

# Modelli Linguistici e Sequence-to-Sequence





Comprendere, Predire e Generare  
Linguaggio Naturale

# Indice dei contenuti

- Introduzione ai modelli linguistici
- Cos'è un modello linguistico
- Modelli N-gram
- Modelli linguistici neurali
- Il framework Sequence-to-Sequence
- Il meccanismo di attenzione
- L'architettura Transformer
- Subword Segmentation
- Applicazioni pratiche
- Sfide etiche e considerazioni pratiche

# Introduzione ai modelli linguistici

I modelli linguistici (Language Models o LM):

- Sistemi **probabilistici** che assegnano probabilità a sequenze di parole
- Predicono quale parola è più probabile che segua una determinata sequenza
- Alla base di numerose applicazioni NLP quotidiane:
  -  Correzione automatica sui nostri smartphone
  -  Assistenti virtuali come Siri, Alexa, Google Assistant
  -  Traduzione automatica
  -  Generazione di testo

## L'evoluzione dei modelli linguistici 🕒

Periodo	Approccio	Caratteristiche
Anni '80-'00	<b>Modelli statistici (N-gram)</b>	Basati su conteggi, context locale
Anni '10	<b>Reti neurali ricorrenti (RNN)</b>	Rappresentazioni dense, memoria limitata
Anni '10-'15	<b>LSTM e GRU</b>	Migliore gestione dipendenze a lungo termine
2017-oggi	<b>Transformer</b>	Basati su attenzione, altamente parallelizzabili
2018-oggi	<b>Modelli pre-addestrati</b>	Transfer learning, miliardi di parametri

 Quanto è importante predire la parola successiva?

Completa la frase: "Il cielo oggi è particolarmente \_\_\_\_\_"






# Cos'è un modello linguistico

Un modello probabilistico che assegna una probabilità a una sequenza di parole

- **Framework left-to-right:**

$$P(w_1, w_2, \dots, w_n) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times \dots \times P(w_n|w_1, \dots, w_{n-1})$$

- **Applicazioni fondamentali:**

- Completamento predittivo 
- Correzione ortografica e grammaticale 
- Generazione di testo 
- Valutazione della fluidità del linguaggio 
- Disambiguazione 

# Valutazione dei modelli linguistici: Perplexity 📏

La perplexity misura quanto il modello è "sorpreso" da un testo di test:

$$Perplexity = 2^{-\frac{1}{N} \sum_{i=1}^N \log_2 P(w_i | w_1, \dots, w_{i-1})}$$

- **Interpretazione:** Se un modello ha perplexity 100, è come se stesse scegliendo uniformemente tra 100 possibili parole ad ogni passo
- **Obiettivo:** Perplexity più bassa = modello migliore

# Caso Aziendale: Settore Editoriale

## Applicazioni:

- Assistenza alla scrittura con suggerimenti contestuali
- Controllo automatico di leggibilità e tono
- Generazione di titoli ottimizzati
- Adattamento stilistico per diversi pubblici

## Esempio reale:

The Associated Press utilizza modelli linguistici per automatizzare la produzione di report finanziari e sportivi di base, generando migliaia di articoli trimestrali sui risultati aziendali e resoconti di partite.



# Modelli N-gram

Il mattone fondamentale dei modelli linguistici statistici

# Principio di funzionamento degli N-gram

Approccio probabilistico basato sull'**assunzione markoviana**: il futuro dipende dal passato solo attraverso il presente

- **Unigram (n=1)**:  $P(w_i)$  - Solo frequenza individuale delle parole
- **Bigram (n=2)**:  $P(w_i|w_{i-1})$  - Solo la parola precedente
- **Trigram (n=3)**:  $P(w_i|w_{i-2}, w_{i-1})$  - Due parole precedenti

**Stima delle probabilità**: conteggio delle frequenze relative

$$P(w_i|w_{i-n+1}, \dots, w_{i-1}) \approx \frac{\text{count}(w_{i-n+1}, \dots, w_i)}{\text{count}(w_{i-n+1}, \dots, w_{i-1})}$$

# Il problema della sparsità

Anche con corpora enormi, molte sequenze valide non appariranno mai!

