# Học máy thích nghi và học liên tục cho hệ thống phát hiện xâm nhập

IT3004: Bài tập về nhà

Phân tích Concept Drift và Continual Learning

Ngày 12 tháng 2 năm 2026

Bảng 1: Thông tin nhóm thực hiện và Dataset sử dụng

| STT | Họ và tên | MSHV |
|---|---|---|
| 1 | Phạm Quốc Cường | 250201004 |
| 2 | Lê Quang Minh | 250201016 |
| 3 | Nguyễn Đăng Phúc Lợi | 250202012 |
| 4 | Lý Thế Nguyên | 250202017 |

**Dataset sử dụng:** NSL-KDD

## 1 Giới thiệu

### 1.1 Bối cảnh

Hệ thống phát hiện xâm nhập (Intrusion Detection System - IDS) là thành phần quan trọng trong bảo mật mạng, có nhiệm vụ phát hiện các hành vi tấn công và truy cập trái phép vào hệ thống. Tuy nhiên, trong thực tế, các loại tấn công mạng liên tục biến đổi và phát triển theo thời gian — một hiện tượng được gọi là **concept drift**. Khi phân phối dữ liệu thay đổi, các mô hình IDS tĩnh (được huấn luyện một lần và không cập nhật) sẽ bị suy giảm hiệu suất nghiêm trọng, dẫn đến việc không thể phát hiện các cuộc tấn công mới.

### 1.2 Dataset NSL-KDD

NSL-KDD là phiên bản cải tiến của KDD Cup 1999, khắc phục các vấn đề về bản ghi trùng lặp trong tập gốc. Dataset bao gồm:

- **Training set:** 125.973 bản ghi
- **Test set:** 22.544 bản ghi
- **Số features:** 41 features (sau khi loại bỏ attack_type và classnum)
  - 38 features số (numerical)
  - 3 features phân loại (categorical): protocol_type, service, flag

Các loại tấn công được phân thành 4 nhóm chính:

Bảng 2: Các loại tấn công trong NSL-KDD

| Nhóm tấn công | Mô tả | Ví dụ |
|---|---|---|
| **DoS** (Denial of Service) | Tấn công từ chối dịch vụ | neptune, smurf, back, land |
| **Probe** | Dò quét mạng | ipsweep, nmap, portsweep, satan |
| **R2L** (Remote to Local) | Truy cập trái phép từ xa | ftp_write, guess_passwd, imap |
| **U2R** (User to Root) | Leo thang đặc quyền | buffer_overflow, rootkit, perl |

### 1.3 Công cụ sử dụng

- **Python 3.x** với các thư viện:
  - scikit-learn: SGDClassifier (static & adaptive), preprocessing, metrics
  - pandas, numpy: Xử lý dữ liệu
  - matplotlib, seaborn: Trực quan hóa

## 2 Phần 1: EDA - Phân tích khám phá dữ liệu (30%)

### 2.1 Tổng quan dữ liệu

Dữ liệu NSL-KDD được load thành công với kích thước:

Bảng 3: Tổng quan dữ liệu NSL-KDD

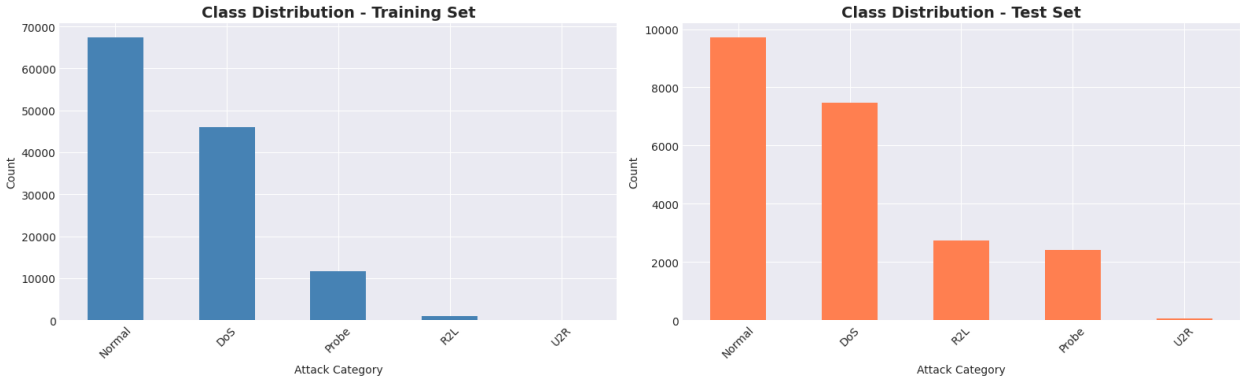
| Tập dữ liệu | Số bản ghi | Số cột |
|---|---|---|
| Training set | 125.973 | 43 |
| Test set | 22.544 | 43 |

