Topic 6: Deep Learning per l'analisi dell'impatto ambientale; Corso: Machine Learning per le applicazioni ambientali

Obiettivi di apprendimento

Al termine di questa unità, gli studenti saranno in grado di:

- Spiegare il concetto e il ruolo delle reti neurali nelle applicazioni ambientali.
- Illustrare i principi fondamentali del deep learning (DL) e la struttura e l'addestramento delle reti neurali.
- Descrivere l'impiego delle reti neurali convoluzionali e ricorrenti nell'analisi dell'impatto ambientale.
- Applicare un modello di deep learning di base all'analisi di dati ambientali (ad esempio inquinamento, copertura del suolo o andamento delle temperature).
- Interpretare i risultati dei modelli di deep learning e discuterne il contributo alle decisioni sostenibili.
- Sviluppare semplici prototipi o analisi di scenario che illustrino applicazioni del deep learning alla sostenibilità.
- Discutere come il DL possa aiutare i decisori nella valutazione dei rischi ecologici.

Introduzione

Il deep learning (DL) è un ramo del machine learning che utilizza reti neurali artificiali per apprendere schemi complessi nei dati. A differenza degli algoritmi tradizionali, i modelli di deep learning possono estrarre automaticamente caratteristiche rilevanti, risultando particolarmente utili per grandi dataset ambientali complessi. Questa lezione introduce i principi fondamentali del deep learning e mostra come tali metodi contribuiscano all'adattamento climatico, alla protezione ecologica e a decisioni sostenibili.

Definizione e differenze: Machine Learning e Deep Learning

L'intelligenza artificiale nacque negli anni Cinquanta, quando alcuni pionieri dell'informatica iniziarono a chiedersi se fosse possibile far «pensare» i computer: una questione le cui implicazioni esploriamo ancora oggi [1]. In sintesi, l'IA mira ad automatizzare attività intellettuali normalmente svolte dagli esseri umani. È quindi un campo generale che comprende il machine

learning e il deep learning, ma anche molti altri approcci che non prevedono necessariamente apprendimento. La Figura 6.1 rappresenta il rapporto tra intelligenza artificiale, machine learning e deep learning mediante tre ovali concentrici.

Il machine learning è un modo di programmare i computer affinché apprendano dai dati e migliorino le proprie prestazioni senza specificare una regola esatta per ogni situazione. Nel machine learning classico, le persone scelgono generalmente le caratteristiche di ingresso (feature) rilevanti, ad esempio temperatura media, precipitazioni o umidità del suolo, e addestrano un algoritmo a effettuare previsioni o prendere decisioni sulla loro base. Questo approccio funziona bene con dati relativamente limitati, strutturati in tabelle, e quando sono importanti semplicità e interpretabilità.

Il deep learning è un tipo particolare di machine learning che utilizza reti neurali artificiali con numerosi strati per apprendere direttamente da dati grezzi, come immagini, suoni o lunghe serie temporali. Anziché progettare manualmente le feature, una rete profonda apprende progressivamente rappresentazioni utili nei suoi strati: dai pixel alle forme e agli oggetti di un'immagine. Richiede generalmente più dati e risorse computazionali, ma è molto efficace in compiti complessi quali riconoscimento di immagini e del parlato e comprensione del linguaggio naturale.

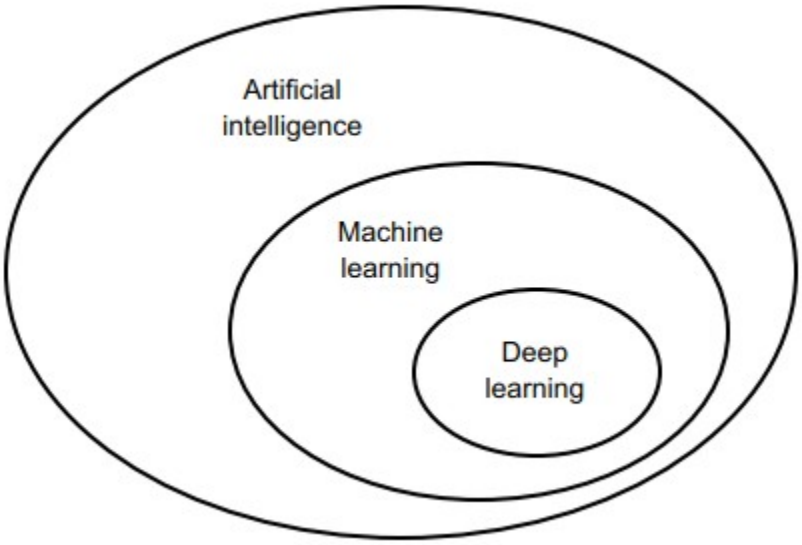

Figura 6.1 Livelli dell'intelligenza artificiale

La distinzione fondamentale è che il machine learning comprende una famiglia più ampia di metodi, in cui solitamente le feature sono definite da esseri umani e i modelli sono più piccoli e interpretabili; il deep learning è un sottoinsieme basato su reti neurali profonde che apprendono automaticamente feature e schemi da molti dati. In pratica, il machine learning è spesso usato per dati ambientali strutturati in fogli di calcolo o database, mentre il deep learning è preferito per immagini satellitari, griglie di sensori e segnali ad alta dimensionalità. La regola da ricordare è: tutto il deep learning è machine learning, ma non tutto il machine learning è deep learning.

L'esplosione dei dati ambientali: satelliti, droni, sensori IoT e modelli climatici

La scienza ambientale moderna va ben oltre le misurazioni manuali: disponiamo di una rete globale di «occhi» e «orecchie» digitali che monitorano la Terra. I satelliti acquisiscono quotidianamente immagini ad alta risoluzione, rilevando fenomeni come lo scioglimento dei ghiacci o l'espansione urbana, mentre i droni mappano foreste e coste in 3D. Milioni di sensori Internet of Things (IoT), collocati negli oceani, nel suolo e nelle città, trasmettono continuamente dati su temperatura, inquinamento e umidità. Ne deriva una grande quantità di informazioni, spesso non strutturate, impossibile da analizzare manualmente.

Oltre alle osservazioni del mondo reale, gli scienziati utilizzano complessi modelli climatici eseguiti su supercomputer per simulare condizioni meteorologiche anche a distanza di decenni.

Combinando simulazioni, immagini satellitari e flussi dei sensori si ottengono dataset con milioni di variabili, detti ad alta dimensionalità. I metodi tradizionali di machine learning possono incontrare difficoltà perché richiedono una semplificazione manuale preliminare dei dati (feature engineering). Con gli attuali volumi di dati, definire regole manuali per ogni immagine o misura è impraticabile.

Il deep learning diventa quindi essenziale. Diversamente da molti algoritmi più datati, che oltre un certo punto non migliorano più, i modelli profondi possono beneficiare di quantità crescenti di dati. Le reti neurali convoluzionali (CNN) individuano schemi nelle immagini satellitari, mentre le reti neurali ricorrenti (RNN) analizzano l'andamento temporale dei sensori climatici. Automatizzando l'analisi, il deep learning trasforma una massa di informazioni grezze in conoscenze utili alla tutela del pianeta.

Perché il deep learning è importante: dati complessi, non lineari e ad alta dimensionalità

Il deep learning è importante per la scienza ambientale perché le relazioni ecologiche sono spesso altamente non lineari: una piccola variazione della temperatura può avere effetti complessi o sproporzionati. Le funzioni di attivazione consentono alle reti profonde di approssimare relazioni curve e complesse. I dati ambientali sono inoltre spesso ad alta dimensionalità, come immagini satellitari con milioni di pixel o misure simultanee di migliaia di sensori. Rispetto ai metodi che soffrono maggiormente della «maledizione della dimensionalità», le architetture profonde sono progettate per elaborare grandi insiemi di dati non strutturati, distinguendo automaticamente schemi significativi dal rumore. Esempi di applicazioni per la sostenibilità includono:

- Mappatura della deforestazione. Le reti neurali convoluzionali (CNN) analizzano immagini satellitari di regioni come l'Amazzonia, rilevando automaticamente le aree disboscate o degradate e aiutando le autorità a intervenire rapidamente.
- Previsione delle alluvioni. Modelli ricorrenti o ibridi apprendono da precipitazioni, livelli dei fiumi e umidità del suolo precedenti per prevedere quando e dove potrebbero verificarsi esondazioni, aumentando il tempo disponibile per prepararsi.
- Classificazione delle barriere coralline. Modelli addestrati su migliaia di fotografie subacquee riconoscono le specie di corallo e rilevano sbiancamento o malattie, accelerando il monitoraggio della salute delle barriere.

- Previsione delle emissioni. I modelli profondi utilizzano dati relativi a consumi energetici, traffico e attività industriali per stimare emissioni di CO_2 giornaliere o future e monitorare i progressi verso gli obiettivi climatici.

