M.Sc. Engg. (CSE) Thesis

Detecting Fake Co-visitation Attack in Recommendation Systems

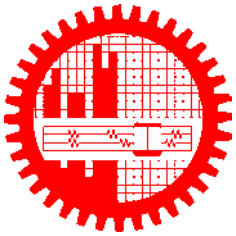
Submitted by

Tropa Mahmood

1014052062

Supervised by

Dr. Muhammad Abdullah Adnan



Submitted to

Department of Computer Science and Engineering

Bangladesh University of Engineering and Technology

Dhaka, Bangladesh

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

December 2021

Candidate's Declaration

I, do, hereby, certify that the work presented in this thesis, titled, "Detecting Fake Co-visitation Attack in Recommendation Systems", is the outcome of the investigation and research carried out by me under the supervision of Dr. Muhammad Abdullah Adnan, Associate Professor, Department of CSE, BUET.

I also declare that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Tropa Mahmood

Tropa Mahmood

1014052062

The thesis titled **"Detecting Fake Co-visitation Attack in Recommendation Systems"**, submitted by Tropa Mahmood, Student ID 1014052062, Session October 2014, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfilment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents on December 22, 2021.

Board of Examiners

1. _____

Dr. Muhammad Abdullah Adnan — Chairman
Associate Professor — (Supervisor)
Department of CSE, BUET, Dhaka

2. _____

Dr. A.K.M. Ashikur Rahman — Member
Professor and Head — (Ex-Officio)
Department of CSE, BUET, Dhaka

3. _____

Dr. Mahmuda Naznin — Member
Professor
Department of CSE, BUET, Dhaka

4. _____

Dr. Md. Shohrab Hossain — Member
Professor
Department of CSE, BUET, Dhaka

5. _____

Dr. Khandker Farid Uddin Ahmed — Member
Professor — (External)
Department of Mathematics
BUET, Dhaka.

Acknowledgement

Foremost, I am thankful to Almighty Allah for his blessings for the successful completion of my thesis. I would like to express my heartiest gratitude, profound indebtedness and deep respect to my supervisor, Dr. Muhammad Abdullah Adnan, Associate Professor, Dept. of CSE, BUET, Dhaka, Bangladesh, for his constant supervision, affectionate guidance and great encouragement and motivation. His keen interest on the topic and valuable advices throughout the study was of great help in completing thesis.

I would also want to thank the members of my thesis committee for their valuable suggestions. I thank Dr. A.K.M. Ashikur Rahman, Dr. Mahmuda Naznin, Dr. Md. Shohrab Hossain and especially the external member Dr. Khandker Farid Uddin Ahmed.

I am especially grateful to Department of Computer Science and Engineering (CSE) of Bangladesh University of Engineering and Technology (BUET) for providing their support during the thesis work. My sincere thanks goes to CSE Office staffs for providing logistic support to me to successfully complete the thesis work.

Finally, I would like to thank my family: my husband Raihan Parvez, parents and all of those who supported me for their appreciable assistance, patience and suggestions during the course of my thesis.

Dhaka
December 22, 2021

Tropa Mahmood
1014052062

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

Recommendation systems are vulnerable to injection attacks by malicious users due to their fundamental openness. One of the vulnerabilities is the fake co-visitation injection attack, which significantly impacts recommendation systems since it modifies the system according to the attacker's wishes. To date, the detection of co-visitation injection attacks are challenging as: (1) the choice of attribute representation of nodes is hard, (2) practical evidence for analyzing and detecting anomalies on real-world data is insufficient, (3) it is challenging to filter between the original and injected co-visitation data in terms of node behaviors. This paper investigates a detection framework that combines attribute and network structure information more synergistically to detect outlier nodes based on CUR decomposition and residual analysis. At first, co-visitation graphs are constructed using association rules, and their nodes attribute representations are developed. Then, both attributes and network structure information are blended in order to identify suspicious nodes. Extensive experiments on both synthetic and real-world data exhibit the efficacy of the proposed detection approach compared with other state-of-the-art approaches. The detection performance can improve by up to 50% for co-visitation injection attacks over the baselines in terms of false alarm rate (FAR) while keeping the highest detection rate (DR).

Chapter 1

Introduction

In this chapter, we describe an an overview of the thesis followed by the motivations and objectives of the research. Then the contributions of the research are briefly outlined. Finally, an overview of the organization of the thesis is provided.

1.1 Overview

Recommendation systems play a crucial role to help users to make informed decision for online purchase or social media contribution. But these systems are vulnerable to fake co-visitation injection attacks due to their open nature. This attack significantly impacts the recommendation systems since it modifies the system according to the attacker's wishes.

A number techniques have been studied to identify fake co-visitation injection attacks in graph-based recommendation systems, and they are continually being refined. Although this detection technique achieves a remarkable outcome in terms of detection rate (DR), but it has a high false alarm rate (FAR).

In a graph-based co-visitation recommendation system, this thesis provides a unified detection approach for protecting against fake co-visitation injection attacks. To detect suspicious nodes, it takes the most influential node properties and combines them with network topology information.Extensive experiments on both synthetic and real-world data exhibit the efficacy of the proposed detection approach compared with other state-of-the-art approaches. The experimental results indicated that our proposed detection approach has a 10.52% lower average false alarm rate (FAR) than the other baselines on the real-world dataset. If we use the largest dataset size, we discover that this disparity can be as wide as 50%.

1.2 Recommendation System

We are in the century of information explosion. An internet user finds an overwhelming number of choices while looking for information about his interests. This rich information base creates a poverty of attention for each item, and the user pays the price for it by getting lost and frustrated on choosing the right option [19]. Recommendation systems play a crucial role in eliminating this challenge in the modern age. Nowadays, they are widely deployed by web services (e.g., YouTube, Amazon, and Google News) and help users make informed decisions for online purchases or social media contributions. The recommendation system uses the user's historical behavior and social data to recommend the relevant data that match the user's preference [20]. Fig.-1.1 and Fig.-1.2 depicts the fundamental framework of a recommendation system and their uses in different web services respectedly. Recommendation system can be formally defined as:

Definition 1.2.1. *A recommendation system (sometimes replacing 'system' with a synonym such as a platform or an engine) is a subclass of information filtering systems that seeks to predict the "rating" or "preference" a user would give to an item [21].*

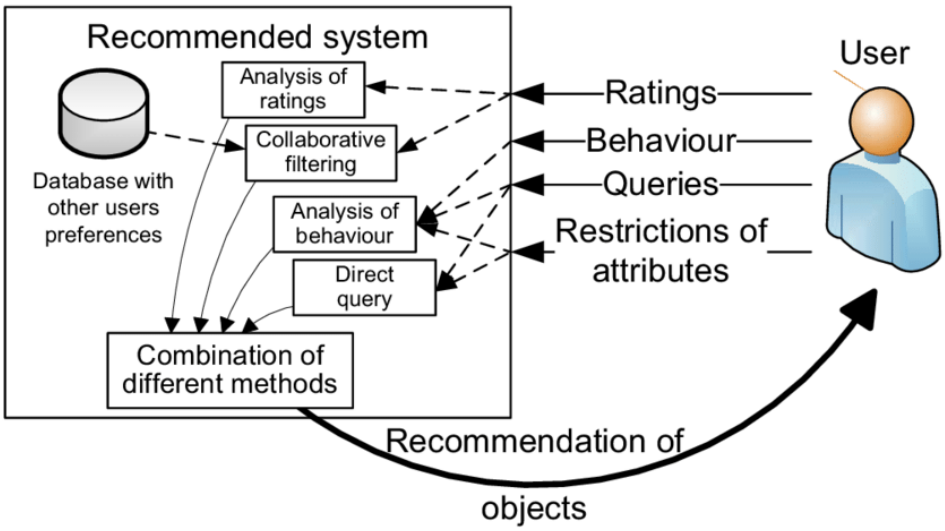


Figure 1.1: Recommendation System [1]

Various types of recommendation algorithms, such as Collaborative Filtering, Content-based, and Hybrid recommendation systems, have been introduced to date. Among them, the most common types of recommendation systems are content-based and collaborative filtering based recommendation systems. Fig.-1.3 depicts the content and collaborative filtering based recommendation system.

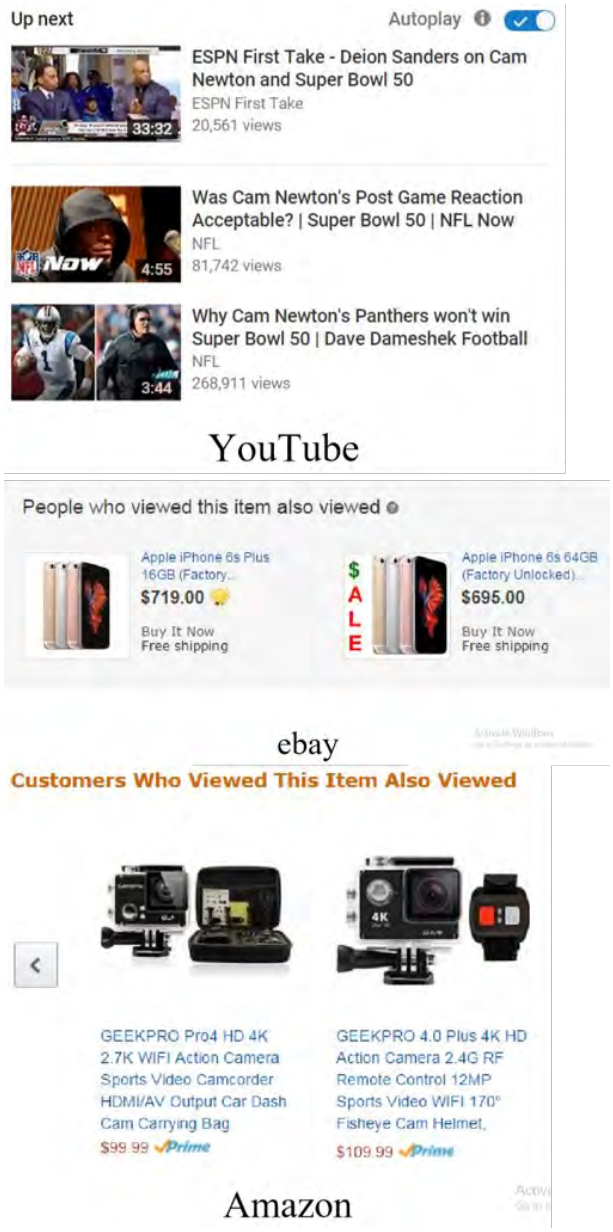


Figure 1.2: Recommendation System in different web services.

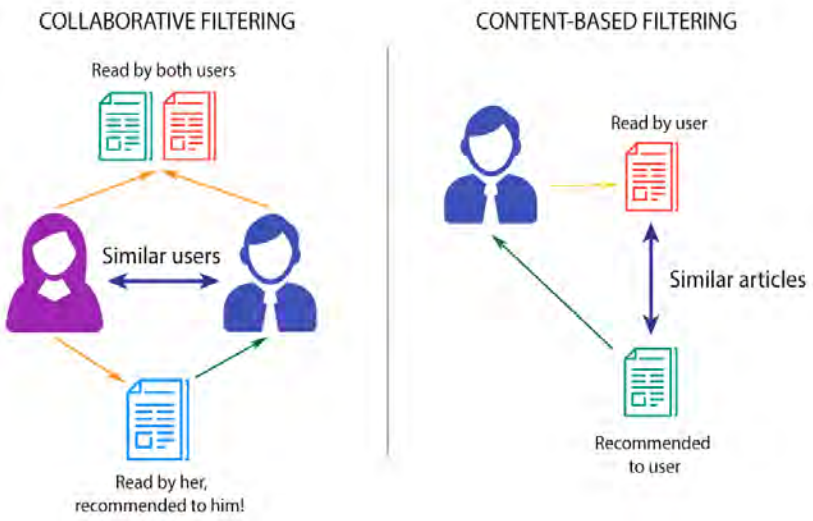


Figure 1.3: Collaborative & Content Based Filtering Recommendation System [2]

1.3 Collaborative Filtering based Recommendation System

Among all types of recommendation systems, Collaborative Filtering based recommendation systems are widely deployed due to their effectiveness and lucidity. This type of recommendation system predicts user preferences for products or services by learning past user-item relationships from a group of users who share the same preferences and taste [22]. Collaborative filtering based recommendation systems can be defined as:

Definition 1.3.1. ***Collaborative filtering Recommendation System:*** *The technique of filtering or evaluating information using the opinions of others is known as collaborative filtering (CF). CF technology brings together the opinions of large interconnected communities on the web, supporting filtering of substantial quantities of data [23].*

In addition to assisting consumers in finding suitable items, they benefit companies producing items or services by raising both selling rate and cross-sales. In such a collaborative filtering based recommendation system, users build profiles by rating certain items, and obtain personalized recommendations for other, unknown items, based on the correlation between their ratings and those of other users. Collaborative recommendation system can be further divided into 2 types: 1) User-Item based RS and 2) Item-Item based RS. In this research we only exploit the Item-Item based RS.

1.3.1 Item-Item based RS

Item-based suggests similar items based on the items users have already liked or positively interacted with. This is also commonly known as “People who viewed this also viewed”

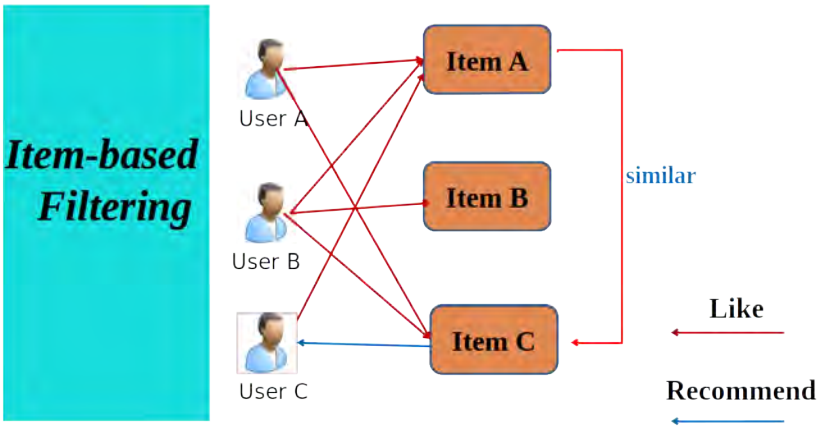


Figure 1.4: Item-Item based Recommendation System.

feature [4]. It looks for similar items based on the items users have already liked or positively interacted with. Instead of matching user-user similarity, item-item CF matches items purchased or rated by a target user to similar items and combines those similar items in a recommendation list.

From fig.-1.4 we can see that user A likes items A and C, user B likes all three items, where user C likes only Item A. Item-based collaborative filtering determines the most similar item of item C to build the recommendation list for user C. Here item A is the most similar item of item C because all the users except user C like item A alongside item C. Therefore, the item-item recommendation system recommends item C to user C.

1.4 Graph-based Recommendation System

Graph-based recommendation systems recently gained massive popularity due to their effectiveness. These recommendation systems use the graph structure to denote the relationship between users-items (user-based) or items-items (item-based). By using the graph, this recommendation system solves the data sparsity problem of a conventional non-graph recommendation system. Fig.-1.5 represents a graph based recommendation system.

In the following subsections, we describe a popular graph representation and graph-based recommendation system used by top-notch web service providers for recommendation purposes.

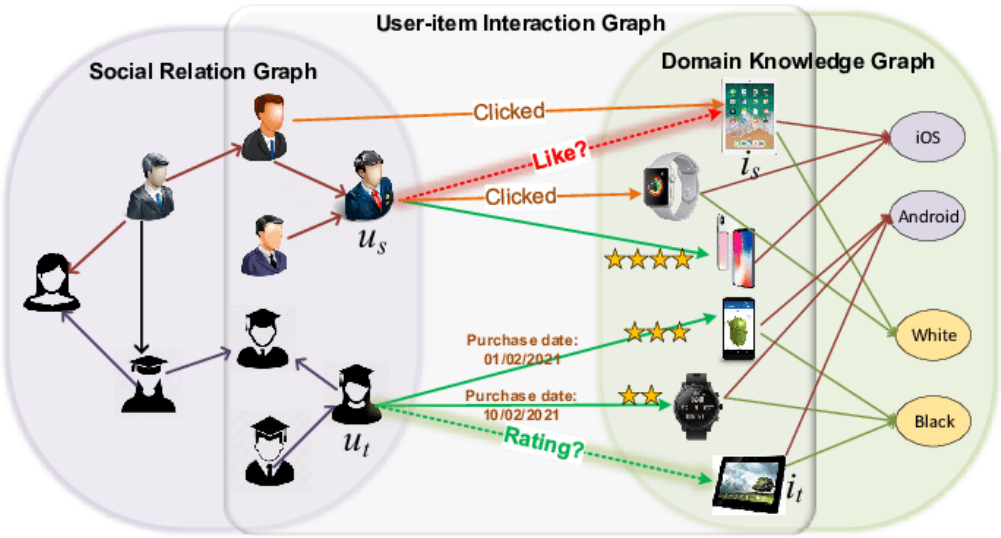


Figure 1.5: The demonstration of graph learning based recommender systems [3].

1.4.1 Co-visitation Graph

A Co-visitation graph is a data structure used as the core component of co-visitation recommendation systems [4]. It uses each item as a node, and if two items are visited by a user within the same session, they are connected by an edge, and their co-visitation score is 1. The co-visitation score will increase with the increasing number of times the two items are co-visited simultaneously. On YouTube, for example, a user is said to have co-visited two videos if they watched one after the other in the same browser session [24]. Fig.-1.6 illustrates an example co-visitation graph.

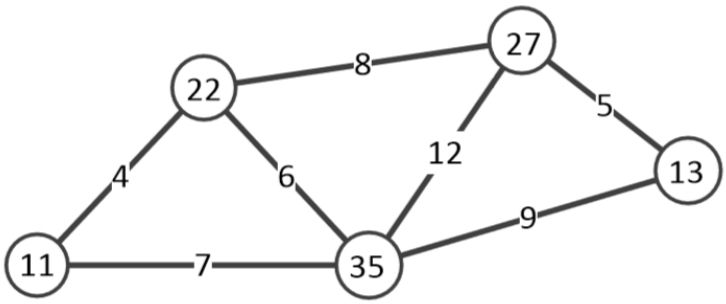


Figure 1.6: Co-visitation graph [4].

Definition 1.4.1. *A co-visitation graph G can be defined as $(\mathbf{V}, \mathbf{E})$, where each node $i \in \boldsymbol{V}$ is an item and edge $e_{ij} \in \boldsymbol{E}$ means that item i and j were co-visited by at least one user. The total number of times items i and j were co-visited is considered as edge weight, w_{ij} and the total*

number of times an item is visited is considered as node weight w_i where,

$$w_i \geq \sum_{j=N(i)} w_{ij} \tag{1.1}$$

where $N(i)$ is the set of neighbors of item i. From equation 1.1, we can see that the node weight of a node i is greater than or equal to the summation of the weights of its edges. This is because users could visit the item i without visiting other items [4].

1.4.2 Co-visitation Recommendation System

One particular type of Collaborative Filtering based recommendation system is called Co-Visitation Recommendation System. It mainly leverages a co-visitation graph to recommend items to a user. The key intuition is that items that were frequently co-visited in the past are likely to be co-visited in the future. Specifically, two popular recommendation tasks are item-item recommendation and user-item recommendation. In an item-to-item recommendation, when a user is visiting an item i, the system recommend the most similar top-k items of i to the user. Intuitively, items i and j are more similar if they are more frequently co-visited; given the number of co-visitations between i and j, they tend to be less similar if they are more popular. Fig.-1.7 represents the primary concept of co-visitation recommendation system. Here we can see that, video A and B are co-visited by many users in the past. So in present, when a new user visits video A, the item-based co-visitation recommendation system analyzes the previous co-visitation history of that item and suggest top-k most similar items to the user. In this case, it will recommend video B.

Figure 1.7: Co-visitation Recommendation System [4].

In a user-to-item recommendation, the system recommends top-k items to a user via considering the user's visiting history. The visiting history could include all items the user has visited if the user logs in the web service, or it could include items the user has visited in a browser session if the user does not log in or the user is an unregistered visitor. The item-item recommendation system and the user-item recommendation system are compared in fig.-1.8. In this research, we only exploit the item-based co-visition recommendation system which is used by many popular

web services such as YouTube to recommend videos according to Google's official report [24], and it is used by Amazon to recommend products according to Amazon's publication [6] due to its effective and simple nature.

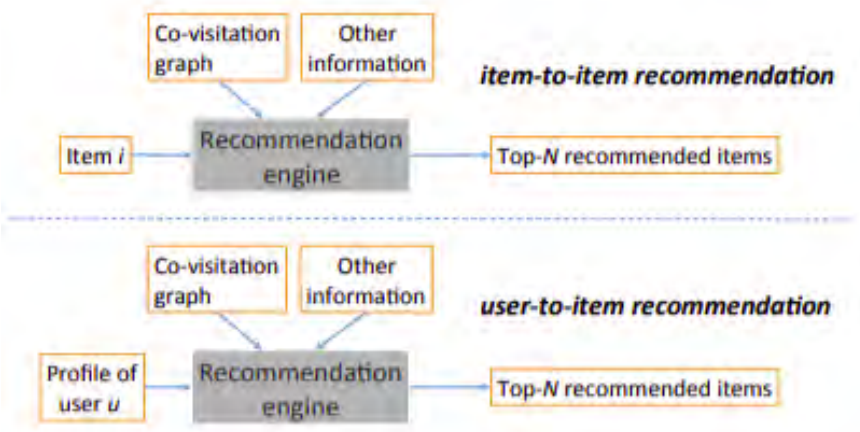


Figure 1.8: Item-item recommendation vs. user-item recommendation [4]

1.4.3 Fake Co-visitation Injection Attack

Unfortunately, these systems are highly vulnerable to fake co-visitation injection attacks due to their openness and fundamental vulnerabilities. Examples of manipulation have been outlined in [4, 25] and include attacks from popular systems like Amazon and eBay [20, 26]. As aforementioned, in addition to helping consumers in finding their desired items, these recommendation system also benefit service providers by raising their selling rate. As a result of this attack, these personalized recommendation services get hampered along with confidence of both customers and businesses on the web services.

What is Fake Co-visitation Injection Attack?

In fake co-visitation injection attack the attacker injects a sufficient number of fake co-visitations to alter the result of recommendation system according to his desire [4]. Attacks against a graph-based co-visitation recommendation system are depicted in fig.-1.9. The attacker first chooses a set of anchor items from the co-visitation graph whose recommendation list he wants to change, then injects fake co-visitation data to promote or demote the target item.

Challenges on Detection Techniques Research

Researches regarding detection techniques of fake co-visitation injection attack faces some significant limitations due to the challenging issues:

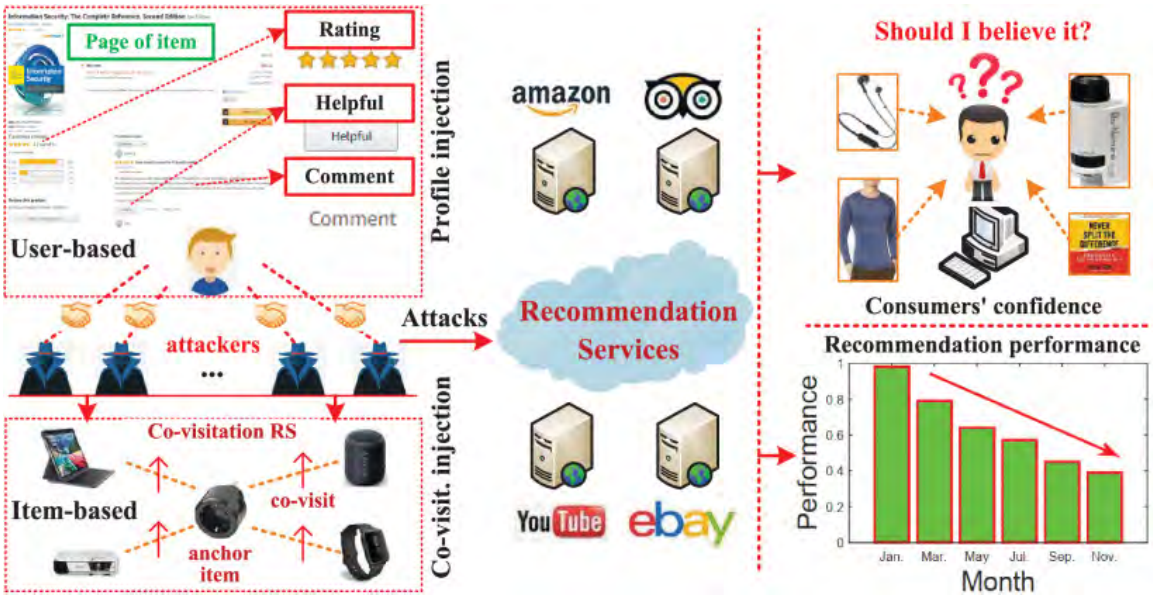


Figure 1.9: Attacks on graph based co-visitation recommendation system [5].

- The unavailablity of real-world co-visitation attacked dataset.
- The choice of attribute representation of nodes is hard
- It is challenging to filter between the original and injected co-visitation data in terms of node behaviors.

1.5 Motivation

Attacks on recommender systems can affect the quality of the prediction for many users, decreasing overall user satisfaction with the system. Such threats may cost users time and money and pose a severe challenge to the recommender system administrators, who have to discover the attackers manually. This vulnerability of recommender systems is even more severe if we think it extends a personalized information system in which an attacker can introduce fake co-visitations to increase the general interest for a set of target resources. Therefore, the demand for protecting users' benefits is becoming a burning issue.

The Need for an detection approach compatible for real-world recommendation system: The primary target of the detection technique is to be compatible for real-world recommendation system. But the real-world co-visitation attack dataset is unavailable. So the existing researches [5, 7, 8] regarding detection techniques of this attack used random graph model to generate co-visitation graph. So it is worth investigating to check how the detection methods perform on real-world dataset by running experiments and construct the co-visitation graph from existing real-world rating dataset by taking the rating timestamp into consideration.

The Need for an detection approach which select the most influential node attributes: To make the system robust to both both small and large scale fake co-visitation injection attacks, it is required to eliminate the noisy and structurally irrelevant attributes of the nodes. These disturbed attributes can decrease the overall detection performance by mistakenly detecting a few genuine items as target items. The existing researches may lack in this point. In paper [7], the disturbed attributes are not removed which results a large false alarm rate (FAR). On the other hand, in paper [5], the author eliminates disturbed data in the early stage of the detection technique. As a result, it brings adverse effects in terms of structured behavior representations.

The Need for an detection approach with small time complexity: The real-world co-visitation graph is highly complex and rich with many redundant data. The detection approach needs to be efficient enough to run on such a big dataset. So it is worth investigating how to improve the overall performance in the most optimized way.

1.6 Thesis Objectives

Our primary objective is to propose a unified detection model which will perform well above the existing baseline methods both in terms of detection rate (DR) and false alarm rate (FR). While analyzing current detection approaches for fake co-visitation injection attack in graph based recommendation system, found them still in the phase of improving its efficiency [4,5,7,8,27,28]. In summary, our proposed methodology needs to meet below requirements.

