Progetto FLC – Anno Accademico 2023-2024 – Team Decompiler2.0

INDICE

1. Introduzione 2

1.1 Obiettivi del progetto 2

1.2 Stack tecnologico e dataset 2

2. Sviluppo del progetto 4

2.1 Large Language Models (LLM) 4

2.1.1 Caratteristiche dei Modelli LLM 4

2.1.2 Punti di Forza 4

2.1.3 Limitazioni 4

2.2 Scelta del Modello 5

2.2.1 Modifica del modelfile 5

2.3 Prompt engineering 6

2.3.1 Few Shot 6

2.3.2 Chaining 7

2.3.3 RAG 7

2.4 Sviluppo Estensione 7

2.4.1 Integrazione LLM 8

3. Conclusioni 8

3.1 Analisi dei risultati ottenuti 8

3.2 Analisi dei limiti 9

3.3 Sviluppi Futuri 9

Riferimenti: 10

1. Introduzione

Oggi, in un'era digitale come la nostra, si pone sempre più l'attenzione alla tutela dei dati personali che circolano in rete. Infatti, i nostri dati vengono coinvolti costantemente ogni volta che navighiamo su un sito web, utilizziamo un'app o acquistiamo online un prodotto. Dalle informazioni di navigazione alle nostre preferenze personali, i dati rappresentano un tesoro per le aziende, ma al tempo stesso anche un rischio per la nostra privacy. È qui che entrano in gioco le privacy policy, ovvero una rappresentazione documentata fondamentale che fa da intermediario tra l'azienda, il sito o la pagina e l'utente, fornendo a quest'ultimo tutte le informazioni riguardanti la raccolta, l'uso e la conservazione dei suoi dati personali. Ciononostante, le privacy policy sono scritte in un linguaggio complesso, legale, e difficilmente comprensibile per l'utente medio senza un'analisi approfondita. Sulla base di ciò è molto utile avere a disposizione uno strumento che consenta all'utente di comprendere velocemente le informazioni chiave contenute in questi documenti e questo progetto si inserisce in questo contesto proponendo una soluzione tecnologica che mira a semplificare l'analisi delle privacy policy mediante l'utilizzo di un'estensione Firefox.

1.1 Obiettivi del progetto

L'obiettivo principale del progetto è stato lo sviluppo di un'estensione Firefox che consente agli utenti di analizzare, riassumere e categorizzare in maniera automatica le privacy policy dei siti web che visitiamo. In particolare, gli obiettivi proposti per il progetto sono:

- Studiare e suggerire quali caratteristiche di una pagina Web di privacy policy sono essenziali da rilevare.
- Creare un'estensione JavaScript che possa eseguire questa analisi in maniera automatica e precisa, restituendo informazioni utili e di semplice interpretazione all'utente finale.
- Dare la possibilità di scaricare, in formato json, tutto ciò che è stato raccolto e analizzato automaticamente durante la navigazione web.

1.2 Stack tecnologico e dataset

La parte saliente del progetto, oltre allo sviluppo dell'esenzione, si è concentrata sull'analisi automatica delle privacy policy, per la quale abbiamo considerato diversi approcci tecnologici, ciascuno valido e con molteplici punti di forza e limitazioni:

1. Approcci sintattici
2. Approcci semantici (distributional semantics, latent semantic analysis, entity recognition)
3. Approcci basati sul deep learning
4. Approcci basati sull'LLM (large language models)

Tutti questi metodi rappresentano un insieme di strategie avanzate per affrontare l'analisi delle policy in modo efficace e dettagliato.
L'approccio che abbiamo identificato come utile e interessante approfondire, e implementare nella nostra estensione è quello basato sui Large Language Models (LLM), integrato con tecniche avanzate di prompt engineering per "allenare" o condizionare le risposte del modello alle nostre necessità.

Il dataset utilizzato, al fine di analizzare e valutare la bontà delle analisi del modello è "OPP-115" (Online Privacy Policy – set di 155 testi), sviluppato specificamente per l'analisi delle privacy policy dei siti web. Il suo obiettivo principale è identificare e categorizzare le pratiche di gestione dei dati personali descritte negli accordi di privacy policy. Questo dataset comprende 115 testi di privacy policy selezionate per rappresentare una gamma diversificata di settori e pratiche. Abbiamo utilizzato una parte di queste policy per testare l'efficacia e la pertinenza delle risposte generate dall'estensione.
Le privacy policy presenti nel dataset sono state accuratamente selezionate da un team di esperti, seguendo uno schema che ricerca in ogni testo piccoli estratti al fine di riassume e definire come il sito gestisce ogni punto della lista seguente:

- First Party Collection/Use
- Third Praty Sharing/Collection
- User Choice/Control
- User Access, Edit, & Deletion
- Data Retention
- Data Security
- Policy Change
- Do Not Track
- International & Specific Audiences
- Other

Queste categorie sono state fondamentali per strutturare e valutare in modo sistematico i testi estratti durante la navigazione web al fine di esaminare le privacy policy.

