

**RELATORI:**

**Prof. Francesco PALMIERI**

**Prof. Massimo FICCO**



UNISA - DINF

EVALUATION OF A MCU-DEPLOYED  
SIAMESE NEURAL NETWORK-BASED IDS

# USING A REAL IoT ATTACK DATASET

---

CORSO DI LAUREA  
MAGISTRALE IN INFORMATICA

---

**Alberto Montefusco**  
**Matr.: 0522501498**

1



## LE MOTIVAZIONI

La proliferazione dei dispositivi IoT ha introdotto sfide significative per la sicurezza informatica, rendendo necessari sistemi di rilevamento delle intrusioni (IDS) adatti agli ambienti IoT-edge.

SOLUZIONE	MOTIVO
<b>Addestramento on-board</b>	Addestrare modelli con dati sul campo e ridurre le violazioni della privacy dei dati
<b>Nuovi modelli di rete</b>	Utilizzare modelli capaci di essere addestrati con dataset limitati e sbilanciati
<b>IoT-IDS per MCU</b>	Implementare IDS su piccole unità microcontrollore (MCU)



## LE MOTIVAZIONI

Il progetto valuta le prestazioni di un IDS basato su una Rete Neurale Siamese (SNN) implementata su un un'unità microcontrollore (MCU).

- La SNN è stata valutata su un dataset IoT-Edge di attacchi realistico creato ad hoc e su TON\_IoT. La creazione del nuovo dataset ha incluso:
  - la progettazione di un'**architettura IoT-Edge-Cloud**;
  - la raccolta e la trasformazione del traffico generato.
- Lo studio valuta le prestazioni del sistema attraverso:
  - lo sviluppo di un **IDS** basato su una **SNN**;
  - il **deploy** della **SNN** su una **MCU**.

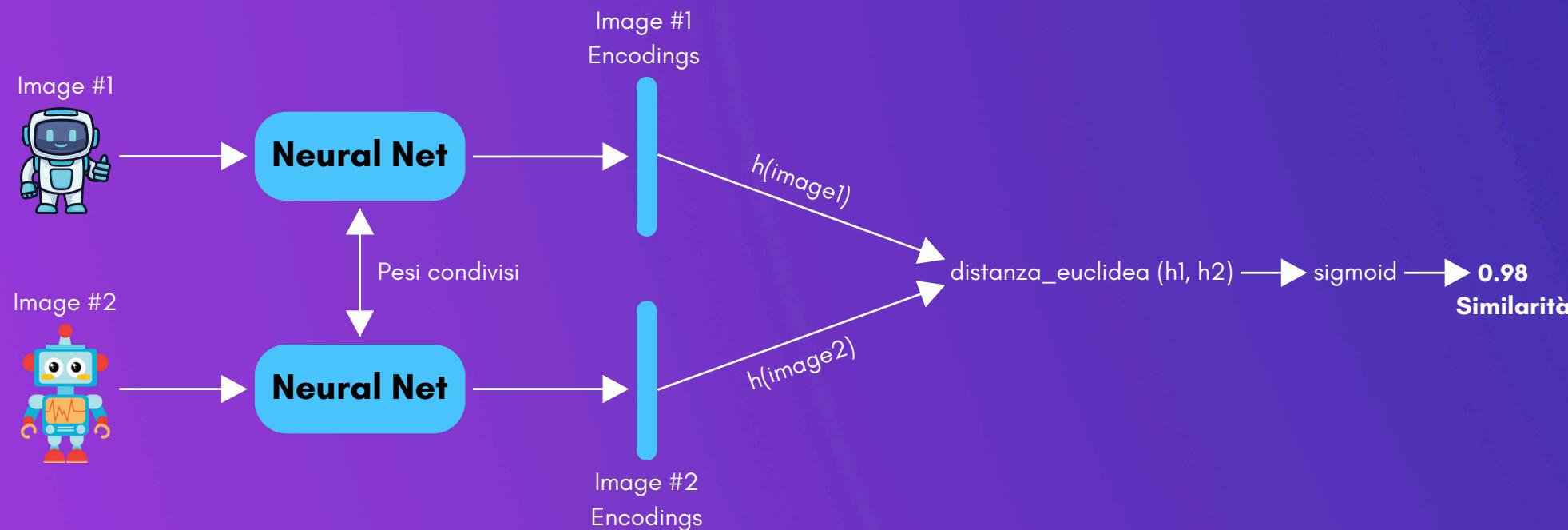


### OBIETTIVO

Implementare un IDS basato su una SNN per l'esecuzione su una piattaforma MCU (ESP32).

