



# ICT Training Center

**Il tuo partner per la Formazione e la Trasformazione digitale della tua azienda**



## Note



# SPRING AI

## GENERATIVE ARTIFICIAL INTELLIGENCE CON JAVA

# Simone Scannapieco

## Corso base per Venis S.p.A, Venezia, Italia

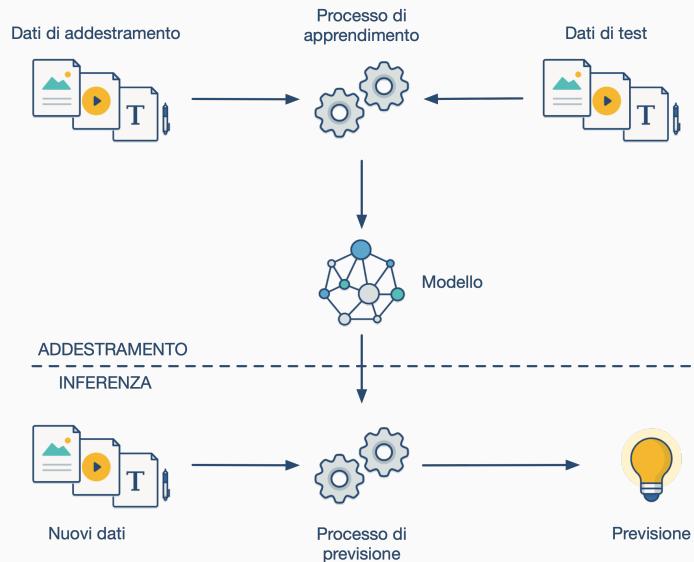
Settembre 2025

## Note

# GENERATIVE AI E NLP

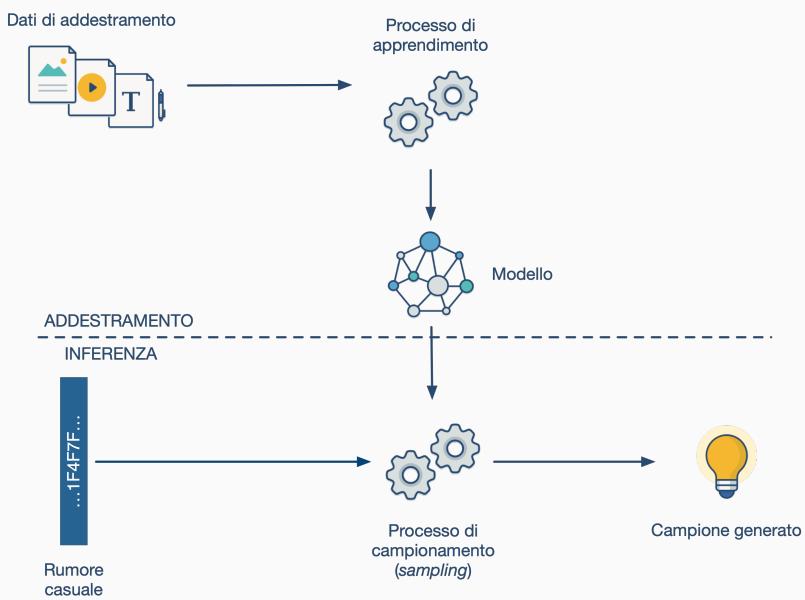
## Note

## MACHINE LEARNING SCHEMA DI WORKFLOW SEMPLIFICATO



©Simone Scannapieco

## Note



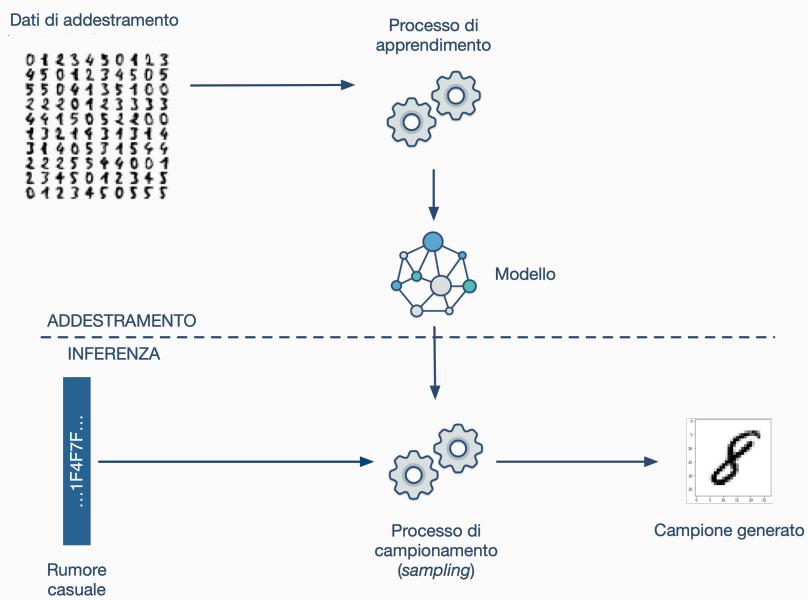
