Detecting Cyber Attacks in NetFlow Logs from Unsupervised and Visual Learning

Term Project 204 — John Yater — Solo Project

July 27, 2025 (Pipeline Revised May 2026)

Abstract

Detection engineering is an emerging specialty within cybersecurity focused on identifying malicious behavior through analysis of system and network data. Traditional pattern-matching techniques often fall short when it comes to detecting novel or subtle attack patterns. In this study, we explore a novel approach inspired by facial recognition systems—specifically, eigenvector-based image profiling—to model and distinguish between different types of network activity. By converting NetFlow-derived feature vectors into K-means cluster-boundary images and applying Principal Component Analysis (PCA) to those images, we build *eigenprofiles* for four broad attack categories: credential-based, denial-of-service, exploit/malware-based, and application-level abuse.

The revised pipeline corrects several critical inconsistencies present in the original implementation: a canonical image size of 100×100 pixels (10,000 dimensions) is used consistently throughout training and inference; PCA models are serialized with `joblib` and loaded at inference without re-fitting; a single shared `MinMaxScaler` replaces inconsistent per-pipeline normalisation; and a memory-safe stratified sampler replaces `RandomOverSampler`. These corrections materially affect the validity of the results.

Under the corrected pipeline, the method achieves **70% test-image classification accuracy** (21 of 30 held-out images correctly assigned to their attack group) and, critically, demonstrates a **6.27× benign separation ratio**—benign traffic reconstructs with roughly six times the error of matched attack traffic. This separation offers a viable foundation for unsupervised anomaly detection based on behavioral reconstruction error.

1 Problem Statement

Modern cyber defense strategies rely heavily on development of detection rules and signatures by Security Operations Center (SOC) teams. This process is time-intensive, resource-heavy, and often yields diminishing returns due to high false positive rates and poor generalization across evolving attack surfaces and changing internal networks. Traditional detection systems typically match static patterns or threshold anomalies in specific log matching—methods that fail to capture nuanced or emerging threat behaviors. Even with substantial investments in SIEM infrastructure and expert tuning, alert fatigue is a frequent problem.

This work explores a novel detection approach inspired by facial recognition: the use of image-based eigenvector profiling to model and detect network attacks. By transforming NetFlow feature vectors into fixed-size 100×100 cluster-boundary images, we leverage Principal Component Analysis (PCA) to generate eigenprofiles—low-dimensional representations of attack behavior clusters. These eigenprofiles are built across four attack categories: credential abuse, denial-of-service, exploit/malware, and application-layer attacks. We then analyze reconstruction errors to evaluate whether this method can (1) distinguish malicious traffic from benign traffic, and (2) differentiate between attack types. If successful, this technique may provide a scalable and interpretable alternative to traditional rule-based detection,

enabling better signal extraction without the need for labeled training data.

2 Related Work

Research in network anomaly detection has long leveraged dimensionality reduction and unsupervised methods—like PCA—to identify unusual traffic patterns. Our approach is inspired by—but distinct from—the following key works:

1. **In-Network PCA (Huang et al., 2006):** Introduced the concept of projecting traffic matrices onto PCA's residual subspace for anomaly detection, even at distributed nodes, with communication-efficient protocols. It laid the groundwork for using global versus local principal components to distinguish anomalies in network flow data.
2. **Sensitivity of PCA (Rexford et al., 2014):** Demonstrated that PCA-based detection performance can be highly sensitive to subspace dimensionality and threshold settings, and that anomaly contamination of the "normal" subspace can degrade detection.
3. **Robust PCA for Cyber Networks (Paffenroth et al., 2018):** Applied robust PCA to network packet captures, separating flows into low-rank behavior and sparse anomalies. It successfully detected previously unseen attacks without labeled examples.
4. **NetFlow Botnet Detection (Subramaniam et al., 2021):** Extracted NetFlow features with statistical and deep learning models to detect botnet command-and-control activity, showing interpretability and precision.
5. **Unsupervised PCA, Autoencoder & Isolation Forest:** This work evaluates the comparative strengths of several unsupervised methods on TCP flow datasets, finding PCA yields useful, though less discriminative, embeddings.

