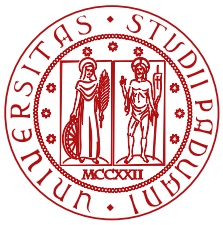

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

Machine Learning per Intrusion Detection in IoT: Una Recensione dello Stato dell'Arte

Relatore
Prof. Bellotto Nicola

Laureando
Antonutti Manuel

ANNO ACCADEMICO 2022-2023

Data di laurea 21 Novembre 2023

Indice

1 Introduzione **1**

2 Internet of Things e Sicurezza **3**

2.1 Definizione di IoT . . . 3

2.2 Applicazioni . . . 3

2.3 Attacchi informatici più comuni . . . 5

2.3.1 Brute Force . . . 5

2.3.2 Spoofing . . . 6

2.3.3 Denial of Service . . . 6

2.3.4 Information gathering . . . 7

2.3.5 MITM . . . 8

2.3.6 Malware . . . 9

2.3.7 Injection . . . 11

3 Intrusion Detection System **13**

3.1 Definizione di IDS . . . 13

3.2 Strategie di posizionamento dell'IDS nella rete . . . 13

3.3 Strategie di rilevamento delle intrusioni . . . 14

3.4 Estrazione di dati analitici dalla rete e dai dispositivi . . . 15

3.4.1 Analisi di processi e log di sistema . . . 15

3.4.2 Analisi del traffico di rete . . . 16

4 Machine Learning **19**

4.1 Definizione di machine learning . . . 19

4.2 Panoramica delle tecniche di machine learning . . . 19

4.3 Supervised learning . . . 20

4.4 Unsupervised learning . . . 26

4.5 Metriche di performance . . . 28

5 Stato dell'arte delle tecniche di ML in IDS **31**
5.1 Confronto tra dataset pubblici 31
5.2 Confronto tra le tecniche ML più utilizzate per IDS 35
5.3 Implementazioni di IDS su sistemi embedded 42
5.4 Maggiori sfide da affrontare degli IDS in ambito IoT 46
5.4.1 Problemi con i dataset 46
5.4.2 Problemi con il machine learning per IDS 47
5.4.3 Problemi nell'implementare IDS su dispositivi IoT 47
5.5 Direzioni future 48

6 Conclusioni **51**

Bibliografia **53**

Capitolo 1

Introduzione

L'Internet of Things (IoT) è un campo tecnologico in rapida crescita, con applicazioni in settori come la domotica, l'agricoltura, l'automazione industriale, la sanità e molti altri ancora. Il grafico in figura 1.1 mostra le previsioni per i prossimi anni di IoT Analytics (azienda che fornisce strategie di marketing e business focalizzate sull'industria 4.0), che sottolineano l'aumento di interesse delle imprese nelle tecnologie IoT, vedendo un probabile raddoppio della spesa globale già per il 2027.

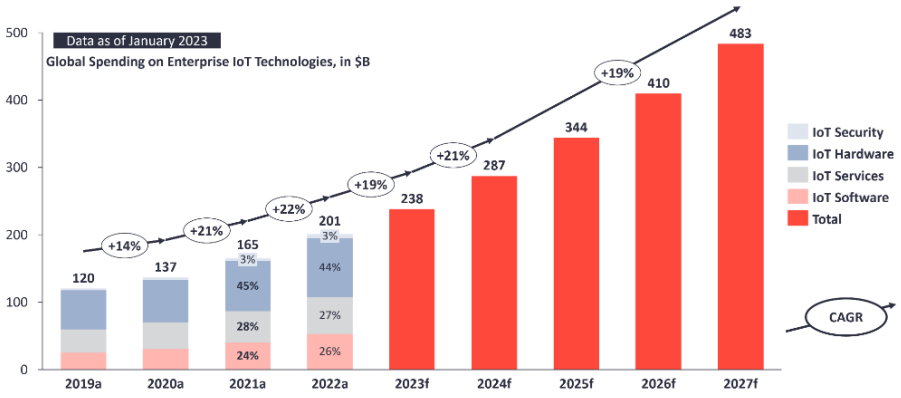

Figura 1.1: Spesa globale sulle tecnologie IoT per l'impresa [1]

Tuttavia, l'aumento di connettività verso la rete comporta una maggiore esposizione alle intrusioni e agli attacchi informatici, soprattutto quando i dispositivi non sono ben protetti, come spesso è il caso nelle reti IoT. Un rapporto di SonicWall (società americana di sicurezza informatica) suggerisce una crescita esponenziale degli attacchi malware a dispositivi IoT (figura 1.2). Il resoconto afferma che in Europa il volume di dispositivi compromessi è salito del 21% tra il 2021 e il 2022 e che le principali vittime sono le istituzioni pubbliche come i governi e la sanità.

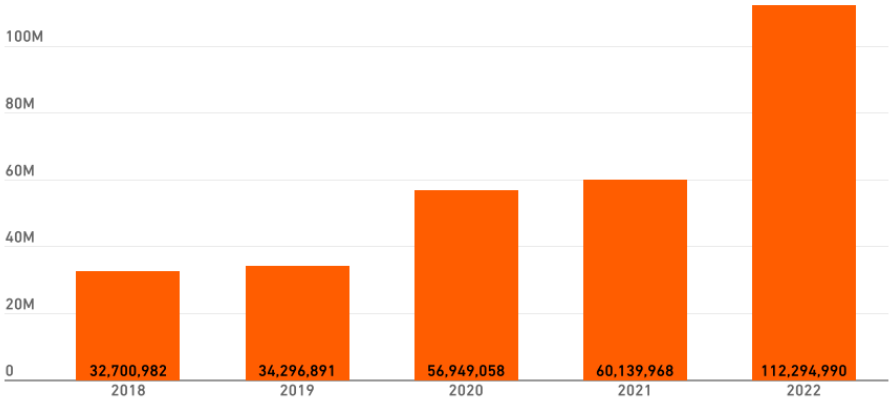

Figura 1.2: Attacchi malware a dispositivi IoT negli ultimi anni [2]

In questo elaborato, si vuole fornire una recensione delle attuali soluzioni basate sull'intelligenza artificiale e una descrizione dei problemi che devono ancora essere affrontati prima che queste tecniche possano essere impiegate su reti reali.

Nel capitolo 2, viene fornita un'introduzione all'IoT e alle sue applicazioni più popolari, oltre ad illustrare le minacce informatiche più comuni.

Nel capitolo 3, vengono introdotti i sistemi di rilevamento delle intrusioni, discutendo le tipologie di IDS e le informazioni che raccolgono per condurre le loro analisi.

Nel capitolo 4, viene fornita un'introduzione alle tecniche di machine learning ed una panoramica dei modelli utilizzati negli studi presi in considerazione da questa recensione.

Nel capitolo 5, vengono discussi alcuni dei dataset più conosciuti per lo sviluppo dei sistemi di rilevamento. Successivamente, vengono comparati i risultati di vari elaborati nell'applicare algoritmi di apprendimento automatico (sia classici che moderni) al problema. Dopodichè, vengono presentati tre studi che descrivono la loro implementazione di un IDS su sistemi embedded, dotati di risorse limitate. Ed infine vengono discusse le problematiche principali che devono ancora essere risolte dalla ricerca, nonché le possibili direzioni future che possono essere considerate.

Capitolo 2

Internet of Things e Sicurezza

2.1 Definizione di IoT

L'Internet of Things, comunemente abbreviato IoT, è un espressione coniata da Kevin Ashton nel 1999 con cui ci si riferisce al processo di estensione della rete internet verso dispositivi e oggetti di uso quotidiano. Questa connettività permette loro di raccogliere e condividere informazioni, rendendoli più interattivi e intelligenti. L'obiettivo principale è quello di creare una network di oggetti univocamente identificati e rintracciabili con la capacità di comunicare sia reciprocamente che verso punti nodali [3].

2.2 Applicazioni

Le possibili applicazioni dell'internet delle cose al giorno d'oggi sono molteplici. Si tratta di una tecnologia che coinvolge diversi ambiti, tra i quali i più rilevanti sono:

- Smart Farming: impiego di sensori e sistemi di controllo remoti per monitorare le condizioni di umidità e temperatura del suolo in tempo reale, supervisione a distanza degli animali e utilizzo intelligente dei macchinari agricoli. L'IoT può aiutare a rendere più efficienti le aziende agricole, sia ottimizzando il processo produttivo al fine di evitare sprechi che automatizzando i compiti più semplici e ripetitivi (come l'irrigazione dei campi).

- Smart Home: introduzione di dispositivi smart nella propria abitazione con cui è possibile regolare automaticamente la temperatura, accendere e spegnere le luci, avviare elettrodomestici, aprire il cancello o il garage da remoto e così via. L'impiego dell'internet delle cose nel settore della domotica può diminuire i consumi, semplificare la gestione delle attività quotidiane ed aumentare la sicurezza (sia da un punto di vista delle intrusioni per-

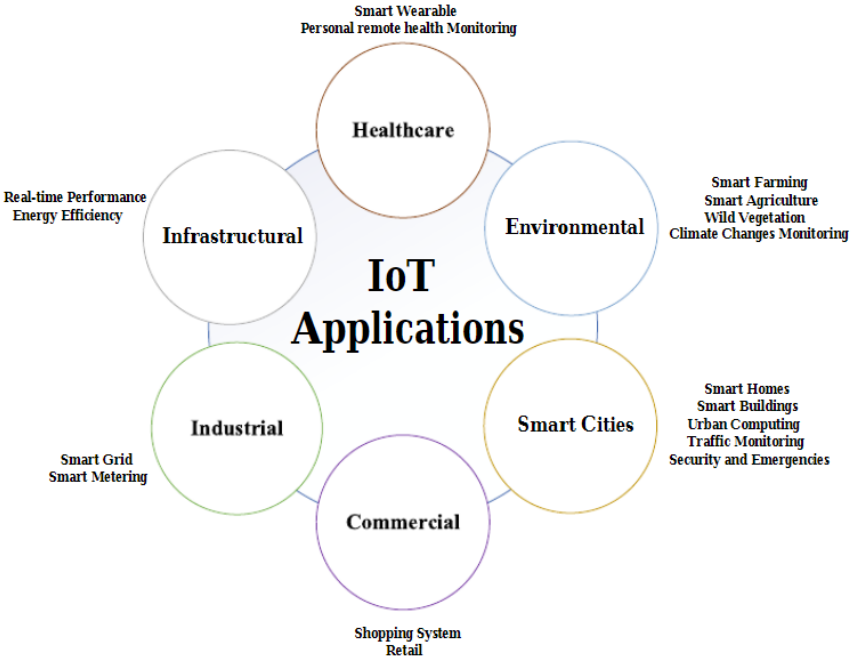

Figura 2.1: Tassonomia delle applicazioni dell'IoT [4]

chè la smart home può notificare al proprietario lo stato dell'antifurto, ma anche nel caso di incendi e altri situazioni anomale).

- Smart Health: utilizzo di dispositivi portatili che monitorano la salute dei pazienti in tempo reale (come lo smart watch, la smart band e il fitness tracker), oppure di strumenti che osservano in automatico lo stato dei pazienti negli ospedali e rilevano anomalie nei parametri vitali.

- Smart City: impiego di sensori e infrastrutture per monitorare il traffico di mezzi in città (con lo scopo di ottimizzare la funzione dei semafori), per controllare i livelli di inquinamento dell'aria e dell'acqua, per migliorare i sistemi di approvvigionamento idrico e di smaltimento dei rifiuti e per implementare sistemi di illuminazione intelligente.

- Smart Grid: utilizzo di una rete di informazione e una rete di distribuzione dell'energia elettrica in un sistema decentralizzato con lo scopo di fornire energia in modo più efficiente. L'IoT in questo settore permette di monitorare le risorse, i consumi e le necessità degli utenti per rispondere di conseguenza.

- Smart Manufacturing: impiego di sensori e sistemi di controllo per monitorare, analizzare e intervenire sui processi di un industria.

- Smart Mobility: si riferisce a tutte le soluzioni intelligenti per la mobilità, principalmente in città, tramite servizi di noleggio e condivisione di auto, bici e monopattini elettrici, nonchè servizi di smart parking e reti di ricarica.

2.3 Attacchi informatici più comuni

L'aumento di connettività introdotto dall'Internet of Things comporta un'espansione della superficie di attacco dei dispositivi interessati. Le vulnerabilità da considerare non sono più confinate solo alla rete interna, ma bisogna considerare anche minacce esterne provenienti dal cloud. Gli attacchi informatici più frequenti alle reti IoT sono i seguenti.

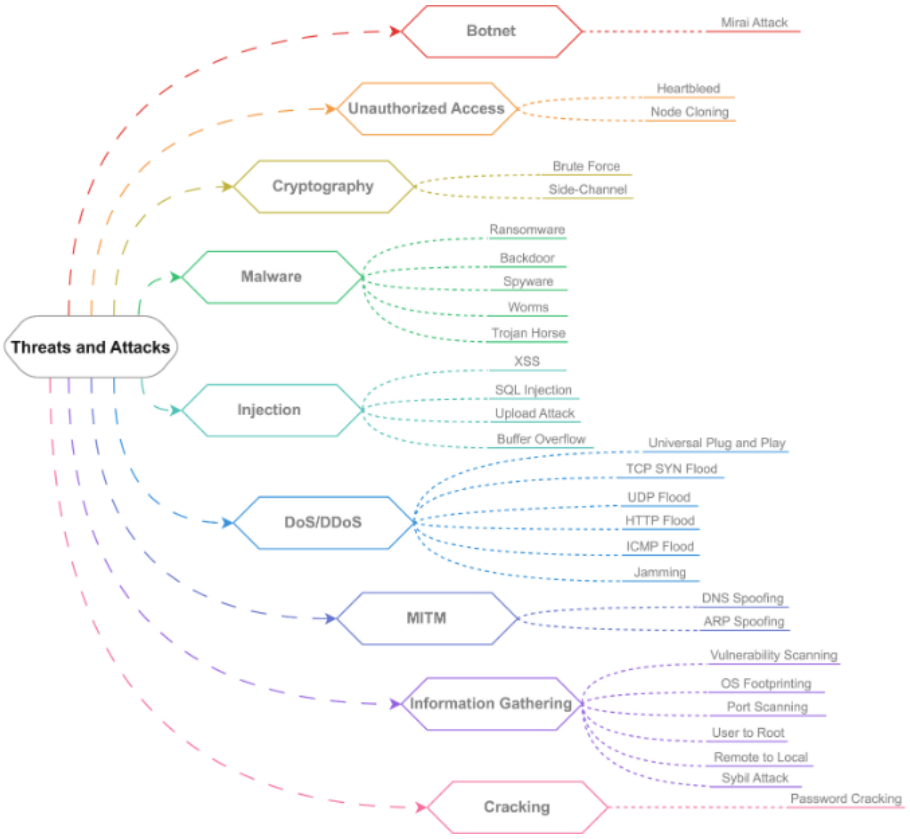

Figura 2.2: Categorie di attacchi informatici nell'IoT [5]

2.3.1 Brute Force

Questa tecnica, nota come attacco di forza bruta, viene utilizzata per acquisire le credenziali delle vittime. L'attacco consiste nel tentare varie combinazioni di utente e password in rapida successione. L'obiettivo è indovinare correttamente le credenziali dell'utente, sfruttando la velocità di immissione dei calcolatori moderni. Questo metodo, sebbene semplice, può rivelarsi

estremamente efficace, soprattutto quando le vittime utilizzano password deboli o facilmente indovinabili. Le parole possono essere generate in maniera sequenziale, aggiungendo un carattere alla volta, oppure possono essere estratte da un dizionario di password comuni. Esistono, inoltre, metodi ibridi che generano credenziali nuove da parole predefinite.

Il malware Mirai [6] è noto per l'utilizzo di una tecnica di attacco basata sulla forza bruta [7] per individuare le credenziali di accesso di dispositivi connessi all'internet, in particolare camere IoT e router, che espongono la porta telnet. Nonostante la semplicità di questo metodo, esso risulta ancora efficace nell'infezione di nuovi dispositivi IoT, come evidenziato dal report [8] di Kaspersky del 2023. Secondo il report, il protocollo telnet è il principale obiettivo di questi attacchi, con circa il 97% di tentativi nella prima metà dell'anno.

2.3.2 Spoofing

Si tratta di una tecnica usata dall'attaccante per mascherare la propria identità e fingersi qualcun altro. Può essere applicata a vari livelli della pila ISO-OSI ed è spesso usata in congiunzione con altri metodi per creare attacchi più sofisticati. Lo spoofing, in molte delle sue forme, è di facile implementazione, perchè non richiede una grande competenze tecnica. Alcuni esempi sono i seguenti [9]:

- ARP Spoofing: l'attore malintenzionato invia un messaggio ARP corrotto sulla rete locale presentando il suo indirizzo MAC e associandolo all'indirizzo IP di un altro utente. Quando gli altri dispositivi vogliono comunicare con l'utente legittimo, finiscono per mandare i messaggi all'hacker. Questa tecnica funziona solo sulle reti locali.

- IP Spoofing: L'attore malintenzionato modifica il campo IP sorgente degli header di rete e di trasporto per sembrare un altro dispositivo. In questo caso, l'obiettivo non è catturare i messaggi di un altro utente, ma rimanere anonimo.

- Email Spoofing: l'attore malintenzionato assume l'indirizzo email di un utente legittimo al fine di ingannare la vittima. Questa tecnica fa uso di ingegneria sociale per carpire informazioni sensibili o per richiedere denaro.