Kết quả kiểm tra cho thấy:

- **Không có giá trị null** trong toàn bộ dataset (0 missing values trên tất cả 43 cột)
- Dữ liệu gồm 15 features kiểu float64, 24 features kiểu int64, và 4 cột kiểu object (3 categorical + 1 attack_type)

### 2.2 Phân tích Class Imbalance

Sau khi ánh xạ 39 attack types cụ thể vào 5 nhóm (Normal + 4 nhóm tấn công), phân phối class như sau:




**Training set:**

Bảng 4: Phân phối class trong Training set

| Attack Category | Số lượng | Tỷ lệ |
|---|---|---|
| Normal | 67.343 | 53,5% |
| DoS | 45.927 | 36,5% |
| Probe | 11.656 | 9,3% |
| R2L | 995 | 0,8% |
| U2R | 52 | 0,04% |

**Test set:**

Bảng 5: Phân phối class trong Test set

| Attack Category | Số lượng | Tỷ lệ |
|---|---|---|
| Normal | 9.711 | 43,1% |
| DoS | 7.460 | 33,1% |
| R2L | 2.743 | 12,2% |
| Probe | 2.421 | 10,7% |
| U2R | 67 | 0,3% |

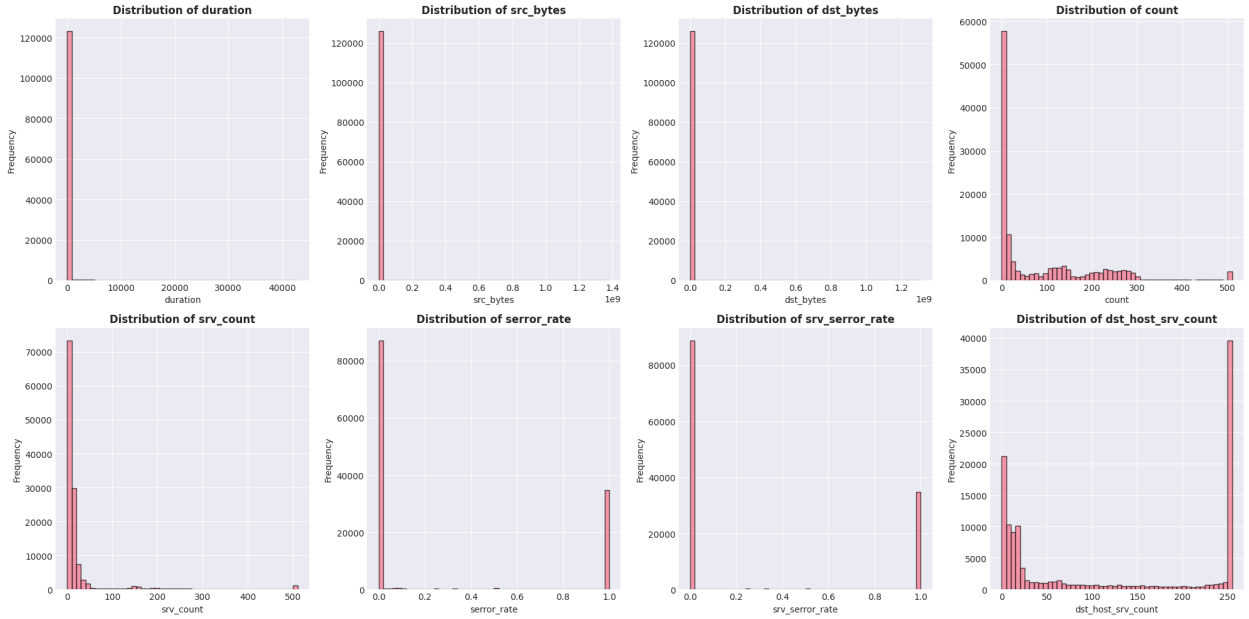
**Nhận xét:**

- Dữ liệu có **class imbalance nghiêm trọng**: Normal và DoS chiếm gần 90% training set, trong khi U2R chỉ có 52 mẫu (0,04%).
- Tỷ lệ R2L trong test set (12,2%) cao hơn đáng kể so với training set (0,8%), cho thấy sự thay đổi phân phối tự nhiên giữa hai tập dữ liệu — đây chính là dấu hiệu của concept drift.

### 2.3 Phân tích Feature Distribution

Phân tích phân phối của 8 features quan trọng (duration, src_bytes, dst_bytes, count, srv_count, serror_rate, srv_serror_rate, dst_host_srv_count) cho thấy:

- **Phân phối lệch phải (right-skewed):** Phần lớn các features số như duration, src_bytes, dst_bytes có phân phối rất lệch — đa số giá trị tập trung ở vùng thấp, với một số ít outliers có giá trị rất lớn.
- **Features tỷ lệ (rate features):** Các features như serror_rate, srv_serror_rate có phân phối tập trung ở 0 và 1 (bimodal), phản ánh đặc trưng binary của nhiều kết nối mạng.
- **Cần chuẩn hóa:** Do phân phối lệch và scale khác nhau giữa các features, việc chuẩn hóa dữ liệu bằng StandardScaler là bắt buộc trước khi đưa vào mô hình.