## Tecniche di smoothing:

- **Add-one (Laplace):** Aggiunge 1 a tutti i conteggi
- **Good-Turing:** Riserva probabilità per eventi mai visti
- **Kneser-Ney:** Considera la diversità dei contesti
- **Interpolazione:** Combina modelli di diversi ordini
- **Backoff:** "Retrocede" a modelli di ordine inferiore quando necessario

# Vantaggi e limitazioni dei modelli N-gram

## Vantaggi:

- Semplicità concettuale
- Efficienza computazionale
- Interpretabilità diretta
- Buone performance in domini specifici

## Limitazioni:

- Incapacità di catturare dipendenze a lungo termine
- Crescita esponenziale dei parametri con  $n$
- Difficoltà con parole rare
- Nessuna nozione di similarità semantica

# Modelli linguistici neurali

Superano i modelli N-gram grazie a:

## 1. Rappresentazioni dense (word embeddings)

- Parole simili hanno vettori simili
- Dimensionalità ridotta

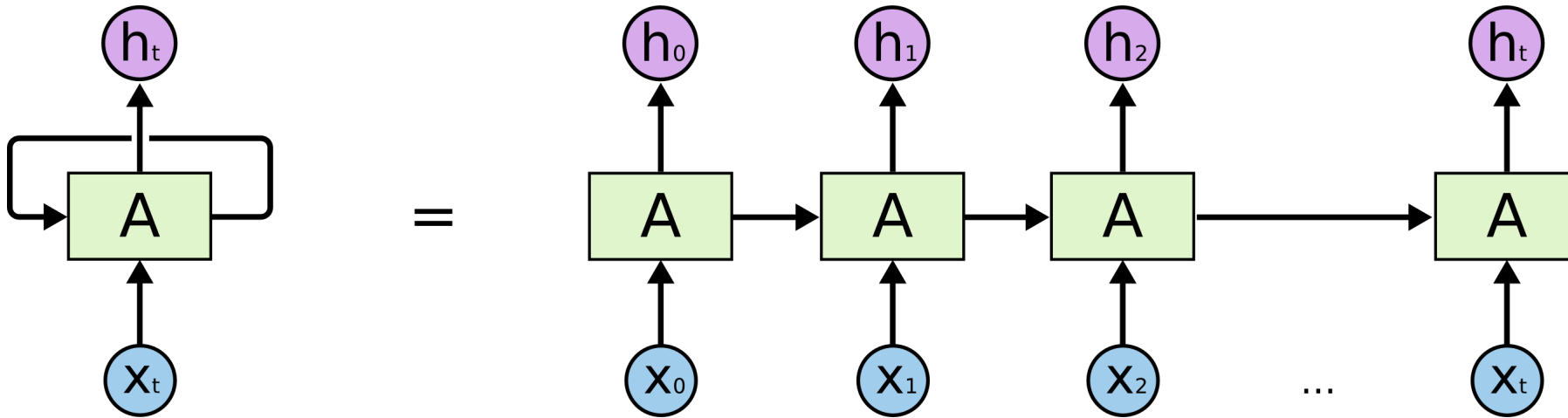
## 2. Reti neurali potenti

- Apprendimento automatico di feature
- Capacità di modellare relazioni complesse

## 3. Generalizzazione migliore

- Performance superiori su sequenze mai viste
- Comprensione di relazioni semantiche

## Modelli basati su reti neurali ricorrenti (RNN)



Le RNN processano le sequenze elemento per elemento, mantenendo uno **stato nascosto** che viene aggiornato ad ogni passo.

