Intelligenza artificiale

Prof.ssa Lucia Migliorelli

lmigliorelli@unite.it

Corso di Sistemi multimediali e web per il turismo

Dipartimento di Scienze Politiche, Università di Teramo

Intelligenza artificiale

- L'intelligenza artificiale (IA) è una branca della scienza che si occupa di sviluppare sistemi in grado di risolvere problemi che normalmente richiederebbero l'intelligenza umana.

- Se un compito di solito richiede il cervello (riconoscere un volto, tradurre un testo, capire un'emozione, ecc.), ma riusciamo a farlo fare a una macchina, allora quello è un sistema di intelligenza artificiale.

- Questa definizione non dipende dalla tecnica usata (algoritmi, reti neurali, ecc.), ma solo dal fatto che la macchina affronti un compito tipico dell'intelligenza umana.

Intelligenza artificiale – cenni storici (1)

- Conferenza di Darmout, 1956.
- Il termine fu coniato da John McCarty e riassumeva nuove tendenze di ricerca nell'ambito della computer science ovvero: provare a «modellare» il comportamento/l'intelletto umano
- L'Intelligenza Artificiale viene dichiarata come nuova disciplina di ricerca e fu così definita: «*The science and engineering of making intelligent machines*» - John McCarthy (1956)

*John McCarthy (**Dartmouth College**), Marvin Minsky (**Harvard University**), Nathaniel Rochester (**IBM**), and Claude Shannon (**Bell Telephone Laboratories**)*

«In 1968, a computer will be the world chess champion» (Simon &Newell, 1958)

«In 20 years, Computers will be able to do any activity humans can do» (Simon, 1965)

Intelligenza artificiale – cenni storici (2)

Le previsioni sono state parzialmente azzeccate:

- *La prima previsione si è realizzata nel 1997 quando la potenza computazionale Deep Blue è riuscito a battere l'allora campione del mondo, Kasparov nel gioco degli scacchi.*

Intelligenza artificiale – cenni storici (3)

Le previsioni sono state parzialmente azzeccate:

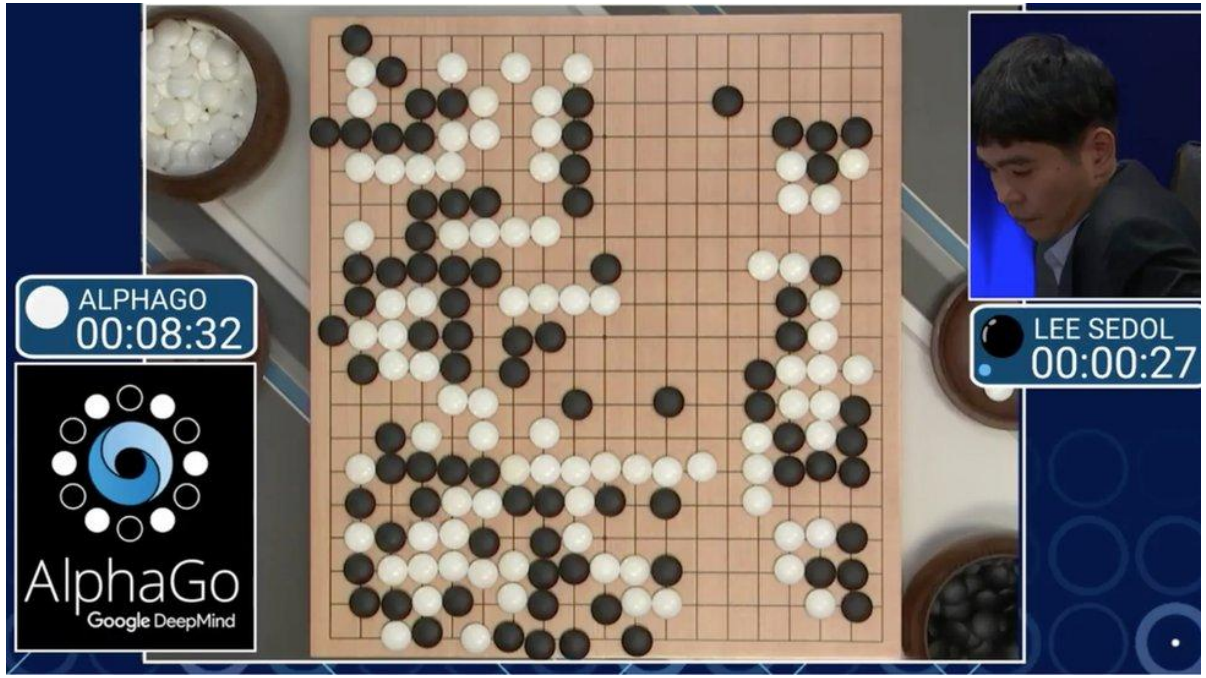
Comprensione del liguaggio Naturale (Siri e Alexa), 2011:

Watson (IBM) è al centro dei due concorrenti, è riuscito a batterli durante il quiz Jeopardy. La macchina era in grado di capire la domanda posta dando una risposta corretta.

Intelligenza artificiale – cenni storici (4)

Le previsioni sono state parzialmente azzeccate:

- Deep Mind è riuscito a battere il campione del mondo di AlphaGo, 2017.



Task vs algoritmo

Task = il compito da risolvere.

Esempio

- Riconoscere un oggetto in una foto (Object Recognition)
- Capire se un tweet è positivo o negativo (Sentiment Analysis)

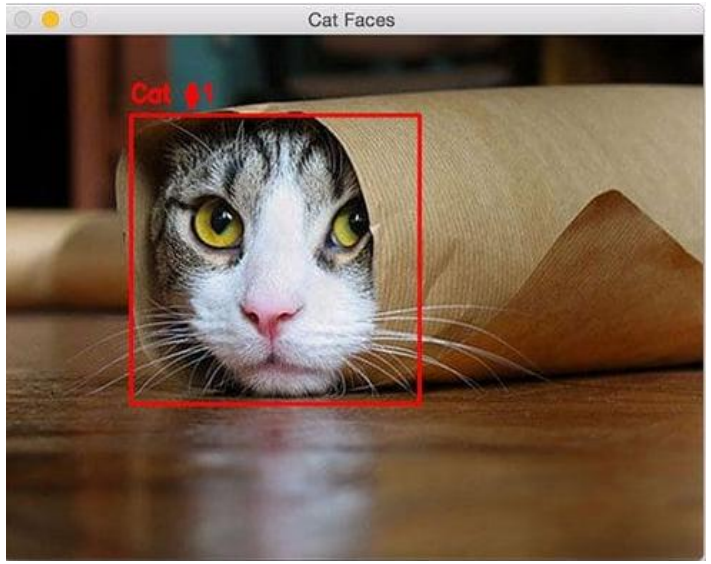
Algoritmo = la serie di passi che servono per risolvere il Task.

È come una ricetta di cucina: il Task è “fare una torta”, l’algoritmo sono i passaggi (mescolare, cuocere, decorare).

TASK = "Riconoscere un gatto nell'immagine"

ALGORITMO = Istruzioni passo-passo

- Leggi immagine
- Analizza pixel
- Decidi l'esemplare



Task vs algoritmo – sentiment analysis

La mia esperienza è stata fantastica

POSITIVE

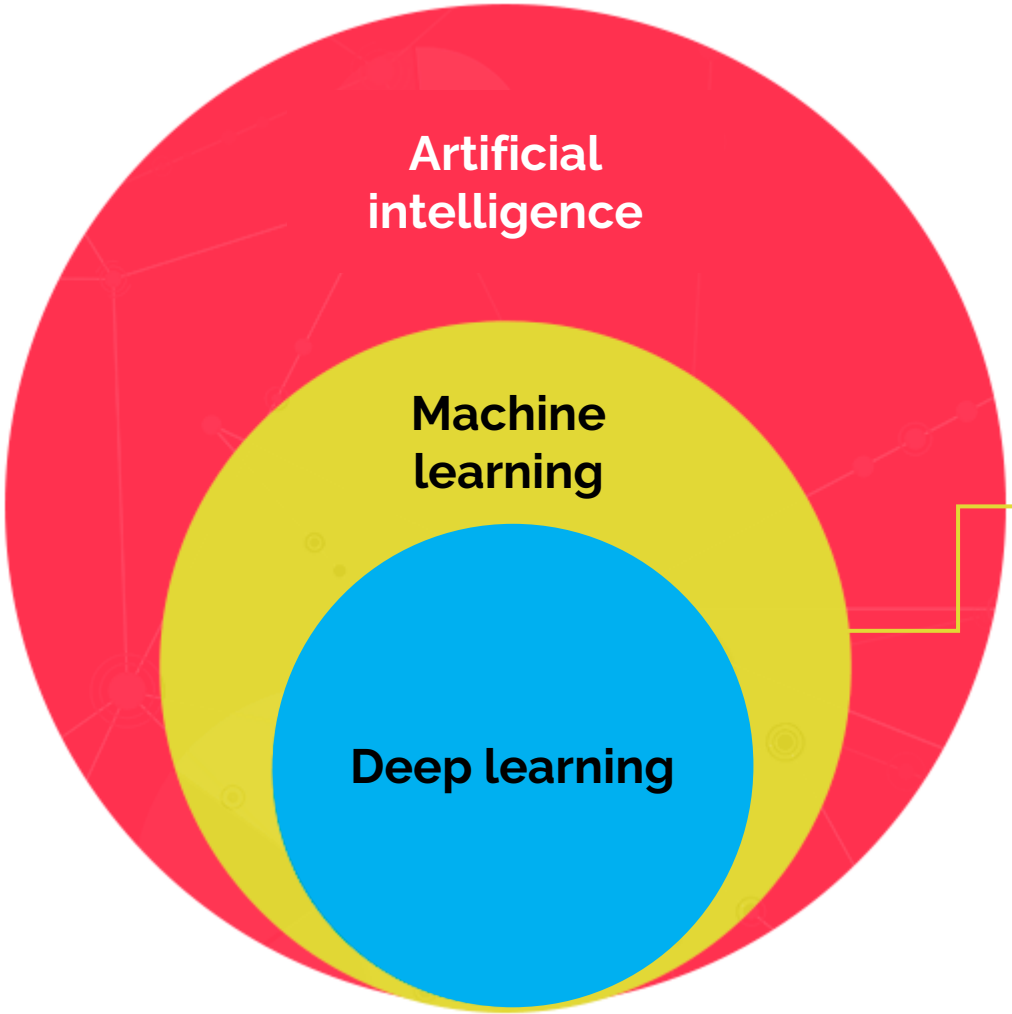
Il prodotto è discreto suppongo

NEUTRAL

Il vostro team di supporto è inutile

NEGATIVE

Machine learning & deep learning (1)



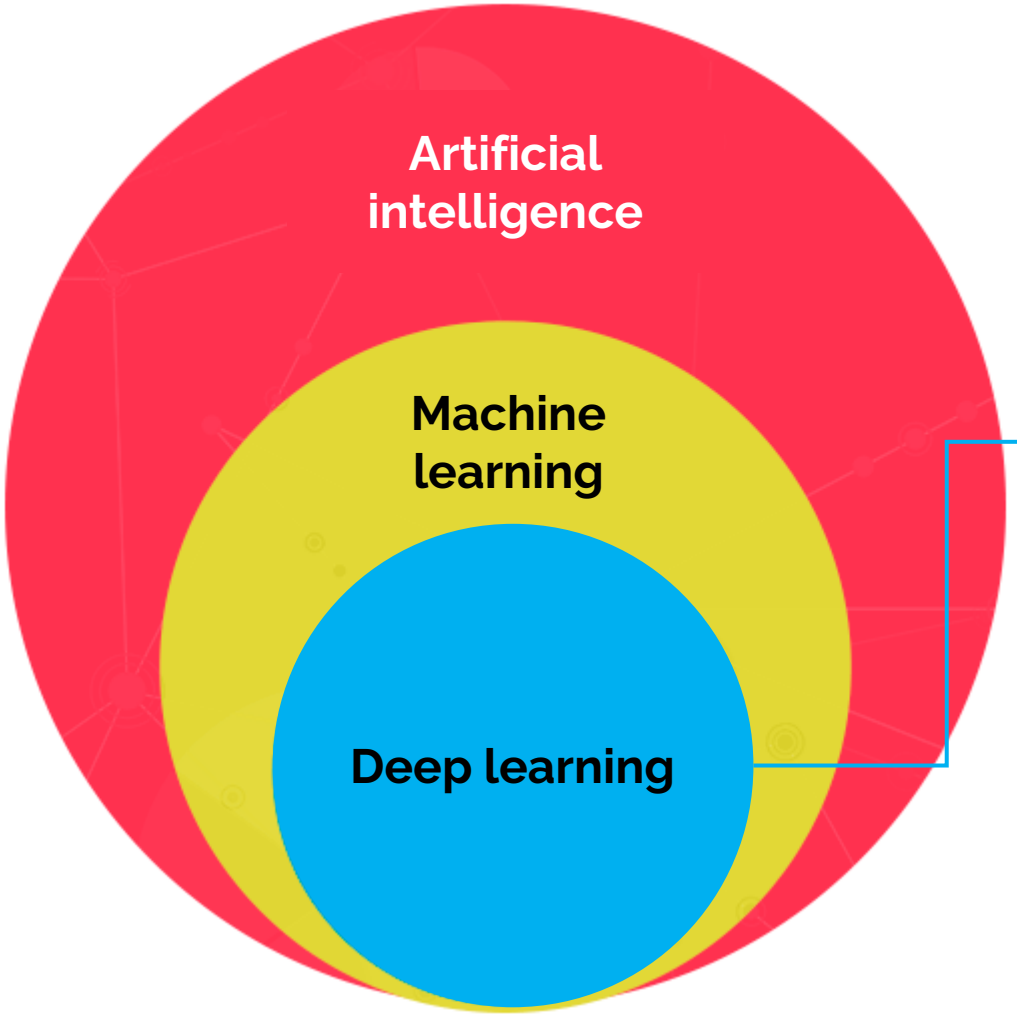
Machine learning) algoritmo che imparano dai dati (invece di scrivere manualmente delle regole)

Si estraggono caratteristiche dai dati (es. nel caso delle immagini estraggo bordi, colori, parole più frequenti…),

poi si usa un algoritmo che impara a classificare o predire (es. decision tree, SVM, k-NN).

Qui l'essere umano sceglie quali caratteristiche dare in pasto al modello di machine learning (handcrafted feature extraction step)

Machine learning & deep learning (2)



Deep learning) È una sottofamiglia del Machine Learning.

Usa reti neurali profonde (con molti strati) che imparano sia le caratteristiche che la regola per prendere decisioni.

Non serve estrarre a mano i bordi o i colori: la rete li impara direttamente dai pixel dell'immagine o dalle parole del testo (in generale si impara direttamente dal dato).

Machine learning & deep learning (3)

Machine learning) per riconoscere gatti, io programmatore dico alla macchina: “guarda le orecchie a punta, i baffi, i colori…” e poi un classificatore decide se le caratteristiche sono associate al cane o al gatto.

Deep learning) mostro milioni di foto alla rete neurale, e lei impara da sola quali caratteristiche distinguono i gatti dai cani..

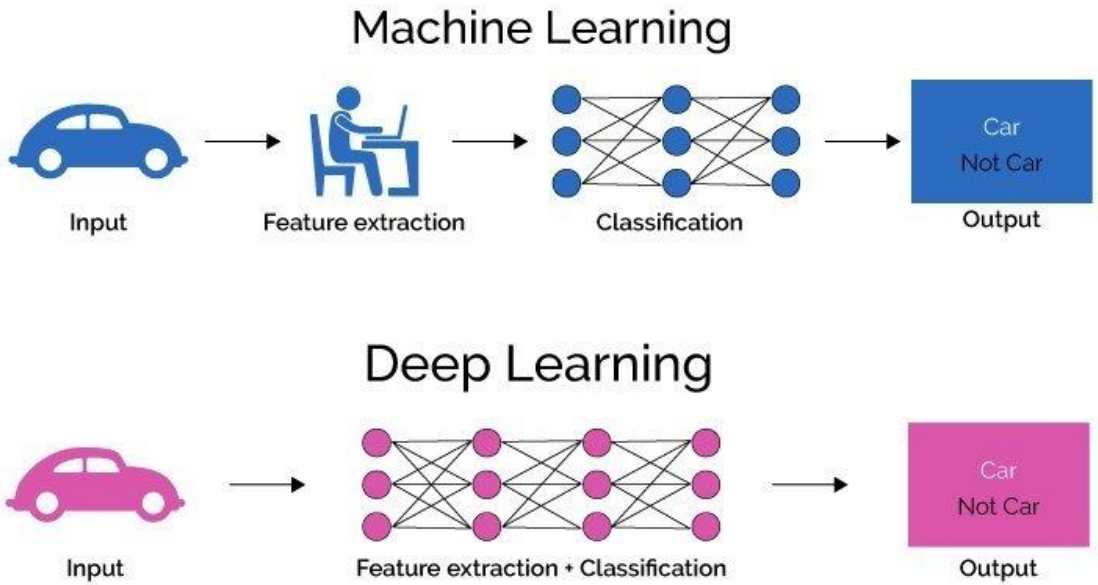
Machine learning = imparare dai dati in senso ampio

Deep learning = un tipo di machine learning basato su reti neurali profonde che fanno tutto «end-to-end»

Machine learning & deep learning (4)

Machine learning) per riconoscere gatti, io programmatore dico alla macchina: “guarda le orecchie a punta, i baffi, i colori…” e poi un classificatore decide se le caratteristiche sono associate al cane o al gatto.

Deep learning) mostro milioni di foto alla rete neurale, e lei impara da sola quali caratteristiche distinguono i gatti dai cani..



Fasi degli algoritmi

Il Machine Learning è un approccio dell'intelligenza artificiale in cui un sistema impara dai dati attraverso esempi.

Ogni algoritmo di Machine Learning funziona in due fasi:

- **Training (apprendimento o addestramento)**
  - La macchina riceve molti esempi (milioni di dati). Da questi, impara le regole per svolgere un certo compito.
  - Debbo costruire un set di addestramento

- **Testing (utilizzo o inference o predizione)**
  - Una volta "addestrato" il modello, lo si usa su nuovi dati per fare previsioni o classificazioni.
  - Testo su dati nuovi **mai visti in fase di addestramento**

Dataset di addestramento (1)

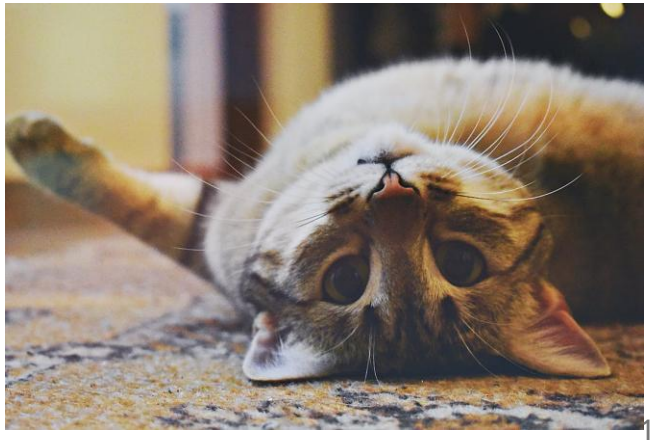
Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*







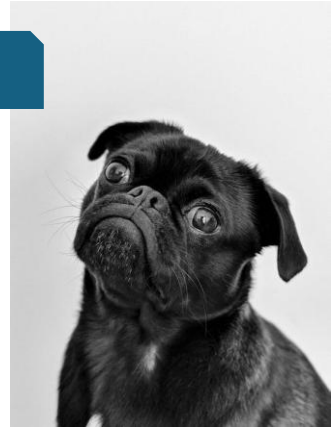



Dataset di addestramento (2)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

cane



gatto

Le immagini devono avere etichette corrette (niente gatti etichettati come cani e viceversa!)

