ROBUREC: Building a Robust Recommender using Autoencoders with Anomaly Detection

Ahmed Aly
Electrical and Computer Engineering
Toronto Metropolitan University
Toronto, Canada
ahmed.aly@torontomu.ca

Dina Nawara
Electrical and Computer Engineering
Toronto Metropolitan University
Toronto, Canada
dina.nawara@torontomu.ca

Rasha Kashef
Electrical and Computer Engineering
Toronto Metropolitan University
Toronto, Canada
rkashef@torontomu.ca

***Abstract*—In the realm of social network analysis and mining, recommendation systems have become indispensable algorithms in assisting users and industries in navigating the available contents or products in various domains and getting the most personalized recommendations to their interests and preferences. However, if the input data has been generated by malicious users, that poses a significant challenge to recommender systems' reliability and efficiency. One of the main threats that poses a challenge to recommender systems is shilling attacks. Shilling attacks tend to manipulate or poison the data in the systems' training phase, leading to biased or compromised recommendations. To address this challenge, we propose a robust recommender system using variational autoencoders (VAE) with Anomaly detection. Our model learns complex and non-linear patterns by exclusively focusing on the user-item interaction data, represented by a binary user-item interaction matrix, making it more resilient to classic shilling attacks. Moreover, our paper incorporates an anomaly detection mechanism, alongside the autoencoder, that analyzes the reconstruction errors, i.e. (MSE) between the original interactions and their reconstructed ones. We test the model on a real-world dataset and evaluate it using Recall@k and NDCG@k. This work enhances the trustworthiness and accuracy of recommendation algorithms, mainly when deployed in social network analysis and mining, where the potential for malicious data manipulation is a critical concern.**

Keywords—Recommender Systems, User-Item Interactions Shillings Attacks, Autoencoders, Anomaly Detection.

I. INTRODUCTION

Vast amounts of information are available online, and navigating the massive number of options to find the most personalized and relevant choice can be overwhelming [1]. In response to such data overload, recommendation systems have emerged as crucial tools for users and industries, revolutionizing the decision-making process in many domains. Recommendation systems are those algorithms that leverage user and item data to deliver personalized attributes that can significantly improve the users' experience. The openness of some recommendation systems, such as collaborative filtering-based recommenders, that are widely used in many domains, especially in E-commerce, leads to these recommenders having significant vulnerabilities to being attacked by malicious raters [2] and that alters and affects the ratings' prediction and hence impacts the authenticity of the generated recommendations by promoting or demoting particular target products. Most of the work done in the literature incorporated traditional methods to address the shilling attacks problem, whether by implementing detectors or building recommenders with limitations in handling complex patterns and relationships in high-dimensional data [3-6]. Additionally, those methods may not fully address the problem of detecting shilling attacks effectively. In this paper, we address these limitations by proposing ROBUREC, a robust recommender system using autoencoders that exclusively focuses on user-item interactions, which makes it more resilient to rating manipulation by fake users. Or, in other words, more resilient to shilling attacks. Alongside its functionality as a recommender, the autoencoder works as an anomaly detector, where we compute the reconstruction errors that indicate if the dataset used has injected fake data or not. We tested the model on real-world datasets (MovieLens 100K, and MovieLens 1M), and evaluated its performance using Recall@k and NDCG@k. The main contributions of this paper can be summarized as follows: 1) Our proposed model incorporates Variational Autoencoders (VAE) to learn the latent representations for users by allowing domain adaptation, i.e., "Drama" and "Comedy", by which the model provides more accurate and personalized recommendations, 2) Our model considers user-item interaction history rather than the ratings, and by doing so, the

ASONAM '23, November 6–9, 2023, Kusadasi, Turkiye

ACM ISBN 979-8-4007-0409-3/23/11…$15.00
https://doi.org/10.1145/3625007.3630112

model is resilient against classic shilling attacks, and the model adapts another level of immunity insurance against shilling attacks, where it implements a thresholding error metric to identify anomalous interactions, by analyzing the reconstruction error between the original input vectors and their reconstructed counterparts. The paper is structured in section II, which gives a background regarding the famously used recommender systems. In section III, we discuss the different types of classic shilling attacks, and section IV sheds light on the state-of-art work in the literature, while section V explains the proposed model. Section VI presents the experimental results, and section VII is the conclusion.

II. Recommender Systems - Background

Many recommender systems are used across different domains [7]. In this section, we highlight some of the most used recommenders.

A. Collaborative Filtering

Collaborative Filtering (CF) is an approach based on recommending a user, i, items based on user/item neighbors [8]. CF-based recommender systems (RSs) take in a rating matrix, R, where $r_{ij} \in R$ represents the rating of user i on item j. There are two main approaches to CF, user-based and item-based [7]. User-based RSs calculate the similarity between a user, i, and all other users through similarity measures (Cosine Similarity, Pearson Coefficient, Spearman Coefficient, etc.) [2,3]. The set of users that are most similar to user i are referred to as i's neighbors; furthermore, i's neighbors are then examined, and items that are rated highly by the neighbors and not rated/watched/purchased by i are then recommended to user i [8]. On the other hand, item-based filtering calculates the similarity between an item j that user i has interacted with and all other items. The most similar items, j's neighbors, are then recommended to the user i. Moreover, CF-based models are often the most vulnerable to shilling attacks [9]. Due to their lack of context awareness and reliance on ratings. Although, they are often easy to implement and thus can serve as a reasonable basis for more complex models.