Most influential attribute selection: Node attributes of the graph need to be defined in such manner, that preserves the structure of the graph and connection between individual nodes. It plays a vital role in outlier detection as the suspicious nodes differ significantly from the standard behavioral features of a benign node. But there may exist noisy and structurally irrelevant attributes which may impede to accurately detect anomalies and may even lead to large false alarm rate. So it is important to determine most influential attributes of the nodes to accurately detect anomalies and improve the overall performance.

High Detection Rate (DR): It is mandatory to have a high detection rate while detecting suspicious nodes as this is the primary parameter to measure the success of the detection approach.

Small False Alarm Rate (FAR): The false alarm rate is desired to be very small for an efficient and accurate detection system. A stand alone high detection rate is not enough to accurately detect anomalies in a graph. If among the detected anomalies, some benign nodes exist along the real attack node(s) then the accuracy of detection is significantly hampered.

Use of Real-world Dataset: Since real-world datasets are rich and unlabeled, it is the primary goal of the detection approach to perform well in real-world dataset. To run successfully in a real-world application, a mature and efficient detection algorithm must be tested with a real-world

dataset.

Small Time Complexity: Because real-world networks are extremely complex, finding suspicious nodes from such a large dataset takes a long time. As a result, having a low time complexity when running the detection algorithm is desirable.

1.7 Contributions

Our key contribution can be summarized as follows:

1. We construct an item-item co-visitation graph from a real world (MovieLens100k [29]) user-item rating dataset. To the best of our knowledge, we are the first to systematically generate a co-visitation graph from a real world rating dataset based on the timestamp of the ratings of a user. Existing detection techniques [5, 7, 8] for defending fake co-visitation injection attacks evaluate the detection performance on synthetic dataset generated using Erdos-Renyi random graph model.

2. While determining node attributes which preserve the structure of the graph and the connection between individual nodes, we incorporate two new unique attributes which represent how enmeshed a node is in the network. These two new attributes along with the typical baseline attributes combinely increase the overall performance of our proposed detection framework over the baselines.

3. We generate an attributed graph from our item-item co-visitation graph and apply CUR decomposition and residual analysis for conducting attribute selection and anomaly detection [18] respectively. As a real-world rich network may consist of many noisy and irrelevant attributes which may eventually bring adverse effects on detection rate, using a detection technique like ours helps to remove the noisy and irrelevant attributes, which will be very beneficial since it increases the overall performance of the detection process. By applying our methodology and running extensive experiments confirmed that, it improves false alarm rate (FAR) performance by upto 50% when compared to other state-of-the-art approaches while keeping the highest detection rate (DR).

4. The proposed detection approach is basically reducing the dimension of the datasets by removing noisy and irrelevant data. As a result, the time complexity of performing the detection algorithm on rich networks decreases.

5. We also discuss some abnormality forensics metrics for real data from diverse perspectives.

1.8 Thesis Organization

The organization of the rest of the thesis is as follows.

In **Chapter 2**, we have presented relevant background studies on various attacks and their detection techniques on graph based recommendation system.

In **Chapter 3**, we have discussed some preliminaries and baseline detection techniques used in our experiment.

In **Chapter 4**, we have formally presented fake co-visitation injection attack model and attack dimensions.

Chapter 5 presents the detection challenges of fake co-visitation injection attack in graph based co-visitation recommender system and give an overview of our proposed detection framework.

Chapter 6 explains our first proposed detection approach in detail. It also discusses the challenges in implementing our proposed scheme and how we addressed those challenges.

Chapter 7 discuss about the extensive experimental results and their analyses.

Finally, **Chapter 8** concludes our thesis. This chapter also includes the outlines of some future works related to this dissertation.

Chapter 2

Literature Review

In the age of big data, a recommendation system is crucial. The most well-known application of recommendation algorithms is on e-commerce platforms [30,31], where they suggest products to their customers and help them to determine which products they need to purchase. The products can be recommended based on the platform's top overall sellers, the consumer's demographics, or an analysis of the consumer's recent purchase activity as a predictor of future purchasing behavior [30]. Nowadays, almost all web service providers (e.g., Facebook, Youtube [24], Amazon [6], Netflix [32], eBay, etc.) use recommendation systems to help users locate relevant information. It not only helps the consumer by recommending their preferred items from a big ocean of data but also benefits the service providers to increase their sales [30, 32] which keeps them up with the competition.

To make the recommendation systems reach their true potential, many studies have been done regarding it that analyze its vulnerability and corresponding prevention so that the systems under study can be made more reliable and secured.

2.1 Collaborative Filtering Recommendation System

Among many popular algorithms for recommender systems, collaborative filtering is the most popular. D. Goldberg et al. were the first who coined the term 'Collaborative Filtering' where they have proposed a manual collaborative filtering technique for Tapestry mailing system [33].

Collaborative filtering suggests items based on the interests of other like-minded users or finds items that are comparable to those that the target user has previously rated. It uses statistical techniques [34] to find the similarity between the user or item vectors. CF can be further classified into two types: user-item and item-item.

The user-Based Collaborative filtering approach is based on the fact that people who shared similar opinions in the past are likely to share the same in the future. This was first introduced in GroupLens for recommending news articles based on other users' interests or ratings

[35]. Further, this concept was used for recommending music albums [36] and for video recommendation [37].

Trying to extend the scope of CF so that it can be applied to large datasets, a need arises for designing a more scalable algorithm. The basic concept behind the approach item-item, considers that the user possesses the same view for similar items, which was first used by [38]. This is also observed as "People who viewed this also viewed" feature in some very popular web service platforms who are using item-item base CF recommendation system.

2.1.1 Co-visitation Recommendation System

One particular type of collaborative filtering-based recommendation system is called the co-visitation recommender system, which takes advantage of co-visitation information between items. It is used and explained by Amazon.com that users might want to add items that are similar to the items present in the shopping cart [6]. After that, In 2010, YouTube reported using it for recommending videos [24]. In the following days, the recommendations were used so broadly by Amazon.com that a Microsoft Research report estimated that 30 percent of Amazon.com's page views were from recommendations [39]. Similarly, Netflix used recommender systems so extensively that their Chief Product Officer, Neil Hunt, provided indication that a substantial percentage (more than 80 percent) of contents watched on Netflix came through recommendations [32].

2.2 Recommendation System Vulnerabilities and Mitigation

To recommend desirable items to consumers, recommendation systems typically rely on user inputs (e.g., previous behavior, product ratings, and reviews). These user contributions strengthen the recommendation database, but they also leave the system vulnerable to a variety of attacks, including shilling, poisoning, and co-visitation attacks. Now we'll go through the work done to model attacks in recommendation systems.

2.2.1 Attacks On Recommendation Systems

Due to the gaining popularity of recommendation systems, it draws the attention of the attackers. Many types of research have been done regarding different types of possible attacks over the past decade. Some of them are mentioned below:

Xing et al. [26] recently proposed pollution attacks to the user-to-item recommendation and demonstrated that YouTube, Amazon, and Google search are vulnerable to the attacks. The goal of pollution attacks is to spoof the system to recommend a target item to a specific victim user.

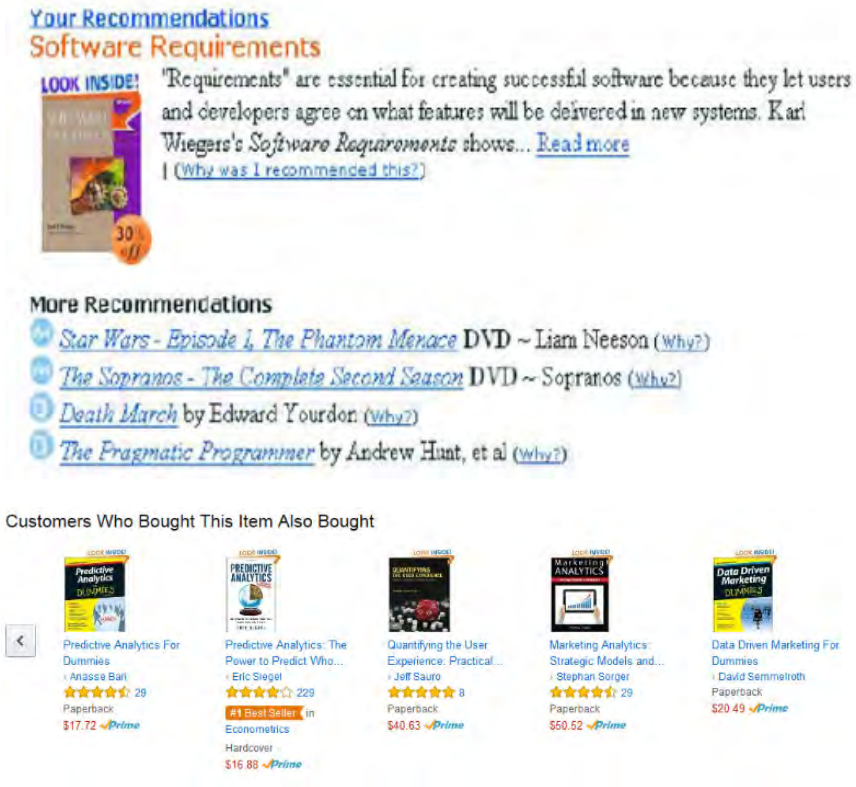


Figure 2.1: Amazon.com shopping cart recommendations. The recommendations are based on the items in the customer's cart:The Pragmatic Programmer and Physics for Game Developers [6]

The key idea is to inject fake information into the victim user's profile (e.g., browsing history) via cross-site request forgery (CSRF) [40]. Pollution attacks suffer from two key limitations:

- Pollution attacks rely on CSRF, which makes it hard to perform the attacks at a large scale.
- Pollution attacks are not applicable to item-to-item recommendation because the attacker cannot change the item that the user is visiting.

A few studies [41–43] have demonstrated that recommender systems (e.g., [6, 35, 44]) leveraging a user-item rating matrix are vulnerable to profile injection attacks (also called shilling attacks). Specifically, in a user-item rating matrix, each row corresponds to a registered user, and each column corresponds to an item; an entry in the matrix is the rating score that the corresponding user gave to the corresponding item; a rating score represents the corresponding user's preference to the corresponding item, and most entries of the matrix are missing since a user only provides feedback about a small number of items. Given such a matrix, these recommender systems infer the values of the missing entries and then recommend users items with the largest inferred values. Profile injection attacks aim to make a target item be recommended to more users. Specifically, in a profile injection attack, an attacker first registers a large number of fake accounts in the service. Then each fake account gives certain rating scores to a carefully chosen subset of items.

These profile injection attacks are not applicable to co-visitation recommender systems that do not rely on the rating matrices.

Calandrino et al. [45] proposed privacy attacks to infer a user's profile (e.g., the products that the user purchased on Amazon) via analyzing the publicly available recommendations that are made by the recommender system. Specifically, in their privacy attacks, an attacker first obtains a partial profile of a user. For instance, on Amazon, some users will review the products that they purchased. Through collecting these publicly available reviews, the attacker can obtain a subset of products that the target user purchased. Then the attacker monitors the temporal changes of item-item recommendation lists of these products. If an item appears in the recommendation lists of a large number of products that are in the target user's partial profile, the attacker infers that the user purchased the item. The authors demonstrated that this privacy attack is feasible on various popular web services, including Amazon, LibraryThing, Hunch, and Last.fm.

Author in [4] proposed a fake co-visitation injection attack in a graph-based co-visitation recommendation system where the attacker injects fake co-visitation to promote or demote a target item. The author claims that this attack can alter the result of the recommendation system of many popular web service providers, including Youtube, eBay, Amazon, Yelp, and LinkedIn.

- This attack does not rely on CSRF.
- It can be performed at scale.
- This type of attacks are applicable to both item-to-item recommendation and user-to-item recommendation.

We will discuss this particular attack model in detail in a later chapter.

2.3 Attack Detection Techniques

In the last two decades, academia and industry have focused a lot of attention on defending against these dangers. We'll go over some recent research findings in this section.

2.3.1 Pollution Attack Detection

There is a lot of work in the field of detecting different types of attacks in recommendation systems. In [46], the author proposed a number of metrics to distinguish between authentic and fake profiles, including the number of prediction-differences (NPD), standard deviation in user ratings, degree of similarity with top neighbors, degree of agreement with other users, and rating deviation from the mean agreement (RDMA).

Weighted deviation from mean agreement (WDMA) and weighted degree of agreement (WDA) were introduced in [25] [47]. Although WDMA is derived from RDMA, it places a greater emphasis on rating deviations for sparse items, resulting in greater information acquisition. A new concept called Length Variance (LengthVar) was proposed, which assesses the difference between a specific profile rating and the system's average rating. Nearest-neighbor classification using kNN, decision-tree learning using C4.5, and support vector machine classifier are the three classification methods utilized in [48]. RDMA, WDA, WDMA, degree of similarity with top neighbors (DegSim), and LengthVar were the attributes used. The attributes that have been covered so far fall into the generic group.

Outlier identification is another way of detecting attackers. These methods might be based on distance, density, clustering, or depth to find outliers.

2.3.2 Outlier Analysis Based Detection Techniques

Due to the gaining popularity of graph-based recommendation systems, which is a type of collaborative filtering-based recommendation system [49], outlier detection has been studied extensively in the last few decades.

Anahita Davoudi and Mainak Chatterjee [50] proposed a classification technique for detection of profile injection attacks in recommendation systems using outlier analysis. They define the attributes that provide the likelihood of a user having a profile of that of an attacker. Using the user-item rating matrix, user-connection matrix, and similarity between users, it is found if the ratings are abnormal and if there are random connections in the network. Then, k-means clustering is used to categorize users into authentic users and attackers.

In [51] The author proposed a structural connectivity model in order to define outliers in graph streams. Trying to solve the sparsity problem of massive networks, statistically, robust models of the connectivity behavior are constructed by dynamically partitioning the network. Also, a reservoir sampling method is presented in order to maintain structural summaries of the underlying network.

In [52], the authors proposed an effective outlier edge detection algorithm. They use the clustering property of social network graphs to give an edge an authentic score in this paper. The discrepancy between the actual and expected number of edges that connect the two sets of nodes that surround the investigating edge determines the edge's genuine score. The edges with low authentic scores are most likely outliers.

2.3.3 Clustering-based Detection Techniques

Bryan Perozzi and colleagues proposed the FocusCO algorithm, which finds the focus, extracts focused clusters and detects outliers in another paper [53]. Furthermore, they demonstrated that

FocusCO scales well with graph size since it performs local clustering of user-interested areas rather than global graph partitioning.

In [54], the authors mentioned that attacker profiles are highly correlated; thus, members of small clusters are considered to be attack profiles. Clustering has the advantage of being completely unsupervised compared to other approaches used for outlier detection.

Outlier nodes detection using clustering algorithms has been studied in few works [55, 56]. David Combe et al. [55] proposed a graph node clustering approach, I-Louvain, that allows splitting the vertices of an attributed graph when numerical attributes are associated with the vertices. They claim that combining the relational information with attributes allows detecting the communities more efficiently than using only one type of information. Falih and others investigated attributed network clustering algorithm ANCA [56] which can operate on undirected, weighted or unweighted, and multi-attributed graphs. It consists of two main components: matrix similarity calculation and clustering. The topological information is encapsulated in attributes that enrich the dimension space. Finally, k-means clustering is used for partitioning the graph vertices.

Residual analysis was used to investigate the residuals between real and estimated data, and it can aid in the understanding of anomalies in general. Anomalies are more likely to have substantial residual errors because their behaviors do not follow the patterns of the bulk of reference examples. Jundong Li and others have done extensive studies on residual analysis for anomaly detection in attributed networks in their work [14]. They introduced a learning framework for anomaly detection that characterizes the residuals of attribute information and their coherence with network information. They found anomalies whose behaviors are singularly distinct from the majority by learning and evaluating the residuals. A list of clustering based detection algorithms are shown in table-2.1.

2.4 Co-visitation Injection Detection

Focusing on co-visitation graphs to find outlier nodes has led to some novel works [5, 7, 8]. The next subsections go through the specifics of the aforementioned studies.

2.4.1 Residual Analysis Based Detection Approach

In [7], Yang and others proposed a unified detection framework to identify anomalies via mining association graphs, which expects to handle coexisted attacks as much as possible. They tried to spot profile injection attacks and co-visitation attacks for the recommendation system. Firstly, they generated a co-rating and co-visitation graph using association rules. Then attribute representations of nodes are developed from diverse perspectives. Finally, attribute information

and connective coherence of graphs are combined to infer suspicious nodes using residual analysis. Fig.-2.2 presents their proposed framework.

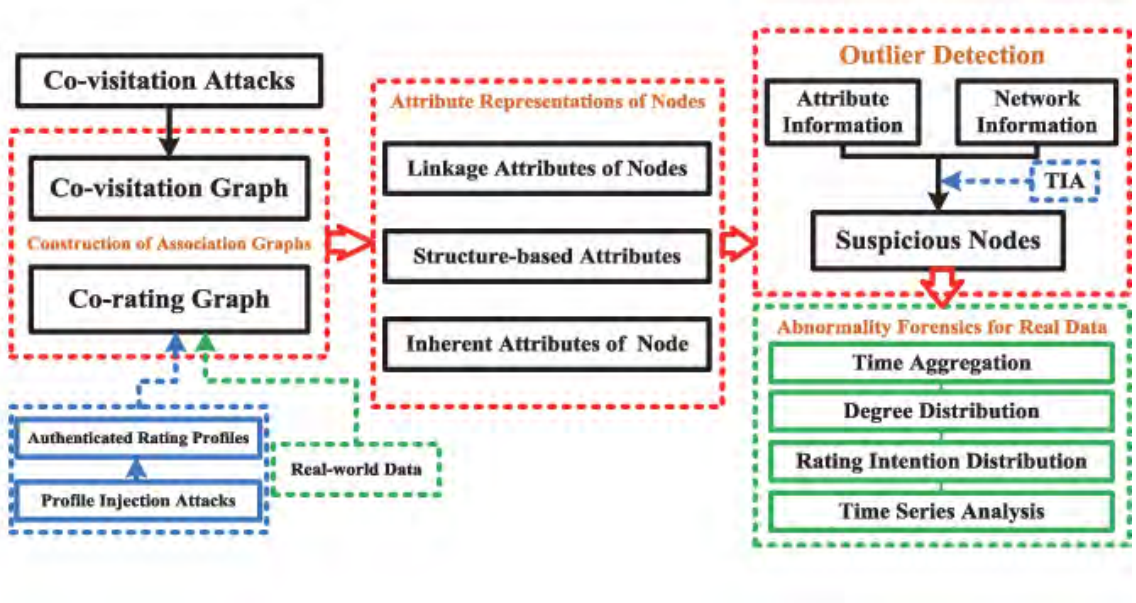


Figure 2.2: The framework of residual analysis based detection method [7]

Limitations: To begin, the authors created a co-visitation graph using a random graph model. They also constructed a co-rating graph using a real-world user-item rating dataset. To detect profile injection attacks, they use a co-rating graph. Co-visitation graph, on the other hand, is used to detect fake co-visitation injection attacks. Second, data that is noisy or structurally unimportant is not removed in this approach. This disturbed data not only reduces overall detection performance by raising the false alarm rate (FAR), but it also takes longer to run the detection algorithm, resulting in a high time complexity.

2.4.2 Higher Order Conditional Random Fields Based Detection approach

In [5], the author presents a unified detection approach to identify profile injection and co-visitation injection attacks using higher-order conditional random fields. For profile injection and co-visitation injection attacks, weighted bipartite graphs and weighted co-visitation graphs are first built from raw real-world and synthetic data. Then, using the bipartite network, they estimate the association relationship between nodes, which is helpful in detecting untrusted links. Then, before anomaly detection, the disturbed data (such as inactive users and unpopular items) is identified and eliminated to reduce false alarm rates while maintaining high detection rates. They use recursive propagation to enhance the relationships in association graphs and incorporate the graphic representation of coupled networks for probabilistic analysis and trustworthiness

evaluation of associative links. Suspicious nodes (users or things) in the source network can be empirically recognized using the redetermined links in the target network and refined cross-network. Fig.-2.3 depicts their proposed framework.

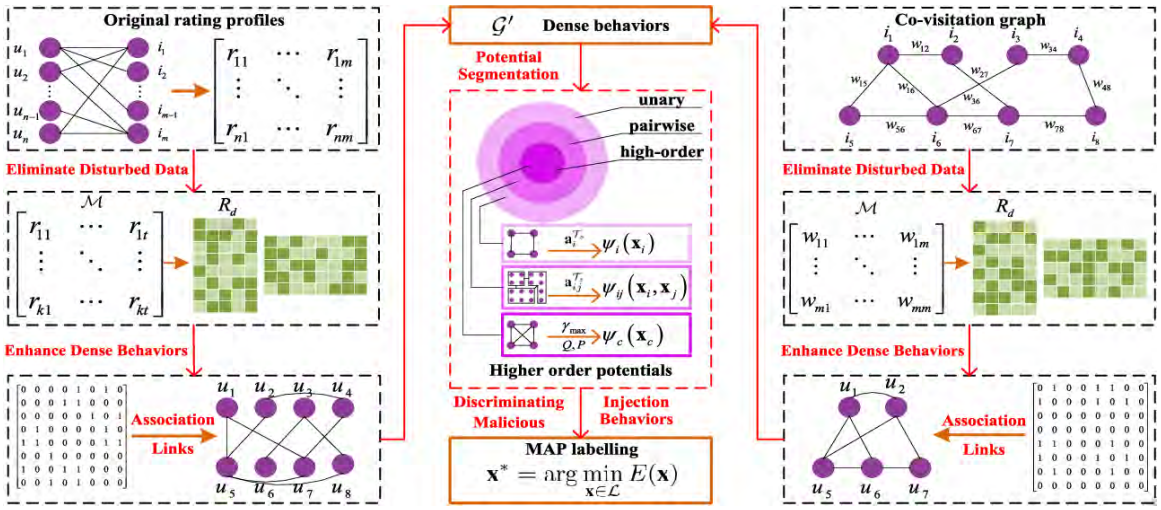


Figure 2.3: The framework of higher-order conditional random fields based detection method [5]

Limitations: Similarly to the previous study, the authors created the co-visitation graph using a random graph model rather than a real-world dataset. In addition, the author removes erroneous data from the graph in order to make the system resistant to both minor and large-scale fake co-visitation injection attacks. However, because this elimination occurs early in the detection process, it has negative consequences in terms of structured behavior representations. As a result, in some circumstances, overall detection performance may suffer.

2.4.3 Probabilistic Inference Based Detection Approach

Another study from Y. Zhang and others [8] presents a unified detection framework to spot malicious threats by investigating probabilistic inference and trustworthiness evaluation of behavioral links according to coupled association networks. Based on both the inherent rating motivation of users and the atomic propagation rules of coupled networks, firstly an association graph is constructed from the original rating matrix. Then, the trustworthiness of link behaviors in the targeted network of coupled association networks is evaluated by exploiting a factor graph model of the coupled network, and concerning links in the targeted network are redetermined. Finally, anomalous users and items can be detected by probabilistic-ally inferring link behaviors and comprehensively evaluating the trustworthiness of nodes in the targeted network.

Limitations: First, the authors used the Erdos-Renyi (ER) random network model to produce a co-visitation graph, but they used a real-world user-item rating dataset to create a co-rating graph. However, only synthetic data is used to detect fake co-visitation injection attacks. Furthermore,

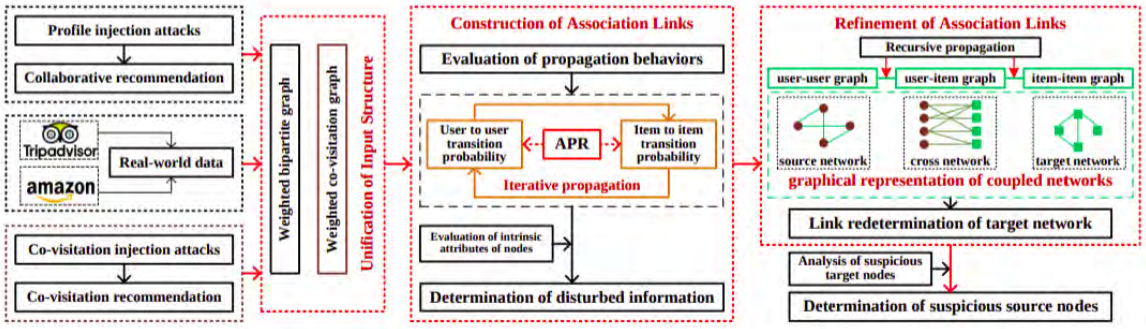


Figure 2.4: The framework of probabilistic inference based Detection approach [8]

the three atomic propagation rules of linked networks used in this study are not always suitable for characterisation of user-item associated network propagation behavior.

2.5 Summary

In this chapter, we have discussed contemporary research works on attacks and their detection techniques on recommendation systems. We also discussed about the limitations of the existing detection techniques. In the next chapter we will briefly discuss about some terminologies followed by fake co-visitation injection attack model and our proposed detection approach for defending fake co-visitation injection attack on graph based recommendation system.

Table 2.1: A list of available community detection algorithms based on attributes [15].

Paper Ref.	**Year**	**Contribution for Attribute-based community detection**	**Technique used**	**Validation methods used**
[57]	2009	SA-Cluster algorithm (Used both node attributes and structural similarities)	• Augmented graph • Neighborhood random walk distance • K-Medoids	• Entropy • Density for the quality of clusters
[58]	2010	Inc-Cluster algorithm	• Augmented graph • An incremental algorithm for Random walk distance	Same as above.

[59]	2013	CESNA algorithm (edge structure and node attributes)	• Probabilistic approach • Takes linear time	F1 score
[60]	2010	GAMER algorithm	Two fold clusters (used sub-space p and quasi-cliques properties)	F1 value
[61]	2011	• Entropy Optimization Algorithm • Augmented Graph Clustering Algorithm	• Entropy minimization • Simple Matching Coefficient	• Rand Index • Louvain for modularity optimization • Cosine Distance
[62]	2012	• SAC1 Algorithm • SAC2 Algorithm	• User attributes and Louvain (Composite modularity) • KNN graph with Louvain	Result Comparison with Louvain and attribute based clustering
[55]	2015	I-Louvain algorithm	• Inertia-based modularity • Fusion Matric Inertia • Used Modularity, relationship information and attributes	Normalized Mutual Information (NMI)
[63]	2016	kNAS algorithm	• Local Outlier Function • k-neighborhood • Similarity score	• Density • Tanimoto Coefficient
[64]	2016	SHC Algorithm	• Cosine similarity • Louvain algorithm	• NMI • Best Q

[65]	2016	aLBCD algorithm	• Action similarity • Euclidean distance	• Accuracy • Adjusted Rand Index
[66]	2016	EM algorithm.	Belief propagation	• Accuracy • Modularity
[67]	2016	• NMMA model • Bayesian nonparametric attribute (BNPA) model	Multinomial distribution	MNI
[68]	2013	Selection method	• Introduced CNMI to manage noise • Introduced mixing parameter	• Modularity • CNMI

Chapter 3

Preliminaries

First, we define certain terms utilized in our paper in this chapter. After that, some existing detection models of co-visitation injection attack are discussed. Table 4.1 gives the notations we use throughout the chapter.

