DevFest Mediterranean 2025

Sant'Agata di Militello

L'illusione del controllo: gestire l'imprevedibilità degli LLM

Gabriele Lo Cascio | MSc in Computer Engineering

GDG Nebrodi
Google Developer Group

Il caso mediatico

il POST

MONDO | Sabato 17 febbraio 2024

Air Canada dovrà risarcire un cliente che aveva ricevuto informazioni fuorvianti da un chatbot

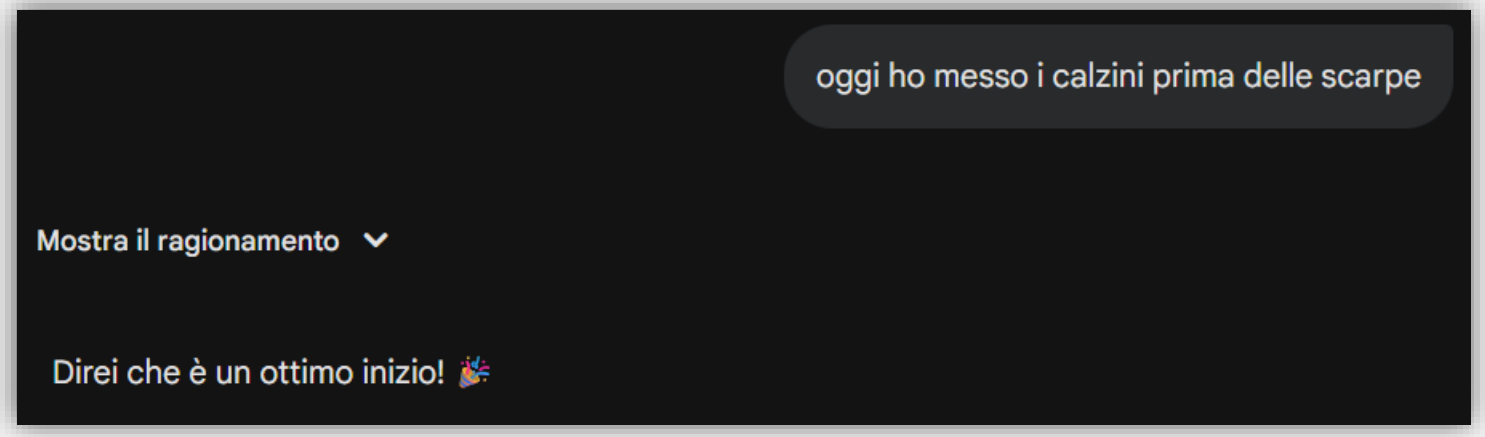
Dopo che inizialmente aveva cercato di non assumersi la responsabilità delle informazioni fornite dall'assistente virtuale

Link all’articolo

GDG Nebrodi
Google Developer Group

Adulazione

- Il modello si comporta come un **people pleaser**.
- Si prioritizza la forma e la cortesia rispetto alla logica e alla verità.
- Il modello non ha un grounding logico, ma solo sociale: dire una sciocchezza con convinzione produce consenso.

