or non-coexist in a single dataset) can be obtained. Having achieved this, the next step is to construct a structure to accommodate all the feature variables under consideration.

Clustering analysis, being widely used at the initial stage of exploring the underlying patterns of data, is adopted here to investigate the structural layout of the feature variables. It aims at categorising these feature variables into different groups according to their similarity. It is important to mention that not all of the clustering techniques are suitable for the current circumstance, since the similarity between pairs of feature variables are not always measured by the proximity of their respective attributes, but by link weights that are generated or inferred from statistical calculation on the given data. Having noticed this premise, only clustering methods built upon the connectivity-based similarity measure are considered as appropriate options for the current situation.

Specifically, hierarchical clustering is selected to perform the task here, since it provides the flexibility to demonstrate the hierarchical structure of the variable layout for such scenarios. Moreover, from practical point of view, the existing link strength can be naturally employed as a similarity (or dissimilarity) measure during the clustering process. In general, hierarchical clustering can be divided into two main types: agglomerative and divisive. For the present work, agglomerative clustering implements a “bottom-up” approach which starts from a singleton feature variable cluster (a cluster containing only one feature variable), and continuously merges feature variable clusters into macro clusters and finally generates a single feature variable cluster encapsulating all the variables in the data set corpus. The time complexity for such an agglomerative clustering algorithm can be as high as $O(N^3)$, since exhaustive scan of the $N \times N$ similarity matrix for the largest similarity has been executed in each of the $N - 1$ iterations, where N denotes the number of variables within the whole data set corpus. An improved version assisted with priority-queue has been proposed to increase its efficiency to $O(N^2 \log N)$ [222]. The pseudo code of this efficient agglomerative clustering algorithm (EACA) for grouping variables is shown in Algorithm 7.

Algorithm 7: Efficient Agglomerative Clustering Algorithm for Feature Variables**Input:** $V = \{v_1, v_2, \dots, v_N\}$: A variable set of N variables M : A similarity matrix for all the pairs of variables

```

1  $\mathcal{A} \leftarrow \emptyset$   $\triangleright$  Active set starts out empty
2 for  $i \leftarrow 1$  to  $N$  do
3    $\mathcal{A} \leftarrow \mathcal{A} \cup \{\{v_i\}\}$ ;
4    $I\{i\} = 1$   $\triangleright$   $I$  indicates clusters available to be merged
5    $P[i] \leftarrow$  Priority queue for  $M[i]$  sorted on similarity  $\triangleright P[i]$ : A priority queue
      storing variables with their respective similarity degrees to  $v_i$  in descending order
6    $P[i].DELETE (M[i][i])$   $\triangleright$  No need of self-similarity
7 end
8  $\mathcal{T} \leftarrow \mathcal{A}$   $\triangleright$  Store the tree as a sequence of merges
9 while  $|\mathcal{A}| > 1$  do
10   $k_1 \leftarrow \arg \max_{k: I[k]=1} P[k].max()$   $\triangleright$  Get index of cluster with highest similarity degree from
      all available priority queues
11   $k_2 \leftarrow P[k_1].max().index()$   $\triangleright$  Get index of cluster with highest similarity degree to
      cluster that is indicated by  $k_1$ 
12   $\mathcal{A} \leftarrow ((\mathcal{A} \setminus \{C_{k_1}\}) \setminus \{C_{k_2}\}) \cup \{C_{k_1} \cup C_{k_2}\}$   $\triangleright$  Remove sets (clusters)  $C_{k_1}$  and  $C_{k_2}$ 
      from  $\mathcal{A}$ , add union set of  $C_{k_1}$  and  $C_{k_2}$  into  $\mathcal{A}$ .  $C_{k_1}, C_{k_2}$ : Sets (clusters) of variables
      regarding  $k_1$  and  $k_2$ 
13   $\mathcal{T} \leftarrow \mathcal{T} \cup \{C_{k_1} \cup C_{k_2}\}$   $\triangleright$  Add  $C_{k_1} \cup C_{k_2}$  as a merging step to  $\mathcal{T}$ 
14   $I[k_2] \leftarrow 0$ ;
15   $P[k_1] \leftarrow \emptyset$ ;
16  for each  $i$  with  $I[i] = 1$  and  $i \neq k_1$  do
17     $P[i].DELETE (M[i][k_1])$ ;
18     $P[i].DELETE (M[i][k_2])$ ;
19     $M[i][k_1] \leftarrow$  UPDATED sim value;
20     $P[i].INSERT (M[i][k_1])$ ;
21     $M[k_1][i] \leftarrow$  UPDATED sim value;
22     $P[k_1].INSERT (M[k_1][i])$ ;
23  end
24 end
Output:  $\mathcal{T}$ 

```

Conversely, divisive clustering is a “top-down” method which starts with all feature variables in a single cluster, then considering every possible way to divide the cluster into two and choose the best decision. This step is performed recursively as the hierarchy going down. Intuitively, the divisive clustering method can be computationally expensive, since there exists $O(2^N)$ possibilities to split data into two clusters, where N denotes the number of feature variables under investigation. However, heuristics can be employed to guide such splitting step and reduce its executional complexity. Fortunately, divisive clustering can be made much more efficient if it is unnecessary to generate a complete hierarchy all the way down to individual variable leaves. Hence, divisive clustering is an appropriate option for the scenario in which the presentation of preliminary hierarchical structure is merely needed. The pseudocode of this efficient divisive clustering algorithm (EDCA) for the present task is shown in Algorithm 8.

Algorithm 8: Efficient Divisive Clustering Algorithm for Feature Variables**Input:** $V = \{v_1, v_2, \dots, v_N\}$: A variable set of N variables M : A similarity matrix for all the pairs of variables K : Number of clusters

```

1  $\mathcal{T} \leftarrow \emptyset$    ▷ Store the tree as a sequence of divided clusters
2  $Q \leftarrow V$      ▷ A priority queue for variable clusters with intra-cluster similarity in ascending
   order
3 while  $Q.LENGTH() < K$  do
4    $\mathcal{C} \leftarrow Q.REMOVE()$  ;
5    $\mathcal{T} \leftarrow \mathcal{T} \cup \{\mathcal{C}\}$  ;
6    $\mathcal{C}_s \leftarrow \emptyset$  ;
7    $\mathcal{C}_s \leftarrow \mathcal{C}_s \cup \{v_s\}$    ▷  $v_s$ : Variable with lowest within-cluster similarity in  $\mathcal{C}$ 
8    $\mathcal{C} \leftarrow \mathcal{C} \setminus \{v_s\}$  ;
9   for each  $v_i \in \mathcal{C}$  do
10    for each  $v_j \in \mathcal{C}$  do
11      COMPUTE average similarity to splinter cluster, named  $sim_j^s$ ;
12      COMPUTE average similarity to current cluster, named  $sim_j^c$ ;
13       $d_j \leftarrow sim_j^s - sim_j^c$  ;
14    end
15    FIND  $v_j \in \mathcal{C}$  with maximum  $d_j$ , named  $v'$  ;
16    if  $sim_{v'}^s > sim_{v'}^c$  then
17       $\mathcal{C}_s \leftarrow \mathcal{C}_s \cup \{v'\}$  ;
18       $\mathcal{C} \leftarrow \mathcal{C} \setminus \{v'\}$  ;
19    end
20  end
21   $\mathcal{T} \leftarrow \mathcal{T} \cup \{\{\mathcal{C}\}\} \cup \{\{\mathcal{C}_s\}\}$  ;
22   $Q.INSERT(\mathcal{C})$ ;
23   $Q.INSERT(\mathcal{C}_s)$ ;
24 end

```

Output: \mathcal{T}

The time complexity of this efficient divisive clustering algorithm is $O(KN^2)$. Since K is usually determined in advance and K is not set to be a large number for most circumstances, the time complexity of this algorithm can be simply denoted as $O(N^2)$.

Note that the variable clustering subsystem works for both the informed data corpus and the uninformed dataset, generating two hierarchical tree structure representing the knowledge for the configurational layout of the variables involved for each of the data collections.

6.2.2 Cluster Recognition Subsystem

Considering a corpus of datasets meeting the presumption (every two of the included datasets having at least one common feature variable existing in both of them), a hierarchical structure of the feature variables involved can be acquired by executing the link extraction and variable clustering steps. Similarly, when encountering a dataset with unspecified variables, the same clustering approach can also be employed to generate a dendrogram representing the hierarchy of these “uninformed” feature variables. In order to obtain particular clusters from the cluster tree, two distinct directions of acquiring feature variable clusters are addressed here:

- 1) Determination for the number of clusters. This usually results in a small quantity of macro clusters. In many situations, macro clusters achieve more attention than the micro ones, since macro clusters can capture more general features in common. For agglomerative clustering, such clusters can be obtained during the final iterations of its clustering process or by back tracking when the complete hierarchical tree has been constructed. In addition, the efficient divisive clustering method can return a pre-specified number of clusters automatically.
- 2) Threshold cut. There may be circumstances where only feature variable clusters with low intra-variance are concerned, thus, a threshold value (typically not too high) can be intuitively set to cut the hierarchical tree \mathcal{T} into subtrees where each subtree represents a particular feature variable cluster meeting the selection criterion. Performing threshold cut may result in a rather large number of micro clusters.

Each time, one of the above mentioned selecting methods can be applied to both the hierarchical clustering trees generated from the informed and uninformed feature variables to obtain a set of model clusters and a set of target clusters, respectively. Having attained such clusters, the next stage is to match them according to their similarity levels from different identifiable aspects. To address this important issue, distinct approaches have been proposed to perform such matching steps with respect to different types of feature variable clusters based on their intrinsic characteristics, as presented below in Section 6.2.2.1 to Section 6.2.2.3. Note that the term “match” used here does not possess its traditional meaning of finding the exact same pattern, but to look for the informed variable cluster which is most similar to the target.

6.2.2.1 Numeric Feature Variable Cluster Recognition

For single continuous numeric variable, it is natural to employ the distribution of its values in the data set as features to describe itself. However, it is time consuming to exploit all of its values at every point to articulate the variable, especially when the size of the dataset is large. Thus, in order to keep the balance between accurate sampling and computation cost, a concise pattern to sample representative variable values is proposed. Particularly, for the present implementation, the 1st to the 9th deciles are used together with the minimum and maximum of the variable values to describe a singleton variable. Therefore, the features of a singleton variable can be presented as a vector, as its format shown in Fig.6.2. An example of the feature vector of a singleton variable is shown in Fig.6.3. However, the representative feature values for a variable can be sampled according to different criteria for different situations if preferred.

Minimum	Decile 1	Decile 2	Decile 8	Decile 9	Maximum
---------	----------	----------	-------	----------	----------	---------

Fig. 6.2 A Sampled Feature Vector for Numeric Variable

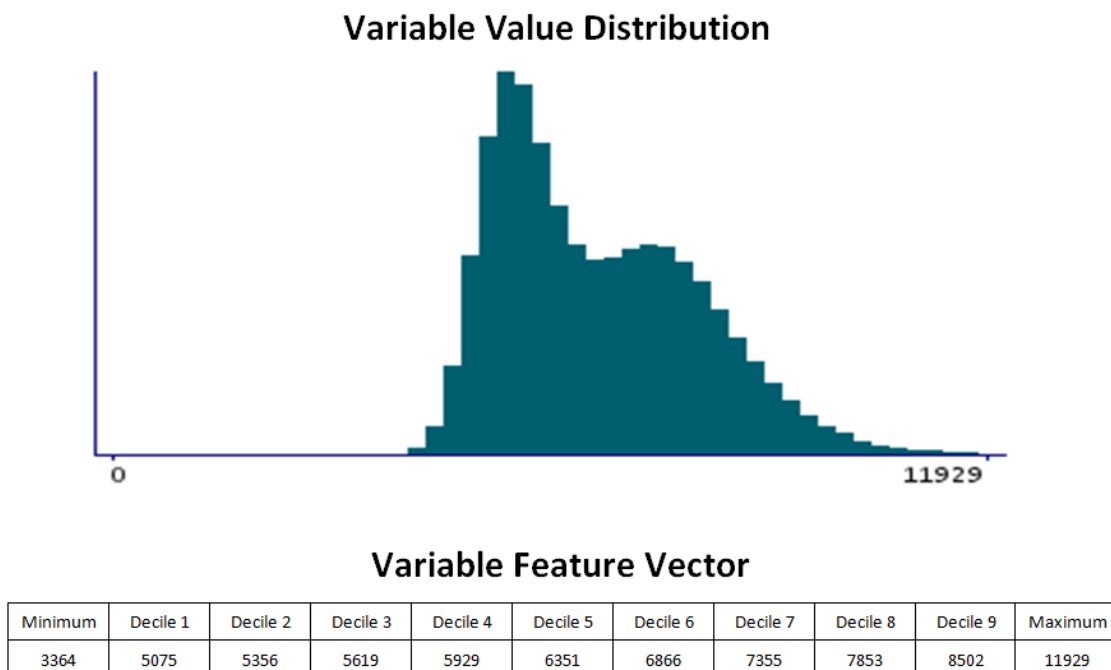


Fig. 6.3 Example of Feature Vector for a Singleton Variable

For two numeric variable clusters, their similarity degree can be measured by comparing their cores. The core for a numeric variable cluster is defined as a vector of the same type to the feature vector for singleton variables, with each vector element containing the average of feature vectors for all the variables within the cluster in the corresponding value field. Namely, a variable cluster can be recognised as a hyper-variable, with the core being its feature vector. Since the values of variables within a cluster can be in general of various ranges, a standardisation preprocessing step is needed for all the variables in advance. A detailed sequential stages for determining the core of a numeric variable cluster is stated as follows:

1. Standardise all the feature variables within the cluster by 0-mean and 1-standard deviation;
2. For each variable within the cluster, sample minimum, maximum and 1st to 9th deciles of its variable values from its corresponding data set, form a feature vector with each vector element containing a value in ascending order;
3. Generate a feature vector of the same size as a core for the variable cluster, fill in each vector element with the weighted average of corresponding values in feature vectors for all the variables within the cluster. Let $V_{\mathcal{C}}^i$ be the i th element for the core vector of Cluster \mathcal{C} , the weighted average value for $V_{\mathcal{C}}^i$ can be calculated as follows:

$$V_{\mathcal{C}}^i = \frac{\sum_{v_j \in \mathcal{C}} V_{v_j}^i \cdot n_j}{\sum_{v_j \in \mathcal{C}} n_j} \quad (6.1)$$

where $V_{v_j}^i$ represents the i th element of the core vector for the variable v_j , and n_j denotes the number of instances in the data corpus under consideration which have a certain value for v_j . The weighted average calculation has the merit of considering the imbalance of dataset size in the examined data corpus (as a data corpus may contain datasets with various numbers of instances).

From the above, given two numeric variable clusters: one cluster including informed variables, the other with uninformed ones, having determined the cores for each of them, the task of matching two clusters is simplified as finding the similarity of these two cores. In practice, distance-based similarity measure is adopted to implement. Namely, the similarity degree for two cluster \mathcal{C}_1 and \mathcal{C}_2 can be defined as:

$$\text{sim}(\mathcal{C}_1, \mathcal{C}_2) = \exp\left(\frac{-\|V_{\mathcal{C}_1} - V_{\mathcal{C}_2}\|}{2}\right) \quad (6.2)$$

where V_{C_1} and V_{C_2} represent the core vectors for C_1 and C_2 , respectively; and $\|V_{C_1} - V_{C_2}\|^2$ denotes the Euclidean distance between V_{C_1} and V_{C_2} .

6.2.2.2 Categorical Feature Variable Cluster Recognition

Apart from numeric variable clusters, it is uneasy to find such statistical information (e.g., deciles, quartiles, averages) for the categorical counterpart and generate a core for the cluster. Particularly, two categorical variable clusters may each include a different number of variables (which is also the case for matching numeric variable clusters), and categorical variables within a cluster may have a different number of possible terms (values), and these terms may either be nominal or ordinal. Combination of these factors makes it a great challenge for the aggregation task. More sophisticatedly, there are often occasions that two variables appearing from different clusters may include similar or even exactly the same variable, but have different term labels in each of the datasets. This makes it impractical to identify them by examining their explicit values. Having noticed the difficulties in tackling with categorical variable clusters, the research focus has been diverted into looking for combinational term distribution information for all the variables within the cluster. A technique aimed at matching categorical feature variable clusters through evolutionary heuristics has been proposed here. The general recognition procedure for variable clusters is established as follows:

1. For each variable within the model (informed) cluster C_m , find the frequency of occurrence for all of its included variable terms, arrange these terms in descending order according to the frequency of occurrence.
2. Create a table (matrix) for C_m , named T_m , with each row in T_m storing an in-cluster variable with the information obtained from Step 1).
3. For the undetected (uninformed) cluster C_n , perform the same as described in Step 1 and Step 2. with the resulting table denoted by T_n .
4. Expand T_m and T_n to the equal size of $N_v \times N_t$, where N_v represents the number of variables for the larger sized cluster between C_m and C_n ; and $N_t = \max\{N_t^m, N_t^n\}$, where N_t^m and N_t^n each denotes the number of terms for variable in C_m and C_n with the largest number of variable terms.
5. For each row in T_m and T_n , fill in the empty element with value 0.
6. For the table of T_m and T_n with empty rows, fill in each empty row with value 0 for all of its elements.

7. Perform an evolutionary algorithm (EA) to shuffle the row vectors within T_n in order to match T_m as much as possible.

An illustration of swapping row vectors in the target table is shown in Fig. 6.4. Taking variable V_A in model table as an example, its involving variable terms A_1, A_2, A_3, A_4 with their respective frequency of occurrence in dataset is arranged in descending order to form a variable term distribution row vector. Elements 0 in both tables are complemented according to Step 5, as described above. It is clear that the row vectors of term distribution information for variables v_b and v_y (highlighted in yellow) in the target table is expected to be swapped in order to better match with the existing model table.

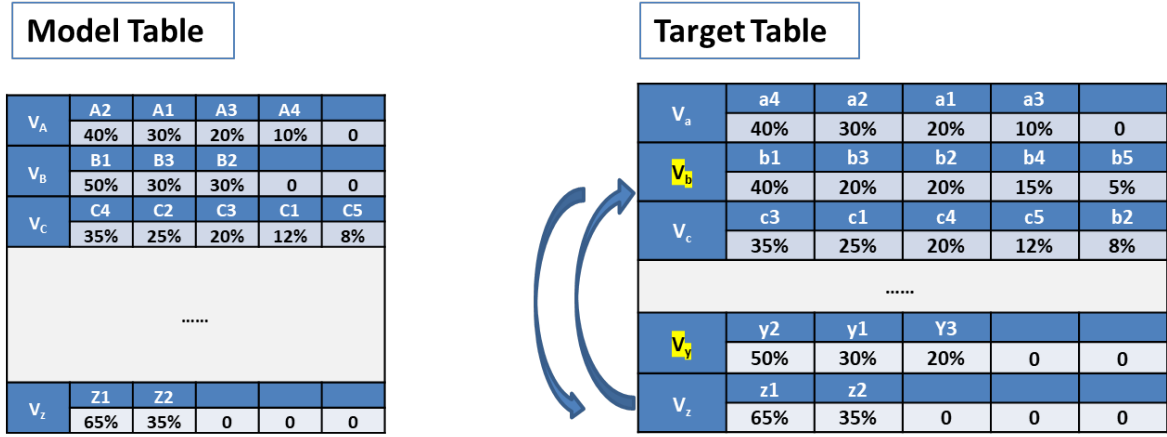


Fig. 6.4 Example of Swapping Row Vectors in Target Table

The implementation of the evolutionary heuristics in Step 7 is detailed in Algorithm 9. In particular, the similarity degree between cluster C_m and C_n (table T_m and T_n) can be calculated by the following formula:

$$\text{sim}(C_m, C_n) = \text{sim}(T_m, T_n) = \exp\left(\frac{-\text{DIST}(T_m, T_n)}{2}\right) \quad (6.3)$$

in which $\text{DIST}(T_m, T_n)$ denotes the distance between T_m and T_n , and can be computed by

$$\text{DIST}(T_m, T_n) = \left(\sum_{i=1}^{N_y} \sum_{j=1}^{N_t} \frac{|T_m^{F_{ij}} - T_n^{F_{ij}}|^2}{N_t} \right)^{\frac{1}{2}} \quad (6.4)$$

where $T_m^{F_{ij}}$ and $T_n^{F_{ij}}$ each represents the frequency of occurrence for the j th term in the i th variable within T_m and T_n , respectively.

