Apache Spark Based Deep Learning for Social Transaction Analysis

Raouf Jmal[1][a], Mariam Masmoudi[2,4][b], Ikram Amous[1][c], Corinne Amel Zayani[3][d] and Florence Sèdes[4][e]

[1]*MIRACL, Enet'Com, Sfax University, Sfax, Tunisia*
[2]*MIRACL, FSEGS, Sfax University, Sfax, Tunisia*
[3]*MIRACL, FSS, Sfax University, Sfax, Tunisia*
[4]*IRIT, Paul Sabatier University, Toulouse, France*

Keywords: Social Network, Social transaction, Trust-attacks, Apache Spark, Spark Streaming, Deep Learning, Elephas.

Abstract: In an attempt to cope with the increasing number of trust-related attacks, a system that analyzes the whole social transaction in real-time becomes a necessity. Traditional systems cannot analyze transactions in real-time and most of them use machine learning approaches, which are not suitable for the real-time processing of social transactions in the big data environment. Therefore, in this paper, we propose a novel deep learning detection system based on Apache Spark that is capable of handling huge transactions and streaming batches. Our model is made up of two main phases: the first phase builds a supervised deep learning model to classify transactions (either benign transactions or malicious transactions). The second phase aims to analyze transaction streams using spark streaming, which transforms the model to batches of data in order to make predictions in real-time. To verify the effectiveness of the proposed system, we implement this system and we perform several comparison experiments. The obtained results show that our approach has achieved more satisfactory efficiency and accuracy, compared to other works in the literature. Thus, it is very suitable for real-time detection of malicious transactions with large capacity and high speed.

1 INTRODUCTION

Nowadays, due to the rapid development of network technologies, social networks have been growing at an incredible rate based on online networking sites (Chun et al., 2008). These networks have become part of people's social lives instead of their real social ways, in which humans make friends, communicate with each other or share data such as video games, movies, pictures and songs (Boyd and Ellison, 2007). On these social sites, users also can comment on other profiles and send private messages. Thus, communication can be defined as a social transaction (Masmoudi et al., 2021), which means the interaction between two users resulting in a change of states or relationships between these users.

[a] https://orcid.org/0000-0002-5761-2353
[b] https://orcid.org/0000-0001-5043-864X
[c] https://orcid.org/0000-0002-5893-9833
[d] https://orcid.org/0000-0002-3296-1020
[e] https://orcid.org/0000-0002-9273-302X

Furthermore, many users always search for ways to establish virtual relationships without meeting people. Thus, they accept the associated risks against the possible benefits that they are convinced to obtain. In this respect, making new relationships and transactions can generate benefits despite their risks. This equality is a fundamental consideration for users' personal safety. However, to trustworthy transactions and interactions between users become a necessity to ensure users' safety and security and identify malicious users. Hence, transactions must be evaluated in order to distinguish malicious transactions from benevolent ones. These malicious transactions are known as trust attacks based on the literature (Masmoudi et al., 2021).

In fact, myriad works in the literature have been proposed to deal with these attacks. Yet, most of these works (Rajesh et al., 2016), (Jayasinghe et al., 2018), (Marche and Nitti, 2020), (Zheng et al., 2021) and (Lee and Jun, 2018) have focused on non-real-time trust attack detection in order to remove the nodes that provide malicious behaviors from the network. With-

Jmal, R., Masmoudi, M., Amous, I., Zayani, C. and Sèdes, F.
Apache Spark Based Deep Learning for Social Transaction Analysis.
DOI: 10.5220/0012202600003584
In *Proceedings of the 19th International Conference on Web Information Systems and Technologies (WEBIST 2023)*, pages 365-372
ISBN: 978-989-758-672-9; ISSN: 2184-3252

out real-time detection, malicious transactions will be passed to the next peer before being detected by the model. These works have applied statistical models and machine learning techniques to detect malicious nodes. Thus, these techniques have several disadvantages, such as mass data processing and feature engineering that requires human intervention as well as a dynamic update system.

