Frequent Itemset Mining for Model Data Reduction

Blake Johns and Ryan Benton
The University of South Alabama, Mobile, USA

Johns.blake921@gmail.com
rbenton@southalabama.edu

Abstract: Network intrusion detection systems (NIDS) are in constant battle to come up with new techniques, whether it is a new machine learning model or viewing the data in a new form, to stay ahead of the curve. Recent work has shown that using Deep Learning, coupled with an image-based representation of packet-level data, achieve a high detection rate for a wide number of attacks. Key to the image construction was removing information that can cause misalignment or potentially bias in the Deep Learning model. However, while image-based Deep Learning is known to be good at extracting relevant information from images, it is proposed that using additional preprocessing can result in both improving training and detection time while potentially providing insight into bytes that are frequently occurring in attacks. To this end, this paper proposes the use of Frequent Itemset Mining (FIM), which is a powerful tool that allows the identification of frequent (commonly occurring) itemsets (bytes) from transactional data (bytes in a packet). By identifying the frequent bytes from within the packets of a flow, the feature space can be greatly reduced, which would permit a quicker classification time. This approach is tested on image representations of CICIDS-2017 data where the first *n* packets of a flow are transformed into an image and then fed into a Convolutional Neural Network (CNN) model architecture. Results show that the use of frequent itemsets to identify frequent byte positions of a packet yield comparable results to images that do not use FIM to reduce the number of packet bytes. The feature reduction results in the need for 99% less data, which also reduces the training complexity. We also show better or comparable results to models using the non-reduced packets when reducing the number of packets in a flow (ranging from eight packets to five).

Keywords: Itemset mining, Network intrusion detection, Pattern mining, Feature selection, Data reduction

1. Introduction

As more of our lives connect to a network, securing the transmitted data continues to grow in importance. To protect this data, cyber security efforts need to be put in place. These efforts come in the form of intrusion detection, which alerts a user or organization of nefarious activity.

Network intrusion detection systems (NIDS) sniff packets on the network and identify threats based on some condition (typically heuristic or signature based). The goal is to achieve high classification accuracy while also keeping the time it takes to detection near real time.

The desire to take advantage of developing convolutional neural networks (CNN) has driven an emergence in the application of image classification and computer vision techniques to identify network intrusion. This involves converting network data to images. Numerous conversions have been proposed using grayscale, red-green-blue (RGB), and red-green-blue-alpha (RGBA) image formats (Ghadermazi et al, 2024; Ho et al, 2022; Kim et al, 2020; Toldinas et al, 2021). These works use various models and data and have shown strong results.

This conversion from network data to images changes and increases the feature space complexity. With the large number of features, proper feature selection methods are crucial. Frequent Itemset Mining (FIM) was originally designed for the discovery of common itemsets in market-basket scenarios (Agrawal and Srikant, 1994). Using a *support* measure, FIM can extract common sets of items (features) that are correlated with each other.

In this paper, we use FIM for feature selection of frequent bytes, represented as pixels, in images created from network data. Specifically, we focus on the pixels created by the image conversion proposed in the SPIN-IDS framework (Ghadermazi et al, 2024). This conversion process converts the packets from a unique flow to a single RGB image, where the row of the image corresponds to a specific packet in the flow and the colour corresponds to the direction of the packet (forward or backward). The driving research questions are as follows:

RQ1 - Can frequent itemset mining be used to reduce the number of packets required to make an accurate detection using the SPIN-IDS framework?

RQ2 - Can frequent itemset mining be used to identify a subset of the original data that achieves similar results to the SPIN-IDS experimental results by the ninth packet?

We propose a novel feature extraction method utilizing itemset mining. This feature extraction method takes an image representing the packets within a flow and identifies frequently occurring byte patterns by converting the image to a transactional data set.

The remainder of the paper is organized as follows. Section 2 discusses related work and Section 3 outlines the experiments. Section 4 presents the results of the experiments. Section 5 identifies areas for future research and Section 6 presents the conclusions.