Fondamenti delle reti neurali profonde

Questo modulo introduce i fondamenti teorici e pratici del deep learning (DL), un sottocampo dell'intelligenza artificiale che ha trasformato l'interpretazione di dataset complessi. Il DL sposta l'attenzione dalla modellazione statistica tradizionale all'estrazione automatica delle caratteristiche.

Mentre metodi statistici tradizionali, come la regressione lineare, dipendono dalla selezione manuale delle variabili e dal loro pretrattamento, le reti neurali profonde (DNN) apprendono autonomamente queste rappresentazioni. Attraverso più strati nascosti, una DNN scompone i dati grezzi in diversi livelli di astrazione.

- **Strati iniziali:** rilevano schemi semplici o caratteristiche di basso livello.
- **Strati più profondi:** combinano tali schemi per identificare strutture complesse di alto livello nei dati.

Questo apprendimento gerarchico è particolarmente utile nell'analisi ambientale. I sistemi naturali sono governati da interazioni spesso non lineari e complesse tra variabili come umidità, pressione atmosferica e particolato.

I modelli tradizionali possono avere difficoltà con queste interazioni, mentre le DNN sono efficaci in:

- **Riconoscimento di schemi:** identificazione di variazioni sottili nei dati climatici che precedono eventi meteorologici estremi.
- **Correlazioni multidimensionali:** descrizione di come fattori diversi, ad esempio umidità del suolo e velocità del vento, concorrano a influenzare gli ecosistemi locali.
- **Accuratezza predittiva:** previsioni più robuste grazie alla rappresentazione di componenti non lineari che modelli più semplici potrebbero trattare come valori anomali.

Con queste basi, i ricercatori possono superare la semplice osservazione dei dati e creare strumenti predittivi adatti alla scala e alla variabilità del pianeta.

Architettura delle reti neurali: il concetto di strato

Una rete neurale può essere interpretata come una funzione matematica che associa un insieme di valori di ingresso a valori di uscita.

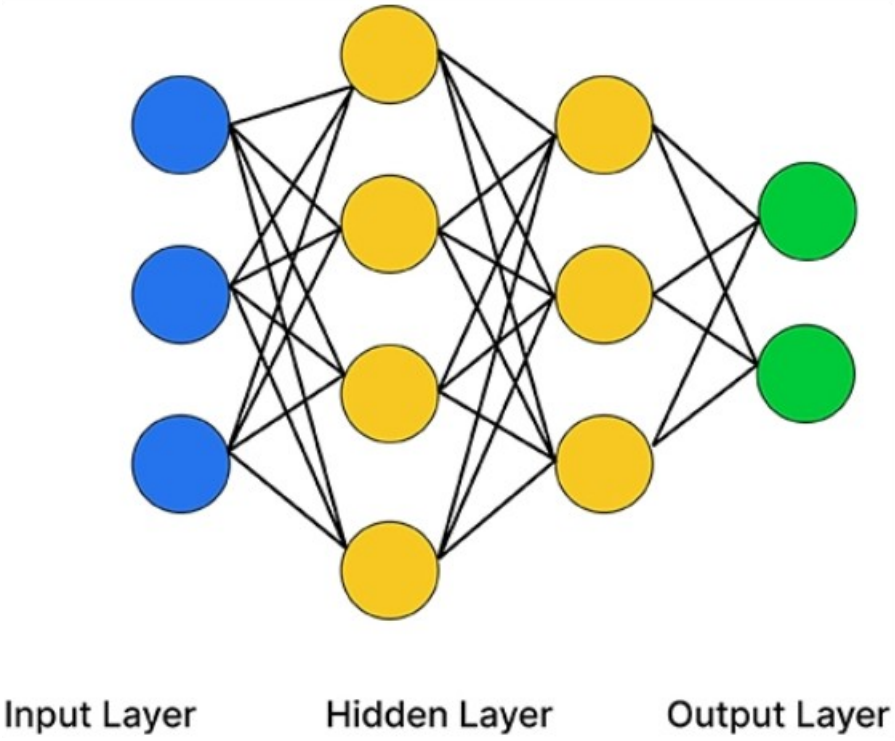

Figura 6.2 Architettura delle reti neurali

Una rete profonda feedforward [1], o percettrone multistrato (MLP), comprende tre tipi di strati (Figura 6.2):

- **Strato di ingresso:** riceve i dati ambientali grezzi senza eseguire calcoli. Il numero di neuroni corrisponde al numero di feature del dataset.
    - *Contesto ambientale: nell'analisi della qualità dell'aria, i neuroni di ingresso possono rappresentare specifiche letture dei sensori.* $\left[x_1 : \text{Temperature}, x_2 : \text{Wind Speed}, x_3 : \text{Humidity}\right]$
- **Strati nascosti:** si trovano tra ingresso e uscita e sono detti «nascosti» perché i loro valori non sono osservati nei dati di addestramento.

- **Heaton** spiega che, durante il passaggio dei dati attraverso gli strati, la rete calcola somme pesate degli ingressi. “Profondo” indica semplicemente una rete con più di uno strato nascosto**.**
- *Funzione:* gli strati nascosti estraggono feature. Quelli iniziali possono rilevare schemi semplici, come cicli giornalieri della temperatura; quelli più profondi li combinano per individuare fenomeni complessi, come le condizioni che precedono un’ondata di calore.

- **Strato di uscita:** lo strato finale produce la previsione (Figura 6.3).
  - *Regressione: un singolo neurone restituisce un valore continuo, ad esempio le precipitazioni previste in mm.*
  - *Classificazione*: più neuroni rappresentano le probabilità di classi diverse, ad esempio classe 0 “Aria sicura” e classe 1 “Aria pericolosa”.

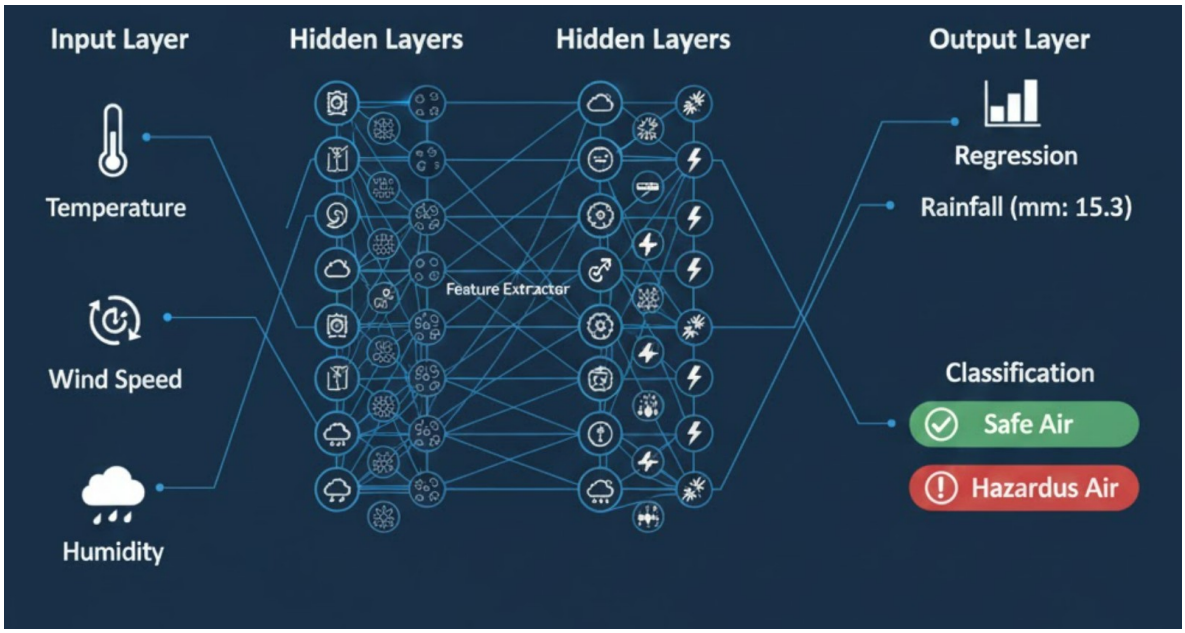

Figura 6.3 Strati delle reti neurali

Funzioni di attivazione: introdurre la non linearità

Se una rete neurale fosse composta soltanto da somme pesate e termini di bias, resterebbe un modello lineare, indipendentemente dal numero di strati: la composizione di funzioni lineari è infatti ancora lineare. Non riuscirebbe quindi a rappresentare le relazioni irregolari e curve dei dati reali.

Per descrivere fenomeni ambientali complessi, come la crescita non lineare dei batteri o le soglie dei temporali, bisogna introdurre una funzione matematica in ciascun neurone: la funzione di attivazione (Figura 6.4).