©Simone Scannapieco

## ➔ Addestramento automatico prevalentemente **non supervisionato**

## Note

# MODELLI GENERATIVI

## SCHEMA DI WORKFLOW SEMPLIFICATO - ESEMPIO MNIST

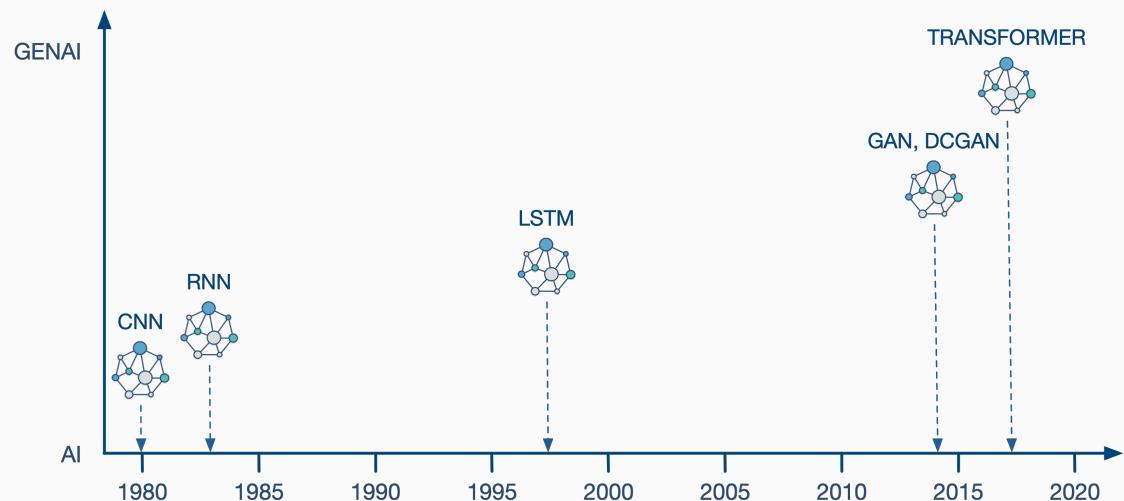


©Simone Scannapieco, Nischal Madiraju

## Note

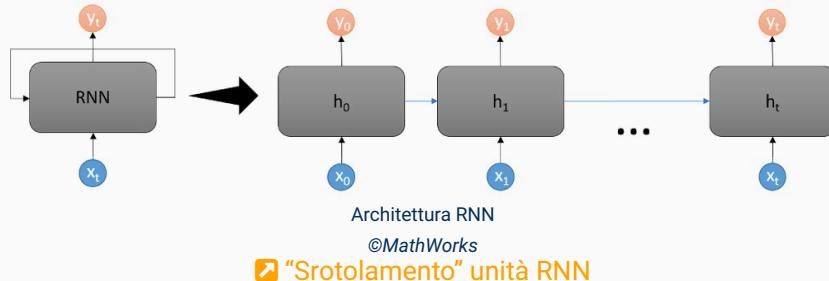
- Definita la nuova era della AI
  - Tre ambiti fondamentali
    - 1 Generazione di testi
    - 2 Generazione di codice
    - 3 Generazione di contenuti multimodali (*diffusion*)
  - Ci focalizziamo su 1

