An Adaptive Machine Learning Framework for Access Control Decision Making

---

Thesis submitted in partial fulfilment of the requirements for the degree of

Master of Research Practice

By
MINGSHAN YOU

Institute for Sustainable Industries and Liveable Cities
(ISILC)

Victoria University
Melbourne, Victoria, Australia

February 2022

Abstract

With the increasing popularity of information systems and digital devices, data leakage has become a serious threat on a global scale. Access control is recognised as the first defence to guarantee that only authorised users can access sensitive data and thus prevent data leakage. However, currently widely used attribute-based access control (ABAC) is costly to configure and manage for large-scale information systems. Furthermore, misconfiguration and policy explosion are two significant challenges for ABAC strategies.

In recent years, machine learning technologies have been more applied in access control decision-making to improve the automation and performance of access control decisions. Nevertheless, existing studies usually fail to consider the dynamic class imbalance problem in access control and thus achieve poor performance on minority classes. In addition, the concept drift problem caused by the evolving user and resource attributes, user behaviours, and access environments is also challenging to tackle.

This thesis focuses on leveraging machine learning algorithms to make more accurate and adaptive access control decisions. Specifically, a minority class boosted framework is proposed to address the possible concept drifts caused by evolving users' behaviours and system environments. Its basic idea is to adopt an incremental batch learning strategy to update the classifier continuously. Within this framework, a boosting window (BW) algorithm is specially designed to boost the performance of the minority class since the minority class is fatal for data protection in access control problems. Furthermore, to improve the overall performance of access control, this study adopts a knowledge graph to mine the interlinked relationships between users and resources. A knowledge graph con-

struction algorithm is designed to build a domain-specific knowledge graph. The constructed knowledge graph is also adopted into an online learning framework for access control decision-making.

The proposed frameworks and algorithms are evaluated and verified through two open-source real-world Amazon datasets. Experimental results show that the proposed BW algorithm effectively boosts the performance of the minority class. Furthermore, using topological features extracted from our constructed access control knowledge graph can improve access control performance in both offline and online learning scenarios.

Statement of Authorship

I, Mingshan You, declare that the Master of Research Practice thesis entitled *An Adaptive Machine Learning Framework for Access Control Decision Making* is no more than 50,000 words in length including quotes and exclusive of tables, figures, appendices, bibliography, references and footnotes. This thesis contains no material that has been submitted previously, in whole or in part, for the award of any other academic degree or diploma. Except where otherwise indicated, this thesis is my own work.

I have conducted my research in alignment with the Australian Code for the Responsible Conduct of Research and Victoria University's Higher Degree by Research Policy and Procedures.

Signature  Date 04 Feb 2022

Acknowledgements

Many individuals have given me continuous support and help during this dissertation and my candidature. I appreciate those who made this paper possible.

First and foremost, I am sincerely grateful to my principal supervisor, mentor and friend, Prof. Hua Wang, who provided me with continuous assistance, guidance, support, and encouragement throughout my candidature. Without his advice and help, the study would not have begun and would not have been completed successfully.

My sincere thanks to my co-supervisor, Prof. Yuan Miao and Dr Kate Nana Wang. They provided me with essential suggestions and gave me valuable time, helping me improve the research quality.

I want to acknowledge Prof. Jinli Cao from La Trobe University. She shared her tremendous research experience and gave me encouragement, guidance and invaluable advice to support my work.

I want to express my gratitude to my friends and colleagues in the PAI (Programming of Applied Informatics) group. Thank you for the companionship and numerous discussions during my research candidature.

I would also like to thank my family for their support during this time. Thanks to my wife and daughter for their encouragement and tolerance for my irregular life during the completion of my thesis.

Publications

Some of the ideas, data, results and figures presented in this thesis have been published or are under review over the course of my Master of Research Practice candidature in journals and conference proceedings. I declare that I, Mingshan You, was the leading contributor and primary author for each of these works. The author has the permission of the publishers to reproduce the contents of these publications for academic purposes. See below for an exhaustive list of the aforementioned publications.

Published paper:

[1] **You, Mingshan**, Jiao Yin, Hua Wang, Jinli Cao, and Yuan Miao. "A Minority Class Boosted Framework for Adaptive Access Control Decision-Making." In International Conference on Web Information Systems Engineering, pp. 143-157. Springer, Cham, 2021.

Paper under review:

[2] **You, Mingshan**, Jiao Yin, Hua Wang, Jinli Cao, Kate Wang, Yuan Miao and Elisa Bertino. "A Knowledge Graph Empowered Online Learning Framework for Access Control Decision-Making.", submitted to World Wide Web (2022).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The digital era has brought people not only a wonderful and convenient life but also increasing concerns and anxieties about personal privacy and data security [1, 2]. Groups and organisations that hold valuable data also suffer from data breaches, causing tremendous financial and reputational loss. After surveying 700 IT security professionals from different countries, a report [3] from McAfee in 2019 revealed that most of them have experienced at least one time of data breach during their careers. At the same time, an endless stream of data leakage incidents is constantly stirring the public's nerves, and therefore people are more and more concerned about data security. According to the Australian Community Attitudes to Privacy Survey 2020 [4], most Australians are highly worried about protecting their personal information in their lives. Data security and data breaches (61%) behind identity theft and fraud (76%) are the second most significant privacy risks identified by Australians in 2020 [4]. Therefore, with the rapid development of information software and hardware technology today, data protection technology has always been a research hotspot.

Among various technologies to protect data security, access control systems are typically employed as the first line of defence [5, 6]. They guarantee that only authorised users can gain access to sensitive resources [7]. This thesis aims

to build an adaptive machine learning-based access control model to make access control decisions in an automatic, adaptive, accurate and efficient way. The research results of this work will significantly improve data security in modern information systems, reduce the human working intensity, and save the cost of system security management.

1.2 Research Problems

Upon in-depth analysis of existing machine learning-based access control algorithms, three problems are waiting for better solutions. (1) the possible concept drifts caused by the evolving user and resource attributes, user behaviours and environment in an information system; (2) the dynamic class imbalance problem existing in a real-world access control request steam; (3) high cardinality categorical data in users' or resources' attributes. In order to build an adaptive machine learning-based access control model with high performance, this work aims to fill in the gaps and provide feasible solutions for these three research problems. The detailed literature survey on machine learning-based access control is presented in Chapter 2. Below are further descriptions of the research problems.

1.2.1 Concept Drift Problem

Concept drift refers to the phenomenon that the statistical distribution of the data that the machine learning algorithm tries to describe and mine changes over time in arbitrary ways. Concept drift often exists in data streams, which is a sequence of data, usually organised in chronological order. Given a data stream denoted as $S = \{d_1, d_2, \cdots, d_t, d_{t+1}, \cdots\}$, where $d_i = \{x_i, y_i\}$ is a labelled sample observed at time step i. Let $S_{(0,t)} = \{d_1, d_2, \cdots, d_t\}$ follow a certain relationship denoted as $f_{\Theta(0,t)}(\cdot)$, if $f_{\Theta(0,t)}(\cdot) \neq f_{\Theta(t+1,t+a)}(\cdot)$, concept drift occurs at time step $t+1$, where a is an arbitrary positive number [8].

If concept drift exists, using traditional batch learning methods to train machine learning algorithms will cause a performance decrease as time passes. There are generally two categories of techniques for avoiding concept drift. One is the

so-called lazy strategy, which means a machine learning model's parameter Θ will not be updated until a concept drift is detected. Obviously, the performance of lazy strategies is subject to the accuracy of drift detection. Researchers further proposed active strategies to cope with concept drift to avoid inaction caused by detection failure when concept drift occurs. Machine learning models are updated through concept drift adaption algorithms once new labelled data is available when active learning strategies are adopted. Compared with batching learning methods, active strategies are essentially online learning methods.

Lazy strategies can be divided into two stages, i.e., concept drift detection and concept drift adaptation. Concept drift detection algorithms include data-based methods and error-based methods. Among them, data-based methods use a distance function/metric to quantify the dissimilarity between the distribution of historical data and the new data [9, 10]. In contrast, error-based methods focus on tracking changes in the online error rate of the base model [10, 11, 12]. Concept drift adaption algorithms have four strategies, namely, (1) base learner evolving, such as configuration of decision tree nodes or neural network structures; (2) base learner parametrization, which means updating the parameters of learners by retraining or fine-tuning; (3) adaptive training set formation, which means adjusting the training set via different algorithms, such as sliding window, instance selection and weighting samples [13, 14]; (4) model ensembles, which means combining the outputs of multiple learners through different fusion rules to get a final decision [15, 16]. These concept drift adaption algorithms also can be used in active learning strategies.

Access control requests and responses are a time-series data stream. Concept drifts widely exist in access control applications because user and resource attributes, user behaviours, and access environment change over time [17]. Over an extended period, user access permissions and user behaviour may change. Therefore, the distribution of access control data and the decision-making pattern may also drift over time. For example, a particular user, Bob, had an employee privilege last year, but this year, he had a manager privilege due to a promotion. In this case, the statistical distribution of his access pattern has changed.

However, existing machine learning-based access control studies did not treat access control data in a data stream manner. They still use traditional batch-

ing learning methods to train and evaluate models. This research will design an adaptive machine learning framework, take the concept drift problem into account, adopt active learning strategy and drift adaption algorithms to handle the concept drift in access control problems.

1.2.2 Dynamic Data Imbalance Problem

Data imbalance is another problem that may cause severe performance decrease apart from concept drift problems for machine learning-based access control. From the perspective of machine learning, access control decision-making is a binary classification problem whose purpose is to establish a machine learning model to predict giving permission or not when a user requests a resource.

For a classification problem, if the number of samples belonging to each class is not equivalent, the dataset is imbalanced or skewed [18]. The class distribution can vary from a slight bias status to a very severe imbalance status. In some extreme cases, the ratio of the minority class to the majority class can be 1:100, 1:1000 or even worse.

Generally speaking, an appropriate machine learning algorithm and a large amount of balanced labelled data can ensure the performance of a predictive classification model. However, the historical data of a real-world access control system is severely imbalanced because most resource requests are permitted, but only very few proportions (less than 5%) are denied[19]. Since machine learning models are trained based on the cumulative loss of all samples, the models tend to pay more attention to the majority class. Therefore, when a dataset is imbalanced or skewed, the performance of the majority class will be exaggerated while the performance of the minority will be unacceptably poor.

For access control problems, the minority class (the permission denied cases) is more crucial than the majority class because incorrectly permitting an illegal request may cause critical data breaches and substantial economic losses. Thus, it is essential to apply practical algorithms to boost the performance of minority classes.

Existing data imbalance coping algorithms, including majority class down-sampling, minority class up-sampling, setting class weights or sample weights,

are trying to keep the samples in two classes or the loss from two classes balanced [20, 21]. These algorithms work well in offline learning scenarios, where the degree of data imbalance is definite and static. For example, it is easy to decide the sampling proportion or the class weight if we know the ratio of two classes is 2:5.

Considering the online learning scenarios, the degree of data imbalance (the proportion of permission deny cases) is dynamically changing over time. Therefore, how to deal with the data imbalance problem is still an open question in this area. This research tries to design an adaptive class imbalance coping algorithm and thus improve the machine learning model's performance in the minority class.

1.2.3 High-cardinality Categorical Data

Access control data is highly confidential to organisations because it relates to their core data and information security. Therefore, the available user and resource attributes are very limited in these open-source access control datasets compared to the real-world scenarios. Furthermore, the available attributes are also encrypted or desensitised before release. General practice for attribute desensitisation is replacing the values with unique integers. For example, in an Amazon employee access dataset [1], users' business titles are represented by title ID, and there are 4,979 unique ID numbers in this dataset. Similarly, users' department names are also represented by 405 unique ID numbers. In this case, no useful semantic meanings can be extracted from these desensitised attributes. Only high cardinality categorical attributes can be used to extract features to represent the corresponding users or resources, which is very challenging.

Generally speaking, there are two kinds of categorical data, i.e., nominal and ordinal. Nominal data is a group of values without order, such as the aforementioned department names and business titles. By contrast, ordinal data is a group of values with an order, such as users' length of service in years.

The general practice for nominal categorical data encoding is the well-known one-hot encoding. However, it only works for low cardinality features. Otherwise, the encoded feature will be too high dimensional to perform well in machine

[1] http://archive.ics.uci.edu/ml/datasets/Amazon+Access+Samples

learning, also known as the 'curse of dimensionality'. Although the label encoding method, which maps the values in the category group into integers, can solve the high dimension problem, the drawback is that it will introduce artificial ordinal relationships between different values. For example, the department encoded in 1 is not necessarily 'smaller' than a department encoded in 10,000. Binary encoding, which encodes the integer results of label encoding into binary digits and then spits the binary digits into multiple columns by bit, is a compromise between one-hot encoding and label encoding.

However, the aforementioned encoding methods are still unsatisfactory for the access control problem. None of them can reflect the interrelationships between different users or attributes who share the same attributes. This thesis will try to adopt the advanced data structure knowledge graph to store and represent high cardinality categorical data. Further, extract topological features from the access control knowledge graph to represent the interrelated relationships between users and resources.

1.3 Contributions

To partly solve the aforementioned problems, this thesis leverages advanced artificial intelligence techniques, including but not limited to machine learning and knowledge graphs, to provide feasible solutions for effective and accurate access control decision-making. The contributions of this thesis are summarised below.

- This thesis proposes an adaptive machine learning framework for access control problems, which adopts consecutive incremental batch learning to adjust the parameters of a machine learning classifier. The framework can capture and adapt to possible concept drifts and real-time changes in access control patterns. As far as we know, this is the first work discussing the machine learning-based access control problem from a data stream perspective.

- This thesis designs a boosting window (BW) algorithm within each consecutive batch to tackle the severe data imbalance problem. The BW algorithm

sets a fixed-size window to hold the samples used to update the classifier parameters. The BW only selects the misclassified samples and controls the class ratio within the window with a preset rate between 0 to 1. The BW algorithm is demonstrated effective in boosting the performance of the minority class (access deny) in access control problems.

- To improve the model's overall performance, this thesis further proposes a knowledge graph empowered online learning framework for access control decision-making. First, an algorithm is designed to construct a knowledge graph from the existing user and resource attributes. Further, this thesis demonstrates how to extract topological features from the established knowledge graph to represent users and resources. To the best of our knowledge, this study is the first try to leverage knowledge graph to extract graph topological features to improve the performance of the access control model.

- This thesis evaluates the proposed frameworks and algorithms and demonstrates their effectiveness and flexibility on two real-world Amazon employee access datasets. The influence of the hyper-parameters of BW on the performance of the minority class is also discussed. Furthermore, the effectiveness of knowledge graph-based topological features is discussed on different imbalance degrees in both online and offline scenarios.

1.4 Thesis Outline

The overall flowchart of the thesis is illustrated in Fig. 1.1. This thesis consists of five chapters in total, and the remaining four chapters are organised as follows.

Chapter 2 introduces the background knowledge of access control, along with the literature survey and the latest research progress. This chapter also presents some fundamental concepts and algorithms of machine learning, which will be used in chapters 3 and 4. For example, chapter 2 formulates classification problems and introduces a gradient descent learning strategy for model parameter optimisation. Other model parameter learning strategies, such as regularisation, offline learning and online learning, are also covered. Furthermore, five

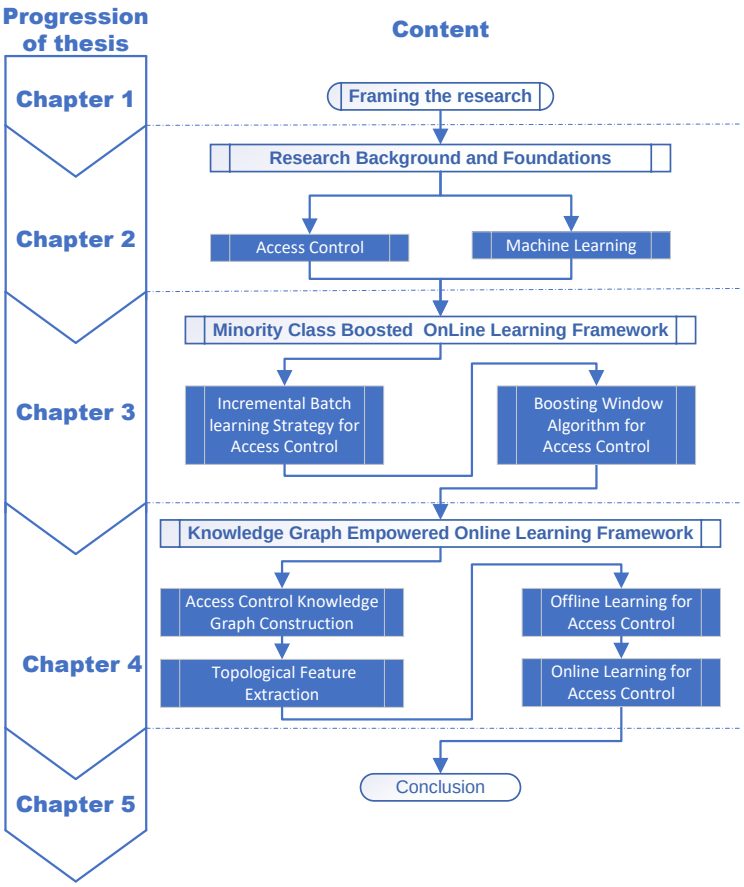

Fig. 1.1. Thesis overall flowchart

well-known classic machine learning algorithms are introduced: Gaussian Naive Bayes, Logistic Regression, Neural Networks, Support Vector Machine, and Random Forest. Finally, Chapter 2 presents some data prepossessing techniques and evaluation metrics for classification problems.

Chapter 3 presents a minority class boosted framework for adaptive access control decision-making. Specifically, this framework employs a continuous incremental batch learning strategy to adapt the concept drift problem instead of a batch learning approach. Furthermore, a boosting window algorithm within the framework is specially designed to boost the performance of the minority class,

thus, decreasing false positive decisions. The proposed framework is evaluated on a well-known Amazon employee access dataset, and results demonstrate the effectiveness and flexibility of the proposed framework and BW algorithm.

Chapter 4 proposes an algorithm to construct an access control knowledge graph from user and resource attributes. Furthermore, an online learning framework for access control decision-making is proposed based on the constructed knowledge graph. Within the framework, topological features are extracted to represent high-cardinality categorical attributes of users and resources. Experimental results show that topological features extracted from the knowledge graph can improve access control performance in both offline and online learning scenarios.

Chapter 5 concludes the findings and contributions of this thesis and discusses possible challenges and potential for future work.

Chapter 2

Background and Foundations

This chapter firstly reviews literature related to data security and access control. Then presents the fundamental concepts and core techniques on machine learning-based access control, including but not limited to: classification problem formulation, model parameter optimisation, classic classification algorithms, data preprocessing techniques and model evaluation metrics.

