Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 1075–1080
DOI: 10.15439/2023F1151
ISSN 2300-5963 ACSIS, Vol. 35

VICRA: Variance-Invariance-Covariance Regularization for Attack Prediction

Aditya Srinivas Menon
0009-0000-9326-028X
Student, Computer Science
Indian Institute of Information Technology Kottayam
Kerala, India
Email: adityasrinivas20bcs8@iiitkottayam.ac.in

Gouri Nair
0009-0007-5793-9966
Student, Electronics and Communication
Indian Institute of Information Technology Kottayam
Kerala, India
Email: gourinair20bec7@iiitkottayam.ac.in

***Abstract*—In cybersecurity, accurate and timely prediction of attacks plays a crucial role in mitigating the risks and impacts of cyber threats. However, traditional attack prediction methods that rely on training Machine Learning (ML) algorithms directly on raw data often suffer from high false alarm rates and low detection rates, leading to inaccurate and unreliable results. To overcome these limitations, this paper presents a novel approach that integrates attack prediction with self-supervision using variance-invariance-covariance regularization (VICReg). The proposed method harnesses VICReg to enhance raw data and generate representations while leveraging self-supervision to learn meaningful features without supervision. Training classic ML algorithms on these refined representations improves prediction accuracy and enhances the robustness of the learning process. We provide a comprehensive description of the proposed method and present an evaluation of its performance on several benchmark datasets. The experimental results demonstrate the superiority of the proposed method over classic ML algorithms.**

***Index Terms*—Self-supervised learning, Deep Learning, Structured Data, Attack Prediction, Wireshark**

I. Introduction and Related Work

CYBERSECURITY is a major concern for businesses, governments, and individuals, as the damage caused by worldwide cybercrime is expected to reach $10.5 trillion annually by 2025. The global cybersecurity workforce is projected to be short 1.8 million people by 2022, with 66% of respondents reporting that they don't have enough capacity to address current threats. Predictive analysis has the potential to give organizations an advantage by allowing them to allocate their defence resources more effectively and automate the process of attack forecasting and prediction.

Some of the most actively studied problems include Network Risk Scoring (NRS) [1], Threat Detection and Classification (TDC) [2], Attack Prediction [3], phishing detection [4], web shell classification and automating security pipelines.

A. Attack Prediction

The number and sophistication of cyberattacks are constantly increasing, making it increasingly difficult for organizations to protect themselves against all likely threats. By predicting and preparing for potential attacks, risks and losses can be minimized. Attack prediction refers to the process of identifying and forecasting potential security threats or vulnerabilities in a system or network. This is a critical aspect of cybersecurity, as it helps organizations to proactively protect themselves against future attacks and to minimize the impact of any breaches that do occur.

TABLE I
Example of Wireshark captured data in table format

No.	Time	Source	Protocol	Length	Info
1	0	192.168.1.2	HTTP	98	GET Tindex.html
2	0.05	192.168.1.1	HTTP	145	HTTP/ 1.1 200 0K
3	0.06	192.168.1.2	HTTP	98	GET /css/style.css
4	0.09	192.168.1.1	HTTP	756	HTTP/ 1.1 200 0K
5	0.1	192.168.1.2	HTTP	98	GET /js/script.js
6	0.14	192.168.1.1	HTTP	903	HTTP/I.1 200 0K

B. Prediction Logic

One of the key tools in addressing cybersecurity threats is the use of network packet analyzers. These tools are designed to capture, analyze, and interpret network traffic, to identify potential security breaches and malicious activities. Among these tools, Wireshark [5] is a free and open-source (GNU General Public License) platform independent tool that serves as a packet analyzer. It is used for network issue resolution, examination, the development of communication protocols and educational purposes. Wireshark intercepts packets and presents them in a table format, with each row representing a single packet and each column displaying various details about the packet.