- **ReLU (Rectified Linear Unit):** funzione che lascia passare i segnali positivi e annulla quelli negativi. $f(x) = max(0, x)$
  - La ReLU è attualmente una delle scelte standard per gli strati nascosti nel deep learning.
  - Principio di funzionamento: trasmette inalterati i valori positivi e blocca quelli negativi.
  - Efficienza computazionale: rispetto a funzioni esponenziali complesse, richiede un semplice confronto con una soglia ed è quindi molto rapida durante l'addestramento.
  - Riduzione del problema del gradiente che svanisce: nelle reti molto profonde i gradienti possono diminuire progressivamente fino a impedire l'apprendimento. La ReLU [1] contribuisce a limitarlo mantenendo un gradiente costante per gli ingressi positivi.
  - Sparsità: restituendo zero per gli ingressi negativi, la ReLU produce rappresentazioni sparse, in cui solo una parte dei neuroni è attiva in un dato momento; ciò richiama l'efficienza del cervello umano.
- **Sigmoide**
  - Classificazione binaria: poiché restituisce valori nell'intervallo (0,1), è adatta allo strato di uscita di un modello binario. Esempio: probabilità di «Alluvione» (>0,5) oppure «Nessuna alluvione» (<0,5).
  - Contesto storico: un tempo usata negli strati nascosti, è stata in gran parte sostituita dalla ReLU perché tende a saturare (le estremità della curva diventano quasi piatte), rallentando l'apprendimento.
- **Softmax:**
  - Mentre la sigmoide si presta a distinguere tra due alternative, la Softmax è progettata per categorie mutuamente esclusive.

- Distribuzione di probabilità: la Softmax trasforma il vettore dei punteggi grezzi dell'ultimo strato in una distribuzione di probabilità le cui componenti sommano esattamente a 1 (100%).
- Applicazione ambientale: classificando la copertura del suolo da immagini satellitari, può assegnare a un pixel le probabilità 85% foresta, 10% area urbana e 5% acqua.
- Decisione: come previsione finale si seleziona di norma la classe con probabilità maggiore.

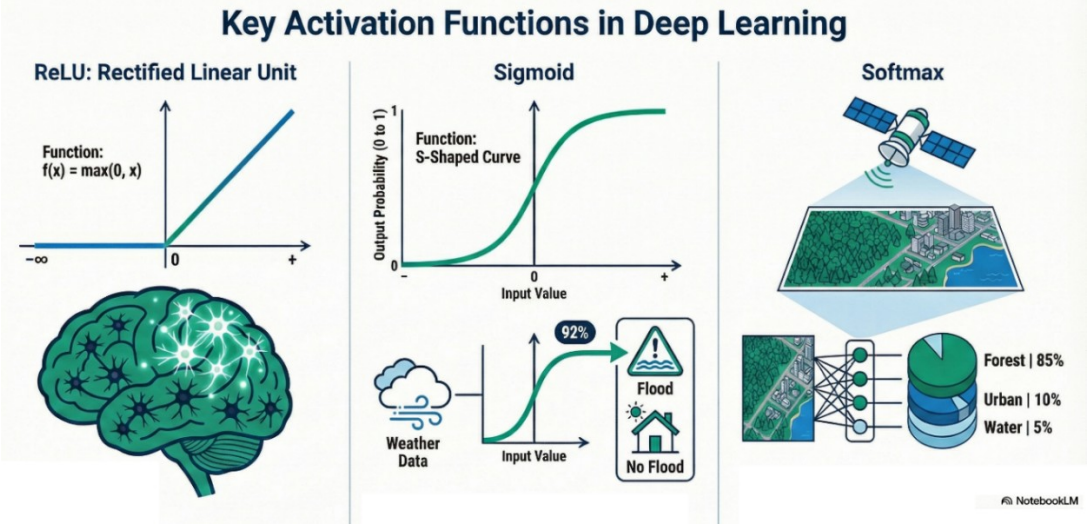

Figura 6.4 Principali funzioni di attivazione

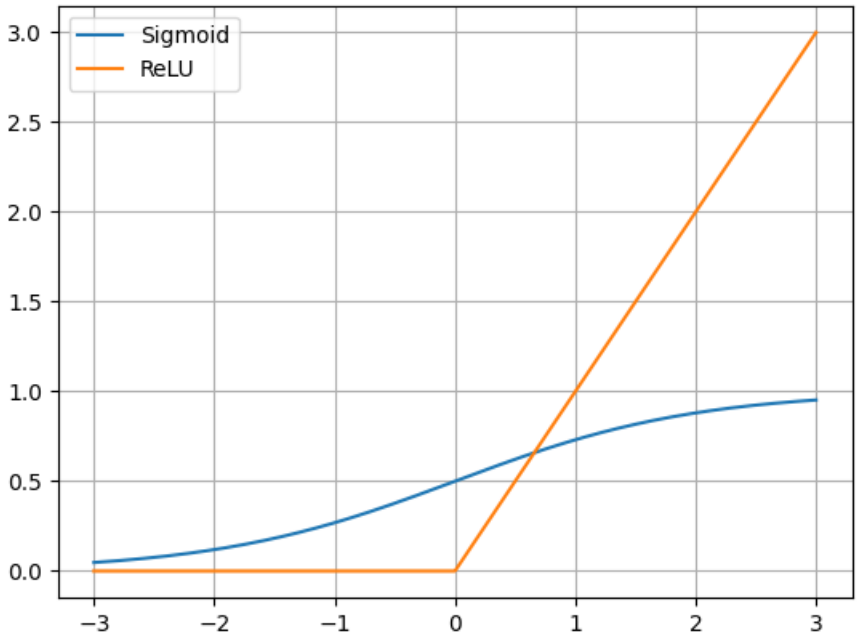

Figura 6.5 Esempio delle funzioni di attivazione sigmoide e ReLU

La Figura 6.5 mostra i grafici delle funzioni di attivazione sigmoide e ReLU.

Addestramento e ottimizzazione

L'apprendimento di una rete neurale è un processo iterativo di tentativi e correzioni. Per apprendere dai dati ambientali complessi, la rete segue tre fasi: propagazione in avanti (formulazione della previsione), calcolo della perdita (misurazione dell'errore) e retropropagazione (correzione dei pesi interni).

- **Funzione di perdita (funzione obiettivo):**

  Per migliorare, una rete deve definire matematicamente l'errore. La funzione di perdita, detta anche funzione obiettivo, restituisce uno scalare che misura lo scostamento tra la previsione della rete e i valori reali di riferimento.

  - **Errore quadratico medio (MSE):** utile nei compiti di regressione, ad esempio per prevedere la concentrazione atmosferica di CO_2 o la temperatura del giorno successivo. Elevando al quadrato le differenze, penalizza maggiormente gli errori grandi.
  - **Perdita di entropia incrociata (cross-entropy):** utilizzata normalmente per la classificazione, ad esempio per decidere se un pixel satellitare rappresenti una

foresta o un’area urbana. Misura quanto la distribuzione di probabilità prevista corrisponde all’etichetta effettiva.

- **Retropropagazione (panoramica concettuale):** dopo aver calcolato la perdita, la rete deve determinare quali pesi e bias hanno contribuito all’errore. Questo avviene mediante la backpropagation.

  L’informazione procede all’indietro, dallo strato di uscita verso quello di ingresso [2]. L’algoritmo calcola il contributo di ciascun peso all’errore finale mediante strumenti matematici.

  - Il gradiente è un vettore che indica la direzione di massimo aumento dell’errore.
  - Calcolando i gradienti, la rete determina come piccole variazioni di ciascun peso influenzerebbero la previsione finale.

- **Discesa del gradiente:** conoscendo i gradienti, la rete aggiorna i pesi mediante un ottimizzatore; un metodo fondamentale è la discesa del gradiente.

  Immagina di trovarti su una montagna avvolta nella nebbia (il paesaggio della funzione di perdita) e di dover raggiungere la valle più bassa, dove l’errore è minimo. Non puoi vederla, ma puoi percepire la pendenza del terreno.

  - **Gradiente:** indica la direzione di salita.
  - **Passo:** per ridurre l’errore, ti sposti nella direzione opposta al gradiente, cioè in discesa.
  - **Tasso di apprendimento (learning rate):** determina l’ampiezza del passo. Se è troppo grande puoi superare il minimo; se è troppo piccolo il processo sarà molto lento.

- **Ottimizzatori comuni**

  La discesa del gradiente è il punto di partenza; il deep learning moderno utilizza ottimizzatori più sofisticati:

  - **Discesa stocastica del gradiente (SGD):** anziché elaborare l’intero dataset in una volta, stima i gradienti usando piccoli gruppi (batch) di dati, rendendo il processo più rapido e dinamico.

- **Adam (Adaptive Moment Estimation):** ottimizzatore molto diffuso che adatta automaticamente il learning rate di ciascun peso, modificando la dimensione degli aggiornamenti durante l'ottimizzazione.