2.1 Access Control

Data exfiltration refers to carrying out an unauthorised data transfer from an information system by malware, or a malicious actor [22]. It occurs in various ways, including database leakages, network traffic, file shares, corporate emails, etc [23]. Since 2000, a large number of data exfiltration incidents have severely damaged the privacy, security, confidentiality and intellectual property of individuals, businesses and governments across the world. Moreover, the situation will become more complicated when threats come from insiders, such as current employees, former employees, contractors, business associates, etc. Although multiple techniques have been employed to prevent information leakage [22], the situation is still not optimistic. According to a report [3] from McAfee, more than half of security professionals claimed that they experienced a data breach.

Access control systems are typically employed as the first line of defence for the protection of data security [24, 25]. The two main components of access control systems are authentication, which is used to verify users' identity, and authorisation, which grants access requests. Access control systems guarantee that only authorised users can access sensitive resources. There are three mainstream access control strategies: role-based access control (RBAC), attribute-based access control (ABAC) and machine learning-based access control. The following subsections will introduce the advantages and disadvantages of these three strategies in combination with Fig. 2.1.

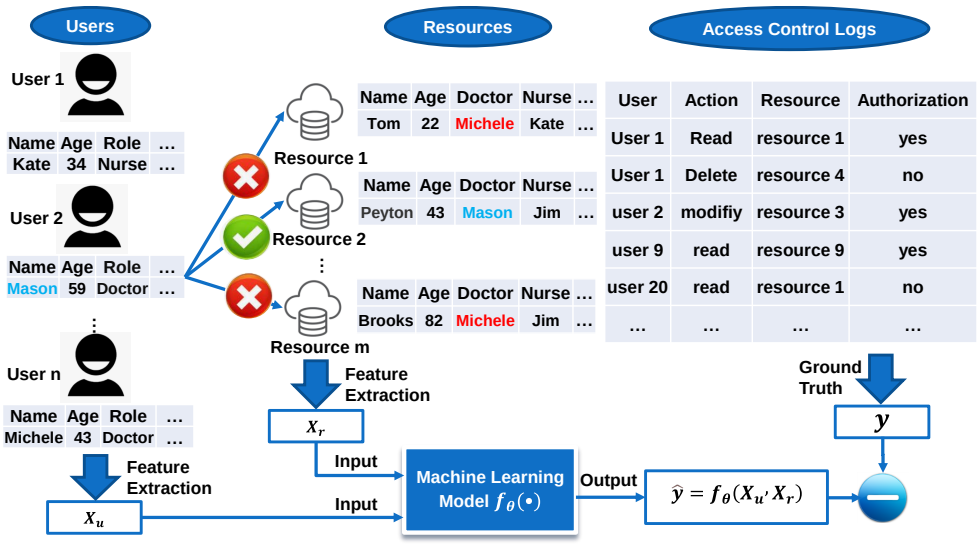

Fig. 2.1. Access control strategy demonstration

2.1.1 Role-based Access Control

Role-based access control (RBAC) strategy is a basic but efficient way to prevent insiders from disclosing sensitive and private information [26]. Within an organisation, roles are assigned to its staff or members based on their job functions. When a user requests a resource, permission is granted only based on the user's role.

As shown in Fig. 2.1, taking a hospital information system as an example, User 1 has been assigned a Nurse role, and User 2 has been assigned a Doctor

role. The system can allow users who have a Doctor role to access all patients' medical records. In this case, all Doctors have access to whole patients' health records, just because they have the same role — Doctor. However, there may be millions of patients in a system, but each Doctor may be only responsible for hundreds of patients. Obviously, the role-based access control strategy is unnecessary enlarged Doctor's permissions to make every Doctor have access to all patients' records.

To summarise, the role-based access control strategy is simple to implement and efficient to operate. Therefore, it has been widely used in early information systems. However, as the scale of information systems increases, the coarse-grained RBAC strategy has the risk of amplifying users permissions [27]. Therefore, more fine-grained access control policies are demanded.

2.1.2 Attribute-based Access Control

Attribute-based access control, also known as policy-based access control, is becoming more and more popular in modern information systems because of its flexibility and expressiveness. It is a technique that allows the specification of fine-grained and context-aware access control policies.

Access policies are Boolean logic statements generated from the attributes of users, resources and other related objects. Here is an example of Boolean logic statement of an attribute-based policy, 'user.age >= 18 OR resource.owner == user.id' or 'TIME > 8:00 AM AND TIME < 5:00 PM'.

Section 2.1.1 discussed that RBAC strategies can give users enlarged permissions, taking a hospital information system as an example, as shown in Fig. 2.1. ABAC strategies can solve the problem by specifying the attributes of users and resources in a Boolean statement. For example, a policy, 'if User.role == Doctor and User.Name == Resource.Doctor, then give the user permission to access the Resource', can narrow Doctors' permission to the medical records of patients under their name only. In this case, as shown in Fig. 2.1, User 2 have access to Resource 2 but have no access to Resource 1 and m because their Doctor's name is Michele instead of Mason.

Since ABAC can limit access to specific resources according to delicately designed Boolean statements, it can adapt to new changes in modern information systems over time, such as the increasing diversity of roles, variety of data access environments and diversity of access devices. So far, many attribute-Based algorithms are proposed to generate Access Control Policies [28, 29**?**]. ABAC has been widely adopted by modern information systems.

However, the ever-changing information technology, evolving lifestyles and habits, the emergence of new technologies such as the Internet of Things (IoT) and distributed information systems have brought new challenges [30, 31]. One is the ever-increasing scale of policies, also known as policy explosion. The considerable policy scale makes it challenging to generate and manage policies manually. The huge policy scale not only reduces the efficiency of the system but also leads to frequent misconfiguration of policies and many other issues. The 2020 Verizon Data Breach Investigations Report shows that among nearly 4,000 investigated data breaches, incidents caused by misconfiguration have risen to fourth place in 2020. Especially from 2019 to 2020, the proportion of incidents caused by misconfiguration has increased substantially by about 5%. Most of the existing ABAC algorithms are incapable of dealing with large scale attributes or dynamic changing attributes.

2.1.3 Machine Learning-Based Access Control

Machine learning is a branch of artificial intelligence (AI), containing multiple data analytical algorithms, such as linear regression, logistic regression, neural networks, support vector machine, decision tree, etc. Machine learning algorithms can learn latent data distribution and patterns and build mathematical decision-making models by automatically learning from data [32, 33]. In the last decades, machine learning algorithms have achieved great progress in some traditional areas like information retrieval, machine translation, automatic speech recognition and computer vision [34, 35, 36, 37, 38]. Recently, researchers have begun to explore adopting advanced machine learning algorithms to build high-performance access control classifiers to overcome the problems of attribute-based access control strategies.

For an access control problem, the system log files record a large amount of historical resource access requests and system responses data, which is a perfect resource of labelled data for machine learning model training [39]. The basic idea of machine learning-based access control is to transform the access control task into a binary classification problem [40]. Machine learning algorithms can work as a classifier to identify patterns behind the historical permission operations by training mathematical decision-making models from data recorded in log files.

Specifically, as demonstrated in Fig. 2.1, samples' features can be extracted from user attributes and resource attributes, borrowing the idea from attribute-based access control. The extracted user features, denoted as X_u and resource features, represented as x_r, work as the inputs of a machine learning model, denoted as $f_\theta(\cdot)$. The samples' ground truth, marked as y, working as the output of a machine learning algorithm, can be extracted from the log files of an access control system. The extracted sample features and corresponding ground truth finally form a labelled dataset, which can be used to train the machine learning model. Once a model is well-trained, it can be used to make decisions for new access control requests. For example, when User 1 sends a new access request to Resource 2, the trained model can predict whether to permit or deny this access.

Researchers have reported promising work applying machine learning algorithms in access control decision-making. For example, Jabal et al. [41] used random forest (RF) and k-nearest neighbours (KNN) algorithms to improve their access control model's performance significantly. Outchakoucht et al. [42] proposed a machine learning-based access control framework for access control in an IoT system. Liu et al. [40] proposed a permission decision algorithm based on random forest, which can achieve a permission decision accuracy of around 93% on a test dataset.

Although some of the works mentioned above have successfully mined decision patterns from log data using machine learning techniques, some problems still exist due to the unique application characteristics of access control: They did not treat this issue from the time series perspective and failed to consider the concept drift problems caused by the evolving user and resource attributes, user behaviours and environment [43, 44]. The studies mentioned above usually failed to consider the dynamic class imbalance problem. When reporting performance,

they focused on the performance of the majority classes and ignored the minority classes.

Simple algorithm transplants without proper tailoring or improvement may lead to severe performance degradation if these problems are not taken seriously. Therefore, this research designs algorithms to deal with the abovementioned problems and proposes an adaptive machine learning framework for access control decision-making.

2.2 Classification Problem

A typical machine learning algorithm can be formulated as Equation (2.1), where $x \in R^n$ is a n-dimensional input feature vector extracted form raw data; $f_\Theta(\cdot)$ is a mathematical mapping function from input to output, which is decided by the specific machine learning algorithm; $f_\Theta(\cdot)$ is also known as the hypothesis of the research problem; $\Theta = [\theta_1, \theta_2, \cdots, \theta_u]$ is the list of trainable parameters of function $f_\Theta(\cdot)$ and u is the total number of parameters in $Theta$; $\hat{y}$ is the predictive output of the machine learning model.

$$\hat{y}^{(t)} = f_\Theta(x) \tag{2.1}$$

To describe the learning (training) process of a supervised machine learning model, denote a labelled dataset as $D = \{X, Y\}$, where $X = [x_1, x_2, \cdots, x_m]$ is the input matrix containing m input vectors and $X \in R^{m \times n}$; $Y = [y_1, y_2, \cdots, y_m]$ is the corresponding output vectors, and $Y \in R^{m \times 1}$; y_i $(i = 1, 2, \cdots, m)$ is also called the ground truth label of input $x_i (i = 1, 2, \cdots, m)$; $\{x_i, y_i\}$ is the i-th labelled sample. If y_i is a real number, the machine learning model is a regression model and the predictive problem is a regression problem. For example, the housing price prediction problem is a regression problem. By contrast, if y_i is a discrete value, $f_\Theta(x)$ is a classification model and the corresponding predictive task is a classification problem. Specifically, When a label y_i is a member of a finite set of classes and the size of the set of classes is two, it is a binary classification problem. For example, for a spam email recognition system, $y_i \in$ {spam, non-spam} and the model used for spam email recognition is also called a

spam classifier. When the size of the class set is larger than two, it is a multiclass classification problem. For example, in a traffic light recognition system, $y_i \in$ {red, green, yellow}.

2.2.1 Cost Function

To train a classifier $f_\Theta(\cdot)$ is to find out the optimised value of its parameters $\Theta = [\theta_1, \theta_2, \cdots, \theta_u]$. To learn the parameters from a given a labelled dataset as $D = \{X, Y\}$, a cost function used to measure the differences between the output of the model $\hat{y}$ and the ground truth y needs to be defined. Equation 2.2 is the widely used cross-entropy cost function [45].

$$L_{(X,Y)}(\Theta) = -\frac{1}{m}\sum_{j=1}^{m}[y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] \tag{2.2}$$

Then, the classifier training problem turns into an optimisation problem, described in (2.3). Algorithms like gradient descent can be used to solve this optimisation problem.

$$\text{optimisation goal:} \quad \min_{\Theta} L_{(X,Y)}(\Theta) \tag{2.3}$$

2.2.2 Gradient Decent

Gradient descent is a general, maybe most widely used, algorithm in machine learning for minimising the cost function $L_{(X,Y)}(\Theta)$ and finding out the optimal model parameters Θ.

The basic process of gradient descent is described as follows: (1) randomly initialise the parameters in Θ with a combination of parameters; (2) compute the cost on the dataset $\{X, Y\}$ with the initialised parameters; (3) simultaneous update the parameters in Θ with Equation (2.4) to ensure the value of cost function $L_{(X,Y)}(\Theta)$ decreases; (4) Keep doing step (3) until reach convergence. Because it is impossible to traverse all parameter combinations, it is uncertain that the local minimum is the global minimum. Therefore, when choosing a

different initial parameter combination, a different local minimum may be found. This is actually an inherent limitation of the gradient descent algorithm.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} L_{(X,Y)}(\Theta), (j = 1, 2, \cdots, u) \tag{2.4}$$

In Equation (2.4), $\frac{\partial}{\partial \theta_j} L_{(X,Y)}(\Theta)$ is the gradient of θ_j and α is the learning rate, which determines step size that walking along the direction of gradient. It is carried out that the direction of the gradient can ensure the fastest descent, and the local minimum will be finally obtained. It is worth mentioning that gradient descent can actually be used to minimise other forms of cost functions in machine learning.

2.3 Model Learning Strategies

2.3.1 Regularisation

When learning parameter Θ from training set, it may turn out to be over-fitting on the training set and therefore have a very poor generalisation performance on the test set. Regularisation is one of the most widely used approaches to prevent over-fitting. In practice, regularisation often leads to slightly higher bias but significantly reduces the variance. This problem is also known in literature as the bias-variance trade-off. The two most widely used types of regularisation are L1 and L2 regularisation, as shown in Equation (2.5) and (2.6), respectively.

$$L_{(X,Y)}(\Theta) = -\frac{1}{m} \sum_{j=1}^{m} [y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] + \lambda \sum_{j=1}^{u} |\theta_j| \tag{2.5}$$

$$L_{(X,Y)}(\Theta) = -\frac{1}{m} \sum_{j=1}^{m} [y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] + \frac{\lambda}{2m} \sum_{j=1}^{u} \theta_j^2 \tag{2.6}$$

The basic idea of regularisation is to add a penalising term to the cost function. L1 regularisation penalises the sum of absolute values of the parameters in Θ, whereas L2 regularisation penalises the sum of squares of the parameters. When

the model is more complex, the value of the penalising term is higher. Therefore, the regularisation term will force the learning algorithm to build a less complex model.

2.3.2 Offline Machine Learning

Offline learning, also known as batch learning, is the standard learning strategy for machine learning. Basically, this method sources a dataset and builds a model on the whole dataset at once. Once the model needs to be partially changed, the only way is retraining, which is time-consuming. In addition, this method stores all data on the server or terminal, which requires high memory.

2.3.3 Online Machine Learning

In contrast to offline learning, online learning is a machine learning method in which data is available sequentially and used to update the best predictor for future data at each step [46]. It is a common technique used in areas of machine learning where it is computationally infeasible to train over the entire dataset. It is also used in situations where it is necessary for the algorithm to dynamically adapt to new patterns in the data, or when the data itself is generated as a function of time, e.g., stock price prediction.

Online learning is data-efficient because once data has been consumed it is no longer required. Technically, this means that storing data is not necessary. This method is also adaptable as it makes no assumption about the distribution of your data. Due to changes or drifts in data distribution, such as changes in customer behaviour, the model can adjust in real-time to keep up with real-time trends.

2.4 Classic Classification Algorithms

Some of the classification algorithms involved in the experiments in this thesis, such as Gaussian Naive Bayes, Logistic Regression, Neural Networks, etc., are briefly discussed below.

2.4.1 Gaussian Naive Bayes

One of the simplest yet most effective algorithms for solving classification problems is the Naive Bayes algorithm. It is a probabilistic method based on Bayes' theorem with the assumption of naive independence between input attributes. Gaussian Naive Bayes is a variant of the Naive Bayes algorithm. It follows Gaussian normal distribution and supports continuous data.

2.4.2 Logistic Regression

Logistic regression is a method for binary classification and it is based on the logistic function (also called sigmoid). The two characteristic features of that function make it particularly convenient for modelling probabilities. These are: 1) it is monotonically increasing 2) its range is between 0-1. As stated before, logistic regression is a probabilistic function, which means that the conditional probability of a data point belonging to a class of interest using the sigmoid function must be fit. The probability of assignment to the opposite class is simply its complement. There are many ways for fitting the best coefficients. In the logistic regression model, the coefficient vector that maximises the joint likelihood of the input data points in the training set having their corresponding label is favoured. As an optimisation technique, gradient descent is most frequently used. What makes logistic regression classifier convenient in text-related tasks is the inspection of its coefficients generated from the training set. Given the high level of ambiguity present in all natural language processing tasks (short messages such as tweets in particular), the insight into the classification criteria allows for further algorithm refinement to better fit its purpose. This feature is especially advantageous when the goal is the extraction of relevant data on a particular topic given user-defined criteria (e.g. posts using specified key-words). In that case, both features determination as well as classifier selection and tuning contribute towards the overall system's sensitivity.

2.4.3 Neural Networks

A neural network is a series of algorithms that endeavour to recognise underlying relationships in a set of data through a process that mimics the way the human brain operates. It contains layers of interconnected nodes. Each node is known as a perceptron and is similar to multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. A neural network has three main components: an input layer, a processing layer, and an output layer. The inputs may be weighted based on various criteria. Within the processing layer, which is hidden from view, there are nodes and connections between these nodes, meant to be analogous to the neurons and synapses in an animal brain. Neural networks are broadly used, with applications for financial operations, enterprise planning, trading, business analytics, and product maintenance. Neural networks have also gained widespread adoption in business applications such as forecasting and marketing research solutions, fraud detection, and risk assessment.

2.4.4 Support Vector Machines

Support vector machines are commonly recognised for their high predictive accuracy [47]. The support vector machines classification method is based on the Structural Risk Minimisation principle from computational learning theory. In contrast to other classification methods, support vector machines need both positive and negative training sets, which are uncommon for other classification methods. These sets are required for the support vector machines to find the decision surface that best separates positive from negative instances of data through a linear hyperplane, which maximises the margin. The document representatives, which are closest to the decision surface, are called the Support. The performance of the support vector machines classifier remains unchanged if documents that do not belong to the support vectors are removed from the training set.

2.4.5 Random Forests

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. Random decision forests correct for decision trees habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance. Random forests are frequently used as 'blackbox' models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

2.5 Data Preprocessing Techniques

2.5.1 One-hot Encoding

One-hot encoding can solve the problem of unequal class weights given to categories within a feature. Its basic strategy is to convert each category value into a new column and assign a 1 or 0 (True/False) value to that column. This has the advantage of not inappropriately weighing a value and thus potentially improving the performance of the classifier. However, it is not very practical when there are many categories. Because encoding like this will result in the formation of as many new columns, this can lead to the "curse of dimensionality" and thus make the model not work properly.

2.5.2 Binary Encoding

Binary encoding is a type of code used primarily to program computers at the most basic level. It consists of a system of ones and zeros, designed to represent either a "true" or a "false" value in logical operations. This technique first encodes the categories as ordinal numbers, then converts these integers to binary codes, and finally splits the numbers in the binary string into separate columns. It is not

as intuitive as one-hot encoding. Its advantage is that it encodes data in fewer dimensions than one-hot encoding.

