UNIVERSITÀ DI PARMA

Dipartimento di Scienze Matematiche, Fisiche e Informatiche

Corso di Laurea in Informatica

Apprendimento e predizione di dati fluviali tramite reti neurali

Learning and prediction of fluvial data through neural networks

Relatore:

Prof. Alessandro Dal Palú

Laureando:

Varesi Marco 264741

ANNO ACCADEMICO 2017-2018

Ai miei genitori.

"Vuoi farmi credere sia impossibile ricordare i propri sogni al risveglio, ho deciso di intrappolarli e raggiungere il mio ad occhi aperti."
Mario Molinari

Indice

Introduzione

Alla base di questo studio vi è l'analisi e la previsione dei dati fluviali.
In particolare, si pone l'attenzione sulla tecnica del reverse flow routing, ovvero la capacità di stimare la variazione nel tempo dell'altezza del fiume a monte utilizzando solo le osservazioni a valle.
L'obiettivo di questa tesi di laurea è quello di utilizzare una rete neurale per ricostruire i livelli del fiume Secchia a monte di un evento misurato, in modo da predisporre simulazioni basate.
Riuscendo a ottenere una buona approssimazione, soprattutto nei periodi di piena, sarà possibile utilizzare la rete per previsioni o ricostruzioni di eventi.
Il presente lavoro si articola in 4 sezioni:

- Nella prima ci si concentra sul concetto di rete neurale per chiarire il principale strumento che andremo ad utilizzare.

  Viene inoltre analizzato il reverse flow problem e le sue soluzioni

- Nella seconda ci si sofferma sull'obiettivo della tesi

- Nella terza si analizzando i possibili metodi per la risoluzione del problema utilizzando a tecnica delle reti neurali

- Nella quarta vengono mostrati e analizzati i risultati ottenuti, con i relativi grafici

Capitolo 1

Background

In questo capitolo andrò ad approfondire il concetto di rete neurale per fornire un background al lettore.
Le reti neurali sono una tecnologia informatica che presenta molte analogie con la struttura del cervello e in particolare con i numerosi neuroni fortemente connessi da sinapsi attraverso cui la computazione si diffonde in parallelo nella corteccia cerebrale.
Analizzerò anche nel dettaglio il problema del reverse flow routing.

1.1 Il cervello umano

I neuroni sono cellule nervose destinate alla produzione e allo scambio di segnali; rappresentano quindi l'unità fondamentale del sistema nervoso.
Sono infatti presenti circa 100 miliardi di neuroni nel cervello umano.
Sono caratterizzati da 3 regioni: il corpo cellulare, chiamato anche soma, i dendriti e l'assone (Figura 1.1).
I dendriti sono ramificazioni sottili, la cui principale funzione è ricevere segnali in entrata, e permettono al neurone di comunicare con molte altre cellule nervose.
L'assone è una sorta di prolungamento, il cui scopo è la trasmissione dei segnali.

La comunicazione tra neuroni viene chiamata sinapsi, ed è formata da una protuberanza sull'assone, chiamata bottone sinaptico, che si incastra con una prominenza sulla superficie di un dendrite.
Il segnale viene trasmesso, da una cellula ad un'altra, non da una connessione elettrica diretta, ma dall'emissione di una piccola quantità di fluidi chiamati neurotrasmettitori, ed avviene in un piccolo spazio denominato fessura sinaptica.
Lo schema è dunque il seguente: vi è un segnale elettrico lungo l'assone, seguito da una trasmissione chimica nella fessura sinaptica, seguito da un segnale elettrico sul lato del dendrite.
La cellula riceve un'intera serie di segnali dai suoi dendriti, li somma all'interno della soma, e se la somma pesata supera il valore di attivazione, emette un impulso elettrico in uscita lungo l'assone, da inviare alle cellule successive.
La sinapsi può svolgere un'azione eccitatoria, facilitando il raggiungimento del valore della soglia, o inibitoria, in cui invece viene ostacolato.

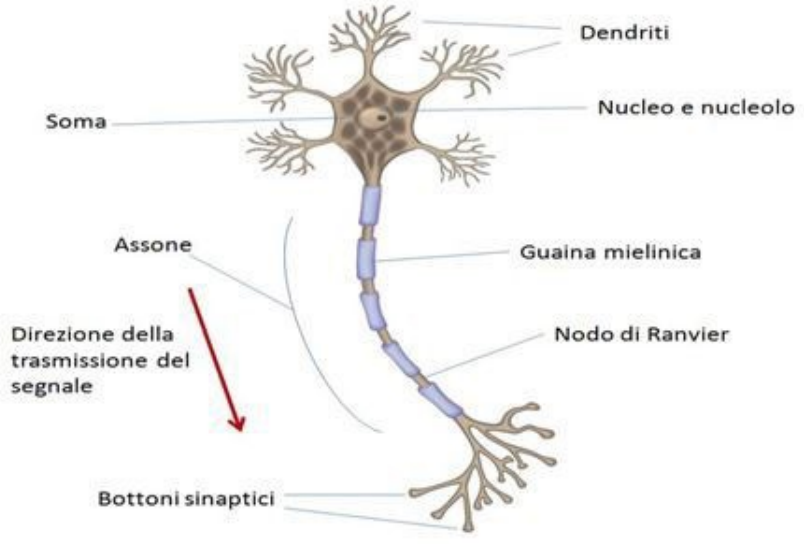

Figura 1.1: La struttura del neurone

1.2 Il neurone artificiale

L'unità utilizzata nelle reti neurali è un modello matematico molto semplificato del neurone biologico.
Ad ogni input x_i è associato un peso w_i con valore positivo o negativo per eccitare o inibire il neurone.
Il bias varia secondo la propensione del neurone ad attivarsi, per variare la soglia di attivazione (Figura 1.2).
L'algoritmo del neurone è il seguente: caricati i valori degli input x_i e dei pesi w_i, viene eseguita la somma dei valori in input pesati con i relativi pesi. Successivamente viene calcolato il valore della funzione di attivazione f utilizzando come argomento il risultato della somma pesata.
L'output del neurone y è quindi il risultato della seguente funzione:

$$y(x) = f(\sum_{i=1}^{d} w_i x_i + w_0)$$

Dove d è numericamente uguale al numero di input totali, e w_0 è il valore del bias.

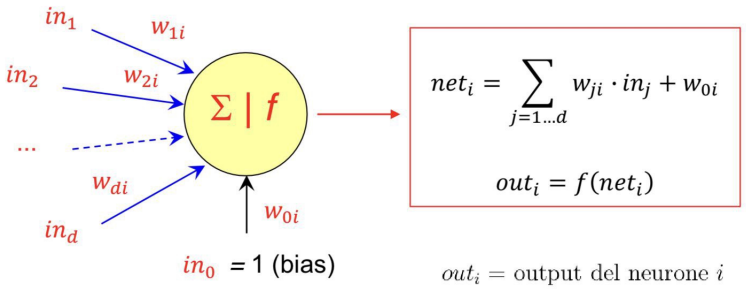

Figura 1.2: La struttura del neurone artificiale

È quindi la funzione di attivazione che determina la risposta del neurone, e ne esistono di diverso tipo.
Le più famose sono le seguenti: a gradino, sigmoide e lineare continua.

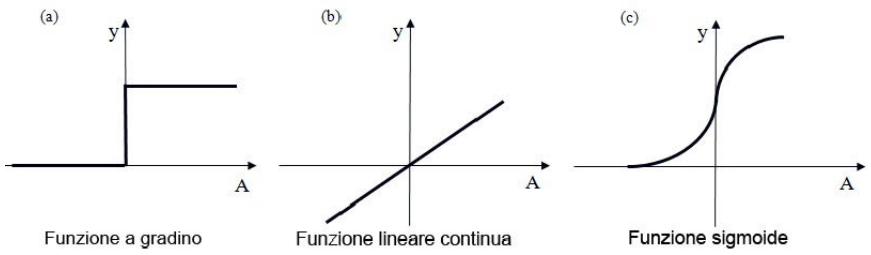

Figura 1.3: Diversi tipi di funzione di attivazione

Le reti neurali sono composte da gruppi di neuroni artificiali organizzati in livelli.
Tipicamente sono presenti: un livello di input, un livello di output, e uno o più livelli intermedi nascosti (hidden layer), ed ogni livello contiene uno o più neuroni.
Considerando che la rete è costituita solo da unità elementari semplici, è la topologia di quest'ultima a determinare (principalmente) la capacità di apprendere.

1.3 L'apprendimento

Una rete neurale non viene programmata per eseguire una certa attività, ma addestrata, utilizzando un algoritmo di apprendimento automatico, mediante una serie di esempi della realtà da modellare.
Nonostante ciò non è in grado di giustificare i risultati.
Occorre inoltre un periodo di allenamento in cui istruire la rete su come comportarsi con l'input che riceve, perché all'inizio non ha nessuna forma di conoscenza.
Quest'ultima infatti è memorizzata nei pesi, e la rete apprende usando tecniche di ottimizzazione per variarne il valore.
Principalmente esistono 2 tipi di apprendimento:

- Supervisionato
- Non supervisionato

1.3.1 Apprendimento non supervisionato

L'apprendimento non supervisionato viene chiamato anche "Learning without a teacher", proprio per sottolineare il fatto che non esiste un insegnante che sorveglia l'apprendimento, e quindi non vengono forniti esempi concreti su cui allenare il modello.
Alla rete infatti vengono presentati solo i valori di input che dividerà autonomamente in gruppi usando misure di similarità, senza usare confronti con output noti, e cercando di mettere input simili nello stesso gruppo.
Una volta che è riuscita a cogliere le regolarità statistiche dei dati, sviluppa la capacità di formare rappresentazioni interne per la codifica dei dati in input e quindi di creare automaticamente nuove classi.

1.3.2 Apprendimento supervisionato

Nell'apprendimento supervisionato viene presentato alla rete un training set, preparato da un supervisore esterno, composto da molte coppie significative di valori (input, output atteso).
I dati in input sono i valori in ingresso della rete e vengono inviati, associati a un determinato peso, ai neuroni dello strato di input.
Il risultato della funzione di attivazione di un neurone, usando come parametro la somma pesata dei valori in ingresso, è il suo output, e viene inviato a tutti i neuroni dello strato successivo con cui è collegato.
Ogni arco avrà associato un proprio peso che andrà ad aumentare o diminuire il valore inviato dal neurone, e diventerà un input del neurone successivo.
L'output dell'ultimo neurone della rete è il risultato finale (Figura 1.4).
Inizialmente i pesi vengono settati con valori casuali, tipicamente bassi, per poi venire gradualmente modificati fino a raggiungere il valore ottimo.
Questa modifica avviene durante il processo di training, in cui l'output y generato dalla rete, può non essere uguale all'output desiderato d.

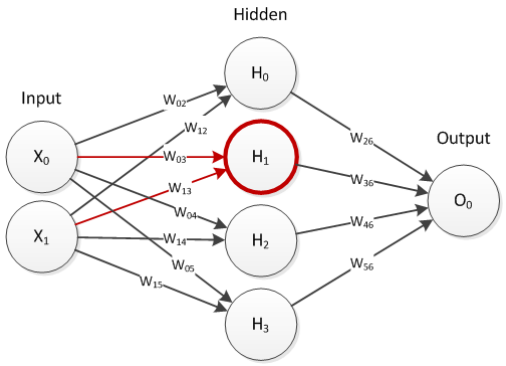

Figura 1.4: La struttura di una rete neurale con un hidden layer

L'errore nell'approssimazione è dato dalla differenza tra il suo output e l'output atteso, ed è molto utile per far capire alla rete quanto sbaglia.
Il principio base dell'Error-correction rules è quello di identificare un segnale d'errore, che può essere $(d - y)$ e modificare i pesi di conseguenza.
Man mano che la rete elabora output, procederà a correggersi, migliorando le risposte variando i pesi, ovviamente aumentando quelli che determinano gli output corretti e diminuendo quelli che generano valori non validi.
Occorre quindi preparare numerosi esempi di funzionamento studiati appositamente, grazia al quale il modello apprenderà il nesso che li unisce, riuscendo quindi a generalizzare, ossia a calcolare nuove associazioni corrette (input,output) processando input esterni al training set.
Il meccanismo di apprendimento supervisionato più utilizzato è la tecnica dell'Error Back-Propagation.
Una volta calcolato l'errore, tramite una funzione di errore E, tra l'output della rete e l'output atteso, propaga all'indietro l'errore (per questo si chiama back-propagation) verso il livello input, addizionando ad ogni peso un incremento positivo o negato determinato dalla sua influenza nella formazione dell'errore.
In modo particolare, l'aggiornamento dei pesi si ottiene retropropagando opportunamente il valore assunto dalla funzione di errore dallo strato di out-

put, attraverso gli strati nascosti, fino allo strato iniziale di input, nel modo seguente:

$$w_{i,j}(t+1) = w_{i,j}(t) - \eta \frac{\partial E}{\partial w_{i,j}}(t+1)$$

dove:

- $w_{i,j}(q)$ = valore alla q-esima iterazione (od epoca) dell'algoritmo, del peso associato all'arco che connette l'i-esimo neurone con il j-esimo neurone
- $\eta \in [0,1]$ = learing rate
- $\frac{\partial E}{\partial w_{i,j}}(q)$ = valore, sempre alla stessa iterazione, assunto dalla derivata parziale della funzione di errore rispetto al peso stesso

Siccome l'output di $E(w)$ è ottenuto in base al valore dei pesi, E è derivabile nei pesi w.
Per trovare la soluzione ottima occorre determinare il vettore w che rende minimo l'errore, ovvero il punto di minimo della funzione E.
Il valore di E può essere ridotto modificando i pesi in direzione opposta al gradiente della funzione, che indica la direzione di massima crescita.
Eseguendo questa modifica più volte saremo in grado di raggiungere un valore di minimo, assoluto o locale, in cui il gradiente è nullo.
Grazie a questa proprietà, l'algoritmo di back-propagation viene anche chiamato discesa del gradiente.
Il learning rate è un elemento essenziale della formula ed è il fattore di apprendimento della rete.
Durante l'algoritmo viene determinato quanto devono variare i pesi, in base all'errore, per permettere un riconoscimento ottimo per quel determinato pattern.
Se questa variazione venisse utilizzata direttamente, la rete apprenderebbe molto bene a riconoscere quel pattern, dando risultati pessimi per gli altri.

