Lezione 13 20/05/2024

Large language models

Sono dei modelli che cercano di predire il testo, a seguito di un prompt. Provano a stimare la parola più probabile dopo un certo discorso. Questi fanno più fatica a lavorare sui numeri, perché invertire due parole ci può stare, invertire due numeri no.

La caratteristica fondamentale è la dimensione del linguaggio, ovvero il poter imparare su una quantità enorme di testi. Poi ci sono dei meccanismi di attenzione, ovvero quella tecnica usata per determinare quanto è importante ciascun token rispetto all'output.

Si vuole quindi avere un singolo modello che sappia risolvere molte tasks.

La AI generativa cambia la modalità di interazione nella gestione, memorizzazione e interrogazione di dati e informazioni.

Si può usare l'interazione in linguaggio naturale non solo per la conoscenza contenuta nei sistemi AI, ma anche per accesso a informazioni strutturate, come interrogazioni SQL. Inoltre ci sono anche modelli multimodali, che possono analizzare testi, audio, dati strutturati.

Si ha quindi human-data interaction, ovvero la possibilità di interagire direttamente con i dati. Per fare questo bisogna poter trasformare il linguaggio naturale in SQL, e poi mi serve uno strumento che data una tabella la può descrivere.

Limiti di questi modelli:

- Le informazioni con cui i sistemi sono addestrati
  - Da dove provengono? Che impatto hanno nella capacità dei sistemi di AI generativa
- Allucinazione
  - Frasi corrette dal punto di vista grammaticali, ma senza senso dal punto di vista del contenuto
- Aggiornamenti temporali
  - Non è possibile facilmente aggiornare le informazioni note ai sistemi di AI generativi

Sulla provenienza dei dati c'è quindi il rischio di bias, di disinformazione, perchè garbage in → garbage out.

Gli attuali meccanismi di guardrails, di regole per evitare output problematici non sono molto efficaci.

Small open source model

I SML sono versioni più piccole dei LLM, che hanno molti meno parametri, e quindi costano molto meno e sono più efficienti da eseguire, cercando di raggiungere gli stessi risultati. Questo semplifica alcune cose, per esempio lo posso eseguire nella mia macchina locale e non rischiare di inviare dati sensibili, poi si può anche customizzare il modello meglio, per fare fine tuning.

La distillazione della conoscenza (knowledge distillation) è la tecnica di trasferire la conoscenza da un modello LLM trainato ad uno più piccolo.

Pruning e quantizzazione: è la rimozione delle parti non necessarie del modello, riducendo la precisione dei loro pesi.

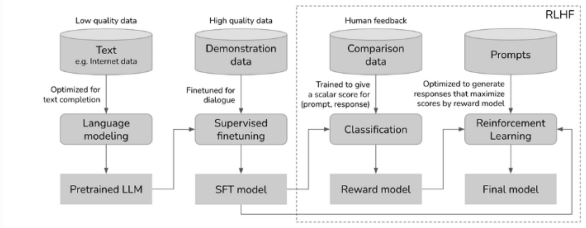
Fine tuning

é una tecnica per adattare un LLM già trainato, su dei dati nuovi. L'obiettivo è quello di allenare il modello su abbastanza tokens che siano rappresentativi del nuovo dominio.

Ci sono due tecniche: quella supervisionata (creo un dataset da 0 e costruisco un processo di reinforcement learning from human feedback (RLHF), e quella che prende una serie di coppie di domande e risposte da cui imparare.

LLM FLow

1. **Building a preference dataset**. By prompting the model in step 2, build a dataset of *{prompt, response 1, response 2}*, where a group of humans decides which response is better, 1 or 2.
2. **Training a reward model.** a model that learns to predict, for every new response
3. **Maximize reward.** Finally train the model to maximize the reward. Train model on a policy that learns to obtain the highest rewards from the reward model.



Il fine tuning invece vuole provare a specializzare il modello. L'idea base è quella di andare ad aggiustare alcuni dei pesi, non tutti, soprattutto gli ultimi layers. Quindi ne vanno modificati in giusto numero, troppi pochi e non cambio niente, troppi e ho la "dimenticanza catastrofica".

Poi c'è la questione della valutazione dei modelli, come faccio a capire quale tra due modelli è il migliore. Esistono molti benchmark, ma sono spesso di tipo sintattico.

