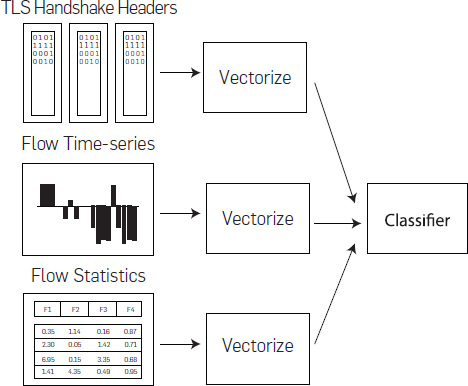
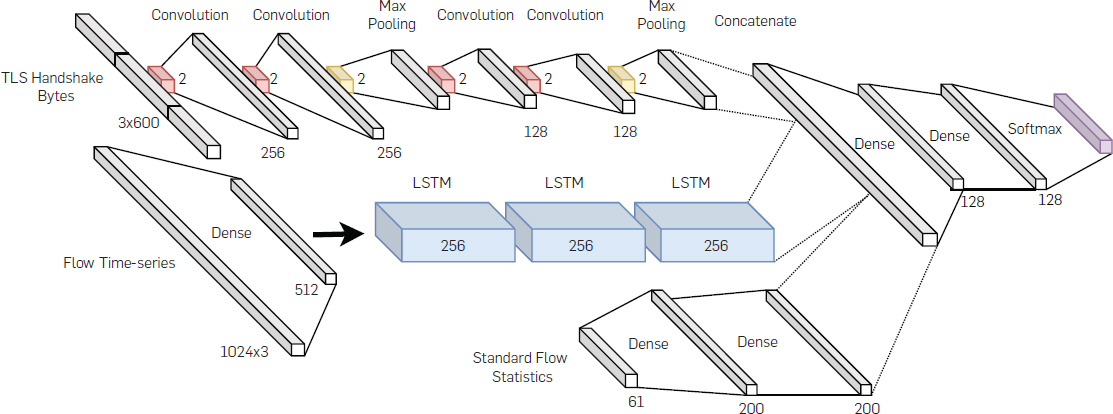
# Traffic Classification in an Increasingly Encrypted Web – Communications of the ACM - 2022

Si basa su articoli che vanno dal 2017 al 2019. SI concentra su DL applicato a HTTP/2 & QUIC quindi traffico cifrato. In pratica prende statistiche dei flussi e byte degli header per allenare DL e sostiene che il modello non deve basarsi su informazioni in chiaro quali SNI per una stabilità futura. Siamo nel 2025 e continuano ad essere usate queste informazioni. Perché non basarsi più su DL ed adottare un metodo più veloce che legge queste informazioni e non usa una DL? Hanno usato **Orange’20 dataset.** For privacy concerns, the IP addresses are masked and the packet pay-loads are removed with the exception of TLS headers. → 800K flussi ma usati 300K TLS e dalle label identificati 120k circa. Nel UCDavis QUIC dataset con 3637 flussi. Hanno usato YAF per estrarre i flussi dai file pcap che non viene aggiornato da 7 anni e Apache Spark per filtrare e preparere tutti i dati. Usano **CICFlowMeter di 7 anni fa per raccogliere statistiche sui flussi**



# A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification 2006

Dal 2006 si parla di ML che utilizza “ payload-independent features such as packet length and inter-arrival time distributions ”

# Network traffic classifier with convolutional and recurrent neural networks for Internet of Things 2017

Presenta una tecnica di NTC unita al DL per l’IoT usando un RNN e un CNN. Usano nDPI per prendere informazioni

# Independent Comparison of Popular DPI Tools for Traffic Classification 2015

Our study includes 2 commercial products (PACE and NBAR) and 4 open-source tools (OpenDPI, L7-filter, nDPI, and Libprotoident)

# TLS Fingerprinting using JA3 for Android Application 2024

TLS JA3 fingerprint fa la detect di applicazioni mobile. Tiene l’abstract / intro da cui prendere spunto. Questa è una tesi quindi non si può citare ma si può copiare. TESI DI MERDA ma cmq ci sta leggerla