Altrimenti l'addestramento algoritmico è compromesso

Dataset di addestramento (3)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*




Serve un numero simile di immagini di cani e di gatti

Se ci sono 10.000 gatti e solo 500 cani, l'algoritmo imparerà a dire sempre "gatto"

Dataset di addestramento (4)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

Serve avere immagini da più prospettive: di fronte, di profilo, sdraiati, in movimento...

Altrimenti il modello riconosce solo pose tipiche (es. cane sempre seduto).

Dataset di addestramento (5)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

Ci sono gatti bianchi, neri, a pelo lungo… cani piccoli, grandi, chiari, scuri.

Se nel dataset ci sono solo gatti neri e cani bianchi, l'algoritmo impara a distinguere il colore del pelo, non l'animale.

Dataset di addestramento (6)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

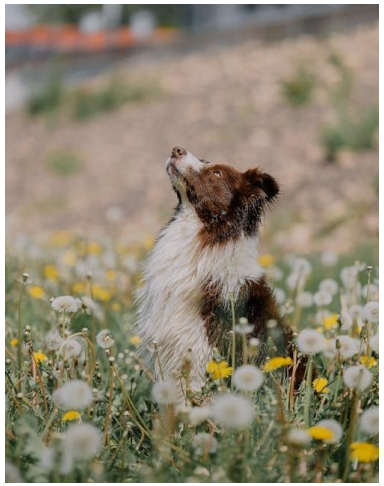


Foto in condizioni diverse (luce naturale, artificiale, scarsa).

Questo evita che il modello associ il tipo di luce a una classe.

Dataset di addestramento (7)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

Foto con più animali, animali parzialmente visibili o occlusi.

Questo aiuta a rendere l'algoritmo più robusto.

Dataset di addestramento (8)

Task) Classificare a partire da un'immagine se è presente un cane oppure un gatto

Come costruisco il dataset di *addestramento algoritmico?*

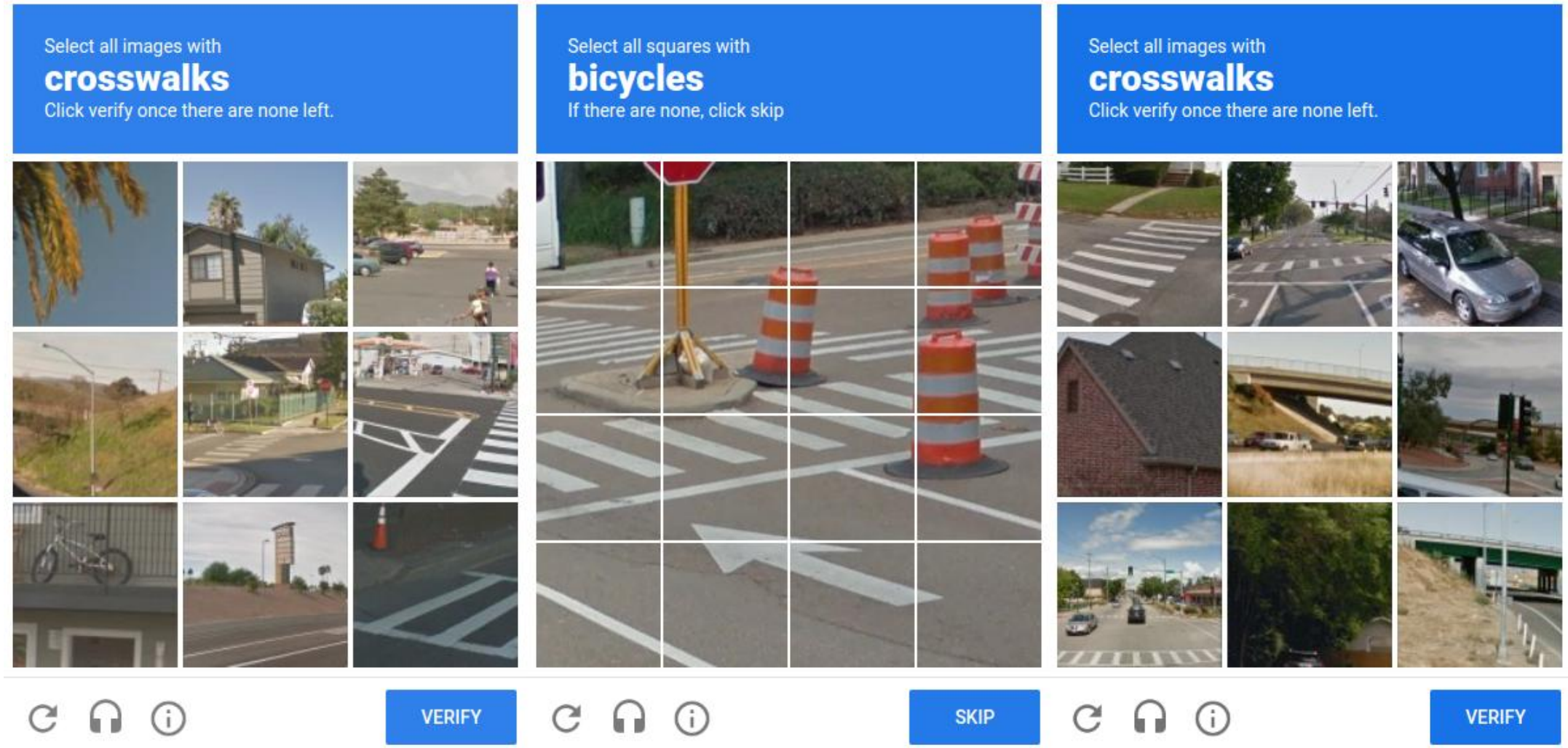
I cani e i gatti non devono comparire sempre sullo stesso sfondo.

Se tutti i gatti sono fotografati in casa e i cani all'aperto, l'algoritmo imparerà a classificare lo sfondo, non l'animale..

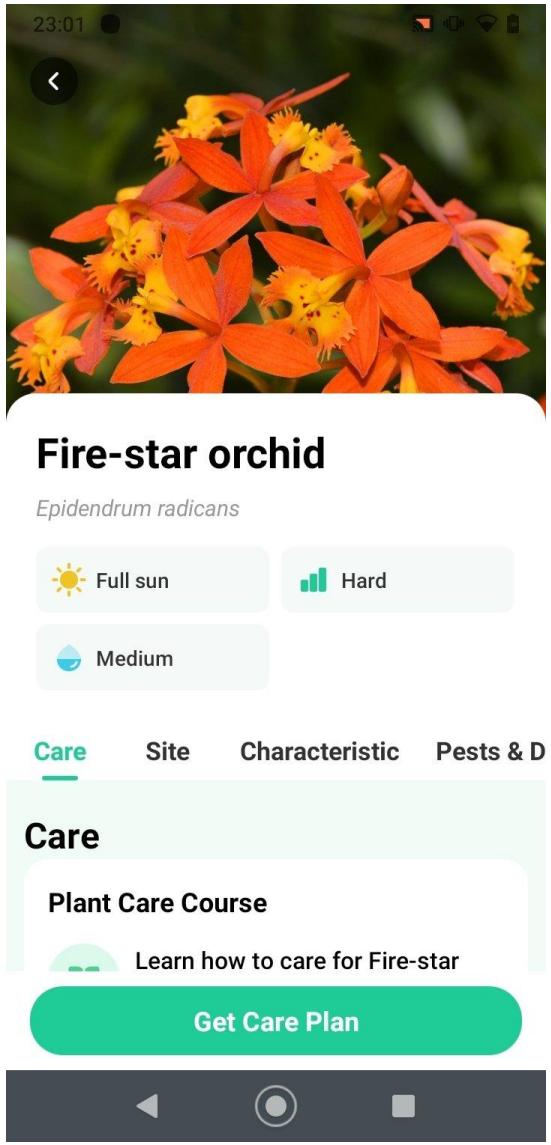
Mai etichettato dati per addestramenti algoritmici?

Mai stati parte di un processo di etichettatura dei dati ai fini degli allenamenti algoritmici?

Mai etichettato dati per addestramenti algoritmici?



Training vs test set (1)



Training set)

Valgono tutti i criteri (bilanciamento, varietà di sfondi, razze, angolazioni, illuminazione…).

Serve a insegnare al modello a riconoscere cani e gatti in situazioni diverse.

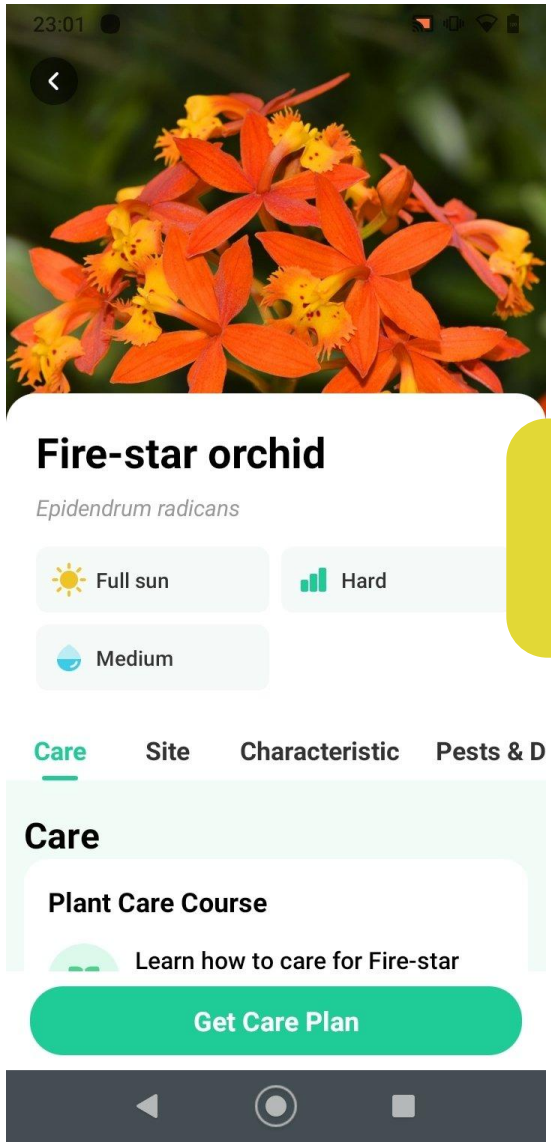
Test set)

Non si applicano i criteri di costruzione (non dobbiamo “prepararlo” per aiutare il modello).

Deve contenere casi realistici, anche squilibrati o difficili, perché il suo scopo è valutare come il modello si comporta nel mondo reale.

Deve rimanere separato dai dati di training!

Training vs test set (2)



Scatto fotografie (raccolgo un test set) per testare un'app di riconoscimento delle piante

Training set)

Valgono tutti i criteri (bilanciamento, varietà di sfondi, razze, angolazioni, illuminazione…).

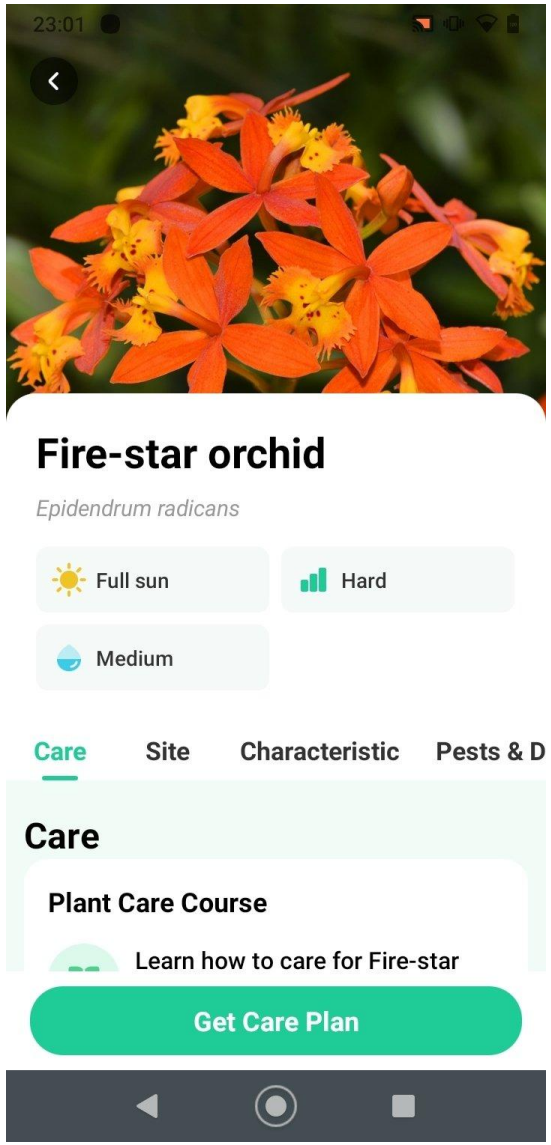
Serve a insegnare al modello a riconoscere cani e gatti in situazioni diverse.

i criteri di costruzione (non dobbiamo "prepararlo" per aiutare il modello).

Deve contenere casi realistici, anche squilibrati o difficili, perché il suo scopo è valutare come il modello si comporta nel mondo reale.

Deve rimanere separato dai dati di training!

Training vs test set (3)



Ogni task (compito da risolvere) richiede dati diversi.

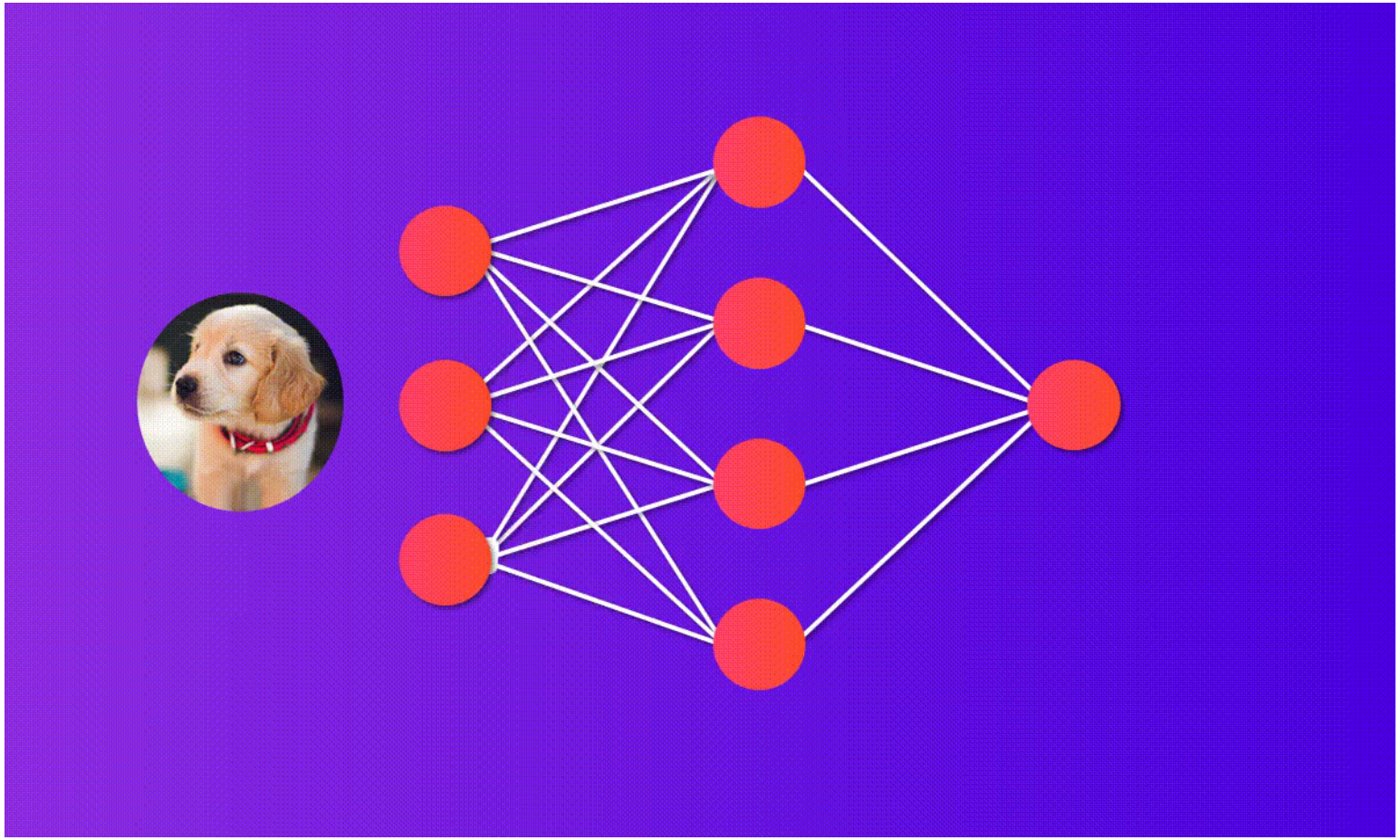
Non esiste un dataset “universale” che vada bene per tutto.

Task: Riconoscere cani vs gatti
Servono immagini etichettate di cani e gatti.Se uso foto di altri animali, il modello non impara a distinguere cani e gatti.

Task: Sentiment Analysis su tweet
Servono testi di tweet con etichetta (positivo, negativo, neutro).
Non avrebbe senso usare immagini o recensioni di prodotti.

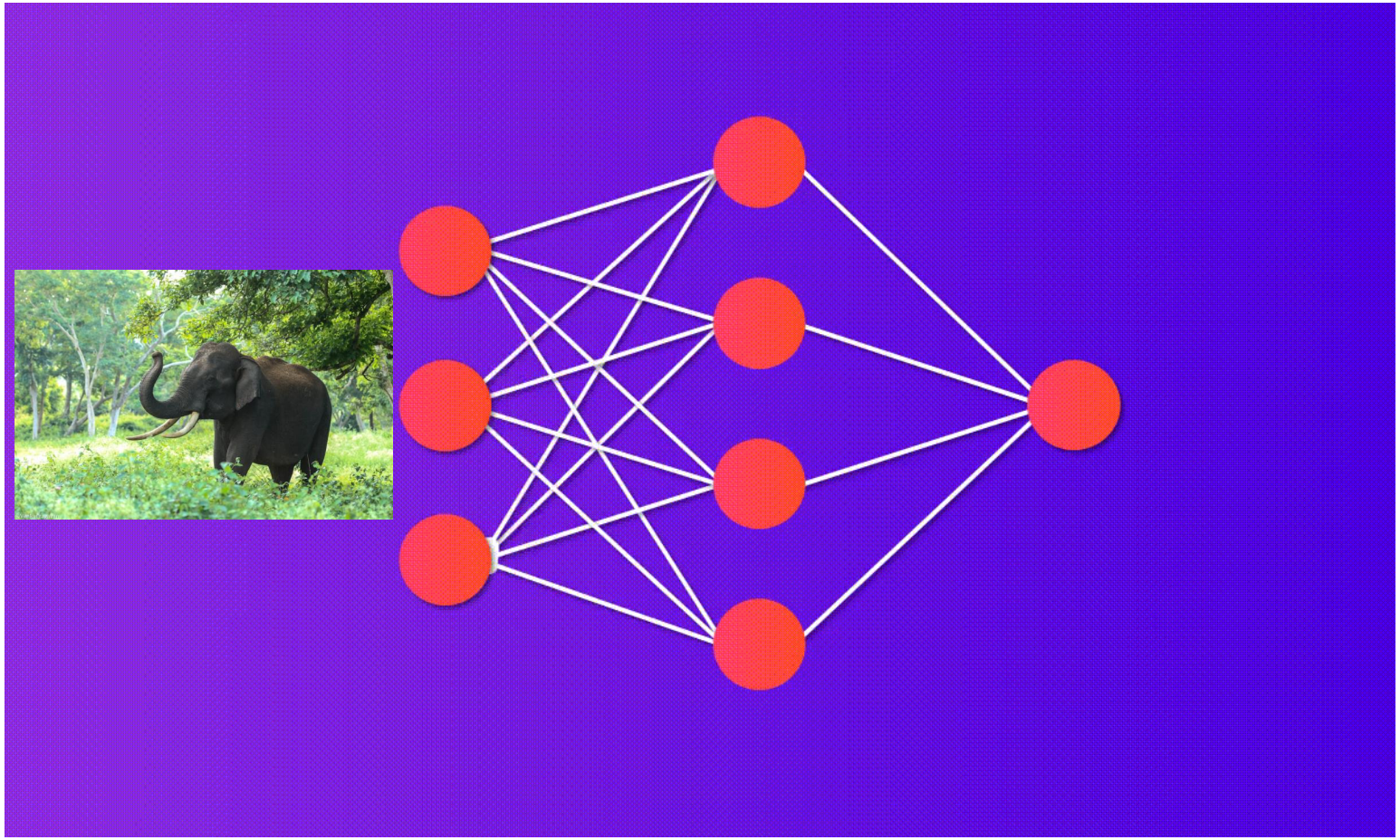
Task: Riconoscere cifre scritte a mano
Dataset MNIST (0–9 scritti a mano).
Non posso riutilizzarlo per riconoscere lettere dell’alfabeto.