2. Related Works

2.1 Image Transformations in Network Intrusion Detection

Network data comes in two forms: raw packets (bytes) and flows. The raw packets include the required information that is dictated by layer 3 of the OSI model (Kumar et al, 2014). A flow consists of the meta-data of a conversation between two endpoints. A single flow is identified by a 5-tuple consisting of source IP, destination IP, source port, destination port, and protocol (Claise, 2008).

Transforming network data to images has been done for both flows and packets. SPIN-IDS takes advantage of the unique tuple to identify the flow but works directly with the packets within that flow to create an image. SPIN-IDS uses the first nine packets in a flow (Ghadermazi et al, 2024). To convert the packet into the RGB (red, green, blue) image format, the Ethernet packet header is discarded along with specific bytes and all flags from the IP and TCP headers. The remaining bytes of the packet, totalling 1486, are converted to hexadecimal byte representation and undergo a decimal conversion (bound by 255) before being put into a vector. The direction of the packet determines the channel to which the data are assigned. Forward packets go to the red channel, backward packets go to the green channel, and the blue channel remains unused. Finally, these packet vectors are combined to create a single image for the flow, at most 9x1486 (height x width) in size.

In a related approach, flow data is converted to both RGB and grayscale images (Kim et al, 2020). This transformation converts non-numeric and symbolic values to numerical values. One-hot encoding is applied to features, such as the protocol, for conversion and all the features are stored within a vector. The feature vectors are joined together to create a single vector with values bound by [0, 255] and reshaped to fit the image space of size (13 x 9) pixels. The new data are fed into 3 channel and single channel image representations for RGB and grayscale images. Additionally, network flows have been converted to RGBA images through numerical conversions (Toldinas et al, 2021). Specifically, the values are converted to unsigned 32-bit integers; if the value is a float, then the remainder is truncated. These values are then mapped to a byte array, where every array is 168 bytes, and every 4 bytes represents a pixel in the ARGB (alpha, red, green, blue) image. This conversion outputs images of size (42 x 1) and a size of (7 x 6).

Rather than using an entire flow, a time window is specified to generate new features with respect to the elapsed time of the window which are converted into an image (Ho et al, 2022). This elapsed window is applied to all flows occurring within that window. It then uses a decision tree for feature selection, the Mahalanobis Distance-Based Oversampling (Abdi and Hashemi, 2016) for class balancing and identifying features with an information gain of greater than 0.001 in at least one attack type. Through this process, 24 features were identified for use in the image conversion process.

Another work uses the DeepInsight Python tool (Alok Al Lab, 2025), a Kernel PCA (KPCA) (Scholkopf et al, 1998), and a Gabor filter to convert network flow data into an image using a suitable user-defined image size (Siddiqi, Ahmed, and Pak, 2022). Missing values and duplicate samples are removed from the data. Non-numerical features were then masked using one-hot encoding before being passed into DeepInsight. Next, KPCA converts the data into a 2-dimensional space. Using an appropriate image size, a Gabor filter is then applied to obtain the data in frequency space while also keeping respect to time space. The image size varies between data sets and is (9 x 9) for the CSE-CIC- IDS 2017 / 2018 and ISCX-IDS 2012 while the NSL-KDD data set uses a 12 x 12 size image. The data is normalized, resulting in a 2-dimensional grayscale image.

2.2 Itemset Mining

An item or itemset comes from the notion of basket data, where the items within the basket represent a transaction and those collections collectively form transactions over time. Frequent itemset mining identifies items that frequently occur together within databases and forms relationships between these items, known as association rules (Agrawal et al, 1993). A persistent itemset mining issue is how memory intensive it is when presented with large amounts of data, particularly many features or a lower support threshold. This was shown in comparisons of the FP-Growth (Han et al, 2004) and Eclat algorithms (Zaki, 2000) followed by the

Eclat and Medic algorithms (Goethals, 2004). A lower support generates more item sets and states that if a database cannot be completely loaded into main memory, then memory problems will arise when generating item sets (Goethals, 2004).

