Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Dipartimento di Informatica – Scienza e Ingegneria

Corso di Laurea in Informatica

Wi-Fi Sensing per Human Identification attraverso dispositvi ESP32

Relatore:
Dr. Andrea Melis

Presentata da:
Fabio Gaiba

Correlatori:
Prof. Dr. Luca Bedogni

Sessione
Anno Accademico 2022/2023

Abstract

Recenti studi esplorano la possibilità di rilevare eventi in una stanza tramite Wi-Fi Sensing. Tale pratica sfrutta l'interazione fra le onde che trasportano segnali Wi-Fi e gli elementi presenti in un ambiente. Queste interazioni vengono dette Channel State Information (CSI) e possono essere analizzate e sfruttate per inferire informazioni sull'ambiente, come localizzazione "device-free", riconoscimento di attività HAR, Human Identification e altro. Per quanto riguarda l'identificazione, non sono mai stati utilizzati dispositivi di fascia bassa come gli ESP32. Essendo piccoli e a basso consumo sono molto versatili, tuttavia la qualità dei dati raccolti è inferiore. In questa tesi, si utilizzano strumenti d'avanguardia per effettuare Human Identification tramite ESP32. Si crea un software che funga da interfaccia fra i dati raccolti e gli algoritmi atti al Wi-Fi Sensing. Per valutare il progetto finale, viene effettuata una raccolta dati in ambiente controllato. Lo strumento mostra un'accuratezza del 95% nel distinguere due utenti mentre del 74% nel distinguerne tre.

"A sound soul
dwells within a sound mind
and a sound body"

Atsushi Okubo

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Vi è un crescente interesse e una sempre maggiore necessità di dotare stanze, uffici e luoghi al chiuso di sistemi capaci di rilevare la presenza di persone. Con la larga diffusione di sistemi Wi-Fi emergono nuove soluzioni per effettuare tali rilevamenti. Il Wi-Fi Sensing è una tecnica di rilevamento sempre più popolare, che riutilizza infrastrutture già esistenti (per la connessione wireless) e consente rilevazioni "device-free".

I dispositivi OFDM, che consentono le connessioni wireless, necessitano di calcolare ogni istante dati CSI. Questi dati catturano le variazione che il segnale subisce mentre viaggia sotto forma di onda in un ambiente. Vengono impiegati dalla tecnologia OFDM per mantenere la qualità della connessione, ma possono essere anche utilizzati per inferire informazioni su che cosa accade nell'ambiente. Tale pratica viene comunemente chiamata Wi-Fi Sensing.

I rilevamenti che possono essere implementati tramite WiFi Sensing sono piuttosto vari. Vanno dal riconoscimento di gesti e attività sino alla stima di parametri biometrici. Le applicazioni sono altrettanto eterogenee. Si parla di spazi intelligenti manovrabili con gesti degli individui senza che questi debbano avere device appositi, alleggerendo gli utenti dalla necessità di indossare/portare dispositivi personali. Oltre che spazi intelligenti, l'impiego del WiFi Sesning ha dimostrato un grande potenziale in scenari di sicurezza e prevenzione di intrusioni. Ci sono anche numerose applicazioni per quan-

to riguarda il monitoraggio di situazioni sanitarie, rilevando segnali vitali e possibili incidenti come cadute o inciampi.

Il problema dell'identificazione delle persone tramite Wi-Fi Sensing è ancora abbastanza inesplorato. Più nello specifico, i sistemi di Human Identification all'avanguardia impiegano dispositivi per la raccolta dati COTS. Sebbene tali dispositivi siano commerciali, non ci sono ricerche di Human Identification che sfruttano dispositivi a basso costo e consumo come gli ESP32.

In questa tesi viene realizzato uno strumento che unisce due tool, uno per la raccolta dati tramite dispositivi ESP32 e uno per effettuare WiFi Sensing a partire da questi dati. Vengono poi raccolti dei dati in alcuni esperimenti e viene valutata l'efficacia dello strumento nell'ambito di Human Identification.

Sono condotte due raccolte dati. La seconda, più estesa e meglio organizzata, mostra risultati molto promettenti. Nel caso con due utenti l'accuratezza del 95% è comparabile a quella di strumenti già ampiamente testati e diffusi. Con tre utenti scende sensibilmente a 74%, risultato comunque molto promettente alla luce dell'utilizzo degli ESP32, che comunque garantiscono una qualità inferiore, e della scarsità di dati.

1.1 Struttura del Documento

In **Scenari applicativi e Stato dell'Arte** vengono affrontati i concetti principali che permettono il WiFi Sensing. Si tratta il calcolo delle matrici CSI e la loro gestione. Successivamente si parla delle procedure di design di algoritmi relativi al WiFi Sensing. Si vedono nel dettaglio tecnologie recenti e d'avanguardia come SHARP e WiFi-CSI-tool ed infine si discute di una serie di ricerche nell'ambito di Human Identification.

In **Analisi progettuale e implementazione** si discutono gli obiettivi principali di questa tesi e di come si sia deciso di procedere nel raggiungerli. Dopo uno sguardo generale sulla realizzazione del progetto si entra più nel

dettaglio identificando le soluzioni necessarie per completare lo strumento. Dopodiché viene mostrata l'implementazione finale.

In **Valutazioni Empiriche** si descrivono sinteticamente gli esperimenti di raccolta dati effettuati. Dopodiché vengono mostrati i risultati della classificazione, valutando le performance e fornendo considerazioni sulla loro natura.

In **Conclusioni e sviluppi futuri** si cerca di trarre delle risposte dai risultati del capitolo precedente. Si prova a rispondere alle domande poste durante la fase di progettazione e infine si affrontano i possibili miglioramenti attuabili in futuro.

Capitolo 2

Scenari applicativi e stato dell'arte

Nel seguente capitolo vengono fornite le conoscenze di base ampiamente utilizzate in tutta la tesi, nonché fondamentali per affrontare l'argomento trattato. Si parte con una breve introduzione al *Wi-Fi sensing* e alle tecniche di raccolta dati CSI. Si passa poi nello specifico ai tipi di algoritmi impiegati in letteratura per il *Wi-Fi sensing*. Successivamente si trattano strumenti allo stato dell'arte come *SHARP* e *ESP32-CSI-Tool*, impiegati nella fase di progettazione e implementazione. Infine si ricapitolano le principali ricerche riguardanti Human Identification.

2.1 Wi-Fi Sensing

Con Wi-Fi si intende una famiglia di protocolli basati sullo standard IEEE 802.11 che permettono comunicazioni wireless. Lo scambio di informazioni avviene tramite onde radio, che attraversando una stanza, interagiscono con gli elementi circostanti. Questa interazione causa modifiche nel segnale. Nel **Wi-Fi Sensing** vengono analizzate tali modifiche per estrarre informazioni riguardanti gli elementi presenti in un determinato luogo. Il WiFi Sensing

riutilizza l'infrastruttura usata per le comunicazioni wireless, per cui è facile da distribuire ed ha un basso costo [1].

Negli scenari di Wi-Fi Sensing dove è l'essere umano l'elemento da rilevare, possono essere inferite una grande varietà di informazioni: movimenti, gesti e informazioni biometriche (battiti del cuore, respiro, etc...). Queste rilevazioni sono anche denominate **Device-Free Sensing**, proprio perché non necessitano di un dispositivo posizionato sulla persona. Le applicazioni sono svariate, alcuni esempi:

1. Riconoscere gesti e movimenti per effettuare operazioni su sistemi come Smart Home (accendere la luce o abbassare il volume della musica tramite un gesto).

2. Sorveglianza e sicurezza laddove strumenti visivi tradizionali come telecamere non sono in grado di rilevare gli intrusi (ad esempio per mancanza di luce, o per mancanza di copertura visiva del luogo). Tecnologie all'avanguardia come *DeMan* [2] presentano una soluzione poco invasiva e piuttosto accurata per rilevare persone stazionarie o in movimento attraverso segnali biometrici.

3. Assistenza Sanitaria, ovvero rilevazione non invasiva di segnali vitali e anche rilevazioni di cadute accidentali in casi in cui è preferibile non indossare dispositivi invasivi. Nella letteratura scientifica sono presenti risultati molto promettenti come *FallDeFi* [3] che raggiungono accuratezze sopra il 90%.

2.2 CSI e Channel Estimation

2.2.1 Channel State Information

Nell'ambito delle telecomunicazioni, con OFDM ci si riferisce ad un tipo di trasmissione digitale di segnale che permette di trasmettere il segnale su più carrier, o in altre parole su più frequenze. In breve, la trasmissione

viene suddivisa in in N flussi paralleli, anche detti subcarrier, scelti in modo ortogonale l'un l'altro così da minimizzare il "cross talk".

Siano Tx e Tr rispettivamente l'antenna trasmettitrice e l'antenna ricevente. Il CSI (Channel State Information) è una metrica usata nei sistemi OFDM che descrive in che modo il segnale (con granularità a livello di subcarrier) si propaga da Tx a Tr attraverso oggetti e persone circostanti. L'ampiezza e la fase del CSI sono influenzate dai fenomeni di attenuazione dell'ampiezza e spostamento di fase[1]. È proprio da questi cambiamenti nel segnale che si possono inferire informazioni sull'ambiente.

2.2.2 Channel Estimation

Channel estimation è il processo tramite il quale vengono stimate le variazioni di segnale nei vari subcarrier. Uno delle tecniche più popolari per fare ciò è quella di trasmettere un insieme di *pilot symbol* che siano noti al ricevente. Un *pilot symbol* è un simbolo completo OFDM (un simbolo può essere descritto come un impulso nella trasmissione digitale) dove i valori di ogni subcarrier sono predefiniti e condivisi sia dal trasmettitore sia dal ricevente.

2.2.3 Definizione operativa di CSI

La matrice CSI H può essere stimata come:

$$y = Hx + n$$

dove y è il vettore del segnale ricevuto, x è il vettore del segnale trasmesso (basato sui *pilot symbols* di comune accordo) e n è un vettore di rumore Gaussiano. Più precisamente H è una matrice di numeri complessi, uno per ogni subcarrier i. Il numero complesso h_i è composto da una parte reale $R(h_i)$ e una parte immaginaria $I(h_i)$. Rappresenta il CFR (Channel Frequency Response) nel seguente modo:

$$h_i = A_i e^{j\phi_i}$$

dove A_i è l'ampiezza del subcarrier i mentre ϕ_i è la sua fase [4].

L'ampiezza e la fase possono essere estratte da h_i con le seguenti equazioni:

$$A_i = \sqrt{I(h_i)^2 + R(h_i)^2}$$

$$\phi_i = \text{atan2}(I(h_i), R(h_i))$$

In altre parole ogni numero complesso h_i che compone la matrice H CSI rappresenta l'impatto che oggetti e persone hanno sulla fase e l'ampiezza del segnale a una certa frequenza f del subcarrier i. Nella maggior parte delle ricerche, i termini CFR e CSI vengono generalmente utilizzati in modo interscambiabile, ed entrambi si riferiscono alla matrice H.

2.2.4 Limitazioni e Pre-Processing

Nei sistemi Wi-Fi reali, le misurazioni CSI sono impattate dalla propagazione multi-path, l'elaborazione del segnale da parte del trasmettitore e del ricevente e da errori hardware e software. Le applicazioni di *WiFi sensing* devono estrarre il canale multi-path che contiene le informazioni degli eventi da rilevare, per cui nella letteratura vengono impiegate un grande varietà di tecniche di processazione del segnale. In generale le tecniche si dividono in 3 categorie[4]:

1. **noise reduction**, ovvero riduzione del rumore. I dati CSI "raw" presentano (come descritto nella definizione operativa) una certa quantità di rumore. Inoltre sono presenti outiliers che potrebbero ridurre in modo sensibile le performance delle applicazioni di WiFi Sensing. Le strategie più popolari sono la rimozione degli outlier e rimozione dell'offset della fase. La seconda strategia menzionata punta a rimuovere gli scostamenti di fase dovuti non a un elemento nell'ambiente ma a errori hardware e software. Ad esempio, gli scostamenti di tempo/frequenza di campionamento (STO/SFO) sono causati da clock e/o frequenze di campionamento non sincronizzati tra ricevitore e trasmettitore.