Training vs test set (4)



Che succede se garantisco in input, in fase di test, la fotografia di un elefante?

Training vs test set (5)

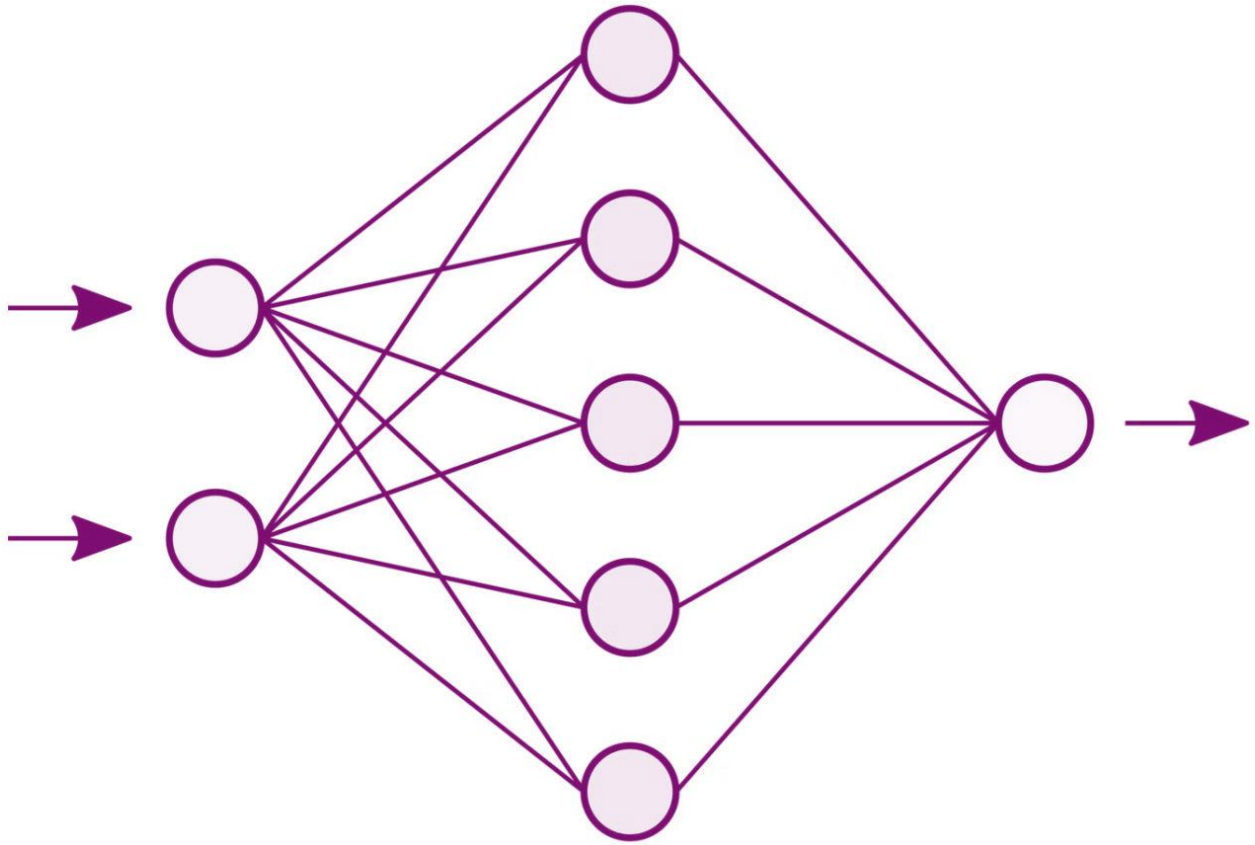


Un classificatore cane/gatto sa solo due cose: “cane” o “gatto”.

Se gli si mostra una foto di un elefante, lui non può rispondere “elefante”, perché non ha mai visto elefanti nei dati di addestramento.

Risponderà comunque “cane” o “gatto”, in base a quale categoria gli sembra “meno lontana” → ma sarà sbagliato.

Reti neurali (1)



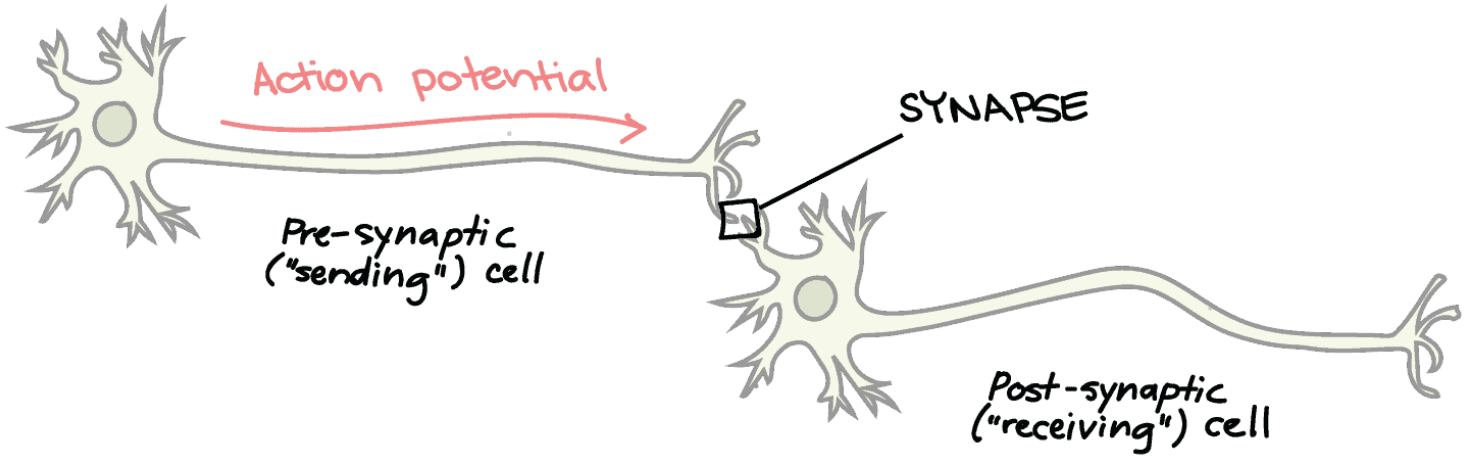
- Le **reti neurali** sono un tipo di algoritmo di Machine Learning.
- Oggi sono tra gli standard più usati per analizzare immagini e audio.
- Una rete neurale è come una scatola nera che riceve un input (es. immagine di un gatto).L'input attraversa tanti "neuroni" collegati tra loro.
- Alla fine la rete produce un output (es. «È un gatto» oppure «è un cane»)

Reti neurali (2)

Ispirazione biologica delle reti neurali)

- Un neurone comunica con altri neuroni attraverso segnali elettrici e chimici.
- Il neurone presinaptico rilascia neurotrasmettitori nello spazio sinaptico.
- I recettori del neurone postsinaptico captano i segnali.
- Se il segnale supera una soglia, il neurone si attiva e genera un potenziale d'azione (uno "spike").
- Questo impulso viaggia lungo l'assone e stimola altri neuroni

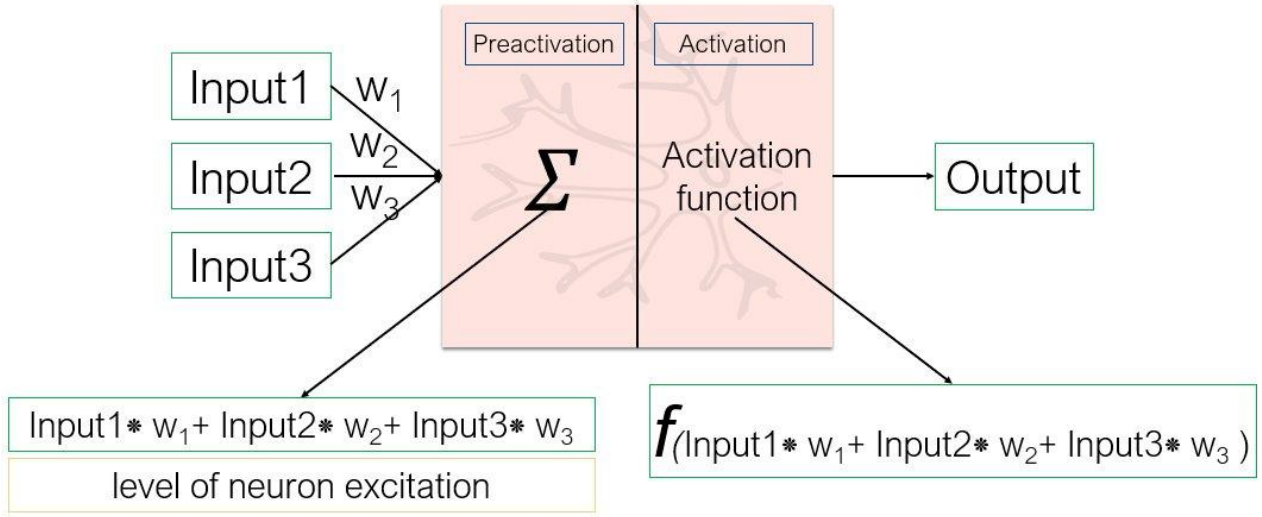
Se ricevo abbastanza segnali, sparo un impulso



Reti neurali (3)

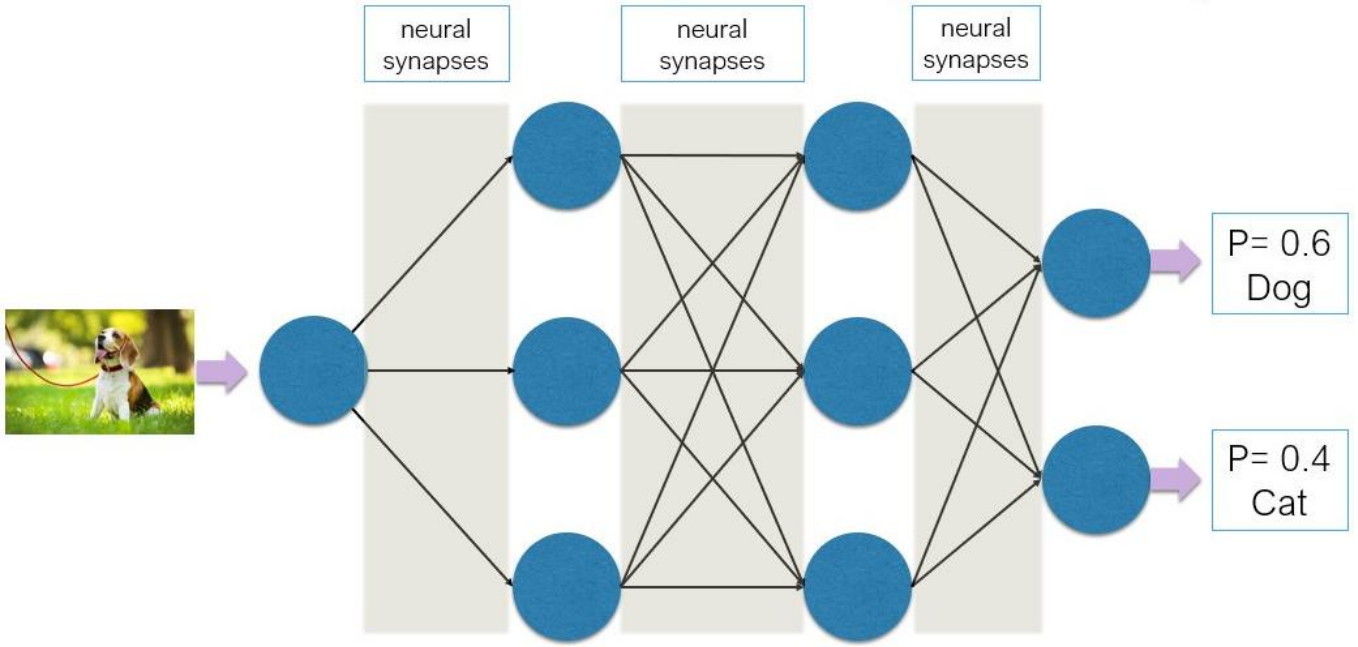
Neurone artificiale

- Un neurone artificiale è un modello matematico che imita questo comportamento:
  - Input: riceve numeri (es. pixel di un'immagine).
  - Pesi: ogni input ha un peso (quanto è importante).
  - Somma + soglia: si calcola un valore totale.
  - Attivazione: se supera una certa soglia → il neurone "si accende" (output = 1), altrimenti resta spento (output = 0).



Se la somma degli input supera la soglia, produco un output

Reti neurali (4)



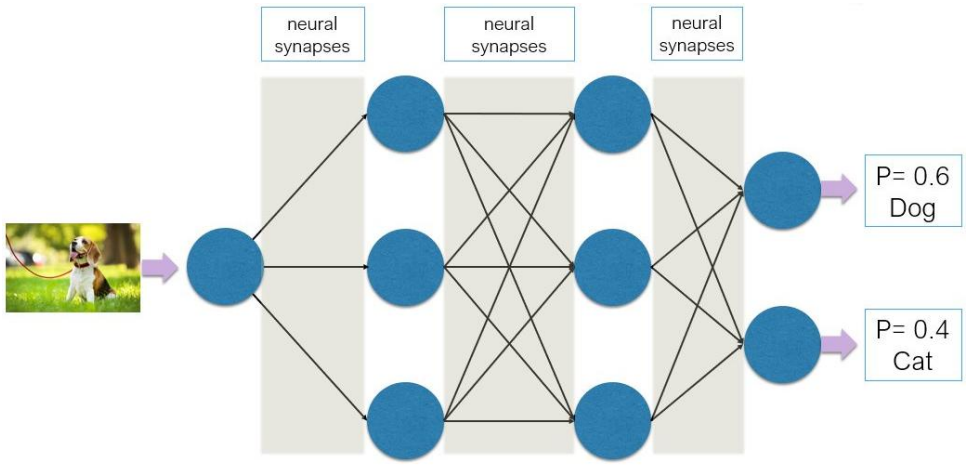
- Una rete neurale è come una scatola che riceve un input (es. immagine di un gatto). L'input attraversa tanti "neuroni" collegati tra loro.

- Alla fine la rete produce un output (es. "È un cane"). L'output è in forma di probabilità

Funzione obiettivo (1)

- Serve a misurare quanto l'output della rete è vicino alla risposta corretta.

- Più l'errore è alto → peggio funziona la rete. **L'allenamento della rete serve proprio a minimizzare questa funzione**.



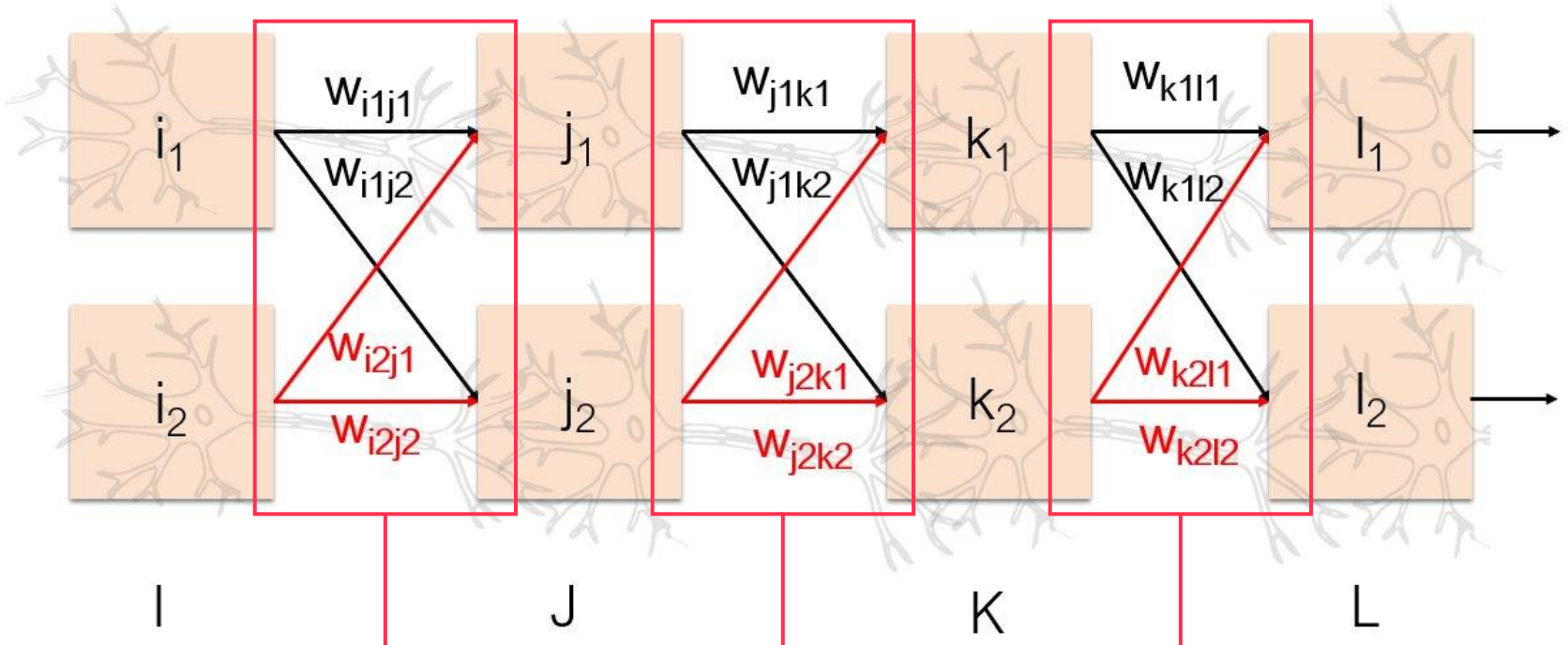
- Task: riconoscere un cane in foto.
- Output rete: "probabilità 0.4 gatto, 0.6 cane".
- L'output deve avvicinarsi il più possibile all'etichetta cane che ha probabilità = 1
- La funzione obiettivo calcola la differenza tra 0.6 e 1.

Funzione obiettivo (2)

- **Obiettivo del gioco)** centrare il bersaglio
- L'etichetta corretta = centro del bersaglio.
- L'output del modello = dove finisce la freccetta.

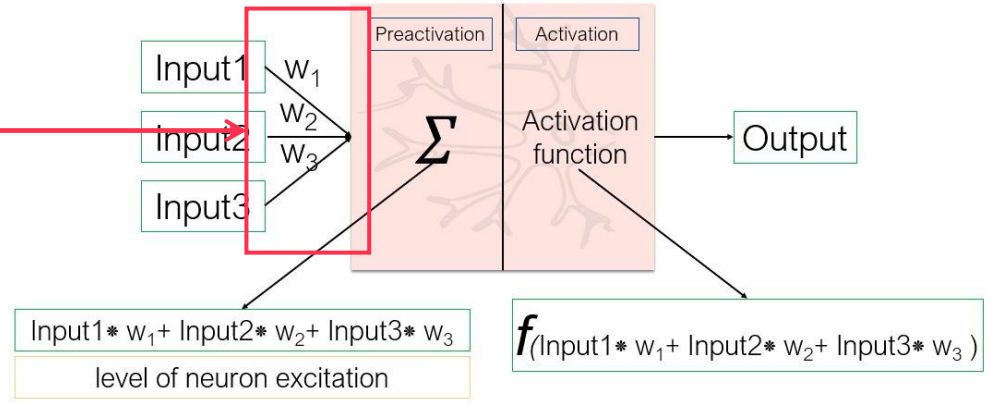
L'allenamento serve a spostare i lanci sempre più vicini al centro.