## FEW-SHOT LEARNING PER IDS

Le **Reti Neurali Siamesi (SNN)** sono particolarmente adatte al **Few-Shot Learning** e all'uso su dispositivi IoT-edge con risorse limitate perché possono individuare nuovi pattern di attacchi senza richiedere un riaddestramento completo: il modello è in grado di valutare quanto due istanze risultino simili o dissimili, adattandosi a nuovi flussi di traffico di attacchi con un numero limitato di esempi.



### VANTAGGI

- Apprendimento con dataset sbilanciati
- Riduzione di dati etichettati per classe
- Flessibilità nell'aggiunta di nuove classi

### SVANTAGGI

- Periodo di training prolungato

Il "FEW-SHOT LEARNING" è una tecnica di apprendimento automatico che consente di addestrare modelli utilizzando un numero estremamente limitato di dati etichettati.

4

## > TESTBED SVILUPPATO

Al fine di acquisire il traffico di rete per la costruzione del dataset, è stato implementato un testbed IoT-edge rappresentativo di un ambiente domestico automatizzato.

Esso comprende una porta intelligente dotata di un lettore di impronte digitali, un sensore di allarme e un rilevatore di movimento in grado di rilevare i movimenti davanti all'ingresso.

I dati generati da questi dispositivi, accessibili tramite un'applicazione dedicata, sono archiviati da un servizio cloud.