2.5.3 Outlier Remove

Outliers are values in the data that are significantly different from the main sample of the data. The presence of outliers can significantly degrade the performance of a classification model. Generally, outlier processing includes the following three methods: Flooring and Capping, Trimming and Replacing.

- Flooring and Capping is a quantile-based technique that partially discards some data at the experimenter's request. For example, implementing the flooring (e.g. 25th percentile) for the lower values and capping(e.g. for the 75th percentile) for the higher values means that values outside the range of 25% to 75% will be removed.

- Trimming removes and completely drops all the outliers. It excludes outlier values from the analysis. By applying this technique, the data becomes thin when more outliers are present in the dataset.

- Replacing refers to replacing outliers with a specific value, for example, the mean, median, mode (the value that appears most frequently in a series of numbers), or other values.

2.5.4 Normalisation

Keeping these features on a similar scale is necessary when solving multi-dimensional feature problems because it will help the gradient descent algorithm converge faster. Mean normalisation is a way to implement feature scaling, the process of bringing all the features of a machine learning problem into a similar scale or range. There are usually two methods of normalisation: Min-Max Normalisation and Z-score Normalisation.

- Min-max Normalisation, also known as deviation normalisation, is a linear transformation of the original data such that the resulting values are mapped between [0,1]. The conversion function is as follows:

$$x^* = \frac{x - \mu}{max - min}, \tag{2.7}$$

where x and x^* is the original and transformed sample value, respectively. max and min are the maximum and minimum value of sample data, respectively. μ is the mean of samples.

- Z-score Normalisation standardises the data based on the mean and standard deviation of the original data. The processed data conforms to the standard normal distribution, that is, the mean value is 0, the standard deviation is 1. The transformation function is as follows:

$$x^* = \frac{x - \mu}{\sigma}, \tag{2.8}$$

where σ is the standard deviation of the sample.

2.6 Evaluation Metrics

This section will briefly introduce the evaluation indexes involved in this study.

2.6.1 Confusion Matrix

A confusion matrix visualises the performance of an algorithm. It is an essential tool for evaluating classification models. It helps to comprehend the classification model's performance on a set of test data to understand the valid values and false by determining how many times the model has given correct and wrong output.

As shown in table 2.1, in a binary classification problem, there are four possible outcome types for each class.

- TP: True Positive means that predicted values are correctly predicted as actual positive.

Table 2.1: Confusion matrix

Total population		Predicted condition	
		Positive (PP)	Negative (NN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

- FP: False Positive means that predicted values are incorrectly predicted as an actual positive. i.e., negative values predicted as positive.
- FN: False Negative means that positive values are predicted as negative.
- TN: True Negative means that predicted values are correctly predicted as an actual negative.

2.6.2 Accuracy

Accuracy, or Acc for short, is the percentage of true results from the total number of cases reviewed as shown in 2.9. It is an effective evaluation option for classification problems that are balanced, unbiased, or do not have a class imbalance. But it does not work well when the target class of the model is very sparse.

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.9}$$

2.6.3 Precision

As shown in 2.10, Precision (or Pre for short) answers the question of what proportion of the predicted positives are true positives. Precision is a valid choice of evaluation metric when we want to be very sure of our prediction.

$$Pre = \frac{TP}{TP + FP} \tag{2.10}$$

2.6.4 Recall

As shown in Equation 2.11, Recall (or Rec for short) represents the percentage of actual positives in the sample whose prediction is positive. Recall is a valid choice

of evaluation metric when we want to capture as many positives as possible.

$$Rec = \frac{TP}{TP + FN} \tag{2.11}$$

2.6.5 F1 Score

As shown in Equation 2.12, the F1 Score (or F1 for short) is a comprehensive indicator which is the harmonic mean of precision and recall. Simply stated the F1 score sort of maintains a balance between the precision and recall for your classifier.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.12}$$

Chapter 3

A Minority Class Boosted Online Learning Framework

This chapter proposes a minority class boosted framework for adaptive access control methods in response to the concept drift and dynamic data imbalance problems. The main content of this chapter is organised as follows. Section 3.1 briefly introduces the background and contributions of this chapter, followed by related works on learning strategy and multiclass imbalanced learning in Section 3.2. Then a detailed description of the proposed framework and boosting window algorithm is presented in Section 3.3. The datasets, evaluation metrics and experiment results are presented in Section 3.4. Finally, Section 3.5 concludes this chapter with a discussion on limitations.

3.1 Introduction

Data exfiltration has in recent years been becoming a top concern across most security-conscious communities, such as governments, armies and other organisations with high-value data [25]. It is also known as data breach or data theft, and many other names. But in essence, it refers to carrying out an unauthorised data transfer from an information system by malware or a malicious actor. According to a report [3] from McAfee in 2019, they found that most IT professionals

have experienced at least one data breach during their career after surveying 700 IT security professionals from different industry and country. On average, they have dealt with six breaches over the course of their professional lives. Data breaches come in different forms, but many of them are related to insider threats. According to the 2020 Verizon Data Breach Investigations Report [48], 30% of data breaches involve internal actors. So, how to prevent data exfiltration from insiders has now become a research hotspot.

Access control is a technology that is typically employed as the first line of defence for the protection of data[24, 25, 49, 50], guaranteeing that only authorised users can gain access to sensitive resources. The latest access control strategy, attribute-based access control (ABAC), grants or denies an operation request based on assigned attributes of the subject and object, environmental conditions and a set of policies specified in terms of those attributes and conditions[25, 51, 52]. As ABAC can create different policies in accordance with administrative requirements, it is more flexible than traditional role-based access control systems [53, 54]. However, the size of the policy increases dramatically as the increasing diversity of roles, data access environments (e.g., homes and cars) and access devices (e.g., smartphones and tablets). The huge policy scale not only reduces the efficiency of the system, but also leads to frequent misconfiguration of policies [55, 56]. Therefore, more and more researchers are beginning to employ machine learning (ML) methods to develop access control strategies.

From the perspective of machine learning, access control decision-making is a binary classification problem. For an access control system, log files record a large number of historical data of resource requests and system responses, which is a perfect labelled data resource for machine learning algorithms to learn potential data distribution and decision patterns [57, 58]. However, due to the unique features of access control, simple algorithm transplants without proper tailoring or improvement may lead to severe performance degradation.

Specifically, there are two main challenges when building a high-performance machine learning based access control decision-making model. The first challenge is the concept drift problem [8]. Access control patterns are changing over time due to the evolving user and resource attributes, user behaviours and environments. Thus, traditional batch learning strategies are not suitable for access

control problems. One possible solution is to adjust the learning strategy online to capture new decision-making patterns, and modify the parameters of the model based on the latest label data over time. Data imbalance is another problem which may cause severe performance decrease besides concept drift for ML algorithms. For access control tasks, there are two classes, access approved (class 1) and access denied (class 0). In most cases, resource access applications are routine working requests and should be approved. Only a very few applications are due to misoperation or illegal access initiated by malicious users. Therefore, the datasets collected from log files are usually imbalanced and lack negative samples. To make matters worse, in terms of security protection systems, negative samples are often more important than the positive sample. It needs more effort to boost the ML models' performance on negative samples (minority class).

To tackle the above-mentioned problems, this chapter proposes a minority-based adaptive access control decision-making framework. In brief, our main contributions are as follows:

(1) We propose an online machine learning framework for access control problems, which adopts consecutive incremental batch learning to adjust the parameters of the ML classifier. The framework is capable of capturing and adapting to possible concept drift and real-time changes in system pattern.

(2) We design a boosting window (BW) algorithm within each consecutive batch to tackle the severe data imbalance problem. BW algorithm sets a fixed-size window to hold the samples used to update the classifier parameters. The boosting window selects only the misclassified samples and controls the class ratio within the window with a preset sample rate of 0 to 1. The designed BW algorithm can effectively boost the performance of the minority class, the more important class in access control problems.

(3) We evaluate the proposed framework and BW algorithm on a real-world Amazon employee access dataset and the results demonstrate their effectiveness and flexibility. We also discuss the influence of the hyper-parameters of BW algorithm on the performance of the minority class.

3.2 Related Work

This section reviews the related works from the perspective of access control and machine learning. Past and recent research progress and gaps between existing research and real-world applications will be presented.

Role-based access control (RBAC) and attribute based access control (ABAC) are two widely used access control methods. The former assigns permissions of systems, resources and networks based on a user's role within an organisation[59]. Due to its simple implementation, RBAC has long been one of the main access control methods. E. Bertino et al. proposed a temporary RBAC model, employing a role triggers strategy to deal with periodic role activations or other temporary roles[60]. To better suit the complicated system environment and provide more fine-grained access control policies, M. J. Moyer designed a generalised ABAC paradigm to create and maintain rich access control policies, which incorporates traditional user roles with subject roles, object roles and environment roles into access control decisions [61]. However, with the development of information system, the permissions of roles are gradually refined and RBAC tends to give users unnecessarily enlarged permissions. Besides, RBAC allows multiple users to share the same permissions, with no restrictions other than roles, which may lead to malicious use of this vulnerability for unauthorised access [62, 63].

In recent years, attribute-based access control has gradually become a mainstream access control strategy because it takes into account additional attributes from both users and requested resources[64]. ABAC is a policy-based method which generate fine-grained context-aware access control policies rather than role-based static permissions [25]. A series of general or domain-specific policy-based ABAC models have been successfully used in cloud computing, real-time systems, collaborative environments, mobile environments, grid computing, web services etc [65, 66, 67, 68].

However, the policy scale has become larger and more complex due to the increase in attributes and the pursuit of system flexibility and versatility. Policy-based ABAC models also caused a number of issues, such as policy misconfiguration and slow response[25, 69]. Therefore, some researchers have started to

explore hybrid ABAC models based on policy generating rules and ML algorithms. For example, S. Dutta et al. proposed a privacy via anomaly-detection system (PALS) to leverage machine learning algorithms to capture physical context collected from attributes and generate context-driven policies[70].

Besides, pure ML approaches have attracted more and more attention in various applications and domains [71, 72, 73, 74]. The main problems for pure ML access control decision models are concept drift and data imbalance. Concept drift is a phenomenon that the statistical distributions of what an ML algorithm tries to describe and predict change over time in an arbitrary way, which often exists in data streams or sequences of data organised in chronological order [8]. Data imbalance refers to that the number of samples belonging to each class in a labelled dataset is not at an equivalent level[75]. This chapter is a practice of trying to deal with both concept drift and data imbalance problems in pure ABAC ML models.

3.3 Methodology

This section first presents the workflow of the proposed consecutive incremental batch learning framework, aiming to tackle the possible concept drift for access control decision-making. Then, we illustrate the principle and implementation details of the boosting window algorithm, which is designed to improve the performance of the minority class, thereby reducing false positives.

3.3.1 Workflow of the Proposed Framework

The proposed framework is essentially a classifier-agnostic consecutive incremental batch learning process for access control applications. Data is divided into a sequence of consecutive batches and a ML classifier working as an ABAC model in this framework would be pretrained and updated according to these data batches. Figure 3.1 shows the main processes in time step t, which includes two stages, the predicting stage and the adaptation stage.

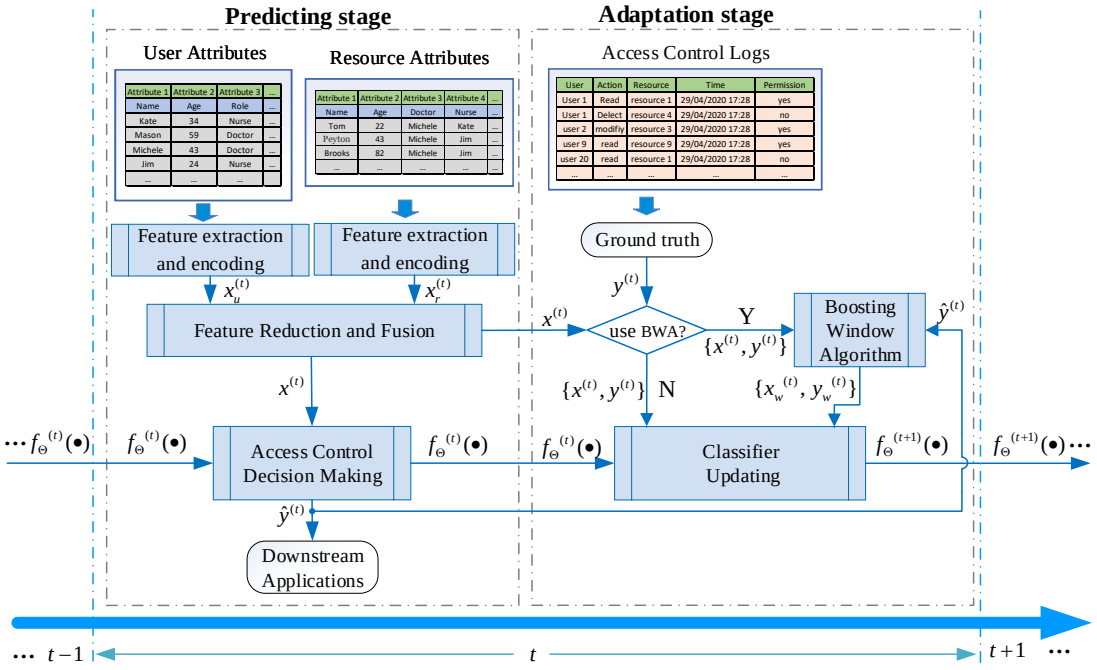

Fig. 3.1. Workflow of the proposed consecutive incremental batch learning framework

At the predicting stage, when a new access request from a user to a resource happens, the trained model can give a predictive result for downstream applications. Take time step t ($t \geq 0$) as an example. Firstly, features are extracted and encoded from user and resource attributes separately, denoted as $x_u^{(t)}$ and $x_r^{(t)}$. The available user attributes may include, for example, user name, age, department, group, position, service years, etc. Resource attributes may include, for example, resource id, name, owner, date created and modified, text description, etc. For non-numeric attributes, appropriate feature extraction or encoding methods are applied to extract features. Then, feature reduction and fusion algorithms are applied to generate the final feature set $x^{(t)}$ from $x_u^{(t)}$ and $x_r^{(t)}$. Let $f_\Theta(\cdot)$ is a randomly initialised binary ML classifier applied in this framework, Θ is the parameter set of $f(\cdot)$, $f_\Theta^{(t)}(\cdot)$ ($t \geq 1$) is the status at time step t, which is actually the classifier updated from $f_\Theta^{(t-1)}$ at time step $t-1$. The access control decision for the data batch at time step t would be made according the result of equation(3.1), which will be used to guide downstream applications.

$$\hat{y}^{(t)} = f_{\Theta}^{(t)}(x^{(t)}), (t \geq 1). \tag{3.1}$$

At the adaptation stage, the ground truth $y^{(t)}$ corresponding to $x^{(t)}$ would be extracted from the verified access control logs. If using BW algorithm to deal with the data imbalance problem, the labelled dataset $D^{(t)} = \{x^{(t)}, y^{(t)}\}$ as well as the predicted result $\hat{y}^{(t)}$ calculated by equation (3.1) are fed as the inputs of BW algorithm. The detailed implementation of BW algorithm will be presented in the following subsection. The output of BW algorithm is a deliberately tailored dataset $D_w^{(t)} = \{x_w^{(t)}, y_w^{(t)}\}$. Finally, the parameters of classifier $f_{\Theta}^{(t)}(x^{\cdot})$ would be fine-tuned based on the tailored dataset $D_w^{(t)}$ and turns into a new status, $f_{\Theta}^{(t+1)}(x^{\cdot})$, which would be used by the predicting stage of time step $t+1$. If the BW algorithm is not used, $f_{\Theta}^{(t)}(x^{\cdot})$ would be fine-tuned on the original labelled dataset $D^{(t)} = \{x^{(t)}, y^{(t)}\}$ directly. No matter if BW algorithm is applied, $f_{\Theta}^{(t)}(x^{\cdot})$ can adapt to any possible concept drifts existing in $D^{(t)}$, because it can be fine-tuned by the latest data batch $D^{(t)}$.

As shown in Fig. 3.1, the classifier, $f_{\theta}^{(t+1)}(\cdot)$, which is used by the window at time step $t+1$ is actually updated from $f_{\theta}^{(t)}(\cdot)$ using the data collected at time step t, $\{x_w^{(t)}, y_w^{(t)}\}$. In other words, the information from the previous window is used to help decision making for the next window.

3.3.2 Boosting Window Algorithm

BW algorithm is the key component of the proposed framework, aiming to boost the performance of the minority class, especially to decrease the false positive rate. The main idea of BW algorithm is to deliberately select a fixed size of samples as the boosting window samples to update the classifier at each time step. Algorithm 3.1 demonstrates how boosting window is implemented at time step t ($t \geqslant 1$).

The number of samples within the boosting window is called boosting window size, denoted as N_w, where N_w should be bigger than the step batch size N_s. The negative sample rate within the boosting window is denoted as r ($0 < r < 1$), which can be used to adjust the boosting strength of the negative samples. The

Algorithm 3.1 Boosting Window Algorithm

Input: $D^{(t)}$=$\{x^{(t)}, y^{(t)}\}$, $\hat{y}^{(t)}$, N_w, r, F_r.

Output: $D_w^{(t)}$=$\{x_w^{(t)}, y_w^{(t)}\}$.

if t=1 **then**
  x_{old}=∅; y_{old}=∅
end if
find out the indexes of all samples idx which satisfy that $y^{(t)} == \hat{y}^{(t)}$
$x^{(t)}$=$x^{(t)}[idx]$, $y^{(t)} = y^{(t)}[idx]$
$x_w^{(t)}$=concatenate(x_{old}, $x^{(t)}$)
$y_w^{(t)}$=concatenate(y_{old}, $y^{(t)}$)
$D_w^{(t)}$=$\{x_w^{(t)}, y_w^{(t)}\}$
if F_r==True **then**
  find out the indexes of all negative samples $idx0$ in $D_w^{(t)}$
  if len($idx0$)> $r * N_w$ **then**
    $idx0 = idx0$[-$r * N_w$: -1]
  end if
  find out the indexes of all positive samples $idx1$ in $D_w^{(t)}$
  if len($idx1$)>(1-r) $* N_w$ **then**
    $idx1$=$idx1$[-(1-r) $* N_w$:-1]
  end if
  idx=$idx0 \cup idx1$
  $x_w^{(t)}$=$x_w^{(t)}[idx,:]$; $y_w^{(t)}$=$y_w^{(t)}[idx]$
else
  if len($x_w^{(t)}$)> N_w **then**
    $x_w^{(t)}$=$x_w^{(t)}$[-N_W:,:]; $y_w^{(t)}$=$y_w^{(t)}$[-N_W:]
  end if
end if
x_{old}=$x_w^{(t)}$; y_{old}=$y_w^{(t)}$
return $D_w^{(t)}$=$\{x_w^{(t)}, y_w^{(t)}\}$

higher r is, the stronger boosting strength would be. The total number of negative samples within the boosting window is $r * N_w$ and the rest are positive samples.