# Unsupervised Detection and Clustering of Malicious TLS Flows 2022

Abstract e intro interessanti su TLS e malware. “Timing-related features have been used by prior work, but our examination found them too sensitive to the specific network setup”. Se il TLS SNI Header è criptato è possibile usare DNS. Da citare poiché utilizzano campi TLS e non per fare label, tramite ML, malware ma questo può estendersi per qualsiasi protocollo. Suddivisa in:

abstract – introduction – related work - discussion ( limiti e improvment ) - conclusion - bib

# Tesi di Magri 2025

Intro – capitolo 2 di teoria – capitolo 3 di teoria – capitolo 4 di teoria – capitolo 5 di teoria – capitolo 6 di teoria – capitolo 7 di teoria misto a confronto tecniche di teoria – capitolo 8 implementazione – conclusione

Siti blacklist / classification: http://dsi.ut-capitole.fr/blacklists/index\_en.php

**Programmi**

- Cisco Joy: <https://github.com/cisco/joy?tab=readme-ov-file> | Non aggiornato da 6 anni | un dataset trovato: fingerprint\_db.json | spiegazione: <https://developer.cisco.com/codeexchange/github/repo/cisco/joy/> | Succeduto da Mercury

- Cisco Mercury: <https://github.com/cisco/mercury> | attualmente in uso da CISCO → **network metadata capture and analysis** | Quello che fa è prendere dei metadati dai pacchetti e visualizzarli ( + metadati di nDPI ) → Una parte di quello che fa nDPI | <https://github.com/cisco/mercury/wiki/Using-Mercury> |

- NFStream: <https://www.nfstream.org/> basato su nDPI <https://github.com/nfstream/nfstream?tab=readme-ov-file> |

- Libprotoident: <https://github.com/LibtraceTeam/libprotoident> | Non aggiornato da 5 anni

Info utili intoduzione / abstract:

Qualsiasi servizio VPN può essere utilizzato dagli utenti per aggirare i filtri o le

firme applicati sui dispositivi di sicurezza della rete. Questi servizi possono essere la fonte di nuovi virus o worm iniettati all'interno della

rete o un gateway per facilitare la fuga di informazioni. In questo documento, abbiamo proposto un approccio innovativo per rilevare l'attività VPN

all'interno della rete. Il sistema proposto analizza la comunicazione tra l'utente e il server per analizzare ed estrarre

caratteristiche dal livello di rete, trasporto e applicazione che non sono crittografate e classificare il traffico in entrata come dannoso,

ovvero traffico VPN o traffico standard. Il traffico di rete viene analizzato e classificato utilizzando pacchetti DNS (Domain Name System) e

traffico basato su HTTPS (Hypertext Transfer Protocol Secure). Una volta classificato il traffico, viene analizzata la connessione in base all'IP del server, alla

porta TCP connessa, al nome di dominio e al nome del server all'interno della connessione HTTPS. Questo aiuta a verificare la

connessione legittima e a segnalare il traffico basato su VPN. Abbiamo lavorato sui cinque principali servizi VPN disponibili gratuitamente e abbiamo analizzato i loro

modelli di traffico; i risultati mostrano il rilevamento efficace dell'attività VPN eseguita dall'utente. Il riconoscimento accurato di protocolli applicativi su traffico cifrato rappresenta una sfida crescente per i sistemi di Deep Packet Inspection (DPI). Strumenti come \textit{nDPI} permettono di identificare numerosi protocolli noti, ma in presenza di protocolli TLS non riconosciuti — come quelli impiegati da VPN, proxy o applicazioni proprietarie — la classificazione si limita a etichette generiche, rendendo complesso il blocco o l’analisi di tali comunicazioni. Nel caso delle VPN, il problema è ulteriormente aggravato dall’uso di un handshake TLS iniziale, spesso riconoscibile, seguito da flussi TCP o UDP cifrati che possono utilizzare indirizzi IP e porte differenti.