2. **signal transform**, ovvero trasformazione del segnale. Sono metodi per l'analisi tempo-frequenza di una serie temporale di misure CSI. Si tenga presente che l'output della trasformazione del segnale in questo contesto rappresenta la frequenza dei pattern di variazione del CSI, piuttosto che la frequenza del carrier.

3. **signal extraction**, cioè una serie di tecniche per estrarre segnali target da delle misurazioni CSI (sia "raw" che pre-processate). Sono utilizzate sia metodi di *filtering* e *thresholding* sia metodi di compressione del segnale. Queste ultime vengono utilizzate anche per diminuire di dimensionalità i dati raccolti, che facilmente arrivano ad occupare centinaia di Gigabyte, diminuendo le risorse richieste (sia di computazione che di storage).

2.3 Raccolta Dati

2.3.1 Strumentazione per la raccolta dati

I recenti avanzamenti nelle tecnologie wireless rendono possibile campionare dati CFR da dispositivi commerciali a basso costo, rendendo le soluzioni di Wi-Fi Sensing molto appetibili e facilmente distribuibili. Di seguito un breve elenco dei software (e relativi apparecchi hardware) più utilizzati nella letteratura:

- La maggior parte delle ricerche sfruttano[1] *Linux 802.11n CSI Tool* [5], che fa affidamento sulla Network Interface Card Intel 5300.

- Atheros CSI Tool [6] è un software usato frequentemente basato su NIC Qualcomm Atheros WiFi.

- Un altro strumento popolare è *Nexmon* [7] che permette ad ogni device con chipset Broadcom di collezionare dati CSI, anche se richiede di modificare il firmware (soluzione abbastanza invasiva che potrebbe presentare danni hardware).

2.3.2 ESP32 e ESP32-CSI-Tool

In questa tesi è stato selezionato come strumento di raccolta dati *ESP32-CSI-Tool* [8]. *ESP32-CSI-Tool* si basa sul framework **EXP-IDF** per chip **ESP32**[9].

Gli ESP32 sono una serie di microcontroller a basso consumo energetico e basso costo con Wi-Fi e Bluetooth integrato. Il framework EXP-IDF permette lo sviluppo di applicazioni su questi chip. La natura di basso consumo e dimensione di questi chip li rende perfetti per essere posizionati a piacere in una stanza e collegarli a dispositivi mobili o facilmente spostabili come smartphone o Raspberry-PI.

Negli ESP32 le informazioni CSI consistono di CFR dei sub-carrier e sono stimati quando i pacchetti vengono ricevuti dal trasmettitore. Ogni risposta nella frequenza di un determinato sub-carrier è registrata in due byte che corrispondono a due unsigned char (in riferimento al linguaggio di programmazione C).

ESP32-CSI-Tool permette di configurare i chip ESP32 in modo tale che stabiliscano una connessione a banda 40MhZ (un chip farà da access point mentre gli altri si connetteranno a esso) e raccolgano dati CSI calcolati a partire dai simboli OFDM LLTF (legacy long training field) tramite porta seriale.

Il tool produce 128 byte di informazioni per pacchetto CSI (oltre ad altre informazioni accessorie come RSSI, timestamp, etc...), che stando alla documentazione di ESP-IDF si traducono in 64 valori CFR. Dopo un'attenta analisi del codice sorgente e alcune domande fatte al ricercatore che si occupa di questo tool occorre fare alcune precisazioni. Con i chip ESP32 si è soggetti a diverse limitazioni, fra cui limitazioni nella *data resolution* e *data throughput*. Se si volessero passare dei dati floating point attraverso la porta seriale ci sarebbe una saturazione del canale. Perciò viene effettuata una quantizzazione dei dati (ovvero un cast a **int8_t**) prima che essi siano raccolti.

Il tool è anche provvisto di alcuni programmi preconfezionati che estrag-

gono dai byte "raw" prima i valori CFR per poi visualizzare l'ampiezza dei valori CSI raccolti in tempo reale.

2.4 Algoritmi per il Wi-Fi Sensing

La larga maggioranza delle ricerche nell'ambito del Wi-Fi Sensing, dopo una fase di raccolta dati e di elaborazione del segnale affrontano la scelta di un algoritmo per effettuare inferenze sui dati. I tipi di algoritmi sono principalmente due: algoritmi basati sulla modellazione e algoritmi basati sull'apprendimento automatico. Esiste anche una terza categoria, che mischia i due approcci ottenendo algoritmi ibridi.

L'obiettivo di questi algoritmi è quello di trovare la funzione di mapping $f(\cdot)$ tale che

$$Y = f(X)$$

dove X sono le misurazioni CSI e Y le inferenze che vogliamo effettuare.

Gli algoritmi della prima categoria puntano a modellare X attraverso modelli teorici basati su teorie fisiche o modelli statistici basati su misurazioni empiriche. La funzione f si ottiene dal modello scelto per X. D'altro canto, per problemi di classificazione (sia essa binaria o multi classe) gli algoritmi orientati all'apprendimento sono generalmente la scelta più popolare. Questo tipo di algoritmi cerca di evincere la funzione di mapping $f(\cdot)$ da un insieme sufficientemente ampio di dati di training X' con rispettivo truth label Y'. L'obiettivo è quello di generalizzare il rapporto fra X' e Y' per poter effettuare previsioni su nuovi dati X.

2.4.1 Vantaggi, Svantaggi e Use Case

Entrambi gli approcci presentano una gamma di vantaggi e svantaggi, per cui il loro impiego comporta una scelta strategica in base all'uso richiesto. Di seguito un breve riassunto che presenta i vantaggi e svantaggi delle soluzioni basate su modellazione[1]:

- Vantaggi:

    1. necessitano di un piccolo (se non nullo) dataset di training e di conseguenza non c'è il bisogno di collezionare molti dati con annesso truth label. Inoltre solitamente non vi è una fase di training, velocizzando ancora di più il processo di creazione della funzione f.
    2. gli algoritmi sono in genere più semplici e hanno costo computazionale basso.

- Svantaggi:

    1. richiede parecchio sforzo costruire e scegliere il modello adatto al problema e trovare i parametri per tale modello.
    2. per avere buone performance, richiede misurazioni accurate e di conseguenza molto preprocessing
    3. essendo per sua natura costruito su misura per ogni caso specifico, è solitamente poco riutilizzabile e versatile. Difficilmente si riesce a scalare o riadattare per nuovi compiti o nuovi ambienti.

Per gli algoritmi basati sull'apprendimento:

- Vantaggi:

    1. Richiedono pochissima elaborazione del segnale (dal momento che solitamente gli algoritmi sono in grado di scovare pattern anche se è presente del rumore)
    2. Possono migliorare all'aumentare del dataset (che potrebbe non essere molto esteso in una prima fase di training).
    3. Nel caso del Deep Learning come categoria di algoritmi di apprendimento automatico (e anche altri), è riutilizzabile (ovvero non c'è bisogno di reiterare da zero nel processo di apprendimento, si può partire da un modello a continuare ad allenarlo con dati nuovi) e

versatile (posso utilizzare modelli allo stato dell'arte allenandoli con dati nuovi per affrontare un nuovo problema).

- Svantaggi:

  1. Collezionare il dati per il training (con relativo truth label) richiede parecchi sforzi.
  2. C'è bisogno di molti dati in situazioni diverse per evitare il facile problema dell'overfitting (l'algoritmo tende ad apprendere pattern presenti nel training dataset che però non generalizzano, avendo buone performance durante il training ma non così buone durante la fase di validazione).
  3. Tendenzialmente la fase di training richiede molte risorse computazionali.
  4. alcuni tipi di algoritmi hanno un importante richiesta di risorse durante la fase di inferenza che li rende particolarmente lenti ad effettuare previsioni (ad esempio **kNN**)

Alla luce di queste informazioni, gli **use case** delle due tipologie possono essere delineati come segue:

- gli algoritmi basati su modellazione sono preferibili nei problemi di **stima**, ovvero quando lo scopo è quantificare una qualsiasi misura (lunghezza, numero di respiri, altezza)

- gli algoritmi basati su apprendimento sono più appropriati in problemi di classificazione, suddivisi in letterature come *detection* nel caso binario (stabilire se una persona c'è o meno, se un evento si verifica o meno) o *recognition* nel caso multi classe (quale persona è, quale gesto sta facendo, quale attività sta eseguendo).

2.4.2 SHARP

Fra i vari problemi che possono essere affrontati con tecniche di Wi-Fi Sensing uno di questi è il riconoscimento delle attività effettuate da individui.

Questa classe di problemi è anche conosciuta come HAR (Human Activity Recognition). Le applicazioni sono fra le più varie e vanno dall'implementare sistemi energetici (climatizzazione dell'ambiente) "smart" sino a soluzioni di assistenza sanitaria per monitorare persone anziane o situazioni critiche.

SHARP, che sta per **Sensing Human Activities through Wi-Fi Radio Propagation**, è un sistema all'avanguardia "device free" per la classificazione automatica di attività umane in spazi al chiuso. Questa sistema HAR sfrutta strumentazioni Wi-Fi tradizionali per analizzare i dati CFR. A differenza degli sistemi dediti a risolvere lo stesso problema di classificazione, SHARP raggiunge accuratezze sopra il 95%. Inoltre questo strumento riesce a generalizzare la funzione di mapping fra input e label anche ad ambienti e individui sconosciuti, senza necessitare di una nuova fase di training[10].

Il sistema SHARP presenta una fase iniziale di sanitizzazione del segnale, per poi passare a una calcolo dei *doppler traces* e infine una fase di allenamento per poi poter testare L'algoritmo. Di seguito esposte le varie fasi leggermente più nel dettaglio:

1. **Il segnale viene preprocessato**. In particolare l'obiettivo di questa fase è l'eliminazione degli artefatti hardware e software che vanno a inficiare sulla fase dei vari campioni CFR.

2. **Vengono calcolati i Doppler Traces**. Il movimento di una persona in una stanza, essendo esso un corpo molle e in movimento, causa una serie di piccoli effetti Doppler chiamati micro-Doppler trace. Questi effetti cambiano se la persona è in movimento, ma rimangono costanti se si tratta di un elemento statico. Dalla computazione di questi micro-Doppler trace viene la generalità di SHARP. Infatti l'idea alla base è che il cambiamento di ambiente non crea micro-Doppler Traces in modo variabile nel tempo. Da notare che la rilevazione di pose statiche non trae particolari benefici dalla computazione di questi effetti.

3. **Crezione dei Dataset**. Il tool è provvisto di un algoritmo preconfezionato che permette di creare dataset (sia per training che per testing)

già pronti per essere dati in input all'algoritmo di deep learning.

4. **Learning**. SHARP presenta un sofisticato sistema di apprendimento attraverso una Rete Neurale Convoluzionale (128,535 parametri allenabili) che effettua previsioni per ogni antenna. In un secondo momento queste previsioni vengono fuse per produrre la decisione finale. Ai fini di questa tesi, solo un antenna viene presa in considerazione per cui non si tiene conto della seconda fase.

2.5 Studi correlati

Nella seguente sezione verranno informalmente esposte gran parte delle ricerche nell'ambito Human Identification, sottolineando le principali strategie che sono state utilizzate assieme ad alcuni aspetti tecnici.

2.5.1 WiWho

WiWho è un framework di WiFi Sensing che consente l'identificazione di una persona all'interno di un gruppo ristretto, sfruttando la tecnologia Wi-Fi 802.11n [11]. Non vi è utilizzo di dispositivi personali.

La ricerca mostra che le informazioni sullo stato del canale (CSI), ottenute attraverso il WiFi di ultima generazione, possono essere utilizzate per riconoscere l'andatura di una persona che cammina. La camminata rappresenta un tratto distintivo per ogni individuo e WiWho sfrutta queste informazioni per l'identificazione delle persone.