2.3.3 Denial of Service

Un attacco Denial of Service (DoS) punta a rendere inaccessibili le risorse e i servizi di qualche sistema, rete o applicazione per un periodo di tempo indeterminato. L'aggressore sovraccarica la vittima con richieste o messaggi, rallentando e talvolta arrestando la macchina bersagliata. Bloccare questa minaccia può essere difficile, perchè l'attaccante può far uso di spoofing e rendersi quasi indistinguibile dagli utenti legittimi. Un attacco DDoS, abbreviazione di Distributed

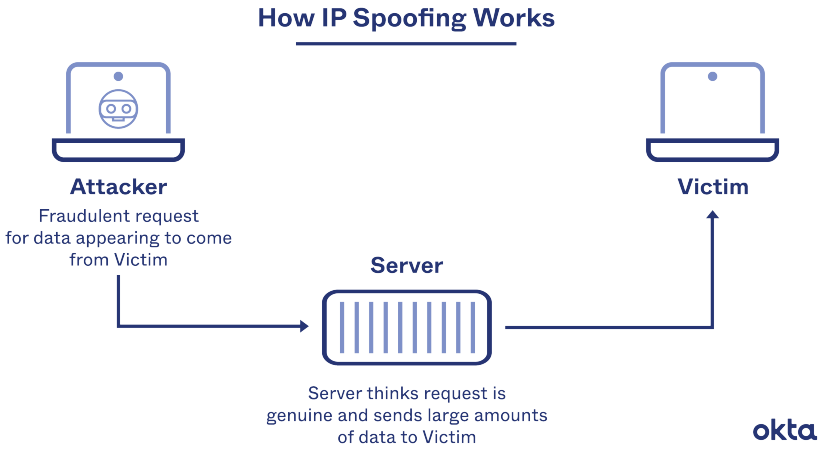

Figura 2.3: Attacco IP Spoofing [10]

Denial of Service, rappresenta una versione molto più potente e avanzata del DoS tradizionale, in quanto si avvale di una moltitudine di dispositivi per condurre l'attacco. Esistono molte varianti di questo attacco che si basano sempre sullo stesso approccio [5].

- SYN Flood: consiste nell'invio di pacchetti di sincronizzazione (con il flag TCP SYN attivo) alla vittima, nella speranza di sovraccaricare le code di connessione e rendere il sistema incapace di rispondere alle richieste legittime. In altre parole, la vittima alloca all'ascolto tante porte di rete quante le richieste di connessione e, una volta esaurite le porte libere, gli utenti legittimi non possono usufruire del servizio.
- UDP Flood: a differenza del SYN Flood, questo attacco non mira a consumare le risorse del server, bensì a sovraccaricarlo con una mole ingestibile di traffico in entrata.
- HTTP Flood: come l'UDP Flood, anche l'HTTP Flood bombarda la vittima con grandi quantità di dati in ingresso. Agisce al livello di applicazione della pila ISO-OSI e ha come obiettivo l'interruzione del servizio web del server bersagliato.
- Jamming: si tratta di una tecnica molto diversa dalle precedenti perchè opera sul mezzo di comunicazione utilizzato dai dispositivi wireless. L'aggressore va a interferire con il segnale radio della vittima (aggiungendo rumore sulla banda della trasmissione), rendendo impossibile lo scambio di dati.

2.3.4 Information gathering

Si tratta spesso della prima fase di un attacco più grande e consiste nel raccogliere informazioni sul sistema da compromettere. Si può implementare in maniera passiva, andando a studiare le

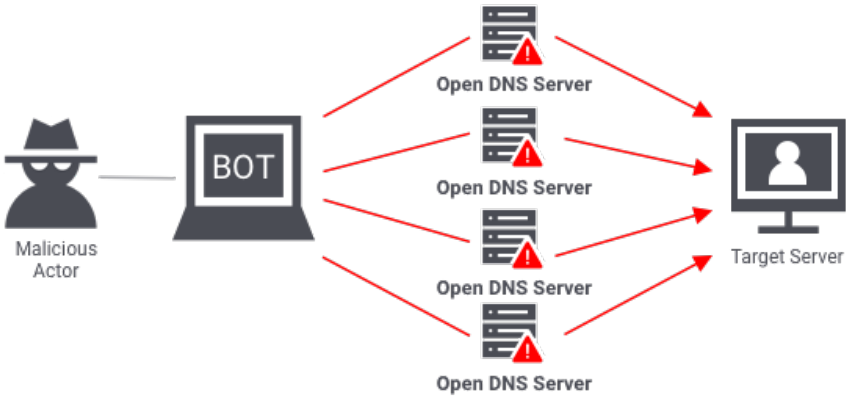

Figura 2.4: Attacco DDoS [11]

informazioni pubblicamente visibili (come ad esempio siti web o documentazione online), oppure in maniera attiva, andando a impiegare strumenti più invasivi come la cattura dei pacchetti in transito sulla rete o la scansione delle porte aperte di una macchina. Di seguito sono riportati alcuni esempi [5].

- Port scanning: consiste nel sondare le porte in ascolto di una macchina alla ricerca dei servizi attivi e delle vulnerabilità del sistema. Alcuni degli strumenti più utilizzati per il port scanning sono Nmap e Hping3 [12].

- Remote to local: si tratta di un attacco in cui un aggressore esterno cerca di ottenere accesso a un sistema o ad una rete locale. Si possono impiegare metodi di forza bruta, ma anche di ingegneria sociale.

- User to Root: comprende tutte le tecniche che cercano di carpire le credenziali d'accesso dell'amministratore di un sistema. L'obiettivo è quello di acquisire maggiori privilegi sulla macchina bersagliata per poi eseguire altri tipi di attacchi più avanzati.

2.3.5 MITM

Questo tipo di attacco (Man In The Middle) prevede l'uso di un dispositivo che fa da tramite tra la vittima e la risorsa di rete. L'attaccante si inserisce in mezzo alla comunicazione con lo scopo di monitorare i pacchetti scambiati tra gli host ed estrarre informazioni sensibili. Questa tecnica è spesso più complessa rispetto ad altri tipi di attacco, come DoS o BruteForce [13]. Il 95% dei server che utilizzano il protocollo HTTPS sono vulnerabili ad attacchi primitivi di tipo MITM e, considerando che la maggior parte dei dispositivi IoT comunica in chiaro, senza alcuna forma di crittografia, si espongono i dati a possibili violazioni [14].

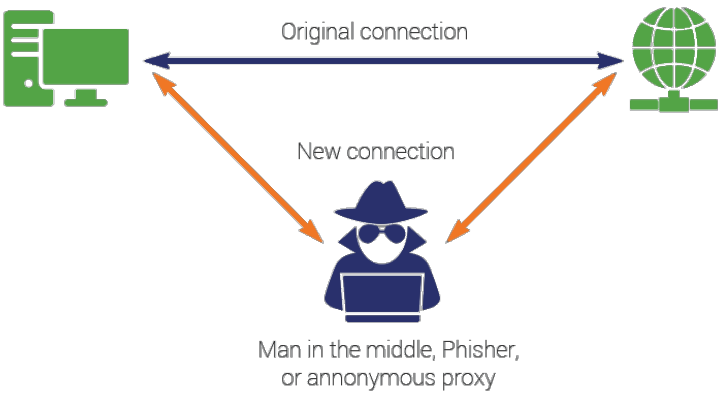

Figura 2.5: Attacco MITM [15]

2.3.6 Malware

Il malware è un termine generico per descrivere tutti i software malevoli progettati per causare danni o disturbi a computer e altri dispositivi. Il malware può avere diversi obiettivi, come rubare informazioni, danneggiare sistemi, spiare la vittima, criptare i file, ottenere il controllo del dispositivo e molti altri ancora [16]. A seguire si elencano alcuni tipi:

- Ransomware: è una forma di malware che utilizza strumenti crittografici per codificare i file di un computer, rendendo impossibile l'apertura e la lettura degli stessi. In cambio della chiave di decifrazione, viene richiesto alla vittima di pagare un riscatto.

  Nel 2021, la regione Lazio venne colpita da un attacco ransomware che bloccò il sistema di prenotazione dei vaccini Covid-19. Un articolo di Wired [17] ha rivelato che il costo per ripristinare i servizi informatici dopo l'incursione del malware è stato di circa 167.000 euro. L'appalto venne aggiudicato a Microsoft.

- Trojan: è una forma di malware che si presenta come software legittimo per ingannare l'utente ad eseguirlo. Una volta attivato, può rubare dati sensibili, installare programmi in background e spiare la vittima.

- Backdoor: è una variante del trojan che permette all'aggressore di accedere al sistema da remoto senza doversi autenticare.

- Worms: sono tipi di malware che si replicano e diffondono autonomamente sulla rete, sfruttando le vulnerabilità dei sistemi o utilizzando tecniche di phishing. Possono essere impiegati per vari scopi, ma principalmente servono a propagare malware.

- Botnet: è una forma di malware che infetta una rete di dispositivi per prenderne il controllo. L'utilizzo primario di una botnet è quello di portare avanti attacchi DDoS sfruttando la grande quantità di bot al suo comando.

  Nel 2016, la botnet Mirai [6] venne impiegata per effettuare attacchi DDoS a vari bersagli, tra cui Dyn, un provider di servizi DNS. Tra le compagnie che subirono un interruzione dei servizi durante questo attacco si registrarono Twitter, Paypal, Amazon, Netflix e molte altre [18]. Questa botnet aveva caratteristiche simili ad un worm nel modo di diffondersi e sfruttava le vulnerabilità delle porte telnet. Nelle prime 20 ore dal suo rilascio sulla rete aveva già infettato 65.000 dispositivi IoT, per poi giungere ad uno stato di equilibrio di circa 200.000 bot. A quel punto Mirai era capace di effettuare attacchi DDoS su larga scala ed utilizzava diverse tecniche tra cui HTTP Flood, UDP Flood e SYN Flood.

Figura 2.6: Categorie di Malware [19]

2.3.7 Injection

Gli attacchi di questa categoria ”iniettano” codice malevolo in siti web e applicazioni legittime. L’obiettivo è quello di far eseguire il codice sui dispositivi di chi utilizza i servizi e i software compromessi. Alcuni esempi di questa tecnica sono riportati di seguito:

- SQL Injection: si tratta di una vulnerabilità dei siti web che permette a un attaccante di interferire con le query che un’applicazione effettua verso il suo database. Questo tipo di attacco consente all’aggressore di visualizzare dati ai quali normalmente non avrebbe accesso. Un attaccante può inserire o “iniettare” codice SQL malevolo all’interno delle caselle di input per essere parte di una query SQL. Se l’applicazione non filtra o non gestisce correttamente questi input, l’attaccante può manipolare le query per eseguire operazioni non autorizzate, come accedere a dati sensibili o modificare record.

  Nel 2018 è stata rilevata una vulnerabilità SQL Injection in Cisco Prime License Manager che consentiva agli aggressori di ottenere l’accesso al terminale dei sistemi su cui era installato il gestore di licenze [20].

- XSS: è l’acronimo di Cross-Site Scripting e consiste in una vulnerabilità dei siti Web poco protetti con cui un malintenzionato riesce a ”iniettare” uno script dannoso nella pagina, che viene quindi incluso nel contenuto dinamico del sito inviato al browser della vittima. Il browser non ha alcun modo di sapere che il codice non sia attendibile e quindi lo esegue [21].

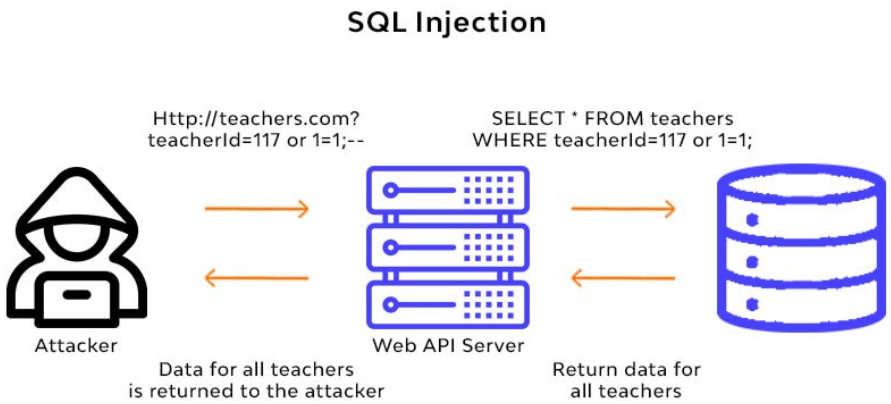

Figura 2.7: Attacco SQL Injection [22]

Capitolo 3

Intrusion Detection System

3.1 Definizione di IDS

L'Intrusion Detection System (IDS) è un sistema composto da software applicativi, e talvolta anche da dispositivi hardware, con lo scopo di rilevare accessi non autorizzati e traffico malevolo nella rete in cui è disposto. Un IDS può aiutare ad accelerare e automatizzare il processo di rilevamento delle minacce, allertando l'amministratore della sicurezza in tempo reale. Le analisi prodotte dall'IDS possono essere collezionate da un SIEM (Security Information and Event Management system) e conglobate con altre fonti per formulare un quadro generale da sottoporre agli analisti della sicurezza. Un IDS capace di intercettare e terminare le comunicazioni in tempo reale è detto Intrusion Prevention System (IPS). Si differenzia da un firewall perchè è disposto all'interno della rete e non al perimetro. Gli IDS/IPS e i firewall sono dispositivi complementari, in quanto i primi offrono una difesa aggiuntiva contro le minacce che superano i secondi [23].

3.2 Strategie di posizionamento dell'IDS nella rete

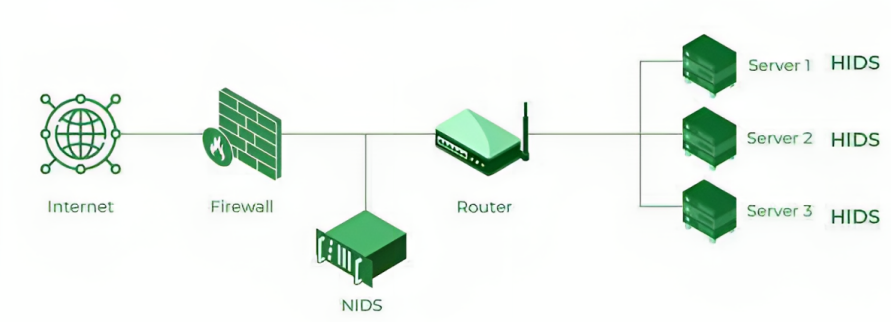

Figura 3.1: Posizione di HIDS e NIDS in una rete LAN [24]

Un sistema di rilevamento delle intrusioni può essere collocato all'interno di una rete LAN in vari modi. Di solito si tende a scegliere tra l'architettura Host-based o quella Network-based, ma recentemente vari studi stanno considerando architetture distribuite come alternative, anche se si tratta di un'area di ricerca ancora aperta [25][26][27].

- HIDS: la prima strategia è chiamata Host-based Intrusion Detection System (HIDS) perché l'IDS viene installato su un host della rete con l'obiettivo di monitorare i processi in esecuzione (alla ricerca di malware), analizzare i log di sistema e il traffico sulla scheda di rete.
- NIDS: un'alternativa all'HIDS è il Network-based Intrusion Detection System (NIDS) che viene collocato all'interno di una rete, spesso come dispositivo a sé stante, con lo scopo di intercettare i pacchetti condivisi tra i nodi della rete, esaminarli e determinare se sono parte di un attacco informatico.
- DIDS: l'ultima strategia da considerare è quella del DIDS, acronimo di Distributed Intrusion Detection System, che consiste in una soluzione ibrida tra HIDS e NIDS. Fondamentalmente, ognuno dei dispositivi host è dotato di un HIDS che comunica le sue analisi ad un server, mentre gli altri nodi della rete (come switch e router) fungono da NIDS per monitorare il traffico dei messaggi. L'implementazione di questo sistema è più complessa rispetto alle alternative perchè richiede di intervenire sull'intera rete da proteggere.