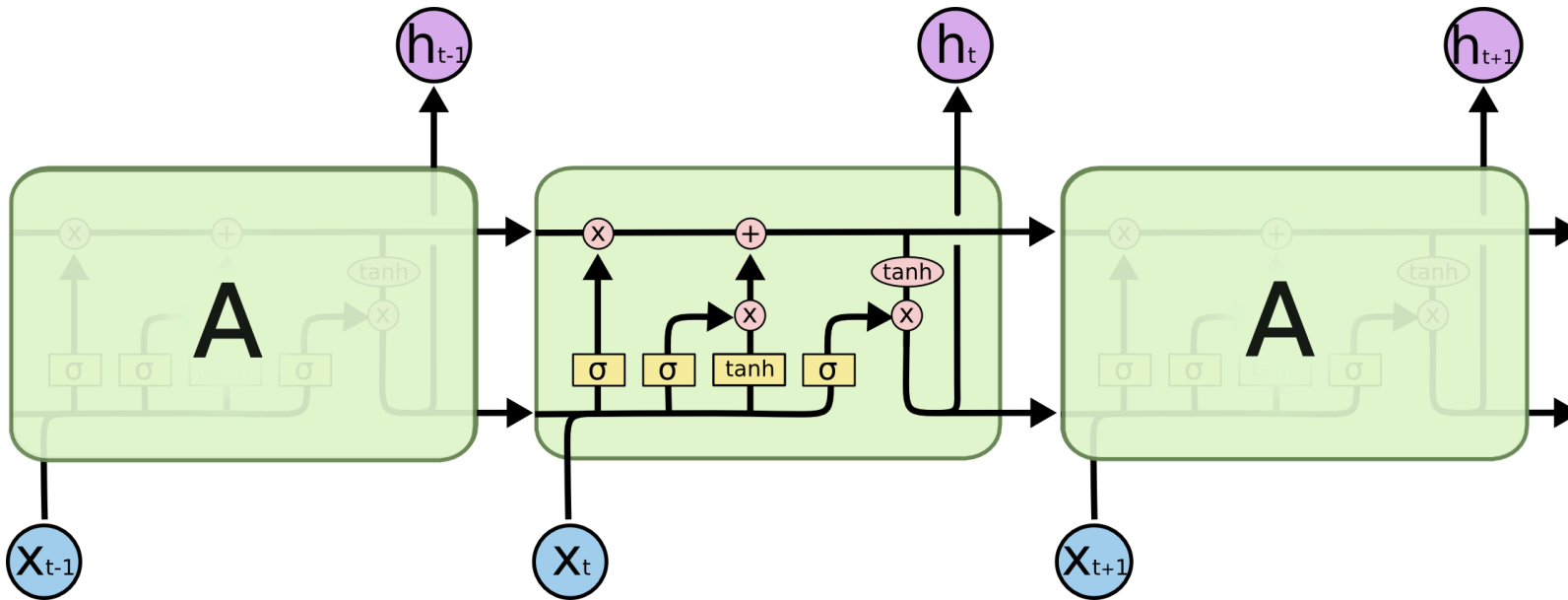
$$h_t = \tanh(W_{hx} \cdot x_t + W_{hh} \cdot h_{t-1} + b_h)$$

$$y_t = \text{softmax}(W_y \cdot h_t + b_y)$$

# Il problema del vanishing gradient e le soluzioni

Le RNN standard hanno difficoltà a catturare dipendenze a lungo termine a causa del **vanishing gradient**.

Soluzioni: LSTM e GRU



- **Long Short-Term Memory (LSTM)**: Celle di memoria con gate di input, forget e output
- **Gated Recurrent Unit (GRU)**: Versione semplificata con gate di reset e update

# Modelli bidirezionali e CNN per il testo

## Modelli bidirezionali

- Due RNN separate:
  - Una da sinistra a destra
  - Una da destra a sinistra
- Combinano informazioni da entrambe le direzioni

## Reti neurali convoluzionali (CNN)

- Applicano filtri su finestre di parole
- Catturano pattern locali (simili a n-gram)
- Vantaggi:
  - Parallelizzabilità
  - Efficienza computazionale



# Weight tying e altre ottimizzazioni

**Weight tying:** Condivide i pesi tra lo strato di embedding e lo strato di output

- Riduce significativamente il numero di parametri
- Migliora la generalizzazione

**Altre ottimizzazioni:**

- **Adaptive softmax:** Gerarchia di classificatori per vocabolari grandi
- **Sampled softmax:** Approssimazione durante l'addestramento
- **Mixture of Softmaxes (MoS):** Combinazione di multiple distribuzioni

# Caso Aziendale: Settore Finanziario

## Applicazioni:

- Analisi automatica di report finanziari
- Identificazione di segnali predittivi di mercato
- Monitoraggio del sentiment in tempo reale
- Generazione di sintesi e report automatizzati

## Esempio reale:

Bloomberg utilizza modelli linguistici avanzati per analizzare migliaia di notizie finanziarie al minuto, identificando eventi rilevanti per il mercato e fornendo insights in tempo reale ai trader.