B. Content-Based Filtering

Models that rely on Content-Based Filtering (CBF) aim to predict a user's rating on unseen items [7]. CBF relies on numericizing items' features into values and calculating the similarity between the items a user has liked with all other items [2,5]. This similarity is calculated using metrics such as cosine similarity [10]. Moreover, tags are assigned to each user to create a user profile; this is done through applying machine learning methods [10]. The user profile reflects the user's tastes. CBF-based models then recommend the user the items with the highest similarity and that satisfy the user's profile. Furthermore, CBF-based models are more resistant to shilling attacks due to their user-specific nature and non-reliance on ratings. Although, if there is a lack of information, whether a minimal rating list or a lack of information to create a user profile, the recommendations will lack accuracy. This recommender system is prone to the cold-start problem, where new users or items don't have interaction or rating history [11].

C. Demographic-Based

Demographic Based (DB) RSs use user demographic data as the basis for the recommendation process [7]. Relevant demographic data includes location, age, language, and gender. Moreover, DB models incorporate the user's ratings and the data of similar users into the recommendation process [10]. Like CBF-based models, DB models offer individualized recommendations [12]; they also do not rely solely on ratings; thus, it can be said they are resistant to shilling attacks. Although, DB models will often not recommend diverse items to users with similar demographic data. Moreover, the ethical issues of accessing users' demographic data arise when DB models are implemented [13].

D. Utility-Based

Utility-based RS models generate a unique utility model for each item-user pair that reflects the usability of an item, j, to user i [14]. Utility models have certain utility-related attributes, such as price and brand, that are each assigned a unique weight for each user [15]. A user-specific utility score for each item is then calculated using a model-specific utility function based on these attributes, and all items are ranked [15]. The items with the highest utility scores are then recommended to the user i. Moreover, note that the weights of the attributes vary for each user over time to capture their changing tastes. Utility-based models do not rely on ratings; thus, they are more resistant to shilling attacks than CF-based models. However, a drawback of utility-based models is that when an item's page does not include sufficient information to fill up all the required attributes, the model will likely not recommend this item to a user, even though it could be a strong fit.

E. Context-Aware

Context-aware recommenders consider additional contextual information related to users or items, allowing for a more comprehensive and accurate recommendation process. In the context of user-based recommenders, contextual information could include factors such as the user's location, time of day, device type, or even their current mood or preferences. By incorporating such contextual data, the system gains a deeper understanding of the user's current needs and preferences, enabling it to generate more relevant and personalized recommendations. Similarly, in item-based recommenders, context refers to the attributes or characteristics of the recommended items. These context-aware recommenders are not designed to replace traditional collaborative filtering methods but are integrated into the existing collaborative filtering framework. The system can refine its understanding of user preferences and item characteristics by augmenting the user-item interaction data with contextual information, resulting in improved recommendation accuracy. Users receive more personalized and relevant suggestions, leading to higher user satisfaction and engagement with the recommendation platform. Additionally, context-aware recommenders can help address the cold-start problem, where new users or items have limited interaction data. By utilizing contextual information, the system can still make relevant recommendations even without extensive historical data [16][17].

F. Hybrid Based

Hybrid-based RS models combine several RS models to mitigate the disadvantages of each and yield better performance [18]. An example of a hybrid-based model is the weighted-hybrid recommender, which uses multiple approaches to calculate an item's affinity score and assigns a greater weight to approaches that predict more accurate scores over time. Another model is the switching hybrid-based RS, which can switch from one approach to another based on circumstance or user preference [10]. Often, hybrid-based models perform better than CF and CBF-based models, as they alleviate their many disadvantages, such as openness to shilling attacks.

Many recommendation systems, as shown in Figure 1, have been adopted in many crucial roles in various domains; hence, a trustworthy recommendation system is a must. Unfortunately, some attackers forge fake and misleading profiles to create biases in user ratings and manipulate the data used to generate recommendations. This attack is called profile injection or shilling attack [19].

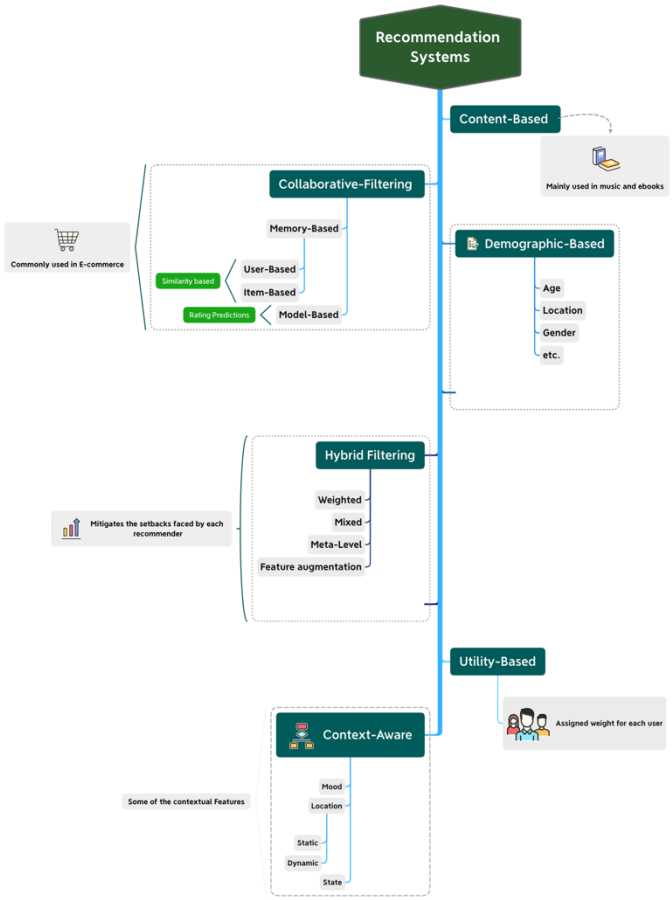

Fig. 1. The state-of-the-art Recommender Systems (RSs)

III. Shilling Attacks

Many types of shilling attacks target recommendation systems. To understand the impact of these attacks, we need to explore their various types to find defenses that can enhance the robustness of recommendation systems. There are two shilling attack types: a push attack and a nuke attack [20]. A push attack aims to assign the target item/items the highest possible rating such that it increases the number of users it's recommended to. On the other hand, a nuke attack aims to assign the targeted item/items the lowest rating possible, decreasing the number of users it's recommended to [21].