3.1 Attributed Network

We consider co-visitation graph as a weighted attributed graph where each node is associated with multiple attributes and each edge has weights. An attributed graph can be defined as:

Definition 3.1.1. *An attributed graph G can be defined as (V, E, F).Where V is a set of n nodes, $V = \{v_1, v_2, ..., v_n\}$ and E is a set of m edges, $E = \{e_1, e_2, ..., e_m\}$ where, $E = \{(u, v) : u, v \in |V|, u, v\}$, F is a set of d attributes, = $\{f_1, f_2, ...f_d\}$ which is associated with each node.*

Fig.-3.1 depicts a attributed network of 6 nodes. Where each node is associated with 5 attributes.

3.1.1 Matrix for mathematical expression of Attributed Network

Attributed graph can be mathematically represented with the help of an adjacency matrix and attributed matrix. Both of the matrices can be defined as:

Definition 3.1.2. *Adjacency Matrix is a $2D$ array of size $V \times V$ where V is the number of vertices in a graph. Let a slot $adj[i][j] = 1$ indicates that there is an edge from vertex i to vertex j. Adjacency matrix for undirected graph is always symmetric.*

Fig.-3.2 represents a adjacency matrix which represents their link relationships in the graph of 5 nodes. It is a boolean matrix where adj[i][j] is set to true if an edge exist between node i and node j, otherwise 0.

[[[]

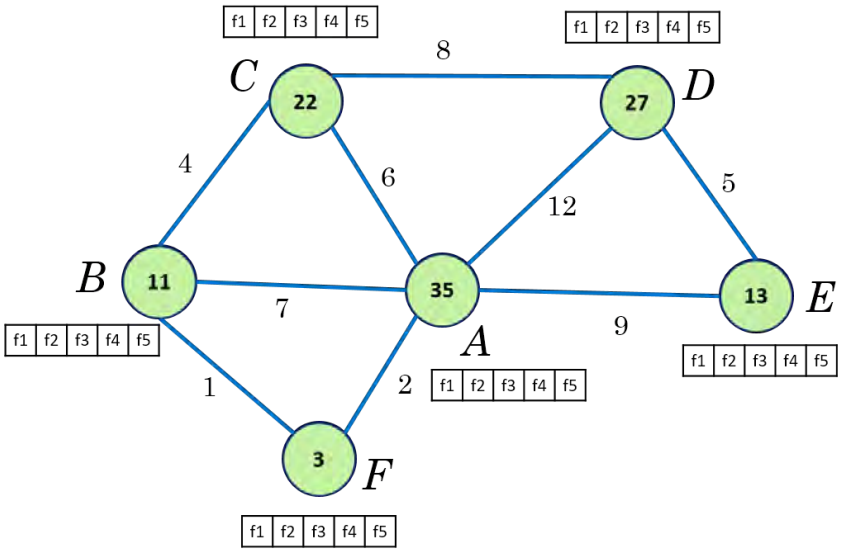


Figure 3.1: Attributed Network

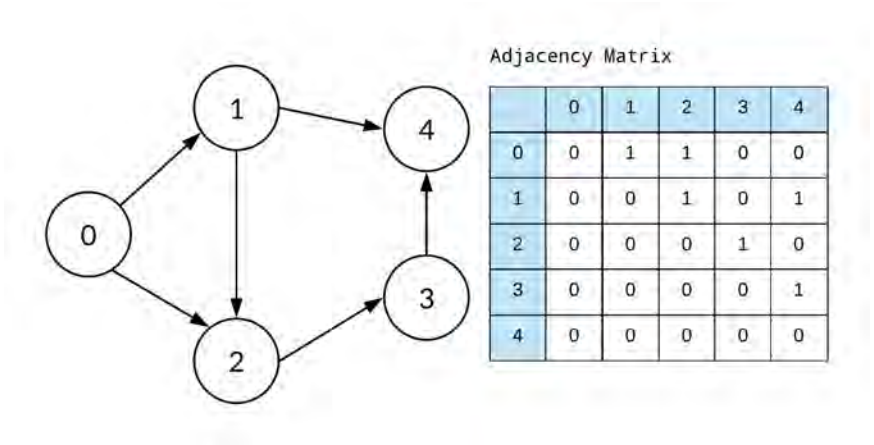


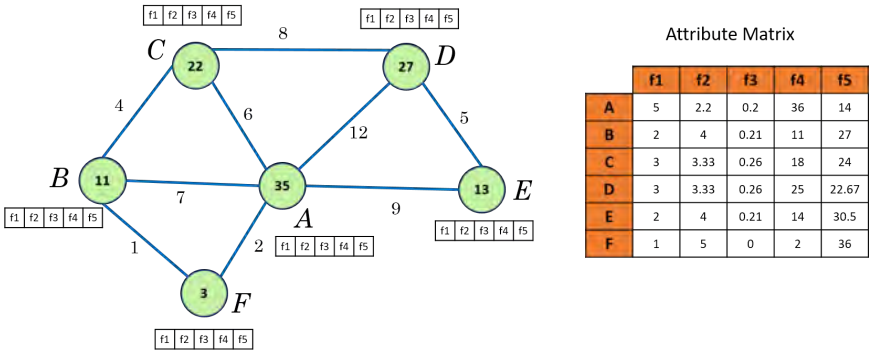
Figure 3.2: Adjacency Matrix [9].

Definition 3.1.3. *Attributed Matrix is a 2D array of size $V \times F$ where V is the number of vertices in a graph and F is the number of attributes associated with each node. A slot $attributed[i][j] = a_j$ indicates that vertex v_i has attributes j whose value is a_j.*

Fig.-3.3 depicts an attributed matrix of an attributed network of 6 nodes. Here each row represents a node in the graph and each column represents an attribute or feature.

Definition 3.1.4. *A diagonal matrix is a square matrix, where all of it's non-diagonal elements are zeros. a matrix $A = [a_{ij}]$ is said to be diagonal if A is a square matrix and $a_{ij} = 0$ when $i \neq j$ [10].*

Fig.-3.4 depicts a nxn diagonal matrix. Two examples are given to demonstrate the diagonal matrix.



	f1	f2	f3	f4	f5
A	5	2.2	0.2	36	14
B	2	4	0.21	11	27
C	3	3.33	0.26	18	24
D	3	3.33	0.26	25	22.67
E	2	4	0.21	14	30.5
F	1	5	0	2	36

Figure 3.3: Attributed Matrix

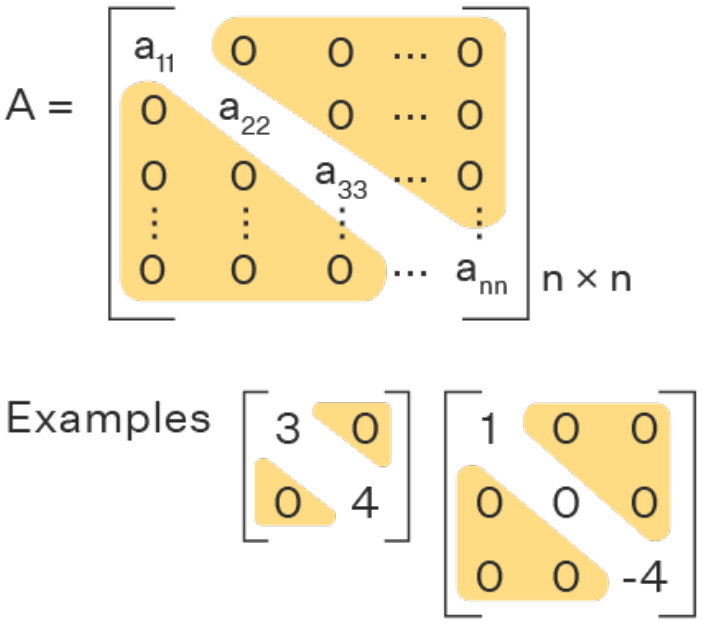


Figure 3.4: Diagonal Matrix [10].

Definition 3.1.5. *An orthogonal matrix is a square matrix A if and only its transpose is as same as its inverse. i.e., $A^T = A^{-1}$, where A^T is the transpose of A and A^{-1} is the inverse of A. So it can also be defined as, A square matrix A is said to be orthogonal if*

- $A^T = A^{-1}$ *(or)*
- $AA^T = A^T A = I$

We know that $AA^{-1} = I$, where I is an identity matrix (of the same order as A) [69].

Fig.-3.5 depicts an example of 3x3 orthogonal matrix.

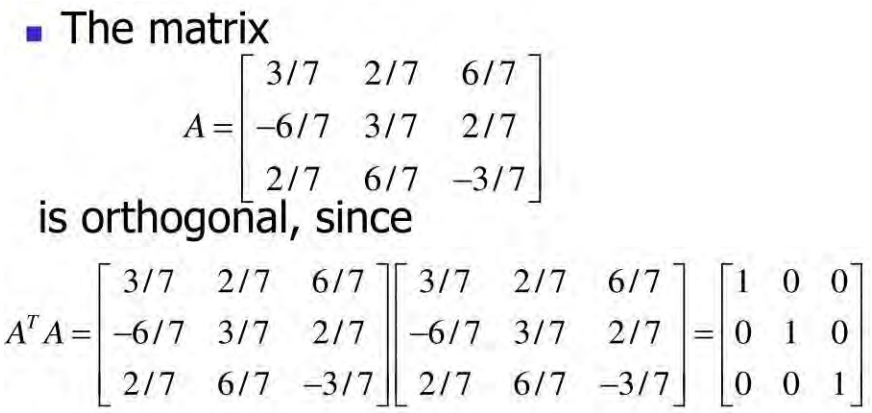


Figure 3.5: Orthogonal Matrix.

3.2 Dimensionality Reduction Techniques

The amount of features (i.e. input variables) in your dataset is referred to as ”dimensionality” in machine learning. Reducing the number of input variables for a dataset is referred to as dimensionality reduction. While adding more features/dimensions improves the performance of any machine learning model, there comes a point where adding more causes performance degradation, which is when the number of features is very large in comparison to the number of observations in your dataset. Several linear algorithms work hard to train efficient models. This is called the ***“Curse of Dimensionality”***. Dimensionality reduction is a set of approaches for reducing the quantity of data while keeping the most critical information and overcoming the dimensionality curse. It has a significant impact on the results of classification and clustering tasks [11].

Definition 3.2.1. *When dealing with high dimensional data, it is often useful to reduce the dimensionality by projecting the data to a lower dimensional subspace which captures the “essence” of the data. This is called dimensionality reduction [70].*

The figure-3.6 illustrates a $3D$ feature space is split into two $1D$ feature spaces, and later, if found to be correlated, the number of features can be reduced even further. A $3D$ classification problem can be difficult to picture, but a $2D$ problem can be mapped to a basic two-dimensional space, and a $1D$ problem to a simple line. The below figure illustrates this concept, where a $3D$ feature space is split into two $1D$ feature spaces, and later, if found to be correlated, the number of features can be reduced even further [11].

There are two components of dimensionality reduction [11]:

- **Feature selection:** In this step, a subset of the original set of variables, or features are selected, to get a smaller subset which can be used to model the problem. It usually involves three ways:

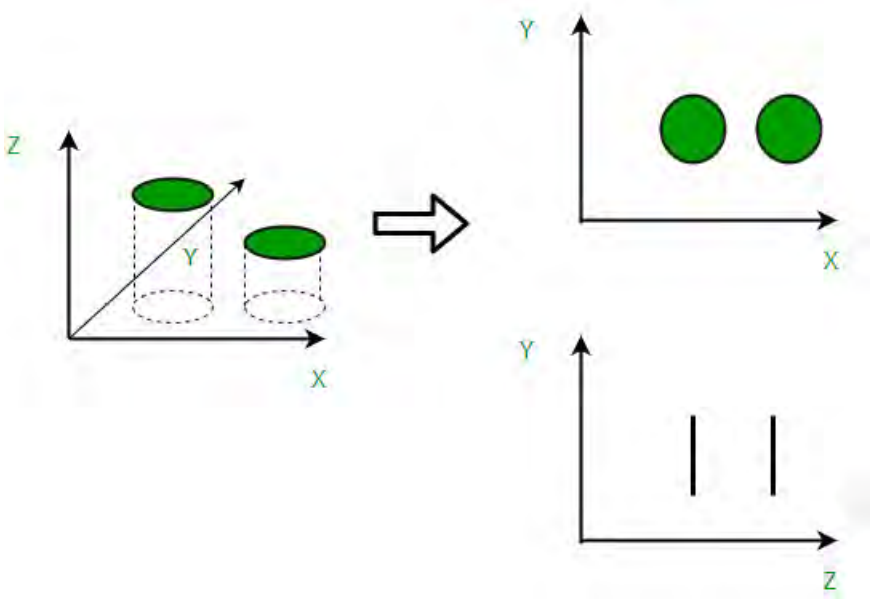


Figure 3.6: Dimensionality Reduction [4].

  - Filter
  - Wrapper
  - Embedded

- **Feature extraction:** This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions.

3.2.1 Methods of Dimensionality Reduction

The various methods used for dimensionality reduction include [11]:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Generalized Discriminant Analysis (GDA)

3.2.2 Principal Component Analysis (PCA)

Principal Component Analysis(PCA) is one of the most popular linear dimension reduction algorithms. This method was introduced by Karl Pearson. It works on the premise that when data from a higher dimensional space is translated to data from a lower dimensional space, the variance in the lower dimensional space should be maximum [11]. It's a projection-based method for transforming data into a set of orthogonal (perpendicular) axes.

Definition 3.2.2. *PCA can be defined as the orthogonal projection of the data onto a lower dimensional linear space, known as the principal subspace, such that the variance of the projected data is maximized [71].*

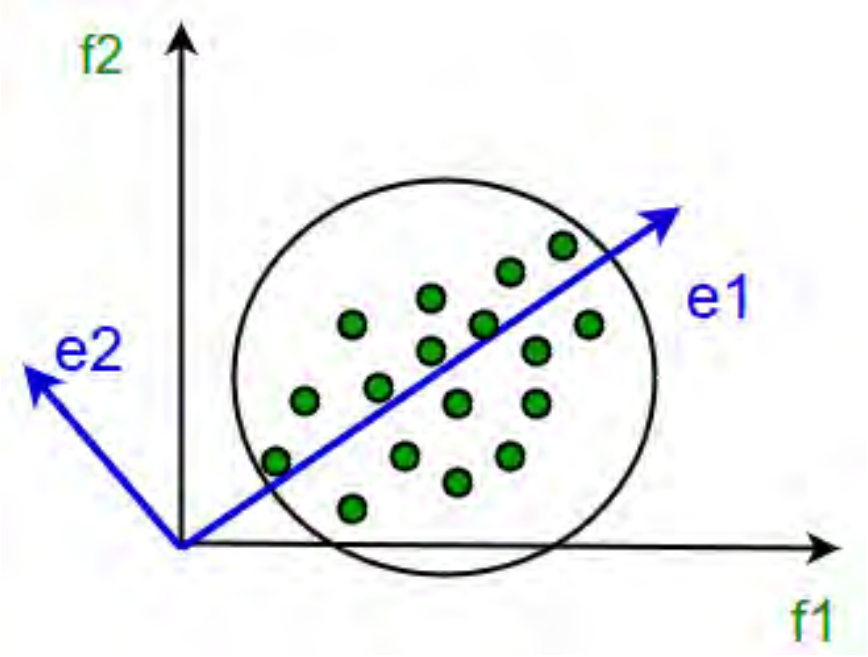


Figure 3.7: Principal Component Analysis [11].

Linear algebra tools, specifically a matrix decomposition like an Eigendecomposition or SVD, can be used to describe and implement the PCA method.

3.2.3 Singular-Value Decomposition (SVD)

Singular-Value Decomposition (SVD) is a matrix decomposition method for reducing a matrix to its constituent pieces in order to simplify certain subsequent matrix operations.

Definition 3.2.3. *The singular value decomposition (SVD) provides another way to factorize a matrix, into singular vectors and singular values. The SVD allows us to discover some of the same kind of information as the eigendecomposition. However, the SVD is more generally applicable.*

For the case of simplicity real-valued matrices are focused for SVD. It can formally presents as:

$$A = U.Sigma.V^T \tag{3.1}$$

Fig.-3.8 represents Singular Value Decomposition(SVD), where A is the real mxn matrix that we wish to decompose, U is an mxm matrix, Sigma (often represented by the uppercase Greek

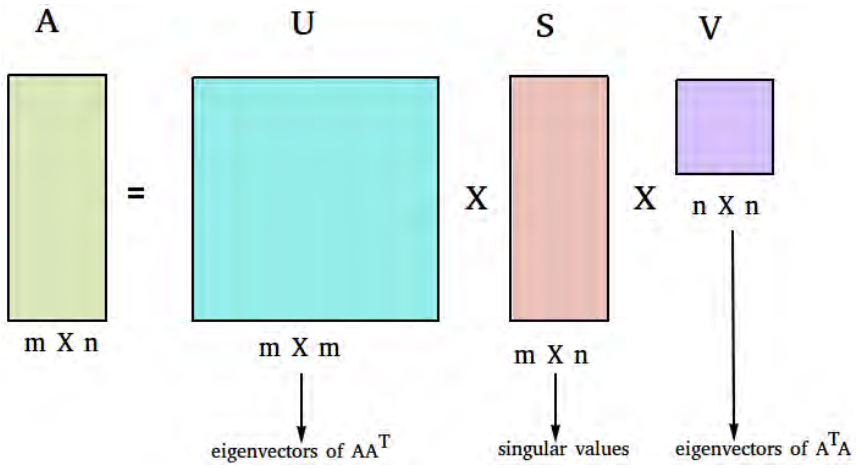


Figure 3.8: Singular-Value Decomposition [12].

letter Sigma) is an mxn diagonal matrix, and V^T is the transpose of an nxn matrix where T is a superscript.

3.2.4 CUR Matrix Decomposition

The PCA and, more broadly, the SVD is basic data analysis methods that express a data matrix as a series of orthogonal or uncorrelated vectors of diminishing relevance. Unfortunately, because these vectors are linear combinations of up to all data points, they are notoriously difficult to interpret in terms of the data and processes that generated them. So, CUR matrix decompositions [72] is developed as an alternative to Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) for improved data analysis. The CUR matrix decomposition chooses columns and rows from the data matrix that have a lot of statistical leverage or much influence. A small number of the most essential features and/or rows can be identified from the original data matrix by implementing the CUR matrix decomposition procedure. As a result, CUR matrix decomposition is a valuable technique for performing exploratory data analysis. CUR matrix decomposition is useful for regression, classification, and clustering in a variety of situations [72, 73] .

Definition 3.2.4. *CUR matrix decomposition is a low-rank matrix decomposition algorithm that is explicitly expressed in a small number of actual columns and/or actual rows of data matrix [72].*

In fig.-3.9 given an $m \times n$ matrix A, we split it into 3 matrices, C, U, and R, where C consists of a small number of actual columns of A, R consists of a small number of actual rows of A, and U is a small carefully constructed matrix that guarantees that the product CUR is "close" to A. Of course, the extent to which $A \approx CUR$, and relatedly the extent to which CUR can be used in

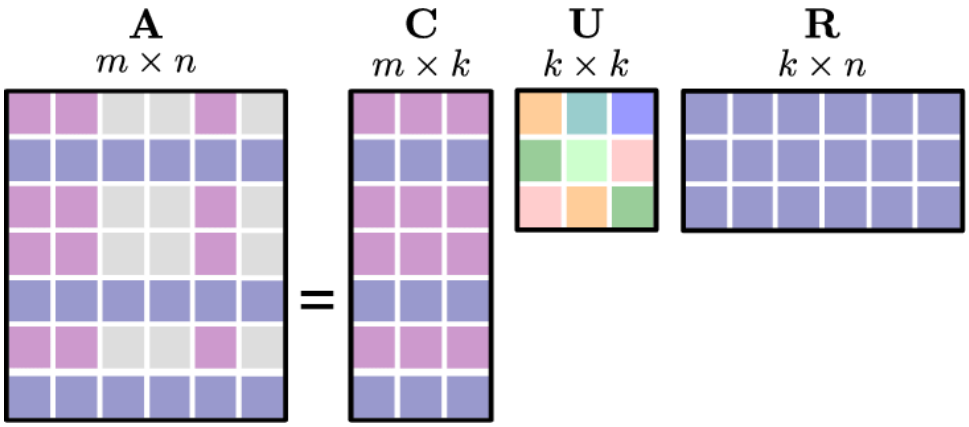


Figure 3.9: CUR Matrix Decomposition [13].

place of A or A_k in data analysis tasks, will depend sensitively on the choice of C and R, as well as on the construction of U [72].

3.3 Residual Analysis

By defining residuals and reviewing residual plot graphs, residual analysis is used to examine the adequacy of a linear regression model [74].

Definition 3.3.1. *Residual(e) refers to the difference between observed value (y) vs predicted value ($\hat{y}$). There is one residual for each data point.*

$$residual(e) = observedValue(y) - predictedValue(\hat{y}) \tag{3.2}$$

After finding out the residuals, a scatter plot of x versus e is drawn to see whether there is a pattern. If the residual plot shows no clear pattern, but just a big blob of points, then the linear regression model is appropriate.

3.3.1 Regression Analysis

Definition 3.3.2. *A set of statistical procedures for estimating relationships between a dependent variable and one or more independent variables is known as regression analysis. It can be used to determine the strength of a relationship between variables and to predict how they will interact in the future [75].*

3.3.2 Regression Analysis Types

There are various types of regression analysis, including linear, multiple linear, and nonlinear. Simple linear and multiple linear models are the most frequent.

Table 3.1: Dataset of 9 data points.

x	1	2	3	4	4	4	5	5	6
y	6	5	5	6	4	10	3	4	3

3.3.2.1 Simple Linear Regression

Definition 3.3.3. *A model that examines the relationship between a dependent variable and an independent variable is known as simple linear regression [75].*

The simple linear model is expressed using the following equation:

$$y = a + bx + \epsilon \tag{3.3}$$

Where, y , x, a , b and ϵ represent dependent variable, independent (explanatory) variable, intercept, slope and residual (error) respectively.

3.3.2.2 Multiple Linear Regression

With the exception that numerous independent variables are utilized in the model, multiple linear regression analysis is quite similar to simple linear regression analysis [75]. The mathematical representation of multiple linear regression is:

$$y = a + bx_1 + cx_2 + dx_3 + \epsilon \tag{3.4}$$

Where, y represents dependent variable, x_1, x_2 and x_3 represent indepent variable , b, c and d stand for slopes and ϵ represent residuals.

3.3.3 Residual Analysis in Outlier Detection

Residual analysis is used to find out residuals between estimated data and true data to spot anomalies in attributed graph as anomalies usually have large residual errors [76]. The reason behind this large residual error is that anomalous instances deviate significantly from majority of reference instances in pattern.

Definition 3.3.4. *An outlier is a point with an unusually large residual.*

We can extend the understanding with below example. Table-3.1 shows a small dataset of 9 data points.

The formula for calculating slope from a dataset is given below:

$$Slope = \frac{(N\sum xy) - (\sum x)(\sum y)}{Nx\sum x^2 - (\sum x)^2} \tag{3.5}$$

Table 3.2: Dataset of 9 data points with residuals.

x	1	2	3	4	4	4	5	5	6
y	6	5	5	6	4	10	3	4	3
$\hat{y}$	6.5	6	5.5	5	5	5	4.5	4.5	4
$e = y - \hat{y}$	-0.5	-1.0	-0.5	1.0	-1.0	5.0	-1.5	-0.5	-1.0

Using equation-3.5 we can calculate the slope from dataset shown in table-3.1. So using the linear equation-3.3, we can estimate the linear Equation of predicted value of y:

$$\hat{y} = -0.5x + 7 \tag{3.6}$$

Using the linear equation of predicted value $\hat{y}$, we can calculate the $\hat{y}$ for all data points in the dataset shown in table-3.2. We can also calculate the residuals for each data point using equation-3.2. Now we can plot the dataset in a 2-dimentional plane where dependent variable (y) is on the vertical axis and the independent variable (x) is on the horizontal axis.

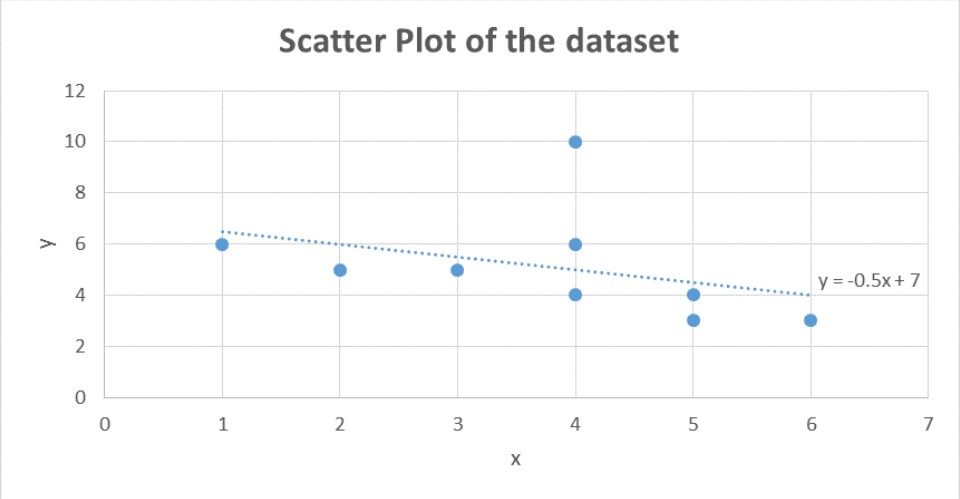


Figure 3.10: Scatter plot of residuals.

From fig.-3.10 and table-3.2, it can be seen that the datapoint (4,10) is an outlier as it has the most residual error 5 which is above standard deviation (2.0) for all the residuals.

3.4 Available Detection Techniques

In fig.-3.9 Schematic of the rank-k CUR decomposition of an m $\times$ n matrix. The components of are formed by small subset of actual columns and rows of the input matrix.

Algorithm 1: K-means algorithm [16] [17]

Input : Data set consisting of n-dimensional vectors, number of clusters k
Output : $S_i, i = 1...k$ containing the clusters

1. Randomly choose k different data points $M_i, i = 1...k$ to serve as the initial centers of the clusters $S_i, i = 1...k$;
2. Assign the data points to the cluster with the closest M_i;
3. Recalculate the centers M_i of each C_i;
4. Repeat from step 2 until there are no more changes in M_i, C_i, $i = 1...k$;
5. Output $S_i, i = 1...k$ containing the clusters.

3.4.1 Detection using K-Means

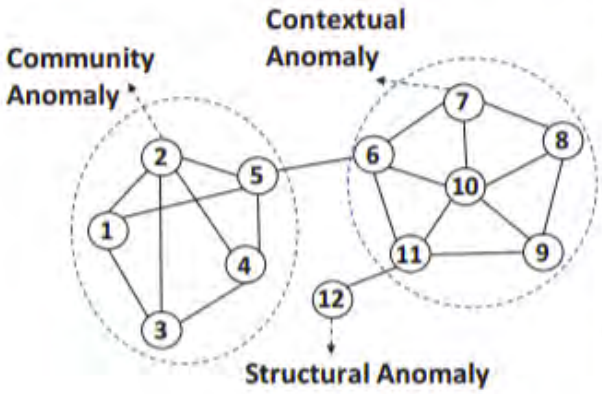
K-means is considered to be the most famous clustering algorithm. The goal of this algorithm is to partition the data points into k clusters such that each point of a cluster has a minimum distance from the center. The center of a cluster is calculated as the mean of the data points it contains. The most common formulation of the method is the two-step iterative refinement technique. In each step data points are allocated to the nearest cluster, then the centers of the clusters are recalculated. When the assignments do not change, the process stops.