Negli ultimi tempi, TLS (Transport Layer Security) è diventato il protocollo principale utilizzato per garantire comunicazioni sicure su Internet. Sistemi di DPI (Deep Packet Inspection) come nDPI offrono già il supporto al riconoscimento di numerosi protocolli cifrati, ma spesso non riescono a distinguere sottoprotocolli specifici o servizi nascosti all’interno di connessioni TLS, come avviene per alcune VPN. Questa tesi propone un approccio semi-automatico per l’identificazione di sottoprotocolli TLS sconosciuti, basato sull’analisi dei metadati dell’handshake TLS (SNI, fingerprint del Client Hello, caratteristiche dei certificati, estensioni utilizzate) e la correlazione con i flussi TCP/UDP successivi. L’obiettivo è generare regole di riconoscimento compatibili con nDPI, migliorandone la capacità di classificazione senza richiedere la decrittazione del traffico. Il metodo è stato validato su dataset di traffico reale, comprendente sia servizi legittimi sia connessioni VPN, mostrando un incremento nella capacità di identificazione e blocco di intere sessioni associate a protocolli crittografati non precedentemente riconosciuti.

\newline

\newline

\noindent

Questa tesi propone una metodologia per l’identificazione e la caratterizzazione di protocolli TLS sconosciuti a \textit{nDPI}, con particolare attenzione alle VPN. La soluzione prevede: (i) l’estrazione di metadati rilevanti dai flussi TLS generici (es. SNI, JA3, certificati, ASN, pattern temporali e dimensionali dei pacchetti); (ii) la correlazione dell’handshake TLS con i successivi flussi dati della stessa sessione VPN tramite criteri temporali, topologici e statistici; (iii) la generazione automatica di un insieme di metadati strutturati e di bozze di regole compatibili con \textit{nDPI}, facilmente integrabili anche da operatori non esperti.

\newline

\newline

\noindent

La metodologia è validata su un dataset di traffico reale contenente diverse tipologie di VPN. I risultati dimostrano un miglioramento significativo nella capacità di correlare flussi TLS generici a traffico VPN completo, riducendo il tempo necessario per creare nuove firme e aumentando l’efficacia del rilevamento rispetto all’utilizzo di \textit{nDPI} in configurazione standard.

Una rete privata virtuale (VPN) è un servizio che nasconde il traffico reale creando un canale protetto da TLS tra l'utente e il server. L'utente all'interno della rete con intenzioni malevole o che desidera nascondere la propria attività all'amministrazione della sicurezza di rete dell'organizzazione può utilizzare i servizi VPN.

1. Problema

L’analisi del traffico di rete cifrato è oggi una sfida cruciale per le aziende e per i fornitori di soluzioni di Deep Packet Inspection (DPI). Strumenti come nDPI sono in grado di riconoscere un ampio numero di protocolli, ma quando si trovano di fronte a protocolli non noti — come nuove VPN, proxy o applicazioni che usano TLS/QUIC — la classificazione si limita a etichette generiche (“TLS”, “QUIC”, ecc.).

La creazione di firme per questi protocolli richiede competenze elevate e un’analisi manuale dei pacchetti, attività che risulta lenta, soggetta a errori e difficile da scalare. Questo limita la capacità di reagire rapidamente a nuove minacce o a traffico non autorizzato.

Nel caso specifico delle VPN, il problema è duplice:

Il flusso TLS iniziale (handshake) può essere riconosciuto tramite indicatori come SNI o JA3, ma non esiste un meccanismo per associare automaticamente tale handshake ai flussi TCP/UDP successivi che trasportano il traffico incapsulato.

Senza questa correlazione, il sistema non è in grado di bloccare l’intero canale VPN, ma solo l’handshake iniziale, lasciando aperta la possibilità di elusione

Le tecniche di identificazione dei protocolli cifrati si dividono principalmente in due categorie:

Metodi statici

Utilizzano indicatori direttamente osservabili nei pacchetti, come:

Server Name Indication (SNI) nel Client Hello TLS

JA3 / JA3S fingerprint delle estensioni TLS

Informazioni dai certificati X.509

Indirizzi IP o ASN di destinazione

Questi approcci sono semplici ed efficienti, ma vulnerabili a tecniche di offuscamento (es. ESNI, SNI encryption, camouflage TLS).

Metodi basati su analisi statistica o Machine Learning

Analizzano il comportamento del flusso cifrato (es. numero di pacchetti, durata, dimensioni medie, distribuzione temporale, rapporti up/down). Alcuni lavori utilizzano algoritmi di classificazione supervisionata o one-class (come One-Class SVM) per distinguere traffico VPN da traffico legittimo. Questi metodi funzionano anche quando i metadati TLS non sono disponibili, ma richiedono dataset di addestramento aggiornati e hanno maggiore complessità operativa.