Parametri (1)



Ogni connessione tra neuroni ha un parametro (un numero, un **peso** associato alla connessione)

Durante l'allenamento i parametri vengono modificati per ridurre l'errore misurato dalla funzione obiettivo.



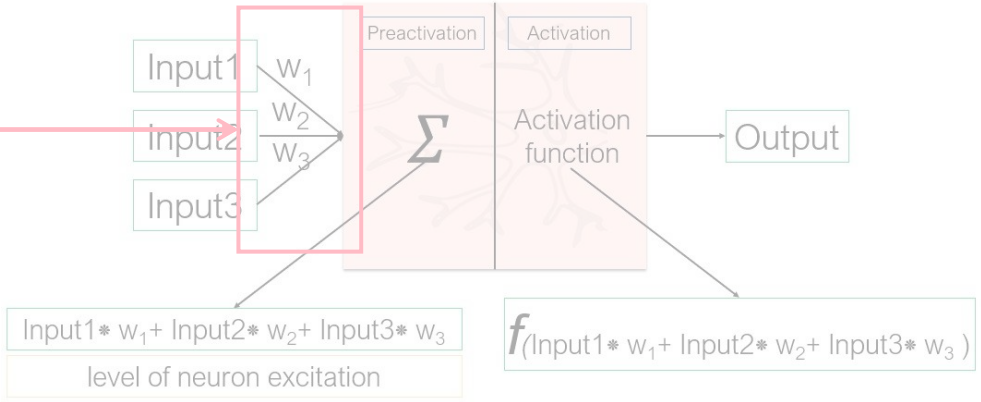
Parametri (2)

I parametri si possono immaginare come i pulsanti di un mixer audio:

- se li regolo male → l'uscita è confusa (rete poco precisa).
- se li regolo bene → il suono è chiaro (rete funziona bene).

alla connessione)

Durante l'allenamento i parametri vengono modificati per ridurre l'errore misurato dalla funzione obiettivo.



Funzione obiettivo e parametri

La funzione obiettivo = misura l'errore

Parametri = manopole che si regolano per ridurre l'errore

Come impara una rete neurale in fase di addestramento?

Prendiamo come esempio la classificazione di cani e gatti da fotografie scattate con lo smartphone

Apprendimento – backpropagation (1)

La rete riceve un input (es. foto di un cane)
e produce un output in forma di probabilità
(es. cane=0.7)

Apprendimento – backpropagation (2)

La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

↓

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Apprendimento – backpropagation (3)

La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Confrontiamo l'output con l'etichetta corretta per sapere quanto siamo lontani dalla verità **(errore = 1-0.7)**

Apprendimento – backpropagation (4)

La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Confrontiamo l'output con l'etichetta corretta per sapere quanto siamo lontani dalla verità **(errore = 1-0.7)**

Che succede a questo errore?

Apprendimento – backpropagation (5)

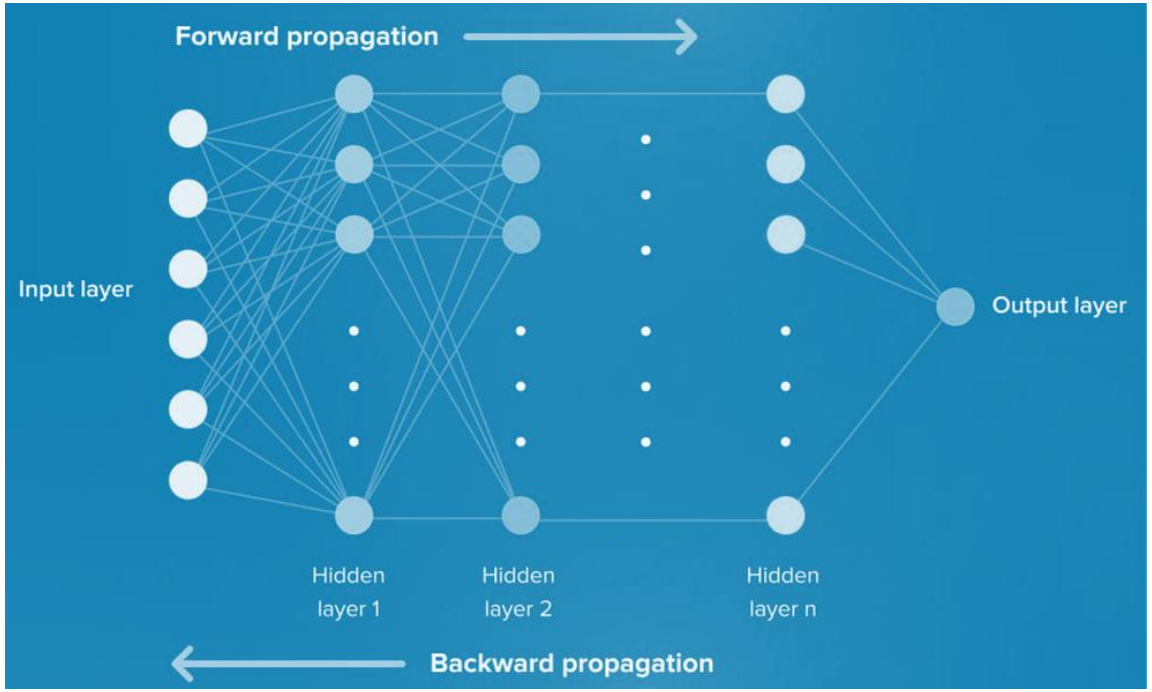
La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Confrontiamo l'output con l'etichetta corretta per sapere quanto siamo lontani dalla verità **(errore = 1-0.7)**

Che succede a questo errore?

Viene retropropagato. Così ogni connessione sa se ha contribuito bene o male al risultato finale



Apprendimento – backpropagation (6)

La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Confrontiamo l'output con l'etichetta corretta per sapere quanto siamo lontani dalla verità **(errore = 1-0.7)**

Che succede a questo errore?

Viene retropropagato. Così ogni connessione sa se ha contribuito bene o male al risultato finale

I pesi vengono leggermente regolati.

- Se un collegamento ha portato a un errore, il suo peso viene ridotto.
- Se ha portato a una risposta giusta, viene rafforzato.

Apprendimento – backpropagation (7)

La rete riceve un input (es. foto di un cane) e produce un output in forma di probabilità **(es. cane=0.7)**

Del dato di training sappiamo l'etichetta quindi confrontiamo l'ouput con l'etichetta effettiva associata al dato **(cane = 1)**

Confrontiamo l'output con l'etichetta corretta per sapere quanto siamo lontani dalla verità **(errore = 1-0.7)**

Che succede a questo errore?

Viene retropropagato. Così ogni connessione sa se ha contribuito bene o male al risultato finale

I pesi vengono leggermente regolati.

- Se un collegamento ha portato a un errore, il suo peso viene ridotto.
- Se ha portato a una risposta giusta, viene rafforzato.

Questo processo si ripete migliaia/milioni di volte con tante immagini di cani e gatti.

L'allenamento si ferma quando: **l'errore diventa molto piccolo, oppure il modello non migliora più** anche se continua ad allenarsi.

Parametri della rete ed intelligenza della rete

GPT-3	GPT-4 «32k»
175B (miliardi)	1.76T (migliaia di miliardi di parametri)

Più parametri = più capacità di riconoscere schemi complessi e generare testo accurato.
Ma significa anche più costi di addestramento e più risorse computazionali.

Large language model (LLM)

- Sono modelli di IA basati su reti neurali (un particolare tipo di rete neurale!).
- Sono addestrati su grandi quantità di testo.
- Servono per comprendere e generare testo (traduzioni, risposte,…)

- **È un modello di linguaggio perché dentro ai suoi parametri (pesi) ha acquisito la capacità di comprendere e generare del testo**

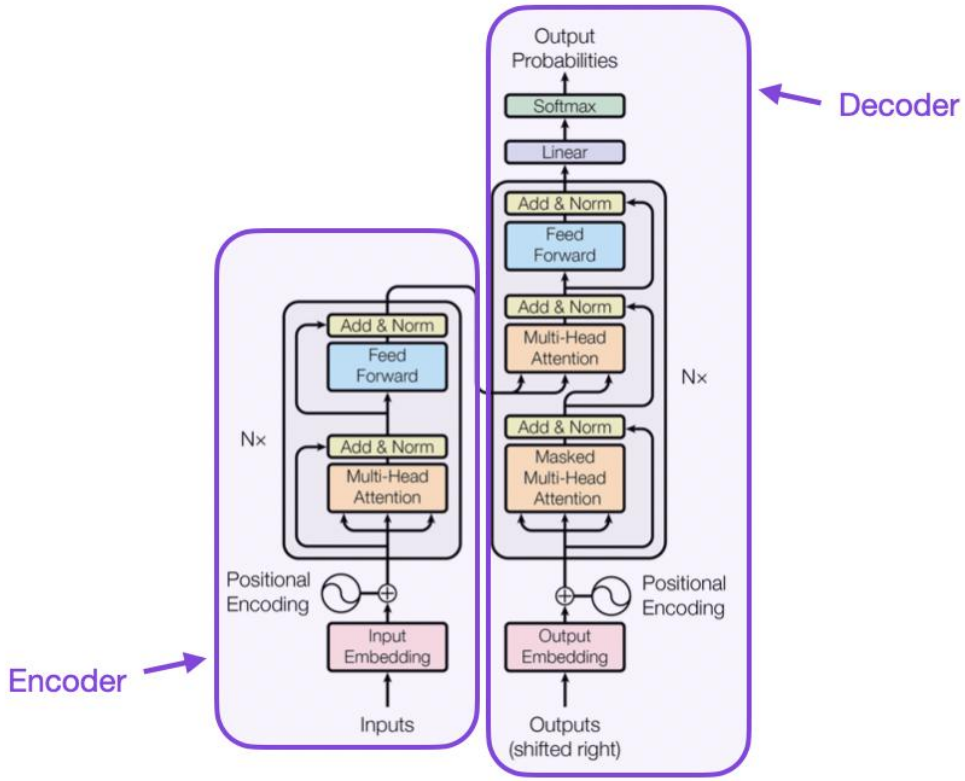


Figure 1: The Transformer - model architecture.

Transformer

- Particolare architettura di reti neurali su cui sono basati i LLM ed hanno un meccanismo al loro interno di attenzione

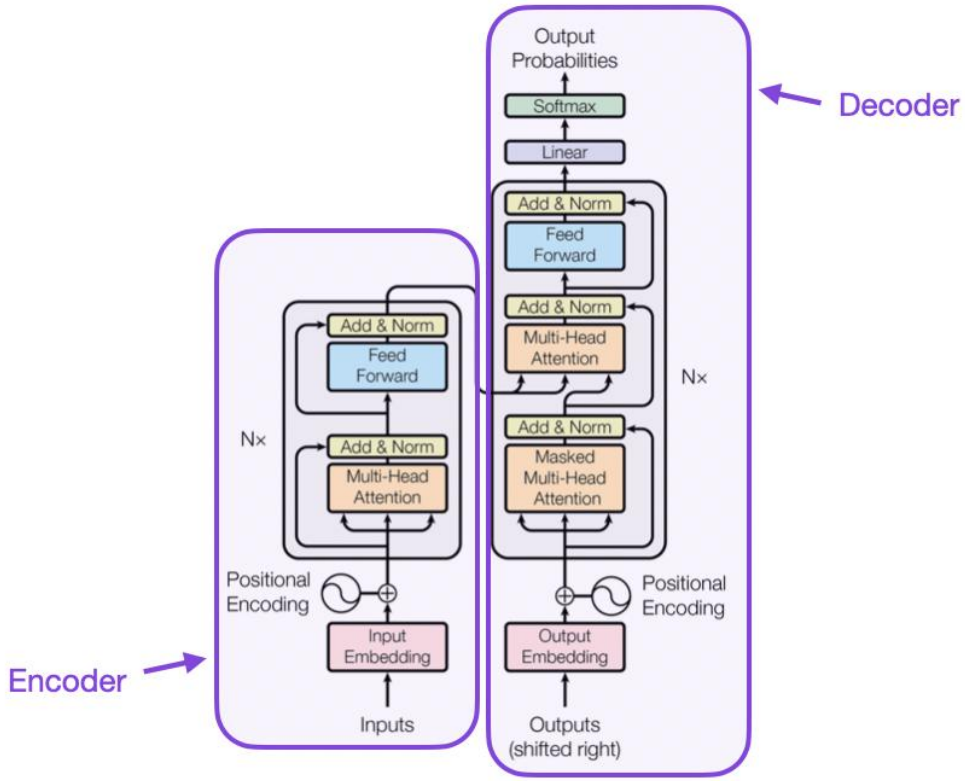
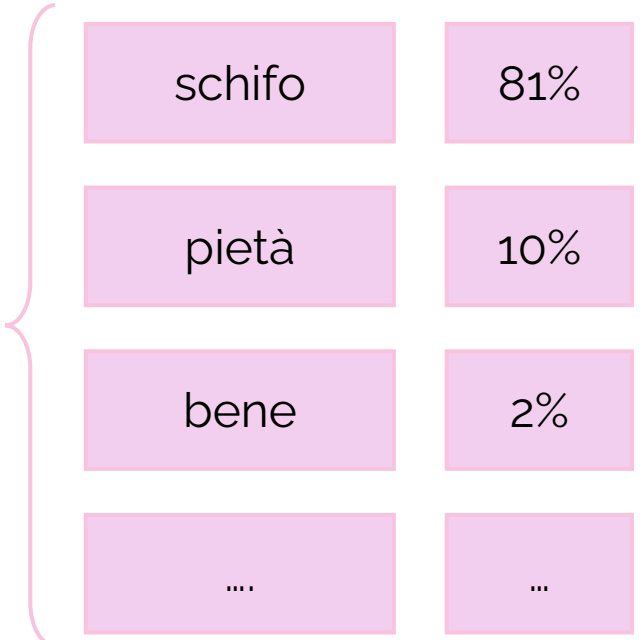


Figure 1: The Transformer - model architecture.

LLM e task da risolvere (1)

- **Next token prediction:** capacità di prendere un testo e suggerire la prossima parola



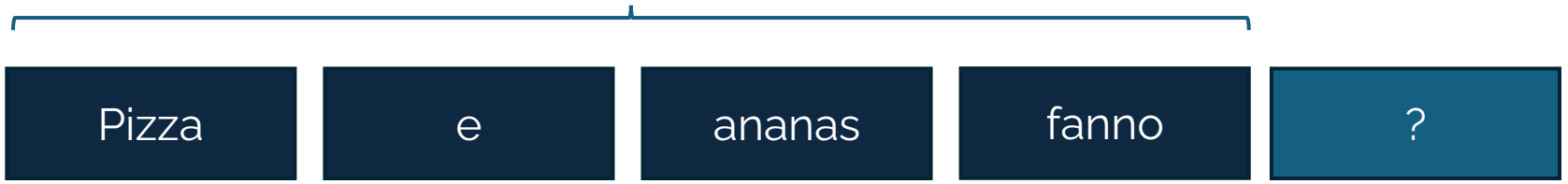
- Un Large Language Model predice la parola successiva in una frase.
- Il modello riceve una sequenza di parole già scritte (Pizza e ananas fanno …). Calcola la probabilità di tutte le possibili parole successive. Assegna una percentuale a ciascuna parola: “schifo” → 81%“pietà” → 10%“bene” → 2%… altre possibilità.
- Sceglie la più probabile che è la parola più probabile secondo i suoi parametri (“schifo”).
- Ripetendo questo processo parola dopo parola, il modello costruisce testi interi (frasi, risposte, articoli, codice…).

LLM e task da risolvere (2)

- **Next token prediction:** capacità di prendere un testo e suggerire la prossima parola

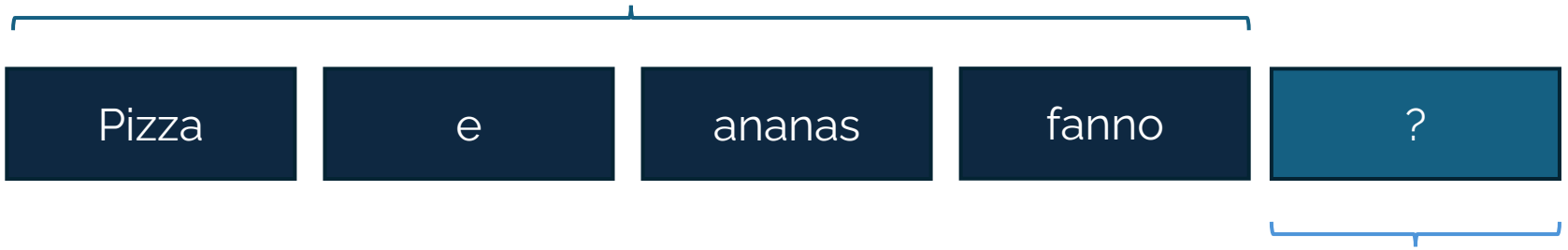
LLM e task da risolvere (3)

- L'idea di base è che questo sia il prompt (ciò che noi utenti diamo alla rete)



LLM e task da risolvere (4)

- L'idea di base è che questo sia il prompt (ciò che noi utenti diamo alla rete)



- La rete risponde «impilando» un token dopo l'altro: next token prediction

LLM e task da risolvere (5)

I Large Language Model sono addestrati per un unico compito: Next Token Prediction (predire la parola successiva).

Tuttavia, quando vengono allenati su **enormi quantità di dati, da questa semplice capacità emergono abilità più complesse**:

- Rispondere a domande
- Scrivere o riscrivere testi
- Generare codice
- Tradurre lingue
- Riassumere contenuti

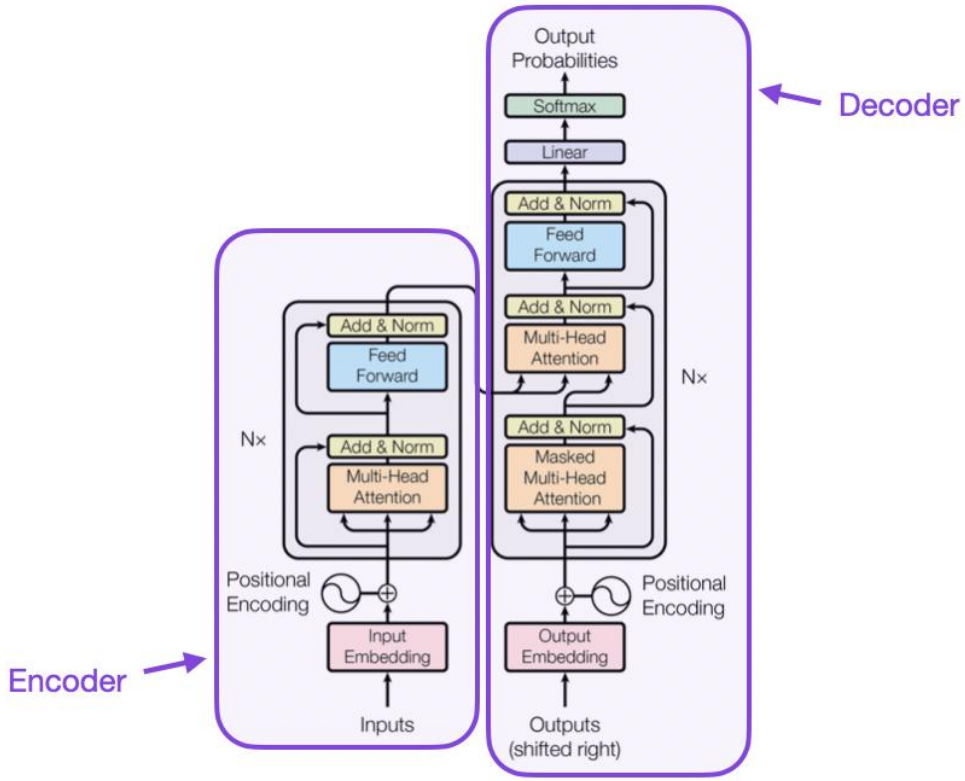
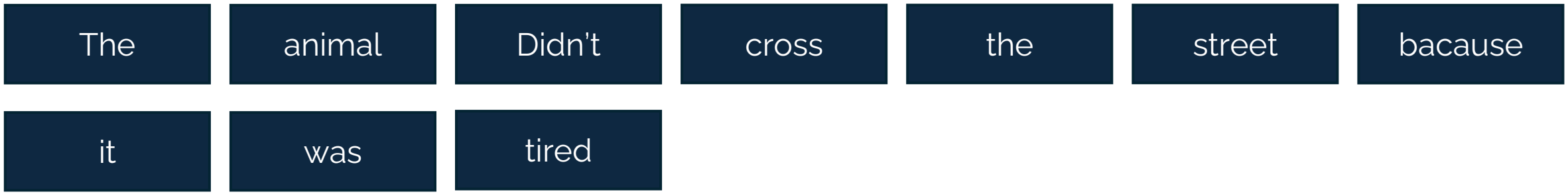


Figure 1: The Transformer - model architecture.