## Note



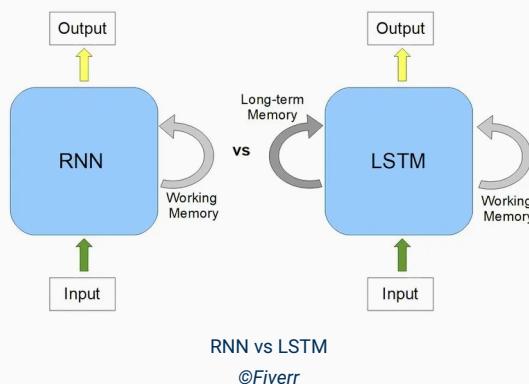
©Simone Scannapieco

## Note



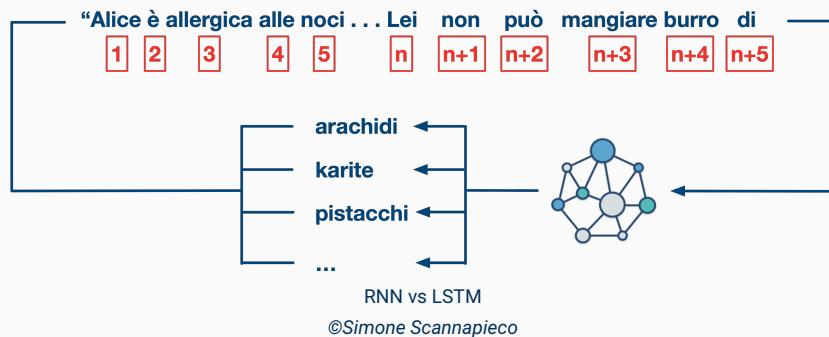
- Ottimizzate per task su dati ordinati/sequenziali
    - Predizione di serie temporali, NLP, speech recognition, video processing
  - Tiene traccia degli *input* precedenti per influenzare l'*input* e l'*output* attuali
  - Scardinano il principio di indipendenza tra *layer*
  - Condividono parametri a livello di *layer* (come CNN)
    - **Pro:** potenziale abbattimento dei tempi di addestramento/grandezza del modello addestrato
    - **Contro:** algoritmo di *backpropagation* specifico, nel tempo (BPTT)
      - Errore deve essere inteso come la somma degli errori fatti fino a quel momento
      - **Gradiente fantasma:** possibilità che addestramento smetta di imparare
      - **Gradiente che esplode:** possibilità di creare modelli instabili, con pesi troppo grandi

## Note



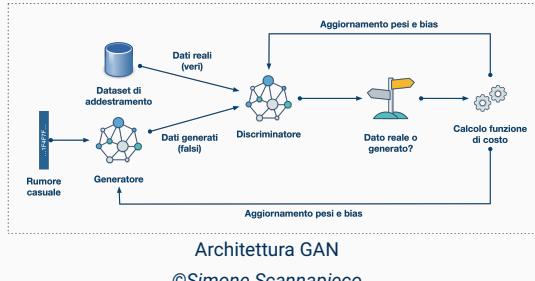
- ➔ Varianti più avanzate delle RNN
    - ➔ La memoria a breve termine può durare migliaia di passaggi (da cui il termine *long*)
    - ➔ Gli *hidden layer* usano unità a tre porte (*in/out/forget*)
    - ➔ Il *forget* permette di tralasciare elementi reputati non più importanti per la decisione dell'*output* corrente

