MM4flow: A Pre-trained Multi-modal Model for Versatile Network Traffic Analysis

Lab seminar (paper review)

Seonghoon Jeong

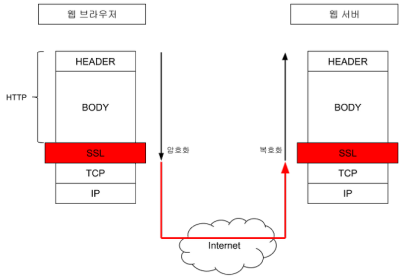
System and Network Security Lab. (SNSec Lab.)
Division of Artificial Intelligence Engineering, Sookmyung Women's University

January 30, 2026

Outline

1. **Preliminaries & Motivation**
2. **Background**: Pre-training & Tokenization
3. **System Design** (MM4flow Architecture)
4. **Evaluation** (Performance & Ablation)
5. **Conclusion and Future Work**
6. **Discussion**: My opinion

Understanding Network Traffic: HTTP Packet Structure



TCP Payload: HTTP Request-Response

Request		Response
POST / HTTP/1.1	⟷ Start line ⟶	HTTP/1.1 403 Forbidden
Host: developer.mozilla.org User-Agent: curl/8.6.0 Accept: */* Content-Type: application/json Content-Length: 345	⟷ Headers ⟶	Server: Apache Date: Fri, 21 Jun 2024 12:52:39 GMT Content-Length: 678 Content-Type: text/html Cache-Control: no-store
	⟷ Empty line ⟶	

{
"data": "ABC123"
}
	⟷ Body ⟶	<!DOCTYPE html>
		<html lang="en">
		(more data…)

HTTP request (GET method)—in detail:

```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

- `GET /doc/test.html HTTP/1.1` → Request Line
- Host … Content-Length: 35 → Request Headers
- Request Headers, Request Line → Request Message Header
- blank line → A blank line separates header & body
- `bookId=12345&author=Tan+Ah+Teck` → Request Message Body

Challenges in Conventional Network Traffic Analysis

1. Heuristic Feature Engineering

- Traditional ML relies on **hand-crafted features**.
- Requires deep domain knowledge.
- Features often fail to capture complex sequential patterns.

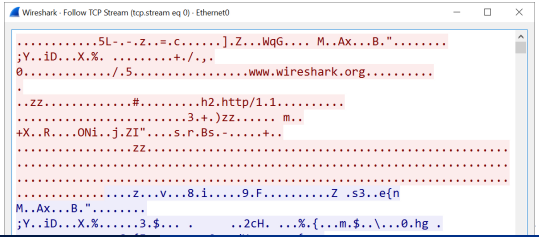
Example: Flow Statistical Features

Category	Features
Packet Size	Min, Max, Mean, Std, Skewness
Inter-arrival Time	Min, Max, Mean, Variance
Throughput	Bytes/sec, Packets/sec
Flags	Count of SYN, ACK, FIN

Labor-intensive & limited representation.

2. Encrypted Traffic (TLS/SSL)

- Encryption hides the payload content.
- **Deep Packet Inspection (DPI)** becomes ineffective.
- Pattern matching (e.g., regex for "GET /admin") no longer works.



Tokenization in Network Traffic Analysis

Tokenization: Breaking down raw data (text, bytes) into smaller units (tokens) for model processing.

NLP Example (Text)

Raw Sentence

```
"Hello, world!"
```

↓

Tokens

```
["Hello", ",", "world", "!"]
```

Network Example (Bytes)

Raw Packet (Hex)

```
47 45 54 20 2f 20 48 54 54 50
```

↓

Tokens (Sub-words/Bytes)