# Il framework Sequence-to-Sequence

Trasformare una sequenza in un'altra sequenza

# Architettura Encoder-Decoder

## Encoder

- Processa la sequenza di input
- Tipicamente una RNN bidirezionale
- Produce una rappresentazione che cattura le informazioni rilevanti

## Decoder

- Modello linguistico condizionato sulla rappresentazione dell'encoder
- Opera in modo auto-regressivo:
  - i. Inizia con token <start>
  - ii. Predice parola per parola
  - iii. Termina con token <end>

## Sfide dell'architettura Seq2Seq base

- **Collo di bottiglia dell'informazione:** Comprimere tutto in un singolo vettore
- **Perdita di informazioni:** Dettagli dall'inizio tendono a "svanire"
- **Disallineamento:** Difficoltà con relazioni non monotone tra input e output
- **Esposizione al bias:** Discrepanza tra addestramento (input corretti) e inferenza (input predetti)

### Tecniche di inferenza:

- **Greedy decoding:** Seleziona sempre la parola più probabile
- **Beam search:** Mantiene le k sequenze più probabili
- **Sampling:** Campiona dalla distribuzione (top-k, nucleus/top-p, temperature)

# Valutazione dei modelli Seq2Seq

La valutazione varia a seconda del compito specifico:

- **Traduzione automatica:**
  - BLEU, METEOR, TER - confrontano con riferimenti umani
- **Riassunto automatico:**
  - ROUGE - misura sovrapposizione di n-gram con riferimenti
- **Generazione di dialogo:**
  - Perplexity, diversità lessicale, valutazione umana
- **Valutazione umana:**
  - Fondamentale per aspetti qualitativi (naturalezza, coerenza, utilità)

## Come comunichereste un messaggio complesso?

Immaginate di dover tradurre una frase lunga e complessa in una lingua straniera, parola per parola...

Sarebbe meglio:

- A) Memorizzare l'intera frase e poi tradurla
- B) Guardare a parti specifiche della frase mentre traducete

# Il meccanismo di attenzione

**Motivazione:** Permettere al decoder di "focalizzarsi" selettivamente su parti dell'input durante la generazione di ciascuna parola

## **Vantaggi:**

- Elimina il collo di bottiglia dell'informazione
- Permette di gestire sequenze lunghe
- Facilita l'apprendimento di allineamenti complessi
- Fornisce interpretabilità



# Tipi di attenzione

## Per copertura

- **Globale:** Considera tutte le parole
- **Locale:** Solo una finestra di parole

