UNIVERSITÀ
DI TORINO

Università degli Studi di Torino

Corso di Laurea Magistrale in Fisica dei Sistemi Complessi

Sviluppo di un modello di machine learning per la predizione della copertura radiomobile

Tesi di Laurea

Relatore/Relatrice
Prof. FARISELLI Piero

Tutor Aziendale
Ing. Fantini Roberto

Candidato
Armato Massimiliano
Matricola 912357

Anno Accademico 2024/2025

Abstract

Il presente lavoro di tesi affronta la problematica dell'ottimizzazione della copertura e della qualità del segnale nelle reti di telecomunicazioni cellulari, un'area di significativo interesse per aziende come TIM S.p.a.. Gli attuali strumenti di pianificazione della rete, come TIMPLAN, pur essendo robusti, presentano limitazioni legate alla loro natura teorica, alla scalabilità per aree estese e ai costi di aggiornamento delle mappe geografiche. L'obiettivo primario di questa ricerca è lo sviluppo di un modello predittivo basato sul Machine Learning, capace di superare tali vincoli, fornendo stime accurate della potenza del segnale ricevuto (RSRP) e della sua qualità (RSRQ).

Il modello si ispira all'architettura proposta da Thrane et al., adottando una rete neurale composita che combina un Multi-Layer Perceptron (MLP) per l'elaborazione di dati numerici (es. coordinate e distanze dalle stazioni radio base) e una Convolutional Neural Network (CNN) per l'analisi di immagini satellitari rappresentanti la conformazione del territorio. Viene inoltre integrato il modello teorico di perdita di percorso UMa_B per guidare l'apprendimento. Il dataset di base è costituito da misure MDT (Minimization of Drive Test) fornite da TIM S.p.A., provenienti da campagne in aree come Perugia e Bologna. Sono state implementate modifiche specifiche per l'adattamento ai dati aziendali, inclusi un metodo di split spaziale e l'adeguamento dell'encoding della frequenza.

I risultati nella predizione dell'RSRP mostrano buone performance, con un errore relativo medio del 2.7% sul dataset di riferimento del paper e del 6.5% sul dataset di Bologna a 2660 MHz, dopo un'accurata pulizia dei dati da duplicati spaziali e temporali. È emerso che il contributo delle immagini satellitari alla predizione dell'RSRP sembra essere futile e richiede ulteriore studio per valutarlo.

La predizione dell'RSRQ, che richiede informazioni su rumore e interferenze, si è rivelata più complessa, raggiungendo un errore relativo medio del 13.3% sul dataset di Bologna nonostante l'aggiunta di features sulle distanze dalle antenne interferenti, sottolineando la necessità di ulteriori approfondimenti.

Questo studio valida la potenzialità di un algoritmo di Machine Learning adattivo ed efficiente per l'ottimizzazione della rete cellulare, consentendo a TIM di prendere decisioni più informate sull'installazione e l'attivazione delle antenne, con benefici in termini di risparmio energetico ed economico, e riduzione delle emissioni di CO_2. Gli sviluppi futuri includono il miglioramento della CNN, la ricerca di un modello fisico-correttivo o l'adozione di un approccio di classificazione per l'RSRQ, l'integrazione di features MDT e/o infrastrutturali aggiuntive (es. tipologia terminale, altezza e tilt dell’antenna trasmissiva) e la potenziale evoluzione verso un algoritmo per la previsione dinamica della copertura radio nel tempo.

Indice

1 Introduzione

Nel contesto attuale, le telecomunicazioni mobili rappresentano una delle infrastrutture più critiche per il funzionamento della società digitale. Con l'espansione esponenziale del traffico dati, la diffusione capillare degli smartphone e l'emergere di applicazioni avanzate come l'Internet of Things, la realtà aumentata e i veicoli connessi, le reti cellulari devono garantire prestazioni elevate, affidabilità e adattabilità in ambienti sempre più complessi.

La qualità dell'esperienza utente dipende fortemente dalla copertura radio e dalla qualità del segnale ricevuto. In questo contesto, parametri come il Reference Signal Received Power (RSRP) e il Reference Signal Received Quality (RSRQ) assumono un ruolo centrale. L'RSRP misura la potenza del segnale di riferimento ricevuto da un dispositivo mobile, mentre l'RSRQ ne valuta la qualità tenendo conto di interferenze e rumore. Questi indicatori sono fondamentali per la selezione della cella servente da parte del terminale, la gestione delle risorse radio e l'ottimizzazione della rete.

Tradizionalmente, la valutazione di questi parametri avviene attraverso strumenti di simulazione di copertura cellulare, validati tramite campagne di Drive Test o attraverso la raccolta passiva di dati MDT (Minimization of Drive Test). Tuttavia, queste metodologie presentano limiti in termini di copertura spaziale, costi operativi e aggiornamento dinamico. In scenari urbani densi, con elevata variabilità ambientale e comportamenti utente non prevedibili, diventa sempre più difficile modellare la propagazione radio con approcci deterministici o empirici.

È in questo contesto che il Machine Learning (ML) si propone come strumento potente e flessibile per la predizione dei parametri radio. Grazie alla capacità di apprendere da grandi quantità di dati eterogenei, i modelli ML possono catturare relazioni non lineari, dipendenze spaziali e temporali, e pattern nascosti che sfuggono alle tecniche tradizionali.

In questo lavoro di tesi verrà quindi presentato un modello di ML utile alla predizione di RSRP ed RSRQ. Si tratta di un algoritmo sviluppato in sinergia con l'azienda TIM S.p.a., la quale, per mezzo della figura di un tutor aziendale, mi ha supportato durante tutto il percorso di studio, modifica e implementazione del modello. L'intento è quello di superare i limiti dei modelli teorici tradizionali, come TIMPLAN, uno strumento proprietario di TIM per la stima di queste metriche radio, offrendo un algoritmo più adattabile ai dati specifici, veloce nella valutazione e meno costoso grazie all'uso di strumenti opensource per la visualizzazione del territorio.

Sfrutteremo una rete neurale composita, basata sulla cooperazione di un multi layer perceptron (MLP) e una rete neurale convoluzionale (CNN), con l'aggiunta di un modello fisico di supporto per aiutare la rete ad imparare relazioni e fenomeni fisici senza dover partire da zero.

La trattazione sarà suddivisa in diversi capitoli:

- Background teorico: verranno spiegati i concetti di telecomunicazione e machine learning usati, in modo da poter comprendere il seguito del lavoro.

- Stato dell'arte: breve elenco della letteratura trovata sull'incontro tra machine learning e telecomunicazioni. Verrà inoltre esposto un articolo di riferimento, su cui ci siamo basati per lo sviluppo dell'algoritmo

- Modello: presentazione della struttura della rete dell'articolo. Vengono inoltre esposte le nostre modifiche al codice, per venire incontro a problemi legati al cambio di dataset.

- Metodologia: descrizione dettagliata di tutti gli step compiuti e le ipotesi formulate per portare a termine il lavoro di tesi.

- Risultati: esposizione dei risultati salienti ottenuti, con aggiunta di grafici e visualizzazioni.
- Sviluppi futuri: proposte di procedure e tecniche per migliorare il lavoro e progredire verso la soluzione ai diversi problemi rilevati nella Metodologia e nei Risultati.
- Conclusioni: trattazione finale e riassuntiva sui passi condotti, sui risultati notevoli ottenuti e sulle possibili soluzioni proposte, per avere una panoramica generale e finale su tutto il lavoro.
- Bibliografia: elenco della letteratura e delle fonti consultate.

2 Background teorico

Per comprendere a pieno il lavoro di tesi svolto, è importante avere a mente alcuni concetti e nozioni base sulle telecomunicazioni e sulle reti neurali. In questo modo potremo visualizzare bene il funzionamento del modello ed il suo scopo.

2.1 Teoria telecomunicazioni

Di seguito vengono introdotti i concetti e le definizioni principali da conoscere nell'ambito delle telecomunicazioni per comprendere l'obiettivo dell'algoritmo.

2.1.1 Rete cellulare

La rete telefonica cellulare è una tecnologia studiata per comunicare a distanza tramite dispositivi portatili, come i nostri smartphone. Questa è composta da antenne, il cui compito è quello di ricevere e trasmettere segnali radio da e verso un dispositivo. Queste antenne sono situate in infrastrutture chiamate stazioni radio base e suddividono il territorio in aree geografiche chiamate celle, contigue e adiacenti (Figura 1).

Figura 1: Rappresentazione concettuale della rete cellulare

Trasmettono continuamente onde radio, che determinano la connessione dell'utente. Queste possono degradarsi a causa di vari fattori, quali distanza dalla sorgente, ostacoli naturali o artificiali, condizioni atmosferiche e curvature terrestre. Questi segnali possono essere a frequenze basse o alte (es. 800MHz e 2660 MHz): le prime sono importanti in quanto hanno un'alta capacità di penetrazione e coprono distanze maggiori, tuttavia sono solitamente associate a larghezze di banda ridotte e hanno quindi una capacità di trasmissione dati più ridotta. Onde ad alta frequenza, invece, sono più sensibili agli ostacoli e coprono distanze minori, ma sono solitamente associate a larghezze di banda maggiori, per cui possono trasportare molti

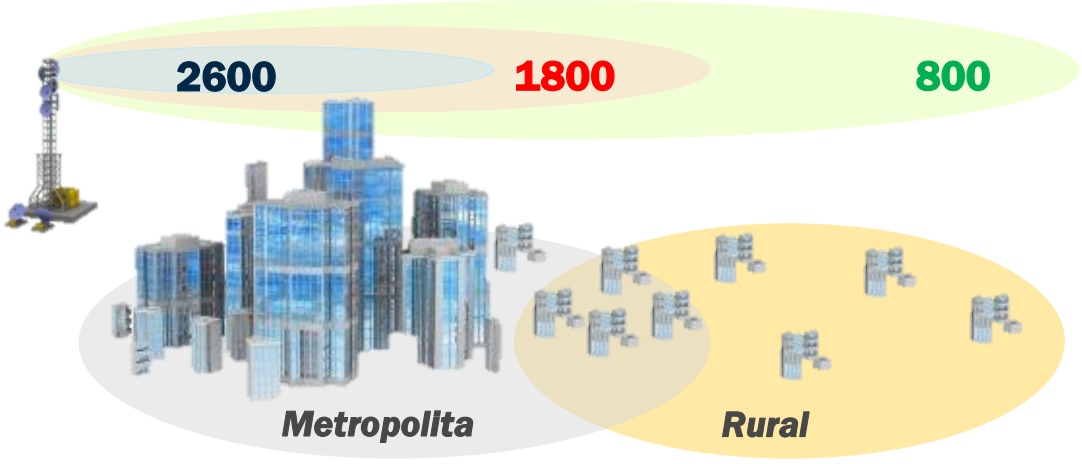

Figura 2: Rappresentazione concettuale della copertura fornita da diverse frequenze

bit al secondo, permettendo download e upload a velocità elevate. Per queste diverse caratteristiche fisiche, la frequenza a 800MHz è tradizionalmente utilizzata per offrire coperture più ampie in zone in cui la necessità di capacità è meno marcata, quindi in zone rurali, mentre le frequenze più alte, a 1800MHz o a 2600MHz, vengono aggiunte come ulteriori "layer" frequenziali per aumentare la capacità nelle aree dove c'è una maggiore concentrazione di popolazione, quindi in zone sub-urbane o urbane (Figura 2)

La necessitrà di offrire capacità di trasmissione dati maggiore ha anche determinato l'evoluzione della tecnologia della rete mobile da 3G a 5G (Figura 3), la quale è caratterizzata dalla qualità e quantità dei servizi forniti:

- **3G**: navigazione e videochiamate base → contenuti testuali e multimediali leggeri;
- **4G**: streaming HD, social media, cloud → video, app complesse, cloud computing.
- **5G**: realtà aumentata, auto connesse, chirurgia remota → esperienze immersive, controllo in tempo reale, automazione industriale.

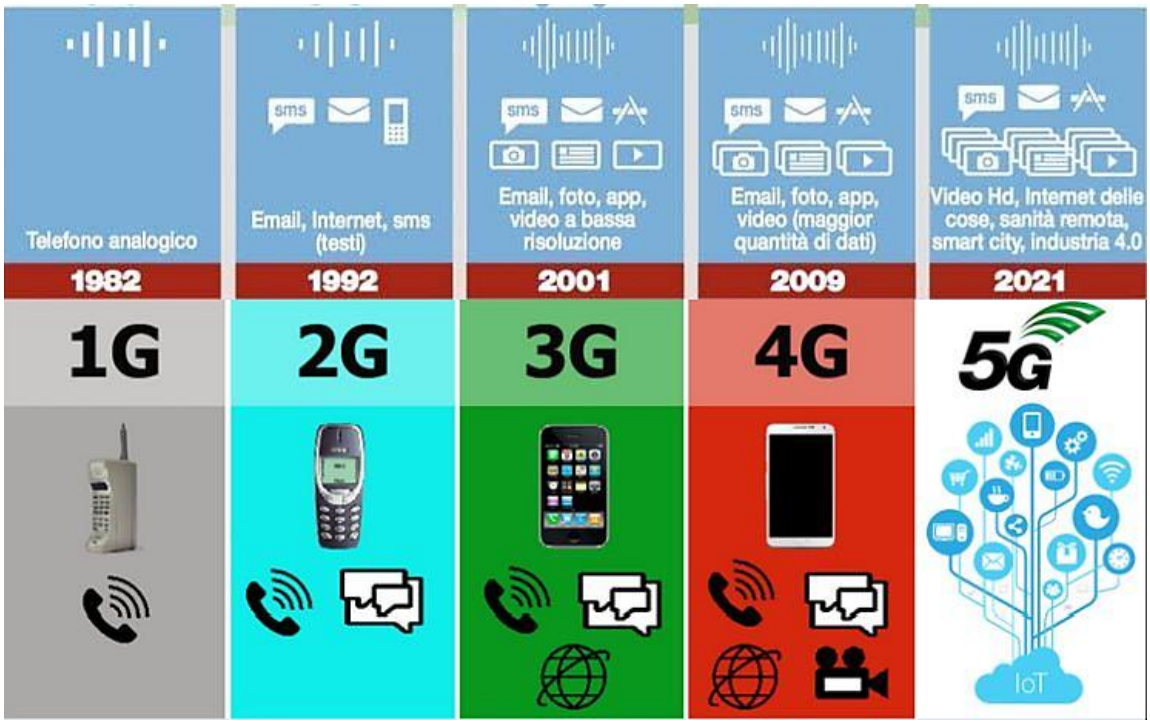

Figura 3: Visualizzazione dell'evoluzione della rete mobile

Questa evoluzione è stata possibile soprattutto grazie all'aumento progressivo della capacità di trasmissione dati.

Le antenne possono supportare anche più frequenze diverse, non troppo distanti tra di loro (ad. esempio 2600 MHz con 1800 MHz, ma non con 800 MHz): concettualmente ciò determina nella rete cellulare la sovrapposizione di due celle (o cosiddetti "layer frequenziali") contemporaneamente.

2.1.2 RSRP ed RSRQ

I seguenti concetti descritti corrispondono alle labels che il modello è incaricato di predire.

RSRP – Reference Signal Received Power

L’RSRP rappresenta la potenza del segnale di riferimento ricevuto da un dispositivo mobile, ed è una misura fondamentale per valutare la copertura radio di una cella. Viene calcolato come la potenza media dei segnali di riferimento trasmessi dalla stazione base (Reference Signals), ed è espresso in dBm.

Valori tipici:

- Buona copertura: > –80 dBm
- Copertura accettabile: –90 / –100 dBm
- Copertura debole: < –110 dBm

L’RSRP è utilizzato dai terminali per i seguenti scopi:

- Selezione della cella iniziale: quando un dispositivo mobile si accende deve agganciarsi ad una cella; pertanto, scansiona le celle disponibili, misura l’RSRP e si aggancia alla cella con RSRP più alto.
- Handover tra celle durante la mobilità: si tratta del passaggio tra una cella all’altra e avviene quando l’RSRP scende sotto una certa soglia.
- Misure MDT per la pianificazione e ottimizzazione della rete (verrà spiegato ampiamente nei paragrafi successivi).

RSRQ – Reference Signal Received Quality

L’RSRQ misura la qualità del segnale ricevuto, tenendo conto non solo della potenza (RSRP), ma anche del rumore e delle interferenze presenti nel canale. È calcolato come rapporto tra RSRP e RSSI (Received Signal Strength Indicator). L’RSSI misura la potenza complessiva che il terminale misura sull’intera banda del segnale. Comprende quindi sia il segnale utile, che eventuali interferenze e il rumore termico. Il rapporto tra RSRP ed RSSI fornisce quindi un’indicazione più completa della effettiva utilizzabilità del segnale rispetto al solo RSRP.

Valori tipici:

- Ottima qualità: > –10 dB
- Qualità media: –10 / –15 dB
- Qualità scarsa: < –15 dB

L'RSRQ è particolarmente utile per:

- Valutare la congestione della cella: se questa infatti è molto affollata, ovvero sono collegati molti utenti contemporaneamente, l'RSSI aumenta e l'RSRQ tende a peggiorare.
- Valutare il livello di interferenza per l'utente: quando l'utente è vicino al bordo della cella, il segnale delle celle adiacenti diventa più forte e produce un'interferenza sul segnale utile che viene rilevata dalla misura di RSSI. Un maggiore RSSI porta ad un peggior livello di RSRQ, e conseguentemente una peggiore qualità per l'utente.

- Ottimizzare l'handover in scenari con interferenze: durante il passaggio da una cella all'altra, il terminale valuta anche la qualità del segnale per scegliere la cella migliore a cui agganciarsi.

- Diagnosi di problemi radio in ambienti urbani complessi: in città, il segnale radio può subire riflessioni, diffrazioni, e ostruzioni da edifici, veicoli, alberi. Questo crea fluttuazioni e interferenze che non sempre si vedono nell'RSRP, ma che l'RSRQ riesce a catturare. Gli ingegneri di rete analizzano l'RSRQ per individuare zone problematiche, dove la qualità del segnale è scarsa anche se la potenza sembra sufficiente. Questo è fondamentale per:
    - Ottimizzare la disposizione delle antenne
    - Regolare i parametri di rete
    - Migliorare l'esperienza utente

Questi concetti introdotti sono importanti, in quanto i dataset saranno costituiti da punti, che rappresentano dispositivi mobili. Questi saranno localizzati spazialmente in celle e riceveranno il segnale trasmesso dalle stazioni radio, misurando RSRP e RSRQ. Il modello ha lo scopo di predire l'RSRP e possibilmente l'RSRQ, data la posizione del dispositivo, la distanza dall'antenna della stazione radio base e la conformazione del territorio intorno al punto.

2.1.3 Modelli UMa

Esistono modelli teorici utili allo scopo di stimare il segnale radio ricevuto che si chiamano UMa (Urban Macro channel model). Questi non calcolano direttamente la potenza o la qualità del segnale, bensì la sua attenuazione durante il tragitto dal trasmettitore (stazione base) al ricevitore (apparecchiatura utente). Inoltre, descrivono la perdita di percorso in scenari macrocellulari urbani, come aree urbane densamente popolate, in cui grandi stazioni base macro vi forniscono copertura.

Questi modelli sono molto complessi e adatti a determinati scenari.

Anche nel nostro algoritmo verrà usato una semplificazione di UMa, ovvero UMa_B [1].

La formula usata per la stima della Path Loss sarà la seguente:

$$PL = 13.54 + 39.08\ \log_{10}(d3d) + 20\log_{10}(fc) - 0.6(h_t - 1.5) \qquad (1)$$

dove i valori numerici sono delle costanti derivanti da campagne di misurazione empirica e regressione statistica condotte da enti come il 3GPP (3rd Generation Partnership Project); fc è la frequenza in GHz, h_t è l'altezza a cui si trova il terminale dell'utente, d3d è la distanza 3d tra stazione radio base e terminale.