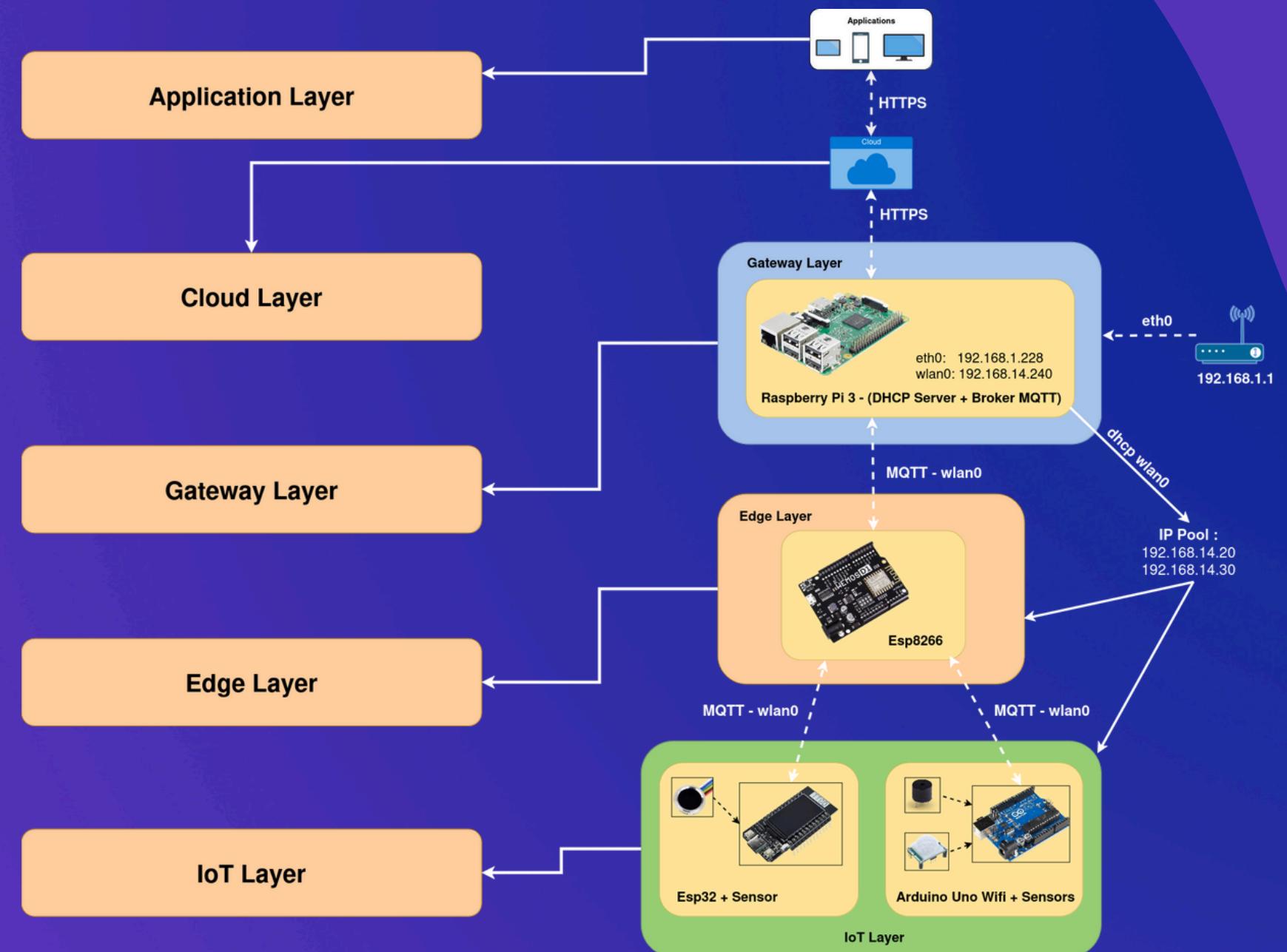


## 5

# TESTBED SVILUPPATO

La logica dell'architettura è suddivisa in 5 livelli:

- 1. IoT Layer:** i dati raccolti dai sensori vengono trasmessi al livello Edge tramite protocollo MQTT.
- 2. Edge Layer:** gestisce lo smistamento delle informazioni ricevute sia dal livello Gateway che dai dispositivi IoT, instradandole in maniera bidirezionale verso il Gateway o i sensori.
- 3. Gateway Layer:** riceve le informazioni dallo strato Edge e le invia al Cloud. Inoltre, riceve dati aggiornati dal Cloud e li inoltra allo strato Edge.
- 4. Cloud Layer:** archivia le informazioni ricevute su Google Drive, garantendo la conservazione e l'accessibilità dei dati.
- 5. Application Layer:** permette di gestire le azioni dei client e di aggiornare le informazioni archiviate su Google Drive.



