Capitolo 1

I modelli LLM

I modelli LLM (*Large Language Model*), come quelli offerti da ChatGPT di OpenAI, sono modelli a reti neurali profonde, una tecnologia sviluppata in questi ultimi anni. Essi hanno inaugurato una nuova era nell'elaborazione del linguaggio naturale (NLP – *Natural Language Processing*). Prima dell'avvento dei modelli LLM, i metodi tradizionali eccellevano in attività differenti, di catalogazione, come la classificazione delle e-mail di spam e il riconoscimento di semplici pattern, che potevano essere catturati con regole specifiche o modelli più semplici. Tuttavia, in genere offrivano prestazioni inferiori nelle attività linguistiche, che richiedevano capacità più complesse di comprensione e generazione, come l'analisi di istruzioni dettagliate, l'esecuzione di analisi contestuali e la creazione di testo coerente e appropriato al contesto. Per esempio, le generazioni precedenti di modelli linguistici non erano in grado di scrivere un'e-mail in base a un elenco di parole chiave, un'attività che oggi è banale per i modelli LLM.

I modelli LLM offrono capacità notevoli nel comprendere, generare e interpretare il linguaggio umano. Tuttavia, è importante chiarire che, quando diciamo che i modelli linguistici "capiscono", intendiamo far riferimento solo alla loro capacità di elaborare e generare testo in modi che ci appaiono coerenti e contestualmente rilevanti, non al fatto che possiedano una coscienza o una comprensione dei concetti simili a quelle umane.

Grazie ai progressi nel deep learning, un sottoinsieme del machine learning e dell'intelligenza artificiale focalizzato sulle reti neurali, i modelli LLM sono stati e sono addestrati su grandi quantità di dati

In questo capitolo

- **Che cosa è un modello LLM?**
- **Applicazioni dei modelli LLM**
- **Fasi di costruzione e utilizzo dei modelli LLM**
- **Introduzione all'architettura a transformer**
- **Utilizzo di grandi dataset**
- **L'architettura GPT**
- **Costruzione di un modello LLM**
- **Riepilogo**

testuali. Questa formazione su larga scala consente loro di catturare maggiori informazioni contestuali e sottigliezze del linguaggio umano rispetto agli approcci precedenti. Di conseguenza, i modelli LLM hanno migliorato significativamente le prestazioni in un'ampia varietà di attività di elaborazione del linguaggio naturale, tra cui la traduzione di testi, l'analisi del sentiment, le risposte alle domande e molto altro.

Un'altra distinzione importante tra i precedenti modelli di elaborazione del linguaggio naturale e i nuovi modelli LLM è che in genere i primi erano progettati per compiti specifici, come la catalogazione del testo, la traduzione linguistica e così via. Mentre quei primi modelli eccellevano nelle loro specifiche applicazioni, i modelli LLM dimostrano una competenza più ampia in una gamma più generale di compiti di elaborazione del linguaggio naturale.

Il successo dei modelli LLM può essere attribuito all'architettura a transformer su cui si basano molti modelli LLM e alle grandi quantità di dati su cui tali modelli vengono formati, che consente loro di catturare un'ampia gamma di sfumature, contesti e modelli linguistici che sarebbero difficili da programmare manualmente.

Questo passo implementativo a modelli basati sull'architettura a transformer e all'utilizzo di grandi dataset di addestramento per addestrare i modelli LLM ha trasformato radicalmente l'elaborazione del linguaggio naturale, fornendo strumenti più efficaci per comprendere il linguaggio umano e interagire con esso.

La discussione di questo capitolo getta le basi per raggiungere l'obiettivo principale di questo libro: comprendere i modelli LLM implementando passo dopo passo tramite codice un modello LLM simile a ChatGPT, basato sull'architettura a transformer.

Che cosa è un modello LLM?

Un modello LLM è una rete neurale progettata per comprendere, generare e rispondere a testi simili a quelli producibili da un essere umano. Questi modelli sono reti neurali profonde addestrate su enormi quantità di dati testuali, che a volte comprendono grandi porzioni dell'intero sapere testuale disponibile oggi al pubblico su Internet.

Il termine *Large* di *Large Language Model* si riferisce sia alle dimensioni del modello in termini di parametri, sia all'immenso dataset sui quali è addestrato. Si tratta di modelli che spesso hanno decine o addirittura centinaia di miliardi di parametri, che sono i pesi regolabili della rete, che durante l'addestramento vengono ottimizzati per prevedere la parola successiva in una sequenza. La previsione della parola successiva è sensata, perché sfrutta la natura sequenziale intrinseca del linguaggio per addestrare i modelli sulla comprensione del contesto, della struttura e delle relazioni all'interno del testo. Tuttavia, questo è un compito molto semplice, e quindi molti ricercatori si sorprendono del fatto che possa produrre modelli così efficaci. Nei prossimi capitoli, discuteremo e implementeremo la procedura di addestramento alla scelta della parola successiva, passo dopo passo.

I modelli LLM utilizzano un'architettura chiamata a *transformer*, che consente loro di prestare un'attenzione selettiva a diverse parti dell'input nel fare le loro previsioni, cosa che li rende particolarmente abili nel gestire le sfumature e le complessità del linguaggio umano.

Poiché i modelli LLM sono in grado di *generare* testo, spesso sono considerati una forma di *intelligenza artificiale generativa*, spesso abbreviata come *GenAI*. Come mostra la Figura 1.1, l'intelligenza artificiale comprende il campo più ampio della creazione di

macchine in grado di eseguire attività che richiedono un'intelligenza di tipo umano, tra cui la comprensione del linguaggio, il riconoscimento di schemi e la presa di decisioni, e include sottocampi come il machine learning e il deep learning.