The boosting window size is a preset fixed value. Based on the data distribution, the window size is big enough to run an algorithm. Furthermore, as time goes by, the number of minority class samples is accumulated based on the sample rate. Regarding the impact on performance, if the window size is too small, the classifier will be fine-tuned more frequently, and the computing cost is relatively high. By contrast, if the window size is too big, the classifier will be too dull to possible concept drifts of the new data.

The inputs of BW algorithm include the labelled dataset $D^{(t)} = \{x^{(t)}, y^{(t)}\}$ at time step t, the corresponding predicted output $\hat{y}^{(t)}$ calculated by $f_{\Theta}^{(t)}(\cdot)$, the boosting window size N_w and the negative sample rate r. The output is the selected dataset $D_w^{(t)}$=$\{x_w^{(t)}, y_w^{(t)}\}$ to boost the negative (minority) samples. F_r is a Boolean variable to indicate if applying the negative sample rate strategy in BW algorithm.

Steps 1 - 3 in Algorithm 3.1 show how to initialise the boosting window, where x_{old} and y_{old} denotes the old samples in the boosting window before the time step t.

Steps 4 - 8 show that the algorithm only picks the misclassified samples in time step t and put them into the boosting window.

If F_r is set to True, BW algorithm will apply a negative sample rate strategy as shown in steps 10 - 19. Specifically, steps 10 - 13 show how to crop the negative sample indexes so that the total number of negative samples in the boosting window is less than or equal to $r * N_w$. Similarly, steps 14 - 17 are to tailor the indexes of positive samples to make the number of positive samples less than or equal to $(1 - r) * N_w$. Steps 18 - 19 set the final samples in the boosting window, $\{x_w^{(t)}, y_w^{(t)}\}$.

Otherwise, if F_r is set to False, the BW algorithm will only keep the total number of samples in the boosting window equals N_w regardless of the ratio of negative samples, as shown in steps 21 - 23.

Step 25 is to prepare data for the following time step $t + 1$. The current samples in the boosting windows will be the old samples for the next time step.

To conclude, the boosting window algorithm boosts the minority class via (1) focusing on and boosting the misclassified samples as shown in steps 4-8, (2) adjusting the negative sample rate in the boosting window as shown in steps 10-19.

3.4 Experiment Results

This section reports the experimental methodology and the evaluation results in detail.

3.4.1 Dataset

This research uses a real-world Amazon employee access dataset[1] to conduct experiments and evaluate the performance of the proposed framework and BW algorithm. It contains 32,769 extremely imbalanced labelled samples, including 30,872 positive samples (access approval) and 1,897 negative samples (access rejection). Each sample contains eight user attributes and one resource attribute. Fig. 3.2 shows the data imbalance status over time, using a sliding window imbalance factor (SWIF) [75] as the indicator, where the sliding window size equals to 100.

3.4.2 Evaluation Metrics

Accuracy, Precision, Recall and F1 Score are in general four basic metrics widely employed in classification model evaluation. Among them, Accuracy, the percentage of correct predictions, is usually the most important metric to evaluate a model's performance. However, it becomes less instructive when data is extremely imbalanced. Taking the dataset used in this chapter as an example, the positive samples are almost count for 95% of the total dataset. Even if the model just directly make all predictions equal 1, its accuracy can reach 95%. Therefore,

[1] https://www.kaggle.com/c/amazon-employee-access-challenge

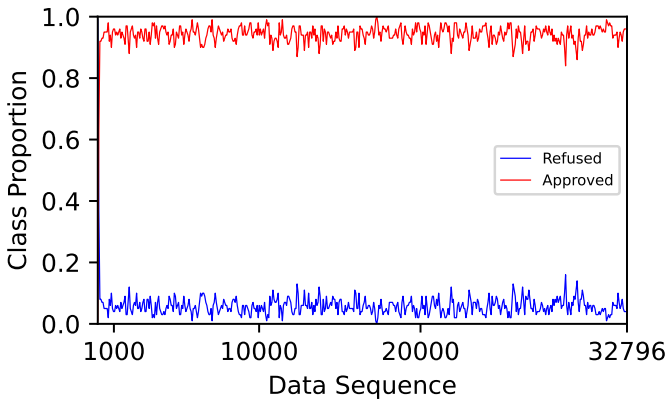

Fig. 3.2. Data imbalance status over time

Precision and Recall, which separately indicate a model's exactness and completeness in each class, are used as a supplement. F1 Score is the harmonic mean of Precision and Recall, thus it is usually considered as the decisive measure to decide which classier is better.

For some special applications, such as access control decision-making and biomedical event extraction[76], Recall is a more significant metric to evaluate the performance of a model. For example, for the access control problem, a false negative result (access request should be approved but refused) only leads to a re-application or manual review. However, a false positive result (access request should be refused but approved) may cause severe consequences, such as data breaches, privacy theft or cyber attacks.

Considering the specificity of the access control problem, we calculate Precision, Recall and F1 Score on class 0 instead of class 1 to show the classifier's performance on negative samples. Besides, we define a relative model cost C as equation (3.2) to represent the total misclassification cost of an access control model.

$$C = N_{FN} \times 1 + N_{FP} \times p, \tag{3.2}$$

where N_{FN} and N_{FP} represent the total number of false negative samples and false positive samples accordingly. Penalty factor p is the ratio of cost caused by a false positive sample and cost of a false negative sample. For access control

problems penalty factor $p \geqslant 1$.

To facilitate comparing the model cost of a series of access control models under different penalty factor settings, we define a normalised relative model cost $\widehat{C}_i$ for the i-th model as equation (3.3).

$$\widehat{C}_i = \frac{C_i}{max(C_1, C_2, \cdots, C_n)}, i = \{1, 2, \cdots, n\}, \tag{3.3}$$

where n is the total number of compared access control models.

3.4.3 Experimental Setting

The step batch size is a hyperparameter of the proposed framework. A larger step batch size means a lower model update frequency. Therefore, the access control decision model will have a slower response to existing concept drift. On the other hand, a smaller step batch size means more frequent model updates, which means a higher computational cost. Information systems can set different step batch sizes according to their own preference. In this chapter, for definiteness and without loss of generality, we set the step batch size to 100.

As the proposed framework is classifier-agnostic, the classifier $f_\Theta(\cdot)$ could be any binary ML classifiers, such as Neural Networks [45, 77], Logistic Regression, Support Vector Machine and Random Forest. Considering the verified universal approximation property, we adopt a full-connected three-layer neural network with ten hidden nodes as the classifier used in this framework.

3.4.4 Performance of Boosting Misclassified Samples

To demonstrate the performance of boosting misclassified sample strategy alone, we set F_r to False, as shown in Algorithm 3.1.

Fig. 3.3 shows the real-time performance on the minority class (class 0) when BW algorithm adopts boosting misclassified sample strategy alone. Baseline is the proposed framework shown in Fig. 3.1 without applying the BW algorithm. Data sequence starts from 1000 because the first 1000 samples are used to pretrain the classifier at time step $t = 1$ and are not used for evaluation.

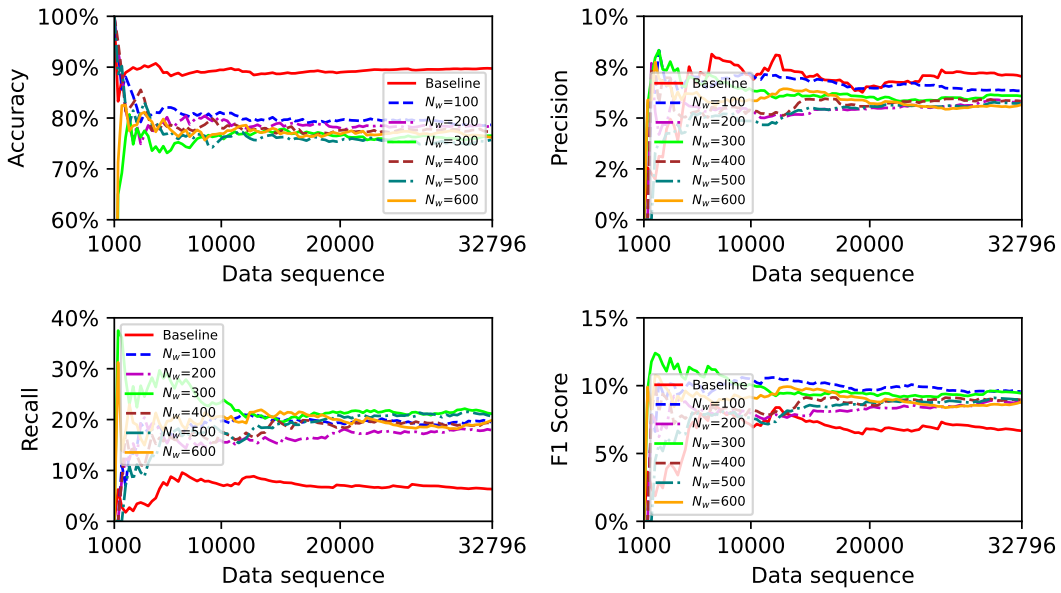

Fig. 3.3. Real-time performance comparison on different boosting window size (class 0)

Unsurprisingly, Baseline achieves the best Accuracy, but gets the worst Precision, Recall and F1 Score. Because there are much more positive samples in the data sequence, if no action is taken to boost the minority class, the classifier will tend to overfit on class 1 and underfit on class 0.

When applying BW algorithm and boosting the misclassified samples, we can see a significant increase in Recall and F1 Score, which are much more important metrics for access control problems with all boosting window size settings. Therefore, the strategy of boosting misclassified samples alone is effective to boost the performance of the minority class and decrease the false positive rate. Among all boosting window size settings, when N_w=300, the access control decision-making model records the best Recall and when N_w=100, it achieves the best F1 Score.

Accordingly, Table 3.1 summarises the overall performance among the whole dataset on the minority class when BW algorithm adopts boosting misclassified samples strategy alone. The window size is the total number of samples kept in a boosting window. We set the window size by grid searching a possible interval and choose the best value for the evaluation set. As shown in Table 3.1, we searched

the window size $N_w \in \{100, 200, 300, 400, 500, 600\}$ and the experimental results show that N_w =300 performs best on the dataset.

Table 3.1: Overall performance comparison on different boosting window size (class 0)

Metrics	Acc(%)	Pre(%)	Rec(%)	F1(%)	$\widehat{C}$ (when $p =$)(%)			
					1	10	100	1000
Baseline	**89.64**	6.91	6.38	6.64	**42.56**	**89.99**	100	100
$N_w = 100$	78.72	6.33	19.47	9.55	87.38	96.27	88.29	86.25
$N_w = 200$	78.32	5.76	17.94	8.72	89.04	98.09	89.96	87.89
$N_w = 300$	76.36	6.05	**21.32**	**9.43**	97.09	98.40	**86.78**	**84.32**
$N_w = 400$	77.20	5.80	19.36	8.93	93.64	98.67	88.68	86.40
$N_w = 500$	75.65	5.69	20.67	8.92	100	100	87.59	85.03
$N_w = 600$	76.15	5.61	19.79	8.74	97.97	99.94	88.42	85.95

As shown in Table 3.1, Baseline achieves the best overall Accuracy at 89.64%, which has an obvious advantage compared with others. However, models with all settings have very poor Precision performance, ranging from 5.61% to 6.91%, which means that there are only about 5 - 7 truly negative samples within every 100 predicted negative samples. The good thing is that with BW algorithm, the Recall on the minority class significantly increases from 6.38% to 21.31% when $N_w = 300$ and the F1 Score has also increased from 6.64% to 9.43%.

Apart from the above-mentioned four metrics, Table 3.1 also analyses the normalised relative model cost $\widehat{C}$ with different penalty factors. $\widehat{C}$ is a much more straightforward metric for decision-makers to choose the best access control decision-making model. When the penalty factor $p = 1$ or p =10, the baseline achieves the best normalised relative model cost. In this case, the cost of false negative samples equals to or slightly less than the cost of false positive samples. Therefore, the most important thing for a classifier is to increase the overall accuracy instead of improving the performance of the minority class. As p increases to 100 or even 1000, the false positive samples cost hundreds of or even thousands of times more than the false negative samples. Accordingly, the models achieving

the best Recall can get the lowest $\widehat{C}$, when N_w=300. The BW algorithm provides the flexibility for decision-makers to choose the best parameter based on their actual penalty factors.

Although both Fig. 3.3 and Table 3.1 have shown the effectiveness of boosting misclassified samples in boosting the minority class, the performance of Recall is far from enough to meet the actual requirements of the access control problem. Therefore, BW algorithm further designed a negative sample rate to adjust the sample ratio in the boosting window. The results with different negative sample rates are discussed in the following section.

3.4.5 Performance Comparison with Different Negative Sample Rates

When setting F_r=True, as shown in Algorithm 3.1, BW algorithm applies a negative sample rate to adjust the sample ratio in the boosting window. We set N_w=300, negative sample rate $r = \{0.2, 0.4, 0.6, 0.8, 0.9, 0.95\}$. Baseline is still the consecutive incremental batch learning framework shown in Fig. 3.1 without applying the BW algorithm.

Fig. 3.4 shows the real-time performance comparison on different negative sample rates on the minority class. Obviously, with the increase of negative sample rate r, the Accuracy of the access control decision-making models decrease monotonically, while their recall increases monotonically. All of the Precisions are at an equivalent low level. Baseline gets the worst F1 Score, followed by the model when $r = 0.2$. Others' F1 Scores are at a similar level.

Fig. 3.4 demonstrates the capability of BW algorithm to boost the Recall of the minority class to a very high level. Table 3.2 gives more details and quantity analyses on the overall performance comparison on different negative sample rates.

As shown in Table 3.2, as the negative sample rates used to update the classifier at each time step increases from around 0.05 (Baseline) to 0.95, the Accuracy decreases from 89.99% to 7.26%. Because with r increases, fewer positive samples

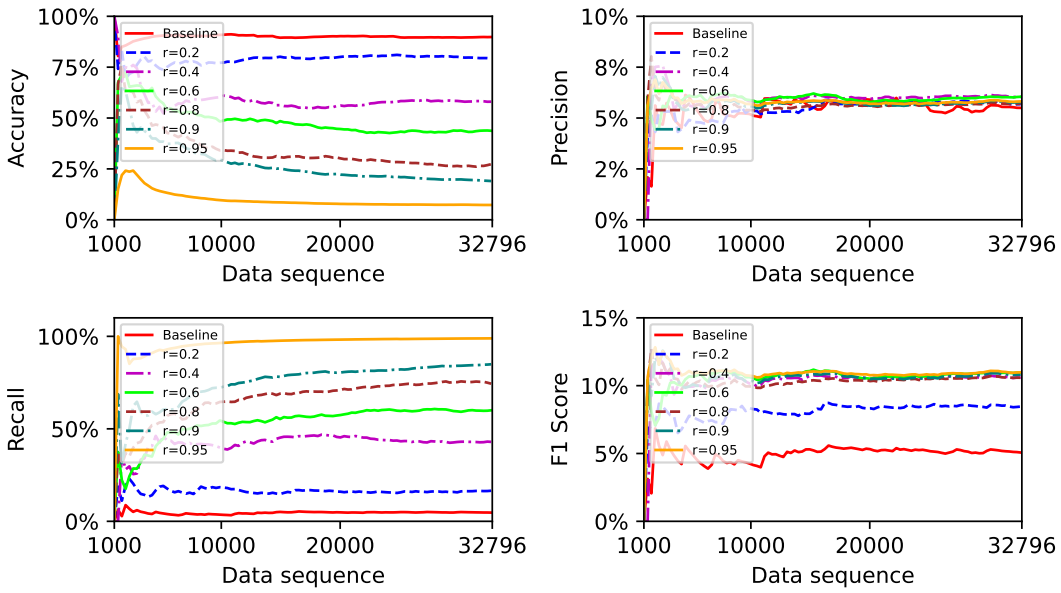

Fig. 3.4. Real-time performance comparison on different negative sample rates (class 0)

are selected to update the classifier and the model will perform worse on the majority class, which would lead to a decrease in Accuracy. On the other hand, as the negative sample rate r increases, the model can identify more negative samples. Thus, the Recall on the negative class dramatically increases from 6.38% to 98.91%. This demonstrates the capability of BW algorithm to control the Recall of the minority class. The F1 Score also increases from 6.64% to 10.96%.

As for normalised relative model cost $\widehat{C}$, when penalty factor p=1 or 10, Baseline with a high positive sample rate achieves the best while the model with $r = 0.95$ gets the worst performance. By contrast, when $p \geq 100$, models with $r = 0.95$ achieves the best $\widehat{C}$ while Baseline gets the worst.

One capability of the BW algorithm is to increase the performance of the minority class by adjusting the sample rate r of the boosting window. If no BW algorithm applies, the original sample rate of the dataset is 0.05, and the F1 score of the minority class is 6.64%, as shown in the Baseline line. When applying the BW algorithm and gradually increasing the sample rate to r =0.95, the F1 score of the minority class has increased from 6.64% to 10.96%. The results on $\widehat{C}$ also

Table 3.2: Sample rate comparison on the minority class

Metrics	Acc(%)	Pre(%)	Rec(%)	F1(%)	$\widehat{C}$ (when $p =$)(%)			
					1	10	100	1000
Baseline	**89.64**	6.91	6.38	6.64	**10.95**	**63.95**	100	100
$r = 0.2$	79.40	5.68	16.47	8.45	22.21	68.58	89.75	87.86
$r = 0.4$	57.98	6.01	42.86	10.54	45.31	76.85	66.43	60.61
$r = 0.6$	43.75	6.03	59.98	10.96	60.65	82.57	51.36	42.93
$r = 0.8$	27.42	5.69	74.32	10.57	78.25	92.08	39.53	28.21
$r = 0.9$	19.02	5.77	84.90	10.80	87.31	95.19	30.15	17.29
$r = 0.95$	7.26	5.80	**98.91**	10.96	100	100	**17.84**	**2.83**

demonstrate that the BW algorithm can decrease the over normalised relative model cost when p =100 and 1000.

For a real-world application, it is hard to say which setting is the best. A recommended practice is to calculate the $\widehat{C}$ of different models according to equation (3.2) and (3.3) based on actual penalty factors.

3.4.6 Discussion

As shown in Table 3.1 and Table 3.2, the performance on class 1 and class 0 are conflicting. In other words, the performance improvement in class 0 will hurt the performance of class 1. In real applications, trade-offs must be made to choose the most appropriate model for a particular application. In this case, specially designed domain-dependent metrics, such as normalised relative model cost $\widehat{C}$, could act as a better decisive metric for model selection.