The algorithm is presented in algorithm 1 which divides n-dimensional datapoints of the input dataset into k different clusters according to their euclidean distance.The goal is to minimize the the distance between the points inside the dataset.

K-means clustering algorithm is one of the most popular clustering algorithm used in community detection. In graph theory and machine learning , community detection in complex network is a long studied problem . In real-world complex graph network each node is associated with additional attributes. The goal of the community detection is to group the graph nodes into clusters that contain dense connections within those clusters and small connections to other clusters considering both attribute and structural information. K-means [16] method only considers the attribute information of nodes while clustering.

3.4.2 Detection using Radar

Radar is a learning framework proposed in [14] that is used for anomaly detection by modeling the residuals of attribute information and its coherence with network information for anomaly detection. Due to the increasing popularity of attributed networks in different domains such as social networks, gene regulatory networks, financial networks, etc, anomaly detection becomes very crucial to solve. An extensive majority of existing anomaly detection algorithms assume that certain properties of anomalies are known in advance in a specific context, such as structural



ID	f1	f2	f3
1	2	22	48
2	11	23	48
3	3	21	47
4	4	22	46
5	3	24	49
6	12	19	51
7	11	38	55
8	10	19	56
9	13	20	49
10	14	20	49
11	11	21	53
12	12	22	52

Figure 3.11: A toy example to describe different types of anomalies in a specific context [14].

anomaly, contextual anomaly and community anomaly. These three types of anomalies using different contexts can be better represented by a toy example in fig-3.11. When only network information is considered, node 12 is considered as a structural anomaly as it does not belong to any communities. On the other hand, taking only attribute information in consideration, node 7 is considered since contextual anomaly as its second attribute $(f2)$ value, deviates significantly from the other nodes. Considering both network and attribute information, node 2 is anomalous. Although its attribute values on $f1$, $f2$ and $f3$ are normal over the entire dataset, its attribute value on $f1$ is relatively higher than the other nodes in the same community (node 1, 3, 4 and 5), therefore, it is referred as a community anomaly.

But in real-world attributed networks, this widely used assumption might not be true. Therefore, it is expedient and beneficial to spot anomalies in a general sense. The challenges which are responsible for detecting anomalies in a more general way are listed below:

- Various types of anomalies coexist which makes the detection more difficult without prior knowledge
- For new types of attacks, node attributes characterizing becomes limited
- Prior knowledge regarding properties of anomalies might not be true.

The author of [14], proposed a principled way of identifying and detecting anomalies via residual analysis. Residual analysis is a method of measuring residual errors between true data and estimated data. Instances with large residual errors tend to be anomalies since their behaviors do not follow the patterns of majority reference instances.

Suppose, true data $\mathbf{X}$ is constructed by selecting structure related attributes and representative instances. $\tilde{\mathbf{X}}$ represents the estimated attributed information. Then the residual error can be well

presented by $\mathbf{X} - \tilde{\mathbf{X}}$. This residual errors can be used to detect contextual anomaly as content patterns of anomalies which deviates significantly from the majority of the normal instances [76]. An objective function of attribute information can be mathematically formulated by considering row sparsity to constrain the number of abnormal instances, is presented below,

$$A : \min_{\mathbf{C},\mathbf{D}} \|\mathbf{X} - \mathbf{C}'\mathbf{X} - \mathbf{D}\|_F^2 + \alpha\|\mathbf{C}\|_{2,1} + \beta \|\mathbf{D}\|_{2,1} \tag{3.7}$$

where α and β control the row sparsity of the coefficient matrix $\mathbf{C}$ and the residual matrix $\mathbf{D}$, respectively. It is noteworthy that , a large norm of **D(i,:)** indicate that the instance has a higher probability to be an anomaly.

The correlation between attribute and network information in attributed networks is also incorporated to detect anomalies in addition to the contextual anomaly. It is assumed that instances having similar patterns are more tend to be linked together in attributed network [14]. Minimizing the following term is exploited to mathematically represent the linkage relationship between instances,

$$B : \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\mathbf{D}(\mathbf{i},:), \mathbf{D}(\mathbf{j},:))^2\mathbf{A}(\mathbf{i},\mathbf{j}) \tag{3.8}$$

Where n is the number of rows and $\mathbf{A}$ is an adjacency matrix which represents their link relationships.

At the end, a anomaly detection framework is integrated to model the residuals of attribute information and its coherence with network information. The learning framework can be mathematically represented by minimizing the following optimization term,

$$\begin{aligned} &\min_{\mathbf{C},\mathbf{D}} A + \gamma B = \|\mathbf{X} - \mathbf{C}'\mathbf{X} - \mathbf{D}\|_F^2 + \alpha\|\mathbf{C}\|_{2,1} + \beta \|\mathbf{D}\|_{2,1} \\ &+\gamma\tfrac{1}{2}\textstyle\sum_{i=1}^{N}\sum_{j=1}^{N}(\mathbf{D}(\mathbf{i},:), \mathbf{D}(\mathbf{j},:))^2\mathbf{A}(\mathbf{i},\mathbf{j}) \end{aligned} \tag{3.9}$$

where $\mathbf{D}$ represents the residual matrix, γ denotes a parameter to balance the contribution of attribute reconstruction and network modeling [14] [7]. By analyzing the residual matrix $\mathbf{D}$, the anomalies can be ranked according to the residual errors.

The key steps of spotting anomalies in an attributed network of frameword Radar are summarized in Algorithm 2. From this algorithm the time complexity can be measured. At each iteration, the most costly operation is the matrix inverse operation requiring $O(n^3)$ caused by matrix update. Fortunately, the linear equation system can speed up the update update operation. So the total time complexity is $\#iterations * (O(n^2d) + O(n^3)$ [14].

Algorithm 2: Anomaly detection in attributed networks via residual analysis (radar) [14]

Input : Adjacency matrix A, attribute matrix X, parameters $\alpha, \beta, \gamma, \varphi$

Output : Top m instances with the highest abnormal scores.

1. Build Laplacian matrix $\mathbf{L}$ from the adjacency matrix $\mathbf{A}$;
2. Initialize $\mathbf{T_R}$, $\mathbf{T_W}$ to be identity matrix;
3. Initialize $\mathbf{D} = (\mathbf{I} + \beta\mathbf{T_R} + \gamma\mathbf{L})^{-1}\mathbf{X}$.
4. **while** objective function in Eq. (6) not converge **do**
5. Update $\mathbf{C}$ by Eq. (14);
6. Update $\mathbf{T_W}$ by setting $\mathbf{T_W}$ (i,i) = $\frac{1}{2\|\mathbf{C(i,:)}\|_2}$.
7. Update $\mathbf{D}$ by Eq. (9);
8. Update $\mathbf{T_R}$ by setting $\mathbf{T_R}$ (i,i) = $\frac{1}{2\|\mathbf{D(i,:)}\|_2}$.
9. **end while**
10. Compute the abnormal score for the i-th instance as $\|\mathbf{D}(i,:)\|_2$
11. Output top m instances with the highest abnormal scores.

Chapter 4

Attack Model

In this chapter we formally define the fake co-visitation injection attack model in graph based co-visitation recommendation system. There are two main aspects of a fake co-visitation injection attack that are needed to describe an attack on a collaborative recommendation system: the attack model, and the attack dimensions. Below we discuss these two concepts.

4.1 Co-visitation Attack Model

Before discussing the details, we first discuss about some terminology used to define the co-visitation attack model.

4.1.1 Attack Model Terminology

`Target Item(`i_t`):` A target item is an item which an attacker wants to promote or demote in a recommendation system.

`Anchor Item(`V_a`):` An anchor item is an item whose recommendation lists haven't included the target item yet in promotion attack. On the contrary, in demotion attack anchor item's recommendation list may contain the target item.The attacker's motive is to get more (promotion) or less (demotion) user impression by appearing (promotion) or disappearing (demotion) target item in the recommendation list of the anchor items.

`Filler Item(`V_f`):` To make the attack more realistic,the attacker selects some filler item randomly and inject fake co-visitations.

`Popularity Threshold(`τ`):` τ is the minimum threshold value for any item to be appeared in the recommendation list of another item.

Table-4.1 gives the principal notations used to exhibit fake co-visitation injection attack model.

Table 4.1: Attributed Networks: Notations.

Notation	Description
V	Set of n instances, $V = \{1, 2, 3, ..., n\}$
E	Set of m edges, $E = \{e_1, e_2, ..., e_m\}$
F	Set of d-dimensional attributes, $F = \{f_1, f_2, ..., f_d\}$ associated with each instance
k	Number of most similar items of item i showing on it's recommendation list.
V_a	Set of p anchor items, $V_a = \{i, ..., p\}$, where each $i \in V$
V_f	Set of q filler items, $V_f = \{j, , ..., q\}$, where each $j \in V$
i_t	Target item
L_i	Top-k recommendation list of item, i
τ	Popularity threshold
w_i	Weight of an item, i
p_i	The probability of a random user visits an item i in a web service.

4.1.2 Fake Co-Visitation Injection Attack

A fake co-visitation injection attack model is an approach of injecting a number of fake co-visitations between a target item and a set of anchor items into a co-visitation graph, based on the knowledge about the recommendation system, its co-visitation information, its threshold value τ, its items, and/or its users [4] (fig-4.1). Suppose, G is a co-visitation graph, which can be defined as $G = \{V, E, F\}$. The goal of co-visitation injection attack is to perform small perturbations on the graph G, leading to the graph $G' = (V', E', F')$, such that the recommendation system works as per the attacker's desire, where each attack can be accomplished by identifying four sets of items: a singleton target item i_t, a set of anchor items with particular characteristics determined by the attacker V_a, a set of filler items usually chosen randomly V_f, and the threshold value, τ.

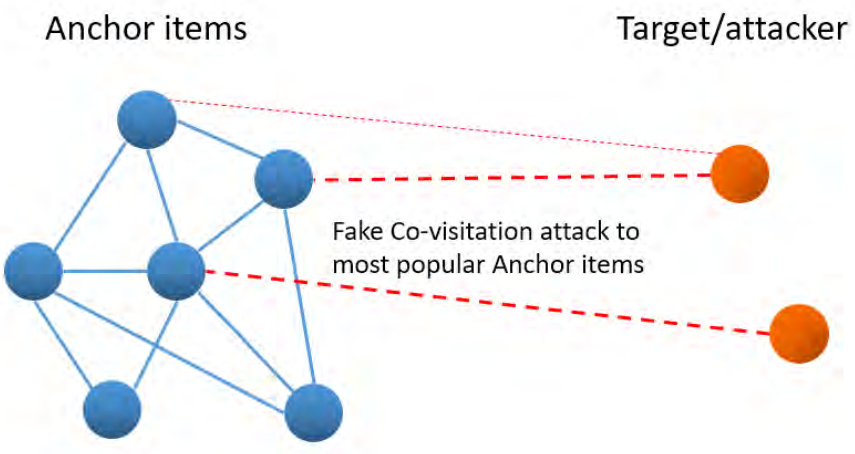


Figure 4.1: Fake Co-visitation Attack

Attack models can be defined by the methods by which they identify the selected anchor items, the proportion of the remaining items that are used as filler items, and the way of injecting fake co-visitation between the target item and each of these sets of items as defined by the function $f(i_t, V_a)$ and $g(i_t, V_f)$ respectively (fig-4.2). Here, $f(i_t, V_a)$ is the function for injecting fake co-visitation between target item i_t and each anchor item i, where $i \in V_a$. On the other hand, $g(i_t, V_f)$ is the function for injecting fake co-visitation between target item i_t and each filler item j, where $j \in V_f$.

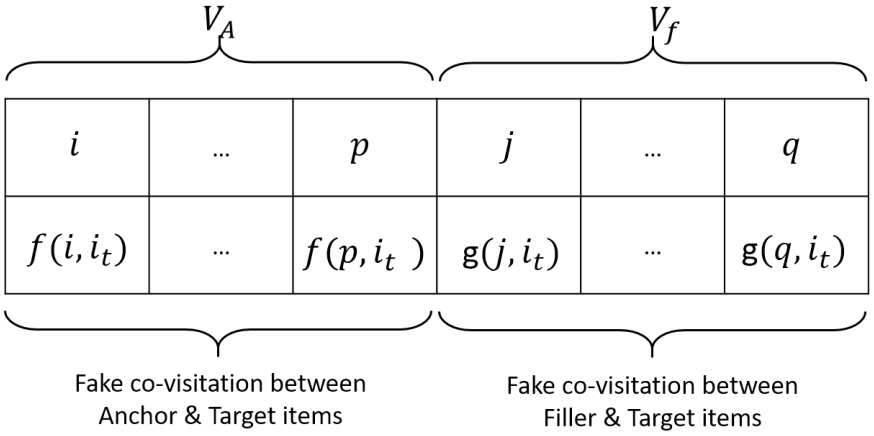


Figure 4.2: Attack Model.

4.1.3 Evaluation Metrics

4.1.3.1 Number of Items

The increased (promotion attack) or decreased (for demotion attack) number of anchor items whose top-k item-item recommendation list include the target item i_t can be used as one of the evaluation metric for measuring promotion attack and demotion attack. For example, the target item, i_t appears in the top-k recommendation list of 5 items before attack. But after promotion attack i_t appears in the top-k recommendation list of 20 items. So the promotion attack's threat is 15 items.

4.1.3.2 User Impressions

The former metric $numberofitems$ does not consider item popularities. But item's popularities have a great impact while measuring threat. If the i_t appears in the top-k recommendation list of a popular anchor item, then it will get more visits. Therefore, new metrics is proposed in [4] , which incorporate item popularities while evaluating threats of promotion and demotion attacks.

Top-k User Impression When a target item i_t appears in the item-item recommendation list of anchor item i, the target item is exposed to any user visits item i. In other words, for every user visit to v_i the target item i_t receives one user impression. The target item i_t will receive more user impressions if the item i receives more visitors in the future. More user impressions could result in more visits to the target item, which could result in more purchases .

The likelihood of converting a user impression into a visit or purchase is highly dependent on the target item's unique ranking position in the item-item recommendation list [4]. The highest-ranked item, for example, may have a higher visit rate than the lowest-ranked one. A user impression as a *top-k user impression* is defined in [4] if the target item is ranked top-k on the recommendation list, to account for the impact of item ranks in the recommendation list, where $k \leq n$.

Probability of Top-k User Impression Measuring *top-k user impressions* for a target item necessitates information of future visits to specific objects, which may not be available at the time of attacks. As a result, the probability of *top-k user impression (UI)* is proposed [4], which is the chance that a random user visits the target item and gets a *top-k user impression*. Suppose the probability of a random user visits an item i in a web service, is denoted by p_i. Let I_{i_t} be the set of items whose top-k recommended items include the target item i_t. Then the probability of top-k user impression of i_t is $UI = \sum_{i \in I_{i_t}} p_i$.

In power law phenomena, the probability p_i is related to the present popularity of the item i. Therefore, p_i is calculated in the following way:

$$p_i = \frac{w_i}{w_1 + w_2 + ... + w_n} \tag{4.1}$$

where n is the total number of instances and w_i is the previous popularity of item i. As a result, *UI (User impression)* metric can be formulated as:

$$UI = \frac{\sum_{i \in I_{i_t}} w_i}{w_1 + w_2 + ... + w_n} \tag{4.2}$$

Promotion Attack's Threat Suppose a target item i_t is in the top-k recommendation list in a group of items denoted as I_{i_t}, where $k \leq n$. This set of items is enlarged to set J_{i_t} after a promotion attack. The threat of promotion attack is defined as *increased probability of top-k user impression (IUI)* [4], which can be formally represented as,

$$IUI = \sum_{i \in J_{i_t} - I_{i_t}} p_i \tag{4.3}$$

Demotion Attack's Threat Assume the set of items whose top-k recommended items include the target item is reduced to be J_{i_t} after a demotion attack. The threat of this demotion attack as

decreased probability of top-k user impression (DUI) [4], which can be defined as follows:

$$DUI = \sum_{i \in I_{i_t} - J_{i_t}} p_i \tag{4.4}$$

With the knowledge and resources available, an attacker's purpose is to maximize the IUI or DUI for a target item.

4.2 Attack dimensions

Fake co-visitation injection attacks can be classified based on the level of knowledge required by the attacker, the attack's aim, and the scale of the attack. The ideal attack against a system, from the attacker's perspective, is one that has the most significant impact with the least amount of work [4]. While the knowledge and effort needed for a successful attack are significant aspects to consider, from a detection point of view, we are more interested to see how these factors combine to define the dimensions of an attack. From this perspective we are primarily interested in the following dimensions:

4.2.1 Attack Intent

The intent of an attack describes the intent of the attacker. Depending on the attacker's intention fake co-visitation injection attack can be further defined in terms of promotion attack and demotion attack. We will formally define these two in the following sections.

- **Promotion Attack** In promotion attacks, an attacker chooses a set of anchor items that do not contain the target item in the recommendation list. Then the attacker injects fake co-visitations between the target and each anchor item to promote the target item and increase user impression(IUI). The more user impression the target item gets, the more it has the chance of being purchased [4]. Fig.-4.3 depicts a promotion attack on co-visitation recommendation system.

- **Demotion Attack** In a demotion attack, an attacker selects a set of items as anchor items whose recommendation list contains the target item. Then the attacker injects fake co-visitations between the anchor and each item in the top-k recommendation list of anchor item, except the target item until the target item disappear from the top-k recommendation list and decrease the user impression (DUI) [4]. The less user impression the target item gets, the less it has the chance of being purchased.The demotion attack on co-visitation recommendation system is presented in fig.-4.4.

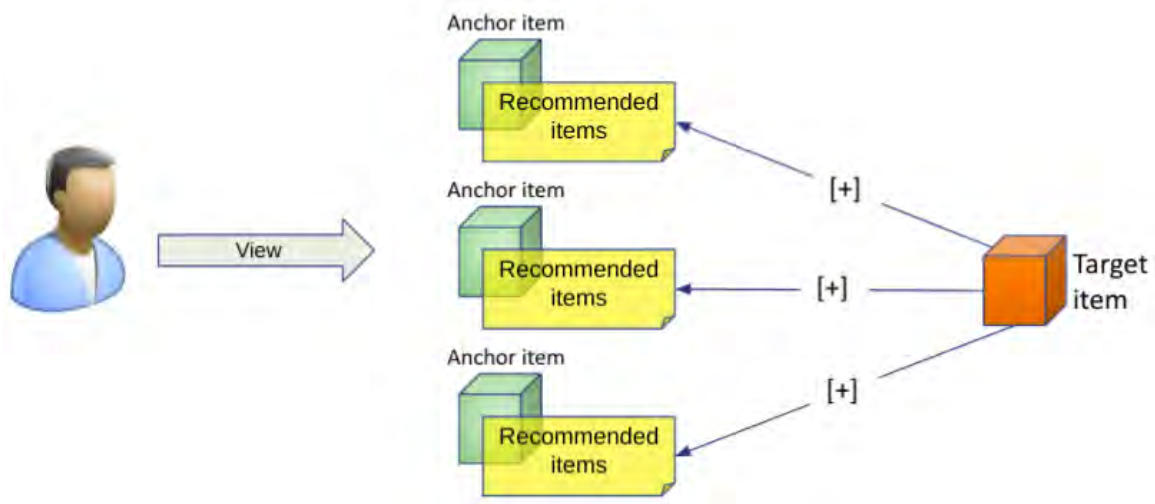


Figure 4.3: Promotion Attacks on graph based co-visitation recommendation system [4]

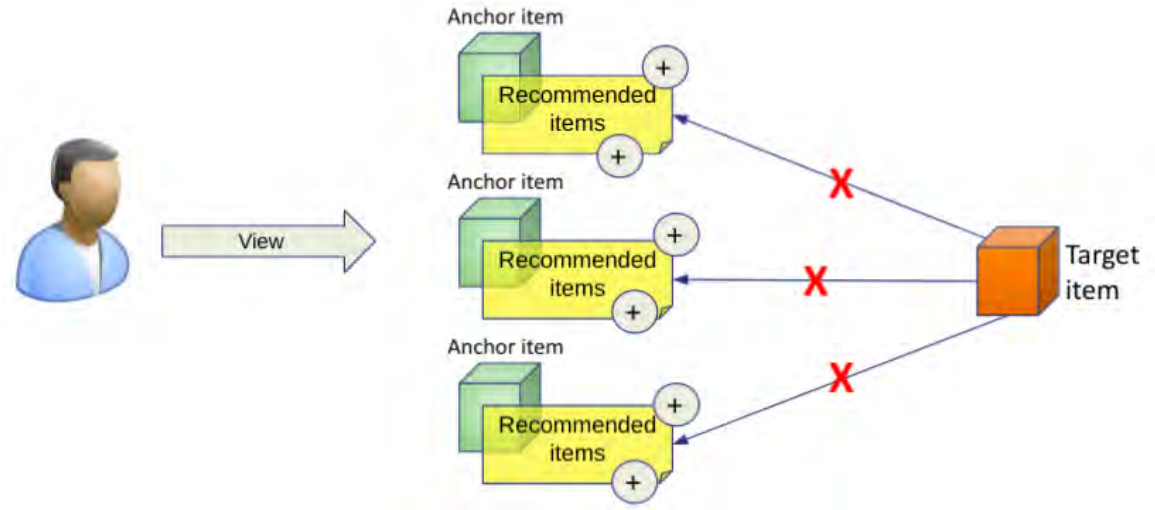


Figure 4.4: Demotion Attacks on graph based co-visitation recommendation system [4]

4.2.2 Attackerś Background Knowledge

Attacks may require some level of knowledge about the items, co-visitation graph, the popularity threshold τ and algorithms in the recommendation system being attacked. An informed attack will generally be more effective than an uninformed attack. The attack success rate is defined as the number of successfully attacked anchor items over the number of anchor items that are selected by our attacks. An anchor item is successfully attacked if the target item appears in its top-k recommendation list [4]. Formally, we define Attack Success Rate (AST) as:

$$\texttt{AST} = \frac{\texttt{\#successfully attacked anchor items}}{\texttt{\#selected anchor items}} \tag{4.5}$$

Attacks with high knowledge result in more significant threats (i.e. IUI for promotion and DUI for demotion) than attacks with medium knowledge, which result in more significant threats than attacks with low knowledge. However, even if just little knowledge is available, attacks can nonetheless produce threats that are about $1/3$ of those produced by attacks using high knowledge [4]. We consider three scenarios where attackers can access different background knowledge of the recommendation system.

- **High knowledge:** An attacker has access to the co-visitation graph G and the popularity threshold τ, which are utilized to tweak the top-k recommended items in this case. Because the attacker is familiar with the essential components of the co-visitation recommendation system, this constitutes a strong attacker. An attacker might either buy this information from a web service insider on the black market or be a web service insider himself. This scenario represents an upper bound of the threats introduced by fake co-visitation injection attacks. We represent high knowledge as a pair (G, τ) [4].

- **Medium knowledge:** In this scenario, an attacker creates a web crawler to collect various things from the web service, as well as their item-item top recommendation lists. Some online services frequently make item-item recommendation lists public, allowing unlogged-in visitors to see them as well. As a result, an attacker has the ability to collect these recommendation lists. Suppose, L_i is the item-item top-k recommendation list of an item i. L is the set of p items whose recommendation lists are collected by the attacker, $L = \{L_1, L_2, ..., L_p\}$. Some web services show items' popularity to users/visitors, and thus an attacker has access to items' popularities. For example, YouTube displays the amount of views (i.e., popularity) of a video to visitors. We express the popularities of all the things that the attacker encountered during collecting L by a set $W = \{(i, w_i)|i \in V\}$, where the set V consists of all the items that the attacker encountered, and w_i is the popularity of item i. We represent medium knowledge as a pair (L, W) [4].

- **Low knowledge:**

  In this scenario, an attacker writes a crawler to collect $L = \{L_1, L_2, ..., L_p\}$, the item-item top-$k$ recommendation lists of p items. We presume, however, that the service does not give item popularity statistics, and that the attacker does not have access to them. This scenario illustrates how even the tiniest piece of information from a co-visitation recommendation system can be leaked to an attacker. Amazon and eBay, for example, are examples of this type of service [4].

4.2.3 Co-visitation graph's structure

In [4], the authors claimed that, Graph structures have relatively small impact on co-visitation attacks.

4.2.4 Attacker's resources and k

Attacker's resources and k have a significant impact on co-visitation attacks. With the increasing number of resources and k, the attacker can inject more fake co-visitations between the target item and a set of anchor items which results in more successful attacks [4].

4.2.5 Anchor Item Selection

An attacker may also perform simple attacks on co-visitation recommendation systems, e.g., inject co-visitations with randomly selected anchor items. The comparison of promotion attacks with two baseline attacks are mentioned below:

- **`Popular Item`:** This attack injects co-visitations between a target item and the most popular item until the target item appears in its top-k recommendation list and then attacks the next most popular item until there are no more fake co-visitations to inject.
- **`Random Item`:** This attack randomly selects an anchor item, injects co-visitations until the target item appears in its top-k recommendation list. This process is repeated until no more fake co-visitations to inject.

4.3 Attack Scenario

In this section, we will discuss about how the attacks are performed in real world.

4.3.1 Promotion Attack

In promotion attack, at first the attacker selects the target item and a set of anchor items. Then he/she determines the inject-able fake co-visitations based on the resource and k. Next, the attacker selects an anchor item from the set of anchor items and check whether the target item is already in the list or not. If the target item is not in the list, then the attacker injects a fake co-visitation between the target and the anchor item. The number of remaining inject-able co-visitations gets updated accordingly. After injecting fake co-visitations the attacker checks if the target is in the recommendation list or not. If it appears in the list, then the attacker selects the next anchor item from the list and injects fake co-visitations. Otherwise, he will inject fake co-visitations as long as the number of remaining inject-able co-visitations is greater than 0 and the target item is not in the recommendation list. Fig-4.5 represents the flow chart of promotion attack.

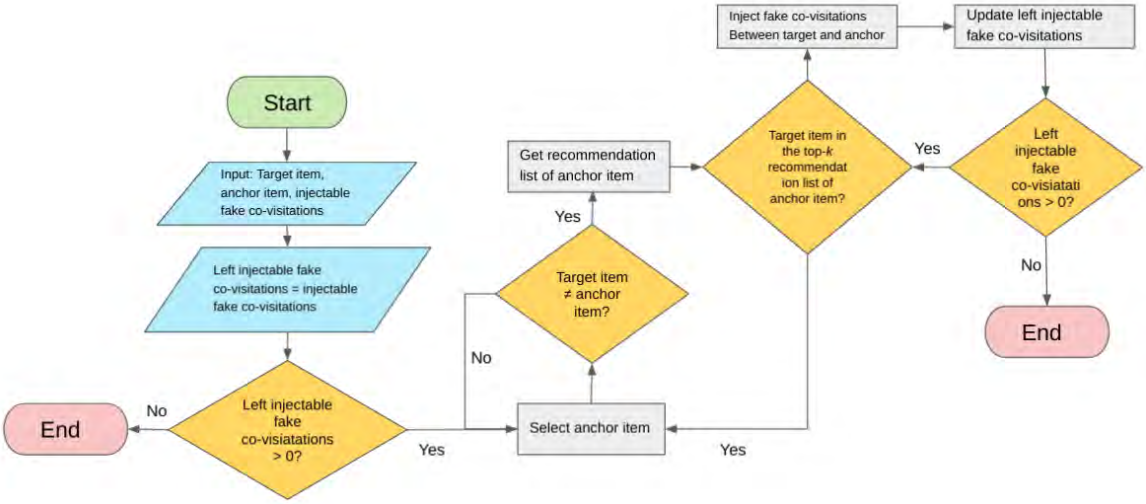


Figure 4.5: Promotion Attack Flow chart

4.3.2 Demotion Attack

On the other hand in demotion attack, the attacker selects the target item and a set of anchor items whose recommendation list does not contain the target item. Then he determines the inject-able fake co-visitations based on the resource and k. Next, he selects an anchor item from the set of anchor items and check whether the target item is in the list or not. If the target item is in the list, then the attacker injects a fake co-visitation between anchor and $(k+1)$th item. The number of remaining inject-able co-visitations gets updated accordingly. After injecting fake co-visitations the attacker check if the target is in the recommendation list or not. If it does not appear in the list, the attacker selects the next anchor item from the list and inject fake co-visitations. Otherwise, he will inject fake co-visitations as long as the number of left inject-able co-visitations is greater than 0 and the target item is in the recommendation list. Fig-4.6 represents the flow chart of demotion attack.