Moreover, all shilling attack models divide the set of all items, I, into four subsets: $I_s, I_F, I_\emptyset$ and I_t. Where, I_s is a set of specially selected items, I_F is a set of filler items that are randomly selected, and I_t is the set of items targeted by the shilling attack. Furthermore, the set $I_\emptyset$ is the set of all items unrated during the shilling attack. In addition, one of the main characteristics that define an attack is the filler size, which refers to the total number of ratings assigned in an attack; in other words, the sum of the items in the sets I_s, I_F and I_t ($Filler\ Size = |I_s| + |I_F| + |I_t|$) [21]. Also, the attack size is the number of fake profiles injected into the recommender system. Below, we shed light on the shilling attack types used against recommendation systems.

A. Random Attack

In a random attack, the set I_s is left null [22]. Additionally, the target items, I_t, are assigned either the maximum or minimum rating depending on whether the attack is a push attack or a nuke attack [21]. Furthermore, the filler items, I_F are randomly assigned ratings according to the standard distribution centered around the mean rating of all items. The benefit of this attack is that it does not require extensive knowledge of the item database to mount; although, it is quite ineffective [23].

B. Average Attack

The average attack model is identical to the random attack model, except for how ratings are assigned to filler items [24]. In an average attack, the ratings for the filler items are assigned by randomly selecting a rating from each item's normal distribution centered around the item's average rating. This attack model requires more knowledge of the item database compared to the random attack, although it is significantly more effective.

C. Love/Hate Attack

In the case of a push attack, all the target items are given the maximum possible rating, while all the filler items are given the minimum rating [25]. Furthermore, vice versa is done in the case of a nuke attack. This attack model requires almost no knowledge of the item database to mount, and it is effective, especially against user and item-based recommender systems.

D. Bandwagon Attack

In the bandwagon attack model, the elements of the item set I_s are selected to be the most popular items in the item database [26]. These items, alongside the target items I_t, are assigned the maximum possible rating. Moreover, there are two variants to the bandwagon attack model, the average bandwagon attack, and the random bandwagon attack [27]. The average bandwagon attack assigns ratings to items in I_F similarly to how they're assigned in the average attack model, while the random bandwagon attack assigns ratings similarly to how

they're assigned in the random attack model. The benefit of bandwagon attacks is that it requires public domain knowledge to mount, the set of the most popular items, yet it is still very effective.

In the case of a nuke attack, the reverse bandwagon attack model is employed. This model is identical to the bandwagon attack model; although, items in the sets I_t and I_s are assigned the minimum rating rather than the maximum rating. Furthermore, like the bandwagon model, there are two sub-models: the average reverse bandwagon attack and the random reverse bandwagon attack, the method in which these sub-models assign ratings to items in I_F are identical to the sub-model bandwagon attack model [21]. The reverse bandwagon attack model has shown strong performance against item-centric recommender systems. Table I summarizes the standard attacks.

Table I: Summary of Standard Shilling Attacks

Attack Model	Required Knowledge	Attack Profile			Intent
		Selected items	Filler Items	Target Items rating	
Random Attack	Low-knowledge	N/A	Random items	min/max	Push/ Nuke
Average Attack	High-knowledge	N/A	Random items	min/max	Push/ Nuke
Bandwagon Attack	Low-knowledge	Popular items	Random items	max	Push
Love/Hate Attack	Low-knowledge	N/A	Max rated items	min/max	Nuke

All the standard shilling attacks depend on manipulating the rating for some items/products. Our motivation was to develop a recommender system that exclusively considers the user-item interactions instead.