Limitazioni attuali

Pochi strumenti integrano in modo nativo la correlazione tra flusso TLS iniziale e successivi flussi cifrati associati. Ad esempio, nDPI identifica l’handshake TLS ma non lo collega automaticamente al traffico dati VPN, rendendo necessario un intervento manuale per generare regole di riconoscimento complete.

"nDPI non riconosce in automatico nuovi protocolli TLS, come quelli usati da VPN, proxy o app nuove, quindi oggi serve un esperto per creare manualmente regole di riconoscimento. Questo rallenta il processo e riduce la capacità di bloccare traffico indesiderato"

problema di detection e automazione

"Progettare e validare una metodologia per l’identificazione e la caratterizzazione di protocolli TLS sconosciuti a nDPI, con particolare attenzione alle VPN, generando automaticamente metadati e bozze di regole pronte per l’integrazione nel motore di DPI."

Questa tesi propone una metodologia e una proof-of-concept per:

Analizzare un file PCAP tramite nDPI e altri strumenti (es. tshark, Zeek).

Isolare i flussi TLS generici con indicatori sospetti di VPN (SNI, JA3, ASN).

Correlare l’handshake TLS con flussi TCP o UDP successivi appartenenti alla stessa sessione VPN, basandosi su:

- Tempistica (flussi avviati entro una finestra temporale dall’handshake)

- ASN o subnet comuni

- Pattern di pacchetti tipici del tunneling VPN

- Risoluzioni DNS recenti dell’host

Generare automaticamente un insieme di metadati strutturati e una bozza di regola nDPI che un operatore può validare e integrare.

Validare la metodologia su un dataset di traffico reale contenente più tipologie di VPN, confrontando i risultati con la classificazione di nDPI in modalità standard.

L’obiettivo finale è ridurre il tempo e le competenze necessarie per estendere nDPI al riconoscimento di nuovi protocolli TLS, fornendo una base automatizzata per la creazione di regole di riconoscimentoI

---

Certo! Ti spiego per bene l’articolo “Fine-grained TLS services classification with reject option” e ti evidenzio i punti chiave utili anche per chi, come te, vuole capire come associare i flussi TLS a servizi/applicazioni.

# Obiettivo e contributi

Il lavoro affronta la classificazione di traffico **TLS** a livello di **servizio web** (non solo “categoria”, ma etichette fini come facebook-web, outlook, opera-autoupdate, ecc.) e introduce esplicitamente la **reject option**: la capacità del modello di **astenersi** e marcare come sconosciuto un flusso che non appartiene a nessuna classe nota (“novel class detection”). Per farlo, gli autori costruiscono un **dataset molto grande** e aggiornato (≈ **140 milioni** di flussi, **~200** servizi) e mostrano che una rete neurale multimodale raggiunge **~97%** di accuratezza di classificazione e rileva **~92%** dei servizi sconosciuti con **5%** di FPR, usando metodi di novelty detection (softmax, energy, gradient-based).

# Come costruiscono il dataset (e perché conta)

* **Sorgente e durata**: raccolta su rete di produzione per **due settimane**, con pipeline IPFIX → NEMEA e un modulo che salva i flussi legati a domini di interesse (tramite SNI) con campionamento controllato. Vengono mantenuti **metadati di pacchetto** e **statistiche di flusso**, mai i payload.
* **Etichettatura**: l’etichetta “servizio” è ricavata mappando **SNI**↦servizio (liste di domini, wildcard tipo \*.youtube.com, doc “whitelist” ufficiali, Netify tool). È anche definita una **gerarchia** per “superclassi” (es. Google, Mozilla) utile a misurare metriche tolleranti agli scambi fra servizi dello stesso fornitore. Hanno selezionato **200** servizi (21 categorie) privilegiando varietà e volume; la copertura dei 200 servizi è circa **45%** dei flussi TLS con SNI osservati (porta 443).
* **Privacy**: rimozione di IP/porte e altri campi identificanti; niente payload; nessun rischio realistico di deanonymization.
* **Perché non solo SNI**: SNI può essere falsificato, potrebbe essere cifrato (ECH), ed è impossibile mantenerne una mappa completa; un modello che apprende i **pattern comportamentali** dei pacchetti può generalizzare al di là del dominio. Gli autori propongono un uso **combinato**: SNI come regola deterministica + modello come sanity check e per scoprire domini non mappati.