4.4 Discussion

In this research, we only exploit promotion and demotion fake co-visitation injection attack with high knowledge as it generates the upper bound of the attack. We also select the most popular item as anchor item since the target item i_t gets the maximum user impression (UI) if it appears on the recommendation list of a popular anchor item.

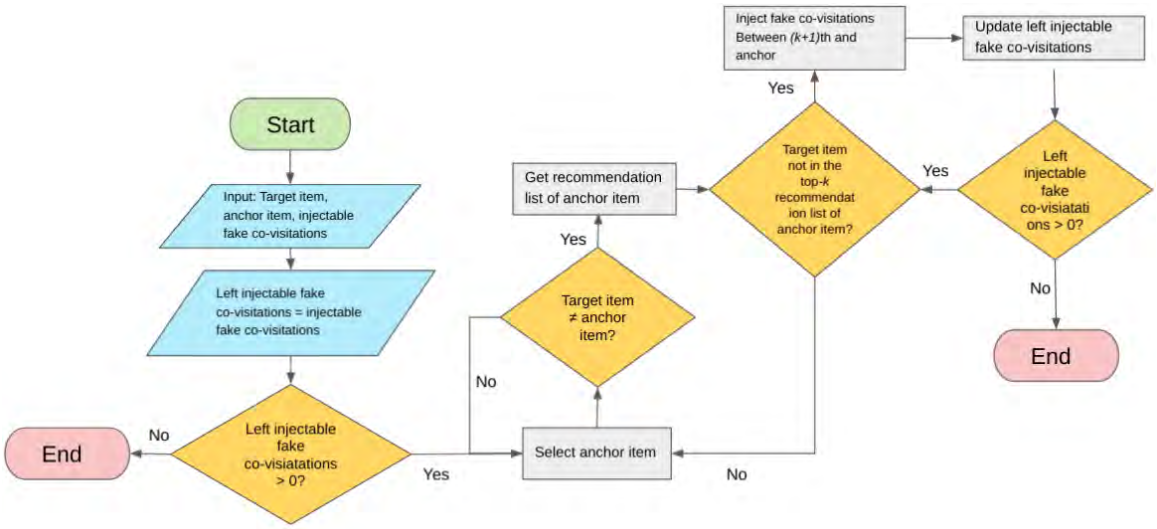


Figure 4.6: Demotion Attack Flow chart

Chapter 5

Proposed Methodology

In this chapter we describe the major components of our proposed approach. Firstly, the co-visitation graph generation process is discussed along with different co-visitation injection attacks. After that, the co-visitation graph is analyzed and node attributes are calculated. In order to detect the suspicious node from the co-visitation graph we develop a framework which incorporates attribute representations of nodes. Then a detection algorithm is formally presented and abnormality forensics are briefly discussed. Lastly, we compare the detection rate and false alarm rate of our proposed detection approach with the state-of-art detection techniques.

The main stages of the presented framework can be briefly described as follows:

1. **Constructing Co-visitation graph:** Based on attacker's background knowledge and intention, different co-visitation attacking scenarios are considered to conduct the attack on both real-world and synthetic dataset. Then, the item-to-item co-visitation graph is constructed. Converting the real world user-item rating dataset into a co-visitation graph is a challenging task.

2. **Determining node attributes:** Similar to [7], we incorporate attribute representations of the nodes derived from topological information along with node and edge weights.

3. **Identifying and Detecting Co-visitation Injection Attacks:** Finally, suspicious nodes are detected by using ANOMALOUS [18] which uses CUR decomposition and residual analysis to score item's abnormality.

The details of each stage is described in the following sections. A basic overview of our outlier node detection stages are shown in 5.1. Table-5.1 gives the principal notations used in this chapter.

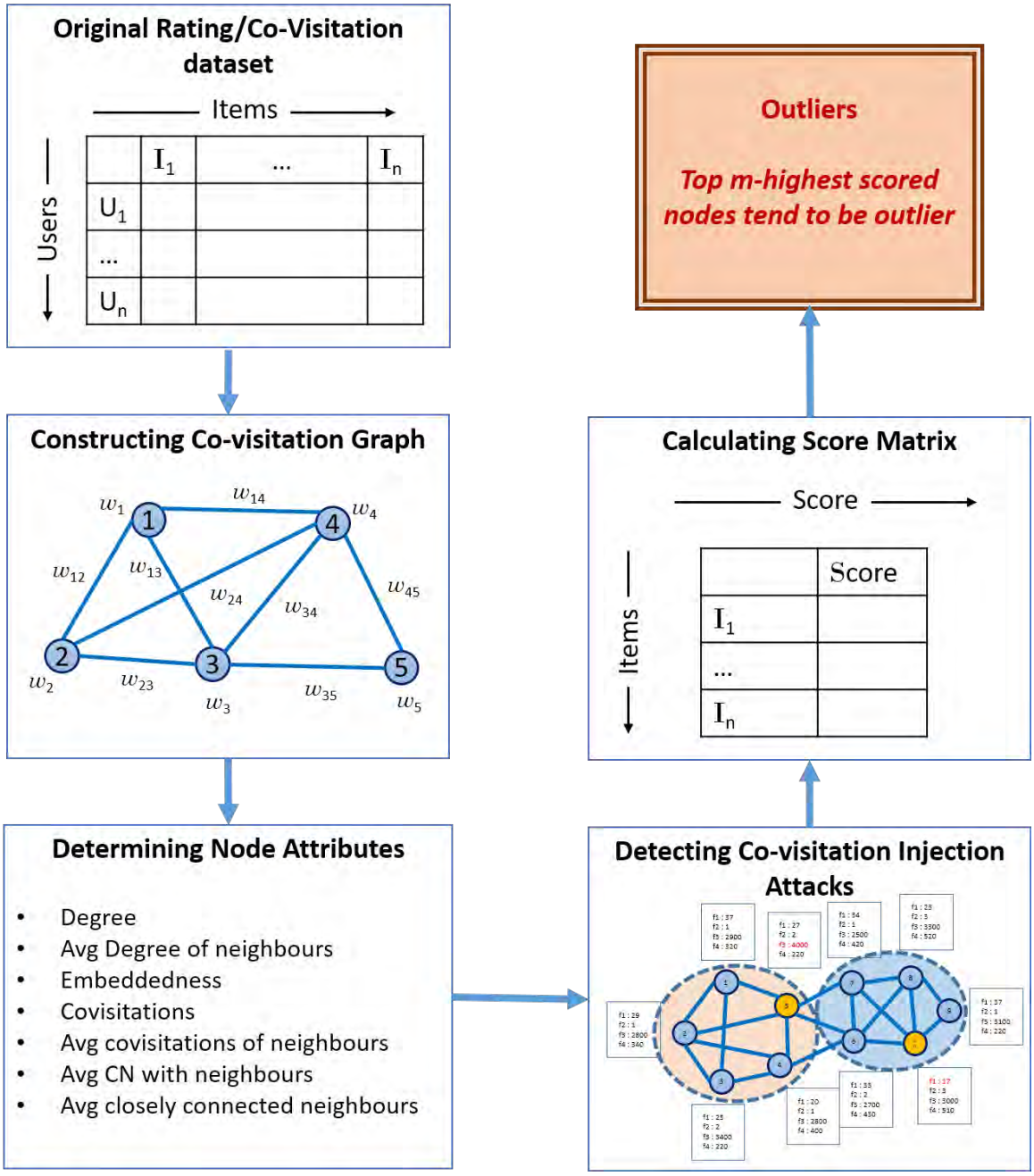


Figure 5.1: Proposed Detection Framework

5.1 Constructing Co-visitation graph

Let , $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ is a co-visitation graph shown in fig.-5.2, where $\mathbf{V} = \{1, 2, ..., n\}$ is the set of n network nodes, $\mathbf{E}$ is a set of m network edges where, edge $e_{ij} \in \mathbf{E}$ means that item i and j were co-visited by at least one user and they are connected by an edge. Each edge has a weight w_{ij} refers to the number of times the two nodes i and j were co-visited and each node has weight w_i which refers to the total number of times the node i is visited.

By analysing linkage and structural attributes of the graph we can transform the co-visitation

Table 5.1: Attributed Networks: Notations.

Notation	Description
V	Set of n instances, $V = \{1, 2, 3, ..., n\}$
E	Set of m edges, $E = \{e_1, e_2, ..., e_m\}$
F	Set of d-dimensional attributes, $F = \{f_1, f_2, ..., f_d\}$ associated with each instance
$N(i)$	Set of neighbors of item, i
k	Number of most similar items of item i showing on it's recommendation list.
V_a	Set of p anchor items, $V_a = \{i, ..., p\}$, where each $i \in V$
V_f	Set of q filler items, $V_f = \{j, , ..., q\}$, where each $j \in V$
i_t	Target item
M_c	Co-visitation matrix. Note that, $M_c(u, j)$ is m_{u_j} (the number of fake co-visitations between nodes u and j) if the node u is an anchor, 0 otherwise.
L_i	Top-k recommendation list of item, i
τ	Popularity threshold
w_i	Weight of an item, i
w_{ij}	Co-visitation weight of an edge, e_{ij} between items i and j
S_{ij}	The similarity between items i and j
p_i	The probability of a random user visits an item i in a web service.
X	Attribute information of all n instances, $X \in \mathbf{R}^{d \times n}$
A	Adjacency matrix, $A \in \mathbf{R}^{n \times n}$
M	Residual matrix of node attributes
$\|A\|_{2,1}$	$l_{2,1}$ norm of matrix $A \in \mathbf{R}^{d \times n}$ is written as $\|A\|_{2,1} = \sum_{i=1}^{d} \sqrt{\sum_{j=1}^{n} A(i,j)^2}$
$\|A\|_F$	Frobenius norm of matrix $A \in \mathbf{R}^{d \times n}$ is written as $\|A\|_F = \sqrt{\sum_{i=1}^{d} \sum_{j=1}^{n} A(i,j)^2}$

graph into an attributed graph ,$\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{F})$, where $\mathbf{F} = \{f_1, f_2, ...f_d\}$ is a set of d attributes, which is associated with each node represented in fig.-5.3. From this network we can construct an adjacency matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$ to describe the linkage relationships between them, where $\mathbf{A}$(i, j) = 1 indicates items i and j are connected with each other. We also construct a attributed matrix $\mathbf{X} \in \mathbf{R}^{n \times d}$ to represent the attribute information of all n instances, where $\mathbf{X}(i, :) \in \mathbf{R}^d$ denotes the attribute information of item i. Fig.-5.4 depicts the adjacency and attributed matrix of graph G.

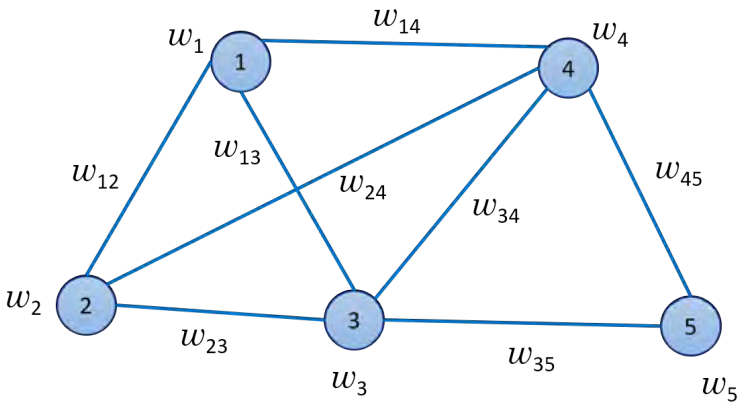


Figure 5.2: Co-visitation Graph

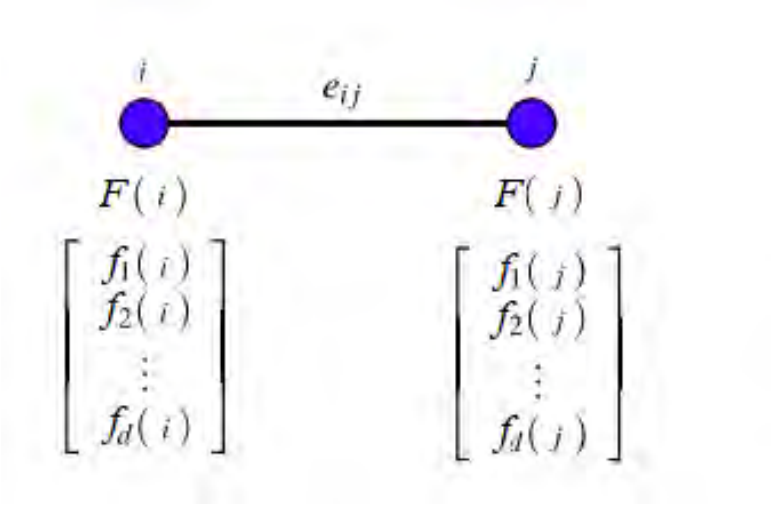


Figure 5.3: Notation related to the triple (node-attributed graph)

5.1.1 Recommendation Process

In a co-visitation recommendation system, it is expected that if two items are co-visited in the past they will also be co-visited in the future. So considering the edge co-visitation weight a similarity measure is introduced between two connected nodes.This similarity measure is used to make the recommendation. We here only consider the item-to-item recommendation. When an user visits an item i, the recommendation system will computes the similarity between i and j, where $j \in N(i)$ and recommend the top-k most similar items of i to the user, whose popularity cross a threshold value, τ . The similarity s_{ij} between item i and item j is calculated as following equation from [24]:

$$S_{ij} = \frac{w_{ij}}{g(w_i, w_j)} \tag{5.1}$$

where $g(w_i, w_j)$ is a function of w_i and w_j. Different web services might use different such

	Adjacency Matrix, A				
	1	2	3	4	5
1	0	1	1	1	0
2	1	0	1	1	0
3	1	1	0	1	1
4	1	1	1	0	1
5	0	0	1	1	0

	Attribute Matrix, X				
	f1	f2	..	..	fd
1	0	1	1	1	0
2	1	0	1	1	0
3	1	1	0	1	1
4	1	1	1	0	1
5	0	0	1	1	0

Figure 5.4: Adjacency matrix and attributed matrix of the co-visitation graph.

functions. For instance, YouTube [24] uses $g(w_i, w_j) = w_i.w_j$. Amazon [6] uses Cosine Similarity between the view vectors (i.e.,an entry of the vector is 1 if the corresponding user viewed the corresponding item, otherwise the entry is 0) of two items as their similarity. Since the entries of the view vectors have binary values, Cosine Similarity between two items of equation 5.1 is reduced to,

$$S_{ij} = \frac{w_{ij}}{\sqrt{w_i.w_j}} \tag{5.2}$$

Given an item i that a user is visiting, the system first ranks the items using their similarities with i and then recommends the top-k items with the most prominent similarities to the user. We denote the top-k recommended items for the item i as a sorted list L_i. Note that this item-item recommendation method favors unpopular items. Specifically, an item with small popularity is more likely to be recommended than an item with large popularity if they have the same number of co-visitations with the item i.

5.1.2 Performing Attack

Several attack scenarios can be generated based on the attacker's background knowledge and intention. In this research, we only exploit promotion and demotion co-visitation injection attack with high knowledge as it generates the upper bound of the attack [4].

5.1.2.1 Promotion Attack with High Knowledge

In promotion attacks, an attacker selects a set of items as anchor items whose top-k recommendation list do not include the target item i_t. Then the attacker injects fake co-visitations between the target item and each anchor item to make the target item appear in its top-k recommendation list [4].

Suppose $V_a = (j, ..., p)$ is the set of anchor items whose top-k recommendation list do not have the target item i_t in it. For an anchor item j where $j \in V_a$ and L_j is the top-k recommendation list. Let, k_j is the k-th ranked item in L_j. In order to make the target item i_t appear in the top-k recommendation list of j, the attacker needs to inject m_{jk} fake co-visitations between the items i_t and j, where m_{jk} satisfies two conditions:

$$s'_{ji_t} > s'_{jk_j}, \forall j \in \vee_a \tag{5.3}$$

$$w_{i_t} + m_{jk} \geq \tau, \forall j \in \vee_a \tag{5.4}$$

where s'_{ji_t} is the similarity between j and the target item i_t and , s'_{jk_j} is the similarity between j and the k-th ranked item k_j after the attack. From Equation 5.1, we can define $s'_{ji_t} = (w_{ji_t} + m_{jk})/f(w_j + m_{jk}, w_{i_t} + m_{jk})$ and $s'_{jk_j} = w_{jk_j}/f(w_j + m_{jk}, w_{k_j})$. Here, w_{ji_t} is the number of co-visitations between j and i_t, w_j is j's popularity, w_{i_t} is i_t's popularity before the attack, and the function f is the normalization factor.

The attacker's intention is to maximize the threat of promotion attack (IUI, increased probability of top-k user impression) [4]. The promotion attack can be formulated as an optimization problem:

$$\texttt{max IUI} = \sum_{j \in V_a} a_j.p_j \tag{5.5}$$

$$\texttt{subject to} \sum_{j \in V_a} a_j.m_{jk} \leq m \tag{5.6}$$

where $a_j \in \{0, 1\}, \forall_j \in V_a$ represents whether the item j is selected as an anchor item or not, and $p_j = \frac{w_j}{w_1+w_2+\cdots+w_n}$ denotes the probability that a random user visits an item j and n is the total number of items.

5.1.2.2 Demotion Attack with High Knowledge

In demotion attack, an attacker selects a set of items as anchor items, whose top-k recommendation list contain the target item, i_t. The aim is to remove the target item from the recommendation list so that the user impression of target item's get decreased [4].

Suppose, j is the anchor item whose top-k recommendation list,L_j includes target item i_t as the u-th ranked item where $(u \leq k)$. We remove the target item from the recommendation list of the selected anchor item by improving the ranking of the $(u+1)$th, $(u+2)$th, ..., $(k+1)$th ranked items until i_t is not on the top-k recommendation list. So demotion attack can be viewed as promotion attack by considering $(u+1)$th, $(u+2)$th, ..., $(k+1)$th ranked items as target items. Therefore, the demotion attack can be formulated as the following optimization problem,

$$\texttt{max DUI} = \sum_{j \in V_a} a_j.p_j \tag{5.7}$$

$$\texttt{subject to} \sum_{j \in V_a} a_j . \sum_{x=u+1}^{k+1} m_{jx} \leq m \tag{5.8}$$

$$\min_{x=u+1}^{k+1} s'_{jx} > s'_{ji_t}, \forall j \in \vee_a \tag{5.9}$$

$$\min_{x=u+1}^{k+1} \{w_x + m_{jx}\} \geq \tau, \forall j \in \vee_a \tag{5.10}$$

$$a_j \in \{0, 1\}, \forall j \in \vee_a \tag{5.11}$$

where $\forall_j \in V_a$. Here V_a denotes the set of anchor items that contain i_t in their top-k recommendation lists. DUI stands for decreased probability of top-k user impression.

5.2 Determining Node Attributes

Similar to [7,8], node attributes are derived from the topological information along with node and edge weights. Finding a good attribute representation of nodes which can preserve the structure of the graph and the connection between individual nodes is challenging. While having higher number of dimensions may require high time and space complexity, lower number of dimensions may result in better detection accuracy. The proposed attributes are listed below:

Degree centrality of a node i ($f1$): The degree of a node i is the number of edges connected to the node [7]. It can be defined as:

$$d_i = \sum_{j=1}^{n} A(i, j) \tag{5.12}$$

where n denotes the number of all nodes in graph G. A is an adjacency matrix converted from G. Note that, $A(i, j)$ is 1 if nodes i and j are interconnected, 0 otherwise.

Average degree of neighbors of a node i ($f2$): Average degree of neighbours of node i can be simply defined as the average number of edges per neighbour node has in the graph. For a given node i, it is advantageous to capture co-visitation characterizations between the linking nodes by measuring the degree centrality of its neighbors [7]. The average degree of neighbors of i is defined as follows to characterize the influence of neighbor nodes,

$$\overline{\mathrm{d}_{\mathrm{N_i}}} = \frac{1}{|N(i)|} \sum_{v=N(i)} \sum_{j=1}^{n} A(v, j) \tag{5.13}$$

where $N(i)$ is the set of neighbors of node i.

Embeddedness ($f3$): The degree to which a node is intertwined in a network [7], is defined as follows,

$$E_{m_i} = \frac{1}{|N(i)|} \sum_{j=N(i)} \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \tag{5.14}$$

where $N(i)$ and $N(j)$ respectively represent the set of neighbors of nodes i and j.

The number of co-visitations of a node i ($f4$): On average, the number of sybil node co-visits on the same anchor is higher than the number of benign node co-visits. Attackers' co-visited behaviors are indicated by the number of co-visitations of a node [7], which can be defined below,

$$NC_i = \sum_{j=1}^{n} M_c(i,j) \tag{5.15}$$

where M_c is a co-visitation matrix converted from G. Note that, $M_c(i,j)$ is m_{ij} (the number of fake co-visitations or co-ratings between nodes i and j) if the node i is an anchor, 0 otherwise.

The average number of co-visitations of neighbors of a node i ($f5$): In response to promotion or demotion assaults, a co-visitation graph is produced using the ErdosRenyi (ER) random rule [4]. The attackers inject enough fake co-visits or co-ratings between anchor nodes and target nodes to achieve their goal. Due to the target node of assaults, co-visits or co-ratings between anchor nodes may not be sufficient. The anchor nodes can be simply estimated as neighbors for each other based on their consistent promotion or demotion behavior. Calculating the average number of co-visitations of a node's neighbors is a good way to define the difference between sybil and benign nodes in order to capture group attack behaviors, which can be defined as follows,

$$\overline{NC_{N_i}} = \frac{1}{|N(i)|} \sum_{v=N(i)} \sum_{j=1}^{n} M_c(v,j) \tag{5.16}$$

where $N(i)$ is the neighbors of i.

The average number of common nodes between a node i and neighbours ($f6$): Generally, two similar nodes may have many common neighbors. But the number of common neighbors will drop drastically between sybil and benign nodes because they are less similar.

$$\overline{N_c(i, N(i))} = \frac{1}{|N(i)|} \sum_{j=N(i)} |N(i) \cap N(j)| \tag{5.17}$$

The average closely connected neighbors of a node i ($f7$): Two nodes can be said as closely connected if their neighbors have common neighbors. This can be defined as follows,

$$\begin{aligned} &\overline{N_c(N(p), N(q))} = \tfrac{1}{|N(i)|} \textstyle\sum_{p,q \epsilon N(i); p<q} Q_{pq}, \\ &Q_{pq} = \begin{cases} 1, & \text{if } |N(p) \cap N(q)| > 1 \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{5.18}$$

5.3 Proposed Detection Approach

The aim of anomaly detection is to spot rare, unexpected and suspicious instances that significantly deviate from the patterns of majority in datasets [77, 78]. To detect anomaly in an attributed network requires seamless integration of network structure information and attribute information. In an attributed network like co-visitation graph, edges represent the connection between data instances and each node has additional attributes. But not all these attributes are strongly hinged to the network structure. These noisy and structurally irrelevant attributes can cause adverse effects on anomaly detection. Therefore, selecting representative attributes that are closely hinged with the network structure is essential for anomaly detection.

In Figure 5.5 as an example, we are most likely to regard each node as an anomaly when doing anomaly detection with all four attributes since each node does not conform to the patterns of the majority. Nonetheless, if we simply consider the attributes f1 and f3, it is clear that node 5 is a community outlier, as the value of its third characteristic (f3) is significantly greater than the values of the other nodes in the same community [79]. Meanwhile, node 10 is an outlier since the value of its first characteristic (f1) differs greatly from the values of its structurally related nodes. This phenomena demonstrates that not all attributes are dependent on the network topology; for example, the binary attribute f2 and the numerical variable f4 are almost randomly assigned values. These noisy and unnecessary features obstruct our ability to capture the relationship between network structure and node properties, perhaps obscuring the underlying anomalies node 5 and node 10.

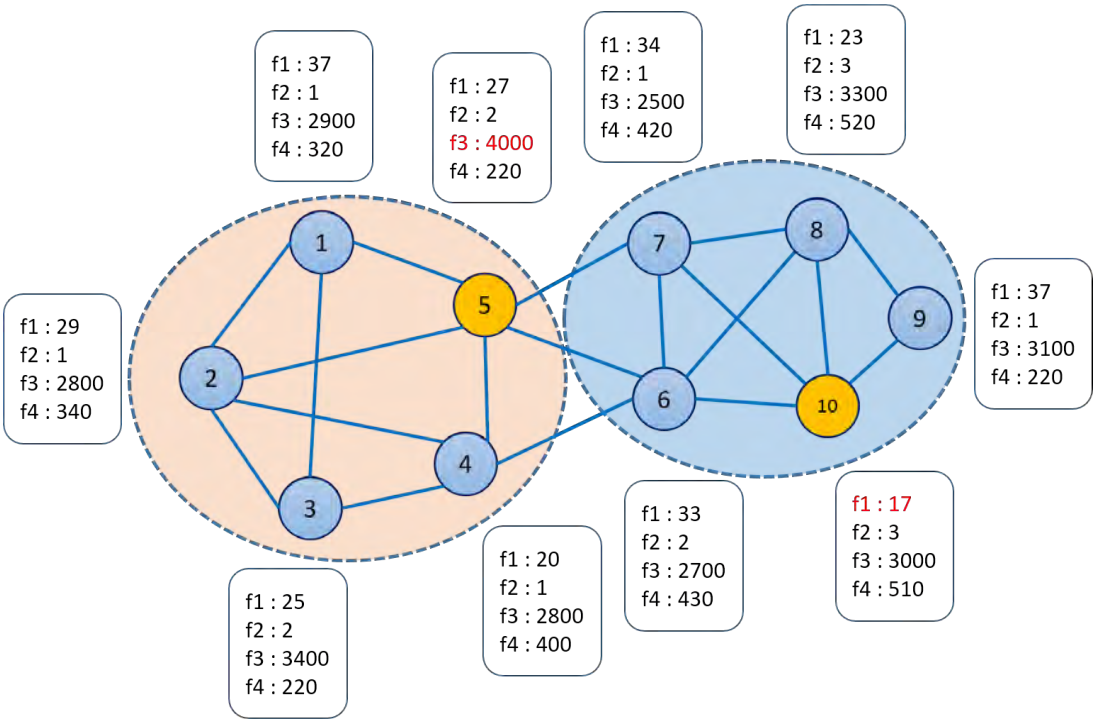


Figure 5.5: Anomaly Detection

In this work, a joint modeling approach for anomaly detection on attributed networks, called ANOMALOUS [18] is applied . ANOMALOUS selects a subset of representative instances on the space of representative attributes that are closely hinged with the network topology based on CUR decomposition and then measures the normality of each instance via residual analysis.