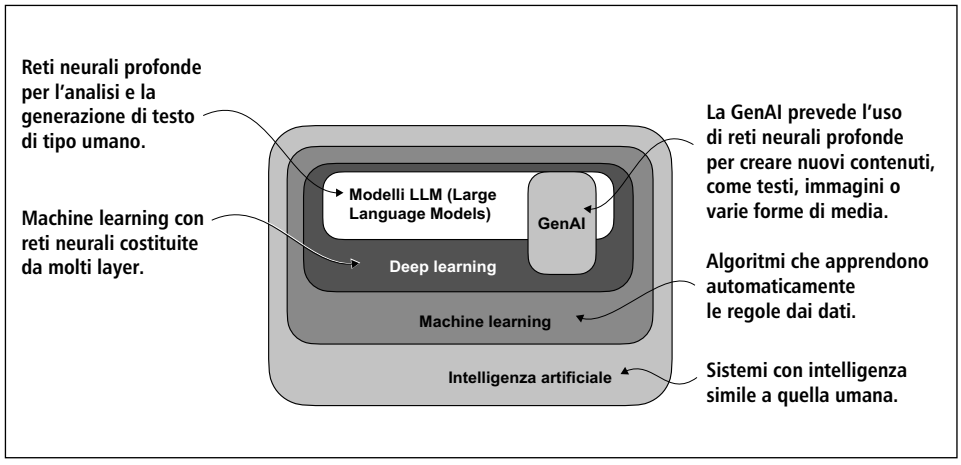

Figura 1.1 Come suggerisce questa rappresentazione gerarchica della relazione tra i diversi campi, i modelli LLM rappresentano un'applicazione specifica delle tecniche di deep learning, utilizzando la loro capacità di elaborare e generare testo di tipo umano. Il deep learning è una branca specializzata del machine learning, che si concentra sull'utilizzo di reti neurali multi-layer. Il machine learning e il deep learning sono campi volti a implementare algoritmi che consentano ai computer di apprendere dai dati e di eseguire attività che in genere richiedono un'intelligenza umana.

Gli algoritmi utilizzati per implementare l'intelligenza artificiale sono al centro del campo del machine learning. In particolare, il machine learning implica lo sviluppo di algoritmi in grado di apprendere, fare previsioni o prendere decisioni sulla base di dati ma senza essere programmati in modo esplicito. Per illustrare ciò, immaginate un'applicazione pratica del machine learning: un filtro antispam. Invece di scrivere manualmente le regole per identificare le e-mail di spam, un algoritmo di machine learning riceve esempi di e-mail pre-etichettate come spam e legittime. Riducendo al minimo l'errore nelle sue previsioni su un dataset di addestramento, il modello impara a riconoscere gli schemi e le caratteristiche indicativi di spam, cosa che gli consente di classificare le nuove e-mail come spam o non spam.

Come illustrato nella precedente Figura 1.1, il deep learning è un sottoinsieme del machine learning che si concentra sull'utilizzo di reti neurali a tre o più layer (chiamate anche reti neurali profonde) per modellare gli schemi e le astrazioni complesse presenti nei dati. A differenza del deep learning, il machine learning tradizionale richiede l'estrazione manuale delle caratteristiche. Ciò significa che ci devono essere esperti umani che si occupano di identificare e selezionare le caratteristiche più rilevanti per il modello.

Sebbene il campo dell'intelligenza artificiale sia oggi dominato dal machine learning e dal deep learning, include anche altri approcci, per esempio l'uso di sistemi basati su regole, algoritmi genetici, sistemi esperti, logica fuzzy o ragionamento simbolico.

Tornando all'esempio della classificazione dello spam, nel machine learning tradizionale, degli esperti umani potrebbero estrarre manualmente certe caratteristiche dal testo

dell'e-mail, come la frequenza di determinate parole trigger (per esempio, "premio", "vincita", "gratis"), il numero di punti esclamativi, l'uso di parole tutte in maiuscolo o la presenza di link sospetti. Questo dataset, creato in base a queste caratteristiche definite, appunto, da esperti, viene poi utilizzato per addestrare il modello. A differenza del machine learning tradizionale, il deep learning non richiede l'estrazione manuale delle caratteristiche. Ciò significa che non è necessario che degli esperti umani identifichino e selezionino le caratteristiche più rilevanti per il modello di deep learning. Tuttavia, sia il machine learning tradizionale sia il deep learning per la classificazione dello spam richiedono comunque la raccolta di etichette, spam/non-spam, che devono essere raccolte da un esperto o dagli utenti.

Esaminiamo ora alcuni dei problemi che i modelli LLM possono risolvere oggi, le sfide che devono affrontare e l'architettura generale dei modelli LLM che implementeremo nel corso del libro.

Applicazioni dei modelli LLM

Grazie alle loro capacità avanzate di analizzare e comprendere dati testuali non strutturati, i modelli LLM hanno un'ampia gamma di applicazioni in vari domini. Oggi, sono impiegati per la traduzione automatica, la generazione di nuovi testi (vedi Figura 1.2), l'analisi del sentiment, la sintesi di testi e molti altri compiti, e sono sempre più utilizzati per la creazione di nuovi contenuti, come la scrittura di narrativa, articoli e perfino codice di programmazione.

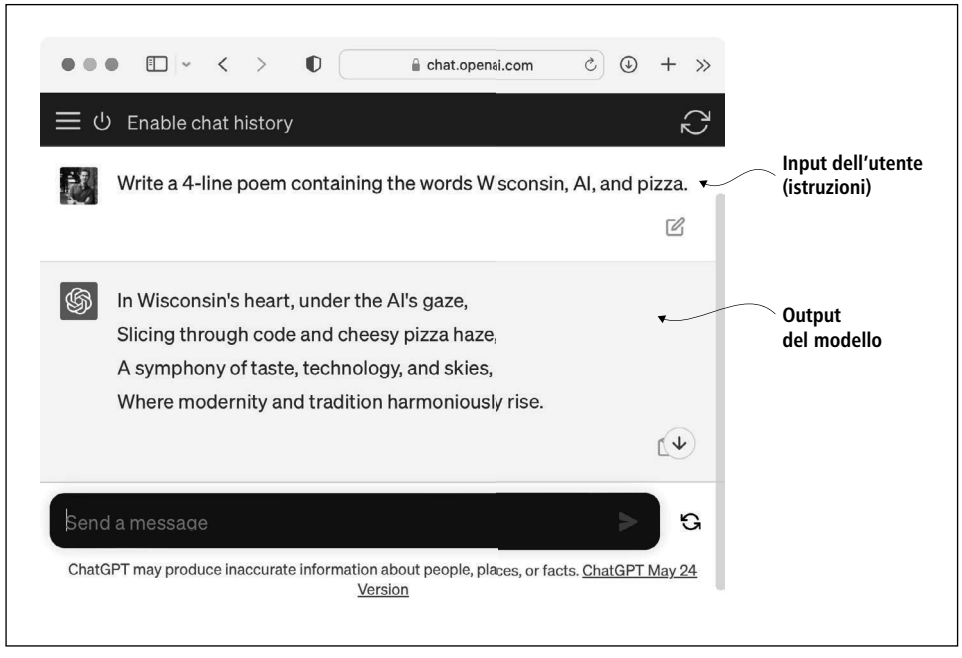

Figura 1.2 Le interfacce LLM rendono possibile la comunicazione in linguaggio naturale tra gli utenti e un sistema. Questa schermata mostra ChatGPT che scrive una poesia sulla base delle specifiche di un utente.