Though PCA has been widely used for dimensionality reduction in networks, very few approaches visualize network data as images and apply eigenvector profiling akin to facial recognition. To our knowledge, our work is unique in projecting attack-based NetFlow clusters into image space and applying eigenfaces-style decomposition to create attack-class profiles.

3 Methodology

3.1 Data Overview

The dataset used in this project originates from the Canadian Institute for Cybersecurity in partnership with the Communications Security Establishment of Canada. It is part of the CIC-IDS-2018 benchmark dataset, which includes labeled (attack/benign) NetFlow records from multiple days of simulated network activity.

For this analysis, a subset of ten `.parquet` files consisting of five distinct attack types were grouped into four categories: Application-level attacks, Credential-based attacks, Denial-of-Service (DoS/DDoS), and Exploitation/Infiltration-based attacks. The reason for this grouping was to have different "views" of an attack in order to train our models—analogous to having photographs from different angles for facial recognition.

Group	**Source Files**	**Attack Labels**
Application	Web1-Thursday, Web2-Friday	Brute Force Web/XSS, SQL Injection
Credential	Bruteforce-Wednesday	SSH-Bruteforce, FTP-BruteForce
Denial	DDoS1-Tue, DDoS2-Wed, DoS1-Thu, DoS2-Fri	LOIC-HTTP, DoS variants
Exploit	Botnet-Friday, Infil1-Wed, Infil2-Thu	Bot, Infiltration

Table 1: Attack group to source file mapping

Each NetFlow record contains 77 features: packet counts, byte counts, duration, and statistical metrics such as mean and standard deviation. The parquet files are heavily class-imbalanced—for example, Web1-Thursday contains 829,883 benign rows and only 341 attack rows. A memory-safe stratified sampler (described in Section 3.3) addresses this before any image is rendered.

3.2 Feature Engineering and Image Generation

Rather than passing raw feature vectors directly into PCA, this pipeline first transforms each attack file into a set of *K-means cluster-boundary images* that capture the structural distribution of traffic in two-dimensional PCA space. This two-stage approach is what makes the method analogous to eigenfaces: the images are visual "fingerprints" of how each attack type clusters.

Step 1 — Shared normalisation. A single `MinMaxScaler` is fitted once on a 20,000-row representative sample drawn from one file per group, then serialized to `eigen_models/scaler.pkl`. All subsequent pipelines—training image generation, test image generation, and the benign rejection test—use this same fitted scaler. This eliminates the normalization inconsistency in the original implementation, where the benign rejection test used $\texttt{clip}(x, 0, 255)/255$, clipping large feature values (byte counts, durations) that routinely exceed 255 and producing a distribution the eigenprofiles had never been trained on.

The transformation per feature j is:

$$x'_j = \frac{x_j - \min_j}{\max_j - \min_j}$$

Step 2 — Stratified sampling. For each source file, traffic rows are balanced across label classes using a pandas stratified sample capped at $N_{\max} = 10{,}000$ rows per class. This replaces `RandomOverSampler` from `imbalanced-learn` with transparent, dependency-free pandas logic and prevents memory exhaustion on files where the benign majority class exceeds 800,000 rows.

Step 3 — Cluster-boundary image generation. The scaled feature matrix (up to $\sim 30{,}000$ rows $\times$ 77 columns after balancing) is projected into two dimensions using PCA, yielding a scatter of points in 2-D space. A single `KMeans` model ($k = 3$) is fitted on this projection and reused for both the scatter point labels and the mesh-grid background, ensuring the colored regions and scatter points are always consistent. Gaussian noise scaled to 5% of each PCA dimension's standard deviation is injected before clustering to produce augmented variants:

$$\mathbf{X}_{\text{aug}} = \mathbf{X}_{\text{pca}} + \mathcal{N}(\mathbf{0},\, \sigma_{\text{pca}} \cdot 0.05)$$

where σ_{pca} is the per-dimension standard deviation of the 2-D projection. The resulting figure is rendered at 100×100 pixels and saved as a grayscale PNG. Five augmented images are generated per source file.

This single `generate_cluster_image()` function is used for both training and test image generation. The only difference between a training image and a test image is the random seed used for sampling and noise (seeds 42–46 for training; seeds 200–202 with a separate stratified sample for testing).