Per quanto riguarda la raccolta di dati CSI, WiWho assume la presenza di 2 endpoint e il sistema dovrebbe lavorare solamente su l'AP. In una fase iniziale WiWho rileva e analizza ogni passo a partire dai dati CSI. In seguito viene preso l'insieme di passi e viene analizzata l'andatura nel suo complesso.

Le caratteristiche della camminata vengono estratte dai dati sotto forma di feature. Ciò viene fatto attraverso dei modelli fisici. Dopodiché vengono comparate con delle camminate "firma" pre-allenate da un classificatore Ma-

chine Learning. L'output di questo classificatore sarà l'etichetta corrispondente all'individuo. Nella fase di Training ogni individuo che darà origine a una classe cammina in un percorso predeterminato per un numero fisso di volte.

L'efficacia di WiWho, stando all'articolo, è stata valutata tramite esperimenti svolti in diverse luoghi, coinvolgendo complessivamente 20 volontari. Il sistema riesce a identificare una persona con un'accuratezza media del 92% in un gruppo di 2 persone e dell'80% in un gruppo di 6 persone. Occorre aggiungere che nella maggior parte dei casi, sono sufficienti 2-3 metri di camminata per riconoscere la camminata di una persona e identificarla.

Per effettuare queste valutazioni, è stata impiegata una Intel 5300 802.11n WiFi NIC con tre antenne, e ad ogni pacchetto corrispondono matrici CSI con 30 campioni CFR ciascuno (corrispondenti a 30 subcarrier).

2.5.2 FreeSense

Similmente a WiWho, FreeSense tenta di identificare un individuo tra un ristretto gruppo di utenti tramite l'analisi della camminata. L'idea è che le differenti caratteristiche fisiche unite al diverso modo di camminare abbiano dei modi specifici di influenzare i segnali WiFi[12].

Il sistema funziona attraverso tre fasi ben definite:

1. **Data collection and noise removal**. I dati CSI vengono raccolti e viene scelto un filtro appropriato per diminuire il rumore.

2. **Feature extraction**. Per estrarre feature dai dati raccolti viene impiegata la tecnica di Principal Component Analysis (PCA) su ogni antenna per ridurre la dimensionalità. Dopodiché la serie di dati viene segmentata e attraverso un modello fisico vengono compresse le informazioni sull'onda.

3. **Classification**. L'identificazione viene eseguita attraverso un classificatore kNN. Date due forme d'onda risultato del passo precedente, i

ricercatori forniscono una formula per calcolare la distanza fra esse. Il classificatore ricerca il label con occorrenza maggiore fra i k esempi più vicini (dove la vicinanza è calcolata proprio attraverso tale formula)

Lo strumento presenta un'accuratezza compresa fra il 75.5% e il 94.5% con numero di utenti che va da 2 a 9. L'accuratezza diminuisce al crescere del numero di utenti, dal momento che più persone ci sono più è probabile che le caratteristiche fisiche e di andatura fra di loro siano simili e dunque difficili da distinguere. I dati sono stati raccolti attraverso un ricevitore con Intel Link 5300 WIFI NIC e un trasmettitore TP-Link TLWR1043ND WIFI. I due dispositivi avevano rispettivamente 2 e 3 antenne. I dati raccolti disponevano di 30 valori CFR per pacchetto.

2.5.3 Rapid

Questo sistema, al fine di ottenere una robustezza maggiore degli altri strumenti all'avanguardia, unisce l'analisi dell'andatura, come nei due articoli precedenti, a un sistema di rilevamento acustico. Questo rende il sistema non più solamente di WiFi Sesning ma anche di Acoustic Sensing [13].

L'articolo effettua in primis un analisi di WiWho, trovandone alcune limitazioni non riportate nell'articolo originale. Il tool analizzato infatti richiede che il percorso di camminata sia parallelo alla linea di vista (LoS) tra il punto di accesso (AP) e il dispositivo client, a una distanza fissa, sia durante il processo di addestramento che durante il processo di identificazione. Questa condizione è ovviamente non sempre vera in casi reali. I ricercatori ricreano l'esperimento mostrato dall'articolo di WiWho[11] e riportano che l'accuratezza cala significativamente all'aumentare della distanza dalla LoS. La soluzione più ovvia sarebbe incrementare maggiormente i dati di training facendo camminare gli utenti per tutti i possibili percorsi, anche se distanti dalla LoS. Ciò comporterebbe tuttavia costi altissimi sia per il training sia per la fase di inferenza (per via delle feature aggiuntive).

I ricercatori procedono denominando l'incompletezza del training data **system noise** (rumore sistemico). Per migliorare la stima di questa system

noise introducono una nuova metrica chiamata CPV-Distance (CFR Power Variance-Distance). Questa metrica dovrebbe stimare la distanza che intercorre fra la persona che cammina e la LoS, in modo tale da fornire un indice del rumore sistemico e di conseguenza un grado di sicurezza riguardo all'inferenza. Nel caso in cui questo grado di sicurezza sia basso, viene proposta una soluzione multi modale, ovvero di coniugare al Wi-Fi Sensing ad altre tecniche di rilevazione. Nel paper viene scelta un tipo di rilevazione acustica basata sul rumore dei passi dell'utente. È necessario, anche in questo caso, trovare una metrica che rispecchia il grado di rumore sistemico presente nei dati raccolti in un dato periodo. In questo caso, similmente ai sistemi di Voice Activity Recognition) viene scelto SNR (Signal to Noise Ratio).

Una volta che tutti questi dati vengono raccolti, il classificatore prenderà in considerazione gli indici riguardanti la system noise e darà più peso al risultato proveniente dai dati più accurati. Per effettuare le classificazioni vengono usati gli SVM.

Sono stati svolti diversi esperimenti per verificare la precisione di questo strumento e l'accuratezza varia fra 92% con 2 utenti sino a 82% nel caso di 6 utenti. Anche in questo caso, per la raccolta dati è stata usata una Intel 5300 NIC.

2.5.4 WFID

A differenza delle ricerche riportate fino a questo punto, lo strumento WFID presenta un approccio leggermente diverso per effettuare Human Indentification. Invece che focalizzarsi sugli effetti che le persone potrebbero avere sulla fase dei segnali, viene data importanza all'ampiezza dell'onda [14].

Viene introdotta una nuova feature, SAF (Subcarrier Amplitude Frequency) che dovrebbe caratterizzare in maniera unica il modo in cui varia l'ampiezza del segnale in base alla camminata di un utente. Il processo può essere riassunto in:

1. **Segmentazione dei dati CSI**. I vari pacchetti vengono processati

in gruppo per catturare variazioni nel segnale. La scelta dei segmenti avviene in modo deterministico attraverso un modello di tipo fisico.

2. **Estrazione dei SAF**. Vengono computate le matrici SAF corrispondenti ai vari segmenti.

3. **Applicazione di PCA**. Ai vari risultati dello step precedente viene applicata la tecnica Principal Component Analysis per ridurne la dimensionalità.

4. **Training con SVM**. I risultati finali sono dati in pasto a un classificatore SVM per il training (e quindi successivamente per fare inferenze).

Il sistema presenta accuratezza del 91.9% in un gruppo di 9 persone e del 93.1% in un gruppo di 6 persone. Per la raccolta dati è stato impiegato un router NW705P con una sola antenna facente da AP. Il ricevitore è un computer equipaggiato con NIC Intel 5300 e 3 antenne. Il vettore di CSI consiste di 90 subcarrier.

Capitolo 3

Analisi progettuale e Implementazione

Questo capitolo spiega quali siano gli obiettivi principali di questa tesi e di come si sia deciso di procedere nel raggiungerli. Si procede poi a dare una visione ad alto livello sulla realizzazione del progetto e di quali siano i componenti da realizzare. Dopodiché si entra più nel dettaglio guardando ESP CSI Tool e SHARP al fine di rendere chiaro quali sono le soluzioni adottate per interfacciarsi a questi strumenti. Infine si illustra l'implementazione, mostrando i punti critici anche con l'aiuto di schemi e frammenti di codice.

3.1 Obiettivo e Motivazioni

Vi è un crescente interesse nell'attrezzare stanze, uffici e in generale luoghi al chiuso con capacità di rilevamento. Spazi come case e uffici "smart" possono implementare funzionalità di rilevamento ed elaborazione tramite WiFi Sensing, alleggerendo gli utenti dalla necessità di indossare/portare dispositivi personali. Per consentire tali spazi intelligenti è necessario un rilevamento senza dispositivi, privo di sforzo, dell'identità e delle attività dell'utente.

Oltre che spazi intelligenti, l'impiego del WiFi Sesning per il rilevamento senza dispositivi ha dimostrato un grande potenziale in scenari di sicurezza

e prevenzione di intrusioni e monitoraggio di situazioni sanitarie. Tuttavia, una questione fondamentale relativa all'identificazione delle persone è che in letteratura sono presenti ancora poche ricerche.

Più nello specifico, i sistemi di Human Identification fanno sempre affidamento su utenti in movimento cercando di analizzare come la loro camminata influenza il segnale Wi-Fi. La possibilità di identificare una persona statica in un determinato ambiente è ancora poco esplorata. Inoltre, sebbene i dispositivi usati per la raccolta dati siano effettivamente COTS, non ci sono ricerche in Human Identification che sfruttano dispositivi così a basso costo come ESP32.

Questo elaborato si pone un duplice obiettivo, uno in chiave progettuale e uno di tipo sperimentale.

In primo luogo si vuole fornire uno strumento che adatti le misurazioni CSI fornite da *ESP32 CSI Tool* per renderle utilizzabili da SHARP. Questo strumento deve inoltre facilitare l'utilizzo di SHARP in modo tale da automatizzare la modifica di parametri. In altre parole si vuole:

1. creare un interfaccia fra la raccolta dati con *ESP32 CSI Tool* e l'algoritmo di WiFi Sensing con SHARP

2. unificare le varie fasi dell'algoritmo di WiFi Sensing per rendere il processo automatico.

In secondo luogo si vuole verificare la possibilità di utilizzare uno strumento innovativo come SHARP per effettuare Human Identification. SHARP infatti è disegnato per problemi di HAR. Tuttavia la sua natura basata su CNN potrebbe permettergli di generalizzare la task di riconoscere l'attività con il riconoscere un individuo. Inoltre i dati forniti a SHARP sono ottenuti tramite schede ESP32, per cui la qualità dei dati appare essere inferiore rispetto a quelli usati dalle tradizionali ricerche su WiFi Sening, almeno in ambito di Human Identification. Chiaramente, lo strumento che si vuole progettare è propedeutico per fare questo tipo di esperimento.

Le due domande alla quale si vuole rispondere sono:

Q1. *Il problema della Human Identification tramite Wi-Fi Sensing può essere affrontato anche con dati raccolti tramite device estremamente semplici e di basso costo come ESP32?*

Q2. *Può uno sistema orientato a risolvere problemi HAR essere adattato per fare Human Identification?*

Il motivo della scelta di questi due strumenti è semplice. Gli ESP32 sono chip a basso costo e basso consumo, installabili pressoché ovunque e addirittura portabili e configurabili tramite smartphone [8]. Riuscire ad ottenere soluzioni di Wi-Fi Sensing per la Human Identification attraverso tale tecnologia è naturalmente un obiettivo interessante e utile. La scelta di SHARP invece, è dovuta al fatto che i sistemi di Human Identification (molti dei quali citati nello Stato dell'Arte) non sono rilasciati sotto licenza Open Source. Inoltre risulta essere interessante verificare se il tipo di preprocessing scelto per SHARP (quindi finalizzato a fare HAR) sia anche in grado di dare risultati in ambito di Human Identification.

3.2 Overview ad alto livello

SHARP è uno sistema creato per funzionare insieme a *Nexmon CSI tool* un tool per raccogliere dati CSI basato su chip Broadcom. Più precisamente i ricercatori hanno sfruttato un router Asus RT-AC86U IEEE 802.11ac Wi-Fi utilizzando una larghezza di banda totale di 80 MHz [15].