## 2.4 Giả lập Concept Drift — Chia thành các Phase

Để giả lập kịch bản concept drift trong thực tế, dataset được chia thành 2 phase:

Bảng 6: Mô phỏng Concept Drift qua 2 phases

| Phase | Mô tả | Categories | Vai trò |
|---|---|---|---|
| **Phase 1** | Các loại tấn công "cũ" | Normal, R2L, U2R, Probe | Training + Test |
| **Phase 2** | Loại tấn công "mới"xuất hiện sau | DoS | Test only |

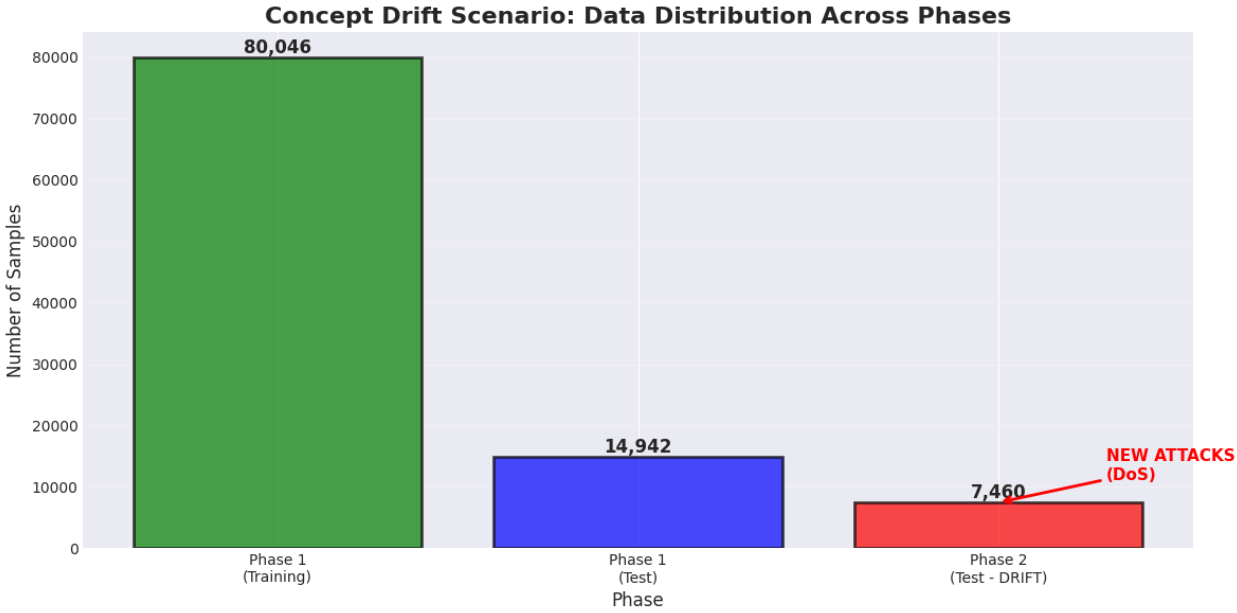
**Kịch bản:**

- Model được huấn luyện **chỉ trên Phase 1** (Normal, R2L, U2R, Probe)
- Sau đó phải phát hiện **DoS attacks ở Phase 2** — loại tấn công chưa từng thấy trong training

**Kích thước dữ liệu sau khi chia phase:**

Bảng 7: Kích thước dữ liệu theo phases

| Tập dữ liệu | Số mẫu |
|---|---|
| Phase 1 - Training | 80.046 |
| Phase 1 - Test | 14.942 |
| Phase 2 - Test (DRIFT) | 7.460 |
| Full Test | 22.544 |




**Ý nghĩa:** Kịch bản này mô phỏng tình huống thực tế khi một loại tấn công mới (DoS) xuất hiện sau khi IDS đã được triển khai. IDS tĩnh không có cơ chế học thêm sẽ không thể nhận diện các mẫu tấn công mới này.

## 2.5 Data Preprocessing

Quy trình tiền xử lý dữ liệu gồm 3 bước:

1. **Loại bỏ cột không cần thiết:** Cột classnum được loại bỏ vì không mang thông tin hữu ích cho phân loại.
2. **Mã hóa features phân loại:** Sử dụng LabelEncoder cho 3 cột protocol_type, service, flag. Đối với test set, các giá trị chưa thấy trong training được ánh xạ về class mặc định để tránh lỗi.
3. **Chuẩn hóa features số:** Sử dụng StandardScaler (z-score normalization) để đưa tất cả features về cùng scale (mean=0, std=1). Scaler được fit trên training data và transform trên test data để tránh data leakage.