The captured packets can be filtered and sorted using various criteria, such as the protocol used, the source and destination addresses, or the specific type of data being transmitted. Table I-B shows an example of Wireshark data displaying six packets in table format. The columns include the time, source IP address, protocol, packet length, and a brief description of the packet

C. Self Supervised Learning

Self-supervised learning (SSL) [6] is a machine learning approach that seeks to acquire data representations without explicit supervision, thereby eliminating the need for labeled data. Through this method, the model autonomously learns valuable features and representations, which can be utilized for downstream tasks. SSL holds significant potential for enhancing the efficiency and effectiveness of learning algorithms in scenarios where labeled data is limited or costly to obtain.

One of the earliest works in SSL was the autoencoder [7], a neural network architecture that learns to reconstruct its input by training on an unlabeled dataset. Another popular SSL technique is contrastive learning [8] which is a method of training a model to distinguish between different representations of the same data.

SSL has been applied to a wide range of tasks such as computer vision [9], natural language processing [10] and speech recognition. SSL is still an active area of research and many questions remain open. For example, there is currently no consensus on the best way to evaluate the quality of the representations learned by SSL methods [11]. Additionally, the effectiveness of SSL for certain tasks or domains is still being explored. The issue of collapsing problem [12] in learning architecture is often mitigated by the presence of hidden biases, which may not have a transparent explanation or interpretation. This ensures that the learning process remains stable and effective. However, the underlying reasons or justifications for these biases may not always be readily apparent or easily interpretable.

D. VICReg

VICReg [13] a study by Meta Research introduced an approach that explicitly addresses the collapse problem by incorporating a straightforward regularization term on the variance of the embeddings along each dimension independently. VICReg, combines this variance term with a decorrelation technique that focuses on reducing redundancy and covariance regularization. By integrating these strategies, VICReg achieves state-of-the-art results on a range of downstream tasks, effectively overcoming the collapse problem and enhancing the quality and diversity of the learned embeddings.

While Self-Supervised Learning (SSL) has garnered substantial interest and recognition in the domains of computer vision and natural language processing (NLP), where large-scale datasets of unlabeled images are readily available (e.g. ImageNet), there has been very less research behind the adoption of SSL to tabular data. We apply self-supervision to predict attacks from tabular data using VICReg in this paper. Following are some of the significant observations:

1) Self supervision using VICReg on tabular data before applying Machine Learning (ML) algorithms helps in improving prediction accuracy.
2) When it comes to attack prediction, swap noise, a complementary approach to existing augmentation techniques in the tabular data setting, proved to be effective.
3) VICRA improves attack prediction accuracy compared to traditional Machine Learning (ML) methods.

Our key contributions can be summarized as follows:

1) We address the problem of Attack Prediction on wireshark features as a Machine Learning (ML) problem. We present the problem as an anomaly detection task for tabular data.
2) We propose a novel technique called VICRA (Variance-Invariance-Covariance Regularization for Attack Prediction) which uses self-supervision to enhance the tabular embeddings using swap noise and show significant increase in performance.
3) By leveraging the inherent structure of data and regularizing the learning process, the method is able to improve prediction accuracy and robustness.
4) We present a pipeline to train attack prediction models on wireshark data using VICRA.
5) We investigate the performance of VICRA attack prediction on popular datasets by comparing it with the current ML approaches.
6) Our VICRA technique improves the accuracy by over 2.48% for NSL KDD, 0.90% for UNSW NB15 and 7.17% for AWID2 than traditional ML approaches.

The rest of the paper is organized as follows. Our proposed approach is described in Section 2. Performance evaluation and findings of the work are shown in Section 3, Section 4 concludes the finding of the work.

II. Proposed Approach

In this section, we formally introduce our proposed VICRA system and highlight the specific areas of the problem that we aim to solve. The architectural overview of the proposed system is shown in Figure II.

The system takes Wireshark features as input and predicts if it's an attack or not. The overall procedure includes the following steps: (1) data preparation, (2) self-supervised learning, (3) embedding cloud generation, and (4) attack prediction. The primary focus of this research is to use self-supervision to enhance the feature embeddings while improving metrics for attack prediction. The processes are then thoroughly explained.

A. Data Preparation

As seen in Figure 1 the dataset is in the form of raw Wireshark features. To obtain useful information from the raw features we clean the data using standard data preparations methods which are mentioned below.