3.5 Conclusion

In conclusion, there is an urgent need for technology to deal with various forms of internal data breaches due to concerns about data security and confidentiality. An accurate access control model based on machine learning can effectively prevent data leakage in companies or organisations around the world. To lead an

intelligent ML-based access control decision-making model, this chapter proposes a consecutive incremental batch learning framework to tackle the possible concept drift in real-world applications. Within the framework, a BW algorithm is specifically designed to deal with the severe data imbalance problem in the access control problem. As the minority class is much more important for the systems data security and privacy protection, BW algorithm focuses on the misclassified sample and designs a boosting window to boost the performance of the minority class. Experimental results on a real-world dataset demonstrate the effectiveness and flexibility of the proposed framework and BW algorithm.

Chapter 4

Knowledge Graph Empowered Online Learning Framework

Although experimental results demonstrated the framework proposed in Chapter 3 can enhance the performance of the minority class, the overall performance is still unsatisfactory because of the limited available attributes and the poor encoding and feature representing methods for high-cardinality categorical data. This chapter constructs an access control domain-specific knowledge graph to better represent user and resource and illustrate relationships between them to assist decision-making.

4.1 Introduction

With the popularisation of information systems and digital devices, enterprises and organisations accumulate a large amount of valuable or sensitive data locally or in the cloud [78, 79]. Once these data are leaked or used maliciously, it will cause significant economic losses or pose a great threat to users' privacy [25, 48, 80]. Secure sensitive information is an important issue to protect customers and then attract users [81, 82]. Access control is recognised as the first defence to guarantee that only authorised users can gain access to sensitive data and thus prevent data leakage [24, 25, 49, 50].

The two main categories of the most widely used access control strategies are role-based access control (RBAC) strategies and attribute-based access control (ABAC) strategies [25, 51, 52]. The former assigns permissions only based on user's roles, which makes it simple to implement and thus widely used in the past [59, 60]. However, with the expansion of the information system scale and the proliferation of users, RBAC strategies are too coarse-grained to meet the needs of sensitive data protection [53, 54]. By contrast, ABAC strategies adopt carefully crafted policies based on multiple attributes from users, environment and resources to assign data access permission. ABAC strategies have become more popular nowadays because they are more fine-grained and flexible than RBAC strategies [83, 84]. For example, the work in [85] proposed an ABAC mining algorithm named Rhapsody to mine ABAC rules from sparse logs.

However, the evolving new information technologies and changes in users' behaviours bring new challenges. One of the biggest problems is the policy explosion, which means the scale of the policy has increased dramatically [55, 56, 86]. The main reason for the policy explosion is that users' roles in organisations are becoming more diverse and people are using more different devices to access data in different places. Policy explosion brings two consequences directly, i.e., decreased efficiency of the system and increased misconfiguration [87, 88].

To overcome these problems, more and more researchers are beginning to explore machine learning based access control strategies, which treat access control decision-making as a binary classification problem. Sample features for machine learning classifier training come from available users, environment and resource attributes etc. The corresponding sample labels come from verified access control log files. Some works have successfully classified access control historical records using machine learning methods with high accuracy. But there is no related work that discusses machine learning based access control from the perspective of a data stream. In reality, access control requests form a data stream to feed into the decision-making models. Therefore, the work in our previous chapter [89] proposed a consecutive batch learning framework to tackle the possible concept drifts by periodically updating the machine learning classifier with new samples.

Furthermore, dynamic class imbalance problems exist in real-world access control applications [77, 90]. In other words, most requests are legitimate and

valid, but there will be a very small number of samples that are denied access due to mishandling or malicious attacks. For access control, a rejected access request usually means a malicious access request, which is the minority class. Misclassification of the minority class will cause severe data leakage. Therefore, improving the classification performance of the minority class (access deny) is vital for an access control problem.

To boost the performance of the minority class for access control, Chapter 3 [89] proposed a Boosting Window (BW) algorithm within an adaptive incremental batch learning framework. Although experimental results demonstrated the work in Chapter 3 can enhance the performance of the minority class, the overall performance is still unsatisfactory because of the limited available attributes and the poor encoding and feature representing methods for high-cardinality categorical data. For example, the manager ID is an essential user attribute related to the possible access permission to a specific system resource. However, in a large organisation, such as Amazon, there will be millions of different manager IDs. In this case, the values of the manager ID are high-cardinality nominal categorical data.

In general practice, one-hot encoding, binary encoding and label encoding are the most popular methods for categorical data encoding [45]. When encoding high-cardinality nominal categorical data, all of them have fatal disadvantages. Label encoding can mislead the classifier because of the big differences between numerical values. For example, the classifier can falsely give more weight to a manager with an ID of 100,000 than 1. One-hot encoding can address this problem but it will result in another serious problem, the curse of dimensionality. Binary encoding adopts binary code to represent ordinal values, working as a compromise between label encoding and one-hot encoding. However, binary encoding fails to represent relationships between different samples with the same attribute value.

In recent years, knowledge graphs have been increasingly used to represent complex data points and relationships in the real world. Existing knowledge graph application areas include question-answering systems, storing research, recommendation systems and supply chain management. Studies show that bringing knowledge graph and machine learning technology can improve the accuracy and performance of machine learning approaches, because knowledge graph provides

a logical way to capture data relationships and drive intelligence into the data itself in a more explainable, accurate and repeatable way. The performance of machine learning algorithms depends on the quality of data. Knowledge graphs capture, persist and make rich contextual information usable to enhance every step of the machine learning processes, from training machine learning models and extracting topological and non-topological features to making predicting decisions.

To better represent user and resource and illustrate relationships between them, this chapter constructs an access control domain-specific knowledge graph to assist decision-making. As an extension of Chapter 3, we leverage knowledge graph to handle user and resource attributes with high-cardinality values to further boost the performance of the minority class. Compared with the work in [89], our main contributions are as follows.

(1) We proposed a knowledge graph empowered online learning framework for access control decision-making. To the best of our knowledge, this study is the first try to leverage knowledge graph to extract graph topological features to improve the performance of the access control model.

(2) We proposed an algorithm to construct a knowledge graph from the existing user and resource attributes. We further demonstrate how to extract features from the established knowledge graph to represent users and resources. The extracted features are fed to a machine learning classifier to make access control decisions based on records in log files.

(3) We evaluate and verify the proposed knowledge graph empowered online learning framework on a much larger open-sourced real-word dataset and discussed the performance on different imbalance degrees in both online and offline scenarios.

The rest of the chapter is organised as follows. Section 4.2 briefly introduces knowledge graph basics, typical graph topological features and link prediction solutions. Section 4.3 presents the workflow of the proposed framework, followed by an access control domain-specific knowledge graph construction algorithm and feature extraction details in section 4.4. Section 4.5 displays the experimental results and concludes the chapter with a discussion on future work.

4.2 Related Work

A knowledge Graph (KG) is a complex data structure consisting of entities, also known as nodes from the perspective of graph theory, and the relationships between them. Apart from the graph-structured data model, both entities and relationships can have multidimensional properties to further describe complex data. KG is often used to represent interlinked facts, allowing both humans and computers to extract useful knowledge and further to do reasoning and prediction based on its contents. Typical ways to analyse a knowledge graph include but are not limited to (1) node classification to predict the type of a given node; (2) link prediction to predict whether two entities are linked or not; (3) community detection to identify densely linked entity clusters and (4) network similarity measurement to evaluate the similarity between two nodes or two networks.

The access control problem can be formulated as a link prediction problem between user entities and resource entities, which is essentially a binary classification problem. Specifically, if an access approve link exists between a user entity u and a resource entity r, the access request $(u \to r)$ will be approve. Otherwise, it will be refused. Once a knowledge graph has been constructed, a variety of graph topological features can be extracted to describe the local or global connections between entities based on homogeneous or heterogeneous subgraphs within the knowledge graph.

A basic solution for link prediction is structural similarity-based unsupervised learning methods, which determine the likelihood of linkage between two nodes based on some similarity or closeness indices deduced from the graph structure. When an index between two nodes exceeds a predefined threshold, they are considered to have a link between them. Common Neighbours (CN) [91] measuring the number of shared nodes between two nodes is the most intuitionistic index to indicate the linkage possibility of them. Similar indices, to name a few, include Adamic Adar (AA) [92], Preferential Attachment (PA) [93] and Resource Allocation (RA) [94]. Their definitions are listed as (4.1)-(4.3) for reference.

$$CN(u, v) = |\mathcal{N}(u) \cap \mathcal{N}(v)|, \tag{4.1}$$

$$AA(u,v) = \sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{\log |\mathcal{N}(w)|}, \tag{4.2}$$

$$PA(u,v) = |\mathcal{N}(u)| * |\mathcal{N}(v)|, \tag{4.3}$$

$$RA(u,v) = \sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{|\mathcal{N}(w)|}, \tag{4.4}$$

where u, v, w are nodes in the target graph, $\mathcal{N}(\cdot)$ denotes the set of nodes adjacent to the specified node in the brackets, $|\cdot|$ denotes the number of distinct nodes in the specified set. These indices are widely used in various domains because of their simplicity and reasonable performance. However, they only considered the node pair's local connectivity and ignored the global structure of a graph.

By contrast, global connectivity indices can provide more overall graph topology information. A well-known index for taking global connectivity into account is the Katz Index (KI), which leverage the length of paths between a pair of nodes to measure their similarity. KI can be calculated as (4.5) [95].

$$KI(u,v) = \sum_{l=1}^{l_{\max}=\infty} \beta^l \cdot \left|\text{path}_{u,v}^l\right|, \tag{4.5}$$

where l is the length of a path between nodes u and v, $\left|\text{path}_{u,v}^l\right|$ is the total number of distinct paths between node u and v with length l, β is a coefficient between 0 and 1 used to adjust the contribution of paths to KI.

Another popular global connectivity index is Average Commute Time (ACT), which calculates the average number of steps required by a random walker starting from node u to reach v and vice versa [96]. The ACT between nodes u and v can be calculated as (4.6) [97].

$$S_{ACT}(u,v) = \frac{1}{l_{uu}^+ + l_{vv}^+ - 2l_{uv}^+} \tag{4.6}$$

where l_{uu}^+, l_{vv}^+ and l_{uv}^+ are the corresponding entries in Laplacian Matrix, L^+.

Obviously, a common drawback for global connectivity indices is relatively higher computation cost compared with local connectivity indices. Decentralized approaches or parallel computing are also incapable of dealing with global graph

computation, because the structural connectivity would be damaged by splitting the graph for decentralized or parallel computing. Therefore, these measures are not suitable for large-scale connected graphs.

Generally speaking, the common advantages of structural similarity-based unsupervised learning methods algorithms include that (1) they do not need labelled data to train a classifier; (2) the link prediction result is explainable based on the definition of the corresponding indices; (3) they often take less computation effort for costly feature engineering and classifier training procedures.

However, there is still no universal feasible method to determine the appropriate threshold for different indices and application domains. Besides, these methods are also criticised for poor performance due to only taking topological features into account and neglecting the attributes of nodes and relationships, which contain rich domain knowledge and play critical roles for most domain-specific link prediction tasks. Therefore, in most cases, when labelled data is available, supervised learning methods are more preferable due to superior performance and the flexibility of feature extraction.

When applying supervised learning methods, both non-topological features and topological features can be used to feed into a machine learning classifier to support link prediction. Non-topological features refer to the attributes of entities and relationships, which contain rich multi-modality domain knowledge. For example, in an access control knowledge graph, the non-topological features of a user entity include sector, department name, job title, job description, etc. By contrast, topological features refer to graph structural features for node representing. In addition to aforementioned local and global connectivity indices, common traditional node topological feature extraction methods in graph theory include Pang Rank [98], Article Rank [99], Betweenness Centrality [100], Harmonic Centrality [101], etc.The performance of supervised machine learning methods for link prediction is determined by the capability of the extracted non-topological and graph topological features as well as the capability of the applied classifier.

4.3 Methodology

We propose a general knowledge graph empowered online learning framework for access control in this section. Firstly, we introduce the workflow of the framework. Then we detail the construction algorithm of an access control domain-specific knowledge graph and the KG-based topological feature extraction method.

4.3.1 Workflow of the Proposed Framework

The supporting information for the access control decision-making problem studied in this chapter includes user attributes, resource attributes and a verified access control log file in chronological order. According to the cardinality of category user attributes and resource attributes, an access control knowledge graph is constructed. The specific knowledge graph construction and refactoring algorithm is given in Section 4.3.2 and a real-world use case is demonstrated in Section 4.4.

Similar to our previous work [89], the proposed framework is essentially a classifier-agnostic consecutive incremental batch learning process for access control decision-making. Within this framework, a randomly initialized binary machine learning classifier works as the access control decision-maker, denoted as $f_{\Theta}^{(0)}(\cdot)$, where Θ is the trainable parameter set of $f(\cdot)$ and (0) means the initialization status of the time step. The classifier $f_{\Theta}^{(0)}(\cdot)$ is constantly updated at each time step as new samples are available for classifier training. We demonstrate the main process of a typical time step t ($t > 0$) in Fig. 4.1, which consists of two stages, namely, the predicting stage and the adaptation stage.

At the predicting stage of the t-th time step, when user u request a resource r, denoted as $(u \to v)^{(t)}$, the classifier $f_{\Theta}^{(t)}(\cdot)$, which is updated at time step $t-1$, will make decision on the access control request $(u \to v)^{(t)}$. Firstly, six feature sets related to this access control request will be extracted from the constructed access control knowledge graph, i.e., $x_{u_N}^{(t)}$, $x_{u_T}^{(t)}$, $x_{r_N}^{(t)}$, $x_{r_T}^{(t)}$, $x_{(u\to r)_N}^{(t)}$ and $x_{(u\to r)_T}^{(t)}$. Among them, $x_{u_N}^{(t)}$, $x_{r_N}^{(t)}$ and $x_{(u\to r)_N}^{(t)}$ are the non-topological feature sets extracted form the user entity, the resource entity and the existing relationships between them. Similarly, $x_{u_T}^{(t)}$, $x_{r_T}^{(t)}$ and $x_{(u\to r)_T}^{(t)}$ are the corresponding topological feature

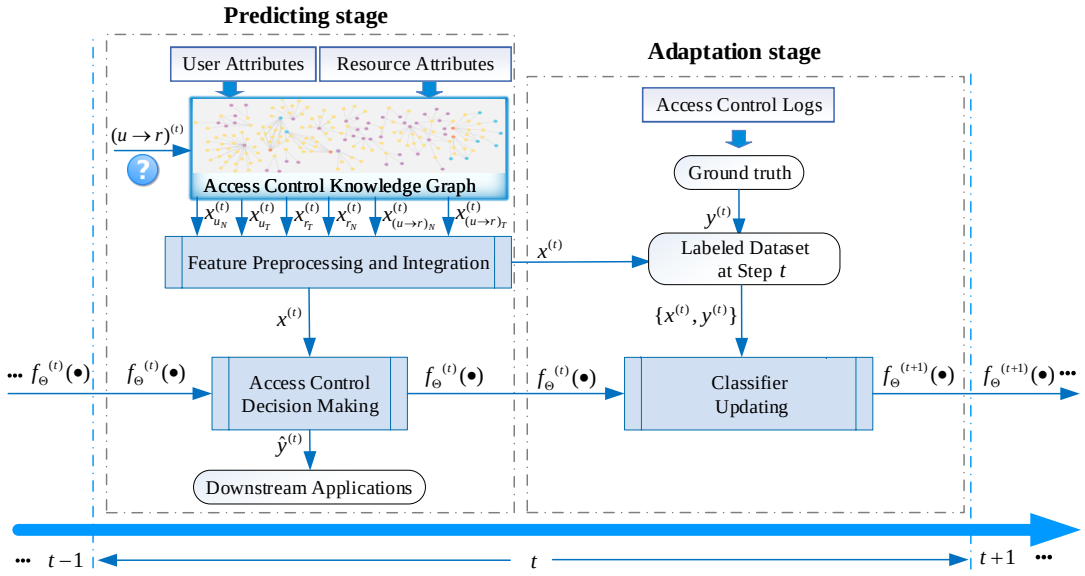

Fig. 4.1. Workflow of the proposed consecutive incremental batch learning framework

sets. The details of the feature extraction process are described in Section 4.3.3. These six feature sets are then preprocessed (outlier replacing and normalization) and integrated into one feature set $x^{(t)}$. Finally, the classifier $f_\Theta^{(t)}(\cdot)$ will make decision on the request $(u \to v)^{(t)}$ according to the result of equation (4.7).

$$\hat{y}^{(t)} = f_\Theta^{(t)}(x^{(t)}), (t > 0). \tag{4.7}$$

At the adaptation stage of the t-th time step, the verified ground truth $y^{(t)}$ corresponding to the request $(u \to v)^{(t)}$ is available and can be extracted from the verified access control log file. Then, the labelled samples $x^{(t)}$, $y^{(t)}$ can be used to finetune the classifier $f_\Theta^{(t)}(\cdot)$. The fine-tuned classifier, denoted as $f_\Theta^{(t+1)}(\cdot)$, will be used at the predicting stage of the $t+1$-th time step. Since the classifier keeps updating with the latest verified samples $\{x^{(t)}, y^{(t)}\}$ at each time step t, it can learn possible new concepts emerging at time step t.

4.3.2 Access Control Knowledge Graph Construction

A knowledge graph consists of a set of entities (with multiple entity labels) and relationships between entities (with multiple relationship types). Each entity or relationship has its identification number and some of them have one or more properties. To construct an access control knowledge graph is to identify all entities including their labels and properties, and all relationships including their relationship types.

4.3.2.1 Attribute Type

We construct an access control knowledge graph $\mathcal{G}$ from existing user attributes and resource attributes information. Apart from the ID attribute, from the perspective of constructing KGs, there are three kinds of attributes in Att_u and Att_r, i.e., Type 1, attributes showing the relationships between users and resources; Type 2, high-cardinality categorical attributes; Type 3, the rest attributes. Let θ be a preset cardinality threshold. If the cardinality of a categorical attribute is larger than θ, it is a Type 2 attribute; otherwise, Type 3. The attribute types work as a guideline for step by step knowledge graph construction, seeing details at Section 4.3.2.2.

4.3.2.2 Algorithm Pseudocode

Let Att_u be the list of users' attribute names, in which an attribute name'userID' is included. X_u denotes the attributes' values according to Att_u. $x_{uid} \in X_u$ is a vector containing all users' ID. Similarly, Att_r is the list of resources' attribute names containing a 'resourceID'. X_r is the attributes' values according to Att_r and $x_{rid} \in X_r$ is a vector containing all resources' ID. We elaborate on the construction process of an access control knowledge graph in Algorithm 4.1.

Algorithm 4.1 Access Control Knowledge Graph Construction

Input: Att_u, X_u, Att_r, X_r, θ.

Output: $\mathcal{G}$.

