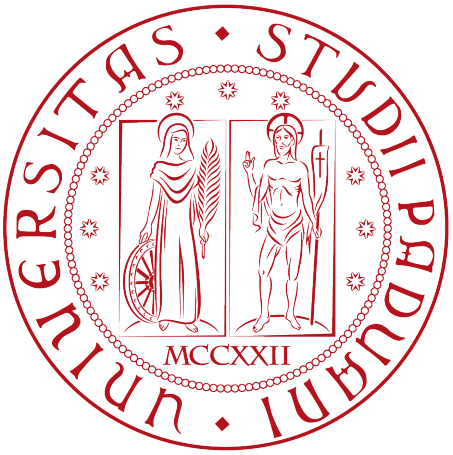
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Applicazione di GPT nei servizi di marketing di un'azienda commerciale

Tesi di laurea triennale

Relatore

Prof. Lamberto Ballan

Laureando

Davide Baggio

Anno Accademico 2022-2023

Uomini forti, destini forti. Uomini deboli, destini deboli.
Non c'è altra strada.

— *Luciano Spalletti*

Sommario

Il presente documento descrive l'esperienza di stage curricolare svolto dal laureando Davide Baggio presso l'azienda Wintech S.p.A., per una durata di circa trecentoventi ore complessive.

L'attività prevedeva la documentazione, analisi, progettazione e codifica di un'applicazione web, in forma di chatbot testuale. Quest'ultimo dev'essere in grado di rispondere in modo intelligente a domande poste in linguaggio naturale, estraendo informazioni dalla documentazione proprietaria aziendale.

Il seguente documento è diviso nei seguenti capitoli:

1. **Contesto aziendale**: viene introdotta l'azienda in cui è stato svolto lo stage, il loro business e i prodotti sul mercato;
2. **Fondamenti teorici**: al lettore vengono fornite delle nozioni sufficienti per fargli capire le tematiche trattate nei successivi capitoli;
3. **Svolgimento dello stage**: vengono descritte le suddivisioni temporali, gli obiettivi, i requisiti e le fasi affrontante durante il corso dello stage;
4. **Il progetto**: vengono elencate le tecnologie adottate, spiegati i casi d'uso e com'è stato implementato il software;
5. **Conclusioni**: viene fatto il bilancio degli obiettivi e dei requisiti con anche una considerazione personale sull'esperienza trascorsa.

Convenzioni tipografiche

Al fine di agevolare la lettura e la comprensione del documento, è stato introdotto un glossario: qualora un termine presente nel testo dovesse comparire in un contesto in cui si presume che il lettore possa non comprenderne il significato, esso verrà evidenziato con la dicitura [g] la prima volta che appare nella tesi.

Nel caso in cui si stia consultando la versione digitale di questo documento, è possibile accedere direttamente al termine nel glossario e/o alla lista degli acronimi, entrambi posizionati alla fine del documento, facendo clic sul termine contrassegnato dal colore azzurro.

I riferimenti a sezioni del documento sono contrassegnati dal simbolo §, seguito dal numero della sezione.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Contesto aziendale

1.1 L'azienda

Figura 1.1: Logo di Wintech S.p.A.

Wintech S.p.A., azienda fondata nel 1987 dal CEO Massimo Gallotta, è un consolidato System Integrator[g] specializzato nel settore dell'Information and Communications Technology (ICT)[g]. In totale l'azienda può contare su circa un centinaio di professionisti altamente qualificati, distribuiti tra Padova (sede principale), Milano e Bassano del Grappa.

Wintech, grazie all'esperienza maturata in questi 36 anni di attività, ha sviluppato strette partnership con importanti realtà nel campo informatico come IBM, Symantec, HP, Microsoft, Sophos e Oracle. Queste collaborazioni permettono all'azienda di offrire ai propri clienti, quali professionisti, imprese, Piccole-Medie Imprese (PMI)[g] pubblica amministrazione, banche e assicurazioni, delle soluzioni personalizzate e innovative che ottimizzano i processi aziendali e soddisfano le diverse esigenze del mercato.

L'azienda si impegna a comprendere appieno le richieste e le aspettative della sua clientela, così da poter fornire soluzioni su misura che si integrino nelle infrastrutture esistenti o ancora da implementare. Offre inoltre servizi di consulenza professionali, tra cui assistenza tecnica, gestione documentale, trasferimento di sistemi su cloud, design di Intranet aziendali, piattaforme di E-Learning e Cyber Security.

1.2 Servizi offerti

In questa sezione vengono descritti tutti i servizi e i prodotti software di Wintech presenti ad oggi sul mercato, con lo scopo di rendere chiaro il dominio applicativo aziendale e poter avere una panoramica generale su ciò che tratta l'azienda.

1.2.1 ERP e Applicazioni Gestionali

Gli Enterprise Resource Planning (ERP)[g] costituiscono uno dei pilastri centrali della gestione aziendale, integrando tutti i processi rilevanti e contribuendo a un'efficiente sincronizzazione delle attività interne grazie a soluzioni personalizzate. L'offerta di Wintech comprende anche gestionali per gli studi professionali, strumenti di tesoreria e soluzioni di business intelligence, il cui obiettivo è quello di migliorare, velocizzare e ottimizzare l'efficienza delle operazioni aziendali tramite l'utilizzo di un software completo e sempre aggiornato.

1.2.2 Digital Transformation

"La trasformazione digitale è quell'insieme di cambiamenti nei comportamenti aziendali e di business collegato e veicolato dalla tecnologia digitale, tramite la quale è possibile traguardare una maggiore competitività di mercato." [1]

Tale processo, sebbene spesso impegnativo, rappresenta una priorità essenziale per le aziende, compresa Wintech, che offre soluzioni avanzate per supportare le aziende nella loro trasformazione digitale, andando ad integrare sistemi che consentano di migliorare i processi di business tramite l'ottimizzazione e l'automatizzazione dei processi attuali, ma anche le loro future evoluzioni.

1.2.3 Cloud

Il Cloud Computing rappresenta una soluzione chiave per le aziende alla ricerca di servizi gestiti efficienti e sicuri, poiché consente di trasferire le risorse informatiche su infrastrutture esterne e altamente specializzate.
Uno dei principali punti a favore dei servizi offerti è la presenza della certificazione UNI CEI EN ISO/IEC 27001:2017, che garantisce standard di sicurezza elevati e un approccio rigoroso alla protezione dei dati aziendali. Attraverso il servizio di full outsourcing, Wintech mira a liberare risorse umane e finanziarie, consentendo alle aziende di concentrarsi sull'innovazione e sullo sviluppo del loro core business senza le limitazioni tipiche della gestione in-house, favorendo una maggiore agilità e scalabilità.

1.2.4 Security

Wintech adotta un approccio proattivo e multidisciplinare alla sicurezza informatica, investendo nella formazione del personale e implementando misure preventive

[1] *Digital Transformation.* URL: https://www.wintech.it/innoviamo/che-innovazione-cerchi/digital-transformation/.

avanzate per proteggere i dati sensibili e garantire la continuità delle attività aziendali. La collaborazione con partner specializzati nel settore della sicurezza informatica può fornire alle aziende la consulenza e le soluzioni necessarie per mitigare i rischi e affrontare le sfide sempre più complesse della sicurezza informatica.

1.2.5 E-Learning e Video Communication

Wintech fornisce anche servizi online di trasmissione video - sia in diretta che on-demand - incentrati sulla formazione a distanza, studiati in modo da poter favorire l'apprendimento, integrando feature che migliorano il rapporto con il relatore o docente.

1.3 Ricerca e Sviluppo

"La velocità di evoluzione del settore è il nostro più grande stimolo, anticiparne il futuro la nostra prerogativa, che cerchiamo sempre di coniugare con le reali necessità delle persone." [2]

Durante la mia permanenza in Wintech sono stato assegnato al reparto di Ricerca e Sviluppo (R&D), un'unità strategica essenziale per il progresso e l'innovazione del settore informatico. In particolare si occupa dell'analisi, progettazione e implementazione di nuove tecnologie, prodotti e servizi per soddisfare le esigenze del mercato e anticiparne le tendenze future. È quindi di fondamentale importanza garantire un vantaggio competitivo all'azienda attraverso l'introduzione di soluzioni all'avanguardia e il miglioramento continuo delle prestazioni e dell'affidabilità dei prodotti/servizi esistenti. Tra i punti cardine di questo ramo aziendale troviamo la sperimentazione e la creazione di prototipi Proof of Concept (PoC)[g] in tempi brevi, in modo tale da poter presentare al board dirigenziale un prodotto funzionante che dia un assaggio delle potenzialità che si è in grado di offrire per poter trarre delle conclusioni in merito.

1.4 Software utilizzati

Durante la mia esperienza in azienda, ho avuto modo di usare quotidianamente un buon numero di strumenti software diversi con lo scopo di agevolare:

- la comunicazione interna;
- la condivisione di file;
- lo sviluppo;
- il versionamento del codice.

[2] *Enrico Merigliano - Responsabile Ricerca e Sviluppo/Tutor interno dello stage.* URL: https://www.wintech.it/siamo/ricerca-e-sviluppo/.

In quanto stagista, i software da me impiegati differiscono sensibilmente da quelli in uso dagli altri dipendenti; ad esempio non mi è stata richiesta l'attività di consuntivazione giornaliera, l'utilizzo dei server di sviluppo/produzione e la repository[g] in GitLab. Fatta questa premessa, gli strumenti da me utilizzati sono i seguenti:

- **Microsoft Office 365**: è la suite di programmi proprietari di Microsoft incentrati sulla produttività, utilizzabili nelle versioni desktop/mobile e web. In particolare:
  - Outlook: client di posta elettronica contenente anche un calendario, agenda delle attività, note e contatti;
  - Teams: piattaforma incentrata sulla comunicazione rapida e la collaborazione all'interno di un gruppo di lavoro. Offre la possibilità di chattare, effettuare chiamate vocali e video, condividere in maniera persistente file o cartelle e la creazione di canali per ogni team e reparto interno all'azienda;
  - Word: software per la produzione di documenti testuali con il template creato da Wintech;
  - Excel: software per la creazione di fogli di calcolo;
  - Powerpoint: software per la produzione di presentazioni.
- **Visual Studio Code**: è un Integrated Development Environment (IDE)[g] di Microsoft finalizzato allo sviluppo di codice su un'ampia gamma di linguaggi di programmazione diversi grazie alla presenza di estensioni apposite. Oltre alle funzioni basilari offre anche il completamento del codice, il debugging, l'integrazione con Git, la presenza di uno o più terminali, la sincronizzazione in cloud delle impostazioni personali e molto altro ancora;
- **GitHub**: è una piattaforma di hosting di codice sorgente proprietaria di Microsoft, organizzato in repository, basata su Git. Tra le features più utili troviamo un Issue Tracking System (ITS)[g], la Continuous Integration (CI)[g] grazie alle Actions e tutto il sistema di collaborazione per lo sviluppo con i fork e le pull request;
- **Windows 11**: è un sistema operativo creato da Microsoft, presente nella maggior parte delle macchine in azienda e ben integrato con i software elencati precedentemente, in quanto tutti di sua proprietà.

Capitolo 2

Fondamenti teorici

In questo capitolo vengono esplorate, ed approfondite, le tecnologie utilizzate alla base per lo sviluppo del progetto, introducendo termini e nozioni su argomenti che verranno trattati nei capitoli successivi.

2.1 Large Language Model

I Large Language Model (LLM)[g], modelli d'intelligenza artificiale ad apprendimento profondo, costituiscono una pietra miliare nell'ambito dell'elaborazione del linguaggio naturale grazie alla loro abilità di comprendere, generare e manipolare il linguaggio umano. Per comprendere meglio cosa essi siano occorre scomporre il nome stesso:

- **Large** - significa che sono addestrati su insieme di dati di grandissime dimensioni e con miliardi, se non trilioni, di parametri;
- **Language** - operano principalmente con il linguaggio umano;
- **Model** - sono usati per trovare pattern o fare predizioni su dei dati.

Questa sezione si propone di esaminare l'architettura, il processo di addestramento e le applicazioni dei LLM, evidenziandone il ruolo che ricoprono nel panorama scientifico e industriale.

2.1.1 Architettura e funzionamento

I moderni LLM, in particolare i Generative Pre-trained Transformer (GPT)[g], si basano sull'architettura transformer, che introduce il concetto di auto-attenzione[g], permettendo al modello di catturare relazioni a lungo raggio tra le parole in un testo. La struttura di un LLM è caratterizzata da una pila di strati di trasformazione, ognuno dei quali contiene meccanismi di auto-attenzione e feed-forward[g]. Questa organizzazione consente ai LLM di catturare le dipendenze nel testo, consentendo loro di comprendere il contesto e generare testi coerenti e significativi.

Transformer[1]

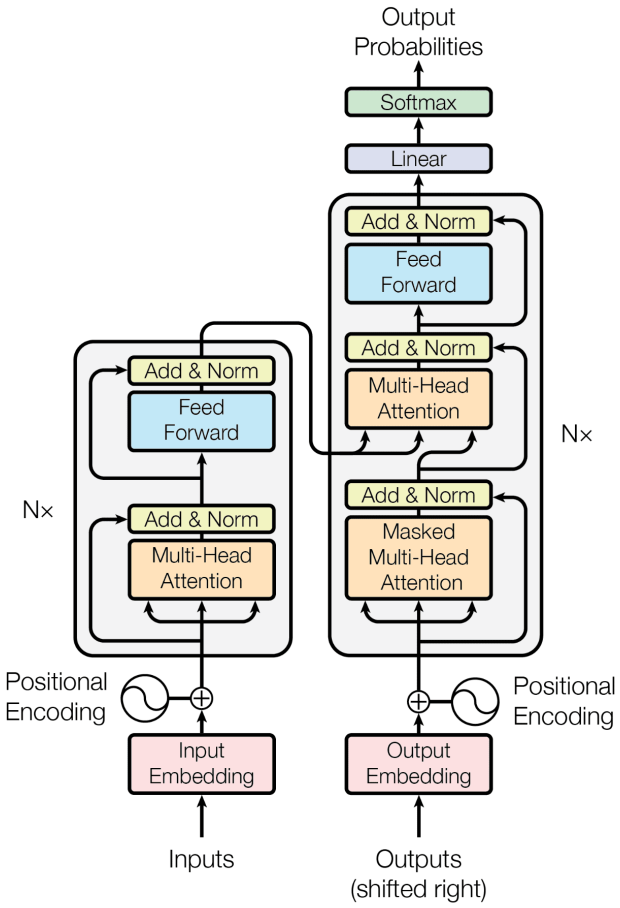


Figura 2.1: Architettura transformer

L'architettura di un transfomer è formata da due parti principali interconnesse: l'encoder e il decoder.
L'encoder (la parte a sinistra della figura 2.1) consiste in uno stack di $N = 6$ strati di trasformazione, ognuno di essi composto da due sottostrati:

1. il primo implementa un meccanismo di auto-attenzione multi head, che a sua volta implementa h teste che ricevono una versione linearmente proiettata delle query, chiavi e valori, ognuna di esse usata per produrre h output in parallelo utilizzati per generare il risultato finale;

2. il secondo è una rete feed-forward completamente connessa composta da due trasformazioni lineari con l'attivazione del Rectified Linear Unit (ReLU) tra di esse:

$$\mathrm{FFN}(x) = \mathrm{ReLU}(\mathrm{W}_1 x + b_1)\mathrm{W}_2 + b_2 \tag{2.1}$$

[1] A. N. Gomez e L. Kaiser e I. Polosukhin A. Vaswani e N. Shazeer e N. Parmar e J. Uszkoreit e L. Jones. «Attention Is All You Need». In: *arXiv:1706.03762* (2017). URL: https://arxiv.org/abs/1706.03762.