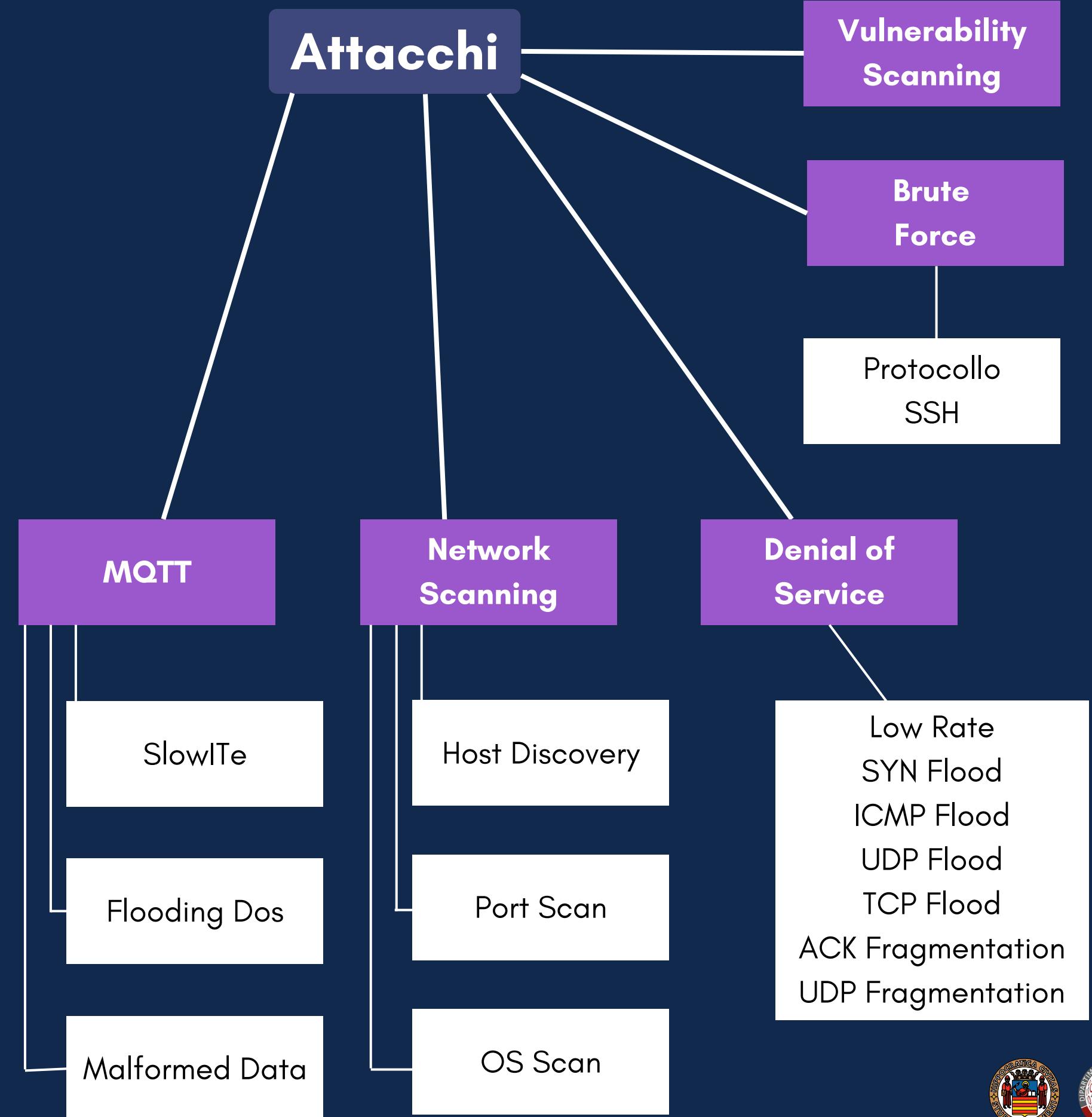
6

## FEW-SHOT IOT-EDGE ATTACK DATASET

Sono stati condotti diversi attacchi informatici suddivisi in quattro classi principali:

1. **Denial of Service** ➤ Tool: hping3
2. **Reconnaissance** ➤ Tool: nmap, Nessus, Vulscan
3. **Brute Force** ➤ Tool: hydra
4. **Attacchi MQTT** ➤ Tool: mqtt-malaria, mqtsa

La scelta di questi attacchi è motivata dalla loro prevalenza e natura generica nel panorama della sicurezza informatica, che li rende specializzati per gli scenari IoT-edge. Per ogni classe sono stati condotti più attacchi in modo che il dataset sia completo e rifletta situazioni reali.



7

## FEW-SHOT IOT-EDGE ATTACK DATASET

Il flusso di traffico viene acquisito utilizzando due PC collegati all'access point del Raspberry Pi:

1. uno dedicato all'esecuzione degli attacchi;
2. l'altro configurato in modalità monitor e dotato di Wireshark per la registrazione passiva del traffico.



8



## FEW-SHOT IOT-EDGE ATTACK DATASET

Le scansioni del traffico di rete alternano periodi di traffico normale e di attacco.

Il **traffico normale** è suddiviso in:

- » traffico "**idle**";
- » traffico "**normale**"

Le scansioni di attacco iniziano e terminano con due minuti di traffico legittimo, includendo una fase di attacco variabile in base al tipo di minaccia.