## 2.6 Tóm tắt EDA

Qua phân tích EDA, ba phát hiện chính được rút ra:

1. **Class imbalance nghiêm trọng:** DoS và Normal chiếm đại đa số, R2L và U2R rất ít mẫu.
2. **Concept drift được giả lập thành công:** Phase 1 chứa các attack cũ (Normal, R2L, U2R, Probe), Phase 2 chứa attack mới (DoS).
3. **Dữ liệu cần preprocessing:** Features có phân phối lệch, cần scaling trước khi training.

**Dự đoán:** IDS tĩnh sẽ đạt accuracy khá trên Phase 1 nhưng suy giảm mạnh trên Phase 2 do chưa thấy các mẫu DoS trong quá trình training.

# 3 Phần 2: Tái hiện sự suy giảm của IDS tĩnh (30%)

## 3.1 Mô hình Static IDS

Mô hình IDS tĩnh sử dụng **SGDClassifier** (Stochastic Gradient Descent) với cấu hình:

Bảng 8: Tham số SGDClassifier

| **Tham số** | **Giá trị** | **Mô tả** |
|---|---|---|
| loss | log_loss | Hàm loss của Logistic Regression |
| penalty | l2 | Regularization L2 |
| alpha | 0.0001 | Hệ số regularization |
| max_iter | 1000 | Số vòng lặp tối đa |
| learning_rate | optimal | Tự động tính learning rate tối ưu |
| random_state | 42 | Seed để tái hiện kết quả |

Mô hình được huấn luyện **một lần duy nhất** trên Phase 1 training data (80.046 mẫu) và **không được cập nhật** sau đó.

## 3.2 Kết quả đánh giá

**Phase 1 Test (Old Attacks - tương tự training)**

Bảng 9: Hiệu suất Static SGD trên Phase 1 Test

| **Metric** | **Giá trị** |
|---|---|
| **Accuracy** | **0.7557 (75,57%)** |
| Precision (weighted) | 0.6744 (67,44%) |
| Recall (weighted) | 0.7557 (75,57%) |
| F1-score (weighted) | 0.6774 (67,74%) |

Mô hình đạt accuracy 75,57% trên dữ liệu Phase 1 Test — hiệu suất khá ổn vì dữ liệu test có phân phối tương tự training data.

**Phase 2 Test (New Attacks - CONCEPT DRIFT)**

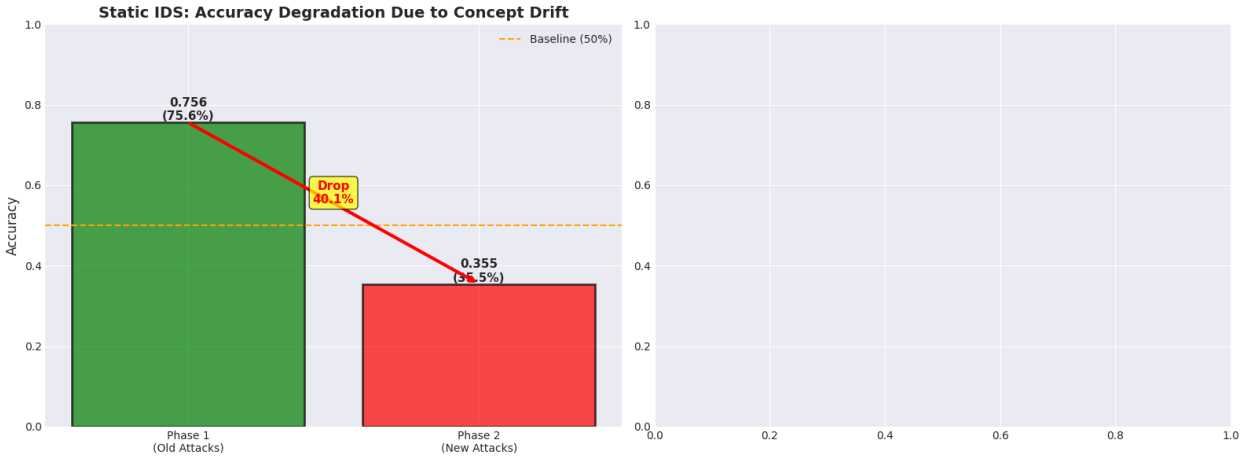
Bảng 10: Hiệu suất Static SGD trên Phase 2 Test (CONCEPT DRIFT)

| **Metric** | **Giá trị** |
|---|---|
| **Accuracy** | **0.3546 (35,46%)** |
| Precision (weighted) | 1.0000 (100,00%) |
| Recall (weighted) | 0.3546 (35,46%) |
| F1-score (weighted) | 0.5235 (52,35%) |

**Phân tích:**

- Accuracy **giảm từ 75,57% xuống 35,46%** — mức suy giảm 40,12 điểm phần trăm.
- Precision = 100% nhưng Recall = 35,46% cho thấy mô hình rất "bảo thủ": khi predict một class nào đó thì đúng, nhưng **bỏ sót 64,54% mẫu DoS** (phân loại nhầm thành các class khác).