In pratica perderebbe tutta l'informazione che aveva appreso precedentemente.
Per evitare ciò si utilizza un fattore moltiplicativo η , il learning rate, che riduce la variazione ottima per quel pattern.
L'applicazione dell'error back-propagation per ogni coppia (input,output atteso) prende il nome di epoca di apprendimento.
Dovranno essere eseguite più epoche per fare in modo che la rete riesca a comprendere il modello statistico dei dati.

1.4 Overfitting e underfitting

La rete neurale deve avere capacità di comprensione del modello statistico del dati e non memorizzare i soli pattern del training set.
Nell'apprendimento l'errore E tende a 0 perché la rete sta imparando meglio.
È molto importante trovare un rapporto adeguato tra le dimensioni del training set e le dimensioni della rete.
Un numero eccessivo di parametri in ingresso e una troppo potente capacità di elaborazione, paradossalmente, rendono difficile alla rete neurale imparare a generalizzare, perché input esterni vengono valutati come troppo dissimili dai sofisticati e dettagliati modelli che conosce (Figura 1.5).
Per evitare ciò si può utilizzare un validation set, ovvero coppie (input,output corretto) su cui calcolare l'errore E.
Questi pattern non saranno presenti nel training set, e quindi non saranno utilizzati per allenare la rete.
In pratica, una volta calcolato l'output, qualunque valore assuma, non causerà la modifica dei pesi e la rete rimarrà invariata.
L'errore calcolato sul validation set è quindi una buona approssimazione della generalizzazione della rete.
Inizialmente l'errore assumerà un valore simile a quello del training set, con cui diminuirà di pari passo durante le epoche di apprendimento.

Una volta che avrà raggiunto il valore ottimale, se si continuerà ad addestrare la rete, quest'ultima inizierà ad imparare il training set, il cui errore continuerà a scendere, ma non il suo modello statistico, causando l'incremento dell'errore del validation set.
D'altro canto si può incorrere nel fenomeno dell'underfitting, causato per esempio dallo scarso numero di variabili nel training set, e la rete non avrà abbastanza parametri per imparare a generalizzare.

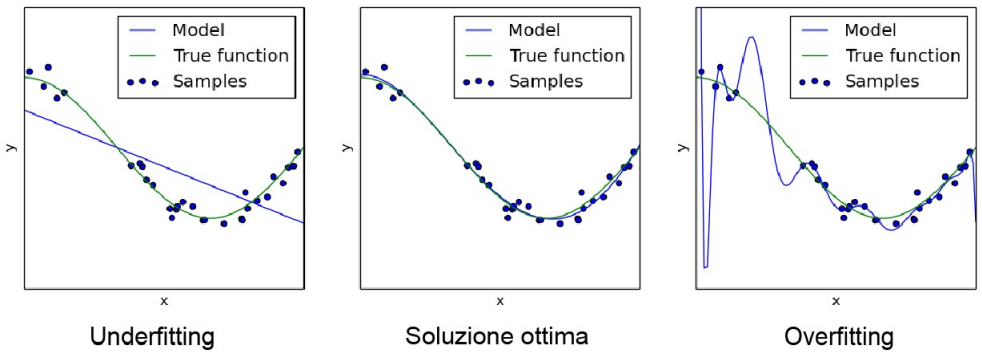

Figura 1.5: Overfitting e Underfitting

1.5 Il problema della regressione

Il machine learning è utilizzato per molteplici scopi in cui è richiesto di prendere delle decisione sui dati.
Principalmente viene impiegato per la risoluzione di due tipologie di problemi:

- Classificazione, il cui scopo è individuare l'appartenenza di un elemento ad una classe.

  L'output predetto è quindi categorico, ossia può assumere solo un numero finito di possibili valori come {SI, NO}, {ALTO, MEDIO, BASSO} ecc...

L'obiettivo è quindi quello di trovare la funzione, che in molti casi non è lineare, che assegni in modo accurato l'attributo classe a nuovi records non classificati.

- Regressione, in cui l'output di un elemento è un valore numerico.

  I pattern in ingresso sono coppie (x,y) e si suppone che esista una funzione $y = f(x)$ incognita.

  L'obiettivo è quindi quello di determinare la funzione che meglio approssima questi dati (Figura 1.6).

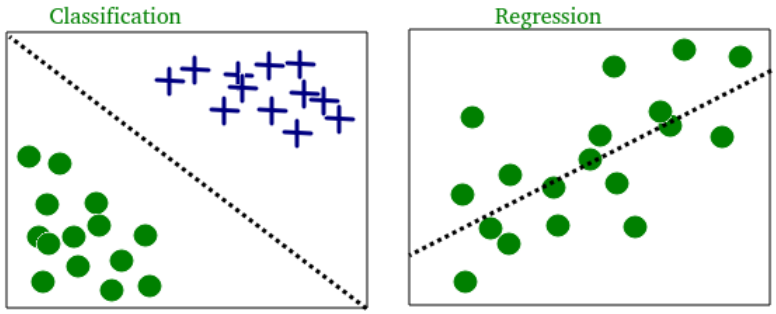

Figura 1.6: Classificazione e Regressione

La regressione viene usata per prevedere il valore di una variabile numerica (variabile dipendente) in base ai valori di uno o più variabili predittive (variabili indipendenti).
Nel più generale modello di regressione multipla, ci sono p variabili indipendenti che possono predire un valore y_i secondo la formula

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_p x_{pi} + \epsilon_i$$

dove

- β_0 = intercetta
- β_i = inclinazione di y_i rispetto alla variabile x_i, tenendo costanti le altre variabili

- ϵ_i = errore in corrispondenza dell'osservazione i

1.6 Tensorflow

Per lo svolgimento di questa tesi ho utilizzato TensorFlow, una libreria software open source per il calcolo numerico ad alte prestazioni, che fornisce API native in linguaggio Python, C/C++, Java, GO, e Rust.
Sviluppato originariamente dai ricercatori del team Google Brain, viene fornito con un forte supporto al machine learning, ed è attualmente usato in dozzine di prodotti commerciali Google come il riconoscimento vocale, Gmail, Google Foto, e Ricerca.
Per installarlo su una macchina, con Ubuntu 16.04 LTS come sistema opertivo, occorre utilizzare il gestore di pacchetti pip di Python.
Se non sono presenti sulla macchina utilizzata sarà necessario installare Python, il gestore dei pacchetti pip, e Virtualenv tramite i comandi:

```
$ sudo apt update
$ sudo apt install python3-dev python3-pip
$ sudo pip3 install -U virtualenv
```

Una volta completata questa operazione è consigliato creare un ambiente virtuale, in modo da isolare l'installazione del pacchetto dal sistema.
Per crearlo occorre scegliere un interprete python e creare una directory ./venv per contenerlo:

```
$ virtualenv --system-site-packages -p python3 ./venv
```

Una volta creato l'ambiente, occorre attivarlo usando un comando specifico della shell:

```
$ source ./venv/bin/activate # sh, bash, ksh, or zsn
```

Una volta attivato virtualenv, il prompt della shell sarà preceduto da (venv).
Per uscire dall'ambiente virtuale sarà necessario usare il seguente comando:

```
(venv) $ deactivate
```

Ora saremo in grado di installare i pacchetti all'interno dell'ambiente virtuale senza influire sulla configurazione del sistema host.
Iniziamo aggiornando pip:

```
(venv) $ pip install --upgrade pip
```

Una volta eseguiti tutti i precedenti passaggi saremo pronti a installare la versione di TensorFlow solo per CPU usando il comando:

```
(venv) $ pip install tersorflow
```

Per poter usufruire delle API native di TensorFlow in un programma Python, occorrerà inserire la seguente riga di codice:

```
import tensorflow
```

Completato il setup, ho seguito la guida "Get Started with TensorFlow" per poter apprendere le nozioni essenziali per utilizzare la libreria.

1.7 Il reverse flow routing

L'idrogramma di un fiume è il grafico che mostra le variazioni nel tempo di alcuni parametri fluviali.
Con reverse flow routing si intende la stima dell'idrogramma di un fiume a monte, relativo alla portata o ai livelli dell'acqua, utilizzando solo le informazioni disponibili a valle del sito selezionato ed eventualmente la lunghezza del fiume.
Questa tecnica presenta due sfide principali; la soluzione potrebbe non essere univoca, e potrebbero verificarsi instabilità durante l'inversione.
I tradizionali tentativi per risolvere il reverse flow routing si basano su due approcci: la soluzione di una forma inversa delle equazioni di Saint Venant e l'applicazione di back-oriented di schemi di routing idrologico.

Considerando le informazioni a valle affette da errori, nella determinazione dell'idrogramma possono presentarsi instabilità e oscillazioni spurie.
Oltre alla procedura riportata sopra, la stima del grafico a monte basato solo su informazioni a valle (osservazioni) può essere eseguito tramite metodi di ottimizzazione.
Queste tecniche mirano a trovare l'idrogramma a monte che instradato a valle, attraverso la tecnica del forward model, corrisponde meglio alle osservazioni disponibili.
D'Oria e Tanda (2012) hanno risolto il reverse flow problem utilizzando un approccio geostatico Bayesiano (GBA) come procedura di ottimizzazione che considera il grafico della portata come una funzione casuale continua che presenta l'autocorrelazione.
Gli autori hanno mostrato le capacità della metodologia GBA, in combinazione con un modello idraulico in avanti (forward hydraulic model) di stimare la portata in una stazione a monte basandosi solo su un idrogramma a valle disponibile: la soluzione era stabile anche in presenza di valori a valle corrotti.
Il forward model, che risolve le equazioni 1-D di Saint Venant, è in grado di descrivere il processo di instradamento idraulico con sufficiente accuratezza.
Il metodo GBA è stato ulteriormente esteso per adottare idrogrammi sull'altezza del fiume anziché quelli sulla portata come osservazioni a valle (D'Oria e Tanda, 2014).
Tutti i lavori precedentemente citati hanno adottato modelli idraulici 1-D in combinazione con diverse procedure di ottimizzazione.
Tuttavia, in molti casi reali, il complesso campo idrodinamico generato dalla propagazione della piena non può essere descritto con precisione con ipotesi 1-D, ed è necessario adottare schemi basati sulle equazioni 2-D, anche se ció pone l'inconveniente di un'indagine dettagliata del terreno e di un maggior costo computazionale (limitato grazie alla parallelizzazione della procedura di stima della matrice Jacobiana).
Un dato molto importante da tenere in considerazione è la presenza di osservazioni corrotte a valle, poiché i dati registrati nelle stazioni sono spesso

influenzati da errori strumentali.
Per risolvere i problemi inversi, caratterizzati da parametri sconosciuti che sono correlati l'un l'altro nello spazio o nel tempo, si utilizza l'approccio bgaPEST.
Il punto fondamentale del bgaPEST, così come di altri metodi basati sull'approccio bayesiano, è il teorema di Bayes, ovvero:

$$p(s|y) \propto L(y|s)p(s) \tag{1.1}$$

dove s è il vettore dei parametri sconosciuti, y è il vettore dei dati misurati, $p(s|y)$ è la funzione di densità di probabilità posteriore di s dato y, $L(y|s)$ è la funzione di verosimiglianza, ovvero la probabilità di y dato s, e $p(s)$ è la funzione di densità di probabilità di s a priori.
s rappresenta quindi il vettore dei valori sconosciuti a monte, mentre y il vettore delle misurazioni osservate a valle.
Il migliore set di parametri s si ottiene massimizzando $p(s|y)$.
La funzione di probabilità $L(y|s)$, della formula 1.1, caratterizza la devianza tra i dati osservati e il modello.
Partendo dai risultati del forward model, ovvero la stima dell'igrogramma a valle in base ai dati disponibili a monte, $L(y|s)$ delinea come un particolare set di parametri s è in grado di riprodurre le osservazioni y nello spazio e/o nel tempo.