I sei strati prima citati applicano la stessa transformazione lineare su tutte le singole parole dell'input, ma ogni strato utilizza dei pesi $(\mathrm{W}_1, \mathrm{W}_2)$ e bias (b_1, b_2) diversi. Inoltre, ogni sottostrato è seguito anche da un livello di normalizzazione che normalizza la somma calcolata tra l'input del sottostrato (x) e l'output generato dal sottostrato stesso $(\text{sublayer}(x))$:

$$\text{layernorm}(x + \text{sublayer}(x)) \tag{2.2}$$

Per quanto riguarda il decoder (la parte a destra della figura 2.1), si può notare che condivide parecchie somiglianze con l'encoder. Anch'esso infatti consiste in una pila di $N = 6$ strati identici, ognuno composto da tre sottostrati:

1. il primo riceve l'output del precedente strato, lo arricchisce con informazioni di posizionamento e implementa l'auto attenzione multi-head su di esso. Mentre l'encoder è progettato per prestare attenzione a tutte le parole nella sequenza di input indipendentemente dalla loro posizione nella sequenza, il decoder è progettato per prestare attenzione solo alle parole precedenti, pertanto la previsione per una parola in posizione i può dipendere solo dagli output noti per le parole che la precedono nella sequenza. Nel meccanismo di attenzione multi-head, ciò è ottenuto introducendo una maschera sui valori prodotti dalla moltiplicazione scalata delle matrici Q, K. Questa mascheratura è implementata sopprimendo i valori delle matrici che altrimenti corrisponderebbero a connessioni non valide:

$$\text{mask}(\mathrm{QK}^T) = \text{mask}\left(\begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} \\ e_{21} & e_{22} & \dots & e_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e_{m1} & e_{m2} & \dots & e_{mn} \end{bmatrix}\right) = \begin{bmatrix} e_{11} & -\infty & \dots & -\infty \\ e_{21} & e_{22} & \dots & -\infty \\ \vdots & \vdots & \ddots & \vdots \\ e_{m1} & e_{m2} & \dots & e_{mn} \end{bmatrix} \tag{2.3}$$

2. il secondo implementa un meccanismo di auto-attenzione multi head simile a quello dell'encoder: riceve le query dal precedente sottostrato e le chiavi e i valori dall'output dell'encoder. Questo consente al decoder di occuparsi di tutte le parole nella sequenza di input;

3. il terzo ed ultimo strato implementa una rete feed-forward completamente connessa identica a quella del secondo sottostrato dell'encoder.

2.1.2 Processo di addestramento

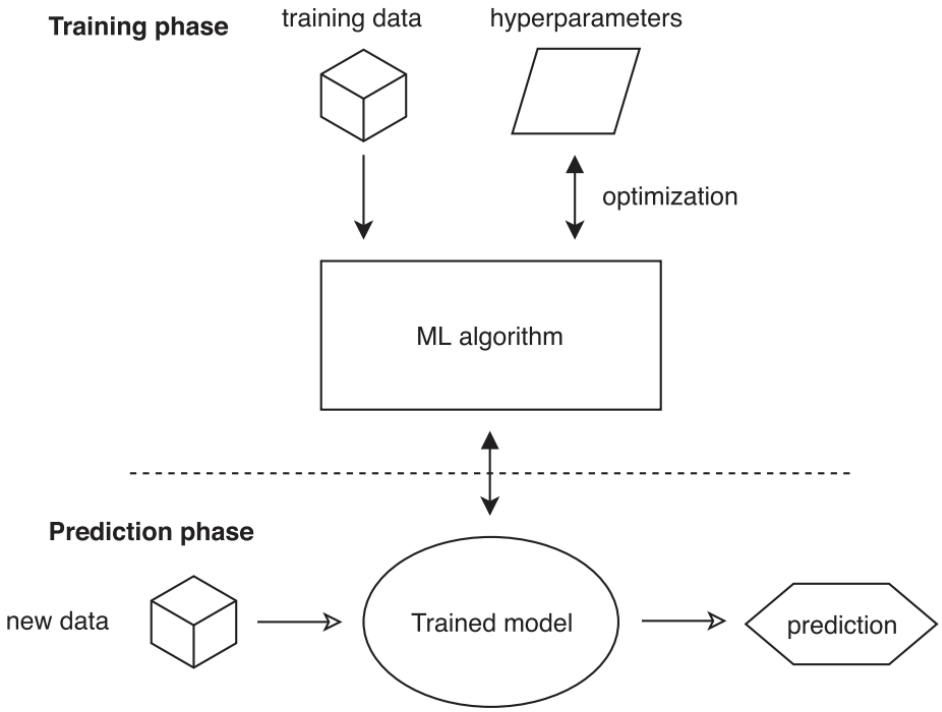


Figura 2.2: Processo di addestramento modelli di machine learning

Il processo di addestramento nei modelli di Machine Learning (ML)[g] costituisce un'essenziale passo nella loro creazione. Nonostante esistano diversi concetti e implementazioni dei vari modelli, è possibile identificare un'architettura comune costituita da due fasi: apprendimento (*training*) e ottimizzazione (*fine-tuning*).[2]

Training
Nella fase di training, il modello viene esposto a un vasto corpus di testi (si parla addirittura di decine di terabyte) e impara a modellare strutture linguistiche, i significati e le associazioni tra le parole. Sviluppa inoltre una rappresentazione profonda e generalizzata del linguaggio, apprendendo le relazioni e i pattern sottostanti ad esso. È importante durante l'addestramento fornire dei feedback sui risultati prodotti, così da migliorare gradualmente la qualità dell'output del modello stesso. Per far ciò esistono diverse tecniche:

- **Apprendimento supervisionato**: si forniscono al modello i dati di input e i corrispondenti valori di output desiderati/ottimali. È compito quindi dell'algoritmo di ML trovare una funzione in grado di mappare gli input verso gli output, cercando di minimizzare gli errori di bias, causati da semplificazioni e assunzioni eccessive, e la varianza, ossia quanto le previsioni prodotte da un modello cambiano al variare del training set;

[2] J. Verbraeken e M. Wolting e J. Katzy e J. Kloppenburg e T. Verbelen e J. S. Rellermeyer. «A Survey on Distributed Machine Learning». In: *ACM Computing Surveys* (2020). URL: https://dl.acm.org/doi/abs/10.1145/3377454.

- **Apprendimento non supervisionato**: a differenza della modalità precedente, al modello vengono forniti unicamente gli input senza etichette, con lo scopo di trovare informazioni intrinseche nei dati stessi e pattern nascosti;
- **Apprendimento semi-supervisionato**: vengono forniti al modello una piccola quantità di dati etichettati e un insieme di dati senza etichette di dimensione nettamente maggiore; tutto ciò ovvia alla problematica del reperimento di dati etichettati, in quanto costosi e molto laboriosi;
- **Apprendimento per rinforzo**: viene utilizzato per addestrare un agente che deve compiere azioni in un ambiente basandosi unicamente sulle sue osservazioni. La qualità di un azione è data da un valore numerico generato da una funzione di ricompensa, con lo scopo di incoraggiare comportamenti corretti da parte dell'agente.

Fine-tuning
Nella seconda fase, quella di fine-tuning, il modello viene addestrato su un dataset specifico di dimensioni ridotte - gli *hyperparameters* - in base all'applicazione desiderata, andando a raffinare le sue abilità e le prestazioni nel dominio di interesse. Più nel dettaglio, il fine-tuning implica la regolazione dei pesi dei parametri del modello pre-addestrato utilizzando un nuovo set di dati di addestramento, l'aggiunta o rimozione di strati, la regolazione delle funzioni di attivazione e degli hyperparameters con l'obiettivo di ottenere un equilibro tra la capacita del modello di generalizzare sui nuovi dati e il rischio di sovrapposizione o sottostima.

La fase di predizione e di addestramento non sono sempre mutuamente esclusive, in quanto con l'apprendimento incrementale il modello viene continuamente addestrato utilizzando nuovi dati generati da egli stesso.

2.1.3 Token

I token sono le unità di base del testo, sia esso in input o in output, che un LLM utilizza per elaborare e generare il linguaggio naturale. A seconda del metodo o schema di tokenizzazione adottato, i token possono essere caratteri, parole intere, sottostringhe o altri segmenti di testo o codice. Il loro utilizzo aiuta il modello a gestire diverse lingue, vocabolari e formati, nonché a ridurre i costi computazionali e di memoria.

Generazione dei token
Il processo che porta alla generazione dei token può essere effettuato utilizzando diversi metodi a seconda della complessità e della variabilità dei testi. Ad esempio, OpenAI utilizza un metodo di tokenizzazione subword chiamato Byte-Pair Encoding (BPE) per i loro modelli basati su GPT. Consiste nell'unire le coppie di caratteri o byte più frequenti in un singolo token, fino a raggiungere un certo numero di token o una dimensione del vocabolario prefissata. È importante sapere che ad ogni token

corrisponde un valore numerico che lo identifica, quindi l'insieme dei token viene successivamente disposto in uno o più vettori in grado di essere interpretato/i dal LLM. Quest'ultimo, per generare la risposta, non fa altro che apprendere la relazione statistica che li lega, in modo tale da poter produrre il prossimo token della sequenza.

Per apprendere meglio il concetto propongo un semplice ed efficace esempio, utilizzando il Tokenizer[3] di OpenAI: la parola di 8 caratteri *progetto* viene spezzata in tre token, ovvero *pro-get-to*, contraddistinti dai seguenti identificatori numerici [1676, 1136, 1462].

Funzione
I token vengono utilizzati in quanto possono aiutare il modello a gestire parole rare o non viste in precedenza, a creare rappresentazioni più compatte e coerenti dei testi e infine per consentire al modello di generare nuove parole (o meglio, token), combinando quelli già esistenti. Il modo in cui la tokenizzazione è diversa nei vari modelli proposti da OpenAI dipende dal numero di token o dalla dimensione del vocabolario che ciascun modello utilizza:

- Ada ha la dimensione del vocabolario più piccola - 50.000 token;
- Babbage - 57.000 token;
- Curie - 57.000 token;
- Davinci ha la dimensione del vocabolario più grande - 60.000 token.

Maggiore è la dimensione del vocabolario, più diversi ed espressivi possono essere i testi che il modello può generare. Tuttavia, maggiore è la dimensione del vocabolario, maggiori sono la memoria e le risorse computazionali richieste dal modello; pertanto la scelta della dimensione del vocabolario dipende dal compromesso tra la qualità e l'efficienza del modello.

Lunghezza del contesto
I LLM sono in grado di gestire una certa quantità di token per interazione: la somma dei token del prompt e quella del testo generato dal modello non deve superare la cosiddetta "lunghezza del contesto", che varia da modello a modello.
Riportando sempre come esempio alcuni prodotti di OpenAI, abbiamo i seguenti valori:

- Ada - 2.049 token;
- Babbage - 2.049 token;
- Curie - 2.049 token;
- Davinci - 4.097 token.

[3] https://platform.openai.com/tokenizer

Nel caso in cui la lunghezza del prompt fosse troppo lunga, il modello genera ugualmente una risposta, tuttavia questa sarà incompleta e troncata una volta raggiunta la soglia massima consentita. È bene quindi tenere in considerazione questo fattore quando si utilizzano i modelli LLM per evitare esperienze d'uso negative o poco efficaci.

2.1.4 Applicazioni trasversali

Le applicazioni dei LLM coprono una vasta gamma di utilizzi che spaziano tra diverse discipline e settori, sfruttando la potenza dell'apprendimento automatico per risolvere problemi complessi e migliorare l'efficienza produttiva in svariate attività. Di seguito propongo un elenco di reali casi d'uso ed applicazioni di modelli d'intelligenza artificiale per fornire un assaggio di cosa sono in grado di offrire alla società al giorno d'oggi:

- **Motori di ricerca**: migliorare i risultati forniti dai motori di ricerca cogliendo il reale intento e scopo che ha spinto l'utente ad effettuare tale operazione, tutto questo grazie alla capacità dei LLM di interpretare il linguaggio naturale. Attualmente sia Google Search che Microsoft Bing hanno implementato tale tecnologia per offrire un servizio migliore ai loro utilizzatori;

- **Generazione di contenuti**: generare contenuti basati sugli input forniti dall'utente è uno dei casi d'uso più comuni, l'obiettivo è aumentare la produttività dei lavoratori o, in alcuni casi, eliminare del tutto la necessità di coinvolgere un essere umano in un determinato processo, qualora l'attività in questione sia sufficientemente semplice. In questa categoria rientrano ad esempio la creazione e il riassunto di testi, la scrittura di codice, la generazione di immagini, ecc.;

- **Estrazione**: data una grande quantità di dati non strutturati, i LLM sono in grado di estrarre informazioni e riorganizzarle in formati ben definiti. Alcuni processi che beneficiano maggiormente di questa funzionalità sono ad esempio la classificazione dei testi, il clustering, il monitoraggio dei social media, l'analisi di dati biomedici, ecc.;

- **Question answering**: è una combinazione della funzione di ricerca e sintesi, inizialmente il modello comprende ciò che l'utente sta richiedendo e restituisce di conseguenza un insieme di informazioni rilevanti, successivamente invece riassume tutte le informazioni del contesto per restituire un'unica risposta coerente. Questa funzionalità porta notevoli benefici ai chatbot e gli assistenti virtuali.

2.1.5 Criticità

Nonostante i LLM stiano diventando una realtà ormai sempre più consolidata al giorno d'oggi, è innegabile constatare il fatto che sono ancora lontani dal rappresentare un prodotto maturo, testato e sicuro. In questa sezione viene fatta una disamina sulle problematiche che derivano da questa tecnologia.

Interpretabilità

Figura 2.3: Schema di un sistema black box

L'interpretabilità dei modelli di ML è una caratteristica cruciale che riguarda la capacità di comprendere e spiegare tutte le decisioni prese da quest'ultimi, in quanto la complessità intrinseca che li caratterizza rende estremamente difficile per l'uomo comprendere il motivo per cui vengono effettuate determinate decisioni. Questo solleva questioni fondamentali riguardo all'affidabilità, all'etica e alla responsabilità dell'uso di questa tecnologia, soprattutto quando viene utilizzata per scopi commerciali. Esiste infatti un settore dedicato, chiamato Explainable Artificial Intelligence (XAI), che si occupa di dare spiegazioni razionali e scientifiche sulle azioni intraprese da un modello black box[g].