1) Missing Values: The data collected might have a lot of missing features. There are many proposed approaches on handling missing data in Wireshark data. For our approach, we perform List-wise Deletion [14] on categorical and binary features followed by Simple Mutation on continuous features. In List-wise Deletion, every case that has one or more missing values is removed whereas in Simple Mutation the missing value is replaced by the mean of the values in that feature.

2) Feature Selection: The Wireshark data that was recorded includes incorrect fields and extra information. In this step, feature selection methods reduce the number of features. In the process of attribute selection, information gain and gain

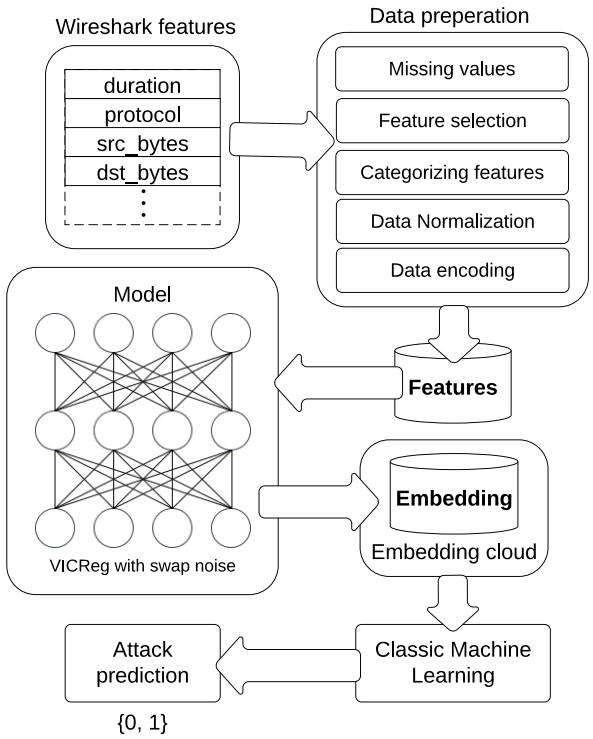

Fig. 1. System architecture of the VICRA System

ratio are commonly employed techniques for assessing the relevance of variables with respect to the target variable [15] [16]. Some columns like IPv4/IPv6 addresses are also removed since the model tends to overfit on these features.

3) Categorizing Features: The features are now categorized into continuous, categorical, discrete and binary for further pre-processing. Categorizing features is a crucial step in pre-processing data for attack prediction. Features are categorized into continuous (e.g., time duration), categorical (e.g., protocol type), discrete (e.g., number of packets), and binary (e.g., event occurred or not) types. By categorizing features, appropriate pre-processing techniques can be chosen for each type.

4) Data Normalization: Continuous features are either normalized or standardized. Log-transformation is performed on skewed data. Log transformation helps mitigate the effect of skewness by reducing the variability in the data and bringing it closer to a normal distribution.

5) Data encoding: Categorical features are one-hot encoded, and discrete features are treated as categorical or binned into ranges. Binary features require no pre-processing.

The output of this phase (given T) is clean and structured data which is fed into the VICReg model for self supervision.

B. Self Supervision

Figure II-B provides an illustration of the VICReg architecture, which encompasses variance, invariance, and covariance regularization. The process begins with a batch of features T obtained from the Data preparation step. From this, two sets of noisy features X and X' are generated and encoded into representations Y and Y'. These representations are then passed through an expander, resulting in the production of embeddings Z and Z'.

To ensure the effectiveness of the embeddings, several regularization techniques are applied. Firstly, the distance between embeddings from the same feature is minimized. Additionally, the variance of each embedding variable within a batch is maintained above a specified threshold. Furthermore, the covariance between pairs of embedding variables over a batch is attracted to zero, promoting decorrelation between the variables. It is worth noting that the two branches in the architecture do not necessarily share the same architecture or weights, although in most experiments, they consist of shared weight Feed Forward Layers (FFL).

To generate the noisy features X and X', swap noise is introduced to the original features, a process that is elaborated upon in Section III-C. After training, the model is then utilized for inference on the features obtained in the previous step. The resulting embeddings are generated and subsequently stored in the embedding cloud for further analysis or downstream tasks.