2.2 Motivazioni aziendali

Come abbiamo già detto, questo lavoro è di interesse per l'azienda TIM S.p.a., pertanto è bene dare una panoramica riguardo alle motivazioni sullo sviluppo di questa ricerca.

2.2.1 TIM

Innanzitutto, introduco l'azienda con una breve descrizione: TIM S.p.A. (Telecom Italia Mobile) è il principale operatore di telecomunicazioni in Italia e uno dei protagonisti europei nel settore delle tecnologie digitali. Fondata ufficialmente nel 1994 a seguito della fusione di diverse società storiche del comparto telefonico italiano, TIM ha ereditato un patrimonio industriale e tecnologico che affonda le radici nel secolo scorso, quando le prime reti telefoniche venivano gestite da concessionarie regionali sotto il controllo dello Stato. La nascita di Telecom Italia ha segnato l'inizio di una nuova fase di liberalizzazione e privatizzazione del settore, in linea con le direttive europee, culminata con l'adozione del marchio unificato TIM nel 2016.

Con sede legale a Milano e direzione generale a Roma, TIM opera oggi su scala nazionale e internazionale, con una presenza significativa anche in Brasile attraverso TIM Brasil, e in San Marino con TIM San Marino. Il gruppo è quotato alla Borsa Italiana e rappresenta uno dei maggiori attori economici del Paese, con un fatturato annuo superiore ai 14 miliardi di euro.

TIM offre una gamma completa di servizi di telecomunicazione: telefonia fissa e mobile, connettività Internet, IPTV, VoIP, cloud computing, cybersecurity e soluzioni IoT. Attraverso la sua divisione TIM Enterprise, l'azienda fornisce soluzioni end-to-end per imprese e pubbliche amministrazioni, contribuendo attivamente alla digitalizzazione del tessuto produttivo italiano. Inoltre, TIM è coinvolta nella gestione della connettività della Pubblica Amministrazione tramite il Sistema Pubblico di Connettività, svolgendo un ruolo strategico nella modernizzazione dei servizi pubblici.

L’azienda si distingue per il forte impegno verso l’innovazione e la sostenibilità. TIM investe costantemente in innovazione, con particolare attenzione alla qualità del servizio, alla copertura radio e all’ottimizzazione delle risorse di rete.

Un esempio di investimento è stato quello del 2011, durante l’asta delle bande di frequenza indetta dal governo italiano. Qui, l’azienda ha speso 1.26 miliardi di euro per comprare diversi pacchetti di bande di frequenza: 15 MHz di frequenza 2660 MHz, 5 da 1800 MHz e 10 da 800 MHz.

Figura 4: Sede TIM Lab Torino

2.2.2 TIMPLAN e obiettivo del lavoro

TIM, inoltre, sviluppa e utilizza strumenti proprietari per la pianificazione e l’analisi della rete, tra cui TIMPLAN. Quest’ultimo è un esempio dell’applicazione pratica di un modello teorico tipo UMa. Utilizza equazioni

complesse, simili a quelle citate per la path loss, e mappe geografiche per l'analisi della conformazione del territorio.

Questo strumento, per quanto accurato nonchè validato da numerose campagne di misure effettuate in campo, presenta delle limitazioni: si tratta di un modello teorico adatto ad un generico territorio o città, quindi, per quanto sia stato migliorato negli anni, l'algoritmo non si adatta perfettamente ai dati; inoltre, per analizzare aree molto estese il tempo richiesto risulta considerevole. Per ultimo, saltuariamente è necessario l'aggiornamento delle mappe geografiche, il quale risulta molto costoso per l'azienda.

Il modello proposto in questo lavoro di tesi cerca di superare questi vincoli: infatti, essendo un algoritmo di machine learning, ha la caratteristica di adattarsi al dataset specifico; completato l'allenamento la valutazione delle label dei punti è molto veloce, anche per dataset di grandi dimensioni; i costi sono limitati, in quanto per la visualizzazione del territorio viene usato un metodo opensource gratuito.

Ci si può chiedere perchè queste misure, RSRP e RSRQ, sono così importanti per l'azienda: il motivo sta nel fatto che queste metriche sono utili a valutare la copertura radio mobile. Con una valutazione attendibile è possibile quindi ottimizzare la copertura. Infatti, possiamo identificare quali e quanti dispositivi vengono serviti dalle antenne, se il segnale è potente oppure scarso e se la qualità del segnale raggiunge i livelli minimi garantiti. Questo assiste gli operatori di rete a prendere decisioni quali installare nuove antenne o spegnerne. Se ad esempio diversi dispositivi rilevano scarso segnale, questo potrebbe essere un motivo per cui aggiungere un'antenna alla rete; oppure se una certa stazione base serve pochi dispositivi, potrebbe essere vantaggioso spegnerla e riassegnare i dispositivi su un'altra cella attiva. Questo può portare sia a un risparmio energetico ed economico, sia ad una riduzione dei consumi, come ad esempio emissioni di CO_2.

2.3 I dati

Ci sono diverse tecniche per ottenere misure nell'ambito delle telecomunicazioni; ne cito quindi due, Drive Test e MDT: la prima viene usata dal paper su cui ci siamo basati per il lavoro, sulla seconda invece si fondano i nostri dati aziendali.

2.3.1 Drive test

I drive test sono una tecnica tradizionale utilizzata dagli operatori di rete mobile per valutare la qualità del servizio offerto in una determinata area geografica. Consiste nel far circolare un veicolo equipaggiato con strumenti di misura (modem, scanner RF, GPS, software di analisi) lungo un percorso prestabilito, per raccogliere dati sulla rete mobile in tempo reale.

Questa tecnica è caratterizzata da diversi vantaggi, ma anche da varie limitazioni.

I vantaggi sono i seguenti:

- Le misure sono precise e controllate.
- È possibile testare scenari specifici (es. Tunnel o zone critiche).
- Controllo totale sull'hardware e sulla configurazione.

Le limitazioni:

- Si tratta di un approccio costoso in termini di tempo, di personale e di carburante, in quanto è necessario incaricare un tecnico, che con un veicolo adatto prenda misure mentre percorre le strade specificate per la raccolta dati.
- Sono presenti vincoli spaziali, poichè si può misurare solo dove il veicolo è in grado di passare.
- È statico temporalmente, in quanto le misure sono legate al momento in cui viene effettuato il test.

2.3.2 MDT

MDT (Minimization Drive Test) invece è una tecnica che consente ai terminali mobili di raccogliere e inviare misure di rete direttamente agli operatori, in modo automatico e distribuito. Le misure possono essere raccolte e inviate in tempo reale oppure registrate nel terminale e inviate successivamente, ad esempio quando il dispositivo è connesso a una rete Wi-Fi.

Vediamo quindi i vantaggi e gli svantaggi legati a questa tecnica.

Vantaggi:

- La copertura spaziale è molto ampia, in quanto non è necessario utilizzare un veicolo per la raccolta dati; pertanto anche aree difficilmente raggiungibili sono incluse.
- I dati sono 'reali': infatti, riflettono l'esperienza utente quotidiana.
- I costi operativi e l'impatto ambientale sono ridotti.

Limitazioni:

- La precisione risulta variabile, in quanto dipende dal tipo di dispositivo e dalla configurazione.
- Il controllo è limitato, dato che l'operatore non può sempre scegliere dove e quando vengono raccolti i dati.
- Dipendenza dal comportamento dell'utente.

2.3.3 Confronto

Entrambe le tecniche raccolgono misure come posizione geografica e temporale, RSRP ed RSRQ, identificativo della cella e frequenza utilizzata.

Aspetto	Drive Test	MDT
Metodo di raccolta	Manuale, veicolo con strumenti	Automatico, tramite smartphone UE
Controllo	Elevato	Limitato
Copertura spaziale	Limitata (solo percorsi veicolari)	Ampia (ovunque ci siano utenti)
Copertura temporale	Puntuale (momento del test)	Estesa (giorni, settimane)
Precisione delle misure	Alta	Variabile, dipende dal terminale
Costo operativo	Alto	Basso

Tabella 1: Tabella di confronto Drive Test - MDT

2.4 Teoria Machine Learning

Per riuscire a seguire la trattazione sulla struttura dell'algoritmo è necessaria un'introduzione al machine learning e al funzionamento delle reti neurali.

2.4.1 Neural Network (NN)

Una rete neurale è una sorta di complessa funzione che prende come input una matrice X e restituisce un output vettoriale Y. X rappresenta l'insieme dei dati, ognuno associato alle proprie caratteristiche, le features. Y, invece,

sono le labels, le etichette, e corrispondono a ciò che si vuole predire partendo da quei dati. Un esempio canonico è quello del mercato immobiliare: immaginiamo di considerare N case, ognuna con m caratteristiche come superficie in metri quadri, altezza del soffitto, colori delle pareti o altro, e dobbiamo stimare il prezzo di ogni casa. L'insieme delle caratteristiche delle N case, quindi, corrisponde a X e il vettore dei prezzi ad Y.

In questo specifico esempio, stiamo stimando dei valori numerici continui: questo è un problema chiamato Regressione. Nel caso in cui volessimo predire valori discreti, si parla di Classificazione.

In questo lavoro di tesi, ci interessa un output numerico continuo, pertanto siamo nell'ambito della Regressione.

In una rete neurale, la prima cosa da fare, a discapito del tipo di problema, è normalizzare la matrice di input.

Si tratta di sottrarre ad ogni misura il valore medio e la deviazione standard delle misure:

$$x_{norm} = \frac{x - \hat{x}}{\sigma} \tag{2}$$

Questo garantisce dati centrati e scalati: infatti, dati grezzi possono avere scale molto diverse (es. altezza in cm vs. stipendio in euro). Se non vengono normalizzati, i pesi associati a features con valori più grandi possono dominare sugli altri, distorcendo l'apprendimento della rete e quindi la predizione del modello. Per quanto riguarda il centramento, questo facilita il lavoro delle funzioni all'interno della rete, in quanto rende la distribuzione maggiormente simmetrica rispetto allo 0.

Configurata la matrice X, la rete quindi provvede al passaggio dall'input all'output: questo, tuttavia, non avviene in un unico step; infatti, X, attraverso apposite trasformazioni, viene ridotta progressivamente in configurazioni matriciali diverse, chiamate strati nascosti, fino a diventare il vettore Y (un esempio nella Figura 5). Questo processo è necessario perché per raggiungere una predizione corretta, la rete deve comprendere complesse

relazioni tra i dati e deve essere in grado di poterli separare anche non linearmente: ciò richiede più passaggi e funzioni non lineari.

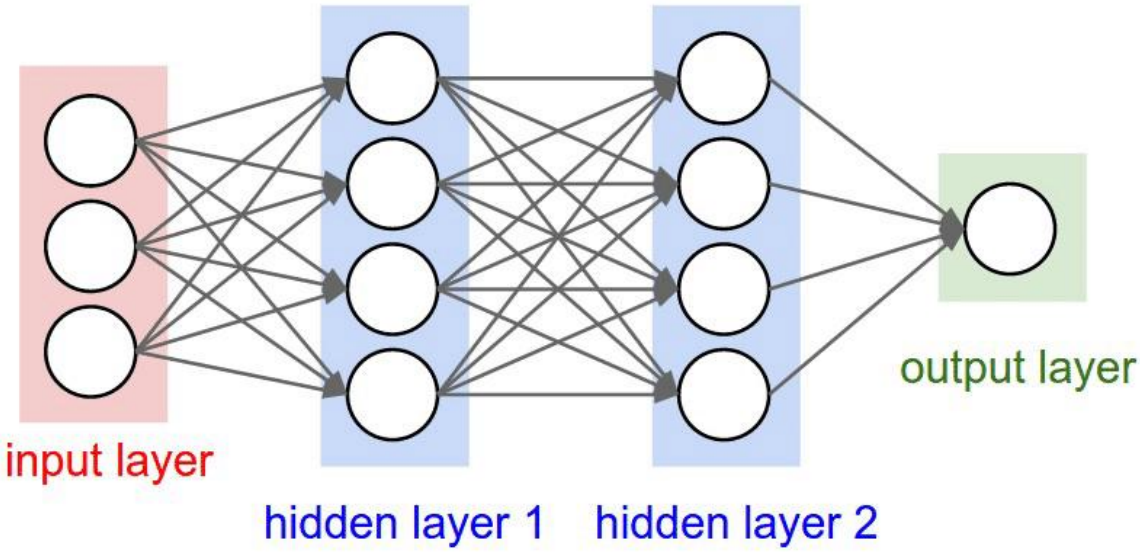

Figura 5: Esempio 2d della struttura della rete neurale per una singola misura.

Per andare da uno strato all'altro si utilizzano quindi delle trasformazioni. Le principali sono le seguenti: la trasformazione lineare, la funzione di attivazione e la Batch Normalization.

La prima definisce una combinazione lineare dell'input della trasformazione:

$$Z = w * x + b \tag{3}$$

dove w è il fattore che moltiplica scalarmente l'input e corrisponde ai pesi del modello che vengono imparati durante l'allenamento; b è il bias. Dal momento che la matrice dei pesi moltiplica scalarmente l'intera matrice x, la

rete si dice Fully Connected; questo concetto sarà diverso nell'analisi delle reti di tipo convoluzionale (CNN).

La funzione di attivazione, invece, agisce subito dopo la trasformazione lineare sul suo output Z:

$$a = g(Z) \tag{4}$$

dove *a* sarà il nuovo strato e g è una funzione non-lineare che serve alla rete per tenere conto di pattern complessi e non linearità. Esistono diversi esempi di funzione di attivazione. Una tra le più importanti è la ReLU (Figura 6), che verrà anche adottata nel nostro modello:

$$g(Z) = \max(0, Z) \tag{5}$$

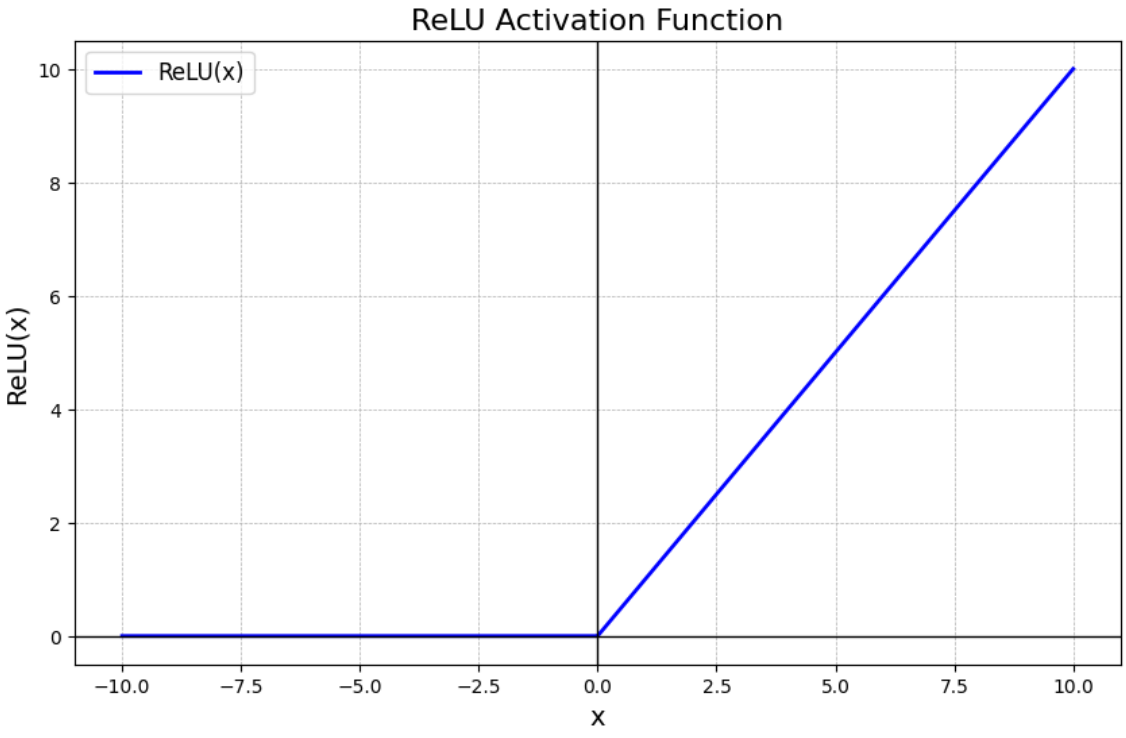

Figura 6: Rappresentazione grafica della funzione ReLU

Questa ha diversi vantaggi:

- È molto semplice da calcolare, il che significa un tempo di addestramento ridotto rispetto ad altre funzioni di attivazione più complesse.
- Aiuta la rete a convergere più velocemente durante la discesa del gradiente (in sostanza quanto sta sbagliando). In pratica, il modello impara più rapidamente e con meno iterazioni.

Esistono numerose altre funzioni di attivazione (Figura 7), ad esempio la sigmoide o la tanh, tuttavia queste sono maggiormente usate in altri problemi, come la Classificazione.

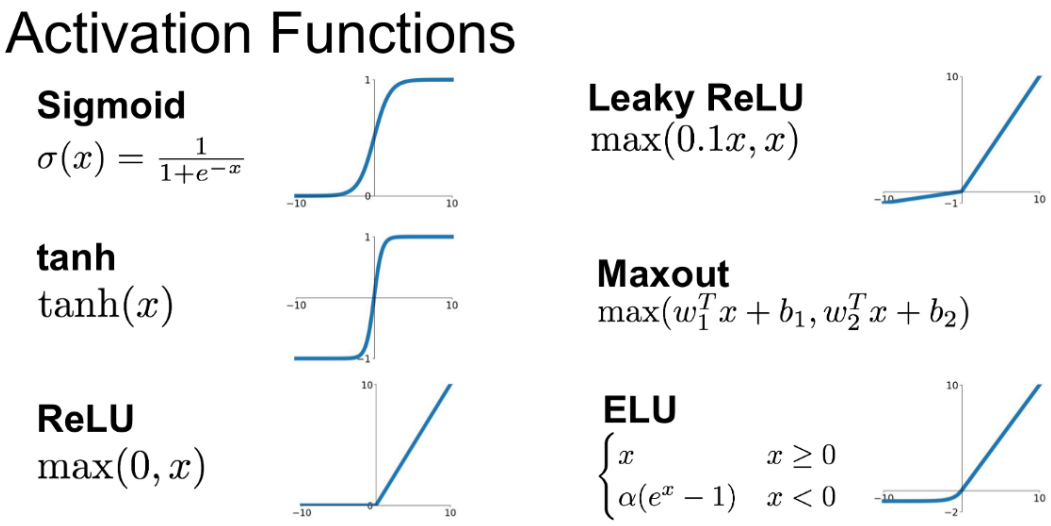

Figura 7: Rappresentazione matematica e grafica delle più importanti funzioni d'attivazione

Inoltre, come per la matrice iniziale, anche lo strato deve essere normalizzato e quindi viene applicata la trasformazione chiamata Batch Normalization. Questa mantiene gli input degli strati con media zero e varianza unitaria,

facilitando l'apprendimento. In sostanza, evita che le attivazioni esplodano o si annullino.

Queste tre funzioni si ripetono in sequenza nella rete a seconda di quanti strati si vogliono e della complissità del modello. Un algoritmo di questo tipo si chiama Multi-Layer-Perceptron (MLP).

2.4.2 Convolutional Neural Network (CNN)

La gestione di un dataset composto da immagini è molto diversa rispetto a quella di un dataset formato da valori numerici: infatti un insieme di immagini non è rappresentato da una matrice, bensì da un tensore; inoltre, ogni singola immagine è essa stessa un tensore.

Per analizzarne uno non si considera ogni valore isolato, bensì porzioni locali, mediante l'uso di un filtro che scorre lungo tutto il tensore; questo approccio diminuisce il carico computazionale e permette di focalizzarsi su zone localizzate del tensore, come farebbe un essere umano quando guarda un'immagine e prova ad elaborarla.