Numerous works have been performed to reduce the memory requirement of itemset mining tasks. The introduction of the CFP-Tree and CFP-array applied lightweight compression techniques to reduce the memory footprint (Schlegel et al, 2011). This compression resulted in a decrease in memory usage of 7x - 25x. Influenced from machine learning, a test-train split method is used to split the data into different partitions (Al-Zeiadi, Mohammed, and Al-Maqaleh, 2025). Refraining from loading all the data into memory allows this method to efficiently handle large datasets.

Itemset mining is traditionally used with transactional data; however, some work attempts to apply this to domains such as image classification. To obtain the image in the proper representation to perform FIM, the image is transformed into a set of visual words (Nowozin et al, 2007). A backtracking FIM approach is applied to these words to extract the frequent itemsets. Then a support vector machine (SVM) is trained on the item sets.

Another improvement uses a custom itemset mining algorithm to create a grouplet, which is structured information of an image (Yao, Bangpeng, and Fei-Fei, 2010). Once the initial grouplets are created, itemset mining is applied to them to prune out infrequent grouplets. The mining of different configurations within images is also proposed (Quack et al, 2007). These configurations are used to separate background information from the primary image and allow focus on the object of interest. Fernando et al (2012) defines an itemset as a pair that contains a visual word and its frequency (w, s). The set of transactions are created from the set of local histograms (local bag-of-words) computed from the image. Frequent Local Histograms (FLH) are then formed, and any frequent mining algorithm can be used to extract patterns which are used directly for image classification.

3. Experiments

The experiments require the generation of images that represent the packets within a network flow. These RGB images are at most 9 pixels in height. The height of the image depends on the number of packets selected as a cap for that flow. These experiments begin with the first nine packets in a flow and continue to reduce this down to the first five.

The design of the experiments is two parts. Firstly, packet bytes are extracted from the RGB image and converted into a transactional representation. This transactional representation treats bytes in a single row as a set of items, allowing the application of itemset mining. After the transformation, the transactions are placed into two datasets. The first data set is the wholistic transactional dataset, which contains every row of pixels from each image. The second data set is created with respect to the position of the row within the image. As each row in the image corresponds to a packet in a flow, this data set contains positionally relative data within the image. Once the transactional data sets have been created, frequent itemsets (sets of items) will be extracted. The criteria for an itemset to be considered frequent is a measure of support, defined as *count(x)/count(dataset) >= MinSupport* where x is an item in the database and *MinSupport* is a decimal value threshold that represents a percentage. This metric is relative to the size of the data and is expected to yield different results between the full transactional data set and the packet position data sets.

Once the frequent itemsets have been extracted, the byte positions within the itemsets will be used as the new feature space. The original data set begins with a maximum size of (9, 1486, 3) representing (packet, bytes, channel). This is a maximum size because not all images will have a height of nine, but it remains the cap. Assuming that there were 10 itemsets extracted that contained eight frequent byte positions, the new size of the feature space is (9, 8, 3).

3.1 Experimental Setup

The experiments were run on a ROG Zephyrus G14 laptop with the Windows operating system. This machine has 16 GB of RAM and an AMD Ryzen 9 6900HS processor. The experiments used the CICIDS 2017 data set (Sharafaldin, Lashkari, and Ghorbani, 2018). Network traffic is converted to images using a Python implementation of the image generation and byte selection method defined by the SPIN-IDS framework (Ghadermazi et al, 2024). Further data transformations and itemset mining are also implemented in Python. The CNN model was built using PyTorch (Paszke et al, 2019).

3.2 Dataset

The CICIDS 2017 data set consists of 13 attacks that cover denial of service (DoS), brute force, web attacks, infiltration, and reconnaissance activities. For these experiments, the network data have been reordered using reordercap and duplicate packets have been removed using editcap; both applications are a part of the Wireshark distribution. The labelling effort is strictly with the IP address of the attacker's machine (Pre-NAT: 205.174.165.73, Post-Nat: 172.16.0.1) after going through the firewall and follows a binary scheme of 0 (benign), 1 (threat). This labelling effort is likely to affect the results of the modelling and itemsets, as many works in the cyber security community that have identified disparities in both the attacks that were effective and the time of which the attack is being executed (Liu et al, 2022).

