SAPIENZA
UNIVERSITÀ DI ROMA

Sintesi di Oggetti tramite Segnali Wi-Fi

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Corso di Laurea in Informatica

Enrico Fortuna
Matricola

Relatore
Prof. Danilo Avola

Correlatori
Prof. Luigi Cinque
Dr. Marco Cascio

Anno Accademico 2021/2022

Sintesi di Oggetti tramite Segnali Wi-Fi
Tesi di Laurea Triennale. Sapienza Università di Roma

Questa tesi è stata composta con LaTeX e la classe Sapthesis.

Email dell'autore:

Sommario

Con l'elevata domanda di traffico di dati wireless e lo sviluppo dell'*Internet of Things* (IoT), le reti Wi-Fi hanno registrato una crescita molto rapida in quanto forniscono un'alta capacità di trasmissione a basso costo e sono facili da implementare.

Quando i segnali wireless si propagano nello spazio, tramite la loro riflessione, rifrazione e dispersione, sono in grado di percepire l'ambiente circostante. Questa capacità, conosciuta come Wi-Fi *Sensing*, ha trovato applicazioni in diversi campi, tra cui il riconoscimento di gesti, la localizzazione e la sintesi di oggetti. Quest'ultima ha suscitato grande interesse all'interno della comunità della *Computer Vision*, anche grazie all'utilizzo di tecniche e sistemi di *deep learning*.

Nella presente tesi viene mostrato come il canale Wi-Fi può essere sfruttato per sintetizzare immagini di oggetti. In particolare, viene estratta l'ampiezza dal *Channel State Information* - CSI di segnali wireless polarizzati in maniera differente e data in input ad una rete neurale a due rami. Una volta addestrata, l'architettura proposta è in grado di sintetizzare oggetti utilizzando esclusivamente l'ampiezza e la polarizzazione dei segnali Wi-Fi.

Indice

Capitolo 1

Introduzione

Negli ultimi anni, con l'aumento della popolarità dei dispositivi wireless e la diffusione dell'*Internet of Things* (IoT), il Wi-Fi ha registrato una crescita molto rapida. Un'importante tecnologia che ha portato al successo del Wi-Fi è il sistema MIMO (*Multiple-Input Multiple-Output*) che, insieme alla tecnica di trasmissione a multiportante OFDM (*Orthogonal Frequency-Division Multiplexing*), utilizza diverse antenne trasmittenti e riceventi per trasferire più dati contemporaneamente, consentendo un elevato *troughput*. Oltre a consentire lo scambio di dati, i segnali wireless possono essere riflessi, dispersi o rifratti dall'ambiente circostante, causando un cambiamento nella loro fase, ampiezza e polarizzazione [29]. Il Wi-Fi *sensing* è una nuova tecnologia che sfrutta le proprietà dei segnali Wi-Fi per rilevare e percepire l'ambiente circostante.

Recentemente le misurazioni del *Channel State Information* (CSI) dei segnali Wi-Fi vengono utilizzate per diversi task di rilevamento. In particolare il CSI descrive il modo in cui il segnale si propaga dal trasmittente al ricevitore con determinate frequenze portanti, lungo percorsi multipli. Per un sistema Wi-Fi con tecnologia MIMO-OFDM, il CSI è una matrice 3D di valori complessi che rappresentano l'attenuazione dell'ampiezza e lo spostamento di fase dei canali Wi-Fi. Una serie di misurazioni del CSI descrive il modo in cui i segnali wireless attraversano gli oggetti e le persone circostanti e può essere utilizzata per diversi scopi di Wi-Fi *sensing* tra cui il riconoscimento di gesti, la localizzazione e la sintesi di oggetti.

Le motivazioni per cui il Wi-Fi *sensing* ha riscosso così tanto successo sono molteplici: primo, sfrutta le infrastrutture già presenti ed utilizzate dalle comunicazioni wireless, quindi è facile da sviluppare ed è a basso costo; secondo, a differenza dei sensori video, preserva la privacy, in quanto non vengono raccolte informazioni sensibili; infine, non soffre di problemi di visibilità, causati da una scarsa illuminazione o dall'occlusione ambientale. In genere le applicazioni basate sui video o foto soffrono di una bassa risoluzione o richiedono l'utilizzo di diversi sensori troppo costosi. Al contrario, le applicazioni basate sul Wi-Fi *sensing* permettono di coprire vaste aree anche utilizzando pochi dispositivi a basso costo.

L'argomento principale di questa relazione riguarda la sintesi di immagini (*Image Synthesis*) a partire dai segnali Wi-Fi. In generale L'*Image Synthesis* si riferisce al processo di generazione di immagini utilizzando come sorgente informazioni diverse dalle immagini stesse.

Per eseguire questo compito, viene utilizzata una rete neurale artificiale a due rami, ispirata da [3], in grado di convertire l'informazione ottenuta dai segnali Wi-Fi in immagini, seguendo uno schema di apprendimento di tipo Maestro-Studente [4]. Più nello specifico, vengono utilizzate le ampiezze nel tempo estratte dal CSI di segnali Wi-Fi polarizzati diversamente, per sintetizzare immagini raffiguranti degli oggetti di comune utilizzo, come una bottiglia di plastica o una lattina di alluminio. L'antenna che riceve i segnali viene polarizzata in maniera differente poiché, come mostrato in [14, 29], la polarizzazione è una proprietà dei segnali Wi-Fi in grado di descrivere le proprietà fisiche degli oggetti.

1.1 Stato dell'arte

Attualmente, sebbene siano stati proposti vari metodi, la sintesi di immagini rimane un problema complesso. Esistono diversi tipologie di *Image Synthesis* in base al task che si vuole risolvere. L'Image Synthesis Vincolata ha lo scopo di sintetizzare una nuova immagine rispettando dei vincoli specificati dall'utente, come un'altra immagine, una descrizione di un testo o degli abbozzi di disegni.

Negli ultimi anni, la Rete Generativa Avversaria (in inglese *Generative Adversarial Network* - GAN) è stata ampiamente utilizzata per la sintesi di immagini. In [11] viene proposto un metodo di traduzione da immagine a immagine utilizzando le *Conditional GANs* (cGANs). Tale metodo si è dimostrato efficace per una varietà di conversioni, generando immagini con una risoluzione di 256x256 pixel.

Con *text-to-image synthesis* ci si riferisce ad un metodo di calcolo che trasforma una descrizione testuale in immagini con una semantica simile alle descrizioni. In [21] Reed, Akata, Yan e altri utilizzano le GANs per un metodo di sintesi da testo a immagine. Il testo in input viene codificato in uno spazio latente utilizzando una Rete Neurale Ricorrente. Sempre Reed, in [22], introduce una rete GAN *What-Where* (GAWWN) alla quale, oltre al testo, viene dato in input un vincolo di posizione.

Altre tecniche di sintesi utilizzano come vincolo non testo ma uno *sketch* ovvero un modo rapido e conveniente per l'utente di disegnare quello che vuole, senza includere troppi dettagli. In [24] vengono sintetizzate immagini realistiche a partire da *sketch* con tratti colorati, in cui il generatore utilizza un autoencoder con strati convoluzionali. Xian e altri, [27], convertono *sketch* in immagini utilizzando in aggiunta le texture degli oggetti, tramite un metodo di sintesi basato su una rete neurale convoluzionale (CNN).

Scene Graph to Image Generation prevede la generazione di un'immagine utilizzando dei grafi, dove i nodi rappresentano gli oggetti, mentre gli archi definiscono le loro relazioni con altri oggetti. Questa tecnica è in grado di generare immagini complesse in maniera più efficiente rispetto alle descrizioni testuali. Diversi sono gli articoli che utilizzano questa tecnica, tra cui [1, 12, 15, 16], ma in tutti i casi la dimensione delle immagini ottenute non supera i 256x256 pixel.

Alcuni studi hanno mostrato la possibilità di estrarre informazioni fisiche dalle onde radio, utilizzando il Wi-Fi *sensing*. In [13], Kato e altri, introducono CSI2Image, un metodo di conversione dal CSI dei segnali wireless ad immagini, usando le GANs. Il *training data* usato per l'addestramento della rete consiste simultaneamente di immagini da 64x64 pixel e CSI compresso. Tale metodo riesce a generare corret-

tamente gli oggetti, eccetto alcuni casi (5%), in cui una bottiglia di plastica viene confusa con una lattina di alluminio. In [29] viene utilizzato un metodo ispirato alla polarimetria radar per rilevare la posizione e il tipo degli oggetti tramite segnali Wi-Fi. L'idea alla base è che oggetti di materiale diverso riflettono le onde wireless in maniera differente, provocando un cambiamento nel *pattern* della polarizzazione dei segnali. Lo studio rivela una precisione media del 95% nel riconoscimento di tre tipi di materiali diversi: rame, alluminio, legno.

1.2 Contributi e outline

In questa relazione viene proposta un'architettura di rete neurale a due rami, in grado di sintetizzare immagini di oggetti a partire dai segnali Wi-Fi. Quest'ultimi vengono ricevuti da un'antenna ricevente polarizzata in maniera differente: prima orizzontalmente e poi verticalmente. L'ampiezza nel tempo ottenuta dal CSI estratto da tali segnali viene sanificata per ottenere *feature* radio consistenti, con lo scopo di migliorare la qualità delle immagini sintetizzate.

I capitoli della tesi sono strutturati come segue:

- **Capitolo 2**: introduce gli aspetti teorici del canale di comunicazione wireless, per comprendere la tecnologia alla base del Wi-Fi *Sensing*.
- **Capitolo 3**: tratta di *Deep Learning*, descrivendo la struttura e gli algoritmi di addestramento delle reti neurali artificiali.
- **Capitolo 4**: mostra l'architettura e il funzionamento del modello utilizzato per sintetizzare le immagini.
- **Capitolo 5**: descrive la raccolta del dataset, la progettazione, le fasi di test e la valutazione del metodo proposto.
- **Capitolo 6**: espone le conclusioni e i possibili sviluppi futuri.

Capitolo 2

Il Canale Wireless

Il seguente capitolo è organizzato come segue: il paragrafo 2.1 definisce le equazioni che caratterizzano una comunicazione wireless; il paragrafo 2.2 approfondisce i concetti precedenti, tenendo in considerazione il ritardo di propagazione dell'onda, il rumore dei segnali e diversi fenomeni tipici di uno scenario reale; il paragrafo 2.3 descrive la metrica RSSI utilizzata per misurare l'intensità dei segnali; i paragrafi 2.4 e 2.5 descrivono, rispettivamente, la tecnica di trasmissione a multi-portante OFDM e il suo metodo di misurazione del canale (CSI); il paragrafo 2.6 definisce l'ampiezza del segnale e il suo procedimento di sanificazione; infine, il paragrafo 2.7 si concentra sul concetto di polarizzazione dei segnali e delle antenne wireless.

2.1 Modello fisico del canale

La principale caratteristica delle comunicazioni wireless, come la trasmissione Wi-Fi, è l'utilizzo della radiazione elettromagnetica (EM) per trasmettere informazioni da un'antenna trasmittente (TX) a un'antenna ricevente (RX). Tale radiazione viaggia a velocità costante, sfruttando una forma d'onda sinusoidale, e si propaga attraverso un mezzo di trasmissione non guidato, come l'aria o lo spazio. Ogni onda EM è generata da oscillazioni tra campi elettrici e magnetici che producono un campo elettromagnetico.

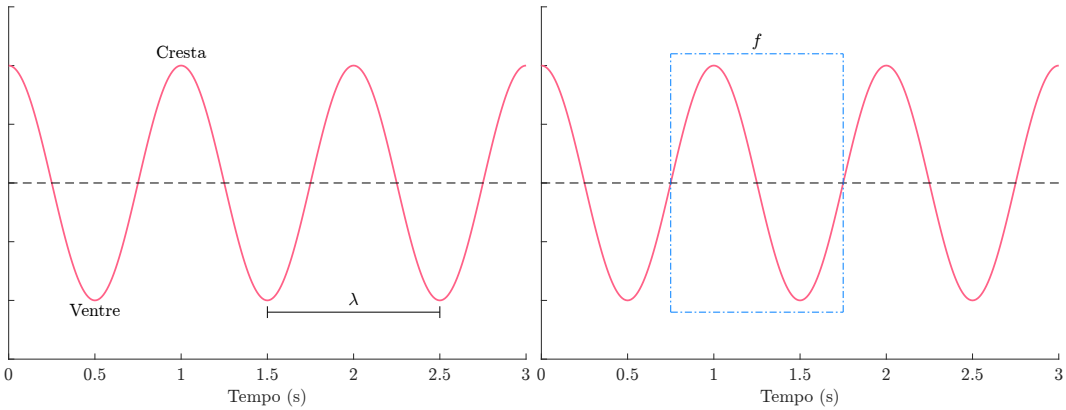
Le principali proprietà della radiazione elettromagnetica sono la lunghezza d'onda e la frequenza. La prima, mostrata nella figura 2.1a, rappresenta la distanza tra due creste o due ventri dell'onda e si misura in metri (m). La seconda, mostrata nella figura 2.1b, è il numero di cicli di oscillazione al secondo e si misura in Hertz (Hz).

Esiste una relazione tra la velocità di propagazione dell'onda v, la lunghezza d'onda λ e la frequenza f che può essere espressa come segue:

$$v = \lambda f \tag{2.1}$$

dove v è la velocità della luce nel vuoto, cioè 3 x 108 m/s o inferiore in un mezzo trasmissivo differente. Di conseguenza, partendo dall'eq. 2.1, λ e f possono essere definiti come:

$$\lambda = \frac{v}{f} \tag{2.2}$$



(a) **(b)**

Figura 2.1. Rappresentazione grafica della forma d'onda: in (a) la lunghezza d'onda λ, definita come la distanza tra due punti consecutivi di massimo (o di minimo) di una funzione periodica, in (b) la frequenza f in secondi, dove la sezione delimitata dal rettangolo blu tratteggiato corrisponde ad un ciclo di oscillazione.

$$f = \frac{v}{\lambda} \tag{2.3}$$

pertanto la frequenza e la lunghezza d'onda sono direttamente proporzionali alla velocità di propagazione e inversamente proporzionali tra loro.
Il campo elettromagnetico cambia a seconda della distanza dalla sorgente trasmittente, come mostrato nella fig. 2.2. Il campo che si trova al di sotto di una lunghezza d'onda dall'antenna TX è chiamato campo vicino (*Near-Field*). All'interno di questa regione, la forza della radiazione diminuisce rapidamente man mano che l'onda si allontana dalla sua sorgente e un ricevitore nelle vicinanze può influenzare la radiazione del trasmettitore. Questa regione del campo elettromagnetico viene infatti utilizzata per implementare la comunicazione NFC (*Near-Field Communication*), dove è necessario che il tag NFC del ricevitore si trovi entro 4cm dal trasmettitore.

Tuttavia, a partire da due lunghezze d'onda di distanza dall'antenna TX, inizia un'altra regione del campo elettromagnetico che prende il nome di campo lontano (*far-field*). In quest'area, se ci si allontana dalla sorgente trasmittente, la potenza del campo EM diminuisce inversamente al quadrato della distanza dall'antenna TX, e un ricevitore non influenza il campo generato dall'antenna trasmittente. Grazie alle sue proprietà, il campo lontano consente le trasmissioni a lungo raggio richieste dalla maggior parte delle applicazioni di Wi-Fi *sensing*, le quali sfruttano i principali metodi di comunicazione radio, come il Bluetooth o la Wi-Fi.

Per definire un modello fisico di un canale wireless si utilizza un'antenna fissa TX che emette un'onda radio nello spazio libero (*free space*). Il campo elettrico e il campo magnetico sono proporzionali e perpendicolari tra di loro. Essi sono perpendicolari anche rispetto alla direzione di propagazione dell'onda, quindi è sufficiente stimarne solamente uno per descrivere il *far-field* della sorgente trasmittente.
In risposta a una sinusoide $cos\ 2\pi ft$ trasmessa nel vuoto e in assenza di un'antenna RX, il campo elettrico al tempo t può essere modellato come segue:

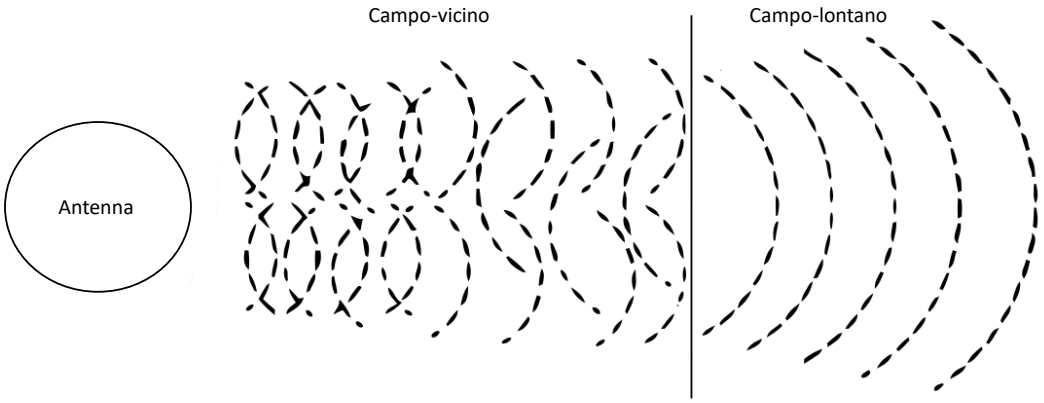


Figura 2.2. Le due diverse regioni del campo elettromagnetico: campo vicino e campo lontano

$$E(f,t,(r,\theta,\psi)) = \frac{\alpha_s(\theta,\psi,f)\ cos\ 2\pi f(t-r/c)}{r} \tag{2.4}$$

dove (r,θ,ψ) è il punto nello spazio in cui viene misurato il campo elettrico E; r è la distanza dal trasmettitore; θ e ψ sono rispettivamente gli angoli orizzontale e verticale tra l'antenna TX e il punto considerato nello spazio. A causa del vuoto, c indica la velocità di propagazione della luce; $\alpha_s(\theta,\psi,f)$ è il diagramma di radiazione nel *far-field* del dispositivo trasmittente nella direzione (θ,ψ), alla specifica frequenza f; il termine α_s è un fattore scalare che tiene in considerazione le perdite dell'antenna.