I dati CSI che raccolgono, oltre ad essere di qualità superiore (essendo stati raccolti con una tecnologia Wi-Fi a 5GHz), sono anche di dimensione incompatibile con dati CSI provenienti da un ESP32. Il numero di subcarrier è infatti di 64 contro i 256 previsti da SHARP. Questo però accade solo nella fase di preprocessing, infatti gli step successivi di SHARP prevedono la scelta di una larghezza di banda e una sezione di banda da considerare.

SHARP prevede inoltre la possibilità che l'AP abbia molteplici antenne. I dispositivi ESP32 usati avevano solamente un'antenna. Dunque tutte le

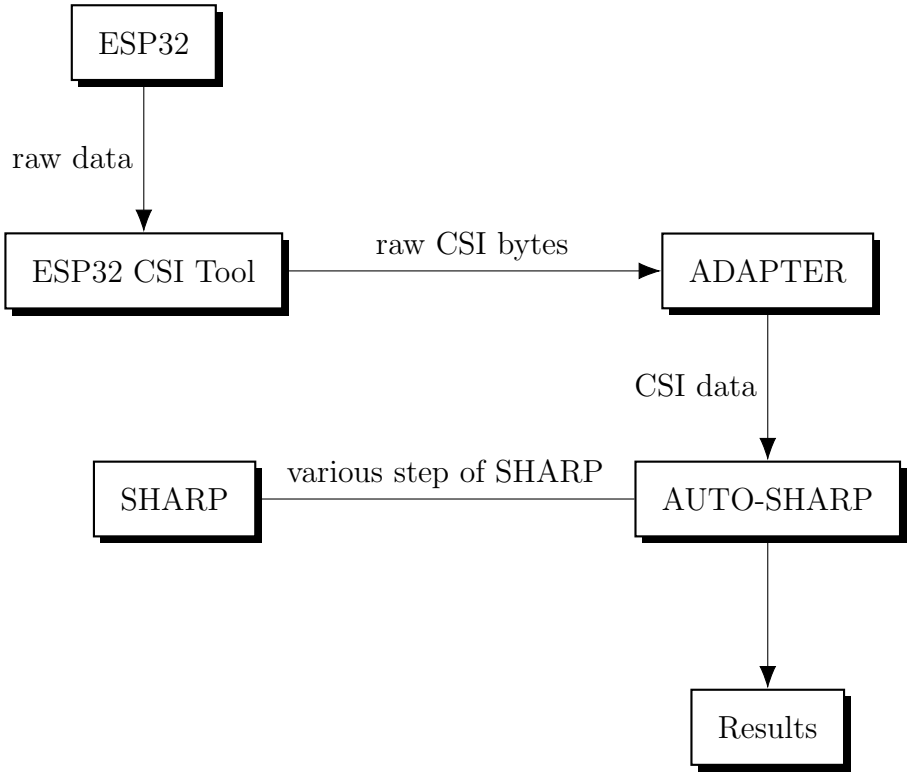

Figura 3.1: Pipeline del progetto completa

sezioni di SHARP che richiedono l'uso di più antenne (fra le quali, scegliere l'antenna più adeguata per la classificazione) non si applicano, e vanno eliminate ai fini di questa tesi.

Il progetto quindi prevede la creazione di 2 componenti fondamentali, uno che faccia da ponte tra i dati CSI raccolti da ESP32 CSI Tool e SHARP, e uno che renda automatico l'utilizzo di SHARP. Per il resto della tesi chiameremo questi due componenti **ADAPETER** e **AUTOSHARP**.

Nel grafico 3.1 è riportata una schematizzazione di come funzioneranno nel loro insieme tutti i tasselli del progetto. Di seguito una breve descrizione:

1. I chip ESP32 stabiliranno una connessione e si invieranno pacchetti.

Da questi pacchetti vengono ricavati i dati CSI.

2. ESP32 CSI Tool si occupa di fornire alle schede ESP32 il programma per effettuare la connessione, scambiarsi pacchetti ed estrarre i dati CSI dalla scheda per inviarli alla porta seriale.

3. ADAPTER si occupa di trasformare i byte "raw" forniti dallo step precedente in vere e proprie misurazioni CFR sotto forma di numeri complessi. I dati subiranno poi alcune modifiche a livello di formato ai fini dell'apprendimento automatico per rimuovere eventuali bias.

4. AUTOSHARP si occupa di gestire ADAPTER. Una volta che i dati vengono trasformati nel formato corretto questo programma si occupa di eseguire ogni fase di SHARP (chiaramente a scelta) avendo fissati i parametri delle varie operazioni. Questo tool introduce anche alcune soluzioni mirate all'efficienza fra cui la parallelizzazione dei processi.

5. Dialogo con SHARP. AUTOSHARP dialoga con SHARP seguendo il suo naturale corso, gestendo man mano la creazione di directory e file. È il tool SHARP che ha l'onere di effettuare le varie fasi di computazione.

6. Il prodotto finale della pipeline è un classificatore (una CNN) e una serie di valutazioni sulla sua performance.

È anche necessario modificare leggermente il codice di SHARP per facilitare le interazioni con l'ADAPTER ed eliminare l'utilizzo di più antenne.

Dopo la realizzazione di tale progetto, si passa a una fase di raccolta dati. Raccolti i dati CSI con opportuna label si fa uso della pipeline per testarne il suo funzionamento e rispondere al quesito *Q2*.

L'idea finale è che un possibile utente che sia interessato ad effettuare Human Identification possa utilizzare questo tool per produrre un classificatore. L'utente dovrebbe raccogliere dati e fornirli allo strumento. Una volta disponibile il classificatore, basta montarlo su un computer collegato a

un ESP32 per produrre rilevamenti in tempo reale. Esistono anche tecniche di compressione di reti neurali che cercano di abbassarne il costo in termini di spazio e di tempo mantenendo lo stesso "fitness" [16]. Attraverso queste tecniche sarebbe possibile addirittura montare la rete neurale finale sopra gli stessi ESP32, rendendoli capaci di rilevare eventi e attivare di conseguenza altri dispositivi collegati (ad esempio una luce, un sistema acustico, etc...).

Da notare che, sebbene l'obiettivo di questa tesi sia fare Human Identification, con qualche modifica di paramenti è facile adattare il tutto per effettuare HAR.

3.3 ESP32 CSI Tool

In questa sezione viene analizzato lo strumento di raccolta dati, con particolare enfasi riguardo al formato dei dati finale così da rendere ovvi quali siano i passi successivi da prendere. Il codice sorgente dello strumento di raccolta dati è facilmente accessibile tramite github [17].

3.3.1 Struttura del tool

Ai fini di un utente, il tool è strutturato secondo directory che contengono i principali componenti. Ai fini di questa tesi, alcune directory come **_components** sono state ignorate, giacché contengono funzioni e costanti di utility non interessanti per la comprensione:

1. **active_ap** è la directory che contiene il codice per l'Access Point. Consente la connessione da parte di una Station (ESP32) o potenzialmente anche di un telefono. Accetta di ricevere pacchetti e di conseguenza pacchetti di risposte CSI.

2. **active_sta** è la directory che contiene il codice per la Station. La Station si connette all'AP (che potrebbe essere un router o un altro ESP32) e invia richieste di pacchetti ricevendo dunque risposte con pacchetti CSI.

```
#if CONFIG_SHOULD_COLLECT_ONLY_LLTF
    int data_len = 128;
#else
    int data_len = data->len;
#endif

int8_t *my_ptr;
#if CSI_RAW
    my_ptr = data->buf;
    for (int i = 0; i < data_len; i++) {
        ss << (int) my_ptr[i] << " ";
    }
#endif
```

Listing 1: Example of a listing.

3. **python_utils** è una directory che raccoglie comodi script in Python. Alcuni servono a visualizzare in tempo reale l'ampiezza di alcuni sub-carrier, altri per aggiungere un timestamp globale ai pacchetti, etc...

3.3.2 Funzionamento e dipendenze

Il tool fa affidamento al framework ESP-IDF[9] per interfacciarsi al microcontroller. In poche parole il framework contiene delle API per ESP32 e diversi script per automatizzare la Toolchain di compilazione e build di applicazioni.

Tramite i software forniti da ESP-IDF è possibile compilare con facilità i programmi presenti nelle directory e caricarli sopra i chip ESP32. Una volta che le schede possiedono i compilati al loro interno si possono lanciare i processi e far partire le comunicazioni. ESP32 CSI Tool colleziona infine i dati CSI dei pacchetti e li stampa sul terminale.

Il codice in C++ qui riportato 1 mostra come i dati CSI vengono rac-

colti. Se l'impostazione **CONFIG_SHOULD_COLLECT_ONLY_LLTF** è attiva allora solo i LLTF verranno collezionati per un totale di 128 valori. Questi valori corrispondo a i componenti reali e immaginari di ogni subcarrier. Dopodiché vengono inseriti nello stream che verrà mandato alla porta seriale.

Da notare il cast a **int8_t** che quantizza i componenti reali e immaginari di ogni valore CFR. Come già menzionato se si volessero passare dati sotto forma di floating point attraverso la porta seriale ci sarebbe una saturazione della porta seriale. Questo rende molto chiaro il fatto che dati CSI forniti da questo tool siano di qualità inferiore rispetto agli altri strumenti moderni (proprio per la natura poco costosa degli ESP32).

3.4 SHARP

Nella seguente sezione viene analizzato lo strumento SHARP che si occupa di fare preprocessing e apprendimento per il classificatore (nonché testing per la valutazione delle sue performance). Nuovamente si vuole porre particolare enfasi sul formato dei dati iniziale e sul funzionamento generale in modo tale da chiarire quali siano i passi implementativi per costruire **ADAPTER** e **AUTOSHARP**. Il codice sorgente del tool è raggiungibile attraverso l'articolo relativo a SHARP [10]

3.4.1 Struttura del Tool

Il sistema SHARP contiene una serie di script in Python che si occupano di dare forma ad ogni step. Questi script lavorano in modo atomico interagendo con il filesystem, generando quindi directory e file come output.

Riportiamo di seguito i passi del programma principali che dovranno essere sfruttati da **AUTOSHARP** per automatizzare il tutto.

I seguenti programmi costituiscono la sanitizzazione della fase. Hanno come obiettivo sbarazzarsi degli artefatti hardware e software che causano offset nella fase del segnale (non rispecchiando l'offset causato dall'ambiente).

Due directory sono coinvolte nella processazione dei dati, *phase_processing* e *processed_phase*.

1. **CSI_phase_sanitization_signal_preprocessing.py**
2. **CSI_phase_sanitization_H_estimation.py**
3. **CSI_phase_sanitization_signal_reconstruction.py**

Per quanto riguarda la computazione delle **doppler traces** si fa riferimento a **CSI_doppler_computation.py** che produrrà file nella cartella *doppler_traces*. La creazione dei dataset e l'allenamento e valutazione del modello sfruttano nuovamente la struttura delle directory già mostrata. La distribuzione dei LABEL viene inferita dalla nomenclatura dei file I risultati vengono trascritti in una serie di file.

In definitiva, le varie fasi prelevano gli input da file presenti in una directory iniziale e per tutte le fasi successive si basano sui nomi delle subdirectory. Questo tipo di organizzazione genera molte difficoltà nella comprensione, specialmente se si presenta il bisogno di rieseguire determinate fasi o se si vuole cambiare un parametro o la struttura dei file iniziali. Inoltre per quanto riguarda l'allenamento e il testing è prevista la possibilità di avere più antenne che ricevono il segnale. Durante la fase di allenamento e inferenza si unisce il risultato di ogni antenna per usare i risultati migliori. Ai fini della tesi, essendo gli ESP32 dotati di una sola antenna, non è interessante esplorare questa possibilità.

3.5 Implementazione di ADAPTER

In questa sezione viene trattata l'implementazione del primo componente: ADAPTER. Tale implementazione consiste in due script: **format.py** e **padding.py**. Si è deciso di separare l'aggiunta di padding in modo che si possa facilmente effettuare la formattazione anche per matrici CSI di dimensioni maggiori (più subcarrier).

```
for i in range(len(csi_arr)):
    if i % 2 == 0:
        imaginary_tmp =  csi_arr[i] * 1.0j
    else:
        complex_arr.append(csi_arr[i] + imaginary_tmp)
```

Listing 2: From bytes to complex number

Il primo script ha due obiettivi fondamentali: **formattazione dei dati** e **divisione del dataset**.