Algorithm 9: Evolutionary Algorithm to Shuffle Variable Term Distribution Vectors**Input:***maxIter*: Maximum number of iterations*popSize*: Population size of solutions δ : Threshold to keep specific percentage best solutions from previous iteration λ : Threshold to perform shuffle step*L*: List to store best possible solutions in descending order

```

1 for  $i \leftarrow 1$  to popSize do
2   Randomly initialise possible solution  $S_i$  for  $T_n$ ;
3   L.INSERT( $S_i$ ) ;
4 end
5 iter  $\leftarrow 0$  ;
6 while iter < maxIter do
7   for  $i \leftarrow L.SIZE() \times \delta + 1$  to L.SIZE() do
8      $L'$ .INSERT( $S_i$ )  $\triangleright L'$ : Another List to store solutions
9   end
10  while L.SIZE() < popSize do
11    for  $j \leftarrow 1$  to  $L'$ .SIZE() do
12      if RANDOM() >  $\lambda$  then
13         $L'_j$ .SWAP( $p, q$ ) ;  $\triangleright$  swap variables with position  $p$  and  $q$  in  $L'_j$ ,  $p$  and  $q$  are
14          randomised in advance
15        COMPUTE sim( $T_m, L'_j$ ) ;
16      end
17     $L$ .INSERT( $L'_j$ ) ;
18  end
19  L.SORT() ;
20  iter  $\leftarrow iter + 1$  ;
21 end
Output: L.GETFIRST()

```

The time complexity for the proposed EA is $O(kl^2)$, with k and l each denoting the maximum number of iterations and the size of the population in EA prespecified for performing the refinement steps, respectively.

6.2.2.3 Mixed-Type Feature Variable Cluster Recognition

For pattern recognition of a mixed-type variable cluster, since the numeric feature variables in the cluster have already been transformed into its categorical counterpart, the task can be conducted through the same procedure as performing categorical feature variable cluster matching. The process of approximating a numeric variable as a categorical output is described in Section 5.1.3.3.

6.3 Experimental Evaluation

6.3.1 Data Preparation

To evaluate the performance of the proposed model, the experimental test is carried out on nine real world data from UCI benchmark datasets [206]. Similar to the data preprocessing steps discussed in Section 5.2.1, each dataset is separated into training part and testing part at first. Then the training part is segmented again into subsets with overlapped variables amongst them according to human knowledge. The details of these data corpora are summarised and shown in Table 6.1.

Table 6.1 Summary of Datasets: Variable Cluster Recognition

Dataset Collection	Type	NDC	ANIED	ANVED	ANVCTD
Wine	N	4	1256	3	1
Twitter	N	7	82038	13	4
News	N	12	3048	7	2
Bank	C	7	6459	4	2
Salary	C	6	5028	6	1
Student-Por	C	4	163	8	3
Automobile	M	3	106	10	2
Internet	M	7	1264	10	4
HeartDisease	M	5	100	16	2

¹. Type: C = Categorical Data; N = Numeric Data; M = Mixed-Type Data

². NDC: Number of Datasets in Collection

³. ANIED: Average Number of Instances in Each Dataset

⁴. ANVED: Average Number of Variables in Each Dataset

⁵. ANVCTD: Average Number of Variables Co-existing in Two Datasets

6.3.2 Experimental Setup and Results

Recall that the recognition of uninformed feature variable clusters involves two consecutive steps: variable clustering and variable cluster identification. Although the output of the entire predicting system is intended to be the matching result between an unspecified variable cluster to the certain specified counterparts, the quality of the clusters generated by the preliminary step will have a direct impact on the subsequent one. Hence, empirical studies on both the variable clustering subsystem and the cluster recognition subsystem have been conducted. The results are reported and discussed below.

6.3.2.1 Experimentation on Feature Variable Clustering

In order to gauge variable clustering quality, the average of Silhouette Coefficient (ASC) [223] is adopted, due to its comprehensive consideration of both cohesion and separation degrees of resultant clusters without referring to the ground truth. This method is applicable to the current situation where no prior classification (group) knowledge about the examined feature variables is available. Mathematically, the formula for calculating Silhouette Coefficient (SC) for a single variable v_i is defined as follows:

$$SC(v_i) = \frac{b(v_i) - a(v_i)}{\max\{a(v_i), b(v_i)\}} \quad (6.5)$$

where $a(v_i)$ represents the average distance between v_i to every other feature variable within the same cluster, $b(v_i)$ denotes the smallest average distance of v_i to all the feature variables in any other cluster, of which v_i is not a member. Thus, let D be a set of all the feature variables within a data corpus with $|D|$ denoting the number of feature variables in D , the ASC for a set of clusters S generated by a clustering algorithm can be formulated as:

$$ASC(S) = \frac{1}{|D|} \sum_{v_i \in D} SC(v_i) \quad (6.6)$$

It is clear from the above definition that $ASC(S) \in [-1, 1]$, with result asymptotic to 1 representing better performance of the clustering approach.

For EACA, the average-linkage (AL) measured by APCC is used to guide the clustering steps. For comparison, spectral clustering, an advanced technique based on graph partitioning which also employs connectivity-based similarity amongst data, is included as a baseline clustering method. However, it does not provide the flexibility to generate different numbers of clusters at one time. The number of feature variable clusters for each data corpus or dataset is estimated by human expert according to their domain knowledge, although the exact

number of groups (clusters) is confirmed. Thus, three adjacent integers which are most likely to outline the categorical information of the discussed feature variables are examined, respectively. Paired t-tests are carried out to validate the significance of the experimental results.

The resultant ASC for the training data corpus and that for the testing dataset are illustrated in Table 6.2 and Table 6.3, with each of the figures shown based on an average over 10 times of n -fold cross validation, where n denotes the number of datasets in the data corpus.

The experimental results show that each of the clustering algorithms generates the best variable structure several times, revealing that the proposed models achieves at least competitive performance compared to the spectral clustering approach in considering both intra-cluster compactness and inter-variable variance. Notably, EACA generate best results more times than the other two approaches, and EDCA performs well for the situation where the number of clusters is not required to be large. Moreover, performing clustering algorithms on training data corpora generally yields a smaller ASC than those performing on testing dataset. This indicates that valuable information between pairs of variables coming from different datasets may not be fully captured during the previous step of link prediction, possibly because the overlapped feature variables in data corpus may not be sufficiently informative for inferencing the correlation between variables coming from different involved datasets.

Table 6.2 Comparison of ASC for Different Clustering Models on Data Corpus

Data Collection	Number of Clusters	EACA	EDCA	Spectral Clustering
Wine	2	0.234 ± 0.012 (v)	0.264 ± 0.027 (v)	0.212 ± 0.014
	3	0.438 ± 0.026 (*)	0.402 ± 0.028 (*)	0.502 ± 0.016
	4	0.274 ± 0.021	0.231 ± 0.034	0.020 ± 0.008
Twitter	6	0.632 ± 0.023	0.605 ± 0.032 (*)	0.648 ± 0.006
	7	0.543 ± 0.025 (v)	0.516 ± 0.036 (v)	0.472 ± 0.004
	8	0.476 ± 0.013 (v)	0.423 ± 0.024	0.412 ± 0.008
News	5	0.383 ± 0.012	0.312 ± 0.023 (*)	0.392 ± 0.017
	6	0.463 ± 0.025 (*)	0.445 ± 0.039 (*)	0.491 ± 0.011
	7	0.512 ± 0.013 (*)	0.491 ± 0.022 (*)	0.554 ± 0.012
Bank	3	0.416 ± 0.012 (v)	0.483 ± 0.033 (v)	0.385 ± 0.026
	4	0.582 ± 0.025 (v)	0.627 ± 0.029 (v)	0.516 ± 0.013
	5	0.503 ± 0.027 (v)	0.483 ± 0.035 (v)	0.431 ± 0.009
Salary	3	0.473 ± 0.019 (*)	0.514 ± 0.024 (*)	0.552 ± 0.008
	4	0.612 ± 0.015 (*)	0.593 ± 0.019 (*)	0.641 ± 0.011
	5	0.677 ± 0.016 (v)	0.552 ± 0.028 (*)	0.623 ± 0.022
Student-Por	4	0.361 ± 0.027 (*)	0.325 ± 0.034 (*)	0.412 ± 0.023
	5	0.452 ± 0.033	0.408 ± 0.025 (*)	0.467 ± 0.016
	6	0.323 ± 0.028	0.288 ± 0.034 (*)	0.331 ± 0.029
Automobile	4	0.523 ± 0.009 (v)	0.556 ± 0.011 (v)	0.489 ± 0.014
	5	0.592 ± 0.021 (v)	0.581 ± 0.013 (v)	0.548 ± 0.011
	6	0.464 ± 0.017 (v)	0.492 ± 0.022 (v)	0.416 ± 0.015
Internet	7	0.313 ± 0.017 (*)	0.251 ± 0.012 (*)	0.321 ± 0.014
	8	0.262 ± 0.015 (v)	0.238 ± 0.004 (v)	0.222 ± 0.008
	9	0.203 ± 0.013 (v)	0.191 ± 0.007 (v)	0.156 ± 0.003
HeartDisease	5	0.453 ± 0.021 (v)	0.405 ± 0.018 (*)	0.423 ± 0.016
	6	0.561 ± 0.036 (v)	0.521 ± 0.038	0.511 ± 0.022
	7	0.433 ± 0.029 (v)	0.411 ± 0.036	0.402 ± 0.032

¹. Spectral clustering is set as the baseline model.

². The best results are highlighted in boldface.

³. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

Table 6.3 Comparison of ASC for Different Clustering Models on Testing Dataset

Data Collection	Number of Clusters	EACA	EDCA	Spectral Clustering
Wine	2	0.287 ± 0.014 (v)	0.304 ± 0.021 (v)	0.264 ± 0.018
	3	0.438 ± 0.026 (*)	0.402 ± 0.028 (*)	0.502 ± 0.016
	4	0.313 ± 0.024 (v)	0.284 ± 0.027 (v)	0.263 ± 0.011
Twitter	6	0.684 ± 0.026 (*)	0.627 ± 0.035 (*)	0.708 ± 0.022
	7	0.607 ± 0.021 (v)	0.554 ± 0.027 (v)	0.532 ± 0.019
	8	0.532 ± 0.018 (v)	0.473 ± 0.019 (v)	0.452 ± 0.011
News	5	0.487 ± 0.017 (*)	0.422 ± 0.019 (*)	0.532 ± 0.014
	6	0.563 ± 0.027 (*)	0.494 ± 0.028 (*)	0.597 ± 0.021
	7	0.552 ± 0.020 (v)	0.521 ± 0.019	0.534 ± 0.022
Bank	3	0.496 ± 0.016	0.552 ± 0.017 (v)	0.485 ± 0.013
	4	0.625 ± 0.018	0.677 ± 0.022 (v)	0.616 ± 0.019
	5	0.525 ± 0.021 (v)	0.543 ± 0.026 (v)	0.493 ± 0.020
Salary	3	0.513 ± 0.029	0.542 ± 0.030 (v)	0.502 ± 0.022
	4	0.664 ± 0.022	0.701 ± 0.025 (v)	0.671 ± 0.019
	5	0.627 ± 0.021 (v)	0.594 ± 0.032 (v)	0.569 ± 0.027
Student-Por	4	0.423 ± 0.033	0.477 ± 0.036 (v)	0.432 ± 0.031
	5	0.503 ± 0.035 (*)	0.528 ± 0.031 (*)	0.567 ± 0.029
	6	0.439 ± 0.031 (*)	0.478 ± 0.035	0.484 ± 0.032
Automobile	4	0.643 ± 0.010 (v)	0.616 ± 0.012 (v)	0.567 ± 0.016
	5	0.713 ± 0.018 (v)	0.668 ± 0.017 (v)	0.646 ± 0.014
	6	0.591 ± 0.019 (v)	0.535 ± 0.021	0.533 ± 0.018
Internet	7	0.431 ± 0.022 (v)	0.374 ± 0.020 (*)	0.402 ± 0.019
	8	0.347 ± 0.012 (v)	0.303 ± 0.018 (v)	0.288 ± 0.016
	9	0.272 ± 0.015 (v)	0.241 ± 0.016 (v)	0.203 ± 0.009
HeartDisease	5	0.532 ± 0.016	0.475 ± 0.022 (*)	0.538 ± 0.019
	6	0.645 ± 0.024 (v)	0.592 ± 0.031	0.611 ± 0.032
	7	0.483 ± 0.022 (v)	0.456 ± 0.028	0.446 ± 0.021

¹. Spectral clustering is set as the baseline model.

². The best results are highlighted in boldface.

³. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

6.3.2.2 Experimentation on Variable Cluster Recognition

Intuitively, the correctness for cluster pattern matching should be determined by the proportion their matched contents. Thus F_1 score, a synthetic measurement, which encapsulates both precision and recall assessments and takes them into balanced consideration, is employed as a criterion to evaluate the correctness of the recognition model. Mathematically, the calculating process for F_1 score can be formularised as:

$$F_1 = 2 \cdot \frac{\textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}} \quad (6.7)$$

where *precision* and *recall* are respectively denoted as:

$$\textit{precision} = \frac{\textit{number of matched variables}}{\textit{number of variables in uninformed cluster}} \quad (6.8)$$

$$\textit{recall} = \frac{\textit{number of matched variables}}{\textit{number of variables in informed cluster}} \quad (6.9)$$

In the present experimentation, a matching is regarded a correct identification if $F_1 \geq \frac{2}{3}$. Note that for two clusters (an uninformed cluster and an informed cluster) prepared to undergo a matching test, they are expected to be similar in intra-variance, since it will increase the chance for the two examined clusters to be similar in a broad sense. Conversely, in an extreme case, performing matching on a singleton variable cluster with a macro cluster including a large number of variables is definitely not desirable. Thus, a threshold cut is essential for both hierarchical trees (one for feature variables in the informed data corpus and the other for feature variables in the unknown testing data) created by previous clustering algorithm to generate pairs of clusters with similar intrinsic characteristic for matching.

As feature variable pattern recognition is a brand new topic. it is impractical to directly compare this work with any existing work with respect to this novel problem. Instead, two well-known matching methods based on bipartite graphs are implemented and refined for comparison: the max flow matching algorithm (MFMA), and the Kuhn-Munkres algorithm (KMA). These approaches have proved working well on finding maximum weighted bipartite matching [224]. Specifically, MFMA assisted with EDCA is selected as a baseline model to conduct the experimentation. Particularly, the variable cluster in a set of informed variable clusters with the highest maximum weight to the uninformed target is selected as its optimum matching. In order to create a weighted link between a variable-pair in which each of the involved feature variables comes from a distinct cluster, a series of steps is required to perform in advance with respect to different variable-pair types:

1. Numeric variable pair: the link weight is measured by the similarity degree between their respective core vector. The concrete calculation step is specified in Eqn. 6.2.
2. Categorical variable pair: the link weight is determined by the similarity of their respective term distribution vector. The computation steps are described in Eqn. 6.3 and Eqn. 6.4.
3. Mixed-type variable pair: the numeric feature variable is transformed into its categorical counterpart at first, and then the same calculation steps as handling categorical variable pair is performed.

The experimental result can be presented as predicting accuracy, which is defined by the percentage of the uninformed variable clusters “correctly” identified over all the unknown clusters generated with a specific dendrogram cut. Table 6.4 to Table 6.6 provide an overview of the evaluation result. As uninformed feature variable cluster detection is a challenging task, it is generally the case that the predicting accuracy is worse than what might be expected in conventional clustering or classification problems, unless the size of a cluster is extremely small, say, just involving one singleton variable. However, it is clearly shown that the general performance of the proposed approach is better than those achievable by both MFMA and KMA, for each of the data types they handled. In addition to this significant observation, the clusters generated by EACA are more likely to be “correctly” identified than those yielded by EDCA, regardless of which matching method being used. This is possibly because agglomerative clustering enables capturing more internal information than divisive clustering, leading to generate more compact clusters. Moreover, it is worth mentioning that for the current task, both MFMA and KMA are likely to match an uninformed variable cluster to an informed counterpart of a larger size (containing more feature variables) rather than a smaller size, due to their intuition to match as more variables as possible. A heuristic may be employed to help guide their matching behaviour. In particular, a normalisation step considering relative maximum matching weight rather than conventional maximum matching weight may be another direction to improve the performance of these matching techniques.

In terms of time complexity, the cost of performing MFMA for the current task is $O(n^2)$, where n denotes the number of feature variables in the uninformed cluster. The time complexity of original KMA is $O(n^4)$, with a modified version to achieve an $O(n^3)$ running time [225]. For numeric variable cluster detection, the efficiency of the compared methods are remarkably inferior to the proposed model using core vector representation, that just has a time complexity of $O(m)$, where m here depicts the number of elements to form the core vector. For the tasks of categorical and mixed-type variable cluster identification, both of the compared approaches are computationally cheaper than the proposed evolutionary algorithm

when handling large sized clusters, with each involving many feature variables, whilst they suffer from worse performance on predicting accuracy.

Table 6.4 Comparison on Accuracy of Successful Matching by Different Algorithms with Numeric Data Collections

Data	Cut	EACA			EDCA		
		Proposed(N)	KMA	MFMA	Proposed(N)	KMA	MFMA
Wine	0.2	0.80 ± 0.03 (v)	0.72 ± 0.03	0.72 ± 0.02	0.76 ± 0.03 (v)	0.70 ± 0.03	0.70 ± 0.03
	0.4	0.58 ± 0.08 (v)	0.56 ± 0.09 (v)	0.56 ± 0.06 (v)	0.51 ± 0.09	0.50 ± 0.08	0.48 ± 0.09
	0.6	0.46 ± 0.16	0.43 ± 0.17	0.43 ± 0.16	0.50 ± 0.10	0.46 ± 0.12	0.45 ± 0.10
Twitter	0.2	0.76 ± 0.05 (v)	0.73 ± 0.04 (v)	0.73 ± 0.06 (v)	0.72 ± 0.05	0.70 ± 0.05	0.70 ± 0.05
	0.4	0.46 ± 0.07 (v)	0.44 ± 0.08 (v)	0.44 ± 0.09 (v)	0.38 ± 0.09 (v)	0.34 ± 0.08	0.34 ± 0.08
	0.6	0.39 ± 0.04 (v)	0.35 ± 0.05 (v)	0.34 ± 0.06	0.32 ± 0.05	0.33 ± 0.06	0.32 ± 0.05
News	0.2	0.82 ± 0.03 (v)	0.78 ± 0.02 (v)	0.78 ± 0.02 (v)	0.77 ± 0.03 (v)	0.75 ± 0.02	0.75 ± 0.03
	0.4	0.38 ± 0.08 (*)	0.34 ± 0.09 (*)	0.34 ± 0.08 (*)	0.43 ± 0.08 (v)	0.40 ± 0.09	0.40 ± 0.09
	0.6	0.39 ± 0.06 (*)	0.36 ± 0.07 (*)	0.36 ± 0.06 (*)	0.48 ± 0.07 (v)	0.43 ± 0.08	0.43 ± 0.08

¹. MFMA with EDCA is set as the baseline model.

². The best results are highlighted in boldface.

³. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

Table 6.5 Comparison on Accuracy of Successful Matching by Different Algorithms with Categorical Data Collections

Data	Cut	EACA			EDCA		
		Level	Proposed(C)	KMA	MFMA	Proposed(C)	KMA
Bank	0.3	0.50 \pm 0.05 (v)	0.39 \pm 0.04 (v)	0.37 \pm 0.05 (v)	0.42 \pm 0.03 (v)	0.33 \pm 0.04	0.33 \pm 0.04
	0.5	0.42 \pm 0.06 (v)	0.28 \pm 0.08 (v)	0.28 \pm 0.08 (v)	0.37 \pm 0.06 (v)	0.24 \pm 0.06	0.24 \pm 0.06
	0.7	0.27 \pm 0.09 (v)	0.12 \pm 0.04 (*)	0.12 \pm 0.04 (*)	0.39 \pm 0.06 (v)	0.21 \pm 0.08	0.21 \pm 0.09
Salary	0.3	0.67 \pm 0.07 (v)	0.62 \pm 0.06 (v)	0.62 \pm 0.06 (v)	0.60 \pm 0.08 (v)	0.51 \pm 0.07	0.51 \pm 0.07
	0.5	0.50 \pm 0.04 (v)	0.38 \pm 0.06 (v)	0.37 \pm 0.07 (v)	0.42 \pm 0.06 (v)	0.31 \pm 0.07	0.31 \pm 0.07
	0.7	0.29 \pm 0.10 (v)	0.13 \pm 0.08 (*)	0.13 \pm 0.09 (*)	0.39 \pm 0.09 (v)	0.25 \pm 0.08	0.25 \pm 0.08
Student-Por	0.3	0.62 \pm 0.05	0.58 \pm 0.06	0.58 \pm 0.06	0.61 \pm 0.06	0.56 \pm 0.06	0.56 \pm 0.05
	0.5	0.45 \pm 0.07 (v)	0.36 \pm 0.06 (v)	0.36 \pm 0.06 (v)	0.38 \pm 0.05 (v)	0.32 \pm 0.07	0.32 \pm 0.07
	0.7	0.42 \pm 0.05 (v)	0.31 \pm 0.04 (v)	0.30 \pm 0.05 (v)	0.33 \pm 0.06 (v)	0.27 \pm 0.05	0.27 \pm 0.05

1. MFMA with EDCA is set as the baseline model.

2. The best results are highlighted in boldface.

3. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

Table 6.6 Comparison on Accuracy of Successful Matching by Different Algorithms with Mixed-Type Data Collections

Data	Cut	EACA			EDCA			
		Collection	Level	Proposed(M)	KMA	MFMA	Proposed(M)	KMA
Automobile	0.4		0.88 ± 0.04 (v)	0.87 ± 0.03	0.87 ± 0.03	0.85 ± 0.03	0.85 ± 0.04	0.85 ± 0.04
	0.6		0.56 ± 0.05 (v)	0.48 ± 0.06	0.47 ± 0.06	0.50 ± 0.05	0.46 ± 0.06	0.46 ± 0.06
	0.8		0.39 ± 0.07 (v)	0.32 ± 0.08	0.32 ± 0.08	0.36 ± 0.08 (v)	0.31 ± 0.07	0.31 ± 0.07
Internet	0.3		0.61 ± 0.04 (v)	0.55 ± 0.04	0.54 ± 0.03	0.58 ± 0.04 (v)	0.53 ± 0.05	0.53 ± 0.05
	0.5		0.48 ± 0.05 (v)	0.35 ± 0.07	0.34 ± 0.06	0.52 ± 0.06 (v)	0.41 ± 0.05	0.41 ± 0.06
	0.7		0.38 ± 0.06 (v)	0.31 ± 0.09	0.31 ± 0.09	0.40 ± 0.06	0.33 ± 0.09	0.33 ± 0.09
HeartDisease	0.4		0.82 ± 0.03 (v)	0.82 ± 0.03	0.80 ± 0.04	0.84 ± 0.04 (v)	0.81 ± 0.05	0.81 ± 0.05
	0.6		0.39 ± 0.03 (v)	0.33 ± 0.04 (v)	0.33 ± 0.04 (v)	0.41 ± 0.05 (v)	0.30 ± 0.04	0.30 ± 0.04
	0.8		0.43 ± 0.03 (v)	0.34 ± 0.05	0.33 ± 0.05	0.38 ± 0.04 (v)	0.32 ± 0.06	0.32 ± 0.06

¹. MFMA with EDCA is set as the baseline model.

². The best results are highlighted in boldface.

³. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

6.4 Summary

This chapter has proposed a predicting model to perform the novel task of pattern recognition for groups of uninformed feature variables. With the support from collections of informed data, such task can be accomplished via searching for similar patterns between the uninformed variable group to the informed counterpart. Particularly, both efficient agglomerative and divisive clustering approaches are designed and implemented to categorise related feature variables into groups according to their respective application scenarios. Different metrics to handle data collections in various types (numeric, categorical and their hybrid) are presented, with experimental evaluation demonstrating the capability of the proposed approach.

Despite such achievements, the efficacy of the proposed work may be further demonstrated with more real-world datasets. Additionally, the link weights used in the current hierarchical clustering model are in the format of crisp value, which are derived from the defuzzification step discussed in Section 5.1.4.2. Note that such defuzzification process may cause losing important information from its original fuzzy interpretation. An advanced clustering approach which exploiting such fuzzy value directly to generate a more informative hierarchical structure is worth investigating. A prospective outlook for this part will be discussed in Chapter 8.

Chapter 7

Feature Variable Identification and Hierarchical Knowledge Reorganisation

Knowledge representation is a branch of artificial intelligence dedicated to capture information about the world in a form that a computer system can utilize to solve complex problems [226]. Nowadays, with the development of modern technology, information data are growing more quickly than anticipated. Making better use of data to generate new knowledge is of great necessity in data mining. However, for traditional offline techniques, although being capable of generating accurate results, their iteration process may not be sufficiently flexible to catch up with the pace of data growth (need to reconstruct the model to accommodate more information, not sufficiently flexible for update and refinement). Recently, online techniques in data mining have increasingly played a more important role for knowledge detection, due to their effectiveness and efficiency in providing prompt reaction to the up-to-date information. In particular, hierarchical representation of knowledge, possessing the advantages of being both informative and intuitive, is widely employed in various application fields, including human resource management [227], biological taxonomy [228, 229], social media and webpage investigation [230, 231], and knowledge graph completion [232], to name but a few. It also enables accepting acquisition of new information and deriving novel structure from existing framework. Thus, It offers a great potential for developing knowledge models which allow updating knowledge on the fly.

In Chapter 6, a hierarchical representation for the knowledge about the layout of domain feature variables was presented. This provides a systematic view on how the domain features are organised in data collections. Simultaneously, a novel predicting model has been proposed to perform the brand new task of identifying patterns for a group of uninformed feature variables in such domain. This may lead to two opposite results: (1) an uninformed

variable cluster is clearly or approximately detected; (2) there currently exists no informative pattern well matching the target uninformed variable cluster. For the former situation, it is aggressive but sensible to perform a further task of singleton variable identification. For the latter circumstance, utilising the information acquired with the uninformed group to enlarge and update the current knowledge structure about the domain feature variables is an advisable option. In this chapter, specific schemes to perform each of the above mentioned tasks with respect to an uninformed feature variable clusters will be discussed. Importantly, the work presented in this chapter, together with the contents included in Chapter 5 and Chapter 6, are integrated to create a massive model for feature variable prediction. The general model of this predicting system will be provided in Section 7.3.

The remainder of this chapter is organised as follows: Section 7.1 introduces the searching mechanism for identifying singleton feature variable in a detected uninformed variable cluster. Section 7.2 outlines the scheme to handle an uninformed variable cluster which has not been identified, with its implementation procedure described in detail. Section 7.3 shows the experimental evaluation for the proposed approaches discussed in this chapter along with a discussion of results. Section 7.4 generalises the contents discussed in Section 5, Chapter 6 and the proposed models described in this chapter, and presents a general model for prediction at feature variable (cluster) level. Finally, Section 7.5 concludes the chapter.

7.1 Feature Variable Identification

In order to determine whether to perform the singleton variable detection or not, a threshold value $\alpha \in [0, 1]$ is set at first. For a pair of an uninformed variable cluster \mathcal{C}_U and its closest informed counterpart \mathcal{C}_K obtained from the variable cluster recognition subsystem (described in Section 6.2.2), if $\text{sim}(\mathcal{C}_U, \mathcal{C}_K) \geq \alpha$, which shows that \mathcal{C}_U and \mathcal{C}_K are highly similar to each other, and may further indicate that a singleton unknown variable v_U in \mathcal{C}_U could possibly have a similar counterpart in \mathcal{C}_K , or more ambitiously, have a copy of itself in \mathcal{C}_K . Inspired by this observation, an iterative searching algorithm is designed to look for the counterpart of v_U in \mathcal{C}_K . Specifically, the hierarchical structure of the feature variables is usually visualized as a dendrogram, with a leaf node corresponding to a singleton feature variable cluster, the root denoting the entire cluster including all the feature variables in the discussed domain, and a node in the dendrogram representing a variable cluster containing all its child nodes. The proposed searching strategy follows the concept of binary search, which iteratively divides the searching space into separated halves by heuristics and finally obtains the searching result. Likewise, each time, the proposed approach works by comparing the target feature

variable to both sides of the hierarchy at first, and selecting the side with the higher similarity degree. This step is conducted repeatedly until reaching the singleton variable cluster. Then the feature variable included in this singleton cluster is regarded as a possible matching. The pseudo code for this searching algorithm is as follows:

Algorithm 10: Algorithm to Identify Uninformed Variable

Input: v_U : An uninformed variable to be identified \mathcal{C}_K : Closet informed variable cluster L : A List to store variable clusters

```

1 Create a singleton variable cluster  $\mathcal{C}_U^*$  for  $v_U$  ;
2  $\mathcal{C} \leftarrow \mathcal{C}_K$  ;
3 while  $\mathcal{C}.\text{SIZE}() > 1$  do
4   |  $L \leftarrow \mathcal{C}.\text{getChildren}()$  ;
5   | for  $\mathcal{C}_l$  in  $L$  do
6   |   | COMPUTE  $\text{sim}(\mathcal{C}_U^*, \mathcal{C}_l)$  ;
7   | end
8   |  $\mathcal{C} \leftarrow \text{FIND } \mathcal{C}_l$  in  $L$  with maximum  $\text{sim}(\mathcal{C}_U^*, \mathcal{C}_l)$  ;
9 end
10 PICK  $v_K$  from  $\mathcal{C}$  ;  $\triangleright \mathcal{C}$  contains a single variable

```

Output: v_K : Counterpart of v_U in \mathcal{C}_K

Note that the computation of $\text{sim}(\mathcal{C}_U^*, \mathcal{C}_l)$ in Algorithm. 10 considers all types of those variable clusters involved (be they numeric variable clusters, categorical variable clusters or mixed-type variable clusters). For each of the cluster pairs, corresponding computation method needs to be selected accordingly. The working mechanism for this has been discussed in Chapter 6. The average time complexity for this iterative searching algorithm is as simple as $O(\log(|\mathcal{C}_K|))$, where $|\mathcal{C}_K|$ denotes the number of variables in the compared informed variable cluster. Occasionally, more than one closet informed variable cluster may be taken into consideration at the beginning, especially when several informed clusters have a similar proximity to the target cluster. For this circumstance, the proposed algorithm is to run through on each of the clusters under the consideration that may contain a variable similar to the target feature variable. The resultant prediction is based on selecting the one with the highest similarity degree.

7.2 Hierarchical Knowledge Reorganisation

There is also the case that the similarity degree between an uninformed variable cluster to its closest counterpart is still less than α , i.e., $\text{sim}(\mathcal{C}_U, \mathcal{C}_K) < \alpha$. Intuitively, when facing such a scenario, it is not wise to perform iterative search for the task of variable identification, since it is natural to believe that if two clusters of variables are not sufficiently similar, they are unlikely to contain similar singleton variables as well. Although chances are that two clusters may include a certain number of similar or identical variables, the portion of such variables within the cluster may be rather too small. This leads to the task of correct identification extremely difficult compared to handling two clusters with a significant higher similarity degree. However, by contrast, this occasion implies that the uninformed variable cluster may contain variables which are not incorporated in the current hierarchical structure. Also, the information covered by the unknown variable cluster can still be utilised to enrich the existing knowledge base. This observation supports the intuition to update and enlarge the obtained hierarchical structure to better accommodate domain feature variables.

A pre-assumption is set up prior to performing such updating step: the hierarchical variable structure needs to be considered as a binary tree with each cluster being a node in the tree. Different from the traditional definition of binary tree [233, 234] which allows a tree node to have a single child node, the tree node in the binary tree discussed here should have either exactly two child nodes or no child node at all. Several properties can be acquired from such an assumed tree structure.

- The top-level cluster which incorporates all the feature variables generated through the variable clustering step (root cluster in the hierarchical structure) should be considered as a root node for the tree.
- The singleton variable cluster should be treated as a leaf node in the tree, and have no child cluster nodes.
- Each cluster node besides the leaf cluster node should have exact two child cluster nodes, including all the feature variables from its child cluster nodes, but excluding any variable from nodes other than its own child cluster nodes.
- Each cluster node besides the root cluster node should have a parent cluster node, with the child cluster node representing a subcluster to the cluster denoted by the parent node.
- Two cluster nodes with the same parent cluster node are considered as siblings.

- The cluster nodes acquired by a specific level cut are considered as neighbours.

When consider inserting a new cluster node into an existing hierarchical structure (binary tree), the crucial point is to determine the correct position in the tree in which the new node to be insert into. Note that the present discussion assumes that both the cluster to be inserted and the referential set of clusters are generated at the same cut level with respect to their own dendrogram. This assumption is intuitive and practically imperative, otherwise it may lead to searching in the entire hierarchical space to determine the closest cluster, which is time consuming and perhaps misdirecting for the updating process. For an incoming cluster \mathcal{C}_x potentially to be inserted, since its closest counterpart in the referential hierarchy has already been detected, named as \mathcal{C}_a , it has a natural appeal to perform the job of cluster insertion guided by \mathcal{C}_a . Assumes that \mathcal{C}_a has a parent cluster \mathcal{C}_p , inserting \mathcal{C}_x into \mathcal{C}_p to make \mathcal{C}_x and \mathcal{C}_a becoming siblings may seem to be an intuitive and direct choice. However, it is possibly not an ideal option, since it violates the preset binary tree structure (\mathcal{C}_p have three child clusters), and it may not correctly reflect the intimacy degree of \mathcal{C}_a to \mathcal{C}_x and that of \mathcal{C}_a to other same-level clusters as well. Hence, a novel approach to overcoming such defects is proposed.

The proposed approach performs a series of heuristics to determine the position in the hierarchical tree into which the incoming variable cluster is to be inserted. Consider \mathcal{C}_y , where \mathcal{C}_y , the neighbour cluster of \mathcal{C}_a , and it should meet the following criteria:

- (1) $sim(\mathcal{C}_x, \mathcal{C}_a) > sim(\mathcal{C}_x, \mathcal{C}_y)$;
- (2) $\mathcal{C}_y = \arg \max_{\mathcal{C}_i} sim(\mathcal{C}_a, \mathcal{C}_i)$ ($1 \leq i \leq t$), where t is the number of neighbour clusters to \mathcal{C}_a with $sim(\mathcal{C}_a, \mathcal{C}_i) < sim(\mathcal{C}_x, \mathcal{C}_a)$.

This means that \mathcal{C}_y is the cluster having the greatest $sim(\mathcal{C}_a, \mathcal{C}_i)$ score among all t neighbour clusters with $sim(\mathcal{C}_a, \mathcal{C}_i) < sim(\mathcal{C}_a, \mathcal{C}_x)$. It is therefore, the selected cluster to guide the inserting procedure. When performing the inserting step, three different scenarios for the relative position of \mathcal{C}_y in the hierarchical variable clustering tree need to be taken into account respectively.

- Scenario 1: \mathcal{C}_y is the sibling of \mathcal{C}_a and is the cluster with the greatest $sim(\mathcal{C}_x, \mathcal{C}_i)$ ($1 \leq i \leq n$) among all n neighbour clusters. An illustration of this scenario is shown in Fig. 7.1a (with the cloud denoting the root cluster, the nodes in blue representing the original cluster, the nodes in red indicating the variable to be inserted into the hierarchy. The same expressions are also used in Fig. 7.2a and Fig. 7.3a).

- Scenario 1-1: $sim(\mathcal{C}_x, \mathcal{C}_a) > sim(\mathcal{C}_y, \mathcal{C}_a)$. For this situation, the steps for insertion is arranged as follows:
 - * Step 1: Delete \mathcal{C}_y as a child of \mathcal{C}_p ;
 - * Step 2: Set \mathcal{C}_x as a child of \mathcal{C}_p ;
 - * Step 3: Create a new variable cluster $\mathcal{C}_{p'}$, set \mathcal{C}_p and \mathcal{C}_y as children of $\mathcal{C}_{p'}$;
 - * Step 4: Delete \mathcal{C}_p as a child of \mathcal{C}_g ;
 - * Step 5: Set \mathcal{C}_p and $\mathcal{C}_{p'}$ as children of \mathcal{C}_g .

An illustration of updated result for this scenario is shown in fig. 7.1b (The nodes in orange represent those clusters requiring update during the process of insertion. The same expression is used in Fig. 7.1c, Fig. 7.2b, Fig. 7.2c, Fig. 7.3b, Fig. 7.3c as well) . It should be noted that when such updating steps are performed, the operations of adding (removing) variables to (from) the related ancestor clusters need to be conducted accordingly. These basic operations are also executed when performing updating process for other scenarios.

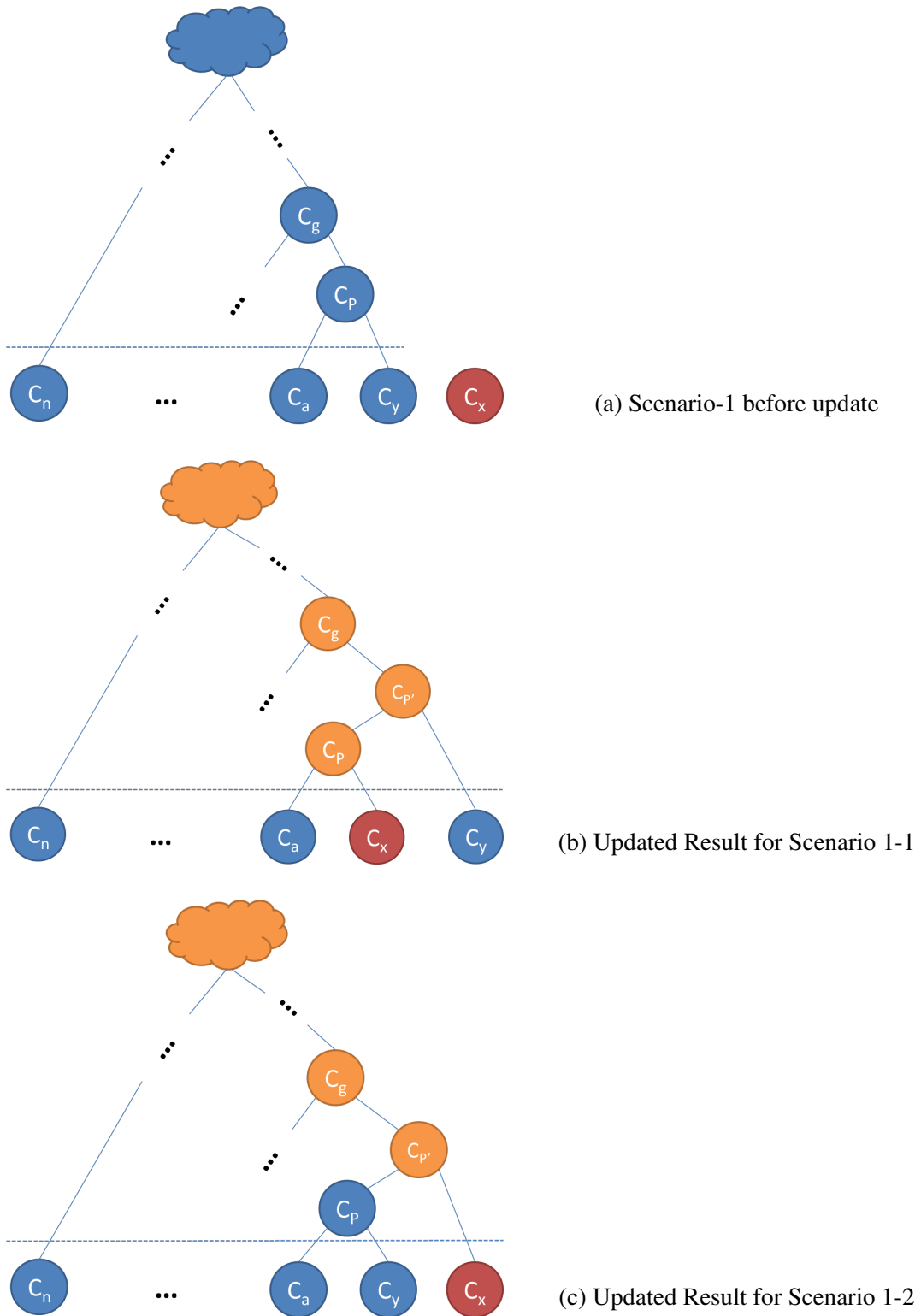


Fig. 7.1 Updating Process for Scenario-1

- Scenario 1-2: $\text{sim}(\mathcal{C}_x, \mathcal{C}_a) \leq \text{sim}(\mathcal{C}_y, \mathcal{C}_a)$, which indicates the similarity degree between \mathcal{C}_x and \mathcal{C}_a is no greater than that of the existing siblings \mathcal{C}_y and \mathcal{C}_a in the hierarchy. For this situation, it is not necessary to change the local hierarchical structure of \mathcal{C}_p being the parent cluster with two children clusters \mathcal{C}_a and \mathcal{C}_y . However, \mathcal{C}_x is still closer to \mathcal{C}_a than any other cluster nodes besides \mathcal{C}_y when considering all the clusters (apart from \mathcal{C}_x itself) generated by the current cut. Thus, it is ideal to merge \mathcal{C}_a and \mathcal{C}_x into a single cluster at earliest stage as possible. Having noticed this inherent characteristic embedded, creating a new cluster to accommodate \mathcal{C}_p and \mathcal{C}_x is expected and required. In particular, the updating procedure is stated as follows:

- * Step 1: Delete \mathcal{C}_p as a child of \mathcal{C}_g ;
- * Step 2: Create a new variable cluster $\mathcal{C}_{p'}$, set \mathcal{C}_p and \mathcal{C}_x as children of $\mathcal{C}_{p'}$;
- * Step 3: Set $\mathcal{C}_{p'}$ as a child of \mathcal{C}_g .