Il diagramma di radiazione (*radiation pattern*) è la rappresentazione grafica delle proprietà di radiazione di un'antenna in funzione della sua direzione angolare; in altre parole, descrive come l'antenna irradia, o riceve, l'energia nello spazio. Un esempio grafico è mostrato nella fig. 2.3. Ogni *pattern* di radiazione è suddiviso in lobi: il lobo principale punta nella direzione avente la massima potenza di radiazione; i lobi laterali sono radiazioni indesiderate con meno potenza; il lobo posteriore è nella direzione opposta al lobo principale e possiede la potenza di radiazione minima.

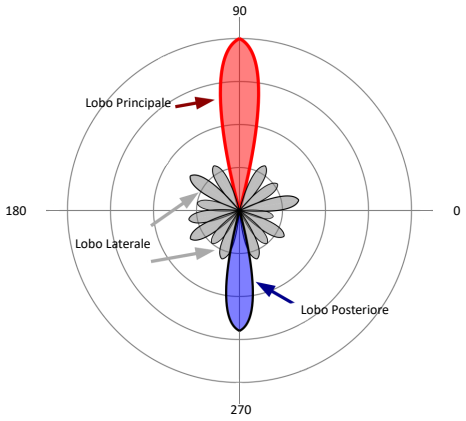


Figura 2.3. Esempio 2D del *radiation pattern* di un'antenna.

Come conseguenza del Principio di Reciprocità dell'Elettromagnetismo, nel *far field* il *radiation pattern* di un'antenna in ricezione è uguale al *radiation pattern* della stessa antenna usata come trasmittente [5]. Di conseguenza, se consideriamo la presenza di un'antenna fissa RX in una posizione specifica $\boldsymbol{x} = (r, \theta, \psi)$, in assenza di rumore, la forma d'onda ricevuta in risposta alla sinusoide precedente, eq. 2.4, può essere definita linearmente[1]

$$E_r(f, t, \boldsymbol{x}) = \frac{\alpha(\theta, \psi, f)\, cos\, 2\pi f(t - r/c)}{r} \tag{2.5}$$

dove $\alpha(\theta, \psi, f)$ è il risultato dei pattern di trasmissione e di ricezione delle antenne TX e RX nella direzione (θ, ψ). Aggiungendo l'antenna RX, il campo elettrico cambia in prossimità di $\boldsymbol{x}$ e questo viene tenuto in considerazione dal *radiation pattern* dell'antenna ricevente.

Entrambe le equazioni 2.4 e 2.5 sono lineari nell'input; ne consegue che il campo EM, ricevuto in $\boldsymbol{x}$ in risposta ad una somma pesata delle onde trasmesse, è la somma pesata delle risposte a ciascuna di queste forme d'onda.

Così, per una data posizione $\boldsymbol{x}$, possiamo definire:

$$H(f) := \frac{\alpha(\theta, \psi, f)\, e^{-j2\pi fr/c}}{r} \tag{2.6}$$

Si ha, quindi, che $E_r(f, t, \boldsymbol{x}) = \Re[H(f)e^{j2\pi ft}]$, dove $\Re$ indica la parte reale di un numero complesso. Pertanto, $H(f)$ è la definizione dell'eq. 2.5 come funzione di sistema di un canale lineare tempo-invariante nel dominio della frequenza e la sua trasformata inversa di Fourier è la corrispondente risposta all'impulso nel dominio temporale. La proprietà tempo-invariante, tuttavia, non vale se le antenne o eventuali ostacoli sono in movimento.
Consideriamo un'antenna fissa TX e il modello precedente definito nell'eq. 2.4. Questa volta, però, il punto $\boldsymbol{x}$ è in movimento, con una velocità v, e allontanandosi dall'antenna trasmittente la distanza r può essere espressa come $r(t) = r_0 + vt$. Il campo elettrico nel punto $\boldsymbol{x}$ in movimento, al tempo t, è definito come:

$$E_r(f, t, (r_0 + vt, \theta, \psi) = \frac{\alpha(\theta, \psi, f)\, cos\, 2\pi f(t - r_0/c - vt/c)}{r_0 + vt} \tag{2.7}$$

Se ci focalizziamo sull'effetto Doppler di $-fv/c$ possiamo riscrivere $f(t - r_0/c - vt/c)$ come $f(1-v/c)t-fr_0/c$. Questo è causato dal movimento del punto osservato rispetto alla sorgente trasmittente. Pertanto la sinusoide cambia da una frequenza f a una frequenza di $f(1 - v/c)$. Di conseguenza, il modello definito dall'eq. 2.5 può essere riscritto come:

$$E_r(f, t, \boldsymbol{x}(t)) = \frac{\alpha(\theta, \psi, f)\, cos\, 2\pi f[(1 - v/c)t - r_0/c]}{r_0 + vt} \tag{2.8}$$

dove $\boldsymbol{x}(t) = (r(t), \theta, \psi)$, con $r(t) = r_0 + vt$.

[1]Con il termine lineare si indica un'equazione, o un'espressione algebrica, in cui ogni termine è una costante o una variabile di primo grado. Si parla quindi di sistema lineare, combinazione lineare, equazione lineare ecc.

L'eq. 2.8 non può essere descritta come una funzione di sistema di un canale lineare tempo-invariante (LTI); tuttavia, ignorando la distanza $r(t)$, possiamo definire il canale come una funzione di sistema, considerando l'effetto Doppler $-fv/c$ piuttosto che la frequenza f.

2.2 Sistema Lineare per il Canale Wireless

Nel paragrafo precedente abbiamo definito il canale wireless come la risposta a un input sinusoidale, $\phi(t) = cos2\pi ft$, il quale si propaga nello spazio libero, osservandone il funzionamento nel caso di un'antenna ricevente fissa e di una in movimento.
Negli scenari reali, tuttavia, è necessario tenere in considerazione due elementi: il ritardo di propagazione dell'onda e diversi fattori di attenuazione del segnale. Il primo, tipicamente indipendente dalla frequenza, è causato dalla perdita di percorso (*path loss*) del segnale in funzione della distanza e dall'ombreggiamento (*shadowing*) provocato dall'ostruzione di grossi ostacoli lungo il percorso, come ad esempio i palazzi. Il secondo, tipicamente dipendente dalla frequenza, è causato dall'interferenza costruttiva e distruttiva dovuta all'effetto multi-percorso (*multi-path effect*). Quest'ultimo fa si che diverse repliche del segnale raggiungano il ricevitore attraverso più percorsi, interferendo tra di loro nell'ampiezza e nella fase. Di conseguenza, secondo le precedenti osservazioni, un canale wireless *multi-path* alla specifica frequenza f, può essere espresso linearmente come:

$$y(f,t) = \sum_i a_i(f,t)\phi(t - \tau_i(f,t)) \tag{2.9}$$

dove $y(f,t)$ è il segnale ricevuto in risposta a una sinusoide trasmessa $\phi(t)$; $a_i(f,t)$ e $\tau_i(f,t)$ sono, rispettivamente, l'attenuazione del segnale e il ritardo di propagazione al tempo t, lungo l'$i-esimo$ percorso tra le antenne TX e RX. Assumendo che ciascun percorso singolo di attenuazione e il ritardo di propagazione siano indipendenti dalla frequenza, per il principio di sovrapposizione, l'eq. 2.9 può essere semplificata come segue:

$$y(t) = \sum_i a_i(t)x(t - \tau_i(t)) \tag{2.10}$$

con $x(t)$ un segnale *bandpass*[2] trasmesso. Tuttavia, anche se assumiamo che le attenuazioni e i ritardi individuali siano indipendenti dalla frequenza, la risposta del canale complessiva può variare con la frequenza. Questo perché percorsi differenti con ritardi differenti causano un effetto di *multi-path fading*.

Visto che l'eq. 2.10 è lineare, un canale wireless *multi-path fading* può essere modellato, al tempo t, come il *channel impulse response (CIR)* di un canale lineare tempo-variante:

$$h(\tau,t) = \sum_i a_i(t)\delta(\tau - \tau_i(t)) \tag{2.11}$$

[2]Un segnale *bandpass* è un segnale a cui viene applicato un filtro utilizzato per selezionare solo le frequenze desiderate.

dove $a_i(t)$ e τ_i sono, rispettivamente, il fattore di attenuazione e il ritardo di propagazione dell'$i-esimo$ percorso, mentre $\delta(t)$ corrisponde alla funzione delta di Dirac. Quindi, confrontando le equazioni (2.9) e (2.11), la relazione tra il segnale ricevuto al tempo t e il segnale *bandpass* x trasmesso al tempo $t-\tau$ può essere definita in termini di risposta all'impulso come:

$$y(t) = \int_{-\infty}^{\infty} h(\tau, t)x(t-\tau)d\tau \tag{2.12}$$

Applicando la trasformata di Fourier (FT) alla risposta all'impulso, è possibile stimare il corrispondente *channel frequency response (CFR)* tempo-variante, definito come:

$$H(f;t) = \int_{-\infty}^{\infty} h(\tau,t)e^{-j2\pi f\tau}d\tau = \sum_i a_i(t)e^{-j2\pi f\tau_i(t)} = |H(f;t)|e^{j\angle H(f;t)} \tag{2.13}$$

dove $|H(f;t)|$ e $\angle H(f;t)$ indicano, rispettivamente, l'ampiezza e la fase del segnale e j è la componente immaginaria.

Infine, nel dominio della frequenza, un canale tempo-variante può essere modellato linearmente come:

$$y(t) = H(f;t)\, x(t-\tau). \tag{2.14}$$

Si noti, tuttavia, che nel caso di un ambiente statico, con tutte le antenne TX e RX fisse, il canale wireless può essere descritto come un sistema lineare tempo-invariante (*linear time-invariant* - LTI) con le equazioni 2.12 e 2.13 specificate come segue:

$$h(\tau) = \sum_i a_i\, \delta(\tau - \tau_i) \tag{2.15}$$

$$H(f;t) = \int_{-\infty}^{\infty} h(\tau)e^{-j2\pi f\tau}d\tau = \sum_i a_i\, e^{-j2\pi f\tau_i} = |H(f)|e^{j\angle H(f)} \tag{2.16}$$

dove le attenuazioni a_i e i ritardi di propagazione τ_i non cambiano col tempo t.

Come ultimo step, bisogna considerare ulteriori fattori casuali, oltre ai fattori che causano il ritardo di propagazione e l'attenuazione del segnale, indicati come rumore. Nel mondo reale, tutti i sistemi di comunicazione devono affrontare tale effetto di rumore come mostrato nella fig. 2.4. È fisicamente impossibile infatti avere un canale senza rumore; solitamente per modellare il rumore del segnale ricevuto viene scelto il rumore Gaussiano bianco additivo (AWGN, *additive white Gaussian noise*). Quindi, in condizioni rumorose, il canale espresso dall'eq. 2.10 può essere definito come:

$$y(t) = \sum_i a_i(t)x(t-\tau_i(t)) + \omega(t) \tag{2.17}$$

dove $\omega(t)$ è la componente AWGN al tempo t.

La componente del rumore è additiva perché, nell'equazione precedente, l'antenna RX calcola la somma tra una componente priva di rumore, ovvero il segnale che

sarebbe stato ricevuto in assenza di rumore, e la componente rumorosa indipendente dal segnale trasmesso. Tale componente è casuale e ad ogni istante di tempo viene estratta da una distribuzione Gaussiana; questa viene scelta perché il rumore si ottiene sommando più fattori indipendenti, che consentono l'applicazione del Teorema del limite centrale[3] [7]. Infine il rumore è chiamato bianco perché ha una potenza uniforme lungo tutta la banda di frequenza.

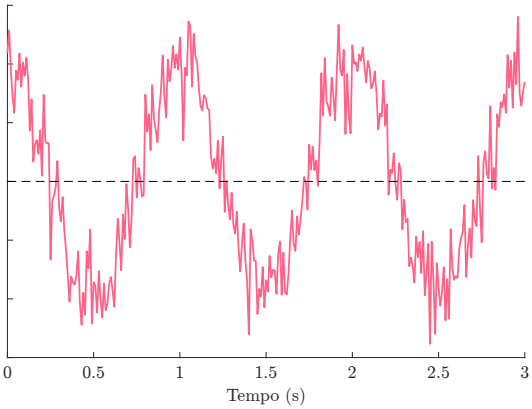


Figura 2.4. Esempio di segnale affetto dal rumore Gaussiano bianco.

2.3 Received signal strength indicator

Man mano che la radiazione EM si allontana dal trasmettitore e incontra degli ostacoli lungo il suo cammino, il segnale ricevuto da un'antenna RX è caratterizzato da una riduzione dell'intensità della potenza d'onda. L'RSSI (*received signal strength indicator*) indica la qualità del segnale tenendo in considerazione le variazioni della potenza del canale wireless. Questa misurazione si è dimostrata adatta per valutare la qualità dei collegamenti nelle reti wireless e nello sviluppo di diverse applicazioni di Wi-Fi *sensing*. Formalmente, l'RSSI si misura in decibel (dB) e di solito viene espresso secondo il *log-distance path loss model*[2], definito come:

$$PL_d = PL_{d_0} + 10n\, log_{10}\frac{d}{d_0} + \chi \tag{2.18}$$

dove PL_d è la perdita di percorso (*path loss*) da una distanza arbitraria d, n è l'esponente di perdita di percorso dipendente dall'ambiente e χ è la variabile casuale Gaussiana con una deviazione standard di σ, che rappresenta il rumore utilizzato per modellare le variazioni di dovute allo *shadowing*. Infine PL_{d_0} è la perdita di percorso ad una distanza d_0 solitamente calcolata sfruttando la perdita di percorso nello spazio libero, derivata dall'equazione di trasmissione di Friss [8], definita come:

$$FSPL = 10\, log_{10}\left(\frac{4\pi df}{c}\right)^2 \tag{2.19}$$

[3]Il Teorema de limite centrale (TLC) afferma che la somma di variabili casuali indipendenti tende ad una distribuzione normale.

dove d è la distanza tra le antenne RX e TX, c è la velocità della luce e f è la frequenza radio ottenuta dividendo la velocità della luce per la lunghezza d'onda.

2.4 Orthogonal Frequency-Division Multiplexing

L'*orthogonal Frequency-Division Multiplexing* (OFDM) è una tecnica di trasmissione utilizzata dalla maggior parte delle comunicazioni wireless moderne, come i sistemi WiFi IEEE 802.11a/b/g/n/ac/ax. Si tratta di un sistema di modulazione a multi-portante, dove un flusso di dati viene trasmesso ad alta velocità utilizzando più segnali ortogonali, noti come sottoportanti, che trasportano informazioni in parallelo; ciascuna sottoportante è modulata utilizzando uno schema di modulazione a basso *symbol rate*.

Applicando un qualsiasi schema di modulazione ad una portante, le bande laterali[4] risultanti da tale processo causano la sovrapposizione tra differenti sottoportanti, creando delle interferenze a qualsiasi frequenza di sovrapposizione. Tuttavia, questo non vale per le frequenze ortogonali, dove la proprietà di perpendicolarità genera tra le sottoportanti adiacenti dei segnali nulli, i quali sono allineati con i picchi delle sottoportanti, come mostrato nella figura 2.5. Quindi il ricevitore è in grado di recuperare il segnale originale senza interferenze, nonostante le bande laterali siano sovrapposte.

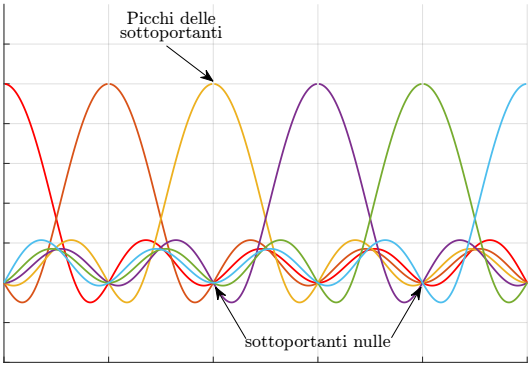


Figura 2.5. Rappresentazione grafica semplificata della tecnologia OFDM, in particolare un esempio della modulazione OFM del segnale.

La combinazione di più frequenze portanti, inoltre, permette di raggiungere un tasso di dati totale paragonabile alle tecniche di modulazione basate su una portante singola all'interno della stessa larghezza di banda, prevenendo interferenze o corruzioni del segnale causate da un eventuale effetto *multi-path*. Questo è possibile utilizzando degli intervalli di guardia, i quali assicurano che nuove repliche ritardate del segnale ricevuto non alterino i tempi e la fase del segnale stesso, poiché i dati vengono campionati solamente in una condizione stabile, come mostrato nella figura 2.6. Pertanto, la tecnica OFDM migliora il classico approccio di *frequency division multiplexing* sfruttando: più portanti per trasmettere l'informazione all'interno di un

[4]Una banda laterale è una banda di frequenze superiore o inferiore rispetto alla frequenza portante.

canale; l'ortogonalità per prevenire l'interferenza causata dalle frequenze sovrapposte; bande di guardia per la stabilità del segnale.