C. Embedding Cloud

The embeddings generated from the self-supervised inference are combined to form an embedding cloud as mentioned in [17]. The embedding cloud is a permanent storage of preprocessed embeddings which are used while training. Once the embedding cloud is generated and saved, we can proceed to train the model on the embeddings for attack prediction.

D. Attack Prediction

The embeddings stored in the embedding cloud, along with the ground truth labels, are utilized to train the Machine Learning (ML) model instead of training the model on the raw features. The self supervision performed regularizes the learning process and leverages the inherent structure of the data. The proposed method is found to yield improved prediction accuracy and greater robustness compared to traditional feature-based approaches. The use of self-supervised learning for generating embeddings has been demonstrated to be a promising approach for training ML models in a variety of applications. Our results suggest that this approach has the potential to be a useful tool in the field of cybersecurity for predicting and mitigating cyber attacks. While the proposed approach shows promising results for attack prediction, further research is required to fully explore its potential and assess its applicability to various types of attack prediction tasks.

III. Evaluation

A. Dataset

For our evaluation, we used three benchmark datasets commonly used in the field of cybersecurity: AWID 2 [18], NSL KDD [19], and UNSW NB15 [20] [21] [22]. These datasets provide a diverse range of attack scenarios and network traffic patterns, allowing us to assess the performance of our proposed approach across different contexts. The AWID 2

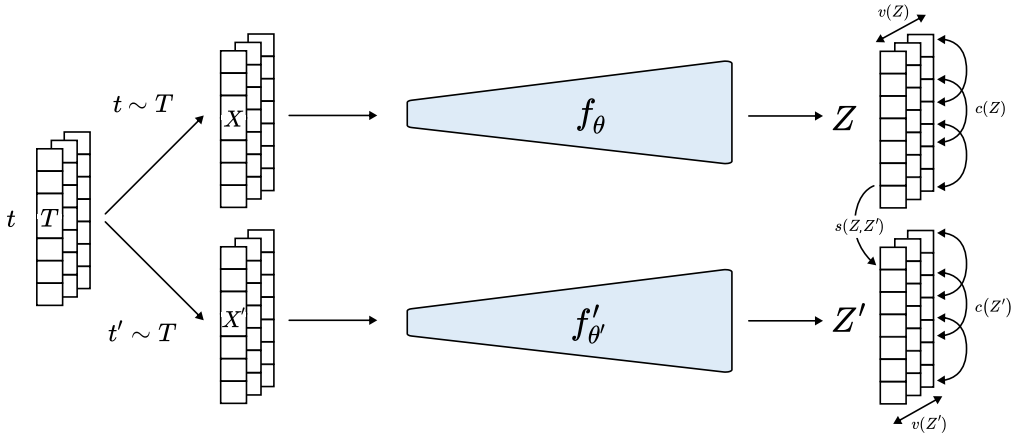

Fig. 2. VICReg architeture chosen for the wireshark data.

dataset contains wireless intrusion detection system (WIDS) data, the NSL KDD dataset is derived from the KDD Cup 1999 dataset, and the UNSW NB15 dataset includes network traffic data with various attack types. Table III-A provides an overview of the dataset distribution for three different datasets: AWID2, NSL KDD, and UNSW NB15. It is also clear from the distribution that in AWID2 dataset close to 97.15% of the data is from the "Normal" label, whereas in NSK KDD and UNSW NB15 datasets, the "Normal" label is only 53.46% and 31.94% respectively.

In order to evaluate the proposed approach, a binary classification scenario was created for each dataset. In this scenario, the "Normal" label was assigned the binary value of 0, while all other attack labels were grouped together and assigned the binary value of 1. This binary classification setup allows for the examination of the model's performance in distinguishing between normal instances and instances associated with various types of attacks. By treating normal instances as the negative class (0) and attacks as the positive class (1), the model can be trained and tested to assess its ability to correctly classify instances as either normal or attack-related. This approach simplifies the problem by focusing on differentiating between normal behavior and malicious activities, enabling the evaluation of the model's effectiveness in detecting and classifying attacks within the given datasets.