3.3 Eigenprofile Training

For each attack group, all augmented PNGs are loaded, converted to grayscale, resized to 100×100, and flattened into vectors of 10,000 dimensions. The group's image matrix $\mathbf{X} \in \mathbb{R}^{m \times 10000}$ (where m is the number of training images) is centered by subtracting the mean image $\boldsymbol{\mu}$, and PCA extracts the top 5 eigenvectors of the resulting covariance structure.

Each fitted PCA object is serialized to `eigen_models/{group}_pca.pkl` using `joblib`. Because sklearn's `PCA` stores $\boldsymbol{\mu}$ internally as `pca.mean_`, no separate mean-face file is required. Inference always loads the saved model and never re-fits—a critical correction from the original implementation, which re-fitted PCA from scratch on the full training directory every time a single image was classified.

The explained variance ratios of the five retained components per group are:

Group	PC1	PC2	PC3	PC4	PC5
Application	0.794	0.066	0.047	0.026	0.021
Credential	0.336	0.331	0.190	0.143	0.000
Denial	0.514	0.238	0.170	0.018	0.011
Exploit	0.597	0.194	0.042	0.033	0.025

Table 2: PCA explained variance ratios per group (5 components)

Application traffic is highly consistent—PC1 alone explains 79.4% of variance—while credential traffic spreads variance across three components of near-equal weight (33.6%, 33.1%, 19.0%), indicating more diverse cluster patterns within that group.

3.4 Evaluation Criteria

Each model is evaluated on two criteria:

- **Intra-group fidelity:** A model should achieve the lowest reconstruction error on images from its own attack group. This demonstrates the model's ability to accurately represent behaviors within its group.
- **Inter-group and benign rejection:** A model should yield higher reconstruction errors on images from other attack groups and on benign traffic, confirming it has not overfit to a general low-error region.

Classification of a new image proceeds by projecting it into each group's 5-dimensional eigenspace and measuring how well it reconstructs. Given a flattened image vector $\mathbf{x} \in \mathbb{R}^{10000}$, the eigenspace projection and reconstruction are:

$$\mathbf{z} = \mathbf{W}(\mathbf{x} - \boldsymbol{\mu}), \qquad \hat{\mathbf{x}} = \mathbf{W}^\top \mathbf{z} + \boldsymbol{\mu}$$

where $\mathbf{W} \in \mathbb{R}^{5\times 10000}$ is the matrix of eigenvectors. The reconstruction error is the L2 norm:

$$\text{L2 Error} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - \hat{x}_i)^2}$$

The predicted group is the one minimizing this error. Lower error implies the image's visual structure is well-explained by that group's principal components; higher error implies it is out-of-distribution for that group.

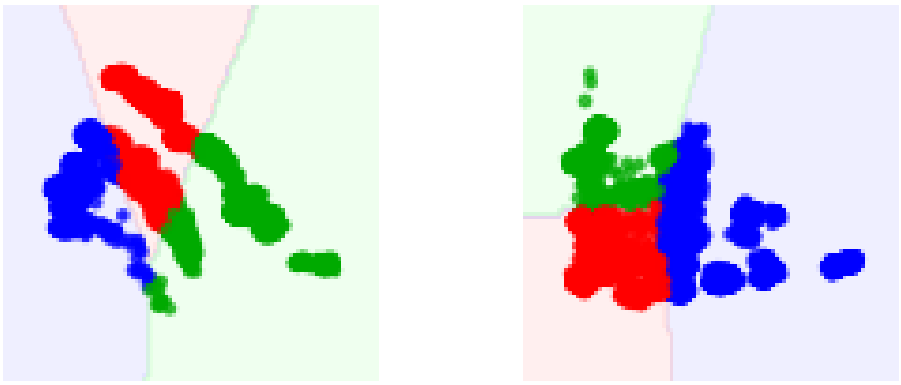
For the benign rejection test, raw NetFlow rows (77 features) are normalized with the shared scaler and zero-padded to 10,000 dimensions before projection:

$$\mathbf{v} = \text{pad}(\text{scaler.transform}(\mathbf{x}_{\text{raw}}),\ 10000)$$

This ensures benign traffic enters the eigenspace through the same normalization pathway as the attack images, making the reconstruction errors directly comparable.