- Mô hình không thể nhận diện phần lớn DoS attacks vì chưa từng thấy pattern của DoS trong quá trình training.




## 3.3 Đo lường Forgetting Measure (FM)

Forgetting Measure (FM) đo lường mức độ "quên"kiến thức của mô hình:
$FM = \frac{1}{T-1} \sum_{j=1}^{T-1} \max(0, a_{j,j} - a_{j,T})$

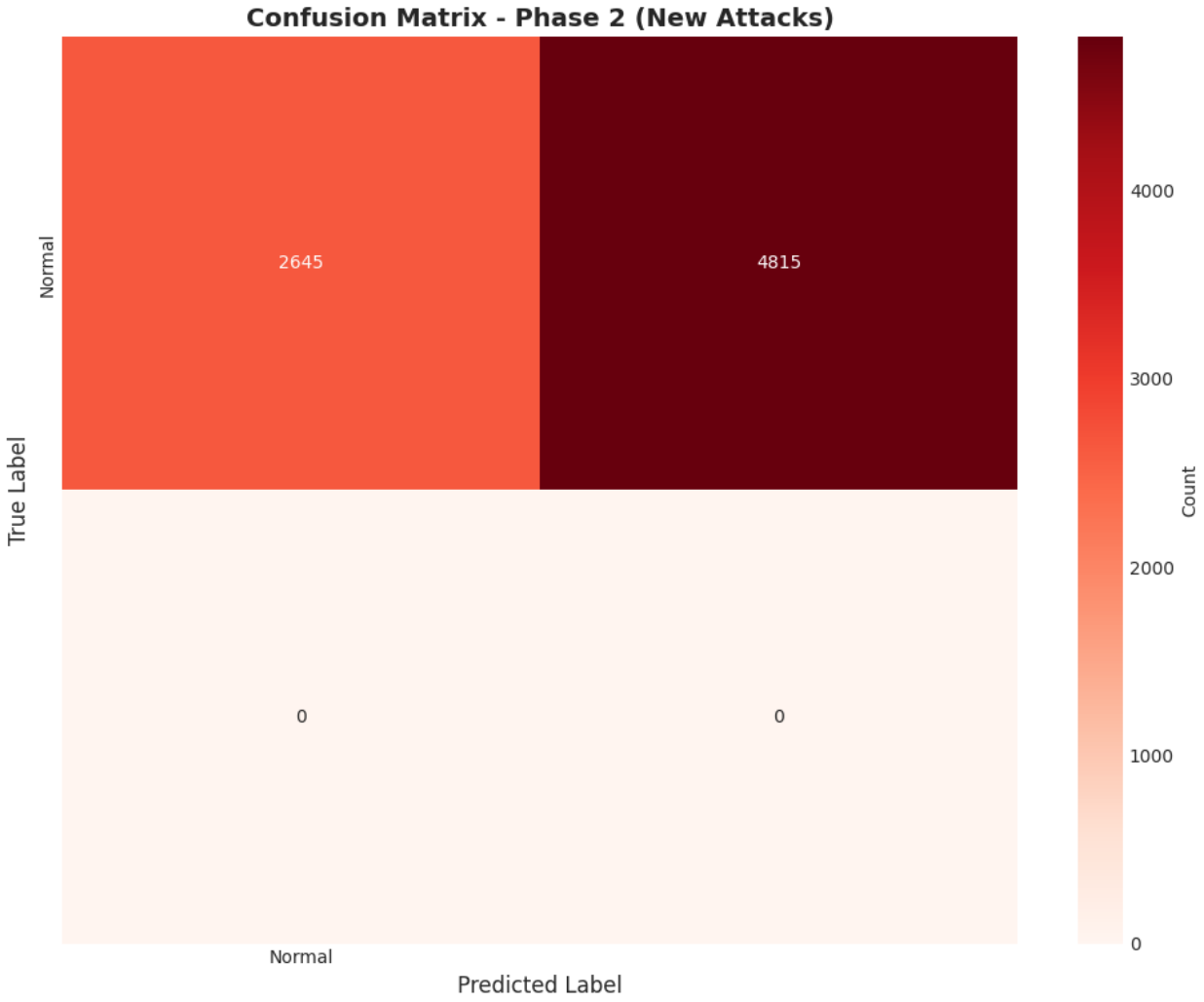
Bảng 11: Tính toán Forgetting Measure

| **Thành phần** | **Giá trị** |
|---|---|
| Accuracy Phase 1 | 0.7557 |
| Accuracy Phase 2 | 0.3546 |
| **FM** | **0.4012** |

**Đánh giá:** FM = 0.4012 > 0.1. Mô hình tĩnh không chỉ không thể học attack mới, mà còn cho thấy khoảng cách hiệu suất rất lớn giữa dữ liệu đã thấy và dữ liệu chưa thấy.