Scansione	Tempo iniziale	Tempo dell'Attacco	Tempo finale
Traffico Normale	5 hrs	<u>nessuno</u>	<u>nessuno</u>
Brute Force	2 min	<u>variabile</u>	2 min
Recon Attack	2 min	<u>variabile</u>	2 min
DoS Attack	2 min	1 min	2 min
MQTT Attack	2 min	2 min	2 min



9

## TRASFORMAZIONE DEI DATASET

Dopo aver completato le scansioni, i passaggi successivi prevedono per **TON\_IoT** e **Few-Shot\_IoT**:

### › Conversione del dataset in file CSV

Solo per **TON\_IoT**:

### › Filtraggio dei dati

### › Bilanciamento delle classi

## Few-Shot\_IoT - TON\_IoT



### Conversione del dataset in file CSV

Dopo l'esecuzione delle scansioni, il dataset viene prima convertito in un file CSV utilizzando lo strumento tshark con uno script Python.

## TON\_IoT



### Filtraggio dei dati

Il dataset viene raffinato per includere solo i protocolli con più di 1.000 campioni, garantendo la rilevanza e riducendo la complessità.



### Bilanciamento delle classi

Vengono selezionati casualmente tanti campioni quanti sono quelli della classe meno rappresentata per un totale di 34.601 campioni per classe.



10

## SELEZIONE DELLE FEATURES

Le **32** feature selezionate rappresentano aspetti chiave del traffico di rete.

Tra queste, i flag TCP (es. SYN, ACK, FIN, RST) e il tempo di arrivo dei pacchetti (IAT) sono fondamentali per rilevare anomalie.

Nel protocollo MQTT, feature come Clean Session, Quality of Service (QoS) e Retain Flag sono cruciali per monitorare connessioni e individuare attacchi come MQTT Flood o Session Hijacking.

Feature in Tshark	Feature in Dataset	Description
type_attack	Attack_Type	Type of attack or benign traffic
rate	Rate	Rate of packet transmission
ip.ttl	Time_To_Leave	Number of hops allowed before packet is discarded
ip.hdr_len	Header_Length	Length of the IP header
_ws.col.Protocol	Protocol_Type	Type of protocol used
tcp.flags.fin	TCP_Flag_FIN	FIN flag for TCP connection termination
tcp.flags.syn	TCP_Flag_SYN	SYN flag for TCP connection initiation
tcp.flags.reset	TCP_Flag_RST	RST flag indicating connection reset
tcp.flags.push	TCP_Flag_PSH	PSH flag for pushing data to the receiver
tcp.flags.ack	TCP_Flag_ACK	ACK flag acknowledging packet receipt
tcp.flags.ece	TCP_Flag_ECE	ECE flag for congestion notification
tcp.flags.cwr	TCP_Flag_CWR	CWR flag for congestion window reduced
frame.len	Packet_Length	Length of the entire packet
frame.time_delta	IAT	Inter-arrival time between packets
ip.flags.mf	Packet_Fragments	Indicates whether the packet is fragmented
tcp.len	TCP_Length	Length of TCP payload
mqtt.conack.flags	MQTT_ConAck_Flags	Flags for MQTT connection acknowledgment
mqtt.conflag.cleansess	MQTT_CleanSession	Clean session flag for MQTT
mqtt.qos	MQTT_QoS	Quality of Service level for MQTT messages
mqtt.conflag.reserved	MQTT_Reserved	Reserved flag in MQTT connection
mqtt.retain	MQTT_Retain	Retain flag for MQTT messages
mqtt.conflag.willflag	MQTT_WillFlag	Will flag indicating last will in MQTT
mqtt.conflags	MQTT_ConFlags	Connection flags for MQTT
mqtt.dupflag	MQTT_DupFlag	Duplicate flag for retransmitted MQTT messages
mqtt.hdrflags	MQTT_HeaderFlags	Header flags in MQTT messages
mqtt.kalive	MQTT_KeepAlive	Keep-alive interval for MQTT connections
mqtt.len	MQTT_Length	Length of MQTT payload
mqtt.msagtype	MQTT_MessageType	Type of MQTT message
mqtt.proto_len	MQTT_Proto_Length	Length of MQTT protocol name
mqtt.conflag.qos	MQTT_Conflag_QoS	Quality of Service flag for MQTT connection
mqtt.conflag.retain	MQTT_Conflag_Retain	Retain flag for MQTT connection
mqtt.ver	MQTT_Version	Version of the MQTT protocol



## RISULTATI

---

I risultati sono stati confrontati con:

- Classificatori di ML tradizionali (SVM, KNN, RF);
- Rete Neurale Convoluzionale (CNN)
  - **Parametri 453253**
- Rete Neaurale Siamese (SNN)
  - **Parametri: 453728**

Utilizzando il dataset generato dal testbed di prova e TON\_IoT.