I modelli LLM possono anche alimentare chatbot e assistenti virtuali sofisticati, come ChatGPT di OpenAI o Gemini di Google, che possono rispondere alle query degli utenti e potenziare i motori di ricerca tradizionali come Google Search o Microsoft Bing. Inoltre, i modelli LLM possono essere utilizzati per un recupero efficace delle conoscenze da grandi volumi di testo in aree specializzate come la medicina e il diritto. Ciò include l'analisi dei documenti, la sintesi di lunghi passaggi e la creazione di risposte a domande tecniche.

In breve, i modelli LLM sono utili per automatizzare quasi ogni compito che implichi l'analisi e la generazione di testi. Le loro applicazioni sono praticamente infinite e, mentre continuiamo a innovare ed esplorare nuovi modi per utilizzare questi modelli, è sempre più chiaro che essi hanno le potenzialità per ridefinire il nostro rapporto con la tecnologia, rendendolo più colloquiale, intuitivo e accessibile.

Pertanto, ci concentreremo sulla comprensione del funzionamento dei modelli LLM, a partire dalle basi, programmando un modello LLM in grado di generare testi. Imparerete anche le tecniche che consentono ai modelli LLM di eseguire query, che vanno dalla risposta alle domande alla sintesi del testo, alla traduzione del testo in diverse lingue e altro ancora. In altre parole, studieremo il funzionamento di assistenti complessi come ChatGPT, costruendone uno passo dopo passo.

Fasi di costruzione e utilizzo dei modelli LLM

Per quale motivo dovremmo creare i nostri LLM? Perché programmare un modello LLM da zero è un esercizio eccellente per comprenderne i meccanismi e i limiti. Inoltre, ci fornisce le conoscenze necessarie per pre-addestrare o perfezionare le architetture LLM open source esistenti, in base ai nostri dataset o compiti specifici del nostro dominio.

> **NOTA**
>
> La maggior parte dei modelli LLM di oggi è implementata utilizzando la libreria di deep learning PyTorch, che è quella che useremo anche noi. I lettori possono trovare un'introduzione completa a PyTorch nell'Appendice A.

La ricerca ha dimostrato che, in termini di prestazioni di modellazione, i modelli LLM personalizzati, ovvero concepiti su misura per specifici compiti o domini, possono superare i modelli LLM generici, come quelli forniti da ChatGPT, rivolti a un'ampia gamma di applicazioni. Esempi di questi includono BloombergGPT (specializzato per la finanza) e modelli LLM pensati su misura per rispondere a domande mediche (vedi l'Appendice B per maggiori dettagli).

L'utilizzo di modelli LLM personalizzati offre diversi vantaggi, in particolare per quanto riguarda la privacy dei dati. Per esempio, molte aziende potrebbero non voler condividere dati sensibili con fornitori terzi di modelli LLM come OpenAI, a causa di problemi di riservatezza. Inoltre, lo sviluppo di modelli LLM personalizzati e più piccoli ne consente l'implementazione direttamente sui dispositivi dei clienti, anche laptop e smartphone, un'idea che aziende come Apple stanno attualmente esplorando. Questa implementazione locale può ridurre significativamente sia la latenza sia i costi di utilizzo dei server. Inoltre, i modelli LLM personalizzati garantiscono agli sviluppatori la massima autonomia, consentendo loro di aggiornare e modificare il modello in base alle necessità.

Il processo generale di creazione di un modello LLM include una fase di pre-addestramento e fine-tuning. Il "pre" in "pre-addestramento" si riferisce alla fase iniziale in cui il modello (anche LLM) viene addestrato su un dataset ampio e diversificato per sviluppare una comprensione generale del linguaggio. Questo modello pre-addestrato fungerà poi da risorsa fondamentale che può essere ulteriormente perfezionata, un processo in cui il modello viene addestrato in modo specifico su un dataset più ristretto, specifico per determinate attività o domini. Questo approccio all'addestramento in due fasi costituito da pre-addestramento e fine-tuning è raffigurato nella Figura 1.3.

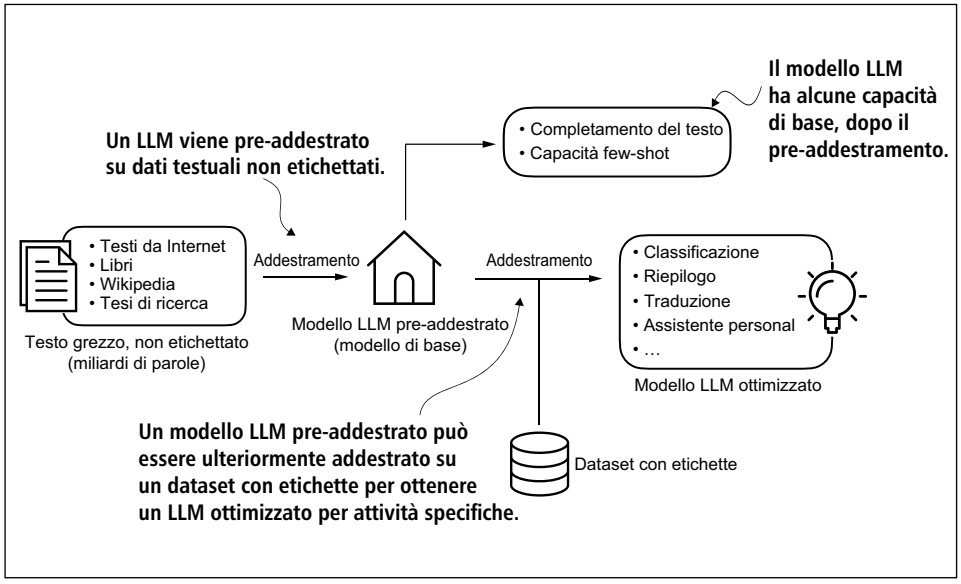

Figura 1.3 Il pre-addestramento di un modello LLM comporta la previsione della parola successiva su grandi dataset testuali. Un modello LLM pre-addestrato può poi essere perfezionato utilizzando un dataset con etichette più piccolo.