Overfitting e generalizzazione nei dati ambientali

Una sfida fondamentale nell'addestramento delle reti neurali è l'overfitting, che si verifica quando un modello apprende troppo bene i dati di addestramento, compresi rumore casuale e anomalie specifiche, anziché relazioni generali. Heaton sottolinea che un modello in overfitting mostra un errore basso sui dati di addestramento e alto su nuovi dati mai osservati: generalizza male.

Conseguenza ambientale: un modello addestrato sui dati meteorologici 2010–2020 potrebbe memorizzare particolari tempeste di quegli anni e non prevedere una siccità nel 2025, cercando schemi storici specifici anziché indicatori climatici generali.

Strategie di mitigazione:

- **Dropout:** tecnica che disattiva casualmente alcuni neuroni durante l'addestramento, inducendo la rete ad apprendere caratteristiche robuste senza dipendere da percorsi specifici.
- **Early stopping (arresto anticipato):** si monitora la prestazione su un insieme di validazione separato e si interrompe l'addestramento quando l'errore di validazione inizia ad aumentare, anche se quello di addestramento continua a diminuire.

Modellazione delle sequenze: dall'MLP all'LSTM

Diversamente dagli algoritmi tradizionali e dalle semplici reti feedforward, che trattano le osservazioni perlopiù indipendentemente, i dati ambientali hanno una natura sequenziale. Per elaborarli si possono impiegare reti Long Short-Term Memory (LSTM) [2].

Le LSTM sono reti neurali ricorrenti (RNN) specializzate nel conservare informazioni a lungo termine e affrontare il problema del gradiente che svanisce. Si può immaginare una sorta di nastro trasportatore parallelo alla sequenza, lungo il quale le informazioni passate possono essere

conservate e recuperate. Le LSTM utilizzano porte apprendibili (di ingresso, uscita e dimenticanza) che regolano il flusso informativo, decidendo cosa conservare e cosa scartare.

Adatte alle applicazioni ambientali, consentono analisi più approfondite di una semplice regressione. Per prevedere la temperatura odierna, un modello LSTM può considerare una sequenza dei precedenti 14 giorni, individuando tendenze di riscaldamento o raffreddamento.

Principali architetture di deep learning in ambito ambientale

Questa sezione illustra le principali architetture neurali impiegate per modellare i sistemi ambientali. Anziché approfondire dimostrazioni matematiche, si concentra sull'intuizione di base: scegliere l'architettura in funzione della struttura dei dati, che possono essere misure tabellari di sensori, immagini satellitari o serie temporali meteorologiche.

Reti feedforward (FFN): le basi

Una rete feedforward, detta spesso percettrone multistrato (MLP), è un modello fondamentale di deep learning. Le informazioni fluiscono in un'unica direzione, dai nodi di ingresso attraverso gli strati nascosti fino all'uscita, senza cicli.

Funzionamento: le FFN approssimano funzioni che associano vettori di ingresso a valori obiettivo. Sono efficaci nell'apprendere schemi globali nei dati tabellari.

Applicazione ambientale: le FFN sono adatte a previsioni statiche, quando l'ordine dei dati è irrilevante o le dipendenze temporali sono deboli.

Esempio: prevedere il livello attuale di inquinamento in un luogo usando velocità del vento, temperatura e densità del traffico misurate nello stesso momento.

Limite: le FFN non dispongono di una «memoria». Elaborano ogni ingresso indipendentemente e risultano meno efficaci quando conta la sequenza, come nelle ondate di calore prolungate o nelle tendenze stagionali.

Reti neurali convoluzionali: elaborazione dei dati spaziali

Mentre le FFN trattano le feature in modo indipendente, molti dati ambientali hanno una struttura a griglia, come immagini satellitari o mappe radar. Le CNN sono specializzate nell'elaborazione di questi dati.

Funzionamento: anziché collegare ciascun neurone a tutti gli altri, le CNN utilizzano piccoli filtri (kernel) che scorrono sull'immagine rilevando schemi locali quali bordi, texture o forme. Ciò

consente di riconoscere caratteristiche, come alberi o edifici, indipendentemente dalla loro posizione nell'immagine (invarianza alla traslazione).

Applicazione ambientale: le CNN sono molto usate nel telerilevamento, ad esempio per mappare la deforestazione in Amazzonia mediante immagini satellitari o classificare lo stato di salute delle barriere coralline dalle fotografie subacquee.

Architetture ricorrenti (RNN, LSTM, GRU): analisi delle serie temporali

I dati ambientali sono spesso sequenziali: conoscere la temperatura di ieri è importante per prevedere quella di oggi. Le reti neurali ricorrenti (RNN) utilizzano una connessione ricorrente che permette di mantenere informazioni nel tempo.

RNN standard: mantengono uno «stato» contenente informazioni su ciò che è stato osservato. Soffrono tuttavia del problema del gradiente che svanisce e possono faticare a ricordare dipendenze lunghe, ad esempio tra una precipitazione di due settimane prima e il livello attuale di un fiume.

Long Short-Term Memory (LSTM): introdotte per affrontare il problema della memoria, presentano una struttura interna specializzata con porte di ingresso, dimenticanza e uscita, che consentono di apprendere quali informazioni ricordare o dimenticare nel lungo periodo.

Gated Recurrent Unit (GRU): variante semplificata delle LSTM, spesso più rapida da addestrare ma con prestazioni simili per molte serie temporali ambientali. Queste architetture sono importanti per la previsione.

Esempio: prevedere il rischio di alluvione dai dati di precipitazione e umidità del suolo degli ultimi 14 giorni, oppure descrivere tendenze climatiche in cui il contesto storico influenza gli stati futuri.

IA responsabile: overfitting ed etica

L'impiego del deep learning nel mondo reale richiede più di un'elevata accuratezza: servono affidabilità e responsabilità etica.

Evitare l'overfitting: un errore comune è creare modelli che memorizzano i dati di addestramento, ad esempio anomalie meteorologiche del 2010, ma non generalizzano al futuro. Tecniche come dropout (disattivazione casuale di neuroni) ed early stopping (arresto quando peggiorano le prestazioni di validazione) migliorano la robustezza.

IA etica e interpretabile: i modelli di deep learning sono spesso definiti «scatole nere». Nelle politiche ambientali è importante garantire trasparenza. La tecnologia non è neutrale: le scelte tecniche hanno anche implicazioni etiche. Quando un modello influenza l'allocazione dell'acqua o la risposta ai disastri, bisogna verificare che non perpetui distorsioni nei dati storici e che le sue previsioni siano interpretabili dagli esperti del dominio.

Dal deep learning alle decisioni di adattamento climatico

Una volta addestrato e validato il modello, l'attenzione si sposta dall'ottimizzazione tecnica all'impatto operativo. Ottenere errori bassi, misurati per esempio con MAE o RMSE, è un traguardo tecnico, ma l'obiettivo finale è trasformare le previsioni numeriche in interventi concreti di adattamento climatico. La modellazione predittiva contribuisce a tre pilastri della sostenibilità:

- **Pianificazione proattiva:** utilizzare previsioni a lungo termine per progettare infrastrutture resilienti, ad esempio dimensionando i sistemi di drenaggio in base all'intensità prevista delle precipitazioni.
- **Riduzione del rischio:** attivare sistemi di allerta precoce nei quali i risultati del modello fanno scattare protocolli di sicurezza.
- **Allocazione delle risorse:** ottimizzare la distribuzione di acqua o energia sulla base delle condizioni ambientali previste.

Modellazione predittiva per l'adattamento climatico

Il deep learning può integrare o sostituire simulazioni fisiche lente con modelli rapidi basati sui dati, favorendo l'adattamento in tempo reale in diversi ambiti:

- Previsione di eventi meteorologici estremi (ondate di calore, tempeste, siccità).
- Previsione di cambiamenti idrologici (alluvioni, variazioni della portata fluviale, scarsità d'acqua).
- Modellazione della probabilità e della propagazione degli incendi boschivi.
- Previsione del deterioramento della qualità dell'aria e degli effetti sulla salute umana.

Questi modelli sostengono pianificazione e infrastrutture resilienti, sistemi di allerta precoce e preparazione delle risposte alle emergenze. Le previsioni vengono collegate a soglie operative e misure specifiche di resilienza. Seguono esempi strutturati come Rilevamento → Azione, che illustrano meccanismi dei modelli, applicazioni reali e risultati di adattamento tratti da studi consolidati.