3.3 Strategie di rilevamento delle intrusioni

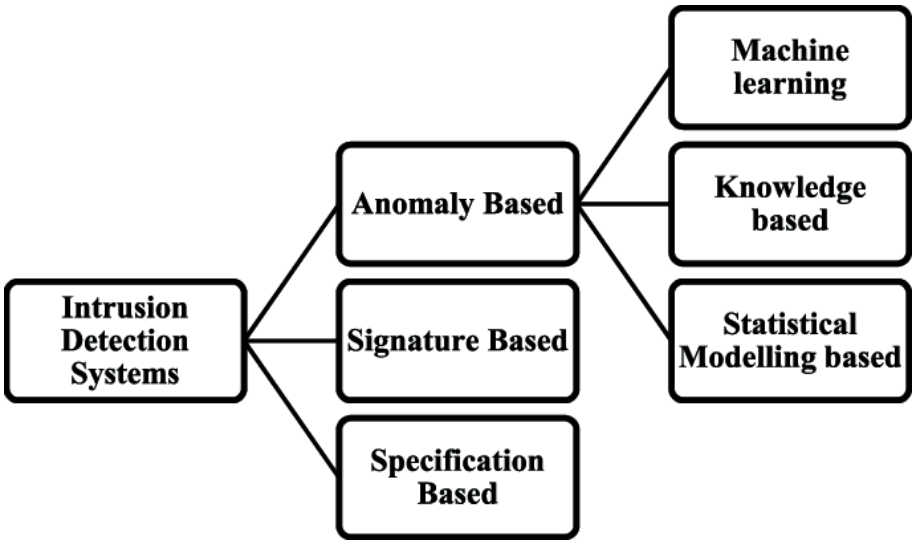

Figura 3.2: Tassonomia delle tecniche di rilevamento delle intrusioni [28]

Un IDS può utilizzare diversi approcci per rilevare intrusioni, alcuni di questi sono:

- Signature-based detection: questo approccio si basa sull'uso di un database contenente migliaia di firme di noti attacchi informatici. Il traffico viene comparato con le varie firme contenute nel database e nel caso in cui ci sia una corrispondenza, viene dato l'allarme. Questa strategia è attualmente la più utilizzata perché presenta un'alta efficacia nel rilevare gli attacchi più comuni, tuttavia si rivela fallimentare nel riconoscere nuovi metodi di intrusione e attacchi conosciuti la cui firma è stata cambiata drasticamente.
- Anomaly-based detection: questo metodo si basa sull'apprendere le caratteristiche che compongono il traffico normale nella rete in cui è applicato per poi identificare comportamenti anomali. Le tecniche di ML rientrano in questa categoria. Questo approccio permette di individuare attacchi informatici mai visti prima (chiamati zero day), ma presenta una più alta percentuale di falsi positivi, quindi una maggiore probabilità di errore. Infatti, qualsiasi attività sulla rete che si discosta dal modello di riferimento viene etichettata come pericolosa, anche se si dovesse trattare di un'operazione innocua.
- Specification-based detection: prevede la definizione di una lista di specifiche che rappresentano il normale comportamento dei dispositivi. Questa tecnica richiede la presenza di un esperto che conosca molto bene il funzionamento della rete. Inoltre, una volta definite le caratteristiche, il sistema risulta essere molto statico e poco adatto a reti variabili.
- Hybrid-based detection: quest'ultimo approccio prevede l'utilizzo combinato di diverse tecniche di rilevamento. Il vantaggio consiste in un miglioramento della precisione di identificazione, ma presenta maggiori costi dal punto di vista computazionale. Le più recenti soluzioni IDS applicano questa strategia [29].

3.4 Estrazione di dati analitici dalla rete e dai dispositivi

Per distinguere tra un'intrusione nel sistema e la normale attività dei dispositivi di una rete, l'IDS necessita di estrarre informazioni dal sistema con lo scopo di elaborarle. A seconda che l'IDS sia installato su un dispositivo della rete (Host-based IDS) o sia un dispositivo autonomo collegato alla rete (Network-based IDS), le fonti di dati possono variare. Nel primo caso, si possono usare i log di sistema e le informazioni sui processi in esecuzione sull'host. Nel secondo caso, si possono usare i dati provenienti dal traffico di rete.

3.4.1 Analisi di processi e log di sistema

Nel caso di Host-based IDS, è possibile analizzare i log di sistema e il comportamento dei processi in esecuzione, oltre che monitorare il traffico sulle porte dell'host. Di conseguenza è possibile controllare l'accesso degli utenti ai file, le modifiche ai file di sistema, i tentativi di accesso

non autorizzato, le modifiche alle credenziali utente e ai registri di sistema [30]. Questo tipo di analisi dipende molto dall'host su cui è installato l'IDS e per questo risulta difficile trovare standard comuni per dispositivi con caratteristiche diverse. In generale, esistono soluzioni per sistemi operativi come Linux, Windows o Mac, ma per dispositivi IoT che utilizzano Contiki NG o FreeRTOS le opzioni sono limitate.

3.4.2 Analisi del traffico di rete

Per analizzare il traffico di rete, l'IDS deve catturare i pacchetti condivisi tra gli host [31]. Spesso questi sono connessi tramite un dispositivo, come uno switch o un router, che si occupa dell'instradamento dei pacchetti. Una tecnica comunemente utilizzata per far arrivare i messaggi all'IDS è il port mirroring, che usa uno switch per inviare i pacchetti sull'interfaccia collegata al sistema di rilevamento, ripetendo quelli che arrivano alle altre porte. In questo modo, l'IDS può monitorare tutto il traffico di rete senza interferire con esso. Tuttavia, l'interfaccia dell'IDS deve essere in grado di gestire lo stesso volume di messaggi di tutte le altre porte messe assieme. Questo può essere un problema se la rete è molto congestionata, perchè si rischia di perdere dei pacchetti. Una volta catturati i pacchetti, si possono ricavare dati sugli host (indirizzi IP e porte del mittente e del destinatario), dati sulla comunicazione (il protocollo in uso, la durata della connesione) e dati statistici sui messaggi (la dimensione e la frequenza dei pacchetti, il numero di byte trasmessi). Un IDS dovrebbe idealmente essere capace di analizzare i pacchetti per intero, così come vengono inviati, ma spesso si ritrova a non avere la capacità computazionale o la memoria necessarie a svolgere questo compito. DI seguito sono riportati alcuni formati utilizzati per l'analisi del traffico di rete.

- File PCAP: il Packet Capture (PCAP) è un formato con cui vengono salvati i pacchetti catturati da software di packet sniffing (come WireShark, TCPdump, Zeek o SolarWinds) [32]. Le informazioni contenute nei pacchetti sono salvate integralmente (sia il payload che gli headers). Di conseguenza questo tipo di file risulta spesso essere molto grande, con ovvi svantaggi dal punto di vista delle prestazioni.

- Formato Flow: il formato Flow si riferisce in maniera generale a tutti i file prodotti da programmi di analisi del traffico di rete che sintetizzano le conversazioni tra dispositivi in una serie di attributi [33]. Questi attributi possono essere estratti dai pacchetti oppure essere calcolate come statistiche sul flusso di dati. Un esempio di formato Flow è il protocollo NetFlow, nato per risolvere il problema del consumo di memoria e tempo introdotto dall'elaborazione di file PCAP. Così, negli anni '90, Cisco sviluppò il protocollo NetFlow che consentiva di riassumere la conversazione di due dispositivi in qualche decina di attributi standard (come gli indirizzi IP del mittente e del destinatario, i byte in

ingresso ed uscita, le flags TCP utilizzate, il massimo e il minimo valore di TTL visto, etc). Nel 2013, l'IETF sviluppò il protocollo IPFIX sulla base della nona versione di NetFlow, introducendo la possibilità di esportare le informazioni del flusso di rete in formati personalizzati. Il vantaggio è una maggiore flessibilità nella raccolta dei dati.

Capitolo 4

Machine Learning

4.1 Definizione di machine learning

Il machine learning è un sottoinsieme dell'intelligenza artificiale da anni al centro dell'attenzione nell'ambito tecnologico. In generale, l'apprendimento automatico è un insieme di tecniche con cui un computer impara dai propri errori, con l'obiettivo di associare i dati forniti in input all'output desiderato. Nella pratica, il calcolatore utilizza un modello matematico per approssimare la funzione che permette questa associazione. Dal risultato del modello si può calcolare la loss function (funzione che misura l'errore commesso dal computer). Il processo di apprendimento consiste nella ricerca di un punto di minimo per la loss function, che risulta ottimizzare la precisione del modello. L'obiettivo del machine learning è quello di giungere ad uno stato in cui l'algoritmo permette di generalizzare correttamente il problema, rispondendo al meglio alle future istanze di test [34].

4.2 Panoramica delle tecniche di machine learning

Le tecniche di apprendimento automatico permettono agli elaboratori di svolgere mansioni sempre più complesse senza essere esplicitamente programmati per farlo. Sebbene la popolarità di questo ambito sia esplosa solo negli ultimi decenni, le origini dell'intelligenza artificiale risalgono al secolo scorso [35]. Le motivazioni che all'epoca rallentarono lo sviluppo in questo campo sono da attribuire alla mancanza di dati e all'insufficiente potenza computazionale. Al giorno d'oggi, l'avanzamento dell'elettronica e il fenomeno del big data hanno accelerato la diffusione del machine learning.

Il machine learning si divide in tre principali categorie: apprendimento supervisionato (supervised learning), apprendimento non supervisionato (unsupervised learning) e apprendimento per rinforzo (reinforcement learning). Inoltre, è bene sottolineare che le tecniche di apprendi-

mento profondo (deep learning) sono forme avanzate di machine learning, per cui si suddividono anch'esse nelle tre classi sopracitate [36]. In questo elaborato saranno presentati esempi di algoritmi facenti parte delle prime due classi perchè sono quelli principalmente utilizzati nello sviluppo degli IDS.

4.3 Supervised learning

L'apprendimento supervisionato è una sottocategoria dell'apprendimento automatico che comprende tutti gli algoritmi che necessitano di dati etichettati (ovvero samples a cui viene assegnato un valore di riferimento, che rappresenta l'output desiderato).

Le due principali categorie di problemi del supervised learning sono la classificazione e la regressione. La prima consiste nell'assegnare gli input a delle categorie predefinite, che possono essere due (classificazione binaria) o più (classificazione multiclasse). La regressione, invece, riguarda i problemi in cui l'output non è discreto, ma continuo, dove quindi assume valori reali.

Siccome il rilevamento delle intrusioni è un problema di classificazione, in questo elaborato non si prende in considerazione alcun algoritmo per la regressione. Di seguito sono riportate le tecniche di classificazione più comuni.

- Logistic Regression: la regressione logistica (LR) è una tecnica con cui le variabili in ingresso vengono combinate linearmente e al risultato viene applicata la sigmoid function.

- Support Vector Machine: questo algoritmo (SVM) costruisce un iperpiano che usa per separare i campioni di classi diverse [37]. La posizione dell'iperpiano viene determinata in base alla distanza dai punti più vicini delle due classi. L'ipotesi è che selezionando un iperpiano che massimizzi la distanza dai punti delle due classi si possa ottenere un errore di generalizzazione inferiore. Nel caso in cui i punti non siano linearmente separabili si può definire una funzione su misura (chiamata kernel) per mappare i campioni ad uno spazio dimensionalmente più grande dove si possa effettuare la divisione.

- Naive Bayes: questo modello (NB) è un classificatore probabilistico basato sulla probabilità condizionata e sul teorema di Bayes. Calcola il prodotto delle probabilità individuali delle caratteristiche della rete bayesiana dati gli input e determina la probabilità delle classi di output. Dopodichè seleziona quella con il valore maggiore.

- k-Nearest Neighbors: questa tecnica (kNN) supervisionata non ha bisogno di addestramento. Il suo funzionamento è molto semplice: si sceglie un valore k positivo, la classe di appartenenza di un nuovo punto è derivata per voto di maggioranza dai k punti più vicini. Il costo computazionale dell'algoritmo dipende dal numero di punti utilizzati nella

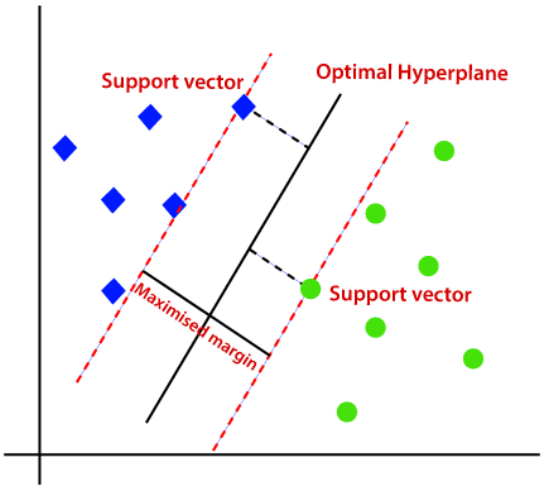

Figura 4.1: Iperpiano di decisione di una SVM [38]

sua inizializzazione, perchè deve confrontare la distanza dei nuovi campioni con il set di punti memorizzati.

- Decision Tree: l'albero di decisione (DT) è una rappresentazione di una funzione che mappa un vettore di valori in input ad una decisione. L'albero è spesso implementato usando una struttura ad albero binario in cui ogni nodo rappresenta una condizione if da soddisfare [39]. La classificazione è conclusa quando si raggiunge una foglia, in cui si trova il numero della classe di appartenenza. L'albero viene costruito scegliendo ricorsivamente l'attributo più significativo per ogni nuovo sotto-albero. La scelta avviene utilizzando il coefficiente IG (Information Gain) oppure l'indice di Gini. Questo algoritmo è da preferire quando l'input è composto da attributi ben strutturati, ovvero che presentano un chiaro e distinto significato (come nel caso delle features che descrivono un flusso di traffico di rete). Non è una buona scelta quando in input vengono forniti dati non strutturati (come immagini, audio, video, testo).

- Random Forest: questo algoritmo (RF) è una ensemble (insieme) di alberi di decisione [41]. Gli alberi tra di loro sono diversi perchè vengono costruiti aggiungendo un elemento di casualità. Praticamente, la scelta dell'attributo per il nuovo sotto-albero avviene tra un sotto-insieme scelto in maniera stocastica degli attributi rimasti. La foresta casuale è molto meno suscettibile all'overfitting rispetto al singolo albero di decisione.

- Perceptron: il percettrone simula il funzionamento di un neurone, prendendo la combinazione lineare degli input e applicandovi una funzione di attivazione, spesso a gradino.

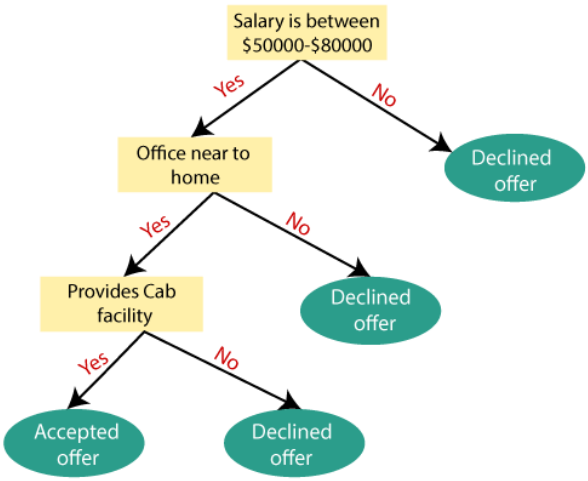

Figura 4.2: Albero di decisione [40]

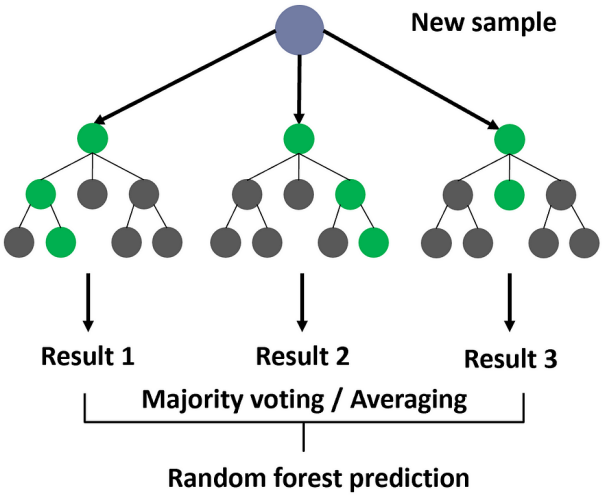

Figura 4.3: Foresta casuale [42]

- Multi-layer Perceptron: questo è un algoritmo (MLP) che si ispira al funzionamento del cervello umano. Si compone di nodi interconnessi, chiamati neuroni, che si dispongono a strati. Ogni neurone si comporta come un percettrone e passa l'output ai nodi dello strato successivo. Ha uno strato di input e uno di output, mentre può avere uno o più strati nascosti.

- Deep Neural Network: quando un MLP possiede due o più strati nascosti si parla di rete neurale profonda (DNN) [44]. L'aumento degli strati permette alla rete di apprendere pattern più complessi. Inoltre il modello risulta capace di fare una selezione autonoma delle features più importanti. Tuttavia, l'aumento dei parametri addestrabili accresce il rischio di overfitting. Questo tipo di rete è preferibile quando le features non sono strutturate

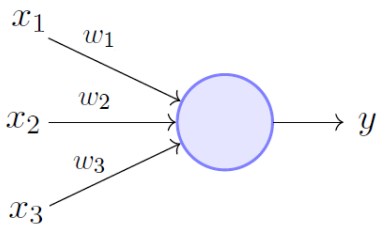

Figura 4.4: Percettrone [43]

(come nel caso di pixel di immagini o frequenze di un audio) e si dispone di un dataset molto ampio.

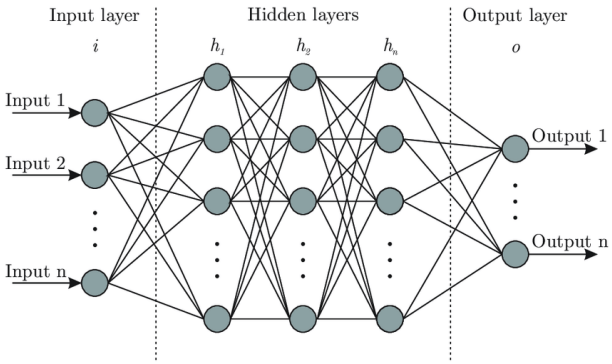

Figura 4.5: Rete neurale profonda [45]

- Convolutional Neural Network: le reti neurali convoluzionali (CNN) sono algoritmi facenti parte dell'insieme delle DNN che utilizzano strati convoluzionali per simulare il funzionamento della corteccia visiva. Le CNN applicano un filtro all'input con ogni strato, permettendogli di individuare caratteristiche particolari. Questo processo permette la semplificazione del contenuto in ingresso per poi essere passato ad un MLP classico. Le reti convoluzionali vengono spesso impiegate per la classificazione di immagini e video [46].