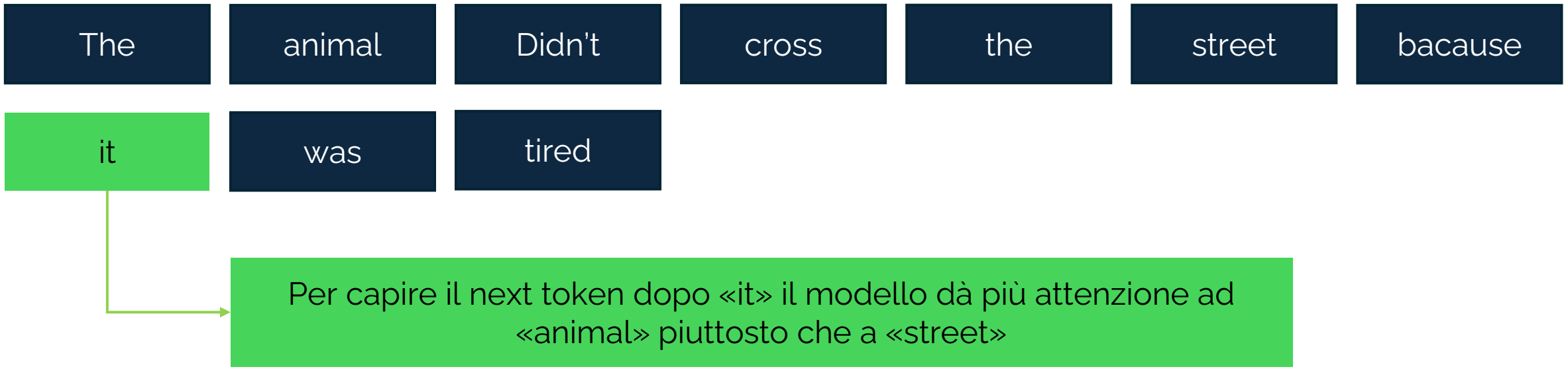
Meccanismo di attenzione nelle architetture transformer (1)

- È una tecnica che assegna pesi diversi alle parti di un input.
- Permette al modello di concentrarsi sulle informazioni più rilevanti.
- Così migliora l'accuratezza delle previsioni.

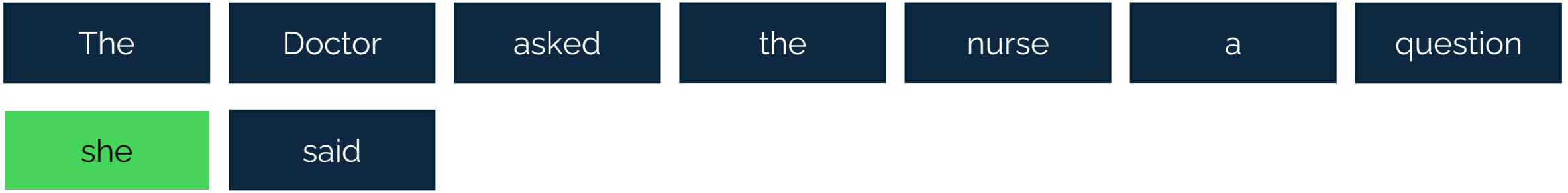


Meccanismo di attenzione nelle architetture transformer (2)

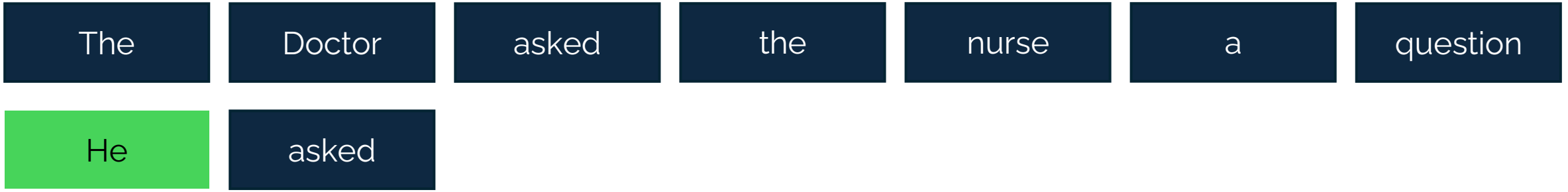
- È una tecnica che assegna pesi diversi alle parti di un input.
- Permette al modello di concentrarsi sulle informazioni più rilevanti.
- Così migliora l'accuratezza delle previsioni.



Meccanismo di attenzione bias



Se il modello vede “She”, l’attenzione tende ad andare su nurse.



Se il modello vede “He”, l’attenzione tende ad andare su doctor.

Meccanismo di attenzione bias

The	Doctor	asked	the	nurse	a	question
she	said					

Se il modello vede "Sh

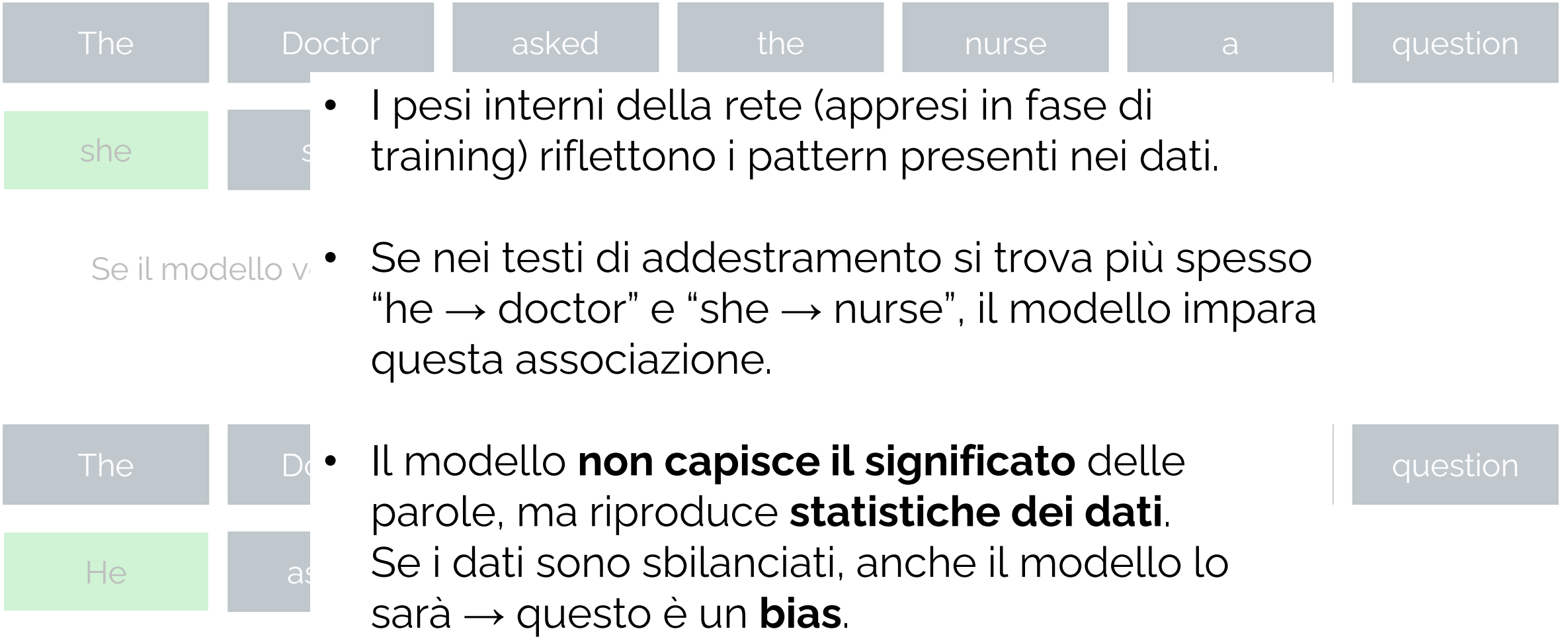
In inglese, sia *doctor* che *nurse* possono essere maschili o femminili, ma il modello associa ruoli di genere perché ha appreso questa correlazione dai dati di addestramento.

Cosa significa?

The	Doctor					question
He	asked					

Se il modello vede "He", l'attenzione tende ad andare su doctor.

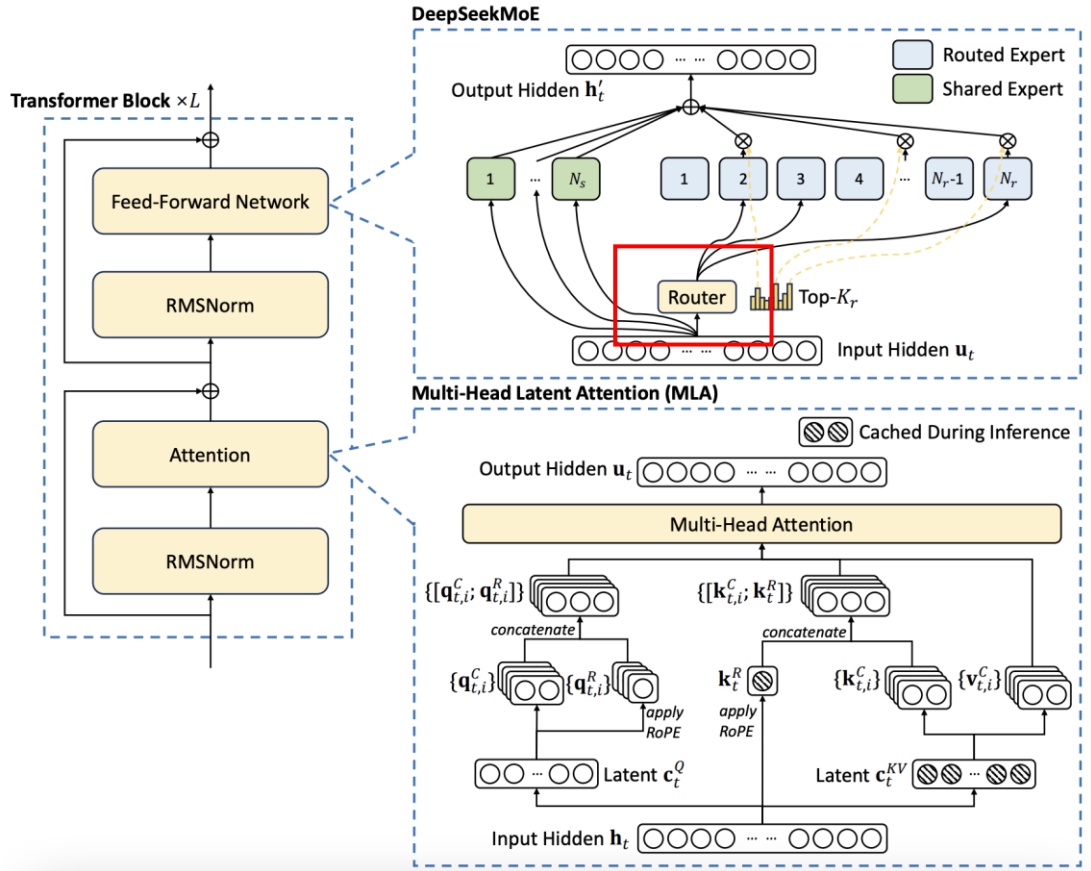
Meccanismo di attenzione bias



- I pesi interni della rete (appresi in fase di training) riflettono i pattern presenti nei dati.
- Se nei testi di addestramento si trova più spesso “he → doctor” e “she → nurse”, il modello impara questa associazione.
- Il modello **non capisce il significato** delle parole, ma riproduce **statistiche dei dati**.
  Se i dati sono sbilanciati, anche il modello lo sarà → questo è un **bias**.

Mixture of experts (1)

- Ultimi modelli LLM (deepseek)
- Un modello Mixture of Experts è formato da tanti “pezzi” specializzati, chiamati esperti.
- Non tutti gli esperti si attivano sempre: c’è un router (o gating network) che decide quale esperto usare a seconda dell’input.

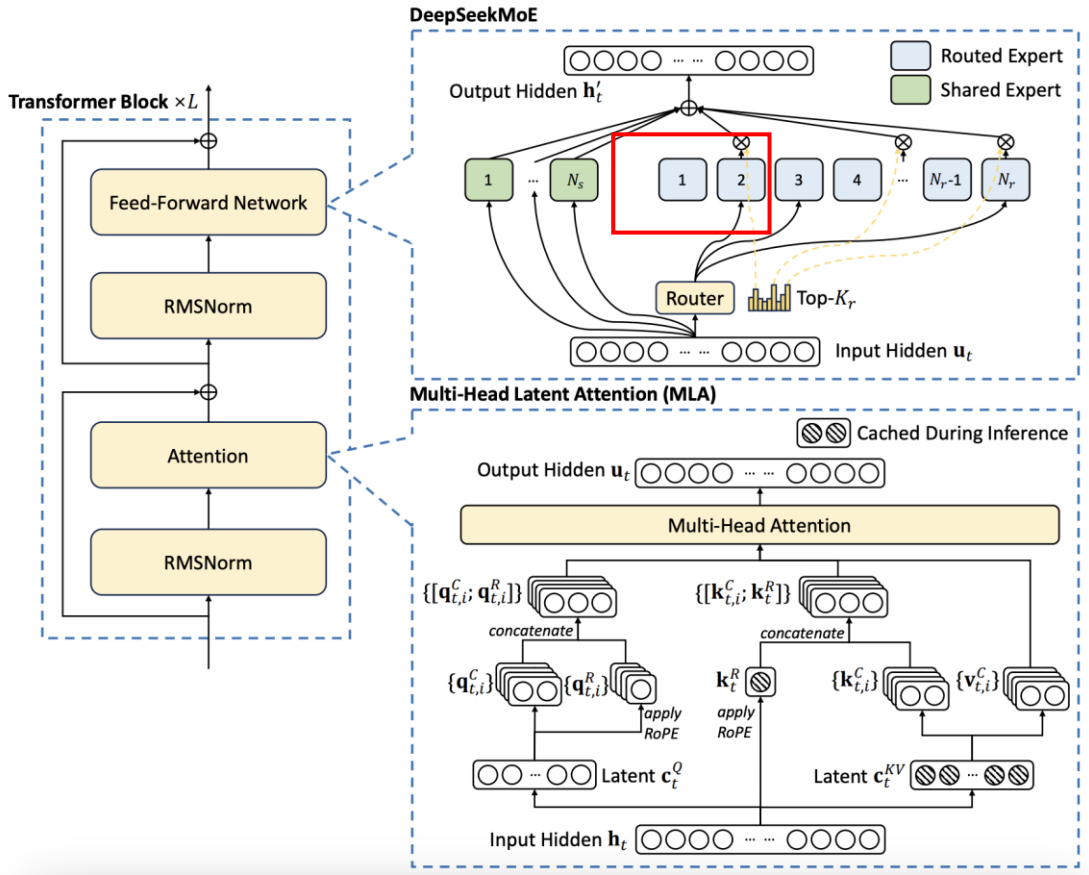


Mixture of experts (2)

- Efficienza: non serve accendere tutta la rete per ogni richiesta → solo alcuni esperti lavorano.
- Scalabilità: posso costruire modelli enormi senza usare sempre tutte le risorse.
- Il modello DeepSeek ha un MoE molto grande → riesce a essere più economico perché suddivide il lavoro tra tanti esperti

---

- L'attivazione degli esperti può cambiare da una richiesta all'altra.
- Questo significa che lo stesso prompt può produrre risposte leggermente diverse → il modello è non deterministico (non sempre risponde uguale allo stesso input (questo succede con tutti gli LLM)).



Transformer classico & Mixture of experts

- In un modello Transformer standard, tutti i neuroni e tutti i blocchi partecipano al calcolo.
- L'attenzione decide quali parole o parti dell'input pesano di più per predire il prossimo token.

- Nei Mixture of experts l'attenzione funziona ancora dentro ogni blocco attivato

- Non tutti i blocchi vengono attivati: è il router che decide quali esperti usare

- Oltre a «dare pesi alle parole» (attenzione), il modello sceglie «quali cervelli specializzati» usare (router)

- Il compito di base non cambia: il modello continua a predire la parola successiva. La differenza è come ci arriva:

- In un Transformer classico: tutte le parti del modello lavorano sempre → più costoso.
- In un MoE: solo alcuni esperti lavorano per quel token → stesso risultato, ma più efficiente.

Allenamento/apprendimento dei LLM (1)

Stato iniziale

Pre-training

Supervised fine-tuning

Alignment

Allenamento/apprendimento dei LLM (2)

Stato iniziale

Il modello non sa fare niente (i suoi pesi sono inizializzati in maniera randomica)

Allenamento/apprendimento dei LLM (3)

Insegnare a un bambino a parlare ascoltando milioni di conversazioni.

Pre-training

Legge tantissimo testo (internet, libri, articoli...).
Impara a fare il compito base: data una frase, indovinare la parola successiva (next token prediction).

Le manopole del mixer (i pesi) vengono regolate per riuscire a “predire” bene.

Allenamento/apprendimento dei LLM (4)

mandare lo studente a scuola: non impara solo a parlare, ma anche a risolvere esercizi di matematica, rispondere a domande di storia, scrivere in modo ordinato.

Supervised fine-tuning

Ora che il modello “sa parlare”, bisogna specializzarlo su compiti specifici.

Lo si addestra con esempi guidati da esseri umani (io ti faccio una domanda → tu mi dai una risposta corretta).

In questo modo affina le sue capacità.

Allenamento/apprendimento dei LLM (5)

È come insegnare le buone maniere: non solo dire le cose giuste, ma anche nel modo giusto

Alignment (RLHF - Reinforcement Learning with Human Feedback)

Anche se sa tante cose, il modello può rispondere in modo strano o poco utile.

Qui intervengono gli umani: leggono le risposte del modello e dicono “questa è migliore di quest’altra”.

Il modello viene quindi allineato per comportarsi più vicino a ciò che gli utenti vogliono.

Allenamento/apprendimento dei LLM (6)

Stato iniziale

Pre-training

Supervised fine-tuning

Alignment

In tutti e tre i casi, il meccanismo matematico che regola l'apprendimento algoritmico è sempre la backpropagation, ma cambia:
Come si costruisce il dataset, la funzione obiettivo (loss function), il risultato atteso.

Allenamento/apprendimento dei LLM (7)

Stato iniziale

Pre-training

Supervised fine-tuning

Alignment

- Qui l'obiettivo è solo predire bene la parola successiva: "Avevi previsto gatto, la parola giusta era cane → hai fatto un errore di X".
- È un errore che si può calcolare con formule classiche (tipo cross-entropy loss).
- La backpropagation propaga questo errore e aggiusta i parametri.