Conoscenza derivata dall'addestramento

L'addestramento dei modelli di ML con dati reali, comporta un importante limitazione nella conoscenza acquisita, la quale può condurre a diverse implicazioni negative e a sfide impegnative per ovviare a tale problematica.
Tra queste, emerge in primo luogo la criticità dell'obsolescenza dei dati, derivata dall'evoluzione costante del panorama informazionale: i modelli non sono quindi in grado di considerare nuovi sviluppi, tendenze o eventi successivi alla data di addestramento. La mancanza di contestualizzazione attuale potrebbe comportare la produzione di risposte inaccurate, o inappropriate, quando i modelli vengono sollecitati a trattare tematiche di attualità. In aggiunta, la mutevolezza del linguaggio e delle dinamiche culturali costituisce un altro aspetto critico, in quanto, nel corso del tempo, l'evoluzione del linguaggio e dei valori culturali può portare all'emergenza di nuovi termini, espressioni e sfumature di significato. Modelli formati su informazioni datate potrebbero incontrare difficoltà nell'identificare e comprendere tali cambiamenti, con conseguenze quali fraintendimenti o rappresentazioni fuorvianti del testo.
In una prospettiva più ampia, l'utilizzo di dati reali nell'addestramento può risultare in un'influenza di bias e in una rappresentazione sbilanciata delle informazioni. Questi dati possono riflettere e amplificare i bias culturali, di genere, razziali e altri pregiudizi presenti nella società durante quel periodo specifico. Ciò può portare il

modello a internalizzare e perpetuare tali bias, con conseguenti effetti negativi sulle risposte e le decisioni che genera.
Infine, la limitazione della conoscenza basata sui dati utilizzati per l'addestramento può arrecare importanti conseguenze sulla capacità di generalizzazione del modello, il quale potrebbe risultare meno versatile nell'applicare ciò che ha appreso a contesti nuovi o differenti da quelli contemplati nei dati di addestramento. Questo concetto riguarda in particolare:

- **Overfitting**: si verifica quando il modello è troppo complesso e si adatta in maniera eccessiva ai dati di addestramento, catturando anche il rumore casuale presente nei dati. Questo porta ad un'eccessiva complessità del modello, che può memorizzare gli esempi di addestramento invece di catturare i pattern sottostanti per imparare a generalizzare. Di conseguenza, il modello si comporta eccezionalmente bene sui dati di addestramento, ma le sue prestazioni su nuovi dati sconosciuti sono scarse;

- **Underfitting**: si verifica quando un modello non è in grado di catturare correttamente le relazioni complesse presenti nei dati di addestramento; in altre parole, il modello è troppo semplice per rappresentare i dati in modo accurato. I segni tipici che lo contraddistinguono sono prestazioni mediocri, sia sui dati di addestramento che su quelli di test, e una curva di apprendimento che converge rapidamente ad un errore elevato.

Allucinazioni
Con in termine allucinazioni ci si riferisce a situazioni in cui il modello genera risultati all'apparenza corretti, ma che in realtà non hanno alcun fondamento sui dati di addestramento o sulla realtà. Nei modelli predittivi possono verificarsi quando un modello ha eccessiva flessibilità e/o troppi parametri, portando a previsioni che si basano su pattern casuali o rumorosi nei dati di addestramento, causando una cattiva generalizzazione e a prestazioni scadenti su nuovi dati.
Le allucinazioni possono essere particolarmente problematiche in contesti in cui le decisioni o le azioni sono guidate dalle previsioni del modello, come nell'elaborazione del linguaggio naturale, nella visione artificiale, nella guida autonoma e in molte altre applicazioni.

2.2 Word embedding

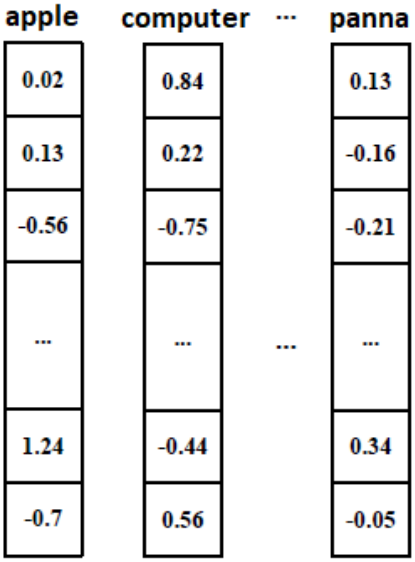


Figura 2.4: Esempio di word embedding

Nell'ambito dell'elaborazione del linguaggio naturale, il word embedding[g] è una tecnica che trasforma un testo in vettori numerici continui di dimensione fissa in uno spazio multidimensionale, catturando le relazioni semantiche e sintattiche che legano le parole in un determinato testo.
Questa sezione si prefigge di esaminare in modo approfondito e esaustivo il word embedding, esplorando le sue metodologie di base, i modelli avanzati e le sue applicazioni in varie sfere. [4]

2.2.1 Metodologie

Esistono diverse metodologie per generare le rappresentazioni in forma vettoriale delle parole, e possono essere classificate in due grandi categorie.

Metodi basati sulla predizione
Questi modelli cercano di prevedere una parola target (o un token) principale all'interno del testo in base alle parole circostanti che costituiscono la frase.
Un esempio noto di un modello di word embedding basato sulla previsione è il *Word2Vec* [5], ossia una rete neurale feedforward a due strati in grado di prevedere una parola target data una finestra di parole circostanti nel contesto. Esistono due approcci distinti che il modello può utilizzare per produrre gli embeddings:

1. **Continuous Bag of Words** (CBOW): viene utilizzato il contesto (cioè le parole circostanti) per prevedere la parola target. Si considera una finestra di

[4] F. Almeida e G. Xexéo. «Word Embeddings: A Survey». In: *arXiv:1901.09069* (2019). URL: https://arxiv.org/abs/1901.09069.
[5] T. Mikolov e K. Chen e G. Corrado e J. Dean. «Efficient Estimation of Word Representations in Vector Space». In: *arXiv:1301.3781* (2013). URL: https://arxiv.org/abs/1301.3781.

parole circostanti e si cerca di massimizzare la probabilità della parola target data la somma delle rappresentazioni delle parole circostanti;

2. **Skip-gram**: invece di prevedere la parola target data una finestra di parole circostanti, questo modello fa il contrario. Prende una parola target e cerca di prevedere le parole circostanti. Questo aiuta a catturare meglio il contesto e le relazioni tra le parole.

Metodi basati sul conteggio
Questi metodi sono basati sull'ipotesi distribuzionale: le parole che appaiono nello stesso contesto condividono un significato analogo, quindi, più una parola condivide lo stesso contesto con altre parole, più simili sono i loro significati.
Coinvolgono l'utilizzo di dati statistici, come la frequenza, e utilizzano la matrice di co-occorrenza per catturare le relazioni semantiche e sintattiche tra le parole. Un modello che sfrutta tali principi è *GloVe* (Global Vectors for Word Representation).

Recenti sviluppi[6]
Le rappresentazioni dei word embedding generati mediante i due modelli sopra citati sono statiche, quindi la rappresentazione vettoriale di ciascuna parola viene determinata unicamente in base all'addestramento del modello generato a priori. Tuttavia, così facendo sorge la questione della polisemia: il significato di una specifica parola può variare a seconda dei diversi contesti in cui essa viene utilizzata. Ad esempio, la parola "apple" solitamente rappresenta un tipo di frutto, ma, nel caso in cui venisse applicata in un contesto tecnologico, potrebbe invece riferirsi all'azienda informatica di Cupertino. Proprio per ovviare a questa problematica, la ricerca in tale campo si sta concentrando sull'integrare la generazione dei word embedding all'interno dei LLM, sfruttando le enormi potenzialità offerte da tali rete neurali per contestualizzare i vocaboli.

GPT
Come menzionato precedente, i modelli GPT si basano sull'architettura transformer, in questo caso vengono impiegati per l'estrazione e la loro capacità di cogliere relazioni a lungo raggio tra le parole in input.

[6] Q. Jiao e S. Zhang. «A Brief Survey of Word Embedding and Its Recent Development». In: *IEEE* (2021). URL: https://ieeexplore.ieee.org/abstract/document/9390956.

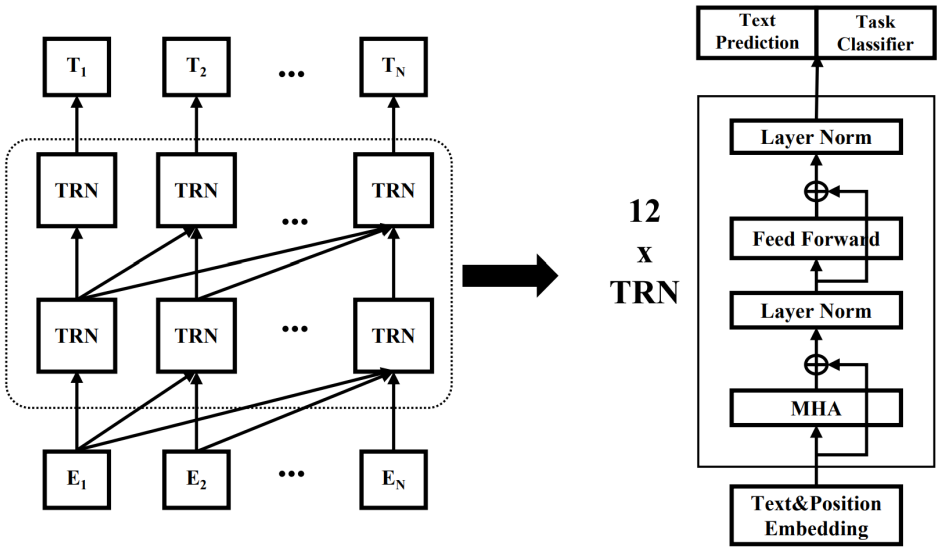


Figura 2.5: Architettura GPT per la generazione di embeddings

2.3 Vector database

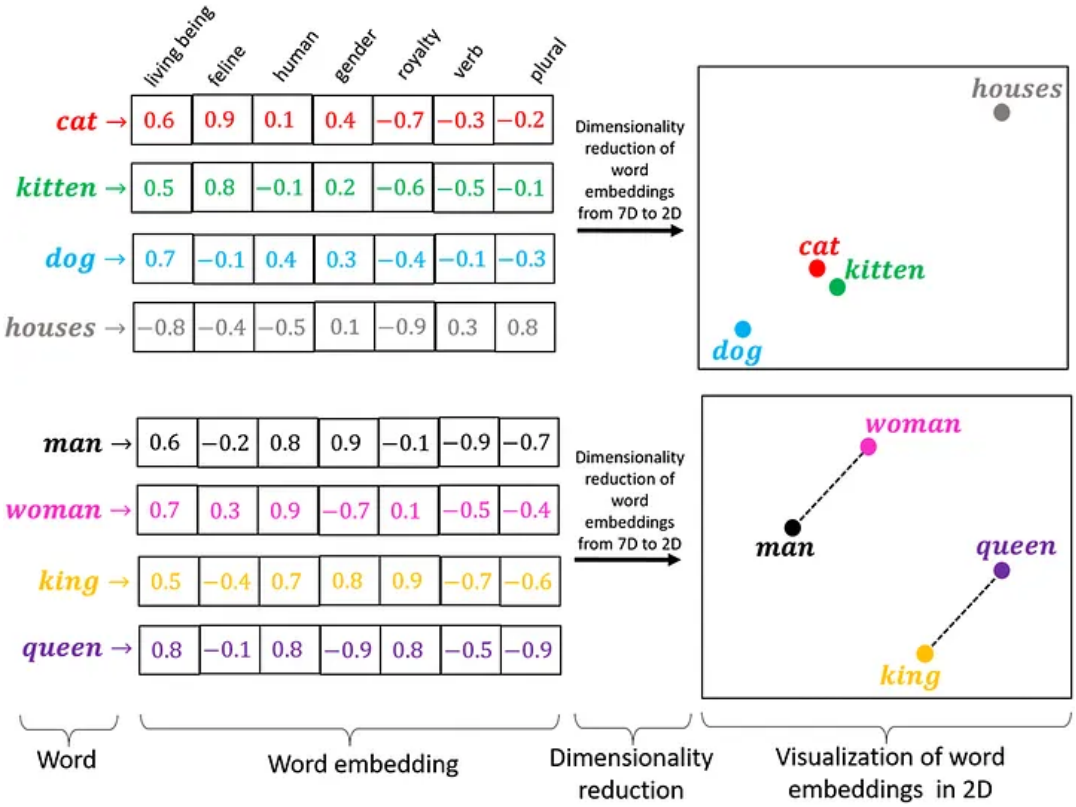


Figura 2.6: Rappresentazione 2D di word embedding in uno spazio vettoriale

Un database vettoriale[g] è un Database Management System (DBMS)[g] specializzato nel memorizzare dati in forma vettoriale ad alte dimensioni, utilizzando tecniche innovative per la memorizzazione, l'indicizzazione e l'elaborazione delle

query. Sono in grado di gestire query impegnative e algoritmi avanzati come la ricerca per similarità del testo, quindi in base al significato semantico del prompt. Le peculiarità che li contraddistingue sono la capacità di gestione dei dati, come operazioni CRUD, l'ingestione ad alta velocità, lo sharding[g], la replicazione e l'integrazione semplificata con linguaggi in ambito ML ampiamente utilizzati come Python e Tensorflow.

2.3.1 Funzionamento

Nei database tradizionali, solitamente vengono effettuate interrogazioni per righe in cui il valore corrisponde esattamente alla query sottoposta. Il modus operandi cambia invece quando si tratta di un database vettoriale, in quanto viene applicata una metrica di similarità per trovare un vettore che sia il più simile possibile alla query eseguita.
Un database vettoriale utilizza una combinazione di diversi algoritmi Approximate Nearest Neighbor (ANN)[g], che ottimizzano la ricerca attraverso l'hashing, la quantizzazione o la ricerca basata su grafi. Questi algoritmi vengono assemblati in una pipeline che fornisce una ricerca rapida e accurata dei vicini di un vettore posto come query.
Poiché il database vettoriale fornisce risultati approssimati, i principali compromessi da tenere in considerazione riguardano l'accuratezza e la velocità: più si desidera che il risultato sia accurato, più l'interrogazione sarà lenta.