Hence, in this paper, we use social network transaction analysis in order to distinguish real-time malicious transactions from benevolent ones. For this reason, we propose a deep learning model based on Apache Spark and spark streaming to process and analyze stream transactions. This model: (i) can process huge transactions efficiently so that we can analyze and bock each transaction in real-time, (ii) is robust enough so that failure will not abort the whole streaming process and (iii) is able to classify transactions into two different classes (either benevolent transactions or malicious transactions).

The remainder of this paper is structured as follows: In section 2, we analyze and compare related-works on attacks and intrusion detection systems using Apache Spark. In section 3, we not only describe our architecture and its phases, but also define the features that will be used to train and create the DL model. In section 4, we show the performance of the proposed approach and experimental results. In Section 5, we discuss the outcomes and implications of our research while emphasizing the importance of future studies. We identify and highlight key directions for further exploration and advancement within the field, opening up opportunities for future researchers to build upon our findings. In Section 6, we summarize the key outcomes and implications of our work.

2 RELATED WORK

Several works, (Abderrahim et al., 2017), (Ekbatanifard and Yousefi, 2019), (Talbi and Bouabdallah, 2020), (Chen et al., 2015), (Meena Kowshalya and Valarmathi, 2017) and (Jafarian et al., 2020), have been suggested to detect trust attacks by evaluating trust associated with transactions. Besides, these models could not detect attacks in real-time, which reflect their inefficiency. However, most works have identified attacks from past transactions (after validation).

In (Abdelghani et al., 2018), authors put forward a machine learning-based trust evaluation model in order to detect malicious nodes by classifying past transactions into two major classes (attacks and none-attacks) using some features related to the four types of trust-related attacks (SPA, BMA, BSA, DA).

(Masmoudi et al., 2019) set forth a trust-related attack detection model based on deep learning to identify the four types of trust-related attacks (BMA, SPA, BSA and DA). This model has performed better results with high Recall (94.4%) and accuracy (95.68%), compared to the work proposed by (Abdelghani et al., 2018), but there was no real-time detection.

In order to prevent all types of trust-related attacks (BMA, BSA, SPA, DA, OSA and OOA) authors, in (Masmoudi et al., 2021), offered a consensus protocol named “PoTA” for the blockchain technology. This protocol is based on a classification technique; named Support Vector Machine. The latter is able not only to determine whether the completed transaction is an attack or not, but also to decide whether to accept or reject a transaction. The model recorded better results with F-measures of 5.22%, compared to the study carried out by (Masmoudi et al., 2019).

In contrast, for real-time attack detection, (Azeroual and Nikiforova, 2022), presented a prototype intrusion detection system that aims to detect anomalies in data through machine learning techniques by using the k-means algorithm for Spark MLlib based cluster analysis. Also, they provided an example of how big data technologies and the above-mentioned services can be used not only for everyday tasks, but also for the protection of all the data produced, collected, processed and transferred.

In (Awan et al., 2021), authors applied two machine learning approaches, namely Random Forest (RF) and Multi-Layer Perceptron (MLP) through the Scikit-ML and the Spark-ML libraries for the detection of DOS attacks. In terms of accuracy, they achieved similar mean accuracy in both approaches. However, in terms of training time and testing time, the big data approach outperforms the non-big data approach since Spark computations in memory happen in a distributed manner.

(Khan and Kim, 2020) developed an Intrusion detection system using Spark and Convolutional-auto encoder (Conv-AE) based deep learning approach. Thus, the conventional ML classifier used Spark MLlib to detect data anomalies, while the Conv-AE deep learning approach is used for misuse detection. The evaluation showed that their system is better than advanced approaches in terms of attack detection accuracy. Whereas the proposed approach did not perform detection in real time. It detected intrusions over a long period of time, as it passed by both anomaly detection and misuse detection.

Several other inherent streaming engines, such as Apache Storm and S4, support native streaming that immediately process data as it arrives. On the other hand, native streaming systems generally have lower throughput. Furthermore, fault tolerance and load balancing for native streaming are more expensive than micro-batching systems, (Singh et al., 2016) and (Hirzel et al., 2014). However, various studies used spark streaming to detect malicious attacks in real time. (Zhang et al., 2018) recommended a real-time detection system using a Random Forest algorithm based on the Apache Spark framework to detect intrusions in high-speed networks. The model used Apache Kafka to continuously send data to spark streaming for processing.