Il primo passo per creare un modello LLM consiste nell'addestrarlo su un ampio corpus di dati testuali, *testo grezzo*, dove "grezzo" si riferisce al fatto che questi dati sono solo puro testo, senza alcuna etichettatura (ma potrebbero essergli stati applicati dei filtri, come la rimozione di caratteri di formattazione o l'imposizione di usare documenti in altre lingue).

NOTA

Coloro che hanno un background relativo al machine learning potrebbero notare che in genere le informazioni di etichettatura sono necessarie per i modelli tradizionali di machine learning e per le reti neurali profonde addestrate tramite il paradigma tradizionale di apprendimento con supervisione convenzionale. Tuttavia, la fase di pre-addestramento dei modelli LLM è differente. In questa fase, i modelli LLM utilizzano l'apprendimento ad auto-supervisione, in cui il modello genera da sé le proprie etichette dai dati di input.

Questa prima fase di addestramento di un modello LLM è chiamata *pre-addestramento*, e crea un primo modello LLM pre-addestrato, chiamato *Modello di base*. Un tipico esempio

di tale modello è GPT-3 (il precursore del modello offerto da ChatGPT). Questo modello è in grado di completare il testo, ovvero di finire una frase iniziata da un utente. Ha anche limitate capacità *few-shot*, ovvero può imparare a svolgere nuove attività basandosi anche solo su pochi esempi, anziché aver bisogno di dati di addestramento estese.
Dopo aver ottenuto un modello LLM pre-addestrato tramite l'addestramento su grandi dataset testuali, in cui il modello LLM viene solo addestrato a prevedere la parola successiva nel testo, possiamo addestrare ulteriormente il modello LLM su dati etichettati, operazione nota anche come *fine-tuning*.
Le due categorie più popolari di fine-tuning dei modelli LLM sono il *fine-tuning per rispondere a istruzioni* e il *fine-tuning per attività di classificazione*. Nel fine-tuning per rispondere a istruzioni, il dataset con etichette è costituito da coppie istruzione-risposta, come una query che chiede di tradurre un testo, accompagnata dal testo tradotto correttamente. Nel fine-tuning per attività di classificazione, il dataset con etichette è costituito da testi ed etichette delle classi, per esempio, e-mail associate alle etichette "spam" e "non spam". Tratteremo le implementazioni del codice di pre-addestramento e fine-tuning di un modello LLM e approfondiremo le specifiche del fine-tuning per rispondere a istruzioni e a classificazioni dopo il pre-addestramento di un semplice modello LLM.

Introduzione all'architettura a transformer

La maggior parte dei modelli LLM di oggi si basa sull'architettura a transformer, un'architettura a rete neurale profonda introdotta nel documento del 2017 *Attention Is All You Need* (`https://arxiv.org/abs/1706.03762`). Per comprendere i modelli LLM, dobbiamo comprendere il funzionamento dei transformer, sviluppati per la traduzione automatica, traducendo testi dall'inglese al tedesco e al francese. La Figura 1.4 mostra una versione semplificata dell'architettura a transformer.
L'architettura a transformer è composta da due sub-moduli: un encoder e un decoder. Il modulo encoder elabora il testo di input e lo codifica in una serie di rappresentazioni numeriche o vettori che catturano le informazioni contestuali dell'input. Poi, il modulo decoder prende questi vettori codificati e genera il testo in output. In un'attività di traduzione, per esempio, l'encoder codificherebbe il testo dalla lingua di origine in vettori, e il decoder decodificherebbe questi vettori per generare il testo nella lingua di destinazione. Sia l'encoder sia il decoder sono costituiti da molti layer collegati da un meccanismo di auto-attenzione. A questo punto, potreste avere molte domande su come gli input vengono pre-elaborati e codificati. Affronteremo queste questioni nell'implementazione passo-passo che svolgeremo nei prossimi capitoli.
Un componente chiave dei transformer e dei modelli LLM è il meccanismo di auto-attenzione (non mostrato in figura), che consente al modello di valutare l'importanza delle diverse parole (o token) in una sequenza, l'una rispetto all'altra. Questo meccanismo consente al modello di catturare le dipendenze a lungo raggio e le relazioni contestuali all'interno dei dati di input, migliorando la sua capacità di generare output coerenti e contestualmente rilevanti. Tuttavia, a causa della sua complessità, rimanderemo ulteriori spiegazioni al Capitolo 3, dove discuteremo e implementeremo passo dopo passo questo meccanismo.