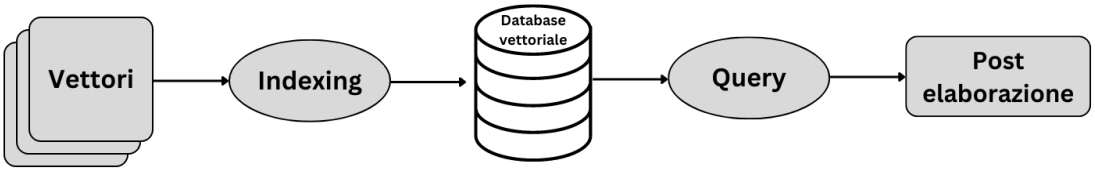


Figura 2.7: Pipeline di un database vettoriale

Gli step sul funzionamento possono essere raggruppati nel seguente modo:

- **Indexing**: i vettori vengono indicizzati, utilizzando algoritmi ANN per la ricerca approssimativa dei vicini più simili, e mappati ad una struttura dati per velocizzare la ricerca;
- **Querying**: il database vettoriale riceve una query e la compara con i vettori indicizzati, cercando i risultati più simili grazie ad una metrica di similarità;
- **Post elaborazione**: il database vettoriale restituisce i vicini più pertinenti, ed in alcuni casi effettua un re-ranking dei vicini utilizzando una metrica di similarità differente;

Capitolo 3

Svolgimento dello stage

3.1 Lo stage secondo Wintech

Figura 3.1: Locandina di Stage-IT 2023

L'azienda Wintech attribuisce molta importanza alla realizzazione di stage universitari, pertanto partecipa regolarmente all'evento Stage-IT, promosso annualmente da Confindustria Veneto Est in collaborazione con i Dipartimenti di Matematica e Scienze Statistiche dell'Università di Padova.
È proprio durante questo evento che ho avuto la possibilità di avere un primo contatto iniziale con l'azienda, presente con un proprio stand presso il quale proponevano tre progetti formativi con lo scopo di introdurre noi studenti universitari nel mondo del lavoro, in particolare nel settore dell'information and communication technology. Grazie a questa esperienza vengono quindi valutate le competenze degli studenti e la capacità di creare prodotti concreti di valore per l'azienda mediante processi di analisi, progettazione e sviluppo codice. Wintech pone particolare attenzione all'esplorazione di percorsi innovativi e all'applicazione di nuove tecnologie, indipendentemente dal quadro di pianificazione aziendale esistente e dagli attuali progetti in corso d'opera.
Tra le opzioni proposte dall'azienda quella che ha colto di più il mio interesse riguardava lo studio e l'applicazione di modelli d'intelligenza artificiale in ambito marketing all'interno di un'azienda di servizi informatici. Si tratta quindi di un progetto che coinvolge tecnologie all'avanguardia rappresentanti un punto di svolta significativo nel campo Information Technology (IT), poiché in grado di generare contenuti con una qualità mai vista precedentemente.

3.2 Obiettivi

In questa sezione vengono elencati tutti gli obiettivi concordati con il tutor aziendale, Enrico Merigliano, nel documento "Piano di Lavoro", redatto per poter iniziare lo stage in azienda. Si farà riferimento ai requisti secondo la seguente notazione:

- **O** - indica gli obiettivi obbligatori, vincolanti in quanto obiettivo primario richiesto dall'azienda;
- **D** - indica gli obiettivi desiderabili, non strettamente necessari ma dal riconoscibile valore aggiunto;
- **F** - indica gli obiettivi facoltativi/opzionali, rappresentanti un valore aggiunto non strettamente competitivo.

Obbligatori

- **O1**: Mappatura di tutte le funzionalità possibili tramite API[g] REST e loro prova tramite testi in ambito di marketing
  - analisi approfondita del sistema e delle parti interessate;
  - studio delle modalità di lavoro attualmente utilizzate per la gestione del marketing;
  - produzione di una completa documentazione correlata alle analisi svolte tramite un documento che mappi tutte le API messe a disposizione con modalità di utilizzo e scopo.
- **O2**: Costi e licenze
  - analisi delle modalità di utilizzo permesse ed il loro costo.
- **O3**: Analisi dei prodotti e delle tecnologie disponibili sul mercato
  - analisi di mercato e dei competitor;
  - proposta di sostituzione o di integrazione fra altri prodotti.

Desiderabili

- **D1**: Verificare che sistema GPT/Azure può essere esercitato su informazioni proprietarie;
- **D2**: Integrazione con il sistema di marketing dei prodotti aziendali;
- **D3**: Produzione di una completa documentazione progettuale.

Facoltativi

- **F1**: Realizzare la presentazione interna utilizzando l'applicazione prodotta;
- **F2**: Predisporre la manutenibilità della piattaforma in modo efficace.

3.3 Requisiti

I requisiti, così come concordato già dal primo incontro con il tutor, sono stati definiti in corso d'opera della progettazione. Questo è dovuto dal fatto che essendo il prodotto software da realizzare un PoC, lo scopo principale era quello di ricerca e di verificare cosa fosse implementabile e cosa invece non lo fosse.
Al termine dello stage i requisiti possono quindi essere racchiusi in cinque elementi distinti:

- **R1**: le tecnologie impiegate devono avere licenze compatibili per scopi commerciali;
- **R2**: le tecnologie impiegate devono rispettare le normative per il trattamento dei dati;
- **R3**: i risultati prodotti non devono essere soggetti ad allucinazioni;
- **R4**: i documenti consultati per fornire le risposte devono essere elencati alla fine di queste ultime;
- **R5**: le collezioni dei database vettoriali devono poter essere separate per permettere al modello di consultare set di documenti diversi;
- **R6**: le chat devono poter essere scaricate per poter tener traccia degli scambi di messaggi avvenuti con il chatbot.

3.4 Pianificazione del lavoro

Sempre come da Piano di Lavoro, la mia attività presso Wintech è stata suddivisa in quattro fasi distinte per un totale di 320 ore, ovvero 8 settimane lavorative a tempo pieno. Di seguito le fasi:

- **Fase 1 - Analisi degli strumenti**
  dal 29/05/2023 al 14/06/2023 (80h).

  Obiettivi:

  - effettuare una ricerca per ottenere tutta la documentazione necessaria su GPT e le API REST messe a disposizione;
  - testare le funzionalità disponibili utilizzando il materiale di marketing a disposizione in azienda e valutarne gli output;
  - mappare le funzionalità trovate in un documento in cui spiegherà come le feature messe a disposizione dalle API possono trasformare degli input in output utilizzabili a fini di marketing in un'azienda commerciale;
  - raccogliere i dati disponibili sulla licenza e il costo dell'utilizzo dello strumento in modo da avere dei dati a disposizione per valutare i competitor e presentare le differenze fra i prodotti nella fase seguente.

- **Fase 2 - Analisi dei competitor**
  dal 15/06/2023 al 21/06/2023 (40h).

  Obiettivi:

  - verificare la presenza sul mercato di prodotti alternativi fra i quali "Co-pilot" di Microsoft Corporation;
  - valutare l'eventuale necessità di integrazione o la sostituzione con essi per il fine di produrre contenuti di marketing testuali e multimediali per l'azienda.

- **Fase 3 - Integrazione negli strumenti aziendali**
  dal 22/06/2023 al 24/07/2023 (180h).

  Obiettivi:

  - verificare la possibilità di eseguire addestramento dello strumento di IA su dati aziendali esistenti;
  - realizzare un software come web app che permetta di inserire degli input testuali o dei settaggi che permettano ad un utente di ottenere come output dei testi e del materiale multimediale sapendo che il software è un proof of concept dell'integrazione dello strumento di IA all'interno della web app proprietaria di marketing esistente in azienda;
  - il software prodotto è corredato da una documentazione di progetto che permetta ad uno stakeholder di comprenderne gli obiettivi ed i risultati raggiunti e ad uno sviluppatore di integrare il lavoro in un software aziendale.

- **Fase 4 - Utilizzo del sistema**
  dal 24/07/2023 al 26/07/2023 (20h).

  Obiettivi:

  - utilizzare il sistema per sfruttarne le capacità dimostrando di poter produrre la presentazione di progetto che verrà effettuata da parte dello studente al board direttivo dimostrando quanto appreso e quanto realizzato. I prodotti saranno la descrizione del lavoro svolto e il materiale di presentazione.

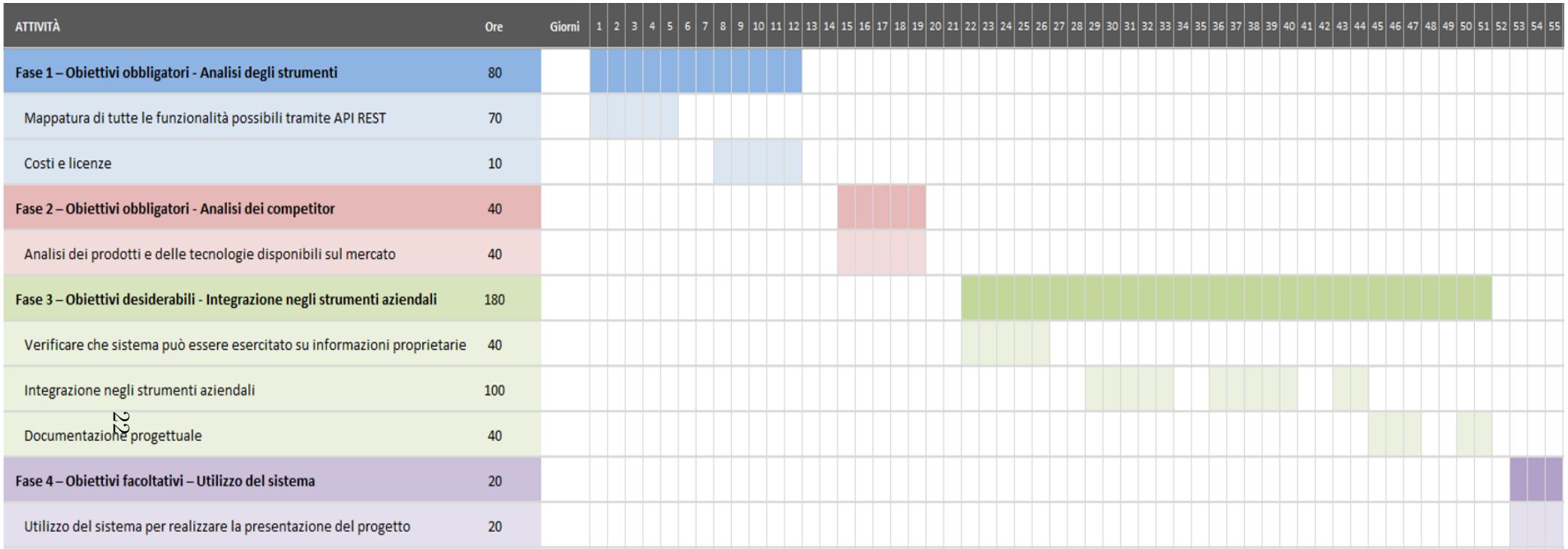


Figura 3.2: Diagramma di Gantt per la suddivisione del lavoro

3.5 Fasi

In questa sezione vengono descritte le prime due fasi del progetto, ovvero quelle incentrate sullo studio e analisi degli strumenti e dei competitor, mentre la terza verrà descritta nel dettaglio nel capitolo successivo. La quarta, la più breve fra tutte per durata temporale, riguarda invece unicamente la realizzazione di una presentazione interna aziendale sul software realizzato, pertanto non verrà approfondita.

3.5.1 Fase 1 - Analisi degli strumenti

Durante la prima fase, l'obiettivo era quello di studiare e analizzare gli strumenti messi a disposizione da OpenAI per capire come poterli utilizzare al meglio per i fini aziendali e, successivamente, realizzare un documento che descrivesse le funzionalità trovate.
Per quanto riguarda la fase di ricerca, ho iniziato a leggere il materiale ufficiale di OpenAI, in particolare le seguenti sezioni:

- **Documentazione**: è la parte più discorsiva, vengono descritte le funzionalità e le modalità di utilizzo dei prodotti messi a disposizione, ponendo particolare attenzione alle best practices e alle limitazioni che derivano dall'usare queste tecnologie;
- **Riferimenti API**: è la parte più tecnica, vengono descritte le API REST messe a disposizione per l'utilizzo dei prodotti, in particolare vengono speficiati i parametri di input/output e le modalità di utilizzo delle stesse;
- **Termini di servizio**: è la parte legale, vengono descritte le condizioni di utilizzo dei prodotti;
- **Privacy policy**: vengono descritte le modalità di trattamento dei dati personali, tematica fondamentale per questo progetto in quanto vengono coinvolti dati appartenenti all'azienda ed è bene assicurarsi che non vengano utilizzati in modo improprio;
- **Pricing**: trattandosi di un progetto che potrà avere risvolti commerciali, è bene informarsi attentamente sui costi di utilizzo dei prodotti per valutare la fattibilità economica del progetto.

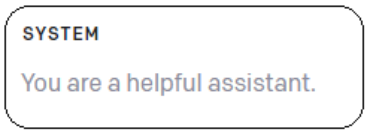


Figura 3.3: Casella di input System

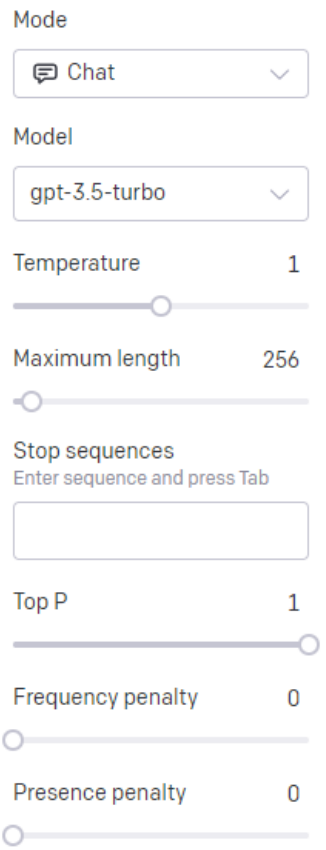


Figura 3.4: Riquadro contenente i settaggi

Prima di redigere il documento, ho testato le funzionalità disponibili utilizzando il playground[1] messo a disposizione da OpenAI, in modo da avere un'idea più chiara di come funzionassero i modelli e i relativi parametri da poter impostare. Questo mi ha permesso di capire quali fossero le potenzialità e i limiti di questi strumenti, cogliendo le diverse sfumature che contraddistinguono i vari modelli.
Tra i settaggi troviamo:

- **System**: consente di inserire un messaggio di input con lo scopo di istruire il modello su come deve comportarsi e qual è il contesto in cui deve operare. Questo parametro è opzionale, ma è consigliato inserirlo per ottenere risultati migliori;
- **Temperature**: consente di controllare la creatività del modello, ovvero la probabilità che il modello generi un token diverso da quello più probabile. Ad esempio, se la temperatura viene impostata a 1, il modello genererà un token diverso da quello più probabile, mentre se viene settata a 0 il modello genererà sempre il token più probabile;
- **Maximum length**: permette di limitare la lunghezza del testo generato in base al numero di token selezionati;

[1] https://platform.openai.com/playground

- **Stop sequences**: per evitare di generare token non necessari, si può inserire una sequenza di interruzione. Ad esempio, le sequenze di interruzione si possono utilizzare per generare una lista con un numero specifico di elementi; in questo caso, utilizzando "11" come sequenza di interruzione, puoi generare una lista con solamente 10 elementi, poiché il completamento si interromperà una volta raggiunto il numero 11;

- **Top P**: è un parametro che permette di filtrare i token generati, in particolare vengono scelti i token che hanno una probabilità cumulativa maggiore o uguale a quella specificata. Ad esempio, se il parametro viene impostato a 0.9, verranno scelti i token che hanno una probabilità cumulativa maggiore o uguale a 0.9, ovvero i token che rappresentano il 90% della probabilità totale;

- **Frequency penalty**: è un parametro utilizzato per scoraggiare il modello dal ripetere troppo frequentemente le stesse parole o frasi all'interno del testo generato Ad esempio, se il parametro viene impostato a 0.9, verranno ridotte le probabilità di generare dei token già presenti del 90%;

- **Presence penalty**: è un parametro utilizzato per incoraggiare il modello a includere una vasta gamma di token nel testo generato.