### OBIETTIVO

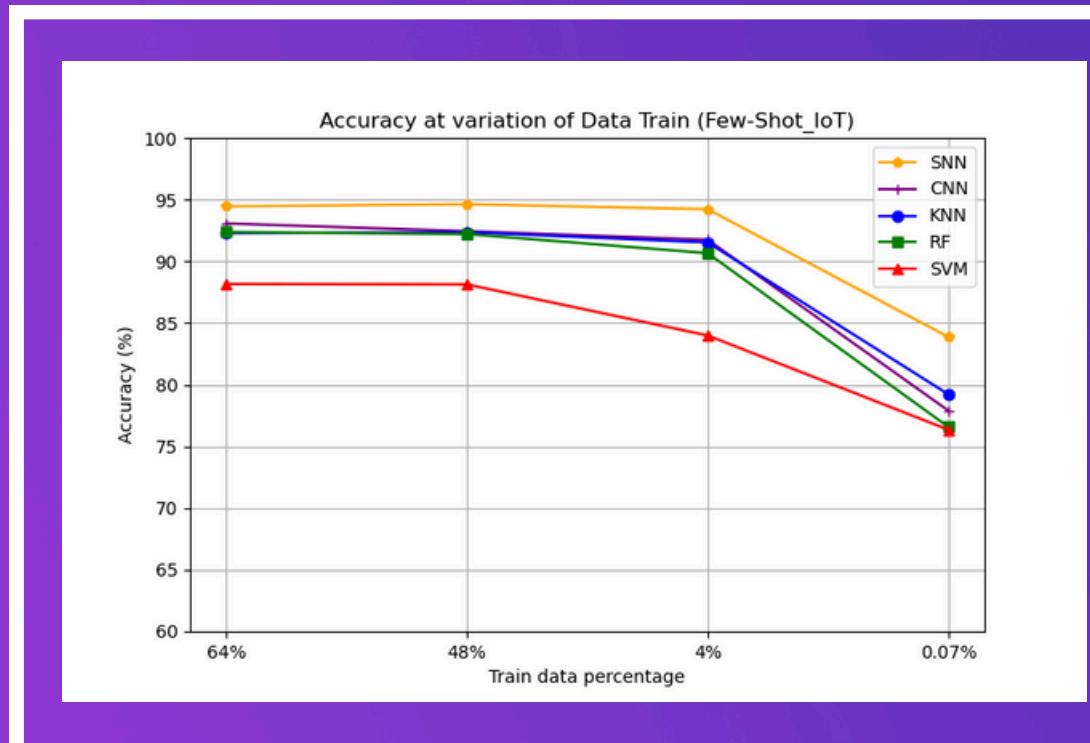
Analizzare come le prestazioni dei modelli degradano al diminuire della quantità di dati utilizzati durante il processo di addestramento per dimostrare l'efficacia delle SNN su dataset ridotti e sbilanciati.