2. Sviluppo del progetto

2.1 Large Language Models (LLM)

I modelli LLM, o Large Language Models, sono una classe di modelli di intelligenza artificiale basati su architetture di reti neurali profonde, specificatamente progettati per comprendere e generare testo umano. Questi modelli sono allenati su vasti dataset di testo e sono in grado di elaborare e produrre risposte coerenti a partire da input testuali, grazie alla loro capacità di apprendere la struttura e il significato del linguaggio naturale. Degli esempi di modelli molto conosciuti sono GPT-4 (Sviluppato da OpenIA), BERT (Sviluppato da Google) e molti altri.

2.1.1 Caratteristiche dei Modelli LLM

Questi modelli hanno diverse caratteristiche importanti che li distinguono e identificano.

- Dimensionalità e Complessità: sono caratterizzati da un numero elevato di parametri, che possono variare da centinaia di milioni a decine di miliardi.

- Pre-Addestramento e Fine-Tuning: Gli LLM vengono prima pre-addestrati su grandi quantità di dati non supervisionati e successivamente perfezionati (fine-tuned) su dataset specifici per compiti particolari. Questo approccio migliora la loro versatilità e precisione in applicazioni specifiche.

- Capacità di Generalizzazione: Grazie alla vasta quantità di dati su cui sono stati addestrati, gli LLM possono generalizzare su una vasta gamma di compiti, anche su quelli per i quali non sono stati specificatamente progettati.

2.1.2 Punti di Forza

- Versatilità: I modelli LLM sono in grado di affrontare una vasta gamma di compiti di elaborazione del linguaggio naturale, come traduzione, riassunto, risposta a domande, e generazione di testo.

- Risposte Contestuali: Possono generare risposte che tengono conto del contesto fornito, rendendo le interazioni più naturali e coerenti.

- Capacità di Apprendere nuove Informazioni: Attraverso il processo di fine-tuning, i modelli possono essere adattati a nuovi domini o compiti specifici, migliorando la loro performance in contesti specializzati.

2.1.3 Limitazioni

- Risorse Computazionali: A causa della loro complessità e dimensione, gli LLM richiedono ingenti risorse computazionali per l'addestramento e l'inferenza, rendendoli inaccessibili senza infrastrutture adeguate.

- Comprensione Superficiale: Nonostante la loro capacità di generare testo coerente, gli LLM non comprendono realmente il significato delle parole come farebbe un essere umano. La loro comprensione è basata su correlazioni statistiche piuttosto che su una vera comprensione semantica.

- Bias e Dati di Addestramento: Gli LLM possono riflettere e amplificare i bias presenti nei dati su cui sono stati addestrati. Questo può portare a risposte inappropriate.

2.1.4 Ollama

Ollama è una piattaforma open-source avanzata per l'intelligenza artificiale, progettata per semplificare la gestione e l'utilizzo di modelli linguistici di grandi dimensioni (LLM) direttamente su dispositivi locali. La piattaforma offre un'ampia gamma di modelli pre-addestrati, come llama 3.1 e Gemma2, che possono essere impiegati immediatamente o personalizzati per soddisfare specifiche esigenze operative. Ollama è stata sviluppata con l'obiettivo di facilitare l'integrazione con diverse tecnologie e applicazioni, eliminando la complessità tipica dei processi di configurazione. Oltre all'utilizzo di modelli preesistenti, la piattaforma consente agli utenti di creare modelfile personalizzati, offrendo la flessibilità di adattare i modelli standard per applicazioni particolari. Questa funzionalità avanzata permette di ottimizzare le prestazioni dei modelli, rendendoli più efficaci per compiti specifici e garantendo soluzioni su misura in base alle necessità del progetto.