I modelli presenti sul playground non erano tutti quelli che OpenAI mette a disposizione tramite API, ma questo non è stato un problema siccome l'obiettivo era quello di testare la qualità della produzione del testo e i modelli presenti erano sufficienti a tale scopo.
In particolare mi sono concentrato sui seguenti modelli:

- **Chat**
  - **gpt-3.5-turbo**: è il modello di terza generazione GPT più performante fornito da OpenAI, ha subito operazioni di fine tuning per renderlo adatto in contesti che prevedano l'utilizzo di una chat. Rappresenta un ottimo compromesso tra costi di utilizzo, tempi di inferenza e qualità dei risultati; OpenAI stessa raccomanda l'utilizzo di questo modello nella maggior parte dei casi d'uso. Token massimi consentiti: 4.096.

- **Completamento di testo**
  - **text-davinci-003**: è un modello molto potente ma dispendioso, in quanto costa 10 volte tanto rispetto a gpt-3.5-turbo. Produce ottimi risultati, richiedendo leggermente più tempo per l'elaborazione dei testi. Token massimi consentiti: 4.097;
  - **text-curie-001**: è un modello di fascia media, produce risultati modesti in poco tempo ad un prezzo inferiore di davinci. Token massimi consentiti: 2.049.

Al termine delle prove il modello che ha soddisfatto maggiormente le aspettative è stato gpt-3.5-turbo, proprio come consigliato da OpenAI, in quanto economico e funzionale.

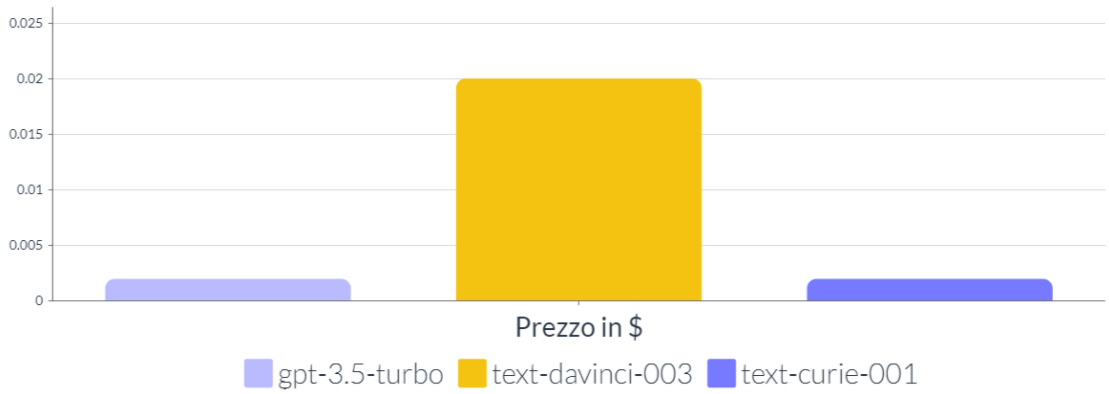


Figura 3.5: Costi dei modelli testati di OpenAI

Per quanto riguarda invece il trattamento dei dati, mi sono accertato in primis che quest'ultimi non venissero utilizzati per altri scopi che non siano strettamente a fini di tutela legale. Inoltre, assieme al tutor ci siamo consultati con il Data Protection Officer (DPO)[g] dell'azienda per avere il parere di una figura più esperta in merito.

Infine, ho redatto il documento riguardante tutte le tematiche sopracitate, in modo tale da poterlo condividere con il resto del team e successivamente discuterne avendo del materiale tangibile per valutare le future direzioni da intraprendere durante il percorso.

3.5.2 Fase 2 - Analisi dei competitor

Durante la seconda fase del progetto, l'obiettivo principale consisteva nello studio approfondito e nell'analisi dei competitor di OpenAI, al fine di valutare la presenza di valide alternative che potessero sostituire, o affiancare, il servizio precedentemente menzionato.

Il tutor ha posto particolare enfasi sulla necessità di raccogliere informazioni dettagliate riguardo a Microsoft Copilot, che era stato annunciato soltanto due mesi prima dell'inizio dello stage. Questo prodotto si presentava come una proposta estremamente simile e vantaggiosa, condividendo numerosi casi d'uso con il servizio di OpenAI. Nonostante la mia ricerca abbia richiesto meno tempo del previsto, a causa del fatto che Microsoft aveva reso disponibile una versione beta solo a un ristretto numero di aziende selezionate, sono riuscito a reperire un sufficiente numero di informazioni. Tali dati mi hanno permesso di affermare con abbastanza sicurezza che Microsoft Copilot, una volta rilasciato ufficialmente, potrebbe rappresentare una valida alternativa ai servizi forniti da OpenAI per gli scopi del progetto di questo stage. L'integrazione con la suite proprietaria di Microsoft e, di conseguenza, con i documenti aziendali ospitati su SharePoint, lo posiziona come un potenziale concorrente.

Nel corso di una ricerca indipendente, ho individuato una serie di modelli gratuiti che possono essere scaricati ed eseguiti in locale sulla propria macchina. Tuttavia, nonostante i promettenti benchmark presenti sul sito GPT4All[2], alcuni test preliminari hanno messo in luce fin da subito che l'esecuzione di tali modelli richiede una potenza computazionale significativa. Questo aspetto si è rivelato problematico, in quanto il computer che avevo in dotazione non era dotato di hardware sufficiente a sostenere tali requisiti computazionali. Attraverso una prova effettuata durante lo sviluppo del PoC, utilizzando un documento di dimensioni limitate, ho constatato che il tempo di esecuzione ammontava a circa 11 minuti per ottenere una risposta di solamente due parole. Tale inferenza si è rivelata eccessivamente elevata in un contesto in cui l'utente si aspetta di ricevere una risposta pressoché immediata, mettendo in luce la necessità di possedere risorse di calcolo più potenti per renderne l'utilizzo efficiente.

Come parte del processo di valutazione, ho creato una dettagliata matrice di confronto utilizzando Excel, tuttavia, le dimensioni di quest'ultima (42 righe e 10 colonne) ne rendono impraticabile l'inserimento nella tesi. All'interno della matrice ho preso in considerazione vari aspetti dei servizi, tra cui: l'utilità per l'azienda, la disponibilità di API, i costi mensili, la qualità della documentazione, la presenza di esempi e di una community attiva, le politiche sulla privacy e la fattibilità dell'utilizzo con l'ingestione dei documenti.
Questa fase di analisi e valutazione dei competitor ha portato ad ottenere una panoramica approfondita delle possibili alternative al servizio di OpenAI, e ha posto le basi per la selezione della soluzione più adatta alle esigenze dell'azienda.

[2] https://gpt4all.io/index.html

Capitolo 4

Il progetto

In questo capitolo viene presentato il progetto realizzato durante lo stage, descrivendone l'implementazione e le principali funzionalità.

4.1 Tech stack

In questa sezione vengono presentate tutte le tecnologie e gli strumenti per lo sviluppo del software finale da me individuati durante la terza fase dello stage.

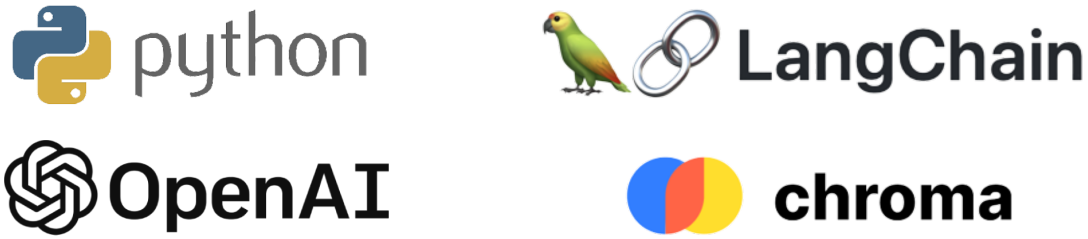


Figura 4.1: Tech stack del progetto

4.1.1 Python

Python è un linguaggio di programmazione multi-paradigma ad alto livello, creato da Guido van Rossum all'inizio degli anni novanta. È rinomato per la sua semplicità, leggibilità e versatilità, in quanto offre una sintassi pulita e concisa, rendendo intuitiva la comprensione e la scrittura del codice. Oltre all'ampia libreria standard, l'ecosistema di questo linguaggio viene ulteriormente arricchito grazie alla comunità di sviluppatori sempre più popolata, offrendo ai programmatori un set di funzioni e metodi adatti a svolgere una vasta gamma di compiti in modo efficiente e rapido.

Tra i punti cardine di Python troviamo la sua natura orientata agli oggetti, la tipizzazione dinamica, la gestione automatica della memoria e la compatibilità multi-piattaforma, tutti fattori che contribuiscono notevolmente alla sua flessibilità e facilità d'uso. Inoltre, con il packet manager pip, è possibile installare e gestire facilmente le dipendenze di un progetto, semplificando quindi il lavoro di sviluppo

e manutenzione utilizzando gli ambienti virtuali. In aggiunta a tutto ciò, Python è in costante aggiornamento ed evoluzione, spingendo quindi molti sviluppatori indipendenti ed aziende ad adottarlo in nuovi progetti, soprattutto in ambiti quali l'IA, il ML e la data science.

4.1.2 LangChain

LangChain è un framework[g] open-source per Python e JS/TS, lanciato nell'ottobre del 2022 con lo scopo di semplificare la creazione di applicazioni che utilizzino i LLM. I punti di forza che caratterizzano questo framework sono:

- i componenti: ovvero astrazioni modulari per lavorare con modelli linguistici, per ciascuna astrazione è presente una collezione di implementazioni;
- le catene: concatenazioni di più componenti per svolgere compiti con una complessità superiore e coprire un'ampia gamma di casi d'uso;
- le integrazioni: vengono messe a disposizione numerevoli integrazioni di popolari servizi come il cloud storage di Amazon, Google e Microsoft Azure, API per fornire ai LLM informazioni aggiornate, sistemi e template per il web scraping e molto altro.

4.1.3 OpenAI

OpenAI è una società no profit fondata a dicembre del 2015 con lo scopo di sviluppare modelli d'intelligenza artificiale e renderli disponibili a livello globale, così che tutti siano partecipi all'evoluzione di una tecnologia che suscita un grande interesse, ma anche numerosi quesiti.
Tra i loro di prodotti di spicco troviamo i modelli GPT, dei LLM basati su reti neurali con centinaia di miliardi di parametri, rendendoli quindi i più preformanti sul mercato al giorno d'oggi. Per la realizzazione del progetto sono quindi stati utilizzati i seguenti modelli, impiegando la relativa implementazione della libreria di OpenAI in LangChain:

- gpt-3.5-turbo: modello per la generazione di testo in linguaggio naturale e codice, ottimizzato particolarmente per casi d'uso che riguardino la presenza di una chat;
- emebeddings-ada-002: modello per la creazione di word embedding a partire da testo in linguaggio naturale.

4.1.4 Chroma

Chroma è un database vettoriale open-source facile ed intuitivo, creato unicamente per memorizzare i word embedding - generati a partire da un modello apposito - e il relativo testo, ottimizzando l'operazione di ricerca semantica. Data la sua efficienza, è il vector store di punta integrato in LangChain, dove il suo utilizzo viene ulteriormente semplificato, e reso compatibile con il tipo degli oggetti del framework, senza tuttavia perdere le principali funzioni fornite.

4.1.5 Streamlit

Streamlit è un framework open-source, sviluppato unicamente per Python, che consente di creare un'interfaccia utente moderna, gradevole e semplice da utilizzare con poche righe di codice. Questo permette di conseguenza agli sviluppatori di poter concentrare le proprie risorse sulla parte di backend. Quindi, per poter realizzare una web app completa e funzionante, non sono richieste competenze con linguaggi tradizionali dello sviluppo web, ampliando di conseguenza le possibilità di creazione e distribuzione di pagine web interattive. Ciò rende Streamlit una scelta ideale per sviluppatori Python che desiderano trasformare rapidamente le proprie analisi dati, modelli machine learning o altre elaborazioni in strumenti accessibili tramite un'interfaccia web user-friendly.

4.2 Casi d'uso

- **UC1 - Seleziona tipologia di risposta**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente seleziona la tipologia di risposta che vuole ricevere dal chatbot
  - **Precondizione**: L'utente ha avviato il chatbot e ha aperto la sidebar
  - **Postcondizione**: Il chatbot è pronto a ricevere la domanda dell'utente
  - **Scenario principale**:
    1. L'utente seleziona la tipologia di risposta che vuole ricevere dal chatbot
  - **Estensioni**:
    1. UC1.1 - Seleziona risposta testuale
    2. UC1.2 - Seleziona risposta immagine
- **UC1.1 - Seleziona risposta testuale**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente seleziona la tipologia di risposta testuale
  - **Precondizione**: L'utente ha avviato il chatbot
  - **Postcondizione**: Il chatbot è pronto a ricevere la domanda dell'utente
  - **Estensioni**:
    1. UC1.1.1 - Selezione database vettoriale
- **UC1.2 - Seleziona risposta immagine**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente seleziona la tipologia di risposta immagine
  - **Precondizione**: L'utente ha avviato il chatbot

  - **Postcondizione**: Il chatbot è pronto a ricevere la domanda dell'utente

- **UC1.1.1 - Selezione database vettoriale**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente seleziona il database vettoriale da cui vuole ricevere la risposta
  - **Precondizione**: L'utente ha selezionato la tipologia di risposta testuale
  - **Postcondizione**: Il chatbot è pronto a ricevere la domanda dell'utente
  - **Scenario principale**:
    1. L'utente seleziona la tipologia di risposta testuale
    2. L'utente seleziona il database vettoriale da cui vuole ricevere la risposta

- **UC2 - Invio del messaggio**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente invia un messaggio al chatbot
  - **Precondizione**: L'utente ha selezionato la tipologia di risposta che vuole ricevere dal chatbot
  - **Postcondizione**: Il chatbot invia la risposta all'utente
  - **Scenario principale**:
    1. L'utente seleziona la tipologia di risposta che vuole ricevere dal chatbot
    2. L'utente invia un messaggio al chatbot

- **UC3 - Creazione database vettoriale**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente crea un database vettoriale
  - **Precondizione**: L'utente ha aperto la sidebar
  - **Postcondizione**: Il database vettoriale è stato creato
  - **Scenario principale**:
    1. L'utente sceglie il nome del database vettoriale
    2. L'utente sceglie la cartella da cui prendere i file
    3. L'utente crea un database vettoriale
  - **Estensioni**:
    1. UC3.1 - Aggiornamento database vettoriale

- **UC3.1 - Aggiornamento database vettoriale**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente aggiorna un database vettoriale

  - **Precondizione**: L'utente ha creato un database vettoriale
  - **Postcondizione**: Il database vettoriale è stato aggiornato
  - **Scenario principale**:
    1. L'utente sceglie il database vettoriale da aggiornare

- **UC4 - Lista documenti caricati nel database vettoriale**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente visualizza la lista dei documenti caricati nel database vettoriale
  - **Precondizione**: L'utente ha creato un database vettoriale
  - **Postcondizione**: L'utente visualizza la lista dei documenti caricati nel database vettoriale
  - **Scenario principale**:
    1. L'utente visualizza la lista dei documenti caricati nel database vettoriale selezionato

- **UC5 - Scarica messaggi della chat**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente scarica i messaggi della chat
  - **Precondizione**: L'utente ha avviato il chatbot e ha inviato almeno un messaggio
  - **Postcondizione**: I messaggi della chat vengono scaricati
  - **Scenario principale**:
    1. L'utente scambia uno o più messaggi con il chatbot
    2. L'utente scarica i messaggi della chat

- **UC6 - Cancella messaggi della chat**
  - **Attore primario**: Utente
  - **Descrizione**: L'utente cancella i messaggi della chat
  - **Precondizione**: L'utente ha avviato il chatbot e ha inviato almeno un messaggio
  - **Postcondizione**: I messaggi della chat vengono cancellati
  - **Scenario principale**:
    1. L'utente scambia uno o più messaggi con il chatbot
    2. L'utente cancella i messaggi della chat

4.3 Implementazione

4.3.1 Backend

Il backend è costituito principalmente da due parti:

- l'ingestione dei documenti e la conseguente creazione/aggiornamento di un database vettoriale;
- l'istanziazione e il settaggio dei parametri del modello di OpenAI gpt-3.5-turbo per renderlo in grado di consultare i documenti.