4. Image Generation

To create the images, the algorithm proposed by the SPIN-IDS framework was used. The implementation was done using Python 3.12 and uses the Python module *scapy* (The Scapy Community, 2025) for packet dissection targeting TCP packets.

Each image represents a single TCP flow and is identified by the flow tuple (source IP, destination IP, source port, destination port, and protocol). A cap for the number of packets in each flow is set to 9 and reduced by 1 until the cap is set to 5. Each time the cap is reduced, a new set of images is created.

Within the packet, the Ethernet header is discarded. We continue to follow the process described by the SPIN-IDS framework and select bytes 3-9 and 11-12 from the IP header, bytes 5-20 of the Transport header, and the entirety of the payload to create a packet vector. These byte values are then converted to decimal equivalent of their hex value and are bound from [0, 255]. If the total number of bytes selected exceeds 1486 then the remainder is discarded. If the number of bytes is less than 1486 then the vector is padded with 0's to ensure that there is uniform shape.

4.1 Image Transformation

This section details the process of transforming images into transactional data sets. This representation is in the form of a CSV where each item separated by a comma is associated with the channel, the byte position, and the value in that byte position. As an example, the packet vector Red Channel [0, 25, 0, 250] would be reduced to [REDb1v32, REDb3v256].

The packet vector [[0, 25, 0, 250], [0, 0, 0, 0], [0, 0, 0, 0]] is read in and flattened to become [0, 0, 0, 25, 0, 0, 0, 0, 0, 250, 0, 0, 0, 0, 0]. Once flattened, the starting indices 0 and 1 are set for the red and green channels, respectively, and every third value from the starting positions is taken to separate the channels. To identify which channel is active in the packet vector, the mean of the values is taken and the channel whose mean is greater than zero is the active channel. The blue channel is disregarded, as this channel is only used for image shape and any non-zero value in this channel indicates that there was bleed over.

Once the channel has been identified, 0's are removed. This removal is done because of an added padding that is done during image creation. The original index positions are stored to avoid losing that information during zero removal. The first arrow in Figure 1 shows the before and after of the transformation. At this stage, the possibly values for each index are 0-255. To combat feature complexity, we employ a binning strategy.

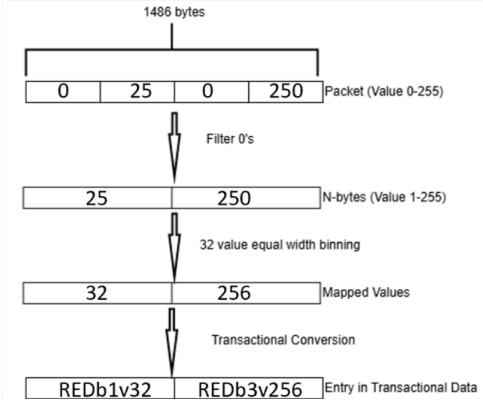

Figure 1: Data Transformation

Equal-width binning is used to reduce the number of possible values from 256 possible values to eight. Each bin covers 32 values, resulting in the eight bins. The second arrow in Figure 1 shows the value 25 in index 0 mapped to the 32 bin and the value 250 in index 1 mapped to the 256 bin.

Finally, the values are converted to their transactional form, which consists of the channel associated with the row in the image, the initial index (byte position), and the binned value. For example, Figure 1 shows 32 mapped to REDb1v32 and 256 mapped to REDb3v250. Following the trace back, the original channel was red, the original index was 1 (despite being in index 0 after the final transformation), and the binned value is 32. With this transformation, the previous vector Red Channel [0, 25, 0, 250] is now a transaction {REDb1v32, REDb3v256}.

For each image, two data sets are created with respect to the data label. The first data set is the wholistic transactional data set. In this dataset, each entry is associated with a transformed packet from a flow. This data set includes every packet of every flow as a complete record of transactions.

The second data set created is focused on the position of the packet (row of an image) within each flow corresponding to a label. For example, if a flow consisted of 4 packets, then each transformed packet vector would go to a separate file: packet 1: packet 1 transactions, packet 2: packet 2 transactions, etc. This is done for every image and generates a single, positionally based data set for the packets of the flows of a label.