Il filtro è una matrice di pesi che moltiplica scalarmente la porzione selezionata per ottenere un nuovo valore. Il risultato dello scorrimento del filtro lungo tutto il tensore è una nuova rappresentazione dell'input con

dimensioni diverse da quelle iniziali. Questo processo si chiama Convoluzione e possiamo osservarne un esempio nella Figura 8.

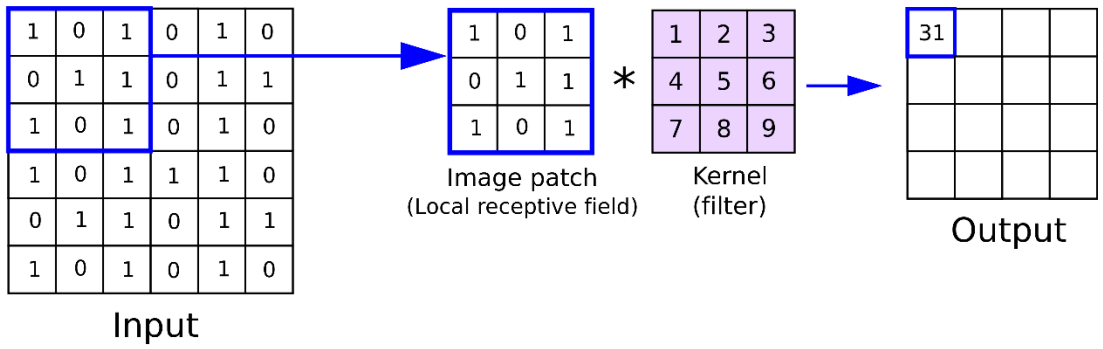

Figura 8: Esempio di applicazione di un filtro a una porzione locale di immagine

Le dimensioni dell'output della convoluzione si basano sulla seguente equazione:

$$n_{out} = \frac{n_{in} + 2p - f}{s} + 1 \tag{6}$$

questa rappresenta una delle dimensioni della matrice di output, dove n_{in} è la dimensione iniziale dell'immagine, p è il padding, s è lo stride ed f è la dimensione del filtro. Lo stride indica di quanto scorre il filtro sull'immagine. Il padding indica il numero di strati formati da 0 che andranno a circondare l'output a fine convoluzione. Questi strati vengono aggiunti affinchè si tenga maggiormente conto dei bordi dell'immagine:

infatti, quando il filtro scorre su di essa, passa più volte sulle zone centrali che sui bordi.

Pertanto, le dimensioni complessive del tensore di output della convoluzione sono le seguenti:

$$d = n_{out} \times n_{out} \times n_f \tag{7}$$

dove n_f corrisponde al numero di filtri che vengono usati ed è un parametro definito nell'inizializzazione (iperparametro).

Quando l'immagine ha raggiunto la nuova rappresentazione, caratterizzata dalle dimensioni desiderate, questa viene compressa in un vettore; pertanto, considerando tutte le immagini, si ottiene una matrice X che riutilizziamo in una rete neurale fully connected (come quella definita nel paragrafo precedente) per ottenere il vettore di output finale.

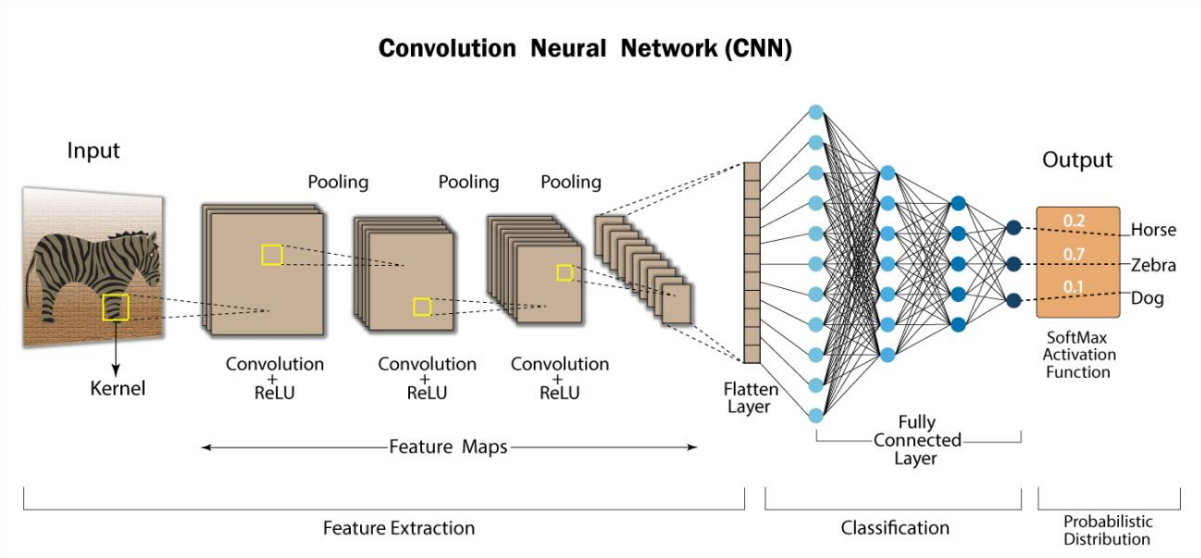

Figura 9: Esempio della struttura di una rete neurale convoluzionale per una singola immagine con task Classificazione.

Dalla Figura 9 possiamo notare il pooling: questa è una trasformazione molto utile che accompagna solitamente la convoluzione e funziona in modo

simile. Considera la configurazione tensoriale ottenuta dalla convoluzione e fa scorrere un filtro su di essa; tuttavia, a differenza della convoluzione, il filtro non serve per calcolare il prodotto scalare, bensì soltanto per selezionare le porzioni locali del tensore. A quel punto per ogni area localizzata si prende o il valore massimo o la media tra i valori a seconda se si stia effettuando il max-poolong o l'average-pooling, come mostrato in Figura 10. Lo scopo di questa trasformazione è tenere in considerazione l'invarianza traslazionale: infatti se un oggetto si sposta leggermente nell'immagine, il pooling aiuta a mantenere la sua presenza nel feature map.

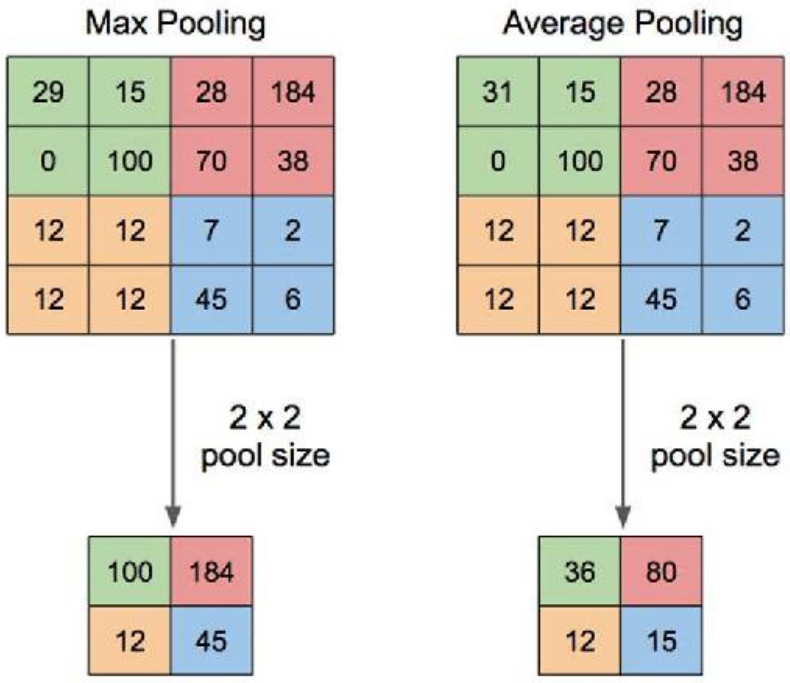

Figura 10: Esempio del funzionamento del pooling e della differenza tra max ed average pooling.

2.4.3 I pesi

Le matrici dei pesi (compresi i filtri nelle convoluzioni) sono l'elemento fondamentale in una rete neurale. Questi valori vengono moltiplicati per

l'input e per gli strati successiva e danno vita all'output. Se sono definiti in maniera scorretta anche il risultato finale lo sarà. Pertanto, è necessario trovare un sistema per riuscire a calcolarli il meglio possibile.

Per prima cosa, vengono inizializzati random, in quanto inizialmente non si hanno informazioni sui loro valori. Ovviamente, ne consegue che non diano risultati soddisfacenti nell'immediato. É necessario un allenamento dell'algoritmo, in modo che esso li corregga progressivamente. Abbiamo quindi bisogno di un metodo per valutare quanto l'algoritmo stia sbagliando: si utilizza la Loss, ovvero una funzione che confronta le label predette con quelle vere dei dati. Ne esistono diversi tipi, ma nella regressione viene solitamente usata la Mean Squared Error (MSE) Loss:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{8}$$

dove y_i sono i valori di output predetti, $\hat{y}_i$ sono le labels vere e N è il numero totali di dati.

Altre due Loss importanti (non verranno trattate in questo lavoro) sono la Cross-Entropy, la quale viene usata nella classificazione binaria, e la Softmax, che viene usata nella classificazione multiclasse.

Definito l'errore, la rete aggiorna i pesi attraverso un processo iterativo chiamato Backpropagation, che sfrutta il gradiente della funzione di perdita rispetto ad essi. In pratica, si calcola quanto ogni peso ha contribuito all'errore e lo si modifica in direzione opposta al gradiente, secondo una certa intensità, determinata da un iperparametro, il learning rate. Questo meccanismo, chiamato Gradient Descent, consente alla rete di apprendere, migliorando le proprie previsioni ad ogni epoca di addestramento:

$$w^{(t+1)} = w^{(t)} - \alpha \frac{\partial L}{\partial w} \tag{9}$$

dove $w^{(t)}$ sono i pesi prima dell'aggiornamento, α è il learning rate e L è la Loss.

Sebbene Gradient Descent sia concettualmente semplice ed efficace, presenta alcune limitazioni pratiche, soprattutto in presenza di dataset di grandi dimensioni o funzioni di perdita complesse. Pertanto, per affrontare questi problemi, si è introdotto il metodo dello Stochastic Gradient Descent (SGD), che aggiorna i pesi utilizzando un sottoinsieme casuale dei dati (mini-batch), riducendo il costo computazionale per ogni iterazione e introducendo una variabilità utile per evitare minimi locali.

$$w^{(t+1)} = w^{(t)} - \alpha g^{(t)} \tag{10}$$

con

$$g^{(t)} = \frac{1}{m} \sum_{j=i_1,\dots,i_m \in \{1,\dots,N\}} \nabla_w L\left(x_j,\ y_j, w\right) \tag{11}$$

Infine, per un'ottimizzazione più sofisticata è stato sviluppato ADAM, che combina i vantaggi di SGD con tecniche di adattamento del learning rate:

$$w^{(t+1)} = w^{(t)} - \alpha \frac{p^{(t)}}{\sqrt{s^{(t)}}} \tag{12}$$

con

$$p^{(t)} = \beta_1 p^{(t-1)} + (1 - \beta_1) g^{(t)} \quad (13)$$

$$s^{(t)} = \beta_2 s^{(t-1)} + (1 - \beta_1) \left(g^{(t)}\right)^2 \quad (14)$$

dove $p^{(t)}$ rappresenta la stima del primo momento, ovvero la media esponenziale dei gradienti, ed $s^{(t)}$, invece, la stima del secondo momento, ovvero la media esponenziale dei quadrati dei gradienti. $p^{(t)}$ cattura la direzione media del cambiamento, mentre $s^{(t)}$ misura la variazione dei gradienti e consente di adattare il passo di aggiornamento in base alla stabilità del gradiente.

3 Stato dell'arte

Per sviluppare questo lavoro di tesi, abbiamo fatto la scelta di non partire totalmente da zero nella modellazione, bensì abbiamo deciso di cercare un articolo scientifico con un obiettivo simile al nostro, copiarne il codice GitHub associato e arricchirlo, modificandolo per adattarlo ai nostri dati.

3.1 Ricerca bibliografica

Grazie ad un'approfondita ricerca su Google Scholar e ArXiv, abbiamo selezionato diversi articoli da analizzare per decidere da quale partire.

I seguenti papers scientifici non sono stati usati per lo sviluppo diretto del modello, tuttavia sono stati comunque studiati in modo da cercare l'articolo adatto; pertanto, è bene darne una veloce panoramica introduttiva.

- **Towards Addressing the Spatial Sparsity of MDT Reports to Enable Zero Touch Network Automation** [2]
    - Questo studio affronta il problema della scarsità spaziale nei rapporti di Minimization of Drive Test (MDT), che compromette le prestazioni dei modelli di Machine Learning per l'automazione della rete. Si distingue per l'utilizzo di modelli generativi, in particolare Generative Adversarial Networks e Variational Autoencoders, per aumentare dati MDT tabulari multidimensionali scarsi, a differenza di molta letteratura che si concentra su dati di immagine o audio.

- **Drive test minimization using Deep Learning with Bayesian approximation** [3]
    - Questo lavoro dimostra come le tecniche di Deep Learning (DL) possano essere utilizzate per prevedere le metriche di qualità del segnale LTE (RSRP, RSRQ, SINR) usando misurazioni dei drive test e fornisce indicazioni su dove sono necessarie ulteriori misurazioni. Si distingue per l'uso di Deep Neural Networks combinate con Convolutional Neural Networks per apprendere una funzione di mappatura da dati basati sulla posizione e immagini satellitari, e per l'incorporazione dell'approssimazione bayesiana per fornire una metrica di incertezza nelle previsioni. L'obiettivo è ridurre i drive test fisici fino al 70% prevedendo la qualità del segnale in località non misurate.

- **Enhanced MDT-Based Performance Estimation for AI Driven Optimization in Future Cellular Networks** [4]
    - Questo articolo si concentra sui rapporti MDT e affronta tre principali tipi di errori nella stima della copertura basata su MDT: errore di posizionamento, errore di quantizzazione e scarsità di rapporti utente (data sparsity). La sua distinzione principale è la caratterizzazione congiunta di tutti e tre gli errori interdipendenti per la prima volta, inclusa l'analisi della loro interazione. Propone un framework per quantificare questi errori, determinarne le distribuzioni e trovare una larghezza ottimale del bin, che minimizzi l'errore complessivo nella stima della copertura basata su MDT, fondamentale per l'automazione della rete basata sull'AI.

- **Empirical Formula for Propagation Loss in Land Mobile Radio Services** [5]
    - Questo articolo deriva una formula empirica per la perdita di propagazione nelle aree urbane per utilizzare computazionalmente il metodo di previsione della propagazione di Okumura. La formula è presentata come A + B log R, dove A e B sono funzioni della frequenza e dell'altezza dell'antenna, e R è la distanza. La sua distinzione chiave è che fornisce un modello matematico empirico (modello Hata) basato su risultati sperimentali per la previsione della forza di campo mediana di base e della perdita di propagazione nei servizi radio mobili terrestri in condizioni specifiche. Include anche fattori di correzione per aree suburbane e aperte e per le altezze delle antenne veicolari.

Questi quindi erano gli articoli selezionati durante la ricerca. Possiamo notare come condividano diversi concetti tra loro, come ad esempio la predizione dell'RSRP, l'uso di MDT, modelli teorici come quelli basati sulle formule di Okumura... tuttavia, nessuno di questi è adatto come base per il nostro lavoro. Infatti, escludendo il secondo articolo [3], di cui parleremo nel paragrafo successivo, gli altri hanno tutti task diversi dal nostro. Nel primo caso, viene proposta una tecnica per arricchire i dati MDT usando metodi di learning non supervisionato, mentre nel terzo viene mostrato come trattare gli errori nella stima della copertura. Nell'ultimo infine si ricerca un approccio totalmente teorico per il calcolo della funzione di perdita. Il nostro obiettivo invece è quello di concentrarci sulla predizione della copertura radio mobile, da un punto di vista pratico e specifico per i dati forniti dall'azienda.

3.2 Base di partenza

Il paper **Drive test minimization using Deep Learning with Bayesian approximation** [3] sembrava un ottimo punto da cui partire. Pertanto, andando a considerare il codice github associato ci siamo accorti che quest'ultimo era basato su un paper più recente, che è proprio quello che abbiamo deciso di usare. Si tratta di **Model-aided deep learning method for path loss prediction in mobile communication systems at 2.6 GHz** di Thrane, Jakob, Zibar, Darko, Christiansen, Henrik Lehrmann [6].

Questo ha l'obiettivo di predire l'RSRP, cercando di prevedere nel modo più preciso possibile la path loss. Ha molti punti in comune con l'altro paper, meno aggiornato: infatti, rimane inalterato l'uso di drive test, oppure la struttura del modello, formato da una rete neurale composita. Tuttavia, aggiunge diversi dettagli, utili per risultati migliori e più precisi. Innanzitutto, estende l'analisi a 2630 MHz, frequenza più critica per le nuove generazioni di reti mobili come il 5G. Aggiunge un modello teorico di supporto, in modo che l'algoritmo venga aiutato nel compito di imparare il fenomeno della perdita di percorso. Inoltre, sfrutta tecniche di machine learning aggiuntive per ottimizzare i risultati, come ad esempio data augmentation.

In sintesi, mentre il primo articolo ha gettato le basi sull'uso del Deep Learning con dati di drive test per la previsione della qualità del segnale e l'introduzione dell'incertezza Bayesiana, il paper successivo rappresenta un'evoluzione concentrandosi specificamente sulla previsione del path loss, introducendo l'importante concetto di apprendimento assistito da modello e ampliando l'analisi a frequenze più alte e rilevanti per il 5G, con una forte enfasi sulla generalizzazione e l'estrazione automatica di feature dalle immagini satellitari.

4 Modello

In questa sezione mostriamo la struttura dettagliata dell'algoritmo, partendo innanzitutto dal modello presentato dal paper scientifico su cui ci siamo basati (Model-aided deep learning method for path loss prediction in mobile communication systems at 2.6 GHz), passando poi alle modifiche implementate. Per completezza, specifico che l'algoritmo è scritto in python e la rete neurale è basata principalmente sulle librerie di pytorch.

4.1 Modello di partenza

Come detto precedentemente, siamo partiti dal modello proposto dall'articolo in [6].

Questo ha come scopo la predizione di RSRP ed RSRQ, partendo da dati ottenuti con Drive Test. Ciò è sufficiente come base per il nostro lavoro, in quanto partiamo da dati MDT e vogliamo predire le stesse labels dell'articolo.

4.1.1 Dati ed input

I ricercatori del Department of Photonics Engineering della Technical University of Denmark hanno effettuato un Drive Test in un'area di circa 1.4 km^2 intorno all'università, nel comune di Kgs. Lyngby in Danimarca.

Le misure sono state effettuate con l'attrezzatura Rohde & Schwarz TSMW Universal Network Analyzer ed il software ROMES, i quali consentono di catturare misure di qualità di segnale con posizione GPS annessa. Vengono

considerate 3 celle, PCI 64, 65, 302; le prime due a frequenza 811 MHz e l'altra a 2630 MHz.
Ad ogni misura è stata associata un'immagine satellitare generata da un'API REST di immagini statiche, tramite il servizio Mapbox. Queste hanno risoluzione 256 x 256 pixel, un livello di zoom 17, che corrisponde a 0.75 metri per pixel, e pertanto coprono un'area di circa 185 x 185 metri.
Le misurazioni con l'aggiunta delle immagini costituiscono il dataset di input per il modello, formato da circa 60000 dati, di cui 7000 selezionati per il test. Nel concreto, l'input è definito da due file csv, uno per le features, features_matrix.csv, e l'altro per le labels, output_matrix.csv. Nel primo le caratteristiche usate sono le seguenti: latitudine e longitudine del punto, distanza orizzontale, verticale e complessiva tra il punto e la stazione radio base, l'identificativo della cella del punto; viene inoltre usata la tecnica del one hot encoding sull'identificativo delle celle, al fine di rendere l'algoritmo in grado di discernere in quale cella si trovi il punto: infatti, vengono proposte 3 colonne, ognuna corrispondente ad una cella, e, per ogni punto, in ogni colonna troviamo 1 se il punto appartiene a tale cella e 0 se non vi appartiene. Questo è utile per associare al dato la frequenza del segnale.
Nel csv di output invece troviamo RSRP ed RSRQ.

