Intelligenza Artificiale come Supporto allo Studio della Matematica

Strumenti, limiti e trucchi pratici

Marco Barchiesi
Università degli Studi di Trieste

6 maggio 2026

Struttura del seminario

1. Come funzionano gli LLM: token, addestramento, attention.
2. Oltre i chatbot: strumenti che *verificano*.
3. Da oracolo a tutor: trucchi di prompt e calibrazione.
4. Errori tipici.
5. Mini-laboratorio, regole operative.

Come funzionano: predizione del prossimo token

Idea di base

Un Large Language Model genera testo prevedendo, passo dopo passo, quale **token** è più plausibile come continuazione del testo dato. Un token è un pezzo di parola, una parola, o un simbolo.

"La capitale della Francia è..." $\longrightarrow$ "Parigi"

Addestramento

Il modello vede testi incompleti, prova a predire la parte mancante e, correggendosi (modificando miliardi di parametri numerici), impara regolarità linguistiche, concettuali e argomentative.

La risposta non viene recuperata da un archivio: viene **generata al momento**, sulla base del contesto.

Attention: cosa il modello "guarda" nel contesto

Attention

Il meccanismo di **attenzione** assegna pesi diversi alle parti del testo: decide quali parole, simboli o frasi sono più rilevanti per produrre il prossimo token.

- In "Marco ha prestato il libro a Luca perché *lui* doveva studiare", il modello usa il contesto per interpretare il pronome.
- In matematica: collega "f continua su K compatto" all'uso che ne farà la dimostrazione tre righe più sotto.

Il diagramma completo della pipeline (tokenizzazione $\rightarrow$ embedding $\rightarrow$ transformer $\rightarrow$ campionamento) è in appendice.

Perché sembrano intelligenti, perché sbagliano

Perché funzionano spesso

Il linguaggio contiene molta conoscenza implicita: definizioni, esempi, stili di prova, codice, analogie, relazioni tra concetti.

"Roma sta all'Italia come Parigi sta alla Francia."

Perché falliscono

Generano risposte **plausibili**, non necessariamente vere: possono mescolare fatti corretti e falsi, inventare citazioni, o produrre ragionamenti logicamente fragili.

Metafora utile

Uno studente che ha letto moltissimo e imita molto bene il linguaggio matematico. Ma imitazione competente $\neq$ verità garantita.

Messaggio centrale

Tesi

Gli strumenti di AI sono utili se li usiamo per aumentare **comprensione**, **esplorazione** e **verifica incrociata**. Sono pericolosi se sostituiscono il controllo logico.

Buon uso

Esempi, intuizioni, esercizi guidati, feedback, debugging di prove.

Uso rischioso

Soluzioni copiate, citazioni inventate, passaggi non verificati.

Regola

L'AI propone. Lo studente controlla.

Oltre i chatbot: strumenti che *verificano*

Idea chiave

Un chatbot **racconta** un risultato. Un sistema di calcolo simbolico o uno script eseguito lo **producono** in modo verificabile.

Calcolo simbolico (CAS)

- **Wolfram Alpha** – gratuito, web. Limiti, integrali, ODE, semplificazioni.
- **SymPy / SageMath** – Python, gratuiti. Stesse cose, programmabile.

LLM con esecuzione di codice

- ChatGPT (Code Interpreter), Claude (esecuzione di codice).
- L'AI scrive uno script Python, lo **esegue**, e mostra il risultato calcolato, non narrato.

Regola pratica

Per ogni asserzione numerica o simbolica non banale: chiedi una verifica indipendente con CAS o codice eseguito. Non accettare il risultato a parole.

Esempio: sfuggire all'effetto "Scatola Nera"

Prompt debole (Opaco)

"Calcola $\int_0^1 x^x \, dx$."

Cosa fa l'AI oggi:
Non sbaglia quasi più. Riconosce l'integrale (il celebre *Sogno del Sofomoro*), lancia Python di nascosto o usa lo sviluppo in serie, e vi stampa il numero esatto ($0.78343\ldots$).

Il Rischio (Scatola nera):
Ricevete il risultato corretto, ma il processo vi è precluso. Vi abituate ad accettare passivamente un numero da un Oracolo.

Prompt forte (Trasparente)

"Scrivi ed esegui codice Python che calcoli $\int_0^1 x^x \, dx$ con `scipy.integrate.quad`, e in parallelo con stima Monte Carlo. Mostra codice ed errore stimato."

Il Vantaggio (Metodo):
Costringete l'AI a lavorare "a carte scoperte". Ottenete due metodi indipendenti, ispezionate il listato Python, capite l'errore statistico e l'operazione diventa un **esperimento riproducibile**.