IV. Literature and Related Work

Recommendation systems are guiding users through the overwhelming abundance of choices in many domains. With that growing reliance on them, they also attract malicious users who seek to manipulate the output recommendations for their gain. These threats pose a severe challenge to the effectiveness of the recommender systems, as they can lead to biased and untrustworthy recommendations. Some strategies carried out by the researchers to face those threats are building robust recommendation systems [28] or constructing smart detectors [29]. Since we aim to build a robust recommender system, we highlight the research work addressing the development of resilient, robust recommenders. For example, the authors in [30] developed the Anti-FakeU model, which aims to complement Graph Neural Networks Recommender Systems (GNN RSs), which are vulnerable to shillings attacks, by implementing a GNN-based detector model. At the core of this model is a user-to-user graph where each vertex represents a user, and each weighted edge represents the similarity between each user and their neighbors. Within this graph, fake profiles will be in a cluster due to their similarity; thus, a Graph Convolutional Neural Network (GCN) is employed to identify this cluster. The predicted cluster is then fed into the GNN RS and is accounted for when making recommendations. On the Gowalla dataset, the Anti-FakeU model has greater recall and accuracy when it comes to detecting fake users, compared to the Principal Component Analysis (PCA), fraudulent action propagation (FAP), and Degree SAD models, in the cases of random, PowerUser, or Rev Attacks. However, the Degree SAD model outperforms it in the cases of popularity and RAPU attacks. Similarly, in [31], the researchers developed a clustering-based model to detect shilling attack groups. The model's first component places all the items every user has rated into a unique document, x, and then places all documents into a document corpus, X. Then, the HDP NLP model is applied to the corpus to acquire the item latent topics. The model's second component uses a specially designed algorithm to replace each item in a document x, with the numbers of related topics, thus constructing User vectors. The third and final component uses the K-means clustering algorithm to detect groups of fake user profiles. On both the Amazon and Netflix datasets, the HDP-KM model had greater recall and precision scores than the other tested models. Those models are the Catch the Black Sheep (CBS) model, a model which ranks users using spam probability; the DeR-TIA model, an unsupervised method that employs RDMA and DegSim (Degree of similarity) metrics; the GD-BKM model, another unsupervised method that also uses K-means clustering, and the Co-Detector model, which is a supervised model. Moreover, the HDP-KM model greatly outperforms traditional RDMA and Degsim-based models, as its recall and accuracy were folded higher than the DeR-TIA model. Another robust recommender system model was developed by the authors in [32], where this model employs various metrics to extract attack features, including a user's Decay-score, Rapidity-score, Diversity-score, RDMA, and DegSim. These attack features are inputted into the last hidden layer of the Neu-UHFLA neural network alongside the user-item latent feature vector. Moreover, the latent feature vectors are found by inputting the sparse rating matrix, R, into the bottom layer of the neural network, which is a fully connected layer. Furthermore, since this model uses neural matrix decomposition, which combines generalized matrix decomposition (GMF) and multi-layer perceptron (MLP), the latent feature vectors are inputted into both GMF layers and MLP layers independently, and both feature vectors are then inputted into the final hidden layer. On the MovieLens100K data set used for testing, the Neu-UHFLA model showed consistent precision and recall when detecting fake profiles, both greater than 0.9, across all attack sizes (1%, 3%, 5%, 10%).

Matrix Factorization (MF) was incorporated in [28], where the researchers aimed to develop a robust recommender model which combines two sequential MF methods to yield reliability predictions regarding the fakeness of profiles; this model is referred to as the RBM model. The first method functions by taking in the sparse interactions/rating matrix R, and applying MF to predict the missing elements in R; then, the error between the predicted data and the original data is calculated. These predictions are then inputted into the second MF method in matrix form. This method calculates the reliabilities of these predictions and thus deduces which item-user interactions are likely due to a shilling attack. When tested on the

MovieLens1M set, the RBM model exhibited outstanding performance in the case of love-hate, perfect-knowledge, and average attacks, scoring above a 4.5 in the prediction value of target item metric in the case of 125-shilling attack profiles or greater (2% attack size). Moreover, the model was slightly less impressive against bandwagon-random, random, and bandwagon average attacks; being weak against random attacks is a notable limitation as they're one of the most used attacks. Furthermore, the model was not tested against other detector models.

Most of the existing literature considered the rating in their models, which still can make their models vulnerable to shilling attacks. Our main contribution is to use the user-item interaction instead of the ratings, which adds more resilience against classic attacks.

V. Proposed Model- ROBUREC

In this paper, we propose a robust deep learning-based collaborative-filtering recommendation system, ROBUREC, that takes in a user-item interaction vector, $X \in \{0,1\}^{|I|}$ as an input (note that $|I|$ indicates the cardinality of the set of items). This interaction vector is binary, meaning each element $x_j \in X$ is either a 1 or a 0, where a 1 indicates that the user has interacted (i.e., rated, purchased, clicked on, etc.) with the j-th item, and a 0 indicates otherwise. Using a binary interaction vector, rather than a user ratings' vector, creates a model more resistant to shilling attacks, as items can no longer be pushed or nuked. Moreover, our model outputs a reconstructed vector, $X' \in R^{|I|}$, where each $x'_j \in X'$ contains the probability that the user will interact with the j-th item. This vector is then sorted, and the items with the highest probability are recommended to the user.

The deep learning model is implemented using a variational autoencoder (VAE), where it has two processes: (i) An encoder that takes the user-item interaction matrix as an input and maps it to a latent representation z. Our encoder consists of multiple layers to gradually reduce the input's dimensionality. (ii) The second process consists of the decoder, which takes the latent representation z as input and reconstructs the original input data. The final layer of the decoder uses a sigmoid activation function to map the output values to the range $[0,1]$, representing the item-user interaction probability. Figure 2 depicts a high level of our proposed model's Architecture.

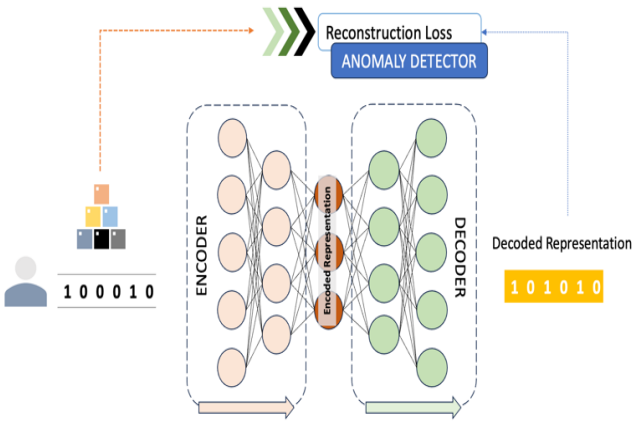

Fig.2. The Proposed Model's High-Level Architecture

The model is trained using two real-world datasets (MovieLens 100K [33] and MovieLens 1M [34]); this phase also serves as a baseline for detecting anomalies. The difference between the reconstructed interactions and the original input is quantified as the reconstruction error as described in equation 1, where the higher the reconstruction error, the higher the indication of anomalies.