A demonstration of the updated result is shown in Figure. 7.1c.

- Scenario 2: $\text{sim}(\mathcal{C}_x, \mathcal{C}_a) > \text{sim}(\mathcal{C}_y, \mathcal{C}_a)$, where \mathcal{C}_y is a direct child cluster to one of \mathcal{C}_a 's ancestors, denoted as \mathcal{C}_g . This scenario is shown in Fig. 7.2a. Note that \mathcal{C}_g has another child cluster, named \mathcal{C}_p . For this situation, two possible sub-scenarios needs to be taken into consideration.

- Scenario 2-1: $\text{sim}(\mathcal{C}_p, \mathcal{C}_x) > \text{sim}(\mathcal{C}_y, \mathcal{C}_x)$. For this scenario, \mathcal{C}_p and \mathcal{C}_x need to be merged at first before being clustered with \mathcal{C}_y . A brief introduction of manipulating the procedure is described as follows:

- * Step 1: Delete \mathcal{C}_p as a child of \mathcal{C}_g ;
- * Step 2: Create a new variable cluster $\mathcal{C}_{p'}$, set \mathcal{C}_p and \mathcal{C}_x as children of $\mathcal{C}_{p'}$;
- * Step 3: Set $\mathcal{C}_{p'}$ as a child of \mathcal{C}_g .

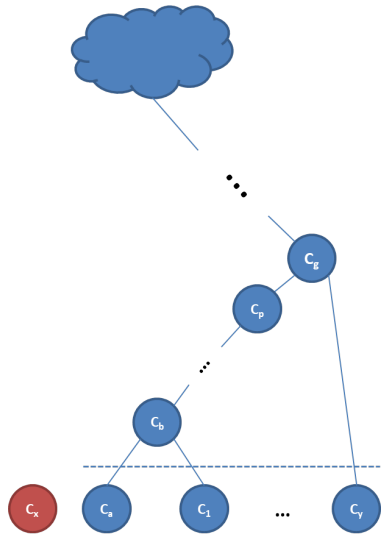
An illustration of this updating process for the described scenario is shown in Fig. 7.2b.

- Scenario 2-2: $\text{sim}(\mathcal{C}_p, \mathcal{C}_x) \leq \text{sim}(\mathcal{C}_y, \mathcal{C}_x)$. For this scenario, the original hierarchical structure needs minor but necessary revision: \mathcal{C}_x and \mathcal{C}_y are supposed to be grouped at first prior to being combined with \mathcal{C}_p . The procedure for such revision is articulated as follows:

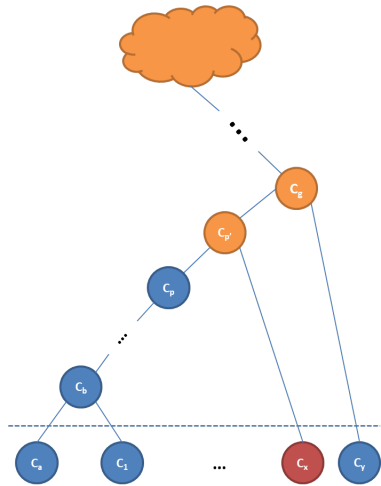
- * Step 1: Delete \mathcal{C}_y as a child of \mathcal{C}_g ;
- * Step 2: Create a new variable cluster $\mathcal{C}_{p'}$, set \mathcal{C}_x and \mathcal{C}_y as children of $\mathcal{C}_{p'}$;

* Step 3: Set \mathcal{C}_p' as a child of \mathcal{C}_g .

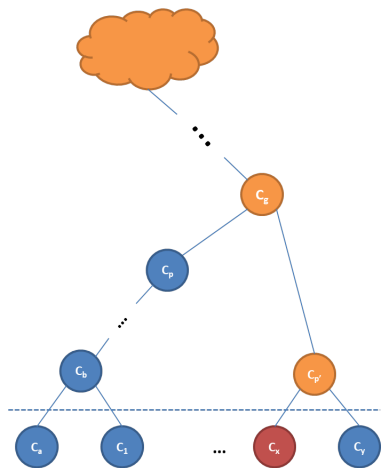
Similarly, a demonstration for such updating operation is presented in Fig. 7.2c.



(a) Scenario-2 before update



(b) Updated Result for Scenario 2-1



(c) Updated Result for Scenario 2-2

Fig. 7.2 Updating Process for Scenario-2

- Scenario 3: $sim(\mathcal{C}_x, \mathcal{C}_a) > sim(\mathcal{C}_y, \mathcal{C}_a)$, where \mathcal{C}_y has already been merged with another cluster \mathcal{C}_b prior to being clustered with \mathcal{C}_a 's ancestors in the existing hierarchy. An presentation of such scenario is revealed in Fig. 7.3a. When facing this situation, two distinct sub-scenarios need to be taken into account.

- Scenario 3-1: $sim(\mathcal{C}_x, \mathcal{C}_y) > sim(\mathcal{C}_b, \mathcal{C}_y)$. For this scenario, a three-step updating process is proposed as follows:

- * Step 1: Delete \mathcal{C}_y as a child of \mathcal{C}_p ;
- * Step 2: Create a new variable \mathcal{C}'_p , set \mathcal{C}_x and \mathcal{C}_y as children of \mathcal{C}'_p ;
- * Step 3: Set \mathcal{C}'_p as a child of \mathcal{C}_p .

An illustration of the updating process for the above scenario is shown in Fig. 7.3b.

- Scenario 3-2: $sim(\mathcal{C}_x, \mathcal{C}_y) \leq sim(\mathcal{C}_b, \mathcal{C}_y)$. For this scenario, it is merely necessary to merge \mathcal{C}_x and \mathcal{C}_p , whilst keeping the basic local structure of \mathcal{C}_p with two children \mathcal{C}_y and \mathcal{C}_b unchanged. The steps for performing the operation can be presented as follows:

- * Step 1: Delete \mathcal{C}_p as a child to its parent \mathcal{C}_q ;
- * Step 2: Create a new cluster \mathcal{C}'_p , set \mathcal{C}_x and \mathcal{C}_p as children of \mathcal{C}'_p ;
- * Step 3: Set \mathcal{C}'_p as a child of \mathcal{C}_q .

An example of updated hierarchy is demonstrated in Fig. 7.3c.

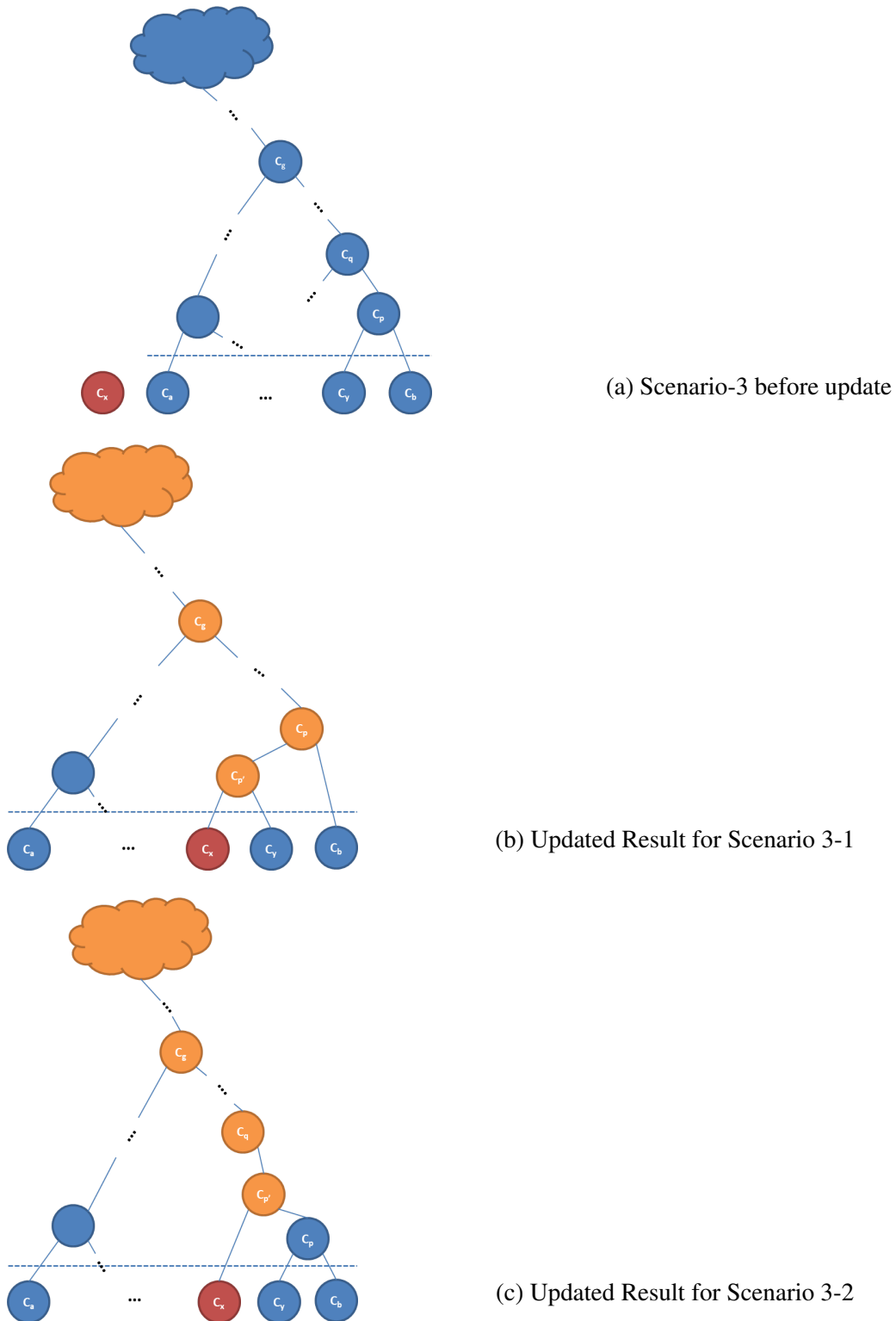


Fig. 7.3 Updating Process for Scenario-3

It is important to point out that during the updating process, the substructure of the inserted nodes remains unchanged, which also guarantees preserving the binary configuration for the entire hierarchical structure. Generally, the proposed approach performs the task of inserting a variable cluster into an existing hierarchical structure while completely preserving its binary tree layout. Note that the updating process can be performed iteratively when a set of undetected feature variable clusters arrives, which forms an online training procedure. Although the inserted cluster is not generally identified clearly, the resulting hierarchical structure provides a relatively informative view for the underlying patterns of this undetected cluster from its sibling cluster and neighbour clusters.

7.3 Experimental Evaluation

This section presents the experimental evaluation of the proposed work for both the tasks of singleton variable identification and hierarchical structure reorganisation.

7.3.1 Experimentation for Singleton Variable Identification

7.3.1.1 Data Preparation

To evaluate the performance of the iterative searching procedure, Algorithm 10 is tested over three categories of datasets: numeric, categorical and mixed-type, respectively. Each category contains four datasets, with their general information depicted in Table 7.1.

Table 7.1 Summary of Datasets: Singleton Variable Identification

Dataset Collection	Type	Number of Feature Variables	Number of Instances
Wine	N	14	5024
Twitter	N	88	564266
News	N	82	40896
Urban	N	146	2639
Bank	C	22	45213
Salary	C	25	30168
Student-Port	C	28	652
Student-Mat	C	28	404
Automobile	M	24	318
Internet	M	57	8848
HeartDisease	M	70	500
Arrhythmia	M	242	1210

¹. Type: N = Numeric Data; C = Categorical Data; M = Mixed-Type Data

7.3.1.2 Experimental setup

In the experiments carried out, each dataset is split into two subsets in equal size. EACA is performed on both subsets to create two hierarchical trees using human knowledge to determine the number of clusters being generated. The test is only conducted on pairs of variable clusters meeting the following selection criteria: (1) each of the clusters in the cluster pair comes from a distinct hierarchical tree, (2) the similarity degree between the two clusters involved in the cluster pair is above a specific level, with the calculation formula having been described in Section 6.2. In the present experimental setting, the level of similarity degree for a pair of numeric variables is set to 0.9, and is set to 0.6 for both categorical variable pairs and mixed-type variable pairs. This setting is practically meaningful since in reality, it is not necessary to force performing singleton variable identification if two involving variables clusters are not sufficiently similar. For comparison, exhaustive search (ES) is adopted as a baseline model.

7.3.1.3 Experimental Result

The experimental result is presented by searching accuracy, which is defined as the ratio between the number of real correct matchings and maximum number of possible correct matchings. The reported results are based on an average 30 runs of 2-FCV. From Table

7.2, it is clear that the proposed searching algorithm performs inferior to ES on all the data collections. This is because ES, with running time of $O(N)$ (N represents the size of the feature variable cluster), traversing the entire searching space, may almost guarantee to find the correct match. This is somewhat expected, as the heuristics embedded in the proposed approach is not sufficiently accurate. However, the advantage of the proposed iterative searching scheme over ES is its time efficiency, with an average of $O(\log(N))$ running time to achieve the predicted result. This indicates that when the size of the cluster is large, a trade off between accuracy and efficiency is worth considering.

Table 7.2 Comparison on Accuracy of Singleton Variable Identification

Dataset Collection	Proposed	ES
Wine	0.92 ± 0.05 (*)	1.00 ± 0.00
Twitter	0.78 ± 0.07 (*)	0.96 ± 0.01
News	0.83 ± 0.04 (*)	1.00 ± 0.00
Urban	0.77 ± 0.07 (*)	1.00 ± 0.00
Bank	0.82 ± 0.03 (*)	1.00 ± 0.00
Salary	0.87 ± 0.05 (*)	1.00 ± 0.00
Student-Por	0.94 ± 0.02 (*)	1.00 ± 0.00
Student-Mat	0.92 ± 0.02 (*)	1.00 ± 0.00
Automobile	0.84 ± 0.04 (*)	1.00 ± 0.00
Internet	0.78 ± 0.05 (*)	1.00 ± 0.00
HeartDisease	0.73 ± 0.06 (*)	1.00 ± 0.00
Arrhythmia	0.78 ± 0.07 (*)	0.98 ± 0.00

¹. ES is set as the baseline model.

². The best results are highlighted in boldface.

³. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

7.3.2 Experimentation on Hierarchical Structure Reorganisation

This section provides an artificial scenario concerning the update process of the hierarchical knowledge structure. It is used to demonstrate the procedures of the proposed approach, for hierarchies generated via both agglomerative and divisive clustering methods. The accuracy of the work is validated on both real-world data obtained from UCI repository [206] and synthetic data.

7.3.2.1 Experimental setup

Provided that the external categorical labels are unavailable for any of the feature variables under investigation, the ground truth of the taxonomic information for the variables is not a natural existence in data. Having noticed of that, in experiment, any “ground truth” is artificially computed by performing the same technique on the entire dataset as it conducts on the training data. That is, without losing fairness, the updated hierarchical structures are compared against those directly generated from the entire data with the same clustering approach. The training data is obtained through the following steps:

1. Perform a hierarchical clustering algorithm with respect to feature variables on the entire dataset.
2. Cut the resulting hierarchical tree with a specified similarity threshold, and obtain a set of clusters with intra-similarity for each of the clusters involved above the mentioned threshold.
3. Remove randomly a cluster from the cluster set acquired from Step 2.
4. For each of the feature variables included in the removed cluster, delete all its information with regard to each of the instances in the entire dataset, and return the resulting subset of the entire dataset as a training dataset for the current experiment.

Having obtained such training data, the same clustering approach conducted on the entire dataset is adopted to perform on this training data with respect to the feature variables involved, and a new hierarchical structure can be learned from this process. Following that, the removed feature variable cluster is expected to be inserted to updating the obtained hierarchy. This process guarantees that the inserted variable cluster includes no overlapped variables with the existing structure.

To evaluate the proposed updating scheme, Normalised Mutual Information (NMI) is employed as the validity index, since it causes no bias for a large number of clusters and provides a relatively reliable conclusion. It generates scores ranging between 0 and 1, especially, with the maximum value indicating that clustering result matches the “ground truth” perfectly. The concept of NMI is described in Section 5.1.3. For present experiment, let π_1, π_2 each be a set of feature variable clusters, their NMI can be calculated as:

$$NMI(\pi_1, \pi_2) = \frac{\sum_{i=1}^{|\pi_1|} \sum_{j=1}^{|\pi_2|} n_{ij} \log\left(\frac{n_{ij}N}{n_i m_j}\right)}{\sqrt{\sum_{i=1}^{|\pi_1|} n_i \log\left(\frac{n_i}{N}\right) \sum_{j=1}^{|\pi_2|} m_j \log\left(\frac{m_j}{N}\right)}} \quad (7.1)$$

where $|\pi_1|$ and $|\pi_2|$ represent the number of clusters within π_1 and π_2 respectively, n_i and m_j each denote the number of feature variables in cluster i and j , n_{ij} depicts the number of feature variables agreed by clusters i and j , and N implies the total number of feature variables under consideration.

As inserting a cluster into an existing variable cluster hierarchy is a brand new topic, with seldom related information available in the literature, a naive insert scheme which only considers the cluster to be inserted to its closest counterpart in the hierarchy is set as a baseline model, named as naive insert (NI). Note that the NMI measure should not be simply performed on two set of clusters just generated by the designated cut, since the influence of the updating procedures (inserting steps) will not be straightforwardly revealed at cut level, but demonstrated at a higher level of the hierarchy when macro clusters involving both the existing micro clusters and the inserted cluster are taken into account. In the present experiment, the best choice for the number of clusters from the macro perspective are determined according to human knowledge.

7.3.2.2 Experimental Results

For a single dataset, the results are based on an average of r runs for each of the comparing methods, with r herein denoting the number of variable clusters generated by a specific cut on the entire data. Let \mathcal{T}_t and \mathcal{T}_e each be a hierarchical variable tree generated from the training dataset D_t and the entire dataset D_e , and cut_α represents a specific cut for both of the trees at the threshold of α , where α denotes the dissimilarity degree between feature variables. Every time, a cluster generated by cut_α from \mathcal{T}_e , which contains variables not included in D_t , is selected to be inserted into \mathcal{T}_t . This step performs successively r times for each of the variable clusters generated from the D_e with cut_α . Usually, in order to generate compact clusters, α , within the range of $[0, 1]$, is set to a moderate number (i.e., $\alpha = 0.3, 0.4$ or 0.5).