Esempi di scenari applicativi per l'adattamento climatico:

- **Applicazioni per la previsione delle alluvioni**
  - **(Rilevamento)** Modelli CNN-LSTM elaborano immagini satellitari e dati mareografici per prevedere la probabilità di inondazioni costiere nelle 48 ore successive, con un'accuratezza superiore del 20% rispetto a modelli tradizionali in scenari di innalzamento del livello del mare. → **(Azione)** I comuni attivano difese costiere adattive, come il ripristino delle mangrovie, e rialzano infrastrutture critiche; le allerte anticipate riducono del 30–50% le perdite economiche grazie a sacchi di sabbia preventivi e deviazioni del traffico [5].
  - **(Rilevamento)** Reti LSTM spazio-temporali integrano precipitazioni radar e livelli fluviali per mappare in tempo reale la profondità delle alluvioni urbane con risoluzione oraria. → **(Azione)** I servizi di emergenza inviano avvisi tramite app, ottimizzano le pompe di drenaggio e aggiornano le norme urbanistiche per edifici resistenti alle alluvioni [6].
- **Gestione del rischio di incendi boschivi**
  - **(Rilevamento)** Modelli ensemble CatBoost-RNN analizzano indici della vegetazione, vento e temperatura da dati MODIS per prevedere quotidianamente la probabilità di innesco degli incendi in regioni di 10.000 km². → **(Azione)** Gli enti forestali programmano incendi controllati nei periodi a basso rischio, inviano ordini di evacuazione graduati via SMS e predispongono mezzi aerei antincendio, riducendo del 25% i costi di spegnimento [7].
  - **(Rilevamento)** Architetture ibride CNN-BiLSTM simulano la velocità di propagazione del fuoco usando dati VIIRS sugli incendi attivi e dati topografici, prevedendo quasi in tempo reale il perimetro dell'incendio. → **(Azione)** Le comunità installano sensori per rilevare precocemente il fumo, ampliano le fasce tagliafuoco con vegetazione autoctona e formano squadre volontarie di risposta rapida [8].
- **Strategie per siccità e ondate di calore**
  - **(Rilevamento)** Modelli profondi ibridi QRNN con wavelet prevedono gli indici di siccità della portata fluviale SDI-3, SDI-6 e SDI-12 per Stoccolma, integrando decomposizioni delle serie temporali per una maggiore accuratezza a diverse scale. → **(Azione)** Le autorità idriche introducono restrizioni con 3–12 mesi di

anticipo, rivedono i contratti di approvvigionamento e adeguano il funzionamento degli impianti idroelettrici per attenuare i deficit previsti [9].

- **(Rilevamento)** Modelli di deep learning e metodi ensemble su simulazioni climatiche europee individuano episodi estremi di caldo e siccità che intensificano le ondate di calore nell'Europa meridionale e occidentale. → **(Azione)** Città come Lisbona e Atene attivano piani di emergenza per il caldo, con centri di raffrescamento, limitazioni al lavoro all'aperto e SMS mirati agli anziani [10].

Caso di studio: previsione delle ondate di calore mediante reti neurali profonde

Fonte dei dati: Open-Meteo

Per questo caso di studio si utilizza Open-Meteo, che offre accesso gratuito tramite API pubblica ai dati meteorologici storici di qualsiasi località, consentendo di selezionare intervallo temporale e variabili.

Per addestrare un modello di deep learning a prevedere le ondate di calore occorrono variabili legate allo stress termico. I dati si scaricano mediante un'interfaccia di rete, specificando posizione geografica (latitudine e longitudine), risoluzione temporale (giornaliera) e variabili meteorologiche.

L'esercitazione si concentra sulla variabile temperature_2m_max, ossia la temperatura massima a 2 metri: è la variabile obiettivo che si desidera prevedere per individuare eventi di caldo estremo.

I dati vengono importati direttamente in Python con Pandas, che interpreta la risposta CSV dell'API come un dataset strutturato adatto al machine learning.

Esempio di deep learning

In questa attività pratica passiamo dalla statistica di base al deep learning moderno. Costruiremo un sistema che non si limiti a segnalare che oggi fa caldo, ma avverta dell'avvicinarsi di ondate di calore pericolose analizzando l'evoluzione climatica nel tempo.

A tal fine usiamo una rete LSTM (Long Short-Term Memory), in grado di riconoscere relazioni temporali nei dati.

Guida passo per passo: eseguire la simulazione

Utilizzeremo Google Colab, una piattaforma cloud gratuita che permette di eseguire il modello direttamente nel browser senza installare software.

1. **Aprire Google Colab:**
   - Accedere a colab.research.google.com tramite il browser.
   - Verrà visualizzata la finestra «Welcome to Colaboratory».
2. **Caricare il notebook:**
   - Fare clic sulla scheda «Upload» nella finestra.
   - Trascinare il file «ML_DL.ipynb» nella finestra oppure scegliere «Choose File» per selezionarlo dal computer.
3. **Eseguire il codice:**
   - Dopo il caricamento, individuare la barra del menu superiore.
   - Selezionare «Runtime» e «Run all», oppure premere Ctrl+F9.
   - *Nota: può comparire l'avviso «Notebook requires high RAM» o «Author not trusted»; selezionare «Run anyway» per procedere.*
4. **Interagire:**
   - Scorrere il notebook durante l'esecuzione. Appariranno tabelle dei dati, barre di avanzamento dell'addestramento e, infine, grafici interattivi.

Requisiti tecnici

L'implementazione utilizza Python e le seguenti librerie specializzate:

- **TensorFlow/Keras:** per costruire e addestrare la rete neurale LSTM.
- **Scikit-learn:** per normalizzare i dati e calcolare metriche di prestazione (MAE, RMSE).
- **Pandas/NumPy:** per elaborare serie temporali ed eseguire calcoli numerici.
- **Folium/Plotly/Seaborn:** per creare grafici previsionali interattivi e mappe geospaziali di supporto alle decisioni.

Fonte dei dati

Open-Meteo funge da «macchina del tempo», fornendo dati meteorologici storici per qualsiasi località. La variabile selezionata per addestrare il modello è «Maximum Temperature (2 m)», cioè la temperatura massima a 2 m da prevedere.

Struttura del notebook: comprendere il codice

Il notebook «ML_DL.ipynb» contiene un flusso di lavoro completo e pronto all'uso, già configurato per scaricare i dati, addestrare la rete neurale e generare automaticamente report interattivi. Non è necessario scrivere altro codice per visualizzare i risultati.

Per adattare la simulazione a città o scenari diversi, gli studenti devono modificare soltanto il **Blocco 1 (Configurazione e librerie).** I Blocchi da 2 a 8 contengono la logica automatizzata di elaborazione, addestramento LSTM e visualizzazione e vanno eseguiti in sequenza **senza modificarli.**

L'attività pratica è suddivisa in otto blocchi logici:

1. Configurazione e librerie. Si imposta il «pannello di controllo» dell'esperimento. Prima di eseguire l'IA, gli studenti possono modificare le seguenti variabili per personalizzare la simulazione per città o scenari diversi:

- **Posizione geografica:** LATITUDE e LONGITUDE. Coordinate della città di riferimento (predefinita: Kyiv, Ucraina, 50.45, 30.52). Cambiarle consente di provare climi differenti, ad esempio la piovosa Sebenico (Croazia), Roma (Italia) o Skopje (Macedonia del Nord).
- **Periodo storico:** START_DATE e END_DATE. Intervallo impiegato per addestrare l'IA, ad esempio dal 2010-01-01 al 2025-12-31. Un maggior numero di anni aiuta generalmente il modello a riconoscere meglio gli schemi.
- **«Memoria» dell'IA:** LOOKBACK_DAYS. Dimensione della finestra mobile (predefinita: 14 giorni), che determina quanti giorni passati sono osservati per formulare una previsione.
- **Controlli di visualizzazione:** DISPLAY_START_DATE e DISPLAY_END_DATE. Regolano l'intervallo mostrato dal grafico interattivo principale (Blocco 6); anziché rappresentare tutti i 15 anni, permettono di concentrarsi su un singolo anno, ad esempio il 2025.
- **Soglie di sicurezza (sistema decisionale):**
  - HEATWAVE_WARNING: soglia di temperatura (ad esempio 27,0 °C) che attiva un avviso «arancione» per la popolazione.
  - CRITICAL_EMERGENCY: temperatura estrema (ad esempio 31,0 °C) che attiva un'allerta critica «rossa» per i servizi di emergenza.
- **Periodo di analisi della previsione:** DISPLAY_FORECAST_START_DATE e DISPLAY_FORECAST_END_DATE. Intervallo specifico (ad esempio agosto 2025)

usato nel Blocco 7 per verificare se il sistema di allerta abbia individuato correttamente le ondate di calore.

2. Caricamento dei dati. Il codice si collega automaticamente all'API Open-Meteo e scarica i dati meteorologici degli ultimi 15 anni. È possibile osservare la conversione dei dati «grezzi» in una tabella strutturata di Pandas (Figura 6.6).

```
Fetching data from Open-Meteo...
Data loaded: 5844 records.
```

	Date	MaxTemp	SolarRad
0	2010-01-01	-6.2	1.83
1	2010-01-02	-6.2	2.66
2	2010-01-03	-11.4	3.29
3	2010-01-04	-12.4	3.87
4	2010-01-05	-8.7	3.60

Figura 6.6 Dati ottenuti da Open-Meteo

3. Preelaborazione: la «finestra mobile». Questa è una differenza fondamentale rispetto agli algoritmi classici. Invece di imparare semplicemente che «ieri faceva caldo, quindi oggi farà caldo», si crea una finestra mobile (lookback) di 14 giorni: il modello osserva la sequenza dal giorno 1 al giorno 14 per prevedere il giorno 15. Ciò permette alla rete di cogliere l'inerzia climatica, ossia l'accumulo progressivo di calore prima di un picco.