Allenamento/apprendimento dei LLM (8)

Stato iniziale

Pre-training

Supervised fine-tuning

Alignment

- Non basta più dire la parola giusta. Il modello deve rispondere in modo utile, sicuro, coerente con le preferenze umane.
- Per farlo, si introduce un modello di ricompensa (reward model), addestrato su valutazioni umane.

Zoom su alignment (RLHF) (1)

Gli umani leggono due (o più) risposte generate dal modello.

Es. alla domanda “Come cucino la pasta?”

- Risposta A: “Metti acqua, portala a ebollizione, aggiungi la pasta…”
- Risposta B: “Compra pasta al supermercato.”

Gli umani dicono: A è meglio di B.

Con questi confronti si allena un reward model, cioè un modello che impara a dare un punteggio (alto = risposta buona, basso = risposta scadente).

Durante l’RLHF, la loss non misura più solo “quanto hai sbagliato la parola successiva”, ma diventa: “Hai generato una risposta che il reward model giudica poco utile → *penalità*”. “Hai generato una risposta che il reward model valuta utile → *ricompensa (→ reward)*”.

Questo punteggio passa nella backpropagation, **che regola i parametri per spingere il modello a dare risposte sempre più vicine a quelle che gli umani apprezzano.**

Zoom su alignment (RLHF) (2)

In fase di allineamento il modello è stato «addestrato» ad essere più inclusivo!

Allineamento curiosità

LLAMA 3.1

15.6 mila miliardi di token !! Per il pre-training

10 milioni di token per il fine-tuning

Modelli base e instruct

Stato iniziale

Pre-training

Supervised fine-tuning

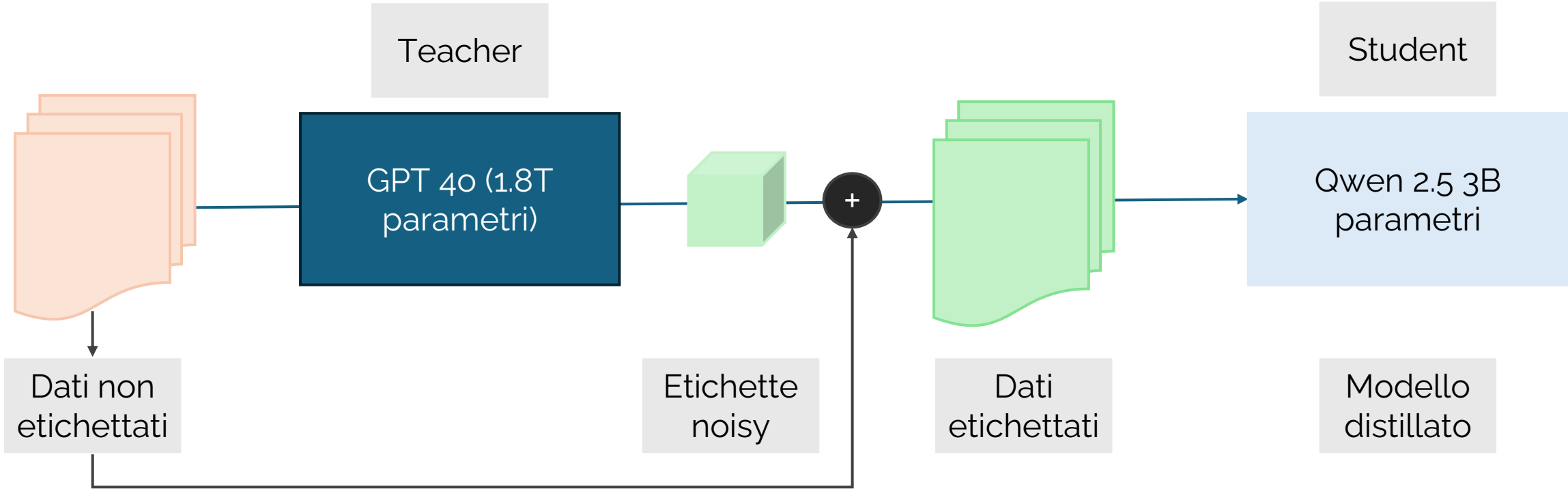
Alignment

Base o foundation model

Instruct model

Knowledge distillation (1)

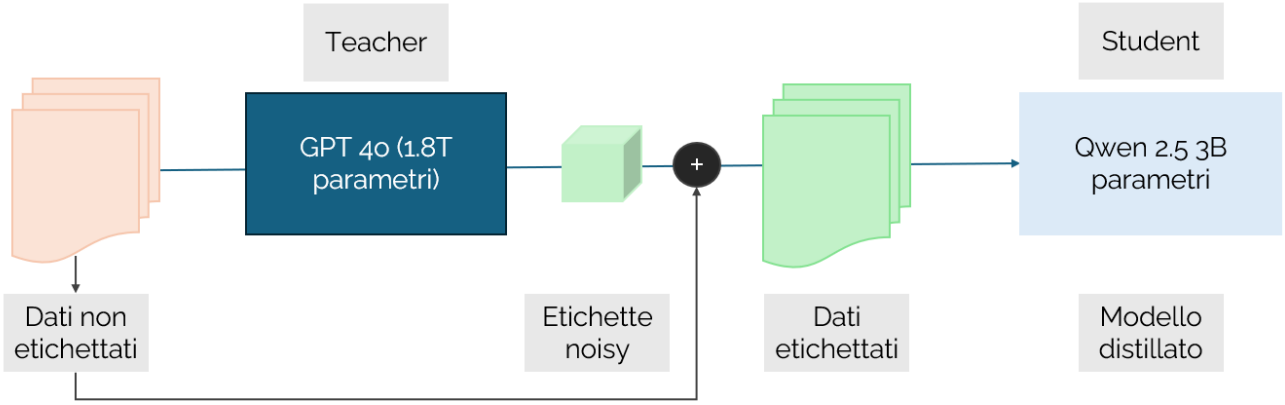
Tecnica di compressione in cui un modello più piccolo (student) apprende da un modello più grande (teacher)



Knowledge distillation (2)

Dati non etichettati

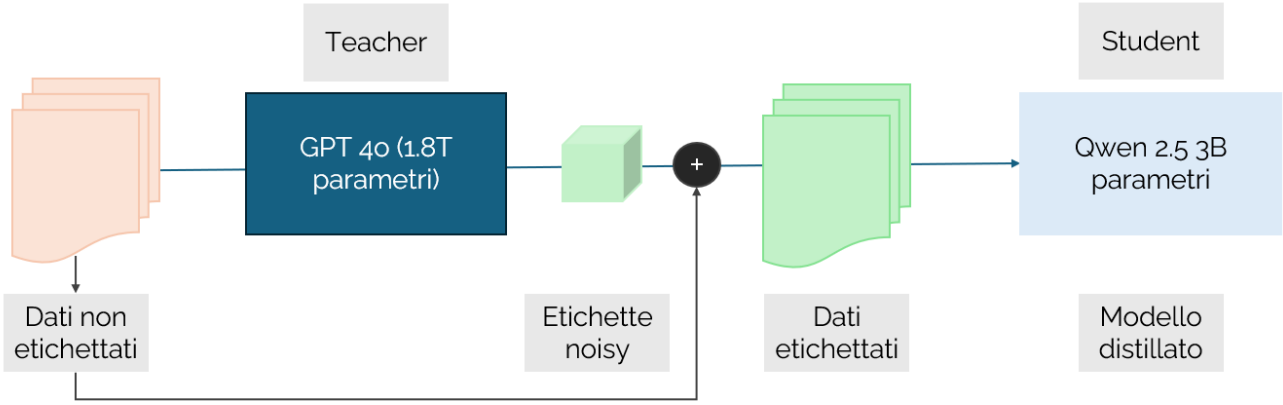
- Sono dati “grezzi” che non hanno risposte pronte (ad esempio frasi senza la continuazione).
- Normalmente non si potrebbe usare questi dati per allenare uno student, perché mancano le etichette



Knowledge distillation (3)

Dati non etichettati

- Sono dati “grezzi” che non hanno risposte pronte (ad esempio frasi senza la continuazione).
- Normalmente non si potrebbe usare questi dati per allenare uno student, perché mancano le etichette



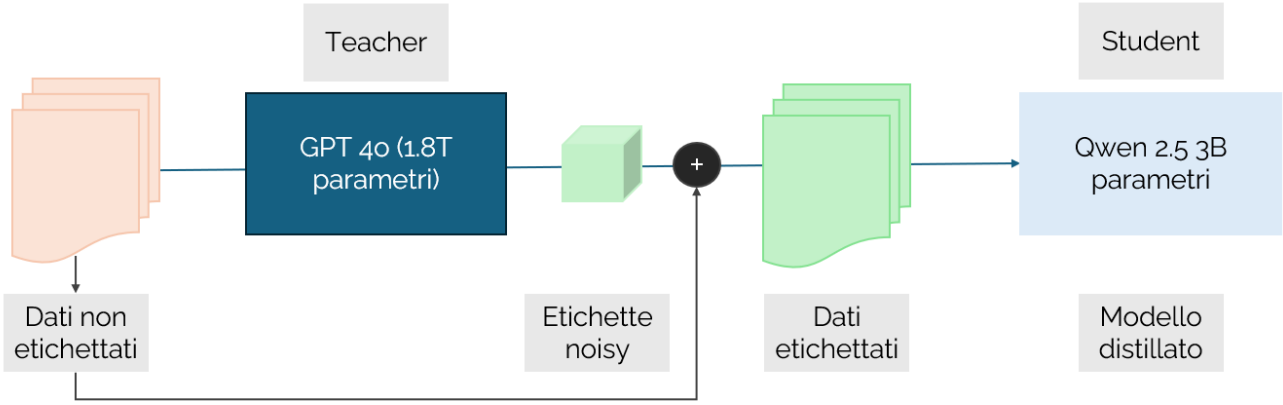
Teacher (es. GPT-4 con 1.8 trilioni di parametri)

- Prende questi dati grezzi e produce delle risposte (le “etichette”).
- Non sono perfette: possono essere noisy (cioè con errori), ma comunque utili.

Knowledge distillation (4)

Dati non etichettati

- Sono dati “grezzi” che non hanno risposte pronte (ad esempio frasi senza la continuazione).
- Normalmente non si potrebbe usare questi dati per allenare uno student, perché mancano le etichette



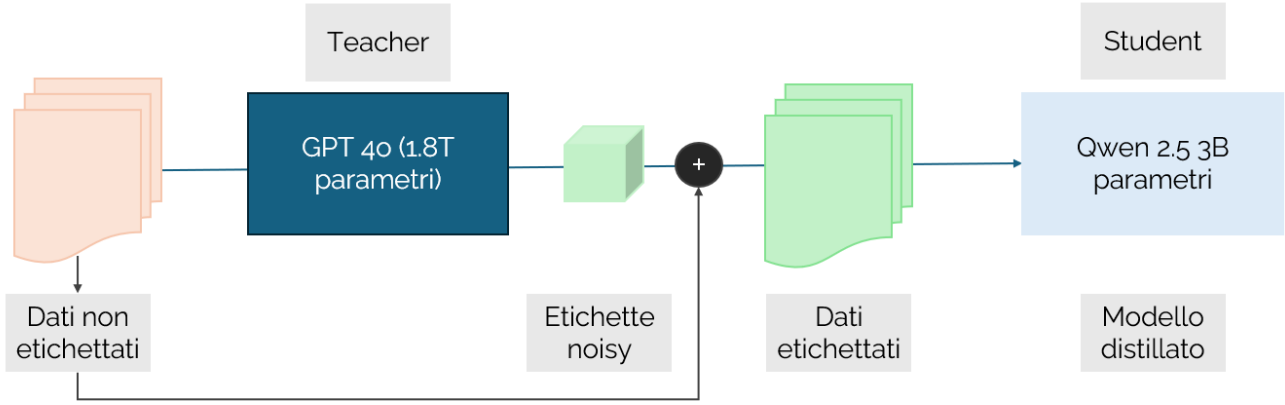
Teacher (es. GPT-4 con 1.8 trilioni di parametri)

- Prende questi dati grezzi e produce delle risposte (le “etichette”).
- Non sono perfette: possono essere noisy (cioè con errori), ma comunque utili.

Etichette noisy + Dati etichettati

- Si combinano le risposte generate dal teacher con eventuali dati già etichettati
- In questo modo si crea un nuovo dataset arricchito, pronto per l’allenamento.

Knowledge distillation (5)

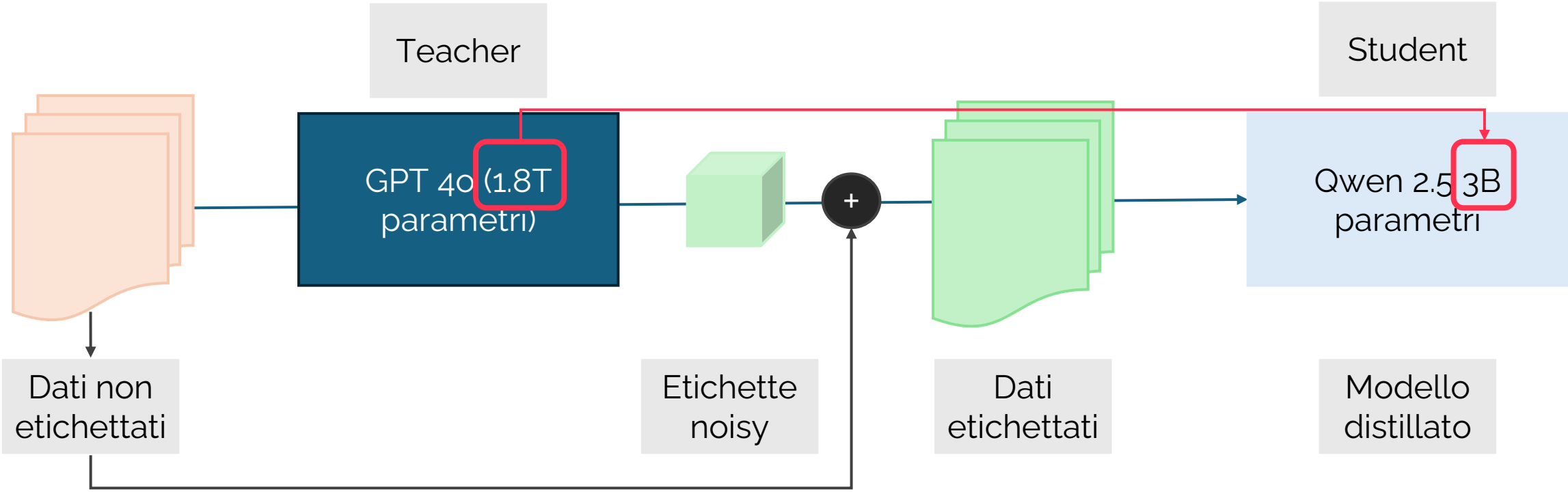


Student (es. Qwen 2.5, 3 miliardi di parametri)

- Viene addestrato su questo dataset “distillato”.
- È molto più piccolo del teacher, ma riesce a imparare da lui e a mantenere buona parte della sua capacità.

Knowledge distillation (6)

Tecnica di compressione in cui un modello più piccolo (student) apprende da un modello più grande (teacher)



Quantization (1)

La **quantizzazione** non cambia l'architettura del modello, ma **riduce la precisione dei numeri** usati nei calcoli.

- Un modello usa normalmente numeri a 32 bit (float32) per rappresentare i pesi e le attivazioni.
- Con la quantizzazione, questi numeri vengono "compressi" in formati più piccoli, ad esempio 16 bit o 8 bit.

- il modello occupa meno memoria,
- gira più veloce (soprattutto su hardware specializzato),
- ma può perdere un po' di accuratezza.

Quantization (2)

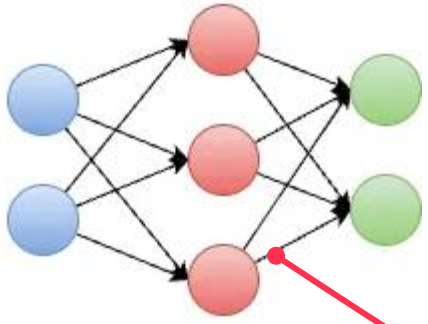
Distillazione = addestro un **nuovo modello più piccolo** (student) che imita il grande (teacher).

Quantizzazione = prendo lo stesso modello e riduco la precisione dei suoi numeri.

Quindi:

- La distillazione crea un nuovo modello.
- La quantizzazione ottimizza quello esistente

Quantization (3)

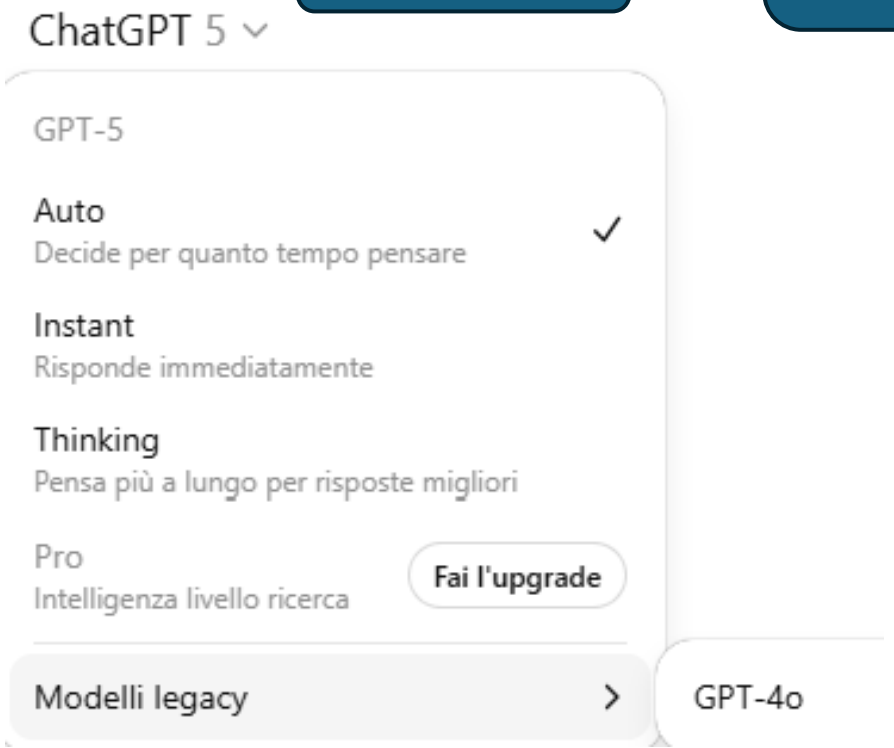


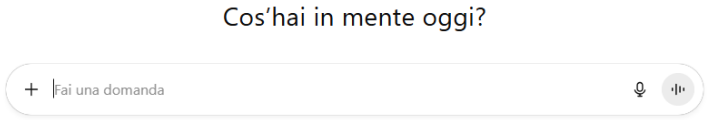
Valori dei pesi (manopole del mixer) della mia architettura

Floating Point				Integer		
1231.4531				1231		
32 bit				**8 bit**		
0.21	-0.37	-2.54		21	37	25
4.5	4.37	-0.78	**Quantization**	45	43	78
5.1	0.01	9.6		51	23	96

Prompting e sistemi (1)

modello → Il modello gira dietro l'interfaccia → Interfaccia, non si interagisce mai direttamente con il modello





Prompting e sistemi (2)

L'interfaccia

- Può applicare filtri di sicurezza (bloccare risposte non appropriate).
- Può aggiungere funzionalità extra (grafici, ricerche web, salvataggi in cloud).
- Può avere altri mini-modelli di controllo che verificano quello che il modello principale ha prodotto.

Server

Server (dove gira il modello)

- È la “macchina” fisica che esegue i calcoli del modello.
- Può stare in America, Europa, Cina, Italia… e questo cambia le regole:
  - Privacy e gestione dei dati (leggi diverse da paese a paese).
  - Velocità e costi (più il server è vicino, più è rapido).