1) Data preparation: Prior to applying self supervision and learning algorithms, the dataset is cleaned using the techniques mentioned in Section 2.1. This includes handling missing values, selecting and categorizing features, data normalization for numerical features and data encoding for categorical features. The post processed data (T) is fed in batches to the self supervised VICReg model after adding noise.

2) Swap noise: Our proposed framework offers a complementary approach to existing augmentation techniques employed in the tabular data setting. As such, we conducted experiments involving the introduction of noise to randomly selected entries within each subset. This was achieved by overwriting the value of a chosen entry with another value randomly sampled from the same column. This augmentation technique is referred to as 'swap-noise'. In a previous study conducted by Michael Jahrer (MJ) [23], a noise creation method known as 'swap noise' was introduced. This method involves randomly swapping a small portion of columns between two samples in order to generate noisy samples for training purposes. In the following section, we present our implementation of the swap noise technique, based on MJ's original approach.

B. Baseline

To evaluate the performance of VICRA, we compared it against several methods commonly used in attack prediction tasks. These methods include traditional machine learning algorithms such as logistic regression, decision trees, and support vector machines, as well as deep learning models such as feed-forward neural networks. Additionally, we implemented our own baseline model that directly trained on the raw features without the self-supervised learning step.

C. Experiment Setup

To conduct the experiment, we first preprocess the AWID2, NSL KDD and UNSW NB15 datasets using the approach mentioned in Section 2.1. The features are then run through the VICReg model for self supervision. The VICReg model is a multi-layer perceptron architecture with stacked layers of linear transformations, batch normalization, and ReLU activation functions. The model consists of an expander module that is responsible for expanding the input features. It takes in features and applies a linear transformation followed by batch normalization and ReLU activation. This process is repeated n times in the expander module. The model is trained for 50 epochs and the representations are generated for each wireshark feature in the dataset. The generated representation

TABLE II
OVERVIEW OF THE DATASETS

AWID2			NSL KDD			UNSW NB15	
Class	Label Count		Class	Label Count		Class	Label Count
Normal	157,749,037		Normal	67,343		Normal	56,000
Impersonation	1,884,378		DoS	45,927		Attack	119,341
Injection	1,530,373		Probe	11,656			
Flooding	1,211,459		R2L	995			
			U2R	52			
Total	**162,375,247**		**Total**	**125,973**		**Total**	**175,341**

TABLE III
RESULTS FOR VICRA ON AWID2, NSL KDD, AND UNSW NB15 ACROSS FOUR DIFFERENT APPROACHES, DECISION TREE, LOGISTIC REGRESSION, MLP AND SVC

		Decision Tree		**Logistic Regression**		**MLP**		**SVC**	
		w VICReg	w/o VICReg	**w VICReg**	w/o VICReg	w VICReg	w/o VICReg	**w VICReg**	w/o VICReg
NSL KDD	Precision ↑	**95.07%**	90.72%	**92.06%**	91.38%	**97.47%**	95.58%	**96.59%**	96.53%
	Recall ↑	**70.79%**	69.36%	**68.23%**	66.33%	69.16%	62.45%	67.70%	65.19%
	F1 Score ↑	**81.16%**	78.62%	**78.37%**	76.86%	**80.91%**	75.54%	**79.60%**	77.82%
	FPR ↓	**0.0485**	0.0937	**0.0777**	0.0827	**0.0237**	0.0382	0.0316	0.0310
	FNR ↓	**0.2921**	0.3064	**0.3177**	0.3367	**0.3084**	0.3755	**0.3230**	0.3481
	Accuracy ↑	**81.29%**	78.52%	**78.57%**	77.27%	**81.42%**	76.98%	**80.25%**	78.85%
UNSW NB15	Precision ↑	67.98%	74.53%	88.87%	90.85%	90.82%	78.36%	89.76%	75.66%
	Recall ↑	85.43%	96.79%	**83.05%**	61.08%	81.34%	92.45%	80.53%	96.67%
	F1 Score ↑	75.71%	84.21%	**85.86%**	73.05%	**85.82%**	84.82%	**84.89%**	84.88%
	FPR ↓	0.1888	0.1552	0.0488	0.0289	0.0386	0.1198	**0.0431**	0.1459
	FNR ↓	0.1457	0.0321	**0.1695**	0.3892	0.1866	0.0755	0.1947	0.0333
	Accuracy ↑	82.50%	88.41%	**91.27%**	85.61%	**91.42%**	89.43%	**90.85%**	89.00%
AWID2	Precision ↑	**89.76%**	87.57%	**73.21%**	67.71%	**73.21%**	57.44%	**64.38%**	58.24%
	Recall ↑	**92.58%**	24.62%	**86.51%**	72.98%	86.51%	92.70%	72.98%	79.69%
	F1 Score ↑	**91.15%**	38.44%	**79.30%**	70.24%	**79.30%**	70.93%	**68.41%**	67.30%
	FPR ↓	0.1077	0.0043	**0.3226**	0.3547	**0.3226**	0.6999	**0.4114**	0.5823
	FNR ↓	**0.0742**	0.7538	**0.1349**	0.2702	0.1349	0.0730	0.2702	0.2031
	Accuracy ↑	90.92%	91.31%	**77.21%**	68.80%	**77.21%**	61.65%	**65.98%**	60.91%