2.2 Scelta del Modello

Inizialmente avevamo deciso di utilizzare come modello llama3.1 da 8B di parametri che ci è sembrato sin da subito perfettamente in linea con le nostre necessità. Tuttavia, dopo aver effettuato una serie di test sul nostro hardware, ci siamo resi conto che il tempo di risposta risultava essere troppo lungo (dai 6 ai 20 minuti per i compiti più complessi). Questa è un'importante limitazione di cui tener conto in fase progettuale, abbiamo deciso allora di procedere con il modello gemma2:2b formato da 2B di parametri. Questo modello è risultato sin da subito più veloce con una perdita di precisone tuttavia trascurabile per le nostre necessità.

2.2.1 Modifica del modelfile

Un Modelfile è un file di configurazione che consente di personalizzare il comportamento di un modello linguistico di grandi dimensioni (LLM). Questo file è essenzialmente uno script o un set di istruzioni che definisce come il

modello dovrebbe rispondere a determinati input, quali parametri utilizzare durante l'elaborazione e come gestire le risposte.

Si tratta quindi di un documento ben strutturato che permette agli utenti di configurare e personalizzare modelli di intelligenza artificiale in modo molto preciso, settando e definendo una serie di parametri e direttive che influenzeranno il funzionamento del modello durante l'interazione e la risposta. Questi parametri sono ben documentati per i modelli e Ollama supporta questo tipo di customizzazione. Quelli più rilevanti approfonditi al fine del progetto sono elencati qui di seguito:

- Personalizzazione del comportamento: è possibile adattare lo standard di risposta del modello alle specifiche esigenze del progetto, garantendo che risponda in un modo che sia più allineato agli obiettivi operativi.

- Creazione di output strutturati: se si ha la necessità che il modello generi l'output in un formato specifico (come JSON, XML, o qualsiasi altro schema), il Modelfile può definire queste strutture, assicurando che le risposte siano sempre formattate correttamente. Ollama implementa un sistema di Template per la risposta, che tuttavia in questo progetto non è stato utilizzato perché ritenuto complesso e di difficile implementazione, essendo legato alla codifica in linguaggio Go-Template a noi attualmente sconosciuta.

2.3 Prompt engineering

Il prompt engineering è una disciplina emergente che si occupa di sviluppare e ottimizzare prompt per sfruttare al meglio i modelli linguistici in un'ampia varietà di applicazioni e contesti di ricerca. Il prompt engineering può essere usato per potenziare le prestazioni dei modelli LLM in compiti sia semplici che complessi, come la risposta a domande e il ragionamento aritmetico. Abbiamo studiato, testato e verificato diverse tecniche di prompt, di seguito elenchiamo le tre più interessanti che sono state implementate nello sviluppo progettuale.

2.3.1 Few Shot

La tecnica del Prompt Few-Shot consente al modello di adattarsi a compiti specifici con pochi esempi di riferimento. Questo è particolarmente utile quando si lavora con dati non strutturati, come il testo di una pagina web, dove la variabilità linguistica è alta. Abbiamo così la possibilità di personalizzare il comportamento del modello senza dover ricorrere a

grandi dataset specifici per il compito, rendendolo versatile e adattabile a diversi contesti.

2.3.2 Chaining

La tecnica di prompt chaining consiste nell'utilizzare più prompt in sequenza facendo in modo che l'output di un prompt diventi l'input per il successivo. Questo permette di suddividere un problema complesso in sotto-problemi gestibili. Questa tecnica risulta utile nel caso in cui si hanno compiti complessi che in genere l'LLM fatica a risolvere, in questo modo ogni sotto compito verrà affrontato con un prompt separato e si giungerà alla risposta finale. Ne consegue che uno dei vantaggi, oltre alla semplificazione delle richieste complesse, è l'ottenere una risposta finale più precisa grazie a questa divisione e risoluzione a catena. Nel nostro caso abbiamo dieci prompt basati sulla risposta del primo. Questo consiste nell' analisi generale del testo della policy con l'obiettivo di estrapolare i punti chiave utili alla sua categorizzazione, effettuata negli step intermedi, e con un ultimo step che sintetizza tutte le informazioni degli step precedenti.

2.3.3 RAG

Questa tecnica combina tecniche di retrieval insieme a quelle di generation (generazione di testo) per migliorare la risposta dell'LLM. L'utilità di questa tecnica si evidenzia in particolar modo quando si lavora con modelli di grandi dimensioni che richiedono risposte molto accurate. In particolare, si accede a fonti di conoscenza/dati esterne per fornire risposte più dettagliate. Il funzionamento si articola in due fasi principali:
Fase di Retrieval in cui il sistema cerca nel database o nei documenti disponibili le informazioni pertinenti al prompt dato, utilizzando un modello di retrieval (es motore di ricerca).
Fase di generazione, una volta recuperate le informazioni rilevanti il modello usa queste informazioni insieme al prompt originale per produrre la risposta.