# Quali feature usano (riutilizzabili anche senza ML)

Input **multimodale**:

1. **PSTATS** = sequenze dei **primi 30 pacchetti con payload** (si ignorano ACK/zero-payload), con **dimensione** del payload, **direzione** (±1) e **inter-arrival time (IAT)**. Le sequenze sono paddate a 30.
2. **FLOWSTATS** = statistiche aggregare: #byte/#pacchetti in entrambe le direzioni, **durata**, **#roundtrip** (cambi di direzione), e **TCP flags** selezionate (escludendo quelle senza varianza, come SYN/ACK).

Preprocessing: **standardizzazione** e **clipping** (quantile per FLOWSTATS; limiti per IAT), impostazioni poi riutilizzate in tutti gli esperimenti. (Gli autori mostrano che togliere standardizzazione+clipping danneggia molto la novelty detection pur impattando poco l’accuracy di classificazione).

# Modello: architettura e perché è “multimodale”

Rete neurale con due rami: **Conv1D** per le sequenze di pacchetto (PSTATS) e **layer lineari** per le statistiche di flusso (FLOWSTATS); concatenazione, livelli fully-connected con **BatchNorm/Dropout/ReLU**, **~1.1M** di parametri. Le PSTATS sono a lunghezza fissa 30.

Gli autori confrontano anche **LightGBM** (stesso input) e trovano che la rete neurale è superiore per classificazione (e soprattutto più adatta alla novel class detection).

# “Reject option” e rilevazione dei servizi sconosciuti

Idea: calcolare per ogni flusso un **novelty score**; se supera una soglia (fissata al **95° percentile** sulla validation → ≈5% FPR), si rifiuta la predizione e si etichetta unknown. Tre strategie:

* **Softmax score** (negativo del max softmax) – semplice ma tende a essere troppo fiducioso out-of-distribution.
* **Energy-based** (LogSumExp dei logit) – implementazione banale, prestazioni migliori del softmax.
* **Gradient-based**: si fa backprop “a etichetta prevista corretta” e si misura la **norma del gradiente** sul penultimo layer. Gli autori lo **estendono** usando **SimLoss** (matrice di similarità fra servizi, es. fra servizi Facebook), così gli errori “intra-gruppo” producono gradienti più piccoli → meno falsi unknown (migliora TPR e AUROC).

Metriche per novelty: **TPR@5%FPR** e **AUROC** misurata solo per **FPR 0–10%**, perché in reti ad alta velocità non è accettabile un FPR elevato. Nota pratica: usando la soglia tarata sulla prima settimana, sulla seconda settimana l’FPR “effettivo” sale a ~**6.5%** → serve **ricalibrare periodicamente** (drift).

# Setup sperimentale e risultati

* **Split noti/sconosciuti**: prendono top-50, top-100, top-150 servizi come “noti” (per volume); **i servizi dello stesso gruppo** vanno tutti o nei noti o negli sconosciuti per evitare leakage. Cross-validation su dieci fold.
* **Classificazione (tutti noti)**: con **191 classi** ottengono **>97%** di accuratezza e **>98.5%** di accuratezza “superclasse” (cioè tollerando scambi interni al fornitore). L’abstract riporta **97.04%**.
* **Confronto ML**: LightGBM 93.32% acc. vs DL **97.41%**; F1 macro 90.09% vs **95.91%**; la rete è meglio anche a parità di feature.
* **Novel class detection**: sullo split **top-100 (91 sconosciute)**, il metodo **gradient-based+SimLoss** ottiene **TPR@5%FPR = 91.94%** e **AUROC ≈ 92.13%**; l’**energy** batte nettamente il softmax ed è proposto come **nuovo baseline** semplice e veloce.
* **Ablation**: togliere **packet times** e/o **direzioni** influisce poco sull’**accuracy**, ma cala la TPR per novelty di **>4%**; senza **standardizzazione+clipping** la TPR crolla di **>8%**; ridurre le sequenze a **20** pacchetti quasi non cambia, a **10** sì (coerente: il **75%** dei flussi ha ≤20 pacchetti). L’uso di **SimLoss** aggiunge ≈ **+1% TPR**.
* **Interpretabilità (XAI)**: con SHAP (o simili) mostrano che **le dimensioni dei pacchetti contano più di tutto**, e che **i primi pacchetti** (handshake + primi application data) pesano di più; per servizi specifici cambiano i “picchi” (es. nel Chrome Remote Desktop API conta tantissimo il **primo pacchetto**). Hanno anche un indizio sul **ClientHello**: un piccolo picco di frequenza attorno **500–550 byte**.