Ingestione
L'ingestione dei documenti avviene tramite il caricamento di tutti i file `pdf`, `pptx`, `docx`, `txt`, salvati in una cartella del server, e la loro successiva conversione in testo. Per la conversione dei file in testo puro sono stati mappati i moduli con le loro relative estensioni:

Tabella 4.1: Mappatura estensioni - moduli

Estensione	Modulo
`pdf`	PyPDFium2Loader
`pptx`	UnstructuredPowerPointLoader
`docx`	Docx2txtLoader
`txt`	TextLoader

Una volta che i file sono stati correttamente convertiti in testo, vengono divisi in chunk[g] di circa 1000 caratteri l'uno, cercando il più possibile di non dividere i paragrafi, le frasi e le parole per mantenere invariato il significato del testo. Questo consente di avere una maggiore precisione nella ricerca semantica, in quanto avendo porzioni ridotte è possibile generare dei word embedding più accurati.
È importante a questo punto generare gli Universally Unique Identifier (UUID)[g] per ogni chunk tramite l'utilizzo di Secure Hash Algorithm 1 (SHA-1)[g], un algoritmo crittografico che, usando il testo contenuto del chunk come input, genera un hash di 160 bit successivamente convertiti in una stringa di 40 caratteri esadecimali.

Ora c'è tutto il necessario per poter istanziare un database vettoriale con Chroma e renderlo permanente, così che le operazioni effettuate su di esso persistano alla chiusura del programma. All'interno di esso vengono caricati tutti i chunk e i relativi UUID, successivamente, per ogni chunk, viene generato e memorizzato un word embedding tramite il modello embeddings-ada-002 di OpenAI.

Grazie agli identificatori univoci, è possibile effettuare due operazioni di ottimizzazione:

- verificare se un chunk è già presente nel database vettoriale e in caso contrario aggiungerlo;

- se un documento precedentemente caricato in Chroma viene modificato, è sufficiente aggiornare unicamente i chunk che hanno subito modifiche, e non generare nuovamente word embedding anche per porzioni di testo rimaste invariate.

Istanziazione e settaggio del modello
LangChain mette a disposizione un oggetto che, dati i seguenti elementi:

- un LLM;
- un retriever di documenti (si può usare anche un database vettoriale con l'apposito metodo `as_retriever()`);
- un numero k che indica la quantità di documenti da restituire;

crea una catena di componenti in grado di accettare una query in linguaggio naturale come input e di restituire in output una risposta che tragga informazioni unicamente dai k documenti più rilevanti. In sostanza a gpt-3.5-turbo viene fornito un prompt che contiene la domanda formulata dall'utente e un k numero di chunk pertinenti ad essa, al modello quindi non spetta nient'altro che comprendere tutto ciò e formulare un'unica risposta finale.

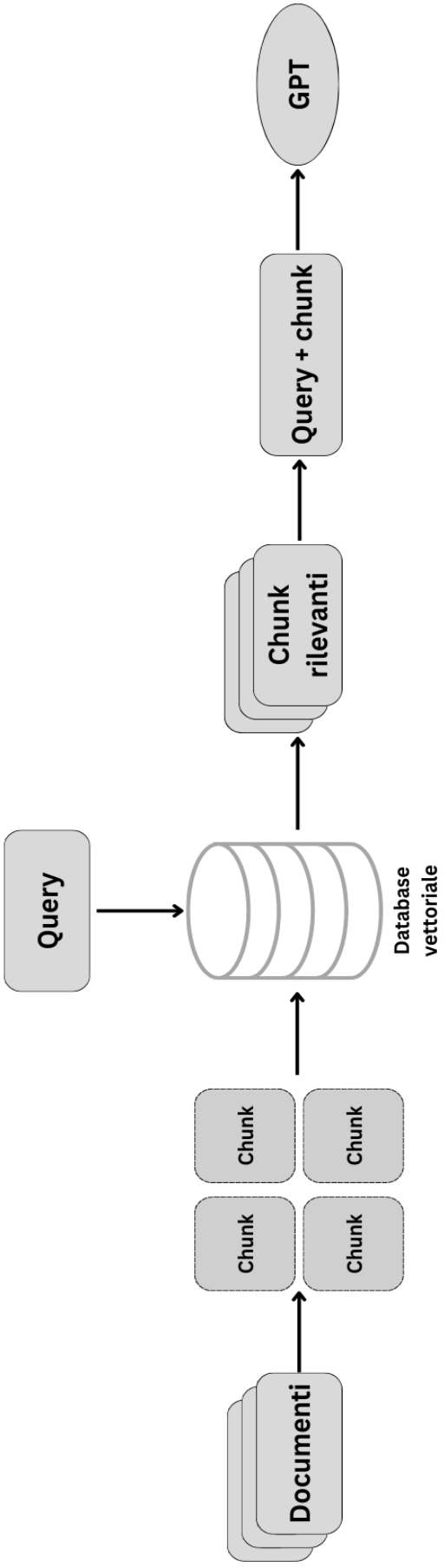


Figura 4.2: Pipeline funzionamento chatbot

4.3.2 Recenti sviluppi di LangChain

Solamente dopo aver concluso lo stage, è stata rilasciata una nuova versione di LangChain che ha introdotto diverse novità. Tra queste, quelle più significative per questo progetto sono:

- **Compressione del contesto**: invece di restituire immediatamente i chunk così come sono dopo la ricerca semantica, è possibile comprimerli utilizzando il contesto della query fornita. In questo modo viene restituita solo l'informazione rilevante, scartando tutto il resto, apportando un considerevole risparmio di denaro, siccome il prompt che viene inviato a OpenAI è più breve, e quindi meno caro;
- **MultiQueryRetriever**: la ricerca di documenti basata sulla similarità può produrre risultati diversi anche con leggere modifiche nella formulazione della query o se le rappresentazioni non catturano accuratamente la semantica dei dati. Grazie a questa nuova funzionalità, è ora possibile automatizzare questo processo generando query che, pur essendo sintatticamente diverse, conservano lo stesso significato. A questo punto, è possibile combinare i risultati ottenuti attraverso l'unione dei documenti duplicati e restituire un insieme di documenti più accurato.

Questo denota come LangChain sia un progetto in continua evoluzione che non smette mai di innovare e apportare migliorie, aiutando gli sviluppatori a creare applicazioni sempre più performanti e scalabili. Infatti, durante tutta la programmazione non ho avuto la necessità di creare neanche una classe, in quanto il framework mette a disposizione tutto ciò che serve per creare una catena di elaborazione di questo tipo.

4.3.3 Frontend

Come spiegato precedentemente, per realizzare l'interfaccia utente è stato scelto di utilizzare il framework Streamlit, in quanto facile ed intuitivo. L'obiettivo dello stage infatti non era realizzare un prodotto già pronto per il mercato, ma piuttosto un prototipo funzionante che potesse essere utilizzato per testare le potenzialità del chatbot in questo specifico contesto applicativo. L'User Interface (UI) è stata quindi implementata unicamente per rendere l'utilizzo del software più piacevole rispetto all'utilizzo di una semplice Command Line Interface (CLI)[g].
Di seguito viene spiegata l'interfaccia grafica e le funzionalità proposte.

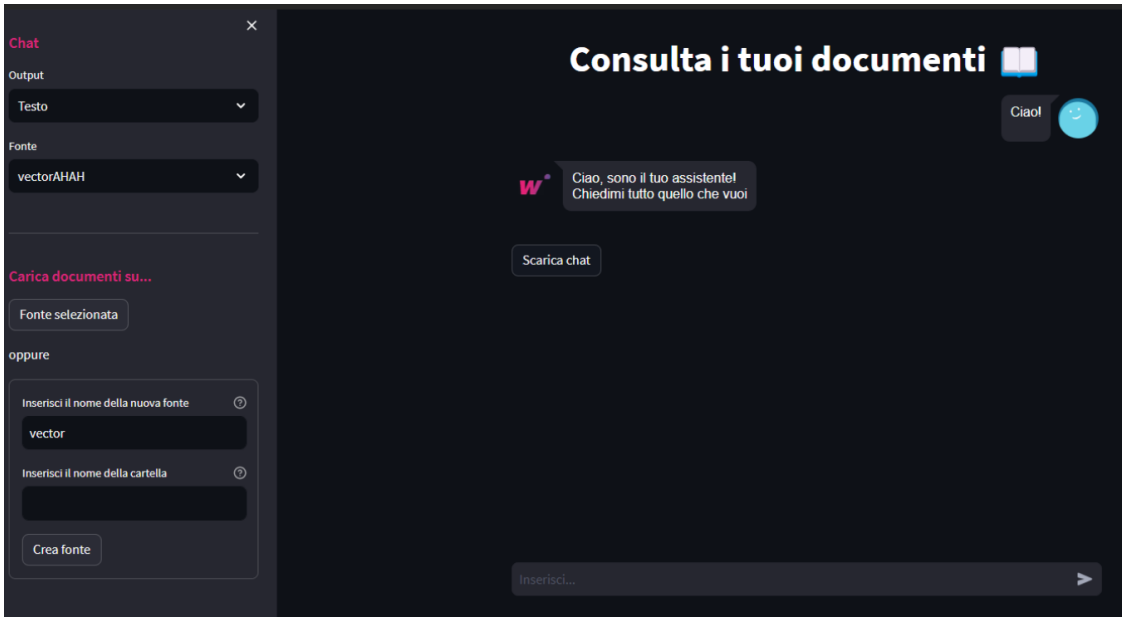


Figura 4.3: Interfaccia della web app

La schermata è volutamente minimale con lo scopo di renderne l'utilizzo facile ed intuitivo. È composta essenzialmente da due sezioni principali che verranno approfondite in seguito, ossia:

1. Sidebar laterale posizionata a sinistra
2. Cronologia dei messaggi bot – utente

Sidebar
È la parte della web app più ricca di contenuti, in quanto fornisce all'utente svariate funzioni per ottenere il massimo delle potenzialità dell'applicazione. Andando in ordine si trovano le seguenti sezioni:

- **Chat**

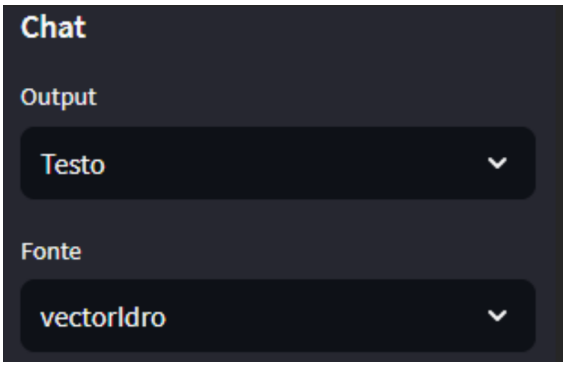


Figura 4.4: Interfaccia - sezione chat

  È composta da due menù a tendina, ovvero:

  – Output: è possibile scegliere se ottenere una risposta testuale o la generazione di un immagine;
  – Fonte: permette di selezionare il vector store di riferimento dal quale il modello recupererà le informazioni sui documenti richiesti dall'utente.

- **Documenti**

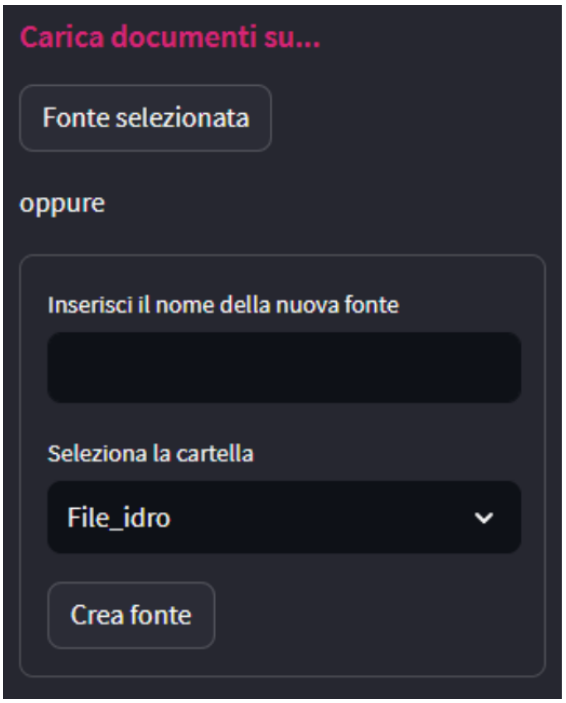


Figura 4.5: Interfaccia - sezione caricamento

In questa sotto sezione si può creare, o aggiornare, un vector store utilizzando i file nei formati `pdf, pptx, docx, txt` salvati in una cartella del server tramite:

  - Fonte selezionata: permette di aggiungere i file al vector store precedentemente selezionato nel menù a tendina sotto la voce "Fonte".
  - Se invece si volesse creare un vector store da zero:
    * inserire il nome desiderato da assegnare al database vettoriale;
    * selezionare dal menù a tendina il nome della cartella contenente i documenti e premere successivamente il bottone "Crea fonte" dedicato.