Table 7.3 Comparison on NMI of Different Updating Process

Dataset	α	Num	Proposed	NI
Twitter(N)	0.3	7	0.66 ± 0.08 (v)	0.55 ± 0.07
Urban(N)	0.4	5	0.57 ± 0.09 (v)	0.48 ± 0.08
Bank(C)	0.4	4	0.70 ± 0.06	0.72 ± 0.08
Salary(C)	0.5	4	0.74 ± 0.08 (v)	0.65 ± 0.10
HeartDisease(M)	0.3	6	0.58 ± 0.06 (v)	0.52 ± 0.05
Arrhythmia(M)	0.5	9	0.68 ± 0.09 (v)	0.58 ± 0.10
Synthetic-1(C)	0.4	6	0.61 ± 0.07 (v)	0.55 ± 0.08
Synthetic-3(N)	0.3	8	0.72 ± 0.05 (v)	0.66 ± 0.07
Synthetic-5(M)	0.4	8	0.67 ± 0.08 (v)	0.61 ± 0.08

¹. (N): Numeric data; (C) Categorical data; (M) Mixed-type data

². Num: Number of macro clusters.

³. NI is set as the baseline model.

⁴. The best results are highlighted in boldface.

⁵. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

The experimental results on NMI measure for both of the updating procedures are reported in Table 7.3. It is clear that the performance of the proposed updating process is universally superior to that achievable using NI. This is because the proposed approach uses more heuristics to infer the appropriate position for inserting the coming feature variable cluster. Importantly, the proposed approach perfectly preserves the binary structure of the hierarchical tree, presenting the underlying knowledge in a well organised and informative manner. On the contrary, NI violates such structure each time when performing the inserting step. As such, it conceal valuable information from the resultant structure that facilitates comparative analyses of the similarity between two pairs of variable clusters: (1) a pair of inserted cluster and its closest counterpart C^* , (2) C^* and its sibling cluster in the original hierarchy.

In terms of time complexity, NI runs $O(l)$ steps to find the position for conducting its inserting operation, which is slightly efficient than the proposed approach with $O(2l)$ of the searching time to determine the appropriate position for insertion, where l denotes the number of clusters generated from \mathcal{T}_l with cut_α . Both methods need additional $O(\log(l))$ of time to update information for all ancestor clusters of the inserted cluster.

7.4 Intelligent System for Variable Detection and Hierarchical Knowledge Reorganisation

This section summarises the contents described in Chapter 5, Chapter 6 and preceding sections of this chapter to provide a comprehensive predicting model for intelligent detection of feature variable and automatic reconstruction of feature variable hierarchical layout. This novel predicting model is adaptable to widespread scenes in real world, particularly when any given information content is obtained from different data sources where parts of the information overlap, and parts of the information are isolated with missed description to them. Actually, there currently exists no effective model to handle such situations while considering all of these data as a whole. In general, this section outlines the general framework of the predicting model and describes the functionality for each of the involving subsystems. Experimental evaluation is also included to present the efficacy of the integrated system.

7.4.1 Generic Framework of Intelligent System

This subsection illustrates the generic framework of the intelligent system for feature variable detection and hierarchical knowledge reorganisation. The system is aimed to perform the task of variable cluster pattern recognition and subsequent jobs of variable identification or variable structure modification. Such ultimate goals are achieved on the basis of feature variable cluster detection. In order to perform the preliminary duty of variable cluster pattern recognition, an intermediate system which includes three subsystems is designed, and these subsystems are named as link extraction subsystem, variable clustering subsystem and cluster recognition subsystem, respectively. These subsystems together with the variable identification subsystem and hierarchical knowledge reorganisation subsystem, are integrated to construct the general variable predicting system, undertaking the task of feature variable detection at both singleton variable level and variable cluster level. The generic framework of the proposed model is illustrated in Fig. 7.4. The fundamental functionalities for each of the subsystems are as follows:

7.4 Intelligent System for Variable Detection and Hierarchical Knowledge Reorganisation

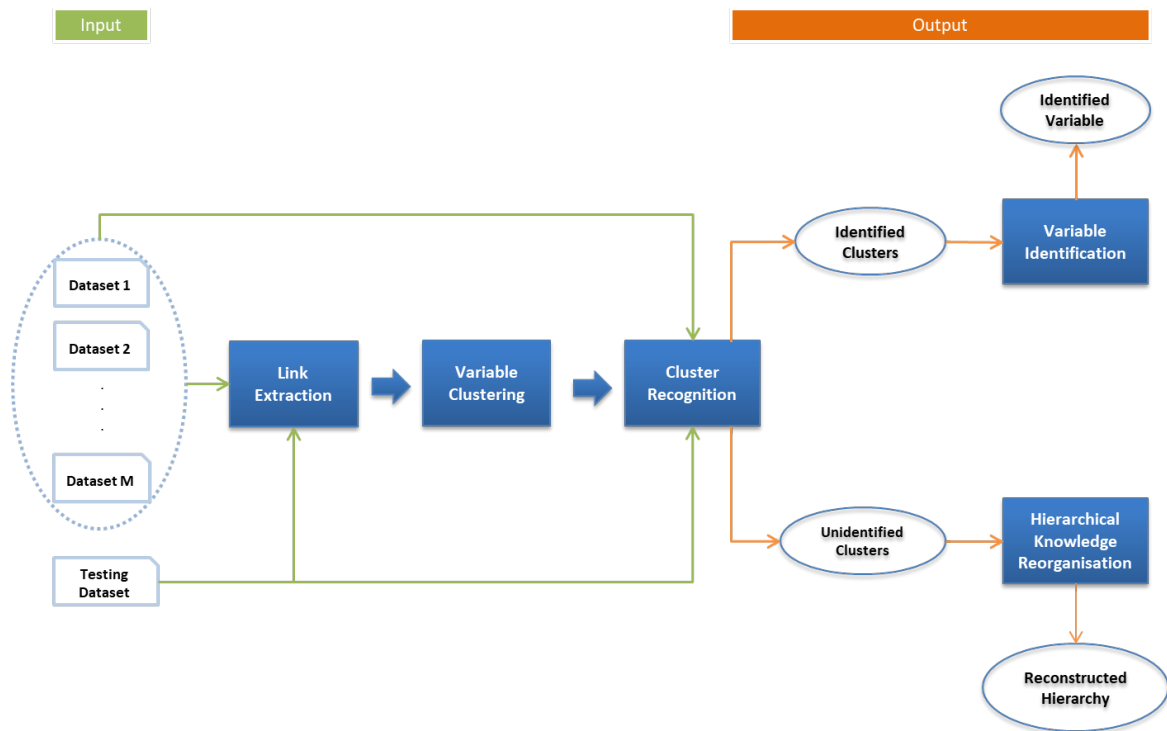


Fig. 7.4 Conceptual Framework of Cluster Based Variable Detection and Hierarchy Reorganisation

- **Link Extraction Subsystem:** Extract both existing and potential links between each pair of feature variables from the domain data corpus. Find the similarity degree for a pair of feature variables by measuring its related link weight. This step is executed for both the training datasets (datasets with already informed variables) and testing dataset (dataset with uninformed or unknown variables).
- **Variable Clustering Subsystem:** For an existing informed data corpus, group all the feature variables into clusters according to their similarity degree acquired from the previous step, so that the variables with higher similarity degrees are gathered together, and the variables with lower similarity degrees are separated into different categories. Likewise, perform the same action to the testing dataset to obtain another set of clusters with uninformed variables.
- **Cluster Recognition Subsystem:** Select an variable cluster from the uninformed cluster set generated by previous clustering step. Attempt to identify it by matching it to the appropriate informed variable cluster. This step may result in two opposite consequences: (1) the uninformed variable cluster is clearly or approximately identified; or (2) there exists no informed variable cluster matching this unknown variable cluster suitably.

- **Variable Identification Subsystem:** When criterion (1) is met, determine the subtree rooted from the informed variable cluster with its own hierarchical structure. For a feature variable within the uninformed variable cluster, iteratively search the obtained subtree structure from top to bottom to find the informed counterpart.
- **Hierarchical Knowledge Reorganisation Subsystem:** When criterion (2) is satisfied, insert the unidentified variable cluster into the hierarchy generated from the informed feature variables to consolidate the size of the hierarchical structure.

7.4.2 Empirical Study of Intelligent System

7.4.2.1 Data Preparation

The empirical study is examined on both real world data from UCI benchmark data sets [206] and on a collection of synthetic datasets. As there rarely exists any corpora of datasets designated for the current study, for a single dataset in UCI benchmark data repository, a series of steps is performed to generate proper data for carrying out the experimentation:

1. Split the entire dataset into a “training” set and a testing set with respect to instances.
2. Further split the “training” set into two separate parts with respect to feature variables, remove one part from the “training” set, and treat the remained one as the real training set.
3. Split the training set into subsets with overlapped variables according to human knowledge, where the instances in each subset may not necessarily represent the same objects or entities. These subsets are grouped together to form a training data corpus.

In the present experimentation, the ratio for the remaining feature variables is set to 0.8, which means that the 80% of the feature variables are employed for training, and all the variables within the entire dataset are used for testing the performance of the predicting model.

To demonstrate the performance of the proposed predicting model on larger sized groups of datasets including more variables, different corpora of synthetic datasets are generated as well to conduct the experiment. Table 7.4 provides a brief review on the characteristics of all datasets employed.

Table 7.4 Summary of Datasets: General Predicting Model

Dataset Collection	Type	NDC	ANIED	ANVED	ANVCTD
Bank	C	7	6459	4	2
Salary	C	6	5028	6	1
Student-Port	C	4	163	8	3
Connect-4	C	6	11034	7	2
Twitter	N	7	82038	13	4
Facebook	N	5	104	5	2
Urban	N	13	203	12	3
News	N	12	3048	7	2
Automobile	M	3	106	10	2
HeartDisease	M	5	100	16	2
Arrhythmia	M	10	121	30	3
Internet	M	7	1264	10	4
Synthetic-1	C	22	15491	18	4
Synthetic-3	N	25	21990	20	2
Synthetic-5	M	15	12020	15	2

^{1.} Type: C = Categorical Data; N = Numeric Data; M = Mixed-Type Data

^{2.} NDC: Number of Datasets in Collection

^{3.} ANIED: Average Number of Instances in Each Dataset

^{4.} ANVED: Average Number of Variables in Each Dataset

^{5.} ANVCTD: Average Number of Variables Co-existing in Two Datasets

7.4.2.2 Experimental Setup

Note that two possible operations may be conducted from the proposed predicting model (searching for singleton variables and updating the hierarchical structure for informed variable clusters), with each being exclusively performed. Thus, both operations are required to be examined. For investigation on performance of singleton variable identification, the accuracy index, defined as the ratio of number of correct predictions over the total number of predictions in searching for singleton feature variables, is employed as the indicator. The NMI measure presented in Section 7.3 is adopted here to gauge the performance on the task of variable hierarchy updating.

Specifically, for the proposed model, APCC is used to measure any correlation between numeric feature variables. Both Max-Min and BSAP metrics with COG step for connected-triple based fuzzy inference are employed to implement the link extraction subsystem, with WJC, LWP, RSS, SR being selected to perform comparison. In implementation of the variable clustering subsystem, EACA and EDCA are adopted to generate hierarchical structure for variable clusters, respectively, with AL being chosen as the linkage measure. Similar to the experimental setup introduced in Section 7.3, the cut level for cluster generation and the number of macro clusters for best describing the taxonomic information of domain variables, are determined with human knowledge. The threshold which guides to decide whether to perform singleton variable identification or hierarchical structure update is set to 0.9 for a numeric variable pair. This threshold is set to 0.6 when considering categorical variable pair or mixed-type variable pair. Moreover, LWP with EDCA for implementation of the framework is selected as the baseline model.

7.4.2.3 Experimental Results

In all experiments carried out, for each corpus of datasets, an n -fold cross validation [174] is conducted at instance level, where n is the number of datasets in the corpus. The following results are based on an average of running 10 times n -fold cross validation. Table 7.5 reveals the performance for each of the compared approaches regarding the accuracy of singleton variable identification. It is clear that the predicting model implemented with connected-triple based fuzzy inference as premise consistently outperforms the neighbour-based metrics (WJC) and the random walk-based metric (SR) across nearly all corpora of datasets. According to the statistics shown, the qualities of FCT implemented by Max-Min and BSAP are similar. Note that RSS and LWP metric can generate better results for specific corpora of datasets. As described in Chapter 5, this has much to do with the distribution

7.4 Intelligent System for Variable Detection and Hierarchical Knowledge Reorganisation

of the feature variables in each dataset within such a corpus. However, the proposed FCT metric demonstrates its competitiveness according to predicting accuracy, regardless of the feature variable distribution in each dataset corpus. This demonstrates the robustness and adaptability of the proposed approach.

Table 7.6 presents the average NMI which measures the quality of variable partition after updating the existing hierarchical structure. Again, the statistic figures show that the predicting model embedded with FCT to measure link strength are more effective than that employing other link inference approaches across the majority of data corpora in different types. Particularly, it reveals that different realisations of FCT (i.e., Man-Min and BSAP) have little variance on the performance of the general predicting system, which to a certain extent demonstrates the robustness of the proposed framework for link prediction to multiple implementations. In addition, according to the illustrated results, EDCA shows its advantage of working on smaller numbers of macro groups, while EACA are likely to perform well on partitions in a comparatively larger number.

Table 7.5 Comparison on Accuracy of Singleton Variable Identification

Data	$cut\alpha$	EACA										EDCA									
		Max-Min	BSAP	WJC	RSS	SR	LWP	Max-Min	BSAP	WJC	RSS	SR	LWP	Max-Min	BSAP	WJC	RSS	SR	LWP		
Bank	0.4	0.77(0.06) (v)	0.78(0.06) (v)	0.54(0.08) (*)	0.72(0.05)	0.66(0.06) (*)	0.74(0.05)	0.75(0.07)	0.58(0.06) (*)	0.69(0.07) (*)	0.71(0.07)	0.73(0.07)	0.75(0.07)	0.75(0.07)	0.58(0.06) (*)	0.69(0.07) (*)	0.71(0.07)	0.73(0.07)	0.75(0.07)		
Salary	0.5	0.81(0.08) (v)	0.83(0.07) (v)	0.69(0.06) (*)	0.78(0.07)	0.70(0.05) (*)	0.72(0.05) (*)	0.84(0.05)	0.75(0.04) (*)	0.73(0.06) (*)	0.68(0.06) (*)	0.78(0.05)	0.86(0.07) (*)	0.84(0.05)	0.79(0.05) (*)	0.78(0.05) (*)	0.72(0.07) (*)	0.89(0.06)	0.78(0.05)		
Student-por	0.4	0.82(0.06) (*)	0.81(0.06) (*)	0.77(0.05) (*)	0.74(0.05) (*)	0.68(0.04) (*)	0.84(0.07) (*)	0.86(0.07) (*)	0.886(0.07) (*)	0.886(0.07) (*)	0.72(0.07) (*)	0.886(0.07) (*)	0.886(0.07) (*)	0.886(0.07) (*)	0.79(0.05) (*)	0.78(0.05) (*)	0.72(0.07) (*)	0.89(0.06)	0.65(0.04)		
Connect-4	0.5	0.67(0.04)	0.67(0.04)	0.61(0.05) (*)	0.64(0.03)	0.58(0.05) (*)	0.62(0.06) (*)	0.72(0.03)	0.73(0.03) (v)	0.66(0.04)	0.62(0.04) (*)	0.74(0.05)	0.78(0.06)	0.74(0.05)	0.63(0.06)	0.66(0.04)	0.62(0.04) (*)	0.65(0.04)	0.74(0.07)		
Twitter	0.3	0.76(0.08)	0.75(0.08)	0.69(0.07) (*)	0.75(0.06)	0.72(0.07)	0.70(0.08) (*)	0.78(0.06)	0.72(0.07)	0.74(0.05)	0.77(0.06)	0.74(0.07)	0.78(0.06)	0.72(0.07)	0.74(0.05)	0.77(0.06)	0.77(0.06)	0.74(0.07)	0.66(0.06)		
Facebook	0.3	0.79(0.05) (v)	0.79(0.05) (v)	0.65(0.06)	0.75(0.06) (v)	0.74(0.05) (v)	0.68(0.05)	0.75(0.04) (v)	0.75(0.05) (v)	0.75(0.04) (v)	0.71(0.06) (v)	0.68(0.05)	0.75(0.04) (v)	0.75(0.05) (v)	0.59(0.05) (*)	0.75(0.04) (v)	0.71(0.06) (v)	0.66(0.06)	0.66(0.06)		
Urban	0.4	0.62(0.03)	0.63(0.03) (v)	0.51(0.04)	0.58(0.02) (v)	0.53(0.03)	0.58(0.04) (v)	0.56(0.05)	0.68(0.04)	0.68(0.04)	0.51(0.04)	0.68(0.04)	0.68(0.04)	0.68(0.04)	0.48(0.06) (*)	0.55(0.04)	0.71(0.05) (v)	0.68(0.04)	(*)0.54(0.03)		
news	0.4	0.71(0.04) (v)	0.72(0.04) (v)	0.64(0.05) (*)	0.70(0.05)	0.68(0.04)	0.62(0.05) (*)	0.68(0.04)	0.68(0.04)	0.68(0.04)	0.68(0.04)	0.62(0.05) (*)	0.68(0.04)	0.68(0.04)	0.67(0.06)	0.75(0.05) (v)	0.71(0.05) (v)	0.68(0.04)	0.68(0.04)		
Automobile	0.5	0.84(0.04) (v)	0.84(0.04) (v)	0.78(0.03)	0.75(0.05)	0.72(0.06) (*)	0.78(0.05)	0.81(0.05) (v)	0.82(0.05) (v)	0.82(0.05) (v)	0.73(0.05) (*)	0.78(0.05)	0.82(0.05) (v)	0.82(0.05) (v)	0.75(0.04)	0.72(0.04) (*)	0.73(0.05) (*)	0.77(0.04)	0.77(0.04)		
HeartDisease	0.3	0.69(0.08)	0.69(0.08)	0.59(0.07)	0.66(0.05)	0.64(0.08)	0.56(0.07)	0.66(0.07)	0.67(0.07)	0.67(0.07)	0.59(0.07)	0.56(0.07)	0.67(0.07)	0.67(0.07)	0.55(0.06)	0.60(0.06)	0.59(0.07)	0.54(0.08)	0.54(0.08)		
Arrhythmia	0.5	0.64(0.07) (v)	0.64(0.07) (v)	0.54(0.08) (v)	0.61(0.06) (v)	0.58(0.07) (v)	0.51(0.06) (v)	0.61(0.06) (v)	0.62(0.06) (v)	0.62(0.06) (v)	0.54(0.05) (v)	0.51(0.06) (v)	0.62(0.06) (v)	0.62(0.06) (v)	0.50(0.07)	0.54(0.07) (v)	0.54(0.05) (v)	0.48(0.07)	0.48(0.07)		
Internet	0.5	0.73(0.08) (v)	0.73(0.08) (v)	0.59(0.07) (*)	0.67(0.07)	0.70(0.06)	0.66(0.07)	0.76(0.07) (v)	0.76(0.07) (v)	0.68(0.06)	0.72(0.07) (v)	0.66(0.07)	0.76(0.07) (v)	0.65(0.08) (*)	0.68(0.06)	0.72(0.07) (v)	0.68(0.08)	0.68(0.08)	0.68(0.08)		
synthetic-1	0.4	0.78(0.05) (v)	0.78(0.05) (v)	0.69(0.07) (v)	0.70(0.06) (v)	0.75(0.07) (v)	0.73(0.05) (v)	0.74(0.06) (v)	0.74(0.06) (v)	0.74(0.06) (v)	0.73(0.06) (v)	0.73(0.05) (v)	0.74(0.06) (v)	0.74(0.06) (v)	0.63(0.07) (*)	0.62(0.05) (*)	0.73(0.06) (v)	0.66(0.06)	0.66(0.06)		
synthetic-3	0.3	0.84(0.03) (v)	0.84(0.03) (v)	0.70(0.04)	0.86(0.04) (v)	0.79(0.03) (v)	0.76(0.04) (v)	0.82(0.04) (v)	0.82(0.04) (v)	0.82(0.04) (v)	0.79(0.03) (v)	0.76(0.04) (v)	0.82(0.04) (v)	0.82(0.04) (v)	0.65(0.03) (*)	0.82(0.05) (v)	0.70(0.04)	0.71(0.03)	0.71(0.03)		
synthetic-5	0.4	0.80(0.06) (v)	0.81(0.06) (v)	0.67(0.05)	0.77(0.06) (v)	0.75(0.07) (v)	0.73(0.06) (v)	0.77(0.05) (v)	0.77(0.05) (v)	0.77(0.05) (v)	0.75(0.07) (v)	0.73(0.06) (v)	0.77(0.05) (v)	0.77(0.05) (v)	0.60(0.06) (*)	0.75(0.07) (v)	0.68(0.06)	0.68(0.06)	0.68(0.06)		