First of all, residual analysis to find out residuals between estimated data and true data is used to spot anomalies in attributed graph as anomalies usually have large residual errors [76]. The reason behind this large residual error is that anomalous instances deviate significantly from majority of reference instances in pattern. True data, X is constructed by selecting structure related attributes and representative instances. Subsequently CUR decomposition is used to select a subset of representative instances on the space of representative attributes that are closely hinged with the network topology. The objective function can be mathematically formulated below:

$$\min_{\mathbf{C},\mathbf{U},\mathbf{R},\mathbf{M}} \|\mathbf{X} - \mathbf{CUR} - \mathbf{M}\|_F^2 + \Psi(\mathbf{M}, \gamma, \varphi) \tag{5.19}$$

where $\mathbf{C} \in \mathbb{R}^{d\times m}$ is a subset of m columns of $\mathbf{X}$, $\mathbf{R} \in \mathbb{R}^{r\times n}$ is a subset of r rows of $\mathbf{X}$, $\mathbf{U} \in \mathbb{R}^{m\times r}$ with m, r and $rank(\mathbf{U})$ as small as possible, the residual matrix of node attributes is denoted by $\mathbf{M}$ and regularization term on $\mathbf{M}$ is Ψ. From Equation 5.19 we see that matrix C and R are generated by selecting m instances and r attributes respectively from the original data and attribute space so that it can approximate matrix X as closely as possible.

To limit the number of anomalies and to control the column sparsity of the residual matrix $\mathbf{M}$, Equation 5.19 can be reformulated as:

$$\min_{\mathbf{C},\mathbf{U},\mathbf{R},\mathbf{M}} \|\mathbf{X} - \mathbf{CUR} - \mathbf{M}\|_F^2 + \gamma \left\|\mathbf{M}^T\right\|_{2,0} + \varphi \operatorname{tr}\left(\mathbf{MLM}^T\right) \tag{5.20}$$

where Laplacian matrix, $\mathbf{L}$ is generated by the adjacency matrix $\mathbf{A}$ of networks.

In addition to the attribute information of the network, structure information is also needed to be incorporated for detecting anomalies [18]. Therefore, Based on Homophily [80–82], if two instances are connected in the network, their attribute patterns in the residual matrix $\mathbf{M}$ ought to be similar after attribute reconstruction. Hence, network structure modeling is achieved by minimizing the following terms:

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\mathbf{M}^T(i,:) - \mathbf{M}^T(j,:)\right)^2 \mathbf{A}\,(i,j) = \operatorname{tr}\left(\mathbf{MLM}^T\right) \tag{5.21}$$

From Equation 5.20, the term $\varphi \operatorname{tr}\left(\mathbf{MLM}^T\right)$ is used to model the correlation between network structure and node attributes and the contribution of the modeling of this correlation is controlled by parameter φ [18].

We further reformulate the objective function of Equation 5.20 of framework ANOMALOUS [18] by replacing the general CUR model and adding α and β which respectively controls the row

sparsity and the column sparsity of matrix $\mathbf{W}$, that selects representative instances and attributes simultaneously-

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{M}} \|\mathbf{X} - \mathbf{XWX} - \mathbf{M}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1} + \beta \left\|\mathbf{W}^T\right\|_{2,1} \\ &+\gamma \left\|\mathbf{M}^T\right\|_{2,1} + \varphi \operatorname{tr}\left(\mathbf{MLM}^T\right) \end{aligned} \tag{5.22}$$

Anomalous nodes can be ranked according to residual errors, by analyzing the residual matrix $\mathbf{M}$ [18]. As aforementioned, the residual error is generally much higher for the anomalous nodes, we empirically consider top m ranked nodes to be outlier nodes via analyzing the results from numerous experiments.

The key steps for detecting anomalies on representative attributes via CUR decomposition and residual analysis of framework ANOMALOUS [18] are summarized in algorithm 3. From we algorithm we can estimate the time complexity of the algorithm. The most costly operation at per iteration is the matrix inverse operation requiring $O(n^3)$ caused by matrix update. Fortunately, the update of $\mathbf{M}$ can be sped up by the linear equation system, which only need $O(n^2d)$ (n is usually greater than d). Similarly, the update operation of $\mathbf{W}$ requires $O(n^2)$ which can be accelerated through calculating each column of $\mathbf{Y}$ in parallel. Consequently, the total time complexity is $\#iterations * (O(n^2d) + O(n^2))$, i.e., $\#iterations * O(n^2d)$ [18].

The basic process of our proposed framework for detecting suspicious nodes in co-visitation graph has shown in fig.-5.1. It is noteworthy that in order to characterize basic connection patterns of nodes in co-visitation graph, all presented attributes are implemented. The top t instances with the highest abnormal scores need to be empirically determined for determining suspicious nodes. So, how to determine a reasonable t should be empirically analyzed according to a list of experiments.

Algorithm 3: ANOMALOUS: A joint modeling approach for anomaly detection on attributed networks [18].

Input : Adjacency matrix $\mathbf{A}$, attribute matrix $\mathbf{X}$, parameters $\alpha, \beta, \gamma, \varphi$.
Output : Top t instances with the highest abnormal scores.

1. Build Laplacian matrix $\mathbf{L}$ from the adjacency matrix $\mathbf{A}$;
2. Initialize $\mathbf{D_R}$, $\mathbf{D_W}'$, $\mathbf{D_W}''$ to be identity matrix;
3. Initialize $\mathbf{M} = \mathbf{X}(\mathbf{I} + \gamma\mathbf{D_R} + \varphi\mathbf{L})^{-1}$.
4. Build orthogonal matrix $\mathbf{P}, \mathbf{Q}$ and diagonal matrix Θ_1, Θ_2 based on Lemma 2 [18].
5. **while** objective function in Eq.-5.22 not converge **do**
6. Update $\mathbf{W}$ by using theorem-1 [18];
7. Update $\mathbf{D_W}'$ by setting $\mathbf{D_W}'(i,i) = \frac{1}{2\|\mathbf{W(i,:)}\|_2}$.
8. Update $\mathbf{D_W}''$ by setting $\mathbf{D_W}''(i,i) = \frac{1}{2\|\mathbf{W^T(i,:)}\|_2}$;
9. Update $\mathbf{M}$, where $\mathbf{M} = (\mathbf{X} - \mathbf{XWX})(\mathbf{I} + \gamma\mathbf{D_R} + \varphi\mathbf{L})^{-1}$;
10. Update $\mathbf{D_R}$ by setting $\mathbf{D_R}(i,i) = \frac{1}{2\|\mathbf{M^T(i,:)}\|_2}$;
11. **end while**
12. Compute the abnormal score for the i-th instance as $\|\mathbf{M}(:,\mathbf{i})\|_2$
13. Output top t instances with the highest abnormal scores.

Chapter 6

Performance Evaluation

In this chapter, we investigate the proposed approach on both synthetic and real-world datasets, evaluate the DR and FAR in comparison with the baselines and show the efficacy of our proposed approach.

6.1 Experimental Setup

6.1.1 Experimental Dataset

To conduct the experiment we use synthetic data (labeled) and real-world data (unlabeled). Although discovering anomalous nodes on real-world dataset is the primary target of our proposed detection approach but Synthetic dataset gives us good indication of our proposed technique's performance.

6.1.1.1 Synthetic Data

We use Erdos-Renyi (ER) random graph model [4] to generate our experimental co-visitation graph. In ER graph, we start with n vertices and no edges, and at each step we add one new edge chosen uniformly with probability $p = 0.15$. We generate a co-visitation graph with $1k$ nodes and $140k$ edges. Each edge has co-visitation weight ranging from 1 to $2.5k$, where each node has a popularity of at most $200k$. Since the most popular items tend to connect with more items, we assign higher popularity to the node having higher degree such that the popularity is greater than the summation of weights of all the edges connected to it. As a result, an item has more popularity if it possesses a higher degree in the co-visitation graph. Empirically, for our experiment we set the popularity threshold, $\tau = 500$, which means items with popularity below 500 will not be shown in the recommendation list of any item. We generate the attacked data by applying promotion and demotion attack separately with high knowledge attack model to the above mentioned ER Graph. We assume, $k = 5$ which refers to the scenario for promotion attack

Table 6.1: Dataset

Synthetic	Number of Nodes	1000
	Number of Edges	140000
	Popularity	200000
	Fake Co-visitation	2400
Real World [29]	Number of Nodes	1682
	Number of Edges	102189
	Fake Co-visitation	500

Table 6.2: Read-world Dataset

MovieLens 100K [29]	5k	102	5023
	30k	502	32122
	70k	1002	78822
	100k	1682	2400

where attacker wants the target item to be in the top-5 recommendation list and for demotion attack, attacker wants the target item to be out of top-5 recommendation list of the anchor item in where the target node already exists. To make the attack more camouflaged, we inject some co-visitations between target and filler items with a minimum probability of $p = 0.002$. We assume here that the attacker can inject at most $2.4k$ fake co-visitations in the co-visitation graph resulting in a significant impact on the recommendation system.

We develop a recommendation system based on the co-visitation information of the graph. We calculate similarity between two connected nodes. Similarity is measured between a node and its neighbors in terms of their edge weight divided by their node weight (popularity). Then return the top 5 most similar items connected to the item.

6.1.1.2 Real-world Dataset

Due to the unavailability of real-world co-visitation attacked dataset, we use MovieLens100k [29] rating dataset. We convert the user-to-item rating data into co-visitation data. We assume here, each movie refers to an item in the co-visitation graph and two movies are said to be co-visited if an user rates both of them at almost the same time and the edge (depicting co-visitation) weight will be 1. We assume, the edge weight will increase if more users co-rated (i.e: co-visited) those two items subsequently. By applying our approach, we get 1682 movie items as node and 102189 connections between them from the MovieLens100k dataset. We assign node popularity by summing up all of its edges' weights with its neighbors. Subset of 100K dataset is used to evaluate the performance on different dataset size. Table-6.2 presents the node and edge count in each subset used in our experiment.

6.1.2 System Specification

We compiled our source code for Generating Co-visitaion graph and executing the attack to run our experiments on a local machine. The local machine is an HP ProBook workstation having Intel Core i5 8th gen processor with 16GM RAM. To store the graph there are several graph database available like Neo4j, ArangoDB, OrientDB, Cassandra etc. Among them we use Neo4j graph database to store our graph. The reasons behind choosing Neo4j are as follows:

- Easy to learn. It has a great community as well as large number of free books and online courses available.
- It uses Cypher which is a great and easy to use graph query language.
- Easier to load huge size of data into Neo4j, with very low memory footprint.
- It delivers the lightning-fast read and write performance, while still protecting data integrity. It is the only enterprise-strength graph database that combines native graph storage, scalable architecture optimized for speed, and ACID compliance to ensure predictability of relationship-based queries.

We have developed our program in Python language and connected to Neo4j database for maintaining our co-visitation graph. All necessary scripts required to perform attack to the graph are also developed in Python language.

Two of the baseline detection methods are developed in Python : K-Means and I-Louvain. And the other two detection systems are developed in Matlab : Radar and ANOMALOUS.

6.1.3 Evaluation Metrics

The Detection Rate (DR) and False Alarm Rate (FAR) are used to measure the detection performance of the proposed approach [25]. DR represents the number of outliers detected divided by the total number of outliers in the specific dataset. FAR refers to the proportion that genuine data is falsely detected as attack behavior [83]. The definitions can be described as follows,

$$DR = \frac{|V_d \cap V_a|}{|V_a|} \tag{6.1}$$

$$FAR = \frac{|V_d \cap V_g|}{|V_g|} \tag{6.2}$$

where, V_d, V_a and V_g refer to the set of the detected nodes, all attack nodes and all genuine nodes, respectively.

Table 6.3: Empirically Determined Parameter Values.

Name	Notation	Value
Popularity Threshold	τ	500
Number of top items in RS list	k	5
Number of clusters in K-means	C	6
Controls row sparsity of W (Radar & ANOMALOUS)	α	0.0005
Controls column sparsity of W (Radar & ANOMALOUS)	β	0.01
Controls column sparsity of M (Radar & ANOMALOUS)	γ	0.1

6.2 Experimental Result Analysis

Table-6.3 gives the empirically determined values of the parameters we used in this research.

6.2.1 Analyzing Different Detection Results

We have exploited different benchmark detection approaches including K-Means, I-Louvain and Radar based detection methods. The results of different detection approaches have been evaluated and the effectiveness of our presented detection approach have been exhibited in terms of detecting fake co-visitation attacks. It is worth mentioning that all the details about parameters used in presented methods are empirically determined. It is also noteworthy that we only evaluate the promotion attack with high knowledge (PH) and demotion attack with high knowledge (DH). All the results of the suggested detection approach including DR and FAR are averaged. All the graphs and results shown in the following subsections are on real-world dataset. In the next subsections we will discuss briefly about the results of individual detection approaches applied on the attacked dataset.

6.2.1.1 K-Means Based Detection

To evaluate the detection performance of K-Means clustering based detection method, we apply K-Means detection approach to our attacked dataset. Based on the rich set of attributes of nodes, the K-Means method divides the whole dataset into C clusters. It is assumed that the attribute information of an anomalous node differs significantly from the majority of other node's attributes. As a result, the anomalous node will belong to the smallest cluster if and only if the network itself is large and has many connections. From our experiment on a real dataset (movielens 100K), we can observe that when the cluster size is small ($C = 2$), the anomalous nodes fall into the bigger cluster as the network size is relatively small and nodes are not closely connected. As a result, suspicious nodes do not differ significantly from the majority of nodes

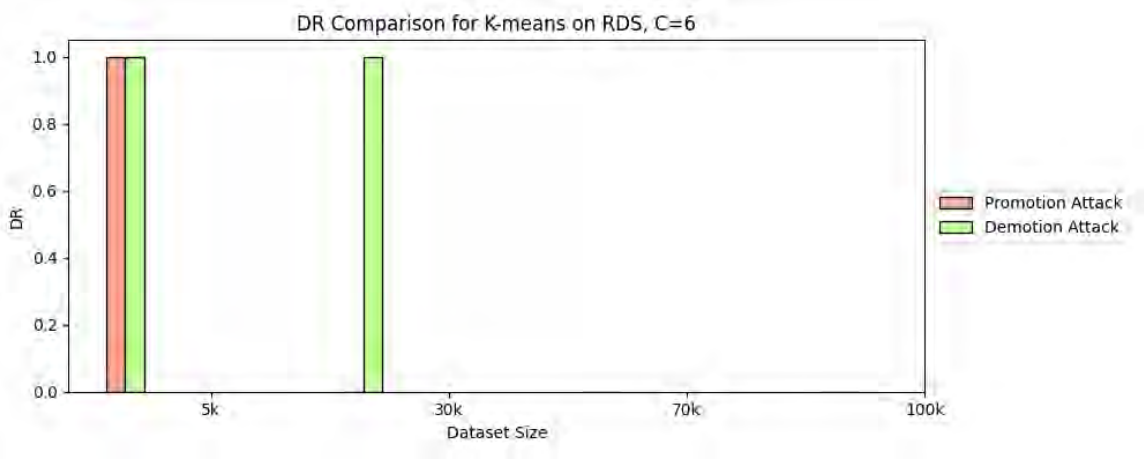


Figure 6.1: DR comparison for k-means on RDS, $C = 6$.

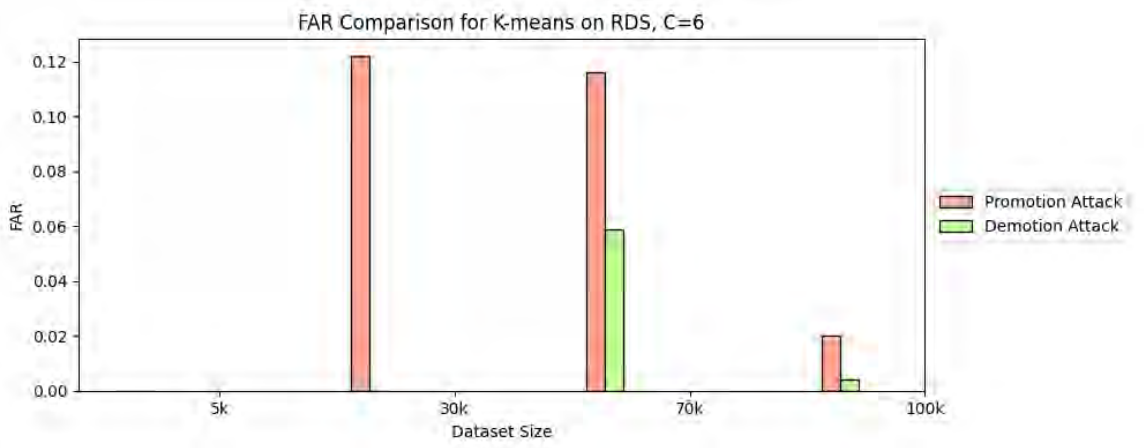


Figure 6.2: FAR comparison for k-means on RDS, $C = 6$.

resulting in the fall into the bigger cluster. But with the increasing size of C, the tendency of anomalous nodes to be in small clusters rises high. The following two graphs on fig-6.1 and fig-6.2 are showing DR and FAR comparison of promotion attack and demotion attack on different sizes of datasets, respectively. From the graph we can observe that the DR and FAR decreases with dataset size. When the dataset size is small the K-Means clustering method performs well with $C = 6$ for both promotion and demotion attack.

6.2.1.2 I-Louvain Based Detection

We also apply I-Louvain, a clustering based approach for detecting suspicious nodes, on our attacked dataset to evaluate its detection performance. Fig-6.3 and fig-6.4 are showing DR and FAR comparison of promotion attack and demotion attack on different sizes of datasets, respectively. From the detection result, we observe that the I-Louvain based clustering method performs well with a small dataset for promotion attack. But when the dataset size increases, its detection rate decreases. The I-Louvain based clustering method does not perform well for

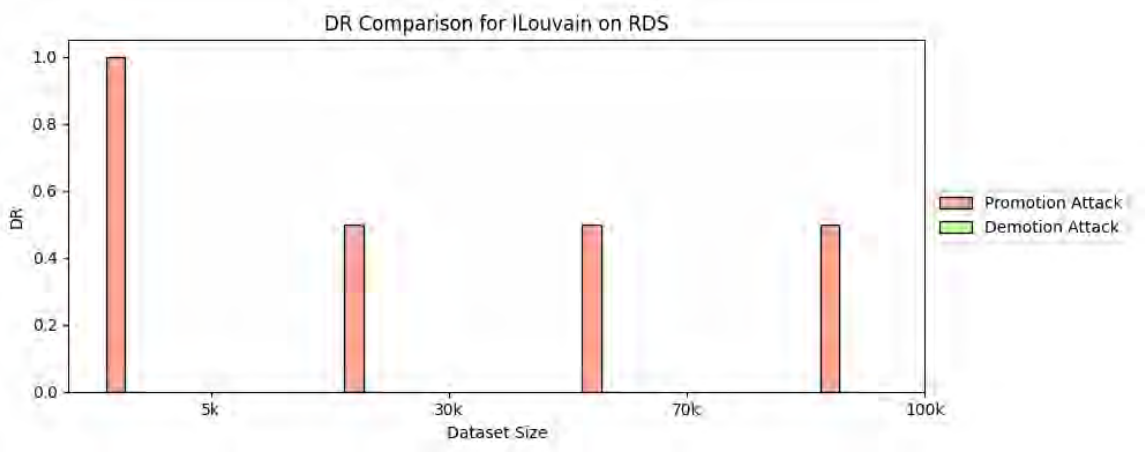


Figure 6.3: DR comparison for ILouvain on RDS.

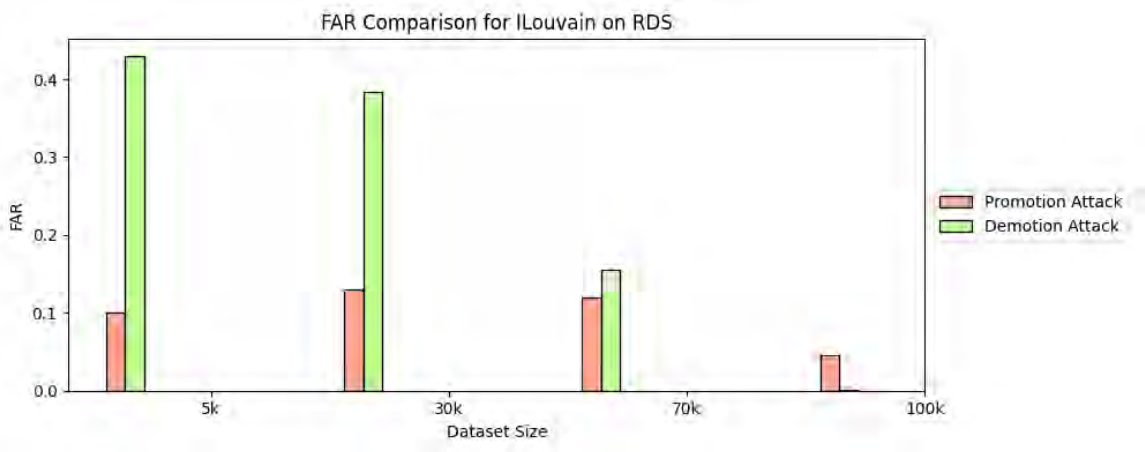


Figure 6.4: FAR comparison for ILouvain on RDS.

demotion attack for any size of dataset. So we can state that, though the I-Louvain algorithm requires less time to execute, it does not work as expected in our experiment for detecting outlier nodes.

6.2.1.3 Radar Based Detection

To evaluate the detection performance of radar based detection approach, we apply Radar on our attacked dataset. The experimental results in terms of DR and FAR values are presented in fig-6.5 and fig-6.6, respectively. By comparing the experimental results on different sizes of datasets, we can observe that Radar obtains the best performance on promotion attack when $m = 6$ regardless of different dataset size with a minimum FAR. On the other hand, for demotion attack radar performs well only when the dataset size is large. But when the dataset size is small, DR decreases. Here m is the number of top nodes from the result which we take as detected nodes. The reason for its excellent detection rate is that in real-world attributed networks, nodes are annotated as anomalies because of a variety of reasons. This detection approach provides a

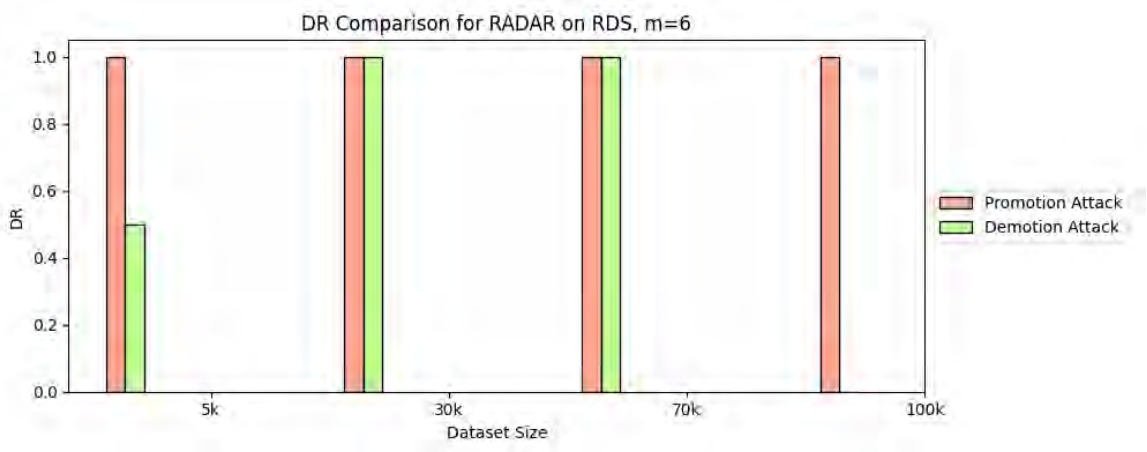


Figure 6.5: DR comparison for Radar on RDS , $m = 6$.

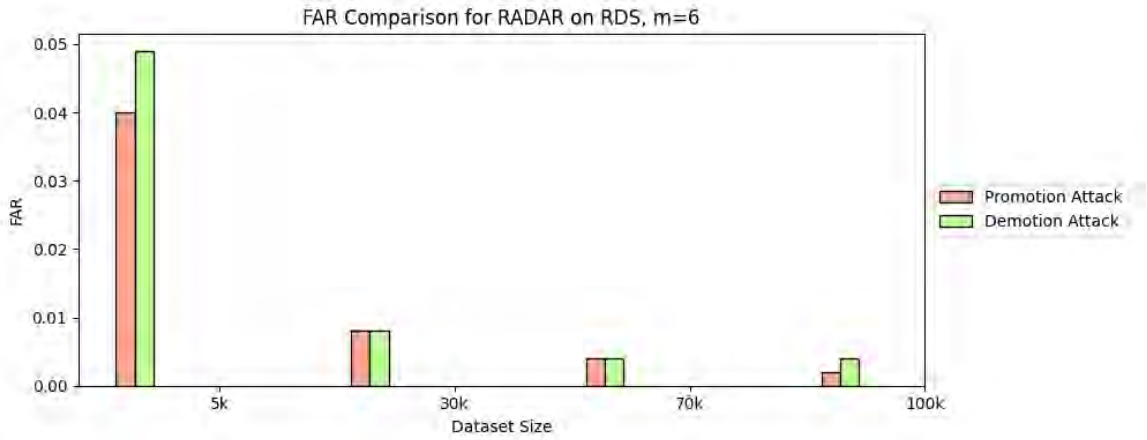


Figure 6.6: FAR comparison for Radar on RDS , $m = 6$.

general way to detect anomalies globally and does not depend on specific properties of anomalies. It is noteworthy that in our experiments we set parameters α, β and γ as 0.0005, 0.01 and 0.1 respectively, which controls the row sparsity of both the coefficient matrix and the residual matrix and also balance the contribution of attribute and network modeling.

From the graph of fig-6.7 and fig-6.8 we can observe that the DR increases with m. m is considered as the top m number of elements with highest residual errors. When the dataset size is small, the anomalous node cannot be detected if m is as small as 2. But DR increases when m is set to be 6 for $5k$ size of dataset. But for a larger dataset, DR increases with even small size of m. It is worth mentioning that we set an optimum value, $m = 6$ for all our experiments and for all sizes of dataset. On the other hand, the FAR increases with m. The DR and FAR trend with respect to m on real dataset are presented in fig-6.7 and fig-6.8.