ChatGPT di OpenAI:

- Modello = GPT-4 / GPT-5.
- Interfaccia = la pagina web/app che usiamo quotidianamente.
- Server = negli Stati Uniti.

DeepSeek:

- Modello = open source (chiunque può scaricarlo e farlo girare sul tuo computer (niente problemi di privacy, perché i dati restano da te) oppure usare l’interfaccia di un’altra azienda, che magari ha i server in Italia o in Europa.

Modelli open source

- Si conoscono i pesi del modello (cioè i numeri che rappresentano ciò che ha imparato).
- Di solito c'è anche un paper che spiega come è stato costruito.
- Non sempre sono pubblicati il codice di training e soprattutto i dati usati per addestrare.
- Si possono scaricare e far girare sul computer o server locale, anche in versione più leggera con la quantizzazione.

- Esempi: LLaMA, Qwen, DeepSeek.

- Vantaggi: controllo, privacy (i dati non escono dalla tua rete), possibilità di adattarli al caso di interesse.
- Svantaggi: bisogna avere server e personale tecnico per gestirli e mantenerli.

Modelli closed source

- Non rilasciano i pesi né il codice.
- Si possono usare solo tramite l'interfaccia dell'azienda che li ha creati.

- Esempio: ChatGPT di OpenAI.

- Vantaggi: scalabilità, non devi preoccuparti di manutenzione.
- Svantaggi: i dati passano sui server dell'azienda (es. USA) → problemi di privacy e compliance.

Application programming interface - API

- È un canale di accesso che un'azienda mette a disposizione per usare il modello.

- **API e modelli Closed**

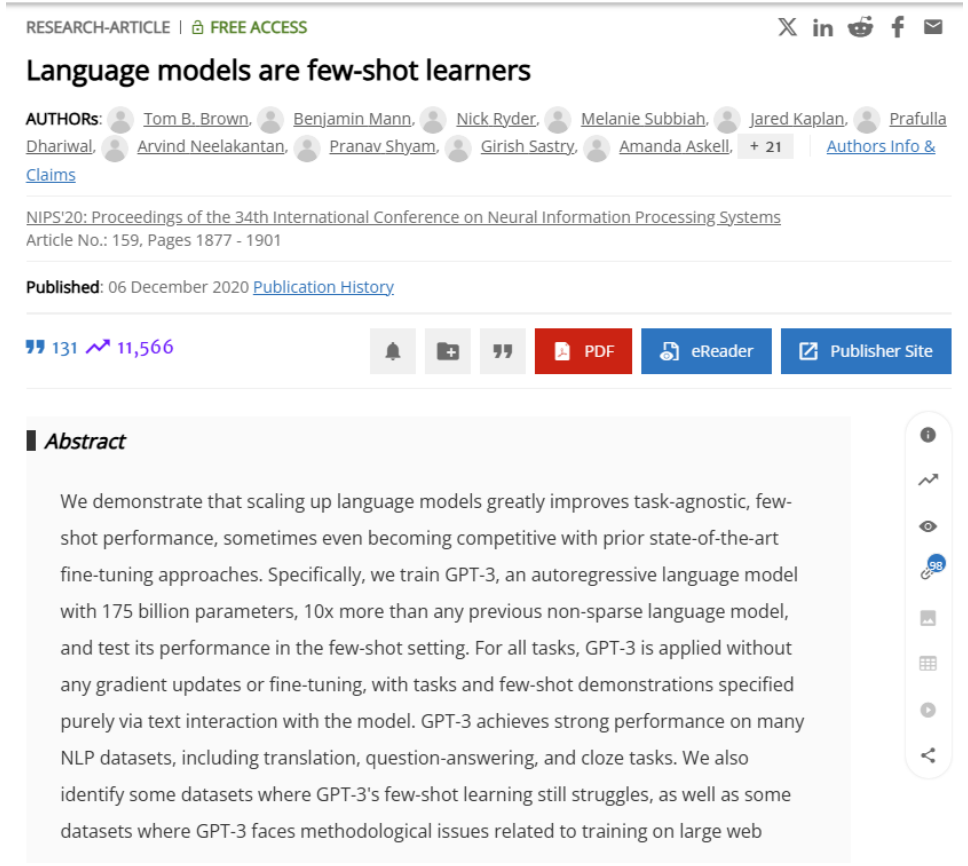
- Nei modelli closed (es. GPT di OpenAI),
- l'unico modo per usarli è via API o interfaccia web dell'azienda. Non si hanno mai i pesi del modello: si inviano i dati ai server dell'azienda che restituisce una risposta.
- È comodo e scalabile, ma i tuoi dati passano fuori dal tuo controllo.

- **API e modelli Open**

- Si possono scaricare LLaMA2/3 o DeepSeek (con i pesi dell'addestramento),
- Si installano su un server in Europa,
- Si crea un'API
  - se la licenza fornita dalle aziende produttrici del modello consente l'uso commerciale → si possono anche offrire le API pubbliche (es. "API basata su LLaMA 3 che gira su server in Europa").
  - se la licenza è solo per ricerca → si possono creare API interne per il laboratorio, ma non venderle o offrirle a clienti.

Prompting e in-context learning (ICL) (1)

- ICL è la capacità dei modelli linguistici (LLM) di imparare sul momento da ciò che scriviamo nel prompt.
- Il modello usa il testo che gli viene fornito come "contesto" per capire meglio cosa vogliamo e per adattarsi al compito, senza bisogno di riaddestramento.
- Prima del 2020 servivano dataset enormi e training lunghi per adattare un modello a un nuovo compito.
- Con ICL, invece, basta scrivere pochi esempi nel prompt.
- Questo approccio è stato descritto nel paper del 2020 "Language Models are ***Few-Shot Learners***" (che ha introdotto GPT-3).



RESEARCH-ARTICLE | FREE ACCESS

Language models are few-shot learners

AUTHORS: Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, + 21 Authors Info & Claims

NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems
Article No.: 159, Pages 1877 - 1901

Published: 06 December 2020 Publication History

131 11,566 PDF eReader Publisher Site

Abstract

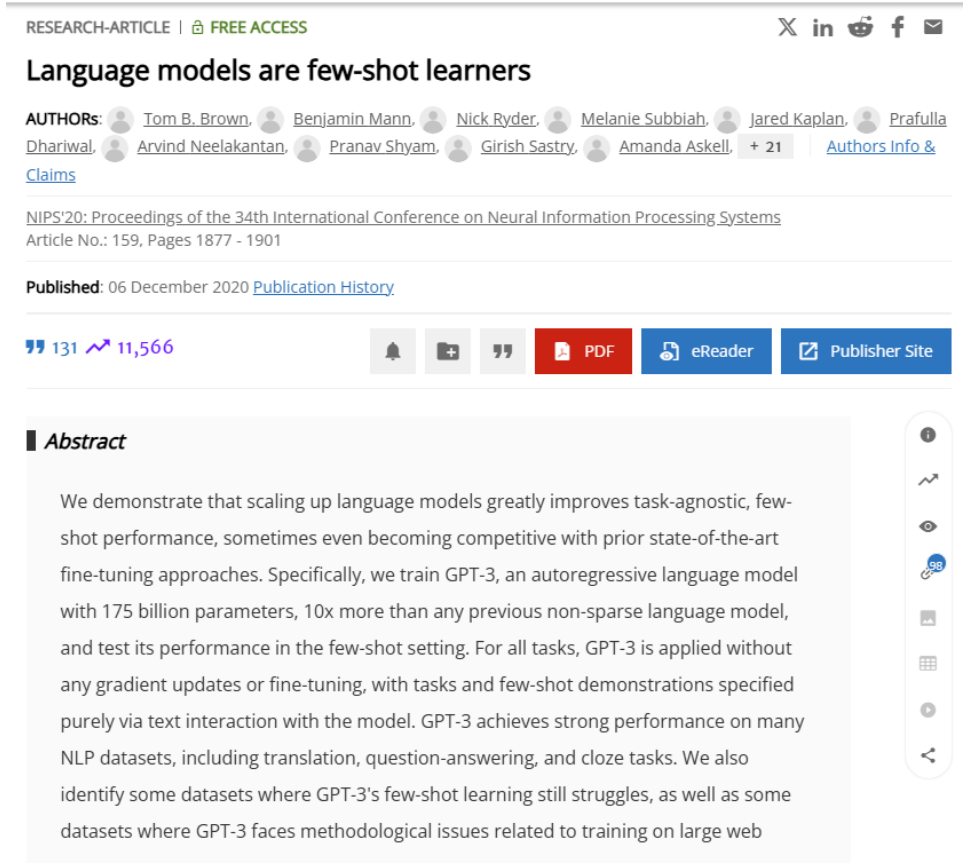
We demonstrate that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even becoming competitive with prior state-of-the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and cloze tasks. We also identify some datasets where GPT-3's few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web

Prompting e in-context learning (ICL) (2)

- ***Zero-shot learning)*** si dà solo l'istruzione → il modello prova a farlo.
  - Es: "Traduci in inglese: Buongiorno"

- ***One-shot learning)*** si dà un esempio ed il modello copia lo stile.
  - Es: "Traduci in inglese: Buongiorno → Good morning. Traduci in inglese: Buonanotte →"

- ***Few-shot learning)*** si danno più esempi → il modello generalizza meglio.
  - Es: dai 3-4 frasi tradotte e poi si chiede la traduzione di una nuova.



RESEARCH-ARTICLE | FREE ACCESS

Language models are few-shot learners

AUTHORS: Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, + 21 Authors Info & Claims

NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems
Article No.: 159, Pages 1877 - 1901

Published: 06 December 2020 Publication History

131 11,566 PDF eReader Publisher Site

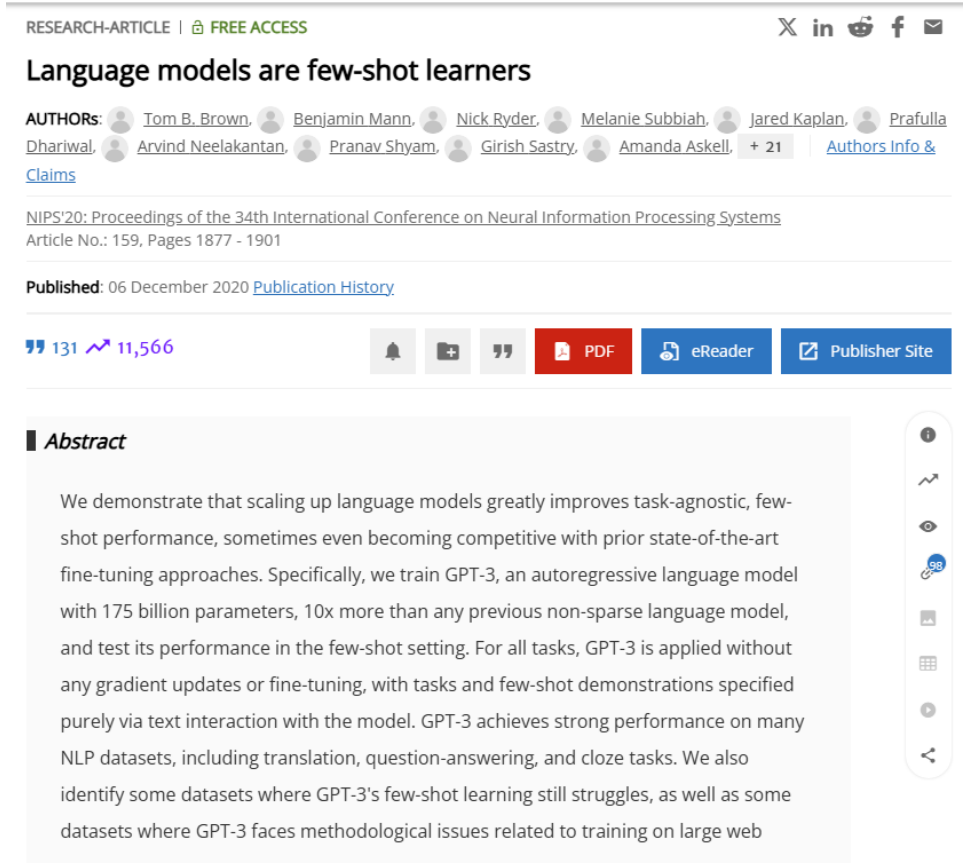
Abstract

We demonstrate that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even becoming competitive with prior state-of-the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and cloze tasks. We also identify some datasets where GPT-3's few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web

Prompting e in-context learning (ICL) (3)

- ***Perché funziona?***

Grazie al meccanismo dell'attenzione: Il modello legge tutto il prompt, dà più peso alle parti rilevanti e costruisce la risposta coerente con il contesto.



RESEARCH-ARTICLE | FREE ACCESS

Language models are few-shot learners

AUTHORS: Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, + 21 Authors Info & Claims

NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems
Article No.: 159, Pages 1877 - 1901

Published: 06 December 2020 Publication History

131 11,566 PDF eReader Publisher Site

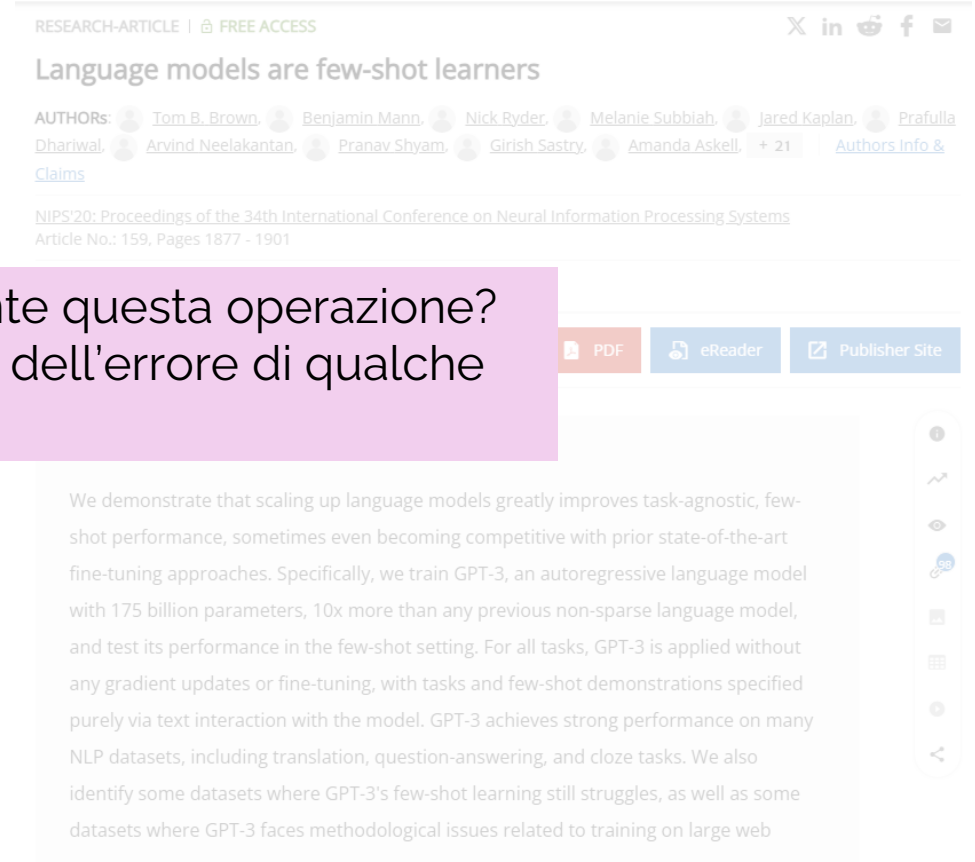
Abstract

We demonstrate that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even becoming competitive with prior state-of-the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and cloze tasks. We also identify some datasets where GPT-3's few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web

Prompting e in-context learning (ICL) (4)

- ***Zero-shot learning)*** si dà solo l'istruzione → il modello prova a farlo.
  - Es: "Traduci in inglese: Buongiorno"
- ***One-shot learning)*** si d
  copia lo stile.
  - Es: "Traduci in ingl
    morning. Traduci in inglese: Buonanotte →"
- ***Few-shot learning)*** si danno più esempi → il modello generalizza meglio.
  - Es: dai 3-4 frasi tradotte e poi si chiede la traduzione di una nuova.

Cambiano i pesi della rete durante questa operazione? Avviene una retropropagazione dell'errore di qualche tipo?



Prompting e in-context learning (ICL) (5)

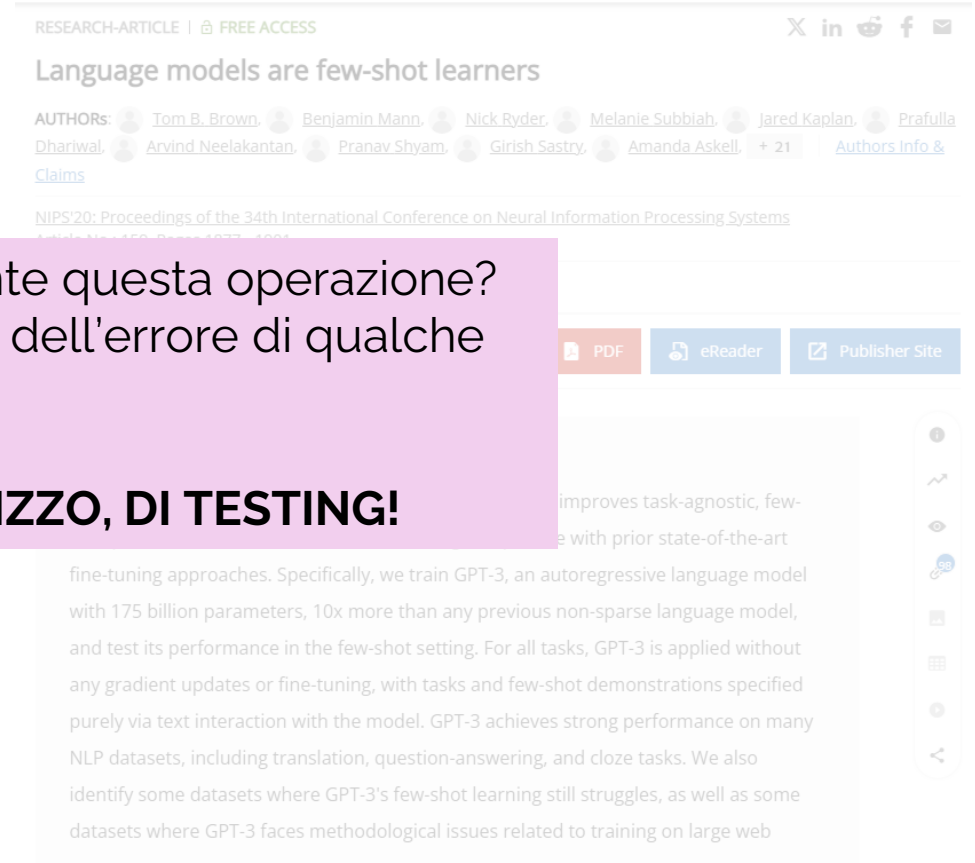
- ***Zero-shot learning)*** si dà solo l'istruzione → il modello prova a farlo.
  - Es: "Traduci in inglese: Buongiorno"

- ***One-shot learning)*** si d
copia lo stile.
  - Es: "Traduci in ingl
morning. Traduci i

- ***Few-shot learning)*** si danno più esempi → il modello generalizza meglio.
  - Es: dai 3-4 frasi tradotte e poi si chiede la traduzione di una nuova.

Cambiano i pesi della rete durante questa operazione? Avviene una retropropagazione dell'errore di qualche tipo?

NO SIAMO IN FASE DI UTILIZZO, DI TESTING!



Chain of thoughts (1)

GPT3 (Prima dei modelli «o»)

Prompt)

In caffetteria ci sono 23 mele. 20 le mangio per merenda poi ne acquistano altre 3 quante mele ci sono?

GPT-3) 0 mele

Chain of thoughts (2)

GPT3 (Prima dei modelli «o»)

Prompt)

In caffetteria ci sono 23 mele. 20 le mangio per merenda poi ne acquistano altre 3 quante mele ci sono?