4.1.2 Algoritmo

Innanzitutto, viene elaborato l'input in modo da suddividere i dati in due sezioni, una dedicata al training e una alla validazione. Lo split dei dati proposto dal paper non è canonico, in quanto sfrutta il fatto che i dati derivano da un drive test: le misure sono state prese su strada nel centro

abitato di Kongens Lyngby, dove è stato sviluppato questo algoritmo; pertanto, è stato creato un box di territorio geografico intorno all'università e tutti i punti localizzati in quell'area sono stati usati per la validazione. Le restanti misure sono state usate per il training.

I due nuovi set di dati vengono quindi normalizzati e salvati nei rispettivi files .npy. Ad ogni punto di input inoltre è associata un'immagine satellitare centrata alle coordinate del punto. Queste sono utili al modello per comprendere la conformazione geografica del territorio. Generati i file, inizia l'allenamento del modello. Viene usata una rete neurale composita: una parte della rete è un MLP che legge i dati numerici, l'altra è una CNN per l'analisi delle immagini. Infine, i due output vengono sommati insieme per ottenere l'output finale.

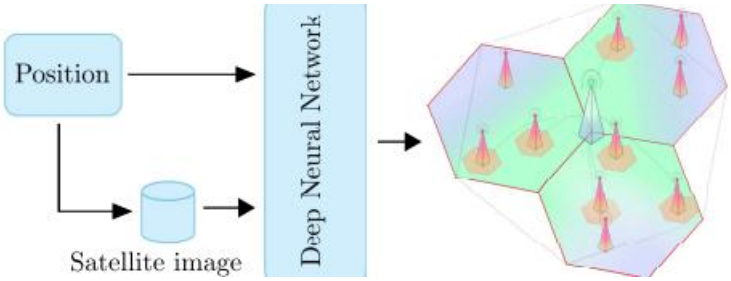

Figura 11: Schema concettuale del funzionamento della rete composita. A destra la rappresentazione schematica di antenne, celle e dispositivi mobili.

Il MLP è composto da blocchi di funzioni (sono impostati 2 blocchi di default ma è inizializzabile ad ogni simulazione): ogni blocco è formato da una trasformazione lineare, una funzione di attivazione ReLU e una normalizzazione dello strato risultante (batch normalization). Gli strati sono impostati a 200 neuroni.

La CNN anch'essa è formata da blocchi in base alla scelta del numero di layers convoluzionali; ogni blocco è composto da una convoluzione, da una funzione LeakyReLU, dalla normalizzazione dello strato ed infine da una trasformazione MaxPooling.

È stato impostato di default 1 solo canale. Pertanto, le immagini vengono trasformate e considerate in scale di grigio. Il numero di convoluzioni si basa sulla lunghezza del vettore del numero di filtri. Ogni valore di questo vettore indica il numero di filtri usati in ogni convoluzione che determinano la dimensione tridimensionale della nuova configurazione.

[200, 100, 50, 25, 12, 1] questo è un esempio di inizializzazione del vettore del numero di filtri: notiamo quindi che le convoluzioni saranno 6.

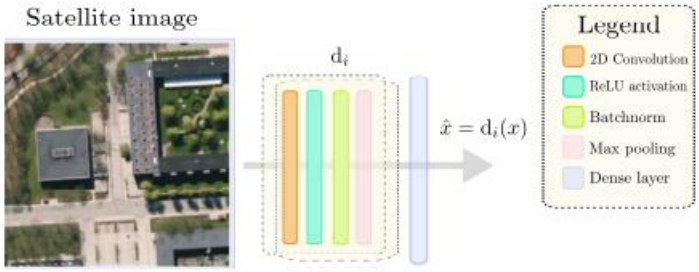

Figura 12: Schema rappresentativo del funzionamento della CNN per singola immagine

L'input, inoltre, prima di venire analizzato dalla rete viene arricchito con un ulteriore feature: si tratta della path loss, che corrisponde all'attenuazione del segnale radio. Questa viene calcolata attraverso le equazioni del modello UMa_B [1], il quale è stato sviluppato per gestire scenari di comunicazione wireless in ambienti urbani macro-cellulari. L'obiettivo è quello di

indirizzare il modello, in modo che non debba imparare l'intero fenomeno della perdita di percorso da zero.

Innanzitutto, viene calcolato un output derivante dall'analisi sinergica tra features e immagini:

$$Y_1 = F + I \tag{15}$$

dove Y_1 è l'output complessivo, F il contributo del MLP, I quello della CNN. In seguito, questo output, essendo ancora tensoriale, viene sottoposto ad una seconda rete neurale, basata su un singolo blocco di MLP, in modo da ottenere uno scalare, e viene sommata alla pathloss:

$$Y = P + NN_2(Y_1) \tag{16}$$

dove P è il contributo derivante dal modello di supporto, ricavato tramite la conversione della path loss in RSRP ed $NN_2(Y_1)$ è il contributo della seconda rete neurale. La pathloss in particolare è ottenuta come:

$$PL = 13.54 + 39.08\ \log_{10}(d3d) + 20\log_{10}(fc) - 0.6(h_t - 1.5) \tag{17}$$

equazione già descritta in precedenza nella sezione 2.1.3. Nota la pathloss, è possibile stimare la potenza ricevuta al terminale, P_{rx}, come:

$$P_{rx} = P_{tx} - PL + offset \tag{18}$$

dove P_{tx} è la potenza di trasmissione della base station e offset è un parametro che tiene conto dei guadagni delle antenne in ricezione e trasmissione, di eventuali perdite lungo i cavi d'antenna, e di perdite legate all'implementazione hardware di ricevitore e trasmettitore. Nota la potenza

complessiva ricevuta dal terminale, è possibile calcolare la potenza associata al segnale di riferimento come:

$$P_{rsrp} = P_{rx} - 10\log_{10}(12N) \tag{19}$$

Dove N dipende dalla banda del segnale trasmesso, e per una trasmissione LTE a 20MHz è pari a N= 100. Il valore di RSRP così stimato viene quindi normalizzato usando la media (μ_{rsrp}) e la deviazione standard (σ_{rsrp}) del valore di RSRP osservato nelle misure, ottenendo:

$$P = \frac{P_{rsrp} - \mu_{rsrp}}{\sigma_{rsrp}} \tag{20}$$

In Figura 13 è mostrato come il valore stimato dal modello analitico per il calcolo della path loss viene usato per ricavare una feature aggiuntiva da passare alla rete neurale, che di fatto andrà quindi a calcolare un fattore correttivo da applicare alla stima prodotta dal modello propagativo analitico.

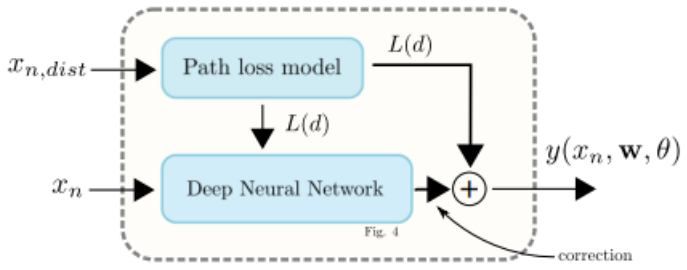

Figura 13: Schema rappresentativo dell'aggiunta del modello di supporto

In Figura 14 viene mostrata la composizione di MLP e CNN in un unico output, il quale viene passato ad una seconda rete neurale per ottenere uno scalare.

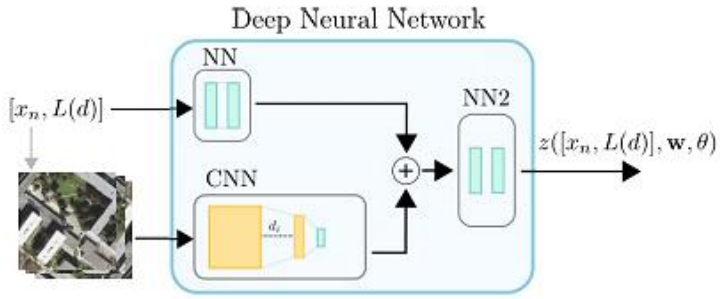

Figura 14: Schema riassuntivo del funzionamento della rete complessiva

Per quanto riguarda l'allenamento, in quanto algoritmo con task regressione, è stata scelta come loss la MSELoss (Mean Square Error Loss), mentre come ottimizzatore è stato usato ADAM. Inoltre, è implementata anche la funzione ReduceLROnPlateau, la quale serve a diminuire il learning rate (con una certa pazienza) per raffinare l'ottimizzazione.

4.1.3 Struttura del modello

Come già detto in precedenza, il modello si basa sul codice presentato dal GitHub "jakthra/PathLossPredictionSatelliteImages" [7] associato al paper di riferimento. In questo paragrafo presento brevemente i files interni al GitHub, in modo da avere un'idea chiara del materiale a disposizione su cui lavorare.

- README.md: questo è il file introduttivo, il quale fornisce le linee guida su come inizializzare ed usare l'algoritmo.
- requirements.txt: file di testo nel quale sono riportate tutte le librerie python necessarie al funzionamento del codice.

- fileGen.py: contenitore delle funzioni per lo split dei dati e la generazione dei files formato .npy.

- drive_test_route_journal.py: direttamente correlato a FileGen.py, contiene al suo interno la descrizione dettagliata della funzione dello split dei dati basato su drive test.

- generate_training_test.py: vengono definiti i csv e la cartella delle immagini da usare nella rete. Inoltre, utilizza FileGen.py per generare i dati splittati.

- raw_data: cartella contenente il dataset, composto da csv ed immagini.

- dataset_factory.py: carica i file numpy generati da generate_training_test.py e definisce le features e le labels da utilizzare nella rete. Inoltre, crea le matrici di input, applicando le trasformazioni necessarie, come data_augmentation o resize dell'immagine.

- model.py: al suo interno è presente l'intera rete neurale, con la definizione delle classi e delle funzioni della CNN, del MLP e del supporto teorico.

- pathloss_38901.py: viene definita la funzione per il calcolo della path loss prevista da Uma_b.

- train.py: il file adibito all'allenamento del modello. Inizializza la rete, definisce gli iperparametri e la loss. Svolge il training e la valutazione.

- predict.py: utile a valutare i pesi e la precisione del modello.

Il codice è programmato interamente in linguaggio python e per la rete sono state usate le funzioni della libreria Pytorch [8].

4.2 Implementazione modifiche

Per rendere il modello compatibile con i dati forniti da TIM è stato necessario apportare alcune modifiche al modello.

4.2.1 Split dei dati

La prima modifica riguarda lo split dei dati nei due set, training e test; questa divisione non è canonica e non è adatta a descrivere i dati dell'azienda: sarebbe stato necessario definire ogni volta un box di coordinate geografiche nei pressi di alcune misure, senza un motivo valido o un qualche criterio. Pertanto, inizialmente è stato definito un metodo di split canonico e generico, basato sull'assegnazione al training set dell'80% dei dati e al test set del 20%, scelti casualmente con probabilità uniforme.

Abbiamo deciso, tuttavia, di ricorrere ad un'ulteriore modifica per questo metodo; infatti, ci siamo accorti che, basandoci su dati con feature spaziali, potrebbe verificarsi il fenomeno del leakage spaziale: se le misure sono molto vicine tra di loro spazialmente, l'algoritmo potrebbe notare una relazione spaziale tra i dati e adattarsi ad essi senza generalizzare. Pertanto, abbiamo definito un metodo che seleziona casualmente i dati, con le stesse proporzioni precedenti, ma aggiunge un vincolo sui dati di test; ognuno di questi, infatti, quando viene scelto, genera intorno a sé un'area circolare (raggio definito nell'inizializzazione) dentro cui non possono essere presi altri dati di test. Se uno di essi viene selezionato dentro quest'area viene scartato.

4.2.2 One hot encoding

Ci siamo interrogati sulla motivazione che ha portato gli autori del paper [6] ad applicare questa tecnica in questo modello per indicare l'identificativo di cella; infatti, le posizioni dei punti e le distanze dalle antenne erano già delle informazioni sufficienti per stimare l'RSRP. Inoltre, l'identificativo della cella non è un'informazione essenziale ed utile per questa stima.

Analizzando più in profondità il codice, ci siamo accorti che questa tecnica viene usata dall'algoritmo per assegnare diverse frequenze ai punti: ogni cella, infatti, ha una stazione radio base che serve il segnale ad una determinata frequenza. Queste vengono poi usate nella stima della path loss. Per quanto riguarda i dati raccolti nelle misure MDT di TIM, però, sono state scelte antenne che trasmettono segnale radio alla stessa frequenza portante; pertanto, abbiamo deciso di togliere il one hot encoding dell'identificativo di cella e modificare il codice in modo che usi nel calcolo della pathloss sempre la stessa frequenza.

5 Metodologia

5.1 Setup iniziale

La prima scelta da fare per questo lavoro è stata quella di decidere su quale ambiente programmare l'algoritmo. Essendone particolarmente pratico, la prima opzione è stata quella di Google Colab.

Ci sono state diverse difficoltà con questo approccio: innanzitutto, il modello originale usava librerie adatte ad una versione di python obsoleta, la 3.10; è stato pertanto necessario l'uso di un ambiente virtuale conda, su cui installare le varie librerie; inoltre, alcune di queste avevano nomi differenti da quelli riportati sul file requirements.txt, presentato su GitHub. Infine, la GPU concessa dalla versione gratuita di Google Colab si è rivelata incompatibile con il codice e probabilmente anche insufficiente dal punto di vista computazionale (questo punto non è stato analizzato con eccessiva profondità). Abbiamo deciso quindi di optare per l'utilizzo delle macchine virtuali di laboratorio TIM. Qui, grazie all'uso di GPU più potenti e Visual Studio Code, siamo riusciti a superare i precedenti problemi.

5.2 Dataset ed input

I dati forniti da TIM sono stati ottenuti con la tecnica MDT. Sono stati suddivisi in diversi dataframe, in quanto sono state condotte campagne in tempi e luoghi diversi. In particolare, in questa trattazione sono stati usati i dataset con punti localizzati nei pressi di Perugia e di Bologna: per il primo sono stati considerati circa 40000 punti, mentre per il secondo circa 7000000, i quali ovviamente sono stati filtrati.

Come spiegato nel capitolo inerente al modello, l'input ha una forma particolare; pertanto, per non apportare ulteriori modifiche al codice, è stato

necessario adattare il dataset a quello proposto dal paper: dai dati MDT sono state prese le coordinate dei punti, gli identificativi delle celle e le misure di RSRP ed RSRQ (come labels). Per ottenere le distanze dalle antenne, è stato usato un algoritmo esterno al modello: questo considera l'identificativo della cella e la frequenza del segnale e conduce una ricerca in un inventory TIM, contenente questi identificativi; quando la cella viene trovata, vengono mostrate la posizione della stazione radio base e le celle adiacenti con le relative coordinate.

Per quanto riguarda le immagini, queste non sono state fornite dall'azienda; è stato pertanto necessario ricercare un generatore di immagini satellitari da associare alle coordinate dei punti. Abbiamo osservato che diversi siti offrivano la possibilità di ottenere immagini satellitari o ancora meglio immagini ortofoto (quest'ultime sono foto fatte da aereo e sono molto simili a quelle proposte nel paper). Tuttavia, era necessario tenere in considerazione 2 vincoli: ci serviva prima di tutto un generatore opensource gratuito e in secondo luogo che desse la possibilità di accedere a librerie python per iterare la procedura di generazione. L'unico riscontro possibile è stato OpenStreetMap [9], un sito opensource che supporta la libreria python staticmap.

Le immagini generate hanno dimensione 300 x 300 pixels e coprono un diametro di circa 1.5 km (si basano su uno zoom di 15 che corrisponde ~4.78 m/pixel). Verranno poi trasformate dal modello in formato 256 x 256 pixels. In Figura 15 è possibile osservare degli esempi della tipologia di immagine che è possibile ottenere.

Figura 15: Esempi di immagini prodotte da OpenStreetMap per mezzo di staticmap

5.3 Training

Creato il dataset, è giunto il momento dell'allenamento dell'algoritmo.

Innanzitutto, avevamo bisogno di controllare che stesse funzionando tutto correttamente: pertanto, abbiamo provato a fare training sui dati proposti dal paper. Abbiamo provato con 1, 10, 30 e 80 epoche. Fatto ciò, abbiamo iniziato ad usare i dati TIM, partendo dal dataset di Perugia. Anche qui abbiamo usato lo stesso numero di epoche nei vari tentativi. La label da predire in ogni prova era l'RSRP, più facile da ottenere con le features specificate.

Abbiamo fatto poi un tentativo anche per stimare l'RSRQ. Già immaginavamo scarsi risultati, in quanto le features di input difficilmente erano sufficienti per definire la qualità del segnale; sarebbero state necessarie informazioni sulle interferenze. Infatti, anche lavorando sul learning rate e provando a ridurlo, i risultati non sono stati soddisfacenti.

In seguito, abbiamo provato ad escludere le immagini, in quanto abbiamo pensato che, basandoci su un modello già preimpostato, non avevamo la certezza che queste fossero davvero necessarie per stimare l'RSRP. Provando entrambi i casi, con e senza immagini, abbiamo ottenuto risultati molti simili con una percentuale di errore aumentata di un fattore trascurabile. Tuttavia, considerando più a fondo il problema, ci siamo accorti che il nostro dataset era composto solo da punti che ricevevano segnale con frequenza 800 MHz: questa frequenza è tale da riuscire facilmente a penetrare gli ostacoli presenti nel territorio; la conformazione del territorio diventa, quindi, meno rilevante e di conseguenza anche le immagini, che sono il modo in cui l'algoritmo la può valutare, perdono di significato. Bisogna però a questo punto chiedersi cosa succede se usiamo frequenze più alte, come 2660 MHz, che hanno un livello di penetrazione più basso rispetto a 800 MHz.

Oltre al dataset di Perugia, TIM mi ha dato la possibilità di accedere ad enormi dataset incentrati su Bologna. Qui i punti hanno frequenze variegate e sono presenti, ad esempio, anche numerose misure con frequenze 2660 MHz. Detto questo, la prima cosa da fare è stata analizzare il csv. Abbiamo dovuto usare una libreria più potente rispetto al classico Pandas [10], ovvero Polars [11], adatta appunto per dataframe di grandi dimensioni. Quindi abbiamo filtrato il dataset prendendo soltanto le misure a frequenza 2660 MHz, e tra queste abbiamo poi selezionato solo le prime 3 celle più popolose.

Visualizzando i punti con la libreria Folium [12], tuttavia, si nota che sono molto vicini tra loro a gruppi. Ho quindi pensato che questo comportamento potesse influenzare la rete: l'algoritmo, infatti, potrebbe generalizzare male, pensando che la vicinanza tra i punti sia un qualche pattern di cui tenere conto. Conducendo il training, infatti, si notano performance della loss inusuali; l'errore sul train set risulta maggiore rispetto al test set, situazione completamente anomala.

Il primo approccio che ho deciso di tentare allora è stato quello di cambiare lo split per considerare questo leakage spaziale.

Si è quindi tornati ad analizzare la questione sulle immagini. Come ci aspettavamo, qui diventano importanti: infatti, togliendole dall'allenamento, si giunge a un'overfitting troppo marcato per poterlo gestire. Con le immagini, invece, si arriva a dei risultati accettabili, anche se non ai livelli del modello con misure a 800 MHz.