```
["GET", " ", "/", " ", "HTTP"
```

Paper Introduction

Title:
MM4flow: A Pre-trained Multi-modal Model for Versatile Network Traffic Analysis

Publication:
ACM SIGSAC Conference on Computer and Communications Security (CCS) 2025

Affiliation:
중국인민해방군국방과기대학—National University of Defense Technology, Changsha, Hunan, China

교신저자 Shaojing Fu는 주로 LLM과 딥러닝으로 이미지,

MM4flow: A Pre-trained Multi-modal Model for Versatile Network Traffic Analysis

Luming Yang
National University of Defense Technology
Changsha, Hunan, China

Lin Liu*
National University of Defense Technology
Changsha, Hunan, China

Junjie Huang
National University of Defense Technology
Changsha, Hunan, China

Zhuotao Liu
Tsinghua University
Beijing, China

Shiyu Liang
Shanghai Jiao Tong University
Shanghai, China

Shaojing Fu*
National University of Defense Technology
Changsha, Hunan, China

Yongjun Wang*
National University of Defense Technology
Changsha, Hunan, China

Abstract

Network traffic analysis is a critical research area, playing an essential role in enhancing network security and ensuring high-quality network services. Existing methods, which primarily rely on a single modality, face two significant limitations. First, while existing approaches may achieve strong performance in specific tasks, they often lack sufficient adaptability for diverse tasks. Second, existing pre-trained models are only trained with GB-scale traffic, with which increases the risk of over-fitting and limiting the models' overall performance. To address these challenges, we propose MM4flow, a pre-trained multi-modal model designed for versatile network traffic analysis. We divide network flows into two modalities: raw byte streams and transmission patterns, which

CCS Concepts

• **Security and privacy** → Network security.

Keywords

Network Traffic Analysis, Multi-modal, Pre-training, Identification

ACM Reference Format:

Luming Yang, Lin Liu, Junjie Huang, Zhuotao Liu, Shiyu Liang, Shaojing Fu, and Yongjun Wang. 2025. MM4flow: A Pre-trained Multi-modal Model for Versatile Network Traffic An alysis . In *Pr oceedings of th e 20 25 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), October 13–17, 2025, Taipei.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3719027.3744804

Introduction: Context & Problem Statement

What is this paper about?

1. A **Pre-trained Multi-modal Model** designed for versatile network traffic analysis.
2. Leveraging **Massive Unlabeled Data** (TB-scale) to learn universal traffic representations.

Limitations of Existing Methods

- **Uni-modal Constraints**: Byte-based models fail on *encrypted tunnels* (Acc ≈ 0.05) due to randomized payloads; Statistical models lack content granularity.
- **Insufficient General Knowledge**: Reliance on small, specific public datasets leads to poor generalization and potential bias.

Problem Statement

How can we effectively fuse multi-modal information (Content + Behavior) from massive real-world traffic to solve the "Going Dark" problem in encrypted traffic analysis?

The Proposed Approach: MM4flow

Multi-modal Modeling

- **Raw Byte Stream**: Captures static content info (~~IP/TCP Headers~~ Payload).
- **Transmission Pattern**: Captures dynamic behavior (Packet Lengths + Direction).

Two-Stage Learning

- Uni-modal Pre-training (on 77.6 TB data)
- Multi-modal Fusion (via Cross-Attention mechanism)

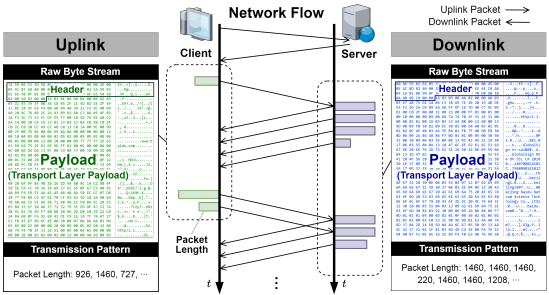


Figure 1: A network flow can mainly be represented by two modalities, raw byte stream and transmission pattern.

Key Contributions

1. **Novel Framework**: The first pre-trained model to integrate Byte and Pattern modalities for versatile analysis.
2. **Scale**: Developed an efficient collecting scheme, utilizing **77.6 TB** of real-world traffic for pre-training (vs. typical GB-scale datasets).
3. **Performance:** Demonstrated state-of-the-art results (e.g., **+84% accuracy** in encrypted tunnel website identification) and superior few-shot generalization.

Summary

MM4flow effectively bridges the gap between content and behavior, enabling robust analysis even under heavy encryption.

Comparison of Pre-training Data Scale

Table 1: The pre-training data comparison with prior pre-trained models for network traffic analysis.

Year	Work	Model Params	PCAP Size	Sample-size × Epochs	The source of pre-training data	Sim. ** PT&SFT
2022	ET-BERT [43]	132.19M	30GB	(1.6M)*	ISCXVPN [22], CIC-IDS2017 [60], CSTNET	◐
2023	YaTC [89]	1.86M	—	(76.8M)*	ISCXVPN [22], ISCXTor [36], USTC-TFC2016 [76], CICIoT2022 [17]	◕
2023	Flow-MAE [27]	85.12M	57.34GB	3.86M×800	CSE-CIC-IDS2018 [60]	◔
2024	NetMamba [73]	1.86M	77.6GB	(19.2M)*	ISCXVPN [22], ISCXTor [36], USTC-TFC2016 [76], CICIoT2022 [17], CrossPlantform [71]	●
2025	TrafficFormer [93]	132.14M	18.8GB	0.6M×160	ISCXVPN [22], CICMalAnal2017 [37], Browser [70]	◔
	MM4flow	**174.40M**	**77.6TB**	**465M×2**	**Real-world Traffic from a Network Gateway**	○

* The sample size of the pre-training data is not mentioned in the paper. Thus we can only estimate it based on *batch-size* × *training-steps* = *sample-size* × *epochs*, which is the product of the pre-training dataset size and epochs.