GPT-3) 0 mele

Prompt)

In caffetteria ci sono 23 mele. 20 le mangio per merenda poi ne acquistano altre 3 quante mele ci sono? **Pensaci passo passo**

GPT-3)
Ci sono 23 mele iniziali. 20 mele vengono mangiate, 23 – 20 = 3
Altre 6 mele vengono acquistate, 6+3= 9 mele

COT

Chain of thoughts (3)

GPT3 (Prima dei modelli «o»)

Prompt)

In caffetteria ci sono 23 mele. 20 le mangio per merenda poi ne acquistano altre 3 quante mele ci sono? **Pensaci passo passo**

GPT-3)
(i) Ci sono 23 mele iniziali. (ii) 20 mele vengono mangiate (iii) Calcolo: 23 – 20 = 3
(iv) Altre 6 mele vengono acquistate (v) Calcolo: 6+3= 9 mele

COT

- Perché il COT funziona? Perché esiste il meccanismo dell'attenzione. Il modello che arriva al passo iv) non si è scordato dei numeri 23 e 20.
- L'attenzione permette al modello di «dare più peso» alle cifre

Chain of thoughts (4)

GPT3 (Prima dei modelli «o»)

Prompt)

In caffetteria ci sono 23 mele. 20 le mangio per merenda poi ne acquistano altre 3 quante mele ci sono? **Pensaci passo passo**

GPT-3)
(i) Ci sono 23 mele iniziali. (ii) 20 mele vengono mangiate (iii) Calcolo: 23 – 20 = 3
(iv) Altre 6 mele vengono acquistate (v) Calcolo: 6+3= 9 mele

COT

- Perché il COT funziona? Perché esiste il meccanismo dell'attenzione. Il modello che arriva al passo iv) non si è scordato dei numeri 23 e 20.
- L'attenzione permette al modello di «dare più peso» alle cifre

- Ecco come nascono i modelli di ragionamento! (es «GPT-40»)

Chain of thoughts (5)

Stato iniziale

Pre-training

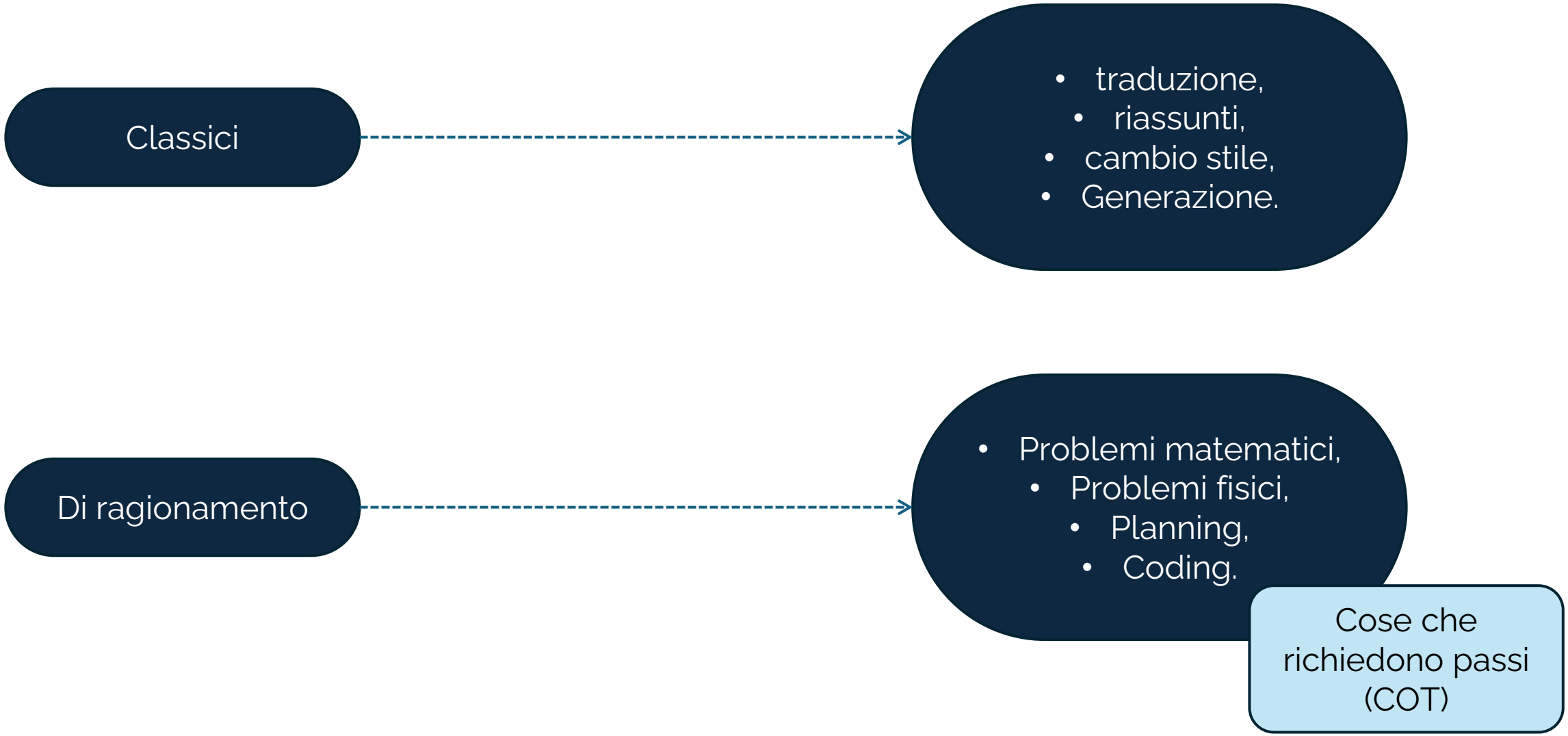
Supervised fine-tuning

Alignment

La COT viene insegnata in fase di Alignment.

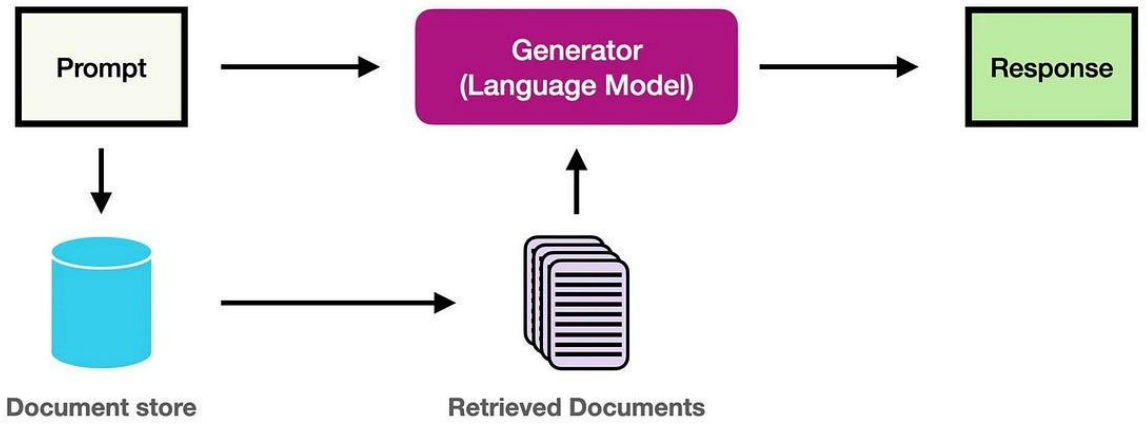
Al modello vengono dati tanti esempi di COT!

Modelli classici vs di ragionamento



Retrival augmented generation (RAG) (1)

- Un RAG non è un nuovo modello, ma un'estensione del concetto di in-context learning.
- In pratica: invece di far affidamento solo sulla memoria del modello (che può essere limitata o datata), il RAG recupera informazioni fresche e pertinenti da una base di conoscenza esterna (knowledge base, documenti, database, ecc.) e le "inietta" nel prompt prima di generare la risposta.



Retrival augmented generation (RAG) (2)

Input dell'utente → lo studente/utente fa una domanda (prompt).

Esempio: "Quali sono i tassi di interesse del mio conto?

Retrieval (recupero) → il sistema prende questa domanda e cerca nei documenti della knowledge base le parti rilevanti. La ricerca avviene di solito usando tecniche di similarità semantica (es. embeddings).

Augmentation (arricchimento del prompt) → **i pezzi di testo trovati vengono inseriti nel prompt, accanto alla domanda dell'utente.**

Quindi il modello non lavora solo con la domanda, ma anche con informazioni aggiuntive e specifiche.

Generation (risposta) → l'LLM (tipo GPT) elabora il prompt arricchito e genera la risposta finale, usando sia la sua conoscenza interna sia le informazioni recuperate.

Retrival augmented generation (RAG) (3)

- Senza RAG: L'utente chiede a GPT: "Qual è la politica di rimborso della mia banca?"
- GPT potrebbe inventare (allucinare) o dare una risposta generica.
- Con RAG: Il sistema cerca nella knowledge base della banca i documenti che contengono la sezione "Politica di rimborso".
- Recupera il testo: "I clienti hanno diritto al rimborso entro 14 giorni lavorativi dalla richiesta scritta".
- Inserisce questo testo nel prompt accanto alla domanda.
- GPT risponde in modo preciso: "La tua banca prevede che i clienti possano ottenere un rimborso entro 14 giorni lavorativi dalla richiesta scritta."

Modelli agentici

- Modelli agentici) Modelli capaci di interagire con il mondo
- Alla base c'è sempre la stessa logica degli LLM: predizione di token uno dopo l'altro.
- La novità è che alcuni token vengono interpretati come azioni invece che come testo normale.
- Esempi di azioni)
  - Aprire una pagina web → token speciale che dice "apri URL"
  - Cliccare un bottone → token che dice "clic su X"
  - Usare una calcolatrice → token che dice "calcola(45+46)"
  - Chiamare un'API → token che dice "call API meteo con parametro=Roma"

Il modello non "fa i calcoli" o "naviga" da solo, ma produce il comando testuale che un sistema esterno ***interpreta e traduce in un'azione reale.***

Recap LLM base vs RAG vs Modelli agentici

LLM base: riceve input → genera testo.

RAG: riceve input → cerca info → integra nel prompt → genera testo.

Agente: riceve input → genera testo + possibili azioni → azione eseguita → genera testo finale.

Benchmark LLM: Quanto è bravo il mio LLM?

https://livebench.ai/#/

Il benchmarking è un processo di confronto sistematico delle performance di diversi modelli di intelligenza artificiale (o di uno stesso modello in versioni diverse).

Un benchmark è la verifica stessa: cioè l'insieme dei test usati per valutare i modelli. Nel caso degli LLM, un benchmark è formato di solito da:

- Training set → i dati usati per addestrare il modello (le "prove di esercitazione" fatte in classe).
- Test set (o evaluation set) → i dati usati per valutare il modello (l'"esame vero"), che il modello non deve aver visto prima, altrimenti la valutazione non è valida.

Benchmark LLM (2)

https://livebench.ai/#/

Come funziona il benchmarking di un LLM

- Si sceglie un benchmark (es. un insieme di domande e risposte).
- Si dà il test al modello: l'LLM deve rispondere senza sapere le soluzioni.
- Si confrontano le risposte dell'LLM con quelle corrette.
- Si calcola la percentuale di successo: quante risposte corrette / totale delle domande.

Benchmark LLM (3)

DROP (reading comprehension + aritmetica)

Il modello deve leggere un testo e rispondere a domande che richiedono capire il contenuto e fare piccoli calcoli.

Es. testo) Nel censimento della città di Springfield risultano 8.000 abitanti, di cui 2.000 non sono cittadini statunitensi.

Domanda)
Qual è la percentuale di non cittadini statunitensi a Springfield?

MMLU (Massive Multitask Language Understanding)

Il modello deve rispondere a domande simili a quelle di un quiz universitario, in ambiti molto diversi (medicina, legge, storia, matematica…).

Es. domanda) Quale vitamina è carente nello scorbuto?

A) Vitamina A
B) B) Vitamina C
C) C) Vitamina D
D) D) Vitamina B1

Risposta corretta: B) Vitamina C

HellaSwag (ragionamento su scenari complessi, buon senso)

Es. domanda) Quale comportamento è moralmente più sbagliato?

A) Parlare al telefono con la mamma durante il turno di lavoro.
B) Aiutare un amico a scappare di prigione.

Risposte/a corrette/a)

- A) e B) (entrambe sono moralmente scorrette).
- Oppure B) (la seconda risposta è considerata più scorretta della prima)

Benchmark LLM (4)

HellaSwag (ragionamento su scenari complessi, buon senso)

Dato "ctx_a“ il modello deve scegliere uno dei 4 “endings” in fase di testing

https://rowanzellers.com/hellaswag/


```
{
 "ind": 14,
 "activity_label": "Wakeboarding",
 "ctx_a": "A man is being pulled on a water ski as he floats in the water casually.",
 "ctx_b": "he",
 "ctx": "A man is being pulled on a water ski as he floats in the water casually. he",
 "split": "test",
 "split_type": "indomain",
 "endings": [
  "mounts the water ski and tears through the water at fast speeds.",
  "goes over several speeds, trying to stay upright.",
  "struggles a little bit as he talks about it.",
  "is seated in a boat with three other people."
 ],
 "source_id": "activitynet~v_-5KAycAQlC4"
}
```

Data Leaking

Leaking tra training e test set

Succede quando dati identici (o troppo simili) finiscono sia nel training set che nel test set.

Il modello li “ricorda” e ottiene un punteggio gonfiato.

Esempio) Nel test set c’è la frase “Roma è la capitale d’Italia”.

La stessa frase è finita anche nel training perciò **il modello non dimostra vera comprensione**, ma solo memoria.

- Fa sembrare il modello più bravo di quanto sia
- Porta a scelte sbagliate in deployment, perché le performance crollano quando il modello affronta dati reali.
- È difficile da scoprire, perché a volte il leakage è sottile (variabili correlate o dati duplicati non ovvi)

Chatbot Arena (1)

Altro modo per valutare LLM con “human-in-the-loop”

- Due chatbot (basati su LLM diversi) vengono messi “uno contro l’altro” e ricevono la stessa domanda da parte dell’utente.
- L’utente vede solo le risposte, senza sapere quale modello le ha generate.
- Dopo averle lette, l’utente vota quale preferisce (o dichiara un pareggio).
- I risultati vengono raccolti in modo anonimo e aggregato: più voti significano un ranking più solido dei modelli.

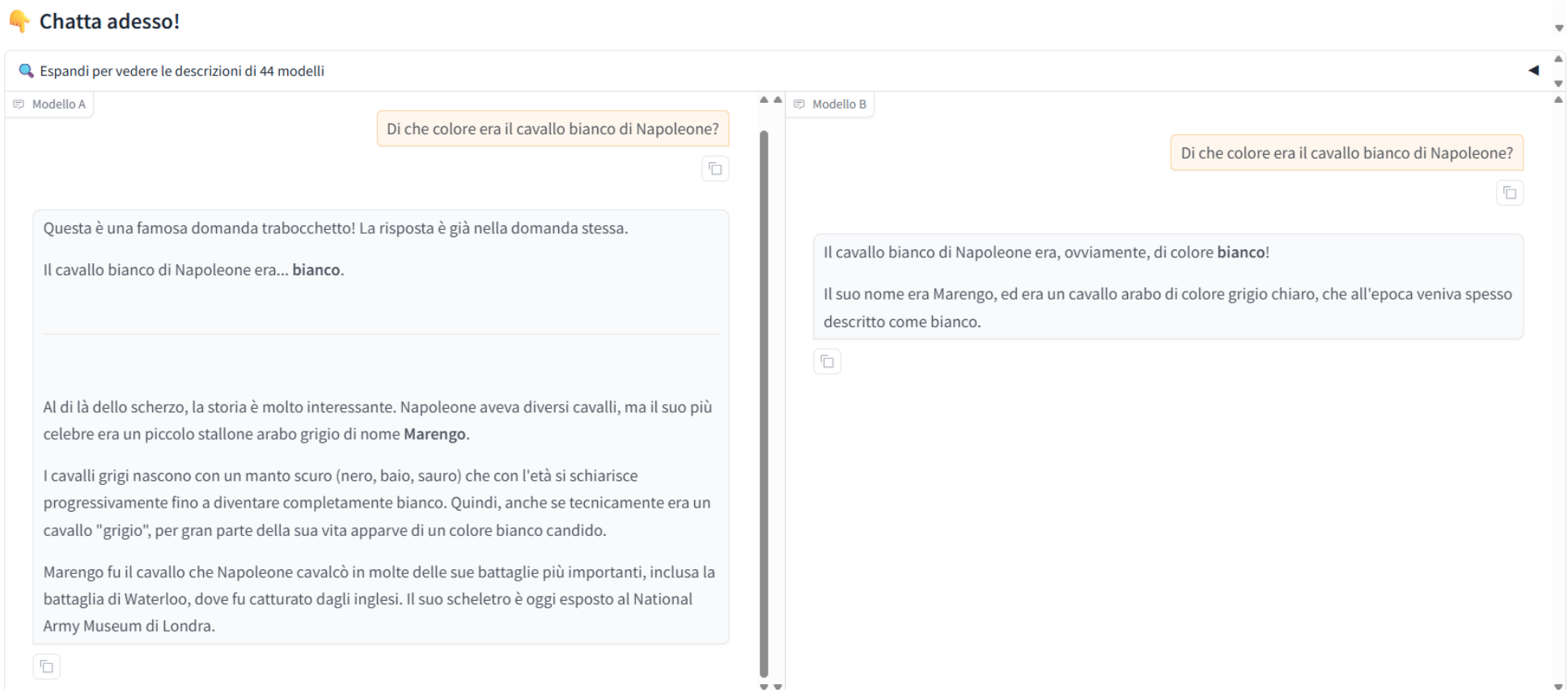
È una modalità di benchmarking “dal basso”: non si usano solo metriche automatiche, ma il giudizio umano diretto per valutare la qualità delle risposte (chiarezza, correttezza, utilità).

Chatbot Arena (2)

Altro

- Du
  do
- L'u
- Do
- I ri
  de

È u



solido

udizio

Safety – LLM (Caso studio OpenAI),quanto è sicuro il mio sistema basato LLM? (1)

Preparedness Framework

Il Preparedness Framework è il modo con cui OpenAI si organizza per prevenire rischi e gestire in sicurezza i modelli di intelligenza artificiale man mano che diventano più potenti.

L'idea è che, così come per nuove tecnologie (farmaci, aerei, centrali nucleari…), serve una valutazione continua dei rischi anche per i modelli di AI. Quindi l'obiettivo del framework è: Garantire che, prima di rilasciare un nuovo modello o una nuova capacità, siano stati:

- valutati i **rischi potenziali**,
- messi in atto dei **controlli di sicurezza**,
- stabilite **procedure di monitoraggio**

https://cdn.openai.com/pdf/18a02b5d-6b67-4cec-ab64-68cdfbddebcd/preparedness-framework-v2.pdf

Safety – LLM (Caso studio OpenAI),quanto è sicuro il mio sistema basato LLM? (2)

4 dimensioni chiave

Rischi per la sicurezza	Possibilità che il modello venga usato per produrre contenuti dannosi (istruzioni su come costruire armi)
Rischi sociali	Impatti sulla società, come la diffusione di disinformazione o l'aumento di disuguaglianze.
Robustezza e affidabilità	Capacità del modello di rispondere in modo corretto, riducendo errori e allucinazioni.
Allineamento agli scopi umani	Quanto il modello segue le intenzioni dell'utente in modo sicuro, senza "deragliare" in comportamenti imprevisti

Safety – LLM (Caso studio OpenAI),quanto è sicuro il mio sistema basato LLM? (2)

Meccanismo valutativo. Ogni nuova capacità del modello (ogni nuovo rilascio) viene:

- Testata con scenari realistici e stress test (simulazioni di uso improprio).
- Classificata per livello di rischio (da basso a critico).
- Confrontata con soglie predefinite: se il rischio supera una soglia, il modello non può essere rilasciato senza ulteriori mitigazioni.