$$Reconstruction\ Error\ (MSE)_{(i,j)} = \left(M_{[i,j]} - M'_{[i,j]}\right)^2 \quad (1)$$

Where $M_{[i,j]}$ is the interaction for user-item (i,j), and$M'_{[i,j]}$ is the reconstructed interaction. We evaluate the recommender using recall@k and normalized discounted cumulative gain (NDCG@K) metrics, as calculated in equations 2 and 3.

$$Recall@k = \frac{No.of\ generated\ relevant\ recommendations}{Total\ relevant\ items} \quad (2)$$

$$NDCG@k = \frac{DCG@k}{IDCG@k}, \quad (3)$$

- DCG@k is the Discounted Cumulative Gain at position k.
- IDCG@k is the Ideal Discounted Cumulative Gain at position k.

$$DCG@K = rel1 + \sum i = 2Klog2(i+1)reli, \quad (4)$$

- *reli* is the relevance score of the item at position i in the ranked list of recommendations.

VI. Experimental Results

ROBUREC is experimented on two real-world datasets, MovieLens 100K and MovieLens 1M, since we needed to check the scalability performance of our model. The model works on the whole dataset or genre-based or domain-based sub-datasets. For instance, in our experiment, the datasets are loaded into two domains, 'Comedy' and 'Drama', and we used 80%-20% training and testing splits. The VAE is trained using Adam optimizer, where we iterated over several epochs. The results are shown below in Figures 3-10, captured for each genre/domain.