1. LWP with EDCA for implementation is set as the baseline model.

2. The best results are highlighted in boldface.

3. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

Table 7.6 Comparison on NMI for Hierarchical Structure Update

Data	cat _α	Num	EACA						EDCA					
			Max-Min	BSAP	WJC	RSS	SR	LWP	Max-Min	BSAP	WJC	RSS	SR	LWP
Bank	0.4	4	0.56(0.08)	0.56(0.08)	0.40(0.05) (*)	0.44(0.03) (*)	0.48(0.06) (*)	0.54(0.07) (*)	0.62(0.06) (v)	0.63(0.06) (v)	0.45(0.07) (*)	0.49(0.05) (*)	0.53(0.05) (*)	0.58(0.06)
Salary	0.5	4	0.63(0.06) (v)	0.63(0.06) (v)	0.49(0.05) (*)	0.52(0.08) (*)	0.46(0.05) (*)	0.48(0.06) (*)	0.67(0.05) (v)	0.67(0.05) (v)	0.45(0.06) (*)	0.63(0.07) (v)	0.52(0.05) (*)	0.58(0.07)
Student-por	0.4	5	0.42(0.04) (v)	0.42(0.04) (v)	0.31(0.06) (*)	0.44(0.05) (v)	0.39(0.06) (v)	0.35(0.05) (v)	0.40(0.06) (v)	0.40(0.06) (v)	0.33(0.05) (v)	0.36(0.06) (v)	0.33(0.05) (v)	0.34(0.04)
Connect-4	0.5	7	0.51(0.06) (v)	0.51(0.06) (v)	0.39(0.06)	0.48(0.05) (v)	0.48(0.07) (v)	0.41(0.06) (v)	0.45(0.05) (v)	0.45(0.05) (v)	0.35(0.05) (*)	0.40(0.06) (v)	0.43(0.06) (v)	0.39(0.06)
Twitter	0.3	7	0.46(0.05) (v)	0.46(0.05) (v)	0.27(0.04) (*)	0.50(0.06) (v)	0.45(0.05) (v)	0.33(0.06) (v)	0.42(0.06) (v)	0.42(0.06) (v)	0.28(0.06) (*)	0.49(0.08) (v)	0.39(0.05) (v)	0.31(0.07)
Facebook	0.3	6	0.69(0.07) (v)	0.69(0.07) (v)	0.55(0.05) (v)	0.64(0.05) (v)	0.68(0.06) (v)	0.59(0.07) (v)	0.64(0.05) (v)	0.64(0.05) (v)	0.53(0.06) (v)	0.58(0.06) (v)	0.60(0.07) (v)	0.56(0.06)
Urban	0.4	5	0.61(0.06) (v)	0.62(0.06) (v)	0.43(0.08) (*)	0.56(0.05) (v)	0.51(0.07) (v)	0.48(0.06) (v)	0.63(0.07) (v)	0.63(0.07) (v)	0.49(0.06) (v)	0.60(0.06) (v)	0.53(0.07) (v)	0.49(0.05)
news	0.4	6	0.74(0.04) (v)	0.75(0.04) (v)	0.52(0.06) (*)	0.68(0.06) (v)	0.70(0.05) (v)	0.61(0.06) (v)	0.70(0.05) (v)	0.70(0.05) (v)	0.50(0.05) (*)	0.66(0.04) (v)	0.66(0.05) (v)	0.63(0.04)
Automobile	0.5	5	0.59(0.06) (v)	0.59(0.06) (v)	0.48(0.04) (*)	0.45(0.06) (*)	0.50(0.04) (*)	0.51(0.06) (*)	0.61(0.05) (v)	0.61(0.05) (v)	0.50(0.06) (*)	0.52(0.05) (v)	0.54(0.04) (v)	0.54(0.05)
HeartDisease	0.3	6	0.49(0.08) (v)	0.49(0.08) (v)	0.40(0.07) (v)	0.43(0.07) (v)	0.47(0.08) (v)	0.41(0.06) (v)	0.46(0.07) (v)	0.46(0.07) (v)	0.34(0.05) (v)	0.36(0.04) (v)	0.41(0.06) (v)	0.36(0.06)
Arrhythmia	0.5	9	0.55(0.05) (v)	0.55(0.05) (v)	0.47(0.04) (*)	0.48(0.05) (v)	0.56(0.04) (v)	0.49(0.06) (v)	0.52(0.04) (v)	0.53(0.04) (v)	0.42(0.06) (*)	0.50(0.05) (v)	0.54(0.04) (v)	0.50(0.04)
Internet	0.5	8	0.70(0.07) (v)	0.71(0.07) (v)	0.42(0.09) (*)	0.66(0.08) (v)	0.63(0.07) (v)	0.56(0.08) (v)	0.61(0.05) (v)	0.61(0.05) (v)	0.35(0.06) (*)	0.54(0.07) (v)	0.55(0.08) (v)	0.46(0.07)
synthetic-1	0.4	6	0.64(0.06) (v)	0.64(0.06) (v)	0.45(0.05) (*)	0.66(0.04) (v)	0.61(0.05) (v)	0.53(0.06) (v)	0.58(0.05) (v)	0.58(0.05) (v)	0.42(0.06) (*)	0.62(0.05) (v)	0.54(0.07) (v)	0.50(0.07)
synthetic-3	0.3	8	0.72(0.05) (v)	0.72(0.05) (v)	0.60(0.06) (v)	0.70(0.04) (v)	0.70(0.05) (v)	0.66(0.05) (v)	0.68(0.06) (v)	0.68(0.06) (v)	0.58(0.07) (*)	0.66(0.06) (v)	0.65(0.04) (v)	0.62(0.04)
synthetic-5	0.4	8	0.61(0.07) (v)	0.61(0.07) (v)	0.42(0.05) (*)	0.52(0.07) (v)	0.55(0.06) (v)	0.48(0.05) (v)	0.57(0.06) (v)	0.57(0.06) (v)	0.45(0.07) (v)	0.51(0.06) (v)	0.52(0.05) (v)	0.46(0.06)

1. Num: Number of macro clusters.

2. LWP with EDCA for implementation is set as the baseline model.

3. The best results are highlighted in boldface.

4. Sign (*) / (v) indicates that the corresponding result is significantly worse / better than that on baseline model of 95% confidence level.

7.5 Summary

This chapter has provided two directions for performing further investigation on feature variable clusters: singleton variable identification and variable hierarchy update, according to different criteria. Both operations are based upon the general concept of binary tree structure. Particularly, the variable hierarchy update procedure can be iteratively performed to consolidate the knowledge base of feature variables within a given problem domain, which may also be exploited as a dynamic online training process, due to its light weight in computation. The implementation for both operations has been detailed in this chapter, with experimental evaluation being presented to demonstrate their efficacy.

Further more, from a general viewpoint, this chapter has generalised the contents discussed in Chapter 5 and Chapter 6, to present a comprehensive reasoning system which aims at executing prediction at variable and variable cluster level. The application scenario for the proposed predicting model has been outlined, with the functionality for each of the involving components described. An empirical study on the proposed predicting model has been carried out and reported, demonstrating its potential for future use.

Chapter 8

Conclusion

This chapter presents a final summary of the research on intelligent prediction at instance level, feature variable level and variable cluster level, as discussed in the preceding chapters. Based on the literature review regarding the popular techniques for prediction of various types, this thesis has provided systematic models to perform the task of instance based missing information estimation, feature variable identification, and variable group pattern recognition. To view from instance perspective, a hybrid model embedded with either hard or soft partition as the premise, guides the implementation of regression analysis in an elaborate manner. To view from feature variable perspective, hidden relationships between domain attributes from different data sources have been investigated via a data-driven approach. An effective hierarchical knowledge structure for feature variables has been established, being flexible and adaptable to update and reorganisation, while facilitating queries for singleton variables. The efficacy of the proposed framework has been experimentally validated, demonstrating its potential in real-world applications. Besides its promising prospect in future use, this chapter points out initial thought for the improvement and refinement of the proposed work.

8.1 Summary of Thesis

The knowledge premise for the proposed work discussed in the thesis has been provided by Chapter 2. Fundamentally, a survey of popular clustering techniques which enable identification of the underlying patterns of data has been extracted, with a description of their general concepts, implementation process and computational complexity analysis. A comprehensive study on existing link prediction techniques is presented, including a wide variety of metrics inspired through different intuitions, with their general framework and technical essentials described. Link prediction, being an emerging research field which attracts great attention in recent years, have set up as a solid foundation and inspired the proposed work. Fuzzy infer-

ence model and state-of-the-art fuzzy link analysis are briefly introduced, as its advantage in reasoning with imprecision and uncertainty has a natural appeal for the task of prediction. In general, the contents included in this chapter provides the basis upon which to develop the subsequent theoretical framework.

Chapter 3 has proposed a novel intelligent system aiming at predicting a missing feature value (in the numeric form) for an object being followed with interest. Initially, it gives a retrospective review on the downsides of the existing popular predicting methods of the similar type, and provides relevant application scenario for the proposed approach. Next, it presents the conceptual framework of the proposed predicting system, and describes the implementation for each of its subsystems in detail, with the computational complexity examined. This chapter further presents a significant application scenario for the proposed model on student academic performance, showing its advantage over compared estimating techniques, demonstrating its efficacy and effectiveness.

An alternative implementation of the proposed framework in Chapter 3 with granular refinement has been presented in Chapter 4, still focusing on the prediction of student academic performance. It improves the working of its predecessor by systematically considering factors with significant influence on student academic record via introducing an offset value generating scheme to the predicting model. Importantly, fuzzy clustering, a soft computing technique which enables accommodating both local and global information within the consideration area, takes the place of the hard clustering approach, thereby providing an informative basis for reasoning. This facilitates generating predicted results in a more readily interpretable manner.

Chapter 5 has started the route to perform investigation on prediction at variable level. It proposes a brand new scenario of measuring relationship between variables from diverse datasets. Initially, a group of two distinct datasets with common feature variables co-appearing in both of them is considered. Afterwards, a data corpus involving a number of datasets with similar associative characteristics is investigated heuristically. The common variables are regarded as a key to inferring the relationships hidden behind those examined variables. This data-driven approach for inter-variable correlation prediction works by exploiting the concept of connected-triples and implemented with fuzzy logic, which radically departs from its conventional crisp representation. As such, it possesses the advantage of being consistent with human logical thinking and generate interpretable predicted result. Through the exploitation of link strength measurements and fuzzy inference, the job of de-

tecting similar or related variables can be accomplished via examining link relation patterns within and across different data sources. Empirical evaluation results have revealed the potential of the proposed work in predicting interesting attribute relations, while involving simple computation mechanisms.

A brand new pattern recognition method for group investigation has been proposed in Chapter 6. In particular, It works on determining the underlying patterns for groups of uninformed feature variables. With the support information from specified data collection, such a task can be accomplished via searching for similar patterns between the uninformed variable group in relation to the informed counterpart. The link pattern prediction scheme presented in Chapter 5, sets up as a basis to assist partitioning feature variables (both informed and uninformed) into different groups, which is essential for the presented work at initial stage. Different metrics to handle data collections in various types (numeric, categorical and their mixed type) have been suggested, with experimental evaluation demonstrates the capability of the proposed techniques.

Chapter 7 has provided two topics for conducting further investigation on feature variable clusters: singleton variable identification and variable hierarchy update, according to different results provided by the variable cluster pattern recognition process. Both of the operations are built upon the basic concept of binary tree structure. Particularly, the variable hierarchy update procedure can be iteratively performed to consolidate the knowledge base of feature variables within the discussed domain, which can also be regarded as an online training process due to its computational efficiency. The implementation for both of the operations are detailed, with experimental evaluation being presented to demonstrate their efficacy. Moreover, Chapter 7 has presented a comprehensive reasoning system which aims at performing prediction at variable and variable cluster level within one common system. The application scenario for the proposed predicting model has been outlined, with the functionality for each of the involving components described. Empirical study on the proposed predicting model has been carried out, demonstrating its potential for future use.

8.2 Future Work

Despite its efficacy and great potential for future application, the proposed work so far in this thesis is open for further investigation. This section highlights interesting directions whose successful implementation will benefit current research significantly.

8.2.1 On Cluster Embedded Regression Analysis

Currently, the number of clusters determined for performing a clustering algorithm is provided by human experts with domain knowledge. Advanced techniques independent of human experts' advice is worth applying and investigating. In Section 5.1.3.3, the elbow method has been employed to determine the number of clusters to guide transforming the numeric data into its categorical counterpart. Besides the elbow method, in the literature, a number of techniques have been proposed to select the ideal k , where k stands for the number of clusters, such as Cross-validation, Bayesian Information based approach, and Silhouette Index based approach [43]. These methods each have their own merits and drawbacks, carefully examining them may help select the best k to improve the predicting accuracy.

The current implementation of the predicting system involves performing a clustering algorithm simply once to partition objects into different groups. However, theoretically, such partitioning may continue to run until all the instances in the training dataset are approximately fitted, with a better approximate fit. In order to solve the over fitting problem (well fit for the training instances but generating less accurate result for the testing data), a terminating criterion can be determined while no significant decrease is observed for the relative regression error.

8.2.2 On Student Academic Performance

While focusing on attributes closely related to academic modules, other important performance indicators, including practicals scores, oral presentation scores and tutor evaluation outcomes, should be considered. To integrate these factors, improved fuzzy clustering offset value generating techniques need to be developed. Also, their relative weighting may be important to be taken into account too. Note that the attributes in the entire dataset are currently chosen by human experts. An additional step to include feature selection [15] techniques in order to make an informed, automated choice of attributes to use would be very beneficial. In addition, how alternative clustering and classification mechanisms may perform in replacing the current use of simple methods is also very interesting to be investigated, although they may lose certain computational efficiency.

8.2.3 On Fuzzy Connected-Triple for Inference

Whilst significantly better performance is achieved by the present implementation over a wide range of compared methods, currently, the fuzzy membership functions used for expressing the linguistic terms are given by human experts. Automatic determination of fuzzy membership values by the use of certain clustering method may be employed to derive the required set of (potentially more objective) linguistic terms, if there is abundant historical data available. Existing literatures [235–237] may provide references to guiding the refinement from different perspectives.

Within the current implementation, while running the step of fuzzy inference, each connected-triple has been treated equally. A better approach might be to aggregate these connected-triples according to the importance of the individual centres of the triples, boosting the reliability of the detected links [238]. Alternative aggregating methods (e.g., arithmetic average and ordered weighted averaging as employed in [239] could be utilised for this). Also, the propagation and aggregation operations developed in social trust network [13, 240, 241] may be adapted for such use to enhance the reliability of the predicting system. Furthermore, the current study conducts an exhaustive search for all potential variable pairs; an aided heuristic metric for disclosing variable pairs with “strong” correlation may help to reduce the time complexity.

For link strength measurement, metrics other than those presently employed may be considered to further improve the modelling performance. Additionally, in the process of estimating correlation between a numeric-categorical variable pair, the current implementation to transform a numeric attribute variable into its categorical counterpart may lead to loss of valuable information from the original data source. Employing a more sophisticated technique for measuring mixed-type variable pair to replace the current approach is worth investigating. Moreover, in this work, only datasets involving categorical values, continuous numeric values and mixed of those are tested. Developing other types of link analysis strategy to handle more types of data (e.g., sequential data, image and visual data) is clearly desirable.

8.2.4 On Variable Cluster Pattern Recognition

Pattern recognition techniques generally aim at providing a reasonable answer for inputs by performing the “most likely” matching of the inputs, while taking into account their statistical variation. The current criterion set for the correct detection of uninformed variable clusters

is based on a hard bounded threshold of predicting accuracy, alternative soft definition of correct detection, such as “most of the attribute variables in the uninformed variable cluster are similar to most of their counterparts found in the informed variable cluster” may interpret the predicted result in a more comprehensible manner that is consistent with human logical thinking. Fuzzy granular computing [155] may offer an instruction for implementing such general concept.

The link weights used in the current hierarchical clustering model are represented in the format of crisp value, derived from the defuzzification step as discussed in Section 5.1.4.2. Note that such a defuzzification process may cause losing important information from its original fuzzy interpretation, an advanced clustering approach which exploits such fuzzy values directly to generate a more informative hierarchical structure is worth investigating. In particular, techniques with fuzzy distance based clustering [242] or clustering on fuzzy data [243] may provide insight to assist implementing this general concept.

Additionally, the current approach for generating feature variable clusters and comparing them is based on the computation of their statistical distributions, more information related to the domain questions may be acquired and included to perform a synthetic and comprehensive investigation on data collection. Advanced techniques in semantic information retrieval [244] and knowledge enrichment [245] may be employed for this.

8.2.5 On Hierarchical Knowledge Representation

The proposed update process for hierarchical knowledge structure can not guarantee that the hierarchy above the cut level is appropriately arranged, since bringing new feature variables into a cluster may change its original intra-similarity degree. A cut at a higher level of the hierarchy could possibly lead to a set of clusters with improper intra-variance indicated by the cut level. Thus, a further step to adjust the position of the macro clusters in the hierarchy which are affected by the updating process would be interesting to investigate.

When considering a set of undetected feature variable clusters to be inserted into an existing hierarchy, the order of insertion may affect the structure for the resulted hierarchical tree. In other words, such updated hierarchy is sensitive to the sequence of cluster insertions. It is desirable to design and implement an intelligent scheme to adapt to this scenario, so as to create a stable structure which is robust to the sequence of insertions.

Moreover, since there seldom exists prior knowledge about dataset corpus, it cannot be guaranteed that a feature variable under consideration exactly belongs to a single cluster. It might possess a certain membership in two or more variable clusters. Thus, fuzzy hierarchical clustering technique [246, 247], which is capable of interpreting such circumstances, can be adopted to generate soft clusters. More ambitiously, such membership values can be further utilised to construct more hierarchical structures linked with each other, resulting in a more complex structure such as “lattice” or “bush”, which enables accommodating more valuable information regarding groups of variables. Last but not least, automated detecting links between feature variable clusters coming from different hierarchical structures is very interesting but this remains as future research.

Appendix A

Publications Arising from Thesis

Several publications have been generated from the research carried out within the PhD project. The resultant publications closely related to the thesis is summarised below, including both papers for academic journals and conferences.

A.1 Journal Articles

1. Z Li, C Shang, Q Shen. Inter-variable correlation prediction with fuzzy connected-triples. *Soft Computing*, (2018)22: 7059-7072.
2. Z Li, C Shang, Q Shen. Variable and variable cluster recognition amongst multiple data sources, under review for publication, 2019.

A.2 Conference Papers

3. Z Li, C Shang, Q Shen. Fuzzy connected-triple for predicting inter-variable correlation. *Advances in Intelligent Systems and Computing: Contributions Presented at the 17th UK Workshop on Computational Intelligence, 2017 (Best Student Paper Award)*.
4. Z Li, C Shang, Q Shen. Fuzzy-clustering embedded regression for predicting student academic performance. *Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on*. IEEE, 2016: 344-351.
5. Z Li, C Shang, Q Shen. Intelligent prediction of student examination scores. Paper presented at 15th UK Workshop on Computational Intelligence, 2015, Exeter, UK.