In this study (Zhou et al., 2018), the authors compared three machines-learning algorithms (Naıve Bayes, Decision Tree and Logistic Regression) based on the online DDoS attack detection system using spark streaming. However, the authors did not use Spark MLlib with Spark streaming to build ML algorithms which decrease the speed of the detection system.

(Abid and Jemili, 2020) suggested a graph-based real-time Intrusion-Detection System (IDS) to detect and classify intrusions using the Spark Machine Learning library and Spark Structured Streaming. Their system achieved great results with good processing speed using only a small cluster. Nonetheless, this approach was not compared to other works.

According to these works and comparisons reported in table 1, we notice that most works used machine learning techniques based on Apache Spark to detect either attacks or intrusions. Nonetheless, deep learning can be used to reconcile complex nonlinear relationships between variables and build up complex behavioral patterns more successfully than machine learning and statistical techniques (Yue et al., 2021). Then, we will use deep learning based on Apache Spark to detect malicious transactions (trust-related attacks) in real-time.

3 SPARK-BASED DEEP LEARNING FOR SOCIAL TRANSACTION ANALYSIS

In our system, we have incorporated a deep learning model on top of Apache Spark. However, it should be noted that Apache Spark does not inherently include a deep learning library, which can complicate the deployment of such models. To overcome this challenge, we utilized the Elephas extension [1], which enables the deployment of deep learning models with Spark. This approach has allowed us to create a system with reduced latency for classifying transactions, ensuring efficient and timely processing. Figure 1 depicts our architecture, which comprises two phases. In the first phase, the Spark-based deep learning model creation, we begin by preprocessing the dataset. Subsequently, a deep learning model is generated and trained. This phase yields a classification model based on Deep Learning capable of producing two transaction labels: malicious and benevolent. Moving to the second phase, spark streaming for social transaction analysis, we read data from a transaction stream and apply specific transformations. Ultimately, the trained model is employed to classify and assign labels to the transaction stream.

[1] https://github.com/maxpumperla/elephas

In the following sub-section, we will present the features used to train the model and provide a detailed overview of the two proposed phases.

3.1 Spark-Based Deep Learning Model Creation Phase

In this phase, we leverage the deep learning model based on Spark to aggregate transaction elements and detect malicious transactions. Transaction elements play a crucial role in distinguishing between malicious and benign transactions. These elements serve as features in training our model. Previous studies in the literature have explored various transaction features to detect trust-related attacks. However, for our approach, we will utilize the features proposed in (Masmoudi et al., 2021) as a basis for our analysis. These features are defined as follows:

- Quality of provider: refers to the Quality of Service (QoS) provided by a user whatever good or bad services.
- User Similarity: refers to the similarity between two users.
- Rating-Frequency: represents the frequency of rating attributed by one user to another.
- Rating-trend: This feature aims to reveal if a user is rather optimistic or pessimistic.
- Vote: Means that a user gives a voting value to the service of another user.
- Trust value: The overall trust value of user U in a social network.
- Vote similarity: refers to the similarity between a user vote in such service and other users' votes.

The features utilized in our approach are detailed in (Masmoudi et al., 2021), (Masmoudi et al., 2019) and (Abdelghani et al., 2018). Once we have defined

Table 1: A Comparison of attacks detection approaches.