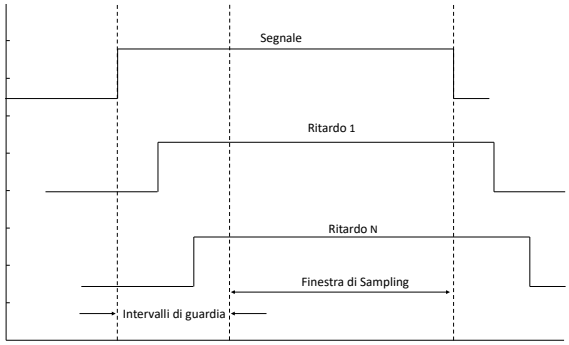


Figura 2.6. Rappresentazione grafica degli intervalli di guardia.

2.5 Channel State Information

Il *Channel State Information* (CSI) è una misura del canale wireless alternativa e più dettagliata rispetto alla metrica RSSI, descritta nel paragrafo 2.3.
Essa è ampiamente utilizzata nelle comunicazioni wireless moderne basate sul sistema OFDM, per ottenere caratteristiche dettagliate della propagazione del segnale dal trasmittente al ricevitore a livello di sottoportante. In questo modo le trasmissioni possono essere adattate alle condizioni del collegamento, consentendo comunicazioni affidabili in una configurazione delle antenne MIMO (*multiple-input multiple-output*), mostrata nella fig. 2.7. Mentre l'RSSI indica la qualità del segnale ricevuto, il CSI può catturare informazioni più ricche riguardo il segnale, come l'ampiezza, la fase, o la frequenza.
Negli ultimi anni, infatti, questa misurazione sta guadagnando grande importanza per lo sviluppo di numerosi applicazioni di Wi-Fi *sensing*, come il rilevamento della presenza umana [9, 26], il riconoscimento dell'attività [19, 28] e la ricostruzione di immagini [13].

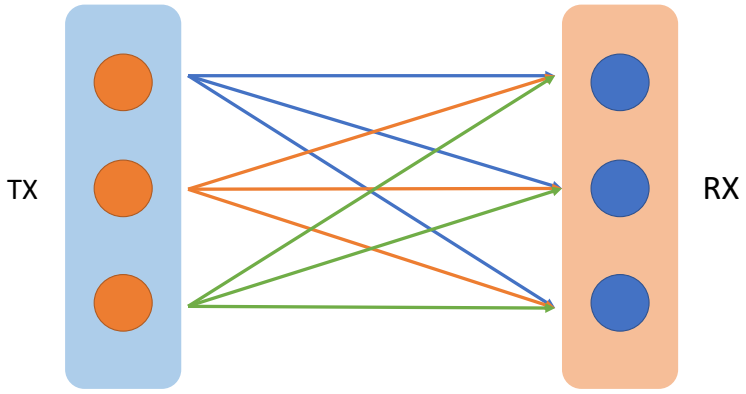


Figura 2.7. Esempio di configurazione MIMO delle antenne. Più antenne TX trasmettono segnali multipli, aventi la stessa frequenza, a più antenne RX.

In una trasmissione Wi-Fi standard, i pacchetti di dati P caratterizzano il segnale scambiato tra gli access point (APs) TX e RX. Il CSI è una metrica di valutazione nel dominio della frequenza, che coinvolge i valori CFR calcolati per tutte le K sottoportanti OFDM relative a ciascun pacchetto $p \in P$ del ricevitore. Dati gli array Θ e Γ delle antenne fisse, rispettivamente di trasmissione e di ricezione, posizionate in un ambiente statico, per ogni sottoportante $k \in K$ della comunicazione wireless stabilita tra le antenne $\theta \in \Theta$ e $\gamma \in \Gamma$, la risposta in frequenza $H(f)_k^{\theta,\gamma}$ può essere specificata come:

$$H(f)_k^{\theta,\gamma} = |H(f)_k^{\theta,\gamma}| e^{j\angle H(f)_k^{\theta,\gamma}} \tag{2.20}$$

dove $|H(f)_k^{\theta,\gamma}|$ è l'ampiezza del segnale, $\angle H(f)_k^{\theta,\gamma}$ è la fase del segnale, j è la componente immaginaria risultante dalla trasformata di Fourier applicata alla risposta all'impulso.
Pertanto, la misurazione finale del CSI stimata su tutte le K sottoportanti, considerando tutte le antenne negli array TX e RX, è una matrice complessa di dimensioni $\Theta \times \Gamma \times K$ definita come:

$$CSI = \begin{bmatrix} H(f)_1^{(1,1)} & H(f)_2^{(1,1)} & \dots & H(f)_k^{(1,1)} \\ H(f)_1^{(1,2)} & H(f)_2^{(1,2)} & \dots & H(f)_k^{(1,2)} \\ \vdots & \vdots & \vdots & \vdots \\ H(f)_1^{(\theta,\gamma)} & H(f)_2^{(\theta,\gamma)} & \dots & H(f)_k^{(\theta,\gamma)} \end{bmatrix} \tag{2.21}$$

dove $H(f)_k^{(\theta,\gamma)}$ è un numero complesso con segno a 8-bit che indica il $k-esimo$ valore CFR della sottoportante sulle antenne $\theta \in \Theta$ e $\gamma \in \Gamma$.

2.6 Ampiezza del segnale

Tramite la misura del CSI di un segnale è possibile ottenere la sua ampiezza. Concettualmente essa indica la variazione tra il valore massimo raggiunto dall'onda e la sua posizione di equilibrio, come mostrato nella figura 2.8.

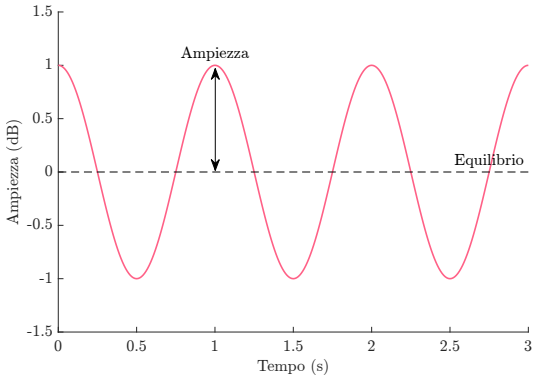


Figura 2.8. Esempio di ampiezza di un'onda EM.

In pratica ci permette di esprimere la potenza del segnale radio riportata nella scala dei decibel. Negli scenari reali, tuttavia, il canale wireless non è privo di rumore e viene quindi modellato come nell'eq. 2.17. Di conseguenza, anche se è

possibile ricavare l'ampiezza del segnale dalla matrice CSI, essa richiede un'ulteriore elaborazione per attenuare possibili rumori causati dalle condizioni ambientali e dalle specifiche del protocollo wireless; questo è possibile utilizzando varie strategie di rimozione dei valori anomali (*outliers*) [20, 6].

2.6.1 Sanificazione dell'ampiezza

Durante la misurazione del CSI possono presentarsi dei valori anomali che, dal momento in cui influenzano la procedura di estrazione dell'ampiezza del segnale, dovrebbero essere rimossi. Sanificare l'ampiezza tramite una strategia basata sul filtraggio consente di eliminare informazioni radio non rilevanti, ovvero di ridurre il rumore causato da fattori esterni, come ad esempio le interferenze radio.
A tal proposito, tramite l'identificatore di Hampel [23], è possibile identificare come valore anomalo un qualsiasi punto che cade al di fuori dell'intervallo chiuso $[\mu - \xi\sigma, \mu + \xi\sigma]$, dove μ e σ sono, rispettivamente, la mediana e deviazione mediana assoluta (*median absolute deviation* - MAD) della sequenza di dati, ed ξ è una costante dipendente dall'applicazione in questione. In maniera più precisa, data una finestra scorrevole di lunghezza fissata, i valori anomali sono identificati dai punti che si allontanano dalla media locale più di ξ MAD all'interno di questa finestra di pacchetti per ciascuna sottoportante. Questi valori vengono poi sostituiti con il numero precedente non anomalo per mantenere una ampiezza coerente con l'informazione. Formalmente, date le ampiezze del segnale estratte dalle misure CSI di $p \in P$ pacchetti wireless trasmessi tra le antenne TX e RX, e considerando la dimensione di una finestra w, la mediana locale è definita come segue:

$$\mu(\Omega^{p,k}) = \Omega^{p,k}_{\lceil w/2 \rceil} \tag{2.22}$$

$$\begin{aligned} \Omega^{p,k} = \Big\{ & |H(f)_k|^{p-\lceil w/2 \rceil}, \ldots, |H(f)_k|^{p+\lceil w/2 \rceil} : \\ & |H(f)_k|^{p-\lceil w/2 \rceil} < |H(f)_k|^{p+\lceil w/2 \rceil} \Big\} \end{aligned} \tag{2.23}$$

dove $\Omega^{p,k}$ è l'insieme contenente i w pacchetti vicini di ampiezza $|H(f)_k|$ della k-esima sottoportante, ordinati in maniera crescente. Si noti che per per semplicità descriviamo l'equazione per un singolo campione, anche se, μ è calcolato su tutte le $\Theta \times \Gamma \times K$ combinazioni di antenne e sottoportanti. Pertanto, il MAD locale utilizzato per rilevare i valori di ampiezza anormali è definito come:

$$\begin{gathered} \sigma(\Sigma^{p,k}) = \mu(|\Sigma^{p,k}_i - \mu\Sigma^{p,k}|) \\ \forall i,\ t.c.\ 1 \leq i \leq w \end{gathered} \tag{2.24}$$

Infine, gli intervalli in cui punti sono valori locali accettabili sono definiti come:

$$limit^{p,k} = \mu(\Omega^{p,k}) \pm \xi * \sigma(\Omega^{p,k}) \tag{2.25}$$

e ogni valore che non rientra in questi intervalli viene sostituito con il precedente valore, in modo da mantenere la coerenza delle informazioni, ottenendo ampiezze sanificate.

2.7 Polarizzazione di un'onda elettromagnetica

Come già detto nei paragrafi precedenti, nel *far-field* la direzione del campo elettromagnetico generato da un'antenna TX è ortogonale alla direzione di propagazione. Un'onda EM avente questa caratteristica prende il nome di onda trasversale. La polarizzazione è una proprietà tipica delle onde trasversali che descrive la direzione di oscillazione del vettore "campo elettrico" $\vec{E}$; quest'ultimo può puntare in qualsiasi direzione nel piano e può essere decomposto in due componenti perpendicolari. Teoricamente ci sono un numero infinito di modi per definire queste due componenti, ma nella maggior parte dei casi si utilizzano le coppie (H, V) o (P, S): la prima sta per componente orizzontale (*Horizontal*) e verticale (*Vertical*), le quali possono essere facilmente espresse tramite gli assi x e y del piano cartesiano; la seconda sta per componente parallela (*Parallel*) e perpendicolare (*Senkrecht*, dal tedesco). Se come componenti ortogonali scegliamo quella orizzontale e verticale, il vettore campo elettrico può essere decomposto come:

$$\vec{E} = E_h \hat{h} cos(kz - \omega t + \phi_h) + E_v \hat{v} cos(kz - \omega t + \phi_v) \tag{2.26}$$

dove $cos\omega t$ rappresenta la cosinusoide; $E_h = |E_h|e^{j\phi_h}$, $E_v = |E_v|e^{j\phi_v}$ sono le due componenti scelte; $\hat{h}$ e $\hat{v}$ sono i vettori unità[5], rispettivamente orizzontale e verticale, che indicano la direzione di oscillazione delle componenti. Tali vettori sono perpendicolari al vettore d'onda k che punta nella direzione z, ovvero nella direzione di propagazione. Infine, ϕ_h e ϕ_v indicano la fase della componente rispettivamente orizzontale e verticale.

Abbiamo visto quindi che la polarizzazione descrive la direzione di oscillazione del campo elettrico; in particolare, se questa avviene sempre sullo stesso piano, l'onda si dice polarizzata, se invece avviene in diverse direzioni casuali, come mostrato nella figura 2.9, l'onda non è polarizzata.

Figura 2.9. Esempio di un'onda elettromagnetica non polarizzata. Il vettore campo elettrico oscilla in maniera casuale in direzioni differenti.

2.7.1 Polarizzazione lineare, circolare ed ellittica

Si dice che un'onda piana è linearmente polarizzata se non c'è differenza di fase tra E_h e E_v, ovvero quando $\Delta_\phi = \phi_h - \phi_v = 0$ o $\Delta_\phi = \pi$: in questo caso il campo

[5]Quando mettiamo un simbolo hat a un vettore, significa che il vettore unità punta in quella direzione, ovvero in questo caso $\hat{h} = (1, 0, 0)$

elettrico oscilla lungo una sola direzione che è costante nel tempo. In base a questa direzione la polarizzazione lineare può essere suddivisa ulteriormente in orizzontale, verticale o inclinata di θ gradi, come mostrato nella figura 2.10. Se invece le due componenti hanno la stessa ampiezza, ma sono sfasate di un quarto di periodo, l'onda è polarizzata circolarmente. In particolare, se $\phi_h - \phi_v = \frac{\pi}{2}$ abbiamo una polarizzazione circolare destrorsa, se invece $\Delta_\phi = -\frac{\pi}{2}$ abbiamo una polarizzazione circolare sinistrorsa. Infine, quando E_h e E_v sono sfasate ma con un'ampiezza differente la polarizzazione si dice ellittica.

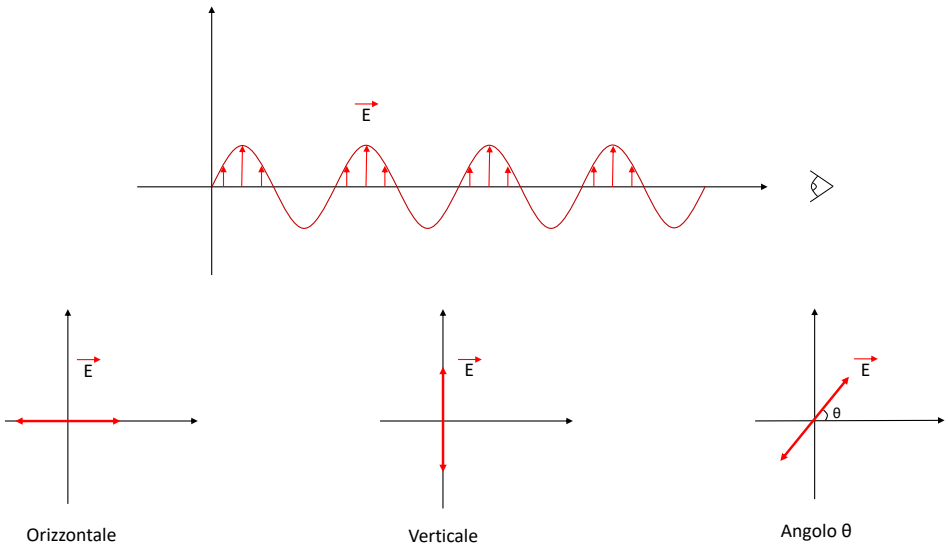


Figura 2.10. Tipi differenti di polarizzazione lineare.

2.7.2 S-polarizzazione e P-polarizzazione

Un altro sistema di coordinate frequentemente utilizzato per descrivere i segnali polarizzati riguarda il "piano di incidenza", costituito dalla direzione di propagazione dell'onda in arrivo e dal vettore perpendicolare alla superficie. In altre parole è il piano su cui viaggia l'onda prima e dopo la sua riflessione o rifrazione ed è perpendicolare alla superficie di incidenza.

La componente del campo elettrico perpendicolare a questo piano è chiamata componente s, mentre quella parallela ad esso è chiamata componente p (fig. 2.11). Di conseguenza si dice *s-polarizzata* l'onda incidente con il vettore campo elettrico normale al piano di incidenza, mentre *p-polarizzata* quella avente il vettore campo elettrico lungo il piano di incidenza.

2.7.3 Polarizzazione di un'antenna

La polarizzazione di un'antenna è definita come la polarizzazione del campo EM prodotto dall'antenna, indipendentemente dal fatto che sia in modalità di trasmissione o ricezione. Per la maggior parte delle antenne è molto facile definire la polarizzazione: essa è determinata semplicemente dalla loro struttura. Un'antenna verticale riceverà meglio i segnali polarizzati verticalmente e allo stesso modo

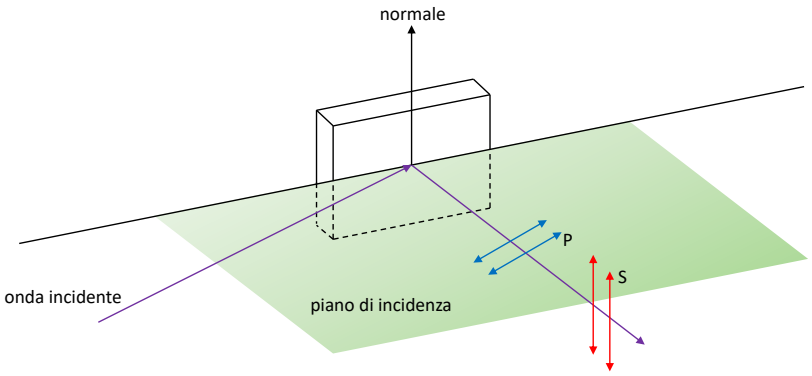


Figura 2.11. In blu viene mostrato il vettore campo elettrico dell'onda p-polarizzata, mentre in rosso quello dell'onda s-polarizzata.

un'antenna orizzontale riceverà meglio i segnali polarizzati orizzontalmente. Se consideriamo il tipo più semplice di antenna, ovvero un'antenna a dipolo, essa avrà una polarizzazione parallela al suo orientamento (fig. 2.12). Quindi basterà ruotarla fisicamente per renderla polarizzata orizzontalmente o verticalmente, anche se questa potrebbe non essere sempre la scelta migliore.

