Università degli Studi di Camerino

Scuola di Scienze e Tecnologie

Corso di Laurea in Informatica (Classe L-31)

Classificazione dei Domini Applicativi di Modelli BPMN attraverso Large Language Models

Laureando
Anthony Eleuteri

Matricola 110386

Relatore
Marco Piangerelli

Correlatore
Ivan Compagnucci

A.A. 2022/2023

Indice

Elenco dei codici

Elenco delle figure

Elenco delle tabelle

1. Introduzione

Nell'ambito dell'Intelligenza Artificiale (IA), il rapido sviluppo di tecnologie avanzate ha promosso notevoli trasformazioni in svariati settori, tra cui l'assistenza sanitaria, la guida autonoma, l'analisi finanziaria e la traduzione automatica[Tak]. Tra le molteplici discipline coinvolte, il Natural Language Processing (NLP) ha sperimentato progressi significativi grazie alla diffusione di modelli di apprendimento profondo (Deep Learning).

Questa tesi indaga sull'applicazione dei modelli di apprendimento profondo, focalizzando in particolare l'attenzione sul modello GPT-2, nell'ambito della classificazione di sequenze di testo relative ai modelli BPMN (Business Process Model and Notation) caratterizzati da specifici domini all'interno di dataset eterogenei. L'obiettivo primario consiste nella realizzazione di un sistema di classificazione in grado di identificare automaticamente il dominio associato a una particolare sequenza di testo, basandosi sulle etichette assegnate agli elementi del processo.

Le etichette, strumenti essenziali in questo contesto, fungono da veicolo di informazioni supplementari concernenti i molteplici aspetti dei modelli BPMN, quali i nomi, le descrizioni, i ruoli, i commenti e altri attributi rilevanti. L'impiego di modelli di apprendimento profondo, come GPT-2, mira a sfruttare la comprensione e l'analisi avanzata della struttura e del contenuto del testo al fine di agevolare la categorizzazione automatica delle sequenze di testo in contesti BPMN.

L'implementazione di un sistema di classificazione basato su modelli di apprendimento profondo per l'identificazione dei domini delle sequenze di testo riveste un'importanza cruciale nell'ambito dell'analisi dei dati e della classificazione automatizzata. Tale sistema potrebbe offrire un sostegno fondamentale agli attori coinvolti nei processi decisionali e nell'interpretazione dei dati, contribuendo così all'ottimizzazione delle operazioni e alla miglior comprensione dei diversi contesti di dominio.

1.1 Obiettivi della ricerca

Gli obiettivi della ricerca sono i seguenti:

- Analizzare l'efficacia dei modelli GPT-2 nella classificazione di sequenze di testo rispetto a domini specifici, con particolare attenzione alla classificazione di testo relativo ai processi BPMN.
- Esplorare l'applicazione dei modelli di Deep Learning nella classificazione di sequenze di testo relative a domini specifici, con particolare attenzione alla classificazione di testo relativo ai processi BPMN.
- Investigare la presenza di correlazioni tra le etichette e i domini dei processi BPMN, nonché tra gli elementi di flusso dei processi e le etichette, al fine di identificare eventuali pattern o relazioni significative che possano migliorare la classificazione automatica delle sequenze di testo.
- Valutare l'impatto delle varie tecniche di classificazione di testo nella categorizzazione automatizzata dei dati testuali, esaminando metriche standard di valutazione e confrontando i risultati ottenuti con modelli di riferimento nel campo del Natural Language Processing.
- Valutare le prestazioni del sistema di classificazione attraverso metriche standard di valutazione, come precisione, recall, e F1-score, e confrontare i risultati ottenuti con modelli di riferimento nel campo del Natural Language Processing.
- Implementare un Front-End intuitivo e User-Friendly per agevolare l'interazione dell'utente con il modello di classificazione realizzato, facilitando la raccolta di feedback, eventuali errori di progettazione e quindi contribuendo alla sua adattabilità e usabilità.

L'obiettivo finale è quello di approfondire la comprensione dell'applicazione dei modelli di deep learning nella classificazione del testo in contesti specifici, come quelli correlati ai processi BPMN, al fine di contribuire allo sviluppo di metodologie e strumenti per l'analisi e la categorizzazione automatizzata dei dati testuali. Inoltre, si prevede di implementare un Front-End intuitivo che consenta un'interazione rapida e semplice con il modello in questione, facilitando così la raccolta di feedback e migliorando l'usabilità complessiva del sistema.

1.2 Struttura della Tesi

La tesi è strutturata nel seguente modo:

1. **Introduzione**
    - **Motivazione**: Questa sezione presenta le ragioni che hanno portato alla scelta del tema di ricerca e dell'argomento trattato.
    - **Obiettivi**: Vengono definiti gli obiettivi della ricerca, delineando cosa si intende ottenere con il lavoro svolto.
2. **Fondamenti Teorici**

- Questo capitolo offre un'analisi approfondita dei concetti chiave relativi al deep learning, ai trasformatori e alla classificazione di testo.