Authors	Objectives	Algorithms	Techniques/libraries	Real-time
(Awan et al., 2021)	DDoS attack detection system	Multi-Layer Perceptron (MLP) and Random Forest (RF)	Spark MLlib Scikit ML	X
(Azeroual and Nikiforova, 2022)	Intrusion detection system (IDS)	K-means	Spark MLlib	X
(Khan and Kim, 2020)	IDS	LR, DT, SVM and conv_AE (without Spark)	Spark MLlib	X
(Patil et al., 2022)	Distributed classification system for DDoS attacks	DecisionTree (DT), NaiveBayes (NB), Multinomial Logistic Regression(MLR), and Random Forest(RF)	Spark MLlib spark streaming Apache kafka Hadoop	✓
(Zhou et al., 2018)	DDoS attack detection system	NB, DT and Logistic Regression	Spark streaming Apache kafka Jpcap (an open-source Java library)	✓
(Abid and Jemili, 2020)	Real-time IDS	K2 algorithm	Spark MLlib spark streaming	✓
(Zhang et al., 2018)	Real-time IDS	Random Forest	Spark MLlib spark streaming Apache kafka	✓
(Masmoudi et al., 2019)	Trust-attack detection system (BMA, SPA, BSA, DA)	MLP	-	X
(Abdelghani et al., 2018)	Trust evaluation model (No specified attacks)	MLP, Naive Bayes, and Random Tree	-	X
(Masmoudi et al., 2021)	Consensus protocol to prevent trust-related attacks (BMA, BSA, SPA, DA, OSA, OOA)	SVM	Blockchain	X

these features and constructed our dataset, we initiate a Spark application, which grants us access to powerful libraries. For example, Spark ML offers various functions to train our dataset, while Spark SQL aids in creating a Spark context, reading the dataset as a dataframe, and facilitating visualizations of transformations. Using the Keras library, we generate a deep learning model by constructing a series of consecutive Dense layers with Dropout and activation functions. To integrate this Keras model with Spark, we define an estimator on top of it. We use the Elephas estimator, an extension of Keras, which enables distributed deep learning models to be executed at scale using Spark. Elephas aims to maintain the simplicity and availability of Keras, facilitating the rapid prototyping of distributed models that can handle large datasets.

3.2 Spark Streaming for Social Transaction Analysis Phase

During this phase, we utilized the Spark Streaming library to predict transaction labels in real-time. The initial step involved preparing multiple new transactions, with each transaction stored in a separate file for testing the model. We started by launching a Spark session and creating a schema to ensure that the streaming data adheres to the correct data types in the transaction files. Next, we configured the stream reading parameters, including the maximum number of new files to consider in each trigger and the file path and formats. Once the stream started, each new file in the specified directory was automatically processed. We then applied the same pipeline used in the data preprocessing phase to the data stream using the transform function. With the prepared data frame stream, we were able to make predictions using the trained model.

To regulate the stream processing, we set trigger parameters to determine the maximum time interval for triggering processing. We also incorporated a

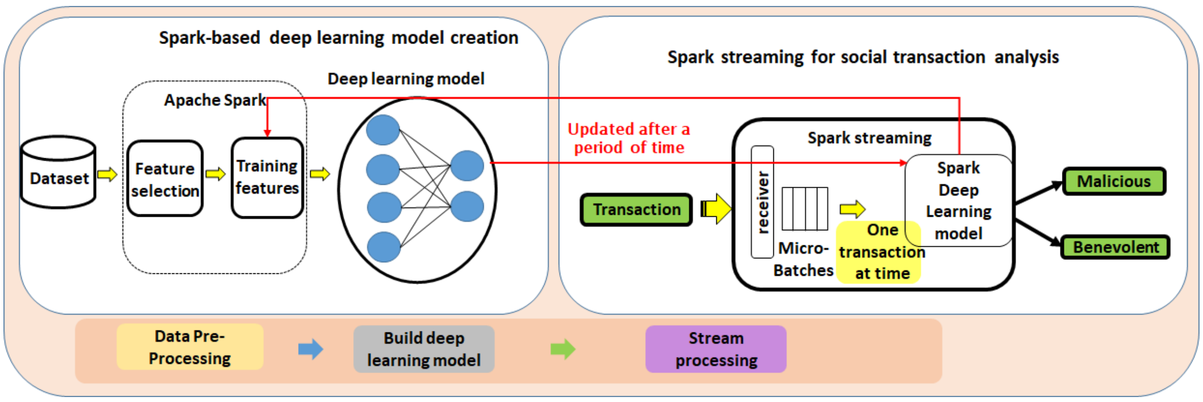

Figure 1: Spark-based deep learning architecture for transaction analysis.

threading function, where the written stream activated the thread and then entered a sleep mode for a short time. This threading mechanism allowed for concurrent execution of multiple tasks, suspending the calling thread for a few seconds. To ensure the streaming query runs and waits for the prediction of each trigger, we utilized a function. After making real-time predictions, each classified transaction was added to the training data, and the model was re-fitted to incorporate the updated information.