Da “oracolo” a tutor socratico

Prompt debole

“Risolvi questo esercizio.”

Rischio: ricevo una soluzione elegante ma opaca; se contiene un errore, potrei non accorgermene.

Prompt migliore

“Non risolvere subito. Fammi 3 domande per capire cosa so, poi dammi un suggerimento alla volta.”

Vantaggio: il modello diventa uno strumento di auto-verifica, non un risolutore opaco.

Trucco 1: separare intuizione e prova

Sto studiando il teorema di Heine–Borel in $\mathbb{R}^n$. Spiegami prima l'intuizione geometrica in 5 righe. Poi dammi due esempi: uno che soddisfa le ipotesi e uno che fallisce. Solo alla fine elenca quali passaggi richiedono una prova rigorosa.

- Utile per definizioni astratte: compattezza, completezza, densità, convergenza debole.
- Non chiedere solo "spiegami": chiedere esempi, non-esempi e confini della definizione.

Trucco 2: generare controesempi

Dammi tre controesempi al seguente falso enunciato: “ogni funzione continua su un aperto limitato è limitata”. Per ciascuno indica quale ipotesi manca rispetto a Weierstrass.

Perché funziona

Molti teoremi diventano chiari quando si capisce *perché* le ipotesi sono necessarie.

Buon esercizio: chiedere all'AI tre controesempi, poi verificarne uno alla lavagna.

Trabocchetti topologici: Compiacenza (*Sycophancy*) vs Ragionamento

Domanda pilotata (Prompt)

"Dimostra che $\mathbb{R}$ con la topologia cofinita è di Hausdorff."

Modelli Standard (Chatbot veloci)

Soffrono di **compiacenza** (*sycophancy*). Pur di assecondare l'ordine "dimostra", assumono che l'enunciato sia vero e **inventano** una prova (es. fingendo che l'intersezione sia vuota).

Modelli Reasoning (es. GPT-5.5)

Sfruttano una catena di pensieri nascosta (*"Thought for..."*). Calcolano i complementari *prima* di rispondere e sventano l'inganno: **"Non si può dimostrare: è falso."**

Verifica logica: U, V intorni aperti non vuoti con $\mathbb{R} \setminus U$ e $\mathbb{R} \setminus V$ finiti. Se $U \cap V = \emptyset$, allora $R = (\mathbb{R} \setminus U) \cup (\mathbb{R} \setminus V)$: assurdo. Lo spazio non è T_2.

Regola d'oro: Mai pilotare la risposta. Invece di chiedere ciecamente *"Dimostra..."*, abituatevi a chiedere: *"Questo enunciato è vero? Se no, forniscimi un controesempio."*

Trucco 3: chiedere una revisione critica

> Ti do una dimostrazione. Non riscriverla. Controlla riga per riga: per ogni passaggio dimmi se segue dalle ipotesi, da un risultato standard, oppure se c'è un gap. Evidenzia eventuali dipendenze circolari.

- Funziona meglio di "la prova è corretta?"
- Costringe il modello a cercare errori invece di confermare.
- Va usato su prove brevi (10–20 righe), non su un intero capitolo.

Trucco 4: testare identità e congetture

Vorrei capire l'identità $\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n}$.

(1) Verificala numericamente per $n = 1, \ldots, 5$ con codice eseguito.

(2) Dammi due strategie di prova: una combinatoria e una algebrica.

(3) Per ciascuna, indica i passaggi che richiedono giustificazione.

- Il test su casi piccoli non prova la formula, ma scopre errori rapidamente.
- Combinare **verifica numerica** (codice) e **discussione di prove alternative** è molto più robusto che chiedere "dimostra".
- Chiedere sempre: "quali casi limite o esempi degeneri devo controllare?"

Trucco 5: calibrazione e “quanto sei sicuro?”

Problema

Gli LLM sono mal calibrati: dichiarano con tono sicuro affermazioni false e raramente dicono spontaneamente “non lo so”.

Per ogni risposta, valuta la tua confidenza in scala 1–5 e indica:
(a) cosa ti rende sicuro;
(b) quale ipotesi o passaggio potrebbe essere sbagliato;
(c) come si potrebbe verificare indipendentemente.

Sotto pressione di auto-verifica, il modello spesso identifica da solo il punto debole della propria risposta.

Bibliografia: massimo scetticismo

Errore comune

Gli LLM possono inventare titoli, autori, numeri di teorema e riferimenti bibliografici, con tono assolutamente convinto.

Trova riferimenti sul teorema X. Per ogni riferimento dammi: autori, titolo, anno, link verificabile, e indica se non sei sicuro. Non inventare nulla: se non trovi, scrivi "non trovato".