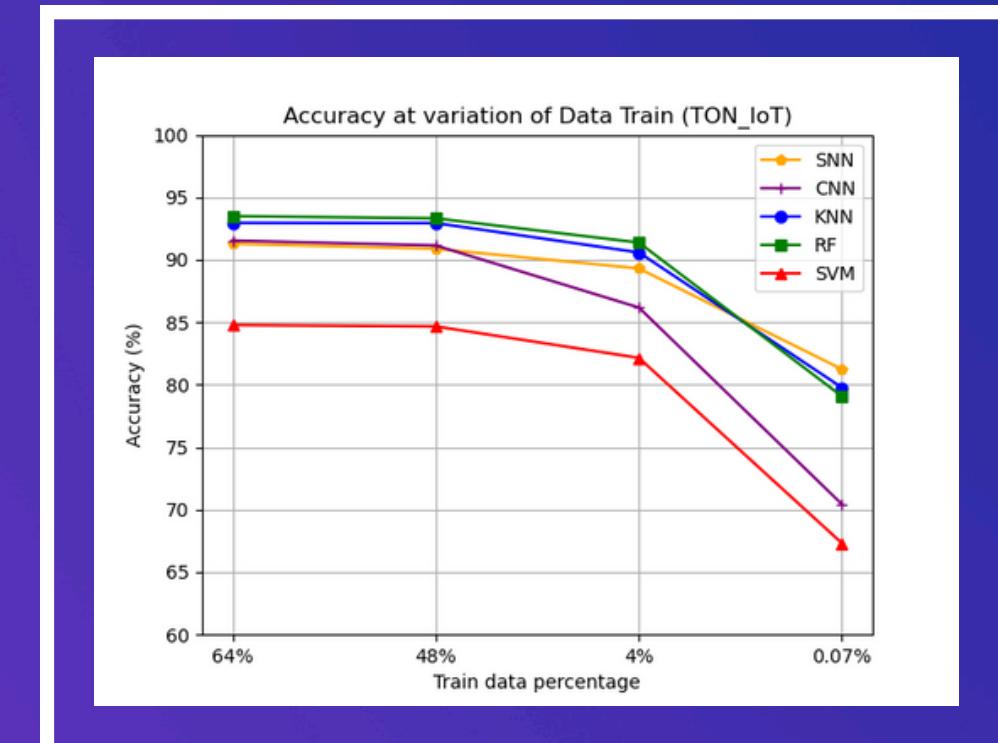
## RISULTATI

I risultati dimostrano l'efficacia della SNN nel mantenere elevate le prestazioni anche quando la quantità di dati di training viene ridotta e soprattutto quando il dataset è sbilanciato. A differenza di altri modelli, la Rete Siamese utilizza come dati di training coppie uniche generate a partire da tutte le possibili combinazioni dei campioni disponibili. Questo approccio la rende particolarmente vantaggiosa in scenari in cui i dataset sono limitati e sbilanciati.

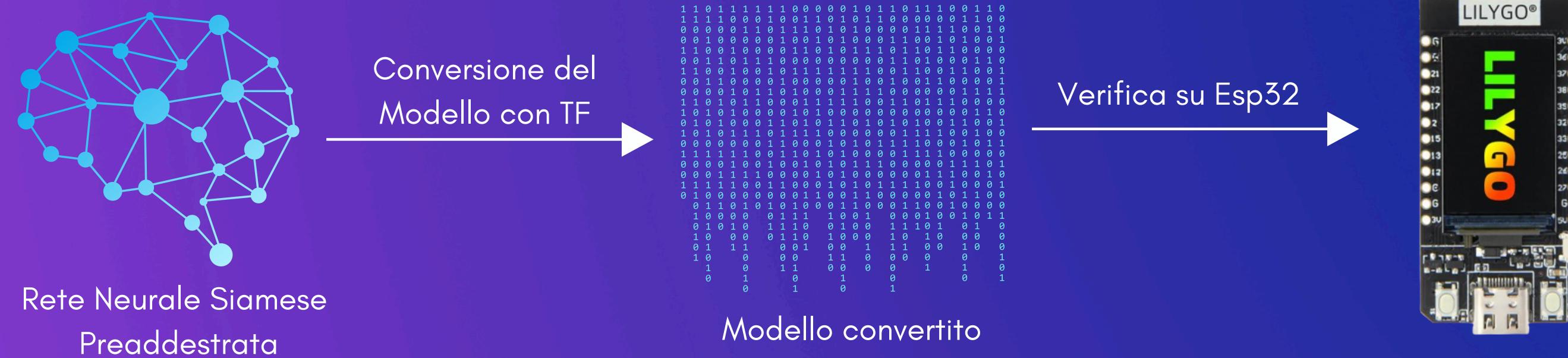
**DATASET SBILANCIATO**



**DATASET BILANCIATO**



# DEPLOY DELLA SNN SU MCU



La SNN è stata implementata su una piattaforma MCU per valutare i tempi di inferenza. È stato scelto l'ESP32 per la sua compatibilità con **TensorFlow Lite**: il modello neurale è stato convertito in un file "model.h", poi sono state generate 100 coppie di test dal dataset creato, che sono state poi tradotte in linguaggio C e caricate sulla scheda.