4. Costruzione e addestramento del modello LSTM (Figura 6.7). Si costruisce il «cervello» del sistema:

- Si utilizza uno strato LSTM che agisce come memoria del sistema, conservando le tendenze importanti e scartando il rumore casuale.
- Si aggiunge il dropout per prevenire l'overfitting, cioè la situazione in cui il modello memorizza gli anni passati ma non apprende schemi meteorologici generali.

```
117/117 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.0029 - mae: 0.0414 - val_loss: 0.0025 - val_mae: 0.0388
Epoch 38/50
117/117 ━━━━━━━━━━━━━━━━━━━━ 1s 8ms/step - loss: 0.0030 - mae: 0.0423 - val_loss: 0.0025 - val_mae: 0.0388
Epoch 39/50
117/117 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.0029 - mae: 0.0413 - val_loss: 0.0025 - val_mae: 0.0389
Epoch 40/50
117/117 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.0030 - mae: 0.0423 - val_loss: 0.0025 - val_mae: 0.0389
Epoch 40: early stopping
Restoring model weights from the end of the best epoch: 35.
Training complete.
```

Figura 6.7 Addestramento LSTM con early stopping

5. Visualizzazione delle prestazioni di addestramento. Si traccia la «curva di apprendimento».

Obiettivo: entrambe le curve dovrebbero diminuire nel tempo, indicando la riduzione dell'errore (loss) del modello.

- **Linea blu (perdita di addestramento):** indica quanto bene il modello apprende dai dati storici.
- **Linea rossa (perdita di validazione):** indica quanto bene il modello si comporta su dati nuovi, non visti durante l'addestramento.

Che cosa osservare nella Figura 6.8:

- **Risultato positivo:** entrambe le curve diminuiscono e rimangono vicine.
- **Overfitting:** se la curva blu diminuisce ma quella rossa inizia ad aumentare, il modello sta memorizzando i dati anziché apprendere schemi generalizzabili.

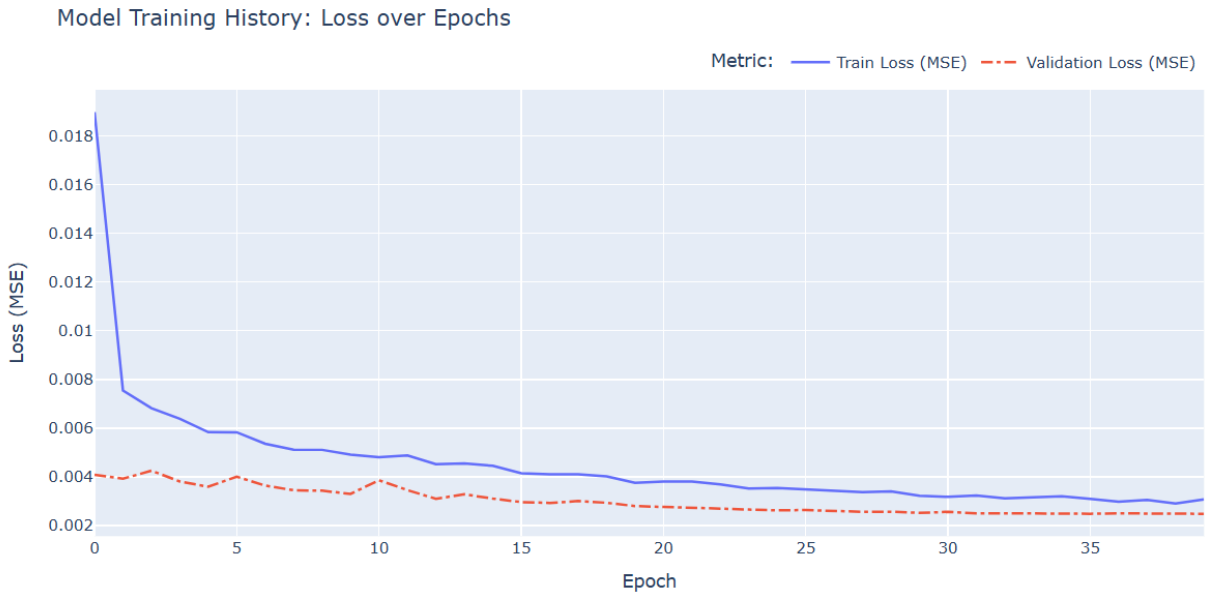

Figura 6.8 Evoluzione dell'addestramento: perdita in funzione delle epoche

6. Valutazione e previsione interattiva. Si utilizza Plotly per creare un grafico interattivo**.**

6a. Scheda di valutazione del modello (metriche):

Prima di osservare i grafici, calcoliamo gli indicatori quantitativi del modello (Figura 6.9).

- **MAE (errore assoluto medio):** lo scostamento medio in gradi delle previsioni, ad esempio di 1,5 °C.
- **RMSE (radice dell'errore quadratico medio):** metrica che penalizza più fortemente gli errori grandi rispetto a quelli piccoli.
- **R² (coefficiente di determinazione):** punteggio che esprime quanta parte della variabilità della temperatura viene spiegata dal modello.

```
--- 📊 WEATHER MODEL REPORT CARD ---

1. Average Error (MAE): 2.14°C
   👉 Explanation: On average, the model's prediction is off by about 2.1°C.
--------------------------------------------------
2. Penalty for Big Mistakes (RMSE): 2.74°C
   👉 Explanation: This metric punishes large errors more heavily.
      If this number is much higher than the Average Error, the model occasionally makes big misses.
--------------------------------------------------
3. Accuracy Score (R²): 0.93 (or 93.1%)
   👉 Explanation: How much of the temperature variation does the model explain?
      (1.0 = Perfect, 0.0 = Useless, Negative = Worse than guessing)
   ✅ Quality: High Confidence.
--------------------------------------------------
```

Figura 6.9 Scheda dei risultati del modello meteorologico

6b. Previsioni (grafico interattivo)

Questa sezione genera un grafico dinamico della serie temporale tramite Plotly (Figura 6.10).

- **Linea rossa:** temperatura prevista dall'IA.
- **Linea blu tratteggiata:** temperatura realmente registrata.
- Cursore interattivo: nella parte inferiore, permette di ingrandire settimane o mesi specifici.

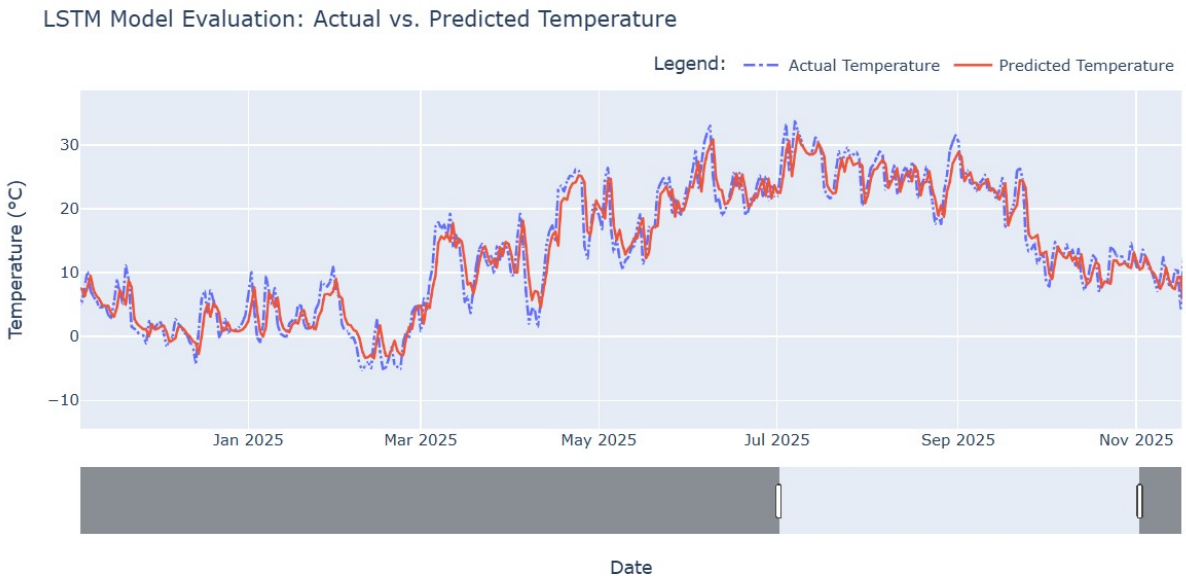

Figura 6.10 Valutazione del modello LSTM: temperatura effettiva e prevista

6c. Valori effettivi e previsti (qualità dell'adattamento)

Si genera un diagramma di dispersione per verificare eventuali distorsioni sistematiche.