- *Ground truth*: libro del corso, arXiv, zbMATH, MathSciNet, pagine degli autori.
- Mai citare un articolo visto solo nella risposta dell'AI.

Prompt pronti da riusare

1. "Non risolvere: dammi solo il prossimo suggerimento."
2. "Fammi un esempio e un non-esempio della definizione."
3. "Quale ipotesi del teorema viene usata in ogni passaggio?"
4. "Cerca un controesempio prima di provare l'enunciato."
5. "Controlla l'ordine dei quantificatori."
6. "Esegui codice Python per verificare numericamente."
7. "Costruisci un esercizio analogo, leggermente più difficile."
8. "Se citi fonti, indica solo riferimenti verificabili."
9. "Dimmi cosa potrebbe essere falso nella mia soluzione."
10. "Se la risposta in italiano è confusa, riformula in inglese."

Mini-laboratorio in aula

Attività

A piccoli gruppi: ognuno usa un LLM per analizzare un enunciato matematico e produce una delle tre uscite seguenti.

1. Un esempio che illustra l'enunciato.
2. Un controesempio a una variante falsa.
3. Una verifica di un passaggio logico sospetto, con codice eseguito se applicabile.

Regola

Il gruppo deve spiegare a voce perché la risposta dell'AI è corretta o dove è insufficiente. Non si legge la risposta dell'AI: la si critica.

Cosa non fare

Evitare

- Copiare soluzioni senza capirle.
- Caricare dati personali o materiale riservato.
- Chiedere “dimostra” senza prima controllare se l’enunciato è vero.
- Fidarsi di bibliografia non verificata.

Preferire

- Prompt socratici.
- Verifica con CAS o codice eseguito.
- Soluzione propria, poi feedback.
- Controesempi prima delle prove.

Integrità accademica

Verifica le policy del corso e dell’Ateneo. In dubbio, dichiara nell’elaborato l’uso che hai fatto dell’AI. La trasparenza protegge te.

Conclusione

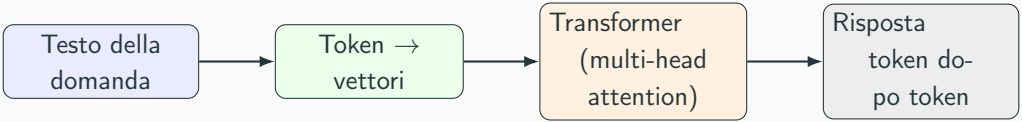
Tre regole operative

1. Usare l'AI per **esplorare**, non per certificare.
2. Chiedere **esempi, non-esempi e controesempi** prima delle soluzioni.
3. Trattare ogni asserzione non banale come una **bozza da verificare** con codice o CAS.

L'obiettivo non è studiare meno.

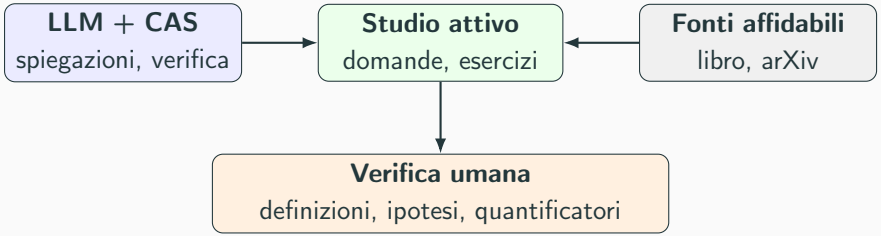
È studiare con più feedback.

Appendice: anatomia di una risposta



Ogni token è rappresentato come un vettore (embedding). L'attention combina questi vettori, decidendo quanto ciascun token influenza gli altri. Strati successivi raffinano la rappresentazione fino al vocabolario di uscita, da cui si campiona il prossimo token.

Appendice: mappa d'uso per uno studente



Lo studio attivo è al centro. LLM+CAS e fonti affidabili sono input paralleli. La verifica umana è l'ultimo filtro, sempre.

Appendice: usare l'AI per costruire esercizi

Genera 5 esercizi sulla convergenza di successioni: due facili, due medi, uno trabocchetto. Non dare subito le soluzioni. Per ogni esercizio indica solo quale definizione o teorema serve.

Uso consigliato

Preparare l'esame: non "fammi il riassunto", ma "interrogami", "dammi feedback", "aumenta la difficoltà".

Appendice: prompt per revisione di una soluzione

Ecco la mia soluzione. Valutala come farebbe un docente:
(1) identifica il primo errore logico, se esiste;
(2) controlla le ipotesi usate;
(3) segnala passaggi troppo rapidi;
(4) non riscrivere la soluzione completa, ma dammi indicazioni per correggerla.