- Recurrent Neural Network: le reti neurali ricorrenti (RNN) sono anch'esse considerate delle reti profonde (DNN) e la loro caratteristica principale è la capacità di memorizzare eventi/stati precedenti, che impattano anche l'output di nuove iterazioni. Il segnale uscente dai neuroni viene riutilizzato come input per gli stessi neuroni al turno seguente. Questo crea un legame sequenziale tra gli input, permettendo alla rete di riconoscere pattern nel tempo. La RNN ha molteplici impieghi tra cui il natural language processing (NLP), l'analisi di video, di meteo, di trend di mercato e di anomaly detection.

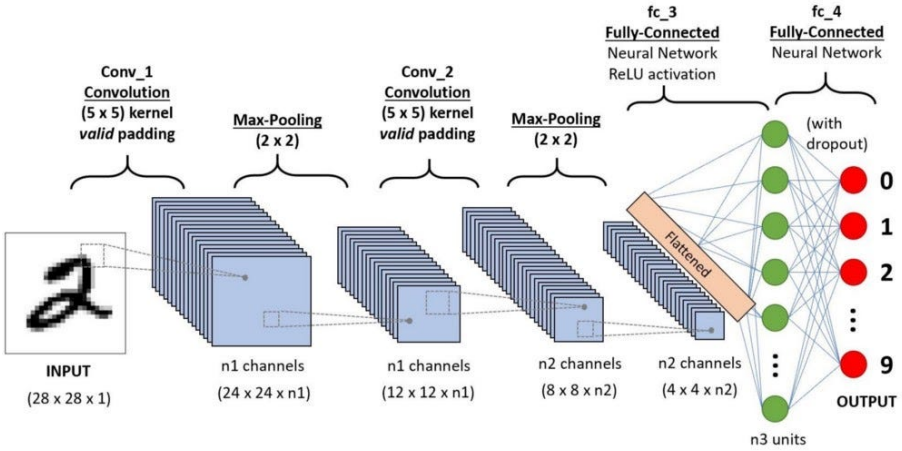

Figura 4.6: Rete convoluzionale [47]

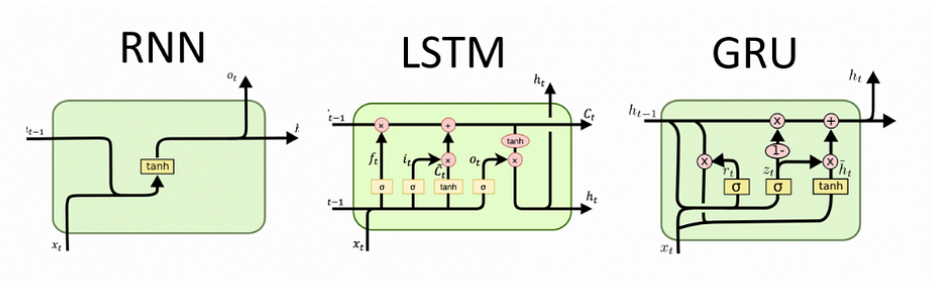

Figura 4.7: Reti neurali ricorrenti: RNN, LSTM e GRU [48]

- Long Short Term Memory: questo algoritmo (LSTM) è un esempio di rete ricorrente (RNN) che tenta di risolvere il problema della scomparsa del gradiente [49]. Attraverso l'impiego di un componente di memoria a lungo termine, la rete mantiene una copia del valore precedente e grazie a componenti chiamate gating units è capace di manipolare il flusso di memoria, aggiornando o dimenticando informazioni nel tempo. L'output della cella non è necessariamente simile al valore ricordato (passato in input alla prossima iterazione). Sebbene questa tecnica introduca un netto miglioramento rispetto a ciò che è possibile fare con una semplice RNN, non risolve il problema della scomparsa del gradiente, ma lo mitiga significativamente.

- Gated Recurrent Unit: questa tecnica (GRU) è molto simile alle reti LSTM, perchè anch'essa presenta una componente di memoria. Tuttavia risulta molto meno complicata, perchè l'output della cella e l'informazione ricordata coincidono, necessitando di meno parametri ed operazioni. Quindi, è più semplice e veloce, ma meno potente.

- Transformer: questo modello è un algoritmo di deep learning molto avanzato che permette di gestire lunghe sequenze di dati, perchè consente di ritenere informazioni su tutti gli input forniti attraverso il meccanismo dell'attenzione [50]. Il costo introdotto da questa tecnica si presenta nella forma di un aumento della complessità, che richiede molto più tempo e risorse per l'addestramento. I transformer sono stati introdotti nel 2017 e sono diventati molto popolari perchè costituiscono le fondamenta di LLM (Large Language Models) come ChatGPT e DALL-E.

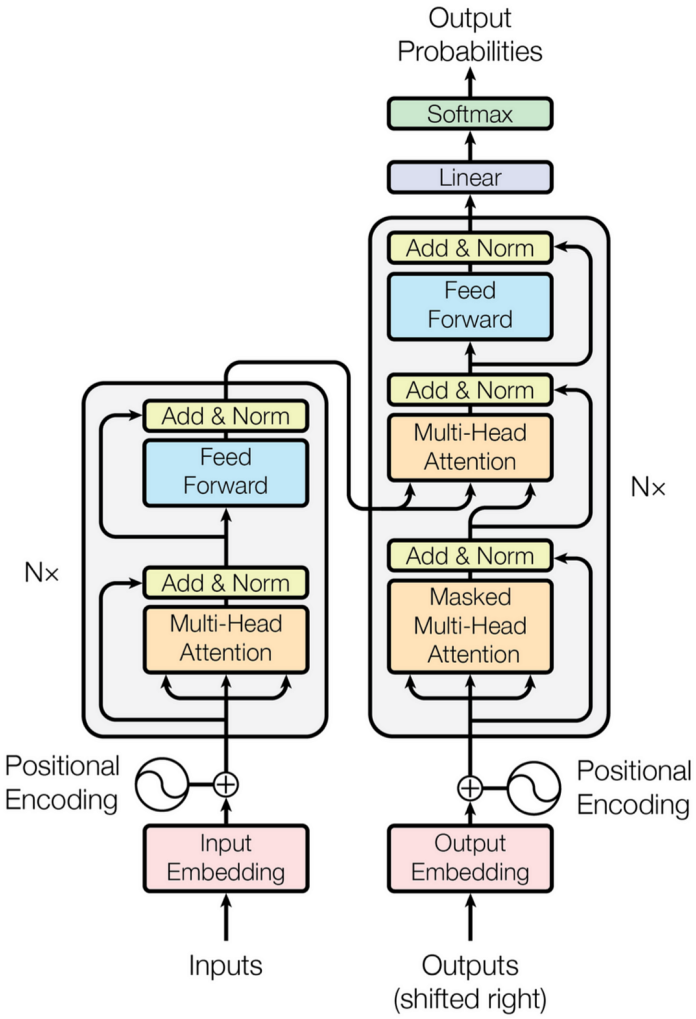

Figure 1: The Transformer - model architecture.

Figura 4.8: Modello del transformer [51]

4.4 Unsupervised learning

L'apprendimento non supervisionato è una sottocategoria dell'apprendimento automatico che comprende tutti gli algoritmi che non richiedono campioni etichettati. Questo approccio è utile a trovare pattern e relazioni nascoste tra i dati.

Le principali categorie di problemi affrontati dall'unsupervised learning sono il clustering e la riduzione della dimensionalità. Il clustering è un processo che cerca di raggruppare campioni di un insieme di dati in base alla somiglianza delle loro caratteristiche. Questo metodo di aggregazione dei dati si basa sulla similarità delle features per raggruppare i campioni in modo significativo. La riduzione della dimensionalità, d'altra parte, è una tecnica che mira a comprimere le informazioni contenute in un campione, riducendo il numero di features necessarie a rappresentarlo. Questo processo sintetizza i dati, riducendo la dimensione dei campioni, ma mantenendo comunque la maggior parte delle informazioni rilevanti.

Alcuni degli algoritmi più conosciuti sono:

- One Class Support Vector Machine: questa tecnica (OCSVM) utilizza l'algoritmo di SVM per creare un ipersfera contenente tutti i record normali. Successivamente, ogni punto che si trova al di fuori di questa ipersfera viene segnalato come anomalia.

- Isolation Forest: questo algoritmo (IF) è simile ad un RF, ma invece di seguire un criterio per dividere i dati, usa la casualità [52]. Sceglie infatti un attributo e un valore di soglia a caso per ogni nodo dell'albero di decisione. L'assunzione su cui si basa l'algoritmo è che le anomalie siano più vicine alla radice dell'albero, perchè essendo molto diverse dai dati normali vengono separate prima dal resto del gruppo. Questo approccio funziona solo se si usano molti alberi, in modo da avere una stima della normalità dei record basata sulla probabilità statistica

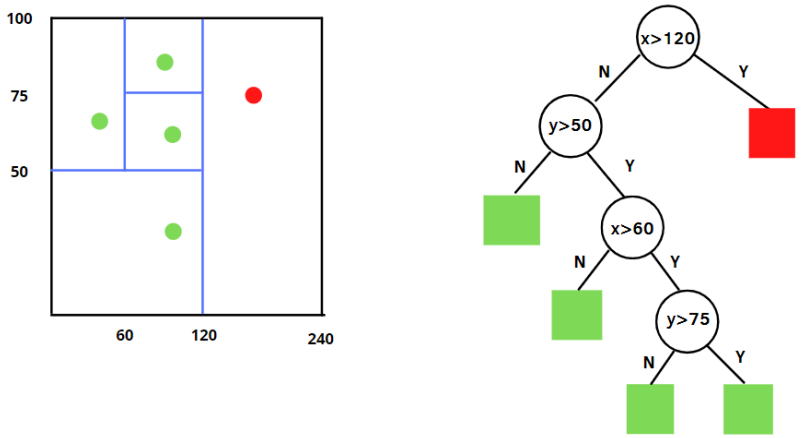

Figura 4.9: Esempio di un albero della isolation forest [53]

- AutoEncoder: questo modello (AE) è un tipo di rete neurale profonda (DNN) che non richiede supervisione [54]. Si basa su due componenti: un encoder e un decoder. L'encoder comprime il vettore di input in una dimensione inferiore, mentre il decoder espande l'output dell'encoder fino a raggiungere la dimensione originale. La rete impara a ricostruire l'input senza usare etichette, perché il suo obiettivo è minimizzare la differenza tra input e output. Durante l'addestramento, l'AE riceve solo esempi normali e apprende una rappresentazione interna di questi dati. Grazie alla compressione, l'AE elimina le informazioni irrilevanti e il rumore. Tuttavia, la ricostruzione introduce anche una distorsione, che è più evidente per i dati anomali, cioè quelli che non seguono la distribuzione dei dati normali. Misurando l'errore quadratico medio tra input e output e scegliendo una soglia appropriata, si può distinguere tra dati normali e anomalie.

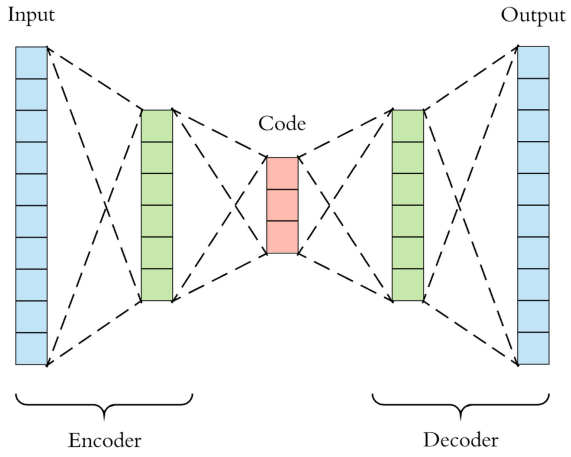

Figura 4.10: Modello dell'autoencoder [55]

- Variational AutoEncoder: si tratta di un modello (VAE) che estende l'AE introducendo un meccanismo di campionamento dello spazio latente (lo strato della rete con minor numero di dimensioni) [56]. In questo modo, la distribuzione dei dati nello spazio latente è più regolare e la ricostruzione è più accurata. Il VAE, infatti, non usa direttamente l'output dell'encoder come input del decoder, ma effettua un campionamento casuale a partire dai parametri dell'output (media e varianza). Questo vettore è poi usato come input del decoder.

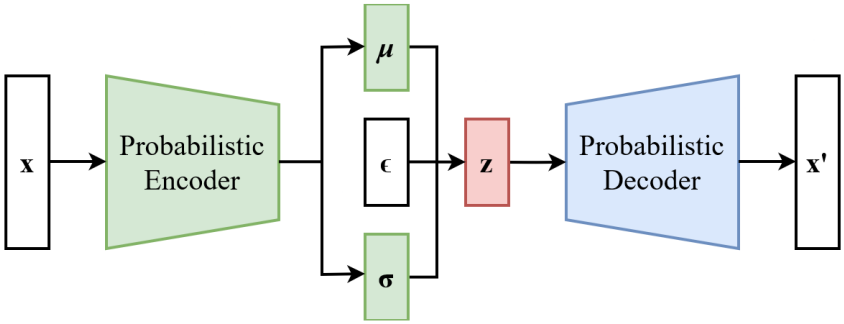

Figura 4.11: Modello del variational autoencoder [57]

4.5 Metriche di performance

Le metriche di performance sono strumenti utili per misurare l'efficacia dei modelli di machine learning. Di seguito si elencano le metriche e i parametri più comunemente utilizzati per il benchmarking di algoritmi di apprendimento automatico.

- TP: acronimo di True Positive che indica il numero di campioni anomali correttamente identificati dal modello.

- FP: acronimo di False Positive che indica il numero di campioni anomali erroneamente classificati dal modello.

- TN: acronimo di True Negative che indica il numero di campioni normali correttamente identificati dal modello.

- FN: acronimo di False Negative che indica il numero di campioni normali erroneamente classificati dal modello.

- ACC: indica l'accuratezza del modello, ovvero la percentuale di previsioni corrette sul numero totale di previsioni

$$\frac{TP + TN}{TP + FP + TN + FN}$$

- Precisione: indica la percentuale di previsioni positive corrette sul totale delle previsioni positive effettuate dal modello

$$\frac{TP}{TP + FP}$$

- Recupero: in inglese chiamato recall, indica la percentuale di previsioni positive corrette sul totale delle istanze etichettate come positive

$$\frac{TP}{TP + FN}$$

- F1-score: è la media armonica tra precisione e recupero ed è da preferire all'accuratezza nei casi di classi sbilanciate

$$\frac{2 * Precision * Recall}{Precision + Recall}$$

Capitolo 5

Stato dell'arte delle tecniche di ML in IDS

5.1 Confronto tra dataset pubblici

Il ruolo dei dataset nella creazione di un sistema di rilevamento delle intrusioni efficace è fondamentale, in quanto la capacità di individuare correttamente anomalie dipende quasi interamente dalla qualità dei dati forniti durante l'allenamento [58]. Purtroppo, nella pratica, ci sono molti ostacoli che impediscono la messa a punto di un dataset analogo. Tra queste, si possono citare la limitata disponibilità di raccolte pubbliche di dati di sicurezza informatica, l'assenza di criteri standardizzati per la selezione delle features da includere nel dataset e la complessità nell'uniformare le distribuzioni degli attacchi informatici. Detto questo, molti ricercatori hanno tentato di creare dei set di dati ampi ed esaustivi, di seguito sono riportati alcuni dei più comunemente utilizzati:

- KDD99 [59]: Pubblicato nel 1999 in occasione di "The Third International Knowledge Discovery and Data Mining Tools Competition", questo è stato uno dei primi dataset utilizzati per lo sviluppo di sistemi di rilevamento di intrusione. Questo dataset si compone di un totale di 4.898.431 di campioni, presenta 41 features e 4 categorie di attacchi. Le intrusioni presenti rientrano nelle seguenti classi: DoS, Probe (una forma di Information Gathering), U2R (User to Root, ottenere i permessi di amministratore) e R2L (Remote to Local, ottenere accesso ad una macchina da remoto). Nonostante sia stato ampiamente utilizzato per la ricerca, presenta vari problemi. Uno di questi è l'enorme numero di record ridondanti che rendeno il dataset sbilanciato. Di conseguenza, gli algoritmi di machine learning finiscono per apprendere solo gli esempi più frequenti, dimenticando quelli meno comuni. Ad oggi questo dataset è da considerarsi obsoleto perchè non riflette adeguatamente il traffico di rete e le minacce informatiche moderne. Oltre a non tenere conto le esigenze dell'IoT.

- NSL-KDD [59] [60]: Questo è un dataset pubblicato nel 2009 derivato dal KDD99, con l'obiettivo di risolvere alcuni dei suoi problemi. Questo dataset si compone di 125.973 campioni di addestramento e 22.544 campioni di test, con le stesse 41 features e 4 categorie di attacco del suo KDD99. Tuttavia, a differenza del KDD99, questo dataset elimina i record ridondanti rendendolo più bilanciato. Sebbene sia una versione migliorata presenta ancora varie limitazioni. Ad esempio, non include alcun attacco recente o sofisticato, come quelli basati su botnet o ransomware. Inoltre, non include alcun traffico di rete proveniente da dispositivi IoT.

- UNSW-NB15 [61]: Pubblicato nel 2015 dalla UNSW Canberra, questo è un dataset un po' più recente e realistico rispetto al KDD99. Questo dataset si compone di un totale di 2.540.044 campioni. Il dataset è stato generato con lo strumento IXIA PerfectStorm per creare una combinazione di attività normali e attacchi informatici simulati. Gli strumenti Argus e Bro-IDS (oggi rinominato a Zeek) sono stati utilizzati per estrarre le 49 features dai pacchetti di rete. Mentre, le intrusioni sono state suddivise nelle seguenti categorie: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode e Worms. Questo dataset introduce molti tipi di attacco rispetto al KDD99. Tuttavia, non presenta alcun esempio specifico per IoT.