** It indicates the similarity (Smi.) between datasets used for supervised fine-tuning (SFT) and the pre-training (PT) data, where ● > ◕ > ◐ > ◔ > ○.

- **MM4flow** utilizes **77.6 TB** of real-world traffic.
- Significant scale advantage over previous SOTA models (typically < 100 GB).

2.1 Multi-modal Nature of Network Flows

Network flows are inherently multi-modal, consisting of:

- **Content Information (Raw Byte Stream)**:
  - Sequence of bytes in packets (Headers + Payload).
  - *Static Characteristic*: Determined by protocols and application data.
- **Behavior Information (Transmission Pattern)**:
  - Side-channel info: Packet length, direction, inter-arrival time.
  - *Dynamic Characteristic*: Affected by network conditions (loss, reorder).

Design Choice in MM4flow:

- **Byte Stream Modality**: Payload bytes (excluding specific header fields like IPs to avoid bias).
- **Transmission Pattern Modality**: Sequence of packet lengths (+/- for direction).
- *Excluded*: Packet timestamps (too unstable).

2.2 Motivation: Why Traditional Methods Fail?

Limitation 1: Uni-modal approaches are insufficient.

Byte-only Models

- Great for unencrypted traffic.
- Fail on Encrypted Tunnels: Payload is randomized; no byte patterns.

Pattern-only Models

- Good for encryption.
- Weak on Complex Apps: Misses fine-grained content distinctions.

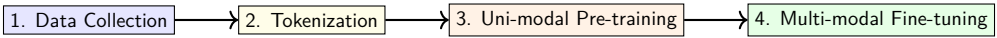
Limitation 2: Data Scale & Bias in Pre-training

- Existing pre-trained models use public datasets (GB-scale).
- **Problems:**
  1. **Scale**: Insufficient for learning general patterns.
  2. **Bias**: Public datasets have specific artifacts (e.g., lab environment IPs).

3.1 MM4flow Overview

4-Stage Pipeline:

1. **Data Collection**: Real-time capture of massive unlabeled traffic.
2. **Tokenization**: Converting flows to tokens (Bytes & Lengths).
3. **Uni-modal Pre-training**: Two BERT models learning separately (Masked Modeling).
4. **Multi-modal Fine-tuning**: Fusing knowledge for specific tasks.



3.1 Overview Architecture

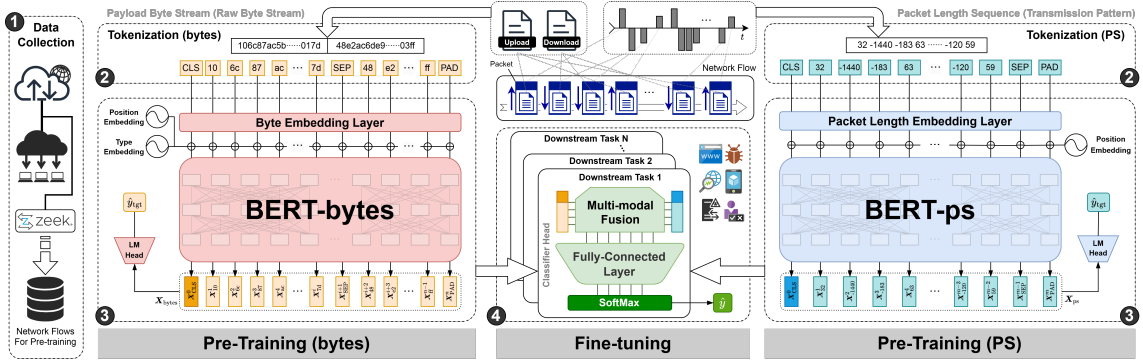


Figure 2: The overview of MM4flow. It consists of four modules: (1) data collection, (2) tokenization, (3) uni-modal pre-training, and (4) multi-modal fine-tuning. BERT-bytes and BERT-ps are applied to represent two modalities of network flows, that is, the payload byte stream (raw byte stream) and packet length sequence (transmission pattern), respectively.

3.2 Data Collection: Scaling to TBs

1. Data Collection → 2. Tokenization → 3. Uni-modal Pre-training → 4. Multi-modal Fine-tuning

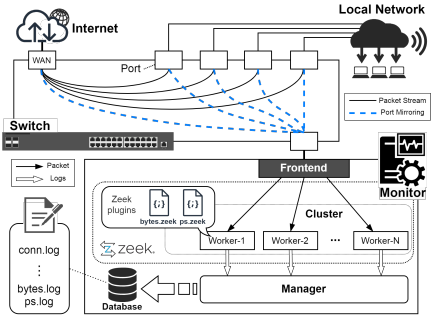
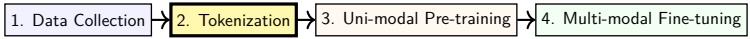


Figure 3: At the network gateway, it is the the architecture for real-time network traffic capture, whose configuration primarily consists of port mirroring and the Zeek cluster.

Novelty: Real-time efficient collection Scheme

- **Tool**: `Zeek` (Network Security Monitor).
- **Method**: Custom plugins (`ps.zeek` , `bytes.zeek`) to extract features directly.
- **Benefit**: Avoids storing massive PCAP files (Storage reduced to 0.6%).
- **Scale**:
  - Collected **77.6 TB** of real-world traffic.
  - **465 Million** network flows after filtering.
  - Significantly larger than prior works (typically 30GB).

3.3 Tokenization: Why One-Byte Tokens?



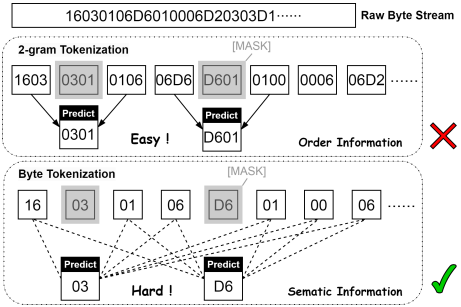


Figure 4: Two types of tokenization for raw byte streams. For 2-gram tokenization applied by [29, 43, 93], the masked token is simply a combination of the preceding token's low byte and the succeeding token's high byte. It is determined by the tokenization itself and has nothing to do with semantic information. For instance, between `06D6` and `0100`, the masked token will inevitably be `D601`. In contrast, byte tokenization requires comprehensive contextual analysis to

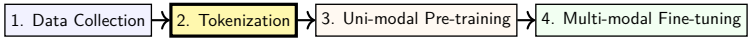
Limitation of 2-gram Tokens:

- MLM becomes a "deterministic" puzzle (e.g., `06D6 [MASK] 0100` → `[MASK]` is `D601`).
- Model fails to learn deep semantics.

Advantage of One-Byte Tokens:

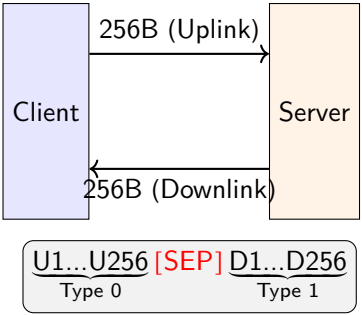
- Forces model to perform **comprehensive contextual analysis**.
- Effectively captures the semantic information of raw byte streams.

3.3.1 Payload Byte Stream

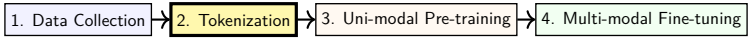


Double-Directional Input

- **Total**: 512 Bytes per flow.
- **Structure**: First 256 bytes from **Uplink** + First 256 bytes from **Downlink**.
- **Why?**: Client and Server typically serve distinct functions (e.g., HTTP GET vs. Response).
- **Modality**: Only Transport Layer Payload (excluding ~~L3/L4 headers~~).



3.3.2 Packet Length Sequence



Capturing Behavior (Transmission Pattern)

- **Modality**: Sequence of packet payload lengths.
- **Input**: First **256 packets** of the flow.
- **Logic**:
  - Positive (+): Uplink data.
  - Negative (−): Downlink data.
- **Filtering**: Exclude functional packets (e.g., ~~SYN~~, ~~ACK~~ with no payload) to focus on data transfer.