- **Asse X:** temperatura effettiva.
- **Asse Y:** temperatura prevista.
- **«Linea della perfezione»:** diagonale rossa tratteggiata; se il modello fosse perfetto, tutti i punti blu cadrebbero su questa linea.

6d. Grafico dei residui

Si genera un grafico dei residui, o errori, nel tempo.

- **Linea dello zero:** rappresenta una previsione perfetta (errore pari a zero).
- **Scopo:** verificare la casualità degli errori. Se i punti appaiono come rumore casuale attorno allo zero, il modello ha catturato gran parte della struttura; schemi sistematici segnalano invece aspetti non modellati.

7. Sistema decisionale per l'adattamento climatico

In questo blocco simuliamo un sistema di allerta precoce. L'intelligenza artificiale non si limita a prevedere un numero: collega la temperatura prevista a decisioni e interventi concreti per la comunità.

Il sistema funziona secondo il principio del semaforo, confrontando la previsione dell'IA con soglie di sicurezza predefinite:

- 🟢 Livello «Normale» (verde)

- Condizione: temperatura prevista inferiore alla soglia di avviso (per esempio < 30 °C).
- Interventi per la comunità:
  - Monitoraggio ordinario delle condizioni meteorologiche.
  - Normale funzionamento dei servizi comunali.
  - *Spiegazione: la situazione è sicura e non richiede interventi.*

- 🟠 Livello «Avviso / Allerta» (arancione)
  - Condizione: la temperatura supera la prima soglia (per esempio ≥ 30 °C, variabile HEATWAVE_WARNING).
  - Interventi per la comunità:
    - Comunicazione: invio di SMS ai gruppi vulnerabili (anziani, persone con patologie croniche ecc.).
    - Assistenza sanitaria: preparazione dei servizi di ambulanza a un possibile aumento delle emergenze.
    - Servizi sociali: visite aggiuntive agli anziani isolati.
- 🔴 Livello «Emergenza critica» (rosso)
  - Condizione: la temperatura raggiunge valori estremi (per esempio ≥ 35 °C, variabile CRITICAL_EMERGENCY).
  - Interventi per la comunità:
    - Infrastrutture: apertura di centri comunali di raffrescamento, ossia spazi climatizzati presso biblioteche, scuole e altri edifici pubblici.
    - Sicurezza sul lavoro: limitazione o divieto dei lavori fisici pesanti all'aperto, come cantieri e manutenzione stradale, nelle ore più calde.
    - Approvvigionamento idrico: predisposizione di punti mobili per la distribuzione di acqua potabile nelle piazze.
    - Rete elettrica: preparazione a possibili interruzioni programmate dovute al sovraccarico causato dai condizionatori.

```
--- HEATWAVE EARLY WARNING SYSTEM (LSTM) ---
Analyzing forecast for the period: 2025-08-01 to 2025-08-31
Date: 2025-08-01 | Forecast=22.2°C | Actual=24.3°C | Error=-2.1°C
  [ACTION]: 🟢 Status Normal.
------------------------------------------------------------
Date: 2025-08-02 | Forecast=24.7°C | Actual=26.4°C | Error=-1.7°C
  [ACTION]: 🟢 Status Normal.
------------------------------------------------------------
Date: 2025-08-03 | Forecast=26.1°C | Actual=27.0°C | Error=-0.9°C
  [ACTION]: 🟢 Status Normal.
------------------------------------------------------------
Date: 2025-08-04 | Forecast=26.3°C | Actual=28.1°C | Error=-1.8°C
  [ACTION]: 🟢 Status Normal.
------------------------------------------------------------
Date: 2025-08-05 | Forecast=27.1°C | Actual=29.0°C | Error=-1.9°C
  [ACTION]: 🟠 ADVISORY. Issue elderly warnings.
------------------------------------------------------------
Date: 2025-08-06 | Forecast=27.7°C | Actual=28.4°C | Error=-0.7°C
  [ACTION]: 🟠 ADVISORY. Issue elderly warnings.
------------------------------------------------------------
Date: 2025-08-07 | Forecast=27.0°C | Actual=23.0°C | Error=4.0°C
  [ACTION]: 🟠 ADVISORY. Issue elderly warnings.
------------------------------------------------------------
```

Figura 6.11 Sistema di allerta precoce per le ondate di calore

8. Visualizzazione geospaziale

In quest'ultimo blocco trasformiamo la previsione temporale in uno strumento geospaziale di supporto alle decisioni. Sulla mappa vengono rappresentate le condizioni di rischio associate alla località scelta e alle soglie configurate.

Applicazione alla comunità: estendendo il modello a una rete di sensori distribuiti in centro, periferia e altre zone, sarebbe possibile identificare spazialmente quartieri più esposti e pianificare interventi mirati.

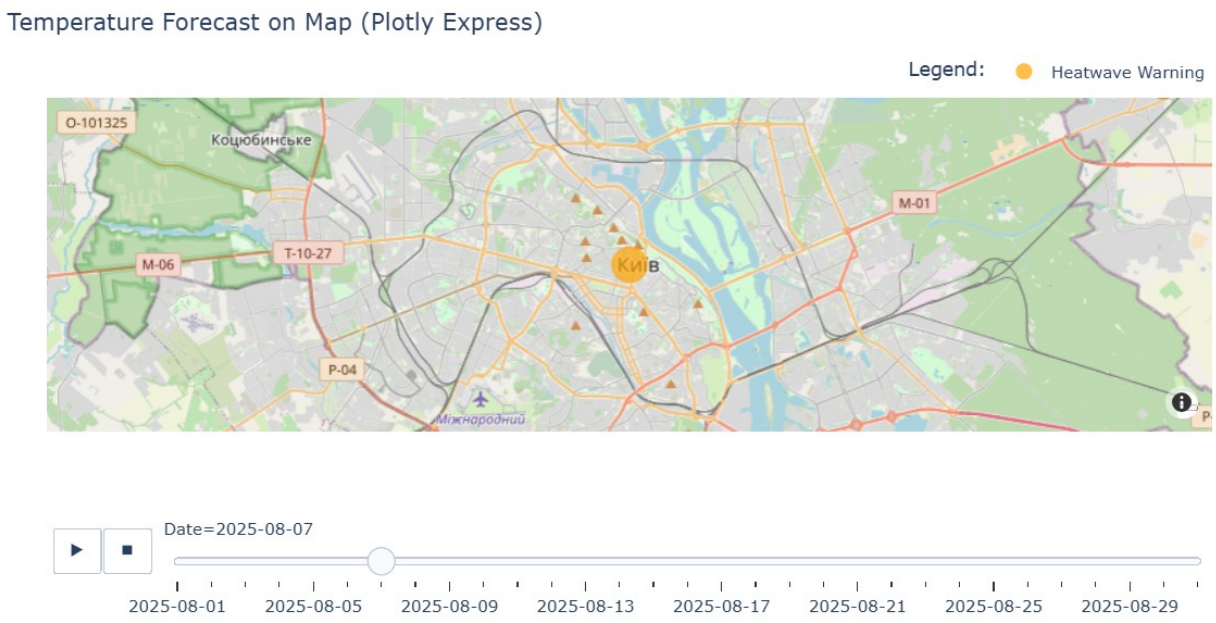

Figura 6.12 Visualizzazione cartografica basata sui parametri CRITICAL_EMERGENCY e HEATWAVE_WARNING

Esercitazioni

In questa esercitazione utilizzerai il deep learning per prevedere eventi meteorologici estremi in una città a tua scelta. Segui le attività per adattare il modello e interpretare le previsioni.

Attività 1: adattamento geografico

- **Apri** il notebook fornito in Google Colab.
- **Individua** le variabili LATITUDE e LONGITUDE nel Blocco 1.
- **Modifica** le coordinate scegliendo una città, ad esempio quella in cui vivi oppure una città calda come Roma.
- **Esegui** il notebook per generare nuove previsioni.
- **Analizza i risultati del sistema decisionale nel Blocco 7:**
  - Il sistema genera troppi messaggi «CRITICAL ALERT» (rossi)? Ciò suggerisce che le soglie potrebbero essere troppo basse.
  - Oppure non genera alcuna allerta nonostante il caldo? Ciò suggerisce che le soglie potrebbero essere troppo alte.

- **Visualizzazione cartografica (Blocco 8):** scorri fino alla mappa interattiva. Compaiono punti rossi o arancioni? Il risultato rispecchia il clima della città selezionata?
- **Taratura:** torna al Blocco 1 e modifica HEATWAVE_WARNING e CRITICAL_EMERGENCY in modo che le soglie siano appropriate per la città.
- **Esegui nuovamente il notebook.**

Domanda: in che modo il cambiamento delle soglie ha reso più utile il sistema di allerta precoce per questa città?