4 EXPERIMENTAL EVALUATION

This section aims to validate the performance of the spark streaming based deep learning model and to compare it with other models available in literature.

4.1 Simulation Setup

This part provides an overview of the dataset that will be used in experimentation and the evaluation metrics that will be applied to evaluate the performance of our approach.
In fact, the realization of the proposed system and all different simulations was carried out using an ”ASUS” laptop with specific properties. The laptop is equipped with an Intel Core i5 processor and has 8GB of RAM memory. It operates with a clock frequency of up to 3.1 GHz and runs on the Windows 10 operating system. These hardware specifications were instrumental in conducting the various simulations and implementing the proposed system effectively. Additionally, the simulation was built upon Apache Spark, an open-source unified analytics engine designed for large-scale data processing. For the development process, Visual Studio Code (Vscode) served as the code editor of choice. Python was the primary programming language used for the development of our deep learning model within this environment.

4.1.1 Data-Set

In order to test our approach, we need a large dataset social transaction elements. To this end, we made use of simulations applied to a real dataset named “Sigcomm [2]”. The latter comprises 76 users, 364 services, 300 devices, 711 users’ interests, 531 social relationships between users, 32000 transactions between users and 285788 proximities. Using this dataset, we performed some simulations in order to generate different instances. These instances are composed of various features related to malicious transactions and benevolent transactions. Based on these simulations, we created a CSV file based on 3200 transactions.

4.1.2 Evaluation Metrics

In this section, we define the evaluation metrics used to test the proposed approach. However, we aim to classify each transaction stream in real-time based on spark streaming. In fact, to evaluate the performance of the proposed system, we considered two cases: case (I) designed to assess the performance of the model in predicting transaction classes and case (II) used to determine the performance of the streaming workflow.

In case (I), we have used a performance metric as defined in table 2 in order to validate the performance of our classification model. To evaluate case (II), Spark provides a web UI to monitor and inspect the status and resource consumption of a Spark application in a web browser. It presents different parameters, such as Spark jobs that show the status, duration, and progress of all jobs and the overall event timeline. It checks for more information about the environment,

[2]https://crawdad.org/thlab/sigcomm2009/20120715/

Table 2: Evaluation metrics evaluation for the classification model.

Metrics	Definition
Precision	PPV = TP / (TP + FP)
Recall	TPR = TP / (TP + FN)
F-measure	F = (2 * PPV * TPR) / (PPV + TPR)
Area Under Precision-Recall Curve	AUPRC = $\int_0^1$ TP/(TP+FP) d(TP/P)
Area Under ROC-curve	AUROC = $\int_0^1$ TP/P d(FP/N)

stage state, etc. The UI improves the production of visualizations and real time metrics, which make it easier to troubleshoot and debug during development. Thus, to evaluate the performance of the streaming query, we used some metrics provided by the web UI, as follows :

- Input Rate : The aggregate data rate that describes the number of loaded records per second between the last trigger and the current trigger.
- Process Rate : The aggregate rate at which Spark processes data that describes the number of loaded records per second in each trigger.
- Batch Duration : The duration of each batch.

4.2 Experimental Results

This sub-section shows the key findings of the simulation experiments in order to check whether our approach can process transaction streams and detect trust-related attacks in real-time using our deep learning based classification model.