```
[CLS] +140 -200 +50 -512 ... [SEP]
```

Note: Packet length is more stable than inter-arrival time against network jitter.

3.3.3 Vocabulary and Scale Analysis

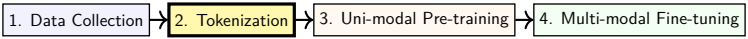


Table 4: The hyper-parameter setting of MM4flow.

MM4flow	BERT -bytes	BERT -ps	Pre-Training	
Vocabulary Size	261*	3005**	Batch Size	128
Max Length	512	256	Learning Rate	2.00E-05
Hidden Size	768		Optimizer	AdamW [46]
Hidden Layers	12		Epochs	2
Attention Heads	12		Steps	908K
Intermediate Size	3072			

* It contains 256 byte values (`0x00-0xFF`) and 5 special tokens.
** It contains 1-1500 packet length with 2 directions and 5 special tokens.
*** The special tokens applied by both BERT-ps and BERT-bytes include `[CLS]`, `[UNK]`, `[SEP]`, `[MASK]`, and `[PAD]`.

Vocabulary Size (Vocab Size):

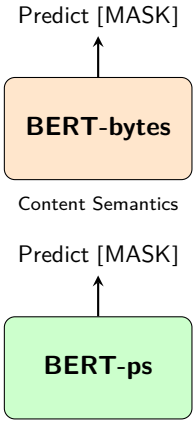
- **BERT-bytes**: 261
  - 256 values (0x00-0xFF) + 5 Special Tokens.
- **BERT-ps**: 3005
  - Packet Lengths: 1 to 1500 (MTU limit).
  - Directional: 1500 (Uplink) + 1500 (Downlink) + 5 Special Tokens.
  - Larger space → Sparse representation.

Max Position Embeddings (Max Len):

- Both fixed at **512**.
- **Bytes**: Cover 512 bytes payload.
- **Packet Sequence**: Covers sequence of 512

3.4 Uni-modal Pre-training

Two independent **BERT** models are trained.



Objective: Masked Language Modeling (MLM).

- Avoids "Modality Modality".

Embedding Definition:

$$e_i^{\text{byte}} = e^{\text{byte}} + e_i^{\text{pos}} + e^{\text{type}} \quad (1)$$

$$e_i^{\text{pl}} = e^{\text{pl}} + e_i^{\text{pos}} \quad (2)$$

(*1*) *Byte Embedding: Token + Position + Type*
($e^{type} \in \{0, 1\}$, *representing Up/Downlink*)
(*2*) *Packet Length Embedding: Token + Position*

3.4.1 Pre-training Strategy: Masked Language Modeling (MLM)

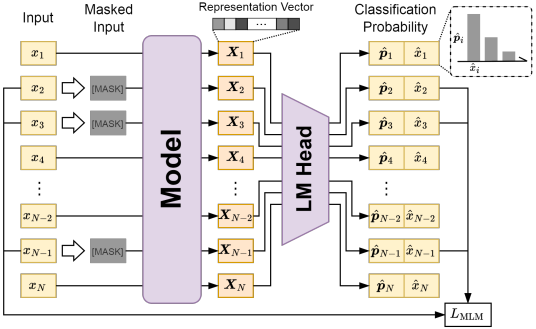


Figure 5: The schematic diagram of Masked Language Modeling (MLM), based on which the pre-training of BERT-ps and BERT-bytes can be achieved.

Fig 5. Masked Language Modeling Strategy.

Masking Strategy (Same as BERT)

- **Selection**: Randomly select **15%** of tokens in the sequence.
- **For selected tokens:**
  - **80%**: Replace with `[MASK]` token.
  - **10%**: Replace with a **random** token.
  - **10%**: Keep the **original** token unchanged.

Purpose

- Forces model to rely on **context** rather than just memorizing the token itself.
- Acts as a Denoising Autoencoder (DAE).

3.4.2 Pre-training Objective (Loss Function)

Masked Language Modeling (MLM) Loss (Eq 3):

$$\mathscr{L}_{\text{MLM}} = -\sum_{i=1}^{k} \log P_r\{\text{MASK}_i = \text{token}_i | \tilde{X}; \Theta\}$$

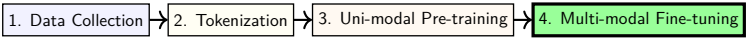
Component Breakdown:

- $\tilde{X}$: The input sequence with **masked** tokens.
- **MASK**$_i$: The i-th masked position.
- **token**$_i$: The **original** ground-truth token.
- Θ: Model parameters being optimized.

Intuition:

- Minimize the **Negative Log-Likelihood** (NLL) of the correct prediction.
- Ensures the model learns **bi-directional context** to fill in the blanks.

3.5 Multi-modal Fusion (Fine-tuning)



Cross-Attention Mechanism works as the bridge.

- **Query**: Concatenation of both outputs ($X_{bytes}||X_{ps}$).
- **Key/Value:** Uni-modal outputs (X_{bytes} and X_{ps}).

$$X'_{bytes} = \text{CrossAttention}(X_{bytes}, X_{bytes}||X_{ps}) \tag{5}$$