Tuttavia, analizzando più approfonditamente il dataset, mi sono accorto che alcuni punti corrispondevano allo stesso identico terminale e differivano soltanto di una distanza temporale di alcuni millisecondi, per cui la posizione spaziale rimaneva invariata. Ecco spiegato il motivo perché nella visualizzazione si osservavano zone con migliaia di punti, ammassati in uno spazio che fisicamente non poteva contenerli tutti. Per risolvere questo problema, ho deciso di ripartire dal dataset originale di Bologna e filtrare i dati sempre sulla stessa frequenza, ma considerando solo le coppie di latitudine e longitudine uniche: in caso di gruppi con la stessa coordinata, i valori di RSRP vengono mediati per ottenere un unico valore. Vengono quindi selezionate le dieci celle più popolose, per un totale di 78000 punti.

A questo punto viene condotta nuovamente l'analisi della rete con e senza immagini: i risultati, purtroppo sono stati diversi dalle nostre aspettative. Infatti, le performance della rete sono sufficientemente buone, tuttavia non si rivela una differenza sostanziale tra l'uso di sole features e modello completo. Gli autori dell'articolo di riferimento affermano che le immagini servono sia per valutare la conformazione del territorio, ma anche come ausilio per discernere tra punti molto vicini tra loro e migliorare la generalizzazione: è probabile quindi che avendo tolto la vicinanza spaziale e temporale tra i punti, le immagini potrebbero non avere più un impatto così grande nell'ottimizzazione della rete. Lo studio di questo problema è attualmente in corso: sono state prodotte mappe di visualizzazione con folium e saliencymap, ma non sono ancora state formulate ipotesi solide.

Per tentare di migliorare le performance complessive del modello, sono stati condotti diversi tentativi con vari approcci: il primo approccio è stato quello di provare a stabilizzare meglio la loss del test modificando i parametri; tuttavia, aumentare il learning rate, la regolarizzazione o il numero di epoche ha sortito solo effetti trascurabili sulla diminuzione dell'errore. Un altro

approccio è stato quello di lavorare sul modello fisico. Come detto in precedenza, viene calcolata una path loss per aiutare l'algoritmo a comprendere il fenomeno dell'attenuazione del segnale. Viene quindi definita un'equazione per la perdita di segnale, la quale contiene al suo interno un offset, definito come iperparametro. Pertanto, l'idea è stata quella di provare prima a togliere l'offset e poi provare a disattivare il modello fisico stesso, aggiungendo però alle features la frequenza dei punti: l'obiettivo è quello di lasciare al modello il lavoro di capire il fenomeno della perdita da zero, ma aiutandolo con l'informazione della frequenza. Purtroppo, anche queste strade hanno portato a risultati trascurabili.

5.4 Stima dell'RSRQ

Un altro task del modello è la predizione dell'RSRQ, che rappresenta la qualità del segnale. Come già detto in precedenza, si tratta di una stima più complessa rispetto a quella dell'RSRP: infatti, per valutare la qualità del segnale avremmo bisogno di informazioni inerenti al rumore e alle interferenze nel segnale. La strada corretta potrebbe quindi essere quella di aggiungere features. Attraverso l'algoritmo esterno, già usato per ottenere la distanza del punto dalla stazione radio base servente, si possono ottenere le posizioni delle antenne delle celle adiacenti. In questo modo siamo in grado di calcolare la distanza del punto di misura dalle possibili antenne interferenti. Queste distanze sono state aggiunte all'elenco delle features.

Inoltre, per aggiungere informazioni utili all'algoritmo, queste distanze sono state usate per calcolare le path loss delle antenne interferenti, che a loro volta sono state aggiunte come ulteriori features all'input.

Il primo approccio a questa metodologia è stato con il dataset di Perugia. Questo è composto da 3 celle servite da una sola antenna. Pertanto, sono risultate soltanto 4 possibili antenne adiacenti in un raggio di 5000 m. Tuttavia, 3 antenne trasmettevano a frequenza 1800 MHz e solo una a 800 MHz. Per interferire con il segnale è necessario che l'antenna servente e quelle interferenti siano alla stessa frequenza. Quindi, dal momento che il

dataset di Perugia contiene soltanto misure a 800 MHz, avremmo dovuto considerare soltanto una cella.

Di conseguenza, per avere più celle interferenti e quindi un caso più generico, si è deciso di utilizzare i dati di Bologna. Li ho filtrati su frequenza 1800 MHz e ho preso le tre celle con la molteplicità maggiore, ognuna servita da un'antenna diversa. Per ognuna di queste, ho identificato per mezzo dell'inventory 4 antenne adiacenti con la frequenza 1800 MHz: ho effettuato una ricerca nel raggio di 1250 m, visualizzando la lista degli identificativi dei nodi adiacenti e ne ho individuati 4 con frequenza 1800 MHz, guardando l'ultima lettera della stringa con cui vengono identificati nel database TIM; queste lettere, infatti, determinano la frequenza supportata a cui l'antenna trasmette:

- E = 800 MHz
- T = 1800 MHz
- L = 2600 MHz

Ottenute le coordinate, ho calcolato le 4 distanze interferenti per ogni punto mettendole in ordine crescente, in modo da avere le antenne interferenti ordinate dalla più vicina a quella più lontana. In conclusione, si ottiene un dataset da circa 60000 misure.

Preparato il dataset, è stato dato ad input al modello per la stima dell'RSRQ; è stata selezionata la modalità data-driven, nella quale vengono aggiunte le features della path loss e la somma delle path loss interferenti, ma non viene definita una correzione fisica da aggiungere all'output della rete composita.

Infatti, la path loss è direttamente correlata con l'RSRP ma non con l'RSRQ e pertanto sarebbe necessario pensare ad una formula totalmente diversa per definire una correzione fisica.

6 Risultati

In questa sezione, analizziamo i risultati ottenuti, ripercorrendo i passi visti nel capitolo della Metodologia.

Innanzitutto, elenchiamo gli iperparametri di default previsti dal modello originale:

- Learning rate (lr): 0.001
- Batch-size: 30
- Weight-decay: 0.00001

Ricordo brevemente lo scopo di ognuno di questi: il learning rate è il passo con cui i pesi vengono aggiornati durante la discesa del gradiente; serve a controllare quanto velocemente la rete si adatta ai dati. Il weight-decay è una forma di regolarizzazione e aiuta a prevenire l'overfitting. Infine, il batch-size indica quanti campioni vengono elaborati in ogni passo di addestramento.

6.1 Dataset paper

Il primo step è stato quello di valutare i risultati ottenuti dando ad input al modello i dati del paper.

È stato necessario trovare una metrica per capire chiaramente quanto l'algoritmo stesse sbagliando, concluso l'allenamento. Pertanto, abbiamo optato per l'errore relativo:

$$err = \frac{y - \hat{y}}{\hat{y}} \tag{21}$$

Dopodichè, facendo la media sui valori assoluti di ogni errore otteniamo l'errore relativo medio.

Abbiamo proceduto quindi all'allenamento della rete secondo diverse modalità: senza l'uso delle immagini, con l'uso delle immagini, per la predizione di RSRQ.

Nel primo caso, abbiamo ottenuto una buona dinamica, con un errore relativo medio del 3.3 %. In Figura 16 viene mostrata la dinamica di un batch di punti a confronto con i corrispettivi valori veri, in modo da avere una panoramica dei valori di RSRP predetti dal modello. È inoltre presente l'andamento dei punti (in arancione nella figura) in cui non è ancora stato sommato il contributo del modello fisico.

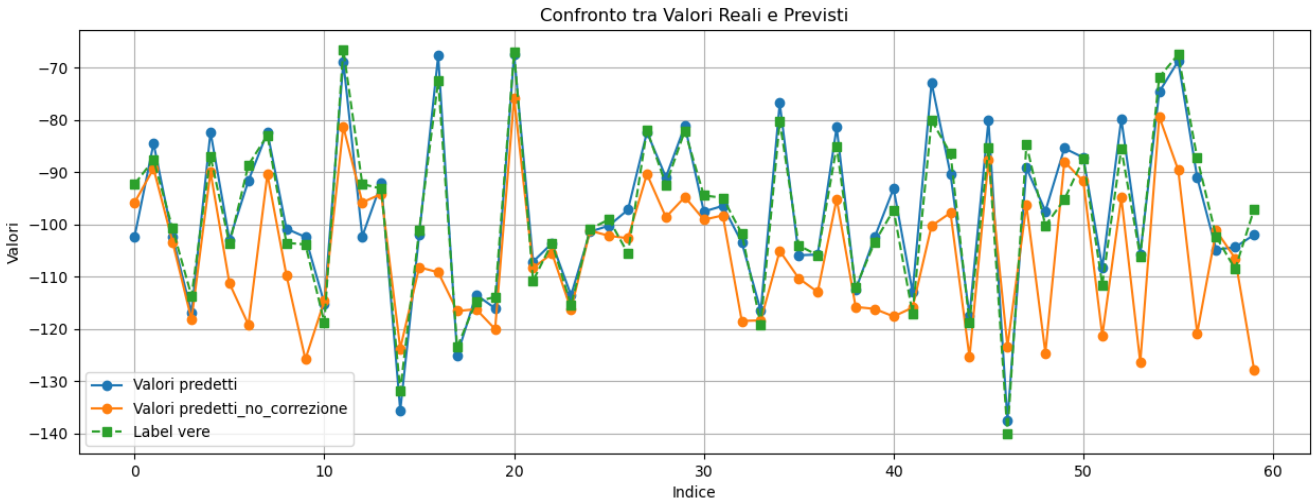

Figura 16: Dinamica di confronto tra valori predetti e labels vere per un batch di 60 punti per l'allenamento senza immagini con il dataset del paper

L’allenamento basato sul modello completo ha dato risultati ancora migliori rispetto al caso precedente: infatti otteniamo un errore relativo medio del 2.7 %. Notiamo quindi come le immagini forniscono un leggero miglioramento nelle performance.

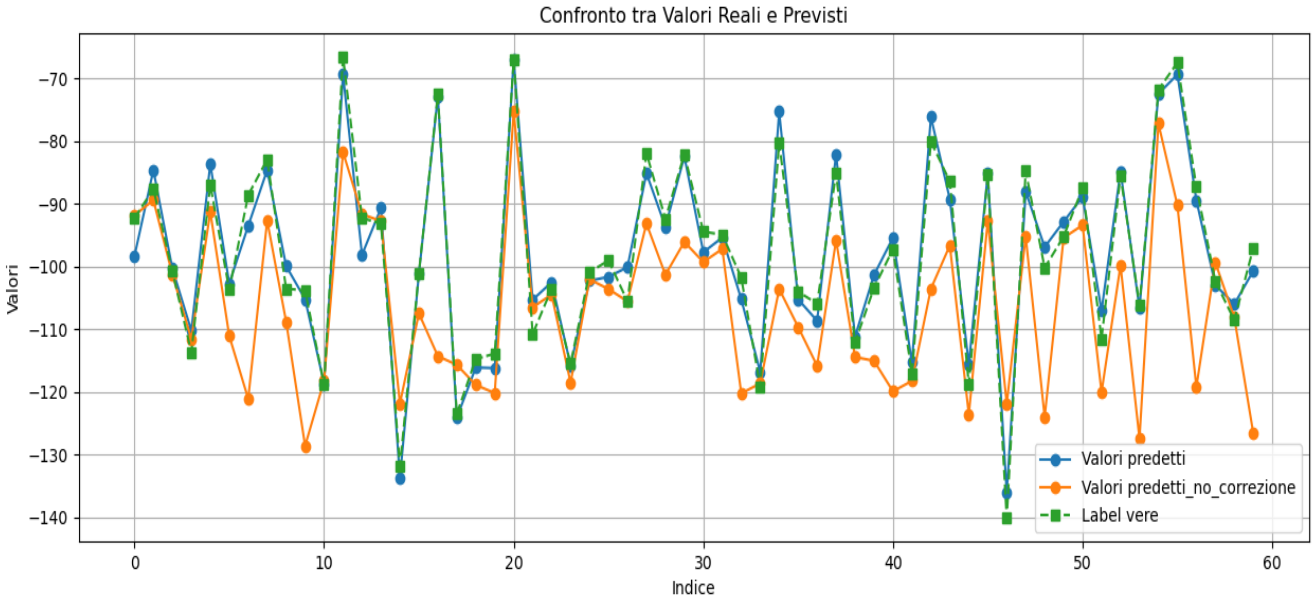

Figura 17: Dinamica di confronto tra valori predetti e labels vere per un batch di 60 punti per l’allenamento del modello completo con il dataset del paper

Per avere un ultimo confronto, abbiamo anche provato a predire l’RSRQ, usando il modello completo. Le performance risultanti sono buone, ma non soddisfacenti come le precedenti: otteniamo un errore relativo medio del 9.3%.

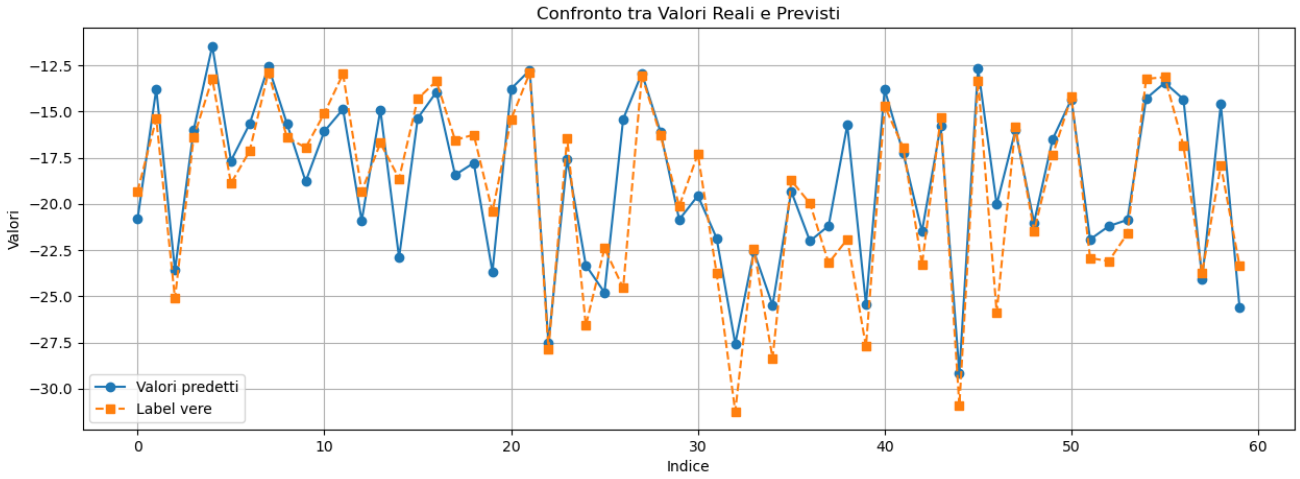

Figura 18: Dinamica di confronto tra valori predetti e labels vere per un batch di 60 punti per l'allenamento del modello completo per la predizione dell'RSRQ con il dataset del paper

Possiamo notare nella Figura 18 che, a differenza delle precedenti, non è presente l'andamento dei valori predetti con la correzione. Infatti, come abbiamo già detto, la path loss non è direttamente correlata all'RSRQ e quindi non possiamo calcolare un contributo basato su un modello teorico fisico da aggiungere all'output della rete. Pertanto, l'algoritmo viene allenato in modalità "data-driven" e quindi senza ulteriori correzioni.

6.2 RSRP – dataset Perugia

Abbiamo quindi fatto il training sul dataset di Perugia, ottenendo i risultati in Figura 19.

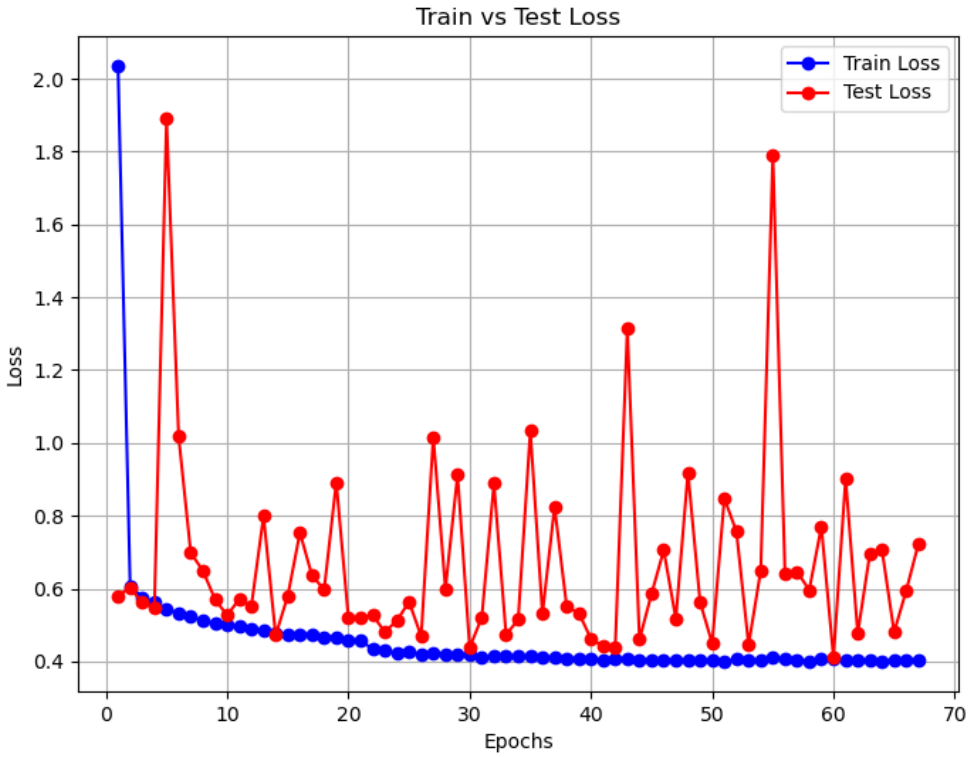

Figura 19: Loss 80 epoche per dataset Perugia (RSRP)

Si tratta del confronto tra la Loss del train con quella del test: possiamo osservare che il test non presenta un grande gap con il train, pertanto non notiamo una situazione di overfitting. È inoltre presente una certa oscillazione, tale da non attivare l'early stopping.

Per quanto riguarda l'errore relativo medio con il dataset di Perugia abbiamo ottenuto circa il 4.5%. In Figura 20 possiamo osservare la visualizzazione della dinamica di un batch di 30 punti.

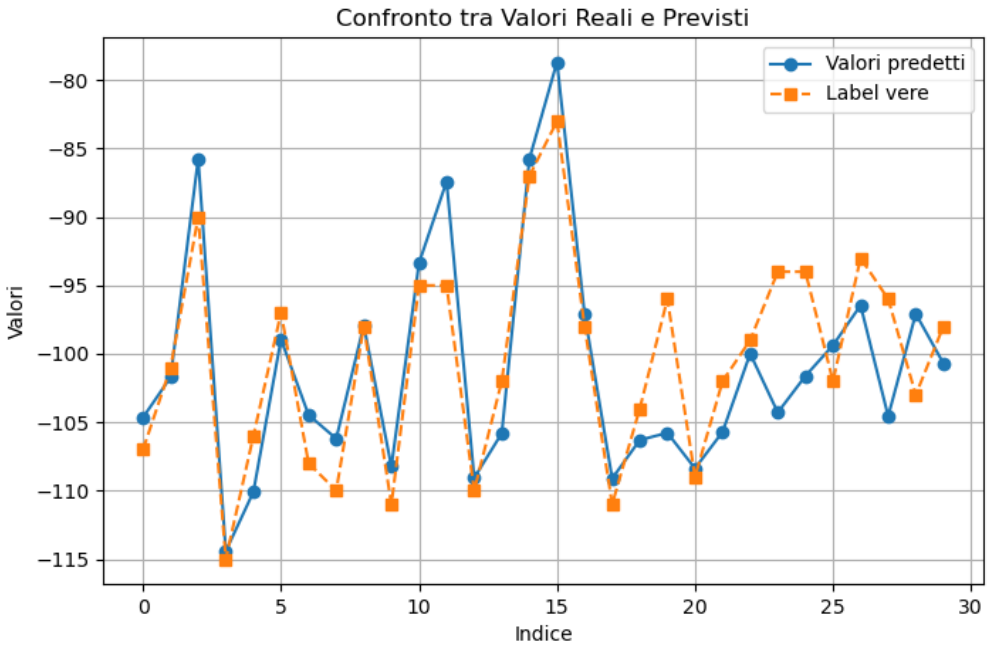

Figura 20: Visualizzazione del confronto tra i primi 30 punti del dataset con le label predette corrispondenti (dataset Perugia-RSRP)

6.3 RSRQ – dataset di Perugia

Con lo stesso dataset, abbiamo tentato di predire l'RSRQ, anche se con scarse attese. Infatti, si è ottenuto un errore relativo medio di circa il 13.2%.