Un altro problema è che questi modelli non dimenticano, quindi se il LLM usa un dato per il training, ma poi questi dati vogliono essere rimossi (per esempio per il diritto all'oblio o per copyright) questo è difficile da fare, è un problema aperto.

Allucinazione

I modelli fine-tunati difficilmente hanno errori di allucinazione.

Retrieval augmented generation architecture

Non sempre possiamo mettere in un LLM le conoscenze come i dati personali. Oppure magari i dati cambiano spesso e quindi non ha senso fare fine tuning. Tutti i casi in cui voglio usare conoscenza aggiornata, oppure non voglio trainare un nuovo modello, una possibile soluzione è la retrieval augmented generation.

C'è una parte dove questi sistemi di retrieval cercano di estrarre informazioni rilevanti, poi c'è una fase generativa dove queste informazioni vengono date in pasto ai LLM.

L'idea è quella di dire per esempio ho un pdf che contiene le specifiche di una nuova auto, dati non contenuti nel LLM, quindi nella fase retrieval il modello retrieval va ad estrarre dal pdf il dato che mi interessa, e poi lo da in mano al modello generativo che lo mette insieme al prompt per dare la risposta.

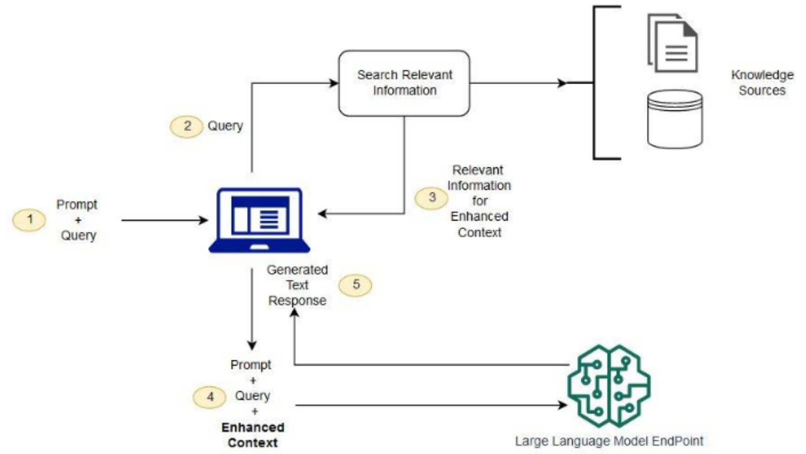
Quindi vado a dare al modello del contesto da usare.

Chunking

é un processo che rompe un documento in blocchi di testo piccoli. Ciascun chunk è poi dato ad un LLM per creare un embedding, e quindi poi la coppia testo-embedding è salvata in un database vettoriale. Tramite questo vettore posso poi fare una ricerca approssimata al posto di una esatta.

Però questo vuol dire che se il LLM viene modificato, bisogna ricostruire l'intero database vettoriale. C'è bisogno dello stesso identico LLM per tutte le entry.

Questa è una soluzione efficace per i sistemi RAG:



Le tipiche soluzioni RAG sono basate su un database vettoriale che salva per ciascun documento i suoi embeddings, la query è quindi tradotta in embedding e un vettore di ricerca è creato. Quindi i pezzi dei documenti trovati sono mandati al LLM come contesto.

Si può utilizzare anche un grafo della conoscenza per query complesse.

RAG Advanced

Limiti:

a volte i sistemi RAG faticano a capire l'intento dietro a

Understanding User Intent: RAG systems sometimes struggle to grasp the exact intent behind a user's query, which is crucial in surfacing the correct information to the LLM.

Dependency on Vector Embeddings: RAG relies heavily on vector embeddings to interpret and match queries with relevant information. While these embeddings are powerful, they are not infallible and can sometimes lead to inaccuracies or oversimplifications in understanding query context.

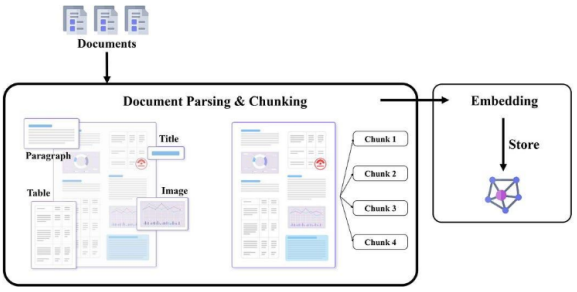
Black Box Nature: The process of generating and comparing vector embeddings is complex and often opaque. With embeddings having potentially hundreds of dimensions, it's challenging to discern how they are structured and how they influence the similarity scores used in semantic search.