- CIC-IDS2017 [62]: Questo dataset è stato pubblicato nel 2017 dal Canadian Institute for Cybersecurity (CIC) dell'Università del New Brunswick. Il dataset contiene un totale di 2'830'743 di campioni di traffico di rete, scaricabili in formato PCAP o CSV. Le intrusioni sono state suddivise in 14 categorie tra cui DDoS, PortScan, DoS Golden-eye, Heartbleed, Web Attack XSS, Web Attack SQL Injection e Botnet. Gli strumenti CICFlowMeter e Wireshark sono stati utilizzati per estrarre le 79 features dai pacchetti di rete. Il dataset è stato creato per essere il più realistico possibile e contiene attacchi comuni e recenti che riflettono i dati del mondo reale, ma rimane comunque un dataset generico non progettato per l'IoT.

- BoT-IoT [63]: Questo dataset, pubblicato nel 2018 dalla UNSW Canberra, si focalizza sul traffico generato dalle botnet, ovvero delle reti di dispositivi infetti controllati da un attaccante. L'estrazione delle 43 features è stato fatto tramite Argus, uno strumento di analisi del flusso di rete. Il dataset comprende più di 72 milioni di record, ma ne esiste una versione ridotta con 3 milioni di campioni. Gli attacchi presenti nel dataset si suddividono nelle seguenti categorie: DDoS, DoS, OS Fingerprinting, Service Scan, Keylogging, Data theft. Il dataset presenta un forte sbilanciamento tra le classi: il traffico malevolo rappresenta quasi il 99% dei record, mentre il traffico normale solo l'1%.

- ToN-IoT [64]: Il dataset ToN-IoT, pubblicato nel 2021 dalla UNSW Canberra, si focalizza sul traffico generato dalle reti di dispositivi IoT e IIoT, ovvero le reti industriali di Internet delle cose. L'estrazione delle 44 features è stata fatta tramite diversi strumenti, ma per il traffico di rete è stato impiegato principalmente Zeek. Il dataset comprende 21'978'632 di record, con attacchi suddivisi tra le categorie di Scanning, DoS, DDoS, Ransomware, Backdoor, Data Injection, XSS, Password Cracking e MITM. Il dataset presenta un forte sbilanciamento tra le classi: il traffico malevolo rappresenta circa il 97% dei record, mentre il traffico normale rappresenta solo il 3%.

- IoT-23 [65]: Questo dataset, pubblicato nel 2020 da Aposemat e finanziato da Avast, contiene diversi attacchi informatici a dispositivi della smart home, come Amazon Echo e lampadine intelligenti. Il dataset è disponibile in formato PCAP e Zeek flow, con una versione completa da 20GB o una versione ridotta da 8GB (composta dai soli flow). Si compone di oltre 300 milioni di record etichettati e 21 features.

- NF-UQ-NIDS-v2 [66]: Questo dataset è stato pubblicato dalla University of Queensland nel 2022 e comprende quattro sottodataset provenienti da diverse fonti: NF-UNSW-NB15, NF-ToN-IoT, NF-BoT-IoT e NF-CSE-CIC-IDS2018. Il dataset contiene un totale di 75.987.976 flussi di dati, di cui 25.165.295 (33,12%) sono normali e 50.822.681 (66,88%) sono attacchi. L'estrazione delle 43 features è stato fatto utilizzando CICFlowMeter, per cui il dataset è reperibile in formato CSV.

- Edge-IIoT [67]: Pubblicato nel 2022, questo dataset offre una panoramica approfondita del traffico di rete generato da dispositivi IoT industriali. Si basa su oltre 20 milioni di record, raccolti da diverse fonti e analizzati con tecniche avanzate. Il dataset presenta 61 features, selezionate da un insieme originario di 1176 features. Contiene diverse categorie di attacchi tra cui DoS, DDoS, Probing, MITM e SQL Injection. Questo lavoro è stato riconosciuto come uno dei migliori nel suo campo, entrando nella top 1% delle pubblicazioni del Web of Science (WoS).

- CICIoT2023 [68]: Questo dataset è stato presentato dal Canadian Institute for Cybersecurity (CIC) nel 2023 ed è composto da 33 attacchi eseguiti in una topologia IoT formata da 105 dispositivi reali, tra cui telecamere, smart TV, termostati, router e altri. Il dataset contiene 46 features estratte dal traffico di rete utilizzando WireShark, TCPDUMP e DPKT. Gli attacchi ricadono in sette grandi categorie: DDoS, DoS, Reconnaissance, Web-based attacks, Brute Force, Spoofing, Mirai.

Tabella 5.1: Confronto di alcuni dei dataset più utilizzati per lo sviluppo di IDS in IoT

Dataset	**Anno**	**Features**	**Attacchi**	**IoT**[1]	**Samples**	**Formato**[2]
KDD99 [59]	1999	41	4	No	4’898’431	Flow, CSV
NSL-KDD [59] [60]	2009	41	4	No	148’517	Flow, CSV
UNSW-NB15 [61]	2015	49	9	No	2’540’044	Flow, PCAP, CSV
CIC-IDS2017 [62]	2017	79	14	No	2’830’743	PCAP, CSV
BoT-IoT [63]	2018	43	10	Si	73’370’443	PCAP, CSV
ToN-IoT [64]	2019	44	9	Si	21’978’632	PCAP, CSV
IoT-23 [65]	2020	21	15	Si	325’307’990	Flow, PCAP
NF-UQ-NIDS-v2 [66]	2022	43	20	Si	75’987’976	CSV
Edge-IIoT [67]	2022	61	14	Si	20’952’648	PCAP, CSV
CICIoT2023 [68]	2023	46	33	Si	46’686’579	PCAP, CSV

[1] Indica se il dataset contiene o meno dati simulati o collezionati specificatamente per dispositivi e reti IoT

[2] Il formato con cui è possible scaricare il dataset. L’etichetta Flow comprende tutti i formati di flusso generati da programmi come Argus, Zeek/Bro, ecc.

5.2 Confronto tra le tecniche ML più utilizzate per IDS

Nello sviluppo di un sistema di rilevamento delle intrusioni basato sulle anomalie (anomaly-based IDS) è fondamentale selezionare un modello di apprendimento adeguato, che garantisca un'alta accuratezza, ma soprattutto un elevato f1-score. Quest'ultimo, infatti, è da preferire quando il dataset è composto da classi sbilanciate [69], come è il caso per molti dei dataset presentati nella sezione precedente.

Nella tabella 5.2 sono riportati alcuni studi che hanno proposto diversi algoritmi (principalmente algoritmi tradizionali del machine learning) e li hanno confrontati in base alle metriche di performance introdotte nel capitolo sul machine learning. Dalla tabella si può dedurre che, tra le tecniche supervisionate, gli alberi di decisione (DT) e le foreste casuali (RF) offrono risultati ottimi. Tra le tecniche non supervisionate, invece, si osserva che algoritmi classici come l'isolation forest (IF) o il OCSVM siano meno efficaci di tecniche avanzate come l'autoencoder (AE) [70]. Quest'ultimo non raggiunge l'accuratezza di DT e RF, ma permette di identificare gli zero days (attacchi informatici mai visti prima) con più sicurezza. Ecco perchè una soluzione ibrida tra RF e AE come quella proposta dallo studio [71] ha avuto molto successo. Infatti, l'algoritmo proposto mantiene alto il valore di F1-score e basso il numero di FP (falsi positivi). In questo studio i modelli di RF e AE vengono allenati separatamente perchè hanno due scopi differenti. La foresta casuale ha l'obiettivo di identificare gli attacchi già conosciuti (simulando l'approccio signature-based degli ids tradizionali), mentre l'autoencoder svolge il compito dell'anomaly detection individuando tutte le minacce nuove. I due modelli vengono poi posizionati in serie (prima l'RF e poi l'AE) per ottenere risultati in accuratezza e f1-score che superano il 99% per il dataset BoT-IoT e il 94% per il dataset CSE-CIC-IDS-2018.

In generale le tecniche di machine learning più classiche possono essere utilizzate per ottenere risultati molto buoni, ma mancano della complessità necessaria per comprendere pattern di più alto livello. Per cui, quando questi algoritmi vengono impiegati su nuove reti e nuovi dati, finiscono per peggiorare drasticamente nella performance generale.

Tabella 5.2: Confronto dei risultati di vari studi sulle tecniche di ML per IDS (2 pagine)

N	Ricerca	Anno	Algoritmo	Tipo classificazione[1]	Dataset	Miglior algoritmo[2]	ACC(%)	F1(%)
1	[72]	2019	LR	Binario Multi	KDD99	DT	92.90	95.40
			NB		NSL-KDD	DT	93.00	93.50
			kNN		UNSW-NB15	RF	90.30	92.40
			DT		CIC-IDS2017	RF	94.00	90.50
			RF					
			SVM					
2	[64]	2019	LR	Binario Multi	ToN-IoT	RF	85.00	86.00
			RF					
			NB					
			SVM					
			kNN					
3	[70]	2020	PCA[3] IF	Binario	CIC-IDS2017	AE	-	96.16
			OCSVM					
			AE					

[1] Classificazione binaria o multi-classe. Negli studi che presentano entrambi i tipi, i valori di ACC e F1 riportati fanno riferimento al caso binario.

[2] Riporta l'algoritmo con le migliori prestazioni (maggiore ACC e F1) in riferimento al dataset preso in esame nello studio.

[3] Principal Component Analysis: è una strategia non supervisionata per la riduzione della dimensionalità che permette di proiettare i dati su dimensioni inferiori come combinazioni lineari. I dati vengono proiettati su una retta (riduzione fino ad una dimensione) e l'algoritmo classifica i punti in base ad una soglia.

Tabella 5.2: Confronto dei risultati di vari studi sulle tecniche di ML per IDS (2 pagine)

N	Ricerca	Anno	Algoritmo	Tipo classificazione[1]	Dataset	Miglior algoritmo[2]	ACC(%)	F1(%)
4	[73]	2023	SVM	Binario Multi	KDD99	RF	99.75	99.89
			DT		UNSW-NB15	RF	99.95	99.52
			RF		CSE-CIC-IDS-2018	DT	99.99	99.99
			kNN					
			MLP					
5	[71]	2023	DT	Binario	BoT-IoT	RF+AE	99.63	99.72
			RF		CSE-CIC-IDS-2018	RF+AE	94.92	95.90
			IF					
			OCSVM					
			AE					
			RF+AE					
6	[68]	2023	LR	Binario Multi	CICIoT2023	RF	99.68	96.53
			Perceptron					
			RF					

[1] Classificazione binaria o multi-classe. Negli studi che presentano entrambi i tipi, i valori di ACC e F1 riportati fanno riferimento al caso binario.

[2] Riporta l'algoritmo con le migliori prestazioni (maggiore ACC e F1) in riferimento al dataset preso in esame nello studio.

[3] Principal Component Analysis: è una strategia non supervisionata per la riduzione della dimensionalità che permette di proiettare i dati su dimensioni inferiori come combinazioni lineari. I dati vengono proiettati su una retta (riduzione fino ad una dimensione) e l'algoritmo classifica i punti in base ad una soglia.

Dopo aver esaminato gli algoritmi più classici del machine learning, si può procedere ad analizzare le tecniche dell'ambito deep learning. Queste soluzioni sono sicuramente più complesse e richiedono maggiori risorse, in termini di memoria, parametri e tempo necessario all'addestramento, ma offrono risultati migliori e si adattano meglio a nuove situazioni.

Nella tabella 5.3 sono messi a confronto vari studi sugli algoritmi di deep learning applicati al rilevamento delle intrusioni. Dalla tabella si può dedurre che gli algoritmi basati su reti neurali ricorrenti e reti convoluzionali sono molto efficaci, oltre che essere estremamente popolari. Infatti, la maggioranza degli studi presi in considerazione (11 su 16) utilizza varianti di RNN o CNN. In particolare, molti degli studi presentati propongono soluzioni basate sull'algoritmo LSTM. Alcune di queste sono delle varianti Bi-LSTM [74] [75], oppure delle ensemble (insieme di algoritmi) [76] [77] o dei modelli ibridi con altre tecniche (AE+LSTM [78] o CNN+LSTM [79] [80] [81]). L'uso massiccio di questo modello è dovuto alla capacità dell'LSTM di gestire lunghe serie di input, rendendolo uno degli algoritmi più comuni nell'anomaly detection di dati sequenziali. D'altra parte, anche le reti convoluzionali possono essere utilizzate per gestire sequenze di input [82] [79] [83] [84] [80]. Infatti, impiegando le 1D-CNN è possibile utilizzare input e filtri monodimensionali. Questo permette ai filtri di scorrere lungo la serie di dati per effettuare la convoluzione. Anche questa architettura offre ottimi risultati che migliorano ulteriormente quando si combinano CNN e LSTM in un modello ibrido.

Tra le altre proposte degli studi troviamo modelli basati sugli ensemble [76] [77]. Entrambi utilizzano un ensemble dei più comuni algoritmi di deep learning, tra cui DNN, CNN e LSTM. Tuttavia, lo studio [76] impiega un secondo strato di ensemble in cui raggruppa LR, RF, SVM e NB. Le tecniche di ensemble risultano in una migliore accuratezza, ma non giustificano un aumento così drastico della complessità.

Altri studi hanno presentato tecniche non supervisionate a base di autoencoder [78] [85]. Le loro conclusioni riportano ottimi risultati, ma, come indicato dal paragrafo sul confronto delle tecniche di machine learning per IDS, conviene affiancare all'AE degli algoritmi supervisionati così da diminuire il numero di falsi positivi.

Infine, gli studi [86] [87] discutono dell'implementazione di un IDS basato sui transformer, che sono dei modelli molto recenti che sfruttano il meccanismo dell'attenzione per estrarre le caratteristiche globali dei record di addestramento. Questa tecnica permette di analizzare sequenze di dati estremamente lunghe (non soffre del problema della scomparsa del gradiente), a costo di una maggiore complessità computazionale. I risultati riportati nella tabella evidenziano le buone prestazioni di questa tecnica, anche quando viene impiegata per la classificazione multiclasse su dataset sbilanciati. In particolare, lo studio [87] dimostra che i transformer siano capaci di etichettare correttamente anche gli esempi di classi meno frequenti e che lo facciano con un grado di confidenza accettabile.

Tabella 5.3: Confronto dei risultati di vari studi sulle tecniche di DL per IDS (3 pagine)

N	Ricerca	Anno	Algoritmo	Tipo classificazione[1]	Dataset	Miglior Algoritmo[2]	ACC(%)	F1(%)
1	[74]	2022	LSTM Bi-LSTM GRU	Binario Multi	NSL-KDD BoT-IoT IoT-23	Bi-LSTM Bi-LSTM LSTM	99.92 99.96 99.80	99.94 99.90 99.88
2	[82]	2022	CNN	Binario	BoT-IoT	CNN	90.75	90.22
3	[78]	2022	AE+LSTM	Binario Multi	NSL-KDD	AE+LSTM	98.88	98.88
4	[88]	2023	DNN	Multi	-	DNN	93.74	93.47
5	[75]	2023	LSTM Bi-LSTM GRU	Binario Multi	ToN-IoT	Bi-LSTM	99.99	100
6	[79]	2023	DNN CNN LSTM CNN+LSTM CNN+DNN	Multi	CIC-IDS2017	CNN+DNN	95	-

[1] Classificazione binaria o multi-classe. Nelle righe che presentano entrambi i tipi, i valori di ACC e F1 riportati fanno riferimento al caso binario.

[2] Riporta l'algoritmo con le migliori prestazioni (maggiore ACC e F1) in riferimento al dataset preso in esame nello studio.

[3] Due strati di ensemble (insieme di algoritmi) in serie, il primo è (DNN, LSTM, ResNet), il secondo è (LR, RF, SVM, NB). Gli output del secondo strato vengono poi passati ad un algoritmo XGB.

[4] Uno strato di ensemble (insieme di algoritmi) composto da (MLP, DNN, CNN, LSTM). Gli output di questo strato vengono poi passati ad una DNN per ottenere il risultato.

[5] Variational Autoencoder con Gated Recurrent Unit e Auto Regression

Tabella 5.3: Confronto dei risultati di vari studi sulle tecniche di DL per IDS (3 pagine)

N	Ricerca	Anno	Algoritmo	Tipo classificazione[1]	Dataset	Miglior Algoritmo[2]	ACC(%)	F1(%)
7	[83]	2023	DNN CNN LSTM	Multi	CIC-IDS2017	CNN	98.61	-
8	[76]	2023	Ensemble Deep 1[3]	Binario	UNSW-NB15 N-BaIoT	Ensemble Deep 1[3] Ensemble Deep 1[3]	99.82 99.80	- -
9	[84]	2023	CNN	Binario Multi	CIC-DDoS2019	CNN	99.75	-
10	[77]	2023	Ensemble Deep 2[4]	Binario Multi	ToN-IoT CIC-IDS2017 SWaT	Ensemble Deep 2[4] Ensemble Deep 2[4] Ensemble Deep 2[4]	99.6 98.7 99.6	99.4 96.7 99.8
11	[80]	2023	LSTM CNN CNN+LSTM	Binario Multi	UNSW-NB15 X-IIoTID	CNN+LSTM CNN+LSTM	93.21 99.84	- -
12	[81]	2023	CNN+LSTM	Binario Multi	KDD99 NSL-KDD UNSW-NB15	CNN+LSTM CNN+LSTM CNN+LSTM	97.05 99.99 94.43	96.01 99.99 93.50

[1] Classificazione binaria o multi-classe. Nelle righe che presentano entrambi i tipi, i valori di ACC e F1 riportati fanno riferimento al caso binario.