Capitolo 2

Obiettivo della tesi

L'obiettivo della tesi è ottenere una rete neurale in grado di risolvere il reverse flow problem, e ottenere di conseguenza una buona stima dei valori dell'altezza del fiume, per poter effettuare previsioni o ricostruzione di eventi.

2.1 Analisi requisiti

Il fiume che prenderemo in considerazione è il Secchia, e in particola la stazione di Bacchello a valle, e quella di ponte Alto a monte(Figura 2.1).
Le informazioni su cui ho potuto lavorare sono le misurazioni in metri, effettuate ogni mezzora, di 6 anni (2011, 2012, 2014, 2015, 2016, 2017) nelle due stazioni precedentemente citate.
La rete neurale dovrá riuscire a prevedere il livello del fiume a monte, avendo a disposizione solo le misurazioni a valle.
Lo scopo è quindi quello di riuscire a studiare la struttura della rete per vedere in che condizioni impara meglio.
Oltre che lavorare sulla struttura della rete e i suoi parametri, dovró anche costruire un training set efficace.
Non avendo nessun tipo di informazione iniziale dovrò stabilire la dimensione del training set, sia in termini di numero di pattern, sia in termini di sample utili per la previsione, per ottenere una misurazione il più precisa possibile.

Il reverse flow routing può infatti essere considerato un problema di regressione multipla in cui, date n misurazioni a valle, si deve determinare l'altezza del fiume a monte.
È importante tenere in considerazione che per ottenere risultati utili, alla luce delle richieste degli ingegneri, è necessaria la massima precisione nei picchi, piuttosto che nei periodi di secca, considerati meno rilevanti.
Inoltre è indispensabile che il tempo di training, così come quello di previsione, siano ragionevoli.
Il percorso del fiume Secchia è però caratterizzato dalla presenza di numerose pianure alluvionali e aree allagabili che influenzano la propagazione della piena, e che quindi rendono più complicato l'apprendimento da parte della rete.

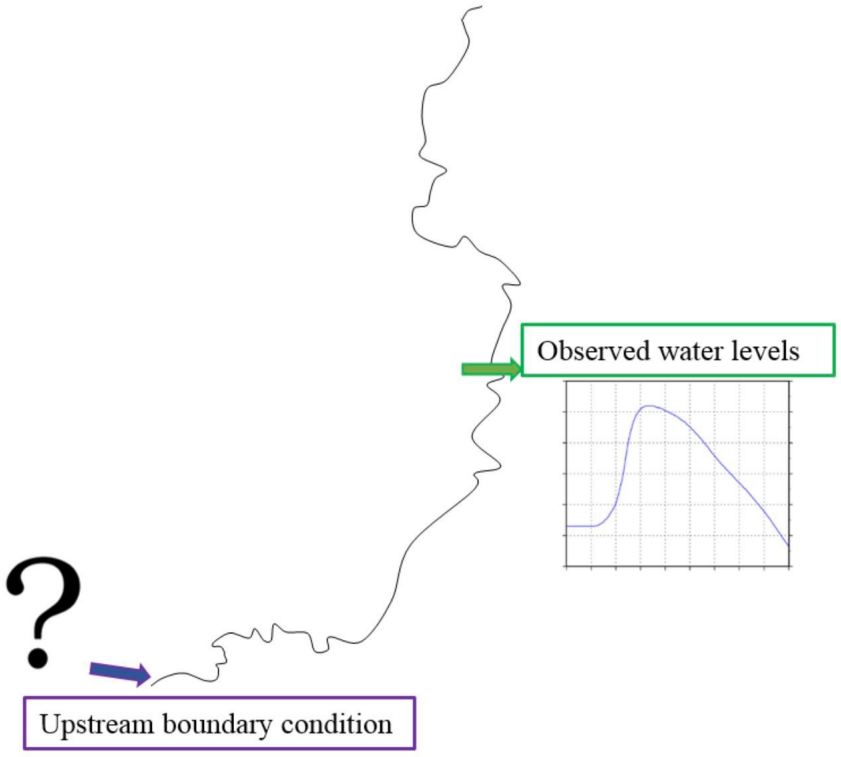

Figura 2.1: Il problema del reverse flow routing per il fiume Secchia

Capitolo 3

Metodi

3.1 La struttura della rete

Il primo obiettivo era riuscire a trovare il modello ottimale per la rete e stabilire un corretto valore di epoche di apprendimento, oltre che un corretto valore del learning rate, in modo da ottenere buoni risultati senza incorrere nell'overfitting.
Dopo numerosi test cercando di approssimare nel modo più corretto possibile il livello del fiume Secchia nell'anno 2017 utilizzando 3 anni per addestrare la rete (2016, 2015 e 2014), il modello che ha raggiunto la maggiore precisione è strutturato nel seguente modo:
Un primo layer composto da 64 unità;
Un secondo, terzo e quarto layer composti da rispettivamente 256, 128 e 64 unità;
Un layer finale composto da una sola unità visto che l'output della rete sarà di un solo valore.
In questo caso tutti gli strati sono densi (fully connected layer), ovvero tutti i nodi di un layer sono connessi a tutti i nodi del successivo, e la funzione di attivazione utilizzata è la Relu (Figura 3.1).
Rimane ancora da individuare il numero di misurazioni in input per ottenere l'approssimazione più accurata possibile.

Anche in questo caso ho utilizzato un metodo pragmatico.
Ho settato la rete con diverse quantità di sample in ingresso per calcolare per ogni previsione dell'anno l'errore quadratico medio sulle misurazioni dell'altezza del fiume.
I risultati del test, dove si può verificare che è 10 il numero ideale di misurazioni in input, sono riportati nella sezione 4.1.
Date quindi 10 misurazioni successive (una ogni mezzora, per un totale di 5 ore di misurazioni) registrate nella stazione di Bacchello, il modello calcola il livello del fiume nella stazione di ponte Alto nello stesso momento della prima misurazione utilizzata in input.

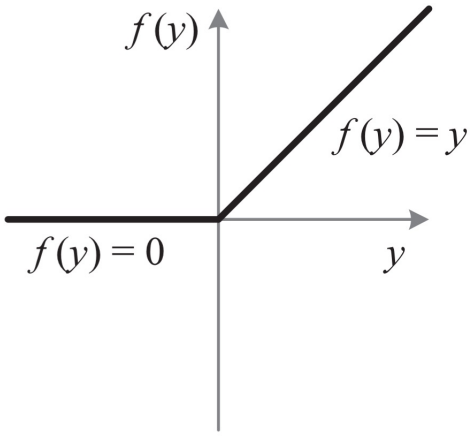

Figura 3.1: Funzione Relu

3.2 Gestione dei dati

Per eseguire il codice è necessario creare due file .txt per ogni anno, uno contenente le misurazioni dell'altezza del fiume in metri misurate ogni 30 minuti nella stazione di Bacchello e uno con quelle di ponte Alto.
Sarà poi il programma a creare la matrice che il modello utilizzerà come dato in input.

Per l'output invece, visto che la stima, date 10 altezze, è di un solo valore, basterá un semplice array con le misurazioni a ponte Alto dell'anno di cui si vorrà calcolare l'approssimazione.
Entrando nel dettaglio, la matrice verrà strutturata nel seguente modo:
Nella i-esima riga saranno presenti l'i-esimo livello del fiume a Bacchello più le successive 9 misurazioni, necessarie per calcolare in modo ottimale l'i-esima altezza nella stazione di ponte Alto, ovvero il valore nella i-esima posizione dell'array di output (Tabella 3.1).
La rete, grazie a queste due strutture (la matrice e l'array), sarà in grado di calcolare il valore dei pesi per ogni arco in modo da ottenere un risultato finale il più accurato possibile rispetto alla reale misurazione.

Matrice di 10 colonne										**Array**
1,89	1,89	1,9	1,89	1,88	1,87	1,86	1,86	1,88	1,87	−0,1
1,89	1,9	1,89	1,88	1,87	1,86	1,86	1,88	1,87	1,87	−0,09
1,9	1,89	1,88	1,87	1,86	1,86	1,88	1,87	1,87	1,87	−0,12
1,89	1,88	1,87	1,86	1,86	1,88	1,87	1,87	1,87	1,85	−0,15
1,88	1,87	1,86	1,86	1,88	1,87	1,87	1,87	1,85	1,85	−0,19
1,87	1,86	1,86	1,88	1,87	1,87	1,87	1,85	1,85	1,82	−0,21
1,86	1,86	1,88	1,87	1,87	1,87	1,85	1,85	1,82	1,81	−0,25
1,86	1,88	1,87	1,87	1,87	1,85	1,85	1,82	1,81	1,79	−0,27
1,88	1,87	1,87	1,87	1,85	1,85	1,82	1,81	1,79	1,76	−0,29
1,87	1,87	1,87	1,85	1,85	1,82	1,81	1,79	1,76	1,75	−0,28

Tabella 3.1: 10 righe della Matrice e dell'Array relativi al 2017

3.3 Analisi del codice

In questo paragrafo andrò ad analizzare nello specifico lo script utilizzato per definire il modello, ricordando che il linguaggio di programmazione utilizzato è python.
Nelle seguenti righe di codice viene definita la struttura della rete come visto precedentemente.

```
import tensorflow as tf

def build_model():
    model = keras.Sequential([
        keras.layers.Dense(64, activatio=tf.nn.relu,
                           input_shape=(train_data.shape[1],)),
        keras.layers.Dense(256, activation=tf.nn.relu),
        keras.layers.Dense(128, activation=tf.nn.relu),
        keras.layers.Dense(64, activation=tf.nn.relu),
        keras.layers.Dense(1)
    ])
    optimizer = tf.train.RMSPropOptimizer(0.0001)
    model.compile(loss='mse', optimizer=optimizer,
                  metrics=['mae'])
    return model
```

Si può notare in modo chiaro la struttura composta da 5 strati e il loro numero di nodi.
Nel primo layer viene specificata la matrice in input, la cui variabile assume il nome di train_data.
Viene anche impostato il valore del learning rate a 0,0001 e viene specificato l'utilizzo di una discesa del gradiente stocastica ovvero il RMSProp (Root Mean Squared Propagation).

RMSProp utilizza lo stesso principio della discesa del gradiente, e l'unica differenza è nell'aggiornamento dei pesi.
In poche parole, invece di utilizzare le derivate indipendentemente per ogni epoca, utilizza le medie ponderate esponenzialmente del quadrato delle derivate.
Per quanto riguarda il loss, valore che identifica quanto è cattiva la previsione del modello su un singolo esempio, viene definito usando mse (mean squared error, l'errore quadratico medio).
La metrica utilizzata per valutare l'accuratezza della rete è mae ovvero mean absolute error (errore assoluto medio).

```
EPOCHS = 300

history = model.fit(train_data, train_label, epochs=EPOCHS,
                    validation_split=0.2, verbose=0,
                    callbacks=[PrintDot()])

test_predictions = model.predict(test_data).flatten()

np.savetxt("predizione.txt", test_predictions, fmt='%1.3f')
```

Nella seconda parte del codice viene impostato il numero di epoche di apprendimento a 300, e viene addestrato il modello specificando diversi fattori. Dobbiamo infatti definire la matrice in input e l'arrey di output, oltre che il numero di epoche precedentemente fissato.
Un altro dato essenziale da inserire è la percentuale di validation set rispetto al training set, in questo caso impostata a 20%.
Il validation set è un elemento molto importante per ottenere buoni risultati, infatti la rete non viene allenata su questi dati, ma vengono utilizzati per ottenere una valutazione obbiettiva dell'accuratezza del modello durante l'addestramento della rete.

La rete però non "impara" da questi dati e non aggiorna i propri pesi in base al risultato ottenuto.
La matrice prima di essere separata nei due set (training e validation) viene mescolata in modo casuale, e di conseguenza anche l'array dell'output, in modo che il validation set sia il più generico possibile e non sia una specifica porzione di un anno.
La funzione PrintDot serve semplicemente a mostrare a video la progressione dell'allenamento, stampando dei punti ogni 100 epoche di addestramento.
La predizione viene poi salvata dentro l'array test_predicition e grazie al comando savetxt viene copiata sul file predizione.txt che potremo utilizzare per stampare il grafico e confrontarlo con i dati reali.