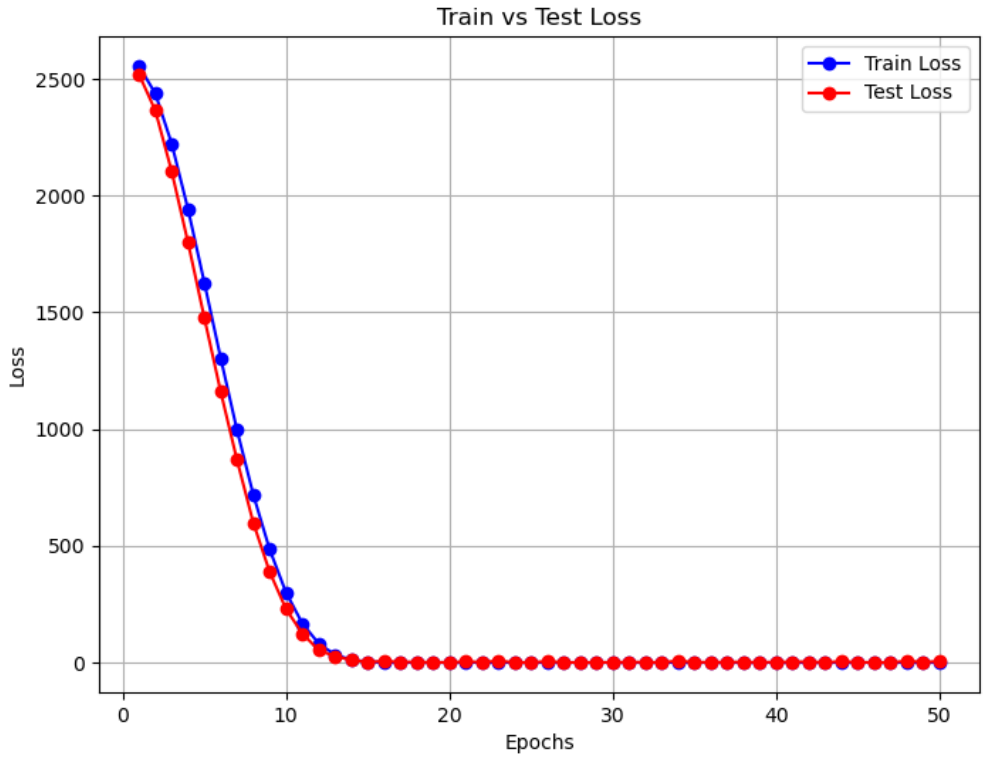

Figura 21: Loss 80 epoche per dataset Perugia (RSRQ)

Si osservi la loss riportata in Figura 21: non bisogna farsi ingannare dalla sovrapposizione degli andamenti di train e test: infatti, la scala è totalmente diversa da quella dell'RSRP. Se prima i valori finali di train e test erano di 0.4 e 0.7, qui sono di circa 5 e 8, ovvero molto maggiori.

Questo errore relativo medio così è in linea anche con quanto possiamo osservare nella Figura 22: infatti, è ben visibile come gli andamenti di valori predetti e valori veri siano molto diversi tra di loro.

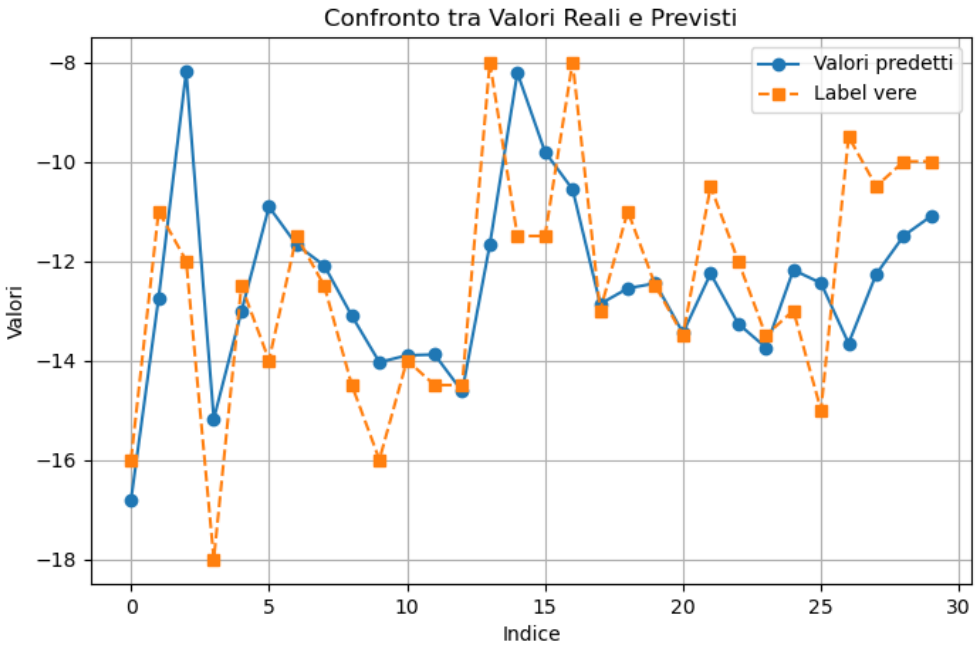

Figura 22: Visualizzazione del confronto tra i primi 30 punti del dataset con le label predette corrispondenti (dataset Perugia-RSRQ).

Abbiamo anche provato a ridurre il learning rate (da 0.001 a 0.0001 e 0.0003), oppure a modificare il batch-size, ma il risultato non ha portato ad una riduzione dell'errore relativo medio.

6.4 Dataset Bologna

Per quanto riguarda la questione immagini satellitari, come precedentemente detto, nel dataset di Perugia, essendoci solo punti che ricevono segnale a 800 MHz, ci aspettiamo che le immagini non siano fondamentali. Infatti, si è verificato che l'errore medio passa da 4.5% a 5.5%, che si ritiene non significativo per gli scopi del modello.

Per verificare l'utilità delle immagini rispetto alla modalità con sole features, abbiamo quindi usato il dataset di Bologna, filtrato per renderlo compatibile con il modello.

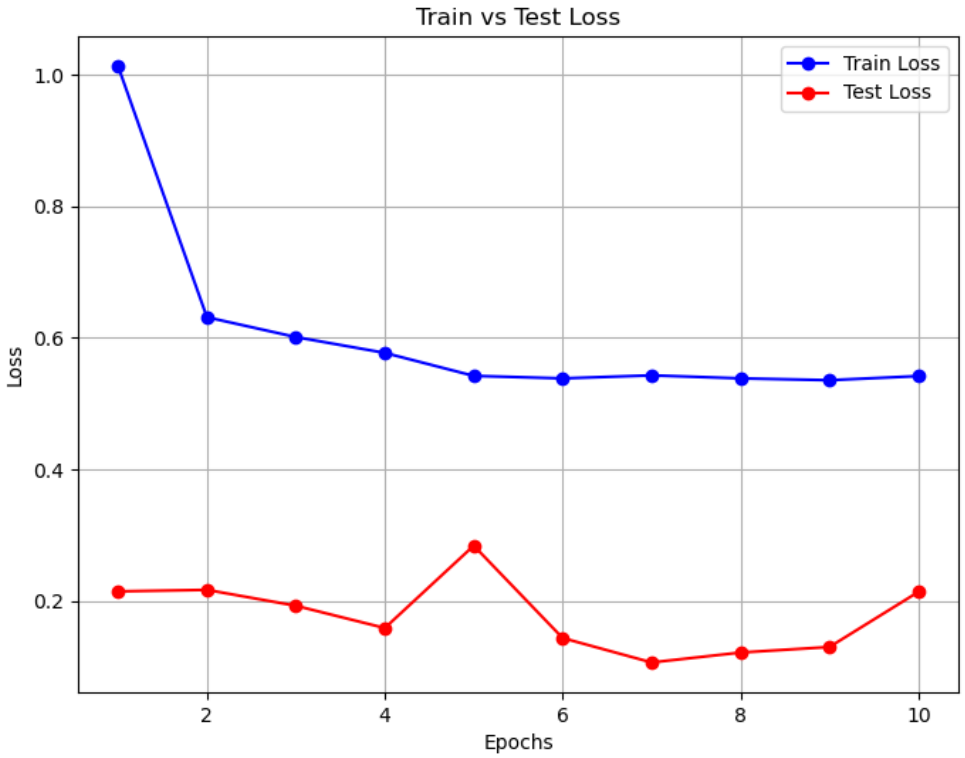

Figura 23: Loss per il primo tentativo con il dataset di Bologna (con immagini)

In Figura 23 possiamo osservare come la situazione degli andamenti sia anomala: la loss del train è superiore con un gap quasi costante rispetto al test.

Ho ipotizzato che il problema fosse il leakage spaziale: infatti implementando un algoritmo di split che esclude i punti vicini spazialmente, i dati selezionati vengono notevolmente ridotti.

In questa situazione, togliendo la funzionalità delle immagini, notiamo un overfitting molto marcato (provare a curarlo non ha prodotto risultati): ciò potrebbe significare che il modello si adatta troppo ai dati di training, senza generalizzare. Intuiamo quindi che il modello sta provando a compensare la mancanza di informazioni visive "inventando" correlazioni nei dati tabellari e che quindi manca una parte cruciale dell'informazione: in questo caso, la morfologia del territorio. Le immagini, quindi, potrebbero fungere da

regolarizzatore naturale: forniscono contesto spaziale e ambientale che aiuta il modello a generalizzare.

Infatti, con le immagini attive l'overfitting diminuisce molto e l'errore relativo risulta essere 6.8%. In Figura 24 e 25 è possibile osservare un esempio della bontà della previsione ottenuta in questo caso.

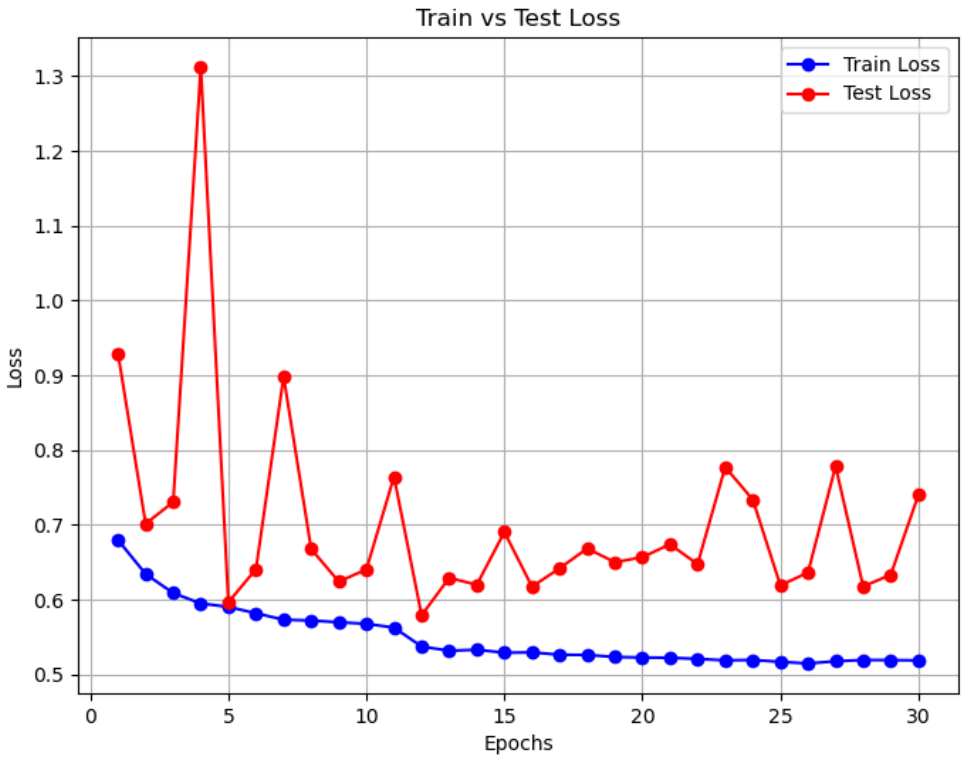

Figura 24: Loss 30 epoche per dataset di Bologna con immagini

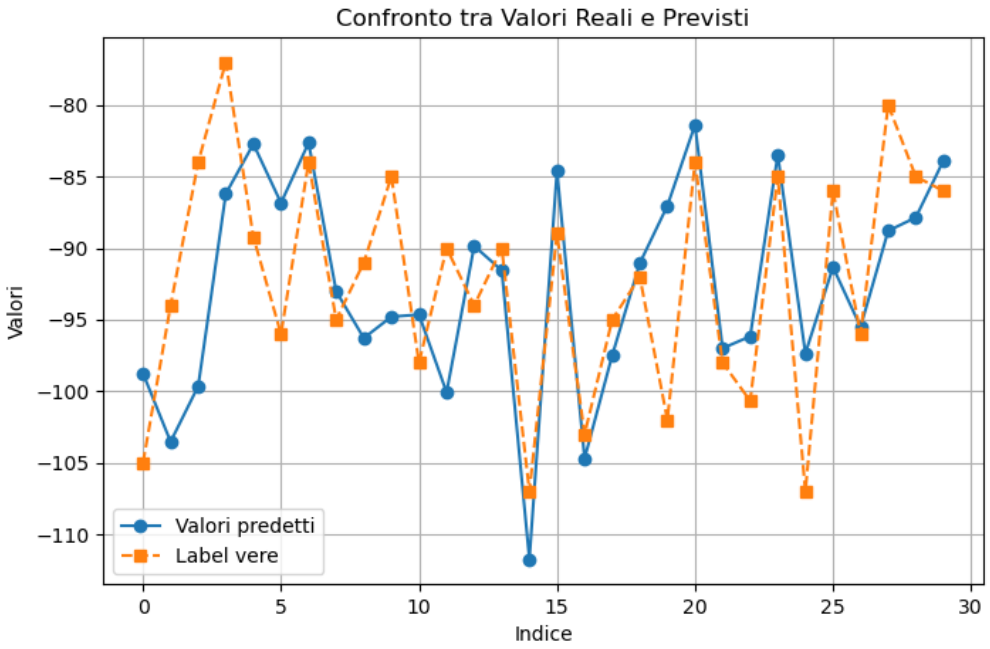

Figura 25: Visualizzazione del confronto tra i primi 30 punti del dataset con le label predette corrispondenti (dataset Bologna-RSRP).

Ispezionando più in profondità il dataset, anche grazie alla visualizzazione con la libreria folium, abbiamo scoperto che diversi punti erano differenti tra loro solo per la coordinata temporale e non spaziale: si osservavano infatti zone con migliaia di punti, ammassati in uno spazio che fisicamente non poteva contenerli tutti, come mostrato in Figura 26.

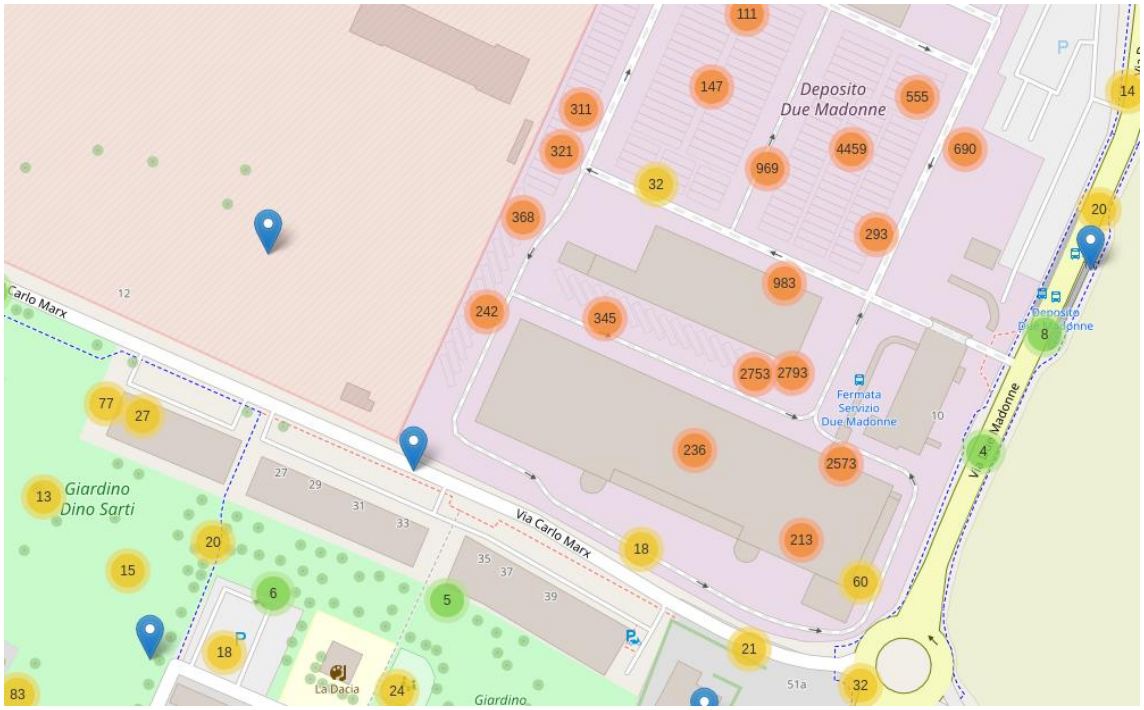

Figura 26: Visualizzazione di una porzione del dataset di Bologna con la libreria Folium. Gli indicatori blu sono singoli punti, mentre le aree numerate mostrano la molteplicità in quel punto.

Pertanto, è stato filtrato il dataset in modo da considerare solo i valori unici per ogni coppia di coordinate, prendendo la media dell'RSRP tra i gruppi di valori uguali. È stata mantenuta la frequenza a 2660 MHz.

Per quanto riguarda le performance, il modello completo ottiene risultati simili ai precedenti, come visibile in Figura 27, ovvero un errore relativo medio del 6.5%.