4.2.1 Experimental Results of our Deep Learning Based Classification Model

We tested our model on a local machine with limited resources and we expected good results. In fact, we achieved a good accuracy rate of 99.6%. We also measured the model performance using Area Under the ROC Curve (AUROC) to determine the extent to which the model is capable of distinguishing between classes and we obtained a value of 0,99. The Area Under the Precision-Recall Curve (AUPRC) also provides a rate of 0,99. We note that our model correctly predicted trust-related attacks. Moreover, F-measures were 99,73% and 99,26% in detecting malicious transactions and benevolent transactions, respectively. Hence, we achieved high scores, as illustrated in figure 2, which validate the performance of our classification model.

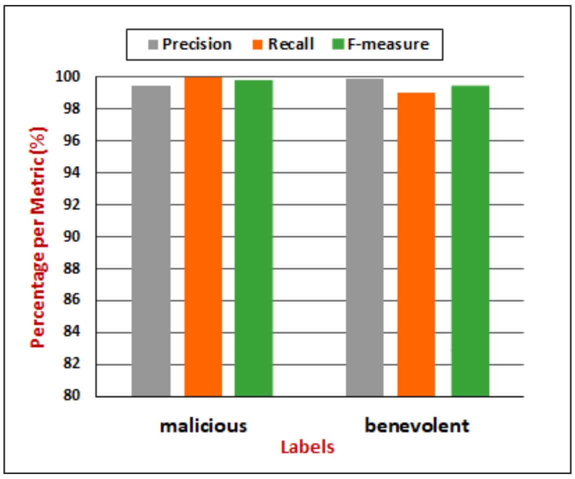

Figure 2: Experimental results of our deep learning based classification model.

4.2.2 Comparison

The majority of the above mentioned works, as reported in Table (1), have been conducted to detect DDoS attacks or intrusions based on Spark, which showed efficient performance in processing data streams and developing different ML techniques.

Nevertheless, all trust management works did not take advantage of Spark to improve the development of trust related-attack detection model in real-time. Thus, we aim to detect malicious transactions in real-time using Apache Spark. To make adequate comparisons, we referred only to trust management works, although they do not support Apache Spark. We compared our model with three previous works, (Abdelghani et al., 2018), (Masmoudi et al., 2019) and (Masmoudi et al., 2021), that were conducted to detect trust-related attacks that represent malicious transactions. These models also used the same dataset (Sigcomm) but for different purposes. In fact, figure 3 plots the f-measure value for the three models. We clearly notice the difference between our work and the works carried out by both (Abdelghani et al., 2018) and (Masmoudi et al., 2019). Our model has increased by approximately 5% compared to other models. In the work conducted by (Masmoudi et al., 2021), the deep learning model needs a larger dataset to improve the model performance than the machine learning model used in (Masmoudi et al., 2021).

4.2.3 Experimental Results of the Stream Processing Query

To evaluate the stream query, we generated 160 transactions and each one is stored in CSV format. These transactions were not used before in the training model in order to validate the efficiency of prediction.

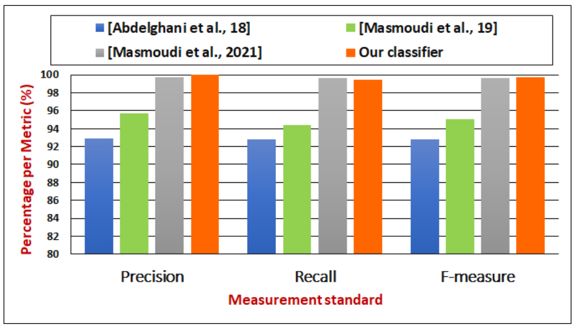

Figure 3: Comparison between different classification models.

As shown in figure 4, we notice that the Process Rate remains stable at 1.4 records/sec average rate at the same Input Rate of 0.5 records/sec average rate. This means that a job with enough processing capacity can process input data. For Batch Duration, increasing the batch size, will lead to high latency. Some streaming systems have an option of high latency in exchange for lower throughput. As demonstrated figure 4, our Batch Duration is oscillating consistently around 700 ms. In fact, the structured streaming achieves both latency and throughput. As it operates on the data, it oscillates as structured streaming processes varying numbers of events over time.