# Cosa significa per chi crea regole o metadati (non-ML)

Anche se il paper è ML-centrico, ci sono spunti **riutilizzabili in un approccio rule-based**:

* Concentrati sui **primi 20–30 pacchetti con payload**: sono quelli più informativi (handshake e primi scambi applicativi).
* Tra i metadati utili: **dimensioni dei pacchetti** (pattern), **direzione**, **tempi inter-arrivo**, **numero di round-trip**, **durata**; l’uso combinato aumenta la discriminabilità, specie per distinguere “sconosciuti”.
* Esistono **marker ricorrenti**: es. un picco di **ClientHello** attorno a 500–550 byte nelle distribuzioni globali; dimensione del **certificato** nelle prime decine di pacchetti; **pattern di round-trip** tipici immediatamente dopo l’handshake. Questi possono diventare **regole** o **feature per NDPI** alla “prima occorrenza”.
* **SNI non basta** (può sparire o essere falsificato); avere regole comportamentali basate su metadati ti rende **robusto** anche in scenari ECH.

# Limiti e note pratiche

* **Drift temporale**: la soglia per il reject tarata su una settimana non garantisce FPR identico nella successiva (sale a ~6.5%): in produzione serve **ricalibrazione periodica**.
* **Imbalance** tra servizi molto popolari e rari; accuracies macro e per superclasse aiutano a leggere meglio le prestazioni.
* **Split realistici**: quando separano noti/sconosciuti rispettano i **gruppi di servizi** per evitare artefatti (punto metodologico importante).

## In una frase

Gli autori pubblicano un grande dataset TLS etichettato via SNI e dimostrano che una rete **multimodale** sui **primi 30 pacchetti** + statistiche di flusso può **identificare servizi** con ≈**97%** di accuratezza e **rigettare** come sconosciuti nuovi servizi con **≈92% TPR a 5% FPR**, con metodi semplici (energy) o più potenti (gradienti+SimLoss), sottolineando che **dimensioni dei pacchetti e primissimi scambi** sono la chiave—informazione utile anche per costruire **regole deterministiche** alla prima occorrenza.

Se vuoi, ti aiuto a tradurre questi insight in un set concreto di **metadati/regole** (per esempio una checklist “8 pacchetti” in stile nDPI) per YouTube, Facebook, ecc., partendo proprio dai campi che qui risultano più informativi.

---

Perfetto, ti faccio una spiegazione approfondita e completa del paper **“Datasets are not Enough: Challenges in Labeling Encrypted Network Traffic”**, visto che è molto in linea con la parte della tua tesi che riguarda **l’etichettatura iniziale dei flussi**.

## 🎯 Obiettivo del paper

Gli autori vogliono far emergere un problema spesso sottovalutato: **anche se un dataset è grande, non significa che sia accurato o utile**, perché la parte davvero difficile (e costosa) è **ottenere etichette corrette** per il traffico di rete, specialmente quando è cifrato (TLS, QUIC, ecc.).

Molti lavori di classificazione del traffico danno per scontato che i dataset etichettati siano “oro colato”, ma nella pratica:

* la **fase di labeling** è piena di trappole,
* gli errori si propagano,
* e le stime di accuratezza dei modelli possono essere **sovrastimate**.