## 3.4 Phân tích Confusion Matrix

Confusion matrix trên Phase 2 cho thấy mô hình tĩnh phân loại phần lớn mẫu DoS thành các class đã biết (Normal, Probe, R2L, U2R), vì nó chỉ có thể predict các class đã được train. Đây là biểu hiện rõ ràng nhất của concept drift: mô hình cố gắng "ép"dữ liệu mới vào các khuôn mẫu cũ.




### 3.5 Tóm tắt Phần 2

Kết quả đã **chứng minh** sự suy giảm của IDS tĩnh khi gặp concept drift:

1. **Hoạt động ổn** trên dữ liệu tương tự training (Phase 1): Accuracy ~75,57%
2. **Suy giảm nghiêm trọng** trên dữ liệu mới (Phase 2): Accuracy ~35,46%
3. **FM = 0.4012** — mức catastrophic forgetting, cho thấy hạn chế cốt lõi của mô hình không thích nghi

## 4 Phần 3: Khắc phục bằng Adaptive Learning (40%)

### 4.1 Phương pháp: SGDClassifier với Online Learning

Để khắc phục hạn chế của IDS tĩnh, nhóm sử dụng **SGDClassifier** với khả năng **online learning** thông qua phương thức partial_fit(). Đây là phương pháp incremental learning cho phép mô hình cập nhật liên tục khi có dữ liệu mới mà không cần retrain từ đầu.

**Cấu hình Adaptive Model:**

| | | |
|---|---|---|
| loss | log_loss | Logistic Regression |
| penalty | l2 | Regularization L2 |
| alpha | 0.0001 | Hệ số regularization |
| max_iter | 1 | 1 pass mỗi lần gọi partial_fit |
| warm_start | True | Giữ lại trọng số từ lần fit trước |
| learning_rate | adaptive | Tự điều chỉnh learning rate |
| eta0 | 0.01 | Learning rate khởi tạo |

**Quy trình Test-Then-Train:** Mô hình adaptive sử dụng chiến lược "test rồi mới train":

1. **Predict:** Dự đoán label cho sample mới
2. **Nhận feedback:** Nhận label thực tế (giả lập analyst/SOC review)
3. **Update:** Cập nhật mô hình với sample mới qua partial_fit()

Chiến lược này giả lập kịch bản thực tế trong SOC (Security Operations Center): IDS đưa ra cảnh báo → analyst xác nhận → hệ thống tự cập nhật.

### 4.2 Phase 1: Online Training trên Old Attacks

Mô hình adaptive được huấn luyện trên Phase 1 theo kiểu mini-batch online:

- **Batch size:** 256 samples
- **Tổng số batches:** 313
- **Thời gian training:** ~0,4 giây
- **Dữ liệu:** 80.046 samples

Kết quả đánh giá trên Phase 1 Test (trước drift):

| | |
|---|---|
| **Accuracy** | **0.7567 (75,67%)** |

Hiệu suất tương đương với Static SGD (75,57%), cho thấy adaptive model học được kiến thức cũ tốt tương đương.

### 4.3 Phase 2: Adaptive Learning trên New Attacks (Test-Then-Train)

Trên Phase 2 (7.460 mẫu DoS), mô hình thực hiện test-then-train cho từng sample:

**Tiến trình learning curve:**

| | |
|---|---|
| 10% (746 mẫu) | 98,12% |
| 20% (1.492 mẫu) | 98,93% |
| 30% (2.238 mẫu) | 99,15% |
| 50% (3.730 mẫu) | 99,46% |
| 80% (5.968 mẫu) | 99,66% |
| 100% (7.460 mẫu) | **99,73%** |

**Kết quả cuối cùng trên Phase 2:**

| | |
|---|---|
| **Accuracy** | **0.9973 (99,73%)** |
| Thời gian | ~9,6 giây |

**Phân tích:** Mô hình adaptive nhanh chóng học được pattern của DoS attacks. Ngay từ 10% dữ liệu đầu tiên, accuracy đã đạt 98,12% và tiếp tục cải thiện. Điều này chứng tỏ khả năng thích nghi vượt trội so với mô hình tĩnh.