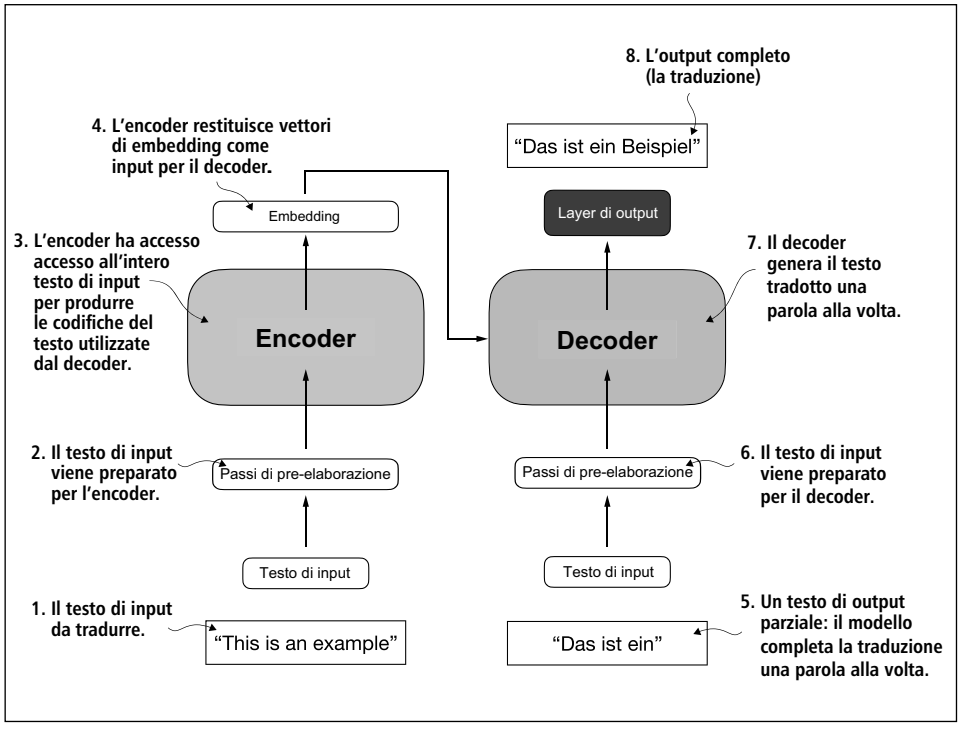

Figura 1.4 Una rappresentazione semplificata dell'architettura originale a transformer, che è un modello di deep learning per la traduzione linguistica. Il transformer è costituito da due parti: (a) un encoder che elabora il testo di input e produce una rappresentazione di embedding (una rappresentazione numerica che cattura molti fattori in varie dimensioni) del testo che il (b) decoder può utilizzare per generare il testo tradotto, una parola alla volta. Questa figura mostra la fase finale del processo di traduzione, in cui il decoder deve generare solo la parola finale ("Beispiel"), dato il testo di input originale ("This is an example") e una frase parzialmente tradotta ("Das ist ein"), per completare la traduzione.

Varianti successive dell'architettura a transformer, come BERT (*Bidirectional Encoder Representations From Transformers*) e i vari modelli GPT (*Generative Pretrained Transformer*), si sono basate su questo concetto per adattare questa architettura a diverse attività. Se siete interessati, potete fare riferimento all'Appendice B per ulteriori suggerimenti di lettura. Un BERT, che è costruito sul sub-modulo encoder del transformer, differisce nel suo approccio all'addestramento da un GPT. Mentre un GPT è progettato per attività generative, un BERT e le sue varianti sono specializzate nella previsione di parole mascherate, dove il modello deve prevedere parole mascherate o nascoste in una data frase, come mostra la Figura 1.5. Questa speciale strategia di addestramento fornisce a BERT alcuni punti di forza nelle attività di classificazione del testo, tra cui la previsione del sentiment e la catalogazione dei documenti. Come applicazione delle sue capacità, al momento della stesura, X (ex-Twitter) utilizza un BERT per rilevare i contenuti tossici.

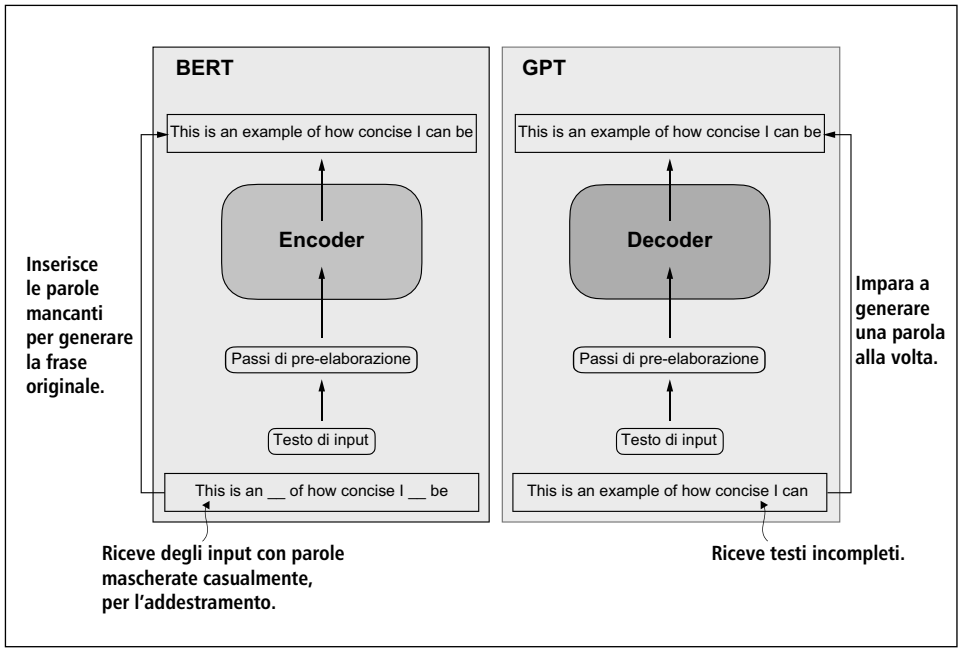

Figura 1.5 Una rappresentazione visuale dei sub-moduli encoder e decoder del transformer. A sinistra, il segmento encoder esemplifica i modelli LLM di tipo BERT, che si concentrano sulla previsione di parole mascherate e sono utilizzati principalmente per attività come la classificazione del testo. A destra, il segmento decoder mostra i modelli LLM di tipo GPT, progettati per attività generative e per produrre sequenze di testo coerenti.

Un GPT, al contrario, si concentra sulla parte decoder dell'architettura a transformer, ed è progettato per attività che richiedono la generazione di testi. Ciò include la traduzione automatica, la sintesi di testo, la scrittura di brani, la scrittura di codice per computer e altro ancora.

I modelli GPT, progettati e addestrati principalmente per eseguire attività di completamento del testo, dimostrano anche una notevole versatilità nelle loro capacità. Questi modelli sono abili nell'eseguire attività di apprendimento sia zero-shot sia few-shot. L'apprendimento zero-shot si riferisce alla capacità di generalizzare rispetto ad attività completamente invisibili, senza alcun esempio specifico precedente. Al contrario, l'apprendimento few-shot implica l'apprendimento da un numero minimo di esempi che l'utente fornisce come input, come mostra la Figura 1.6.

Transformer vs. LLM

I modelli LLM di oggi si basano sull'architettura a transformer. Quindi, transformer e modelli LLM sono termini che spesso vengono usati come sinonimi, in letteratura. Tuttavia, non tutti i transformer sono LLM, poiché i transformer possono essere utilizzati anche per la visione artificiale. Inoltre, non tutti i modelli LLM sono transformer, poiché ci sono modelli LLM basati su architetture ricorrenti e convoluzionali. La motivazione principale alla base di questi approcci alternativi è quella

di migliorare l'efficienza computazionale dei modelli LLM. Resta da vedere se queste architetture LLM alternative potranno competere con le capacità dei modelli LLM basati su transformer e se saranno poi adottate nella pratica. Per semplicità, utilizzo il termine "LLM" per riferirmi ai modelli LLM basati su transformer, simili a GPT. I lettori interessati possono trovare riferimenti bibliografici che descrivono queste architetture nell'Appendice B.

