Ingegneria dei Dati HW1

Integrazione LLM nei casi studio

Gruppo:
- Pietro Barone
- Alessandro di Mario
- Guido Petrolini
- Douglas Ruffini

Caso studio: Open Ceres

Ha l'obiettivo di estendere il tuo Knowledge Graph (KG) con le informazioni contenute nelle pagine web dedicate alle entità del grafo

Fase di annotazione:

Annoti pagine web di input usando la conoscenza del tuo KG per scoprire le informazioni rilevanti nella pagina

1. Supponi di conoscere l'entità nella pagina web e cerchi nel DOM le informazioni che hai nel KG
2. Per ogni oggetto trovato cerchi un'etichetta candidata tra i vari elementi a lui vicini nella pagina
3. Scegli la label giusta (potrebbe differire per lo stesso oggetto tra pagina web e KG) con un dizionario che scrivi a mano per ogni predicato nel KG

Fase di estrazione:

Dalle posizioni delle coppie oggetto-label nella pagina web crei delle nuove coppie per espandere il tuo KG

1. Crei un grafo dei candidati con nodi formati da coppie oggetto-label per ogni coppia di stringhe nella pagina e metti degli archi tra i nodi per la loro similarità (se due label o oggetti condividono font, colore, grandezza…). Facendolo per ogni nodo estratto hai un grafo completo
2. Dalla fase di annotazione hai ricavato uno o più nodi corretti ed è verosimile che altri nodi corretti abbiano alta similarità con loro. Con algoritmi di label propagation ricrei delle community di coppie oggetto-label simili tra loro e quindi corrette

Utilizzo degli LLM in Open Ceres (1)

Possibili punti di integrazione di un LLM nel sistema OpenCeres:

Generazione di sinonimi per i predicati nel KG

Per gli elementi che trovi sia nel DOM della pagina che nel tuo KG ma con label diverse e che non hai nel dizionario dei sinonimi, generi una lista di sinonimi (multilingua magari?)
Input: Label nel KG
Output: Lista di predicati
Costo e latenza: basso (input e output di piccole dimensioni), lo fai in pochi casi
Benefici: tieni aggiornato il tuo dizionario di sinonimi delle label in modo automatico

Validazione finale delle coppie oggetto-label

Validazione delle coppie oggetto-label ottenute nella prima fase di annotazione
Input: coppia label-oggetto estratta dalla pagina web
Output: true/false se la coppia è corretta oppure no
Costo e latenza: basso ma se lo fai per tutte le coppie estratte diventa medio-alto
Benefici: rimuove errori di mapping e mantiene ordinato il dizionario dei sinonimi

Utilizzo degli LLM in Open Ceres (2)

Possibili punti di integrazione di un LLM nel sistema OpenCeres:

Filtraggio dei nodi candidati

Filtraggio delle coppie oggetto-label presi come nodi candidati nella fase di estrazione
Input: Coppia oggetto-label del nodo candidato
Output: true/false se la coppia è sensata oppure no
Costo e latenza: basso ma se lo fai per tutte le coppie estratte diventa medio-alto
Benefici: riduci in cascata tutto il proseguo dell' algoritmo di label propagation perché hai meno istanze

Verifica pre-inserimento nel KG

Verifica finale prima di aggiungere le nuove coppie nel KG, verifica solo semantica non di veridicità (es. Budget deve essere numerico)
Input: coppia label-oggetto
Output: true/false se la coppia è corretta oppure no
Costo e latenza: basso ma se lo fai per tutte le coppie estratte diventa medio-alto
Benefici: correggi errori grossolani e non popoli il KG con cose errate (errori della pagina web)

Caso studio: WEIR

Sfrutti le omogeneità locali e globali per correggere le ridondanze a livello di schema e istanza. Ha l'obiettivo di generare wrapper per ogni sorgente, allineare i vari schema e trovare delle label esplicative per i dati che hai

Ridondanze:

- A livello di schema se si usano nomi o approssimazioni/unità di misura diverse per stessa entità
- A livello di istanza se la trovo in più fonti

In generale immagini l'esistenza di un'unica grande tabella H con un set di fonti = {s1, s2, …}

- Ogni sorgente pubblica solo una vista di H
- Ogni sorgente usa subset di attributi/colonne (proiezione)
- Ogni sorgente usa subset di istanze (selezione) e potrebbe aggiungere errori

Metodologia:

- Genero regole a partire da tutti i punti testuali fissi del DOM nei vari template
- Valido le regole confrontandole con dati estratti da altre regole per altre pagine con una distance function (se tirano fuori cose simili probabilmente sono delle regole corrette)

Utilizzo degli LLM in WEIR

Possibili punti di integrazione di un LLM nel sistema WEIR:

Verifica di dominio

Fai analizzare al LLM la pagina per fargli valutare la semantica e farti dire se rientra nel dominio da te scelto. La usi solo per fonti meno fidate e più rumorose
Input: snippet HTML (da ottimizzare per dargliene meno possibile)
Output: true/false se la pagina rientra nel tuo dominio
Costo e latenza: medio-alto, per questo da fare su poche sorgenti
Benefici: riduce il pool di pagine da cui generare regole snellendo tutto il resto dell' algoritmo

Risoluzione conflitti

Risolve conflitti tra unità di misura o approssimazioni basandosi sulla coerenza semantica. Lo fai definendoti un convertitore con regole scritte a mano e, non potendo gestire tutti i casi, usi l'LLM solo come ultima spiaggia
Input: stringhe conflittuali e un minimo di contesto della pagina
Output: stringa che risolve il conflitto
Costo e latenza: medio
Benefici: migliori la qualità dei dati e riduci errori dovuti a formati diversi e inconsistenti

Latenza e costi dei task

Questa slide presenta un confronto empirico dei costi monetari e delle latenze per i task di estrazione dati analizzati. È importante sottolineare che i valori riportati sono stime molto approssimative, calcolate per fornire un ordine di grandezza sull'impatto computazionale e valutare la fattibilità di queste integrazioni in scenari di produzione reali.

Task	Token Input	Token Output	Costo (1.000 call)	Latenza stimata
Verifica dominio	550	2 (true/false)	~$0.08	~0.4s
Risoluzione conflitti	150	20 (stringa che risolve conflitto)	~$0.03	~0.6s
Generazione sinonimi	55	50	~$0.04	~0.8s
Validazione finale	55	2 (true/false)	~$0.01	~0.3s
Filtraggio nodi	55	2 (true/false)	~$0.01	~0.3s
Verifica pre-inserimento	55	2 (true/false)	~$0.01	~0.3s

GPT-4o-mini
È un modello proprietario di grandi dimensioni ma fortemente ottimizzato per essere rapido ed economico. I prezzi base sono di circa $0.15 per 1 milione di token in input e $0.60 per 1 milione in output.

Task	Token Input	Token Output	Costo (1.000 call)	Latenza stimata
Verifica di dominio	550	2 (true/false)	~$0.025	~0.15s
Risoluzione conflitti	150	20 (stringa che risolve conflitto)	~$0.006	~0.20s
Generazione di sinonimi	55	50	~$0.004	~0.25s
Validazione finale delle coppie	55	2 (true/false)	~$0.0004	~0.10s
Filtraggio coppie nodi candidati	55	2 (true/false)	~$0.0004	~0.10s
Verifica pre-inserimento nel KG	55	2 (true/false)	~$0.0004	~0.10s

Llama 3 8B
È una soluzione generativa molto più leggera e snella, dotata di soli 8 miliardi di parametri. I costi sono di circa $0.05 per 1 milione di token in input e $0.08 in output.