## Per meccanismo

- **Soft:** Pesi continui, differenziabile
- **Hard:** Selezione discreta

## Per interazione

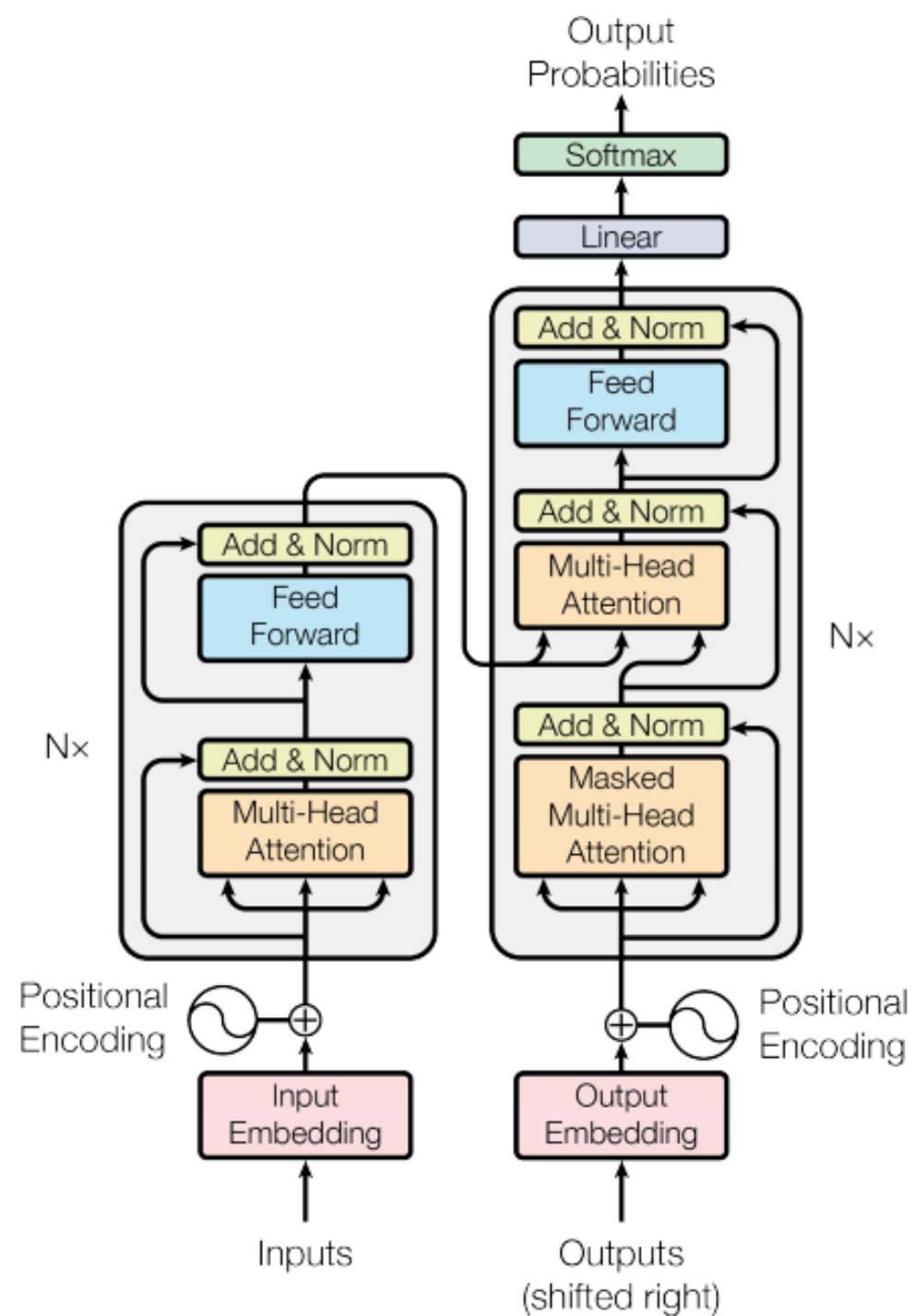
- **Self-Attention:** Sequenza interagisce con se stessa
- **Cross-Attention:** Interazione tra sequenze diverse

## Per complessità

- **Single-Head:** Un unico meccanismo
- **Multi-Head:** Attenzione in parallelo con diverse proiezioni

# L'architettura Transformer 🤖

"Attention is All You Need" (Vaswani et al., 2017)



# Principi fondamentali del Transformer

## Innovazioni chiave:

- **Parallelizzazione completa:** Elabora l'intera sequenza contemporaneamente
- **Self-attention:** Ogni posizione interagisce con tutte le altre
- **Rappresentazioni posizionali:** Informazioni esplicite sulla posizione
- **Architettura encoder-decoder:** Mantiene la struttura generale Seq2Seq

## Vantaggi rispetto a RNN/CNN:

- Cattura efficacemente dipendenze a lungo termine
- Addestramento molto più veloce
- Path length costante tra qualsiasi coppia di posizioni
- Maggiore scalabilità

## Immaginate di essere un Transformer!

Come processereste questa frase?