Questa tecnica ha il vantaggio di dare risposte precise e informazioni aggiornate basate su dati in tempo reale. Nel nostro progetto abbiamo utilizzato dieci prompt ognuno dei quali consiste in una domanda che fa riferimento ad ogni singola categoria evidenziata nel dataset.

2.4 Sviluppo Estensione

Il codice scritto segue le indicazioni standard per lo sviluppo di una estensione Firefox, ci siamo riferiti alla documentazione ufficiale seguendo le linee guida. L'estensione sviluppata fornisce diverse funzionalità interessanti che approfondiamo di seguito. Analisi automatica delle pagine: l'estensione esegue in background, durante la navigazione, l'analisi del testo, salvando i dati ottenuti in riferimento alle policy privacy nella cache del browser, rendendoli successivamente disponibili all'utente. Interfaccia grafica accattivante e semplice: abbiamo sviluppato un'interfaccia che in maniera

diretta e comprensibile restituisce all'utente un'enorme quantità di informazioni e analisi effettuate durante la navigazione. Abbiamo inoltre inserito la possibilità di veder graficate queste informazioni rispetto a tutte le altre raccolte fino a quel momento. Integrazione prompt engineering: abbiamo inserito tre pulsanti che effettuano automaticamente delle richieste al modello seguendo le tecniche di prompt engineering descritte in questa relazione. Questo è un interessante vantaggio che è possibile usare per valutare quanto il modello migliora con questi metodi. Oltre a questi punti che riteniamo essere i più interessanti, abbiamo inserito una serie di ulteriori funzionalità utili e intrigate per fornire un prodotto quanto più completo.

2.4.1 Integrazione LLM

Un ulteriore aspetto positivo di Ollama sono le sue API, queste facilitano l'integrazione dei modelli AI in diverse applicazioni, consentendo personalizzazione, addestramento e utilizzo dei modelli tramite richieste programmatiche. Le API sono progettate per risposte in tempo reale, ideali per applicazioni interattive come chatbot e assistenti virtuali. Inoltre, sono facilmente integrabili con sistemi esistenti tramite richieste HTTP standard e risposte in JSON, supportando l'elaborazione multilingue e in diversi domini. Ollama fornisce documentazione completa per facilitare l'uso e l'integrazione delle API, rendendole uno strumento potente e flessibile per sviluppatori che desiderano sfruttare l'intelligenza artificiale nei loro progetti.

Nel contesto del nostro progetto, l'API a cui viene inviata la richiesta è l'endpoint /api/generate, utilizzando il metodo POST. Questo metodo consente di inviare dati al server per la loro elaborazione. I parametri trasmessi includono il nome del modello, il prompt, e l'opzione stream. Quest'ultima, se impostata su false, come nel nostro caso, indica che si desidera ricevere una risposta completa dal server, anziché un flusso di dati continuo. Inoltre, il parametro temperature, regola il livello di creatività della risposta generata dal modello, mentre il formato della risposta è specificato come JSON. Infine, il parametro keep_alive, definisce la durata per cui la connessione deve rimanere attiva.

3. Conclusioni

3.1 Analisi dei risultati ottenuti

I risultati del progetto confermano il raggiungimento degli obiettivi stabiliti, dimostrando che l'estensione è in grado di analizzare automaticamente le

privacy policy con un livello di accuratezza soddisfacente. L'uso del dataset OPP-115, che include una vasta gamma di privacy policy, ha fornito una base solida per valutare l'estensione. Infatti, i test sono stati effettuati confrontando i risultati ottenuti dall'estensione con quelli presenti nel dataset e ciò ha dato riscontri positivi. Inoltre, l'impiego di tecniche di prompt engineering, come few-shot, chaining e Retrieval-Augmented Generation (RAG), ha notevolmente migliorato la precisione e l'efficacia dell'estensione.
L'integrazione di un modello personalizzato ha ulteriormente garantito la coerenza delle risposte generate, producendo output consistenti e utili. L'analisi quantitativa mostra che l'estensione è in grado di classificare correttamente le privacy policy secondo categorie predefinite, con una notevole accuratezza. Tuttavia, l'analisi qualitativa suggerisce che ci sono margini di miglioramento, specialmente in contesti con policy particolarmente complesse, lunghe o ambigue.