4.2 Frequent Itemset Extraction

To extract frequent itemsets we use the Python module mlxtend. This module supplies a priori approach, priori stating that if an itemset is frequent, then all subsets must also be frequent. To determine what itemsets are frequent, the minimum support is initially set to 0.4 (40%).

4.3 Modelling

A CNN is used for the classification of image data. The model is implemented as specified by the SPIN-IDS framework to ensure a likewise comparison. This framework uses four convolution layers, three max pooling, batch normalization, two instances of dropout, and two dense layers. The model implementation is performed using the PyTorch deep learning library in Python and is evaluated using the accuracy metric.

5. Results

5.1 Itemset Extraction

Itemsets were extracted from all data sets using a minimum support of 0.4. Looking at the itemsets present in more than 40% of the data in each transactional database, we can identify four key bytes for potential classification: byte {2, 5, 7, 19} rather than the entire 1486 bytes included in the image.

Analysis of packet 1 data sets shows that the highest itemset support within the benign label is 0.99 and 1.0 within the threat label. Looking at itemsets with support greater than 0.8 support, the itemsets are {REDb19v128, REDb2v64, REDb5v64} with a support of 0.93 occurring together within the benign label. The threat label for packet 1 consist of longer itemsets such as {REDb2v64, REDb19v32, REDb5v64, REDb7v64, REDb18v160, REDb21v32, REDb20v128} with the highest singleton itemset (itemset of size 1) support being {REDb5v64} as 1.0 or 100% of the time occurring in a threat packet. This analysis is applied to every set of packet transactional data.

Through itemset mining, we can reduce the dataset from 9x1486x3 to 9x26x3. This new feature space contains the byte positions that were frequent in both threat and benign labels, as well as the entire transactional data set and the packet positional data sets. The feature space extracted is as follows: byte positions {0, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 26, 27, 28, 30}.

5.2 Model Performance

To adequately answer the research questions, we evaluate the model by examining overall accuracy. Overall accuracy is calculated by (TP +TN) / (P + N) where TP is true positive, TN is true negative, total positive P, and total negative N.

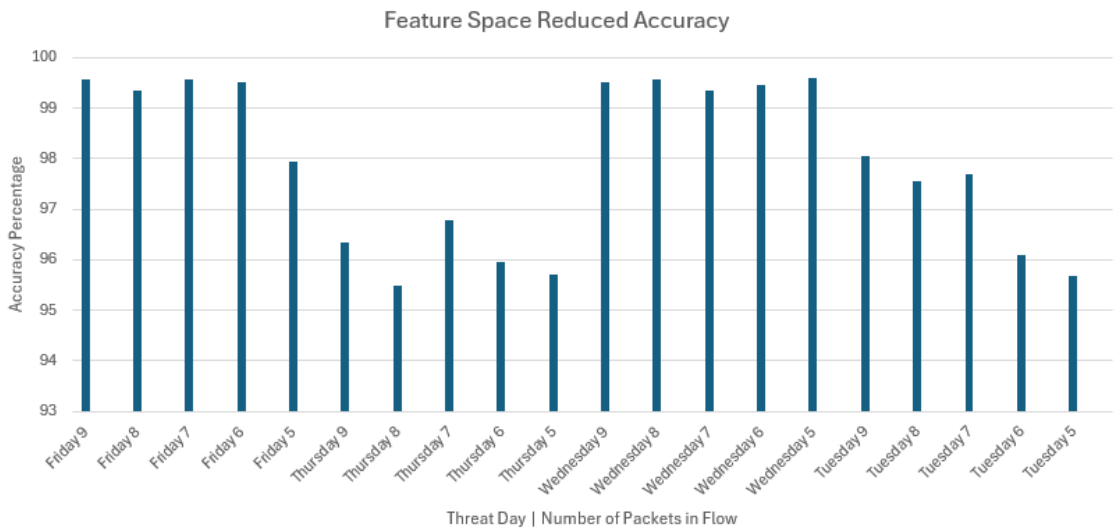

Figure 2: Frequent Itemset Reduced Feature Space Model Accuracy

For each day, the model was trained separately from the others. In doing so, an accuracy of more than 95% was achieved on all days as seen in Figure 2. Furthermore, the amount of time required to train the model with the reduced feature space is noticeably shorter than the model without a reduced feature space as seen in Figure 3.