[2] Riporta l'algoritmo con le migliori prestazioni (maggiore ACC e F1) in riferimento al dataset preso in esame nello studio.

[3] Due strati di ensemble (insieme di algoritmi) in serie, il primo è (DNN, LSTM, ResNet), il secondo è (LR, RF, SVM, NB). Gli output del secondo strato vengono poi passati ad un algoritmo XGB.

[4] Uno strato di ensemble (insieme di algoritmi) composto da (MLP, DNN, CNN, LSTM). Gli output di questo strato vengono poi passati ad una DNN per ottenere il risultato.

[5] Variational Autoencoder con Gated Recurrent Unit e Auto Regression

Tabella 5.3: Confronto dei risultati di vari studi sulle tecniche di DL per IDS (3 pagine)

N	Ricerca	Anno	Algoritmo	Tipo classificazione[1]	Dataset	Miglior Algoritmo[2]	ACC(%)	F1(%)
13	[89]	2023	DNN	Binario Multi	NF-UQ-NIDS-V2	DNN	98.23	98.22
14	[85]	2023	AE VAE VAE+GRU+AR[5]	Binario	SWaT	VAE+GRU+AR[5]	96.58	84.81
15	[86]	2023	Transformer	Multi	ToN-IoT	Transformer	99.46	99.44
16	[87]	2023	CNN+Transformer	Multi	KDD99 NSL-KDD CIC-IDS2017	CNN+Transformer CNN+Transformer CNN+Transformer	97.85 91.04 91.06	- - -

[1] Classificazione binaria o multi-classe. Nelle righe che presentano entrambi i tipi, i valori di ACC e F1 riportati fanno riferimento al caso binario.

[2] Riporta l'algoritmo con le migliori prestazioni (maggiore ACC e F1) in riferimento al dataset preso in esame nello studio.

[3] Due strati di ensemble (insieme di algoritmi) in serie, il primo è (DNN, LSTM, ResNet), il secondo è (LR, RF, SVM, NB). Gli output del secondo strato vengono poi passati ad un algoritmo XGB.

[4] Uno strato di ensemble (insieme di algoritmi) composto da (MLP, DNN, CNN, LSTM). Gli output di questo strato vengono poi passati ad una DNN per ottenere il risultato.

[5] Variational Autoencoder con Gated Recurrent Unit e Auto Regression

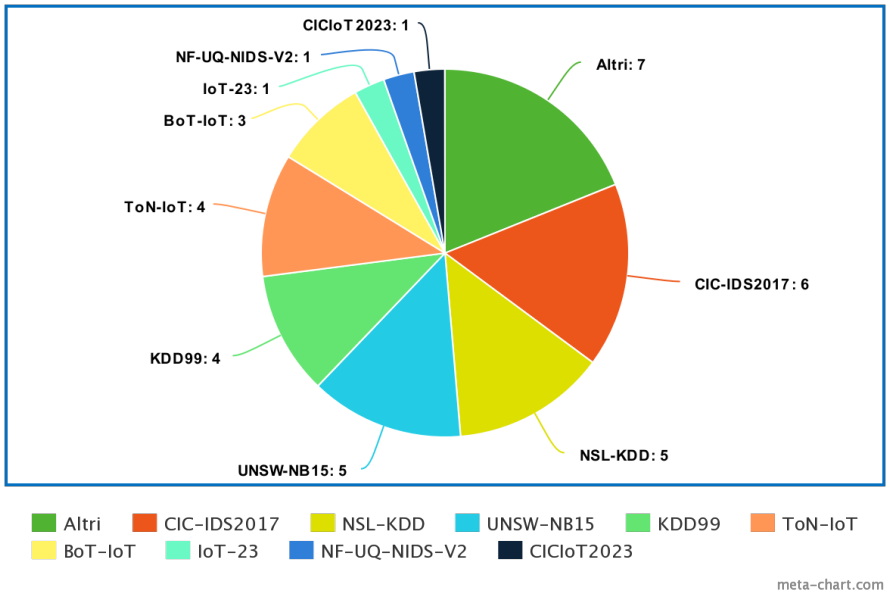

Figura 5.1: Dataset utilizzati dagli studi proposti

5.3 Implementazioni di IDS su sistemi embedded

Nelle precedenti sezioni sono stati analizzati vari dataset e modelli per lo sviluppo di sistemi di rilevamento efficaci, ma non sono state fornite implementazioni per valutare effettivamente la fattibilità di un IDS nell'IoT. Infatti, nel caso in cui si volesse utilizzare uno dei dispositivi della rete come IDS, si dovrebbero considerare le poche risorse disponibili come parte del problema. Nell'ambito tinyML (l'apprendimento automatico applicato ai sistemi embedded), è spesso necessario tenere conto dei consumi energetici, di memoria e di tempo degli algoritmi di machine learning, perchè si dispone di minori risorse rispetto a quelle dei classici computer. Alcuni studi hanno affrontato questo problema e hanno tentato di fornire risultati sperimentali per verificare l'attuabilità di un IDS nell'IoT.

Il primo di questi studi, dal nome "A Lightweight Optimized Deep Learning-based Host-Intrusion Detection System Deployed on the Edge for IoT"[90], ha creato un Host-based IDS per dispositivi IoT che permette di fare inferenza all'edge. Tuttavia, il testo riporta l'importanza di integrare un nodo al livello di fog per gestire centralmente le analisi dei singoli HIDS. Questo approccio trasforma il sistema in un Distributed IDS, permettendo una gestione più coordinata delle minacce alla sicurezza. In questa ricerca si esplora l'uso di una rete convoluzionale (CNN) che viene precedentemente allenata su un server (training offline) per poi essere testata su diverse schede di prototipazione IoT, come Raspberry 4, Arduino Portenta H7, ESP-WROOM-32, NodeMCU V3 e Arduino UNO. I risultati di questo studio concludono che sviluppare un HIDS generico multipiattaforma per dispositivi IoT non sia possibile a causa delle differenze

nell'hardware e nel software. Per cui, si suggerisce di personalizzare l'IDS in base alla scheda di produzione utilizzata. In figura 5.2 sono riportati i risultati di accuratezza e dimensione degli algoritmi proposti in questo studio. Il grafico, invece, descrive il modello di tensorflow-lite consigliato dagli autori in base al dispositivo su cui viene impiegato. Come si può vedere, con le prestazioni di un NodeMCU è preferibile utilizzare il modello ottimizzato e pre-ottimizzato (tMOP). Con un ESP32 è meglio scegliere il modello ottimizzato (tMO). Ed infine con schede più performanti conviene impiegare il modello di base di tensorflow-lite (tM).

Model	Model's depth	Accuracy	Model's size
Original Model (OM)	7 layers, 16 Features	99.74%	343 Kb
Pre-Optimized Original Model (POM)	5 layers, 16 Features	98.75%	120 Kb
"tflite" Model (tM)	7 layers, 16 Features	99.74%	106 Kb
"tflite" Model Optimized (tMO)	5 layers, 7 Features	97.21%	4237 bytes
"tflite" Model Optimized and Pre-Optimized (tMOP)	5 layers, 7 Features	96.69%	2704 bytes

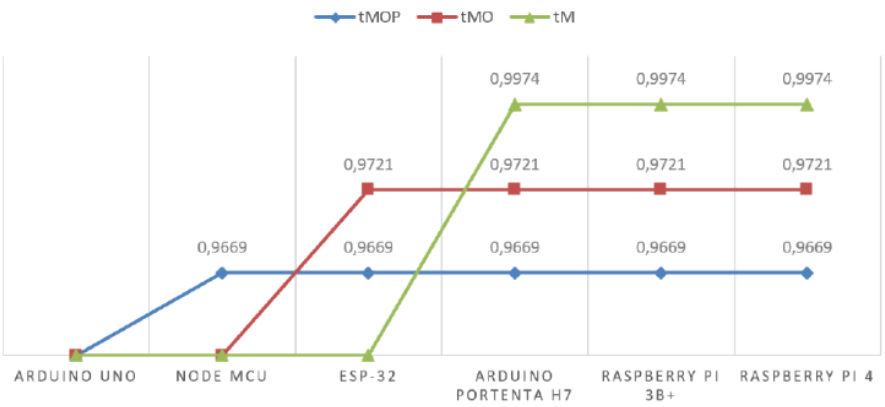

Figura 5.2: Risultati del primo studio e grafico che descrive il modello di tensorflow consigliato in base alla scheda utilizzata [90]

Il secondo studio preso in esame, dal titolo "Network Intrusion Detection System in a Light Bulb"[91], ha proposto la realizzazione di un sistema di rilevamento delle intrusioni basato sulla rete (NIDS), da implementare su una lampadina IoT. Per raggiungere questo obiettivo, gli autori hanno impiegato l'algoritmo del decision tree, mentre le schede adottate per la sperimentazione sono state l'Arduino UNO, la NodeMCU Lua Lolin V3 (che presenta un chipset ESP8266) e l'ESP32. I dataset utilizzati per l'addestramento e la valutazione delle prestazioni del sistema sono stati il BoT-IoT e il ToN-IoT. Per affrontare il problema delle classi sbilanciate, i ricercatori hanno selezionato un sottoinsieme equilibrato dei dati. Successivamente, hanno utilizzato la libreria scikit-learn per effettuare l'addestramento offline del decision tree e hanno esportato il modello da Python a C senza l'uso di librerie come Tensorflow Lite, in quanto è stato sufficiente replicare l'albero con una serie di condizioni if (figura 5.3). I risultati di questo studio evidenziano come sia possibile applicare l'albero di decisione a sistemi con risorse limitate, mantenendo alta la precisione del modello e consumando pochissima memoria. In particolare, il sistema proposto ha ottenuto una precisione media del 99% e un tempo di inferenza che varia

dagli 0.8 ai 15 microsecondi in base al dataset (figure 5.4 e 5.5). Il modello è stato in grado di funzionare su una lampadina intelligente compatibile con Tuya, che è una delle piattaforme IoT più diffuse al mondo.

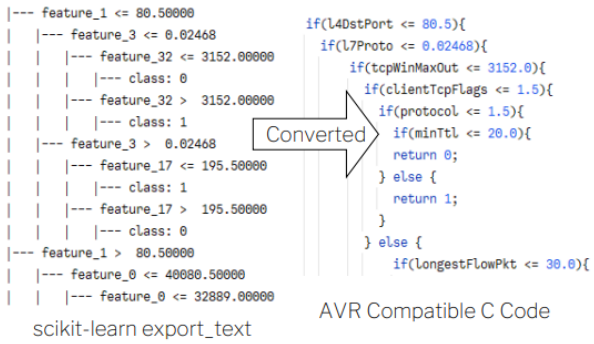

Figura 5.3: Esportazione albero di decisione [91]

Approach	Inference Time / Flow	Size -O0 (-O3)	BSS -O0 (-O3)
Our System	1.53µs	4174B (3414B)	96B (96B)
sklearn-porter	3.40µs	8902B (8670B)	8B (4B)

Size is measured with the Linux tool 'size' given after compilation with GCC at optimisation level 0 (-O0) and 3 (-O3) respectively. Here, BSS (block starting symbol) size can be considered the size of global variables.

Figura 5.4: Risultati del secondo studio in termini di tempo di inferenza e memoria consumata dall'albero di decisione confrontati con il modello di sklearn [91]

	Traditional IDS Datasets				*IoT IDS Datasets*					
Model	*CSE-CIC-IDS2018*		*UNSW-NB15*		*ToN-IoT*		*BoT-IoT*		*MQTT-IoT-IDS2020*	
Depth	**BAcc.**	**F1**	**BAcc.**	**F1**	**BAcc.**	**F1**	**BAcc.**	**F1**	**BAcc.**	**F1**
5	97.07%	96.98%	99.76%	99.76%	95.73%	95.77%	99.90%	99.90%	99.74%	99.74%
6	97.78%	97.73%	99.76%	99.76%	96.30%	96.40%	99.94%	99.94%	99.75%	9.75%
10	98.09%	98.05%	99.76%	99.76%	98.79%	98.80%	**99.96%**	99.96%	**99.93%**	99.93%
12	**98.13%**	98.09%	**99.76%**	99.76%	**99.03%**	99.04%	99.36%	99.36%	99.92%	99.92%
Best Acc.	98.13%		99.76%		99.03%		99.96%		99.92%	

Figura 5.5: Risultati del secondo studio in accuratezza e f1-score [91]

Il terzo studio preso in considerazione, intitolato "Energy consumption of on-device machine learning models for IoT intrusion detection"[92], ha implementato un sistema di rilevamento delle intrusioni IoT utilizzando un dataset specializzato in traffico anomalo per la smart home. Il dataset contiene traffico di smartphone, termostati, controlli della luce, sensori di movimento e altri ancora. Sono stati testati e comparati diversi algoritmi di machine learning tra cui logistic regression (LR), decision tree (DT), random forest (RF), naive bayes (NB), kNN e multilayer

perceptron (MLP o ANN). I tempi di training e di inferenza dei vari modelli sono stati comparati su tre diversi dispositivi: un server, un laptop ed un raspberry pi 4. Infine, il modello è stato esportato usando la libreria MicroMLGen per essere compatibile con la scheda ESP32 Azure IoT Kit, con cui sono stati misurati i tempi di inferenza per gli algoritmi sopracitati salvo per kNN e ANN, perchè troppo costosi. I risultati dimostrano che fare inferenza per IDS su dispositivi IoT è possibile, ma bisogna valutare bene quale modello impiegare. Gli algoritmi di logistic regression e naive bayes sono molto rapidi in fase di training, ma non soddisfano in fase di testing con la loro bassa precisione e i tempi di inferenza leggermente più alti. Gli algoritmi di decision tree e random forest performano molto meglio, ma il secondo ha tempi di inferenza e consumi maggiori rispetto al primo seppur provvedendo ad un leggero aumento della precisione. Questo miglioramento è spesso piccolo, per cui è possibile utilizzare il decision tree nel caso si preferisca risparmiare risorse. I risultati sono consultabili in figura 5.6.

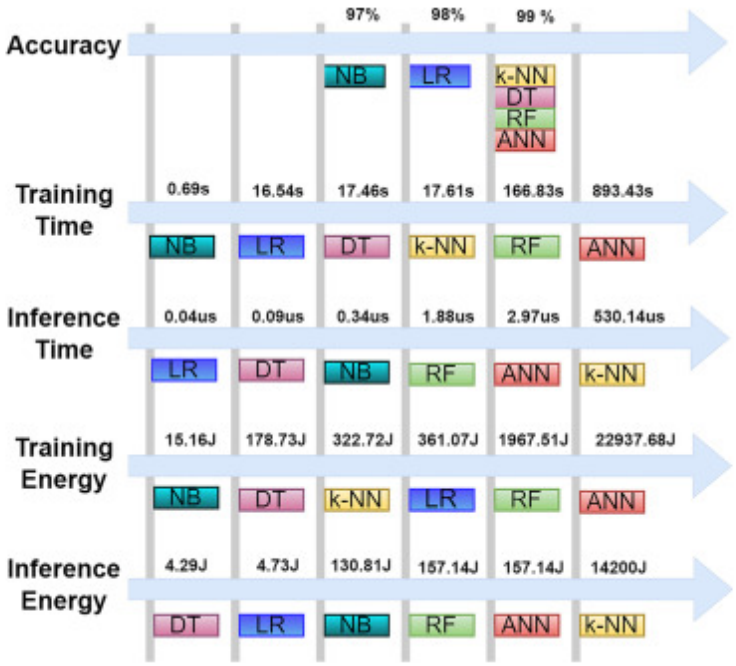

Figura 5.6: Risultati del terzo studio [92]

5.4 Maggiori sfide da affrontare degli IDS in ambito IoT

Lo sviluppo di sistemi di rilevamento delle intrusioni per l'Internet of Things è un'area della ricerca ancora molto immatura, che si trova ad affrontare parecchie sfide [93]. Sebbene negli ultimi anni si siano fatti molti passi avanti con l'avanzamento del deep learning e con l'aumento delle risorse a disposizione dei microcontrollori, risulta evidente che ci sia ancora molto lavoro da fare.

5.4.1 Problemi con i dataset

Tra i principali problemi con i dataset vi è la mancanza di uno standard nella scelta delle feature usate per fare classificazione. Questo significa che esportare un modello di machine learning e testarlo su un altro set di dati non è possibile. Ogni dataset propone un suo insieme di caratteristiche, spesso provenienti da informazioni sui pacchetti IP e TCP/UDP o da statistiche sul traffico (come la grandezza media in byte dei pacchetti) [59] [61], ma alcune volte estratte da protocolli diversi (come MQTT [67]), per cui risulta molto difficile fare confronti tra diversi insiemi di dati. Inoltre, in alcuni casi, un determinato set di feature può migliorare la performance del modello, come può peggiorarla notevolmente [94] [95].