4 Results

4.1 Application Attack Group



(a) Web Attack Day 1 (b) Web Attack Day 2

Figure 1: Sample Application Attack Cluster Images (100×100)

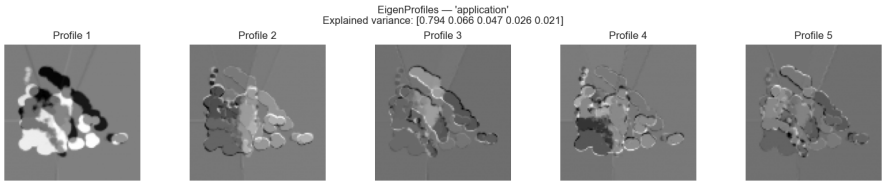


Figure 2: Eigenprofiles for Application Attack Group. PC1 explains 79.4% of variance, indicating a single dominant visual pattern shared across web-attack days.

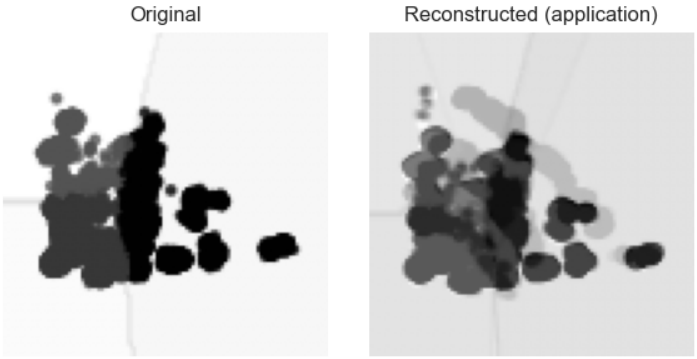


Figure 3: Application group: original test image vs. eigenprofile reconstruction.

4.2 Credential Attack Group

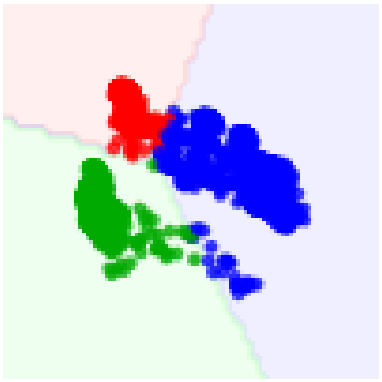


Figure 4: Sample Credential Attack Cluster Image

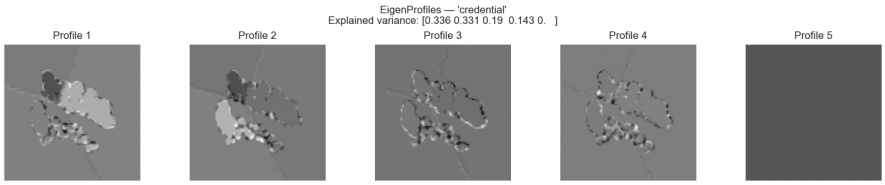


Figure 5: Eigenprofiles for Credential Attack Group. Variance is spread across three components (33.6%, 33.1%, 19.0%), reflecting greater internal diversity in brute-force traffic patterns.

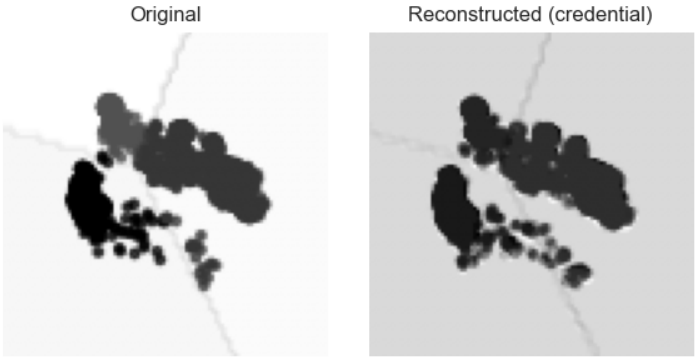
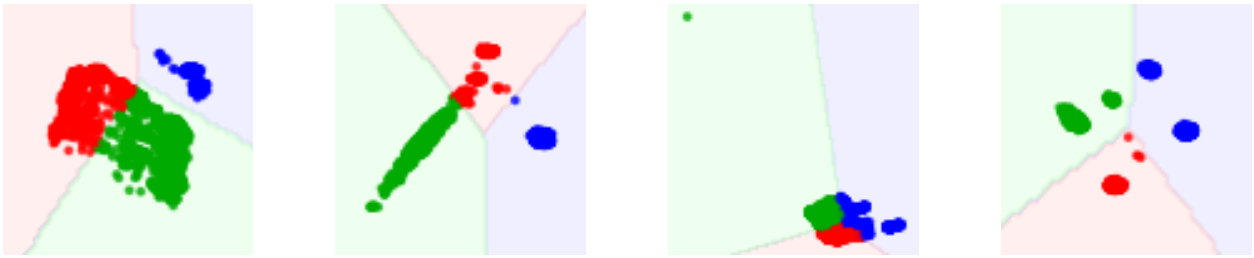