Per quanto riguarda la formattazione dei dati, le misure da prendere sono

1. I dati che vengono passati in input sono stringhe di valori raccolti direttamente dalla porta seriale sotto formato **csv**. Bisogna in primo luogo isolare la matrice di byte CSI raw interessata.

2. Dopodiché è necessario effettuare un "cast" che renda le varie stringe array di byte. Potrebbe succedere che alcuni artefatti siano stati introdotti all'interno dei valori (come caratteri non leggibili, o più valori mischiati fra loro). è quindi necessario eliminare i valori non corretti per mantenere la sanità del dataset.

3. A questo punto rimane solo da fare un cast da valori interi a numeri complessi. Come già riportato durante l'introduzione dei chip ESP32, i byte riportano alternativamente la parte reale e immaginaria di un certo valore CFR. Il codice che esegue il cast viene qui riportato 2

Per la divisione del dataset invece si ha la necessità di dividere in training e test prima di qualsiasi step di elaborazione del segnale. Questo perché si vuole evitare qualsiasi tipo di **data leakage**, che porterebbe alla sovrastima delle performance del modello [18]. Un altro aspetto importante da considerare è quelli di pareggiare la grandezza dei dataset. Infatti durante la fase di raccolta dati potrebbe accadere che i dati per un certo label siano troppi rispetto agli altri. Per evitare di introdurre un bias dovuto alla frequenza

maggiore, si è deciso di pareggiare tutti i label in favore del minimo. La grandezza del dati per il test è pari al 5% dei dati totali.

Il prodotto di questo script è un file dove sono stati serializzati sotto forma di array i valori CSI. Questa scelta è stata effettuata poiché velocizza il caricamento da parte dei successivi script e perché anche SHARP usa la stessa tecnica.

A questo punto i dati sarebbero già pronti per essere dati in input a SHARP se nonché il preprocessing di quest ultimo è stato progettato per pacchetti a banda 80MhZ, quindi un totale di 256 subcarrier. Per terminare l'adattamento **padding.py** aggiunge 192 valori nulli di padding.

3.6 Implementazione di AUTOSHARP

In questa sezione viene trattata l'implementazione del secondo componente: AUTOSHARP. L'implementazione è effettuata attraverso uno script chiamato **sharping.py**. Questo script serve ad automatizzare le fasi di SHARP e pulire e gestire il filesystem, al fine facilitare diverse run. Vi sono infatti più parametri che possono essere modificati e risulta utile poterli cambiare in modo sistematico ed unico.

La maggiore difficoltà nell'utilizzo di SHARP è che molti step di computazione vengono saltati se esistono già file con lo stesso nome target. Inoltre lo strumento non è coerente con la strutturazione delle directory fra passaggio e passaggio. AUTOSHARP si prende l'onere, tramite l'aggiunta di opzioni da terminale, di garantire l'esecuzione di tutte le fasi.

In figura 3.2 una schematizzazione del flusso del programma. Il titolo delle frecce corrisponde alle flag passabili al programma che indicano di rieseguire una determinata parte.

Un ulteriore appunto va fatto riguardo la parallelizzazione dei passi di preprocessing. Come già menzionato, SHARP esegue computazioni su blocchi di file. Riceve dunque in input una serie di directory, e per ogni directory avvia l'elaborazione del segnale. Questa elaborazione del segnale è "single

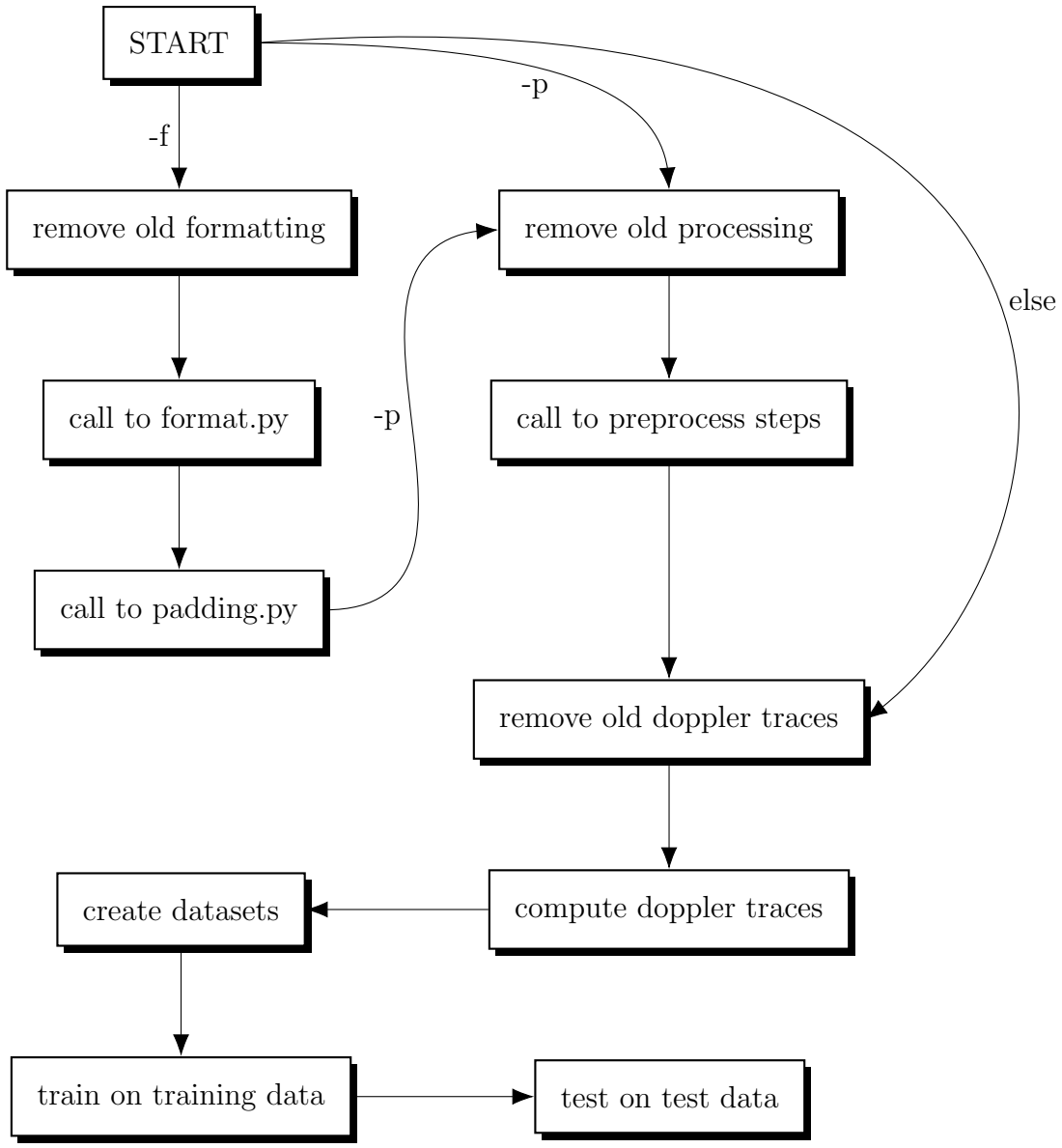

Figura 3.2: Funzionamento di AUTOSHARP

```
def usePool(commands):
    pool = ThreadPool(numberOfCore)
    pool.map(subprocess.run, commands)
    pool.close()
    pool.join()
```

Listing 3: Parallelizzazione dei lanci di SHARP

threaded". In altre parole, se si vogliono processare n file, un solo core della CPU verrà usato e l'$i-esimo$ file verrà processato solo quando il file $i-1$ ha finito. In computer con processori multi-core risulta evidente il vantaggio di gestire più esecuzioni in parallelo di SHARP, specialmente visto il carico computazionale elevato del passo di sanitizzazione delle fasi. Sia C_n il numero di core, il costo computazionale parallelizzando le esecuzioni di SHARP diventerebbe $\frac{n}{C_n}$, che risulta essere significativo ai fini di questa tesi.

Per implementare la parallelizzazione viene usata la libreria Python **multiprocessing** [19].

Nel codice qui riportato 3 si può vedere l'implementazione attraverso la funzione **usePool**. Come input prende una lista di comandi. Viene inizializzata una **pool** (un gestore di thread) in base al numero di core disponibili e vengono mappati alla pool i vari comandi da eseguire. Questa generalizzazione permette di eseguire lo stesso step di computazione su molteplici file approfittandosi di tutti i core della macchina.

Prima di proseguire nell'esecuzione si aspetta che tutti i processi abbiano terminato l'elaborazione. Questo poiché la fasi successive di elaborazione hanno come vincolo quello di avere tutti i file allo stesso punto dell'elaborazione.

Sono state effettuate anche una serie di modifiche al codice sorgente di SHARP stesso. Specialmente riguardo la fase di training e di test. Infatti SHARP tenta di verificare le sue previsioni utilizzando tutte le antenne disponibili, aspetto che non riguarda gli ESP32 utilizzati. Si ha deciso quindi di eliminare il codice relativo alle diverse antenne in favore di più semplicità.

Capitolo 4

Valutazioni Risultati

In questo capitolo viene descritta inizialmente la struttura dell'esperimento, trattando accuratamente la fase di raccolta dati per garantire ripetibilità. Dopodiché si procede a mostrare i risultati della classificazione, valutando i risultati e fornendo considerazioni sulla loro natura.

4.1 Descrizione Esperimento

Sono state effettuate due raccolte dati. Una iniziale per sperimentare il corretto funzionamento del tool e misurare le sue performance in modo prematuro. Successivamente è stata ripetuta la raccolta dati con alcune modifiche e con l'obiettivo di ottenere un dataset più largo.

La raccolta dati è stata effettuata in una stanza di 24 m^2, contenente 4 scrivanie disposte come in figura 4.1. Le schede ESP32 sono state disposte a 2 m di distanza, rappresentate in figura da due router dotati di antenna (l'immagine riporta due antenne, mentre gli ESP32 ne possiedono una sola).

4.2 Prima raccolta dati

Per la prima raccolta dati sono stati scelti 3 volontari con caratteristiche fisiche diverse. L'esperimento ha seguito il seguente procedimento, ripetuto

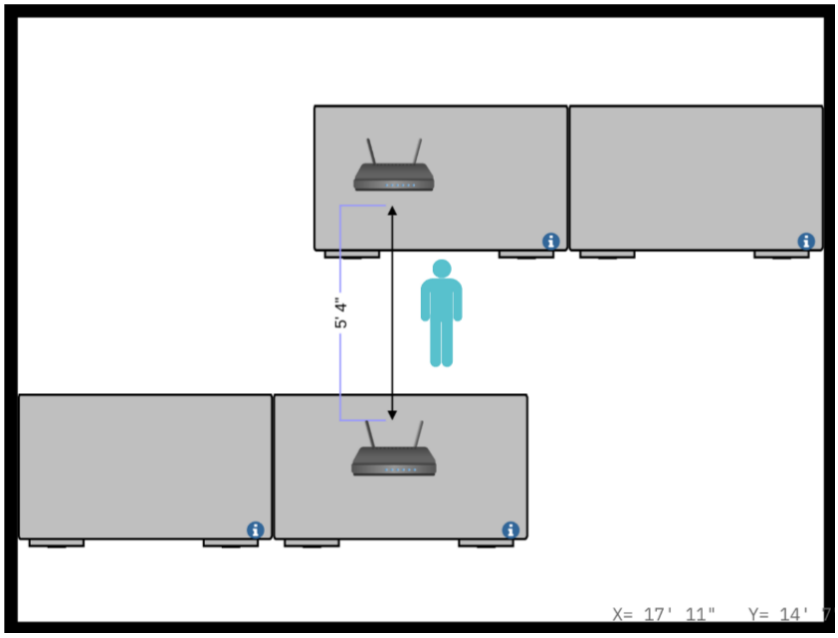

Figura 4.1: Laboratorio dove sono stati condotti gli esperimenti

per ogni individuo:

1. il volontario si posiziona in mezzo alle due antenne, perpendicolare alla LoS.