is stored in the embedding cloud as a json object before using it for attack prediction.

As seen in [18] we choose four Machine Learning approaches, Decision Tree, Logistic Regression, Multi Layer Perceptron (MLP) and Support Vector Machines (SVM) for attack prediction. As a way to demonstrate the importance of self supervision and test whether it works, we train attack prediction models both on raw features T and representations Z. For each dataset four such models are trained on raw features and the self supervised representations and the metrics are logged for comparison.

D. Evaluation Metrics

The most commonly deployed performance metrics for validating the performance of ML and DL methods for attack prediction are Accuracy, F1 Score, Precision and Recall.

- Precision is defined as the ratio of total number of correctly predicted packets by total number of predicted packets.
- Recall is defined as the ratio of total number of correctly predicted packets by the sum of correctly predicted packets and the number of missed packets.
- F1-score: Given precision and recall, F-score is defined as the Harmonic mean of precision and recall
- Accuracy is defined as the ratio of the total number of correctly predicted packets to the total number of packets in the dataset.

E. Results

The results are shown in Table III. It can be observed that the models trained on self supervised VICReg embeddings perform better in the given metrics compared to the models

trained on raw features. The experiment was done using four different approaches for attack prediction to show that self supervision helps improve prediction metrics regardless of the choice of the model. On an average across the four methods, self supervision improves the accuracy by over 2.48% for NSL KDD, 0.90% for UNSW NB15 and 7.17% for AWID2 than training the models on raw features. It is also to be noted that for datasets like AWID2 with over 97.15% data labeled as normal the improvement in accuracy is significantly higher compared to datasets like NSL KDD and UNSW NB15 where the percentage of data labeled as normal is 53.46% and 15.97% respectively.

IV. Conclusion

We tackle the challenge of Attack Prediction on wireshark features as a Machine Learning (ML) problem by framing it as an anomaly detection task for tabular data. To address this, we introduce a novel technique called VICRA (Variance-Invariance-Covariance Regularization for Attack Prediction). VICRA leverages self-supervision to enhance tabular embeddings using swap noise, resulting in a significant performance boost. By incorporating the underlying data structure and applying regularization during the learning process, VICRA improves prediction accuracy and robustness. We present a comprehensive pipeline for training attack prediction models on wireshark data using VICRA. To evaluate the effectiveness of VICRA, we conduct extensive experiments on popular datasets and compare its performance with existing ML approaches. Our results demonstrate that VICRA achieves substantial accuracy improvements, surpassing traditional ML approaches by over 2.48% for NSL KDD, 0.90% for UNSW NB15, and 7.17% for AWID2 datasets. Overall, VICRA offers a promising solution for enhancing attack prediction capabilities in the context of Wireshark data analysis.