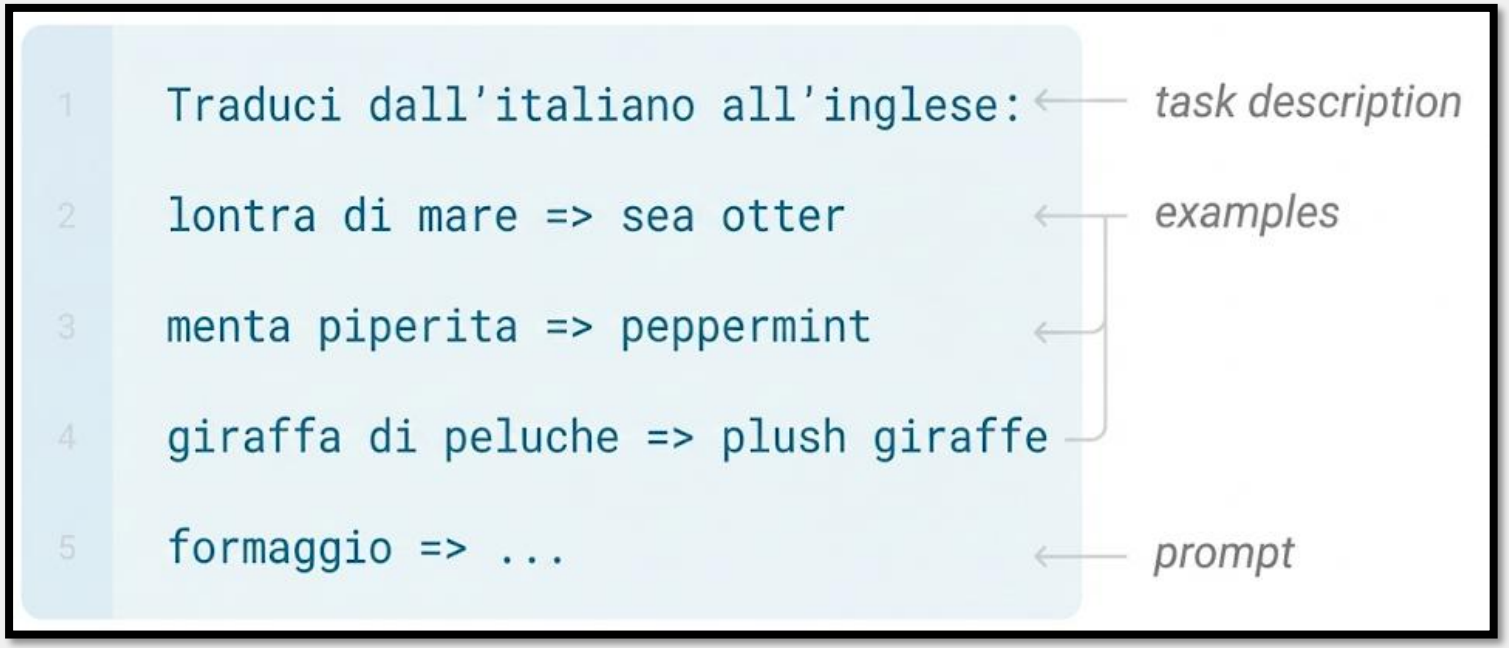


Link alla chat

GDG Nebrodi
Google Developer Group

Verso il ragionamento

- Per indurre un determinato comportamento si applica la tecnica del **prompting**.
- Chiariamo il problema e definiamo gli obiettivi da raggiungere.
- Capacità emergenti: il modello apprende nuovi task dagli esempi nel contesto.