Attività 2: mettere alla prova la «memoria» (ottimizzazione degli iperparametri)

- **Individua la variabile LOOKBACK_DAYS = 14 nel Blocco 1:** indica l'intervallo di memoria dell'IA, ossia quanti giorni precedenti vengono considerati per prevedere il giorno successivo.
- **Modifica il valore da 14 a 3 giorni.**
- **Esegui** nuovamente l'intero notebook («Runtime» → «Run all») per riaddestrare il modello con questa «memoria breve».
- **Scorri** fino al grafico di valutazione nel Blocco 6b.

Domanda: confronta ora il comportamento della linea rossa (previsione) con quello ottenuto usando 14 giorni:

- **Effetto di ritardo:** la linea rossa si sposta verso destra? Il modello si accorge del caldo soltanto dopo che si è verificato?
- **Accuratezza dei picchi:** il modello non raggiunge i valori massimi della temperatura, sottostimando il caldo?
- **Ipotesi:** le ondate di calore non si sviluppano istantaneamente. Perché l'IA può aver bisogno di osservare gli ultimi 14 giorni per prevedere correttamente un evento estremo?

Attività 3 (⚠️ avanzata e facoltativa): inversione dello scenario (test dell'«anticiclone invernale»)

Contesto: i cambiamenti climatici comportano non soltanto ondate di calore, ma anche episodi di freddo estremo, ad esempio durante irruzioni di aria polare.

- **Fase 1 (cambiare periodo):** nel Blocco 1, modifica DISPLAY_FORECAST_START_DATE e DISPLAY_FORECAST_END_DATE selezionando un intervallo invernale.

- **Fase 2 (soglie):** nel Blocco 1, imposta CRITICAL_EMERGENCY su un valore di gelo, ad esempio -15.0 °C.
- **Fase 3 (inversione della logica):** passa al Blocco 7 (sistema decisionale).
    - Individua la riga di codice: if forecast_temp >= CRITICAL_EMERGENCY:
    - **Azione:** sostituisci l'operatore >= (maggiore o uguale) con <= (minore o uguale).
    - *Nota: ora il sistema attiva l'allerta soltanto quando la temperatura scende sotto la soglia di gelo.*
- **Esecuzione:** esegui il Blocco 7.

Domanda: il sistema segnala correttamente i giorni più freddi come «CRITICAL ALERTS»? Discuti perché adattare le soglie e la logica del sistema è fondamentale quando cambia il rischio ambientale.

Attività 4: discussione

Interpreta i risultati delle attività 1–3 in chiave ambientale. Sulla base della frequenza e distribuzione delle allerte, discuti l'utilità del modello per le decisioni locali e i suoi limiti.

Valutazione

1. Qual è lo scopo principale di un modello LSTM (Long Short-Term Memory) rispetto a una semplice media mobile per prevedere le ondate di calore?
    A. Smussare le fluttuazioni giornaliere e produrre un grafico annuale più regolare delle temperature.
    B. Individuare errori tecnici e rumore nei dati grezzi dei sensori.
    C. Cogliere schemi temporali complessi e l'«effetto di accumulo» del calore nel tempo.
    D. Eliminare la necessità di stazioni meteorologiche fisiche in località remote.

2. Qual è la conseguenza negativa più probabile di ridurre la «memoria» dell'IA (LOOKBACK_DAYS) da 14 a 3 giorni?
    A. Il modello diventa più sensibile al rumore e produce frequenti falsi allarmi.
    B. Il modello manifesta un «effetto di ritardo», riconoscendo l'ondata di calore soltanto quando la temperatura è già elevata.
    C. Il modello passa automaticamente dal deep learning alla regressione lineare.
    D. Il modello non può più essere eseguito perché il dataset diventa troppo piccolo.

3. Qual è il principale rischio dell'overfitting in un modello di deep learning per le previsioni ambientali?
   A. Produce previsioni troppo generiche e trascura le caratteristiche meteorologiche locali.
   B. Memorizza particolari eventi storici (rumore) e non riesce a prevedere accuratamente condizioni meteorologiche nuove e mai osservate.
   C. Richiede una potenza computazionale molto superiore a quella disponibile.
   D. Non è in grado di elaborare dati non lineari, come umidità o velocità del vento.

4. Dal punto di vista dell'amministrazione comunale, qual è il principale vantaggio strategico di questo sistema predittivo?
   A. Consente di mobilitare risorse di emergenza (come centri di raffrescamento e interventi sulla rete) diversi giorni prima che si verifichi la crisi.
   B. Garantisce un'accuratezza del 100%, eliminando del tutto falsi allarmi ed eventi non rilevati.
   C. Elimina la necessità di decisioni umane, consentendo una gestione interamente automatica della città.
   D. Riduce automaticamente la temperatura fisica della città ottimizzando i semafori.

5. Quale fenomeno spiega meglio perché un modello di IA potrebbe prevedere temperature molto più elevate nel centro di una città rispetto alle zone rurali circostanti?
   A. L'effetto «isola di calore urbana» (UHI), per cui le superfici artificiali trattengono e riemettono più calore rispetto a quelle naturali.
   B. Un'anomalia locale della pressione atmosferica che si forma frequentemente sopra gli edifici alti.
   C. Un errore di calibrazione dei sensori satellitari che sovrastima la riflessione delle superfici vetrate.
   D. Un «effetto di ritardo» dovuto al traffico intenso che rallenta il flusso d'aria.

Riferimenti bibliografici e approfondimenti

[1] F. Chollet, Deep Learning with Python, 2nd ed. Shelter Island, NY, USA: Manning Publications, 2021 [Online]. Available: https://sourestdeeds.github.io/pdf/Deep%20Learning%20with%20Python.pdf, Accessed: Dec. 15, 2025.

[2] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. arXiv:2106.11342, 2021. [Online]. Available: https://arxiv.org/abs/2106.11342, Accessed: Dec. 15, 2025.

[3] Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016 [Online]. Available: http://www.deeplearningbook.org, Accessed: Dec. 15, 2025

[4] J. Heaton, *Applications of Deep Neural Networks*. arXiv:2009.05673, 2020. [Online]. Available: https://arxiv.org/abs/2009.05673, Accessed: Dec. 15, 2025.

[5] Bilal Hassan, Areg Karapetyan, Aaron Chung Hin Chow, and Samer Madanat. Climate Adaptation-Aware Flood Prediction for Coastal Cities Using Deep Learning. arXiv:2510.26017 [cs.CV], 2025 [Online]. Available: https://arxiv.org/abs/2510.26017, Accessed: Dec. 15, 2025

[6] Qianqian Zhou, Shuai Teng, Zuxiang Situ, Xiaoting Liao, Junman Feng, Gongfa Chen, Jianliang Zhang, and Zonglei Lu. A deep-learning-technique-based data-driven model for accurate and rapid flood predictions in temporal and spatial dimensions. Hydrol. Earth Syst. Sci., 27, 1791–1808, 2023 [Online]. Available: https://hess.copernicus.org/articles/27/1791/2023/hess-27-1791-2023.pdf , Accessed: Dec. 15, 2025

[7] M. Nasourinia and K. Passi, “Wildfire Prediction in British Columbia Using Machine Learning and Deep Learning Models: A Data-Driven Framework,” *Big Data and Cognitive Computing*, vol. 9, no. 11, p. 290, 2025. [Online]. Available: https://doi.org/10.3390/bdcc9110290, Accessed: Dec. 15, 2025.

[8] Hyunglok Seo and Yeonjoo Kim. Enhancing Alaskan wildfire prediction and carbon flux estimation: a two-stage deep learning approach within a process-based model. *Environ. Res. Lett.*, 19, 124040, 2024 [Online]. Available: https://iopscience.iop.org/article/10.1088/1748-9326/ad8bdc, Accessed: Dec. 15, 2025

[9] Babak Mohammadi, Mohammed Abdallah, Rachid Oucheikh, Okan Mert Katipoğlu, and Majid Cheraghalizadeh. Enhancing streamflow drought prediction: integrating wavelet decomposition with deep learning and quantile regression neural network models. *Earth Sci. Inform.*, 18, 232, 2025 [Online]. Available: https://link.springer.com/article/10.1007/s12145-025-01736-w, Accessed: Dec. 15, 2025

[10] Elisa Felsche and Ralf Ludwig. Applying machine learning for drought prediction in a perfect model framework using data from a large ensemble of climate simulations. *Nat. Hazards Earth Syst. Sci.*, 21, 3679–3691, 2021 [Online]. Available: https://nhess.copernicus.org/articles/21/3679/2021/nhess-21-3679-2021.pdf, Accessed: Dec. 15, 2025