## Note



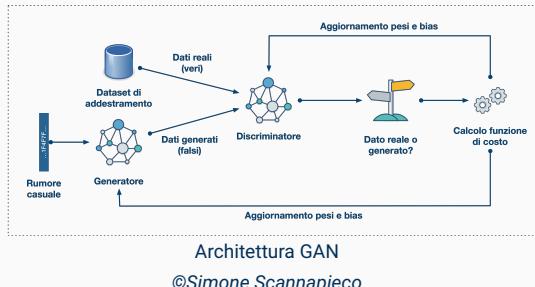
- ➡ Cosa succede se  $n \approx 5$ ?
    - ➡ CNN ed LSTM si comportano in maniera simile
  - ➡ Cosa succede se  $n \gg 5$ ?
    - ➡ LSTM probabilmente avrà probabilità alte in corrispondenza della frutta secca...
    - ➡ ...RNN assicurerebbe una crisi anafilattica ad Alice!

## Note



- Precursore di *Deep Convolutional GAN* – DCGAN (2016)
  - Prima rivoluzione nel modo in cui si creano contenuti artificiali
  - Due reti neurali che competono
    - Generatore
      - **Obiettivo:** migliorare la qualità dei contenuti che produce, ingannando il discriminatore
      - Crea un campione di dati
    - Discriminatore
      - **Obiettivo:** migliorare l'abilità di discriminare dati reali da quelli prodotti
      - Decide se il campione è generato oppure preso dal campione reale (classificazione binaria)

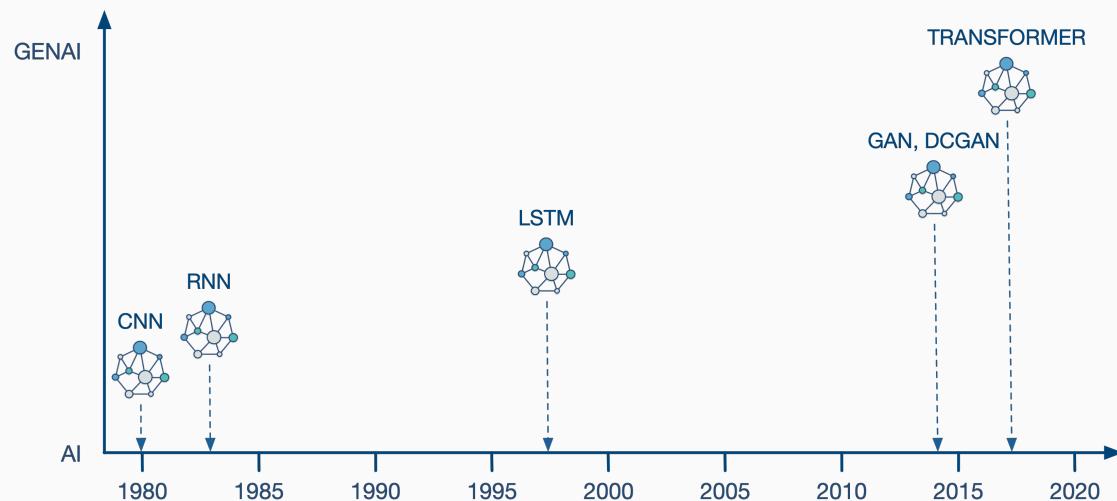
## Note



- 1 Addestrare il discriminatore sui dati reali (*dataset* di addestramento)
  - 2 Generare i dati falsi per il discriminatore
  - 3 Addestrare il discriminatore sui dati falsi
  - 4 Addestrare il generatore con l'*output* del discriminatore