## ➤ DEPLOY DELLA SNN SU MCU

I risultati indicano che l'accuratezza ottenuta è paragonabile a quella del modello eseguito localmente su un laptop, tuttavia i tempi di inferenza risultano troppo elevati per garantire un'efficace rilevazione di anomalie in tempo reale.

Metrica	Avg	Std Dev	Min	Max
Acc (%)	95.00	-	-	-
Pred Time (s)	7.5036	0.0000	7.5027	7.5040

# CONCLUSIONI

## » **Risultati:**

Il Few-Shot Learning per l'addestramento on-board su dispositivi IoT-Edge è efficace anche in situazioni in cui i dataset sono sbilanciati.

## » **Limitazioni nello studio:**

- Elevato tempo di inferenza

## » **Sviluppi futuri:**

- Ampliare il dataset con nuovi attacchi non solo di rete
- Alleggerire il modello SNN tramite operazioni di pruning e quantizzazione
- Sviluppo di un tool per covertire reti neurali in maniera semplice e veloce





**GRAZIE**



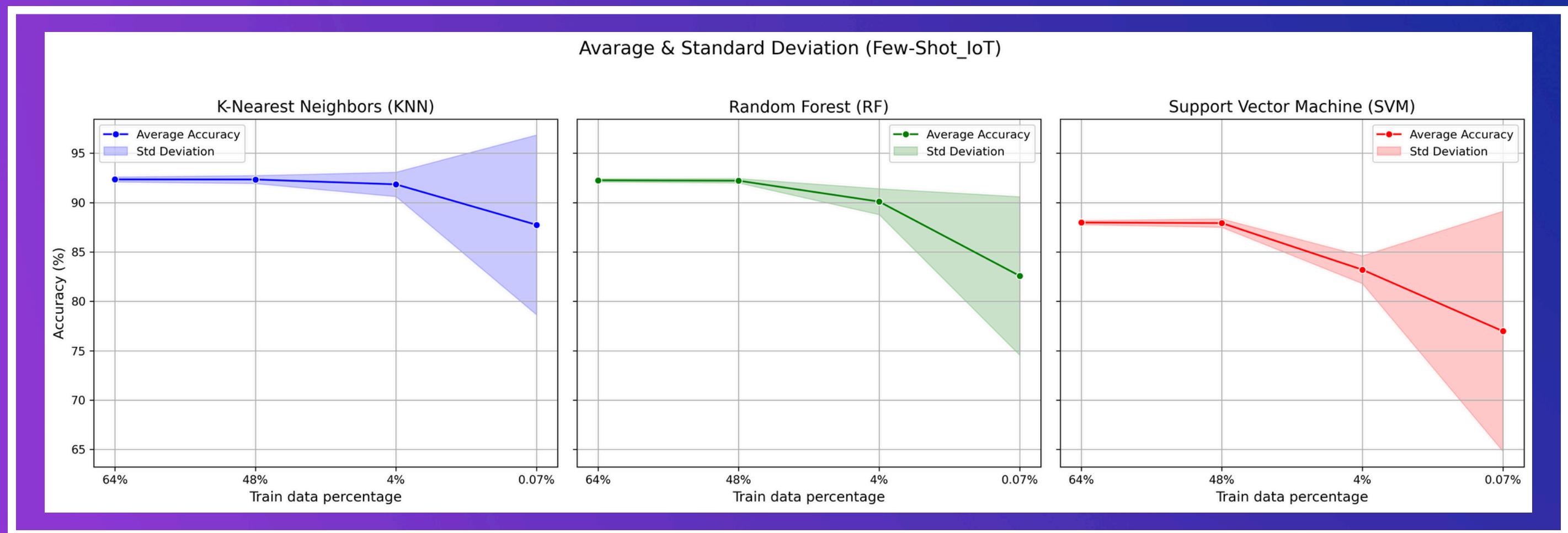
**UNISA - DINF**



## RISULTATI

---

### DATASET SBILANCIATO





# RISULTATI

---

## DATASET BILANCIATO