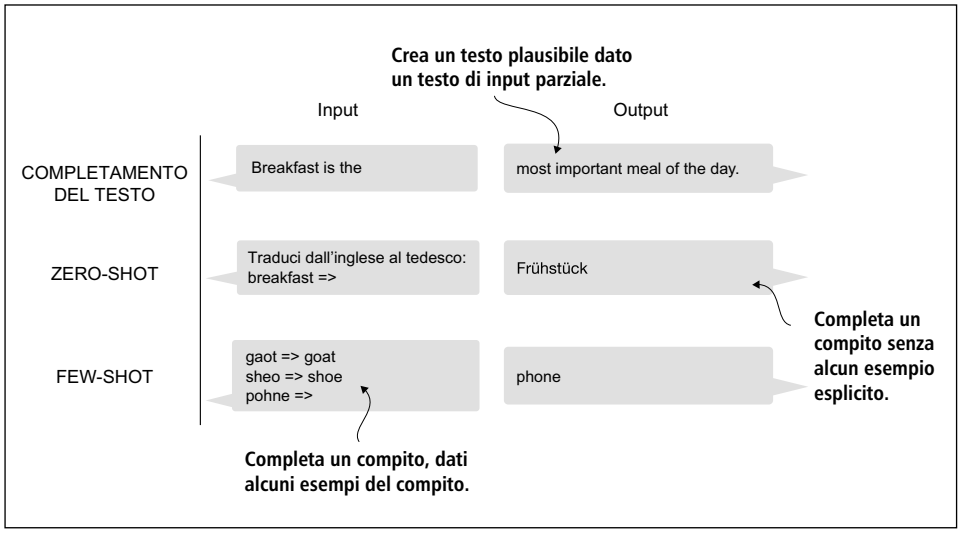

Figura 1.6 Oltre al completamento del testo, i modelli LLM di tipo GPT possono risolvere vari task in base ai loro input senza bisogno di riaddestramento, fine-tuning o modifiche specifiche dell'architettura del modello. A volte, è utile fornire nell'input alcuni esempi del target (un'operazione few-shot). Tuttavia, i modelli LLM di tipo GPT sono in grado di svolgere i loro compiti anche senza un esempio specifico (un'operazione zero-shot).

Utilizzo di grandi dataset

I grandi dataset di addestramento dei modelli di tipo GPT e BERT rappresentano *corpora* di testo diversi e completi, che comprendono miliardi di parole e includono una vasta gamma di argomenti e linguaggi, naturali e di programmazione. Per fornire un esempio concreto, la Tabella 1.1 riassume il dataset utilizzato per il pre-addestramento di GPT-3, che è servito come pattern di base per la prima versione di ChatGPT.

La Tabella 1.1 riporta il numero di token, dove un token è un'unità di testo che un modello legge, e il numero di token di un dataset è più o meno equivalente al numero di parole e segni di punteggiatura presenti nel testo. Il Capitolo 2 affronta l'argomento della tokenizzazione: il processo di conversione del testo in token.

La conclusione è che la scala e la diversità del dataset di addestramento impiegato consentono a questi modelli di funzionare bene in diversi compiti, rispettando la sintassi, la semantica e il contesto del linguaggio, anche se in alcuni casi sono richieste altre conoscenze generali.

Tabella 1.1 Il dataset di pre-addestramento del modello LLM di GPT-3.

Nome dataset	Descrizione	Numero di token	Proporzione nei dati di addestramento
CommonCrawl (filtrato)	Scansione del Web	410 miliardi	60%
WebText2	Scansione del Web	19 miliardi	22%
Books1	Corpus di libri da Internet	12 miliardi	8%
Books2	Corpus di libri da Internet	55 miliardi	8%
Wikipedia	Testo di alta qualità	3 miliardi	3%

I dettagli del dataset GPT-3

La Tabella 1.1 menziona i dataset utilizzati per GPT-3. La colonna Proporzioni riassume la distribuzione dei dati campionati, con arrotondamenti. Sebbene i valori riportati nella colonna Numero token raggiungano in totale i 499 miliardi, il modello è stato addestrato solo su 300 miliardi di token. Gli autori del documento GPT-3 non hanno specificato perché il modello non sia stato addestrato su tutti i 499 miliardi di token.

Per contestualizzare, considerate le dimensioni del dataset CommonCrawl, che da solo è composto da 410 miliardi di token e richiede circa 570 GB di spazio di archiviazione. In confronto, le iterazioni successive di modelli di tipo GPT-3, come LLaMA di Meta, hanno ampliato il loro ambito di addestramento per includere fonti di dati aggiuntive, come i documenti di ricerca di Arxiv (92 GB) e le Q&A relative al codice di StackExchange (78 GB).

Gli autori del documento relativo a GPT-3 non hanno condiviso pubblicamente il dataset di addestramento, ma un dataset comparabile pubblicamente disponibile è *Dolma: An Open Corpus of Three Trillion Tokens for LLM Pretraining Research*, di Soldaini et al. 2024 (`https://arxiv.org/abs/2402.00159`). Tuttavia, la raccolta può contenere opere protette da copyright e i termini di utilizzo possono dipendere dal caso d'uso previsto e dal paese.

La natura pre-addestrata di questi modelli li rende incredibilmente versatili per un'ulteriore e successivo fine-tuning dei compiti, motivo per cui sono anche noti come modelli base. Il pre-addestramento dei modelli LLM richiede l'accesso a risorse significative ed è molto costoso. Per esempio, il costo del pre-addestramento GPT-3 è stimato in 4,6 milioni di dollari in termini di crediti di cloud computing (`https://mng.bz/VxEW`).

Fortunatamente, sono disponibili molti modelli LLM pre-addestrati open source, che possono essere utilizzati come strumenti generici per scrivere, estrarre e modificare testi che non facevano parte dei dati di addestramento. Inoltre, i modelli LLM possono essere ottimizzati per attività specifiche con dataset relativamente più piccoli, riducendo le risorse computazionali necessarie e migliorando le prestazioni.