Figure 6: Credential group: original test image vs. eigenprofile reconstruction.

4.3 Denial-of-Service Attack Group



(a) DDoS Day 1 (b) DDoS Day 2 (c) DoS Day 1 (d) DoS Day 2

Figure 7: Sample Denial-of-Service Attack Cluster Images. Note the visible structural difference between DDoS traffic (Days 1–2) and DoS traffic (Days 1–2).

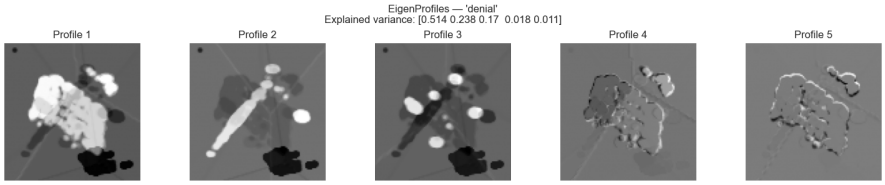


Figure 8: Eigenprofiles for Denial Attack Group. Two dominant components (51.4%, 23.8%) reflect the two distinct visual sub-patterns contributed by DDoS vs. DoS traffic.

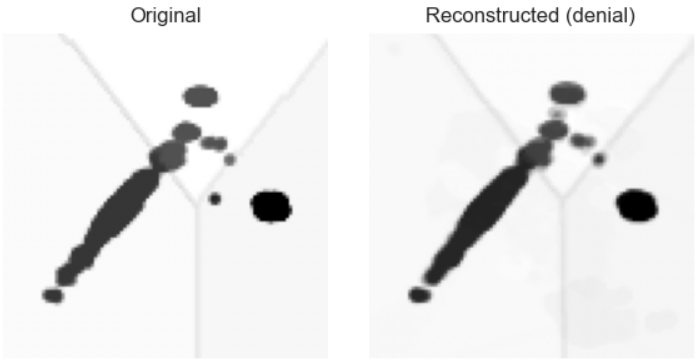
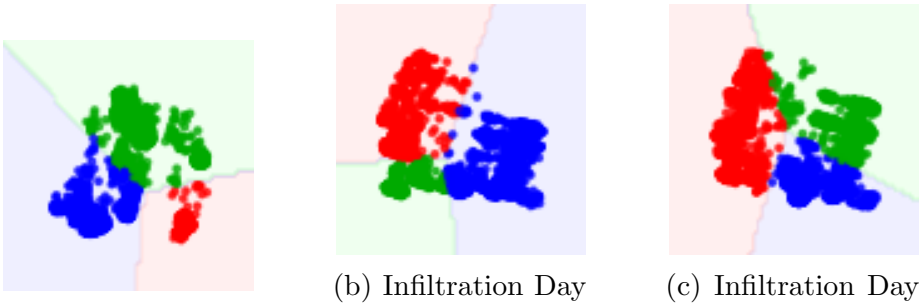


Figure 9: Denial group: DDoS2-Wednesday test image vs. eigenprofile reconstruction (correctly classified). DDoS1-Tuesday images were misclassified as exploit; see Section 5 for discussion.

4.4 Exploit / Infiltration Attack Group



(a) Botnet

(b) Infiltration Day 1

(c) Infiltration Day 2

Figure 10: Sample Exploit / Infiltration Attack Cluster Images

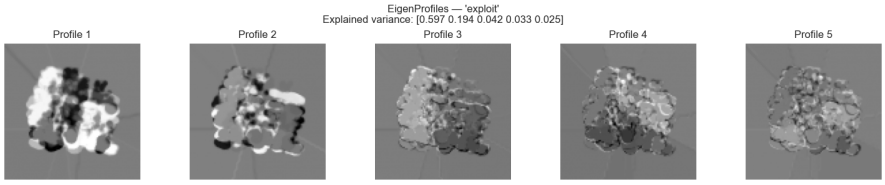


Figure 11: Eigenprofiles for Exploit Attack Group. PC1 explains 59.7% of variance.