With these results, initial research questions can be answered. To answer RQ1, we applied the SPIN-IDS framework using a CNN that has been trained and tested with the reduced dataset. Rather than 1,486 features, this CNN is trained using only the 23 identified features. Furthermore, there are four variations of the model trained, each trained on a different number of packets from within the flow (ranging from the first five packets through the first eight packets). All models will use a 70:30 train / test split.

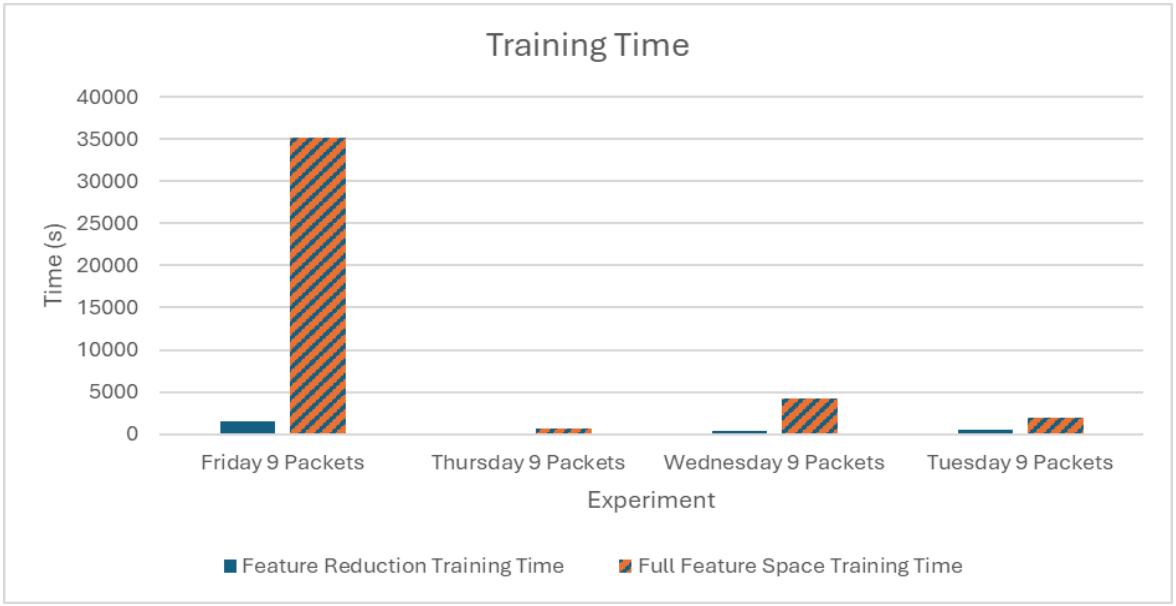

Figure 3: Training Time Comparison between Reduced Feature Space and Full Feature Space

With the reduced data set, the model has shown results of 99% accuracy on Friday with 8 packets (continuing to maintain an accuracy of above 96% down through the fifth packet). Thursday varied on the results, likely due to previously mentioned issues with the dataset but varied between 95-96% accuracy across all runs. Wednesday experiments maintained an accuracy of 99% throughout the reduced number of packets and Thursday saw a gradual decline from 98% to 95% going from nine packets to five.

Given that the SPIN-IDS framework reports 98% overall accuracy across all days, using this reduced data set achieves comparable results at earlier packets within a flow.

Answering RQ2 requires that the CNN model be trained for a binary classification task rather than the author's original multi-class classification. Rerunning the experiments set to answer RQ1 with the full feature space allow us to answer RQ2. In most experiments, the model using the reduced feature space either outperforms the model with the full feature space or has comparable results (differing by 0.5% accuracy). These results can be seen in Figure 4. There are experiments where the model using the full feature space never converged, specifically Thursday 9, Thursday 7, Wednesday 9, Wednesday 6, and Tuesday 6. Modifications to these model runs were not done as to keep experimental processes as closely comparable as possible.