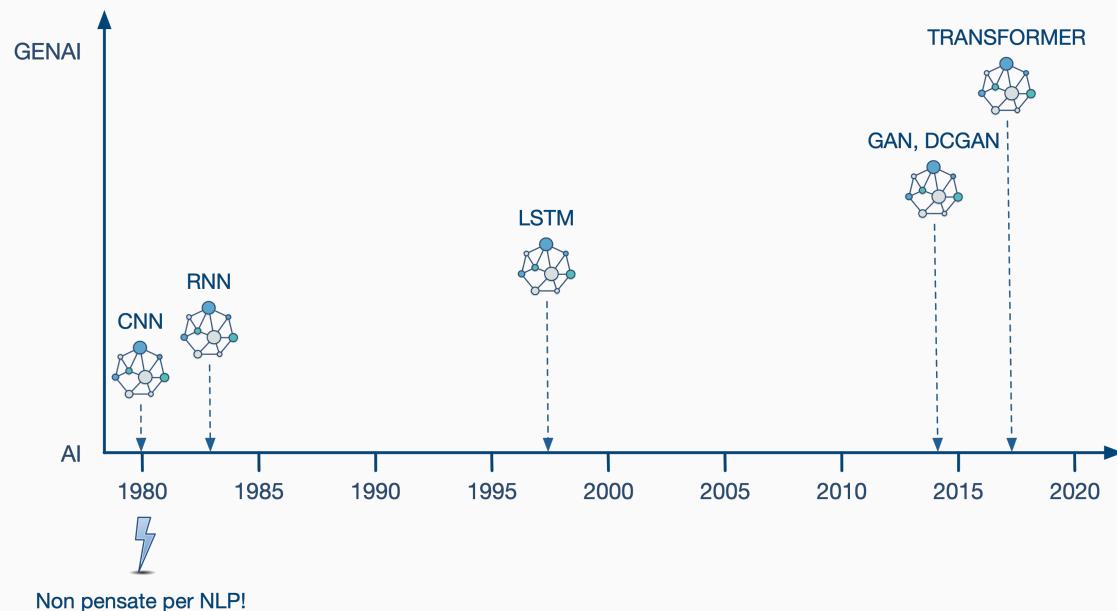
➔ ... giusto per farvi capire la complessità del processo...