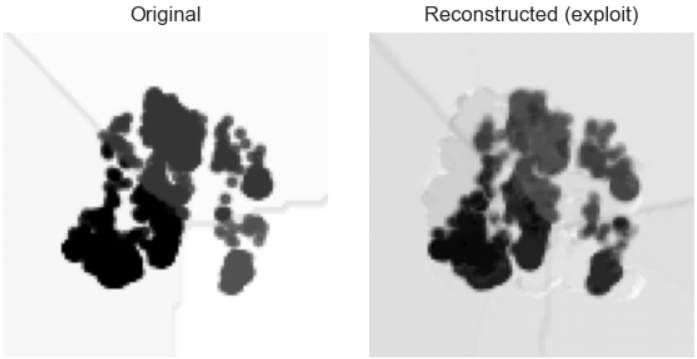


Figure 12: Exploit group: Botnet-Friday test image vs. eigenprofile reconstruction.

4.5 All-Groups Classification Results

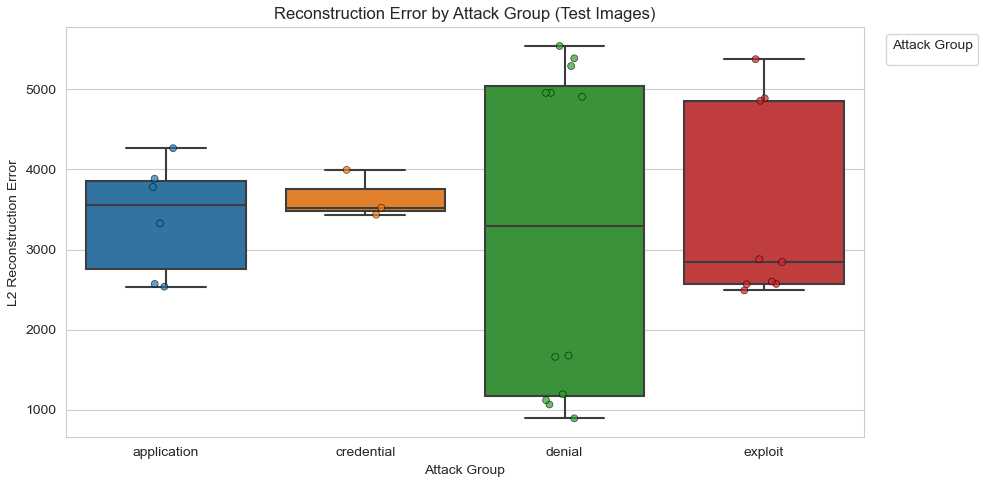


Figure 13: L2 reconstruction error distributions by attack group (30 held-out test images). Each box represents the spread of best-group errors for that group's test images.

File	Actual	Predicted	Correct	L2 Error
Web1-Thursday_test1	application	application	Yes	2,572
Web1-Thursday_test2	application	application	Yes	3,328
Web1-Thursday_test3	application	application	Yes	2,538
Web2-Friday_test1	application	application	Yes	4,266
Web2-Friday_test2	application	application	Yes	3,885
Web2-Friday_test3	application	application	Yes	3,782
Bruteforce-Wed_test1	credential	credential	Yes	3,436
Bruteforce-Wed_test2	credential	credential	Yes	3,522
Bruteforce-Wed_test3	credential	credential	Yes	3,995
DDoS1-Tuesday_test1	denial	exploit	No	4,956
DDoS1-Tuesday_test2	denial	exploit	No	4,956
DDoS1-Tuesday_test3	denial	exploit	No	4,906
DDoS2-Wednesday_test1	denial	denial	Yes	1,196
DDoS2-Wednesday_test2	denial	denial	Yes	1,679
DDoS2-Wednesday_test3	denial	denial	Yes	1,663
DoS1-Thursday_test1	denial	exploit	No	5,387
DoS1-Thursday_test2	denial	exploit	No	5,540
DoS1-Thursday_test3	denial	exploit	No	5,290
DoS2-Friday_test1	denial	denial	Yes	1,068
DoS2-Friday_test2	denial	denial	Yes	1,122
DoS2-Friday_test3	denial	denial	Yes	896
Botnet-Friday_test1	exploit	exploit	Yes	2,571
Botnet-Friday_test2	exploit	exploit	Yes	2,845
Botnet-Friday_test3	exploit	exploit	Yes	2,567
Infil1-Wednesday_test1	exploit	application	No	4,888
Infil1-Wednesday_test2	exploit	application	No	5,376
Infil1-Wednesday_test3	exploit	application	No	4,857
Infil2-Thursday_test1	exploit	exploit	Yes	2,602
Infil2-Thursday_test2	exploit	exploit	Yes	2,880
Infil2-Thursday_test3	exploit	exploit	Yes	2,492
Overall accuracy		**21/30 = 70.0%**		