3.4 Tecniche per cercare di aumentare la precisione della rete

Un buon metodo per cercare di raggiungere una maggiore precisione è inserire più misurazioni possibili nella fase di addestramento.
Un primo approccio può essere quello di utilizzare più anni per allenare il modello, dando alla rete la possibilità di apprendere da una quantità maggiore di informazioni.
Un'altra soluzione può essere l'introduzione di un soglia.
I problemi principali nell'approssimazione dell'altezza del fiume sembrano essere concentrati nei picchi in cui si raggiungono altezze molto elevate, anche superiori agli 8 metri.
Stabilendo quindi un valore minimo, e andando a settare tutte le misurazioni inferiori a quest'ultimo a tale valore, la rete si concentrerà maggiormente sui periodi in cui il livello del fiume raggiunge valori significativi, ottenendo quindi una maggiore accuratezza.
Il valore della soglia non deve essere troppo elevato altrimenti si correrà il rischio di eliminare troppe informazioni utili.

L'obiettivo è quello di ridurre gli errori quando il fiume è in secca, ottenendo di conseguenza, calcolando l'errore medio, un ottimo indice di accuratezza della previsione durante i picchi.
Nel caso del fiume Secchia, impostando la soglia uguale a mezzo metro, riusciremo a liberarci di molte misurazioni, probabilmente affette da errore, e poco importanti per il risultato finale.
Quest'ultime, non solo assumono un valore numericamente basso, ma oscillano in modo periodico, palesando un possibile errore strumentale dovuto al ridotto livello dell'acqua da misurare.

3.4.1 Interpolazione

Avendo a disposizione i dati relativi a 6 anni, un altro modo per incrementare il training set è l'utilizzo dell'interpolazione.
Con interpolazione si intende un metodo per individuare nuovi punti a partire da un insieme finito di punti conosciuti.
Nello specifico andrò a utilizzare uno dei metodi più semplici, l'interpolazione lineare.
Per ogni coppia di punti dati consecutivi, chiamati in questo caso (x_a, y_a) e (x_b, y_b), definiamo come funzione interpolante nell'intervallo $[x_a, x_b]$ la:

$$f(x) = \frac{x - x_b}{x_a - x_b} y_a - \frac{x - x_a}{x_a - x_b} y_b \tag{3.1}$$

Utilizzando questa formula, se volessimo calcolare il valore della funzione con argomento $2,5$, essendo il punto medio tra 2 e 3 , $f(2,5)$ avrá come valore il punto medio tra $f(2)$ e $f(3)$.
Come analizzerò nel dettaglio nel capitolo successivo, l'utilizzo di tutte queste tecniche ha aumentato di molto la precisione rispetto alla prima approssimazione calcolata.
Si possono notare infatti tanti miglioramenti tra la previsione utilizzando solo 3 anni nel training rispetto a 5 anni interpolati e con l'introduzione della soglia.

Capitolo 4

Risultati

In questo capitolo mostrerò i vari test e relativi risultati ottenuti, andando ad analizzare nello specifico ogni prova eseguita.
Il setup della rete utilizzato è descritto nel dettaglio nella sezione precedente. Il modello è strutturato in 5 layer fortemente connessi, il validation set é composto dal 20% del training set, e il learning rate è settato a $0,0001$.
Tutti i neuroni utilizzano la Relu come funzione di attivazione.
La macchina utilizzata presenta le seguenti caratteristiche:

- Sistema Operativo: Ubuntu 16.04 LTS 64-bit
- Processore: Intel i7-6500U (4 core per $2,50$ GHz)
- Scheda grafica: GeForce GTX 950M

Il tempo di training, utilizzando le misurazioni relative a 3 anni, è di 8 minuti. Salirà a 13 quando invece adopereremo 5 anni.
Chiaramente, triplicando i dati presenti nel training set grazie all'interpolazione, triplicherà anche il tempo, che raggiungerà i 39 minuti.

4.1 Numero di sample in input

Nel primo test cercherò di individuare il numero di sample, ovvero il numero delle osservazioni misurate a Bacchello ogni 30 minuti, in ingresso per otte-

nere l'output, ovvero la previsione dell'altezza del fiume a ponte Alto, il più accurato possibile.
Il risultato è dato dai grafici 4.1, 4.2 e 4.3.
L'analisi è stata effettuata con 6 prove, rispettivamente 5, 8, 10, 12, 15 e 20 sample in input, con un training set contenente le misurazioni di 3 anni.
Utilizzando come metro di giudizio l'errore quadratico medio della differenza tra la stima del livello del fiume e la reale altezza di quest'ultimo, il numero di sample ottimale in input è 10, per tutti i 3 anni utilizzati.
Meno sample non bastano a catturare gli effetti di propagazione dell'acqua nel fiume durante i 14 km di distanza tra le 2 stazioni utilizzate, mentre l'utilizzo di più sample, al contrario, non permettono un training efficace a parità di input disponibili.

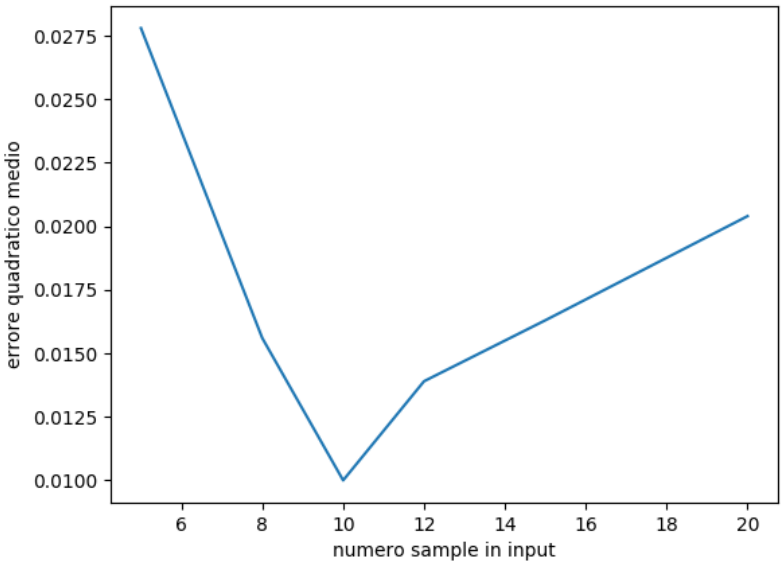

Figura 4.1: Anno 2015

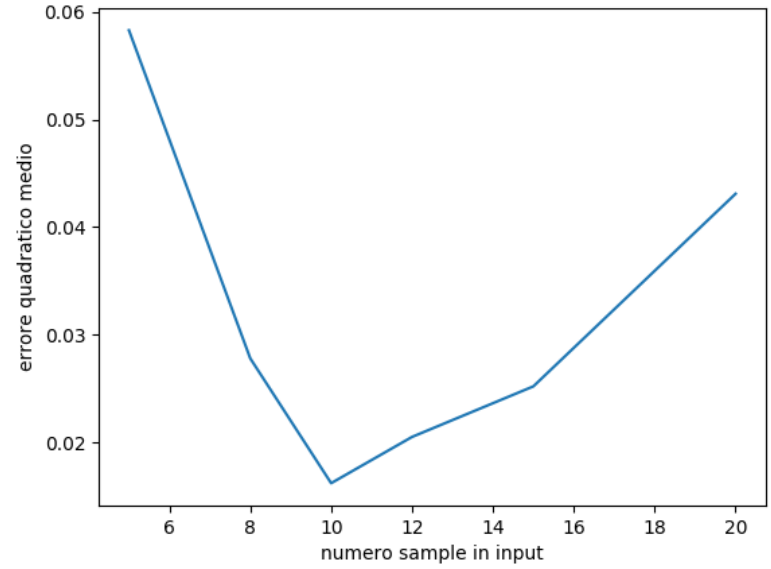

Figura 4.2: Anno 2016

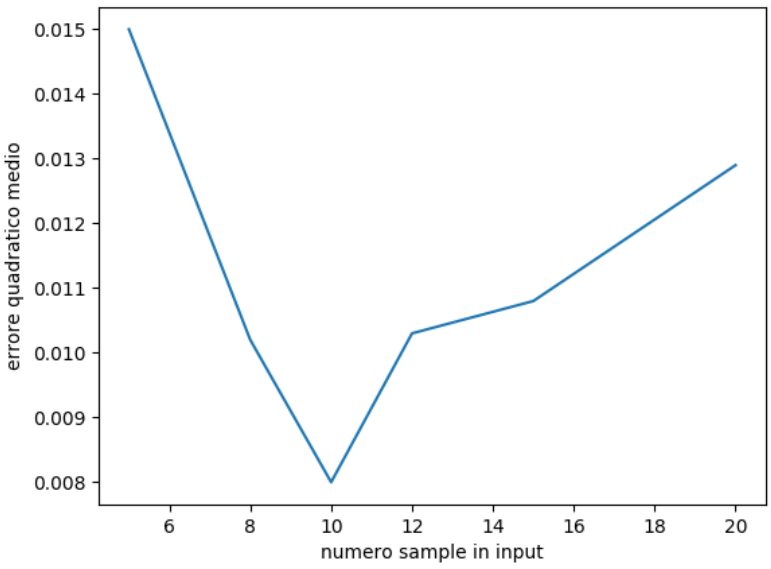

Figura 4.3: Anno 2017

4.2 Test con la rete

Inizialmente ho lavorato usando solo 4 anni, 3 utilizzati per allenare la rete, ovvero per il training set, e uno per il test set.
Gli anni impiegati sono i seguenti: 2014, 2015, 2016 e 2017.
Ho stimato, a rotazione, l'altezza del fiume per ogni anno, in modo da verificare che la rete fosse in grado di approssimare con successo tutti gli anni che avevo a disposizione.
Nei grafici utilizzo il colore azzurro per rappresentare il reale andamento del livello dell'acqua, mentre l'arancione rappresenta il valore stimato dalla rete.

4.2.1 Predizione di un anno dati 3 anni

Come si puó notare dal grafico 4.4 la stima dell'anno 2017 è buona quando l'altezza del fiume si mantiene bassa (Figura 4.5).
La rete peró non riesce a gestire al meglio i picchi di piena, nonostante ciò l'errore quadratico medio ha un valore di 0,84 centimetri (l'errore massimo misurato nei picchi é di 40 cm).
Per quanto riguarda l'approssimazione del 2016 (vedere grafico 4.6) la situazione è analoga a quella del 2017, e l'errore quadratico medio assume un valore di 1,62 centimetri.