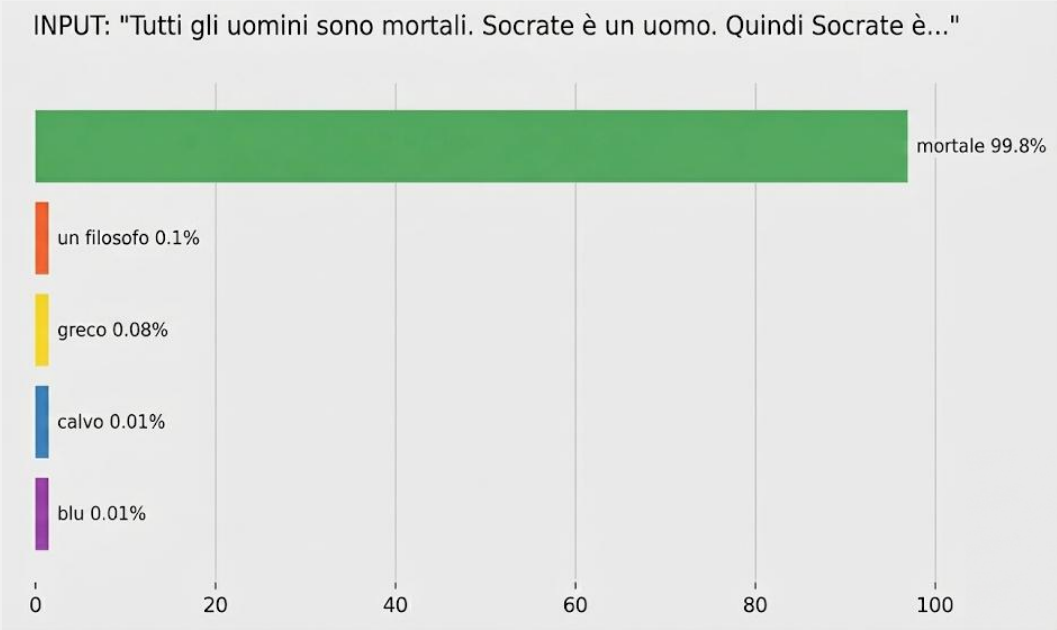


Language Models are Few-Shot Learners

GDG Nebrodi
Google Developer Group

Il Ragionamento

- Non è una funzionalità inserita esplicitamente, ma è una **abilità emergente**.
- Da notare che il ragionamento è puramente linguistico. Non c'è alcun modulo di logica interno.
- La statistica impone «mortale» come prossima parola, ma non è detto sia quella più logica.



Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

GDG Nebrodi
Google Developer Group

Limitazioni

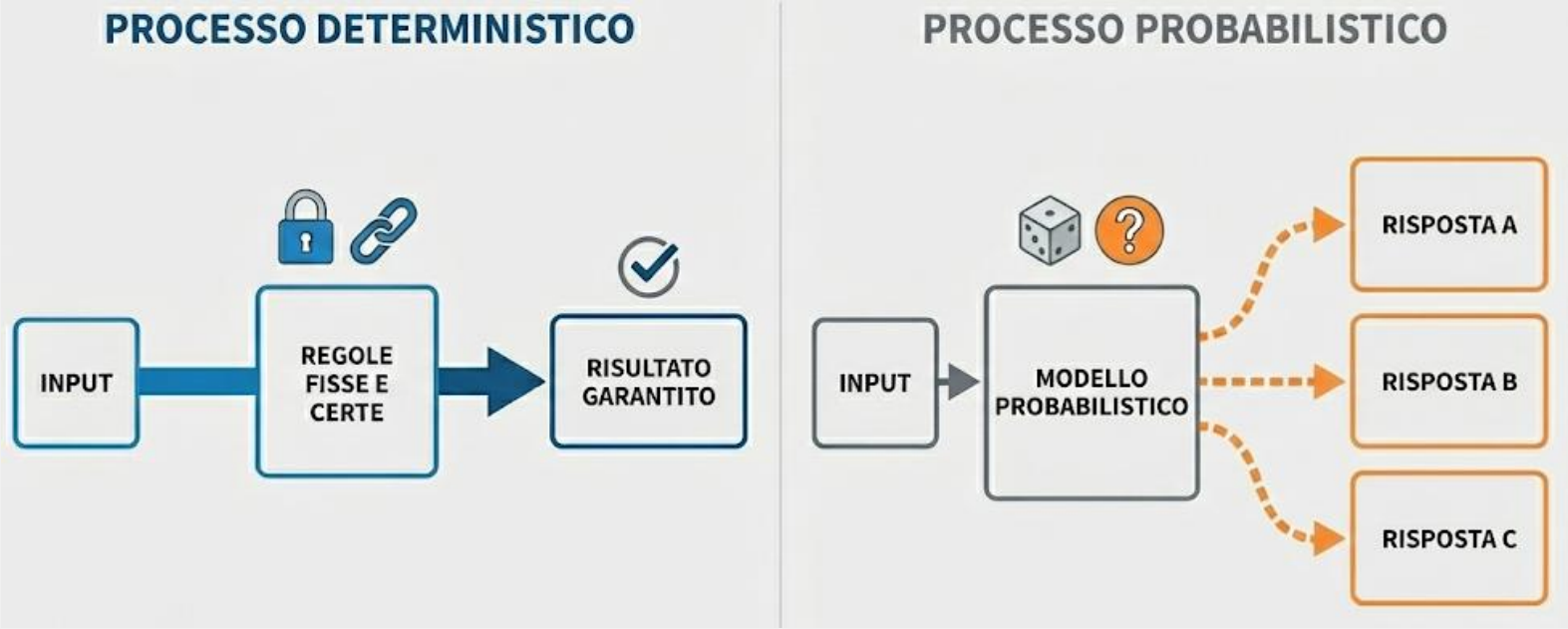
- Alla fine della partita il modello è pur sempre **probabilistico**.
- I prompt sono suggerimenti, non regole e non offrono garanzie matematiche di esecuzione.
- Il ragionamento «passo-passo» riduce gli errori. Ma può capitare che un passaggio intermedio errato nel ragionamento fornisca un risultato falso seppur estremamente convincente.

Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting

GDG Nebrodi
Google Developer Group

Il modello come Interprete

- Sfruttiamo l’LLM come **interprete**.
- Gestiamo l’imprevedibilità vincolando l'output in una struttura rigida.
- Blindiamo l'esecuzione delegando la decisione agli algoritmi deterministici.

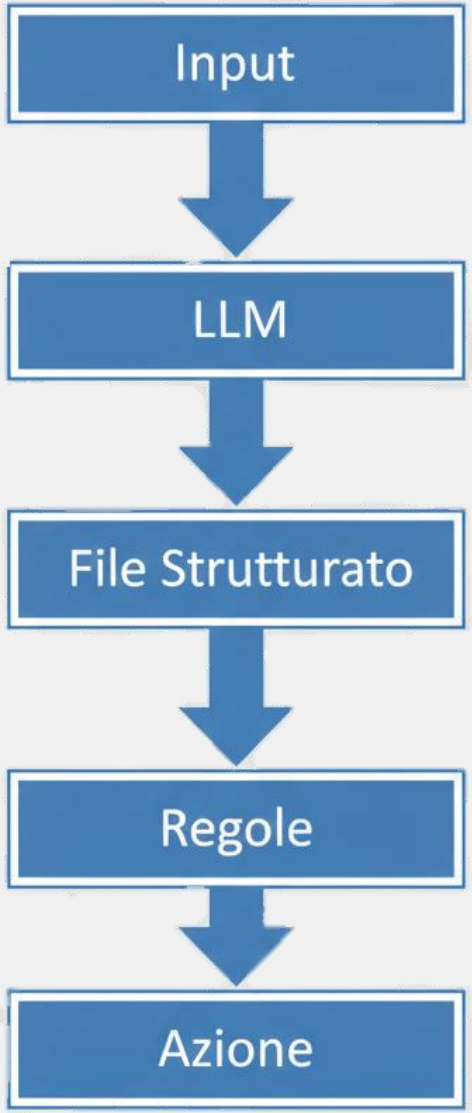


SatLM: Satisfiability-Aided Language Models Using Declarative Prompting

GDG Nebrodi
Google Developer Group

Architettura

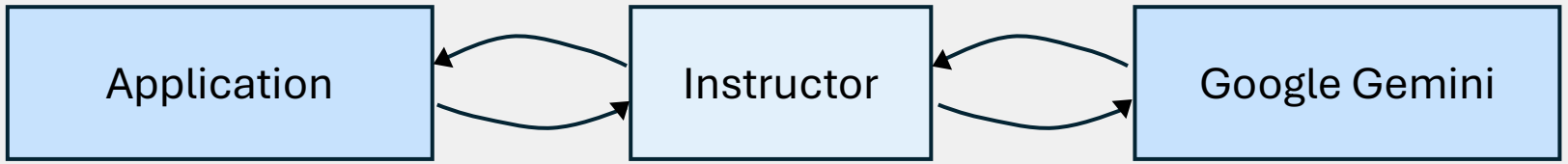
- La richiesta viene elaborata dal modello.
- Viene prodotto un file strutturato: JSON , XML.
- Il file generato viene validato tramite delle regole.
- Sulla base dei parametri ricevuti si esegue un algoritmo.



GDG Nebrodi
Google Developer Group

Una possibile soluzione: Instructor

- Gli LLM potrebbero formattare le risposte in modo diverso ogni volta.
- Offre validazione automatica, gestione dei retry e supporto allo streaming per output sempre affidabili.
- Compatibile con Google Gemini.
- Offre supporto per i principali linguaggi di programmazione: Go, Python, Ruby, Rust e altri.