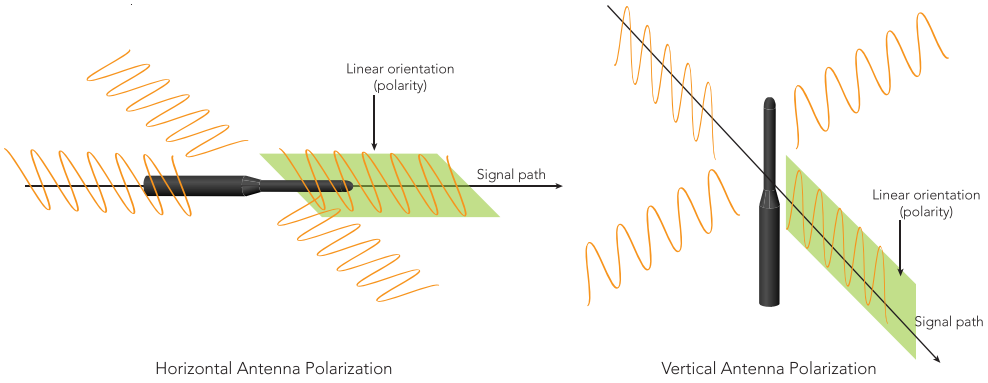


Figura 2.12. Funzionamento della polarizzazione delle antenne Wi-Fi.[17]

Quando abbiamo una coppia di antenne TX/RX la loro polarizzazione può influenzare la potenza del segnale ricevuto e quindi, per raccogliere un segnale con la massima potenza possibile, la polarizzazione dell'antenna di ricezione deve corrispondere alla polarizzazione dell'antenna di trasmissione. Il fattore di corrispondenza, o di perdita, della polarizzazione può essere misurato come:

$$\rho = |p_t^T p_r|^2 \tag{2.27}$$

dove p_t e p_r rappresentano rispettivamente gli stati di polarizzazione dell'antenna di trasmissione e di ricezione.
Per le comunicazioni radio terrestri si è riscontrato che una volta che un segnale viene trasmesso, la sua polarizzazione rimane invariata; tuttavia, poiché il segnale

ricevuto è la somma del segnale diretto più i segnali riflessi, le riflessioni dovute agli oggetti lungo il cammino possono cambiare la polarizzazione.

Capitolo 3

Deep Learning

Nel seguente capitolo, dopo una breve introduzione sul Deep Learning §3.1, vengono descritti l'architettura §3.2 e gli algoritmi di apprendimento delle reti neurali artificiali §3.3. Il Paragrafo 3.4 mostra l'implementazione e il funzionamento delle Reti Neurali Ricorrenti (RNN) ed alcune sue varianti come la Rete Neurale Ricorrente Bidirezionale (B-RNN) e la *Long Short Term Memory* (LSTM). In seguito vengono mostrate tipologie di reti più complesse come le Reti Neurali Convoluzionali (CNN), paragrafo 3.5, e gli Autoencoder §3.6.

3.1 Deep Learning

Il *Deep Learning* è un ramo del *Machine Learning* e dell'*Artificial Intelligence* (fig. 3.1), costituito da un insieme di tecniche ed algoritmi basati su un'architettura a strati (da qui l'aggettivo *Deep*, profondo). Tale struttura, organizzata in maniera gerarchica, simula il funzionamento del cervello umano e prende il nome di Rete Neurale Artificiale.

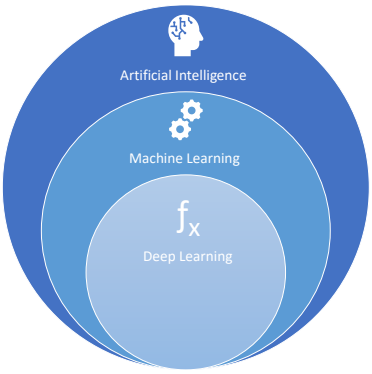


Figura 3.1. Deep Learning come sottoinsieme del Machine Learning e dell'Artificial Intelligence.

Più precisamente, il Deep Learning è una classe di algoritmi del Machine Learning che, partendo da un'informazione non elaborata chiamata *raw input*, estrae progressivamente funzionalità e caratteristiche di alto livello, definite *feature*. Ogni strato della rete riceve in ingresso l'informazione di output del livello precedente,

esegue le computazioni e lo passa al livello successivo, definendo una gerarchia di concetti. Per esempio, nell'elaborazione delle immagini, i livelli inferiori possono identificare i bordi, mentre quelli superiori possono identificare i concetti più rilevanti come lettere, cifre o volti.

Negli ultimi anni le diverse architetture di *Deep Learning* hanno riscosso grande successo, soprattutto nel campo della *Computer Vision*, dimostrandosi utili nel riconoscimento automatico di oggetti e persone a partire da immagini e video.

3.2 Rete Neurale Artificiale

Una Rete Neurale Artificiale è un modello matematico che si ispira al funzionamento di un sistema neurale biologico. In particolare, imita i processi di elaborazione che permettono a un cervello umano di apprendere e di riconoscere.
Una rete neurale è costituita da più strati, ciascuno formato dall'interconnesione di più unità elementari, chiamate neuroni o percettroni, che lavorano in modo parallelo e non lineare, come mostrato nella fig. 3.2.

L'apprendimento è uno dei concetti fondamentali della computazione neurale ed è possibile grazie alla plasticità dei neuroni biologici. Le sinapsi sono in grado di modificare la propria struttura in base agli eventi esterni per dare una risposta che sia il più adeguata possibile.
La rete neurale artificiale simula questo processo di apprendimento, ricevendo dei valori in ingresso tramite uno strato di input e restituendoli in uscita mediante uno strato di output. Le connessioni dei neuroni artificiali sono pesate e vengono aggiornate mediante degli algoritmi di apprendimento affinché la rete restituisca in output il valore più opportuno.

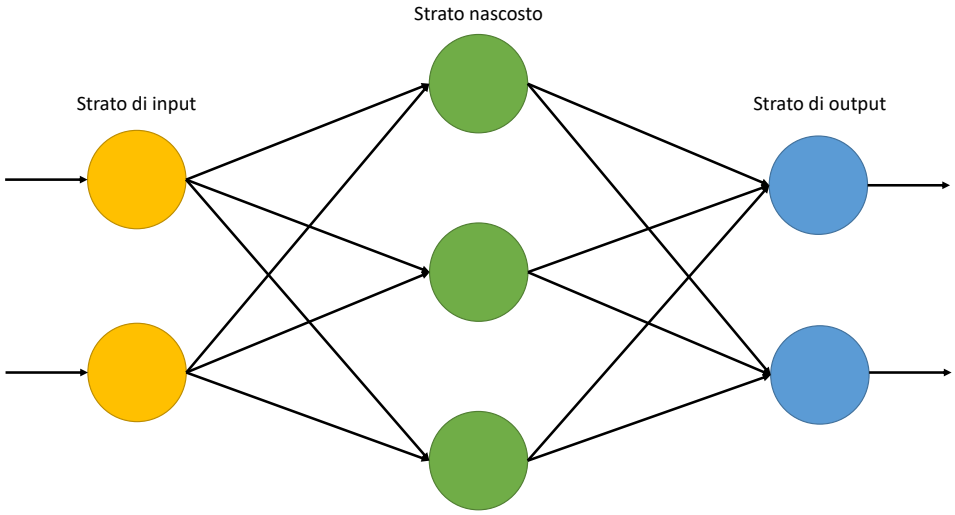


Figura 3.2. Semplice struttura di una rete neurale artificiale a due strati.

3.2.1 Percettrone Multistrato

Il Percettrone Multistrato è il tipo di rete neurale artificiale più semplice, formata da almeno tre strati: lo strato di input, lo strato di output ed uno o più strati nascosti. Ciascuno strato è formato da uno o più neuroni con una funzione di attivazione non lineare. Lo strato di input riceve l'informazione dall'esterno e la trasmette al primo strato nascosto a cui è connesso, senza applicare nessun tipo di computazione. Lo strato nascosto, invece, riceve i dati dallo strato precedente, esegue dei calcoli, e invia il risultato allo strato successivo. Infine lo strato di output è costituito da neuroni che hanno il compito di elaborare l'informazione in ingresso e trasmetterla sotto forma di valori di uscita.

Ogni strato è connesso completamente allo strato successivo, senza formare cicli e senza connessioni trasversali. Per questo motivo l'informazione fluisce solamente dallo strato di ingresso a quello di uscita e questo tipo di rete viene chiamata *feed-forward* (fig. 3.3). In seguito vedremo come è strutturato ogni neurone all'interno della rete.

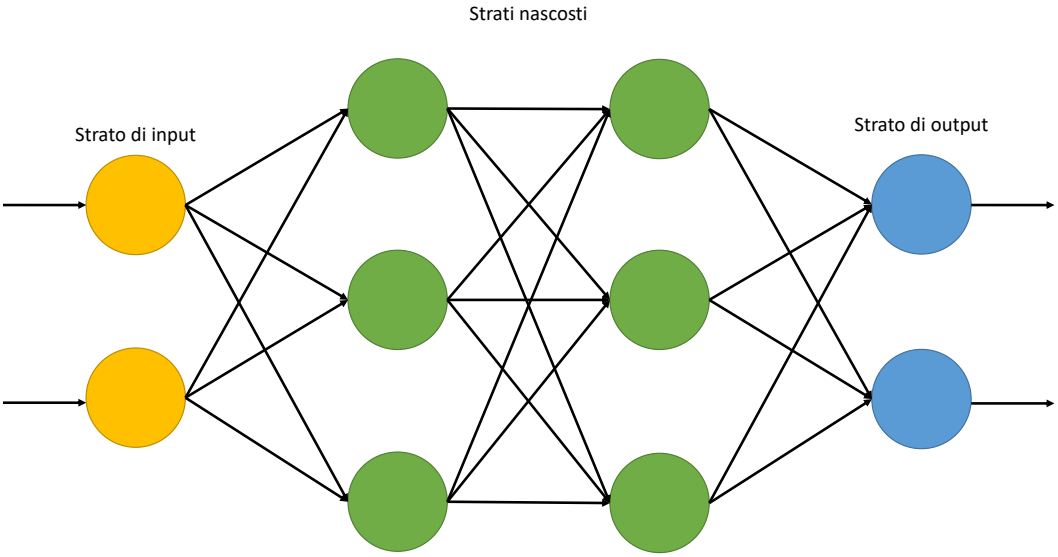


Figura 3.3. Architettura tipica di un percettrone multistrato.

3.2.2 Neurone artificiale

Il Neurone Artificiale, o Percettrone Elementare, è l'elemento più semplice di una rete neurale artificiale che corrisponde al neurone reale che si trova nel cervello umano.
Nel sistema nervoso ogni neurone è formato da quattro elementi principali: i dendriti, il soma, l'assone e i bottoni sinaptici. I dendriti sono connessi ai bottoni sinaptici di altri neuroni dai quali ricevono degli impulsi elettrici che vengono trasmessi al soma; quest'ultimo somma i segnali ricevuti e, se il risultato è maggiore di un certo valore minimo (*bias*), viene propagato ad un altro neurone attraverso l'assone.
Nel caso di una rete artificiale, il *j-esimo* neurone (fig. 3.4) riceve dei valori in input

provenienti da altri neuroni o dall'esterno. Questi valori vengono sommati dalla funzione net_j definita come segue:

$$net_j = w_{j0} + \sum_{i=1}^{n} w_{ji} x_i \tag{3.1}$$

dove w_{j0} è il *bias* associato al neurone j che può attivare o inibire il neurone; w_{ji} è il peso associato al'*i-esimo* valore di ingresso x_i. Infine, la funzione di attivazione $g(net_j)$, che cambia in base al problema trattato, calcola il valore di uscita y_j corrispondente al neurone j.

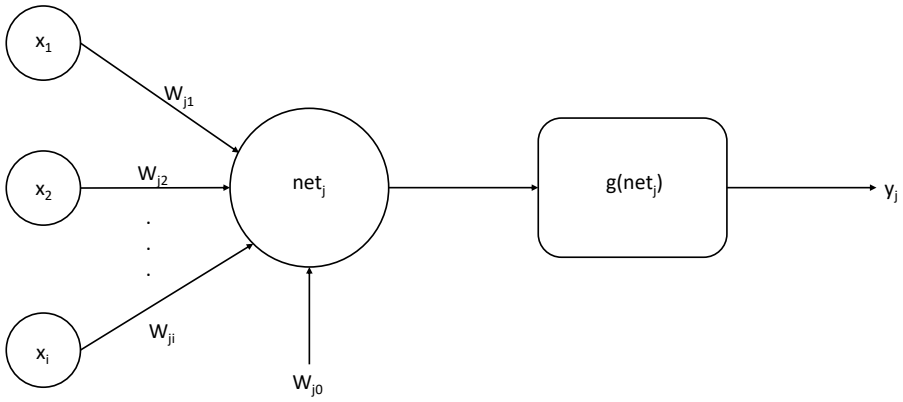


Figura 3.4. Architettura di un neurone artificiale.

3.2.3 Funzione di Attivazione

La funzione di attivazione ha lo scopo di limitare il valore di uscita in un intervallo compreso tra [-1,1] o [0,1]. Se il valore di uscita supera una certa soglia la funzione ritorna un valore vicino a 1, con lo scopo di attivare il neurone e propagare la sua attività nella rete.
Originariamente le funzioni di attivazione usate erano la funzione sigmoidea

$$g(net_j) = \frac{1}{1 + e^{-net_j}} \tag{3.2}$$

e la funzione tangente iperbolica

$$g(net_j) = \frac{e^{2net_j} - 1}{e^{2net_j} + 1} \tag{3.3}$$

Un'altra funzione di attivazione è la funzione *Heaviside*, o funzione a gradino, il cui valore è 0 per argomenti negativi e 1 per argomenti positivi:

$$g(net_j) = \begin{cases} 1 & \text{se } net_j \geq 0 \\ 0 & \text{se } net_j < 0 \end{cases} \tag{3.4}$$

Tuttavia, la funzione di attivazione maggiormente usata è la funzione a rampa ReLU (*Rectified Linear Unit*):

$$g(net) = x^+ = max(0, net_j) \tag{3.5}$$

Nel caso dei neuroni dello strato di output, la funzione di attivazione è diversa. Essi utilizzano la funzione *softmax* definita come segue:

$$g(net_j) = \frac{e^{net_j}}{\sum_{k=1}^{K} e^{net_j}} \tag{3.6}$$

dove K indica il numero di classi. Questa funzione, infatti, è spesso utilizzata in vari metodi di classificazione multi-classe.

3.3 Algoritmi di Training

Gli algoritmi di *Training*, o di apprendimento, servono per addestrare la rete neurale; essi aggiornano il valore delle connessioni pesate affinché la rete sia in grado di apprendere e fare le giuste predizioni. Esistono vari paradigmi di apprendimento:

- *Apprendimento supervisionato*: viene fornito alla rete un insieme di dati chiamato *training set*. Questi dati di addestramento sono formati da coppie (X_p, D_p) che indicano rispettivamente il *p-esimo* ingresso (x_p) e la *p-esima* uscita desiderata (d_p). Si confronta il valore di uscita predetto y_p con quello desiderato d_pe si cerca di minimizzare l'errore di previsione.
- *Apprendimento non supervisionato*: i pesi della rete vengono modificati tenendo in considerazione solamente un insieme di dati in ingresso X_p. Solitamente questi algoritmi vengono utilizzati per analizzare e raggruppare *cluster* di dati privi di *label*.
- *Apprendimento con rinforzo*: non richiede la presentazione di coppie di input/output etichettate. In questo paradigma un algoritmo appropriato cerca un particolare modo di operare: inizia osservando l'ambiente esterno in cui è immerso ed ogni azione che impatta sull'ambiente fornisce un *feedback* usato dall'algoritmo stesso nella fase di apprendimento.

3.3.1 Regola delta

L'algoritmo di *training* utilizzato da un neurone elementare viene chiamato Regola Delta. Questo è un tipo di apprendimento supervisionato che cerca di minimizzare l'errore di previsione usando un metodo di discesa del gradiente. Dato in input un *training set* non vuoto, la funzione costo E è definita come segue:

$$E = \sum_{p=1}^{T} \frac{1}{2}(d_p - y_p)^2 \tag{3.7}$$

ovvero rappresenta l'errore quadratico medio (MSE, da *Mean Square Error*) tra y_p e d_p. Si deve cercare di minimizzare questa funzione affinché la rete restituisca una risposta adeguata; per fare questo si utilizza il metodo di discesa del gradiente. Scendere lungo il gradiente significa spostarsi lungo il punto più basso di una funzione. Nel nostro caso, il gradiente è costituito dalle derivate parziali della funzione costo rispetto ai pesi della rete:

$$\nabla E(w) = [\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, ..., \frac{\partial E}{\partial w_n}] \tag{3.8}$$

Dal gradiente si ottiene la regola di discesa del gradiente usata per modificare i pesi delle connessioni del neurone. Dunque, un peso w_i si modifica come segue:

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} \tag{3.9}$$

in cui η è una costante chiamata *learning rate* che serve per definire la velocità con cui il neurone apprende. La formula per aggiornare i pesi è la seguente:

$$w_{i,t} = w_{i,t-1} + \Delta w_i \tag{3.10}$$

3.3.2 Backpropagation

Il paradigma di apprendimento più usato dal percettrone multistrato è la *Back-propagation* (o retropropagazione dell'errore). Questo algoritmo amplia la regola delta alle reti costituite da più strati. Lo scopo è sempre quello di minimizzare la funzione costo E, stavolta definita come segue:

$$E = \sum_{p=1}^{T} \sum_{j=1}^{n} \frac{1}{2}(d_{pj} - y_{pj})^2 \tag{3.11}$$

ovvero indica l'MSE tra tutti i valori desiderati d_{pj} e i corrispondenti valori di uscita predetti y_{pj}. Questa formula ci permette di calcolare l'errore dei neuroni dello strato di output.
Per lo strato nascosto non sappiamo quali siano i valori di uscita desiderati, dobbiamo quindi retropropagare l'errore dallo strato di output e aggiornare i pesi delle connessioni. Questo è possibile tramite la generalizzazione della regola delta:

$$\Delta w_{jk} = -\eta \delta_j o_k \tag{3.12}$$

dove η è il *learning rate*; o_k è l'output del neurone k, che corrisponde all'input del neurone j; δ_j è l'errore prodotto dal neurone j.