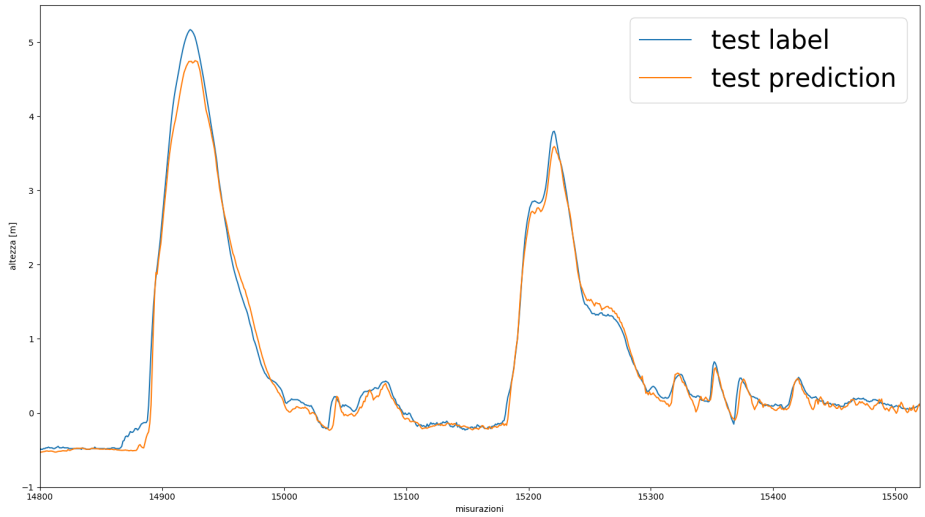

Figura 4.4: Previsione anno 2017, con training di 3 anni, dal 5 al 20 novembre

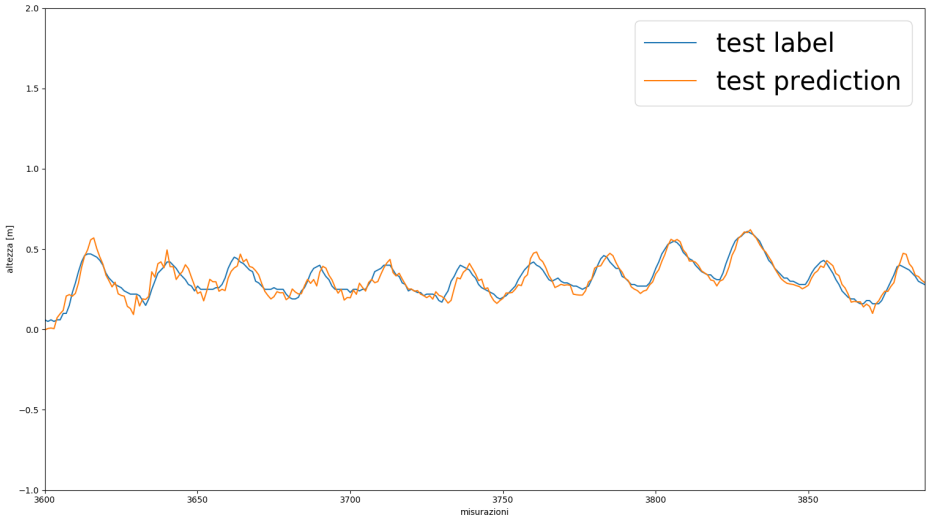

Figura 4.5: Previsione valori bassi dell'anno 2017 in una finestra di 7 giorni

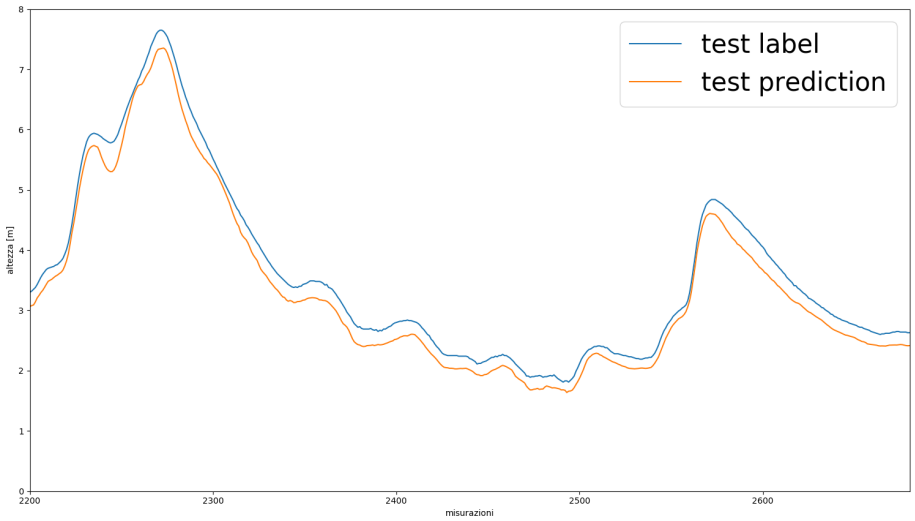

Figura 4.6: Previsione anno 2016, con training di 3 anni, dal 15 al 25 febbraio

Per quanto riguarda gli anni 2015 e 2014 i risultati sono contrastanti.
Nel primo caso il modello prevede bene anche i periodi di piena (Figura 4.7), anche se si può notare qualche piccolo errore di approssimazione negli ultimi 3 picchi dell'anno, dove l'errore misurato è di 13 cm, 19cm e 20 cm.
Per il 2014 il discorso è diverso, la gestione dei picchi da parte della rete non è buono.
Il periodo di piena nella prima parte dell'anno non viene calcolato con successo, inoltre possiamo notare errori non irrilevanti che fanno salire l'errore quadratico medio a 5,16 cm (Figura 4.8), mentre quello del 2015 ha un valore di 1 cm.
La stima del livello del fiume a gennaio presenta un dislivello rispetto ai dati reali dai 20 ai 40 cm per poi raggiugere 1 metro in corrispondenza del 19 gennaio.
Il 19 gennaio infatti c'è stata una rotta del Secchia a valle del ponte Alto e la chiusura finale dell'argine si è verificata dal 21 al 22 gennaio.
Dopo aver effettuato la misurazione a ponte Alto buona parte dell'acqua

è uscita dal fiume andando ad allagare le campagne e i paesi circostanti, pertanto le misurazioni a Bacchello sono state ”invalidate”.
Quando la rete infatti prova a ricostruire i livelli del 19 gennaio utilizzando i dati di Bacchello non può certo ricostruire correttamente i valori e andrà a sottostimare l'altezza del fiume.
La rete non riesce giustamente a gestire il volume mancante, visto che assume sempre che tutta l'acqua arrivi a valle.

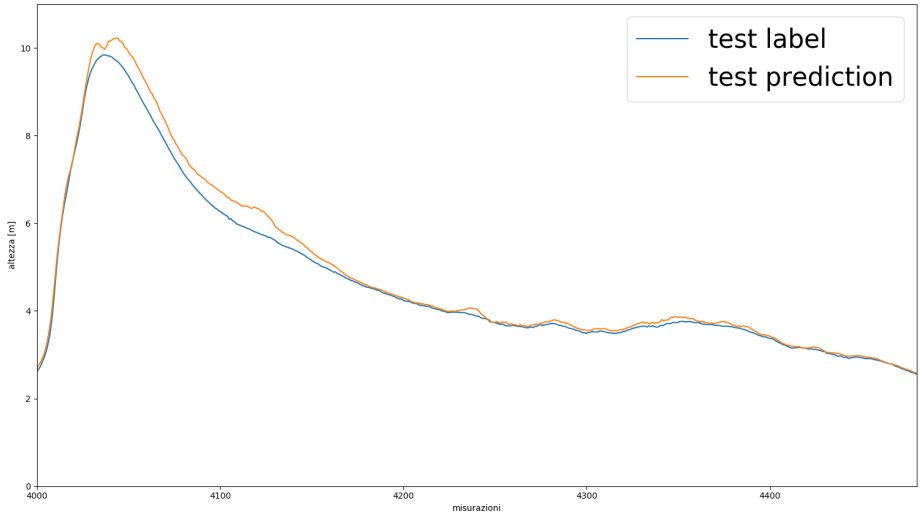

Figura 4.7: Previsione anno 2015, con training di 3 anni, dal 25 marzo al 4 aprile

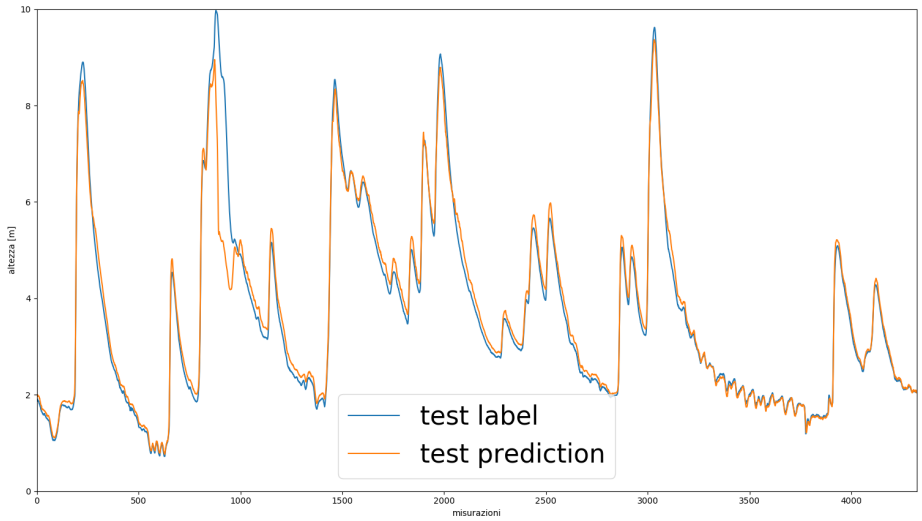

Figura 4.8: Previsione anno 2014, con training di 3 anni, dal 1 gennaio al 31 marzo

Nonostante gli errori concentrati nei picchi, le approssimazioni seguono molto bene l'andamento del fiume durante tutto l'anno.
Qui di seguito, nella figura 4.9, mostro la stima del livello del fiume durante tutto il 2014.

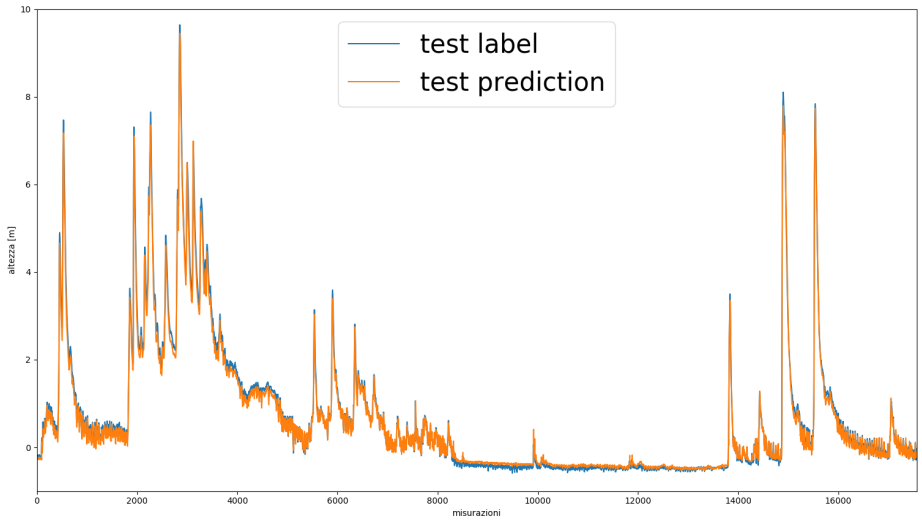

Figura 4.9: Previsione anno 2014 con training di 3 anni

4.2.2 Predizione di un anno dati 5 anni

Come era prevedibile, utilizzando più dati per allenare la rete si ottengono stime più precise.
Per gli anni 2017 e 2016 si può notare qualche miglioramento nella gestione dei picchi anche se l'approssimazione è comunque inferiore al valore reale (Figure 4.10 e 4.11).
L'errore quadratico scende a 0,74 cm per il 2017 e a 1,45 cm per il 2016.
Nel 2015 otteniamo come precedentemente ottimi risultati nei picchi tranne nell'ultimo periodo dell'anno dove però possiamo notare un lieve miglioramento (l'errore di 19 cm per il picco che raggiunge i 4,5 metri scende a 7 cm), anche se si è persa un po' di precisione nelle misurazioni basse (Figura 4.12).
Il 2015 infatti è l'unico anno in cui l'errore medio aumenta e assume il valore di 1,2 cm.
Dobbiamo tenere in considerazione che i valori interessanti per previsioni o ricostruzioni di eventi sono i picchi e non i valori bassi.
È quindi preferibile ottenere una maggiore precisione durante i periodi di piena rispetto a quelli di secca.