3. **Metodologia**

   - Vengono descritte le modalità di raccolta dei dati, la progettazione del Dataset personalizzato, l'architettura del modello per la classificazione e le procedure di addestramento e valutazione.

4. **Tecniche Utilizzate**

   - Vengono presentate le tecniche e gli strumenti utili al fine di svolgere l'esperimento, incluse le librerie per la gestione dei dati.

5. **Risultati Sperimentali**

   - Vengono presentati i risultati dell'esperimento, inclusi i dati raccolti, le metriche di valutazione e le analisi qualitative.

6. **Discussione Finale**

   - Questo capitolo discute in dettaglio i risultati ottenuti, confrontandoli con la letteratura esistente e evidenziando le implicazioni pratiche e le direzioni future della ricerca.
   - Infine, vengono riassunti i principali risultati e le conclusioni della ricerca, insieme a considerazioni finali e possibili sviluppi futuri.

La presente tesi si propone di contribuire alla comprensione e all'avanzamento delle tecnologie di NLP, fornendo nuove intuizioni e metodologie per la classificazione di testo proveniente dalle etichette dei modelli BPMN in domini specifici.

2. Fondamenti Teorici

In questa sezione, si procederà all'esposizione dei fondamenti teorici fondamentali che costituiscono la base concettuale su cui è basata la tesi. Particolare attenzione sarà dedicata all'illustrazione dei concetti e degli strumenti impiegati per l'analisi dei dati, fornendo così un quadro esaustivo e approfondito delle basi teoriche su cui si fonda il lavoro di ricerca.

2.1 Che cos'è il Machine Learning?

Il Machine Learning è il termine generico utilizzato per descrivere il processo di un sistema computazionale che esegue attività senza istruzioni esplicite, indentificando modelli di dati e prevedendo risultati accurati per scenari sconosciuti o nuovi. In poche parole il Machine Learning si trova all'intersezione tra informatica e statistica attraverso il quale i computer ricevono la capacità di apprendere senza essere programmati esplicitamente.

UC Berkeley [Ber] suddivide il sistema di apprendimento di un algoritmo di Machine Learning in tre parti principali:

1. **Processo decisionale**: Una ricetta di calcoli o passaggi che prende in considerazione i dati e prova a prevedere quale tipo di modello il tuo algoritmo sta cercando di trovare.

2. **Funzione di errore**: Un metodo per misurare quanto fosse buona l'ipotesi confrontandola con esempi noti, se disponibili. Il processo decisionale è andato bene? In caso contrario, come quantificare "quanto grave" è stata la mancanza?

3. **Processo di aggiornamento/ottimizzazione**: Un metodo in cui l'algoritmo esamina l'errore e quindi aggiorna il modo in cui il processo decisionale arriva alla decisione finale, quindi la prossima volta l'errore non sarà così eccezionale.

Poiché un algoritmo di apprendimento automatico si aggiorna in modo autonomo, la precisione analitica migliora ad ogni esecuzione poiché apprende da solo dai dati che analizza. Questa natura iterativa dell'apprendimento è unica e preziosa perché avviene senza l'intervento umano, consentendo all'algoritmo di scoprire intuizioni nascoste senza essere specificatamente programmato per farlo.

Sruttando dunque l'apprendimento automatico, uno sviluppatore può migliorare l'efficienza di un'attività che coinvolge grandi moli di dati senza la necessità di input umani manuali.

Potenti algoritmi di apprendimento automatico possono essere utilizzati per migliorare la produttività dei professionisti che lavorano nella scienza dei dati, nell'informatica

e in molti altri campi, esistono numerosi algoritmi di apprendimento automatico comunemente utilizzati. Ciascuno di questi algoritmi di apprendimento automatico può avere numerose applicazioni in una varietà di contesti educativi e aziendali diversi:

- **Regressione Lineare**: Uno dei modelli più semplici e comunemente usati nel machine learning. Viene utilizzato per analizzare la relazione tra una o più variabili indipendenti e una variabile dipendente continua. Ad esempio, può essere impiegato per prevedere il prezzo di una casa basandosi su variabili come la dimensione della casa, il numero di stanze, eccetera.

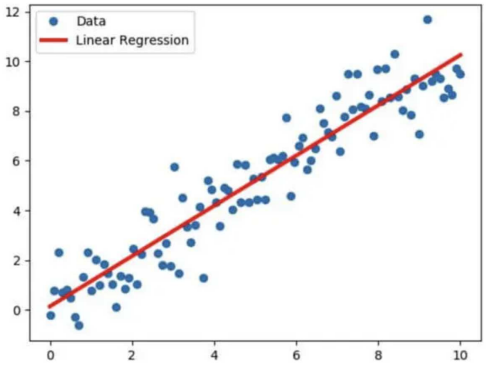

Figura 2.1: Grafico di una Regressione Lineare.