2. rimane fermo per all'incirca 5 minuti

3. la raccolta dati si conclude e si passa al prossimo volontario

Oltre ai tre volontari, si è deciso di raccogliere dati anche in assenza di individui nella stanza, sempre per 5 minuti.

Nella tabella 4.1 sono riportati i dati dei 3 volontari, v1, v2 e v3. Con **em** si fa riferimento all'assenza di individui. Ai fini dell'esperimento sono stati misurati massa e altezza, rispettivamente con incertezza di $\pm 0.5\ kg$ e $\pm 1\ cm$. Sono stati scelti questi dati per scegliere individui di caratteristiche differenti poiché in letteratura sono quelli più significativi[20].

Sono riportati il numero di campioni raccolti, il numero di campioni adibiti alla fase di training e quelli per il testing. La quantità di dati raccolti

	v1	**v2**	**v3**	**em**
altezza	168 cm	179 cm	191 cm	/
massa	68 kg	71 kg	77 kg	/
n. campioni	24.065	21.254	20.552	41.664
n. train	20.190	20.190	20.190	20.190
n. test	1.063	1.063	1.063	1.063

Tabella 4.1: Statistiche sulla prima raccolta dati

mentre la stanza è vuota risulta essere circa il doppio rispetto agli altri scenari. Questo è dovuto al fatto che, non avendo nessun ostacolo fra le antenne i chip sono stati in grado di scambiarsi molti più pacchetti con successo. Prima di essere dati in input a SHARP i dati sono stati troncati a favore dello scenario meno popoloso, in modo da non introdurre bias di frequenza durante la fase di training.

4.2.1 Risultati

In questa sezione verranno trattati i risultati prodotti a partire dai primi dati raccolti. Di seguito un breve sunto delle metriche mostrate e alcuni commenti riguardo il loro significato.

1. **Confusion Matrix**. Una tabella dove ad ogni riga ed ogni colonna corrisponde una classe. Le colonne rappresentano le prediction effettuate dal classificatore mentre le righe i label corretti. Nella diagonale quindi si trovano il numero di prediction corrette.

2. **Precision**. La precision si ottiene dalla seguente formula riguardante una singola classe C: $\frac{TP}{TP+FP}$ dove TP sta per true positive e FP sta per false positive. I primi sono i casi predetti appartenti a C che effettivamente erano della classe C mentre i secondi sono i casi predetti C che invece appartenevano ad un'altra classe. Una precision di 0.8

significa che l'80% delle volte che il classificatore dice che l'input $a \in C$ ha ragione. In altre parole simboleggia l'accuratezza del dire sì.

3. **Recall**. Fissata una classe C questa metrica è espressa dalla formula $\frac{TP}{TP+FN}$ dove TP sta per true positive e FN sta per false negative. I secondi si riferiscono a tutte le volte che si è predetta una classe $X \neq C$ mentre l'input apparteneva alla classe C. Un recall dello 0.8 significa che il 20% delle volte dice che $x \notin C$ mentre in realtà $x \in C$. In altre parole simboleggia l'accuratezza del dire no.

4. **Accuracy**. Percentuale di risposte corrette in totale.

Successivamente verranno effettuate considerazioni riguardo i suddetti risultati al fine di effettuare una seconda raccolta dati atta a migliorare le performance.

Valutazioni e Considerazioni

	v1	**v2**	**v3**	**em**
altezza	168 cm	179 cm	191 cm	/
massa	70 kg	71 kg	77 kg	/
Precision	0.54	0.36	0.37	0.92
Recall	0.43	0.49	0.28	1

Tabella 4.2: Risultati prima raccolta

Basandoci sui dati raccolti, il sistema presenta una **accuracy** generale del 55%. Come riferimento prendiamo in considerazione il classificatore R che classifica in maniera casuale. R presenta in media una accuracy del 25%. Il nostro classificatore quindi ha performance migliori dell'indovinare a caso. In figura 4.2 è possibile visualizzare la confiusion matrix mentre nella tabella 4.2 si possono vedere i dati numerici corrispondenti ai risultati.

Questi risultati vanno visti nel loro complesso. Il classificatore risulta perfettamente in grado di rilevare la presenza di una persona o meno. La

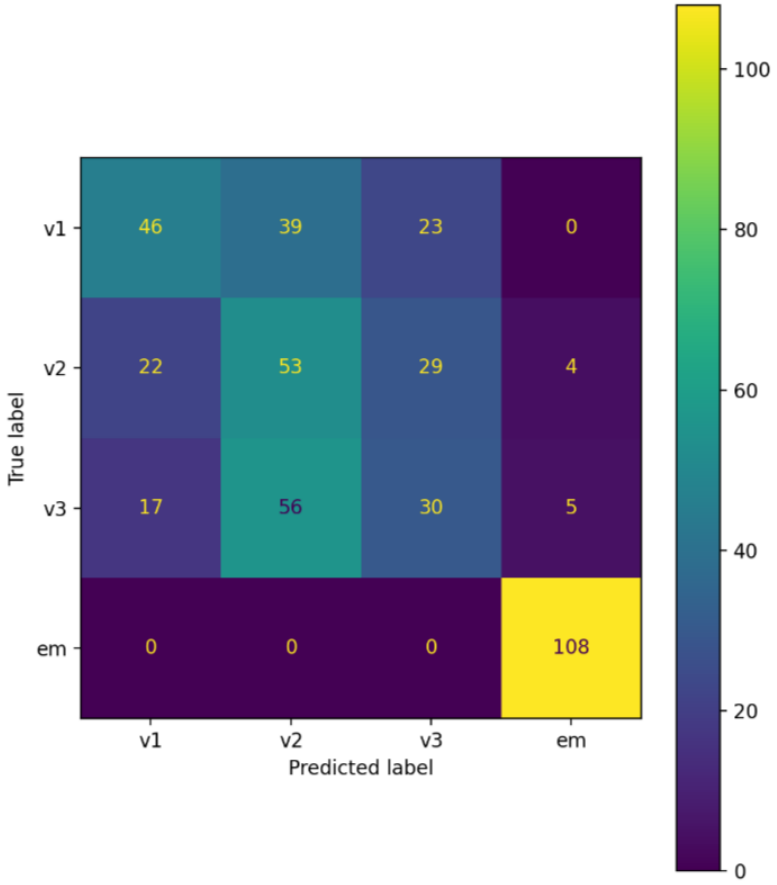

Figura 4.2: Confusion matrix del test data, prima raccolta

classe **em** infatti presenta una **precisione** del 92% e **recall** perfetto. Questo vuol dire che non ci sono stati falsi negativi riguardo all'assenza di persone. Non è mai successo che il sistema affermasse assenza di individuo mentre ci fosse effettivamente qualcuno. La grande accuratezza nel rilevare la presenza di un individuo fa crescere la bontà del sistema anche nel compito di Human Identification.

Il sistema presenta grandi difficoltà nel distinguere i tre utenti $v1, v2, v3$ con risultati particolarmente insoddisfacenti per quanto riguarda $v3$. Alla luce di questi risultati, sono necessarie alcune considerazioni, sopratutto in vista di una seconda raccolta dati:

- Il sistema non è a disposizione di abbastanza dati. La CNN ha oltre 100.000 parametri allenabili mentre il numero di campioni in input non supera gli 80.000. Questa evidente carenza nel dataset non permette alla rete di avere un bias sufficientemente basso atto a produrre previsioni corrette. Persino in fase di allenamento l'accuratezza generale non supera l'80%.

- Tutta le ricerche che provano ad effettuare Human Identification si basano sul riconoscimento di soggetti in movimento. Le interferenze prodotte da una camminata sono molto più significative rispetto a quelle di un soggetto statico. Questo è un primo fattore che spiega la grande difficoltà nel riconoscere individui statici.

- Il tool SHARP effettua prima di passare alla fase di training computazioni per estrarre le "doppler traces". Questi micro effetti doppler sono generati dalle onde che interagiscono con corpi in movimento. Questa informazione sul tool rafforza la spiegazione sulla scarsa capacità di classificazione nella prima raccolta dati

Queste considerazioni ci spingono a modificare leggermente l'impostazione di raccolta dati. In primo luogo si vuole ingrandire il dataset per poter

dare a SHARP maggiori possibilità di apprendere pattern. In secondo luogo, per sfruttare al massimo le possibilità del sistema risulta opportuno far muovere i volontari.

4.3 Seconda raccolta dati

La seconda raccolta dati è stata condotta nello stesso laboratorio della prima 4.1. Sono stati scelti di nuovo tre volontari, due in comune con la raccolta precedente. L'esperimento ha seguito il seguente procedimento per ogni individuo:

1. il volontario si posiziona a due metri di distanza dai dispositivi
2. il volontario inizia a camminare in modo perpendicolare alla LoS per quattro metri
3. il volontario continua a camminare per 20 minuti
4. si interrompe la raccolta dati e si passa al prossimo volontario

In questo scenario si è deciso di non raccogliere dati riguardanti la stanza vuota. Il tool è già perfettamente in grado di rilevare la presenza di un essere umano anche statico per cui risulta più interessante dedicarsi totalmente al compito di Human Identification

	v1	**v2**	**v3**
altezza	163 cm	168 cm	179 cm
massa	58 kg	70 kg	71 kg
n. train	90.250	90.250	90.250
n. test	4.750	4.750	4.750

Tabella 4.3: Statistiche sulla seconda raccolta dati

Nella tabella 4.3 sono riportati i dati dei tre volontari con annessi il numero di campioni raccolti. In questo caso si riporta solamente il minimo

$\min(v1, v2, v3)$ che è stato usato come troncamento in tutti e tre i casi. Nuovamente sono stati misurati massa e altezza, rispettivamente con incertezza di $\pm 0.5\ kg$ e $\pm 1\ cm$.

Visto che nella letteratura scientifica le migliori performance si ottengono nella classificazione con due utenti soltanto, si è deciso di affrontare prima un problema ridotto con due soli utenti per poi introdurre il terzo.

4.3.1 Classificazione con due utenti

	v1	**v2**
altezza	163 cm	168 cm
massa	58 kg	70 kg
Precision	0.95	0.96
Recall	0.96	0.95

Tabella 4.4: Risultati seconda raccolta, 2 user

Il sistema presenta in questo scenario una **accuracy** del 95%. Questo risultato supera di gran lunga quello della prima raccolta dati, portando il tool sullo stesso piano di quelli all'avanguardia, mostrati nel capitolo **Stato dell'arte**. In figura 4.3 è possibile visualizzare la confusion matrix mentre nella tabella 4.4 sono riportate le metriche.

4.3.2 Classificazione con tre utenti

Introducendo un terzo utente, il sistema presenta una **accuracy** del 74%. Questo risultato è sempre superiore rispetto alla prima raccolta dati, ricordando che in questo caso il compito è solo e soltanto Human Identification. In figura 4.4 è possibile visualizzare la confusion matrix mentre nella tabella 4.5 sono riportate le metriche.