Nel caso di un neurone dello strato di output l'errore δ_j corrisponde a:

$$\delta_j = y_j(1 - y_j)(d_j - y_i) \tag{3.13}$$

Mentre per un neurone dello strato nascosto δ_j è definito come:

$$\delta_j = y_i(1 - y_j) \sum_k \delta_k w_{kj} \tag{3.14}$$

dove d_j è il valore desiderato proveniente dal *training set* e y_i è il valore predetto; $\sum_k \delta_k w_{kj}$ indica la somma pesata degli errori generati dai nodi connessi a j.

La formula per aggiornare i pesi è la seguente:

$$w_{jk,t} = w_{jk,t-1} + \Delta w_{jk} \tag{3.15}$$

3.3.3 Overfitting e regolarizzazione

Spesso può succedere che la rete neurale si abitui ai dati del *training set*. Questo problema è noto come *overfitting*; esso avviene quando ci sono troppi parametri rispetto al numero di dati disponibili e la rete non riesce a "generalizzare", cioè non è in grado di dare una risposta adeguata ricevendo in ingresso valori mai visti.

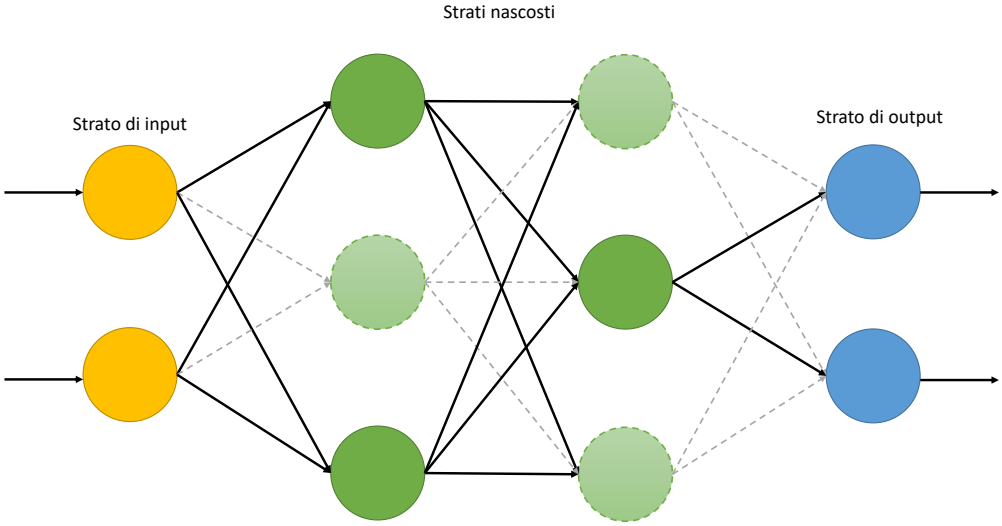


Figura 3.5. Utilizzo della tecnica di *dropout* su un percettrone multistrato.

Per affrontare l'*overfitting* si possono utilizzare dei meccanismi noti come tecniche di Regolarizzazione. Una delle più utilizzate, che prende il nome di Regolarizzazione L2 [18], somma un termine di regolarizzazione alla funzione costo di partenza E_0:

$$E = E_0 + \frac{\lambda}{2n} \sum_i {w_i}^2 \tag{3.16}$$

dove λ è il parametro di regolarizzazione; n è la grandezza del *training set*; $\sum_i {w_i}^2$ è la somma del quadrato dei pesi della rete. In questo modo si possono apprendere dei valori alti per i pesi della rete solo se si migliora la funzione costo E_0. Diminuendo λ la rete minimizza E_0, mentre aumentando λ si minimizzano i pesi; così facendo si attenua *l'overfitting*, poiché i pesi bassi rendono la rete più semplice, con una rappresentazione dei dati più robusta e meno complessa.

Un'altra tecnica di regolarizzazione è la Regolarizzazione L1, definita come segue:

$$E = E_0 + \frac{\lambda}{n} \sum_i |w_i| \tag{3.17}$$

Tale tecnica, in maniera simile alla Regolarizzazione L2, fa apprendere alla rete dei pesi bassi, riducendoli di una quantità costante che tende a zero.

Un'ultima tecnica di prevenzione per *l'overfitting* è conosciuta con il nome di *dropout*. L'idea alla base di questa tecnica è di ignorare temporaneamente e in maniera casuale un certo numero di neuroni (comprese le loro connessioni) dallo strato nascosto, come mostrato nella fig. 3.5. Di conseguenza i neuroni rimanenti dovranno apprendere caratteristiche più robuste, e il risultato è una rete capace di generalizzare in modo migliore.

3.3.4 Epoche e convergenza

Con epoca si intende un ciclo di elaborazione dell'intero *training set*. Solitamente si addestra la rete per un numero finito di epoche, suddivise in *batch* per vincoli di memoria. La rete esegue la fase di addestramento per più epoche e modifica i pesi delle connessioni alla fine di ogni ciclo; lo scopo è di minimizzare l'errore di predizione con l'aumentare delle epoche.
Un altro obiettivo da raggiungere è la convergenza sul *training set* ovvero ottenere un andamento decrescente dell'errore e un andamento crescente dell'accuratezza sui dati di addestramento, come si può vedere nella figura 3.6.

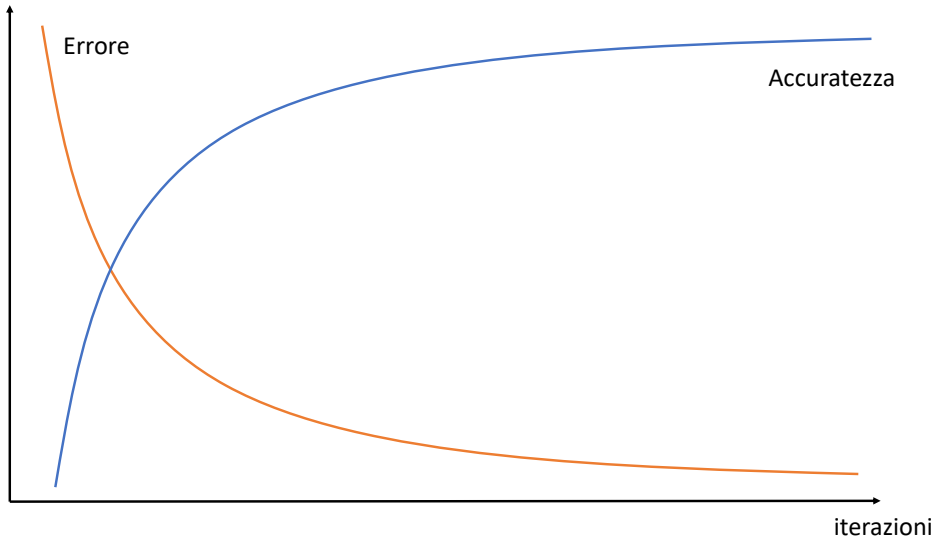


Figura 3.6. Convergenza di una rete sui dati di addestramento.

Tuttavia l'obiettivo finale della fase di *training* è di rendere la rete "generalizzata", cercando di trasferire la precisione ottenuta sui dati di addestramento ad un insieme di dati mai visti prima dalla rete, chiamato *validation* o *test set*. Per raggiungere tale scopo occorre, se ritenuto necessario, utilizzare le tecniche di prevenzione dell'*overfitting* viste nel paragrafo precedente.

3.4 Reti neurali ricorrenti

Le Rete Neurale Ricorrente (RNN) è un tipo di percettrone multistrato che consente di simulare la memoria a breve termine del cervello umano. A differenza della classica rete neurale *feed forward*, i neuroni che compongono una RNN permettono di controllare l'aspetto temporale e sequenziale dei dati di input, grazie alla presenza di cicli e connessioni trasversali. In una RNN standard, mostrata nella figura 3.7, i neuroni degli strati nascosti utilizzano una funzione di attivazione sigmoidea mentre, per la fase di addestramento, adottano il metodo della *backpropagation* nel tempo.

Per gestire gli aspetti temporali si utilizza uno stato interno che corrisponde, formalmente, all'insieme di tutti i valori di uscita dei neuroni degli strati nascosti. Questa memoria, ad ogni istante di tempo t, esegue una predizione y. Più nel dettaglio, una RNN riceve come input una sequenza di valori $x_1, x_2, \ldots, x_n \in R^n$ per calcolare una successione $h_1, h_2, \ldots, h_t \in R^m$, utilizzata a sua volta per eseguire una predizione, restituendo come output una successione $y_1, y_2, \ldots, y_m \in R^k$. Il tutto avviene eseguendo per t iterazioni le seguenti equazioni:

$$\boldsymbol{h}_t = \sigma(W_{xh}\boldsymbol{x}_t + W_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_h) \tag{3.18}$$

$$\boldsymbol{y}_t = softmax(W_{hy}\boldsymbol{h}_t + \boldsymbol{b}_y) \tag{3.19}$$

dove $\boldsymbol{h}_t$ corrisponde all'equazione del neurone dello strato nascosto e rappresenta lo stato di memoria al tempo t; σ è la funzione di attivazione sigmoidea; W_{xh} è la matrice contenente i pesi delle connessioni tra lo strato nascosto e quello di input; W_{hh} è la matrice dei pesi dello strato nascosto con se stesso in instanti di tempo successivi; y_t è l'equazione del neurone nello strato di output e rappresenta il valore di uscita al tempo t; W_{hy} è la matrice dei pesi tra lo strato nascosto e lo strato di output; $\boldsymbol{b}_h$ e $\boldsymbol{b}_y$ sono vettori di *bias*.

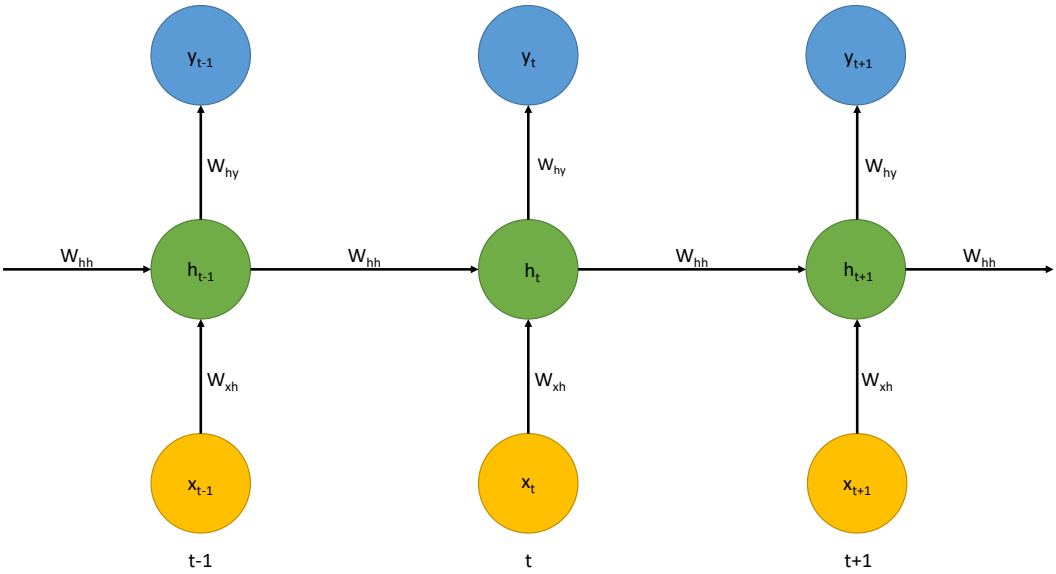


Figura 3.7. Architettura di una Rete Neurale Ricorrente.

Una possibile variante della rete neurale ricorrente è la *Bidirectional Recurrent Neural Network* (B-RNN). Questa rete possiede due strati nascosti che permettono di processare i dati in entrambe le direzioni (da qui il termine bidirezionale). Come abbiamo visto in precedenza, in una RNN standard ogni neurone di ciascuno strato, eccetto il primo, può ottenere informazioni dal passato. Una B-RNN, mostrata nella figura 3.8, permette di ottenere informazioni anche dal futuro. Le equazioni che definiscono una B-RNN sono le seguenti:

$$\boldsymbol{h}_t = \sigma(W_{xh}\boldsymbol{x}_t + W_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_h) \tag{3.20}$$

$$\boldsymbol{z}_t = \sigma(W_{xz}\boldsymbol{x}_t + W_{zz}\boldsymbol{z}_{t+1} + \boldsymbol{b}_z) \tag{3.21}$$

$$\boldsymbol{y}_t = softmax(W_{hy}\boldsymbol{h}_t + W_{zy}\boldsymbol{z}_t + \boldsymbol{b}_y) \tag{3.22}$$

dove $\boldsymbol{h}_t$ indica i valori degli strati nascosti nelle direzioni in avanti (*forward*), mentre $\boldsymbol{z}_t$ indica quelli nelle direzioni indietro (*backward*).

Anche se con le reti neurali ricorrenti sono stati raggiunti ottimi risultati, esse non sono facili da utilizzare per i problemi che richiedono dipendenze temporali a lungo termine. Utilizzando la funzione sigmoidea, se nella nostra rete sono presenti molti istanti di tempo t, il gradiente della funzione di errore decade rispetto al tempo e si va incontro al problema del *vanishing gradient*; l'annullamento del gradiente rende la nostra rete difficile da addestrare e non ci permette codificare le dipendenze temporali a lungo termine.

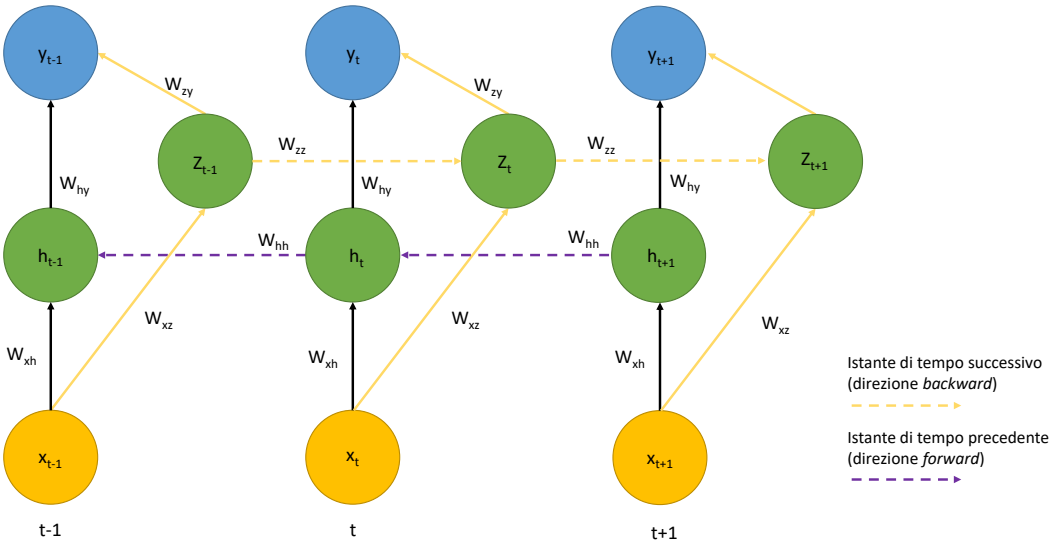


Figura 3.8. Architettura di una Rete Neurale Ricorrente Bidirezionale.

3.4.1 Backpropagation nel tempo

Nel caso delle reti neurali ricorrenti, l'algoritmo di apprendimento utilizzato è la *backpropagation* nel tempo. Si tratta sempre di un algoritmo di *training*

supervisionato, che ha lo scopo di aggiornare i pesi delle connessioni tra i neuroni per minimizzare l'errore tra i valori di uscita desiderati e quelli predetti. Come osservato in precedenza, possiamo considerare una RNN come t copie di una stessa rete ripetute nel tempo, che comunicano tra loro condividendosi i pesi delle connessioni. Di conseguenza, possiamo modificare l'algoritmo di *training* per calcolare l'errore per ogni istante di tempo t, come segue:

$$E_{totale}(t_0, t_n) = \sum_{t=t_0}^{t_n} E(t) \tag{3.23}$$

dove t_0 e t_n sono, rispettivamente, l'istante t di tempo in cui inizia e finisce l'apprendimento della nostra rete neurale.

Per quanto riguarda la modifica da applicare ai pesi, possiamo modificare l'equazione 3.9 considerando ogni istante di tempo t:

$$\Delta w_{jk} = -\eta \frac{\partial E_{totale}(t_0, t_n)}{\partial w_{jk}} = -\eta \sum_{t=t_0}^{t_n} \frac{\partial E(t)}{\partial w_{jk}} \tag{3.24}$$

Una soluzione al problema del *vanishing gradient* accennato nel paragrafo precedente, consiste nell'utilizzo di un'ulteriore algoritmo di *backpropagation* modificato, chiamato *backpropagation* nel tempo troncato. In questo algoritmo viene definito un numero massimo di istanti di tempo, oltre il quale l'errore non si può più propagare, andando, però, a sacrificare le dipendenze temporali a lungo termine.

