BFIS: Efficient Unknown Protocol Feature Extraction Method For Satellite Communication Systems

Xianwen Ling[1], Kun Zhang*[1] and Rong Tong[2], Dianying Chen[1]
[1] Nanjing University of Science and Technology, China
lingxw@njust.edu.cn, zhangkun@njust.edu.cn, cdy0507@163.com
[2] Singapore Institute of Technology Singapore, Singapore
tong.rong@singaporetech.edu.sg

2025 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) | 979-8-3315-7206-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/APSIPAASC65261.2025.11249386

Abstract*—With the rapid development of fields such as the Internet and satellite communications, the number of network protocol types has gradually increased, including numerous proprietary or unknown protocols. However, most research has focused on common network protocols, neglecting satellite communication protocols. Given the relatively simple features of captured bitstream data, extracting key bitstream sequences as protocol features is an effective method for satellite protocol feature extraction. Based on the analysis of satellite protocol structures, including DVB compliant with ETSI standards, this paper proposes a feature extraction method utilizing the BFIS algorithm and frequent sequence splicing via the FSS algorithm. This approach dynamically stores subsequence frequencies in a feature-analysis matrix, calculates similarities between different modes, and is employed to extract features of unknown satellite protocols based on DVB. The clustering results are then mapped to actual protocols. Experimental results on both simulated satellite protocol data and the ISCX VPN-nonVPN dataset show that the BFIS algorithm significantly improves accuracy, achieving 97.62% accuracy on DVB datasets and 95.16% accuracy on ISCX VPN-nonVPN dataset, demonstrating its effectiveness in extracting satellite protocol features.

I. Introduction

The rapid evolution of satellite communication technologies has driven the emergence of novel protocols, yet unidentified protocols pose critical risks including communication failures and security breaches [1], [2]. Current research predominantly focuses on known protocol analysis through rule-based and statistical methods [3], while solutions for unknown protocol recognition remain underdeveloped. Recent advances leverage data-driven approaches such as deep learning [4] and multi-feature fusion [5] to address dynamic network environments. To overcome limitations in handling complex unknown protocols, this work proposes a machine learning-based framework integrating statistical pattern mining and adaptive feature extraction, enabling robust protocol identification without prior knowledge while maintaining computational efficiency.

With the diversification of network protocols and the increasing complexity of network attack methods, traditional methods struggle to meet the growing demand for efficient, flexible, and accurate analysis [6], [7]. Therefore, this paper proposes a feature extraction and pattern recognition method based on statistical learning and machine learning, which effectively extracts valuable frequent patterns from complex satellite protocol traffic.

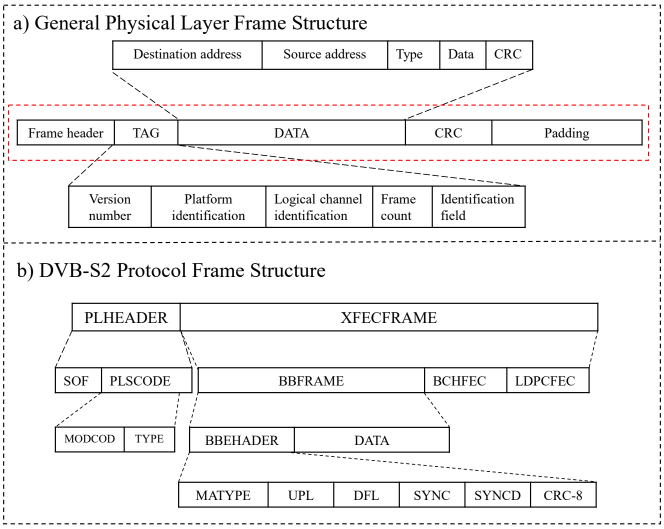

Fig. 1. **Physical Layer Frame Structure in Satellite Communication Protocols.** The illustrated frame architecture delineates the integrated organization of satellite communication protocols, which underpins BFIS's efficient feature extraction mechanism.

In summary, the main contributions of this paper include:

- Propose a feature extraction method for the physical layer of satellite protocols. This method combines the FSS algorithm for frequent sequence concatenation, utilizing a feature analysis matrix to dynamically store subsequence frequencies and calculate the similarity between different patterns. It overcomes the problem of low feature extraction efficiency in traditional bitstream protocols.
- Introduce a general frame structure to represent the physical layer of the satellite protocol, with protocol frame structures of DVB-S2, DVB-S2X, DVB-RCS, and DVB-RCS2 analyzed based on ETSI standards, and validation is performed using public datasets.
- Present experimental results showing that based on the

proposed feature extraction algorithm, mapping clustering results to actual protocols yields accuracy rates of 97.62% and 95.16% on simulation-generated datasets and the ISCX VPN-nonVPN dataset, outperforming traditional advanced algorithms.

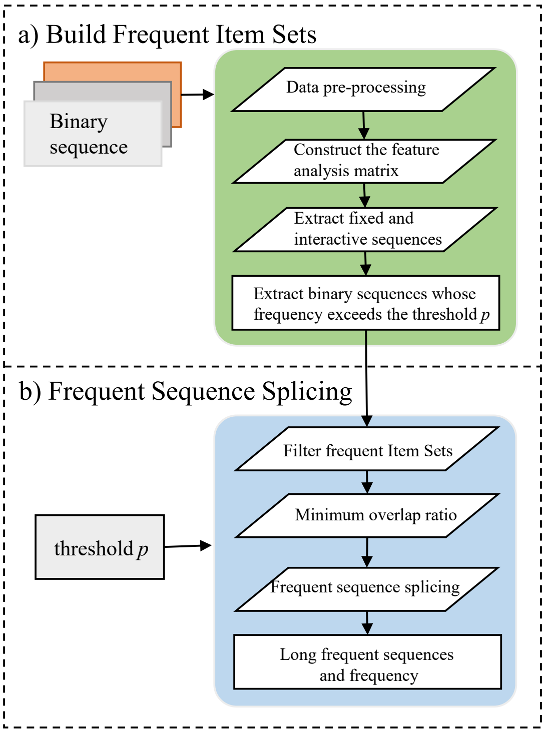
a) Build Frequent Item Sets
Binary sequence
Data pre-processing
Construct the feature analysis matrix
Extract fixed and interactive sequences
Extract binary sequences whose frequency exceeds the threshold p
b) Frequent Sequence Splicing
Filter frequent Item Sets
Minimum overlap ratio
threshold p
Frequent sequence splicing
Long frequent sequences and frequency

Fig. 2. **Schematic diagram of feature extraction.**

II. Related Works

A. Protocol Feature Structure

Current research on DVB satellite protocols predominantly focuses on DVB-S2, DVB-S2X, DVB-RCS, and DVB-RCS2 standards. As shown in Fig. 1, we select DVB-S2 as the representative case, constructing a frame structure compliant with ETSI standards [8] that comprises XFECFRAME timeslots, pilot blocks, and PLHeader segments (SOF + PLSCODE). While DVB-S2X extends the MODCOD schemes from DVB-S2, the RCS/RCS2 protocols however employ burst-based encapsulation with TCP/IP and Generic Stream Encapsulation (GSE) formats for return channel signaling [9]. Salih and Hameed conducted an comprehensive review of the DVB-S2 satellite communication system, emphasizing its enhanced performance characteristics over previous standards [10].

B. Protocol Feature Extraction

Traditional methods predominantly extract packet-level features (IP/port identifiers) and traffic statistics (packet size/delay) through manual rule design [11]–[13]. While effective in controlled environments, these approaches struggle with computational efficiency and generalization in dynamic networks due to dependency on predefined features [14]. Emerging machine learning techniques address these limitations through automated pattern discovery. Support Vector Machines(SVMs) enable statistical feature classification [15], while graph neural networks(GNNs) model protocol relationships through graph convolutional operations [16]. Hybrid architectures(e.g., CNN-LSTM) improve accuracy by jointly learning spatiotemporal packet features [17], demonstrating superior adaptability to encrypted traffic and unknown protocol scenarios compared to conventional methods.

III. Methodology

A. Pre-processing

Preprocessing forms a critical foundation in unknown protocol recognition, preprocessing methods mainly address two tasks:

a) **Uniform data length**: This operation focuses on eliminating consecutive repeated values at the end of protocol data and ensuring data consistency. In practical scenarios, when the effective length of protocol data is too short, large amounts of consistent padding is often added to meet the length standards specified by the protocol. These trailing padding fields hold no substantial meaning, and as protocol data containing such padding increases, the data distribution is disrupted, ultimately interfering with the accuracy of protocol recognition.

b) **Removal of “0x00” values**: The “0x00” value is very common across various protocol data, and its semantics can vary depending on the protocol. Removing this value can effectively reduce the ambiguity of data features and highlight the key characteristics of the protocol data. We will keep the protocol data consistent after removal.

For a given frame of length n, it extracts subsequences of length l using a sliding window technique. Let $frame = [b_1, b_2, \cdots, b_n]$, the extracted subsequence is $sequence = [b_i, b_{i+1}, \cdots, b_{i+l-1}]$, and it is converted into a decimal number using the formula:

$$decimal = \sum_{j=0}^{l-1} b_{i+j} \times 2^{l-1-j} \tag{1}$$

where,$b_{i+j} \in \{0, 1\}$. Fig. 2 shows the schematic diagram of feature extraction, included building frequent itemsets and frequent sequence splicing.

B. Feature extraction

a) **Building Frequent Itemsets**: In this study, the frequent itemsets are constructed using the BFIS (Build Frequent Item Sets) as summarized in Algorithm 1. The following outlines the algorithm process and the principles and operations involved in each key step.

The method initializes a defaultdict structure as feature_matrix$[k] = \sum_m$ local_matrix$_m[k]$, where m iterates over all local matrices. Multithreading processes all data frames in parallel, merging local matrices into a global feature matrix that dynamically stores subsequence occurrence frequencies. The proposed algorithm leverages defaultdict properties to construct the query matrix in a single step, replacing traditional multi-scan approaches and eliminating the need for repetitive scanning. For each local result, key-value pairs update the feature matrix at corresponding keys, with non-existent keys automatically created and initialized to 0. The query matrix is generated by linking frequency information with binary sequences, where each key-value pair in the feature matrix directly maps to a query matrix entry. Finally, for the classification of frequent itemsets, for each element in the query matrix, the total frequency is first calculated:

$$total\_frequency = \sum_i frequencies[i] \tag{2}$$

If the number of sequences is 1 or the total frequency equals the first frequency value, the itemset is marked as a “fixed” type and the frequent itemset is generated. Otherwise, the Jaccard similarity between different sequences is calculated. Let the sets corresponding to two sequences be s_1 and s_2, the Jaccard similarity is given by the formula:

$$sim = \frac{|s_1 \cap s_2|}{|s_1 \cup s_2|} \tag{3}$$

When $sim < repe_{rate}$, the itemset is marked as “interactive” type. In Fig. 3, we compare the frequent sequence mining time of different feature extraction algorithms. Our BFIS algorithm can accelerate the calculation process when processing large-scale data due to the use of multithreading for parallel computation. A binary sequence is used to represent a set of feature items. It can generate interactive frequent item sets according to sequence similarity, avoiding the fixed candidate set generation process like Apriori and FEMSA(feature extraction method of statistical analysis), thus improving the flexibility and adaptability of the algorithm [18], [19].

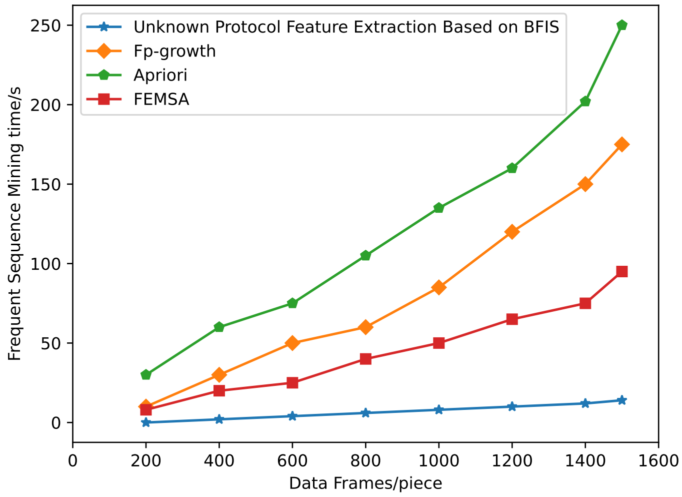

Fig. 3. **Comparison of frequent sequence mining time for different algorithms with varying data frame sizes.**

The computational complexity of the BFIS algorithm is analyzed as follows:

Algorithm 1: BFIS Algorithm

```
Input: Binary Data Frames, l, Similarity Threshold
Output: Frequent Item Set $FIS$
Initialize feature matrix: $FM \leftarrow \emptyset$
foreach $frame\ f \in data\_frames$ in parallel do
    for $i \leftarrow 0$ to $|f| - l$ do
        $s \leftarrow f[i : i + l]$
        $decimal \leftarrow$ BinaryToDecimal($s$)
        $FM[decimal] \leftarrow FM[decimal] + 1$
    end
end
$query\_matrix \leftarrow \emptyset$
foreach $(decimal, f) \in FM$ do
    $query\_matrix[decimal] \leftarrow \{sequences :$
      $[$DecimalToBinary$(decimal, l)], frequencies :$
      $[f]\}$
end
$FIS \leftarrow \emptyset$
foreach $(decimal, data) \in query\_matrix$ do
    $s \leftarrow data.sequences$  $f \leftarrow data.frequencies$
     $total\_f \leftarrow$ Sum($f$)
    if $|s| = 1$ or $total\_f = f[0]$ then
        $fis[decimal] \leftarrow \{type :$ fixed$, sequences :$
          $s, frequencies : f\}$
    else
        for $i \leftarrow 0$ to $|s| - 1$ do
            for $j \leftarrow i + 1$ to $|s|$ do
                $s1, s2 \leftarrow$ Set$(s[i])$, Set$(s[j])$
                $similarity \leftarrow |s1 \cap s2| / |s1 \cup s2|$
                if $similarity \geq repe\_rate$ then
                    $fis[decimal] \leftarrow \{type :$
                     interactive$, sequences :$
                     $[s[i], s[j]], frequencies :$
                     $[f[i], f[j]]\}$
                end
            end
        end
    end
end
return Frequent Item Set
```

- **Time Complexity:** $\mathcal{O}(n \cdot m/p)$, where n is the number of data frames, m is the frame length, and p is the number of parallel threads.
- **Space Complexity:** $\mathcal{O}(k)$, where k is the number of distinct subsequences.
- **Comparison with Traditional Methods:**
  - *Apriori:* $\mathcal{O}(n \cdot m \cdot 2^l)$ - requires multiple scans to generate candidate sets
  - *FP-growth:* $\mathcal{O}(n \cdot m \cdot \log m)$ - requires FP-tree construction
  - *BFIS Advantage:* Through parallel processing and single-scan strategy, the complexity is reduced by approximately a factor of p (where p=8 threads in our

experiments), resulting in 6-8x speedup compared to sequential Apriori.

b) **Frequent Sequence Splicing**: Frequent sequence splicing employs the BFIS algorithm to merge short frequent sequences through overlap-based concatenation, where dual-loop frame scanning first extracts length-l subsequences from preprocessed data to build a frequency matrix recording subsequence positions and occurrence counts. Candidate sequences meeting the minimum overlap ratio p (overlap length/shorter sequence length) are then iteratively merged, with deduplication finally outputting maximally extended protocol patterns retaining comprehensive sequence coverage.

In the Algorithm 2, the first step is to sort the input frequent itemsets based on the starting position pos, which ensures that the comparisons are done in order, reducing unnecessary redundant comparisons. Then, the merging loop begins, initializing an empty set. The outer loop iterates over each sequence x, extracting its sequence content, starting position $posx$, length $lenx$, and frequency $freq$. The inner loop iterates over each sequence y that comes after x, extracting the relevant attributes: $seqx$, $seqy$, $posy$, and $leny$. If the starting positions and sequence contents are the same, the loop skips that pair. Otherwise, the overlap length is calculated:

$$overlap\_len = \max(0, posx + lenx - posy) \tag{4}$$

If the overlap length is greater than 0 and the proportion of the overlap relative to the length of the smaller sequence reaches the minimum overlap ratio, then a new concatenated sequence is created:

$$new\_sequence = seqx + seqy[overlap\_len] \tag{5}$$

The new sequence and its corresponding frequency and starting position are then appended to the set of long frequent sequences. Finally, using a dictionary comprehension, the elements in the long frequent sequences are deduplicated by using the sequence content as the key. The corresponding values are extracted and returned as the final output, ensuring that each sequence in the output set is unique.

C. Hierarchical clustering

This study determines the number of clusters based on the length of the unique label list and performs clustering using the agglomerative clustering algorithm in hierarchical clustering. After setting the number of clusters as $n_{clusters}$, the training feature matrix is employed for model training. An empty list is initialized to store cluster centers. For each cluster label, the following operations are executed: training features belonging to the cluster are selected; if the features are non-empty, their mean is calculated as the cluster center and added to the list; if the features are empty, an alternative center is computed using the mean of the entire training feature matrix. This mechanism ensures effective computation of cluster centers even when empty clusters exist. Finally, the cluster center list is converted into a numpy array for subsequent computations.

Algorithm 2: FSS Algorithm

```
Input: List of frequent sequences, min overlap ratio $p$
Output: Set of merged frequent sequences $LFS$
Sort $FIS$ by starting position
$LFS \leftarrow \emptyset$
foreach $x \in FIS$ do
    $s_x, p_x, l_x, f_x = \text{Extract}(x)$
    foreach $y$ after $x$ do
        $s_y, p_y, l_y = \text{Extract}(y)$
        if $p_x == p_y \wedge s_x == s_y$ then
            continue
        end
        $o = \max(0, p_x + l_x - p_y)$
        if $o > 0 \wedge o/\min(l_x, l_y) \geq p$ then
            $s_{new} = s_x + s_y[o:]$
            $LFS.\text{add}(s_{new}, \min(f_x, f_y), p_x)$
            break
        end
    end
end
return Unique($LFS$)
```

TABLE I
DATASETS

Protocol	Number of data frames/piece
DVB-S2	2000
DVB-S2X	2000
DVB-RCS	2000
DVB-RCS2	2000
ARP	15492
DNS	1575
TCP	1139
UDP	13143

IV. Experiment

As shown in Table I, the protocols studied in this paper include DVB satellite protocol, binary protocols, TCP protocol, UDP protocol, and others. In the context of simulating DVB protocol data using ETSI standards and combining it with publicly available pcap data [20] for research, the data preprocessing workflow presents multi-stage characteristics that are highly adaptable to the provided code. The parameters in this paper are elaborated in Table II. In our comparative algorithms, the parameters of other algorithms are also set to the same values.

TABLE II
Model training parameters

Parameters	Value
Initial sequence length l, Similarity Threshold	[12,0.5]
Minimum overlap ratio p	0.5
n_clusters	len(unique labels)
epochs	10

TABLE III
CLASSIFICATION EVALUATION METRICS FOR FOUR ALGORITHMS

Protocol	BFIS			Apriori			FEMSA			Fp-growth		
	Prec.%	**Rec.%**	**F1.%**	**Prec.%**	**Rec.%**	**F1.%**	**Prec.%**	**Rec.%**	**F1.%**	**Prec.%**	**Rec.%**	**F1.%**
DVB-RCS	**100.00**	91.67	95.65	94.18	91.67	92.91	95.16	91.67	93.38	100.00	91.67	95.65
DVB-RCS2	92.31	**100.00**	**96.00**	91.88	94.33	93.09	91.67	95.33	93.46	0.00	0.00	0.00
DVB-S2	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
DVB-S2X	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	100.00	99.67	99.83	48.00	100.00	64.86
Accuracy		**97.92**			96.50			96.67			72.92	
Macro avg	**98.08**	**97.92**	**97.91**	96.52	96.50	96.50	96.71	96.67	96.67	62.00	72.92	65.13
Weighted avg	**98.08**	**97.92**	**97.91**	96.52	96.50	96.50	96.71	96.67	96.67	62.00	72.92	65.13
ARP	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	49.00	**100.00**	**66.00**
DNS	58.44	50.37	54.11	65.79	27.99	39.27	54.00	72.00	62.00	0.00	0.00	0.00
TCP	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	0.00	0.00	0.00
UDP	93.38	95.13	94.25	90.92	**98.02**	94.34	96.00	92.00	94.00	0.00	0.00	0.00
Accuracy		95.16			95.10			95.00			49.09	
Macro avg	87.95	86.38	87.09	89.18	81.50	83.40	87.00	90.00	88.00	12.00	25.00	16.46
Weighted avg	94.89	95.16	95.01	94.28	95.10	94.20	96.00	95.00	95.00	24.09	49.09	32.32

The raw data is first converted into a binary stream for processing. We employ a frequent itemset construction algorithm to extract representative features: a sliding window traverses the data frames, converting binary sequences within the window into decimal values and counting their occurrence frequencies. Multithreaded parallel processing using a thread pool accelerates frequent itemset generation, aggregating high-frequency binary sequences and their frequencies into a feature matrix. The feature set is further optimized by selecting and merging highly similar itemsets to construct refined feature representations.

After building the feature matrix, features are extracted from both training and validation sets. To ensure consistency, all possible feature keys are first extracted and sorted from the training set. The validation set then uses the same feature keys for extraction, followed by a standardization process to eliminate scale differences, producing standardized feature vectors for subsequent clustering tasks.

We train the model on the training set using the agglomerative clustering algorithm. The number of clusters is set to match the total count of protocol types, ensuring each protocol type corresponds to a unique cluster. Validation set data points are assigned to the nearest cluster centers by calculating Euclidean distances between each validation instance and all cluster centers.

The protocol recognition performance is quantified through four core metrics derived from confusion matrix elements (TP, TN, FP, FN):

- **Precision**: $P = \frac{TP}{TP+FP}$ (Minimizing false alarms)
- **Recall**: $R = \frac{TP}{TP+FN}$ (Maximizing true detections)
- **F1-Score**: $F1 = \frac{2PR}{P+R}$ (Harmonic balance)
- **Accuracy**: $A = \frac{TP+TN}{TP+TN+FP+FN}$ (Overall correctness)

For multi-class evaluation:

- **Macro Avg**: $\frac{1}{N}\sum_{i=1}^{N} Metric_i$ (Class-balanced perspective)
- **Weighted Avg**: $\frac{\sum(w_i Metric_i)}{\sum w_i}$ (Sample-size weighted)

Based on the classification evaluation results presented in the Table III, the BFIS algorithm demonstrates exceptional performance across all tested protocols, particularly excelling in DVB-related protocol classification. It achieves an overall accuracy of 95.16%, second only to Apriori and FEMSA, but still outperforming other algorithms like Fp-growth (with an accuracy of 49.09%). The algorithm's precision, recall, and F1 score remain consistently high, reflecting its efficiency and reliability.

In ARP protocol classification, BFIS achieves 100% precision, recall, and F1 score, perfectly identifying all ARP protocol, similar to Apriori and FEMSA. In DNS protocol, while BFIS's precision (58.44%) lags behind Apriori (65.79%), its recall (50.37%) outperforms Apriori (27.99%), highlighting BFIS's ability to better avoid false negatives. In TCP protocol classification, BFIS maintains its dominant performance with 100% precision, recall, and F1 score, outperforming all other algorithms. In the classification of UDP protocol, BFIS achieves solid results with precision (93.38%), recall (95.13%), and F1 score (94.25%), which, although slightly lower than FEMSA, are still far superior to Fp-growth's 0%. The macro-average and weighted-average metrics also reflect BFIS's strong performance, with values of 87.09% and 95.01%, respectively. Fp-growth shows 0.00% for certain protocols due to its inability to extract meaningful features from these specific protocol structures within the given parameter constraints.

Overall, the BFIS algorithm excels in protocol classification, particularly for DVB protocols, showcasing its effectiveness in accurately and efficiently classifying ARP, TCP, and UDP protocol. Despite some limitations in DNS protocol precision, its high recall ensures a more comprehensive recognition capability, making BFIS a highly reliable and effective choice for protocol classification tasks.

V. Conclusion

This study addresses the challenge of unknown protocol identification in satellite communications by proposing a feature extraction method based on the BFIS algorithm. Through multi-threaded parallel processing and dynamic frequency storage mechanisms, the method achieves 97.62% accuracy on DVB protocol datasets and 95.16% accuracy on the ISCX VPN-nonVPN dataset, significantly outperforming traditional algorithms. Future work will explore integration with deep learning techniques to further enhance the recognition capability for complex protocol patterns.

Acknowledgements

This work was supported by the Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant (No. KYCX25_0758) and the Jiangsu Provincial Science & Technology Major Project of China under Grant (No. BG2024043).

References

[1] T. Pratt and J. E. Allnutt, *Satellite communications*. John Wiley & Sons, 2019.

[2] X. Zhang, Y. Wang, X. Qin, Z. Zhang, H. Zhou, and X. Shen, "Link-level performance analysis of dvb standards in ultra-dense leo satellite-terrestrial networks," in *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, IEEE, 2024, pp. 01–05.

[3] B. Ma, C. Yang, M. Chen, and J. Ma, "Grammatch: An automatic protocol feature extraction and identification system," *Computer Networks*, vol. 201, p. 108 528, 2021.

[4] R. Ma and S. Qin, "Identification of unknown protocol traffic based on deep learning," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, IEEE, 2017, pp. 1195–1198.

[5] X. Zheng and H. Li, "Identification of malicious encrypted traffic through feature fusion," *IEEE Access*, vol. 11, pp. 80 072–80 080, 2023.

[6] A. Rovira-Sugranes, A. Razi, F. Afghah, and J. Chakareski, "A review of ai-enabled routing protocols for uav networks: Trends, challenges, and future outlook," *Ad Hoc Networks*, vol. 130, p. 102 790, 2022.

[7] Y. Liu, F. Zhang, Y. Ding, J. Jiang, and S.-H. Yang, "Sub-messages extraction for industrial control protocol reverse engineering," *Computer Communications*, vol. 194, pp. 1–14, 2022.

[8] ETSI, *European telecommunications standards institute*, https://www.etsi.org, Accessed: 2025-01-13, 2025.

[9] A.-F. B. A. Bachir, M. Zhour, and M. Ahmed, "Modeling and design of a dvb-s2x system," in *2019 5th International Conference on Optimization and Applications (ICOA)*, IEEE, 2019, pp. 1–5.

[10] O. M. Salih and A. Q. Hameed, "An overview performance of dvb-s2 link system," in *AIP Conference Proceedings*, AIP Publishing, vol. 3002, 2024.

[11] M.-G. Kim and H. Kim, "Anomaly detection in imbalanced encrypted traffic with few packet metadata-based feature extraction.," *CMES-Computer Modeling in Engineering & Sciences*, vol. 141, no. 1, 2024.

[12] R. E. Nogales and M. E. Benalcázar, "Analysis and evaluation of feature selection and feature extraction methods," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 153, 2023.

[13] Y. Yang, Y. Yan, Z. Gao, *et al.*, "A network traffic classification method based on dual-mode feature extraction and hybrid neural networks," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4073–4084, 2023.

[14] Z. Liu, X. Zha, G. Song, and Q. Yao, "Unknown wireless network protocol feature extraction method based on sequence association," in *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, IEEE, 2020, pp. 1916–1922.

[15] L. Zou, X. Luo, Y. Zhang, X. Yang, and X. Wang, "Hcdttsvm: A network intrusion detection method based on decision tree twin support vector machine and hierarchical clustering," *IEEE Access*, vol. 11, pp. 21 404–21 416, 2023.

[16] H. Zhang, L. Yu, X. Xiao, *et al.*, "Tfe-gnn: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2066–2075.

[17] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "Cnn-lstm: Hybrid deep neural network for network intrusion detection system," *IEEE Access*, vol. 10, pp. 99 837–99 849, 2022.

[18] X. Yuan, "An improved apriori algorithm for mining association rules," in *AIP conference proceedings*, AIP Publishing, vol. 1820, 2017.

[19] P. Van Huong, N. H. Minh, *et al.*, "Improving the feature set in iot intrusion detection problem based on fp-growth algorithm," in *2020 International Conference on Advanced Technologies for Communications (ATC)*, IEEE, 2020, pp. 18–23.

[20] C. I. for Cybersecurity, "Vpn-nonvpn dataset (iscxvpn2016)," in *Canadian Institute for Cybersecurity Data Collection*, University of New Brunswick, 2016.