References

[1] N. Paltrinieri, L. Comfort, and G. Reniers, "Learning about risk: Machine learning for risk assessment," *Safety Science*, vol. 118, pp. 475–486, 2019. doi: https://doi.org/10.1016/j.ssci.2019.06.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753518311184

[2] F. Ullah, H. Naeem, S. Jabbar, S. Khalid, M. A. Latif, F. Al-turjman, and L. Mostarda, "Cyber security threats detection in internet of things using deep learning approach," *IEEE Access*, vol. 7, pp. 124 379–124 389, 2019. doi: 10.1109/ACCESS.2019.2937347

[3] X. Fang, M. Xu, S. Xu, and P. Zhao, "A deep learning framework for predicting cyber attacks rates," *EURASIP Journal on Information Security*, vol. 2019, no. 1, p. 5, May 2019. doi: 10.1186/s13635-019-0090-6. [Online]. Available: https://doi.org/10.1186/s13635-019-0090-6

[4] O. A. Akanbi, I. S. Amiri, and E. Fazeldehkordi, "Chapter 1 - introduction," in *A Machine-Learning Approach to Phishing Detection and Defense*, O. A. Akanbi, I. S. Amiri, and E. Fazeldehkordi, Eds. Boston: Syngress, 2015, pp. 1–8. ISBN 978-0-12-802927-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128029275000010

[5] [Online]. Available: https://www.wireshark.org/

[6] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum, "A cookbook of self-supervised learning," 2023.

[7] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. doi: 10.1126/science.1127647. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1127647

[8] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006. doi: 10.1109/CVPR.2006.100 pp. 1735–1742.

[9] D. Wang, Y. Zhang, K. Zhang, and L. Wang, "Focalmix: Semi-supervised learning for 3d medical image detection," 06 2020. doi: 10.1109/CVPR42600.2020.00401 pp. 3950–3959.

[10] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[11] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020. doi: https://doi.org/10.48550/arXiv.2003.04297

[12] C. Zhang, K. Zhang, C. Zhang, T. X. Pham, C. D. Yoo, and I. S. Kweon, "How does simsiam avoid collapse without negative samples? a unified understanding with self-supervised contrastive learning," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=bwq6O4Cwdl

[13] A. Bardes, J. Ponce, and Y. LeCun, "VICReg: Variance-invariance-covariance regularization for self-supervised learning," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=xm6YD62D1Ub

[14] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," *Journal of Big Data*, vol. 8, no. 1, p. 140, Oct 2021. doi: 10.1186/s40537-021-00516-9. [Online]. Available: https://doi.org/10.1186/s40537-021-00516-9

[15] W. I. D. Mining, "Data mining: Concepts and techniques," *Morgan Kaufinann*, vol. 10, pp. 559–569, 2006.

[16] M. Asaduzzaman, M. S. Majib, and M. M. Rahman, "Wi-fi frame classification and feature selection analysis in detecting evil twin attack," *2020 IEEE Region 10 Symposium (TENSYMP)*, pp. 1704–1707, 2020. doi: 10.1109/TENSYMP50017.2020.9231042s

[17] A. S. Menon and K. Anand, "X-abi: Toward parameter-efficient multilingual adapter-based inference for cross-lingual transfer," in *Data Management, Analytics and Innovation*, N. Sharma, A. Goje, A. Chakrabarti, and A. M. Bruckstein, Eds. Singapore: Springer Nature Singapore, 2023, pp. 303–317.

[18] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 184–208, 2016.

[19] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009. doi: 10.1109/CISDA.2009.5356528 pp. 1–6.

[20] N. Moustafa, "Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic," Ph.D. dissertation, 2017. [Online]. Available: http://hdl.handle.net/1959.4/58748

[21] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015. doi: 10.1109/MilCIS.2015.7348942 pp. 1–6.

[22] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "NetFlow datasets for machine learning-based network intrusion detection systems," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Springer International Publishing, 2021, pp. 117–135. [Online]. Available: https://doi.org/10.1007%2F978-3-030-72802-1_9

[23] M. Jahrer, "Porto seguro's safe driver prediction solution." [Online]. Available: https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/discussion/44629#250927