Un altro problema da considerare è dovuto alla diversa distribuzione dei dati tra un dataset e l'altro. Ognuno dei dataset presentati in questo elaborato sono stati raccolti da reti, dispositivi e strumenti diversi. Di conseguenza, gli indirizzi IP, le porte, i protocolli e la durata delle connessioni risultano differenti. Un Anomaly-based IDS non può essere impiegato su una rete per cui non è stato addestrato, perchè noterebbe le differenze e le etichetterebbe come anomalie, generando continuamente errori di classificazione. Al fine di ottenere un modello capace di generalizzare correttamente il problema, è necessario allenarlo e valutare i risultati di test su più dataset, effettuando quella che si chiama valutazione (o convalida) incrociata [96]. L'impiego di multipli set di dati durante l'addestramento è il primo passo verso un IDS non più specifico, ma generale [66]. Tuttavia, per raggiungere tale obiettivo rimane fondamentale determinare uno standard per le feature estratte dai formati Flow [93].

Un'altra importante considerazione riguarda la classificazione multiclasse del tipo di attacco rilevato, perchè, avendo dataset molto sbilanciati nel quantitativo di sample per ogni classe, l'addestramento risulta meno efficace e porta ad una performance inferiore rispetto alla classificazione binaria [97]. Questa situazione si verifica nella maggior parte dei dataset descritti nelle sezioni precedenti. Per aggirare parzialmente questo problema, si possono usare tecniche come il sottocampionamento, che consiste nel ridurre il numero di sample delle classi più numerose [98], o il sovracampionamento, che consiste nel generare sample sintetici per le classi meno frequenti [99].

5.4.2 Problemi con il machine learning per IDS

Per allenare gli algoritmi di machine learning per il rilevamento delle intrusioni è necessario adottare alcuni accorgimenti. Innanzitutto, nella maggior parte dei casi si devono raccogliere esempi di traffico della rete su cui andrà installato l'IDS [100] e nel caso si impieghi una tecnica di apprendimento supervisionato, sarà necessario eseguire delle simulazioni degli attacchi informatici ed etichettare i record manualmente. Questo processo richiede tempo ed è poco pratico. Alcuni approcci non supervisionati richiedono solo la raccolta del traffico considerato normale (come ad esempio gli autoencoder). Tuttavia, utilizzare tecniche non supervisionate spesso comporta un aumento dei falsi positivi.

Un'altra considerazione da fare riguarda la modalità di addestramento e rilascio del modello. Esistono due metodi per addestrare un modello di rilevamento delle intrusioni: il primo è un approccio di pre-addestramento, in cui l'algoritmo viene allenato prima del suo rilascio nella rete, mentre il secondo è un approccio di addestramento online, in cui l'algoritmo impara costantemente anche dopo il suo rilascio nella rete. Il primo metodo è una soluzione statica che non si aggiorna, mentre il secondo è in grado di adattarsi alle variazioni della rete con maggiore facilità. Tuttavia introduce rischi di attacchi avversariali, ovvero tecniche con cui un attaccante riesce a manipolare o confondere il sistema di rilevamento per fingersi legittimo. Esempi possono essere la modifica dei pacchetti malevoli in modo che sembrino parte del traffico normale [101].

5.4.3 Problemi nell'implementare IDS su dispositivi IoT

Una prima considerazione riguarda le limitate risorse disponibili ai dispositivi IoT. Spesso, i sistemi embedded soffrono di memoria limitata, il che rende difficile l'uso di modelli di apprendimento automatico complessi. Inoltre, anche se si riuscisse a implementare un algoritmo di deep learning con la scarsa quantità di memoria a disposizione, sarebbe necessario far fronte al grande consumo di energia che questo richiederebbe [102]. Negli ultimi anni, l'ambito TinyML (machine learning su microcontrollori) ha fatto molti passi avanti, ma è ancora un settore molto giovane che presenta varie limitazioni.

Un altro dei problemi principali che rallentano lo sviluppo di sistemi di rilevamento di intrusioni per l'Internet of Things è dovuto alla forte eterogeneità della sua struttura, sia a livello di dispositivi che a livello di protocolli [103]. Ogni dispositivo IoT è dotato di diverse risorse a livello hardware rispetto agli altri dispositivi IoT. Può supportare diversi protocolli di comunicazione, di rete, di trasporto e di applicazione (come riportato in figura 5.7), per cui risulta complesso determinare delle features standard per i Network-based IDS. Inoltre, può utilizzare sistemi operativi diversi, il che rende difficile creare un Host-based IDS generico perchè do-

vrebbe essere compatibile con più sistemi. Ognuno dei dispositivi IoT necessiterebbe di un IDS sviluppato su misura [90].

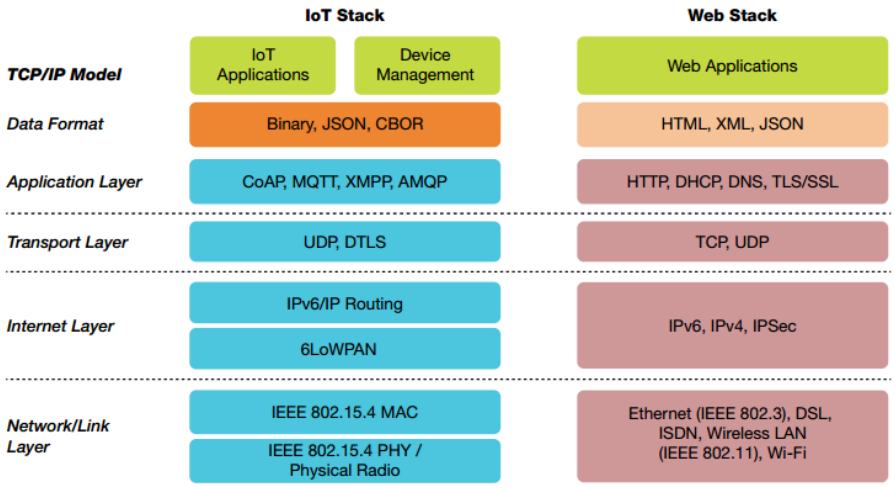

Figura 5.7: Confronto tra la pila TCP/IP per l'IoT e per il Web [104]

Oltre alle problematiche introdotte dalle caratteristiche dell'IoT sono da considerare quelle dovute al funzionamento delle reti. Infatti, se l'obiettivo degli IDS è quello di individuare anomalie nel traffico di rete per identificare attacchi informatici, potrebbe essere facilmente ingannato da altre tipologie di anomalie che accadono normalmente nelle reti, il cui risultato sarebbe un aumento dei falsi positivi rilevati. Esempi di questi eventi sono guasti, imprevisti, rallentamenti del traffico ed errori di trasmissione. Inoltre, nel caso in cui l'IDS dovesse analizzare un volume di traffico di rete che non riesce a gestire, finirebbe per rifiutare i messaggi che arrivano dopo aver riempito il buffer, per cui c'è il rischio che un attacco non venga rilevato [102].

5.5 Direzioni future

Dopo aver discusso i principali problemi che rendono difficile implementare i sistemi di rilevamento delle intrusioni basati su machine learning nell'ambito IoT si può passare a discutere delle direzioni future che la ricerca dovrebbe considerare.

In primo luogo c'è bisogno di sviluppare degli standard per le features estratte dalle reti e dai log di sistema [105]. L'impiego di una soluzione comune in questo contesto permetterebbe ai modelli di machine learning di essere comparati in maniera più esaustiva. Per esempio, si riuscirebbe a confrontare i risultati degli algoritmi se utilizzati su dataset nuovi. Questo aiuterebbe a dimostrare l'efficacia dei modelli nel generalizzare.

Un'altra possibilità da considerare è l'impiego di tecniche di feature selection per ridurre la dimensionalità degli input. Non solo aiuterebbe a prevenire l'overfitting degli algoritmi, ma permetterrebbe di diminuire il costo computazionale dell'inferenza, aumentando la velocità di risposta dell'IDS e diminuendo il consumo di energia. Inoltre, risulterebbe essere un'ottima soluzione alla gestione del big data dell'IoT.

Dei dataset proposti in questo elaborato, la maggior parte presenta distribuzioni sbilanciate di campioni nelle diverse classi, con un numero di record molto differente per ogni categoria. Come, per esempio, il BoT-IoT che si compone di 99% di esempi malevoli contro il misero 1% di traffico normale. Inoltre, tutti i dataset sono generati sinteticamente in laboratorio. Infatti, nessuno di essi contiene dati di traffico reale, sia per motivi di privacy (poche imprese sono disposte a rilasciare informazioni sulla loro rete aziendale) che per motivi pratici. Il processo di etichettatura dei campioni è altamente dispendioso per le reti reali, perchè richiede un esperto che analizzi molto attentamente il traffico e i log di sistema. Non conoscendo i metodi impiegati dall'attaccante si perde molto più tempo nell'identificare i campioni malevoli rispetto al simulare le intrusioni. Di conseguenza, una possibile direzione futura potrebbe essere quella dell'impiego delle Honeynet (reti di dispositivi fittizie che esistono solo per attirare malintenzionati e studiarne il comportamento) per la generazione di nuovi dataset [106]. Questa tecnica permette di scoprire i metodi realmente utilizzati dagli aggressori, perchè tutto il traffico generato all'interno di queste reti, che si discosta dal comportamento prefissato per i singoli host, è considerato un'intrusione al 100%. Un'altra possibilità può essere l'impiego di modelli generativi come le GAN (Generative Adversarial Network) per la sintesi di record nuovi [107], che aiuterebbe a risolvere il problema dello sbilanciamento delle classi e aumenterebbe la protezione contro attacchi avversariali.

Infine, rimane da considerare l'impiego di una architettura SDN (Software Defined Network [108]) per la gestione e il monitoraggio su più livelli del traffico di rete. Le SDN sono applicazioni software con l'obiettivo di creare una rete di dispositivi monitorabile e programmabile centralmente. Facendo uso del protocollo OpenFlow possono accedere ai nodi (gli switch, i router o i firewall) e gestire la rete. Con l'impiego di una SDN è possibile implementare un IDS possa rilevare attacchi informatici in modo più efficiente, perchè fornisce una aumento della visibilità e del controllo sulla rete [109].

Capitolo 6

Conclusioni

In questo elaborato sono stati recensiti alcuni dei più recenti studi sugli IDS basati su tecniche di machine learning e deep learning nell'ambito IoT. I risultati sono stati comparati per determinare i modelli più efficaci e promettenti. Sono state considerate modalità di addestramento sia supervisionate che non supervisionate, confrontando vantaggi e svantaggi di entrambe. Tra le tecniche tradizionali, gli alberi di decisione e le foreste casuali prevalgono per l'accuratezza e f1-score. Tuttavia, non hanno dimostrato grandi capacità nell'individuare zero days, dove invece l'autoencoder si è rilevato molto più efficace, a scapito di un più alto rateo di falsi positivi. Tra le tecniche di deep learning sono state comparate CNN, LSTM, VAE, transformers e altre ancora. Non esiste un modello chiaramente superiore, perchè tutte le proposte hanno raggiunto risultati simili. Tuttavia, gli algoritmi più promettenti sembrano essere i transformers e i metodi a ensemble.

In questa tesi sono stati presentati vari dataset, alcuni selezionati per popolarità e altri per recentezza, e si sono discusse le problematiche che li affliggono, tra cui la mancanza di un insieme standard di caratteristiche per la classificazione e la distribuzione sbilanciata di campioni nelle classi.

Sono stati presentati tre studi sull'implementazione di anomaly-based IDS su sistemi embedded. Essi dimostrano che è possibile applicare tecniche di machine learning in condizioni di risorse limitate e che l'inferenza richiede solo delle frazioni di millisecondo.

Nell'ultima parte dell'elaborato vengono presentate alcune direzioni future che la ricerca potrebbe considerare per migliorare i sistemi attuali. Per affrontare alcuni dei problemi con i dataset è possibile ricorrere all'uso di GAN, per sintetizzare nuovi campioni e arricchire il dataset, oppure si possono raccogliere record reali con l'impiego delle Honeynet. Per quanto riguarda le sfide poste dall'eterogeneità dell'IoT, un'idea interessante è quella di combinare l'IDS con l'architettura SDN, che permette di centralizzare il controllo della rete e di ottenere maggiore visibilità.

Questo studio ha incontrato alcune limitazioni nella revisione della letteratura, dovute alla scarsità di ricerche che combinano machine learning, intrusion detection e iot, che sono temi molto specifici e distinti. Inoltre, è stato difficile confrontare i diversi approcci e risultati, a causa della diversità di tecniche e dataset impiegati nei vari studi.

In conclusione, questa tesi offre una panoramica degli algoritmi di machine learning applicati al problema del rilevamento di intrusioni nelle reti IoT. Dopo aver fornito una presentazione generale del contesto, analizza vari studi che mostrano lo stato attuale della ricerca. Inoltre, evidenzia le principali sfide da superare e presenta alcune possibili direzioni di sviluppo per la ricerca futura in questo ambito.

Bibliografia

[1] Philipp Wegner. *IoT Analytics Global Market Report 2023*. https://iot-analytics.com/product/state-of-iot-spring-2023/. 2023.

[2] SonicWall. *2023 SonicWall Cyber Threat Report*. https://www.sonicwall.com/medialibrary/en/white-paper/2023-cyber-threat-report.pdf. 2023.

[3] Treccani. *Internet of things*. 2023. url: https://www.treccani.it/enciclopedia/internet-of-things_%28Lessico-del-XXI-Secolo%29/.

[4] Rosilah Hassan, Faizan Qamar, Mohammad Kamrul Hasan, Azana Hafizah Mohd Aman e Amjed Sid Ahmed. «Internet of Things and its applications: A comprehensive survey». In: *Symmetry* 12.10 (2020), p. 1674.

[5] Alyazia Aldhaheri, Fatima Alwahedi, Mohamed Amine Ferrag e Ammar Battah. «Deep learning for cyber threat detection in IoT networks: A review». In: *Internet of Things and Cyber-Physical Systems* (2023).

[6] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis et al. «Understanding the mirai botnet». In: *26th USENIX security symposium (USENIX Security 17)*. 2017, pp. 1093–1110.

[7] Aohui Wang, Ruigang Liang, Xiaokang Liu, Yingjun Zhang, Kai Chen e Jin Li. «An inside look at IoT malware». In: *Industrial IoT Technologies and Applications: Second EAI International Conference, Industrial IoT 2017, Wuhu, China, March 25–26, 2017, Proceedings 2*. Springer. 2017, pp. 176–186.

[8] Kaspersky. *IoT Threat Report 2023*. https://securelist.com/iot-threat-report-2023/110644/. 2023.

[9] Kaspersky. *What is Spoofing & How to Prevent it - Kaspersky*. 2023. url: https://www.kaspersky.com/resource-center/definitions/spoofing.

[10] Okta. *IP Spoofing Unraveled: What It Is & How to Prevent It*. https://www.okta.com/identity-101/ip-spoofing/. 2023.

[11] OneLogin. *DDoS Attack: What Is It and How to Prevent It*. 2023. url: https://www.onelogin.com/learn/ddos-attack.

[12] Jean-Paul A Yaacoub, Hassan N Noura, Ola Salman e Ali Chehab. «Ethical hacking for IoT: Security issues, challenges, solutions and recommendations». In: *Internet of Things and Cyber-Physical Systems* 3 (2023), pp. 280–308.

[13] Mohammed Husamuddin e Mohammed Qayyum. «Internet of Things: A study on security and privacy threats». In: *2017 2nd International Conference on Anti-Cyber Crimes (ICACC)*. 2017, pp. 93–97. doi: 10.1109/Anti-Cybercrime.2017.7905270.

[14] Hamidreza Fereidouni, Olga Fadeitcheva e Mehdi Zalai. «IoT and Man-in-the-Middle Attacks». In: *arXiv preprint arXiv:2308.02479* (2023).

[15] The SSL Store. *Executing a Man-in-the-Middle Attack in just 15 Minutes*. 2021. url: https://www.thesslstore.com/blog/man-in-the-middle-attack-2/.

[16] Malwarebytes. *Sicurezza informatica Malwarebytes per utenti privati e aziendali*. url: https://it.malwarebytes.com/malware/.

[17] Luca Zorloni. *Regione Lazio, quanto è costato l'attacco ransomware*. https://www.wired.it/article/regione-lazio-attacco-ransomware-costo-microsoft/. 2021.

[18] Kelli Young. *Cyber Case Study: The Mirai DDoS Attack on Dyn*. CoverLink Insurance - Ohio Insurance Agency Home. 2022. url: https://coverlink.com/case-study/mirai-ddos-attack-on-dyn/.

[19] Clare Stouffer. *Ten types of malware*. 2021. url: https://us.norton.com/blog/malware/types-of-malware.

[20] Kaspersky. *Attacchi SQL injection e come prevenirli*. url: https://www.kaspersky.it/resource-center/definitions/sql-injection.

[21] Kaspersky. *What is a Cross-Site Scripting attack? Definition & Examples*. https://www.kaspersky.it/resource-center/definitions/what-is-a-cross-site-scripting-attack.

[22] Aryan Garg. *SQL Injection: The Cyber Attack Hiding in Your Database*. https://www.analyticsvidhya.com/blog/2023/02/sql-injection-the-cyber-attack-hiding-in-your-database/. 2023.

[23] IBM. *What is an intrusion detection system (IDS)?* https://www.ibm.com/topics/intrusion-detection-system. 2023.

[24] GeeksforGeeks. *Difference between HIDs and NIDs*. https://www.geeksforgeeks.org/difference-between-hids-and-nids/. Accessed on November 5, 2023. 2023.

[25] Murat Özalp, Cihan Karakuzu e Ahmet Zengin. «Distributed intrusion detection systems: A survey». In: *Academic Perspective Procedia* 2.3 (2019), pp. 400–407.

[26] Elena Fedorchenko, Evgenia Novikova e Anton Shulepov. «Comparative review of the intrusion detection systems based on federated learning: Advantages and open challenges». In: *Algorithms* 15.7 (2022), p. 247.