### 4.4 Kiểm tra Forgetting sau Adaptive Learning

Sau khi học Phase 2, đánh giá lại trên Phase 1 Test để kiểm tra liệu mô hình có "quên"kiến thức cũ:

| | | |
|---|---|---|
| Phase 1 Accuracy | 0.7567 | 0.6572 |
| **Adaptive FM** | — | **0.0995** |

**Đánh giá:** FM = 0.0995 < 0.1. Mô hình có suy giảm nhẹ trên dữ liệu cũ (giảm ~10%) nhưng không ở mức catastrophic — đây là trade-off chấp nhận được để đổi lấy khả năng detect attack mới.

## 5 So sánh tổng hợp và phân tích Metrics

### 5.1 Bảng so sánh chi tiết

| | | | |
|---|---|---|---|
| Phase 1 Accuracy (trước drift) | 0.7557 | 0.7567 | +0.0010 |
| Phase 2 Accuracy (new attacks) | 0.3546 | **0.9973** | **+0.6428** |
| Phase 1 Accuracy (sau drift) | 0.7557 (không đổi) | 0.6572 | -0.0985 |
| Forgetting Measure (FM) | 0.4012 | **0.0995** | **-0.3017** |


### 5.2 Phân tích Continual Learning Metrics

Ba metrics được sử dụng:

- **Average Accuracy (AA)**: $AA = \frac{1}{T}\sum_{j=1}^{T} a_{T,j}$
- **Forgetting Measure (FM)**: $FM = \frac{1}{T-1}\sum_{j=1}^{T-1} \max(0, a_{j,j} - a_{j,T})$
- **Backward Transfer (BWT)**: $BWT = \frac{1}{T-1}\sum_{j=1}^{T-1}(a_{T,j} - a_{j,j})$

**Average Accuracy (AA)** - Average Accuracy đo hiệu suất trung bình trên tất cả các tasks sau khi hoàn thành học. Trong đó T là số phase, và Accuracy_i được đo sau khi mô hình đã học xong tất cả các phase.

| Model | Phase 1 | Phase 2 | AA |
|---|---|---|---|
| Static SGD | 0.7557 | 0.3546 | **0.5552** |
| Adaptive SGD | 0.6572 | 0.9973 | **0.8273** |

**Nhận xét:** Adaptive SGD có AA = 0.8273, cao hơn đáng kể so với Static SGD (AA = 0.5552). Mặc dù Adaptive SGD giảm nhẹ trên Phase 1, nhưng bù lại bằng hiệu suất vượt trội trên Phase 2, dẫn đến AA tổng thể cao hơn 27,21 điểm phần trăm.

**Forgetting Measure (FM)**- FM đo mức độ quên kiến thức cũ sau khi học kiến thức mới

| Model | FM |
|---|---|
| Static SGD | 0.4012 |
| Adaptive SGD | **0.0995** |

**Nhận xét:** Static SGD có FM = 0.4012 vì hoàn toàn không thể xử lý attack mới, tạo ra khoảng cách lớn. Adaptive SGD giảm FM xuống 0.0995 — gần ngưỡng "không quên– nhờ cơ chế online learning giúp cập nhật dần dần thay vì thay thế hoàn toàn kiến thức cũ.

**Backward Transfer (BWT)** - BWT đo tác động của việc học task mới lên hiệu suất task cũ

| Model | BWT |
|---|---|
| Static SGD | 0.0000 |
| Adaptive SGD | **-0.0995** |

**Nhận xét:** Static SGD có BWT = 0 vì không học Phase 2 nên Phase 1 accuracy không đổi. Adaptive SGD có BWT = -0.0995, nghĩa là sau khi học DoS attacks mới, accuracy trên old attacks giảm ~10%. Đây là biểu hiện của **Stability-Plasticity Dilemma** — mô hình phải đánh đổi một phần ổn định để có khả năng thích nghi.

### 5.3 Tổng hợp 3 Metrics

| | | | |
|---|---|---|---|
| **AA** (cao = tốt) | 0.5552 | **0.8273** | Adaptive (+49,0%) |
| **FM** (thấp = tốt) | 0.4012 | **0.0995** | Adaptive (-75,2%) |
| **BWT** (gần 0 = tốt) | **0.0000** | -0.0995 | Static |

**Kết luận metrics:** Adaptive SGD vượt trội ở 2/3 metrics quan trọng nhất (AA và FM), trong khi chỉ nhượng bộ nhẹ ở BWT. Tổng thể, Adaptive SGD là lựa chọn tốt hơn cho môi trường có concept drift.