\\ Step1: create User and Resource entities.

for *uid* in unique(x_{uid}) **do**

```
    CREATE (u:User {u.userID=uid})
end for
for rid in unique(x_rid) do
    CREATE (r:Resource {r.resourceID=rid})
end for
\\ Step2: create properties or relationships for User entities from user attributes.
for attn, x_u in zip(Att_u, X_u-x_uid) do
    \\ Step2.1: create relationships from User entities to Resource entities from Type 1 attributes.
    if x_u shows a relationship between users and resources then
        for uid, attv in zip(x_uid, x_u) do
            Let the set of resourceIDs indicated by attv be denoted as Rid_t.
            for rid_t in Rid_t do
                MATCH (u:User userID:uid)
                MATCH (r:Resource resoureID:rid_t)
                CREATE (u)-[ref:HASattn (ref.attnProperty=attv)]->(r)
            end for
        end for
        \\ Step2.2: create new types of entities from the Type 2 attributes.
    else if x_u is a category feature AND cardinality(x_u) > θ then
        for uid, attv in zip(x_uid, x_u) do
            CREATE (a:attn {a.attnProperty=attv})
            CREATE (u)-[ref:HASattn]->(a)
        end for
        \\ Step2.3: create new properties for User entities from the Type 3 attributes.
    else
        for uid, attv in zip(x_uid, x_u) do
            MATCH (u:User userID:uid)
            SET u.attnProperty=attv
        end for
    end if
```

end for
\\ Step3: create properties or relationships for Resource entities from Resource attributes.
for $attn$, x_r in zip(Att_r, X_r-x_{rid}) **do**
  \\ Step3.1: create relationships from Resource entities to User entities from Type 1 attributes.
  if x_r shows a relationship between resources and users **then**
    for rid, $attv$ in zip(x_{rid}, x_r) **do**
      Let the set of userIDs indicated by $attv$ be denoted as Uid_t.
      for uid_t in Uid_t **do**
        MATCH (r:Resource resoureID:rid)
        MATCH (u:User userID:uid_t)
        CREATE (r)-[ref:HAS$attn$ (ref.$attn$Property=$attv$)]->(u)
      end for
    end for
    \\ Step3.2: create new types of entities from the Type 2 attributes.
  else if x_r is a category feature AND cardinality(x_r) $> \theta$ **then**
    for rid, $attv$ in zip(x_{rid}, x_r) **do**
      CREATE (a:$attn$ {a.$attn$Property=$attv$})
      CREATE (r)-[ref:HAS$attn$]->(a)
    end for
    \\ Step3.3: create new properties for Resource entities from the Type 3 attributes.
  else
    for rid, $attv$ in zip(x_{rid}, x_u) **do**
      MATCH (r:Resource resourceID:rid)
      SET r.$attn$Property=$attv$
    end for
  end if
end for
\\ Step4: refactor the above-established access control knowledge graph.
for rel in Rel **do**
  MATCH (u1:User)-[:rel]->()<-[:rel]-(u2:User)

63: CREATE (u1)-[:SHARE*rel*]-(u2)
64: MATCH (r1:Resource)-[:*rel*]->()<-[:*rel*]-(r2:Resource)
65: CREATE (r1)-[:SHARE*rel*]-(r2)
66: **end for**
67: **return** an access control knowledge graph $\mathcal{G}$

Some executive statements of the pseudocode in Algorithm 4.1 are written in Cypher query language, which is the graph query language for the Neo4j graph database. The naming convention thus follows the Cypher coding standards, where entity labels are in CamelCase; property keys are in camelCase and relationship types are in upper-case, such as FOLLOWS in a social media knowledge graph. As listed below, the process of access control knowledge graph construction can be divided into four main steps:

Step 1: create User and Resource entities as shown in lines 2-7. According to the userID and resourceID attributes, we create two entity types with a User label and Resource label respectively. For each unique userID *uid* in x_{uid}, we create a User entity with a userID property as shown in lines 2-4. Similarly, we create Resource entities with a resouceID property based on the *rid* in x_{rid} as shown in lines 5-7.

Step 2: create properties or relationships for User entities from user attributes as shown in lines 9-32. In line 9, *attn* refers to the attribute name traversing Att_u; x_u is the corresponding attribute values and X_u-x_{uid} means the relative complement of x_{uid} in X_u. In other words, X_u-x_{uid} means all attributes' values except x_{uid}. Steps 2.1-2.3 give details on how to create properties or relationships for User entities based on three attribute types defined in Section 4.3.2.1.

Step 2.1: create relationships from User entities to Resource entities based on Type 1 attributes as shown in lines 11-19. When a user attribute *attn* indicates a relationship between users and resources, we search the particular User entity and Resource entity based on the attribute value *attv* and create a HAS*attn* relationship between, where HAS*attn* is the relationship type in upper-case format. We further record the import attribute information in *attv* as a property of the created relationship, named *attn*Property. In line 12, *attv* means the attribute value of *attn* corresponding to the user with userID=*uid*. In line 13, Rid_t means a temporary resourceID set to distinguish it from x_{rid} line 5.

Step 2.2: create new types of entities for high-cardinality categorical user attributes as shown in lines 21-25. To better represent high-cardinality categorical features, we create new entities with a label named *attn* and a property named *attn*Property to record the value of the high-cardinality categorical user attribute. Then, we create a relationship with a type of HAS*attn* to indicate that the user with userID=*uid* has a relation with the newly created entity.

Step 2.3: create new properties for User entities from the Type 3 attributes. The rest of user attributes are all added as the properties of the User entities as shown in lines 28-31.

Step 3: create properties or relationships for Resource entities from resource attributes as shown in lines 34-57. The process of Step 3 is similar to Step2. To avoid redundancy, we no longer describe the detailed process in words.

Step 4: refactor the above-established access control knowledge graph as shown in lines 61-61. We add a SHAREref relationship between User Entities who share the same *attn* entities created in line 23. The subgraph consisting of SHAREref relationships and User entities can be used to extract topological features to represent the original user attribute $attn \in Att_u$. Similarly, we also add a SHAREref relationship between Resource Entities who share the same *attn* entities created in line 49 to facilitate the topological feature extraction from information provided by the original user attribute $attn \in Att_r$.

After the aforementioned four steps, an access control knowledge graph $\mathcal{G}$ is established for topological feature extraction.

4.3.3 Feature Extraction for Access Control

To train the classifier $f_\Theta(\cdot)$ for access control, we use the log file containing access control requests and their corresponding verified decision (approval or refuse) to form labelled samples. Specifically, for each request from user u to resource r at time step t, denoted as $(u \to v)^{(t)}$, six sets of features can be exacted from the access control knowledge graph $\mathcal{G}$ constructed with Algorithm 4.1, namely, $x_{u_N}^{(t)}$, $x_{u_T}^{(t)}$, $x_{r_N}^{(t)}$, $x_{r_T}^{(t)}$, $x_{(u\to r)_N}^{(t)}$ and $x_{(u\to r)_T}^{(t)}$, as shown in Fig. 4.1.

Among them, $x_{u_N}^{(t)}$, $x_{r_N}^{(t)}$ and $x_{(u\to r)_N}^{(t)}$ are the non-topological feature sets extracted form the User entity u, the Resource entity r and the existing relationships

between them $u \to v$. These three non-topological feature sets can be exported from the properties of entities u, r and relationships $u \to r$.

By contrast, $x_{u_T}^{(t)}$, $x_{r_T}^{(t)}$ and $x_{(u \to r)_T}^{(t)}$ are the corresponding topological feature sets. $x_{u_T}^{(t)}$ is extracted from a subgraph of the constructed access control knowledge graph $\mathcal{G}$ which consists of User entities and relationships between them. Similarly, $x_{r_T}^{(t)}$ is extracted from a subgraph containing Resource entities and relationships between them. Both $x_{u_T}^{(t)}$ and $x_{r_T}^{(t)}$ are the topological features extracted to present the entities. The extracted topological features include but are not limited to (1) centrality scores which determine the importance of distinct nodes in a graph, such as page rank scores and betweenness scores; (2) community detection scores which indicate how groups of nodes are clustered or partitioned, as well as their tendency to strengthen or break apart, such as the weakly connected component id and triangle count of an entity. $x_{(u \to r)_T}^{(t)}$ is extracted from a subgraph containing User entities, Resource entities and relationships between User and Resource entities. $x_{(u \to r)_T}^{(t)}$ is used to present the closeness of entities u and r based on the graph with relationships between u and t. The possible features of $x_{(u \to r)_T}^{(t)}$ include but are not limited to Adamic Adar scores and common neighbours.

4.4 Experiment Results

This section introduces a real-world access control dataset. Then provides a use case of the access control knowledge graph construction algorithm described in Algorithm 4.1 on this dataset. Finally, we compare the access control performance on topological features extracted from the established knowledge graph and nontopological features. Results show that the proposed knowledge graph empowered method outperforms nontopological methods in both offline and online scenarios.

4.4.1 Dataset

The experiments of this chapter are conducted on an open-source real-world Amazon employee access dataset[1]. The dataset contains a file listing all user and resource attributes and a time-series log file containing 684,374 user to resource access control requests and the corresponding permission records. The dataset is extremely imbalanced with 10,911 (1.59%) access rejection and 673,463 (98.41%) access approval. The dynamic data imbalance status is shown in Fig. 4.2. Subplot (a) shows the overall imbalance factor of the refused requests and approved requests. The overall imbalance factor of the refused requests gradually converges to 1.59% after a fluctuation at the early stages and the approved requests converge to 98.41%. Subplot (b) shows the sliding window imbalance factor [75] of the two classes when the sliding window size is set to 100.

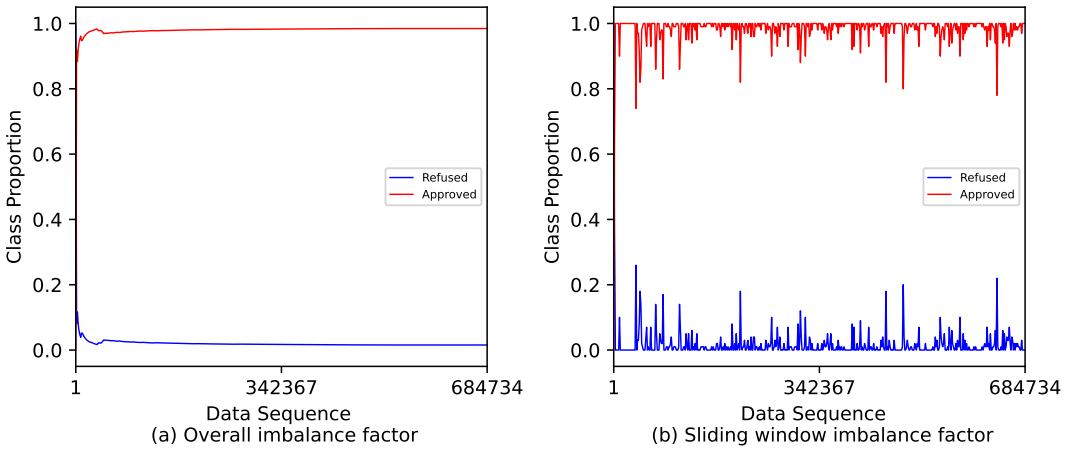

Fig. 4.2. Dynamic data imbalance statuses over time

Table 4.1 lists the basic information of the attribute file. means Not Applicable As shown in Table 4.1, there are 36,063 unique users and 33,252 unique resources. We set the cardinality threshold θ=300. Based on the three attribute types defined in Section 4.3.2.1, the corresponding attribute type is listed in the Type column. The type information can be used to guide the knowledge graph

[1]http://archive.ics.uci.edu/ml/datasets/Amazon+Access+Samples

construction, which is described in Section 4.4.2. For the Type 1 attribute, the cardinality is not applicable (NP).

Table 4.1: Dataset information

Attribute file	Attribute name	Type	Cardinality	Description
User	PERSON_ID	userID	36,063	ID of the user
	RESOURCE_LIST	Type 1	NP	list of resource ID that a users can possibly have access to
	MGR_ID	Type 2	3,207	manager ID
	DEPTNAME	Type 2	405	department description ID
	BUSINESS_TITLE	Type 2	4,979	title ID
	TITLE_DETAIL	Type 3	56	title description ID
	COMPANY	Type 3	49	company ID
	JOB_CODE	Type 3	13	job code ID
	JOB_FAMILY	Type 3	70	job family ID
	ROLLUP_1	Type 3	12	user grouping ID
	ROLLUP_2	Type 3	111	user grouping ID
	ROLLUP_3	Type 3	12	user grouping ID
Resource	RESOURCE_ID	resourceID	33,252	ID of the resource
	RESOURCE_TYPE	Type 3	3	group, system or host

4.4.2 Access Control Knowledge Graph Construction

According to the dataset introduced in Section 4.4.1, we construct an access control knowledge graph following the steps in Algorithm 4.1. The main process and intermediate knowledge graph construction results are summarised in Table 4.2.

In Step 1, based on the user attribute PERSIN_ID, 36,063 User entities with a userID property are created and 33,252 Resource entities with a resourceID property are created using the RESOURCE_ID attribute.

Table 4.2: Usecase of Algorithm 4.1

Step	Used information	Created entity	Created relationship	Created property
Step 1	PERSON_ID, RESOURCE_ID	User entities, Resourc entities	none	u.userID, r.resourceID
Step 2.1	RESOURCE_LIST	none	HAS_P_ACCESS	none
Step 2.2	MGR_ID, DEPTNAME, BUSINESS_TITLE	Manager, Department, Title	HAS_MANAGER, HAS_DEPT, HAS_TITLE	m.managerID, d.deptID, t.titleID
Step 2.3	TITLE_DETAIL, COMPANY, JOB_CODE, JOB_FAMILY, ROLLUP_1, ROLLUP_2, ROLLUP_3	none	none	u.titleDetail, u.company, u.jobCode, u.jobFamily, u.rollup1, u.rollup2, u.rollup3
Step 3.1	none	none	none	none
Step 3.2	none	none	none	none
Step 3.3	RESOURCE_TYPE	none	none	r.resourceType
Step 4	HAS_P_ACCESS, HAS_MANAGER, HAS_DEPT, HAS_TITLE	none	SHARE_P_USER, SHARE_MANAGER, SHARE_DEPT, SHARE_TITLE	none

In Step 2, more entities, relationships and properties are created based on the three types of user attributes. Specifically, in Step 2.1, a HAS_P_ACCESS relationship is created between User and Resource entities to show the possibility of access requests based on Type 1 RESOURCE_LIST attributes. In step 2.2, three Type 2 attributes, namely, MGR_ID, DEPTNAME and BUSINESS_TITLE, are used to create three types of entities. The newly created entity labels are Manager, Department and Title respectively. The attribute values are added as the entity properties as shown in Table 4.2. The m.managerID means we created

a manageID property for Manager entities. Similarly, d.deptID and t.titleID are properties added to Department and Title entities respectively. Furthermore, a HAS_MANAGER relationship is created between User and Manager entities. Similarly, a HAS_DEPT and a HAS_TITLE relationship is also created between User and Department entities as well as User and Title entities. Finally, in Step 2.3, 7 Type 3 attributes are added as the properties of User Entities, as shown in Table 4.2.

In Step 3, only a Type 3 attribute, RESOURCE_TYPE, is available and we add a resourceType property to Resource entities.

In Step 4, a SHARE_P_USER relationship is created between two Resource entities who have HAS_P_ACCESS relationship with the same User Entity. Similarly, three relationships, namely, SHARE_MANAGER, SHARE_TITLE and SHARE_DEPT, are created respectively between two User entities who have HAS_MANAGER/ HAS_TITLE/ HAS_DEPT relationships with the same Manager/ Title/ Department entity.

Finally, an access control knowledge graph $\mathcal{G}$ is constructed based on the Amazon access control dataset. The data model (schema) of $\mathcal{G}$ is illustrated as Fig. 4.3. A circle presents a type of entity with a bold label inside. Below the label is the total number of entities with that label. The properties of the corresponding entities are also listed inside the circle. An arrow represents a directed relationship. We also specify the relationship type and the total number of relationships along the arrow.

4.4.3 Feature extraction

We implement the access control knowledge graph $\mathcal{G}$ on the Neo4j [1] graph data platform, which provides a convenient way for both topological and nontopological feature extraction from existing entities, relationships and subgraphs of a knowledge graph. The topological features adopted in this chapter are implemented with the Neo4j Graph Data Science Library [2].

[1] https://neo4j.com/

[2] https://neo4j.com/docs/graph-data-science/current/algorithms/

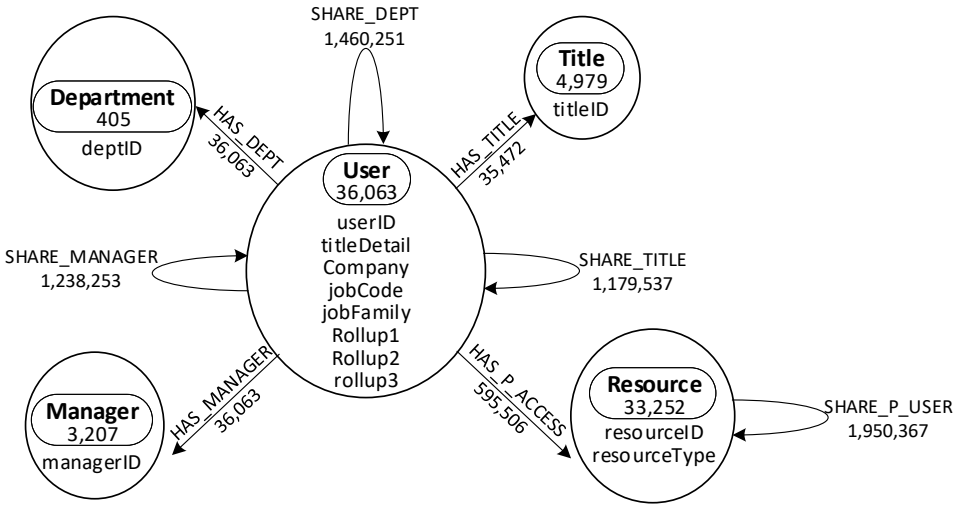

Fig. 4.3. The data model of the constructed access control knowledge graph $\mathcal{G}$

For an access control request from a user u to a resource r, six feature sets, i.e., x_{u_N}, x_{u_T}, x_{r_N}, x_{r_T}, $x_{(u\to r)_N}$, $x_{(u\to r)_T}$ are extracted from the User entities, Resource entities and their relationships. Table 4.3 presents our feature extraction strategies in detail. As shown in the first row, x_{u_N} is extracted from the properties of the User entity u. x_{u_T} presents the topological features extracted from subgraphs containing User entities and the relationships between them, as shown in the second row. The listed features are extracted to represent the importance or connectivity characteristics in the subgraphs. Similarly, x_{r_N} and x_{r_T} are the nontopological and topological features of Resource entity r. In this usecase, no properties are added to the relationship SHARE_P_USER, therefore, $x_{(u\to r)_N}$ is none. We select two link prediction topological features, i.e., preferentialAttachment [1] and totalNeighbor [2], to present $x_{(u\to r)_T}$ in this chapter.