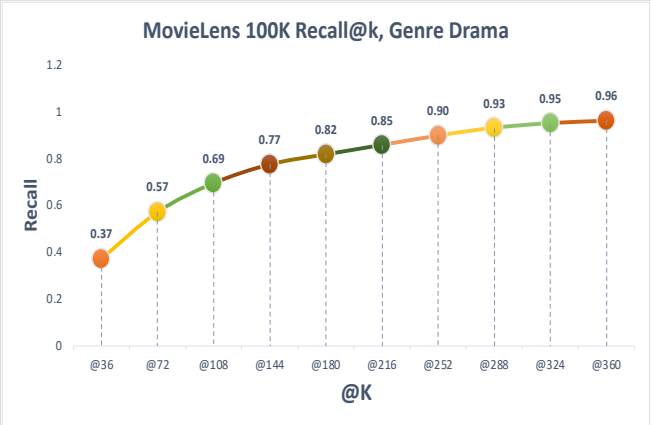

Fig.3. MovieLens 100K Recall@k, Drama

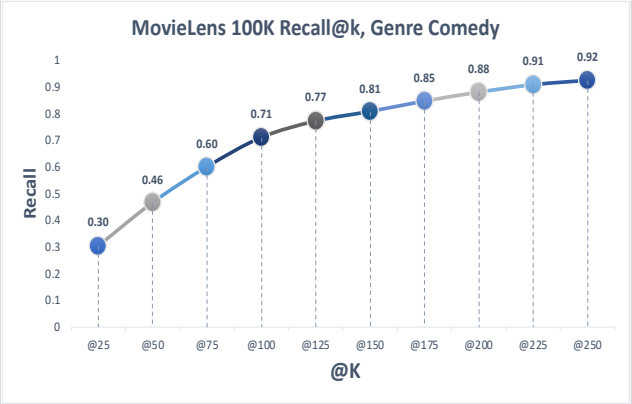

Fig.4. MovieLens 100K Recall@K, Comedy

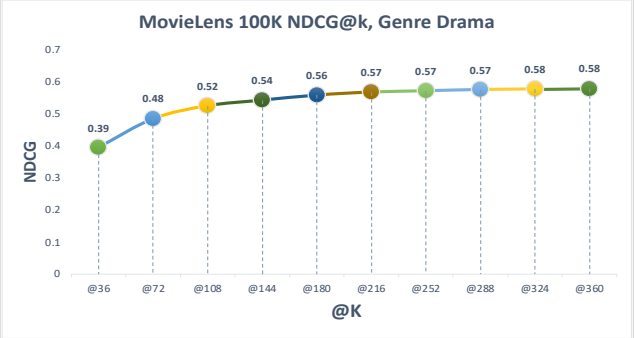

Fig.5. MovieLens 100K NDCG@k, Drama

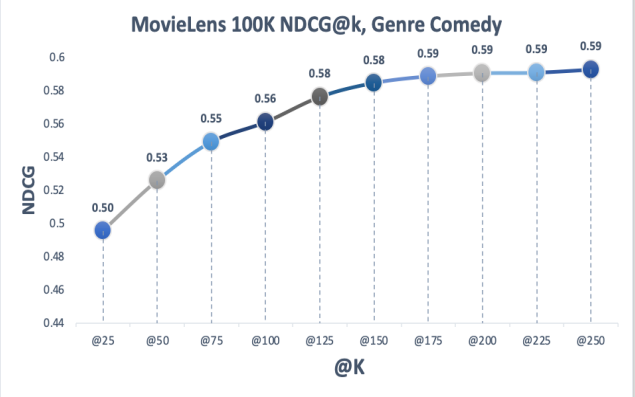

Fig.6. MovieLens 100K NDCG@k, Comedy

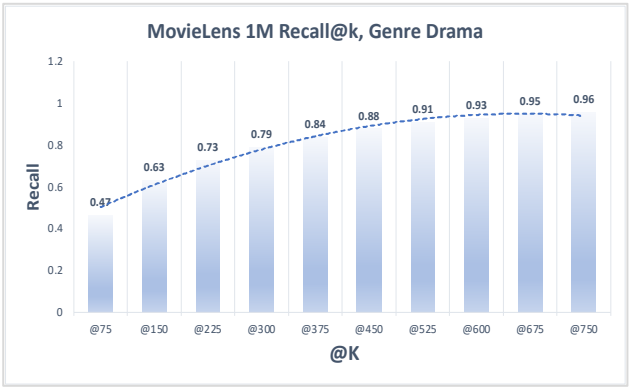

Fig.7. MovieLens 1M Recall@k, Drama

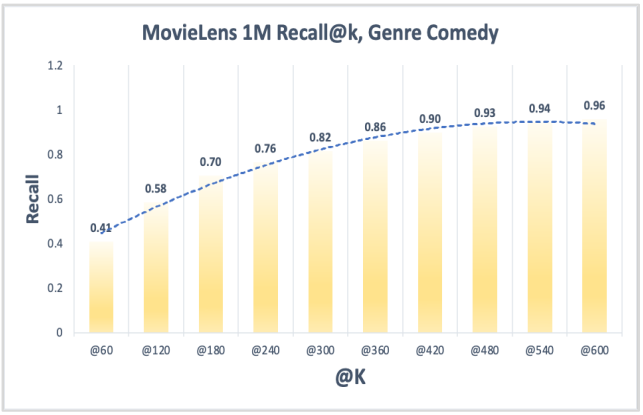

Fig.8. MovieLens 1M Recall@k, Comedy

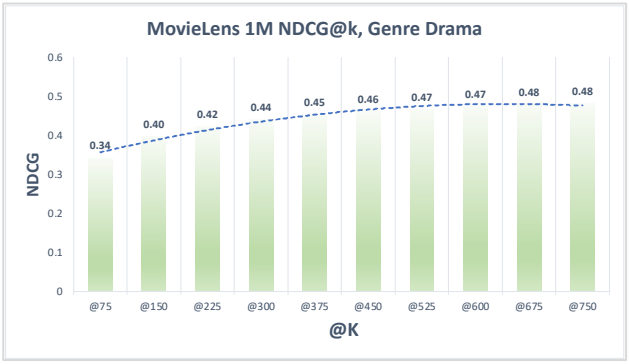

Fig.9. MovieLens 1M NDCG@k, Drama

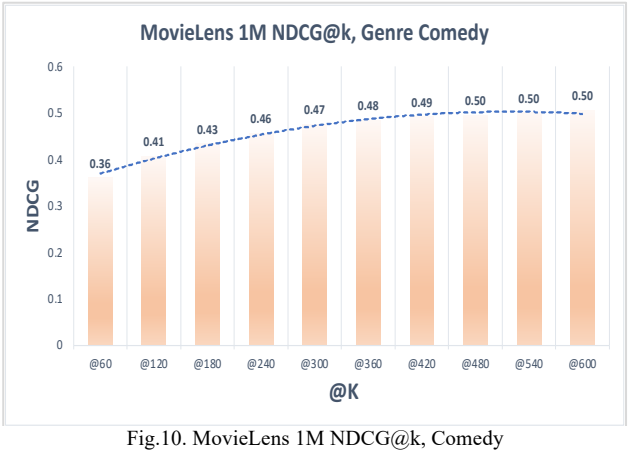

Fig.10. MovieLens 1M NDCG@k, Comedy

As mentioned in the above sections, autoencoders can work as anomaly detectors in recommendation systems [35] by comparing the reconstructed vector against the input one, where higher error values indicate that the input data has been tampered with. Figures 11 and 12 depict the MSE obtained for MovieLens 100K and 1M with slight tampering with the input vectors by adding noise, where we highlight the percentage of MSE increase when the model has the injected data as an input.

In addition to the above evaluation, we benchmark our proposed model against the state-of-art ones [3], [4] where ROBUREC could outperform the rest of the models in terms of Mean Average Recall MAR@k and Mean NDCG@k when MovieLens 1M is used. Figure 13 shows that ROBUREC outperforms both D2D-TM and Multi_VAE models.