### 5.4 Phân tích Learning Curve

Learning curve trên Phase 2 cho thấy:

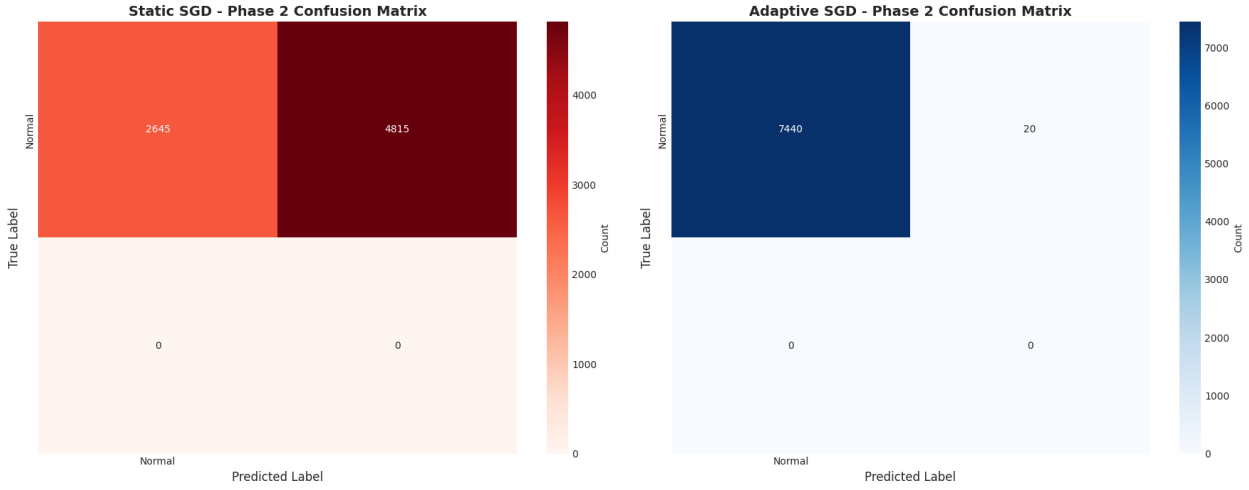
- **Adaptive SGD** bắt đầu từ accuracy ~98% ngay từ các mẫu đầu tiên và nhanh chóng hội tụ lên 99,73%. Điều này cho thấy mô hình có khả năng "học nhanh"(fast adaptation) nhờ cơ chế gradient descent liên tục.
- **Static SGD** giữ nguyên accuracy ~35,46% trên toàn bộ Phase 2, tạo thành đường ngang thấp — minh chứng cho việc mô hình không thể tự cải thiện nếu không có cơ chế cập nhật.

- Khoảng cách giữa hai đường ngày càng rõ rệt theo số mẫu được xử lý, chứng tỏ lợi ích của adaptive learning tích lũy theo thời gian.

### 5.5 Phân tích Confusion Matrix

**Static SGD trên Phase 2:** Mô hình tĩnh phân loại phần lớn mẫu DoS thành các class cũ (Normal, Probe...) vì chỉ "biết"các class đã train. Hầu hết DoS attacks bị bỏ sót hoặc phân loại sai.

**Adaptive SGD trên Phase 2:** Sau khi online learning, mô hình adaptive nhận diện chính xác gần như toàn bộ DoS attacks (chỉ 20 mẫu bị sai trong 7.460 mẫu). Điều này chứng tỏ partial_fit() giúp mô hình nhanh chóng hình thành decision boundary mới cho class DoS.




## 6 Kết luận

Thí nghiệm cho thấy mô hình IDS tĩnh gặp hiện tượng Catastrophic Forgetting) khi đối mặt với sự thay đổi của các loại tấn công mạng (concept drift), khiến độ chính xác sụt giảm nghiêm trọng (từ 75,57% xuống còn 35,46%.) Để khắc phục, phương pháp Adaptive Learning sử dụng SGDClassifier với cơ chế partial_fit() đã giúp hệ thống tự cập nhật liên tục, nâng độ chính xác khi phát hiện các cuộc tấn công mới lên mức 99,73%. Mặc dù mô hình thích nghi phải đối mặt với thách thức Stability-Plasticity, đánh đổi khoảng 10% hiệu suất trên dữ liệu cũ để học kiến thức mới, nhưng xét về tổng thể, nó vẫn vượt trội hơn mô hình tĩnh với điểm số Accuracy trung bình (AA) cao hơn 27,21% và chỉ số Forgetting Measure (FM) thấp hơn đáng kể.

## 7 Tài liệu tham khảo


1. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications.*

2. Losing, V., Hammer, B., & Wersing, H. (2018). Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, 275, 1261-1274.

3. Gomes, H. M., et al. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning*, 106(9-10), 1469-1495.

4. Lopez-Paz, D., & Ranzato, M. (2017). Gradient Episodic Memory for Continual Learning. *Advances in Neural Information Processing Systems (NeurIPS).*

5. Scikit-learn documentation: SGDClassifier — partial_fit() for incremental learning. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html