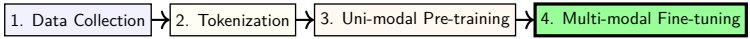
$$X'_{ps} = \text{CrossAttention}(X_{ps}, X_{bytes}||X_{ps}) \tag{6}$$

This allows each modality to "attend" to the most relevant features of the OTHER modality before final classification.

Two-Stage Fine-tuning:

- **Stage 1 (Warm-up)**: Freeze Pre-trained BERTs, train only the Fusion head.
- **Stage 2**: Full fine-tuning of all parameters.

3.5.1 Final Representation & Classification



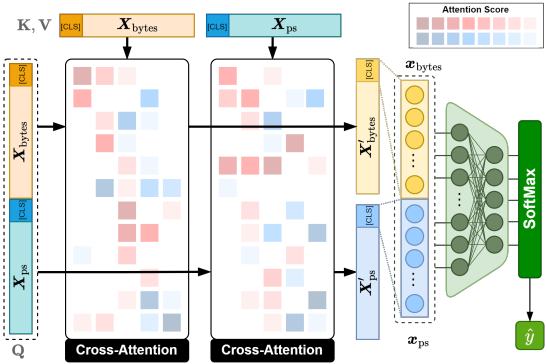


Figure 6: The architecture of the multi-modal fusion module base on cross-attention mechanism.

1. Fused Representations (Eq 5-6)

$$X'_{bytes} = \mathrm{CA}(X_{bytes}, X_{bytes} \parallel X_{ps}) \quad (5)$$

$$X'_{ps} = \mathrm{CA}(X_{ps}, X_{bytes} \parallel X_{ps}) \quad (6)$$

Refined sequence representations via Cross-Attention.

2. Classification (Eq 7)

$$p = \mathrm{Softmax}(W[x'_{bytes} \parallel x'_{ps}] + b) \quad (7)$$

- x'_{bytes}, x'_{ps}: Final [CLS] embeddings.
- $\parallel$: Concatenation.

3.5.2 Two-Stage Fine-tuning Strategy

1. Data Collection → 2. Tokenization → 3. Uni-modal Pre-training → 4. Multi-modal Fine-tuning

Strategy Overview: directly fine-tuning may damage pre-trained weights due to unstable gradients from the random head.

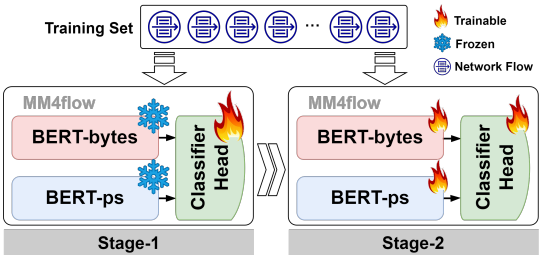


Figure 7: There are two phases in the fine-tuning process, including warm-up and full-parameter fine-tuning.

Stage 1: Warm-up

- **Action**: **Freeze** pre-trained BERT parameters.
- **Update**: Train **only** the Fusion Head (FC layer).
- **Goal**: Stabilize the classification head.

Stage 2: Full Fine-tuning

- **Action**: **Unfreeze** all parameters.
- **Update**: Train entire model with a

4.1 Experimental Settings

Pre-training Data (Scale):

- **Volume**: 77.6 TB pcap (collected over 1 week).
- **Scale:** 465M flows, 249.9B Byte tokens.
- **Comparison**: **3 orders of magnitude larger** than existing works (e.g., ET-BERT, NetMamba).

4.1 Experimental Settings

Pre-training Data (Scale):

- **Volume**: 77.6 TB pcap (collected over 1 week).
- **Scale:** 465M flows, 249.9B Byte tokens.
- **Comparison**: **3 orders of magnitude larger** than existing works (e.g., ET-BERT, NetMamba).

System Environment:

- **Hardware**: 8 × NVIDIA RTX 6000 Ada.
- **Training Time**: ≈ 350 hours (2 epochs).
- **Batch Size:** 512 (effective).

4.1 Experimental Settings

Pre-training Data (Scale):

- **Volume**: 77.6 TB pcap (collected over 1 week).
- **Scale:** 465M flows, 249.9B Byte tokens.
- **Comparison**: **3 orders of magnitude larger** than existing works (e.g., ET-BERT, NetMamba).

System Environment:

- **Hardware**: 8 × NVIDIA RTX 6000 Ada.
- **Training Time**: ≈ 350 hours (2 epochs).
- **Batch Size:** 512 (effective).

Baselines (11 Approaches):

- **Byte-based (6)**: CNN, EBSNN, ET-BERT, YaTC, NetMamba, TrafficFormer.
- **Flow-based (5)**: AppScanner, ETC-PS, FlowLens, FS-Net, GraphDApp.

4.1 Experimental Settings

Pre-training Data (Scale):

- **Volume**: 77.6 TB pcap (collected over 1 week).
- **Scale:** 465M flows, 249.9B Byte tokens.
- **Comparison**: **3 orders of magnitude larger** than existing works (e.g., ET-BERT, NetMamba).

System Environment:

- **Hardware**: 8 × NVIDIA RTX 6000 Ada.
- **Training Time**: ≈ 350 hours (2 epochs).