Facendo riferimento alla confusion matrix, il tool ha una grande accuratezza nel riconoscere il primo utente, con Precision e Recall rispettivamente

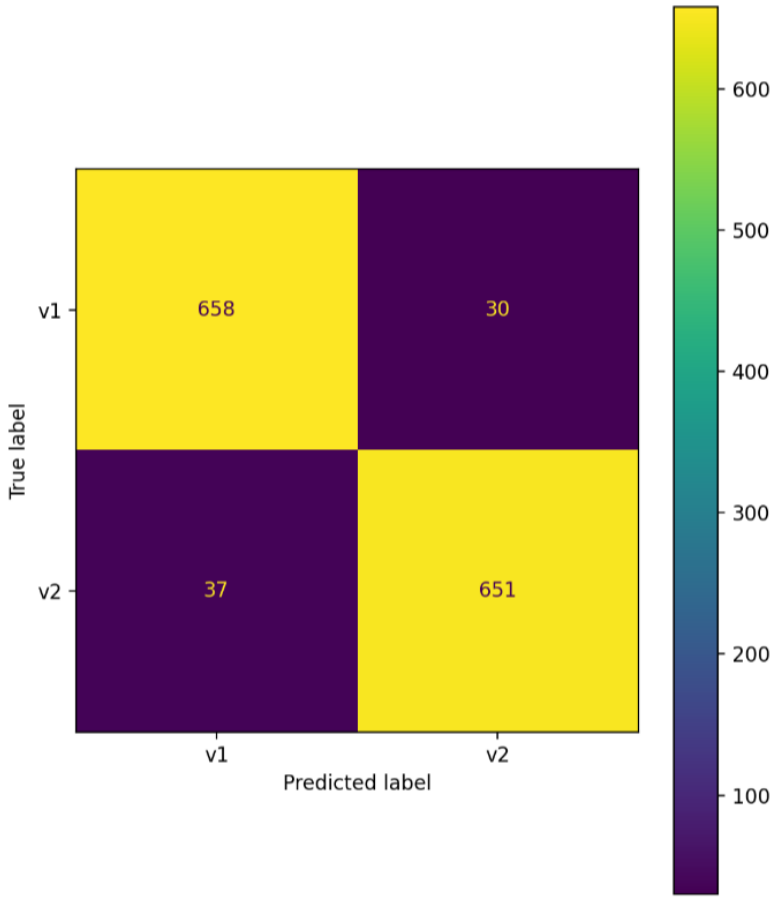

Figura 4.3: Confusion matrix del test data, seconda raccolta, 2 user

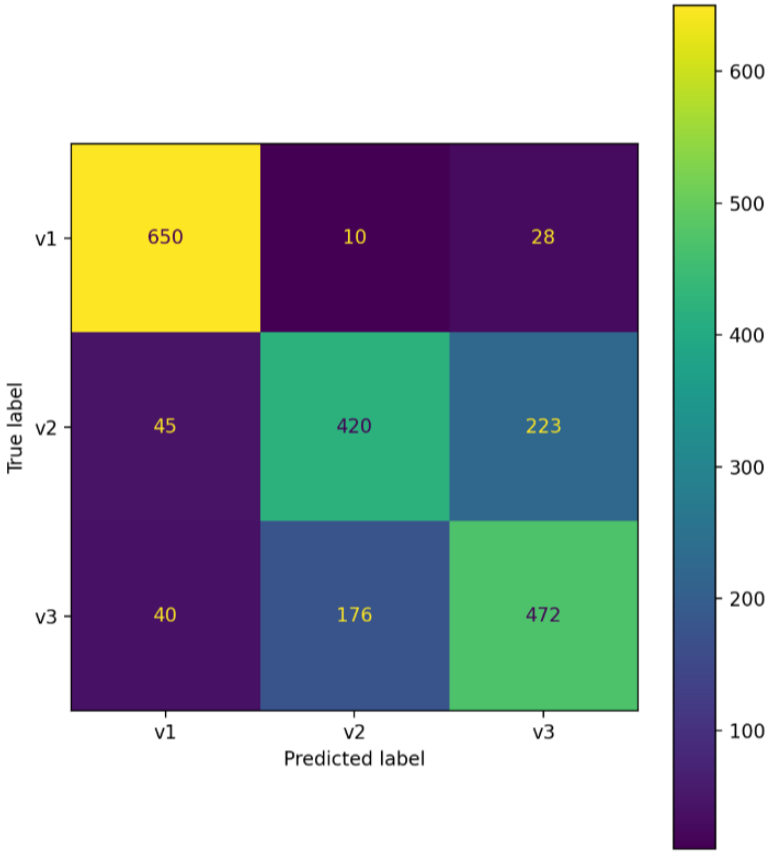

Figura 4.4: Confusion matrix of test data, seconda raccolta, 3 user

	v1	**v2**	**v3**
altezza	163 cm	168 cm	179 cm
massa	58 kg	70 kg	71 kg
Precision	0.88	0.69	0.65
Recall	0.94	0.61	0.67

Tabella 4.5: Risultati seconda raccolta, 3 user

di 0.84 e 0.94. Il calo di performance rispetto alla classificazione con due utenti è nel distinguere fra utente v2 e utente v3. Nonostante il sistema sia di gran lunga migliore di un classificatore casuale R anche solo considerando queste due classi, comunque sbaglia all'incirca una volta su tre (questa stima è in negativo, nel caso di v3 la situazione è meglio di così).

4.3.3 Risultati e Considerazioni

Alla luce di questi nuovi risultati si possono effettuare una serie di considerazioni sul sistema nel suo complesso. Inoltre, è possibile avanzare ipotesi sul calo di accuratezza nello scenario con tre utenti.

- L'aumentare del dataset e il far muovere i volontari ha aumentato di molto l'accuratezza del sistema. Negli articoli i dataset sono di gran lunga maggiori (si parla di ore di raccolta dati, con grandezze finali nell'ordine dei gigabyte contro i 150Mb raccolti). è dunque legittimo pensare che un dataset ancora maggiore possa migliorare l'accuratezza dello strumento, specialmente nel caso di tre utenti (con due le performance sono già incredibilmente alte).

- Il parametro più significativo al fine di riconoscere un utente potrebbe essere il peso piuttosto che l'altezza. Infatti v2 e v3 hanno peso molto simile, mentre differiscono sensibilmente in altezza (più di quando facciano v1 e v2). Questo fatto suggerisce che la massa influenzi di più le onde rispetto all'altezza.

Capitolo 5

Conclusioni e sviluppi futuri

La tesi ha affrontato il problema del Wi-Fi Sensing, tecnologia trattata in letteratura negli ultimi anni. I dispositivi WiFi OFDM per funzionare al meglio, calcolano ogni istante dei dati riguardanti le onde elettromagnetiche, in particolare in che modo l'ambiente le modifica. Tramite questi dati sull'ambiente è possibile estrarre informazioni sulle persone (o la loro assenza) che si trovano in esso.
Questa tesi ha voluto concentrarsi sul compito della Human Identification. In altre parole, si vuole riconoscere quale individuo è presente nell'ambiente a partire dai dati raccolti. Questo tipo di compito è stato approfondito in letteratura, ma mai con dispositivi a basso costo come gli ESP32.
Questa tesi si occupa quindi di fornire uno strumento ponte fra due sistemi allo stato dell'arte, *ESP-CSI-Tool* e *SHARP*. Il primo al fine di effettuare raccolta dati sui dispositivi mentre il secondo per procedere al WiFi Sensing su tali dati. Tale ponte viene strutturato in due componenti chiamati **ADAPTER** e **AUTOSHARP**. Dopo la creazione di questo strumento si passa poi alla realizzazione di un esperimento al fine di valutare il progetto e rispondere ad alcune domande.

5.1 Conclusioni

Nella seguente sezione si vogliono discutere i risultati conseguiti soprattutto in relazione ai due quesiti prefissati nella fase di progettazione. Le due domande erano:

Q1. *Il problema della Human Identification tramite Wi-Fi sensing può essere affrontato anche con dati raccolti tramite device estremamente semplici e di basso costo come ESP32?*

Q2. *Può uno sistema orientato a risolvere problemi HAR essere adattato per fare Human Identification?*

Nel rispondere a queste due domande si tiene fede al secondo esperimento, che avendo più dati a disposizione è certamente più preciso nei suoi risultati.

La risposta ad entrambe le domande, alla luce dei risultati prodotti, è positiva. Nello scenario con due utenti si raggiunge un'accuratezza generale del 95%, mentre in quello con tre utenti si arriva al 74%.

Se nel primo caso si ottengono risultati paragonabili a quelli in letteratura, lo scenario con tre utenti risulta produrre valutazioni inferiori. Occorre esplorare le possibili ragioni al fine di giustificare la risposta positiva sia a Q1 che a Q2.

In primo luogo, la strumentazione di raccolta dati, ovvero gli ESP32, sono di fascia estremamente bassa. La qualità dei dati raccolti è inferiore rispetto a quella delle ricerche analizzate nello Stato dell'Arte. In altre parole, è naturale che l'accuratezza non sia totalmente paragonabile.

In secondo luogo, nonostante i dati del secondo esperimento siano circa 3 volte più numerosi rispetto al primo, ancora si è in uno stato di povertà di dati. Le ricerche all'avanguardia, come SHARP [10], si allenano su ore e ore di raccolta dati. La dimensione dei dati raccolti è nell'ordine delle decine di Gigabyte. Nel secondo esperimento si arriva al più a 150Mb. è facile convincersi che con dataset più popolosi la precisione del sistema potrebbe migliorare ulteriormente, rivaleggiando strumenti allo stato dell'arte come *WFID* o *WiWhoo*.

La seconda domanda in particolare sembra avere una risposta definitivamente positiva, sopratutto in vista delle precedenti considerazioni. SHARP pare essere totalmente in grado di essere riadattato come sistema di Human Identification invece che Activity Recognition. Questo indica che il tipo di preprocessing effettuato sui dati, come il design della CNN, non riconoscano solo il tipo di attività, ma riescano anche ad estrarre informazioni sul come quell'attività viene effettuata. SHARP non solo riconosce che un individuo sta camminando, ma è in grado di associare una particolare camminata ad un certo utente, con precisione molto superiore rispetto alla scelta casuale.

5.2 Sviluppi Futuri

In base alle considerazione effettuate sui risultati, ci sono una serie di accorgimenti e di miglioramenti che possono essere fatti. Questi miglioramenti sono volti all'ampliare il progetto di ADAPTER e AUTOSHARP. Inoltre ci sono delle possibili modifiche che potrebbero migliorare il tool di raccolta dati e SHARP. Infine si vogliono esplorare ulteriori esperimenti per poter rispondere ad altre domande, con l'obiettivo di esplorare più a fondo l'impiego di device a basso costo nel Wi-Fi Sensing.

5.2.1 Possibili miglioramenti di ADAPTER e AUTOSHARP

Più che un completo strumento CLI, l'implementazione del progetto risulta essere un prototipo non totalmente finito. Ci sono alcuni punti deboli che per mancanza di tempo non hanno ricevuto attenzione:

- Lo script che implementa AUTOSHARP non è totalmente personalizzabile. è necessario infatti modificare diverse costanti presenti nel sorgente per ottenere i risultati voluti. In particolare il nome dei file in input e i parametri volti al preprocessing sono hardcoded nel sorgente. Questo rappresenta un enorme miglioramento rispetto al procedimento

originario (riscrivere i comandi da zero ogni volta). Tuttavia si potrebbe migliorare ulteriormente con un file di configurazione esterno allo script.

- ADAPTER non è totalmente generalizzato. Si potrebbero astrarre alcune proprietà sui file di input tipo dimensione dei dati e formato. Inoltre ci sono una serie di ottimizzazioni che potrebbero essere implementate. Adapter infatti scandisce più volte l'intero input per effettuare i vari passaggi. Per favorire la velocità al posto della leggibilità si potrebbero eseguire tutti gli step in un solo ciclo. Tale modifica non è utile ai fini della tesi ma risulta importante se si riusare il tool con dataset molto più estesi.

- Potenziare l'interfaccia CLI. Per adesso il tool prende in input alcuni parametri (come -r per effettuare un reset o -p per rieseguire il preprocessing). Sarebbe molto utile ampliare l'interfaccia addirittura facendo una procedura guidata su più passi per rendere il tool più immediato e di facile comprensione.

5.2.2 Modifiche a ESP-CSI-Tool e SHARP

Gli strumenti all'avanguardia utilizzati sono pienamente funzionanti. Tuttavia si trovano ancora in fase di sviluppo e risultano quindi incompleti. Ci sono alcune modifiche che li potrebbero migliorare, specialmente in vista di interfacciarsi con altri strumenti. Entrambe queste modifiche erano state esplorate durante lo sviluppo della tesi, ma per motivi di tempo (e mancanza di documentazione) sono state abbandonate poiché non totalmente utili ai fini della tesi stessa.