3.4.2 Long-short Term Memory

Una possibile soluzione al problema del *vanishing gradient* che non vada ad intaccare le dipendenze temporali a lungo termine, è l'utilizzo di una rete *Long-Short Term Memory* (LSTM). Una LSTM è un miglioramento di una RNN standard, in cui vengono aggiunte delle celle di memoria per salvare informazioni che riguardano dipendenze temporali a lungo termine. Al posto del tradizionale neurone usato negli strati nascosti di una RNN, viene utilizzata un'unità di memoria. Quest'ultima, mostrata nella fig. 3.9, introduce il meccanismo di *gating* per controllare i processi di memorizzazione: è formata da tre unità moltiplicative, definite *gate* appunto, che stabiliscono se un'informazione debba essere salvata o rimossa tramite "cancelli" che si aprono o si chiudono.

Più nel dettaglio, ogni unità di memoria è formata da:

- *Nodo di input* $\boldsymbol{g_t}$: questa unità riceve in ingresso i valori x_t dallo strato di input al tempo t e h_{t-1} dallo strato nascosto al tempo $t-1$. Calcola la loro somma pesata e applica la funzione di attivazione *tanh* come segue:

  $$\boldsymbol{g}_t = \phi(W_{xg}\boldsymbol{x}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \tag{3.25}$$

  dove ϕ è la funzione tangente iperbolica e W la matrice dei pesi di connessione tra due unità.

- *Input gate* $\boldsymbol{i_t}$: unità che decide se i valori di ingresso verranno memorizzati o meno nell'unità di memoria. Riceve in ingresso sempre i valori x_t e h_{t-1},

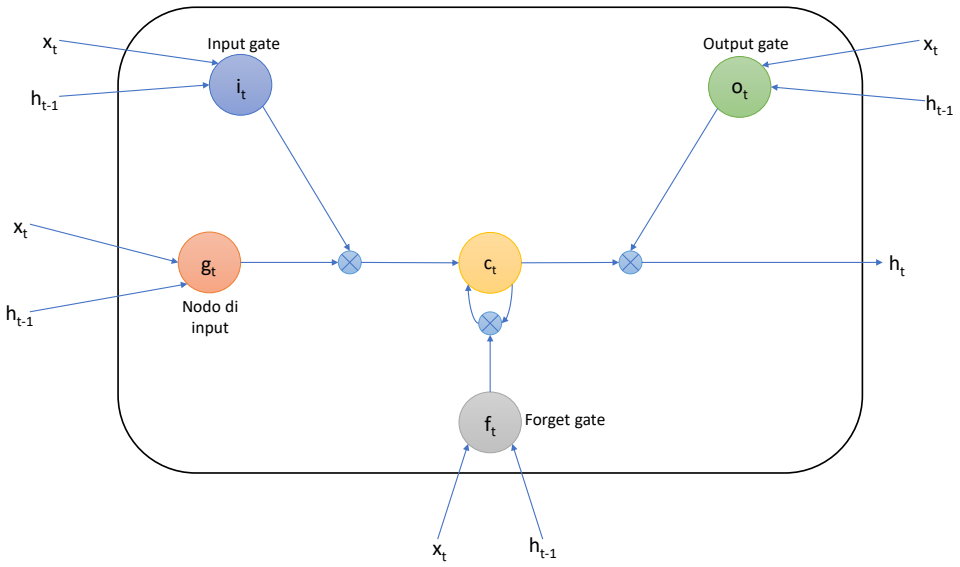


Figura 3.9. Unità di memoria di una LSTM standard.

ma sulla loro somma pesata applica una funzione sigmoidea e moltiplica il risultato con il valore del nodo di input. Se il valore risultante è 1 i valori di ingresso saranno memorizzati nella memoria. La formula è la seguente:

$$\boldsymbol{i}_t = \sigma(W_{xi}\boldsymbol{x}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{3.26}$$

dove σ è la funzione sigmoidea.

- *Forget gate* $\boldsymbol{f_t}$: questa unità, come suggerisce il nome, determina quali informazioni dell'unità di memoria devono essere dimenticate. Riceve in ingresso i valori x_t e h_{t-1} rispettivamente dallo strato di input e dallo strato nascosto e ne calcola la somma pesata, alla quale applica la funzione sigmoidea. Il risultato viene moltiplicato per ogni valore dello stato interno dell'istante di tempo precedente c_{t-1}. Se il risultato è 1 viene salvata l'informazione, altrimenti viene eliminata.

$$\boldsymbol{f}_t = \sigma(W_{xf}\boldsymbol{x}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{3.27}$$

- *Stato interno* $\boldsymbol{c_t}$: è il nucleo di ogni unità di memoria. Contiene un arco ricorsivo avente un peso unitario grazie al quale è possibile evitare il problema del *vanishing gradient*. Questo perché l'arco tra istanti di tempo adiacenti ha un peso costante.

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \phi(W_{xc}\boldsymbol{x}_t + W_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \tag{3.28}$$

- *Output Gate* $\boldsymbol{o_t}$: stabilisce quale sarà il valore di output dell'unità di memoria. Riceve in input x_t e h_{t-1}, applica una funzione sigmoidea sulla loro somma pesata e moltiplica il valore risultante con il valore c_t, su cui di solito viene

prima applicata la funzione *tanh*, in modo da limitare il valore di uscita delle celle di memoria allo stesso intervallo di valori.

$$\boldsymbol{o}_t = \sigma(W_{xo}\boldsymbol{x}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{3.29}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t) \tag{3.30}$$

dove $\boldsymbol{h}_t$ è lo stato nascosto al tempo t.

3.5 Reti neurali convoluzionali

Una Rete Neurale Convoluzionale (*Convolutional Neural Network* - CNN) è una variante della classica rete neurale *feed-forward* che si ispira ai processi biologici; infatti il modo in cui i neuroni sono connessi tra di loro ricorda l'organizzazione della corteccia visiva animale. In questa rete i neuroni rispondono solo agli stimoli di una regione ristretta del campo visivo, nota come campo ricettivo. In seguito, i campi ricettivi di neuroni differenti si sovrappongono parzialmente in modo da coprire l'intero campo visivo. Questi processi biologici possono essere implementati tramite un'operazione di convoluzione che permette di replicare il funzionamento della corteccia visiva animale.

Una CNN, mostrata nella fig. 3.10, è costituita da quattro operazioni principali (convoluzione, non linearità, *pooling* e classificazione) implementate tramite i seguenti strati:

- *Strato di convoluzione*: questo strato effettua l'operazione di **convoluzione** che permette di estrarre delle *feature map* da un'immagine data in ingresso. Durante questa operazione vengono mantenute le informazioni spaziali sui pixel che costituiscono l'immagine e la rete apprende le *feature* grazie all'utilizzo di filtri. Un filtro è una piccola matrice che scorre su tutta l'immagine di input e calcola il prodotto scalare tra i pixel generando una *feature map*.

- *Strato ReLU*: per convenzione viene applicato dopo uno strato di convoluzione con lo scopo di introdurre **non linearità** al sistema. Lo strato applica una funzione di attivazione ReLU che cambia tutti i valori di input negativi ricevuti a 0.

- *Strato di pooling*: strato utilizzato per ridurre la dimensione delle *feature map*. Questo è possibile attraverso vari metodi: sommando tutti i valori all'interno di un filtro oppure calcolandone il massimo o la media.

- *Strato fully connected*: strato formato da un percettrone multistrato avente una particolare funzione di attivazione. Riceve le *feature* dagli strati di convoluzione e di *pooling* e le utilizza per la **classificazione** dell'immagine data in ingresso alla rete.

- *Strato di perdita*: di solito costituisce l'ultimo strato di una CNN e serve per calcolare la differenza tra i valori desiderati e quelli predetti, durante la fase di *training*.

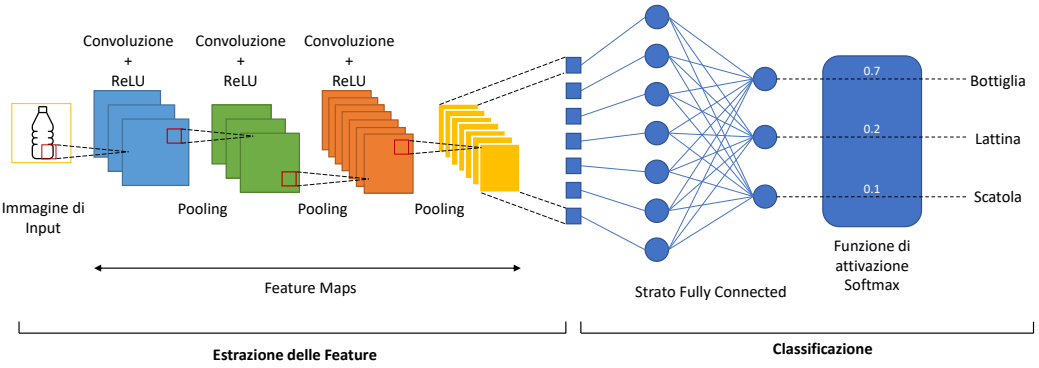


Figura 3.10. Generica architettura di una rete neurale convoluzionale.

Le reti convoluzionali sono particolarmente utili per problemi che riguardano l'elaborazione di immagini con lo scopo di riconoscere oggetti, scene e volti. Inoltre, possono essere utilizzate anche per classificare dati diversi da immagini, come segnali e audio.

3.6 Autoencoder

Un autoencoder è un tipo di rete neurale utilizzata per apprendere una rappresentazione compressa ed efficiente dei dati di input. Gli autoencoder sono costituiti da due componenti principali: un codificatore (*encoder*) e un decodificatore (*decoder*), come mostrato nella fig. 3.11. Il primo elabora i dati di input e li mappa in uno spazio latente di dimensione inferiore, che rappresenta una versione compressa dell'input. Il secondo prende la rappresentazione latente e la utilizza per ricostruire l'input originale. L'obiettivo dell'addestramento di un autoencoder è di apprendere un processo di codifica-decodifica che ricostruisca i dati di ingresso nel modo più accurato possibile. Per raggiungere l'obiettivo l'autoencoder viene addestrato utilizzando una funzione di *loss* che misura la differenza tra l'input ricostruito e quello originale; l'*encoder* e il *decoder* sono addestrati simultaneamente per minimizzare questa *loss*.

Formalmente, il codificatore e il decodificatore possono essere espressi, rispettivamente, tramite due funzioni E_ϕ e D_ψ, tali che:

$$\begin{aligned} E_\phi &: X \to Z \\ D\psi &: Z \to X \end{aligned} \tag{3.31}$$

dove X indica l'insieme dei messaggi decodificati, mentre Z l'insieme dei messaggi codificati.

Per ogni $x \in X$, con $z = E_\phi(x)$ si indica il codice, meglio conosciuto come rappresentazione latente o vettore latente. Al contrario, per ogni $z \in Z$ si scrive $x' = D_\psi(z)$ per riferirsi al messaggio decodificato.

Nel caso più semplice, sia l'*encoder* che il *decoder* sono costituiti da percettroni multistrato (MLP). Se per esempio consideriamo un Codificatore MLP E_ϕ a uno strato, la fase di codifica può essere definita come:

$$E_\phi = z = \sigma(Wx + b) \tag{3.32}$$

dove σ indica la funzione di attivazione, come la funzione sigmoidea o la ReLU, W è la matrice dei pesi e b è il vettore di *bias*.

La fase di decodifica di D_ψ si può definire come:

$$D_\psi = x' = \sigma'(W'z + b') \tag{3.33}$$

dove σ' W' e b' indicano, rispettivamente, la funzione di attivazione, i pesi e il *bias* del *decoder*, i quali possono anche differire da quelli dell'*encoder*.

Durante l'addestramento si cerca di minimizzare l'errore tra x e x' tramite la *loss function* che, nel caso di una MSE, viene espressa come:

$$\mathcal{L}(x, x') = ||x - x'||^2 = ||x - \sigma'(W'(\sigma(Wx + b)) + b'||^2 \tag{3.34}$$

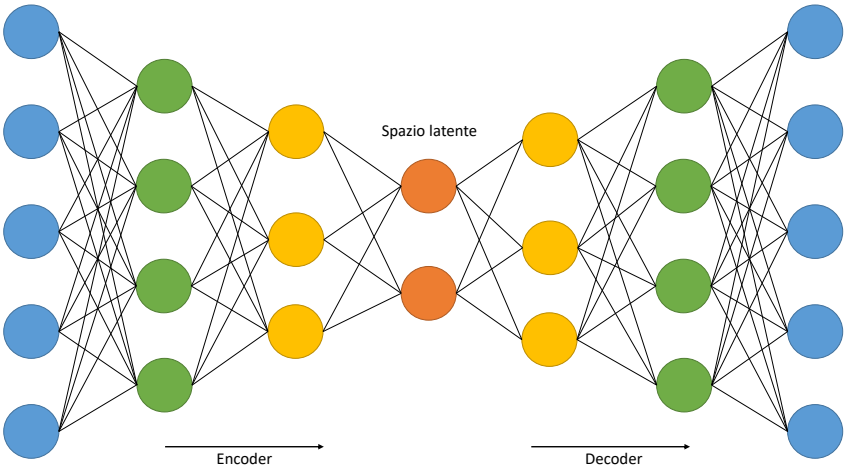


Figura 3.11. Tipica struttura di un autoencoder.

3.6.1 Tipologie di Autoencoder

In base alla particolare architettura di rete utilizzata ed al tipo di problema che si vuole risolvere, solitamente gli autoencoder si suddividono in:

- *Vanilla autoencoders*: sono il tipo più semplice di autoencoder e sono costituiti da un singolo strato nascosto tra l'encoder e il decoder. Solitamente vengono utilizzati per ridurre le dimensioni dell'input e per l'estrazione delle feature.
- *Denoising autoencoders*: tipo di autoencoder addestrato per ricostruire una versione pulita dei dati di input danneggiati o rumorosi. Vengono utilizzati per attività di *denoising* di dati e *anomaly detection*.
- *Variational autoencoders*: questa tipologia di autoencoder è in grado di generare nuovi sample di dati simili a un determinato dataset di partenza. Si basano sull'idea di approssimare una distribuzione complessa utilizzando una distribuzione più semplice, come la Gaussiana.

- *Convolutional autoencoders*: gli autoencoder convoluzionali sono progettati appositamente per elaborare dei dati aventi una struttura a griglia, come le immagini. Utilizzano gli strati delle reti neurali convoluzionali sia nell'encoder che nel decoder per processare i dati di input.

- *Recurrent autoencoders*: questo tipo di autoencoder viene utilizzato per elaborare dati sequenziali, come *time series*, *natural language text* o *speech signal*. Utilizzano i *layer* delle rete neurali ricorrenti (RNN) nel codificatore e nel decodificatore per elaborare l'input, i quali consentono loro di acquisire le dipendenze temporali nei dati.

In conclusione gli autoencoder sono un potente strumento per l'apprendimento di rappresentazioni efficienti di dati, utilizzato per un'ampia gamma di applicazioni; sono particolarmente utili per lavorare con dataset voluminosi e complessi e possono essere combinati con altri tipi di reti neurali per risolvere compiti più complicati.

Capitolo 4

Sintesi di Oggetti tramite Segnali Wi-Fi

Questo capitolo è strutturato come segue: nel paragrafo 4.1 viene introdotta l'architettura e il metodo di deep learning utilizzati per la sintesi di oggetti; nel paragrafo 4.2 viene spiegato quali sono le feature ed il loro processo di estrazione; infine vengono descritte più in dettaglio l'architettura §4.3 e la strategia di apprendimento §4.4 della rete neurale utilizzata.

4.1 Metodo

Per sintetizzare la immagini di oggetti tramite i segnali Wi-Fi, è stata progettata una rete neurale a due rami paralleli, mostrata nella fig. 4.1. L'architettura proposta simula un rapporto di tipo maestro-studente, in cui il primo supervisiona la fase di addestramento del secondo, con lo scopo di trasferire l'informazione visiva ottenuta dalle immagini all'ampiezza estratta dai segnali Wi-Fi raccolti.

Il maestro è costituito da un autoencoder convoluzionale 2D (CNN-AE), quindi sia il codificatore che il decodificatore sono reti CNN. Lo studente, invece, è un autoencoder ibrido (H-AE), che condivide con il maestro la componente di *decoding*, mentre come encoder utilizza una rete LSTM. Quest'ultima viene utilizzata per gestire l'aspetto temporale delle ampiezze in modo da ottenere una rappresentazione latente del segnale, che verrà poi trasferita nel dominio visivo tramite il decoder convoluzionale. In questo modo, a partire dai segnali, lo studente impara a generare correttamente le immagini, grazie alla supervisione del maestro (per maggiori dettagli sull'architettura si veda il paragrafo 4.3).

Il *training set* utilizzato durante la fase di addestramento è costituito da un insieme di coppie $< Immagine,\ Ampiezze >$. L'immagine è stata ottenuta da una foto di diversi oggetti di comune utilizzo a cui è stato rimosso lo sfondo. Le ampiezze vengono estratte dal CSI dei segnali Wi-Fi, i quali sono influenzati dalla presenza del tipo di oggetto mostrato dall'immagine.

Prima di essere date in input alla rete, le ampiezze vengono sanificate e, successivamente, convertite dal dominio della frequenza al dominio del tempo, tramite la trasformata inversa di Fourier. La conversione viene eseguita in quanto la polarizzazione è una caratteristica che descrive la direzione di oscillazione del vettore

campo elettrico nel dominio temporale. In altre parole, se esiste una legge (funzione) matematica che descrive la variazione di questa direzione nel tempo, l'onda si dice polarizzata [25].