Repository Instructor

GDG Nebrodi
Google Developer Group

Lo Schema

- Definizione base dell’oggetto e del tipo dei suoi attributi.

```
class Utente(BaseModel):
        nominativo: str
        anni: int
        città: str
```

- Aggiunta di ulteriori vincoli che verranno successivamente validati.

```
class Utente(BaseModel):
        nominativo: str
        anni: int = Field(gt=0, lt=120)
        città: str
```

- Ecco il JSON che ci aspettiamo venga restituito dal modello.

```
{
  "nominativo": "Gabriele Lo Cascio",
  "anni": 26,
  "città": "Palermo"
}
```

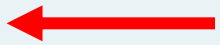
GDG Nebrodi
Google Developer Group

Validazione

- Sfruttiamo ancora Pydantic e aggiungiamo una funzione che ci assicura che il campo nominativo sia fatto come segue «Nome[spazio]Cognome».

```
class Utente(BaseModel):
        nominativo: str
        anni: int
        città: str

        @field_validator('nominativo')
        def nominativo_deve_contenere_spazi(cls, nominativo):
                if ' ' not in nominativo :
                        raise ValueError('Il nominativo deve contenere almeno uno spazio')
                return nominativo
```



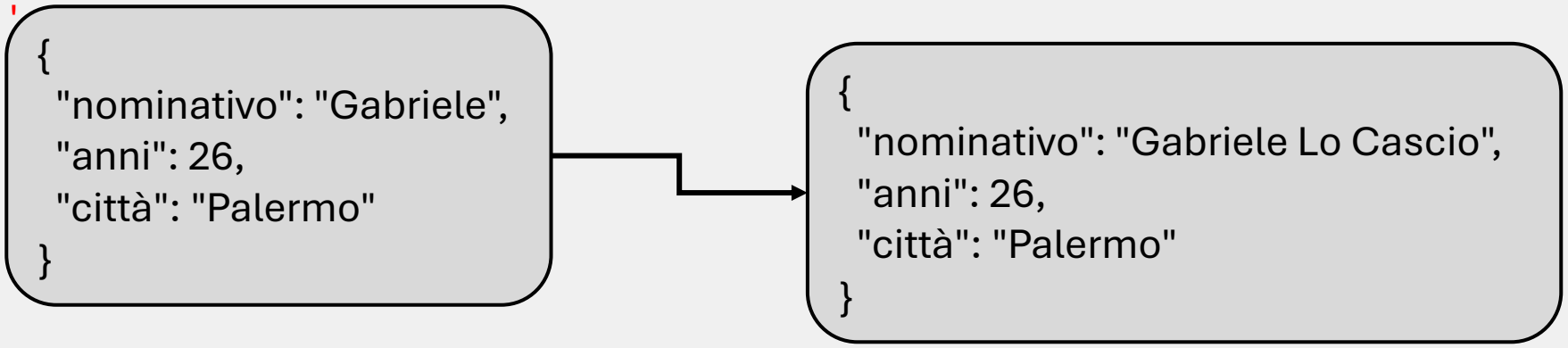
GDG Nebrodi
Google Developer Group

Eventuale fase di Retry

- Il campo «nominativo» non presenta spazi'.
  Evidentemente il modello ha perso di vista il cognome o magari non è stato fornito.

- Si invia lo schema ricevuto e l'errore causato e si ritenta più volte questa fase finché si ottiene conformità allo schema.

- Il numero di tentativi è limitato ed è importante progettare una fallback per preservare la resilienza dell'applicazione.



GDG Nebrodi
Google Developer Group

What if....?

- Applichiamo quello che abbiamo visto al caso di Air Canada.

```
class RichiestaUtente(BaseModel):
        stato_biglietto: Literal["non_prenotato", "prenotato"] = Field(description="Stato attuale della prenotazione del cliente")
        reason: str

        @field_validator('stato_biglietto')
        def validate_policy(cls, v):
                if v == "prenotato":
                        raise ValueError("Policy Violata: Le tariffe lutto non sono retroattive. La richiesta deve avvenire PRIMA dell'emissione.")
                return v
```

GDG Nebrodi
Google Developer Group

DevFest Mediterranean 2025

Sant'Agata di Militello

/Grazie a tutti per l'attenzione

Gabriele Lo Cascio

MSc student in

Computer Engineering,
University of Palermo

GDG Nebrodi
Google Developer Group