"Il gatto che spavenò il topo, che il formaggio attirò, è nero"

**Pensate all'attenzione tra parole** - Questa frase è un esempio di costruzione sintattica annidativa complessa

# Subword Segmentation

La segmentazione a livello di subword affronta il problema del **vocabolario aperto**:

- **Il problema delle parole fuori vocabolario (OOV):**
  - Vocabolari enormi per lingue morfologicamente ricche
  - Inefficienza nella gestione di forme flesse
  - Neologismi e nomi propri non gestibili
- **Principio della segmentazione subword:**
  - Dividere le parole in unità più piccole e frequenti
  - Catturare componenti morfologici significativi
  - Permettere di rappresentare qualsiasi parola come sequenza di subword

# Algoritmi principali di segmentazione

## Byte Pair Encoding (BPE)

- Inizia con caratteri singoli
- Unisce iterativamente le coppie più frequenti
- Usato in GPT, RoBERTa

## WordPiece

- Simile a BPE ma diverso criterio di selezione
- Massimizza la verosimiglianza del corpus
- Usato in BERT

## SentencePiece

- Tratta il testo come sequenza di Unicode bytes
- Gestisce qualsiasi lingua senza pre-tokenizzazione
- Ideale per modelli multilingue

**\*\*Esempio di segmentazione\*\*:** "inimmaginabile" → "in + immagin + abile"

# Impatto sui modelli linguistici e Seq2Seq

La segmentazione subword ha trasformato i modelli NLP moderni:

- **Vocabolari più compatti:** 30-50K token invece di centinaia di migliaia
- **Eliminazione del problema OOV:** Qualsiasi parola può essere rappresentata
- **Migliore generalizzazione morfologica:** Riconoscimento di pattern anche in parole rare
- **Efficacia multilingue:** Condivisione di subword tra lingue con radici comuni
- **Gestione di neologismi e nomi propri:** Scomposizione in componenti familiari

# Architettura dei Transformers

Panoramica delle Varianti e dei loro Utilizzi



Architettura	A cosa serve meglio (tipi di task)	Esempi di modelli celebri	Punti di forza	Limiti tipici
Encoder-only	<ul style="list-style-type: none"> <li>- Classificazione di testo (sentiment, topic, ecc.)</li> <li>- Named-Entity Recognition (NER)</li> <li>- Feature extraction / embedding per retrieval o clustering</li> <li>- Similarità semantica / semantic search</li> </ul>	BERT, RoBERTa, ALBERT, DistilBERT, ELECTRA	<ul style="list-style-type: none"> <li>• Cattura finemente le relazioni nel contesto bidirezionale</li> <li>• Ottimo per "capire" un testo e produrre un vettore o un'etichetta</li> </ul>	<ul style="list-style-type: none"> <li>• Non genera testo autonomamente</li> <li>• Sequence length spesso limitata</li> </ul>
	<ul style="list-style-type: none"> <li>- Language modeling autoregressivo</li> <li>- Generazione di</li> </ul>		<ul style="list-style-type: none"> <li>• Eccellente fluidità</li> </ul>	<ul style="list-style-type: none"> <li>• Non "vede" il futuro della</li> </ul>

Architettura	A cosa serve meglio (tipi di task)	Esempi di modelli celebri	Punti di forza	Limiti tipici
Encoder-Decoder (seq2seq)	<ul style="list-style-type: none"> <li>- Traduzione automatica</li> <li>- Riassunti (summarization)</li> <li>- Parafrasi, riscrittura controllata</li> <li>- Question Answering estrattivo/generativo</li> <li>- Data-to-text, code-to-comment, ecc.</li> </ul>	T5, BART, mBART, Pegasus, FLAN-T5	<ul style="list-style-type: none"> <li>• Encoder capisce l'input completo; decoder genera un output nuovo → flessibilità nei compiti input→output</li> <li>• Molto forte su task supervisionati con coppie (input, output)</li> </ul>	<ul style="list-style-type: none"> <li>• Modelli più pesanti (doppia rete)</li> <li>• Add-to-cost per training/inference rispetto a soli encoder o soli decoder</li> </ul>

## Come Scegliere l'Architettura Giusta

- Serve "capire" un testo e restituire un'etichetta o un embedding?  
→ Encoder-only
- Serve "scrivere" contenuti lunghi partendo da un prompt breve?  
→ Decoder-only
- Serve trasformare A in B (due sequenze diverse), ad es. tradurre, sintetizzare o spiegare codice?  
→ Encoder-Decoder

# Traduzione automatica neurale

## Evoluzione:

- Sistemi basati su regole
- Sistemi statistici (SMT)
- Sistemi neurali (NMT) con RNN
- Transformer-based NMT

## Sfide specifiche:

- Lingue a basse risorse
- Fenomeni linguistici complessi
- Valutazione della qualità
- Adattamento al dominio