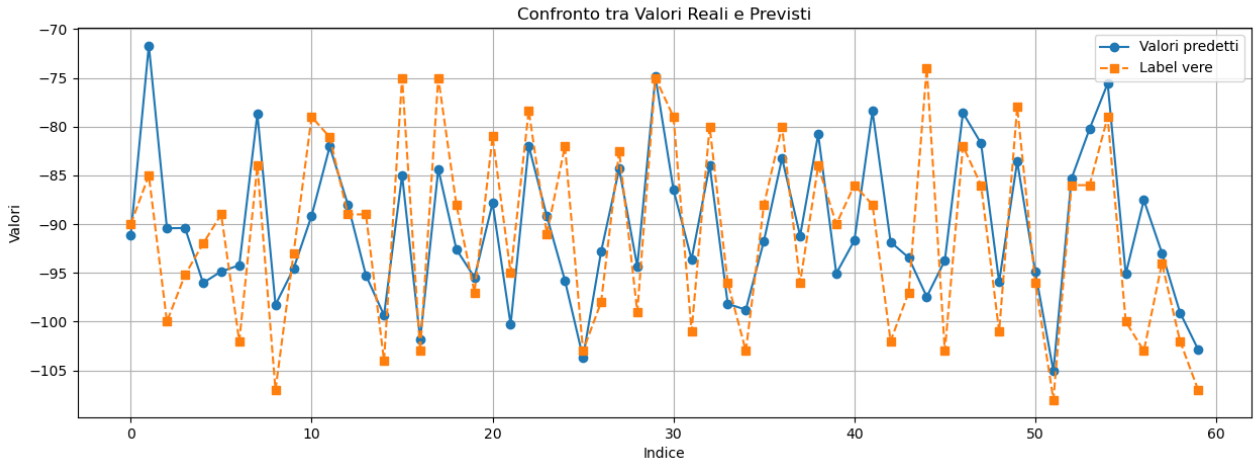

Figura 27: Visualizzazione del confronto tra i primi 60 punti del dataset con le label predette corrispondenti (dataset Bologna-RSRP).

Tuttavia, con queste correzioni il modello senza immagini non risente più di overfitting e ha errore del 7%. Pertanto, il peggioramento delle performance a causa dell'esclusione della CNN è molto ridotto rispetto al caso in cui si sfruttano le immagini, come anche è osservabile confrontando

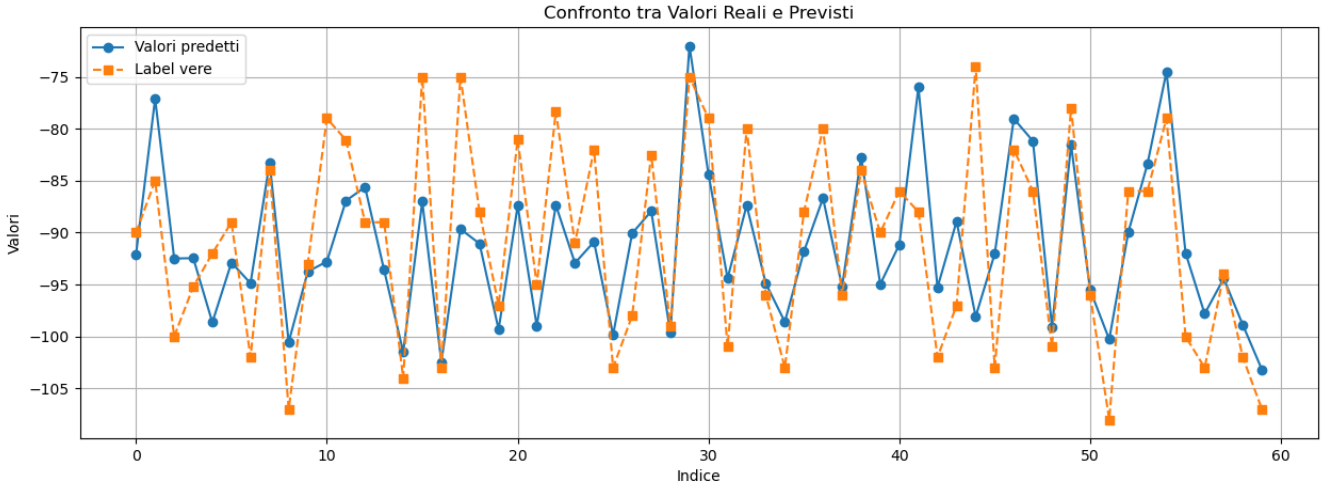

Figura 28: visualizzazione del confronto tra i primi 60 punti del dataset con le label predette corrispondenti per modello senza immagini (dataset Bologna-RSRP).

*gli andamenti in Figura27 e 28, ed è simile a quanto si era rilevato usando il dataset dell'articolo di riferimento.*Possiamo quindi ipotizzare che quanto detto dagli autori potrebbe essere corretto: le immagini sono molto utili sia per aiutare il modello a comprendere la conformazione del territorio ma soprattutto ad analizzare dataset con punti molto vicini tra di loro.

Per meglio comprendere l'impatto dell'uso delle immagini satellitati, abbiamo analizzato esempi di immagini del dataset con la tecnica della saliency map e della colormap.

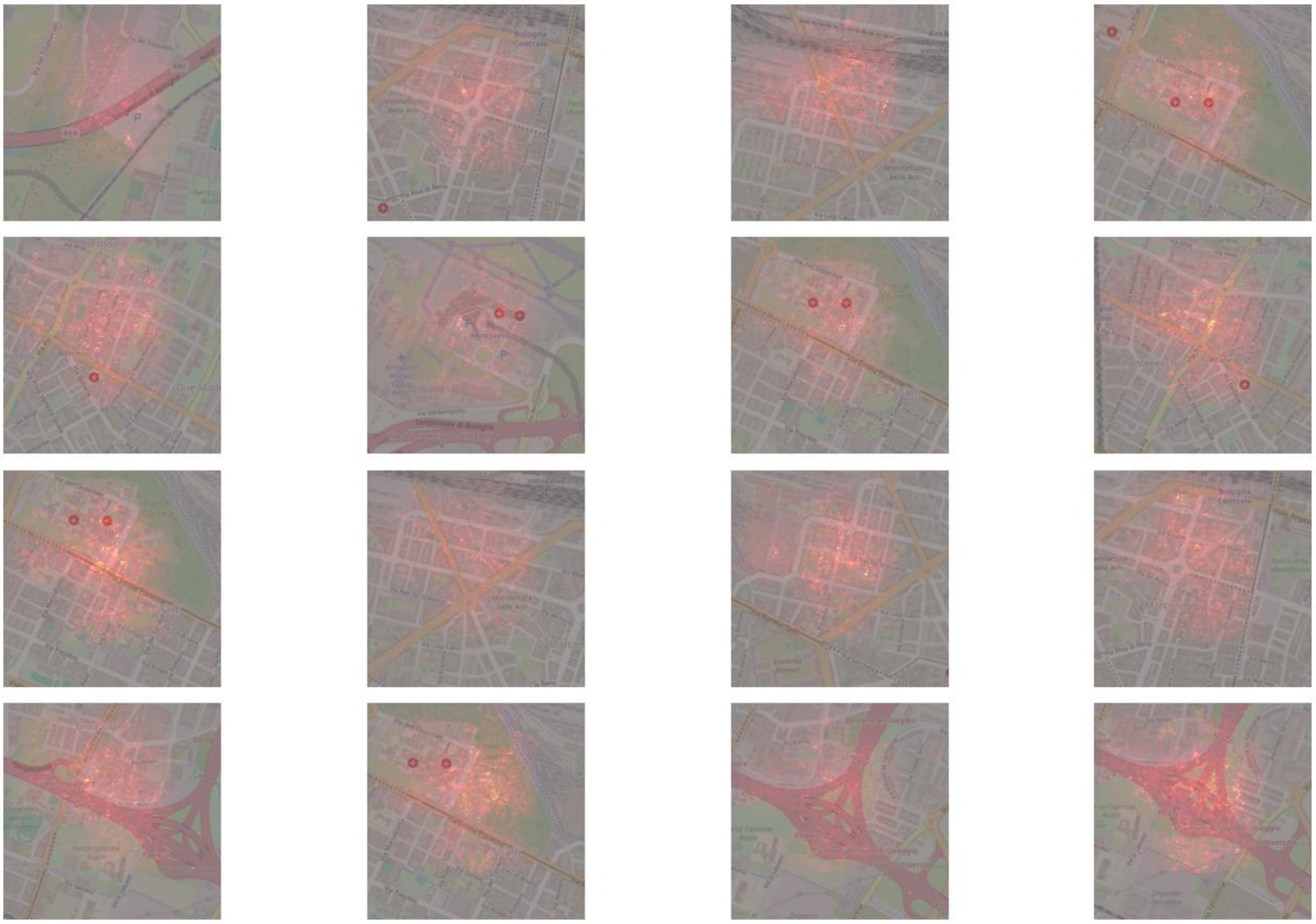

Figura 29: Batch di 16 immagini di esempio con saliencymap aggiunta all'immagine.

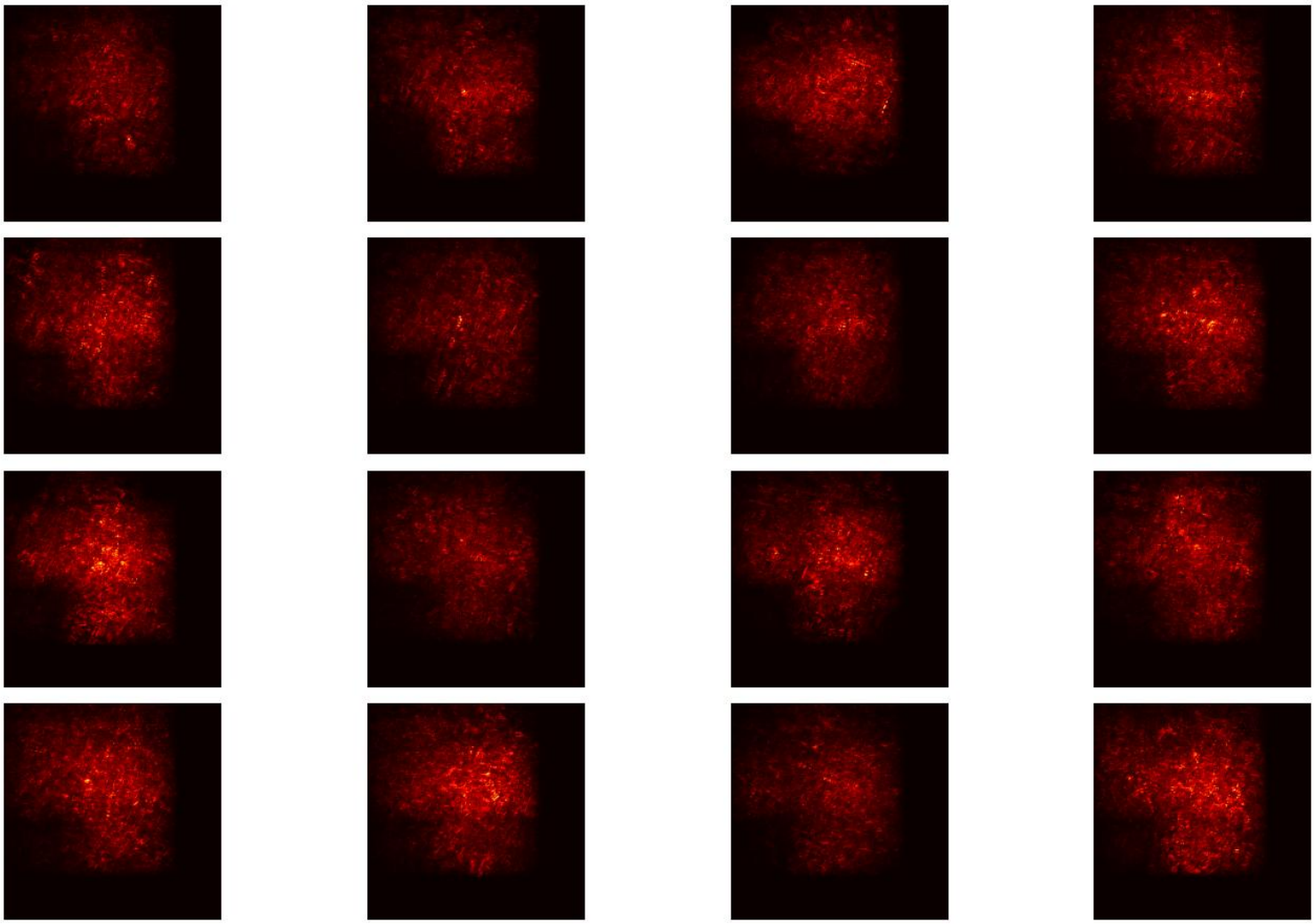

Figura 30: Colormap del batch di immagini di esempio

Possiamo osservare dalla colormap in Figura 30 e dalla saliencymap in Figura 29, che l'algoritmo concentra l'attenzione sul centro dell'immagine. Il fatto che i pixel evidenziati non la coprano interamente è positivo, in quanto significa che l'algoritmo si sta focalizzando su una parte dell'immagine e non su tutta. In sostanza, starebbe imparando che deve guardare sempre il centro dell'immagine per stabilire l'RSRP. Inoltre notiamo che non si sta concentrando su particolari dettagli o sulla conformazione del territorio, come strade, prati alberi o edifici, bensì guarda

la zona centrale in modo quasi compatto. Questo potrebbe essere un segnale di overfitting: potrebbe aver trovato una scorciatoglia nel dataset, piuttosto che imparare una relazione causale. Non è nemmeno da escludere la possibilità di underfitting e quindi di una CNN troppo semplice, inadatta all'analisi di queste immagini.

Aggiungendo la funzione del dropout nei moduli della convoluzione (con probabilità del 30%) e varie trasformazioni di data augmentatio alle immagini (come normalizzazione, colorjitter, crop...) si possono ottenere risultati migliori sulle saliency map e colormap, come possiamo vedere in Figura 31.

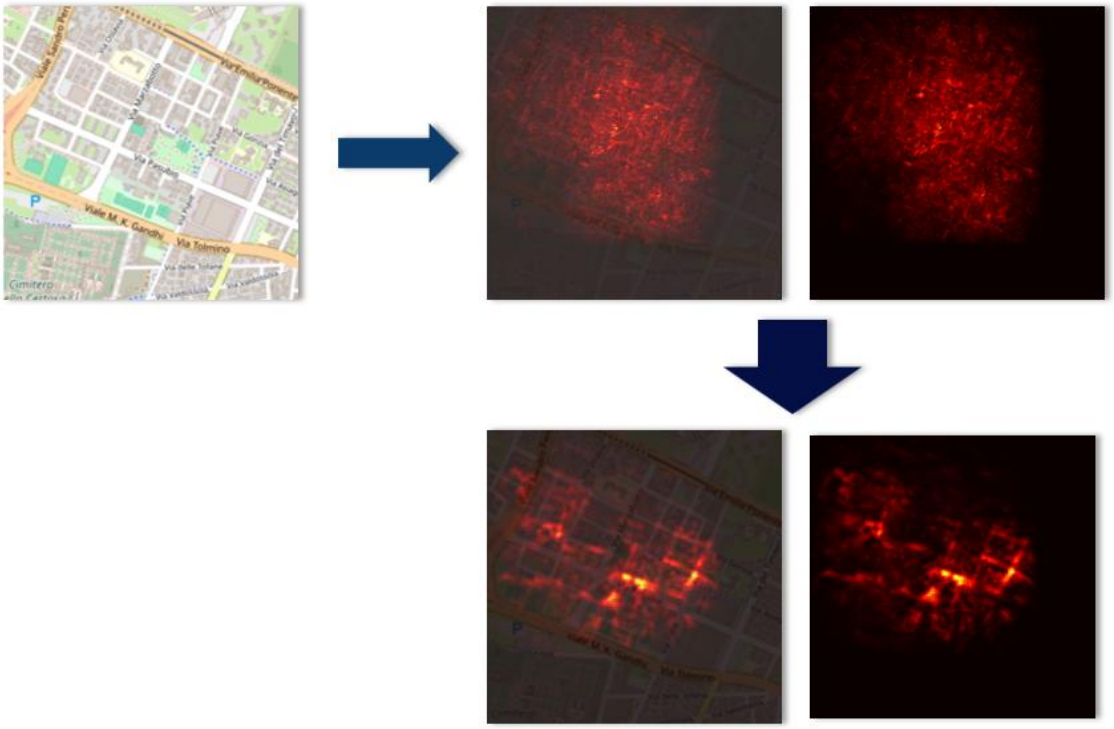

Figura 31: Esempio dell'applicazione della salincy map su un'immagine del dataset prima e dopo il miglioramento della CNN

Sembra che adesso la rete si stia concentrando su zone specifiche dell'immagine, come ad esempio le strade, e non solo il centro della figura. Tuttavia, valutando le performance, queste sono rimaste quasi invariate alle precedenti.

Sarebbero necessari altri studi e tentativi per approfondire la questione e interpretarla in maniera più chiara e rigorosa.

6.5 RSRQ – dataset Bologna

La stima per RSRQ è stata fatta principalmente con misure filtrate dal dataset di Bologna a frequenza 1800 MHz.

I primi tentativi hanno prodotto risultati tutt'altro che incoraggianti.

Infatti, come osserviamo dalla Figura 32, i valori della loss raggiungono picchi enormi in confronto a quelle precedenti; inoltre, l'errore relativo medio risultante è del 24.1%, assolutamente insufficiente per i propositi dell'azienda.

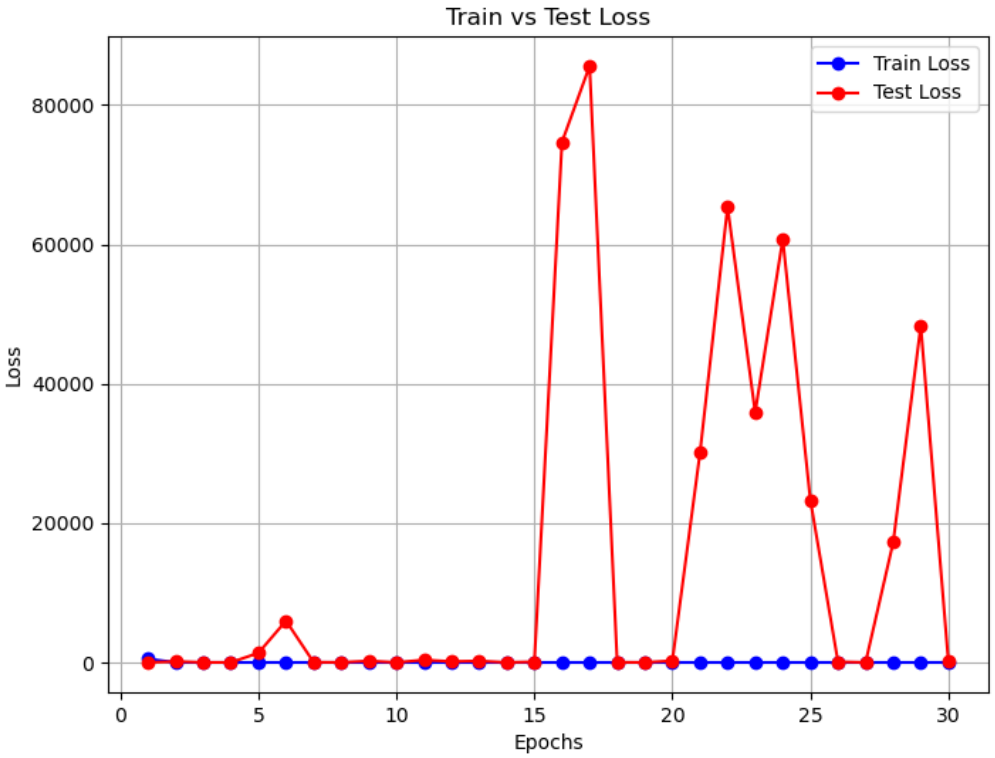

Figura 32: Loss 30 epoche RSRQ Bologna

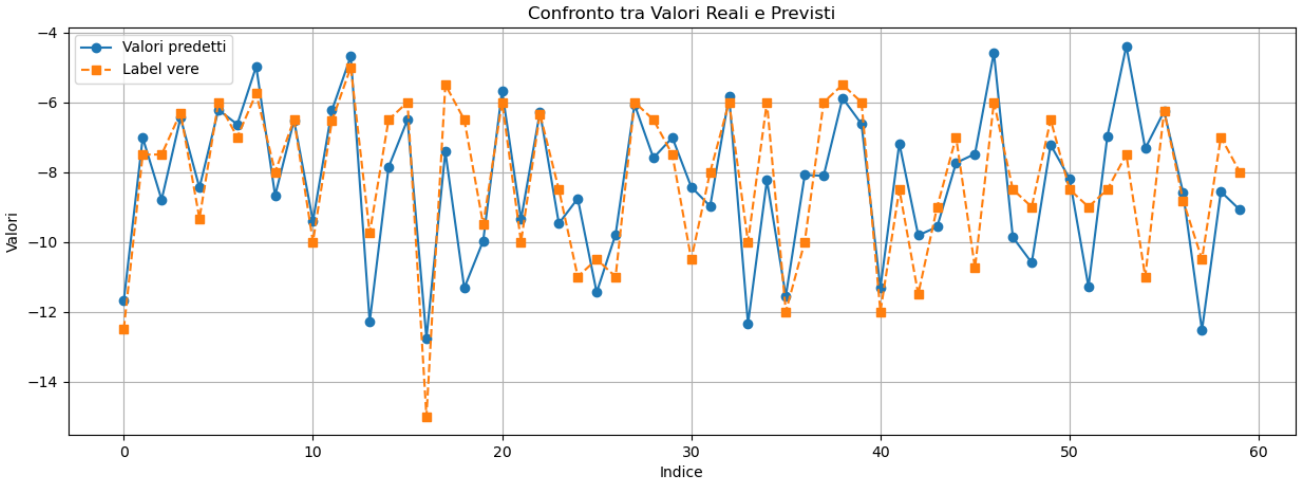

Figura 33: Visualizzazione del confronto tra i primi 60 punti del dataset con le label predette corrispondenti (dataset Bologna-RSRQ primo tentativo).