- **Batch Size:** 512 (effective).

Baselines (11 Approaches):

- **Byte-based (6)**: CNN, EBSNN, ET-BERT, YaTC, NetMamba, TrafficFormer.
- **Flow-based (5)**: AppScanner, ETC-PS, FlowLens, FS-Net, GraphDApp.

Evaluation Datasets (6 Diverse Tasks):

- **Malware**: DataCon2020.
- **Proxy**: DataCon2021-p1, DataCon2021-p2.
- **Application**: Browser, NUDT_MobileTraffic, CSTNET-TLS1.3.

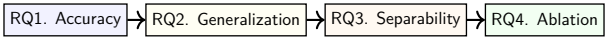
Dataset

Table 5: The additional details about these datasets for evaluations and their scale after processing.

Dataset	Year	Inst	PCAP Size	Class Num	Class Samples*	Downstream Task	Description
DataCon2020 [18]	2020	QIAN-XIN	6.6GB	2	5000	Malware encrypted traffic detection	This dataset is derived from the TLS/SSL encrypted traffic generated by malware and normal software (both are exe types) in the Tianqiong sandbox.
DataCon2021-p1 [19]	2021	THU	1.2GB	6	500	Encrypted proxy traffic classification	The encryption proxy softwares include Firefox, V2ray, Clash, Lantern, Netch, and Shadowsocks. The traffic is generated by accessing a list of websites using Selenium automatically.
DataCon2021-p2 [19]	2021	THU	4.6GB	22	1000	Website traffic identification under encrypted tunnel	The traffic was generated by accessing different websites through the same encryption proxy software.
Browser [70]	2020	UT	7.4GB	4	5000	Browser traffic classification	It included network traffic from scraping the Alexa Top 1,000 websites on Chrome, Firefox, Samsung Internet, and UC Browser on mobile devices.
NUDT_MobileTraffic [90]	2023	NUDT	707.0GB	300	200	Mobile application traffic identification	It is a mobile network traffic dataset published by the Network Forensics Research Lab. Based on a traffic capture software they developed, network flows are accurately labeled and collected.
CSTNET-TLS1.3 [42]	2022	IIE CAS	10.9GB	80	300	TLS 1.3 website traffic identification	This traffic dataset are acquired from Alexa Top-5000 deployed with TLS1.3, which is collected on CSTNET.

* It is the sample count per classes for training set rather than the whole dataset.

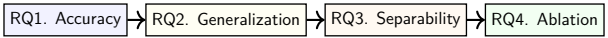
4.2 Research Questions (RQs)



RQ1: Accuracy Comparison (Versatility)

- Does MM4flow achieve superior performance across diverse tasks?

4.2 Research Questions (RQs)



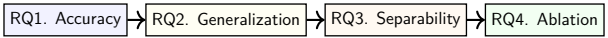
RQ1: Accuracy Comparison (Versatility)

- Does MM4flow achieve superior performance across diverse tasks?

RQ2: Generalization (Few-Shot)

- How well does MM4flow perform with limited labeled data?

4.2 Research Questions (RQs)



RQ1: Accuracy Comparison (Versatility)

- Does MM4flow achieve superior performance across diverse tasks?

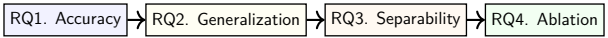
RQ2: Generalization (Few-Shot)

- How well does MM4flow perform with limited labeled data?

RQ3: Separability Analysis

- Does pre-training enhance the distinctiveness of flow representations?

4.2 Research Questions (RQs)



RQ1: Accuracy Comparison (Versatility)

- Does MM4flow achieve superior performance across diverse tasks?

RQ2: Generalization (Few-Shot)

- How well does MM4flow perform with limited labeled data?

RQ3: Separability Analysis

- Does pre-training enhance the distinctiveness of flow representations?

RQ4: Ablation Study

- How do different components (Uni-modal Pre-training, Cross-Attention) contribute?