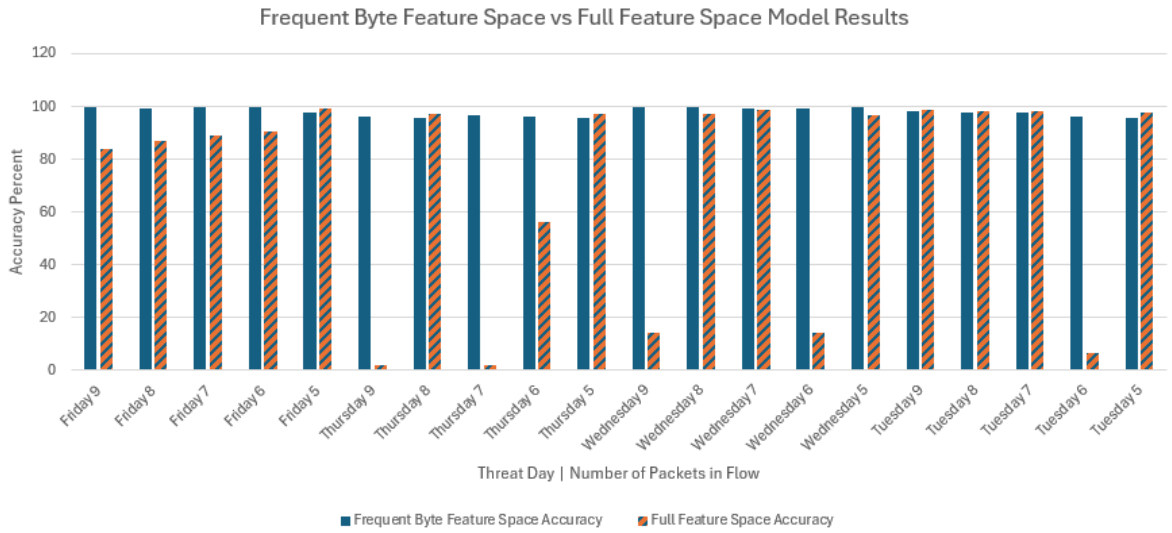

Figure 4: Comparing Model with Reduced Feature Set to Full Feature Set

6. Areas of Future Research

The data transformation efforts proposed here have given the ability to view the data in the form of transactions. In addition, the data split into subsets based on the position of the packet within the flow can be used to consider sequential rule mining. In this sense, frequent itemsets can be extracted from each packet positional dataset and merged to create a set of ordered sequential occurrences with respect to the associated bytes. This makes more sense given that the flows are based on the protocol that two endpoints are talking across, and if there is common behaviour among that protocol during conversation, then these rules could bring light to it at the byte level.

A limitation of this work is that we only used the CICIDS-2017 dataset. While the results here are promising, future work should include testing against other datasets while evaluating the impact of the feature selection method against real-time performance.

7. Conclusions

In this work, we have proposed a novel data transformation approach and application of itemset mining for the extraction of frequent packet bytes represented as images. This effort has brought forth a new way to view network data, as a transactional data set. Through this transformation, it has successfully been shown that frequent itemsets of various complexity exist not only in the wholistic transactional data set but also with respect to the packet position within the flow itself.

These itemsets can be used to reduce the feature space for image classification of image-based network intrusion detection. With results comparable to other CNNs, the proposed data transformation and feature selection method uses 1.5% of the original data (reducing the feature space from (9, 1486, 3) to (9, 23, 3)). With the reduction of data, noise is also filtered out, and accuracy is sustained at less packets within the flow (testing from the first nine packets to the first five packets in a flow).

Ethics declaration: No ethical clearance was required for the research conducted in this paper.

AI Declaration: AI tools were not used.

References

Abdi, L. and Hashemi, S., 2015. To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE transactions on Knowledge and Data Engineering*, *28*(1), pp.238-251.