[1]https://neo4j.com/docs/graph-data-science/current/alpha-algorithms/preferential-attachment/

[2]https://neo4j.com/docs/graph-data-science/current/alpha-algorithms/total-neighbors/

Table 4.3: Feature extraction details

Feature	Source	Features
x_{u_N}	User entity	u.userID, u.titleDetail, u.Company, u.jobCode, u.jobFamily, u.Rollup1, u.Rollup2, u.rollup3
x_{u_T}	subgraphs: (u1:User)-[rel:SHARE_MANAGER]-(u2:User), (u1:User)-[rel:SHARE_DEPT]-(u2:User), (u1:User)-[rel:SHARE_TITLE]-(u2:User)	PageRank, ArticleRank, Betweenness, Degree, Closeness, Louvain, HarmonicCloseness, LabelPropagation, WCC, triangleCount, Modularity
x_{r_N}	Resource entity	r.resourceID, r.resourceType
x_{r_T}	subgraph: (r1:Resource)-[rel:SHARE_P_USER]-(u2:User)	PageRank, ArticleRank, Betweenness, Degree, Closeness, Louvain, HarmonicCloseness, LabelPropagation, WCC, triangleCount, Modularity
$x_{(u\to r)_N}$	relationship: SHARE_P_USER	none
$x_{(u\to r)_T}$	subgraph: (u:User)-[rel:HAS_P_ACCESS]-(r:Resource)	preferentialAttachment, totalNeighbor

4.4.4 Offline Learning Performance Comparison

To verify the effectiveness of the proposed knowledge graph empowered framework, we compared the access control decision-making performance of using topological features extracted from established knowledge graph and nontopological features from original user and resource attributes on both online and offline

scenarios.

Firstly, we verify the performance improvement of topological features on five different classifiers, i.e., naive Bayes (GNB), logistic regression (LR), neural network (NN), random forest(RF), and support vector machine (SVM). We use the scikit-learn [1] library to implement these classifiers. Considering the importance of class 0 (request rejection) in access control problems, results on both class 0 and the macro average on class 1 and class 0 are reported in Table 4.4. The results in Table 4.4 are conducted on a balanced dataset consisting of all negative samples of the original Amazon dataset introduced in Section 4.4.1 and the same number of positive samples randomly selected from the original dataset. Both the negative and positive samples keep the same order as the original dataset.

Although accuracy (Acc) is the most-used metric for evaluating classification models, it only works on balanced datasets. For severe imbalanced datasets, the results of accuracy (Acc) can be misleading and unreliable. Since the F1 score is an evaluation metrics combining two competing metrics, i.e. precision (Pre) and recall (Rec), we mainly discuss the F1 score when comparing the performance of topological and nontopological features. Δ F1 is the growth rate between the F1 score achieved on topological and nontopological features, calculated by Equation (4.8).

$$\Delta \text{F1} = \frac{F1_{\text{Topo}} - F1_{\text{Nontopo}}}{F1_{\text{Nontopo}}} \times 100\%, (t > 0). \tag{4.8}$$

As shown in Table 4.4, RF classifier achieves the best performance on all metrics with topological features, highlighted with bold fonts. Using topological features extracted from the access control knowledge graph increases the F1 score on class 0 from 70.08% to 73.51%, which achieves an increase of 4.89%. Actually, an improvement of 4.21% also achieved on macro average F1 score by using topological features.

The performance on NN and LR classifiers are also boosted on both macro average and class 0 with topological features. However, for the GNB classifier, the macro average F1 score is improved from 45.43% to 60.43% with a cost of the decrease of F1 score of class 0 from 66.49% to 58.59%. It means that topological

[1] https://scikit-learn.org/stable/

features increase the performance on class 1 but decrease on class 0 when using the GNB classifier. By contrast, the SVM classifier increases the F1 score of class 0 from 59.92% to 66.69% but the macro average f1 score decreases from 61.04% to 35.21%. Generally speaking, it is fair to say that the topological feature can improve access control performance in the offline learning scenario.

Table 4.4: Performance comparison of different classifiers on offline scenario

Classifier	Feature	Acc(%)	Macro average				Class 0			
			Pre(%)	Rec(%)	F1(%)	Δ F1	Pre(%)	Rec(%)	F1(%)	Δ F1
GNB	Nontopo	53.56	57.10	53.02	45.43		52.43	90.87	66.49	
	Topo	60.51	60.71	60.59	60.43	↑33.01%	62.57	55.09	58.59	↓11.89%
LR	Nontopo	61.04	61.14	61.09	61.02		62.61	57.55	59.97	
	Topo	62.89	63.08	62.96	62.83	↑2.97%	65.05	57.97	61.31	↑2.23%
NN	Nontopo	60.63	61.07	60.75	60.38		63.68	52.04	57.27	
	Topo	61.46	61.65	61.53	61.39	↑1.66%	63.51	56.39	59.74	↑4.31%
RF	Nontopo	70.65	70.74	70.69	70.64		72.52	67.81	70.08	
	Topo	**73.61**	**73.64**	**73.63**	**73.61**	**↑4.21%**	**74.86**	**72.22**	**73.51**	**↑4.89%**
SVM	Nontopo	61.07	61.18	61.13	61.04		62.69	57.38	59.92	
	Topo	50.51	47.95	49.83	35.21	↓42.31%	50.62	97.71	66.69	↑11.31%

To further verify the improvement effectiveness of topology features in different data imbalance statuses, we use an RF classifier, which performs the best in Table 4.4, to conduct experiments on different class proportions in an offline scenario, as shown in Table 4.5. Consistent with Table 4.4, topological features can improve the access control performance of both macro average and the minority class (class 0) on different degrees of imbalanced datasets. However, with the increase of data imbalance, the performance of the algorithm gradually deteriorates, but the results are still much better than a random decision. Specifically, topological features improve the macro average f1 score by 4.60%, 1.30% and 1.29% respectively when the class 0 accounts for 30%, 10%, 1.59% (the original dataset) in the dataset. Furthermore, topological features are superior in improving the performance on class 0, which records an increase of 10.30%, 5.28% and 33.83% accordingly.

Table 4.5: Offline learning performance comparison on different data imbalance statuses

Class 0	Feature	Acc(%)	Macro average				Class 0			
			Pre(%)	Rec(%)	F1(%)	Δ F1	Pre(%)	Rec(%)	F1(%)	Δ F1
30%	Nontopo	76.87	73.72	66.87	68.41		68.75	41.90	52.07	
	Topo	78.57	75.76	69.89	71.56	↑4.60%	71.02	48.21	57.43	↑10.30%
10%	Nontopo	89.41	67.70	59.16	61.49		43.63	21.38	28.69	
	Topo	89.56	68.58	59.82	62.28	↑1.30%	45.26	22.67	30.21	↑5.28%
1.59%	Nontopo	98.01	53.29	51.08	51.48		8.10	2.63	3.97	
	Topo	97.98	54.24	51.55	52.15	↑1.29%	9.99	3.62	5.32	↑33.83%

4.4.5 Online Learning Performance Comparison

We also conduct online learning experiments on different degrees of imbalance statuses to verify the effectiveness of topology features in improving access control performance. Table 4.6 shows the overall performance comparison results. The time step size is set as 1/1000 of the dataset size. Topological features improve the macro average f1 score by 2.37%, 2.7% and 1.45% respectively when the class 0 accounts for 30%, 10%, 1.59% (the original dataset) in the dataset. In particular, topological features are superior in improving the performance on class 0, which records an increase of 7.28%, 17.10% and 24.31% accordingly.

Table 4.6: Overall performance comparison of online learning

Class 0	Feature	Acc(%)	Macro average				Class 0			
			Pre(%)	Rec(%)	F1(%)	Δ F1	Pre(%)	Rec(%)	F1(%)	Δ F1
30%	Nontopo	73.41	67.72	61.06	61.78		59.51	30.94	40.71	
	Topo	73.68	67.97	62.31	63.25	↑2.37%	59.26	34.58	43.68	↑7.28%
10%	Nontopo	90.50	74.46	55.09	56.76		57.80	11.06	18.57	
	Topo	90.33	71.83	56.18	58.29	↑2.70%	52.32	13.72	21.74	↑17.10%
1.59%	Nontopo	98.39	64.77	51.70	52.76		31.03	3.53	6.33	
	Topo	98.38	65.14	52.17	53.53	↑1.45%	31.75	4.49	7.87	↑24.31%

Fig. 4.4 shows the real-time macro average performance comparison of online learning. The red lines show access control performance comparison when using topological and nontopological features on a dataset with class 0: class 1 = 3:7. Similarly, the brown lines and blue lines present the results of the original dataset and a dataset with class 0: class 1 = 1:9. Fig. 4.4 demonstrates that topological features can improve the overall f1 score without decreasing the accuracy.

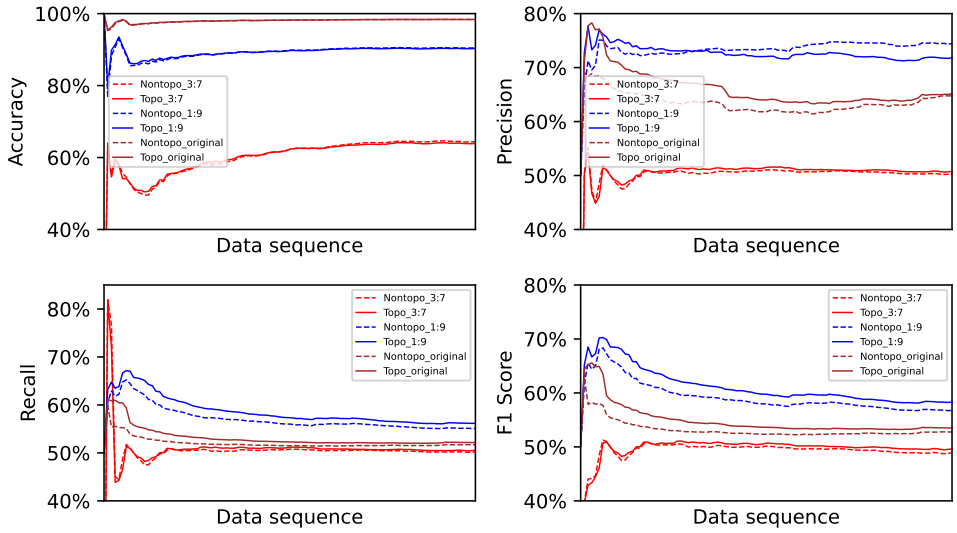

Fig. 4.4. The real-time macro average performance comparison of online learning

Similarly, Fig. 4.5 shows the real-time performance comparison of online learning on class 0 (the minority class). Though the trends are the same with Fig. 4.4, the degree of improvements are larger in Fig. 4.5 .

4.4.6 Discussion

Results shown in Tables 4.5 and 4.6 demonstrated the effectiveness of topological features in improving the access control performance in both offline and online scenarios. However, for privacy and security reasons, the Amazon access control dataset only provides 12 categorical user attributes and 2 resource attributes. These attributes use ID numbers to distinguish different values to prevent sensitive data leakage. It is very challenging to achieve high predictive performance

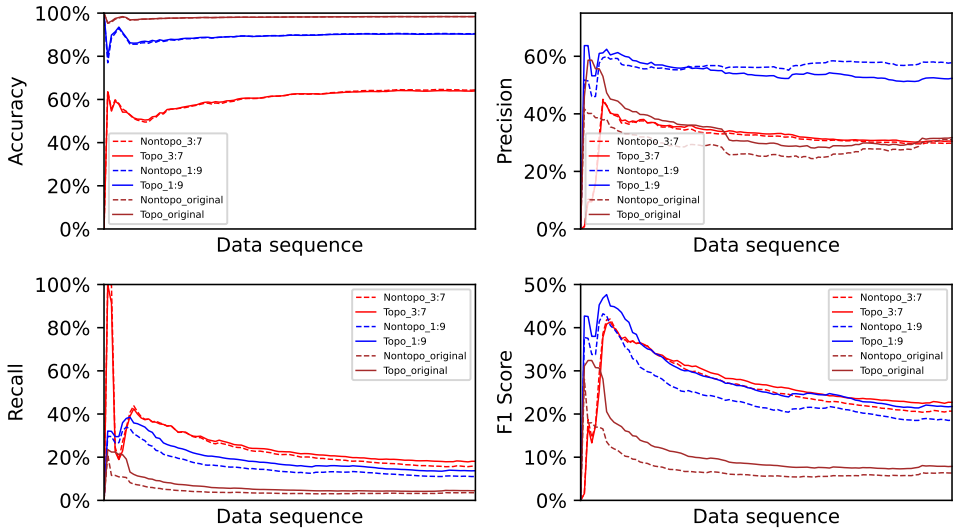

Fig. 4.5. The real-time performance comparison of online learning on class 0

without more text attributes to provide rich semantic information for mining. Therefore, the overall performance and the minority class performance is still unsatisfactory.

In fact, the problem of data insufficiency, especially the lack of attributes information, is common for access control. ABAC rule mining algorithms also suffer from severe overall performance deficiency caused by the poor quality of available real-world access control datasets. For example, the work in [85] proposed an iterative rule mining algorithm, named Rhapsody, to automatically mine ABAC rules from sparse logs and prevent over-permissiveness. They reported the F1 scores of five ABAC rule-based algorithms including Rhapsody on the same Amazon dataset with us. The range of the reported F1 scores is from 0.01 to 0.35, which is equivalent to our method. However, they only choose the top eight most requested resources and their corresponding requests to form eight instances for algorithm evaluation instead of evaluating the algorithm on the whole log file as we do. Therefore, the generalisation performance of their algorithm is not guaranteed.

4.5 Conclusion

To better encode high-cardinality categorical user and resource attributes and improve the machine learning-based access control performance, we proposed a knowledge graph empowered online learning framework for access control decision-making. As a combination of machine learning and knowledge graph, the proposed framework incorporates machine learning algorithms in both online and offline learning modes. It explores latent topological hierarchies and dependencies between users and resources. Through transferring tabular user and resource attributes into a comprehensive knowledge graph, the topological features from the established knowledge graph were extracted to represent uses and resources. Experimental results show that topological features outperform nontopogical features encoded by binary encoding method in both online and offline settings.

Chapter 5

Conclusion and Future Works

This chapter summarises the main work and contributions of this study in Section 5.1 and points out the limitations of this work and some promising research topics for future work in Section 5.2.

5.1 Summary

Data leakage is one of the public's most significant concerns in this era and also a bottleneck that restricts the continued vigorous development of information technology. Motivated by the problem of how to prevent internal personnel data leakage, this thesis locates the critical technology of data protection – Access Control. This research explores machine learning-based access control, one of the technologies that may replace manual access control decisions in the era of big data.

To provide feasible solutions to the concept drift and data imbalance problems existing in MLAC algorithms, this thesis adopts incremental batch learning into access control decision-making and proposes an adaptive machine learning framework to update classifiers periodically in Chapter 3. Furthermore, a boosting window algorithm is designed to specially enhance the performance of the minority class (access deny). The proposed framework and BW algorithm were evaluated on a real-world Amazon access control dataset. Experimental results

showed that the F1 score of the minority class was improved from 6.64% to 10.96% when increasing the sample rate from 0.05 to 0.95.

In Chapter 4, this thesis leverages a knowledge graph to extract topological features from high cardinality categorical attributes to improve the overall performance of both the minority and the majority classes. Experimental results on real-world Amazon access control dataset demonstrated that using topological features extracted from knowledge graph significantly improved the F1 score the minority class and the macro average results in both online and offline learning modes at different imbalance statuses.

This research is significant to academia, the industry and the public:

- This research is a pioneer study to use machine learning methods in the Access control decision-making field. This research provides a more accurate and effective access-control decision-making mechanism for open distributed information systems. Additionally, feasible solutions are also offered for the problems of data imbalance and concept drifts.

- The novel framework proposed in this research can help the industry develop a new access control system to overcome the challenge caused by policy misconfigurations, massive policy scales, and enormous access-control entities. In this way, it can help the industry significantly improve decision-making efficiency, reduce human's working intensity, and save the cost of system management.

- This work is also an effective response to the public's concern about data security as it helps reduce the risk of a data breach.

5.2 Future Work

Although the efforts made in this thesis, the overall performance is still not satisfactory. One of the main reasons is that the open-source datasets applied in this study are desensitised. All textual attributes of users and resources have been encoded as integers. Therefore, no meaningful semantic features can be extracted by advanced deep learning techniques.

Therefore, this study can be further extended and improved from the following aspects in the future.

(1) Build a standard open-source large scale practical access control dataset with rich original user and resource attributes. As one of the three pillars of AI technology development, high-quality data is essential for machine learning and deep learning algorithms to achieve acceptable performance. However, few organisations are willing to disclose their data because of the importance and sensitivity of access control data to information systems. There should be some way in the future to find a balance between keeping data secure and providing more possibilities for researchers. In the same way that Imagenet [102, 103] has significantly advanced computer vision, we believe that a large scale standard access control dataset with rich user and resource attributes information can also lead to the flourishing and advancement of the MLAC field.

(2) Apply graph embedding techniques to extract more abstract interlinked relationships between user and resource attributes. Chapter 4 just get the feet wet on leveraging graph connectivity features to improve the performance of machine learning models. The results are very promising in both online and offline scenarios. In future, more deep learning-based graph embedding algorithms, such as Deepwalk [104], node2vec [105], Graph Convolutional Networks (GCN) [106], GraphSAGE [107] and Graph Attention Networks (GAT) [108], can be explored to extract high-level entity and relationship features to further improve the access control performance.

List of References

[1] Y.-F. Ge, M. Orlowska, J. Cao, H. Wang, and Y. Zhang, "Mdde: multitasking distributed differential evolution for privacy-preserving database fragmentation," *The VLDB Journal*, pp. 1–19, 2022.

[2] R. U. Rasool, U. Ashraf, K. Ahmed, H. Wang, W. Rafique, and Z. Anwar, "Cyberpulse: a machine learning based link flooding attack mitigation system for software defined networks," *IEEE Access*, vol. 7, pp. 34 885–34 899, 2019.

[3] McAfee, "Grand Theft Data II : The Drivers and Shifting State of Data Breaches," McAfee, Tech. Rep., 2019. [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/reports/restricted/rp-data-exfiltration-2.pdf

[4] Lonergan, *Australian Community Attitudes to Privacy Survey 2020*. Office of the Australian Information Commissioner, 2020. [Online]. Available: https://www.oaic.gov.au/assets/engage-with-us/research/acaps-2020/Australian-Community-Attitudes-to-Privacy-Survey-2020.pdf

[5] V. C. Hu, D. F. Ferraiolo, R. Chandramouli, and D. R. Kuhn, *Attribute-Based Access Control*. Artech House, 2017.