4.3 RQ1: Accuracy Comparison

RQ1. Accuracy → RQ2. Generalization → RQ3. Separability → RQ4. Ablation

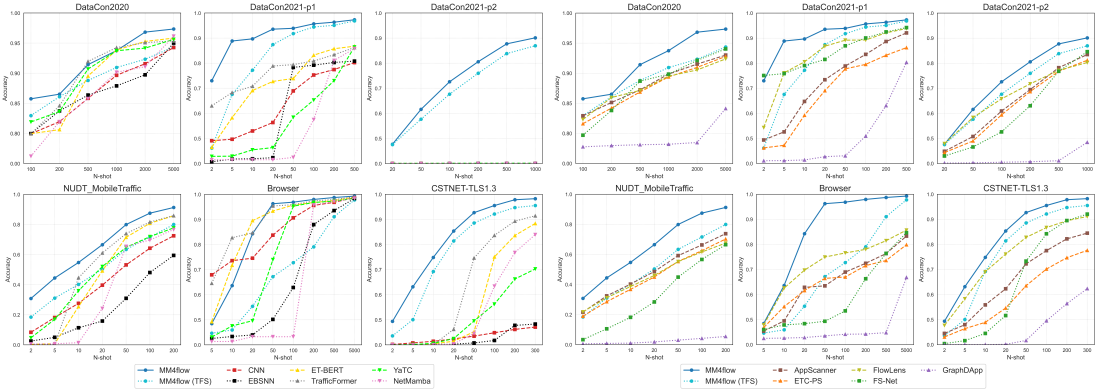
Table 2: The identification results of MM4flow and baselines on six public datasets.

Dataset / Method	DataCon2020 [18] Acc	macro-P	macro-R	macro-F_1	DataCon2021-p1 [19] Acc	macro-P	macro-R	macro-F_1	DataCon2021-p2 [19] Acc	macro-P	macro-R	macro-F_1
AppScanner [69]	0.9302	0.9236	0.9362	0.9281	0.9209	0.9052	0.9267	0.9135	0.8346	0.8302	0.8383	0.8330
ETC-PS [81] ‘	0.9280	0.9214	0.9339	0.9258	0.8622	0.8221	0.8629	0.8360	0.8118	0.8068	0.8153	0.8095
FlowLens [9]	0.9238	0.9172	0.9293	0.9216	0.9380	0.9264	0.9406	0.9323	0.8027	0.7967	0.8062	0.7998
FS-Net [45]	0.9405	0.9350	0.9426	0.9383	0.9423	0.9275	0.9449	0.9351	0.8460	0.8398	0.8461	0.8419
GraphDApp [66]	0.8414	0.8388	0.8543	0.8391	0.8033	0.7886	0.8295	0.7887	0.4599	0.5237	0.4540	0.4485
CNN [75]	0.9427	0.9368	0.9462	0.9407	0.8025	0.7577	0.7735	0.7514	0.0570	0.0560	0.0572	0.0560
EBSNN [80]	0.9799	0.9800	0.9799	0.9799	0.7002	0.7236	0.7002	0.7071	0.3649	0.4011	0.3489	0.3361
ET-BERT [43]	0.9562	0.9563	0.9562	0.9563	0.7302	0.8364	0.7303	0.6946	0.0574	0.0615	0.0574	0.0533
YaTC [89]	0.9562	0.9563	0.9562	0.9563	0.7302	0.8364	0.7303	0.6946	0.0574	0.0615	0.0574	0.0533
NetMamba [73]	0.9616	0.9579	0.9623	0.9600	0.8107	0.7732	0.7697	0.7598	0.0325	0.0015	0.0455	0.0029
TrafficFormer [93]	0.9537	0.9476	0.9595	0.9522	0.8577	0.7877	0.7592	0.7617	0.0298	0.0100	0.0478	0.0091
MM4flow	**0.9734**	**0.9699**	**0.9752**	**0.9724**	**0.9731**	**0.9637**	**0.9728**	**0.9676**	**0.9011**	**0.8963**	**0.9002**	**0.8976**

Dataset / Method	Browser [70] Acc	macro-P	macro-R	macro-F_1	NUDT_MobileTraffic [90] Acc	macro-P	macro-R	macro-F_1	CSTNET-TLS1.3 [42] Acc	macro-P	macro-R	macro-F_1
AppScanner [69]	0.8331	0.8145	0.8131	0.8129	0.7211	0.7186	0.7211	0.7178	0.8451	0.8384	0.8395	0.8369
ETC-PS [81]	0.7988	0.7791	0.7732	0.7750	0.6843	0.6814	0.6843	0.6806	0.7764	0.7689	0.7719	0.7679
FlowLens [9]	0.8572	0.8386	0.8369	0.8371	0.6611	0.6640	0.6611	0.6580	0.9114	0.9075	0.9083	0.9068
FS-Net [45]	0.8466	0.8284	0.8289	0.8275	0.6549	0.6570	0.6549	0.6531	0.9208	0.9153	0.9176	0.9160
GraphDApp [66]	0.6684	0.6662	0.6612	0.6540	0.1562	0.2624	0.1562	0.1373	0.6236	0.6975	0.6168	0.6207
CNN [75]	0.9865	0.9859	0.9864	0.9861	0.7202	0.7212	0.7202	0.7191	0.4210	0.4045	0.4104	0.4038
EBSNN [80]	0.9835	0.9822	0.9838	0.9830	0.5995	0.6101	0.5995	0.5931	0.4554	0.4533	0.4430	0.4052
ET-BERT [43]	0.9893	0.9892	0.9892	0.9892	0.8578	0.8618	0.8578	0.8581	0.8839	0.8768	0.8763	0.8748
YaTC [89]	0.9835	0.9836	0.9835	0.9835	0.7771	0.7903	0.7772	0.7769	0.7026	0.7267	0.7027	0.6941
NetMamba [73]	0.9839	0.9839	0.9827	0.9833	0.7663	0.7759	0.7663	0.7657	0.8394	0.8371	0.8313	0.8267
TrafficFormer [93]	0.9892	0.9889	0.9892	0.9891	0.8583	0.8612	0.8583	0.8586	0.9144	0.9080	0.9095	0.9073
MM4flow	**0.9929**	**0.9926**	**0.9926**	**0.9926**	**0.9111**	**0.9116**	**0.9111**	**0.9110**	**0.9826**	**0.9814**	**0.9822**	**0.9817**

4.4 RQ2: Generalization (Few-Shot)

RQ1. Accuracy → RQ2. Generalization → RQ3. Separability → RQ4. Ablation

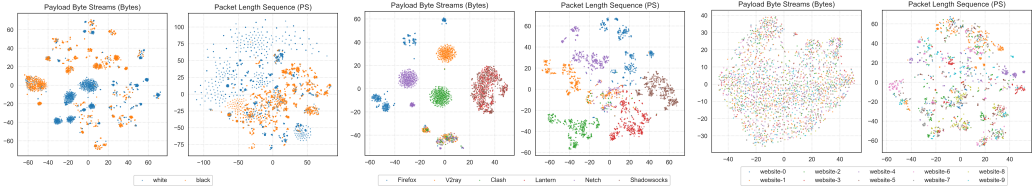


(a) Payload Byte Stream (Bytes) **(b) Packet Length Sequence (PS)**

Figure 8: The performance comparison on few-shot settings