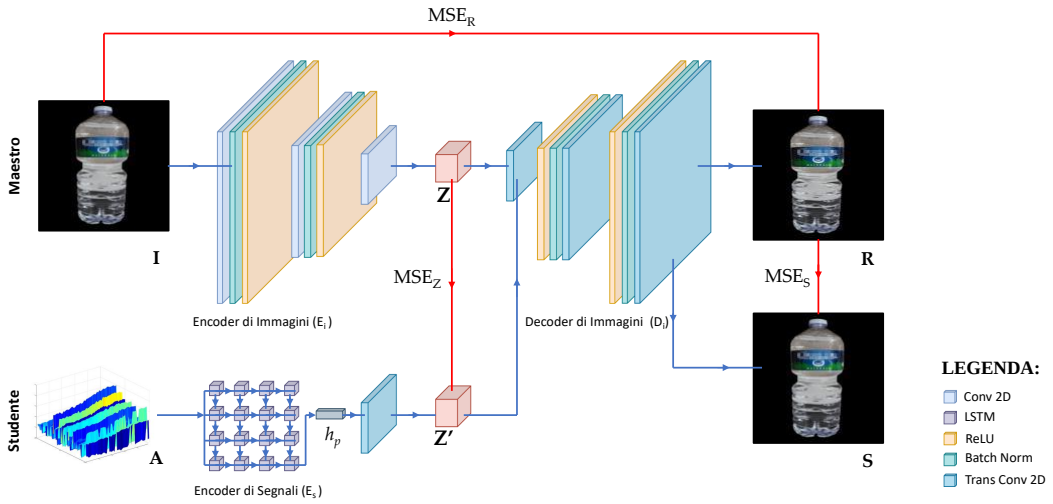


Figura 4.1. Architettura del modello proposto per la sintesi di oggetti tramite segnali Wi-Fi. Data una coppia $< Immagine, Ampiezze >$ come input, l'informazione visiva estratta dalle immagini (Z) viene trasferita alle feature ottenute dalle ampiezze (Z'). In seguito questa informazione viene convertita dal dominio radio al dominio visivo attraverso il decoder di immagini (D_i). Le frecce rosse indicano la supervisione del maestro sullo studente.

Una volta che la rete è stata addestrata, le immagini vengono sintetizzate utilizzando solamente il ramo studente e le ampiezze dei segnali.

4.2 Estrazione delle feature

La sintesi di immagini necessita di caratteristiche misurabili e specifiche dell'oggetto in osservazione, definite *feature*, che non contengano informazioni ridondanti. La *feature* utilizzata in questo lavoro corrisponde all'ampiezza dei segnali Wi-Fi polarizzati in maniera differente.

4.2.1 Ampiezza

Per ottenere delle feature significative nel dominio radio è stata estratta l'ampiezza nel tempo dal CSI dei segnali Wi-Fi.
In una configurazione MIMO-OFDM, date le antenne $\theta \in \Theta$ e $\gamma \in \Gamma$, rispettivamente, di trasmissione e di ricezione, per ogni sottoportante $k \in K$, il CSI è una matrice di numeri complessi di dimensioni $\Theta \times \Gamma \times K$ (mostrata nell'equazione 2.21 del paragrafo 2.5). Ciascuna entry della matrice rappresenta la risposta in frequenza del canale wireless (CFR) per ogni pacchetto $p \in P$ scambiato tra le antenne TX e RX, ed è definita come:

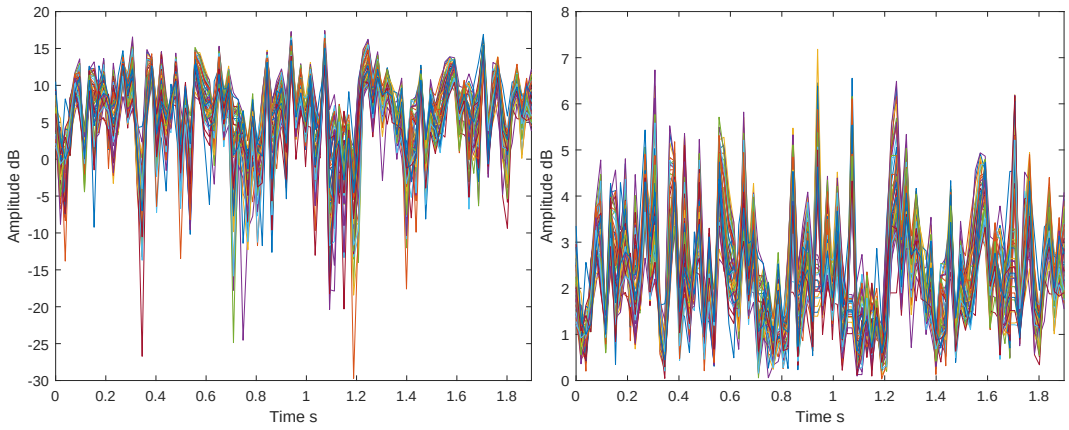
$$H(f)_k^{\theta,\gamma} = |H(f)_k^{\theta,\gamma}|e^{j\angle H(f)_k^{\theta,\gamma}} \tag{4.1}$$

dove $|H(f)_k^{\theta,\gamma}|$ corrisponde all'ampiezza del segnale.
Applicando la trasformata inversa di Fourier (IFFT) all'equazione 4.1, si ottiene la corrispondente risposta all'impulso (CIR) nel dominio temporale:

$$H(t)_k^{\theta,\gamma} = IFFT(H(f)_k^{\theta,\gamma}) \tag{4.2}$$

In questo modo le ampiezze vengono convertite dal dominio della frequenza al dominio del tempo.

Per valutare il metodo proposto, il dataset è stato raccolto con $\Theta = 1$ e $\Gamma = 1$ antenne, sfruttando un canale da 20MHz con $K = 50$ sottoportanti. Successivamente l'ampiezza è stata estratta da $H(t)_k^{\theta,\gamma}$ e filtrata secondo la procedura di sanificazione, descritta nel paragrafo 2.6.1, per ridurre il rumore causato dalle condizioni ambientali e rimuovere eventuali valori anomali. Quest'ultimi costituiscono informazioni irrilevanti da processare che possono influenzare l'estrazione di *feature* utili e di conseguenza vanno rimossi. La figura 4.2 mostra il grafico dell'ampiezza nel tempo prima e dopo il processo di sanificazione.



(a) **(b)**

Figura 4.2. Esempio di ampiezza nel tempo estratta dal CSI per 100 pacchetti di dati. In (a) l'ampiezza grezza, in (b) l'ampiezza sanificata.

4.2.2 Polarizzazione

In questo lavoro viene sfruttata una proprietà specifica dei segnali wireless: la polarizzazione delle onde riflesse da un oggetto.
Quando un'onda incidente polarizzata riflette su oggetti differenti, si comporta in maniera diversa in base al tipo di materiale e alla *texture* della loro superficie. Questo perché, come mostrato nel paragrafo 2.7, la polarizzazione è una proprietà che descrive la direzione di oscillazione del campo elettrico e quando un segnale riflette su un oggetto il risultato è un cambiamento nella direzione del campo elettrico e, di conseguenza, nel tipo di polarizzazione.

Per esempio, le superfici lisce riflettono le onde lasciando la polarizzazione intatta; il metallo assorbe e riflette l'onda, modificando il pattern di polarizzazione; le superfici ruvide, invece, disperdono le onde e causano un cambiamento della polarizzazione (fig. 4.3). È stato inoltre dimostrato [29], che si può dedurre il tipo degli oggetti misurando la polarizzazione dei segnali wireless che riflettono su di essi.

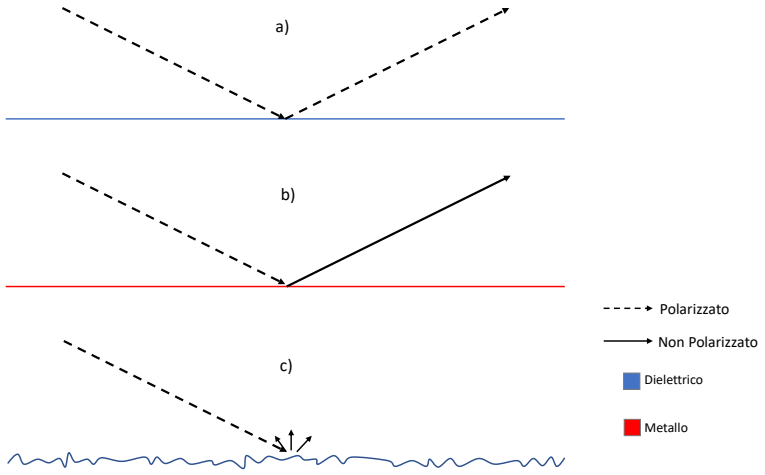


Figura 4.3. Riflessione del segnale polarizzato su una superficie dielettrica liscia (a), una superficie metallica liscia (b), una superficie ruvida non metallica (c).

Nel presente lavoro, la valutazione della polarizzazione è possibile utilizzando delle antenne che ci permettono di ricevere polarizzazioni differenti. In particolare, il segnale Wi-Fi trasmesso da un'antenna TX polarizzata orizzontalmente, viene misurato da un'antenna ricevente polarizzata prima orizzontalmente e poi verticalmente. La polarizzazione delle antenne viene determinata dal loro orientamento, come mostrato nel paragrafo 2.7.3. Successivamente, le due misurazioni vengono combinate in maniera differente per valutare il modello proposto.

4.3 Architettura di rete

L'architettura di rete sfrutta i due rami paralleli per eseguire il trasferimento di dominio ovvero per passare dal dominio radio dei segnali al dominio visivo delle immagini.

Il ramo superiore è costituito da un autoencoder convoluzionale (CNN-AE) che si comporta da maestro. In particolare, l'encoder E_i è formato da tre strati convoluzionali 2D e ciascuno, eccetto l'ultimo, è seguito da una *batch normalization* e una funzione di attivazione ReLU. L'encoder D_i, invece, presenta una struttura inversa rispetto a quella dell'encoder E_i, con strati trasposti convoluzionali 2D al posto di strati convoluzionali semplici e con una funzione di attivazione tangente iperbolica $Tanh$ che segue l'ultimo strato di trasposizione.

Riguardo il ramo inferiore, esso agisce da studente ed è composto da un autoencoder ibrido (H-AE) in grado di gestire le informazioni nel dominio radio. La

componente di encoding, infatti, è un codificatore di segnali E_s formato da uno strato LSTM con p unità, una per pacchetto, e una convoluzione 2D trasposta applicata all'output dell'ultima unità, per modificare le sue dimensioni affinché possa essere data in input al decodificatore. Quest'ultimo, è condiviso con il decoder di immagini D_i del ramo maestro.

La LSTM è stata utilizzata poiché, come dimostrato in [3], è in grado di apprendere *feature* radio a bassa dimensione da una sequenza di ampiezze estratte dal CSI di dati wireless. Nell'architettura proposta, tali *feature* vengono tradotte nel dominio visivo dalla componente D_i per sintetizzare le immagini.

Formalmente, data una sequenza di ampiezze $A = \{a_{1,1}, a_{1,2}, \ldots, a_{P,K}\}$, con $k \in K$ sottoportanti, per ciascun pacchetto $p \in P$ la corrispondente unità $LSTM_p$ può essere definita come:

$$
\begin{aligned}
i_p &= \sigma(W_{ai} a_p + U_{hi} h_{p-1} + \Psi_{ci} c_{p-1} + b_i) \\
f_p &= \sigma(W_{af} a_p + U_{hf} h_{p-1} + \Psi_{cf} c_{p-1} + b_f) \\
o_p &= \sigma(W_{ao} a_p + U_{ho} h_{p-1} + \Psi_{co} c_{p-1} + b_o) \\
c_p &= f_p \odot c_{p-1} + i_p \odot tanh(W_{ac} a_p + U_{hc}\, h_{p-1} + b_c) \\
h_p &= o_p \odot tanh(c_p)
\end{aligned}
$$

dove a, h e c indicano, in ordine, l'input, l'*hidden state* e lo stato interno (*cell state*); i, f e o sono, rispettivamente, l'*input gate*, il *forget gate* e *l'output gate*; W, U, Ψ sono le matrici dei pesi per i corrispettivi *gates*, *hidden states* e connessioni *peep-hole*[1]; infine b indica i vettori di bias che vengono sommati ad ogni gate o allo stato interno.

Si noti che l'encoder E_s riceve in input solamente il vettore h_p proveniente dall'ultima unità LSTM, il quale cattura una rappresentazione latente dell'intera sequenza di input. In seguito, una convoluzione 2D trasposta viene applicata al vettore h_p per prepararlo alla traduzione nel dominio della visione. Il risultato è un nuovo spazio latente Z' che permette alla componente D_i di generare immagini.

4.4 Apprendimento supervisionato

Per addestrare il modello proposto vengono utilizzate N coppie $< I, A >_n$, dove I è l'immagine dell'oggetto, mentre A è l'insieme delle ampiezze estratte dal CSI del segnale Wi-Fi. Per ogni coppia $n \in N$, il codificatore E_i prende in input l'immagine I e crea una codifica ridotta Z chiamata spazio latente. In seguito il decodificatore D_i utilizza Z per cercare di riprodurre l'immagine originale I, creando una nuova immagine R. Formalmente, possiamo definire queste operazioni come:

$$
\begin{aligned}
Z &= E_i(I) \\
R &= D_i(Z)
\end{aligned}
\tag{4.3}
$$

[1]Le connessioni *peep-hole* sono una modifica dell'architettura LSTM base, che permettono ai *gates* di non dipendere solo dal precedente stato nascosto h_{t-1}, ma anche dal precedente stato interno c_{t-1}, aggiungendo un ulteriore termine nelle equazioni del *gate*.

La funzione di loss tra l'immagine originale I e quella ricostruita R viene calcolata come segue:

$$MSE_R = \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} (I_{i,j} - R_{i,j})^2 \tag{4.4}$$

dove W e H corrispondono alla larghezza e all'altezza delle immagini; $I_{i,j}$ e $R_{i,j}$ indicano il valore del pixel in posizione (i, j), rispettivamente, dell'immagine originale e di quella ricostruita.
Nel ramo inferiore, invece, l'encoder E_s riceve in ingresso le ampiezze A espresse nel dominio temporale e calcola una rappresentazione latente h_p. Si ricorda che prima di essere dato in input al decoder, al vettore h_p viene applicato uno strato di convoluzione 2D trasposto, in modo da generare una nuova rappresentazione Z' avente la stessa dimensione di Z.
Infine, il decoder condiviso D_i prende in input Z' e genera una immagine S, passando così dal dominio radio al dominio della visione. Formalmente, possiamo esprimere i passaggi precedenti come segue:

$$\begin{aligned} h_p &= E_s(I) \\ Z' &= 2DTransConv(h_p) \\ S &= D_i(Z') \end{aligned} \tag{4.5}$$

Il trasferimento di dominio è possibile collegando lo spazio latente delle immagini Z a quello delle ampiezze Z'. Questo avviene calcolando per l'encoder E_s la seguente funzione:

$$MSE_Z = \frac{1}{|Z|} \sum_{i=1}^{|Z|} (Z_i - Z'_i)^2 \tag{4.6}$$

dove $|Z|$ indica la dimensione di Z.

Inoltre, per migliorare ulteriormente la capacità di sintesi dello studente, viene calcolato l'MSE anche tra le immagini S generate dallo studente e quelle R ricostruite dal maestro. Formalmente, questo errore è calcolato come segue:

$$MSE_S = \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} (R_{i,j} - S_{i,j})^2 \tag{4.7}$$

dove WH corrisponde al numero di pixel delle immagini e con gli indici (i, j) si indica sempre la posizione del pixel nelle immagini.

L'obiettivo finale della rete è di minimizzare la seguente funzione di *loss* totale:

$$\underset{Totale}{\mathcal{L}} = MSE_R + MSE_Z + MSE_S \tag{4.8}$$

Capitolo 5

Test e Valutazioni

Il capitolo è strutturato come segue: nel paragrafo 5.1 viene descritta la procedura di raccolta dei segnali che compongono il dataset; nel paragrafo 5.2 vengono dati maggiori dettagli riguardo la configurazione degli esperimenti e l'implementazione della rete neurale; infine, nel paragrafo 5.3 vengono mostrati i risultati quantitativi e qualitativi dei vari test eseguiti.

5.1 Dataset

Il dataset utilizzato per valutare il metodo proposto è stato ottenuto attraverso una serie di raccolte. In particolare, sono stati catturati i segnali Wi-Fi per 4 oggetti differenti: una bomboletta spray (bomboletta), una bottiglia d'acqua (bottiglia), una confezione di latte (latte) e una lattina di una bibita (lattina).
La bomboletta è in acciaio, 40 ml di volume, cilindrica, con della schiuma da barba sotto pressione all'interno; la bottiglia è di plastica, 1,5 l di volume, rettangolare con una superficie irregolare e contiene dell'acqua naturale; la confezione di latte è di carta, rettangolare, 1 l di volume, con del latte al suo interno; la lattina è fatta di alluminio, 33 cl di volume, cilindrica e contiene una bibita gassata. Rispetto allo studio presentato in [29], sono stati aggiunti e testati tre materiali differenti: acciaio, plastica e carta.

Gli oggetti sono stati collocati tra un'antenna trasmittente (TX) e una ricevente (RX), senza nessun altro oggetto tra di esse, e poste a 80cm di distanza tra loro. In totale sono stati acquisiti 800 segnali: 200 per ogni oggetto, di cui 100 con l'antenna RX polarizzata orizzontalmente, fig.5.1a, e, successivamente, altre 100 con l'antenna RX polarizzata verticalmente, fig.5.1b. La polarizzazione dell'antenna trasmittente è rimasta orizzontale durante tutto l'esperimento. Ciascuna acquisizione è durata 10 secondi, permettendo lo scambio di circa 600 pacchetti, attraverso il canale da 20 MHz di un collegamento Wi-Fi a 2.4 GHz.
Le raccolte sono avvenute sempre nella stessa stanza, senza impiegare nessun meccanismo di schermatura sull'arredamento. In questo modo si è potuto replicare un ambiente Wi-Fi reale, in cui diverse connessioni wireless si propagano influenzandosi a vicenda. Una foto del setting dell'esperimento è mostrata nella figura 5.2.