Più avanti, implementeremo il codice per il pre-addestramento e lo useremo per pre-addestrare un modello LLM a fini didattici. Tutti i calcoli sono eseguibili su normale hardware di livello consumer. Dopo aver implementato il codice di pre-addestramento, vedremo come riutilizzare i pesi disponibili pubblicamente, e come caricarli nell'architettura che implementeremo, il che ci consentirà di saltare la costosa fase di pre-addestramento quando dovremo perfezionare il modello LLM.

L'architettura GPT

La tecnologia GPT è stata presentata nel documento *Improving Language Understanding by Generative Pre-Training* (https://mng.bz/x2qg) di Radford et al. di OpenAI. GPT-3 è una versione ampliata di questo modello, con più parametri e addestrata su un dataset più ampio. Inoltre, il modello originale offerto da ChatGPT è stato creato perfezionando GPT-3 su un ampio dataset di istruzioni utilizzando un metodo descritto nel documento *InstructGPT* di OpenAI (https://arxiv.org/abs/2203.02155). Come avete avuto modo di vedere nella Figura 1.6, questi sono modelli di completamento del testo e possono svolgere altre attività, come la correzione ortografica, la classificazione o la traduzione linguistica. Questo è davvero notevole, dato che i modelli GPT sono pre-addestrati su un compito di previsione della parola successiva relativamente semplice, come mostra la Figura 1.7.

Figura 1.7 Nel compito di pre-addestramento dei modelli GPT, il sistema impara a prevedere la parola successiva in una frase esaminando le parole che l'hanno preceduta. Questo approccio aiuta il modello a comprendere come si compongono normalmente le parole e le frasi, formando una base che può essere applicata a vari altri compiti.

Il compito di previsione della parola successiva è una forma di apprendimento ad auto-supervisione, che è una forma di auto-etichettatura. Ciò significa che non abbiamo bisogno di compilare esplicitamente le etichette per i dati di addestramento in modo esplicito, ma possiamo usare la struttura degli stessi dati: possiamo usare la parola successiva in una frase o in un documento come etichetta per ciò che il modello dovrebbe prevedere. Poiché questo compito di previsione della parola successiva ci consente di creare etichette "al volo", per addestrare i modelli LLM potete usare enormi dataset testuali non etichettati.

Rispetto all'architettura originale a transformer che abbiamo trattato in precedenza, l'architettura generale GPT è relativamente semplice. In sostanza, è solo la parte decoder senza l'encoder (Figura 1.8). Poiché i modelli a decoder come i GPT generano testo prevedendo il testo una parola alla volta, sono considerati modelli *autoregressivi*. I modelli autoregressivi incorporano i loro output precedenti come input per le previsioni future. Di conseguenza, in un GPT, ogni nuova parola viene scelta in base alla sequenza che la precede, il che migliora la coerenza del testo prodotto.

Architetture come GPT-3 sono significativamente più grandi rispetto al modello originale a transformer. Per esempio, il transformer originale ripeteva i blocchi encoder e decoder per sei volte. GPT-3 ha 96 layer transformer e 175 miliardi di parametri in totale. GPT-3 è stato introdotto nel 2020, il che, secondo gli standard del deep learning e dello sviluppo di modelli LLM, è considerato "molto tempo fa". Tuttavia, architetture più recenti, come i modelli Llama di Meta, si basano ancora sugli stessi concetti di fondo, introducendo solo piccole modifiche.

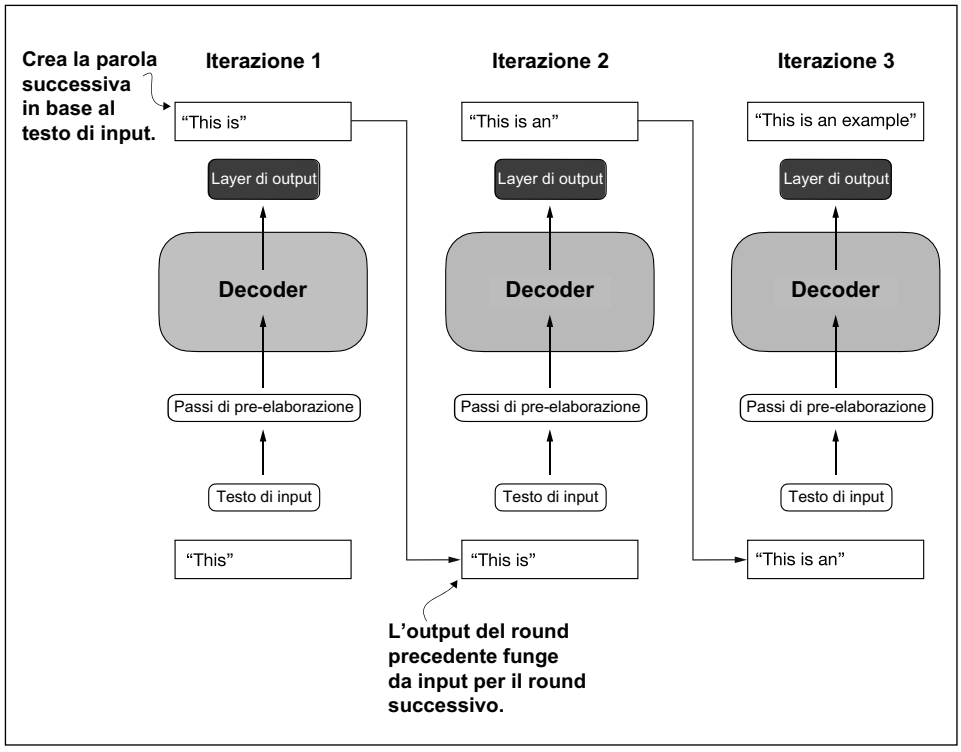

Figura 1.8 L'architettura GPT impiega solo la parte decoder del transformer originale. È progettata per l'elaborazione unidirezionale, da sinistra a destra, il che la rende adatta ad attività di generazione del testo e previsione della parola successiva, per generare testo in modo iterativo, una parola alla volta.

Di conseguenza rimane fondamentale comprendere il funzionamento di un GPT. Per questo mi concentro sull'implementazione di un'architettura GPT, fornendovi al contempo indicazioni sulle varianti impiegate da altri modelli LLM.