Agrawal, R., Imieliński, T. and Swami, A., 1993, June. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data* (pp. 207-216).

Agrawal, R.S. and Srikant, R., 1994, September. R. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB* (pp. 487-499).

Al-Zeiadi, M.A. and AI-Maqaleh, B.M., 2025. Incremental Closed Frequent Itemsets Mining based Approach Using Maximal Candidates. *IEEE Access*.

Alok AI Lab (2025) pyDeepInsight. Available at: https://github.com/alok-ai-lab/pyDeepInsight (Accessed: October 2, 2025)

Claise, B., 2008. *Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information* (No. rfc5101).

Fernando, B., Fromont, E. and Tuytelaars, T., 2012, October. Effective use of frequent itemset mining for image classification. In *European conference on computer vision* (pp. 214-227). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ghadermazi, J., Shah, A. and Bastian, N.D., 2024. Towards real-time network intrusion detection with image-based sequential packets representation. *IEEE Transactions on Big Data*, *11*(1), pp.157-173.

Goethals, B., 2004, March. Memory issues in frequent itemset mining. In *Proceedings of the 2004 ACM symposium on Applied computing* (pp. 530-534).

Han, J., Pei, J., Yin, Y. and Mao, R., 2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data mining and knowledge discovery*, *8*(1), pp.53-87.

Ho, C.M.K., Yow, K.C., Zhu, Z. and Aravamuthan, S., 2022. Network intrusion detection via flow-to-image conversion and vision transformer classification. *IEEE Access*, *10*, pp.97780-97793.

Kim, J., Kim, J., Kim, H., Shim, M. and Choi, E., 2020. CNN-based network intrusion detection against denial-of-service attacks. *Electronics*, *9*(6), p.916.

Kumar, S., Dalal, S. and Dixit, V., 2014. The osi model: overview on the seven layers of computer networks. *International Journal of Computer Science and Information Technology Research*, *2*(3), pp.461-466.

Liu, L., Engelen, G., Lynar, T., Essam, D. and Joosen, W., 2022, October. Error prevalence in nids datasets: A case study on cic-ids-2017 and cse-cic-ids-2018. In *2022 IEEE Conference on Communications and Network Security (CNS)* (pp. 254-262). IEEE.

Nowozin, S., Tsuda, K., Uno, T., Kudo, T. and BakIr, G., 2007, June. Weighted substructure mining for image analysis. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1-8). IEEE.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. and Desmaison, A., 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, *32*.

Quack, T., Ferrari, V., Leibe, B. and Van Gool, L., 2007, October. Efficient mining of frequent and distinctive feature configurations. In *2007 IEEE 11th International Conference on Computer Vision* (pp. 1-8). IEEE.

Siddiqi, M.A. and Pak, W., 2022. An Optimized and Hybrid Framework for Image Processing Based Network Intrusion Detection System. *Computers, Materials & Continua*, *73*(2).

Schlegel, B., Gemulla, R. and Lehner, W., 2011, March. Memory-efficient frequent-itemset mining. In *Proceedings of the 14th international conference on extending database technology* (pp. 461-472).

Schölkopf, B., Smola, A. and Müller, K.R., 1998. Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5), pp.1299-1319.

Sharafaldin, I., Lashkari, A.H. and Ghorbani, A.A., 2018. *Toward generating a new intrusion detection dataset and intrusion traffic characterization*. *ICISSp*, *1*(2018), pp.108-116.

The Scapy Community (2025) Scapy. Available at: https://scapy.net/ (Accessed: October 2, 2025)

Toldinas, J., Venčkauskas, A., Damaševičius, R., Grigaliūnas, Š., Morkevičius, N. and Baranauskas, E., 2021. *A novel approach for network intrusion detection using multistage deep learning image recognition*. *Electronics*, *10*(15), p.1854.

Yao, B. and Fei-Fei, L., 2010, June. Grouplet: A structured image representation for recognizing human and object interactions. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 9-16). IEEE.

Zaki, M.J., 2002. Scalable algorithms for association mining. *IEEE transactions on knowledge and data engineering*, *12*(3), pp.372-390.