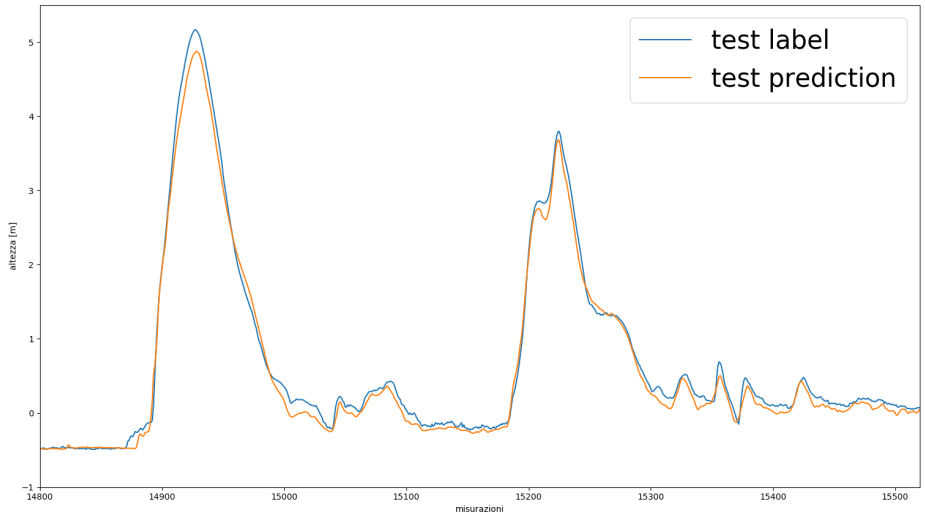

Figura 4.10: Previsione anno 2017, con training di 5 anni, dal 5 al 20 novembre

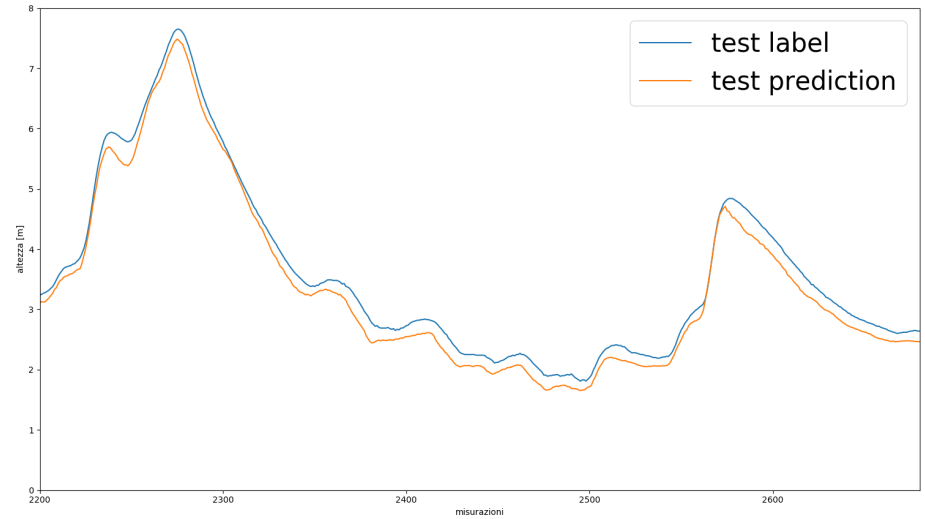

Figura 4.11: Previsione anno 2016, con training di 5 anni, dal 15 al 25 febbraio

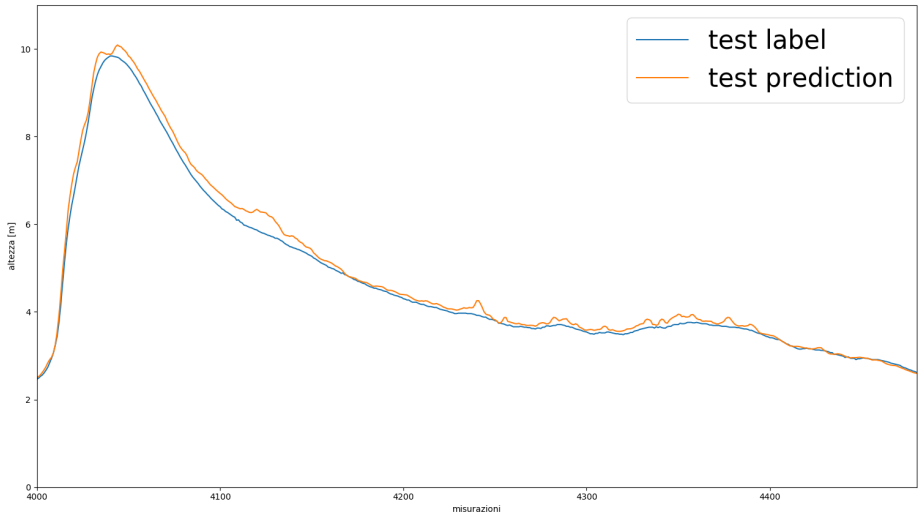

Figura 4.12: Previsione anno 2015, con training di 5 anni, dal 25 marzo al 4 aprile

Grazie alla maggior quantità di dati su cui addestrare il modello, la stima sull'anno 2014 migliora notevolmente.
L'unico errore rilevante è la piena del 19 gennaio, giustificato dalla rottura dell'argine, che la rete non riesce a predire (l'errore è ancora molto elevato e raggiunge il valore di 60 cm), e anche una piccola imprecisione di 15 cm durante la giornata del 31 luglio, e una meno rilevante intorno al 18 settembre.
I restanti picchi dell'anno sono approssimati in modo corretto, con qualche piccolo errore che non supera i 10 cm (Figura 4.13).

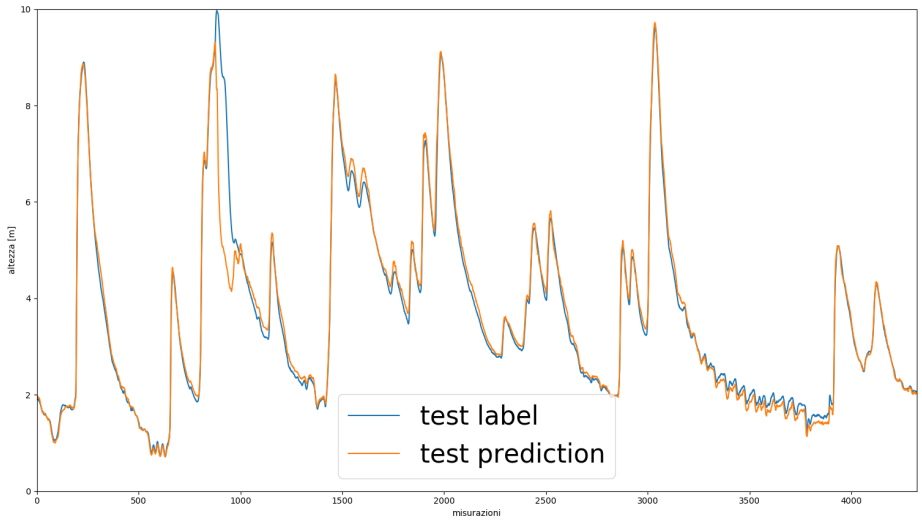

Figura 4.13: Previsione anno 2014, con training di 5 anni, dal 1 gennaio al 31 marzo

Anche i nuovi anni introdotti, ovvero il 2011 e il 2012, hanno un comportamento simile ai precedenti.
Il modello riesce a stimare bene i valori bassi ma ha qualche piccola imprecisione, concentrata nei picchi più significativi, che raggiungono il valore di 30 e 20 cm rispettivamente per il 2012 (figura 4.14) e il 2011 (figura 4.15)

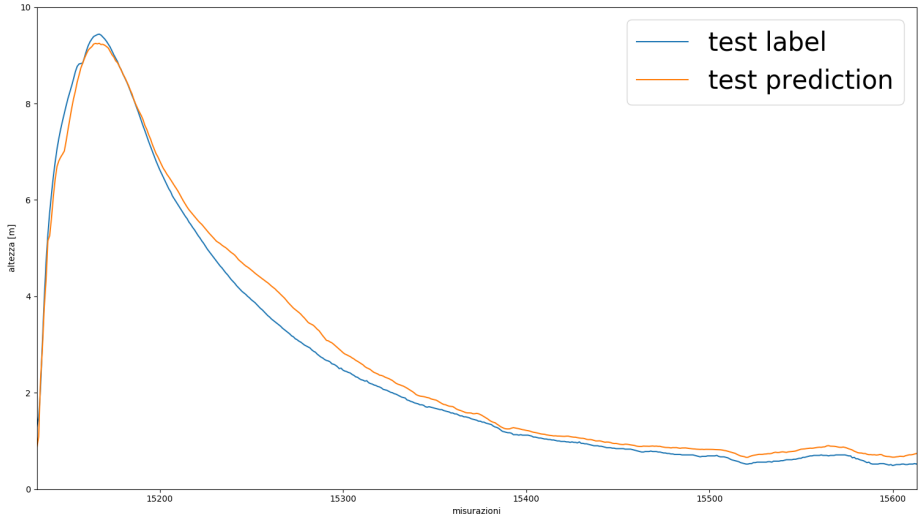

Figura 4.14: Previsione anno 2012, con training di 5 anni, dal 11 al 21 novembre

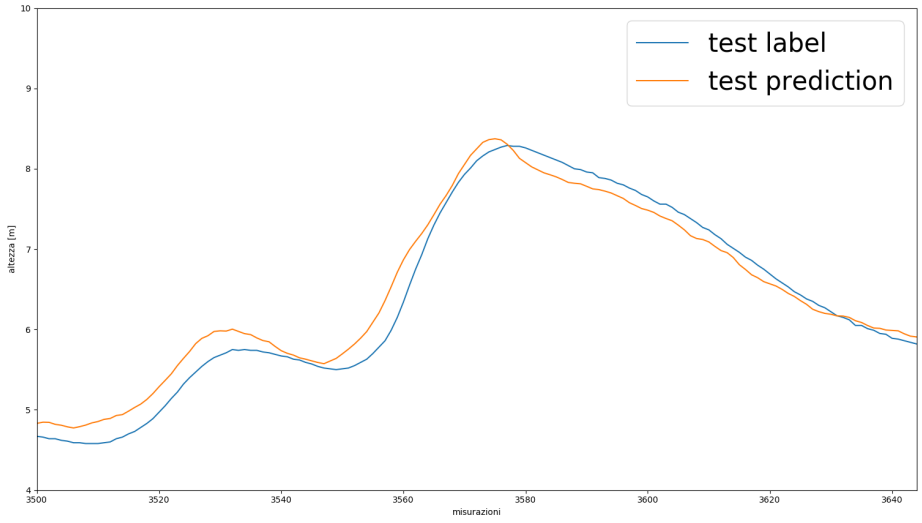

Figura 4.15: Previsione anno 2011, con training di 5 anni, dal 14 al 17 marzo

4.3 Test stabilendo una soglia minima

Siccome il problema principale era riuscire a stimare il livello del fiume Secchia durante i periodi di piena, per aiutare la rete abbiamo stabilito una soglia minima di 50 cm.
In poche parole ho impostato un valore minimo e ho modificato i dati andando ad assegnare a tutte le misurazioni inferiori al mezzo metro il valore della soglia.
In questo modo la rete si concentra maggiormente sui picchi e non perde tempo a imparare tutte le minime ma numerose variazioni del livello dell'acqua, il cui moto armonico (come si può vedere dalla figura 4.16) sembra far intuire che siano presenti errori causati dallo strumento utilizzato per misurare l'altezza del fiume.
Gli esperti del settore confermano infatti che per valori rilevati molto bassi ci sono errori di lettura dovuti al modo in cui è costruito il sistema di rilevazione. Inoltre questi dati sono di interesse molto basso rispetto ai periodi di piena.
Come ci si poteva aspettare, considerando che il modello si sarebbe concentrato esclusivamente sui livelli del fiume significativi, negli anni 2017 e 2016 (Figure 4.17 e 4.18) l'approssimazione dei picchi è migliorata ulteriormente ma solo di qualche cm, sono infatti ancora presenti errori, più o meno di una quindicina di cm per il 2017 (che in due punti raggiungono il valore di 30 cm) e di 20 cm per il 2016.
Tali errori sono comunque considerati molto buoni al fine di previsioni o ricostruzioni di eventi.
Questo è il feedback degli esperti, che conferma la ottima riuscita dell'approccio usato.

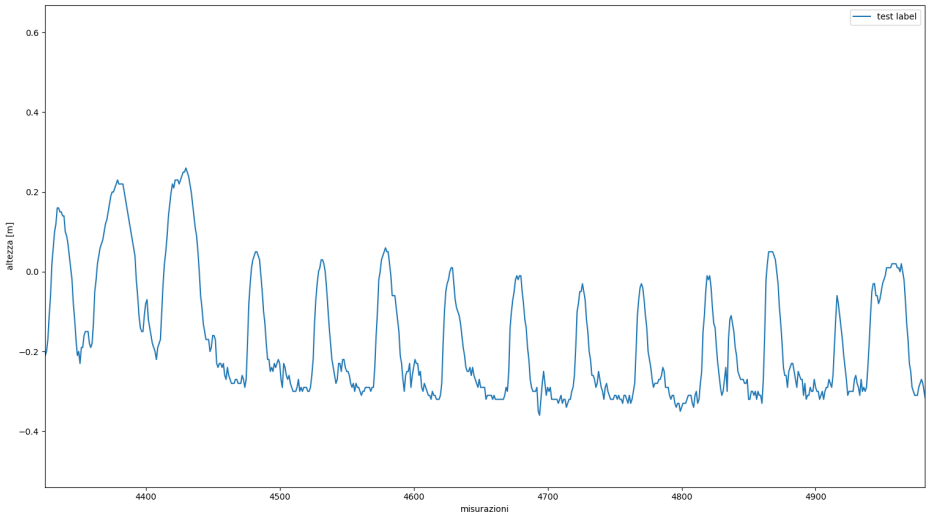

Figura 4.16: Moto armonico delle misurazioni nel 2017 in una finestra di 2 settimane