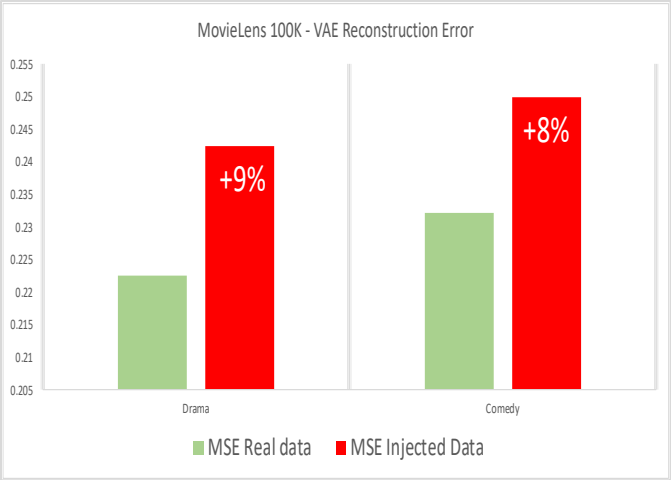

Fig.11. MovieLens 100K Reconstruction Error

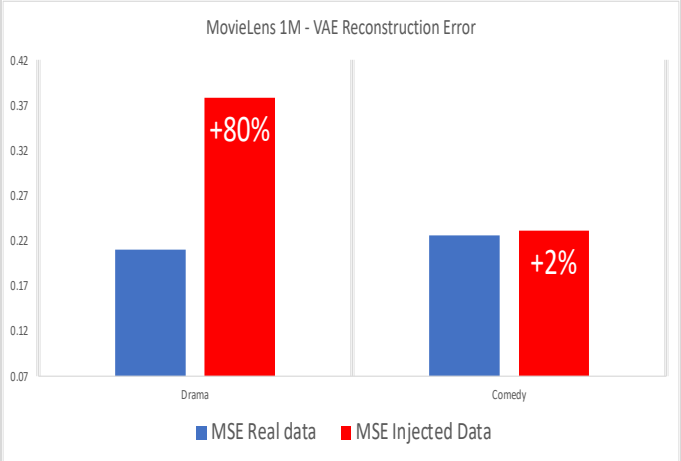

Fig.12. MovieLens 1M Reconstruction Error

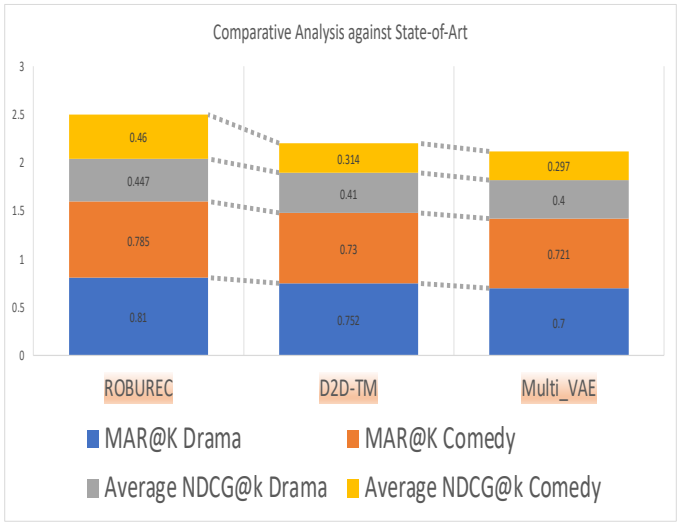

Fig.13. ROBUREC in comparison with state-of-art-models

VII. Conclusion and Future Directions

This paper proposes ROBUREC, a deep-learning model for collaborative-filtering recommendation systems based on variational autoencoders (VAE). Our model utilizes user-item interaction history to learn the latent representation of the input data. We experimented with our model on two real-world datasets, MovieLens 100K and Movielens 1M, and evaluated it using Recall@K, and NDCG@k. The reconstruction error is utilized as a feature in the model to identify anomalies. A higher reconstruction error indicates a higher likelihood of anomalies in the input data. ROBUREC can work on domain-based data, allowing filtering on different genres such as ('Drama' and 'Comedy') experimented by our model. We also compared our model against the state-of-art ones, where our model outperformed them in terms of MAR@K, and MeanNDCG@k. ROBUREC showed efficiency with enhanced anomaly detection capabilities. In future work, we will consider different noise distributions on the input data to map more advanced attack types. Further exploration of peer-to-peer architectures [36][37] is essential to assess how the suggested decentralized models perform in comparison to alternative distributed network frameworks. As we look ahead, the adoption of blockchain [38]-[41] is expected to grow as organizations seek innovative ways to streamline processes, reduce fraud, and enhance transparency.

VIII. References

[1] B. C. Kwon, S.-H. Kim, T. Duket, A. Catalán, and J. S. Yi, "Do People Really Experience Information Overload While Reading Online Reviews?," *International Journal of Human–Computer Interaction* , vol. 31, 2015.

[2] C. Li and Z. Luo, "Detection of shilling attacks in collaborative filtering recommender systems," in *2011 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2011.

[3] L. Nguyen and T. Ishigaki, "D2D-TM: A Cycle VAE-GAN for Multi-Domain Collaborative Filtering," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019.

[4] D. Liang, R. G. Krishnan, M. D. Hoffman and T. Jebara, "Variational Autoencoders for Collaborative Filtering," in *WWW '18: Proceedings of the 2018 World Wide Web Conference*, 2018.

[5] F. Zhang and S. Wang, "Detecting Group Shilling Attacks in Online Recommender Systems Based on Bisecting K-Means Clustering," *IEEE Transactions on Computational Social Systems,* vol. 7, no. 5, 2020.

[6] F. W. a. b, M. G. a. b, J. Y. c, Z. W. b, K. L. d and X. W. e, "Ready for emerging threats to recommender systems? A graph convolution-based generative shilling attack," *Information Sciences,* vol. 578, 2021.

[7] Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim and R. Kashef, "Recommendation systems: Algorithms, challenges, metrics, and business opportunities," *applied sciences,* vol. 21, 2020.

[8] G. Gupta and R. Katarya, "Recommendation analysis on item-based and user-based collaborative filtering," in *International conference on smart systems*, 2019.

[9] M. Ebrahimian and R. Kashef, "Efficient Detection of Shilling's Attacks in Collaborative Filtering Recommendation Systems Using Deep Learning Models," in *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 2020.

[10] M. H. Mohamed, M. H. Khafagy and M. H. Ibrahim, "Recommender Systems Challenges and Solutions Survey," in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, 2019.

[11] S. Natarajan, S. Vairavasundaram, S. Nataraja and A. H. Gandomi, "Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data," *Expert Systems with Applications,* vol. 149, 2020.

[12] A. Abu-Issa, H. Nawawreh, L. Shreteh, Y. Salman, Y. Hassouneh, I. Tumar and M. Hussein, "A Smart City Mobile Application for Multitype, Proactive, and Context-Aware Recommender System," in *International Conference on Engineering and Technology (ICET)*, 2017.