Innanzitutto, come già accennato nello **Stato dell'arte** e nella descrizione del progetto, il tool ESP-CSI-Tool poò essere migliorato. Dopo aver contattato il ricercatore che lo ha sviluppato, è emerso che il tool non sfrutta tutti i dati accessibili del device ESP32. Più nello specifico sarebbe possibile collezionare matrici CSI più grandi. In questo modo si potrebbero raccogliere

più dati in meno tempo, e i dati raccolti sarebbero più precisi (disponendo di informazioni su più subcarrier). Per implementare ciò bisognerebbe mettere mano al codice sorgente del tool, lavoro non troppo complicato. La fase più time intensive sarebbe quella di testing delle modifiche. Infatti come già accennato nei capitoli precedenti, gli ESP32 hanno difficoltà a mandare una grande quantità di dati attraverso la porta seriale. Inviare matrici più grandi (si arriva addirittura a raddoppiare la dimensione) causerebbe molta difficoltà nell'inviare dati. Bisognerebbe diminuire il tasso di raccolta dati e esplorare una serie di dettagli hardware per cercare di aggirare il sistema. Per mancanza di tempo, una volta individuato il problema si è deciso di procedere senza risolverlo. Questo anche in combinazione con l'idea di testare la fattibilità di WI-Fi Sesning con pacchetti molto corti. In questo senso, l'avere meno dati più che rappresentare un problema è una costrizione interessante da sperimentare.
In secondo luogo il sistema SHARP presenta alcuni punti che potrebbero essere migliorati. Un aspetto è la gestione dei file e delle sotto cartelle, che è stata parzialmente modificata durante lo sviluppo della tesi. Occorrerebbe rendere ancora più ovvio il funzionamento. Il tool AUTOSHARP cerca di nascondere tale funzionamento per lasciare all'utente finale solo gli aspetti importanti di cui preoccuparsi. Un altro aspetto chi si è gestito parzialmente è quello di gestione del caso con più antenne. Vista l'assenza di documentazione e la difficoltà di comprensione di un codice che è ancora in sviluppo, si è optato per rimuovere totalmente questa possibilità. Sarebbe interessante invece renderla facoltativa e personalizzabile in base al tipo di input presentato.

5.2.3 Ulteriori esperimenti

Raccogliere dati si è dimostrato un compito più difficile del previsto. In particolar modo avere volontari disposti a camminare per decine di minuti è molto difficile e il compito in sè risulta essere tedioso per i volontari stessi. Per questo sono state effettuate solamente due raccolte dati. Di seguito

una sere di ulteriori esperimenti che con più tempo e più volontari sarebbe interessante esplorare.

- Ripetere l'esperimento due con più utenti e per più tempo. Un dataset molto più esteso permetterebbe in primis di aumentare l'affidabilità dei risultati ottenuti. Inoltre si potrebbe esplorare come all'aumentare degli utenti cambino le performance. Si potrebbero poi effettuare più allenamenti con diverse combinazioni di utenti per individuare le caratteristiche fisiche che più influenzano i risultati.

- Ripetere il primo esperimento di raccolta dati per più tempo. Nel primo esperimento i volontari erano statici e sarebbe molto interessante vedere se con un dataset più largo le performance migliorerebbero. Il riconoscimento di umani statici è un ambito inesplorato in letteratura e quindi sarebbe interessante stabilire se sia possibile o meno.

- Effettuare un esperimento di Human Activity Recognition. Il tool SHARP è originariemnte orientato a fare HAR, per cui sarebbe molto interessante usarlo in combinazione agli ESP32 per testare come device meno potenti influenzino l'accuratezza di questo tool.

Glossario

AP Access Point. 15, 17, 19, 23, 26

API Application Programming Interface. 27

CFR Channel Frequency Response. 7–11, 14, 16–18, 25, 28, 30

CLI Command Line Interface. 49

CNN Convolutional Neural Network. 22, 25, 40, 49

COTS Commercial Off The Shelf. 2, 22

CSI Channel State Information. 1, 5, 7–11, 15, 16, 23, 25, 27–30, 50

HAR Human Activity Recognition. i, 22, 23, 26, 48, 52

kNN k Nearest Neighbour. 13, 16

LLTF Legacy Long Training Field. 10, 28

LoS Line of Sight. 17, 18, 36, 41

NIC Network Interface Card. 9, 16

OFDM Orthogonal Frequency-Division Multiplexing. 1, 6, 7, 10, 47

PCA Principal Component Analysis. 16, 19

SFO Sampling Frequency Offsets. 8

STO Sampling Time Offsets. 8

SVM Support Vector Machine. 18, 19

Bibliografia

[1] Y. Ma, G. Zhou, and S. Wang, "Wifi sensing with channel state information: A survey," *ACM Comput. Surv.*, vol. 52, no. 3, jun 2019. [Online]. Available: https://doi.org/10.1145/3310194

[2] C. Wu, Z. Yang, Z. Zhou, X. Liu, Y. Liu, and J. Cao, "Non-invasive detection of moving and stationary human with wifi," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2329–2342, 2015.

[3] S. Palipana, D. Rojas, P. Agrawal, and D. Pesch, "Falldefi: Ubiquitous fall detection using commodity wi-fi devices," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 4, jan 2018. [Online]. Available: https://doi.org/10.1145/3161183

[4] S. M. Hernandez and E. Bulut, "Wifi sensing on the edge: Signal processing techniques and challenges for real-world systems," *Commun. Surveys Tuts.*, vol. 25, no. 1, p. 46–76, jan 2023. [Online]. Available: https://doi.org/10.1109/COMST.2022.3209144

[5] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11n traces with channel state information," *ACM SIGCOMM CCR*, vol. 41, no. 1, p. 53, Jan. 2011.

[6] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity wifi," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '15.

New York, NY, USA: ACM, 2015, p. 53–64. [Online]. Available: http://doi.acm.org/10.1145/2789168.2790124

[7] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free your csi: A channel state information extraction platform for modern wi-fi chipsets," in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation Characterization*, ser. WiNTECH '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 21–28. [Online]. Available: https://doi.org/10.1145/3349623.3355477

[8] S. M. Hernandez and E. Bulut, "Lightweight and Standalone IoT Based WiFi Sensing for Active Repositioning and Mobility," in *21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM) (WoWMoM 2020)*, Cork, Ireland, Jun. 2020.

[9] Espressif, "Espressif/esp-idf: Espressif iot development framework. official development framework for espressif socs." [Online]. Available: https://github.com/espressif/esp-idf

[10] F. Meneghello, D. Garlisi, N. Dal Fabbro, I. Tinnirello, and M. Rossi, "SHARP: Environment and Person Independent Activity Recognition with Commodity IEEE 802.11 Access Points," *IEEE Transactions on Mobile Computing*, pp. 1–16, 2022.

[11] Y. Zeng, P. H. Pathak, and P. Mohapatra, "Wiwho: Wifi-based person identification in smart spaces," in *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2016, pp. 1–12.

[12] T. Xin, B. Guo, Z. Wang, M. Li, Z. Yu, and X. Zhou, "Freesense: Indoor human identification with wi-fi signals," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–7.

[13] Y. Chen, W. Dong, Y. Gao, X. Liu, and T. Gu, "Rapid: A multimodal and device-free approach using noise estimation for

robust person identification," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 3, sep 2017. [Online]. Available: https://doi.org/10.1145/3130906

[14] F. Hong, X. Wang, Y. Yang, Y. Zong, Y. Zhang, and Z. Guo, "Wfid: Passive device-free human identification using wifi signal," in *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, ser. MOBIQUITOUS 2016. New York, NY, USA: Association for Computing Machinery, 2016, p. 47–56. [Online]. Available: https://doi.org/10.1145/2994374.2994377

[15] Meneghello, Francesca and Garlisi, Domenico and Dal Fabbro, Nicolø' and Tinnirello, Ilenia and Rossi, Michele, "Research data - università degli studi di padova," https://researchdata.cab.unipd.it/624/, 2022.

[16] Z. Li, H. Li, and L. Meng, "Model compression for deep neural networks: A survey," *Computers*, vol. 12, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2073-431X/12/3/60

[17] Hernandez, Steven M. and Bulut, Eyuphan, "Esp32 csi tool," https://github.com/StevenMHernandez/ESP32-CSI-Tool, 2020.

[18] C. Yang, R. A. Brower-Sinning, G. A. Lewis, and C. Kästner, "Data leakage in notebooks: Static detection and better processes," 2022.

[19] "multiprocessing — Process-based parallelism — docs.python.org," https://docs.python.org/3/library/multiprocessing.html#module-multiprocessing, [Accessed 29-Jun-2023].

[20] Z. Wang, M. Ma, X. Feng, X. Li, F. Liu, Y. Guo, and D. Chen, "Skeleton-based human pose recognition using channel state information: A survey," *Sensors (Basel)*, vol. 22, no. 22, Nov. 2022.

Ringraziamenti

Il sentiero della laurea triennale mi ha fatto crescere molto. Sono fiero degli obiettivi raggiunti, ma occorre riconoscere che questi obiettivi non sono solo miei. Desidero dedicare alcune parole di ringraziamento a coloro che hanno reso possibile questo cammino, accompagnandomi con sostegno e affetto lungo il percorso

In primo luogo devo ringraziare i miei genitori. Se c'è stata una costante nella mia educazione, questa è la libertà. Libertà di scegliere, di sbagliare e di affrontare le conseguenze delle proprie azioni. Grazie Mamma e Papà per sopportare un figlio talvolta difficile. Grazie al mio fratellino, che mi ha accompagnato con solidarietà nei momenti difficili e che mi rende orgoglioso ogni giorno che passa.

Ringrazio di cuore i miei "Amici di Quartiere", che conosco fin da quando ero piccolo. Ringrazio Fabio, amico preziossimo nonchè compagno di allenamento per eccellenza. Sono stati molti i giorni in cui la solita sessione di allenamento era il momento più bello e sereno della mia giornata. Ringrazio anche Giovanni e Rui, fedeli soldati che mi hanno accompagnato in un viaggio incredibile verso la sicurezza in sè stessi. Ringrazio le mie ragazze, che conosco da un po' troppi anni. Mi avete dato tante di quelle dritte (a volte più storte che dritte) e mi avete visto crescere insieme a voi. Da un noioso rompiscatole fino a un simpatico rompiscatole. Mi avete consolato quando ne avevo bisogno, strigliato quando era necessario e preso in giro praticamente sempre (vi capisco).

Occorre anche ringraziare i miei compagni di Università. Questo è un

traguardo accademico dopotutto, e senza il loro aiuto ci sarebbero tanti, troppi esami che starei ancora dando. Grazie in particolare a Geno, mio fedele compagno e caro amico, che mi ha sopportato e aiutato. Grazie anche a Luca, il mio "mentore", a cui continuo tuttora a chiedere spiegazioni e aiuti tecnici, proprio come un contadino si rivolge all'usuraio. Grazie ad Apo, un fondamentale soldato che si è imbarcato insieme a me in numerose avventure. Grazie al resto della brigata Ranzani, con cui ho condiviso intense sessioni di studio e chiacchere.

Ringrazio le mie amichette norvegesi, che hanno sconvolto la mia personalità (in positivo) rendendomi più attivo e chissà, forse più simpatico. Assieme a voi ho potuto esprimere il mio lato avventuriero.

Ringrazio il club di Mario Kart, che mi ha fatto compagnia in tante serate volte al relax. Dopo intense giornate di studio bisognava usarlo.

Ringrazio tutti gli altri amici che non hanno contribuito in modo così diretto alla mia carriera accademica. Anche voi siete stati essenziali, se non all'universtà alla mia crescita personale. Vi devo molto.

Infine intendo ringraziare il mio relatore e il mio co-relatore che mi hanno guidato nella stesura del mio primo elaborato accademico. Con loro ringrazio anche tutta la cricca di Ulisse, gruppo molto stimolante che ricorda sempre quante cose da esplorare ci siano.

Devo tante cose alle persone che circondano la mia vita. La mia gratitudine è ampia e genuina nei confronti di tutti voi.