[6] Y.-F. Ge, M. Orlowska, J. Cao, H. Wang, and Y. Zhang, "Knowledge transfer-based distributed differential evolution for dynamic database fragmentation," *Knowledge-Based Systems*, vol. 229, p. 107325, 2021.

[7] F. Khalil, H. Wang, and J. Li, "Integrating markov model with clustering for predicting web page accesses," in *Proceeding of the 13th Australasian world wide web conference (AusWeb07)*. AusWeb, 2007, pp. 63–74.

[8] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2018.

[9] J. Gama, R. Sebastiao, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine learning*, vol. 90, no. 3, pp. 317–346, 2013.

[10] A. Liu, "Concept drift adaptation for learning with streaming data," Ph.D. dissertation, 2018.

[11] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Brazilian symposium on artificial intelligence*. Springer, 2004, pp. 286–295.

[12] M. Baena-García, J. Campo-Ávila, R. Fidalgo-Merino, A. Bifet, R. Gavald, and R. Morales-Bueno, "Early drift detection method," 01 2006.

[13] E. Cohen and M. Strauss, "Maintaining time-decaying stream aggregates," in *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2003, pp. 223–233.

[14] A. Tsymbal, M. Pechenizkiy, P. Cunningham, and S. Puuronen, "Dynamic integration of classifiers for handling concept drift," *Information fusion*, vol. 9, no. 1, pp. 56–68, 2008.

[15] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 377–382.

[16] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[17] M. Tang, J. Yin, M. Alazab, J. Cao, and Y. Luo, "Modeling of extreme vulnerability disclosure in smart city industrial environments," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4150–4158, 2020.

[18] J. Yin, M. Tang, J. Cao, H. Wang, M. You, and Y. Lin, "Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning," *World Wide Web*, pp. 1–23, 2021.

[19] F. Cirillo, *The Pomodoro Technique*. New York, New York, USA: Crown Publishing Group, 2018.

[20] Y. Yang and Z. Xu, "Rethinking the value of labels for improving class-imbalanced learning," *arXiv preprint arXiv:2006.07529*, 2020.

[21] B. Wang and J. Pineau, "Online bagging and boosting for imbalanced data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3353–3366, 2016.

[22] F. Ullah, M. Edwards, R. Ramdhany, R. Chitchyan, M. A. Babar, and A. Rashid, "Data exfiltration: A review of external attack vectors and countermeasures," *Journal of Network and Computer Applications*, vol. 101, pp. 18–54, 2018.

[23] Y.-F. Ge, J. Cao, H. Wang, J. Yin, W.-J. Yu, Z.-H. Zhan, and J. Zhang, "A benefit-driven genetic algorithm for balancing privacy and utility in database fragmentation," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019, pp. 771–776.

[24] F. Paci, A. Squicciarini, and N. Zannone, "Survey on access control for community-centered collaborative systems," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–38, 2018.

[25] D. Servos and S. L. Osborn, "Current research and open problems in attribute-based access control," *ACM Computing Surveys (CSUR)*, vol. 49, no. 4, pp. 1–45, 2017.

[26] Y.-F. Ge, J. Cao, H. Wang, Y. Zhang, and Z. Chen, "Distributed differential evolution for anonymity-driven vertical fragmentation in outsourced data storage," in *International Conference on Web Information Systems Engineering*. Springer, 2020, pp. 213–226.

[27] H. Wang, J. Cao, and Y. Zhang, "Ticket-based service access scheme for mobile users," in *Proceedings of the twenty-fifth Australasian conference on Computer science-Volume 4*, 2002, pp. 285–292.

[28] H. Wang and L. Sun, "Trust-involved access control in collaborative open social networks," in *2010 fourth international conference on network and system security*. IEEE, 2010, pp. 239–246.

[29] H. Wang, L. Sun, and E. Bertino, "Building access control policy model for privacy preserving and testing policy conflicting problems," *Journal of Computer and System Sciences*, vol. 80, no. 8, pp. 1493–1503, 2014.

[30] Y.-F. Ge, J. Cao, H. Wang, Z. Chen, and Y. Zhang, "Set-based adaptive distributed differential evolution for anonymity-driven database fragmentation," *Data Science and Engineering*, vol. 6, no. 4, pp. 380–391, 2021.

[31] Y.-F. Ge, W.-J. Yu, J. Cao, H. Wang, Z.-H. Zhan, Y. Zhang, and J. Zhang, "Distributed memetic algorithm for outsourced database fragmentation," *IEEE Transactions on Cybernetics*, vol. 51, no. 10, pp. 4808–4821, 2020.

[32] J. Yin, M. You, J. Cao, H. Wang, M. Tang, and Y.-F. Ge, "Data-driven hierarchical neural network modeling for high-pressure feedwater heater group," in *Australasian Database Conference*. Springer, 2020, pp. 225–233.

[33] W. Wang, W. Wang, and J. Yin, "A bilateral filtering based ringing elimination approach for motion-blurred restoration image," *Current Optics and Photonics*, vol. 4, no. 3, pp. 200–209, 2020.

[34] A. M. Alvi, S. Siuly, and H. Wang, "Developing a deep learning based approach for anomalies detection from eeg data," in *International Conference on Web Information Systems Engineering*. Springer, 2021, pp. 591–602.

[35] L. Xiao, Y. Xue, H. Wang, X. Hu, D. Gu, and Y. Zhu, "Exploring fine-grained syntactic information for aspect-based sentiment classification with dual graph neural networks," *Neurocomputing*, vol. 471, pp. 48–59, 2022.

[36] M. Tawhid, N. Ahad, S. Siuly, K. Wang, and H. Wang, "Data mining based artificial intelligent technique for identifying abnormalities from brain signal data," in *International Conference on Web Information Systems Engineering*. Springer, 2021, pp. 198–206.

[37] R. U. Rasool, K. Ahmed, Z. Anwar, H. Wang, U. Ashraf, and W. Rafique, "Cyberpulse++: A machine learning-based security framework for detecting link flooding attacks in software defined networks," *International Journal of Intelligent Systems*, vol. 36, no. 8, pp. 3852–3879, 2021.

[38] S. Subramani, H. Wang, H. Q. Vu, and G. Li, "Domestic violence crisis identification from facebook posts based on deep learning," *IEEE access*, vol. 6, pp. 54 075–54 085, 2018.

[39] H. Wang, Y. Wang, T. Taleb, and X. Jiang, "Special issue on security and privacy in network computing," *World Wide Web*, vol. 23, no. 2, pp. 951–957, 2020.

[40] A. Liu, X. Du, and N. Wang, "Efficient access control permission decision engine based on machine learning," *Security and Communication Networks*, vol. 2021, 2021.

[41] A. A. Jabal, E. Bertino, J. Lobo, D. Verma, S. Calo, and A. Russo, "Flap–a federated learning framework for attribute-based access control policies," *arXiv preprint arXiv:2010.09767*, 2020.

[42] A. Outchakoucht, A. Abou El Kalam, H. Es-Samaali, and S. Benhadou, "Machine learning based access control framework for the internet of things," *Machine Learning*, vol. 11, no. 2, 2020.

[43] L. Sun, J. Ma, H. Wang, Y. Zhang, and J. Yong, "Cloud service description model: an extension of usdl for cloud services," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 354–368, 2015.

[44] K. Cheng, L. Wang, Y. Shen, H. Wang, Y. Wang, X. Jiang, and H. Zhong, "Secure k k-nn query on encrypted cloud data with multiple keys," *IEEE Transactions on Big Data*, vol. 7, no. 4, pp. 689–702, 2017.

[45] J. Yin, M. Tang, J. Cao, and H. Wang, "Apply transfer learning to cybersecurity: Predicting exploitability of vulnerabilities by description," *Knowledge-Based Systems*, vol. 210, p. 106529, 2020.

[46] J. Yin, M. Tang, J. Cao, H. Wang, and M. You, "A real-time dynamic concept adaptive learning algorithm for exploitability prediction," *Neurocomputing*, vol. 472, pp. 252–265, 2022.

[47] J. Yin, J. Cao, S. Siuly, and H. Wang, "An integrated mci detection framework based on spectral-temporal analysis," *International Journal of Automation and Computing*, vol. 16, no. 6, pp. 786–799, 2019.

[48] Verizon, "Data breach investigations report," Verizon, Tech. Rep., 2020. [Online]. Available: https://enterprise.verizon.com/resources/reports/2020-data-breach-investigations-report.pdf

[49] P. Vimalachandran, H. Liu, Y. Lin, K. Ji, H. Wang, and Y. Zhang, "Improving accessibility of the australian my health records while preserving privacy and security of the system," *Health Information Science and Systems*, vol. 8, 10 2020.

[50] H. Wang, J. Cao, and Y. Zhang, "A flexible payment scheme and its role-based access control," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, pp. 425– 436, 04 2005.

[51] H. Wang, J. Cao, and Y. Zhang, "Ticket-based service access scheme for mobile users," *Australian Computer Science Communications*, pp. 285–292, 02 2002.

[52] X. Sun, H. Wang, and A. Plank, "An efficient hash-based algorithm for minimal k-anonymity," *Proc Thirty-First Aust Conf Comp Sci*, vol. 74, pp. 101–107, 01 2008.

[53] H. Wang, Y. Zhang, and J. Cao, "Effective collaboration with information sharing in virtual universities," *IEEE Trans. Knowl. Data Eng.*, vol. 21, pp. 840–853, 06 2009.

[54] X. Sun, H. Wang, J. Li, and J. Pei, "Publishing anonymous survey rating data," *Data Min. Knowl. Discov.*, vol. 23, pp. 379–406, 11 2011.

[55] H. Wang, Y. Wang, T. Taleb, and X. Jiang, "Editorial: Special issue on security and privacy in network computing," *World Wide Web*, vol. 23, 07 2019.

[56] E. Kabir, A. Mahmood, H. Wang, and A. Mustafa, "Microaggregation sorting framework for k-anonymity statistical disclosure control in cloud computing," *IEEE Transactions on Cloud Computing*, vol. PP, pp. 1–1, 08 2015.

[57] F. Zhang, Y. Wang, S. Liu, and H. Wang, "Decision-based evasion attacks on tree ensemble classifiers," *World Wide Web*, vol. 23, 09 2020.

[58] H. Wang and L. Sun, "Trust-involved access control in collaborative open social networks," in *2010 fourth international conference on network and system security*. IEEE, 09 2010, pp. 239–246.

[59] R. S. Sandhu, "Role-based access control," in *Advances in computers*. Elsevier, 1998, vol. 46, pp. 237–286.

[60] E. Bertino, P. A. Bonatti, and E. Ferrari, "Trbac: A temporal role-based access control model," in *Proceedings of the fifth ACM workshop on Role-based access control*, 2000, pp. 21–30.

[61] M. J. Moyer and M. Abamad, "Generalized role-based access control," in *Proceedings 21st International Conference on Distributed Computing Systems*. IEEE, 2001, pp. 391–398.

[62] H. Wang, L. Sun, and E. Bertino, "Building access control policy model for privacy preserving and testing policy conflicting problems," *Journal of Computer and System Sciences*, vol. 80, 12 2014.

[63] J. Zhang, H. Li, X. Liu, Y. Luo, F. Chen, and H. Wang, "On efficient and robust anonymization for privacy protection on massive streaming categorical information," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, pp. 1–1, 09 2015.

[64] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, "Attribute-based access control," *Computer*, vol. 48, no. 2, pp. 85–88, 2015.

[65] M. Gupta, F. M. Awaysheh, J. Benson, M. Al Azab, F. Patwa, and R. Sandhu, "An attribute-based access control for cloud-enabled industrial smart vehicles," *IEEE Transactions on Industrial Informatics*, 2020.

[66] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2130–2145, 2018.

[67] S. Ding, J. Cao, C. Li, K. Fan, and H. Li, "A novel attribute-based access control scheme using blockchain for iot," *IEEE Access*, vol. 7, pp. 38 431–38 441, 2019.

[68] H. Zhong, Y. Zhou, Q. Zhang, Y. Xu, and J. Cui, "An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare," *Future Generation Computer Systems*, vol. 115, pp. 486–496, 2021.

[69] J. Li and B. Zhang, "An ontology-based approach to improve access policy administration of attribute-based access control," *International Journal of Information and Computer Security*, vol. 11, no. 4-5, pp. 391–412, 2019.

[70] S. Dutta, S. S. L. Chukkapalli, M. Sulgekar, S. Krithivasan, P. K. Das, and A. Joshi, "Context sensitive access control in smart home environments," in *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*. IEEE, 2020, pp. 35–41.

[71] K. Srivastava and N. Shekokar, "Machine learning based risk-adaptive access control system to identify genuineness of the requester," in *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough*. Springer, 2020, pp. 129–143.

[72] J. He, J. Rong, L. Sun, H. Wang, Y. Zhang, and J. Ma, "A framework for cardiac arrhythmia detection from iot-based ecgs," *World Wide Web*, vol. 23, 09 2020.

[73] H. Li, Y. Wang, H. Wang, and B. Zhou, "Multi-window based ensemble learning for classification of imbalanced streaming data," *World Wide Web*, vol. 20, pp. 1–19, 11 2017.

[74] H. Jiang, R. Zhou, L. Zhang, H. Wang, and Y. Zhang, "Sentence level topic models for associated topics extraction," *World Wide Web*, vol. 22, 11 2019.

[75] J. Yin, M. Tang, J. Cao, H. Wang, M. You, and Y. Lin, "Adaptive online learning for vulnerability exploitation time prediction," in *International Conference on Web Information Systems Engineering*. Springer, 2020, pp. 252–266.

[76] M. Miwa and S. Ananiadou, "Adaptable, high recall, event extraction system with minimal configuration," *BMC bioinformatics*, vol. 16, no. 10, pp. 1–11, 2015.

[77] J. Yin, M. Tang, J. Cao, H. Wang, M. You, and Y. Lin, "Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning," *Word Wide Web*, vol. 1, no. 1, pp. 1–23, 2021. [Online]. Available: https://doi.org/10.1007/s11280-021-00909-z

[78] R. U. Rasool, U. Ashraf, K. Ahmed, H. Wang, W. Rafique, and Z. Anwar, "Cyberpulse: A machine learning based link flooding attack mitigation system for software defined networks," *IEEE Access*, vol. 7, pp. 34 885–34 899, 2019.

[79] H. Wang and L. Sun, "Trust-involved access control in collaborative open social networks," *2010 Fourth International Conference on Network and System Security*, pp. 239–246, 2010.

[80] Z.-G. Chen, Z. hui Zhan, H. Wang, and J. Zhang, "Distributed individuals for multiple peaks: A novel differential evolution for multimodal optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, pp. 708–719, 2020.

[81] K. Cheng, L. Wang, Y. Shen, H. Wang, Y. Wang, X. Jiang, and H. Zhong, "Secure kk-nn query on encrypted cloud data with multiple keys," *IEEE Trans. Big Data*, vol. 7, pp. 689–702, 2021.

[82] J. Zhang, H. Li, X. Liu, Y. Luo, F. Chen, H. Wang, and L. Chang, "On efficient and robust anonymization for privacy protection on massive streaming categorical information," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, pp. 507–520, 2017.

[83] H. Wang, L. Sun, and E. Bertino, "Building access control policy model for privacy preserving and testing policy conflicting problems," *J. Comput. Syst. Sci.*, vol. 80, pp. 1493–1503, 2014.

[84] H. Jiang, R. Zhou, L. Zhang, H. Wang, and Y. Zhang, "Sentence level topic models for associated topics extraction," *World Wide Web*, vol. 22, pp. 2545–2560, 11 2019.

[85] C. Cotrini, T. Weghorn, and D. Basin, "Mining abac rules from sparse logs," in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2018, pp. 31–46.

[86] F. Khalil, H. Wang, and J. Li, "Integrating markov model with clustering for predicting web page accesses," in *Proceeding of the 13th Australasian world wide web conference (AusWeb07)*. AusWeb, 2007, pp. 63–74.

[87] W. Liu, Y. jiao Gong, W. neng Chen, Z. Liu, H. Wang, and J. Zhang, "Coordinated charging scheduling of electric vehicles: A mixed-variable

differential evolution approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 5094–5109, 2020.

[88] H. Hu, J. Li, H. Wang, and G. Daggard, "Combined gene selection methods for microarray data analysis," in *Knowledge-Based Intelligent Information and Engineering Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 976–983.

[89] M. You, J. Yin, H. Wang, J. Cao, and Y. Miao, "A minority class boosted framework for adaptive access control decision-making," in *International Conference on Web Information Systems Engineering*. Springer, 2021, pp. 143–157.

[90] J. Yin, M. Tang, J. Cao, H. Wang, and M. You, "A real-time dynamic concept adaptive learning algorithm for exploitability prediction," *Neurocomputing*, 2021.

[91] S. Daminelli, J. M. Thomas, C. Durán, and C. V. Cannistraci, "Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks," *New Journal of Physics*, vol. 17, no. 11, p. 113037, 2015.

[92] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social networks*, vol. 25, no. 3, pp. 211–230, 2003.

[93] A.-L. Barabási, R. Albert, and H. Jeong, "Scale-free characteristics of random networks: the topology of the world-wide web," *Physica A: statistical mechanics and its applications*, vol. 281, no. 1-4, pp. 69–77, 2000.

[94] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *The European Physical Journal B*, vol. 71, no. 4, pp. 623–630, 2009.

[95] L. Dong, Y. Li, H. Yin, H. Le, and M. Rui, "The algorithm of link prediction on social network," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[96] K. Abbas, A. Abbasi, S. Dong, L. Niu, L. Yu, B. Chen, S.-M. Cai, and Q. Hasan, "Application of network link prediction in drug discovery," *BMC bioinformatics*, vol. 22, no. 1, pp. 1–21, 2021.

[97] P. Srilatha and R. Manjula, "Similarity index based link prediction algorithms in social networks: A survey," *Journal of Telecommunications and Information Technology*, 2016.

[98] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[99] J. Li and P. Willett, "Articlerank: a pagerank-based alternative to numbers of citations for analysing citation networks," in *Aslib Proceedings*. Emerald Group Publishing Limited, 2009.

[100] U. Brandes and C. Pich, "Centrality estimation in large networks," *International Journal of Bifurcation and Chaos*, vol. 17, no. 07, pp. 2303–2318, 2007.

[101] M. Marchiori and V. Latora, "Harmony in the small-world," *Physica A: Statistical Mechanics and its Applications*, vol. 285, no. 3-4, pp. 539–546, 2000.

[102] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[103] G. Bargshady, X. Zhou, R. C. Deo, J. Soar, F. Whittaker, and H. Wang, "Enhanced deep learning algorithm development to detect pain intensity from facial expression images," *Expert Systems with Applications*, vol. 149, p. 113305, 2020.

[104] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.

[105] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[106] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[107] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1025–1035.

[108] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.