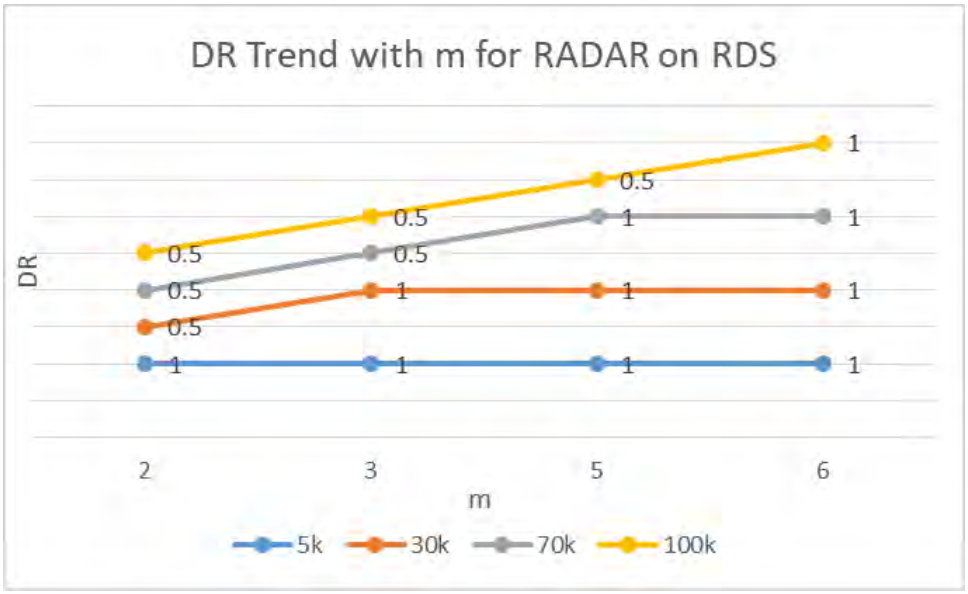


Figure 6.7: DR trend with m for Radar on RDS.

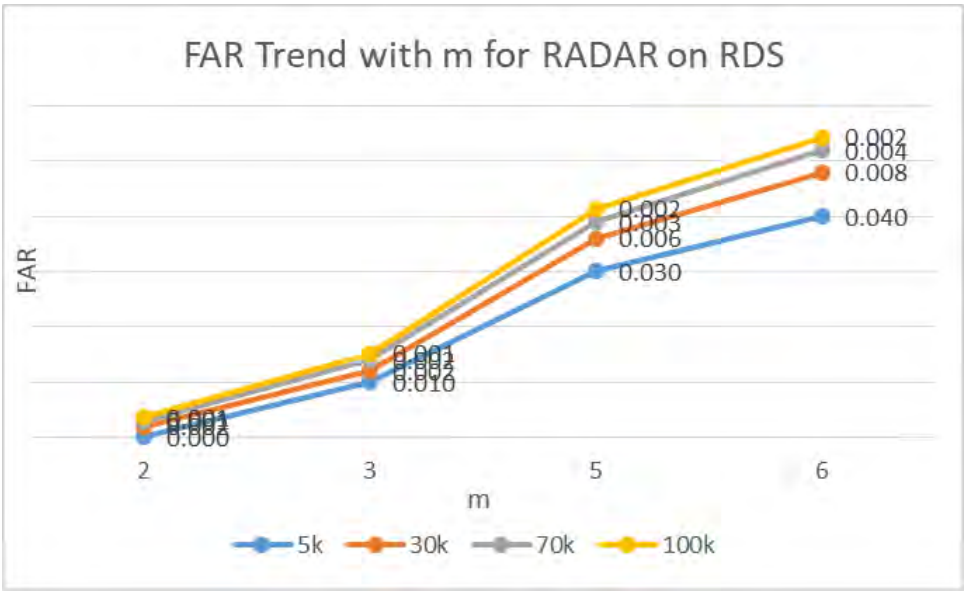


Figure 6.8: FAR trend with m for Radar on RDS.

6.2.1.4 Our proposed detection approach

In our proposed detection approach, we applied ANOMALOUS approach for populating suspicious nodes at the top of the residual error list. Then to evaluate the detection performance we have calculated detection rate and false alarm rate. From fig-6.9 and fig-6.10, we can observe that our method has obtained the best detection rate with a very minimum FAR. It performs well for both promotion and demotion attack regardless of dataset size. It is noteworthy that average false alarm rates are higher with small datasets. But when the dataset size increases FAR decreases. We take $m = 3$ in consideration, which means our suspicious nodes are in the top 3 of the residual error list containing high residual error score. Similar to Radar, parameters α, β

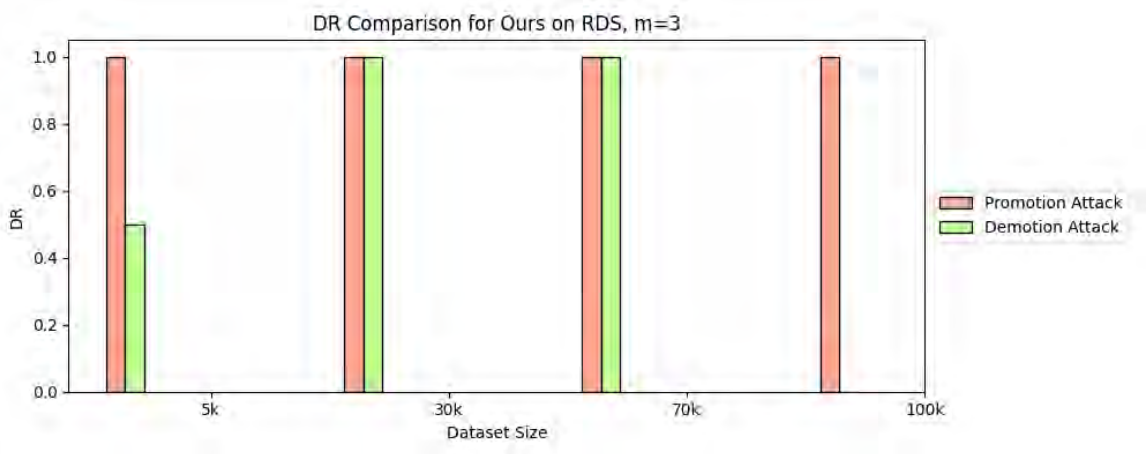


Figure 6.9: DR comparison for Anomalous on RDS , $m = 3$.

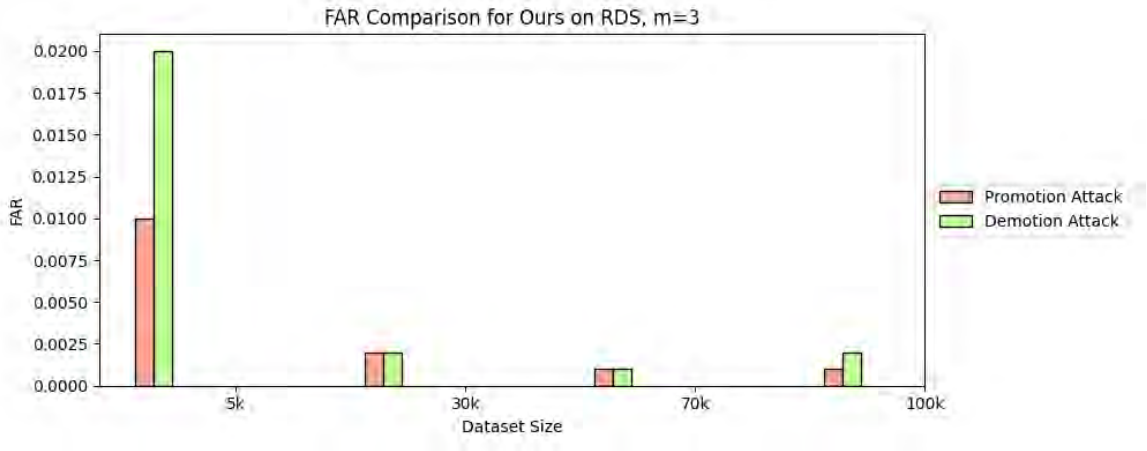


Figure 6.10: FAR comparison for Anomalous on RDS , $m = 3$.

and γ are set to be 0.0005, 0.01 and 0.1 respectively.

We can observe from fig-6.11 that the DR obtains the best performance regardless of the size of m for all dataset sizes. m is considered as the top m number of elements with highest residual errors. On the other hand, the FAR increases with m in fig-6.12. The DR and FAR trend with respect to m on real dataset are presented in fig-6.11 and fig-6.12, respectively.

6.2.2 Analyzing The Importance of Node Attributes

Attribute selection plays a vital role in the detection process as a large number of attribute representations may increase the FAR as well as it may increase the time and space complexity. Therefore, to evaluate the effectiveness of attributes, extensive experiments on real-world dataset have been conducted. Fig-6.13 and fig-6.14 presents the DR and FAR comparison of individual attributes on real-world dataset for promotion and demotion attack. From the experiment, we can see that two attributes $f2$ and $f5$ from table:6.4 are more significant in the detection process compared to others. From our observation we found that along with these two significant

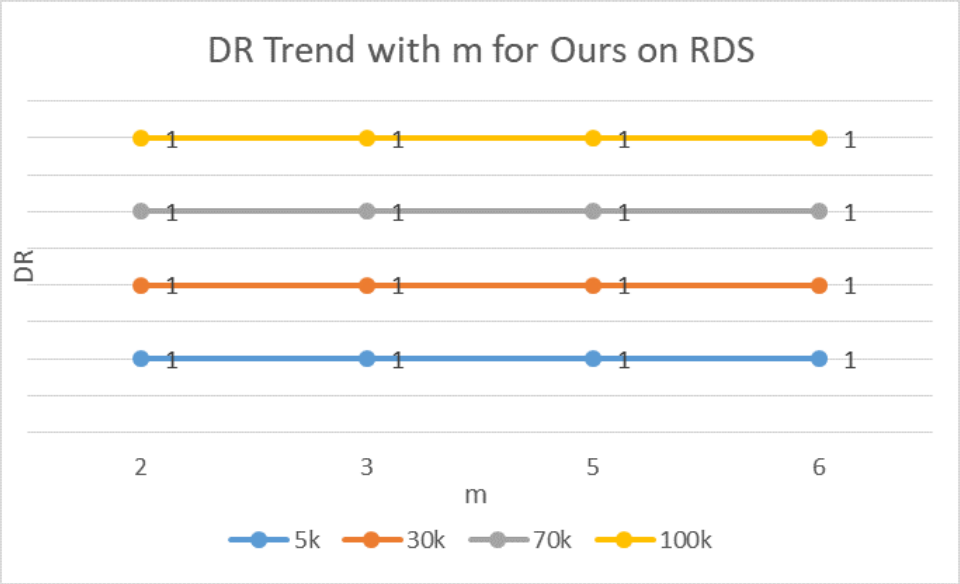


Figure 6.11: DR trend with m for ANOMALOUS on RDS.

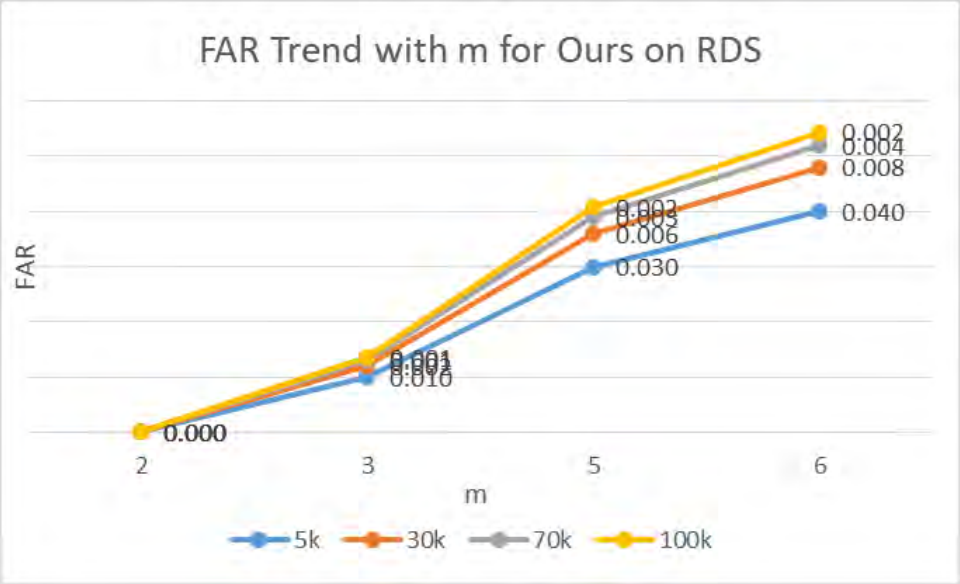


Figure 6.12: FAR trend with m for ANOMALOUS on RDS.

attributes, combination of other five attributes gives better results compared to considering only these two.

Fig-6.15 and fig-6.16 presents the impact of our proposed two attributes ($f6$ and $f7$) on DR and FAR for real-world dataset. Here, the red bar represents the DR and FAR when only $f2$ and $f5$ are considered. The green bar is for all the baseline attributes $f1$ to $f5$. The last but not the least, the blue bar represents DR and FAR when considering all the baseline attributes along with our two proposed attributes. From both the graphs we can see that, the proposed detection approach shows a better performance when considering all seven attributes.

Table 6.4: Definition of Node Attributes

Name	Definition	Mathematical Representation
Degree ($f1$) [7]	Degree of node i denotes the number of neighbor nodes.	$d_i = \sum_{j=1}^{n} A(i,j)$ (6.3)
Average degree of neighbors of a node i ($f2$) [7]	Average degree of neighbour can be simply defined as the average number of edges per neighbour node has in the graph.	$\overline{\mathrm{d_{N_i}}} = \frac{1}{\lvert N(i)\rvert} \sum_{v=N(i)} \sum_{j=1}^{n} A(v,j)$ (6.4)
Embeddedness ($f3$) [7]	The degree to which a node is intertwined in a network.	$E_{m_i} = \frac{1}{\lvert N(i)\rvert} \sum_{j=N(i)} \frac{\lvert N(i) \cap N(j)\rvert}{\lvert N(i) \cup N(j)\rvert}$ (6.5)
The number of co-visitations of a node i ($f4$) [7]	The number of co-visitations of a node indicates co-visited behaviors conducted by attackers.	$NC_i = \sum_{j=1}^{n} M_c(i,j)$ (6.6)
The average number of co-visitations of neighbors of a node i ($f5$) [7]	Used to characterize the difference between sybil and benign nodes	$\overline{NC_{N_i}} = \frac{1}{\lvert N(i)\rvert} \sum_{v=N(i)} \sum_{j=1}^{n} M_c(v,j)$ (6.7)
The average number of common nodes between a node i and neighbours ($f6$)	Two nodes are interconnected closely if the node and its neighbors have common neighbors between them	$\overline{N_c(i, N(i))} = \frac{1}{\lvert N(i)\rvert} \sum_{j=N(i)} \lvert N(i) \cap N(j)\rvert$ (6.8)
The average closely connected neighbors of a node i ($f7$)	Two nodes can be said as closely connected if their neighbors have common neighbors	$\overline{N_c(N(p), N(q))} = \frac{1}{\lvert N(i)\rvert} \sum_{p,q \epsilon N(i); p<q} Q_{pq}$, $Q_{pq} = \begin{cases} 1, & \text{if } \lvert N(p) \cap N(q)\rvert > 1 \\ 0, & \text{otherwise} \end{cases}$

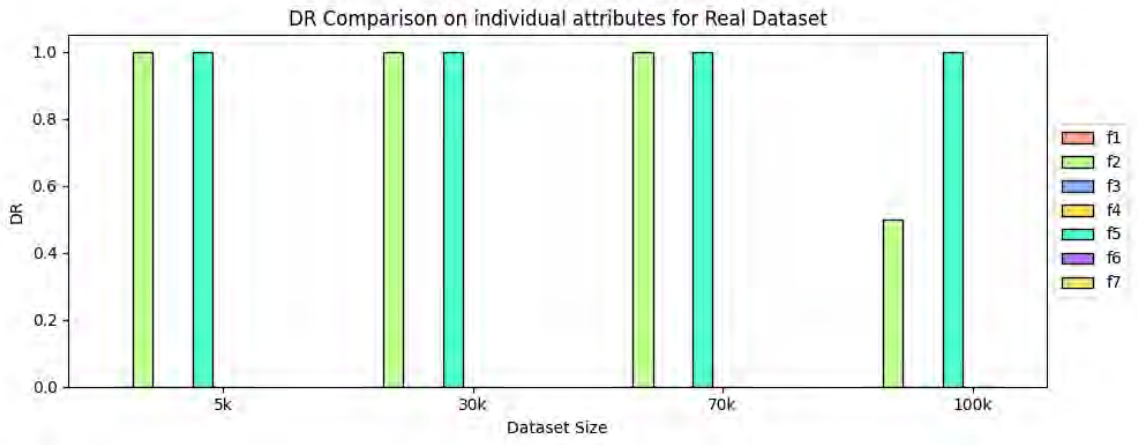


Figure 6.13: DR Comparison of Individual Attributes on Real-World Dataset.

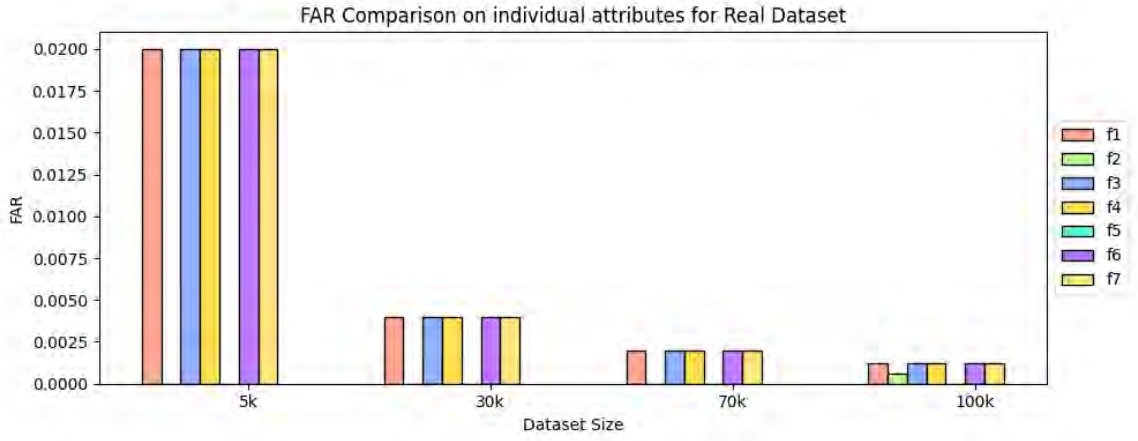


Figure 6.14: FAR Comparison of Individual Attributes on Real-World Dataset.

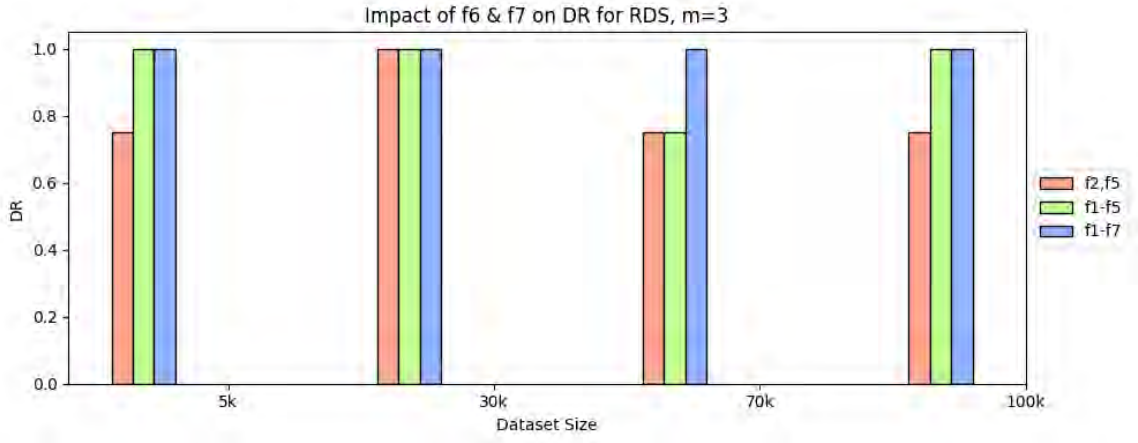


Figure 6.15: Impact of our proposed attributes($f6$ and $f7$) on DR for RDS, $m = 3$.

6.3 Comparison of Different Detection Results

DR and FAR comparison of different benchmark detection techniques along with our proposed approach to evaluate the effectiveness of our proposed approach are shown in fig-6.19, fig-6.20 ,

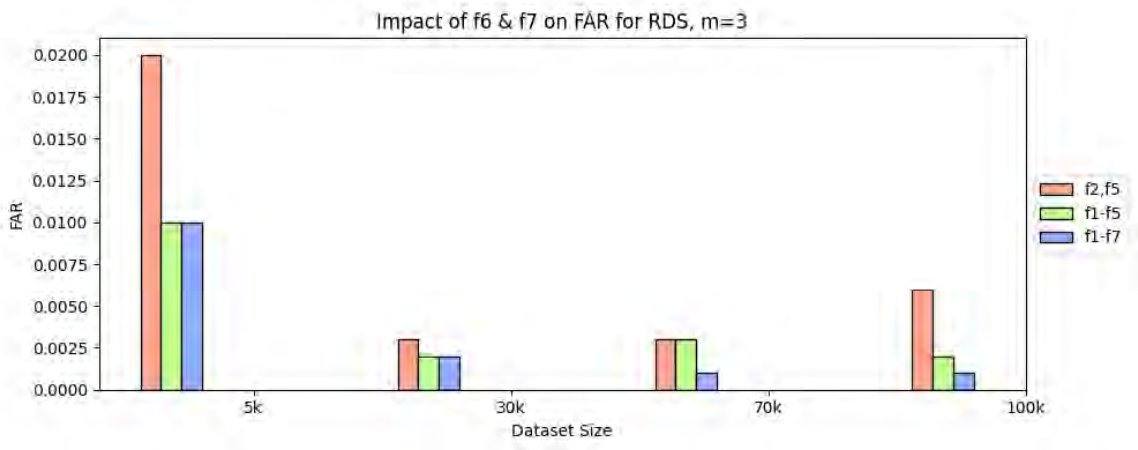


Figure 6.16: Impact of our proposed attributes($f6$ and $f7$) on FAR for RDS, $m = 3$.

fig-6.21 and fig-6.22. From experimental results from the synthetic dataset presenting in fig-6.17 and fig-6.18, we can observe that the results are really promising . Table 6.5 shows a list of conducted experiments in terms of DR and FAR on synthetic dataset. However, our prime purpose is to detect anomalies from real-world dataset. Table 6.6 shows a list of conducted experiments in terms of DR and FAR on real dataset. Fig-6.19 and fig-6.20 are showing the comparison result for promotion attack while fig-6.21 and fig-6.22 are representing the comparison result for demotion attack on different sizes of dataset. By analyzing the pattern of detection rate for different detection methods, we can observe that clustering based methods (Kmeans, ILouvain) worked well when the dataset is small for both synthetic and real-world dataset. But when the dataset size increases, the attacker node can not be detected for both type of attacks (promotion and demotion) even if the clustering size is increased to $C = 6$. When we analyze the false alarm rate (FAR) for clustering based methods (K-Means, I-Louvain) in real-world dataset, we observe that K-Means performance is far better than I-Louvain's as the average K-Means FAR is 34.82% smaller than I-Louvain's. However, analyzing the results from synthetic dataset we found that, the gap between K-Means and I-Louvain is much greater in terms of FAR.

For demotion attack we have observed that it follows the declining trend for DR, i.e: as the dataset size grows DR decreases which is true for both clustering based methods. Moreover we have observed significant performance drop for I-Louvain in comparison to K-Means method. In terms of FAR the result gives little mixed trend. For K-Means the FAR increases and then drops again with the increment of dataset size. But for I-Louvain it is almost the same trend as we have found in promotion attack, a declining trend with increment of dataset size.

On the other hand, Radar performs well with small dataset even if m is small, here m refers to the number of detected nodes with highest residual errors. But when working with larger dataset, we observe that m needs to be larger in order to get better performance.

Next it is found that our proposed technique is more scalable and consistent throughout the experiments compared with other methods. It outperforms the baselines by a significant margin

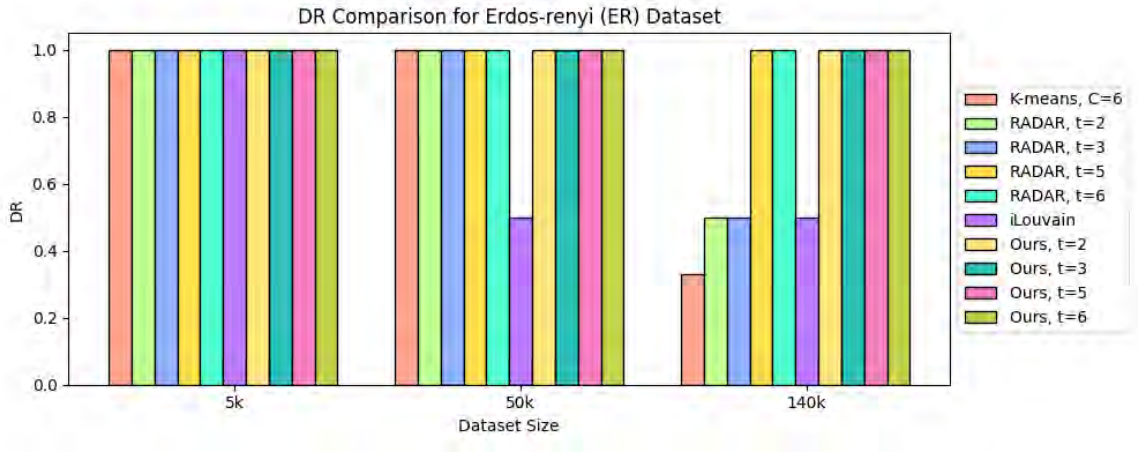


Figure 6.17: DR comparison on Synthetic(ER) dataset for promotion attack.

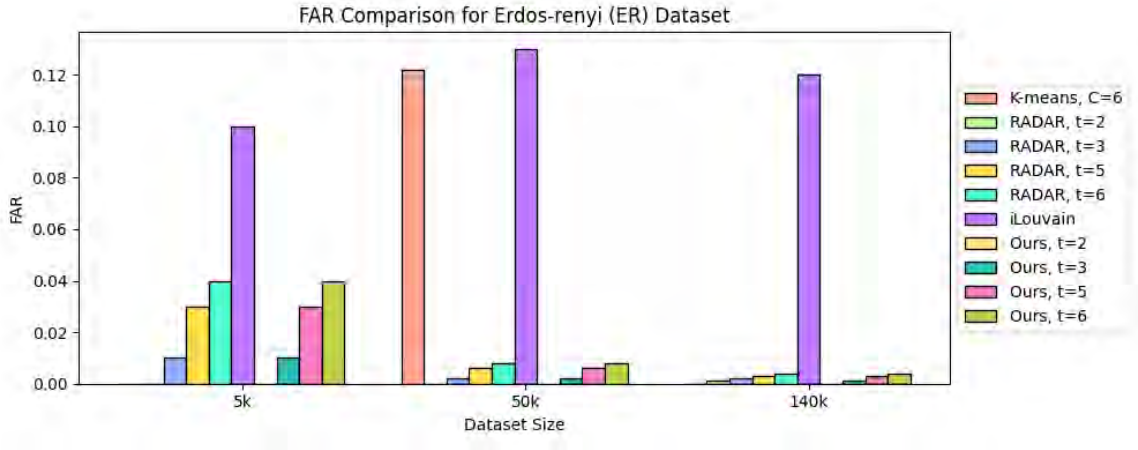


Figure 6.18: FAR comparison on Synthetic(ER) dataset for promotion attack.