Generic Training Data: Embedding models are typically trained on generic datasets, which might not capture the specific nuances or contexts relevant to certain queries. This can lead to superficial similarities being drawn between different pieces of content.

???

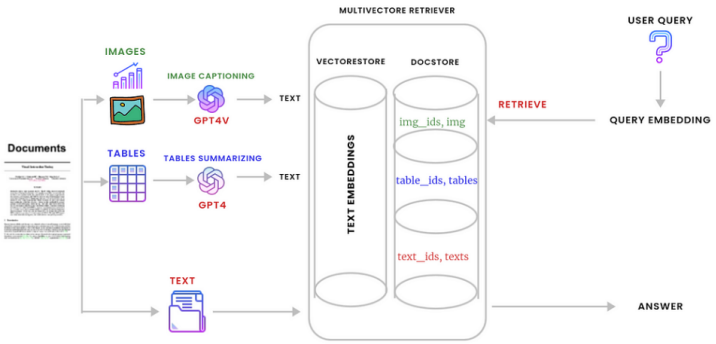
TEXTUAL INFORMATION

- **Document Structure Recognition:** A pdf is organized in different types of content blocks like paragraphs, tables, and charts. This ensures that the divided text blocks are complete and independent semantic units.
- **Robustness in Complex Document Layout:** complex layouts, such as multi-column pages, border-less tables, and even tables with merged cells.
  - Rule based approach (pypdf)
  - Deep Learning based approach (pdfparser)



MULTIMEDIA FILE

- Documents sometimes include not only text, but also images and tables
- A multimodal apporach for embedding is mandatori



IN CONTEXT

- Sometimes enduser query lack of context and does not provide enough information to LLM
  - Fine tuning
  - Provide context
- Use KG for providing context
  - See later

Chain of tought

- In chain-of-thought prompting, the input question is followed by a series of intermediate natural language reasoning steps that lead to the final answer
- *"Let's think step by step"*

Access control

Come faccio a garantire il controllo degli accessi alla conoscenza di un LLM? Una persona potrebbe non essere autorizzata ad accedere ad informazioni sensibili.

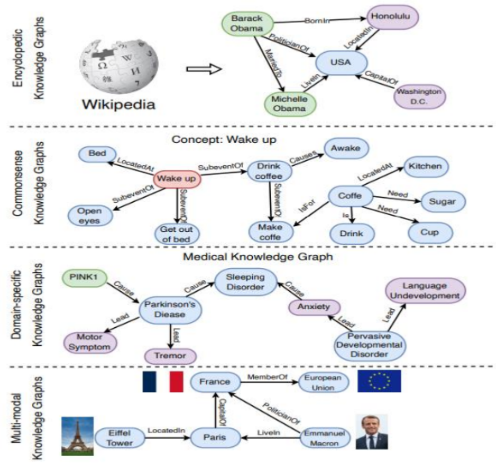
Mentre nei database posso imporre delle regole sull'accesso, nei LLM non posso.

Nei RAG questo controllo lo posso fare, rispetto agli accessi ai chunk.

Knowledge graph

I grafi di conoscenza è una base di conoscenza usata da google per dare dei risultati aggiuntivi alle ricerche (quelli che escono a lato delle ricerche). Questi dati sono sintatticamente o semanticamente collegati alla ricerca.

- **Types of Knowledge Graphs**
- **Encyclopedic KGs**: general knowledge by consolidating information from diverse sources such as encyclopedias, databases, and expert insights.
  - Wikidat, dbpedia
- **Common-sense KGs:** Focused on everyday knowledge, these KGs encompass information about objects, events, and their relationships.
  - ConceptNet
- **Domain-Specific KGs:** Tailored to specific fields like medicine, finance
  - UMLS in the medical domain
- **Multi-Modal KGs:** these KGs incorporate images, sounds, and videos, serving purposes such as image-text matching or visual question answering.
- IMGpedia and MMKG seamlessly blend textual and visual information for comprehensive knowledge representation.



Possono essere usati in RDF, partendo da del testo.

Posso anche usarlo per arricchire un LLM.

Strumenti

Ci sono due strumenti principali: LLAMAINDEX

???

ha meccanismi di indicizzazione su vector, dove si possono avere vettori più generali e vettori più specifici per le architetture RAG.

??

LANGCHAIN

...