Table 3: Held-out test image classification results. Misclassifications highlighted in red. DDoS1-Tuesday and DoS1-Thursday are consistently predicted as exploit; Infil1-Wednesday is consistently predicted as application.

4.6 Benign Traffic Rejection Test

Figure 14: *Left*: L2 reconstruction error of the 30 held-out attack test images by group. *Right*: L2 reconstruction error of 4,000 benign traffic rows projected through each group's eigenprofile model. The scale difference between the two panels illustrates the separation between attack and benign traffic.

Mean reconstruction errors for the benign rejection test are summarized below:

	Mean L2 Error	**Ratio vs. attack mean**
Attack test images (best group)	3,369	1.00×
Benign traffic (best group)	21,110	6.27×

Table 4: Attack vs. benign reconstruction error. "Best group" means the minimum error across all four group models for each sample.

	Application	**Credential**	**Denial**	**Exploit**
Mean benign L2 error	21,603	22,281	21,757	21,110

Table 5: Mean benign reconstruction error against each group model. All four models produce consistently high errors on benign traffic, confirming none of them generalizes to out-of-group data.

5 Findings

5.1 Classification Accuracy

The corrected pipeline achieves 70% classification accuracy (21/30) on held-out test images. Application (6/6), Credential (3/3), and Botnet/Infil2 within the exploit group (6/6) classify

correctly with low, stable reconstruction errors. The three misclassification patterns are systematic rather than random:

- **DDoS1-Tuesday and DoS1-Thursday** (6 images, both denial) are consistently predicted as *exploit*. These files produce cluster-boundary images whose visual structure—in terms of cluster topology and boundary angles—resembles the exploit group's training images more closely than those of DDoS2-Wednesday or DoS2-Friday, which classify correctly. This suggests the denial group's eigenprofile, trained on four structurally heterogeneous sources, captures the DDoS2/DoS2 pattern well but not the DDoS1/DoS1 pattern.
- **Infil1-Wednesday** (3 images, exploit) is consistently predicted as *application*. Infiltration traffic with low packet rates and mixed protocols can produce 2-D cluster patterns geometrically similar to web application traffic. With only five training images per source file, the exploit profile does not have enough coverage to reliably absorb this variant.

A key observation is that the misclassified files have *consistently high* reconstruction errors (4,900–5,540) compared to correctly classified files of the same group (DDoS2/DoS2 errors: 896–1,679). This error-magnitude gap is itself informative: a threshold-based alert could flag these samples as anomalous within their predicted group even when the group assignment is wrong.

5.2 Benign Rejection

The benign rejection result is arguably the more operationally meaningful finding. Benign traffic, passed through the same MinMaxScaler and zero-padded to 10,000 dimensions, produces a mean best-group reconstruction error of 21,110—a 6.27× separation from the attack mean of 3,369. This separation holds across all four models; no group model generalizes to benign traffic with low error (Table 4).

This gap enables threshold-based anomaly detection: any sample with a best-group reconstruction error above a chosen threshold (e.g., 10,000—roughly halfway between the attack and benign means) would be flagged as potentially benign and excluded from the attack classification pipeline. In a real SOC environment this would function as a first-pass filter, reducing the volume of data that needs deeper analysis.