Appendix B

Datasets Employed in Thesis

The datasets employed in the thesis are widely originated from the UCI machine learning repository [206], which have been collected and summarised from real-world problem scenarios and are generally public available. Other datasets are drawn from academic papers or artificially generated. Table B.1 presents a brief description for the properties of these employed datasets.

Table B.1 Summary of Data Used in Thesis

Dataset	Type	Feature	Instance
Arrhythmia	M	279	3630
Automobile	M	26	1040
Bank	C	25	45213
Connect-4	C	42	67557
Coverttype	M	54	581012
Facebook	N	19	500
HeartDisease	M	75	1598
Insurance	M	86	9000
Internet	M	72	10104
Mushroom	C	36	8124
Music	N	54	40909
News	N	84	21330
Salary	C	30	30168
SAP50A	M	3	50
SAP50B	M	3	50
Student-Mat	M	33	220
Student-Por	M	33	459
Twitter	N	78	966494
Synthetic-1	C	350	278838
Synthetic-2	C	540	496460
Synthetic-3	N	410	439800
Synthetic-4	N	560	358668
Synthetic-5	M	190	180000
Synthetic-6	M	340	201000
Urban	N	148	2436
Wine	N	13	3760

¹: Type: C = Categorical Data; N = Numeric Data; M = Mixed-Type Data

Appendix C

List of Acronyms

<i>n</i>-FCV	<i>n</i> -fold cross-validation
AAC	Adamic-Adar coefficient
AL	Average-Linkage
APCC	Absolute Pearson correlation coefficient
ASC	Average of Silhouette coefficient
BSAP	Bounded sum-Algebraic product
CELR	Clustering embedded linear regression
CL	Complete-Linkage
CN	Common neighbours
COG	Centre of gravity
CP	Connected-Path
CT	Commute time or Connected-Triple
EA	Evolutionary algorithm
EACA	Efficient agglomerative clustering algorithm
ECTS	European credit transfer system
ES	Exhaustive search

EDCA	Efficient divisive clustering algorithm
FMPT	Frequency of most popular term-pair
FCC	Fuzzy link prediction based on local clustering coefficient
FCELR	Fuzzy clustering embedded linear clustering
FCO	Fuzzy link prediction based on cluster overlapping
FCT	Fuzzy connected-triple
FOM	Fuzzy link prediction based on order-of-magnitude metric
FIS	Fuzzy inference system
HAC	Hierarchical agglomerative clustering
HPM	Hierarchical probabilistic model
HT	Hitting time
JC	Jaccard coefficient
KMA	Kuhn-Munkres algorithm
KNN	<i>K</i> -Nearest-Neighbours
LCC	Local clustering coefficient
LP	Local path
LWP	Local weighted path
MCMC	Markov Chain Monte Carlo
MFMA	Max flow matching algorithm
MI	Mutual information
MIC	Maximal information coefficient
NA	Network analysis
NCT	Normalised commute time
NHT	Normalised hitting time

NMI	Normalised mutual information
OD	Overlapping degree
PA	Preferential attachment
PAM	Partitioning around medoids
PF	PropFlow
RP	Rooted Pagerank
PS	PageSim
RA	Resource allocation
ROCK	Robust clustering using links
RSS	Relation strength similarity
SBM	Stochastic block model
SC	Spectral clustering
SL	Single-Linkage
SMLR	Simple multi-variable linear regression
SMO	Sequential minimal optimization
SNA	Social network analysis
SR	SimRank
SSE	Sum of square errors
SVM	Support vector machines
SYN	Synthetic weight
UCI	University of California, Irvine
WJC	Weighted Jaccard coefficient
WRA	Weighted resource allocation

References

- [1] Aliaksei Sandryhaila and Jose MF Moura. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014.
- [2] Foster Provost and Tom Fawcett. Data science and its relationship to big data and data-driven decision making. *Big data*, 1(1):51–59, 2013.
- [3] Efraim Turban, Ramesh Sharda, and Dursun Delen. Decision support and business intelligence systems (required). *Google Scholar*, 2010.
- [4] Chengqi Zhang and Shichao Zhang. *Association rule mining: models and algorithms*. Springer-Verlag, 2002.
- [5] David Silverman. *Interpreting qualitative data*. Sage, 2015.
- [6] Christopher Westphal and Teresa Blaxton. Data mining solutions: methods and tools for solving real-world problems. 1998.
- [7] David J Hand. Principles of data mining. *Drug safety*, 30(7):621–622, 2007.
- [8] Kevin Dunbar. How scientists think in the real world: Implications for science education. *Journal of Applied Developmental Psychology*, 21(1):49–58, 2000.
- [9] Colin Robson and Kieran McCartan. *Real world research*. John Wiley & Sons, 2016.
- [10] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.
- [11] Hsi-Mei Hsu and Chen-Tung Chen. Aggregation of fuzzy opinions under group decision making. *Fuzzy sets and systems*, 79(3):279–285, 1996.
- [12] Nico Potyka and Matthias Thimm. Probabilistic reasoning with inconsistent beliefs using inconsistency measures. In *IJCAI*, pages 3156–3163, 2015.
- [13] Patricia Victor, Chris Cornelis, and Martine De Cock. *Trust networks for recommender systems*, volume 4. Springer Science & Business Media, 2011.
- [14] Shin-ya Kobayashi, Andrzej Piegat, Jerzy Pejaś, Imed El Fray, and Janusz Kacprzyk. *Hard and Soft Computing for Artificial Intelligence, Multimedia and Security*. Springer, 2017.

-
- [15] Ren Diao. *Feature selection with harmony search and its applications*. PhD thesis, Aberystwyth University, February 2014. URL <http://hdl.handle.net/2160/13457>.
- [16] Vasant Dhar. Data science and prediction. *Communications of the ACM*, 56(12): 64–73, 2013.
- [17] Charles Nyce and A Cpcu. Predictive analytics white paper. *American Institute for CPCU. Insurance Institute of America*, pages 9–10, 2007.
- [18] Jasmine Zakir, Tom Seymour, and Kristi Berg. Big data analytics. *Issues in Information Systems*, 16(2), 2015.
- [19] Keith A Marill. Advanced statistics: linear regression, part ii: multiple linear regression. *Academic emergency medicine*, 11(1):94–102, 2004.
- [20] George AF Seber and Alan J Lee. *Linear regression analysis*, volume 329. John Wiley & Sons, 2012.
- [21] Theodore B Trafalis and Huseyin Ince. Support vector machine for regression and applications to financial forecasting. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 6, pages 348–353. IEEE, 2000.
- [22] A Pérez, V Sánchez, R Baeza, MC Zamora, and J Chirife. Literature review on linear regression equations for relating water activity to moisture content in floral honeys: development of a weighted average equation. *Food and bioprocess technology*, 2(4): 437, 2009.
- [23] Astrid Schneider, Gerhard Hommel, and Maria Blettner. Linear regression analysis: part 14 of a series on evaluation of scientific publications. *Deutsches Ärzteblatt International*, 107(44):776, 2010.
- [24] Chenping Hou, Feiping Nie, Dongyun Yi, and Yi Wu. Feature selection via joint embedding learning and sparse regression. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1324, 2011.
- [25] Mohammad Goodarzi, Matheus P Freitas, and Richard Jensen. Feature selection and linear/nonlinear regression methods for the accurate prediction of glycogen synthase kinase-3b inhibitory activities. *Journal of chemical information and modeling*, 49(4): 824–832, 2009.
- [26] Roger J Lewis. An introduction to classification and regression tree (cart) analysis. In *Annual meeting of the society for academic emergency medicine in San Francisco, California*, volume 14, 2000.
- [27] Peter Müller, Fernando Quintana, and Gary L Rosner. A product partition model with regression on covariates. *Journal of Computational and Graphical Statistics*, 20(1): 260–278, 2011.
- [28] Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. John wiley & sons, 2005.

-
- [29] Samprit Chatterjee and Ali S Hadi. *Sensitivity analysis in linear regression*, volume 327. John Wiley & Sons, 2009.
- [30] Paul T Von Hippel. 4. regression with missing ys: An improved strategy for analyzing multiply imputed data. *Sociological Methodology*, 37(1):83–117, 2007.
- [31] Paul D Allison. Multiple imputation for missing data: A cautionary tale. *Sociological methods & research*, 28(3):301–309, 2000.
- [32] Brian S Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. Hierarchical clustering. *Cluster analysis*, 5, 2011.
- [33] Sanjiv K Bhatia and Jitender S Deogun. Conceptual clustering in information retrieval. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(3):427–436, 1998.
- [34] Sadaaki Miyamoto. *Fuzzy sets in information retrieval and cluster analysis*, volume 4. Springer Science & Business Media, 2012.
- [35] Andrea Baraldi and Palma Blonda. A survey of fuzzy clustering algorithms for pattern recognition. i. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(6):778–785, 1999.
- [36] Robert J Schalkoff. Pattern recognition. *Wiley encyclopedia of computer science and engineering*, 2007.
- [37] James C Bezdek, James Keller, Raghu Krishnapuram, and Nikhil Pal. *Fuzzy models and algorithms for pattern recognition and image processing*, volume 4. Springer Science & Business Media, 1999.
- [38] Keh-Shih Chuang, Hong-Long Tzeng, Sharon Chen, Jay Wu, and Tzong-Jer Chen. Fuzzy c-means clustering with spatial information for image segmentation. *computerized medical imaging and graphics*, 30(1):9–15, 2006.
- [39] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of computational biology*, 6(3-4):281–297, 1999.
- [40] Javier Herrero, Alfonso Valencia, and Joaquin Dopazo. A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, 17(2):126–136, 2001.
- [41] Badrul M Sarwar, George Karypis, Joseph Konstan, and John Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the fifth international conference on computer and information technology*, volume 1, pages 291–324, 2002.
- [42] Kyoung-jae Kim and Hyunchul Ahn. A recommender system using ga k-means clustering in an online shopping market. *Expert systems with applications*, 34(2):1200–1209, 2008.
- [43] Trupti M Kodinariya and Prashant R Makwana. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95, 2013.

-
- [44] Stef Van Buuren and Willem J Heiser. Clustering objects into groups under optimal scaling of variables. *Psychometrika*, 54(4):699–706, 1989.
- [45] Haixun Wang, Wei Wang, Jiong Yang, and Philip S Yu. Clustering by pattern similarity in large data sets. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 394–405. ACM, 2002.
- [46] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, volume 4, pages 9–56, 2008.
- [47] Natthakan Iam-On, Tossapon Boongoen, Simon Garrett, and Chris Price. A link-based approach to the cluster ensemble problem. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2396–2409, 2011.
- [48] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.
- [49] Ben Taskar, Ming-Fai Wong, Pieter Abbeel, and Daphne Koller. Link prediction in relational data. In *Advances in neural information processing systems*, pages 659–666, 2004.
- [50] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.
- [51] Mark EJ Newman and Juyong Park. Why social networks are different from other types of networks. *Physical Review E*, 68(3):036122, 2003.
- [52] Luca Maria Aiello, Alain Barrat, Rossano Schifanella, Ciro Cattuto, Benjamin Markines, and Filippo Menczer. Friendship prediction and homophily in social media. *ACM Transactions on the Web (TWEB)*, 6(2):9, 2012.
- [53] Cuneyt Gurcan Akcora, Barbara Carminati, and Elena Ferrari. Network and profile based measures for user similarities on social networks. In *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, pages 292–298. IEEE, 2011.
- [54] Junichiro Mori, Yuya Kajikawa, Hisashi Kashima, and Ichiro Sakata. Machine learning approach for finding business partners and building reciprocal relationships. *Expert Systems with Applications*, 39(12):10402–10407, 2012.
- [55] Sen Wu, Jimeng Sun, and Jie Tang. Patent partner recommendation in enterprise social networks. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 43–52. ACM, 2013.
- [56] Milen Pavlov and Ryutaro Ichise. Finding experts by link prediction in co-authorship networks. In *Proceedings of the 2nd International Conference on Finding Experts on the Web with Semantics*, volume 290, pages 42–55. Citeseer, 2007.

-
- [57] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology*, 58(7): 1019–1031, 2007.
- [58] Stanley Wasserman and Joseph Galaskiewicz. *Advances in social network analysis: Research in the social and behavioral sciences*, volume 171. Sage Publications, 1994.
- [59] Wadhah Almansoori, Shang Gao, Tamer N Jarada, Abdallah M Elsheikh, Ayman N Murshed, Jamal Jida, Reda Alhadj, and Jon Rokne. Link prediction and classification in social networks and its application in healthcare and systems biology. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 1(1-2):27–36, 2012.
- [60] Andrea Franceschini, Damian Szklarczyk, Sune Frankild, Michael Kuhn, Milan Simonovic, Alexander Roth, Jianyi Lin, Pablo Minguez, Peer Bork, Christian Von Mering, et al. String v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic acids research*, 41(D1):D808–D815, 2012.
- [61] Frédéric Varone, Karin Ingold, Charlotte Jourdain, and Volker Schneider. Studying policy advocacy through social network analysis. *European political science*, 16: 322–336, 2017.
- [62] Sung-Heui Bae, Alexander Nikolaev, Jin Young Seo, and Jessica Castner. Health care provider social network analysis: a systematic review. *Nursing outlook*, 63(5): 566–584, 2015.
- [63] Xian Zheng, Yun Le, Albert PC Chan, Yi Hu, and Yongkui Li. Review of the application of social network analysis (sna) in construction project management research. *International journal of project management*, 34(7):1214–1225, 2016.
- [64] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308, 2006.
- [65] Qiang Shen and Tossapon Boongoen. Extending data reliability measure to a filter approach for soft subspace clustering. *IEEE Transactions on Knowledge and Data Engineering*, 24:649–664, 2012.
- [66] Qiang Shen and Tossapon Boongoen. Social network inspired approach to intelligent monitoring of intelligence data. In *Social Network Mining, Analysis and Research Trends: Techniques and Applications*, volume 24, pages 79–100. IGI Publishing, 2012.
- [67] Steven T Zech and Michael Gabbay. Social network analysis in the study of terrorism and insurgency: From organization to politics. *International Studies Review*, 18(2): 214–243, 2016.
- [68] Andrea Fronzetti Colladon and Elisa Remondi. Using social network analysis to prevent money laundering. *Expert Systems with Applications*, 67:49–58, 2017.
- [69] Zan Huang and Dennis KJ Lin. The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing*, 21(2): 286–303, 2009.

-
- [70] Natthakan Iam-On and Tossapon Boongoen. Comparative study of matrix refinement approaches for ensemble clustering. *Machine Learning*, 98(1-2):269–300, 2015.
- [71] Heitor Murilo Gomes, Jean Paul Barddal, Fabrício Enembreck, and Albert Bifet. A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)*, 50(2):23, 2017.
- [72] William E Winkler. Overview of record linkage and current research directions. In *Bureau of the Census*. Citeseer, 2006.
- [73] Stacie B Dusetzina, Seth Tyree, Anne-Marie Meyer, Adrian Meyer, Laura Green, and William R Carpenter. Linking data for health services research: a framework and instructional guide. 2014.
- [74] _____, Changjing Shang, and Qiang Shen. Intelligent prediction of student examination scores. In *Proceedings of 15th UK Workshop on Computational Intelligence*, 2015.
- [75] _____, Changjing Shang, and Qiang Shen. Fuzzy-clustering embedded regression for predicting student academic performance. In *Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on*, pages 344–351. IEEE, 2016.
- [76] _____, Changjing Shang, and Qiang Shen. Fuzzy connected-triple for predicting inter-variable correlation. In *Proceedings of the 17th annual workshop on computational intelligence*, pages 49–62, 2017.
- [77] _____, Changjing Shang, and Qiang Shen. Inter-variable correlation prediction with fuzzy connected-triples. *Soft Computing*, 22(21):7059–7072, 2018.
- [78] Pavel Berkhin. A survey of clustering data mining techniques. In *Grouping multidimensional data*, pages 25–71. Springer, 2006.
- [79] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.
- [80] Leonard Kaufman and Peter J Rousseeuw. Partitioning around medoids (program pam). *Finding groups in data: an introduction to cluster analysis*, pages 68–125, 1990.
- [81] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. *Information systems*, 25(5):345–366, 2000.
- [82] Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.
- [83] Zengyou He, Xiaofei Xu, and Shengchun Deng. Squeezer: an efficient algorithm for clustering categorical data. *Journal of Computer Science and Technology*, 17(5): 611–624, 2002.
- [84] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

-
- [85] Alex Pothen, Horst D Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM journal on matrix analysis and applications*, 11(3): 430–452, 1990.
- [86] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.
- [87] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2005.
- [88] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416, 2007.
- [89] Donghui Yan, Ling Huang, and Michael I Jordan. Fast approximate spectral clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 907–916. ACM, 2009.
- [90] Matthew A Jaro. Probabilistic linkage of large public health data files. *Statistics in medicine*, 14(5-7):491–498, 1995.
- [91] Luis Gravano, Panagiotis G. Ipeirotis, Hosagrahar Visvesvaraya Jagadish, Nick Koudas, Shanmugaelayut Muthukrishnan, Lauri Pietarinen, and Divesh Srivastava. Using q-grams in a dbms for approximate string processing. *IEEE Data Eng. Bull.*, 24 (4):28–34, 2001.
- [92] Gonzalo Navarro. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88, 2001.
- [93] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. Effects of user similarity in social media. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 703–712. ACM, 2012.
- [94] Cuneyt Gurcan Akcora, Barbara Carminati, and Elena Ferrari. User similarities on social networks. *Social Network Analysis and Mining*, 3(3):475–495, 2013.
- [95] Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1):1–38, 2015.
- [96] Mark EJ Newman. Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102, 2001.
- [97] Gobinda G Chowdhury. *Introduction to modern information retrieval*. Facet publishing, 2010.
- [98] Evgenia Dimitriadou, Markus Barth, Christian Windischberger, Kurt Hornik, and Ewald Moser. A quantitative comparison of functional mri cluster analysis. *Artificial intelligence in medicine*, 31(1):57–71, 2004.
- [99] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

-
- [100] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.
- [101] Michael Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet mathematics*, 1(2):226–251, 2004.
- [102] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.
- [103] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.
- [104] Linyuan Lü, Ci-Hang Jin, and Tao Zhou. Similarity index based on local paths for link prediction of complex networks. *Physical Review E*, 80(4):046122, 2009.
- [105] Danh Bui Thi, Ryutaro Ichise, and Bac Le. Link prediction in social networks based on local weighted paths. In *Future Data and Security Engineering*, pages 151–163. Springer, 2014.
- [106] Hung-Hsuan Chen, Liang Gou, Xiaolong Luke Zhang, and C Lee Giles. Discovering missing links in networks using vertex similarity measures. In *Proceedings of the 27th annual ACM symposium on applied computing*, pages 138–143. ACM, 2012.
- [107] Yuxiao Dong, Jing Zhang, Jie Tang, Nitesh V Chawla, and Bai Wang. Coupledlp: Link prediction in coupled networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 199–208. ACM, 2015.
- [108] Jinzhu Zhang. Uncovering mechanisms of co-authorship evolution by multirelations-based link prediction. *Information Processing & Management*, 53(1):42–51, 2017.
- [109] Tossapon Boongoen, Qiang Shen, and Chris Price. Disclosing false identity through hybrid link analysis. *Artificial Intelligence and Law*, 18(1):77–102, 2010.
- [110] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.
- [111] Zhenjiang Lin, Irwin King, and Michael R Lyu. Pagesim: A novel link-based similarity measure for the world wide web. In *Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on*, pages 687–693. IEEE, 2006.
- [112] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [113] David F Gleich. Pagerank beyond the web. *SIAM Review*, 57(3):321–363, 2015.
- [114] Graham Brightwell and Peter Winkler. Maximum hitting time for random walks on graphs. *Random Structures & Algorithms*, 1(3):263–276, 1990.
- [115] Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 243–252. ACM, 2010.