3.2 Analisi dei limiti

Il progetto, seppure bene formulato e sviluppato ha alcune limitazioni tecniche. Gli aspetti principali sono legati all'hardware utilizzato ma ci sono anche altri punti da considerare.

La potenza computazionale disponibile non è stata sufficiente per gestire in modo ottimale i modelli di grandi dimensioni come llama 3.1, abbiamo riscontrato tempi di lavorazioni troppo lunghi, limitando così la possibilità di utilizzare modelli di più efficienti e performanti. Un'altra delle principali limitazioni riguarda il prompt engineering: i dati di addestramento e i risultati delle sessioni di lavoro non vengono "memorizzate" una volta che la sessione è terminata. In pratica, ogni sessione con il modello è indipendente, il che significa che tutti i prompt e le risposte generate vengono "perse" al termine della sessione e durante la stessa superato un certo numero massimo di token, rendendo così altamente inefficienti queste tecniche su larga scala e per un uso continuativo. Un'altra limitazione o potenziale problema sta nel fatto che il modello di Ollama risponde in linguaggio naturale, che poi attraverso opportune funzioni viene trasformato in formato json rispetto alle necessità dell'estensione stessa. Queste funzioni risultano funzionare bene solo se il modello risponde seguendo un determinato standard, caratteristica che abbiamo cercato di "insegnare" in maniera definitiva con la modifica del modelfile. Tuttavia, abbiamo notato sperimentalmente che il modello, in alcuni casi, risponde con un formato che non segue il pattern, creando così malfunzionamenti nell'estensione e conseguente incapacità di valutare la pagina.

3.3 Sviluppi Futuri

Per superare le limitazioni attuali e migliorare l'efficacia, ci sono diverse direzioni promettenti che sarebbe molto interessante seguire:

1. L'utilizzo di hardware con capacità computazionali migliori renderebbe possibile l'utilizzo di un modello più performate, meglio addestrato e più preciso nel linguaggio. Avremmo dei conseguenti tempi di esecuzione ridotti e un incremento dell'usabilità per l'utente.
2. L'esplorazione di nuove tecniche di prompt potrebbe contribuire a migliorare ulteriormente la precisione e l'efficienza del sistema, tenendo però a mente le limitazioni intrinseche di queste tecniche in abito di sessioni e massimi token del contesto tenuti in considerazione per generare una risposta.
3. Una possibile soluzione e miglioramento da affiancare alle tecniche di prompt, sarebbe quella di effettuare un fine tuning del modello sfruttando il dataset a disposizione. Questo approccio, affiancato ad un hardware sufficientemente potente, potrebbe dare risultati molto positivi e migliorare di gran lunga la capacità del modello di classificare correttamente i testi in analisi. Si creerebbe così un primo modello specializzato nell'analisi delle privacy policy.
4. Un'ulteriore modifica che ridurrebbe la possibilità di incorrere in errori sarebbe quella di inserire nel modelfile un template di risposta, che formalizzi al modello la necessità di rispondere seguendo un determinato standard json. Questa modifica assicurerebbe una risposta più veloce, meglio gestita e sicuramente entro gli standard.

Tutte queste modifiche migliorerebbero drasticamente i risultati ottenuti, andando a creare un prodotto molto utile in contesti reali, che sarebbe possibile e interessante provare a distribuire sotto licenza.

Riferimenti:

- Swilson_acl_2016: https://usableprivacy.org/static/files/swilson_acl_2016.pdf
- Database OPP-115: https://www.usableprivacy.org/data
- Estensione Chrome di spunto: https://github.com/privacy-tech-lab/privee
- Ollama: https://github.com/ollama/ollama/tree/main
- Model gemma 2b: https://ollama.com/library/gemma2
- Modelfile: https://github.com/ollama/ollama/blob/main/docs/modelfile.md
- API Ollama: https://github.com/ollama/ollama/blob/main/docs/api.md
- Prompt engineering: https://www.promptingguide.ai/it

Il lavoro è stato svolto da team de-compiler2.0 composto da Angelo Milella e Ida Tufilli. Abbiamo trovato molto interessante e attuale l'analisi di questo argomento e di questa tematica, ringraziamo pertanto, il Professor. Floriano Scioscia per averci dato la possibilità di scoprire questa interessante realtà che vorremmo portare avanti nelle nostre carriere professionali e personali. Abbiamo scoperto il mondo,

inizialmente sconosciuto, dietro le estensioni Chromium e i sempre più attuali modelli LLM, che fanno e faranno parte sempre di più delle nostre vite.