[13] S. Milano, M. Taddeo and L. Floridi, "Recommender systems and their ethical challenges," *AI & SOCIETY ,* vol. 35, 2020.

[14] Y. Zheng, "Utility-based multi-criteria recommender systems," in *SAC '19: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019.

[15] P. Jariha and S. K. Jain, "A state-of-the-art Recommender Systems: An overview on Concepts, Methodology and Challenges," in *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, 2018.

[16] D. Nawara and R. Kashef, "Context-Aware Recommendation Systems in the IoT Environment (IoT-CARS)–A Comprehensive Overview," *IEEE Access,* vol. 9, 2021.

[17] D. Nawara and R. Kashef, "MCARS-CC: A Salable Multicontext-Aware Recommender System," *IEEE Transactions on Computational Social Systems ,* 2022.

[18] B. Walek and V. Fojtik, "A hybrid recommender system for recommending relevant movies using an expert system," *Expert Systems with Applications,* vol. 158, 2020.

[19] A. K. Verma and V. S. Dixit, "A Comparative Evaluation of Profile Injection Attacks," in *Advances in Data and Information Sciences*, Singapore, Springer, 2018.

[20] M. Si and Q. Li, "Shilling attacks against collaborative recommender systems: a review," *Artificial Intelligence Review,* vol. 53, 2020.

[21] N. Praveena and K. Vivekanandan, "A Survey on Detection Approaches of Shilling Attacks in SAN," in *5th International Conference on Computing Methodologies and Communication (ICCMC)*, 2021.

[22] P. K. Patiala and S. Goel, "Shilling attack models in recommender system," in *2016 International Conference on Inventive Computation Technologies (ICICT)*, 2016.

[23] B. R, W. CA, M. B and B. RD, "Securing collaborative filtering against malicious attacks through anomaly detection," in *4th workshop on intelligent techniques for web personalization*, 2006.

[24] B. Mobasher, R. Burke, R. Bhaumik and C. Williams, "Effective Attack Models for Shilling Item-Based Collaborative Filtering Systems," in *Proc. of the 2005 WebKDD Workshop*, 20015.

[25] F. Zhang, "Analysis of Love-Hate Shilling Attack Against E-commerce Recommender System," in *International Conference of Information Science and Management Engineering*, 2010.

[26] Z. Liu and M. Larson, "Adversarial item promotion: Vulnerabilities at the core of top-n recommenders that use images to address cold start," in *Proceedings of the Web Conference*, 2021.

[27] T. Kumari and P. Bedi, "A Comprehensive Study of Shilling Attacks in Recommender Systems," *International Journal of Computer Science Issues (IJCSI),* vol. 14, 2017.

[28] S. Alonso, J. Bobadilla, F. Ortega and R. Moya, "Robust Model-Based Reliability Approach to Tackle Shilling Attacks in Collaborative Filtering Recommender Systems," *IEEE Access ,* vol. 7, 2019.

[29] A. P. Sundar, F. Li, X. Zou, T. Gao and E. D. Russomanno, "Understanding Shilling Attacks and Their Detection Traits: A Comprehensive Survey," *IEEE Access,* vol. 8, 2020.

[30] X. You, C. Li, D. Ding, M. Zhang, F. Feng, X. Pan and M. Yang, "Anti-FakeU: Defending Shilling Attacks on Graph Neural Network based Recommender Model," in *WWW '23: Proceedings of the ACM Web Conference 2023*, 2023.

[31] S. Wang, H. Wang, H. Yu and F. Zhang, "Detecting shilling groups in recommender systems based on hierarchical topic model," in *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, 2021.

[32] T. U. o. T. T. C. Wanqiao Yuan Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Y. Xiao, X. Jiao and Y. Ming, "Neural Network Detection of Shilling Attack Based on User Rating History and Latent Features," in *2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, 2019.

[33] *https://grouplens.org/datasets/movielens/100k/.*

[34] *https://grouplens.org/datasets/movielens/1m/.*

[35] X. Wang, H. Zhao, Y. Wang, H. Tao and J. Cao, " Supervised Prototypical Variational Autoencoder for Shilling Attack Detection in Recommender Systems," in *International Conference on Data Mining and Big Data*, 2023.

[36] Manjunath, Y. S. K., & Kashef, R. F. (2021). Distributed clustering using multi-tier hierarchical overlay super-peer peer-to-peer network architecture for efficient customer segmentation. Electronic Commerce Research and Applications, 47, 101040.

[37] Kashef, R., & Niranjan, A. (2017, December). Handling Large-Scale Data Using Two-Tier Hierarchical Super-Peer P2P Network. In Proceedings of the International Conference on Big Data and Internet of Thing (pp. 52-56).

[38] Jebamikyous, H., Li, M., Suhas, Y., & Kashef, R. (2023). Leveraging machine learning and blockchain in E-commerce and beyond: benefits, models, and application. Discover Artificial Intelligence, 3(1), 3.

[39] Saleminezhadl, A., Remmele, M., Chaudhari, R., & Kashef, R. (2021). IoT Analytics and Blockchain. arXiv preprint arXiv:2112.13430.

[40] Schmid, P., Schaffhäuser, A., & Kashef, R. (2023). IoTBChain: Adopting Blockchain Technology to Increase PLC Resilience in an IoT Environment. Information, 14(8), 437.

[41] Yeh, T. Y., & Kashef, R. (2020). Trust-Based collaborative filtering recommendation systems on the blockchain. Advances in Internet of Things, 10(4), 37-56.