(a) (b)

Figura 5.1. Tipi differenti di polarizzazione delle antenne: in (a) l'antenna RX polarizzata orizzontalmente, mentre in (b) polarizzata verticalmente.

Figura 5.2. Tipico setting utilizzato durante la raccolta dei segnali Wi-Fi.

5.2 Dettagli di implementazione

In merito alla configurazione sperimentale, sono stati utilizzati due access point (AP), uno smartphone per scattare le foto degli oggetti e due laptop. Entrambi gli AP sono un ESP32-WROOM-32: un microcontrollore a basso costo e a bassa potenza con Wi-Fi e bluetooth integrato. Uno viene configurato come AP, cioè antenna trasmittente, mentre l'altro come Active Station (STA), ovvero funziona da antenna ricevente. I dispositivi ESP32 salvano il CSI del segnale, utilizzando il tool presentato in [10], come file CSV. Successivamente sono stati pre-processati tramite il software Matlab R2022b per estrarre dal CSI l'ampiezza dei segnali che, dopo essere stata sanificata e convertita nel dominio del tempo, è stata utilizzata in

coppia con le foto per sintetizzare le immagini degli oggetti. La fotocamera dello smartphone è da 50 Mp, con una risoluzione di 8165 x6124 pixel. Il primo laptop, connesso ad AP, è un laptop Acer Aspire 3, mentre il secondo laptop, connesso a STA, è un HP Pavilion Gaming 16. L'ampiezza del canale wireless è di 20MHz con 50 sottoportanti. Prima di eseguire la raccolta dei segnali, descritta nel paragrafo precedente, sono stati eseguiti diversi studi di ablazione per valutare correttamente il metodo proposto.

Per quanto riguarda i parametri delle reti neurali, il dataset è stato suddiviso in *training set*, l'80% dei dati, e validation set, il restante 20%. La rete è stata addestrata per 200 epoche usando l'Adam optimizer con un learning rate di 0.0002 e con i valori dei parametri $\beta 1$ e $\beta 2$ impostati, rispettivamente, a 0.5 e 0.999. Il *batch size* è stato settato a 16. La rete è stata implementata tramite il framework Pytorch e la sua libreria TorchVision, mentre tutti i test sono stati eseguiti su Google Colab che ha permesso di utilizzare un ambiente compatibile con CUDA.

5.3 Valutazione della sintesi di oggetti

In questo paragrafo vengono mostrati i vari test eseguiti per valutare le prestazioni dell'architettura proposta. Gli esperimenti si dividono in due fasi principali: la prima si concentra sulla polarizzazione dei segnali mentre la seconda sul numero di pacchetti scambiati tra le antenne.

Per valutare la qualità delle immagini vengono utilizzati due metriche quantitative: l'Errore Quadratico Medio (*Mean Square Error - MSE*) e l'Indice di Similarità Strutturale (*Structural Similarity Index - SSIM*). Mentre il primo si limita a confrontare i pixel tra l'immagine originale e quella sintetizzata, il secondo cerca di quantificare la somiglianza tra le due in termini di luminosità, contrasto e struttura. L'SSIM calcola un valore che varia da 0 a 1, dove un valore più vicino a 1 indica una maggiore similarità tra le immagini. Queste ultime, prima di essere date in input alla rete, sono state ridimensionate a 128×128 pixel (fig. 5.3).

(a) bomboletta **(b)** bottiglia **(c)** latte **(d)** lattina

Figura 5.3. Immagini originali dei quattro oggetti utilizzati per gli esperimenti.

Le dimensioni scelte hanno fornito un buon compromesso tra qualità visiva e prestazioni della rete. In seguito sono state convertite in tensori e normalizzate; la conversione è necessaria poiché in PyTorch ogni oggetto è un tensore, ovvero, una matrice multidimensionale di numeri. La normalizzazione è un processo che converte i valori dei pixel in un range specifico, [0, 1] o [-1, 1], che di solito coincide con quello delle funzioni di attivazione; normalizzando ogni canale dell'immagine si ottimizzano

le prestazioni del modello durante la fase di addestramento, aiutandolo a prevenire l'*overfitting* e a migliorare la capacità di generalizzazione.

5.3.1 Test 1: Tipo di polarizzazione

La prima fase dei test si è concentrata sulla valutazione della capacità di sintesi utilizzando polarizzazioni differenti. Si ricorda che i segnali sono stati raccolti cambiando l'orientamento dell'antenna RX, mentre l'antenna TX è rimasta polarizzata in maniera orizzontale per tutti i test. Il dataset di partenza, descritto nel paragrafo 5.1, è stato suddiviso in 4 sottoinsiemi:

- ***H***: contiene solo i segnali ricevuti con l'antenna RX polarizzata orizzontalmente;
- ***V***: include solo i segnali con RX polarizzata verticalmente;
- ***S***: contiene la somma dei segnali tra l'insieme H e l'insieme V;
- ***R***: comprende il rapporto tra i segnali contenuti in H e V.

Per ogni sottoinsieme di input POL, la rete è stata addestrata per 200 epoche, utilizzando solo 50 pacchetti. In questo modo si è tenuto basso il tempo di esecuzione, focalizzandosi maggiormente sulla polarizzazione. Il risultato quantitativo è riportato nella tabella 5.1.

Tabella 5.1. Risultati quantitativi, considerando differenti tipi di polarizzazione, con 50 pacchetti.

$\#Pacchetti$	POL	$MSE \downarrow$	$SSIM \uparrow$
50	H	0.038	0.569
	V	**0.028**	**0.672**
	S	0.044	0.537
	R	0.052	0.560

Come si può osservare, l'MSE è vicino a zero per tutti i tipi di polarizzazione, dimostrando che il ramo studente è in grado di sintetizzare gli oggetti. Tuttavia, con $POL = H$ e $POL = R$, alcune immagini mostrano un effetto "fantasma", che invece non è presente utilizzando le altre due polarizzazioni. Tale effetto può essere osservato nella figura 5.4.

Riguardo la metrica SSIM, il valore più elevato viene raggiunto da V. Anche dal punto di vista qualitativo, come si può vedere dalla fig. 5.4, i risultati migliori si ottengono utilizzando solamente i segnali polarizzati verticalmente. Le immagini risultano infatti più nitide e la sagoma degli oggetti possiede una definizione maggiore.

Figura 5.4. Confronto qualitativo basato sull'insieme POL, utilizzando 50 pacchetti. Nel caso di H e R, le immagini della bottiglia mostrano un effetto "fantasma", con la sagoma della bomboletta che si sovrappone alla bottiglia. Tale effetto non è presente per V e S.

Per concludere la prima fase degli esperimenti, sono stati effettuati gli stessi test ma utilizzando 100 pacchetti. I risultati quantitativi sono presentati nella tabella 5.2 e sono coerenti con quelli del caso precedente. Anche dal punto di vista qualitativo (fig. 5.5) le immagini generate utilizzando $POL = V$ mostrano una qualità dell'immagine superiore. Utilizzando $POL = R$ in alcuni casi la bomboletta viene scambiata con la lattina, mentre con $POL = H$ la lattina viene scambiata con il latte.

Tabella 5.2. Risultati quantitativi, considerando differenti tipi di polarizzazione, utilizzando 100 pacchetti.

$\#Pacchetti$	POL	$MSE \downarrow$	$SSIM \uparrow$
100	H	0.051	0.503
	$\boldsymbol{V}$	**0.030**	**0.628**
	S	0.033	0.586
	R	0.064	0.509

Figura 5.5. La figura mostra i risultati qualitativi basati sul tipo di polarizzazione, utilizzando 100 pacchetti. Si può notare che $POL = R$ scambia la bomboletta con la lattina e viceversa. Nel caso di $POL = H$, la lattina viene scambiata con la confezione di latte.

5.3.2 Test 2: Numero di pacchetti

In seguito ai risultati ottenuti nel primo test, la seconda fase si è concentrata esclusivamente sull'insieme V. La rete è stata addestrata per 200 epoche, utilizzando un numero di pacchetti compreso tra 50 e 400. La tabella 5.3 riassume i risultati quantitativi.

Tabella 5.3. Risultati quantitativi basati sul numero di pacchetti, considerando solamente la polarizzazione V.

POL	$\#Pacchetti$	$MSE \downarrow$	$SSIM \uparrow$
V	**50**	**0.028**	**0.672**
	100	0.030	0.628
	200	0.036	0.586
	400	0.030	0.579

Come si può vedere, cambiando il numero di pacchetti, si verifica una variazione nella metrica SSIM. Al contrario, L'MSE tra i pixel rimane quasi invariato per tutti i casi. I test di questa fase dimostrano nuovamente che utilizzando 50 pacchetti si ottiene il risultato migliore. Se invece si aumenta il numero di pacchetti, le *feature* estratte catturano rumore.
La figura 5.6 mostra l'andamento della loss MSE e della metrica SSIM sia per il training set che per il validation set, con $\#Pacchetti = 50$.

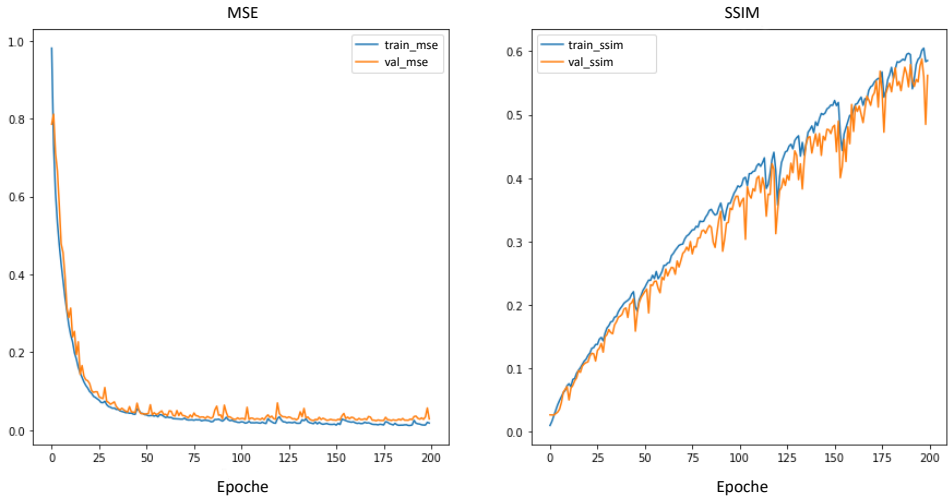


Figura 5.6. Grafici della loss MSE e dell'SSIM del modello durante la fase di addestramento e di valutazione.

I grafici suggeriscono che non vi è alcun segnale di overfitting.
Il confronto qualitativo tra le immagini originali e quelle sintetizzate dal ramo studente è mostrato nella figura 5.7, dove gli oggetti generati sono estremamente simili all'output atteso.

Figura 5.7. Confronto qualitativo finale tra le immagini originali (a), e quelle generate con $POL = V$ e $\#Pacchetti = 50$ (b).

Capitolo 6

Conclusioni

In questo lavoro è stata utilizzata un'architettura di rete capace di sintetizzare gli oggetti usando esclusivamente i segnali Wi-Fi, che vengono polarizzati in maniera differente cambiando l'orientamento delle antenne. La rete ha utilizzato una strategia di apprendimento che simula un modello maestro-studente, in cui il primo coordina il processo di sintesi del secondo solo in fase di addestramento.
La fase sperimentale ha previsto sia la raccolta dei segnali che compongono il dataset, sia i vari test effettuati per valutare la rete. I risultati ottenuti hanno dimostrato che le *feature* estratte riescono a sintetizzare correttamente gli oggetti caratterizzati da 4 materiali differenti: acciaio, alluminio, carta e plastica. Inoltre, posizionando l'antenna ricevente in maniera perpendicolare rispetto a quella trasmittente, cioè polarizzando l'antenna TX orizzontalmente e quella RX verticalmente, si migliorano le capacità di sintesi del modello. Si è visto infine che l'utilizzo di un numero di pacchetti troppo elevato (≥ 100) non permette alla rete di sintetizzare correttamente tutti gli oggetti.
Alla luce di quanto analizzato, si possono proporre modifiche al framework per consentire migliori sviluppi futuri:

- creare un sistema di antenne più sofisticato. Per esempio utilizzare in contemporanea un maggior numero di antenne riceventi, ognuna polarizzata in maniera differente;
- utilizzare antenne a dipolo al posto dei dispositivi ESP32. Queste permettono una migliore gestione della polarizzazione e sono di qualità superiore;
- affiancare l'architettura proposta con una Rete Generativa Avversaria. Le GAN, infatti, si sono dimostrate molto efficaci per vari task di sintesi;
- raccogliere un dataset più grande, contenente un maggior numero di oggetti di materiale diverso.

Bibliografia

[1] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4561–4569, 2019.

[2] Yuri Assayag, Horacio Oliveira, Eduardo Souto, Raimundo Barreto, and Richard Pazzi. Indoor positioning system using synthetic training and data fusion. *IEEE Access*, 9:115687–115699, 2021.

[3] Danilo Avola, Marco Cascio, Luigi Cinque, Alessio Fagioli, and Gian Luca Foresti. Human silhouette and skeleton video synthesis through wi-fi signals. *International Journal of Neural Systems*, 32(05):2250015, 2022.

[4] Danilo Avola, Marco Cascio, Luigi Cinque, Alessio Fagioli, Gian Luca Foresti, and Cristiano Massaroni. Master and rookie networks for person re-identification. In *Computer Analysis of Images and Patterns: 18th International Conference, CAIP 2019, Salerno, Italy, September 3–5, 2019, Proceedings, Part II 18*, pages 470–479. Springer, 2019.

[5] Yigit Bakirli, Ali Selek, and Mustafa Secmen. Broadband compact quasi yagi antenna for uhf wireless communication systems with enhanced performance at uhf ism bands. *Radioengineering*, 29(3):460–470, 2020.

[6] Xiaochao Dang, Xiong Si, Zhanjun Hao, and Yaning Huang. A novel passive indoor localization method by fusion csi amplitude and phase information. *Sensors*, 19(4):875, 2019.

[7] Hans Fischer and Hans Fischer. *The central limit theorem from Laplace to Cauchy: changes in stochastic objectives and in analytical methods*. Springer, 2011.

[8] H.T. Friis. A note on a simple transmission formula. *Proceedings of the IRE*, 34(5):254–256, 1946.

[9] Liangyi Gong, Wu Yang, Zimu Zhou, Dapeng Man, Haibin Cai, Xiancun Zhou, and Zheng Yang. An adaptive wireless passive human detection via fine-grained physical layer information. *Ad Hoc Networks*, 38:38–50, 2016.

[10] Steven M. Hernandez and Eyuphan 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, June 2020.

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[12] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[13] Sorachi Kato, Takeru Fukushima, Tomoki Murakami, Hirantha Abeysekera, Yusuke Iwasaki, Takuya Fujihashi, Takashi Watanabe, and Shunsuke Saruwatari. Csi2image: Image reconstruction from channel state information using generative adversarial networks. *IEEE Access*, 9:47154–47168, 2021.

[14] Chenning Li, Zhichao Cao, and Yunhao Liu. Deep ai enabled ubiquitous wireless sensing: A survey. *ACM Computing Surveys (CSUR)*, 54(2):1–35, 2021.

[15] Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. *Advances in Neural Information Processing Systems*, 32, 2019.

[16] Gaurav Mittal, Shubham Agrawal, Anuva Agarwal, Sushant Mehta, and Tanya Marwah. Interactive image generation using scene graphs. *arXiv preprint arXiv:1905.03743*, 2019.

[17] Ruckus Networks. Radios, antennas and other wi-fi essentials, 2011. `https://webresources.ruckuswireless.com/pdf/wp/wp-wifi-essentials.pdf`.

[18] Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78, 2004.

[19] Michele Polese, Marco Giordani, Marco Mezzavilla, Sundeep Rangan, and Michele Zorzi. Improved handover through dual connectivity in 5g mmwave mobile networks. *IEEE Journal on Selected Areas in Communications*, 35(9):2069–2084, 2017.

[20] Kun Qian, Chenshu Wu, Zheng Yang, Yunhao Liu, and Zimu Zhou. Pads: Passive detection of moving targets with dynamic speed using phy layer information. In *2014 20th IEEE international conference on parallel and distributed systems (ICPADS)*, pages 1–8. IEEE, 2014.

[21] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016.

[22] Scott E Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. *Advances in neural information processing systems*, 29, 2016.

[23] Zhao Ren, Yongjian Kang, Yingying Fan, and Jinchi Lv. Tuning-free heterogeneous inference in massive networks. *Journal of the American Statistical Association*, 114(528):1908–1925, 2019.

[24] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2017.

[25] Wikipedia. Polarizzazione della radiazione elettromagnetica — Wikipedia, the free encyclopedia. `http://it.wikipedia.org/w/index.php?title=Polarizzazione%20della%20radiazione%20elettromagnetica&oldid=129401018`, 2023. [Online; accessed 02-March-2023].

[26] Dan Wu, Daqing Zhang, Chenren Xu, Hao Wang, and Xiang Li. Device-free wifi human sensing: From pattern-based to model-based approaches. *IEEE Communications Magazine*, 55(10):91–97, 2017.

[27] Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Texturegan: Controlling deep image synthesis with texture patches. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8456–8465, 2018.

[28] Fu Xiao, Jing Chen, Xiaohui Xie, Linqing Gui, Lijuan Sun, and Ruchuan Wang. Seare: A system for exercise activity recognition and quality evaluation based on green sensing. *IEEE Transactions on Emerging Topics in Computing*, 8(3):752–761, 2018.

[29] Diana Zhang, Jingxian Wang, Junsu Jang, Junbo Zhang, and Swarun Kumar. On the feasibility of wi-fi based material sensing. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2019.