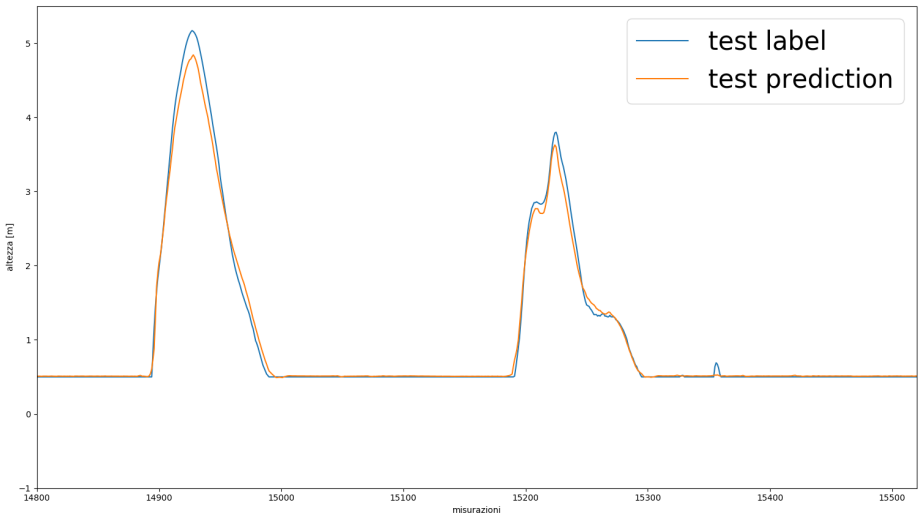

Figura 4.17: Previsione anno 2017, con soglia di 50 cm, dal 5 al 20 novembre

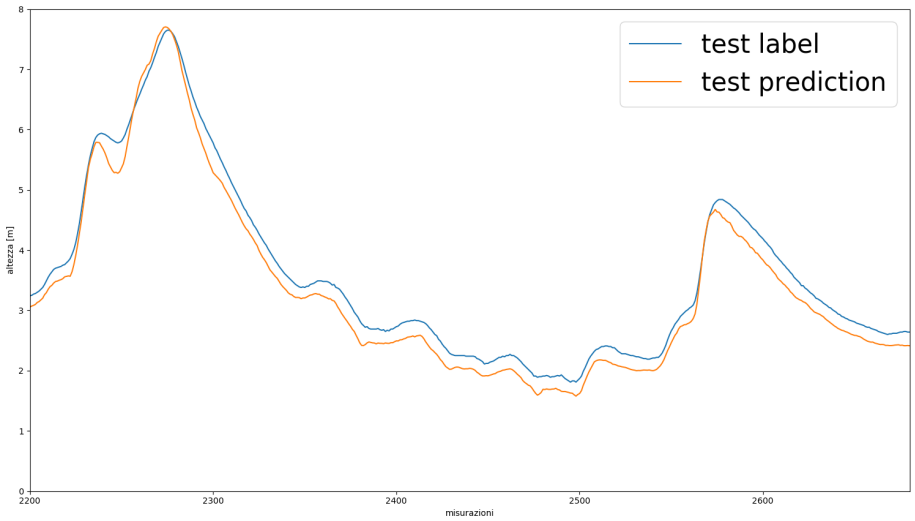

Figura 4.18: Previsione anno 2016, con soglia di 50 cm, dal 15 al 25 febbraio

Per quanto riguarda invece il 2011 e il 2012 il risultato ottenuto è davvero ottimo.
La stima durante i periodi di piena è molto efficace e non sono presenti errori rilevanti nel calcolo dei picchi (in media nel 2011 risulta un'imprecisione di 10 cm, nel 2012 invece di 20 cm, Figure 4.19 e 4.20).
Anche per l'anno 2014 si può notare un'ottima approssimazione e gli unici problemi sono relativi al solito 19 gennaio (il cui errore ha un valore di 65 cm) e al 31 luglio dove il livello del fiume viene sottostimato di 15 cm.

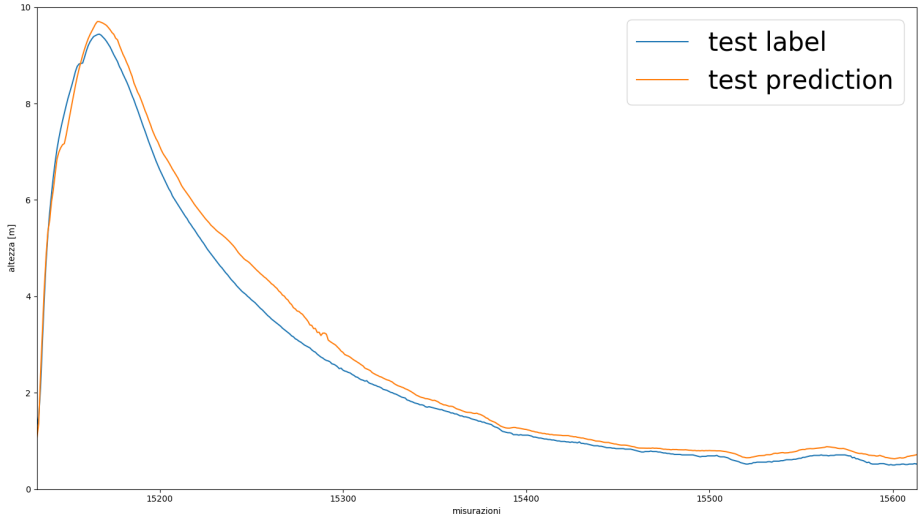

Figura 4.19: Previsione anno 2012, con soglia di 50 cm, dal 11 al 21 novembre

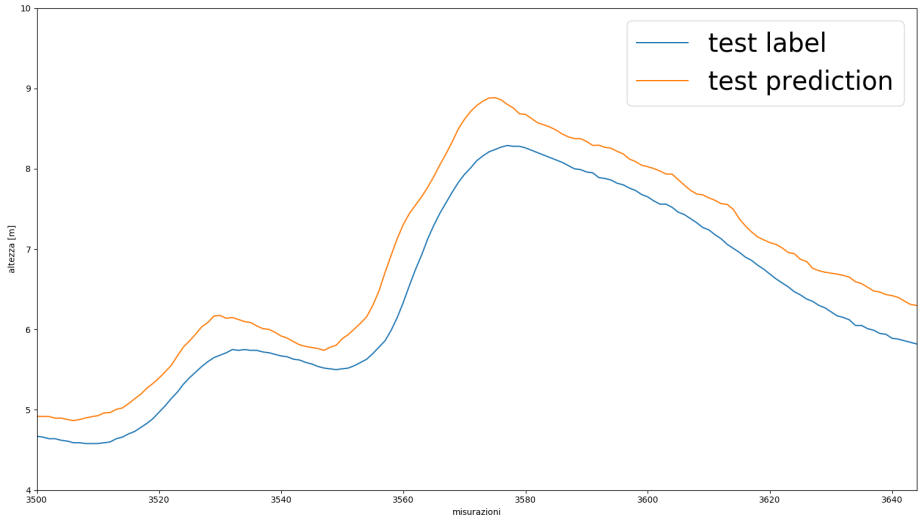

Figura 4.20: Previsione anno 2011, con soglia di 50 cm, dal 14 al 17 marzo

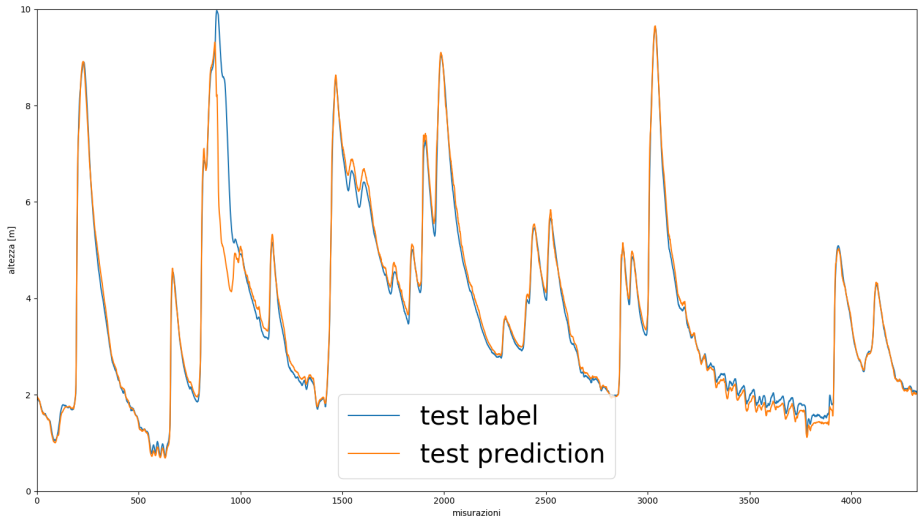

Figura 4.21: Previsione anno 2014, con soglia di 50 cm, dal 1 gennaio al 31 marzo

L'approssimazione del 2015 al contrario non migliora nei punti critici (i soliti 3 picchi a fine anno che la rete non era in grado di calcolare), anzi vi è un leggero peggioramento (Figura 4.22).
Naturalmente l'imprecisione sembra essere limitata in quei punti visto che gli altri periodi di piena vengono gestiti in maniera ottima.

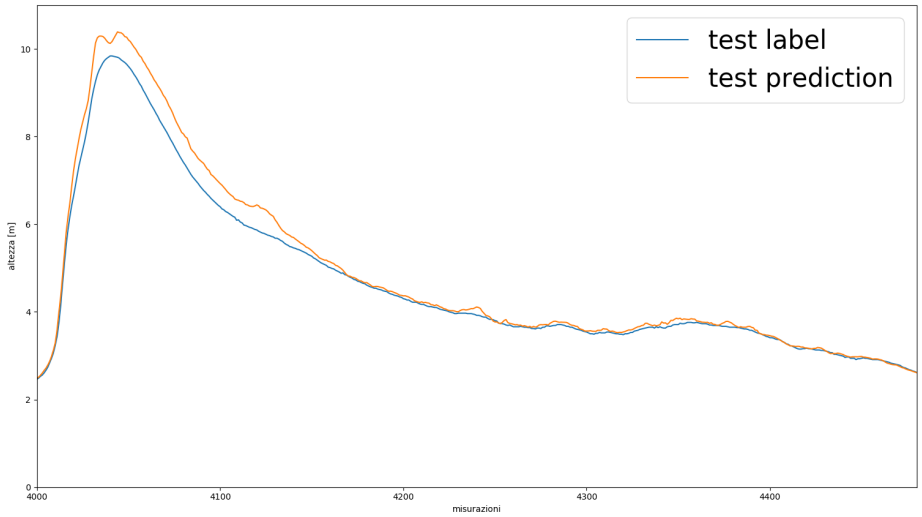

Figura 4.22: Previsione anno 2015, con soglia di 50 cm, dal 25 marzo al 4 aprile

L'unico effetto visibile dell'introduzione della soglia è stato l'aumento di precisione nei picchi di una decina di cm per il 2011 e il 2012.
Per gli altri anni il miglioramento (o peggioramento per il 2015) è minimo, ciò mostra che la rete era già in grado di gestire al meglio i periodi di piena.

4.4 Test utilizzando il metodo dell'interpolazione lineare

Per aumentare ulteriormente il numero di dati da utilizzare nel training set ho utilizzato il metodo dell'interpolazione lineare, ovvero ho aggiunto, nello specifico, altre 2 misurazioni consistenti tra due misurazioni originali.
Avendo a disposizione più informazioni su cui addestrare la rete quello che ci si aspetta è che sarà possibile aumentare il numero di sample in input per ottenere una maggiore precisione.
Per calcolare il valore delle nuove misurazioni da inserire ho semplicemente considerato il passaggio di un valore al successivo come se l'acqua aumentasse in modo costante in quella mezzora, ed ho quindi ricavato l'altezza del fiume dopo 10 e 20 minuti, utilizzando la formula 3.1 descritta nella sezione 3.4.
Il numero di sample in input ottimale in questo caso non é 28 (9 misurazioni originali interpolate, quindi 27, più una) ma 34 ovvero come se stessimo utilizzando 12 misurazioni originali.
La rete corrispondente permette di estendere maggiormente il numero di sample influenzate da un evento, grazie al fatto di avere a disposizione un numero triplicato di osservazioni in input.
Tutti gli anni vengono approssimati con un'ottima accuratezza, andando ad aumentare di molto la precisione generale della rete.
Il 2017 in figura 4.23 non si rilevano grossi errori, che infatti non superano i 10 cm durante tutto l'anno, e l'errore quadratico medio è di $0,51$ cm.
Lo stesso vale per il 2016 (in Figura 4.24) dove vengono gestiti in modo perfetto i picchi più elevati anche se rimane qualche sottostima dell'altezza di 15 cm intorno al 18 maggio, e l'errore quadratico medio è di $0,88$ cm.
Per l'anno 2015 (Figura 4.25), dove avevamo già un ottima approssimazione riusciamo finalmente a ottenere un ulteriore miglioramento dei 3 picchi finali (si parla comunque di una sottostima di 15 cm).
L'errore quadratico medio assume un valore di $0,61$ cm.