Per tentare di migliore questo risultato, è stato necessario aggiungere le distanze interferenti (come spiegato nel capitolo Metodologia).

Come visibile in Figura 34, l'allenamento è stato condotto per 100 epoche per ottimizzare al massimo le performance.

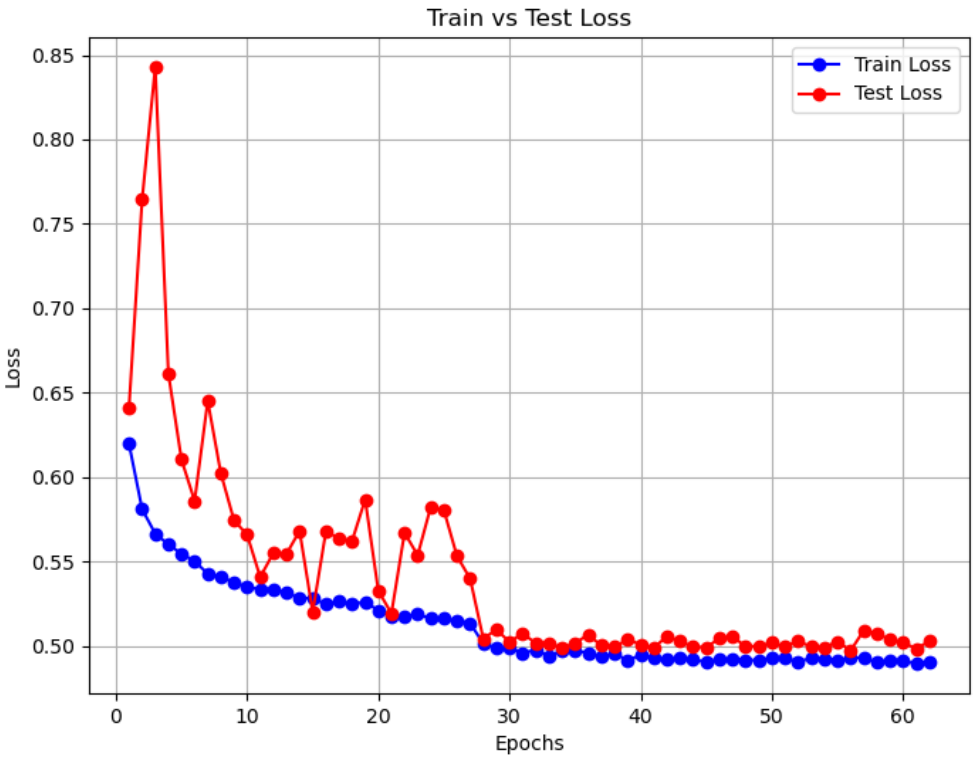

Figura 34: Loss 100 epoche dataset bologna per la stima dell'RSRQ

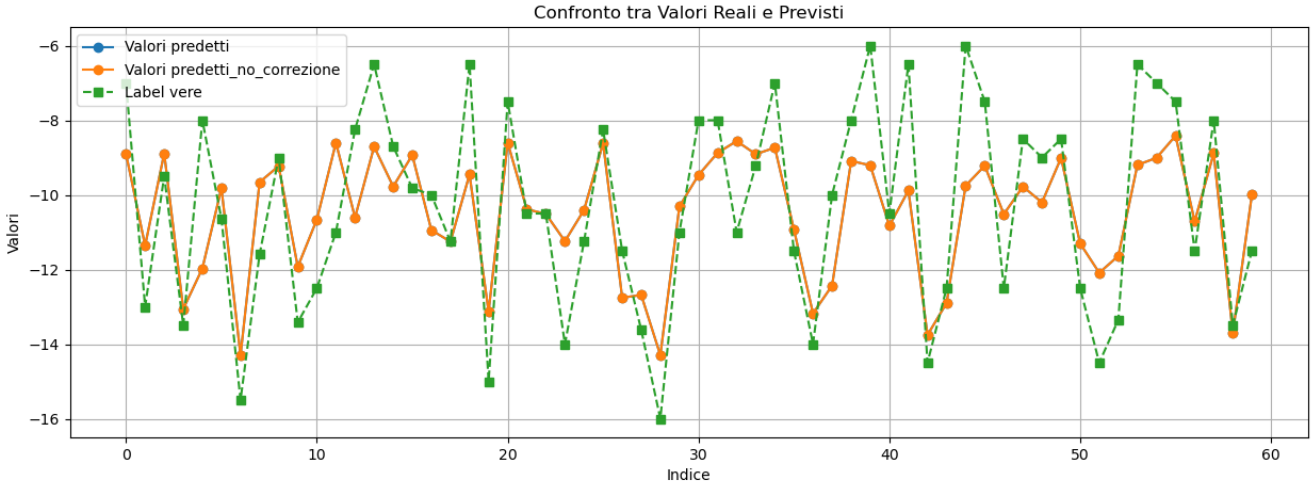

Figura 35: Visualizzazione del confronto tra i primi 30 punti del dataset con le label predette corrispondenti (dataset Bologna-RSRQ).

Si è ottenuto un errore relativo medio del 13.3%. Sono stati provati diversi tentativi per migliorare questa performance: provare l'allenamento senza immagini oppure usando immagini con zoom 17, minore rispetto a quello usato nelle simulazioni precedenti o ancora con l'aggiunta di un qualche modello fisico di supporto; per quest'ultimo abbiamo provato a stimare una pathloss come per l'RSRP, con la differenza che questa dipendeva dalle interferenze delle altre antenne. Tuttavia, ogni tentativo non ha prodotto risultati minori del 14% di errore.

Ad esempio, nelle figure 36 e 37 possiamo osservare l'andamento dei punti con l'uso di immagini a zoom 17 e, come appena affermato, l'errore relativo medio è del 14.6%.

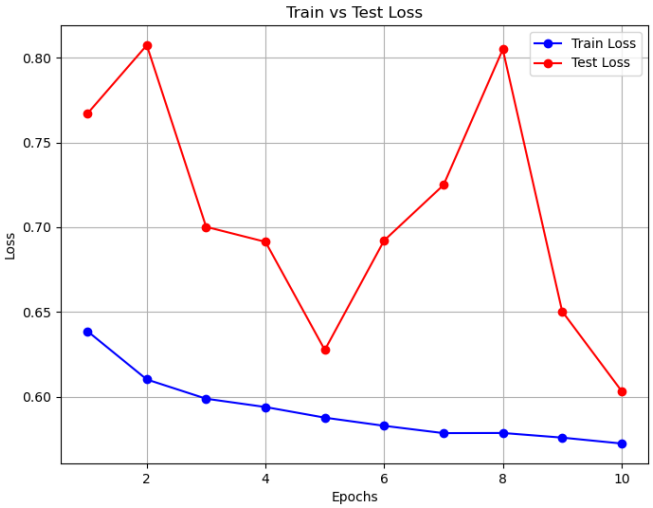

Figura 36: Loss 10 epoche per tentativo con immagini zoom 17 (dataset Bologna - RSRQ)

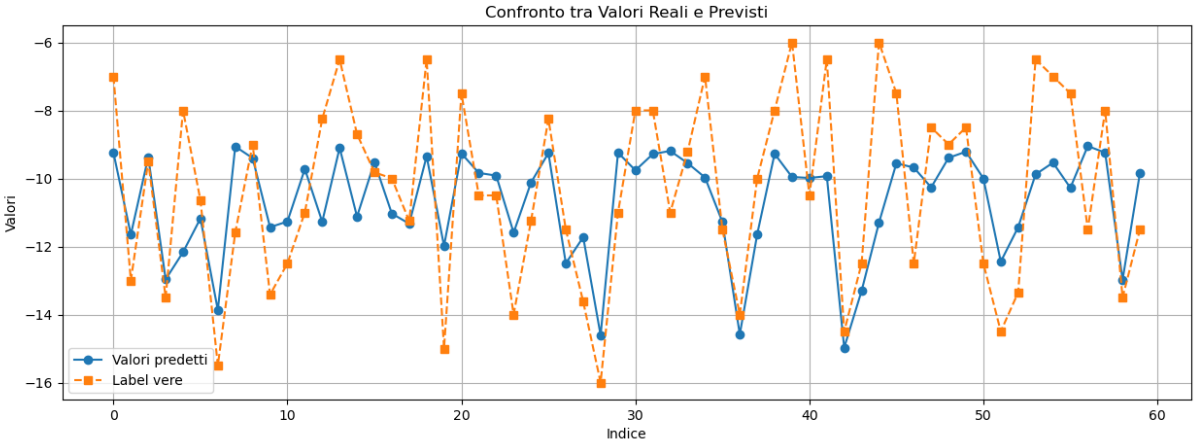

Figura 37: Visualizzazione batch di 60 punti a confronto con i corrispettivi valori veri (caso immagini zoom 17)

7 Sviluppi futuri

I miglioramenti a questo modello predittivo possono essere molteplici, da quelli legati a modifiche interne per aumentare le performance della rete, ad aggiunte basate su nuovi possibili obiettivi.

Per quanto concerne la stima dell'RSRP, sicuramente è necessario fare chiarezza sulla questione delle immagini: bisogna verificare con ulteriori mezzi interpretativi se sono sufficienti soltanto le features numeriche oppure no. Si potrebbe anche provare a migliorare la CNN aggiungendo strati e neuroni o lavorando sulle trasformazioni di data augmentation, per osservarne le conseguenze sui risultati prodotti.

Le performance della predizione dell'RSRQ non sono ancora pienamente soddisfacenti. Abbiamo pensato a due strade da percorrere per poter raggiungere risultati migliori. La prima è quella di trovare una formula correttiva all'output come la path loss per la stima dell'RSRP. Si potrebbe provare a ottenere il SINR (Signal to Interference plus Noise Ratio), un indicatore fondamentale della qualità del segnale nelle telecomunicazioni, che misura la potenza del segnale desiderato rispetto alla somma di segnali di interferenza e rumore. Questa come si può intuire è direttamente correlato con l'RSRQ ed è più facile da calcolare rispetto a questo.

Un'altra via, invece, potrebbe essere quella di cambiare il modello da un task di Regressione a uno di Classificazione. L'idea è quella di trasformare i valori numerici in classi come "Buono", "Scarso", "Ottimo", riferiti alla qualità del segnale. Pertanto, sarebbe necessario cambiare anche le funzioni stesse della rete, usando sigmoidi e softmax nelle attivazioni, e la Cross-Entropy per la Loss.

Per rendere il modello ancora più completo, sarebbe utile aggiungere features ulteriori presenti negli MDT: queste potrebbero essere tipologia del terminale, posizione temporale della misura, frequenza di trasmissione, inclinazione e altezza dell'antenna.

Per quanto riguarda la tipologia del terminale, l'idea potrebbe essere quella di considerare nel dataset i dieci terminali più numerosi; dopodichè si graficano i punti di questi terminali in un diagramma RSRP/distanza dall'antenna. A quel punto si raggruppano i punti in base al terminale e si ordinano i terminali osservando visualmente l'ordine dei gruppi in base alla posizione nel diagramma. Si catalogano i terminali in base alla loro posizioni, identificandoli con categorie come "Scarso", "Medio", "Buono".

Infine, l'informazione viene aggiunta nell'input usando la tecnica del one-hot-encoding.

In ultimo, come progetto a lungo termine, si potrebbe pensare ad un algoritmo che preveda l'evoluzione della copertura radio mobile nel tempo, aggiungendo, togliendo o spegnendo le antenne. Questo lavoro potrebbe portare notevoli benefici all'azienda dal punto di vista della riduzione dei consumi e dell'efficienza energetica

8 Conclusione

Questo lavoro di tesi ha avuto come obiettivo principale lo sviluppo di un modello di Machine Learning per la predizione dell'RSRP (Reference Signal Received Power) e dell'RSRQ (Reference Signal Received Quality) nelle reti di telecomunicazioni mobili. Partendo da un algoritmo basato su un paper di riferimento ("Model-aided deep learning method for path loss prediction in mobile communication systems at 2.6 GHz"), l'approccio ha combinato una rete neurale composita (MLP e CNN) con un modello fisico di supporto (UMa_B) per stimare l'attenuazione del segnale radio (path loss). L'obiettivo è stato quello di superare i limiti dei modelli teorici tradizionali, come TIMPLAN, offrendo un algoritmo più adattabile ai dati specifici, veloce nella valutazione e meno costoso grazie all'uso di strumenti opensource per la visualizzazione del territorio.

Il lavoro ha comportato l'adattamento del modello a dati reali forniti da TIM S.p.a., ottenuti tramite la tecnica MDT (Minimization of Drive Test). Sono state implementate diverse modifiche significative, tra cui un nuovo metodo di split dei dati per prevenire il "leakage spaziale" e l'eliminazione del "one-hot encoding" per l'identificativo delle celle, in quanto i dati aziendali utilizzavano una frequenza uniforme.

I risultati ottenuti sono stati analizzati su diversi dataset:

• Dataset del paper di riferimento: Il modello ha mostrato buone performance nella predizione dell'RSRP, con un errore relativo medio del 3.3% senza immagini e del 2.7% con immagini, confermando il leggero miglioramento apportato dall'informazione visiva. Per la predizione dell'RSRQ, l'errore si è attestato intorno al 9.3%, dimostrando una maggiore complessità del task.

• Dataset di Perugia (RSRP): Con misure a frequenza 800 MHz, le immagini si sono rivelate non fondamentali, con un errore relativo medio del 4.5% che aumentava solo a 5.5% senza di esse. Questo è attribuito all'alta capacità di

penetrazione delle onde a 800 MHz, che rende meno influente la conformazione del territorio.

• Dataset di Bologna (RSRP): Per le frequenze più alte (2660 MHz), le immagini hanno dimostrato un comportamento ambiguo. Inizialmente, il dataset presentava problemi di duplicazione spaziale dei punti, causando performance anomale e overfitting: qui le immagini sembravano avere un ruolo cruciale. La risoluzione del problema dei punti duplicati (mediando i valori RSRP per coordinate uniche) ha permesso di ottenere un errore relativo medio del 6.5% con immagini e del 7% senza, rendendo nuovamente non fondamentali le immagini nella generalizzazione e nella comprensione della morfologia del territorio. L'analisi tramite saliency map ha evidenziato come l'algoritmo si focalizzi sul centro delle immagini, suggerendo un funzionamento inadeguato della CNN e quindi la necessità di ulteriori studi per interpretare il focus della rete e prevenire potenziali scorciatoie di apprendimento.

• Dataset di Bologna (RSRQ): La predizione dell'RSRQ è rimasta complessa, anche con l'introduzione di feature aggiuntive relative alle distanze dalle antenne interferenti. Con un errore relativo medio del 13.3%, i tentativi di migliorare le performance non hanno prodotto riduzioni significative dell'errore.

Gli sviluppi futuri del modello sono molteplici e promettenti. Per l'RSRP, sarà fondamentale chiarire ulteriormente il ruolo delle immagini e ottimizzare la CNN. Per l'RSRQ, si potrebbe esplorare la possibilità di trovare una formula correttiva fisica basata sul SINR o di trasformare il problema in un task di classificazione. L'integrazione di nuove feature provenienti dai dati MDT (come la tipologia di terminale, la posizione temporale o i parametri dell'antenna) potrebbe arricchire ulteriormente l'input del modello. A lungo termine, un progetto ambizioso potrebbe mirare a prevedere l'evoluzione della copertura radio mobile nel tempo, supportando decisioni strategiche per l'installazione o lo spegnimento di antenne, con notevoli benefici in termini di risparmio energetico, economico e riduzione delle emissioni di CO_2 per TIM S.p.a..

In sintesi, questo lavoro ha dimostrato l'efficacia dell'apprendimento automatico, assistito da modelli teorici, nella previsione delle metriche di copertura radio, ponendo le basi per ulteriori miglioramenti e per un'ottimizzazione più intelligente e sostenibile delle reti mobili.

9 Bibliografia

[1] «3Gpp tr 38.900 release 15,» [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2991.

[2] J. Shodamola, H. Qureshi, U. Masood e A. Imran, Towards Addressing the Spatial Sparsity of MDT Reports to Enable Zero Touch Network Automation, IEEE, 2021.

[3] J. Thrane, M. Artuso, D. Zibar e H. L. Christiansen, Drive test minimization using Deep Learning with Bayesian approximation, Kgs. Lyngby 2800, Denmark: IEEE, 2018.

[4] H. N. QURESHI, A. IMRAN e A. ABU-DAYYA, Enhanced MDT-Based Performance Estimation for AI Driven Optimization in Future Cellular Networks, Department of Electrical Engineering, Qatar University, Doha, Qatar: IEEE, 2020.

[5] M. Hata, Empirical Formula for Propagation Loss in Land Mobile Radio Services, IEEE, 1980.

[6] J. Thrane, M. Artuso, D. Zibar e H. L. Christiansen, Model-aided Deep Learning Method for Path Loss Prediction in Mobile Communication Systems at 2.6 GHz, IEEE, 2020.

[7] Jakthra, «PathLossPredictionSatelliteImages,» [Online]. Available: https://github.com/jakthra/PathLossPredictionSatelliteImages.

[8] «Pytorch,» [Online]. Available: https://pytorch.org/.

[9] «OpenStreetMap,» [Online]. Available: https://www.openstreetmap.org/.

[10] «Pandas,» [Online]. Available: https://pandas.pydata.org/.

[11] «Polars,» [Online]. Available: https://pola.rs/.

[12] «Folium,» [Online]. Available: https://pypi.org/project/folium/.

[13] M. Charitos, D. Kong, J. Cao, D. Berkovskyy, A. A. Goulianos e T. Mizutani, LTE-A Virtual Drive Testing for Vehicular Environments, IEEE, 2017.

[14] R. Enami, D. Rajan e J. Camp, RAIK: Regional Analysis with Geodata and Crowdsourcing to Infer Key Performance Indicators, Department of Electrical Engineering, Southern Methodist University: IEEE, 2018.

[15] H. Gokcesu, O. Ercetin, G. Kalem e S. Ergüt, QoE Evaluation for Adaptive Video Streaming: Enhanced MDT with Deep Learning, arXiV, 2022.

[16] G. Bini, F. D. Corpo e C. Carlini, IL FIXED WIRELESS ACCESS DI TIM AL SERVIZIO DEL PAESE, Notiziario tecnico TIM, 2020.