- **Altro**
  - Svuota chat
  - Lista documenti caricati

Messaggi bot-utente

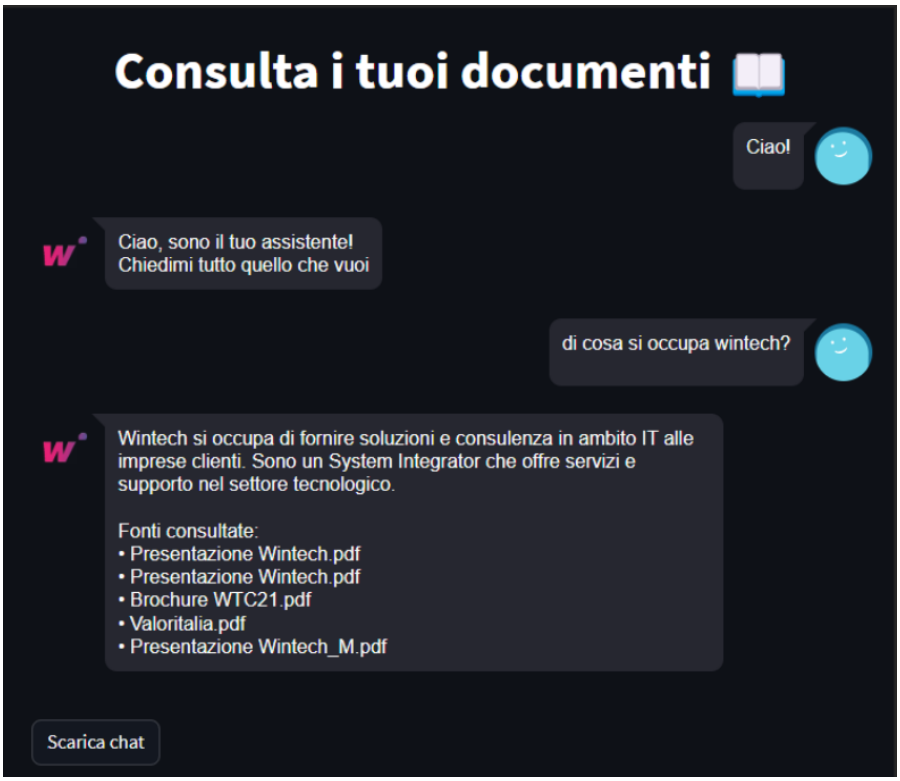


Figura 4.6: Esempio di un possibile scambio di messaggi con il chatbot

In questo esempio dimostrativo è possibile notare che i messaggi a destra, come in tutte le chat, rappresentano quelli inviati dall'utente, mentre quelli a sinistra sono i messaggi generati dal bot. Ognuno di essi termina con una lista di tutti i file che l'IA ha consultato per poter genere la risposta visualizzata. In basso a sinistra

all'ultimo messaggio inviato dal bot si trova il pulsante "Scarica chat" che consente di salvare un file `txt` contenente l'intera conversazione.

4.4 Testing

Data la natura del progetto, l'implementazione di un testing automatico non è stata un'opzione percorribile, poiché il modello utilizza informazioni specifiche relative all'azienda, per le quali non esistono strumenti predefiniti in grado di verificare la correttezza delle risposte generate. La creazione di un dataset personalizzato per il testing non avrebbe coperto totalmente le informazioni presenti nelle centinaia di documenti trattati e, inoltre, avrebbe messo in secondo piano tutta la parte che riguarda l'esplorazione delle potenzialità che offre il chatbot, ossia uno dei principali scopi di questo progetto di ricerca e sviluppo. Oltretutto, non bisogna trascurare la quantità significativa di tempo che sarebbe stata sottratta alla terza fase, in quanto la realizzazione di questo sistema avrebbe comportato una riduzione considerevole del numero di ore a disposizione per l'implementazione delle feature richieste e per lo sviluppo di eventuali migliorie. Date queste premesse, si è preferito procedere con il testing manuale, che consiste nel formulare domande e verificare la qualità delle risposte mediante un confronto con le aspettative dell'utente. A tal proposito, per garantire la correttezza e l'efficacia del modello, sono state organizzate riunioni settimanali con il tutor aziendale, durante le quali si discutevano i progressi compiuti e si valutava il funzionamento del chatbot.

Inoltre, sono stati condotti test occasionali coinvolgendo persone esterne all'azienda al fine di ottenere un feedback più ampio e verificare la capacità del modello di rispondere correttamente a domande su contesti totalmente diversi, come ad esempio la documentazione proveniente dall'Autorità di Bacino distrettuale dell'Appennino Settentrionale [1]. Questi incontri hanno fornito preziosi spunti di riflessione che hanno contribuito alla formulazione di alcuni dei requisiti descritti nella sezione §3.3.

4.4.1 Risultati prodotti

In questa sezione vengono mostrati gli output generati dal modello per dimostrarne la capacità di produrre testo e di rispondere alle domande, usando come unica fonte i documenti presenti all'interno del database vettoriale.
Per effettuare questo test ho usato come documento da esaminare la presente tesi - in formato `pdf` - e ho formulato dei quesiti di diversa natura, in modo tale da verificare che il modello fosse in grado di rispondere accuratamente a sezioni differenti della tesi.

[1] https://www.appenninosettentrionale.it/itc/

Di seguito vengono riportati i risultati ottenuti durante l'esecuzione del software.

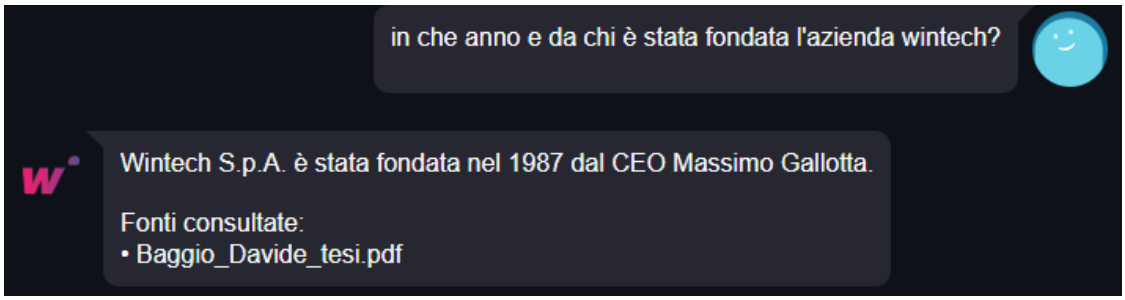


Figura 4.7: Output domanda riguardante Wintech

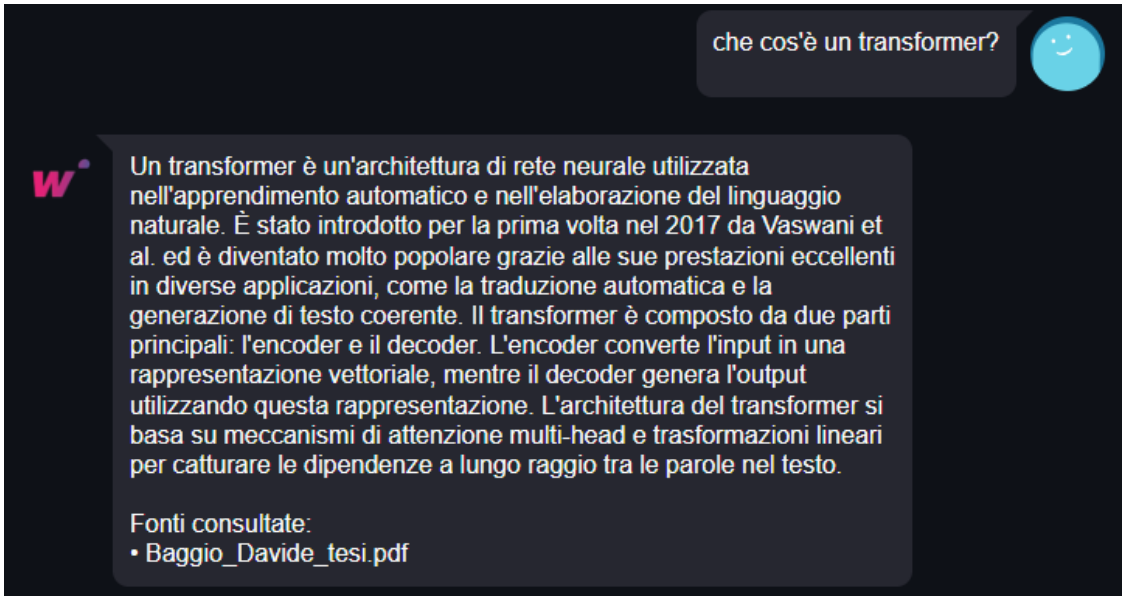


Figura 4.8: Output domanda riguardante i transformer

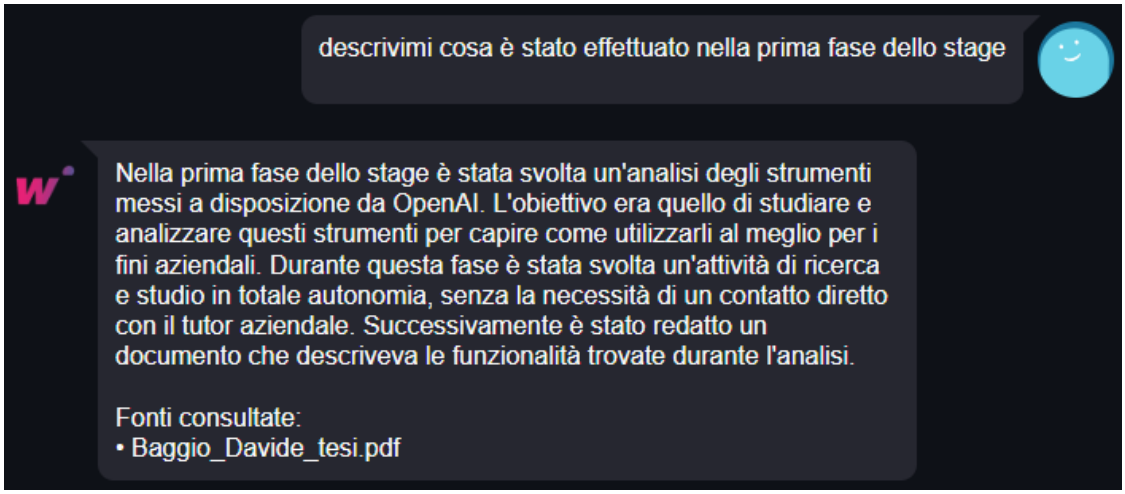


Figura 4.9: Output domanda riguardante la prima fase dello stage

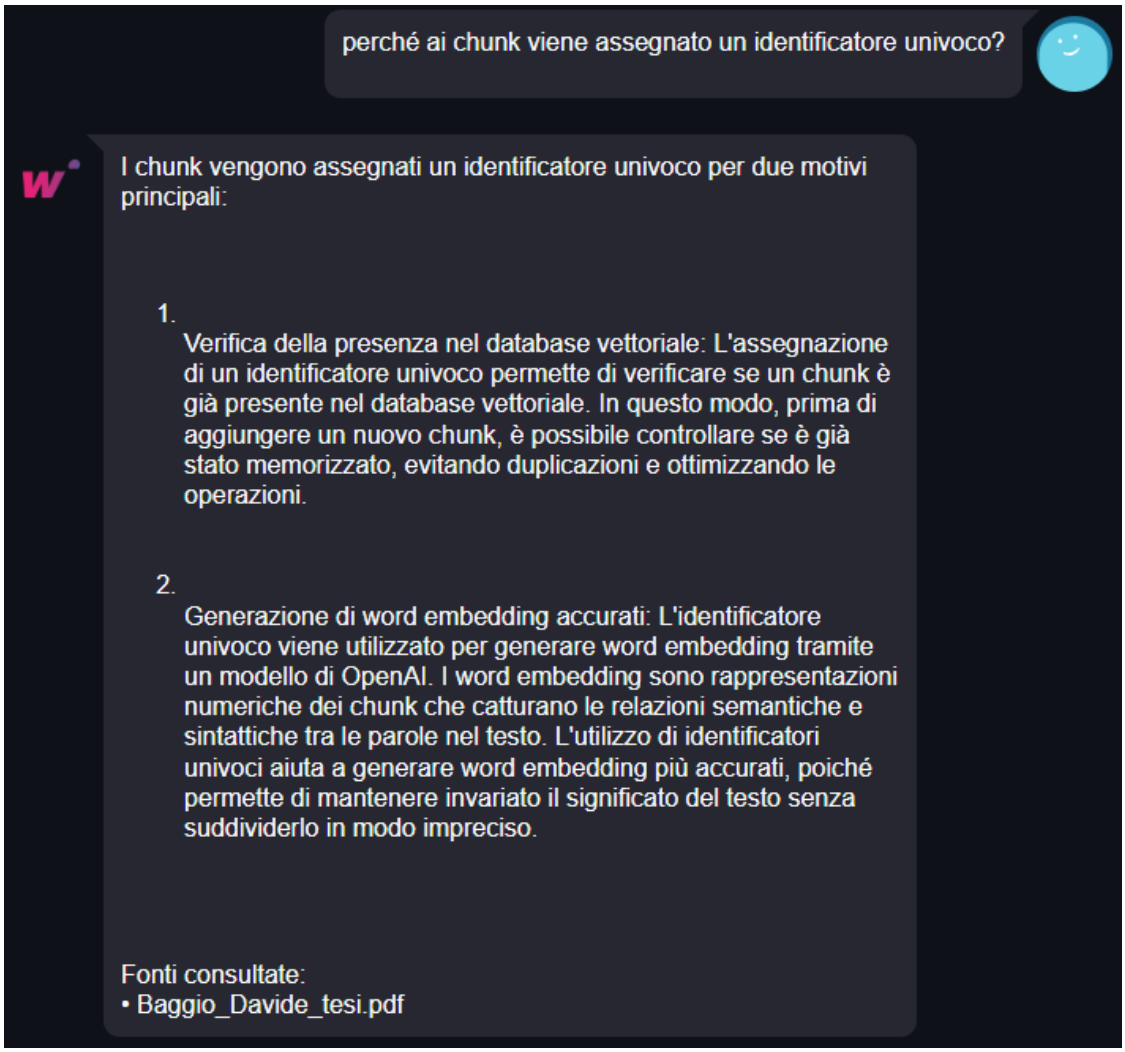


Figura 4.10: Output domanda riguardante gli identificatori dei chunk

Capitolo 5

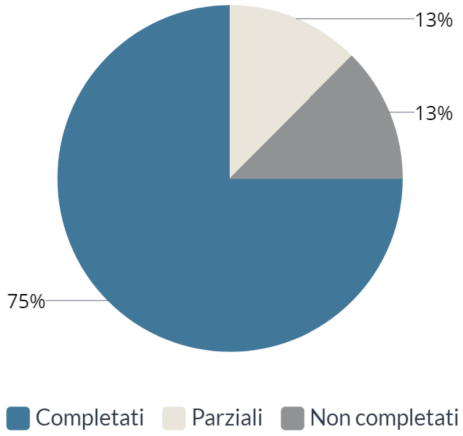
Conclusioni

5.1 Bilancio degli obiettivi

Nella seguente tabella vengono elencati tutti gli obiettivi, presenti nella sezione §3.2 della tesi, con il relativo stato di completamento al termine dello stage.