## Note



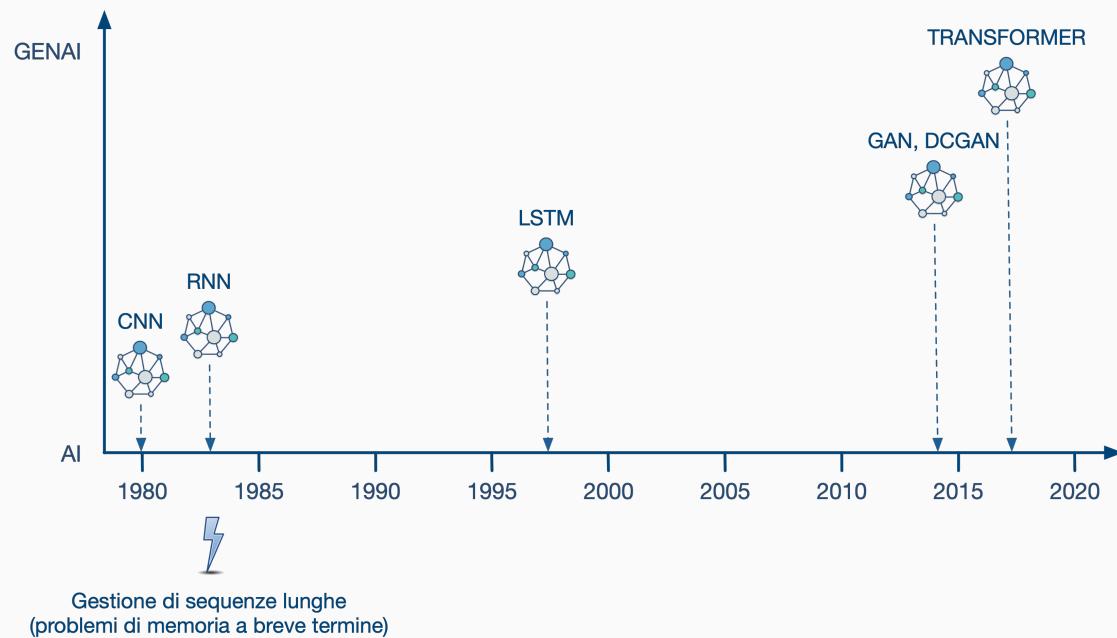
©Simone Scannapieco

## Note



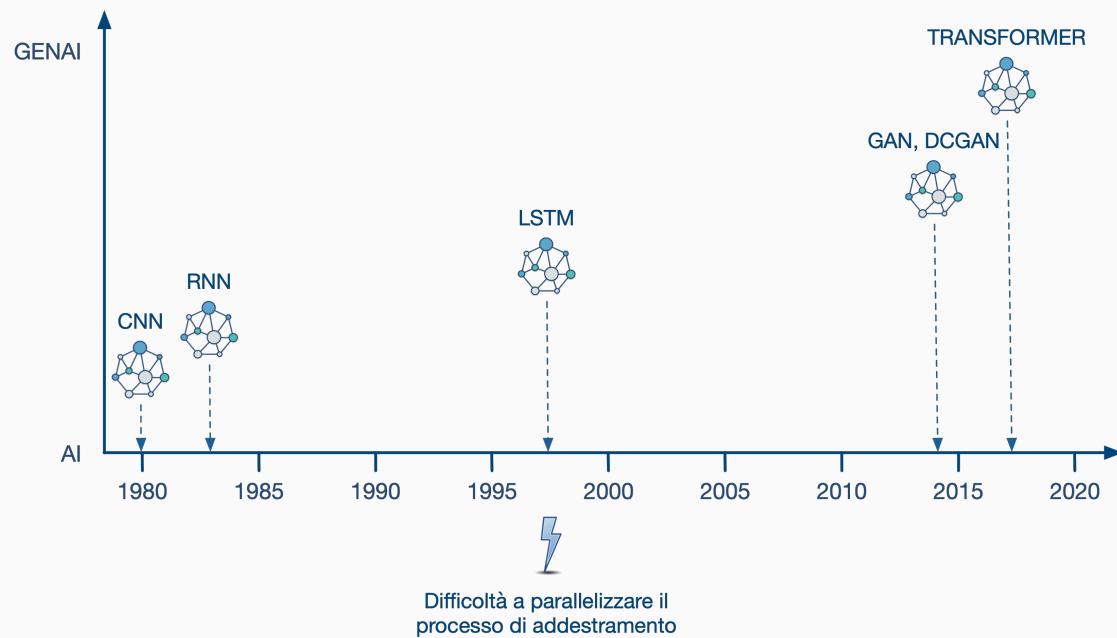
©Simone Scannapieco

## Note



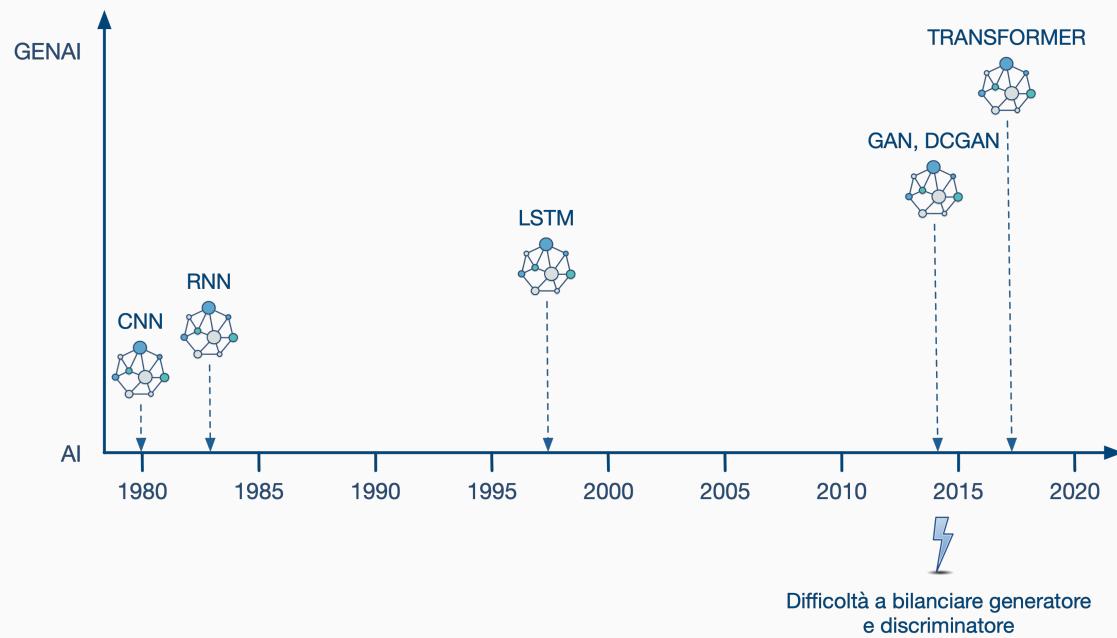
©Simone Scannapieco

## Note



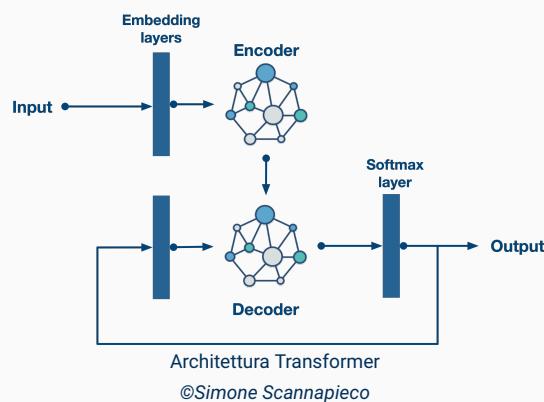
©Simone Scannapieco

## Note



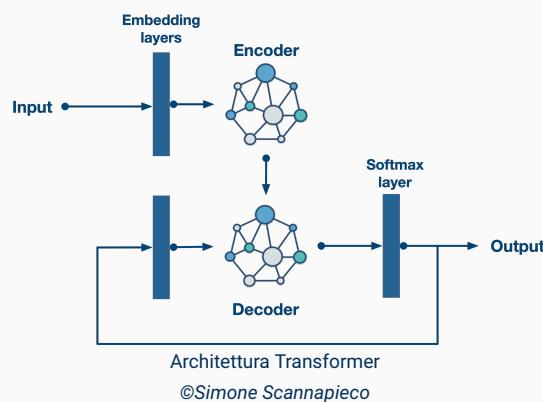
©Simone Scannapieco

## Note



- ➡ Architettura rivoluzionaria rispetto a RNN ed LSTM
    - ➡ *Encoder*
      - ➡ **Obiettivo:** elaborare l'*input* e generare rappresentazioni che catturano il suo significato
    - ➡ *Decoder*
      - ➡ **Obiettivo:** ottenere le rappresentazioni dell'encoder come **contesto** per generare l'*output* (es. un testo)
    - ➡ *Softmax layer*
      - ➡ **Obiettivo:** restituire la scelta del *decoder* reputata da lui più probabile

## Note



- Prime strutture basate su Encoder-Decoder soffrivano di un problema di *bottleneck*
    - Ultimo *layer* nascosto dell'encoder doveva modellare **tutto** il contesto utilizzato dal decoder
    - Decoder in stallo finché l'encoder non finiva il suo lavoro
  - **Attention** come soluzione al problema
    - Decoder ottiene informazione sul contesto da ciascun *layer* nascosto dell'encoder
    - Gestione del contesto più efficace e persistente nel tempo
    - Gestione dell'addestramento ad alto grado di parallelizzazione. . .
    - . . . quindi, su dataset di addestramento **estremamente grandi**

## Note