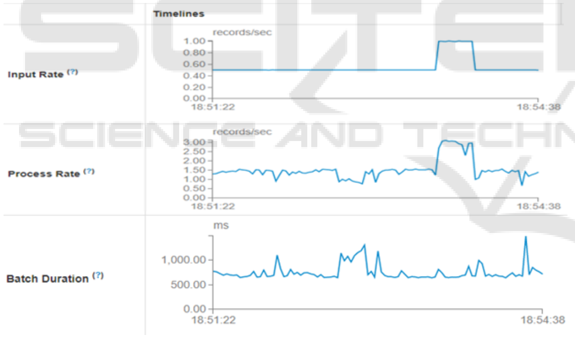

Figure 4: Experimental results of the stream processing query.

In addition, we compared the performance of our Structured Streaming with two models that used real-time metrics in the research community. The first model proposed by (Abid and Jemili, 2020) was able not only to process 60635 records in one second, but also to collect 613027 records per second. These results are illogical since the obtained rows are over the capacity of jobs to process input data in order to achieve high latency. To evaluate our stream processing query, the processing rate is twice greater than the Input Rate. Thus, our stream is better than that in (Abid and Jemili, 2020) with low latency. The second model developed by (Zhou et al., 2018) can collect an average of 200,000 records per second, while the first model has an average processing time of 546 ms. Compared with the two proposed models, our system achieved better results as the average time to process input data is only 333 ms.

5 DISCUSSION

Different patterns have been utilized to efficiently and effectively evaluate trustworthiness. Our findings have surpassed two significant benchmarks. Firstly, we achieved a high f-measure was 99,73% in detecting malicious nodes, surpassing the performance of previous studies (Abdelghani et al., 2018), (Masmoudi et al., 2019) and (Masmoudi et al., 2021). Secondly, we focused on developing a practical system that caters to our specific requirements.

In this regard, we leveraged Apache Spark to enable real-time detection. By striking a balance between processing time and accuracy, we were able to generate comprehensive reports without compromising transaction processing. Given the challenge of obtaining a real-world environment to create a dataset, we relied on existing research that focused on extracting effective features for classifying malicious nodes across various types of trust-related attacks (Masmoudi et al., 2021).

In our future work, we aim to prioritize two key areas. Firstly, we plan to focus on developing solutions for effectively collecting and retrieving sensitive trust management information. This involves implementing mechanisms that ensure the secure and efficient handling of such data to enhance the overall trust management process. Secondly, we intend to optimize transaction storage to improve the system's performance and resource utilization. By devising innovative approaches for storing and accessing transactions, we aim to enhance scalability and reduce storage overhead.

6 CONCLUSION

In general, the trustworthiness of social transactions has been considered as an interesting axis in the research community. For that, to evaluate the trustworthiness of transactions, we proposed in this paper a new system composed of two phases. First, we built a new deep learning model based on Apache Spark to classify transactions into two classes; either malicious or benevolent. Second, we applied a real-time module using spark streaming library to analyze transactions. We also made use of the DL model to make predictions in real-time. Based on our experimental results, we achieved better classification scores compared with those obtained in (Masmoudi et al., 2019), (Masmoudi et al., 2021) and (Abdelghani et al., 2018). We also compared our Structured Streaming process to other similar works, and we got better results.

REFERENCES

Abdelghani, W., Zayani, C. A., Amous, I., and Sèdes, F. (2018). Trust evaluation model for attack detection in social internet of things. In *International Conference on Risks and Security of Internet and Systems*, pages 48–64. Springer.

Abderrahim, O. B., Elhdhili, M. H., and Saidane, L. (2017). Tmcoi-siot: A trust management system based on communities of interest for the social internet of things. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 747–752. IEEE.

Abid, A. and Jemili, F. (2020). Intrusion detection based on graph oriented big data analytics. *Procedia Computer Science*, 176:572–581.

Awan, M. J., Farooq, U., Babar, H. M. A., Yasin, A., Nobanee, H., Hussain, M., Hakeem, O., and Zain, A. M. (2021). Real-time ddos attack detection system using big data approach. *Sustainability*, 13(19):10743.