-
- [116] Virinchi Srinivas and Pabitra Mitra. *Link prediction in social networks: role of power law distribution*. Springer, 2016.
- [117] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.
- [118] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.
- [119] Roger Guimera and Luis A Nunes Amaral. Functional cartography of complex metabolic networks. *nature*, 433(7028):895, 2005.
- [120] Harrison C White, Scott A Boorman, and Ronald L Breiger. Social structure from multiple networks. i. blockmodels of roles and positions. *American journal of sociology*, 81(4):730–780, 1976.
- [121] Patrick Doreian, Vladimir Batagelj, and Anuska Ferligoj. *Generalized blockmodeling*, volume 25. Cambridge university press, 2005.
- [122] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.
- [123] Roger Guimera, Marta Sales-Pardo, and Luis AN Amaral. Classes of complex networks defined by role-to-role connectivity profiles. *Nature physics*, 3(1):63, 2007.
- [124] Yen-Liang Chen, Li-Chen Cheng, and Ching-Nan Chuang. A group recommendation system with consideration of interactions among group members. *Expert systems with applications*, 34(3):2082–2090, 2008.
- [125] Zhen Liu, Qian-Ming Zhang, Linyuan Lü, and Tao Zhou. Link prediction in complex networks: A local naive bayes model. *EPL (Europhysics Letters)*, 96(4):48007, 2011.
- [126] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.
- [127] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.
- [128] Roger Guimera, Stefano Mossa, Adrian Turttschi, and LA Nunes Amaral. The worldwide air transportation network: Anomalous centrality, community structure, and cities’ global roles. *Proceedings of the National Academy of Sciences*, 102(22):7794–7799, 2005.
- [129] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.
- [130] Jennifer L Reed, Thuy D Vo, Christophe H Schilling, and Bernhard O Palsson. An expanded genome-scale model of escherichia coli k-12 (i jr904 gsm/gpr). *Genome biology*, 4(9):R54, 2003.

-
- [131] Roger Guimerà and Marta Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52):22073–22078, 2009.
- [132] Tianhua Chen. *Induction of accurate and interpretable fuzzy rules*. 2017.
- [133] Ebrahim H Mamdani. Application of fuzzy algorithms for control of simple dynamic plant. In *Proceedings of the institution of electrical engineers*, volume 121, pages 1585–1588. IET, 1974.
- [134] Michio Sugeno. *Industrial applications of fuzzy control*. Elsevier Science Inc., 1985.
- [135] J-SR Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, 23(3):665–685, 1993.
- [136] Gang Feng. A survey on analysis and design of model-based fuzzy control systems. *IEEE Transactions on Fuzzy systems*, 14(5):676–697, 2006.
- [137] J Maiers and Yosef S Sherif. Applications of fuzzy set theory. *IEEE Transactions on Systems, Man, and Cybernetics*, (1):175–189, 1985.
- [138] Michio Sugeno. An introductory survey of fuzzy control. *Information sciences*, 36(1-2):59–83, 1985.
- [139] Yan Zhao and Huijun Gao. Fuzzy-model-based control of an overhead crane with input delay and actuator saturation. *IEEE Transactions on Fuzzy Systems*, 20(1):181–186, 2012.
- [140] María José Gacto, Marta Galende, Rafael Alcalá, and Francisco Herrera. Metsk-hde: A multiobjective evolutionary algorithm to learn accurate tsf-fuzzy systems in high-dimensional and large-scale regression problems. *Information Sciences*, 276:63–79, 2014.
- [141] W Pedrycz and Athanasios V Vasilakos. Linguistic models and linguistic modeling. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(6):745–757, 1999.
- [142] Hisao Ishibuchi, Shingo Mihara, and Yusuke Nojima. Parallel distributed hybrid fuzzy gbml models with rule set migration and training data rotation. *IEEE Transactions on fuzzy systems*, 21(2):355–368, 2013.
- [143] Jesús Alcalá-Fdez, Rafael Alcalá, María José Gacto, and Francisco Herrera. Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets and Systems*, 160(7):905–921, 2009.
- [144] Jing Tao Yao, Athanasios V Vasilakos, and Witold Pedrycz. Granular computing: perspectives and challenges. *IEEE Transactions on Cybernetics*, 43(6):1977–1989, 2013.
- [145] Clarence W De Silva. *Intelligent control: fuzzy logic applications*. CRC press, 2018.
- [146] Zdenko Kovacic and Stjepan Bogdan. *Fuzzy controller design: theory and applications*. CRC press, 2005.

-
- [147] Seda Sahin, Mehmet R Tolun, and Reza Hassanpour. Hybrid expert systems: A survey of current approaches and applications. *Expert systems with applications*, 39(4):4609–4617, 2012.
- [148] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *nature*, 435(7043):814, 2005.
- [149] Tamás Nepusz, Andrea Petróczy, László Négyessy, and Fülöp Bazsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77(1):016107, 2008.
- [150] Shihua Zhang, Rui-Sheng Wang, and Xiang-Sun Zhang. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications*, 374(1):483–490, 2007.
- [151] Matteo Brunelli and Michele Fedrizzi. A fuzzy approach to social network analysis. In *2009 International conference on advances in social network analysis and mining*, pages 225–230. IEEE, 2009.
- [152] Jonathan L Gross and Jay Yellen. *Graph theory and its applications*. Chapman and Hall/CRC, 2005.
- [153] Alain Barrat, Marc Barthelemy, Romualdo Pastor-Satorras, and Alessandro Vespignani. The architecture of complex weighted networks. *Proceedings of the national academy of sciences*, 101(11):3747–3752, 2004.
- [154] Susan Bastani, Ahmad Khalili Jafarabad, and Mohammad Hossein Fazel Zarandi. Fuzzy models for link prediction in social networks. *International Journal of Intelligent Systems*, 28(8):768–786, 2013.
- [155] Ronald R Yager. Intelligent social network analysis using granular computing. *International Journal of Intelligent Systems*, 23(11):1197–1219, 2008.
- [156] Duncan J Watts and Steven H Strogatz. Collective dynamics of small world networks. *nature*, 393(6684):440, 1998.
- [157] Qiang Shen and Tossapon Boongoen. Fuzzy orders-of-magnitude-based link analysis for qualitative alias detection. *IEEE Transactions on Knowledge and Data Engineering*, 24(4):649–664, 2012.
- [158] Tossapon Boongoen, Qiang Shen, and Chris Price. Fuzzy qualitative link analysis for academic performance evaluation. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 19(03):559–585, 2011.
- [159] Tossapon Boongoen and Qiang Shen. Order-of-magnitude based link analysis for false identity detection. In *Proceedings of the 23rd International Workshop on Qualitative Reasoning*, pages 7–15, 2009.
- [160] P-T Chang, K-C Hung, K-P Lin, and C-H Chang. A comparison of discrete algorithms for fuzzy weighted average. *IEEE Transactions on Fuzzy Systems*, 14(5):663–675, 2006.

-
- [161] John Neter, William Wasserman, and Michael H Kutner. Applied linear regression models. 1989.
- [162] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
- [163] János Podani. New combinatorial clustering methods. In *Numerical syntaxonomy*, pages 61–77. Springer, 1989.
- [164] Pan Su, Changjing Shang, and Qiang Shen. Link-based approach for bibliometric journal ranking. *Soft Computing*, 17(12):2399–2410, 2013.
- [165] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.
- [166] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10: 207–244, 2009.
- [167] Leona S Aiken, Stephen G West, and Steven C Pitts. Multiple linear regression. *Handbook of psychology*, 2003.
- [168] Jeng-Fung Chen and Quang Hung Do. Training neural networks to predict student academic performance: A comparison of cuckoo search and gravitational search algorithms. *International Journal of Computational Intelligence and Applications*, 13 (01):1450005, 2014.
- [169] VO Oladokun, AT Adebajo, and OE Charles-Owaba. Predicting students academic performance using artificial neural network: A case study of an engineering course. *The Pacific Journal of Science and Technology*, 9(1):72–79, 2008.
- [170] Indriana Hidayah, Adhistya Erna Permanasari, and Ning Ratwastuti. Student classification for academic performance prediction using neuro fuzzy in a conventional classroom. In *Information Technology and Electrical Engineering (ICITEE), 2013 International Conference on*, pages 221–225. IEEE, 2013.
- [171] Michael Fire, Gilad Katz, Yuval Elovici, Bracha Shapira, and Lior Rokach. Predicting student exams scores by analyzing social network data. In *Active Media Technology*, pages 584–595. Springer, 2012.
- [172] Khairul A Rasmani and Qiang Shen. Data-driven fuzzy rule generation and its application for student academic performance evaluation. *Applied intelligence*, 25(3): 305–319, 2006.
- [173] P. Cortez and A. Silva. Using data mining to predict secondary school student performance. In *Proceedings of 5th Future Business Technology Conference (FUBUTECH 2008)*, pages 5–12, Porto, Portugal, April 2008. EUROSIS.
- [174] Yoshua Bengio and Yves Grandvalet. Bias in estimating the variance of k-fold cross-validation. In *Statistical modeling and analysis for complex data problems*, pages 75–95. Springer, 2005.

-
- [175] Bertan Ari and H Altay Güvenir. Clustered linear regression. *Knowledge-Based Systems*, 15(3):169–175, 2002.
- [176] AT Chamillard. Using student performance predictions in a computer science curriculum. In *ACM SIGCSE Bulletin*, volume 38, pages 260–264. ACM, 2006.
- [177] Charles G Petersen and Trevor G Howe. Predicting academic success in introduction to computers. *AEDS Journal*, 12(4):182–191, 1979.
- [178] Timothy P Cronan, Phillip R Embry, and Steven D White. Identifying factors that influence performance of non-computing majors in the business computer information systems course. *Journal of Research on Computing in Education*, 21(4):431–446, 1989.
- [179] Richard F Deckro and Henry W Woundenberg. Mba admission criteria and academic success. *Decision Sciences*, 8(4):765–769, 1977.
- [180] Siu-Man Raymond Ting and Tracy L Robinson. First-year academic success: A prediction combining cognitive and psychosocial variables for caucasian and african american students. *Journal of college student development*, 1998.
- [181] Jens Bennedsen and Michael E Caspersen. Optimists have more fun, but do they learn better? on the influence of emotional and social factors on learning introductory computer science. *Computer Science Education*, 18(1):1–16, 2008.
- [182] Kevin J Keen and Letha Etzkorn. Predicting students’ grades in computer science courses based on complexity measures of teacher’s lecture notes. *Journal of Computing Sciences in Colleges*, 24(5):44–48, 2009.
- [183] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.
- [184] Richard J Hathaway and James C Bezdek. Recent convergence results for the fuzzy c-means clustering algorithms. *Journal of Classification*, 5(2):237–247, 1988.
- [185] John F Kolen and Tim Hutcheson. Reducing the time complexity of the fuzzy c-means algorithm. *Fuzzy Systems, IEEE Transactions on*, 10(2):263–267, 2002.
- [186] Charles Desforges and Alberto Abouchaar. The impact of parental involvement, parental support and family education on pupil achievements and adjustment: A literature review. Technical report, Research report, 2003.
- [187] James K Luiselli, Robert F Putnam, Marcie W Handler, and Adam B Feinberg. Whole-school positive behaviour support: effects on student discipline problems and academic performance. *Educational psychology*, 25(2-3):183–198, 2005.
- [188] Bart Rienties and Lisette Toetenel. The impact of learning design on student behaviour, satisfaction and performance: A cross-institutional comparison across 151 modules. *Computers in Human Behavior*, 60:333–341, 2016.
- [189] Wen-June Wang. New similarity measures on fuzzy sets and on elements. *Fuzzy sets and systems*, 85(3):305–309, 1997.

-
- [190] Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.
- [191] Soumen Chakrabarti, Martin Ester, Usama Fayyad, Johannes Gehrke, Jiawei Han, Shinichi Morishita, Gregory Piatetsky-Shapiro, and Wei Wang. Data mining curriculum: A proposal (version 1.0). *Intensive Working Group of ACM SIGKDD Curriculum Committee*, 140, 2006.
- [192] David J Marchette and Carey E Priebe. Predicting unobserved links in incompletely observed networks. *Computational Statistics & Data Analysis*, 52(3):1373–1386, 2008.
- [193] Myunghwan Kim and Jure Leskovec. The network completion problem: Inferring missing nodes and edges in networks. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 47–58. SIAM, 2011.
- [194] Björn Bringmann, Michele Berlingerio, Francesco Bonchi, and Arisitdes Gionis. Learning and predicting the evolution of social networks. *IEEE Intelligent Systems*, 25(4):26–35, 2010.
- [195] Albert-Laszlo Barabási, Hawoong Jeong, Zoltan Nédá, Erzsebet Ravasz, Andras Schubert, and Tamas Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical mechanics and its applications*, 311(3-4):590–614, 2002.
- [196] Jianhan Zhu, Jun Hong, and John G Hughes. Using markov models for web site link prediction. In *Proceedings of the thirteenth ACM conference on Hypertext and hypermedia*, pages 169–170. ACM, 2002.
- [197] Linyuan Lü and Tao Zhou. Link prediction in weighted networks: The role of weak ties. *EPL (Europhysics Letters)*, 89(1):18001, 2010.
- [198] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR)*, 49(4):69, 2017.
- [199] R Duncan Luce and Albert D Perry. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116, 1949.
- [200] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.
- [201] Jiye Liang and Zhongzhi Shi. The information entropy, rough entropy and knowledge granulation in rough set theory. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12(01):37–46, 2004.
- [202] Marion E Reid, Christine Lomas-Francis, and Martin L Olsson. *The blood group antigen factsbook*. Academic Press, 2012.
- [203] Stephen M Stigler. Francis galton’s account of the invention of correlation. *Statistical Science*, pages 73–79, 1989.

-
- [204] David N Reshef, Yakir A Reshef, Hilary K Finucane, Sharon R Grossman, Gilean McVean, Peter J Turnbaugh, Eric S Lander, Michael Mitzenmacher, and Pardis C Sabeti. Detecting novel associations in large data sets. *science*, 334(6062):1518–1524, 2011.
- [205] Glad Deschrijver, Chris Cornelis, and Etienne E Kerre. On the representation of intuitionistic fuzzy t-norms and t-conorms. *IEEE transactions on fuzzy systems*, 12(1): 45–61, 2004.
- [206] Kevin Bache and Moshe Lichman. Uci machine learning repository, 2013.
- [207] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.
- [208] Chi-hau Chen. *Handbook of pattern recognition and computer vision*. World Scientific, 2015.
- [209] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: a brief review. *Computational intelligence and neuroscience*, 2018, 2018.
- [210] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564, 2016.
- [211] Ronan G Reilly and Noel Sharkey. *Connectionist approaches to natural language processing*. Routledge, 2016.
- [212] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 2018.
- [213] Pil-Soo Kim, Dong-Gyu Lee, and Seong-Whan Lee. Discriminative context learning with gated recurrent unit for group activity recognition. *Pattern Recognition*, 76: 149–161, 2018.
- [214] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869, 2017.
- [215] Muhammad Ali Masood and MNA Khan. Clustering techniques in bioinformatics. *IJ Modern Education and Computer Science*, 1:38–46, 2015.
- [216] Qiang Liu, Pan Li, Wentao Zhao, Wei Cai, Shui Yu, and Victor CM Leung. A survey on security threats and defensive techniques of machine learning: A data driven view. *IEEE access*, 6:12103–12117, 2018.
- [217] Samet Akçay, Mikolaj E Kundegorski, Michael Devereux, and Toby P Breckon. Transfer learning using convolutional neural networks for object classification within x-ray baggage security imagery. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1057–1061. IEEE, 2016.
- [218] Phil Mars. *Learning algorithms: theory and applications in signal processing, control and communications*. CRC press, 2018.

-
- [219] Han Feng. The application of artificial intelligence in electrical automation control. In *Journal of Physics: Conference Series*, volume 1087, page 062008. IOP Publishing, 2018.
- [220] Said Broumi and Irfan Deli. *Correlation measure for neutrosophic refined sets and its application in medical diagnosis*. Infinite Study, 2015.
- [221] N Ilyasova, A Kupriyanov, R Paringer, and D Kirsh. Particular use of big data in medical diagnostic tasks. *Pattern Recognition and Image Analysis*, 28(1):114–121, 2018.
- [222] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press, 2008.
- [223] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.
- [224] Justus Schwartz, Angelika Steger, and Andreas Weißl. Fast algorithms for weighted bipartite matching. In *International Workshop on Experimental and Efficient Algorithms*, pages 476–487. Springer, 2005.
- [225] Roy Jonker and Anton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987.
- [226] Frank Van Harmelen, Vladimir Lifschitz, and Bruce Porter. *Handbook of knowledge representation*, volume 1. Elsevier, 2008.
- [227] Cheng Ling Tan and Aizzat Mohd Nasurdin. Human resource management practices and organizational innovation: assessing the mediating role of knowledge management effectiveness. *Electronic journal of knowledge management*, 9(2):155, 2011.
- [228] Luis Fernando de Mingo, Fernando Arroyo, Miguel Angel Diaz, and Juan Castellanos. Hierarchical knowledge representation: Symbolic conceptual trees and universal approximation. *International Journal of Intelligent Control and Systems*, 12(2):142–149, 2007.
- [229] Roberto Teixeira Alves, MR Delgado, and Alex Alves Freitas. Knowledge discovery with artificial immune systems for hierarchical multi-label classification of protein functions. In *International Conference on Fuzzy Systems*, pages 1–8. IEEE, 2010.
- [230] Yanran Li, Wenjie Li, and Sujian Li. A hierarchical knowledge representation for expert finding on social media. *methods*, 7(9):10, 2015.
- [231] Gerasimos Spanakis, Georgios Siolas, and Andreas Stafylopatis. Exploiting wikipedia knowledge for conceptual hierarchical clustering of documents. *The Computer Journal*, 55(3):299–312, 2012.
- [232] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, pages 2965–2971, 2016.
- [233] Rowan Garnier and John Taylor. *Discrete mathematics: proofs, structures and applications*. CRC press, third edition, 2009.

-
- [234] Steven S. Skiena. *The Algorithm Design Manual*. Springer, second edition, 2009.
- [235] Tzung-Pei Hong and Chai-Ying Lee. Induction of fuzzy rules and membership functions from training examples. *Fuzzy sets and Systems*, 84(1):33–47, 1996.
- [236] Heng-Da Cheng and Jim-Rong Chen. Automatically determine the membership function based on the maximum entropy principle. *Information Sciences*, 96(3-4): 163–182, 1997.
- [237] Hossein Pazhoumand-Dar, Chiou-Peng Lam, and Martin Masek. Automatic generation of fuzzy membership functions using adaptive mean-shift and robust statistics. In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence*, pages 160–171. SCITEPRESS-Science and Technology Publications, Lda, 2016.
- [238] Tossapon Boongoen, Changjing Shang, Natthakan Iam-On, and Qiang Shen. Fuzzy orders-of-magnitude based link analysis for qualitative alias detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41:1705–1714, 2011.
- [239] Pan Su, Changjing Shang, Tianhua Chen, and Qiang Shen. Exploiting data reliability and fuzzy clustering for journal ranking. *IEEE transactions on fuzzy systems*, 25: 1306–1319, 2017.
- [240] Nele Verbiest, Chris Cornelis, Patricia Victor, and Enrique Herrera-Viedma. Trust and distrust aggregation enhanced with path length incorporation. *Fuzzy Sets and Systems*, 202:61–74, 2012.
- [241] Patricia Victor, Chris Cornelis, Martine De Cock, and Enrique Herrera-Viedma. Practical aggregation operators for gradual trust and distrust. *Fuzzy Sets and Systems*, 184 (1):126–147, 2011.
- [242] H-P Kriegel and Martin Pfeifle. Hierarchical density-based clustering of uncertain data. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 4–pp. IEEE, 2005.
- [243] Mohammad GhasemiGol, Hadi Sadoghi Yazdi, and Reza Monsefi. A new hierarchical clustering algorithm on fuzzy data (fhca). *International Journal of Computer and Electrical Engineering*, 2(1):1793–8163, 2010.
- [244] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.
- [245] Muyu Zhang, Bing Qin, Mao Zheng, Graeme Hirst, and Ting Liu. Encoding distributional semantics into triple-based knowledge ranking for document enrichment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 524–533, 2015.
- [246] Christian Borgelt and Rudolf Kruse. Agglomerative fuzzy clustering. In *International Conference on Soft Methods in Probability and Statistics*, pages 69–77. Springer, 2016.
- [247] Michal Konkol. Fuzzy agglomerative clustering. In *International Conference on Artificial Intelligence and Soft Computing*, pages 207–217. Springer, 2015.