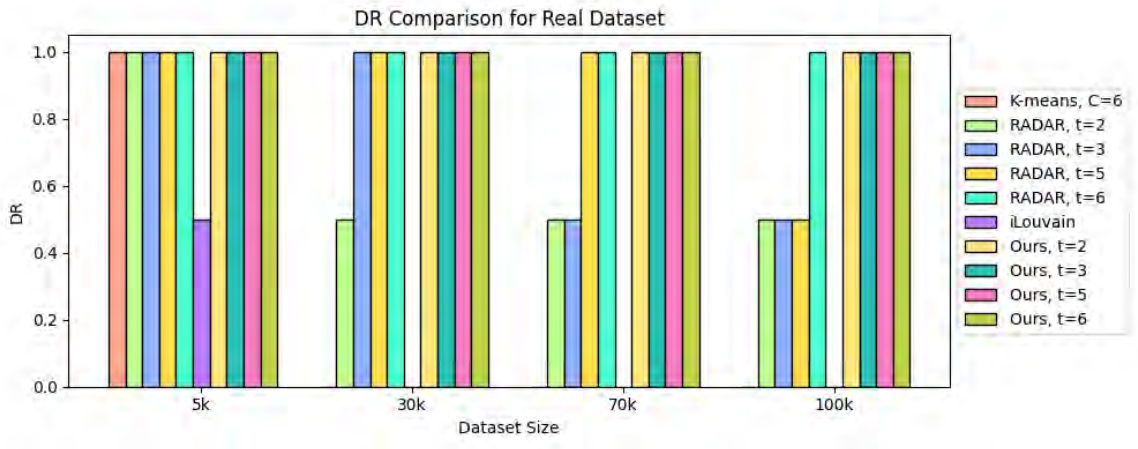


Figure 6.19: DR comparison on Real dataset for promotion attack.

in both synthetic and real-world dataset in terms of DR and FAR. From our experiment we can firmly say that all the clustering based methods have performed fairly but due to the fact that they are solely depending upon node attributes, they have been outperformed by Radar and our

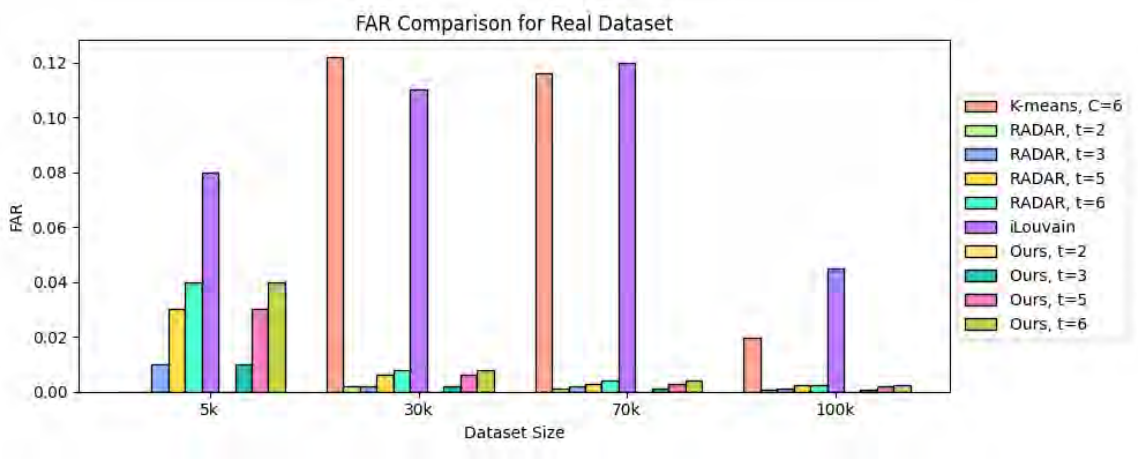


Figure 6.20: FAR comparison on Real dataset for promotion attack.

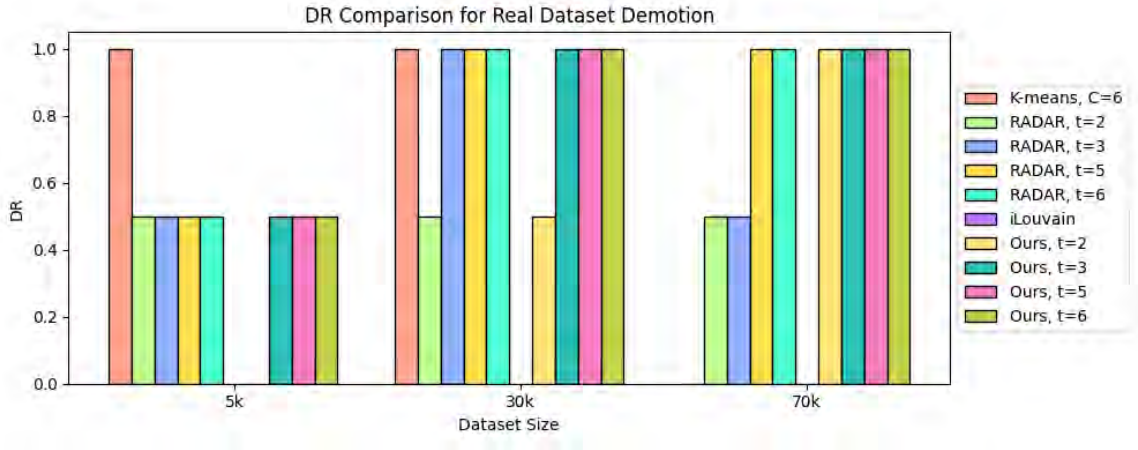


Figure 6.21: DR comparison on Real dataset for demotion attack.

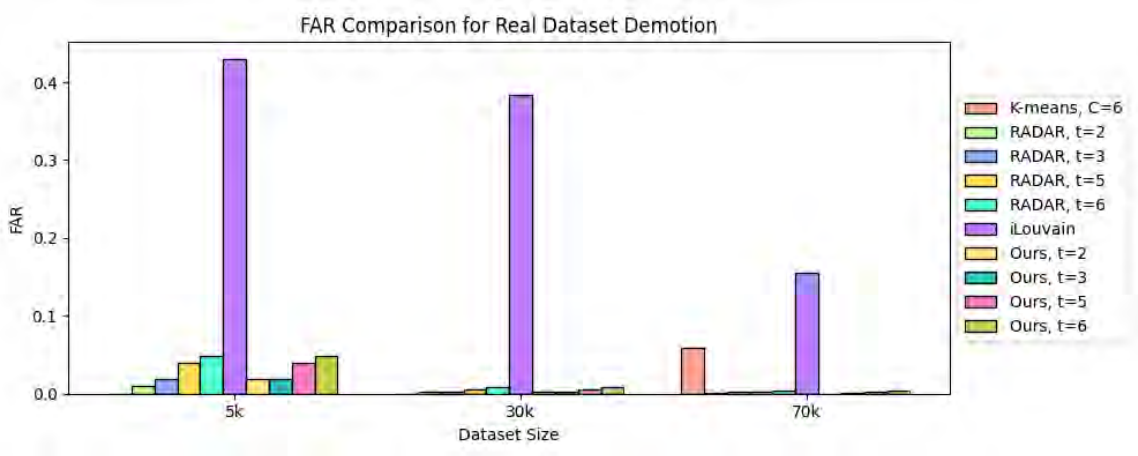


Figure 6.22: FAR comparison on Real dataset for demotion attack.

technique as the latter two use not only the attributes of nodes but also the topological properties of association graph. So from next on-wards, we will focus on performance improvement between Radar and our approach. When we analyze the detection rate (DR) for residual analysis

Table 6.5: Detection Results of the Proposed Approach Compared with Baselines on Synthetic Dataset

Co-visitation attack	Attack Scenario	ER Dataset size	Metric	Benchmarks									
				K-means, C=6	RADAR, m=2	RADAR, m=3	RADAR, m=5	RADAR, m=6	iLouvain	Ours, m=2	Ours, m=3	Ours, m=5	Ours, m=6
Promotion	High Knowledge	5K	DR	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			FAR	0.0000	0.0000	0.0100	0.0300	0.0400	0.1000	0.0000	0.0100	0.0300	0.0400
		50K	DR	1.00	1.00	1.00	1.00	1.00	0.50	1.00	1.00	1.00	1.00
			FAR	0.1220	0.0000	0.0020	0.0060	0.0080	0.1300	0.0000	0.0020	0.0060	0.0080
		140K	DR	0.33	0.50	0.50	1.00	1.00	0.50	1.00	1.00	1.00	1.00
			FAR	0.0000	0.0010	0.0020	0.0030	0.0040	0.1200	0.0000	0.0010	0.0030	0.0040

Table 6.6: Detection Results of the Proposed Approach Compared with Baselines on Real-World Dataset

Co-visitation attack	Attack Scenario	Real Dataset size	Metric	Benchmarks									
				K-means, C=6	RADAR, m=2	RADAR, m=3	RADAR, m=5	RADAR, m=6	iLouvain	Ours, m=2	Ours, m=3	Ours, m=5	Ours, m=6
Promotion	High Knowledge	5K	DR	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0000	0.0000	0.0100	0.0300	0.0400	0.1000	0.0000	0.0100	0.0300	0.0400
		30K	DR	0.0	0.5	1.0	1.0	1.0	0.5	1.0	1.0	1.0	1.0
			FAR	0.1220	0.0020	0.0020	0.0060	0.0080	0.1300	0.0000	0.0020	0.0060	0.0080
		70K	DR	0.0	0.5	0.5	1.0	1.0	0.5	1.0	1.0	1.0	1.0
			FAR	0.1160	0.0010	0.0020	0.0030	0.0040	0.1200	0.0000	0.0010	0.0030	0.0040
		100K	DR	0.0	0.5	0.5	0.5	1.0	0.5	1.0	1.0	1.0	1.0
			FAR	0.0196	0.0006	0.0012	0.0024	0.0024	0.0451	0.0000	0.0006	0.0018	0.0024
Demotion	High Knowledge	5K	DR	1.0	0.5	0.5	0.5	0.5	0.0	0.0	0.5	0.5	0.5
			FAR	0.0000	0.0098	0.0196	0.0392	0.0490	0.4300	0.0196	0.0196	0.0392	0.0490
		30K	DR	1.0	0.5	1.0	1.0	1.0	0.0	0.5	1.0	1.0	1.0
			FAR	0.0000	0.0020	0.0020	0.0060	0.0080	0.3840	0.0020	0.0020	0.0060	0.0080
		70K	DR	0.0	0.5	0.5	1.0	1.0	0.0	1.0	1.0	1.0	1.0
			FAR	0.0588	0.0010	0.0020	0.0030	0.0040	0.1550	0.0000	0.0010	0.0030	0.0040
		100K	DR	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
			FAR	0.0036	0.0012	0.0018	0.0030	0.0036	0.0012	0.0012	0.0018	0.0030	0.0036

based methods (Radar and our approach) in real-world dataset for promotion attack, we observe our technique's performance is better than Radar's when m is small. In fact our technique showed almost 100% detection rate where Radar's were lower. Now to give some advantage to Radar, we increased m and found that Radar can detect all the attacker nodes same as our technique can but with larger FAR. Focusing on FAR comparison when m is small and Radar's DR is less than ours, the average FAR for our technique is 0.00 in contrast to Radar's 0.0009. Going forward, when m is large and DR is same and maximum for both, the average FAR for our approach is 10.52% smaller than Radar's. If we take the largest dataset size for our real-world data, then we find this gap to be as large as 50%. However, analyzing the results from the synthetic dataset we found that the gap between ours and Radar's is smaller in terms of FAR.

For Demotion attack Radar's detection performance is observed to be better than ours when dataset is small and m is also small. With the increment of dataset size and m this gap diminishes and our technique came out on top. But for saturated value of m we have found the DR performance to be the same for both the techniques. In terms of FAR our technique outperforms Radar's but at saturated value of m we have found the FAR performance to be the same for both the techniques, a declining trend with increment of dataset size.

6.3.1 Abnormality Forensics for Real Application

To improve the detection performance of malicious nodes in real-world dataset is the ultimate goal of our proposed approach. In reality, whether the system is attacked or not and the scale of attack is unknown. This is why we investigate abnormality forensics metrics for real data from diverse perspectives:

1. Due to resource limitation, attacker can inject a bounded number of co-visitations into the network. So attacker may not inject so many fake data between target and filler item rather inject most of the fake co-visitation data between target and anchor item. So we can investigate the suspicious item's neighborhood. It is highly possible that the suspicious item is not densely connected with the network.

2. By analyzing historical co-visitation data any sudden spike in the graph can be observed. If an unpopular item suddenly has many co-visitations with some items within a short span of time, then there is a high chance that an attacker is trying to promote this item.

6.4 Limitations

Limitations of our experimental results need to be taken into account. Our study is subject to few external limitations as it is based on analysis and design of a data-driven algorithm. As we discuss, the validity of our study could be threatened by following limitations: (1) the choice of original dataset of MovieLens data; (2) the implementation of co-visitation attacks, such as the choice of anchor items; (3) the choice of an original Erdos-Renyi (ER) random graph [4] used in co-visitation injection attacks; (4) designing the attack as such, attacker is with high knowledge attacking for promotion and demotion attack. Different implementation details of above situations might lead to different detection results at each step. To mitigate any bias in these respects, we provide corresponding suggestions, including: (1) exploiting similar and original base datasets (e.g., MovieLens-1M, MovieLens-25M, etc.) is beneficial to obtain more co-visitation data; (2) changing the value of τ [4] might lead to slightly different result; (3) to select different parameter value while generating Erdos-Renyi (ER) random graph and (4) designing attack with different knowledge level like medium or low and also for demotion purpose.

Chapter 7

Conclusion

By influencing the recommendation result, a co-visitation injection attack can have a significant impact on the recommender system, jeopardizing the reputation of personalized recommendation services. Based on attribute representations and associated connections of nodes in the co-visitation network, we describe an approach for identifying fake co-visitation attacks in this paper. Using CUR decomposition, the technique reduces noisy data and extracts the most significant attributes from a co-visitation attributed graph. Outlier nodes are identified via residual analysis depending on the fact that outlier nodes have a substantial residual error. In comparison to baseline approaches, our method has a lower time complexity. The effectiveness of our approach has been demonstrated through extensive experimental findings on both synthetic and real-world datasets. In terms of detection rate and false alarm rate as performance parameters, there has been significant improvement over a few baselines. We found that our method improves detection accuracy in terms of FAR by more than 10% on average, with a maximum of 50% improvement, when compared to the best resulting baseline methods. Nonetheless, it is yet to achieve the ultimate level (DR of 100% and zero FAR). Therefore, more research is needed to improve the detection technique's robustness and accuracy.

7.1 Future Work

The scope of future research includes:

- Despite our extensive experiments on real-world dataset, practical evidences for analyzing and detecting anomalies on real-world data are insufficient, which led us to generate co-visitation graph from Movielens100K rating dataset. As future work we like to expand our experiment on real-world co-visitation attacked dataset directly.
- Also experimenting larger datasets (MovieLens1M, MovieLens25M) are in our future roadmap.

- In this research, we have considered the upper bound attack which consist of high knowledge with promotion and demotion attack model to conduct our attack. Designing attack model with different knowledge level like medium or low for both promotion and demotion purpose would be our next agenda.

- Several parameters, including those of the proposed detection model α,β, γ and t threshold value, need to be fine-tuned. The row sparsity of the coefficient matrix and the residual matrix is controlled by parameters α and β in Eq.-5.22. Parameter γ is for balancing the contribution of attribute reconstruction and network modeling (see Eq.-5.22). The number of nodes with the largest residual errors is called the threshold t. These factors were determined empirically in this study. Unfortunately, we were unable to discover an effective method for identifying ideal parameters automatically. This could be a future constraint of our work that we need to address.

References

[1] A. Eckhardt, "Various aspects of user preference learning and recommender systems," vol. 471, pp. 56–67, 01 2009.

[2] L. Ngoupeyou Tondji, *Web Recommender System for Job Seeking and Recruiting*. PhD thesis, 02 2018.

[3] S. Wang, L. Hu, Y. Wang, X. He, Q. Sheng, M. Orgun, L. Cao, F. Ricci, and P. Yu, "Graph learning based recommender systems: A review," 05 2021.

[4] G. Yang, N. Z. Gong, and Y. Cai, "Fake co-visitation injection attacks to recommender systems.," in *NDSS*, 2017.

[5] Z. Yang, Q. Sun, Y. Zhang, and W. Wang, "Identification of malicious injection attacks in dense rating and co-visitation behaviors," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 537–552, 2020.

[6] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[7] Z. Yang, Q. Sun, Y. Zhang, L. Zhu, and W. Ji, "Inference of suspicious co-visitation and co-rating behaviors and abnormality forensics for recommender systems," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2766–2781, 2020.

[8] Z. Yang, Q. Sun, and Y. Zhang, "Probabilistic inference and trustworthiness evaluation of associative links toward malicious attack detection for online recommendations," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, pp. 1–1, 09 2020.

[9] CODEPATH, "Graphs."

[10] CUEMATH, "Diagonal matrix."

[11] A. Uberoi, "Introduction to dimensionality reduction."

[12] N. S. Chauhan, "Dimensionality reduction with principal component analysis (pca)."

[13] N. Erichson, S. Voronin, S. Brunton, and J. Kutz, "Randomized matrix decompositions using r," 08 2016.

[14] J. Li, H. Dani, X. Hu, and H. Liu, "Radar: Residual analysis for anomaly detection in attributed networks.," in *IJCAI*, pp. 2152–2158, 2017.

[15] Y. Asim, A. Majeed, R. Ghazal, B. Raza, W. Naeem, and A. K. Malik, "Community detection in networks using node attributes and modularity," *Int J Adv Comput Sci Appl*, vol. 8, no. 1, pp. 382–388, 2017.

[16] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[17] A. Bóta, M. Krész, and B. Zaválnij, "Adaptations of the k-means algorithm to community detection in parallel environments," in *2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pp. 299–302, IEEE, 2015.

[18] Z. Peng, M. Luo, J. Li, H. Liu, and Q. Zheng, "Anomalous: A joint modeling approach for anomaly detection on attributed networks.," in *IJCAI*, pp. 3513–3519, 2018.

[19] H. A. Simon, "Designing organizations for an information-rich world," *International Library of Critical Writings in Economics*, vol. 70, pp. 187–202, 1996.

[20] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proceedings of the 34th Annual Computer Security Applications Conference*, pp. 381–392, ACM, 2018.

[21] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender systems handbook*, pp. 1–35, Springer, 2011.

[22] M. A. Hameed, O. Al Jadaan, and S. Ramachandram, "Collaborative filtering based recommendation system: A survey," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, p. 859, 2012.

[23] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web*, pp. 291–324, Springer, 2007.

[24] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, and D. Sampath, "The youtube video recommendation system," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, (New York, NY, USA), p. 293–296, Association for Computing Machinery, 2010.

[25] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, (New York, NY, USA), p. 542–547, Association for Computing Machinery, 2006.

[26] X. Xing, W. Meng, D. Doozan, A. C. Snoeren, N. Feamster, and W. Lee, "Take this personally: Pollution attacks on personalized services," in *22nd {USENIX} Security Symposium ({USENIX} Security 13)*, pp. 671–686, 2013.

[27] J. Song, Z. Li, Z. Hu, Y. Wu, Z. Li, J. Li, and J. Gao, "Poisonrec: An adaptive data poisoning framework for attacking black-box recommender systems," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, (Los Alamitos, CA, USA), pp. 157–168, IEEE Computer Society, apr 2020.

[28] A. Davoudi, "Effects of user interactions on online social recommender systems," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, (Los Alamitos, CA, USA), pp. 1444–1448, IEEE Computer Society, apr 2017.

[29] Grouplens, "Movielens 100k dataset."

[30] J. B. Schafer, J. A. Konstan, and J. Riedl, "E-commerce recommendation applications," *Data mining and knowledge discovery*, vol. 5, no. 1, pp. 115–153, 2001.

[31] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pp. 207–216, 1993.

[32] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–19, 2015.

[33] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[34] L. Ungar and D. P. Foster, "A formal statistical approach to collaborative filtering," *CONALD'98*, 1998.

[35] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pp. 175–186, 1994.

[36] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "word of mouth"," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 210–217, 1995.

[37] W. Hill, L. Stead, M. Rosenstein, and G. Furnas, "Recommending and evaluating choices in a virtual community of use," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 194–201, 1995.

[38] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.

[39] A. Sharma, J. M. Hofman, and D. J. Watts, "Estimating the causal impact of recommendation systems from observational data," in *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pp. 453–470, 2015.

[40] W. Zeller and E. W. Felten, "Cross-site request forgeries: Exploitation and prevention," *The New York Times*, pp. 1–13, 2008.

[41] M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre, "Collaborative recommendation: A robustness analysis," *ACM Transactions on Internet Technology (TOIT)*, vol. 4, no. 4, pp. 344–377, 2004.

[42] S. K. Lam and J. Riedl, "Shilling recommender systems for fun and profit," in *Proceedings of the 13th international conference on World Wide Web*, pp. 393–402, 2004.

[43] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams, "Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Transactions on Internet Technology (TOIT)*, vol. 7, no. 4, p. 23, 2007.

[44] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[45] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "" you might also like:" privacy risks of collaborative filtering," in *2011 IEEE Symposium on Security and Privacy*, pp. 231–246, IEEE, 2011.

[46] P.-A. Chirita, W. Nejdl, and C. Zamfir, "Preventing shilling attacks in online recommender systems," in *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pp. 67–74, ACM, 2005.

[47] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, "Detecting profile injection attacks in collaborative recommender systems," in *The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*, pp. 23–23, IEEE, 2006.

[48] C. A. Williams, B. Mobasher, and R. Burke, "Defending recommender systems: detection of profile injection attacks," *Service Oriented Computing and Applications*, vol. 1, no. 3, pp. 157–170, 2007.

[49] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, p. 355–369, Mar. 2007.

[50] A. Davoudi and M. Chatterjee, "Detection of profile injection attacks in social recommender systems using outlier analysis," in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 2714–2719, IEEE, 2017.

[51] C. C. Aggarwal, Y. Zhao, and P. S. Yu, "Outlier detection in graph streams," in *2011 IEEE 27th International Conference on Data Engineering*, pp. 399–409, 2011.

[52] H. Zhang, S. Kiranyaz, and M. Gabbouj, "Outlier edge detection using random graph generation models and applications," *Journal of Big Data*, vol. 4, no. 1, p. 11, 2017.

[53] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, (New York, NY, USA), p. 1346–1355, Association for Computing Machinery, 2014.

[54] B. Mehta and T. Hofmann, "A survey of attack-resistant collaborative filtering algorithms.," *IEEE Data Eng. Bull.*, vol. 31, no. 2, pp. 14–22, 2008.

[55] D. Combe, C. Largeron, M. Géry, and E. Egyed-Zsigmond, "I-louvain: An attributed graph clustering method," in *International Symposium on Intelligent Data Analysis*, pp. 181–192, Springer, 2015.

[56] I. Falih, N. Grozavu, R. Kanawati, and Y. Bennani, "Anca : Attributed network clustering algorithm," in *Complex Networks & Their Applications VI* (C. Cherifi, H. Cherifi, M. Karsai, and M. Musolesi, eds.), (Cham), pp. 241–252, Springer International Publishing, 2018.

[57] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *Proc. VLDB Endow.*, vol. 2, p. 718–729, Aug. 2009.

[58] Y. Zhou, H. Cheng, and J. Yu, "Clustering large attributed graphs: An efficient incremental approach," vol. 689-698, pp. 689–698, 12 2010.

[59] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," *2013 IEEE 13th International Conference on Data Mining*, Dec 2013.

[60] S. Günnemann, I. Färber, B. Boden, and T. Seidl, "Subspace clustering meets dense subgraph mining: A synthesis of two paradigms," vol. 845-850, pp. 845 – 850, 01 2011.

[61] J. Cruz, C. Bothorel, and F. Poulet, "Entropy based community detection in augmented social networks," pp. 163–168, 10 2011.

[62] A. Dang and E. Viennet, "Community detection based on structural and attribute similarities," in *ICDS 2012*, 2012.

[63] P. Boobalan, D. Lopez, and X. Gao, "Graph clustering using k-neighbourhood attribute structural similarity," *Applied Soft Computing*, vol. 47, 06 2016.

[64] J. Xi, W. Zhan, and Z. Wang, "Hierarchical community detection algorithm based on node similarity," *International Journal of Database Theory and Application*, vol. 9, pp. 209–218, 06 2016.

[65] N. Helal, R. Ismail, N. Badr, and M. Mostafa, "An efficient algorithm for community detection in attributed social networks," pp. 180–184, 05 2016.

[66] S. Kataoka, T. Kobayashi, M. Yasuda, and K. Tanaka, "Community detection algorithm combining stochastic block model and attribute data clustering," *Journal of the Physical Society of Japan*, vol. 85, 07 2016.

[67] Y. Chen, X. Wang, J. Bu, B. Tang, and X. Xiang, "Network structure exploration in networks with node attributes," *Physica A: Statistical Mechanics and its Applications*, vol. 449, 01 2016.

[68] H. Elhadi and G. Agam, "Structure and attributes community detection: Comparative analysis of composite, ensemble and selection methods," in *Proceedings of the 7th Workshop on Social Network Mining and Analysis*, SNAKDD '13, (New York, NY, USA), Association for Computing Machinery, 2013.

[69] CUEMATH, "Orthogonal matrix."

[70] K. P. Murphy, *Machine learning: a probabilistic perspective*. 2012.

[71] Y. Anzai, *Pattern recognition and machine learning*. Elsevier, 2012.

[72] M. W. Mahoney and P. Drineas, "Cur matrix decompositions for improved data analysis," *Proceedings of the National Academy of Sciences*, vol. 106, no. 3, pp. 697–702, 2009.

[73] Oracle.com, "Cur matrix decomposition."

[74] tutorialspoint.com, "Residual analysis."

[75] corporatefinanceinstitute.com, "Regression analysis."

[76] H. Tong and C.-Y. Lin, "Non-negative residual matrix factorization with application to graph anomaly detection," in *Proceedings of the 2011 SIAM International Conference on Data Mining*, pp. 143–153, SIAM, 2011.

[77] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, July 2009.

[78] C. C. Aggarwal, "Outlier analysis," in *Data Mining: The Textbook*, (Cham), pp. 237–263, Springer International Publishing, 2015.

[79] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han, "On community outliers and their efficient detection in information networks," KDD '10, (New York, NY, USA), p. 813–822, Association for Computing Machinery, 2010.

[80] M. Davis, W. Liu, P. Miller, and G. Redpath, "Detecting anomalies in graphs with numeric labels," in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, (New York, NY, USA), p. 1197–1202, Association for Computing Machinery, 2011.

[81] N. Liu, X. Huang, and X. Hu, "Accelerated local anomaly detection via resolving attributed networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, p. 2337–2343, AAAI Press, 2017.

[82] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[83] C.-Y. Chung, P.-Y. Hsu, and S.-H. Huang, "A novel approach to filter out malicious rating profiles from recommender systems," *Decis. Support Syst.*, vol. 55, p. 314–325, Apr. 2013.

Generated using Postgraduate Thesis LaTeX Template, Version 1.03. Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.

This thesis was generated on Monday 14th March, 2022 at 3:59am.