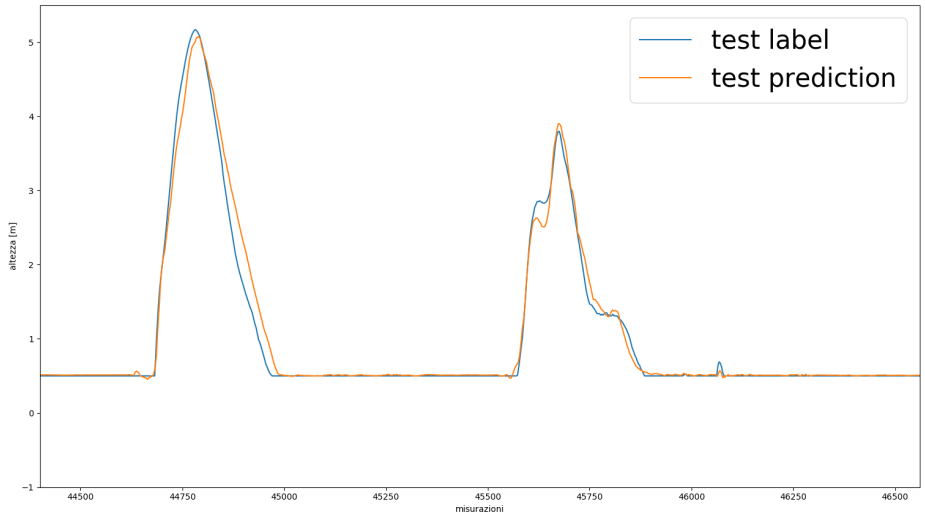

Figura 4.23: Previsione anno 2017, con interpolazione e soglia, dal 5 al 20 novembre

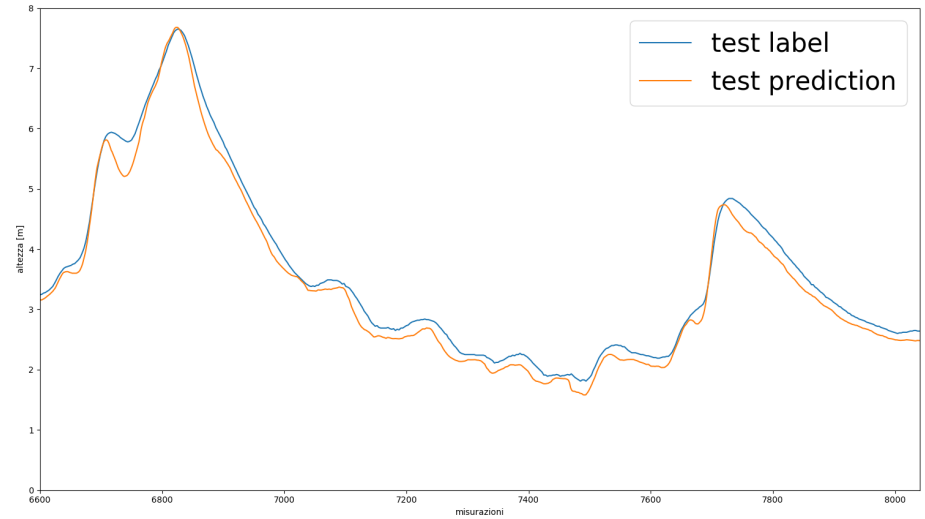

Figura 4.24: Previsione anno 2016, con interpolazione e soglia, dal 15 al 25 febbraio

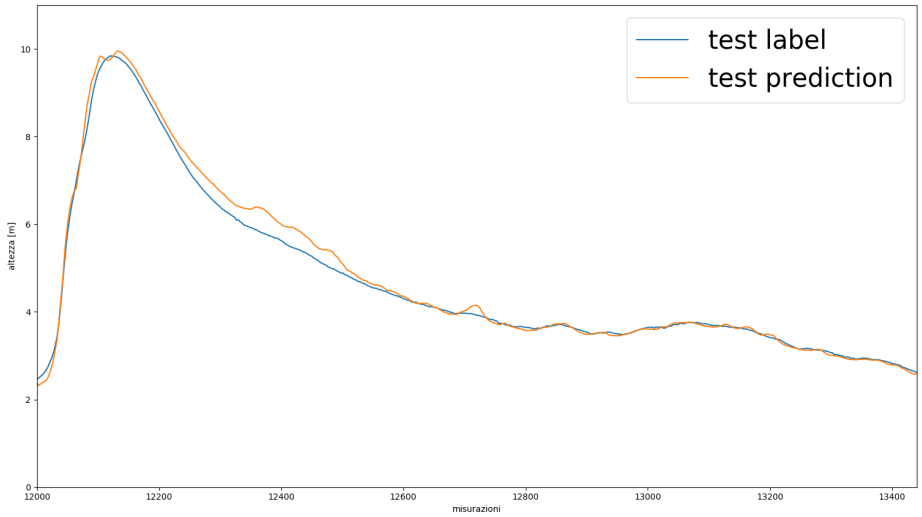

Figura 4.25: Previsione anno 2015, con interpolazione e soglia, dal 25 marzo al 4 aprile

Nei grafici mostrati in figura 4.26 e 4.27 si può notare come il modello riesca a ottenere un'approssimazione molto simile ai reali valori del fiume.
Non sono presenti errori rilevanti di sottostima, e nei picchi l'errore si stabilizza sui 10 cm massimo, anche se in certi periodi di piena del 2012 otteniamo una sovrastima di una ventina di centimetri.
L'errore quadratico medio nel 2011 è di $1,5$ cm, mentre quello del 2012 è di 1 cm.

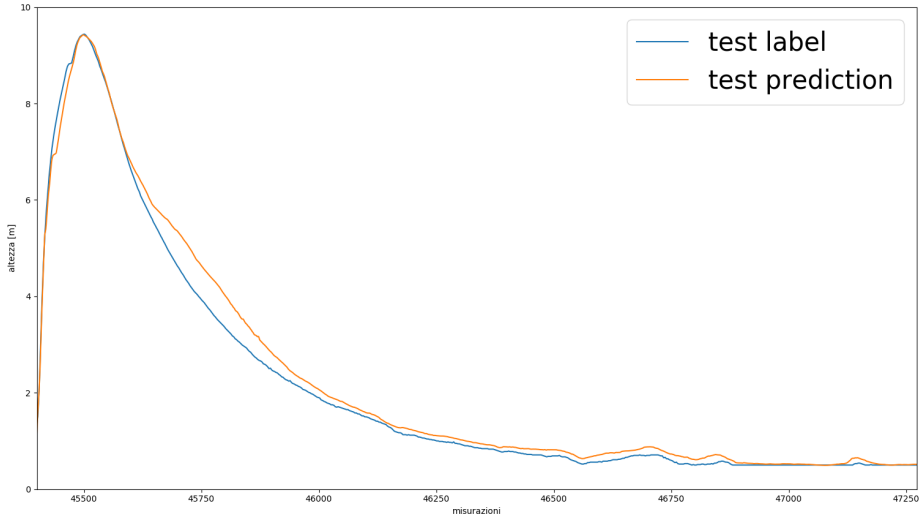

Figura 4.26: Previsione anno 2012, con interpolazione e soglia, dal 11 al 21 novembre

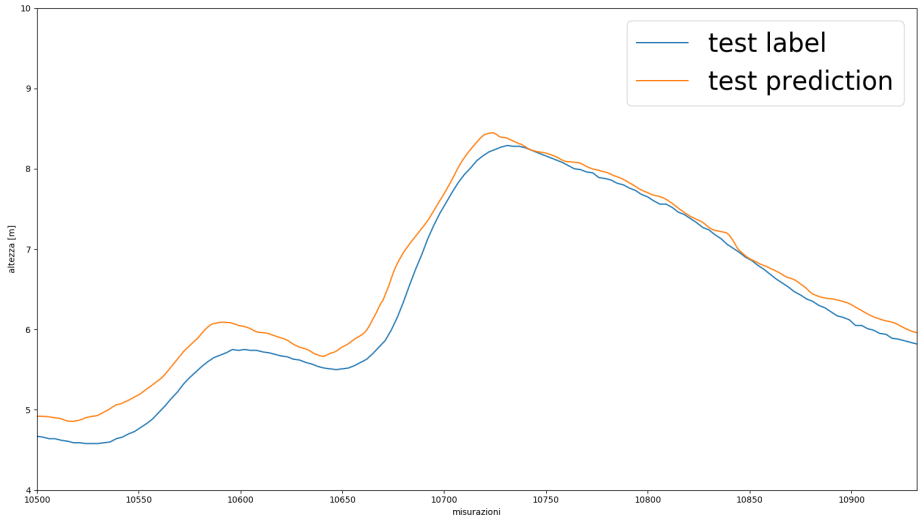

Figura 4.27: Previsione anno 2011, con interpolazione e soglia, dal 14 al 17 marzo

L'anno 2014 (Figura 4.28) come i precedenti non sembra presentare imprecisioni rilevanti nei periodi di piena se non durante la giornata del 19 gennaio dove l'errore è sempre elevato e numericamente uguale a 50 cm.
Come abbiamo già visto l'errore è dovuto alla rottura dell'argine e alla conseguente inondazione del Secchia.
Per assurdo la differenza che si registra tra la stima e i dati reali dovrebbe essere una buona approssimazione della quantità di acqua mancante, ovvero quella che ha allagato le campagne circostanti.

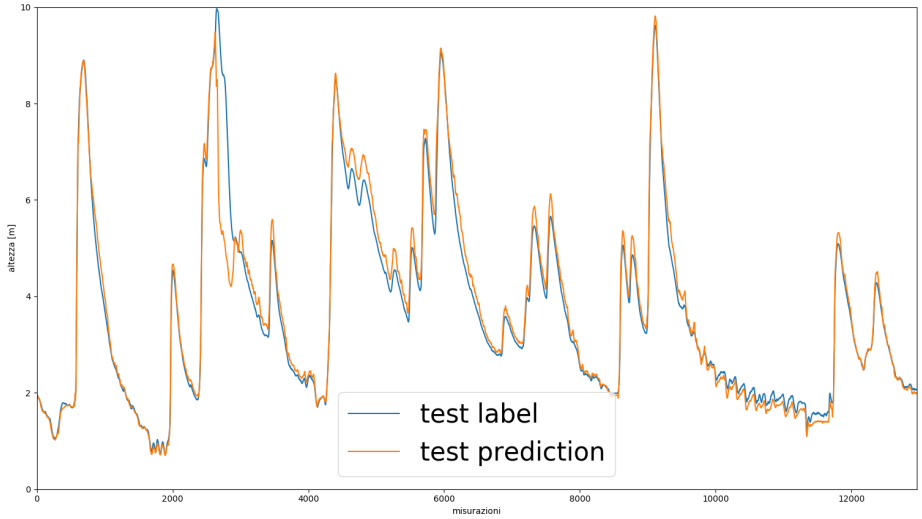

Figura 4.28: Previsione anno 2014, con interpolazione e soglia, dal 1 gennaio al 31 marzo

Capitolo 5

Conclusioni

Questo studio ha cercato di risolvere il reverse flow problem utilizzando esclusivamente le reti neurali.
Per ottenere la massima precisione, soprattutto nei periodi di piena, è risultato necessario introdurre l'interpolazione lineare e una soglia sulle misurazioni rilevate.
I risultati dei test sono molto buoni e in genere offrono uno scenario utilizzabile per ulteriori approfondimenti.
Tuttavia, è importante tenere presente che questa tesi si è concentrata esclusivamente sulla previsione di un solo valore.
L'implementazione di una rete con un numero maggiore di valori in output può migliorare ulteriormente la precisione generale.
Come possibili sviluppi futuri si potrebbe realizzare una rete che utilizza come parametri in ingresso le misurazioni di più stazioni, o la possibilità di gestire una breccia, come quella del 19 gennaio, che attualmente non viene considerata.
Si potrebbe anche combinare il modello analizzato con un simulatore, in grado di generare esempi fluviali realistici, e quindi di ottenere training set sempre più ricchi.

Bibliografia

[1] Simon Haykin. *Neural Networks: A comprehensive Foundation*. Prentice Hall PTR, 2nd edition, 1998.

[2] K. M. Mohiuddin A. K. Jain, Jianchang Mao. Artificial neural networks: a tutorial. *IEEE*, 1996.

[3] M. D'Oria and Tanda. Reverse flow routing in open channels: A bayesian geostatical approch. 2012.

[4] Mignosa P. D'Oria, M. and Tanda. Bayesian estimation of inflow hydrographs in ungauged sites of multiple reach systems, advances in water resources. 2014.

[5] R. Vacondio A. Dal Palù P. Mignosa A. Ferrari, D'Oria and Tanda. Discharge hydrograph estimation at upstream-ungauged sections by coupling a bayesian methodology and a 2-d gpu shallow water model. 2018.

Ringraziamenti

Desidero innanzitutto ringraziare il Professore Dal Palù per i preziosi insegnamenti durante questi 3 anni e per la disponibilità e precisione dimostratemi durante il periodo di stesura di questa tesi.
Inoltre, ringrazio sentitamente il Dott. Ing. Renato Vacondio e la Dott.ssa Ing. Alessia Ferrari che mi hanno fornito dati e supporto durante il tirocinio.
Un grande ringraziamento a mia madre e mio padre che hanno reso possibile tutto questo grazie al loro supporto, sia morale che economico.
Un ringraziamento generale, ma non per questo meno importante, ai miei fratelli, Andrea e Luca, ai miei zii, cugini, e alle mie nonne, che nel mio cuore hanno sempre un posto speciale.
Inoltre, vorrei esprimere la mia sincera gratitudine ai miei compagni di corso, e in particolare a Samuel, Luca, Davide, Matteo, e Pietro, che mi hanno supportato e sopportato in questi 3 anni.
Per ultimi, ma non meno importanti, i miei amici, che mi hanno sempre sostenuto durante tutto il percorso.
Una dedica speciale al Deca Imperiale, che ogni giorno ha condiviso con me gioie, sacrifici e successi, senza voltarmi mai le spalle.
L'aver raggiunto questo traguardo grazie alla vostra amicizia e al vostro sostegno lo rendono ancora più prezioso.