4.5 RQ3: Embedding Analysis (Separability)

RQ1. Accuracy → RQ2. Generalization → RQ3. Separability → RQ4. Ablation



(a) DataCon2020 (b) DataCon2021-p1 (c) DataCon2021-p2

(d) NUDT_MobileTraffic (e) Browser (f) CSTNET-TLS1.3

Figure 9: The t-SNE visualization of pre-trained embeddings on six public datasets.

4.6 RQ4: Ablation Study (Component Analysis)

RQ1. Accuracy → RQ2. Generalization → RQ3. Separability → RQ4. Ablation

Table 3: Ablation study of MM4flow on six public datasets.

Dataset / Model		DataCon2020		DataCon2021-p1		DataCon2021-p2		NUDT_MobileTraffic		Browser		CSTNET-TLS1.3	
		Acc	macro-F_1	Acc	macro-F_1	Acc	macro-F_1	Acc	macro-F_1	Acc	macro-F_1	Acc	macro-F_1
BERT-ps	TFS	0.9400	0.9380	0.9603	0.9535	0.8759	0.8715	0.7820	0.7809	0.8498	0.8299	0.9585	0.9560
	SFT	0.9493	0.9473	0.9680	0.9623	0.9026	0.8999	0.8193	0.8184	0.8752	0.8580	0.9757	0.9742
BERT-bytes	TFS	0.9556	0.9539	0.7883	0.7352	0.0427	0.0072	0.0033	0.0000	0.9862	0.9855	0.4492	0.3872
	SFT	0.9721	0.9710	0.8146	0.7638	0.0471	0.0125	0.8898	0.8902	0.9905	0.9903	0.9633	0.9601
MM4flow w/o cross-attention	TFS	0.9406	0.9385	0.9669	0.9602	0.8723	0.8671	0.7815	0.7822	0.9519	0.9514	0.9598	0.9571
	SFT	0.9727	0.9715	0.9709	0.9643	**0.9028**	**0.8981**	0.9080	0.9085	0.9920	0.9918	0.9800	0.9788
MM4flow	TFS	0.9437	0.9416	0.9685	0.9610	0.8694	0.8650	0.7614	0.7607	0.9777	0.9776	0.9552	0.9524
	SFT	**0.9734**	**0.9724**	**0.9731**	**0.9676**	0.9011	0.8976	**0.9111**	**0.9110**	**0.9929**	**0.9926**	**0.9826**	**0.9817**

* TFS denotes the model Trained From Scratch without pre-trained parameters.

** SFT refers to the model trained by Supervised Fine-Tuning with pre-trained parameters, i.e. MM4flow in other tables and figures.

4.6 RQ4: Ablation Study (Modality Importance)

RQ1. Accuracy → RQ2. Generalization → RQ3. Separability → RQ4. Ablation

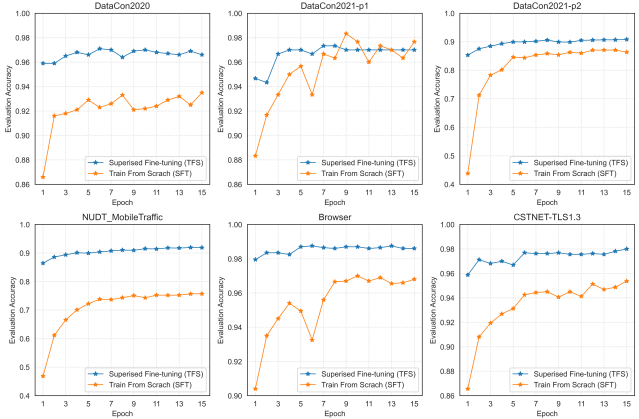


Figure 10: The accuracy change during supervised fine-tuning and training from scratch on six public datasets.

- Multi-modal is better than uni-modal.
- Multi-modal fusion is effective.
- SFT is better than TFS.
- Pre-training makes learning stable.

Conclusion and Takeaways

1. **Versatility**: First multi-modal pre-trained model handling both Byte Stream and Transmission Pattern.
2. **Scale**: Trained on 77.6 TB of real-world traffic (collecting via efficient Zeek plugin).
3. **Performance:** Solves the "Encrypted Tunnel" problem where traditional DL models fail.

Future Work

- **New Modalities**: Mining additional flow features for more comprehensive representation.
- **Real-time Deployment**: Addressing inference speed and hardware dependency (Transformer overhead) for high-speed networks.

Discussion: Reviewer's Opinion

나의 생각

- **Strength**: 직관적인 연구 결과와 학교 레벨에서 수행하기 힘든 압도적 스케일(77TB)의 실험.
- **Question**: 기존 기술(BERT, Cross-Attention)의 조합임에도 불구하고 Top-tier(CCS)에 게재된 요인은?
  - 명확한 문제 정의(Encrypted Tunnel)와 이를 해결하기 위한 모달리티 융합의 논리적 타당성.
- **Limitation**: TB-scale 데이터셋의 다양성은 인정하나, 수집 환경(ISP, Campus 등)에 따라 모델의 편향(Bias)이 발생할 수 있음.

해볼 만한 것 (Future Directions)

- **Domain Integration**: 프로토콜 사양서(Spec)를 피쳐 엔지니어링에 반영하여 체계적인 Analyzer로 발전.
- **XAI**: 탐지 결과에 대한 설명 및 시각화(Visualization).
- **Efficiency**: 실시간 탐지를 위한 모델 경량화(Distillation).
- **Adaptation**: 네트워크 환경 변화(Concept Drift)에 대응하는 적응형 학습.

Thank You!

Q & A