Azeroual, O. and Nikiforova, A. (2022). Apache spark and mllib-based intrusion detection system or how the big data technologies can secure the data. *Information*, 13(2):58.

Boyd, D. M. and Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. *Journal of computer-mediated Communication*, 13(1):210–230.

Chen, R., Bao, F., and Guo, J. (2015). Trust-based service management for social internet of things systems. *IEEE transactions on dependable and secure computing*, 13(6):684–696.

Chun, H., Kwak, H., Eom, Y.-H., Ahn, Y.-Y., Moon, S., and Jeong, H. (2008). Comparison of online social relations in volume vs interaction: a case study of cyworld. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, pages 57–70.

Ekbatanifard, G. and Yousefi, O. (2019). A novel trust management model in the social internet of things. *Journal of Advances in Computer Engineering and Technology*, 5(2):57–70.

Hirzel, M., Soulé, R., Schneider, S., Gedik, B., and Grimm, R. (2014). A catalog of stream processing optimizations. *ACM Computing Surveys (CSUR)*, 46(4):1–34.

Jafarian, B., Yazdani, N., and Haghighi, M. S. (2020). Discrimination-aware trust management for social internet of things. *Computer Networks*, 178:107254.

Jayasinghe, U., Lee, G. M., Um, T.-W., and Shi, Q. (2018). Machine learning based trust computational model for iot services. *IEEE Transactions on Sustainable Computing*, 4(1):39–52.

Khan, M. A. and Kim, J. (2020). Toward developing efficient conv-ae-based intrusion detection system using heterogeneous dataset. *Electronics*, 9(11):1771.

Lee, S. and Jun, C.-H. (2018). Fast incremental learning of logistic model tree using least angle regression. *Expert Systems with Applications*, 97:137–145.

Marche, C. and Nitti, M. (2020). Trust-related attacks and their detection: A trust management model for the social iot. *IEEE Transactions on Network and Service Management*, 18(3):3297–3308.

Masmoudi, M., Abdelghani, W., Amous, I., and Sèdes, F. (2019). Deep learning for trust-related attacks detection in social internet of things. In *International Conference on e-Business Engineering*, pages 389–404. Springer.

Masmoudi, M., Zayani, C. A., Amous, I., and Sèdes, F. (2021). A new blockchain-based trust management model. *Procedia Computer Science*, 192:1081–1091.

Meena Kowshalya, A. and Valarmathi, M. (2017). Trust management for reliable decision making among social objects in the social internet of things. *IET Networks*, 6(4):75–80.

Patil, N. V., Krishna, C. R., and Kumar, K. (2022). Ssk-ddos: distributed stream processing framework based classification system for ddos attacks. *Cluster Computing*, 25(2):1355–1372.

Rajesh, G., Raajini, X. M., and Vinayagasundaram, B. (2016). Fuzzy trust-based aggregator sensor node election in internet of things. *Int. J. Internet Protoc. Technol.*, 9(2/3):151–160.

Singh, M. P., Hoque, M. A., and Tarkoma, S. (2016). A survey of systems for massive stream analytics. *arXiv preprint arXiv:1605.09021*.

Talbi, S. and Bouabdallah, A. (2020). Interest-based trust management scheme for social internet of things. *Journal of Ambient Intelligence and Humanized Computing*, 11(3):1129–1140.

Yue, Y., Li, S., Legg, P., and Li, F. (2021). Deep learning-based security behaviour analysis in iot environments: a survey. *Security and Communication Networks*, 2021.

Zhang, H., Dai, S., Li, Y., and Zhang, W. (2018). Real-time distributed-random-forest-based network intrusion detection system using apache spark. In *2018 IEEE 37th international performance computing and communications conference (IPCCC)*, pages 1–7. IEEE.

Zheng, G., Gong, B., and Zhang, Y. (2021). Dynamic network security mechanism based on trust management in wireless sensor networks. *Wireless Communications and Mobile Computing*, 2021.

Zhou, B., Li, J., Wu, J., Guo, S., Gu, Y., and Li, Z. (2018). Machine-learning-based online distributed denial-of-service attack detection using spark streaming. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE.