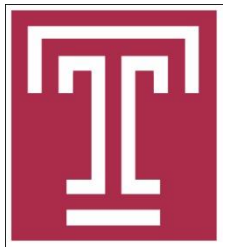
CrossAlert: Enhancing Multi-Stage Attack Detection through Semantic Embedding of Alerts Across Targeted Domains

Nadia Niknami, Vahid Mahzoon, Jie Wu

Dept. of Computer and Info. Sciences

Temple University



CNS2024

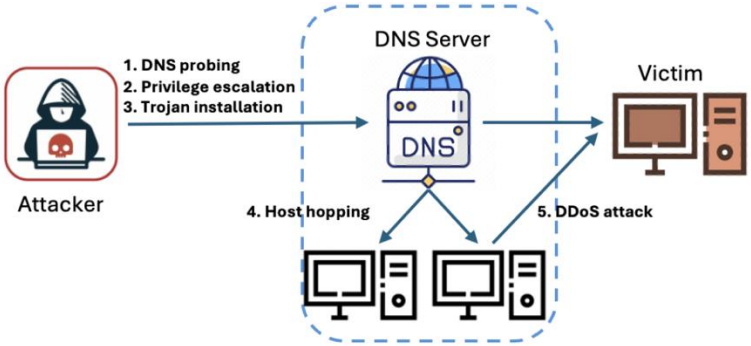
Outline

- Intrusion Detection System(IDS)
- Multi-Stage Attacks (MSAs)
- Problem Statement
- The Proposed Approach: CrossAlert
- Results and Discussion

Intrusion Detection System (IDS)

- IDS is a network security tool that monitors network traffic and devices for known malicious activity, suspicious activity or security policy violations.

Multi-Stage Attacks (MSAs)



- MSAs consist of a series of steps that appear harmless individually but are dangerous when combined.
- Requires identifying both individual steps and their relationships.

Problem Statement

Cyber-attacks have become increasingly complex and distributed.

IDSs generate numerous alerts during MSAs, but high false positives and poor detection performance are common due to high dimensionality and diverse features.

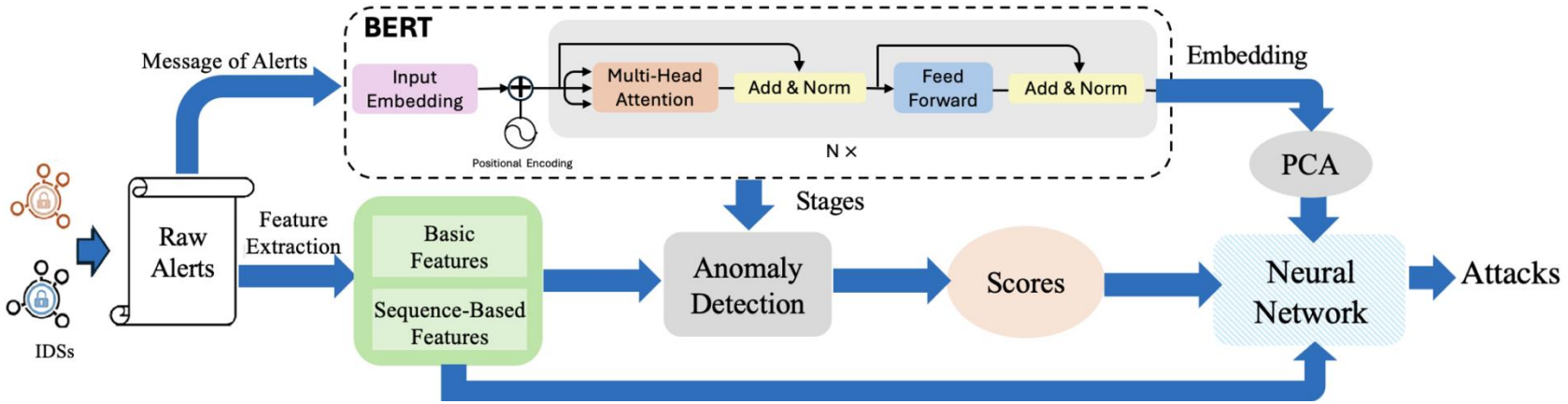
IDS models often fail when applied to different environments or domains, leading to performance degradation.

Attack Stages and Alerts

- In order to detect MSAs, it is crucial to identify alerts related to each potential stage.
- The attacker's attack actions are implemented step by step. A whole attack scenario includes multiple stages and each stage has its characteristics.
- We categorize alerts into four attack stages: scan, exploit, get-access-privilege, and post-attack, as shown in Table.

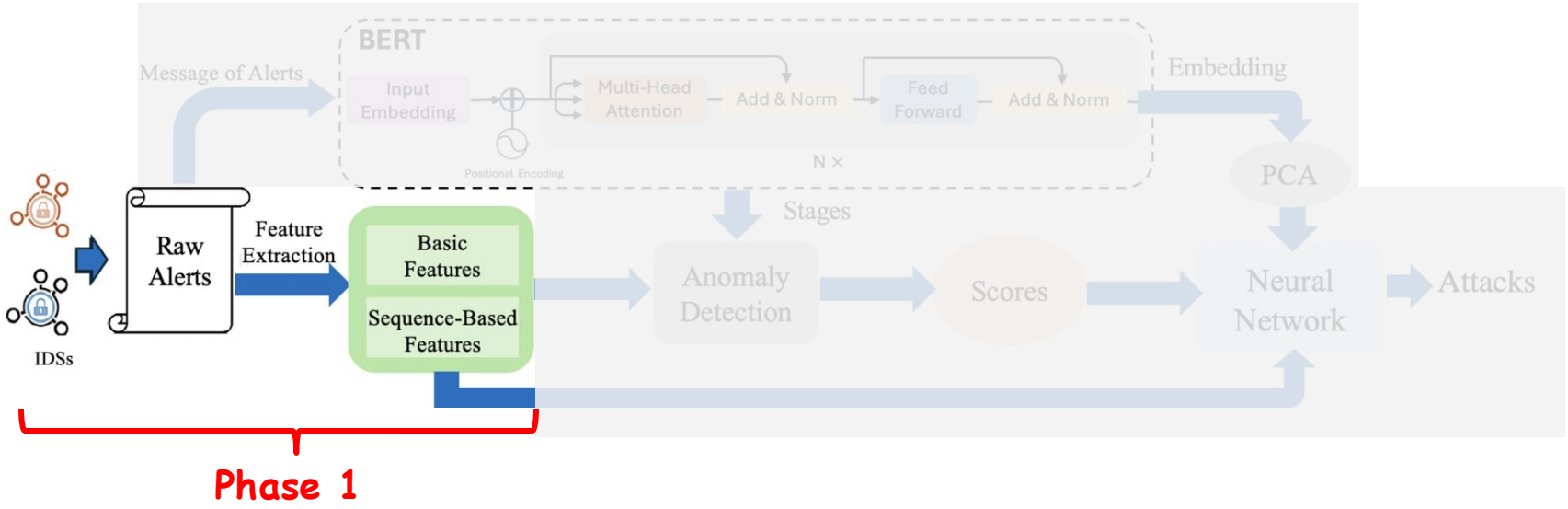
No.	Alert Stage	Alert Types
1	Reconnaissance or Scan	IP address scan, Port scan, Version scan, Vulnerability scan, Social engineering
2	Exploit	Malicious file in network traffic and host, Command injection, Vulnerability attack
3	Get Access	SSH login, RDP login, shell connect
4	Post-Penetrate or Post-Attack	Data transfer, command & control, backdoor communication

CrossAlert



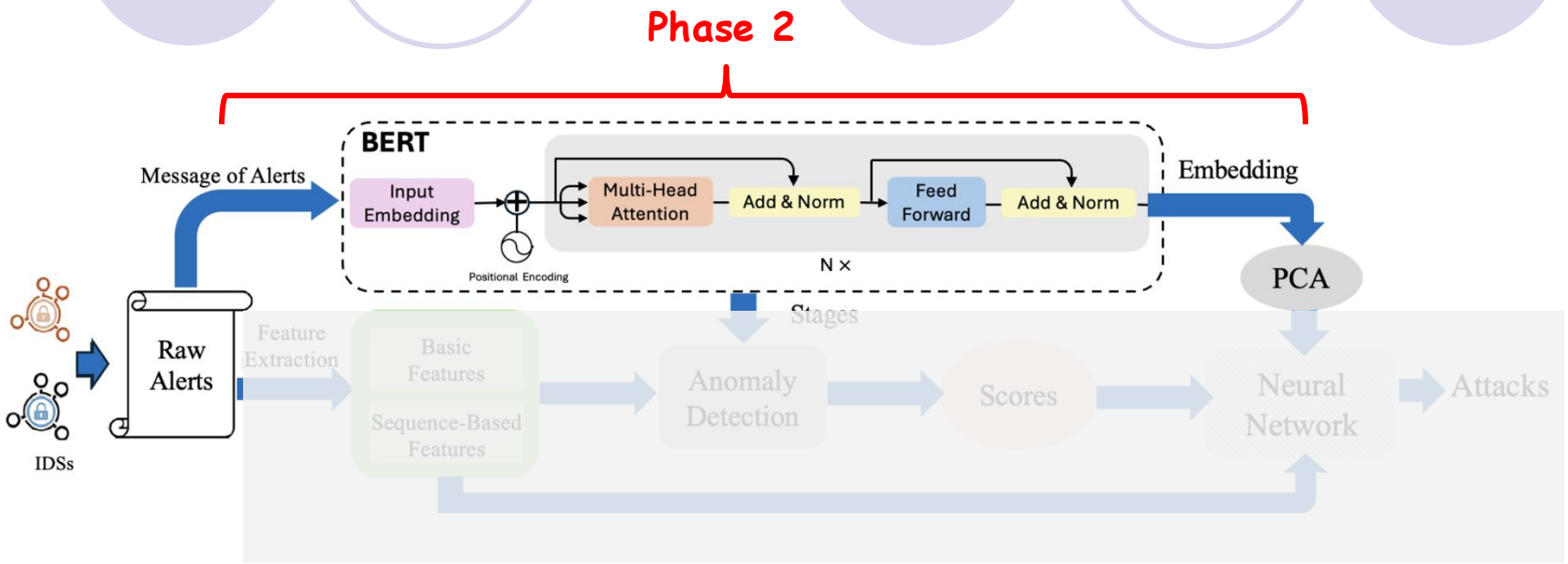
- **Objective**: Enhance MSA detection through the analysis of high-dimensional alerts and multi-faceted integration.
- **Method**: Leverages Natural Language Processing (NLP) and Prototypical Networks to address issues like false positives and domain shift.
- **Approach**: Combines semantic embeddings (from alert messages), anomaly scores, and extracted features to detect hidden attack patterns

Phase (1): Alert Preprocessing and Feature Extraction



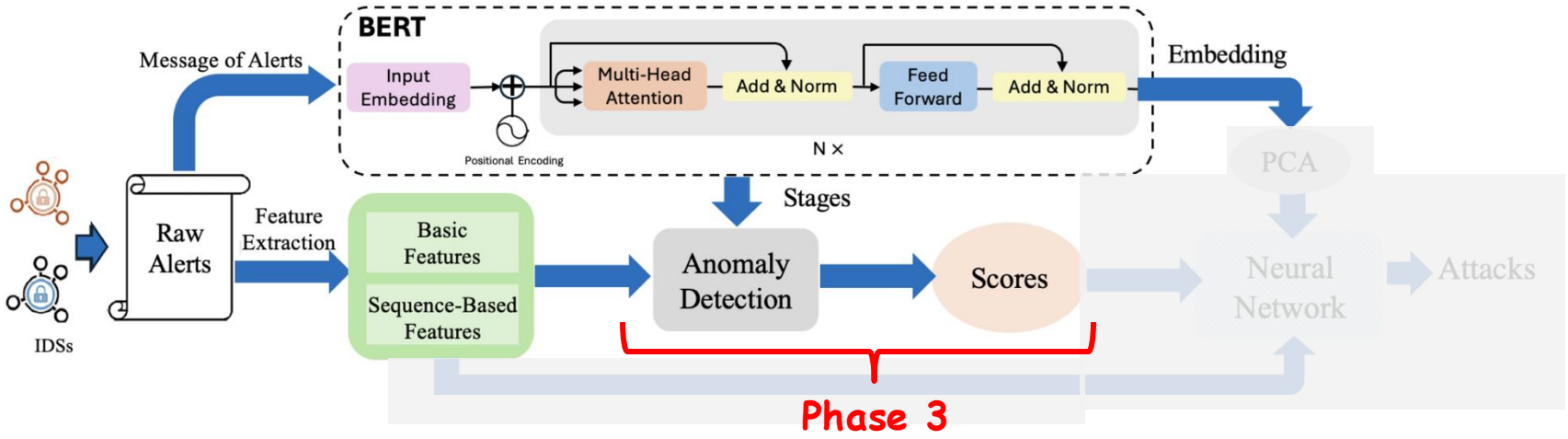
- **Feature extraction** involves deriving both:
  - **Basic features**, which are independent of the alert sequence
    - Such as **Source IP (SIP)** and **Destination IP (DIP)**
  - **Sequence-based features**, which capture temporal information from the alert sequence.
    - **Attack stage** and the counts of alerts sharing the same message from either the same SIP, DIP, or their combination

Phase (2): Semantic Embedding with BERT



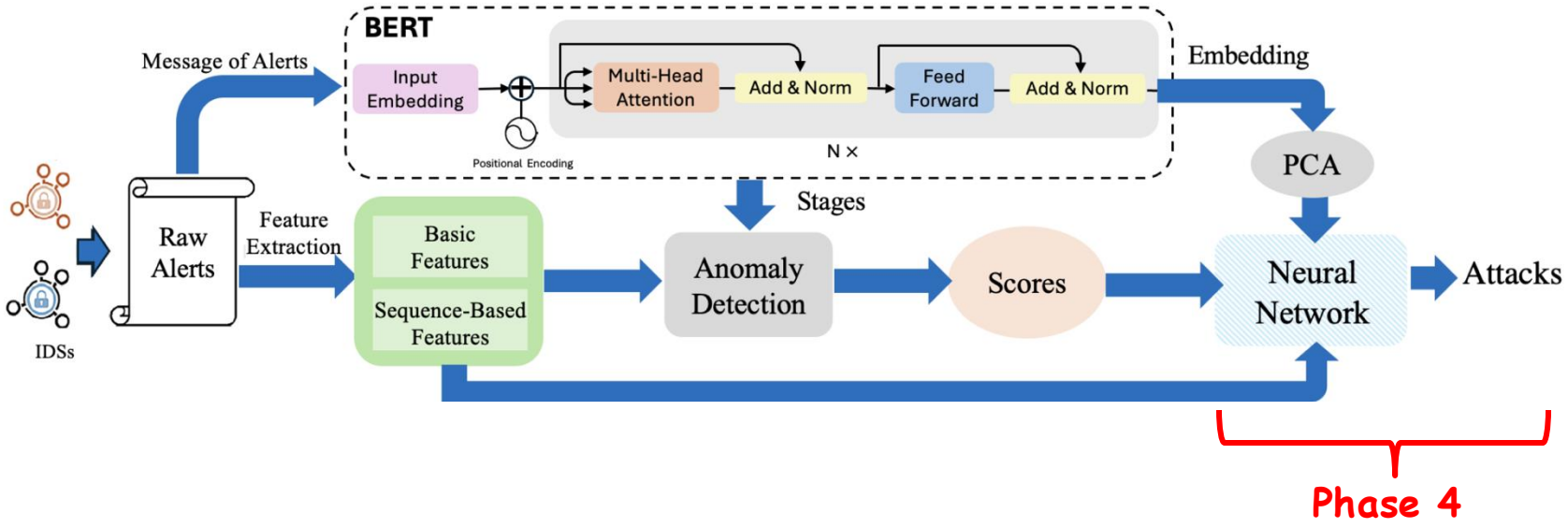
- Converts alert messages into **meaningful embeddings**.
- Similar attack methods result in similar alert information. Therefore, by learning alert semantic representations from a large number of alert sequences, it is possible to effectively represent alerts.
- **BERT** (Bidirectional Encoder Representations from Transformers) is leveraged to capture and analyze the semantic similarities between different alert messages, enabling more effective representation and understanding of alerts.

Phase (3): Anomaly Score



- The Basic and Sequence-based features are fed into an anomaly detection module to calculate the anomaly score for each alert.
- Isolation Forest is an unsupervised anomaly detection technique that employs multiple decision trees to compute anomaly scores, which are then averaged.

Phase (4): Classification



- Principal Component Analysis (PCA) is applied to compress the BERT embeddings for alert messages.
-
- The compressed embeddings are concatenated with the obtained anomaly score, the Basic and Sequence-based features, and then fed into a Feed Forward Neural Network that classifies the alerts to detect multi-stage attacks.

Pseudo-Code

Algorithm 1 Cross-Alert Algorithm

Input $\{a_1, a_2, \ldots, a_N\}$: Alerts in chronological order, $\{(a_{s_1}, y_{s_1}), \ldots, (a_{s_M}, y_{s_M})\}$: Subset of labeled alerts
Output Trained Classifier f_W with parameters W
$B \leftarrow$ Manually determine the alert stage of very few alerts
Fine-tune BERT model with a classifier head over B
for each alert a_i in $\{a_1, a_2, \ldots, a_N\}$ **do**
    $F_i \leftarrow \text{EXTRACTFEATURES}(a_i)$
    $S_i \leftarrow \text{BERTCLASSIFIER}(a_i)$ ▷ Alert stage
    $E_i \leftarrow \text{BERTEMBEDDING}([a_{i-5}, \ldots, a_i, \ldots, a_{i+5}])$
end for
Train Isolation Forest on $\{(F_1, S_1), \ldots, (F_N, S_N)\}$
for each alert a_i in $\{a_1, a_2, \ldots, a_N\}$ **do**
    $AN_i \leftarrow \text{ISOLATIONFOREST}(a_i)$ ▷ Anomaly score
end for
Randomly initialize network parameters W
$L \leftarrow 0$ ▷ Initialize the loss
while Accuracy is improving **do**
    for each labeled alert **do**
        $P_i \leftarrow \text{CLASSIFIER}(E_i, F_i, S_i, AN_i)$
        $L \leftarrow L + \text{CrossEntropy}\,(P_i, y_i)$
    end for
    Perform Adam optimizer on W to minimize L
end while

NLP and BERT in CrossAlert

- NLP Role:
  - Extracts semantic similarity between alerts using BERT (Bidirectional Encoder Representations from Transformers).
- Why BERT?
  - Effective in understanding the context and meaning of alert messages, which helps in identifying similarities between multi-stage attacks.
- Prototypical Networks:
  - Address domain shift by generalizing to new domains with limited labeled data

T-SNE for Different Stage of Attack

- BERT, without fine- tuning on cybersecurity task, struggles to distinguish between the different stages of alert messages (Fig 3).
- The clusters for each stage are not well-defined, and there is significant overlap between the stages.
- We manually labeled a small subset of alert messages to determine the alert stage. Then we fine-tuned BERT with a classifier head on this data and obtained embeddings for all the alerts (Fig 4).

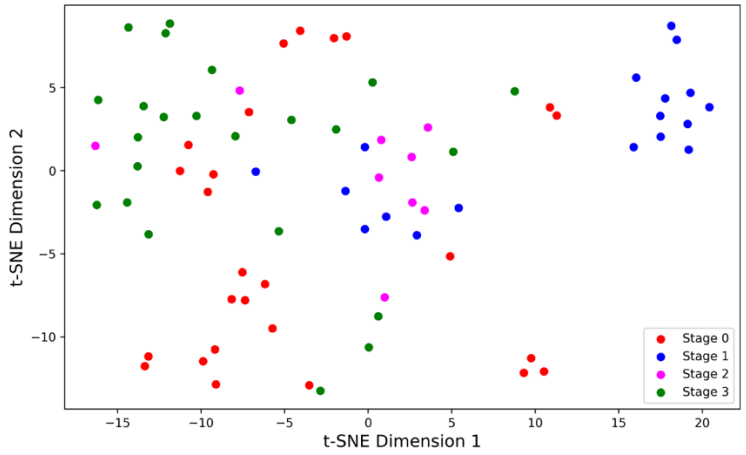


Fig. 3: t-SNE for different stages without prior knowledge.

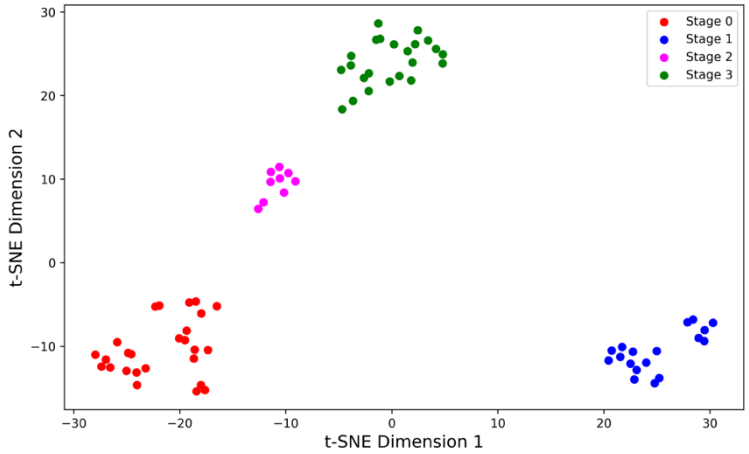


Fig. 4: t-SNE for different stages with prior knowledge.

Similarity between Alert Messages

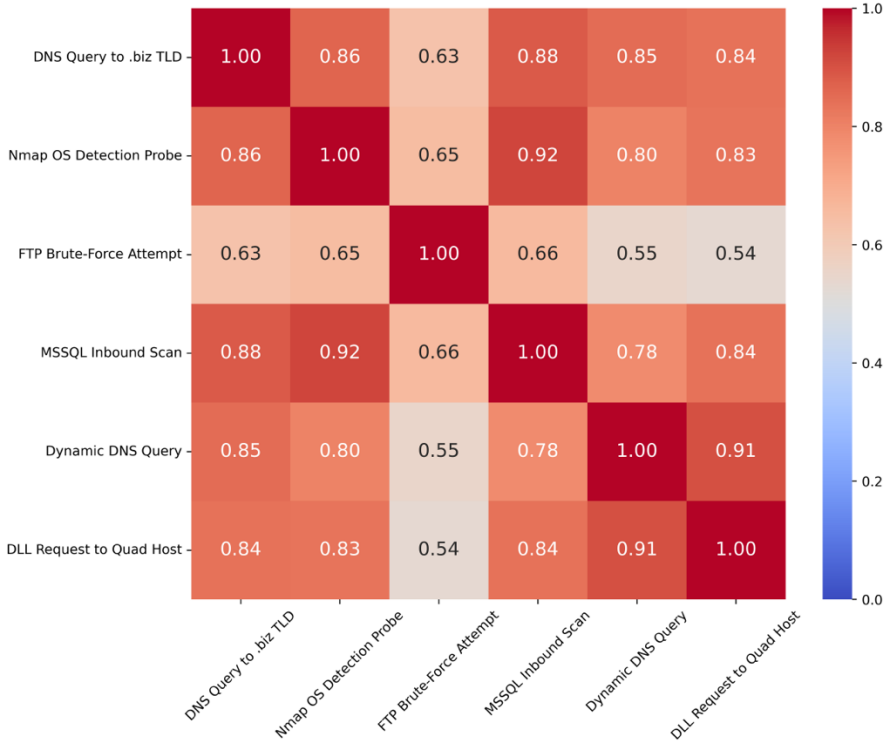


Fig. 5: Similarity heatmap for alert of Scan stage.

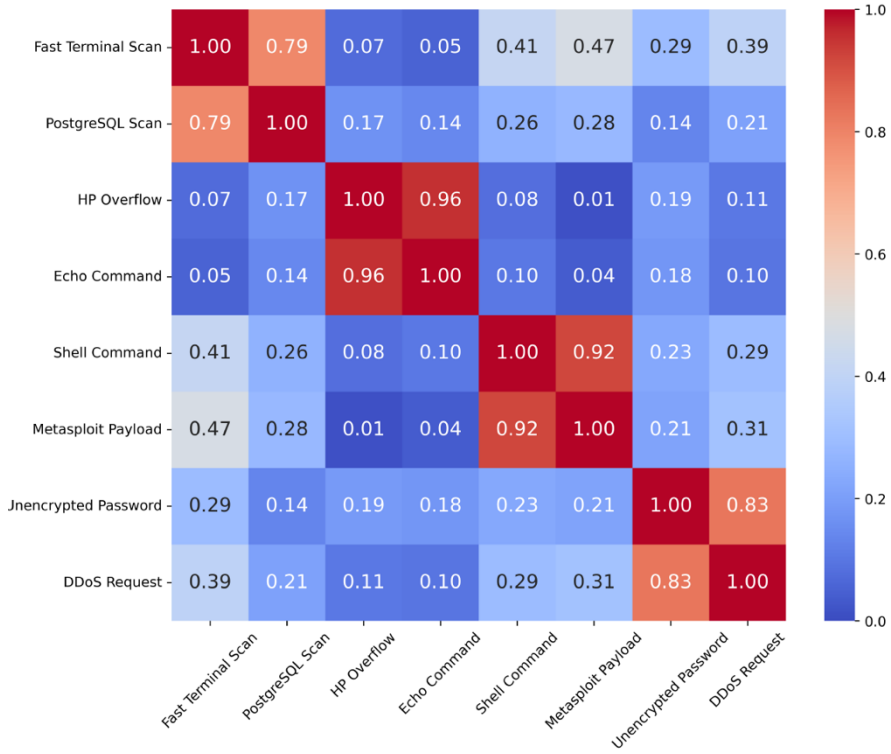


Fig. 6: Similarity heatmap for alerts.

Features Contribution

Integration of basic and sequence-based features significantly improved detection performance, the higher Precision, Recall, and F1-scores across both datasets.

Note that the metric values obtained by isolation forest are still not acceptable, as the isolation forest results in many false predictions.

TABLE IV: Summary of features contribution to initial alert ranking in anomaly detection.

Datasets	Basic Features			Sequence-based Features			Basic and Sequence-based Features		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
DARPA 2000	0.41	0.28	0.34	0.51	0.42	0.46	0.64	0.54	0.58
ISCX 2012	0.33	0.15	0.21	0.44	0.22	0.29	0.51	0.37	0.45

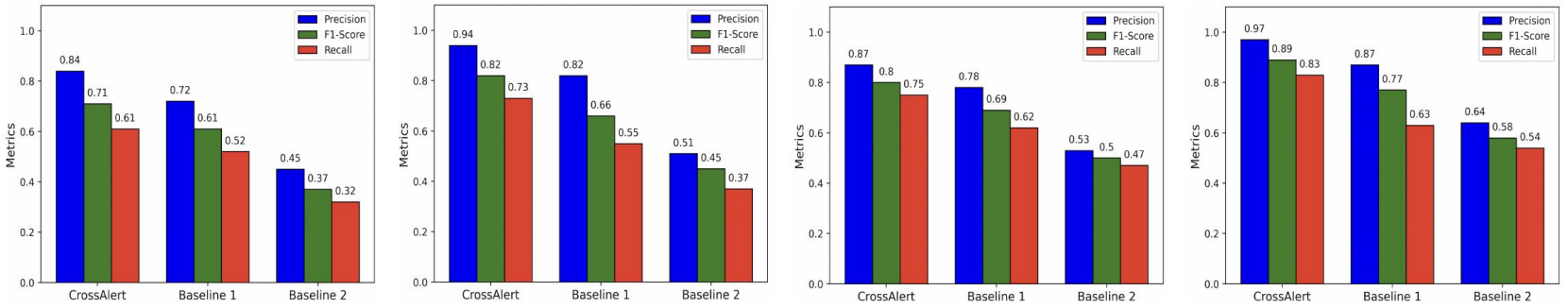
Within Dataset

- Scenario 1: training with 10 MSAs and 100 normal labels
- Scenario 2: training with 20 MSAs and 200 normal labels.

- Baseline1: CrossAlert without BERT embeddings or alert stage, relying on Basic and Sequence-based features with anomaly scores.

- Baseline2: an isolation forest framework.

- Our methodology effectively integrates information from semantic embeddings, Basic and Sequnece-based features, and anomaly scores to outperform the baselines.

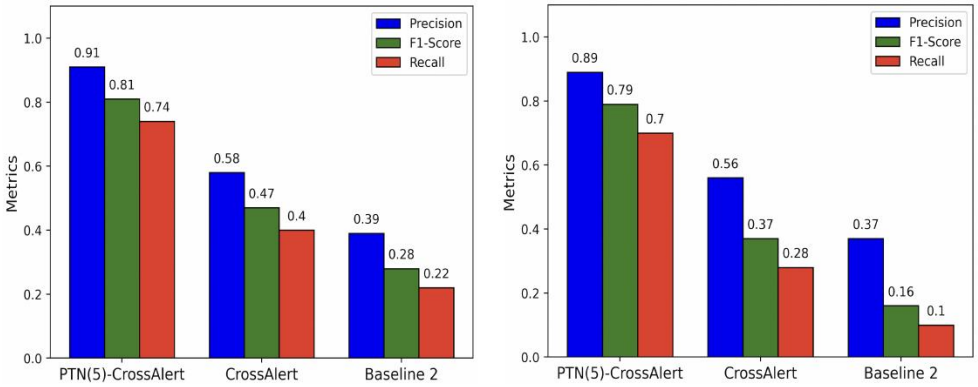


(a) Scenario 1 - ISCX dataset (b) Scenario 2 - ISCX dataset (c) Scenario 1 - DARPA dataset (d) Scenario 2 - DARPA dataset

Fig. 7: Comparing the performance of detection models on different datasets. Scenario 1: training dataset with 10 MSAs and 100 normal labels; Scenario 2: training dataset with 20 MSAs and 200 normal labels.

Domain Shift

- PTN(k)-CrossAlert: the model can only see k MSA and normal samples from the target dataset in addition to the entire source dataset.
- Baseline2: an isolation forest framework.
- Results show that our methodology with the PTN, outperforms the baselines across all metrics in both domain shift scenarios.

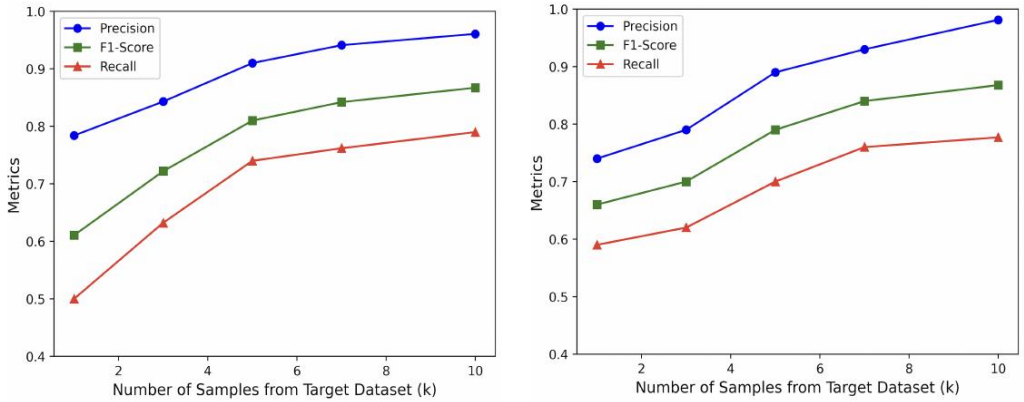


(a) ISCX to DARPA (b) DARPA to ISCX

Fig. 8: Evaluation of detection models' performance in domain shift problem.

Impact of k on PTN-CrossAlert

- We illustrate the performance of our methodology, PTN(k)-CrossAlert, as the value of k varies, allowing the model to see different numbers of samples from the target dataset.
- It can be observed that as the number of available samples from the target dataset increases, the metric values improve.



(a) ISCX to DARPA (b) DARPA to ISCX

Fig. 9: Evaluation of PTN(k)-CrossAlert for different values of k for ISCX to DARPA.

Conclusion

- CrossAlert is a a novel, robust approach for detecting multi-stage attacks using semantic embedding and anomaly detection.

- CrossAlert reduces false positives, enhances multi-stage attack detection, and improves performance across various domains