Appendice: prompt per studio attivo

Interrogami su questo argomento. Fammi una domanda alla volta. Dopo ogni mia risposta, dimmi se è corretta, chiedimi di precisare un punto, e solo alla fine dammi un riepilogo degli errori ricorrenti.

Appendice: prompt verifica logica

About Theorem 1, check whether the proof is logically correct step by step. For each step, verify that it follows from earlier steps, hypotheses, or standard results, and point out any gaps, unjustified implications, or hidden assumptions.

Appendice: prompt editoriale

I am writing a research paper in mathematics. Please review the attached text as an academic editor with expertise in mathematical writing. Your goals are to:
- Improve clarity, coherence, and conciseness while preserving mathematical precision and logical structure.
- Ensure definitions, theorems, lemmas, and proofs are presented in a clear, professional, and consistent style.
- Check that the mathematical notation is standard, unambiguous, and typeset correctly (if relevant to LaTeX use).
- Suggest improvements to sentence flow, transitions, and readability without changing the meaning.
- Identify passages that might benefit from rephrasing for better comprehension or academic tone.

Provide comments and proposed revisions section by section. Keep the language formal but natural, and maintain the author's original voice.

Appendice: prompt Gem/Project

Research-math mode. Treat me as an expert reader in probability, GMT, calculus of variations, and analysis. Be concise, dense, and technically rigorous. Preserve my notation and level of generality. Do not silently rename objects, change assumptions, or weaken claims...

Tutors virtuali

Tutor virtuale su ChatGPT, usa le note del corso come database. Potete chiedergli spiegazioni, approfondimenti, esercizi ecc.

https://chatgpt.com/g/g-67c61d9d44648191aa969952f120aa4e-ai-tutor-probabilita

Similmente, tutor virtuale su Gemini.

https://gemini.google.com/gem/17hgkGKtLTyU2ZgOsb2XcRsOi2qq0acSP?usp=sharing

1. Benchmark AI: che cosa è stato testato

Obiettivo

Valutare modelli AI su compiti realistici di ricerca matematica:

- dimostrazioni in GMT e PDE;
- produzione di LaTeX compilabile;
- testi utilizzabili in articoli, seminari e note tecniche.

Modelli confrontati

GPT-5, Gemini 3 Pro, Claude Sonnet. I modelli premium sono inclusi nel kit replicabile, ma non eseguiti automaticamente.

Prompt	Area	Punto critico
P1	GMT	Cantor non-uniforme
P2	PDE	Schauder parabolico
P3	LaTeX	Preambolo + TikZ
P4	BV	Minimi + arXiv-ready

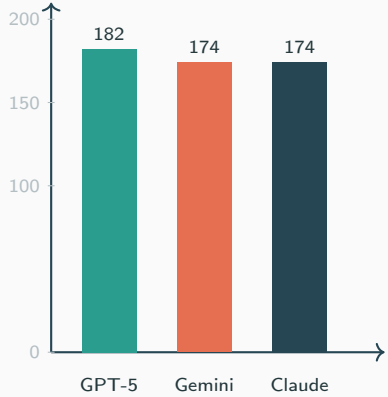
Metrica

Sei dimensioni: correttezza, LaTeX, completezza, chiarezza, costo, tempo.

$$\text{pesato} = 2 \times \text{qualità} + \text{efficienza}.$$

2. Classifica complessiva

#	Modello	Qualità	Totale	Pesato
1	**GPT-5**	75/80	107/120	**182/200**
2	Gemini 3 Pro	71/80	103/120	174/200
2	Claude Sonnet	73/80	101/120	174/200



Risultato chiave

GPT-5 vince complessivamente: è il più equilibrato tra rigore matematico, LaTeX compilabile, completezza e rapidità. Gemini e Claude sono pari nel pesato, ma con profili diversi.

3. Raccomandazioni operative

GPT-5: scelta di default

- migliore equilibrio complessivo;
- LaTeX compilabile nei test;
- adatto ad articoli, note tecniche e bozze rifinite.

Gemini 3 Pro: sintesi affidabile

- risposte compatte e corrette;
- buon LaTeX;
- utile per controlli rapidi e formule standard.

Claude Sonnet: rigore, non produzione finale

- dimostrazioni più dettagliate;
- ottimo per brainstorming teorico;
- problematico su macro e file LaTeX finali.

Attenzione

Claude fallisce i due test LaTeX-intensivi: macro malformate, pacchetti mancanti, colori non definiti, sintassi bibliografica incompatibile.

Regola pratica: Claude per pensare; GPT-5 o Gemini per produrre LaTeX che compila.