## 🧩 Perché il labeling è così difficile nel traffico cifrato

1. **TLS/QUIC nasconde il payload**, quindi non si può “vedere” direttamente cosa c’è dentro.
2. Restano disponibili solo **metadati**:
   * SNI (se non cifrato, ma ECH sta cambiando lo scenario),
   * certificati TLS,
   * dimensioni pacchetti, tempi, direzioni,
   * handshake pattern.
3. **Questi indizi non sempre bastano** a distinguere i servizi in modo certo, specialmente se:
   * un dominio serve più applicazioni (es. google.com → Google Search, Gmail, Drive),
   * o se un’app si appoggia a CDN comuni (Cloudflare, Akamai).

## 📦 Tecniche comuni di labeling (e i loro problemi)

Il paper analizza **quattro strategie principali** di labeling usate nei dataset di traffico cifrato e spiega per ognuna i limiti.

### 1. **DNS-based labeling**

* Idea: associare un flusso TLS al nome di dominio ottenuto dalla query DNS che lo ha preceduto.
* Problemi:
  + DNS caching: la query può essere stata fatta ore prima,
  + DNS-over-HTTPS/DNS-over-TLS non è intercettabile,
  + domini condivisi (CDN).

### 2. **SNI-based labeling**

* Idea: usare il Server Name Indication del Client Hello TLS.
* Problemi:
  + Alcune connessioni non hanno SNI (es. IP diretti),
  + ECH (Encrypted Client Hello) lo renderà invisibile,
  + Domini generici o multi-servizio.

### 3. **JA3/JA3S fingerprinting**

* Idea: associare l’impronta delle opzioni TLS (JA3) a un’applicazione nota.
* Problemi:
  + Cambiamenti frequenti nelle librerie TLS,
  + Collisioni tra app diverse,
  + Non sempre univoco per distinguere sotto-servizi.

### 4. **Controllo in ambiente strumentato**

* Idea: generare traffico in laboratorio da app note e catturarlo.
* Problemi:
  + Non rappresenta il traffico reale (dataset poco realistico),
  + Copertura limitata (non si possono catturare tutte le varianti).

## 🚨 Errori tipici di labeling

Gli autori mostrano che nei dataset pubblici ci sono spesso:

* **Etichette sbagliate** (un flusso di YouTube etichettato come Google Video, ecc.),
* **Classi troppo generiche** (tutto etichettato come “Google”),
* **Rumore** (flussi non correlati inclusi nella cattura di un’app),
* **Mancata sincronizzazione temporale** (es. un’etichetta di dominio assegnata a un flusso partito dopo ore).

## 🔍 Caso di studio: quanto influiscono gli errori

Loro hanno preso un dataset di traffico cifrato con etichette create via SNI e DNS, e hanno:

* ripulito manualmente le etichette,
* confrontato le performance di un classificatore prima e dopo la pulizia.

**Risultato**:  
Il modello sembrava avere un’accuratezza altissima con i dati sporchi, ma in realtà stava imparando rumore o scorciatoie.  
Con dati puliti, l’accuratezza reale era più bassa ma più realistica.

## 💡 Messaggio chiave per la tua tesi

Quello che sottolineano è proprio il problema che vuoi affrontare:

* La **prima etichettatura** (first occurrence labeling) è un **collo di bottiglia**.
* Serve un **processo assistito** (semi-automatico) che estragga **metadati utili** al riconoscimento e che aiuti un umano a confermare/associare il flusso all’applicazione.
* Senza una buona etichettatura iniziale, qualsiasi sistema (regole nDPI, ML, ecc.) rischia di essere basato su dati sbagliati.

## 📌 Cosa puoi prendere per il tuo lavoro

* La lista dei **metodi di labeling esistenti** e dei loro limiti è utilissima per motivare la tua ricerca.
* Puoi citare il fatto che **SNI-based e DNS-based labeling** sono fragili → giustifica il bisogno di un sistema che usi anche:
  + dimensioni pacchetti,
  + timing,
  + JA3/JA3S,
  + certificati.
* Il paper rafforza l’idea che **l’assistenza all’umano nella prima associazione** sia una sfida aperta e non risolta.

Se vuoi, posso estrarti **una tabella comparativa dei metodi di labeling** con vantaggi, svantaggi e possibili mitigazioni, così la puoi riusare nella tua tesi per spiegare perché serve il tuo approccio.  
Vuoi che la prepari?

---