Tabella 5.1: Bilancio degli obiettivi

Nome	Stato
O1	Completato
O2	Completato
O3	Completato
D1	Completato
D2	Non completato
D3	Completato
F1	Parzialmente completato
F2	Completato



Quasi la totalità di tutti gli obiettivi concordati è stata portata a termine, ad eccezione di due obiettivi non obbligatori, ovvero:

- **D2** - Integrazione con il sistema di marketing dei prodotti aziendali.
  In corso d'opera del progetto, insieme al tutor, si è optato per non integrare il software da me prodotto all'interno della piattaforma proprietaria (*WoW*), in quanto sarebbe stato troppo dispendioso e non utile ai fini dell'obiettivo dello stage. Implementare la web app all'interno dell'ecosistema avrebbe sottratto una quantità di tempo significativa a due attività irrinunciabili per questo progetto, ossia la ricerca e la sperimentazione;
- **F1** - Realizzare la presentazione interna utilizzando l'applicazione prodotta.
  L'obiettivo è stato classificato come parziale poiché, sebbene la presentazione sia stata effettivamente creata attraverso l'applicazione sviluppata, è stato necessario apportare alcune modifiche al contenuto al fine di adattarlo alle

esigenze personali. Questo approccio innovativo ha evidenziato la versatilità dell'applicazione e la sua capacità di essere impiegata con successo in contesti reali.

Il bilancio degli obiettivi evidenzia il solido progresso compiuto nel corso dello stage, portando a termine un prodotto che rappresenta una solida base per le prossime fasi di sviluppo, qualora l'azienda decidesse di procedere con l'implementazione di tale software all'interno dei propri prodotti.

5.2 Conoscenze acquisite

Durante lo svolgimento del progetto, ho avuto l'opportunità di acquisire preziose competenze lavorative e interpersonali all'interno di un'azienda, in particolare in un team di ricerca e sviluppo. Il focus del progetto è stato senza dubbio molto stimolante, in quanto mi ha permesso di esplorare campi dell'intelligenza artificiale che precedentemente mi erano sconosciuti, acquisendo quindi maggior conoscenza e consapevolezza sul funzionamento dei modelli di ML in un contesto di generazione di contenuti, recupero delle informazioni e question & answering.

Un aspetto che ho senza dubbio sviluppato, e migliorato giorno dopo giorno, è quello della gestione del progetto: lavorare attivamente all'interno di un team mi ha permesso di migliorare la mia abilità nell'organizzare le attività, assegnare le risorse con efficienza e, soprattutto, comunicare in modo più efficace. Infatti, grazie alle frequenti interazioni ho capito quanto sia fondamentale comunicare regolarmente con altri membri del reparto per aggiornarsi sullo stato di avanzamento, effettuare brevi dimostrazioni sulle funzionalità implementate, ma anche esprimere dubbi e perplessità su che scelta intraprendere. Questa esperienza ha evidenziato quanto sia importante una comunicazione chiara e concisa per poter raggiungere senza ostacoli durante il percorso un obiettivo comune.

Dal punto di vista delle conoscenze tecniche, ho avuto l'opportunità di poter utilizzare tecnologie presenti sul mercato da pochissimi mesi e che rappresentano quindi una novità assoluta e lo stato dell'arte nel settore IT. Questo mi ha permesso di ampliare la mia conoscenza delle librerie Python specificamente sviluppate per contesti che riguardano l'utilizzo delle IA: LangChain si è rivelato essere un framework molto performante per poter implementare algoritmi e lavorare efficientemente in questo settore.

5.3 Valutazione retrospettiva personale

Nei due mesi lavorativi trascorsi in Wintech ho potuto apprendere molte nozioni e dinamiche che difficilmente avrei potuto imparare altrimenti. Essere integrato per la prima volta in un ambiente professionale mi ha permesso di avere un assaggio di come funzioni dall'interno un'azienda, quali sono gli obiettivi a cui questa punta e poter quindi concretizzare di conseguenza, in maniera pratica, tutte le nozioni teoriche che ho appreso durante il mio percorso di studi universitario.

Ho trovato estremamente gratificante essere coinvolto in situazioni reali durante lo stage, poiché ciò mi ha permesso di colmare il divario esistente tra la teoria e l'applicazione pratica. Questa esperienza mi ha consentito di sviluppare una comprensione più approfondita di come i concetti e i principi teorici possano essere effettivamente messi in pratica in un ambiente professionale. Inoltre, ho migliorato notevolmente le mie capacità di risolvere problemi e di condurre analisi critiche, adottando un approccio pragmatico e orientato alle soluzioni.
Credo che l'opportunità di stage offerta dall'Università di Padova non sia solo un'esperienza significativa, ma rappresenti anche un valore aggiunto in quanto agevola l'inserimento degli studenti informatici nel mondo del lavoro, un aspetto di vitale importanza ai giorni nostri.

Al termine della mia esperienza di stage mi sono trovato ad affrontare un bivio: accettare la proposta del mio tutor di proseguire la mia permanenza in azienda o continuare il percorso di studi universitario conseguendo una laurea magistrale. La scelta non è stata affatto semplice, considerando che ho avuto la fortuna di trascorrere due mesi in un'azienda accogliente, composta da professionisti con molta esperienza e conoscenza da condividere. Tuttavia, ho optato per la seconda opzione, poiché credo che un corso di laurea magistrale possa fornirmi ulteriori conoscenze per arricchire il mio bagaglio culturale.

Acronimi e abbreviazioni

ANN Approximate Nearest Neighbor. 17, 47

API Application Program Interface. 47

BPE Byte-Pair Encoding. 9

CI Continuous Integration. 47

CLI Command Line Interface. 47

CRUD Create Read Update Delete. 17, 48

DBMS Database Management System. 16, 48

DPO Data Protection Officer. 48

ERP Enterprise Resource Planning. 48

GPT Generative Pre-trained Transformer. 9, 15, 19, 20, 25, 29, 48

ICT Information and Communication Technology. 48

IDE Integrated Development Environment. 49

IT Information Technology. 44, 48

ITS Issue Tracking System. 49

LLM Large Language Model. 5, 9–12, 15, 29, 34, 49

ML Machine Learning. 8, 12, 17, 29, 44, 49

PMI Project Management Institute. 49

PoC Proof of Concept. 20, 27, 49

SHA-1 Secure Hash Algorithm 1. 49

UI User Interface. 37

UUID Universally Unique Identifier. 33, 50

XAI Explainable Artificial Intelligence. 12

Glossario

API

Application Programming Interface, indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma. La finalità è ottenere un'astrazione, di solito tra l'hardware e il programmatore o tra software a basso e quello ad alto livello semplificando così il lavoro di programmazione. 19, 20, 23, 25, 27, 29, 46

ANN

Approximate Nearest Neighbor, indica un algoritmo che, dato un insieme di punti nello spazio, trova per ogni punto il punto più vicino in un altro insieme di punti, in modo approssimato. 17, 46

Auto-attenzione

è un meccanismo di apprendimento automatico che mette in relazione diverse posizioni di una sequenza per calcolare una rappresentazione di quella sequenza. Nell'elaborazione del linguaggio naturale, questo processo di solito considera il rapporto tra le parole della stessa frase. 5–7, 47

Black Box

è un sistema che può essere osservato in termini di input e di output, senza alcuna spiegazione sul come esso funzioni. 12, 47

Chunk

è una porzione di testo che viene suddivisa in base ad un determinato parametro, ad esempio la lunghezza in caratteri. 33, 34, 36, 47

CLI

Command Line Interface, indica un'interfaccia utente a caratteri, per l'interazione con un computer, in cui l'utente impartisce comandi al computer digitando delle righe di testo predefinite. 37, 46

CI

Continuous Integration, indica una pratica di sviluppo software che prevede che gli sviluppatori integrino frequentemente il proprio lavoro con quello degli altri membri del team. Ogni integrazione viene verificata da un processo di build automatizzato che permette di individuare errori di integrazione in modo rapido. 4, 46

DPO
Data Protection Officer, è una figura professionale che si occupa di garantire il rispetto della normativa sulla protezione dei dati personali all'interno di un'organizzazione. 26, 46

DBMS
Database Management System, indica un sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione efficiente di database. 46

Database vettoriale
è un database specializzato nel memorizzare array multidimensionali, sono utilizzati per rendere le operazione CRUD efficienti su tali strutture dati. 16, 17, 29, 33, 34, 39, 40, 48

ERP
Enterprise Resource Planning, indica un sistema di gestione che integra tutti i processi di business rilevanti di un'azienda (vendite, acquisti, gestione magazzino, contabilità, ecc.) in un unico sistema software. 2, 46

Feed-forward
nel campo delle reti neurali, è un'architettura in cui le connessioni tra i nodi non formano un ciclo. I dati si muovono in una direzione, dal nodo di input al nodo di output. 5–7, 48

Framework
indica un'architettura logica di supporto su cui un software può essere progettato e realizzato, spesso facilitandone lo sviluppo da parte del programmatore. 29, 30, 36, 37, 44, 48

GPT
Generative Pre-trained Transformer, è un modello di linguaggio basato su trasformatori che utilizza l'apprendimento automatico per generare testo e comprendere il linguaggio naturale in modo avanzato. Viene addestrato su enormi quantità di testo presi da diverse fonti, come libri, articoli, pagine web e molto altro. L'obiettivo principale di questi modelli è di catturare e apprendere i modelli di linguaggio, la semantica e le relazioni tra le parole al fine di generare testo coerente e significativo. 5, 9, 15, 19, 20, 25, 29, 46

ICT
Information and Communications Technology, indica l'insieme dei metodi e delle tecnologie utilizzate per la gestione e l'elaborazione delle informazioni, in particolare l'uso di computer e apparecchiature di telecomunicazione per archiviare, recuperare, trasmettere e manipolare dati. 1, 46

IT
Information Technology, indica l'insieme dei metodi e delle tecnologie che realizzano la trasmissione, la ricezione e l'elaborazione di dati e informazioni. 18, 44, 46

IDE
Integrated Development Environment, indica un ambiente di sviluppo ovvero un software che, in fase di programmazione, supporta i programmatori nello sviluppo e debugging del codice sorgente di un programma. 4, 46

ITS
Issue Tracking System, indica un software che gestisce e mantiene una lista di problemi, di solito denominati *issue*, riscontrati durante lo sviluppo di un progetto software. 4, 46

LLM
Large Language Model, sono modelli di linguaggio che utilizzano l'apprendimento automatico e l'intelligenza artificiale per generare testo e comprendere il linguaggio naturale in modo avanzato. Sono modelli neurali profondi che vengono addestrati su enormi quantità di testo presi da diverse fonti, come libri, articoli, pagine web e molto altro. L'obiettivo principale di questi modelli è di catturare e apprendere i modelli di linguaggio, la semantica e le relazioni tra le parole al fine di generare testo coerente e significativo. 5, 9–12, 15, 29, 34, 46

ML
Machine Learning, indica un insieme di metodi di analisi dati che automatizzano la costruzione di modelli analitici. È un ramo dell'intelligenza artificiale basato sull'idea che i sistemi possono imparare dai dati, identificare modelli e prendere decisioni con il minimo intervento umano. 8, 12, 17, 29, 44, 46

PMI
Le *Piccole e Medie Imprese* sono imprese di dimensioni relativamente contenute che svolgono un ruolo significativo nell'economia di molti paesi. Rappresentano una vasta gamma di aziende che variano in base al settore, al numero di dipendenti, al fatturato e ad altri parametri.. 1, 46

PoC
Proof of Concept, indica una realizzazione incompleta o abbozzata di un determinato progetto o metodo, allo scopo di dimostrarne la fattibilità o la fondatezza o per dimostrare una determinata funzionalità. 3, 20, 27, 46

Repository
indica un ambiente di un sistema informativo in cui vengono gestiti i metadati, attraverso tabelle relazionali. In particolare, un repository è un archivio in cui vengono raccolti e ordinati dati e informazioni. 4, 49

SHA-1
Secure Hash Algorithm 1, è una funzione crittografica che prende in input un messaggio di lunghezza variabile e restituisce in output un valore di lunghezza fissa, detta digest, di 160 bit. 33, 46

Sharding
è una tecnica di partizionamento dei dati che divide i dati in più database. In questo modo, ogni database può essere gestito da un server separato, riducendo così il carico di lavoro di ogni server. 17, 50

System Integrator
indica un'azienda che si occupa di integrare hardware e software di terze parti per realizzare un sistema completo. 1, 50

UUID
Universally Unique Identifier, è un identificativo standard usato per identificare in modo univoco informazioni in un sistema informatico. 33, 46

Word Embedding
è una tecnica che trasforma un testo in vettori numerici continui di dimensione fissa in uno spazio multidimensionale, catturando le relazioni semantiche e sintattiche che legano le parole in un determinato testo. 14, 15, 29, 33, 34, 50

Bibliografia

Siti web consultati

Digital Transformation. URL: https://www.wintech.it/innoviamo/che-innovazione-cerchi/digital-transformation/ (cit. a p. 2).

Enrico Merigliano - Responsabile Ricerca e Sviluppo/Tutor interno dello stage. URL: https://www.wintech.it/siamo/ricerca-e-sviluppo/ (cit. a p. 3).

Paper

A. Vaswani e N. Shazeer e N. Parmar e J. Uszkoreit e L. Jones, A. N. Gomez e L. Kaiser e I. Polosukhin. «Attention Is All You Need». In: *arXiv:1706.03762* (2017). URL: https://arxiv.org/abs/1706.03762 (cit. a p. 6).

G. Xexéo, F. Almeida e. «Word Embeddings: A Survey». In: *arXiv:1901.09069* (2019). URL: https://arxiv.org/abs/1901.09069 (cit. a p. 14).

K. Chen e G. Corrado e J. Dean, T. Mikolov e. «Efficient Estimation of Word Representations in Vector Space». In: *arXiv:1301.3781* (2013). URL: https://arxiv.org/abs/1301.3781 (cit. a p. 14).

M. Wolting e J. Katzy e J. Kloppenburg e T. Verbelen e J. S. Rellermeyer, J. Verbraeken e. «A Survey on Distributed Machine Learning». In: *ACM Computing Surveys* (2020). URL: https://dl.acm.org/doi/abs/10.1145/3377454 (cit. a p. 8).

S. Zhang, Q. Jiao e. «A Brief Survey of Word Embedding and Its Recent Development». In: *IEEE* (2021). URL: https://ieeexplore.ieee.org/abstract/document/9390956 (cit. a p. 15).