Sebbene il modello a transformer originale, costituito da blocchi encoder/decoder, fosse stato progettato esplicitamente per la traduzione linguistica, anche i modelli GPT, nonostante la loro architettura più ampia ma più semplice basata solo sul decoder e mirata alla previsione della parola successiva, sono in grado di eseguire attività di traduzione. Inizialmente, questa capacità è stata un po' una sorpresa per i ricercatori, poiché è emersa da un modello addestrato principalmente su un compito di previsione della parola successiva, un'attività che non aveva come obiettivo specifico la traduzione.

La capacità di eseguire attività per cui il modello non era stato esplicitamente addestrato è un *comportamento emergente*. Questo perché tale capacità non viene insegnata esplicitamente durante l'addestramento, ma *emerge* come conseguenza naturale dell'esposizione del modello a grandi quantità di dati multilingue in vari contesti. Il fatto che i modelli GPT possano "apprendere" i modelli di traduzione tra le lingue ed eseguire attività di traduzione anche se non sono stati specificamente addestrati per farlo dimostra i vantaggi e le capacità di questi modelli linguistici generativi. Possiamo usarli per compiti differenti senza utilizzare modelli differenti.

Costruzione di un modello LLM

Ora che abbiamo gettato le basi per comprendere i modelli LLM, scriviamone uno partendo da zero. Prenderemo l'idea su cui si basa GPT e affronteremo l'impresa questo in tre fasi, come delineato nella Figura 1.9.

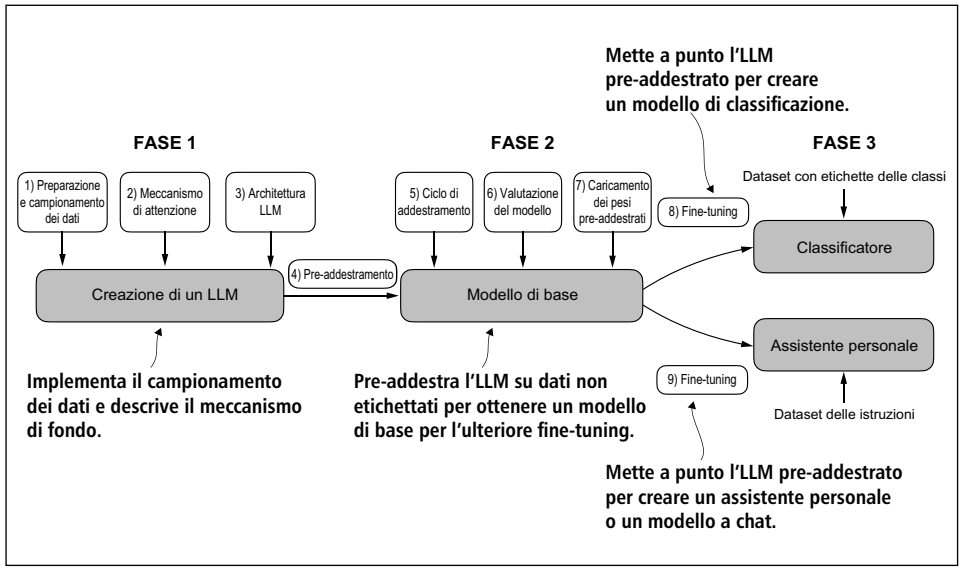

Figura 1.9 Le tre fasi della programmazione di un modello LLM sono l'implementazione dell'architettura LLM e del processo di preparazione dei dati (Fase 1), il pre-addestramento di un modello LLM per creare un pattern di base (Fase 2) e il fine-tuning del pattern di base per trasformarlo in un assistente personale o classificatore di testi (Fase 3).

Nella Fase 1, apprenderemo i passi fondamentali di pre-elaborazione dei dati e programmeremo il meccanismo di attenzione che è al cuore di ogni modello LLM. Nella Fase 2, impareremo a programmare e pre-addestrare un modello LLM di tipo GPT, in grado di generare nuovi testi. Esamineremo anche i fondamenti della valutazione dei modelli LLM, che è essenziale per sviluppare sistemi efficaci di elaborazione del linguaggio naturale.
Pre-addestrare un modello LLM da zero è un'impresa significativa, che richiede migliaia o milioni di dollari in termini di costi di elaborazione. Pertanto, la Fase 2 è rivolta principalmente all'implementazione dell'addestramento per scopi educativi, utilizzando solo un piccolo dataset. Inoltre, fornisco anche esempi di codice per caricare pesi liberamente disponibili.
Infine, nella Fase 3, prenderemo un modello LLM pre-addestrato e lo perfezioneremo per rispondere a istruzioni come rispondere a query o classificare testi, le attività più comuni in molte applicazioni e ricerche.

Riepilogo

- I modelli LLM hanno trasformato l'elaborazione del linguaggio naturale, che in precedenza si basava principalmente su sistemi basati su regole esplicite e metodi statistici più semplici. L'avvento dei modelli LLM ha introdotto nuovi approcci basati sul deep learning, che hanno portato a progressi nella comprensione, generazione e traduzione del linguaggio umano.
- I modelli LLM di oggi prevedono principalmente due fasi di addestramento:
  - innanzitutto, vengono pre-addestrati su un ampio *corpus* di testo non etichettato, utilizzando come etichetta la previsione della parola successiva in una frase;
  - poi vengono perfezionati su un dataset target più piccolo ed etichettato, per rispondere a istruzioni o eseguire attività di classificazione.
- I modelli LLM si basano sull'architettura a transformer. L'idea chiave di tale architettura è un meccanismo di attenzione che fornisce al modello LLM un accesso selettivo all'intera sequenza di input mentre genera l'output una parola alla volta.
- L'architettura originale a transformer è composta da un encoder per l'analisi del testo e da un decoder per la generazione del testo.
- I modelli LLM per la generazione del testo e l'esecuzione di istruzioni, come GPT-3 e ChatGPT, implementano solo moduli decoder, semplificando l'architettura.
- Per il pre-addestramento dei modelli LLM sono essenziali grandi dataset composti da miliardi di parole.
- Mentre il compito generale di pre-addestramento dei modelli di tipo GPT è quello di prevedere la parola successiva in una frase, questi modelli LLM esibiscono proprietà emergenti, come la capacità di classificare, tradurre o riassumere i testi.
- Una volta pre-addestrato un modello LLM, il pattern di base risultante può essere perfezionato in modo più efficiente per diverse altre attività.
- Su attività specifiche, i modelli LLM ottimizzati su dataset personalizzati possono avere prestazioni migliori dei modelli LLM generici.