[27] Enrique Mármol Campos, Pablo Fernández Saura, Aurora González-Vidal, José L Hernández-Ramos, Jorge Bernal Bernabé, Gianmarco Baldini e Antonio Skarmeta. «Evaluating Federated Learning for intrusion detection in Internet of Things: Review and challenges». In: *Computer Networks* 203 (2022), p. 108661.

[28] Vishwa Teja Alaparthy e Salvatore Domenic Morgera. «A multi-level intrusion detection system for wireless sensor networks based on immune theory». In: *IEEE Access* 6 (2018), pp. 47364–47373.

[29] Elijah M Maseno, Zenghui Wang, Hongyan Xing et al. «A systematic review on hybrid intrusion detection system». In: *Security and Communication Networks* 2022 (2022).

[30] IBM. *Log Collection Tool*. https://www.ibm.com/docs/en/datacap/9.1.9?topic=tools-log-collection-tool. 2023.

[31] Angela Horneman e Nathan Dell. «Smart collection and storage method for network traffic data». In: *Book Smart Collection and Storage Method for Network Traffic Data* (2014).

[32] SolarWinds. *What Is Packet Capture (PCAP)?* https://www.solarwinds.com/resources/it-glossary/pcap.

[33] Rick Hofstede, Pavel Čeleda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto e Aiko Pras. «Flow monitoring explained: From packet capture to data analysis with netflow and ipfix». In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 2037–2064.

[34] Shai Shalev-Shwartz e Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[35] Jaime Carbonell, Ryszard Michalski e Tom Mitchell. «An overview of machine learning». In: *Machine learning* (1983), pp. 3–23.

[36] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[37] Corinna Cortes e Vladimir Vapnik. «Support-vector networks». In: *Machine learning* 20 (1995), pp. 273–297.

[38] Anshul Sain. *Support Vector Machines(SVM) - A Complete Guide for Beginners*. https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/. 2021.

[39] Ross Quinlan. «Induction of decision trees». In: *Machine learning* 1 (1986), pp. 81–106.

[40] Siddharth Yadav. *Machine Learning Basics: Decision Tree Regression*. https://towardsdatascience.com/machine-learning-basics-decision-tree-regression-1d73ea003fda. 2019.

[41] Leo Breiman. «Random forests». In: *Machine learning* 45 (2001), pp. 5–32.

[42] Roi Yehoshua. *Random Forests*. https://medium.com/@roiyeho/random-forests-98892261dc49. 2018.

[43] Gerry Saporito. *What is a Perceptron?* https://towardsdatascience.com/what-is-a-perceptron-210a50190c3b. 2021.

[44] Yann LeCun, Yoshua Bengio e Geoffrey Hinton. «Deep learning». In: *nature* 521.7553 (2015), pp. 436–444.

[45] Lavanya Shukla. *Designing Your Neural Networks*. https://towardsdatascience.com/designing-your-neural-networks-a5e4617027ed. 2021.

[46] Alex Krizhevsky, Ilya Sutskever e Geoffrey E Hinton. «Imagenet classification with deep convolutional neural networks». In: *Advances in neural information processing systems* 25 (2012).

[47] Nushaine Ferdinand. *A Simple Guide to Convolutional Neural Networks*. https://towardsdatascience.com/a-simple-guide-to-convolutional-neural-networks-751789e7bd88. 2019.

[48] Mahmud Oyinloye. *Recurrent Neural Networks Report*. https://medium.com/@le.oasis/recurrent-neural-networks-report-c70e6f05cc9e. 2022.

[49] Sepp Hochreiter e Jürgen Schmidhuber. «Long short-term memory». In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser e Illia Polosukhin. «Attention is all you need». In: *Advances in neural information processing systems* 30 (2017).

[51] Maxime. *What is a Transformer?* https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04. 2019.

[52] Fei Tony Liu, Kai Ming Ting e Zhi-Hua Zhou. «Isolation forest». In: *2008 eighth ieee international conference on data mining*. IEEE. 2008, pp. 413–422.

[53] Siddharth Sharma. *Anomaly Detection With Isolation Forest*. https://betterprogramming.pub/anomaly-detection-with-isolation-forest-e41f1f55cc6. 2021.

[54] Dana H Ballard. «Modular learning in neural networks». In: *Proceedings of the sixth National Conference on artificial intelligence-volume 1*. 1987, pp. 279–284.

[55] Siddharth Yadav. *Generating Images with Autoencoders*. https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368. 2019.

[56] Diederik Kingma e Max Welling. «Auto-encoding variational bayes». In: *arXiv preprint arXiv:1312.6114* (2013).

[57] Erfan Eshratifar. *Variational Auto Encoder (VAE) for the Numerai Dataset*. https://medium.com/mlearning-ai/variational-auto-encoders-vae-for-the-numerai-dataset-2709dcc7e449. 2022.

[58] Laura Morán-Fernández, Verónica Bólon-Canedo e Amparo Alonso-Betanzos. «How important is data quality? Best classifiers vs best features». In: *Neurocomputing* 470 (2022), pp. 365–375.

[59] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu e Ali A Ghorbani. «A detailed analysis of the KDD CUP 99 data set». In: *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee. 2009, pp. 1–6.

[60] Canadian Institute for Cybersecurity. *NSL-KDD*. https://www.unb.ca/cic/datasets/nsl.html. 2018.

[61] Nour Moustafa e Jill Slay. «UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)». In: *2015 military communications and information systems conference (MilCIS)*. IEEE. 2015, pp. 1–6.

[62] Iman Sharafaldin, Arash Habibi Lashkari e Ali A Ghorbani. «Toward generating a new intrusion detection dataset and intrusion traffic characterization.» In: *ICISSp* 1 (2018), pp. 108–116.

[63] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova e Benjamin Turnbull. «Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset». In: *Future Generation Computer Systems* 100 (2019), pp. 779–796.

[64] Abdullah Alsaedi, Nour Moustafa, Zahir Tari, Abdun Mahmood e Adnan Anwar. «TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems». In: *Ieee Access* 8 (2020), pp. 165130–165150.

[65] Sebastian Garcia, Agustin Parmisano e Maria J Erquiaga. «IoT-23: A Labeled Dataset with Malicious and Benign IoT Network Traffic. 2020». In: *URL: https://www. stratosphereips. org/datasets-iot23. doi: http://doi. org/10.5281/zenodo* 4743746 (2021).

[66] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa e Marius Portmann. «NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems». In: *Big Data Technologies and Applications*. A cura di Zeng Deze, Huan Huang, Rui Hou, Seungmin Rho e Naveen Chilamkurti. Cham: Springer International Publishing, 2021, pp. 117–135. isbn: 978-3-030-72802-1.

[67] Mohamed Amine Ferrag, Othmane Friha, Djallel Hamouda, Leandros Maglaras e Helge Janicke. «Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning». In: *IEEE Access* 10 (2022), pp. 40281–40306. doi: 10.1109/ACCESS.2022.3165809.

[68] Euclides Carlos Pinto Neto, Sajjad Dadkhah, Raphael Ferreira, Alireza Zohourian, Rongxing Lu e Ali A. Ghorbani. «CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment». In: *Sensors* 23.13 (2023). issn: 1424-8220. doi: 10.3390/s23135941. url: https://www.mdpi.com/1424-8220/23/13/5941.

[69] Purva Huilgol. *Accuracy vs. F1-Score*. https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2. 2019.

[70] Miel Verkerken, Laurens D'hooge, Tim Wauters, Bruno Volckaert e Filip De Turck. «Unsupervised machine learning techniques for network intrusion detection on modern data». In: *2020 4th Cyber Security in Networking Conference (CSNet)*. IEEE. 2020, pp. 1–8.

[71] Chao Wang, Yunxiao Sun, Wenting Wang, Hongri Liu e Bailing Wang. «Hybrid Intrusion Detection System Based on Combination of Random Forest and Autoencoder». In: *Symmetry* 15.3 (2023), p. 568.

[72] Ravi Vinayakumar, Mamoun Alazab, KP Soman, Prabaharan Poornachandran, Ameer Al-Nemrat e Sitalakshmi Venkatraman. «Deep learning approach for intelligent intrusion detection system». In: *Ieee Access* 7 (2019), pp. 41525–41550.

[73] Pierpaolo Dini, Abdussalam Elhanashi, Andrea Begni, Sergio Saponara, Qinghe Zheng e Kaouther Gasmi. «Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity». In: *Applied Sciences* 13.13 (2023), p. 7507. url: https://doi.org/10.3390/app13137507.

[74] Imtiaz Ullah e Qusay H. Mahmoud. «Design and Development of RNN Anomaly Detection Model for IoT Networks». In: *IEEE Access* 10 (2022), pp. 62722–62750. doi: 10.1109/ACCESS.2022.3176317.

[75] Sahar Soliman, Wed Oudah e Ahamed Aljuhani. «Deep learning-based intrusion detection approach for securing industrial Internet of Things». In: *Alexandria Engineering Journal* 81 (2023), pp. 371–383.

[76] Hany Mohamed, Ahmed Hamza e Hesham Hefny. «An Efficient Intrusion Detection Approach Using Ensemble Deep Learning models for IoT.» In: *International Journal of Intelligent Engineering & Systems* 16.1 (2023).

[77] Riccardo Lazzarini, Huaglory Tianfield, Prof Charissis et al. «A Stacking Ensemble of Deep Learning Models for IoT Network Intrusion Detection». In: *A Stacking Ensemble of Deep Learning Models for IoT Network Intrusion Detection* (2023).

[78] Mohamed Mahmoud, Mahmoud Kasem, Abdelrahman Abdallah e Hyun Soo Kang. «Ae-lstm: Autoencoder with lstm-based intrusion detection in iot». In: *2022 International Telecommunications Conference (ITC-Egypt)*. IEEE. 2022, pp. 1–6.

[79] Omar Elnakib, Eman Shaaban, Mohamed Mahmoud e Karim Emara. «EIDM: deep learning model for IoT intrusion detection systems». In: *The Journal of Supercomputing* (2023), pp. 1–21.

[80] Hakan Can Altunay e Zafer Albayrak. «A hybrid CNN+ LSTMbased intrusion detection system for industrial IoT networks». In: *Engineering Science and Technology, an International Journal* 38 (2023), p. 101322.

[81] Jiawei Du, Kai Yang, Yanjing Hu e Lingjie Jiang. «Nids-cnnlstm: Network intrusion detection classification model based on deep learning». In: *IEEE Access* 11 (2023), pp. 24808–24821.

[82] Belal Ibrahim Hairab, Mahmoud Said Elsayed, Anca D Jurcut e Marianne A Azer. «Anomaly detection based on CNN and regularization techniques against zero-day attacks in IoT networks». In: *IEEE Access* 10 (2022), pp. 98427–98440.

[83] Jinsi Jose e Deepa V Jose. «Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset». In: *International Journal of Electrical and Computer Engineering (IJECE)* 13.1 (2023), pp. 1134–1141.

[84] Shalaka Mahadik, Pranav M Pawar e Raja Muthalagu. «Efficient Intelligent Intrusion Detection System for Heterogeneous Internet of Things (HetIoT)». In: *Journal of Network and Systems Management* 31.1 (2023), p. 2.

[85] Yi Ren, Kanghui Feng, Fei Hu, Liangyin Chen e Yanru Chen. «A Lightweight Unsupervised Intrusion Detection Model Based on Variational Auto-Encoder». In: *Sensors* 23.20 (2023), p. 8407.

[86] Peng Wang, Xiaodan Wang, Yafei Song, Jieyu Huang, Peng Ding e Zhou Yang. «TransIDS: A Transformer-based approach for intrusion detection in Internet of Things using Label Smoothing». In: *2023 4th International Conference on Computer Engineering and Application (ICCEA)*. IEEE. 2023, pp. 216–222.

[87] Ruizhe Yao, Ning Wang, Peng Chen, Di Ma e Xianjun Sheng. «A CNN-transformer hybrid approach for an intrusion detection system in advanced metering infrastructure». In: *Multimedia Tools and Applications* 82.13 (2023), pp. 19463–19486.

[88] Albara Awajan. «A novel deep learning-based intrusion detection system for IOT networks». In: *Computers* 12.2 (2023), p. 34.

[89] Monika Vishwakarma e Nishtha Kesswani. «DIDS: A Deep Neural Network based real-time Intrusion detection system for IoT». In: *Decision Analytics Journal* 5 (2022), p. 100142.

[90] Idriss Idrissi, Mostafa Mostafa Azizi e Omar Moussaoui. «A lightweight optimized deep learning-based host-intrusion detection system deployed on the edge for IoT». In: *International Journal of Computing and Digital System* (2021).

[91] Liam Daly Manocchio, Siamak Layeghy e Marius Portmann. «Network Intrusion Detection System in a Light Bulb». In: *2022 32nd International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE. 2022, pp. 1–8.

[92] Nazli Tekin, Abbas Acar, Ahmet Aris, A Selcuk Uluagac e Vehbi Cagri Gungor. «Energy consumption of on-device machine learning models for IoT intrusion detection». In: *Internet of Things* 21 (2023), p. 100670.

[93] Giovanni Apruzzese, Luca Pajola e Mauro Conti. «The cross-evaluation of machine learning-based network intrusion detection systems». In: *IEEE Transactions on Network and Service Management* 19.4 (2022), pp. 5152–5169.

[94] Wei Wang, Yiqiang Sheng, Jinlin Wang, Xuewen Zeng, Xiaozhou Ye, Yongzhong Huang e Ming Zhu. «HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection». In: *IEEE access* 6 (2017), pp. 1792–1806.

[95] Raisa Abedin Disha e Sajjad Waheed. «Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique». In: *Cybersecurity* 5.1 (2022), p. 1.

[96] Anjum Farah. «Cross Dataset Evaluation for IoT Network Intrusion Detection». Tesi di dott. The University of Wisconsin-Milwaukee, 2020.

[97] Mantas Bacevicius e Agne Paulauskaite-Taraseviciene. «Machine Learning Algorithms for Raw and Unbalanced Intrusion Detection Data in a Multi-Class Classification Problem». In: *Applied Sciences* 13.12 (2023), p. 7328.

[98] Md Ochiuddin Miah, Sakib Shahriar Khan, Swakkhar Shatabda e Dewan Md Farid. «Improving detection accuracy for imbalanced network intrusion classification using cluster-based under-sampling with random forests». In: *2019 1st international conference on advances in science, engineering and robotics technology (ICASERT)*. IEEE. 2019, pp. 1–5.

[99] David Gonzalez-Cuautle, Aldo Hernandez-Suarez, Gabriel Sanchez-Perez, Linda Karina Toscano-Medina, Jose Portillo-Portillo, Jesus Olivares-Mercado, Hector Manuel Perez-Meana e Ana Lucila Sandoval-Orozco. «Synthetic minority oversampling technique for optimizing classification tasks in botnet and intrusion-detection-system datasets». In: *Applied Sciences* 10.3 (2020), p. 794.

[100] Tommaso Zoppi, Andrea Ceccarelli, Tommaso Puccetti e Andrea Bondavalli. «Which algorithm can detect unknown attacks? Comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection». In: *Computers & Security* 127 (2023), p. 103107.

[101] Xuezhou Zhang, Xiaojin Zhu e Laurent Lessard. «Online data poisoning attacks». In: *Learning for Dynamics and Control*. PMLR. 2020, pp. 201–210.

[102] Youssef Abadade, Anas Temouden, Hatim Bamoumen, Nabil Benamar, Yousra Chtouki e Abdelhakim Senhaji Hafid. «A Comprehensive Survey on TinyML». In: *IEEE Access* (2023).

[103] Ansam Khraisat e Ammar Alazab. «A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges». In: *Cybersecurity* 4 (2021), pp. 1–27.

[104] Maxime Lefrançois. *Introduction to the Internet-of-things*. `https://ci.mines-stetienne.fr/teaching/maj-info/iot/2017/1/`. 2017.

[105] Ankit Thakkar e Ritika Lohiya. «A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions». In: *Artificial Intelligence Review* 55.1 (2022), pp. 453–563.

[106] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee e Ali A Ghorbani. «Toward developing a systematic approach to generate benchmark datasets for intrusion detection». In: *computers & security* 31.3 (2012), pp. 357–374.

[107] Vikash Kumar e Ditipriya Sinha. «Synthetic attack data generation model applying generative adversarial network for intrusion detection». In: *Computers & Security* 125 (2023), p. 103054.

[108] Olivier Flauzac, Carlos González, Abdelhak Hachani e Florent Nolot. «SDN based architecture for IoT and improvement of the security». In: *2015 IEEE 29th international conference on advanced information networking and applications workshops*. IEEE. 2015, pp. 688–693.

[109] Javed Ashraf, Nour Moustafa, Asim D Bukhshi e Abdullah Javed. «Intrusion detection system for SDN-enabled IoT networks using machine learning techniques». In: *2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE. 2021, pp. 46–52.

[110] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin e Kuang-Yuan Tung. «Intrusion detection system: A comprehensive review». In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 16–24. issn: 1084-8045. doi: https://doi.org/10.1016/j.jnca.2012.09.004. url: https://www.sciencedirect.com/science/article/pii/S1084804512001944.

[111] Suad Mohammed Othman, Nabeel T Alsohybe, Fadl Mutaher Ba-Alwi e Ammar Thabit Zahary. «Survey on intrusion detection system types». In: *International Journal of Cyber-Security and Digital Forensics* 7.4 (2018), pp. 444–463.