# Riassunto automatico e generazione di testo

## Riassunto automatico:

- **Estrattivo:** Seleziona frasi esistenti
- **Astrattivo:** Genera nuove frasi
- **Applicazioni:** Notizie, documenti legali, letteratura scientifica, meeting

## Generazione di testo creativo:

- **Tipi:** Narrativa, poesia, sceneggiature, marketing
- **Sfide:** Coerenza a lungo termine, originalità, controllo stilistico
- **Uso:** Assistenza creativa, generazione di contenuti, scrittura collaborativa

**Impatto nel settore assicurativo:** Condensazione di report di sinistri e documentazione medica, permettendo ai periti di valutare rapidamente i casi e accelerando il processo di gestione.

# Sistemi di dialogo e assistenti virtuali 🤖

## Componenti:

- **NLU:** Comprensione del linguaggio
- **Gestione del dialogo:** Contesto e decisioni
- **NLG:** Generazione di risposte

## Tipi:

- **Task-oriented:** Focalizzati su compiti specifici
- **Open-domain:** Conversazione generale
- **Ibridi:** Combinano entrambi gli approcci

**Impatto nel settore bancario:** Assistenti virtuali gestiscono richieste comuni come controllo del saldo, trasferimenti e informazioni su prodotti, riducendo il carico sui call center e migliorando la soddisfazione del cliente.

# Analisi e generazione di codice

Un'applicazione emergente con enorme potenziale:

- **Completamento di codice:** Suggerimento di linee o blocchi successivi
- **Traduzione tra linguaggi:** Conversione da un linguaggio all'altro
- **Documentazione automatica:** Generazione di commenti chiari
- **Debugging assistito:** Identificazione e correzione di errori
- **Generazione da descrizioni:** Creazione di codice da specifiche testuali

**Impatto nello sviluppo software:** Strumenti come GitHub Copilot aumentano la produttività fino al 30%, particolarmente per compiti come scrittura di test, funzioni boilerplate e conversione tra formati di dati.

## Come hanno cambiato la tua vita?

Quali di queste applicazioni utilizzi regolarmente?

Come hanno cambiato il tuo modo di lavorare o comunicare?

Quali limitazioni o problemi hai riscontrato?



# Sfide etiche e considerazioni pratiche

L'adozione di modelli linguistici avanzati solleva importanti questioni:

## **Bias e fairness**

- Bias di genere, razziali, socioeconomici, culturali
- Strategie: diversificazione dei dati, debiasing, valutazione disaggregata

## **Contenuti dannosi o ingannevoli**

- Disinformazione, contenuti offensivi, phishing, violazioni di copyright
- Prevenzione: filtraggio, watermarking, educazione degli utenti

## **Efficienza e sostenibilità**

- Costi energetici, emissioni di carbonio, accessibilità
- Ottimizzazione: distillazione, quantizzazione, pruning, architetture efficienti

# Implementazione in produzione

Portare modelli linguistici avanzati in ambienti reali presenta sfide specifiche:

## Considerazioni tecniche

- **Latenza:** Tempi di risposta per applicazioni interattive
- **Scalabilità:** Gestione di volumi variabili di richieste
- **Monitoraggio:** Tracking delle performance nel tempo
- **Aggiornamento:** Strategie per l'evoluzione dei modelli

## Best practices

- **Architettura a più livelli:** Modelli di diverse dimensioni per diversi compiti
- **Caching:** Memorizzazione di risposte comuni
- **Fallback graceful:** Meccanismi di backup
- **A/B testing:** Valutazione comparativa di diverse versioni

# Conclusione

I modelli linguistici e Seq2Seq rappresentano una rivoluzione nel modo in cui le macchine comprendono e generano linguaggio:

- Da semplici modelli N-gram a sofisticate architetture Transformer
- Il meccanismo di attenzione come svolta concettuale fondamentale
- La segmentazione subword come soluzione al problema del vocabolario aperto
- Applicazioni che trasformano settori e creano nuove possibilità
- Sfide etiche e pratiche che richiedono un approccio responsabile

**Il futuro:** Modelli più efficienti, controllabili e allineati con i valori umani

**Grazie per l'attenzione! 🙏**

**Domande?**