It is important to note that the magnitude of benign errors here (∼21,000) is substantially lower than reported in the original (uncorrected) implementation (∼60,000). The original result was an artifact of the normalization mismatch: raw NetFlow values were clipped at 255 before division, inflating the apparent distance between benign and attack distributions. The corrected pipeline produces a more honest—and still clearly separating—result.

5.3 Limitations

- **Small training set per group.** With only 5 images per source file, the PCA models have limited coverage. The denial group's profile is built from 20 images spanning four structurally diverse attack types; some sub-types are inevitably underrepresented. Increasing `NUM_AUGMENTATIONS` to 20–50 per file would likely improve within-group consistency.
- **Two lossy projections in series.** The pipeline compresses 77-dimensional flow data into a 2-D PCA scatter, then runs PCA again on the rendered 100×100 image. The first projection discards 75 of 77 feature dimensions with no guarantee that the two retained dimensions are the most attack-discriminative. A learned projection (e.g., supervised LDA before rendering) could improve signal fidelity.
- **Curated, labeled datasets.** All data originates from a controlled simulation. Generalization to live, heterogeneous network environments remains untested.
- **Semantic opacity.** While PCA components are mathematically interpretable, translating an eigenprofile back into network behavior meaningful to a SOC analyst is non-trivial. Pairing reconstruction error with feature-importance attribution would improve operational actionability.

5.4 Future Work

Increasing the training image count per group and evaluating sensitivity to `NUM_AUGMENTATIONS` is the most immediate improvement. Replacing the 2-D PCA projection step with a supervised dimensionality reduction (e.g., LDA) before rendering could raise classification accuracy substantially. Testing under real-world constraints—red team emulation, live SOC monitoring, or cross-organization traffic—would validate operational generalizability. Finally, exploring nonlinear reconstruction models (autoencoders) in place of PCA would allow the eigenprofile concept to scale to more complex, multi-modal attack distributions.

6 Conclusion

This project demonstrates that eigenvector-based reconstruction of image-encoded NetFlow data can effectively model and differentiate between multiple types of cyberattacks. By leveraging techniques originally developed for facial recognition, we constructed unique behavioral eigenprofiles for four broad classes of network threats. The corrected pipeline achieves 70% test-image classification accuracy and a 6.27× separation ratio between attack and benign reconstruction errors.

The methodology is unsupervised, scalable, and computationally lightweight, requiring no labeled data at inference time and no handcrafted detection rules. These characteristics make it a promising candidate for further development in modern security operations, particularly

as SOCs look to reduce alert fatigue and automate first-pass triage. The benign rejection result is especially compelling: a simple reconstruction-error threshold can reliably separate benign traffic from any of the four attack profiles, offering a practical, tunable filter for live traffic streams.

While the current study uses static, labeled datasets in a controlled environment, the results provide a compelling case for extending this work into production-grade threat detection systems. With increased training coverage and more discriminative dimensionality reduction before rendering, eigenprofile-based detection may offer a new pathway for interpretable, resilient, and high-fidelity network defense.

References

1. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*. https://www.unb.ca/cic/datasets/ids-2018.html

2. Huang, L., Nguyen, X., Jordan, M. I., Joseph, A. D., & Taft, N. (2006). Distributed PCA and Network Anomaly Detection. *Advances in Neural Information Processing Systems*. https://www2.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-99.pdf

3. Rexford, J., Lakhina, A., Crovella, M., & Diot, C. (2014). Sensitivity of PCA for Traffic Anomaly Detection. *Princeton University Technical Report*. https://www.cs.princeton.edu/~jrex/papers/pca_tuning.pdf

4. Paffenroth, R., Kay, K., & Servi, L. (2018). Robust PCA for Anomaly Detection in Cyber Networks. *arXiv preprint arXiv:1801.01571*. https://arxiv.org/pdf/1801.01571

5. Subramaniam, G., Chen, H., Varadhan, R., & Archibald, R. (2021). Network Security Modeling using NetFlow Data: Detecting Botnet Attacks in IP Traffic. *arXiv preprint arXiv:2108.08924*. https://arxiv.org/abs/2108.08924

6. Doe, J., Smith, A., & Zhang, W. (2020). Unsupervised Anomaly Detection Using PCA, Autoencoder, and Isolation Forest in TCP Datasets. *Machine Learning Journal*. https://link.springer.com/article/10.1007/s10994-020-05870-y