

# Appendice: Introduzione al Deep Learning 🤖🧠

## Un Tuffo nel Cuore dell'Intelligenza Artificiale Moderna

# Indice dell'Appendice




1. Cos'è il Deep Learning?
2. Perché è diventato così popolare?
3. Concetti di Base: Neuroni, Strati, Funzioni di Attivazione
4. Reti Neurali Feedforward (FNN)
5. Addestramento: Funzione di Perdita, Backpropagation, Ottimizzatori
6. Architetture Comuni (CNN, RNN - cenni)
7. Deep Learning per l'NLP (cenni)
8. Strumenti e Framework Popolari
9. Considerazioni Etiche e Limitazioni

# 1. Cos'è il Deep Learning?

- Branca del **Machine Learning (ML)**, che è un sottocampo dell'**Intelligenza Artificiale (IA)**.
- Utilizza **Reti Neurali Artificiali** con **molteplici strati** (da cui "profondo").
- Capace di apprendere **rappresentazioni gerarchiche dei dati** automaticamente da dati grezzi.
  - *Esempio immagini*: bordi -> forme -> oggetti.

## 2. Perché il Deep Learning è diventato così popolare?

Convergenza di tre fattori chiave:

1.  **Grandi quantità di dati (Big Data):** Essenziali per addestrare modelli complessi.
2.  **Potenza computazionale (GPU):** Ha reso l'addestramento di modelli grandi fattibile.
3.  **Miglioramenti algoritmici:** Nuove architetture, funzioni di attivazione, tecniche di ottimizzazione.

### 3. Concetti di Base: Neuroni Artificiali

- Unità fondamentale: **neurone artificiale** (o nodo).
- Riceve input, esegue una **somma pesata**, aggiunge un **bias**.
- Passa il risultato attraverso una **funzione di attivazione**.

### 3. Concetti di Base: Strati (Layers)

I neuroni sono organizzati in strati:

- **Strato di Input (Input Layer):** Riceve i dati grezzi.
- **Strati Nascosti (Hidden Layers):**  
Elaborazioni intermedie. Più strati nascosti = rete "profonda".
- **Strato di Output (Output Layer):** Produce il risultato finale.

### 3. Concetti di Base: Funzioni di Attivazione

Introducono la **non-linearità**, permettendo di apprendere relazioni complesse.

- **Sigmoide:**  $\sigma(x) = 1 / (1 + e^{(-x)})$  (output tra 0 e 1)  
(Nota: Immagine illustrativa non disponibile al momento)
- **ReLU (Rectified Linear Unit):**  $\text{ReLU}(x) = \max(0, x)$  (efficiente, mitiga vanishing gradient)
- **Tanh:** Output tra -1 e 1
- **Softmax:** Per classificazione multi-classe (output come distribuzione di probabilità)

## 4. Reti Neurali Feedforward (FNN / MLP)

- Tipo più semplice di rete neurale artificiale.
- Informazione si muove in **una sola direzione**: input -> hidden -> output.
- Nessun ciclo o connessione all'indietro (in inferenza).
- Anche note come **Multi-Layer Perceptrons (MLP)**.



## 5. Addestramento: Funzione di Perdita (Loss Function)

- Misura quanto le previsioni del modello si discostano dai valori reali (target).
- Obiettivo dell'addestramento: **minimizzare la funzione di perdita**.

Esempi:

- **Mean Squared Error (MSE)**: Per problemi di regressione (prevedere un numero).

$$\text{MSE} = (1/n) * \sum (y_{\text{true}} - y_{\text{pred}})^2$$

- **Cross-Entropy Loss**: Per problemi di classificazione (prevedere una categoria).

## 5. Addestramento: Backpropagation e Discesa del Gradiente

- **Backpropagation:** Algoritmo per calcolare il **gradiente** della funzione di perdita rispetto a ogni peso/bias.
  - Il gradiente indica la direzione per ridurre la perdita.
- **Discesa del Gradiente (Gradient Descent):** Algoritmo di ottimizzazione che aggiorna iterativamente i pesi/bias usando i gradienti.
  - **Learning Rate:** Dimensione dei passi durante l'aggiornamento.

*(Nota: Immagine illustrativa per Backpropagation/Discesa del Gradiente non disponibile al momento)*


## 5. Addestramento: Ottimizzatori (Optimizers)

Varianti della discesa del gradiente per un addestramento più veloce e stabile.

- **SGD (Stochastic Gradient Descent)**: Usa un singolo esempio (o un piccolo batch) alla volta.
- **Adam (Adaptive Moment Estimation)**: Adatta il learning rate per ciascun parametro. Molto popolare ed efficace.
- Altri: AdaGrad, RMSProp, Adadelta.

## 6. Architetture Comuni (Cenni)

Oltre alle FNN, architetture specializzate:

-  **Reti Neurali Convoluzionali (CNN):**
  - Efficaci per dati a griglia (es. immagini).
  - Usano strati convoluzionali (filtri) per pattern locali.
  - In NLP: classificazione testi (pattern locali di parole).

-  **Reti Neurali Ricorrenti (RNN):**
  - Per dati sequenziali (testi, serie temporali).
  - Connessioni cicliche (memoria).
  - Varianti: LSTM, GRU (per dipendenze a lungo termine).
  - !

## 7. Deep Learning per l'NLP (Cenni)

Il Deep Learning ha rivoluzionato l'NLP:

- **Word Embeddings:** Word2Vec, GloVe, FastText (reti neurali per rappresentazioni vettoriali di parole).
- **Modelli Sequence-to-Sequence (Seq2Seq):** Encoder-Decoder per traduzione, riassunto.
- **Architettura Transformer:** Self-attention, standard per NLP avanzato (BERT, GPT).

## 8. Strumenti e Framework Popolari

- **TensorFlow (Google)**: Ecosistema completo, con Keras (API di alto livello).
- **PyTorch (Meta)**: Flessibile,