- **Regressione Logistica**: Questo algoritmo è utilizzato per la classificazione binaria, ovvero per prevedere la probabilità di appartenenza di un'osservazione a una delle due classi. È ampiamente utilizzato in applicazioni come il rilevamento di spam nelle email o la previsione di crediti inadempiuti.

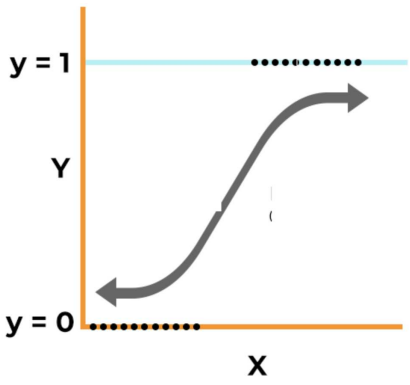

Figura 2.2: Grafico di una Regressione Logistica.

- **Reti Neurali**: Ispirate al funzionamento del cervello umano, le reti neurali sono costituite da neuroni artificiali interconnessi in strati. Possono apprendere da insiemi di dati complessi e sono adatte a una vasta gamma di compiti, tra cui riconoscimento di immagini, traduzione automatica e previsione del mercato azionario.

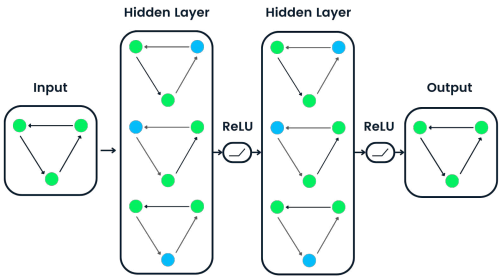

Figura 2.3: Grafico di una Rete Neurale.

- **Alberi Decisionali**: Questi modelli utilizzano una struttura ad albero per rappresentare le decisioni e le loro conseguenze. Sono facili da interpretare e possono essere utilizzati per problemi di classificazione e regressione. Gli alberi decisionali sono spesso utilizzati in settori come il marketing, dove possono aiutare a identificare i segmenti di mercato più redditizi.

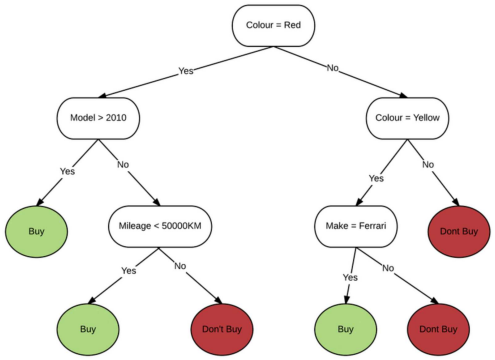

Figura 2.4: Grafico di un Albero Decisionale.

- **Foreste Casuali**: Si tratta di un insieme di alberi decisionali, ognuno dei quali viene addestrato su un sottoinsieme casuale dei dati di addestramento. Le previsioni dei singoli alberi vengono poi combinate per ottenere una previsione più accurata e robusta. Le foreste casuali sono apprezzate per la loro capacità di gestire grandi insiemi di dati e di gestire problemi complessi di classificazione e regressione.

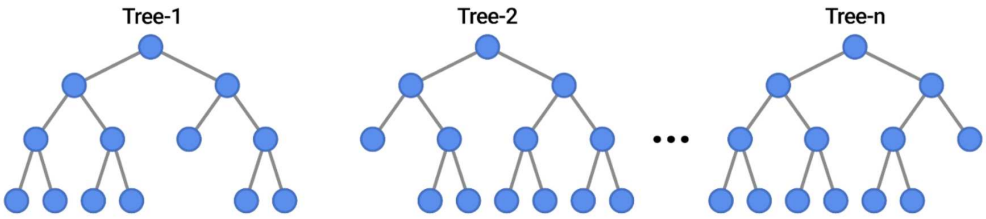

Figura 2.5: Grafico di una Foresta Casuale.

2.2 Che cos'è il Deep Learning?

In questa sezione, sarà esplorata e introdotta la definizione e il contesto del deep learning, con particolare attenzione sul funzionamento della tecnologia, le sue differenze rispetto alle reti neurali tradizionali, le sue applicazioni e il suo potenziale. Si farà un'attenzione particolare anche sulle limitazioni del deep learning per fornire una visione completa e approfondita di questa importante area dell'intelligenza artificiale.

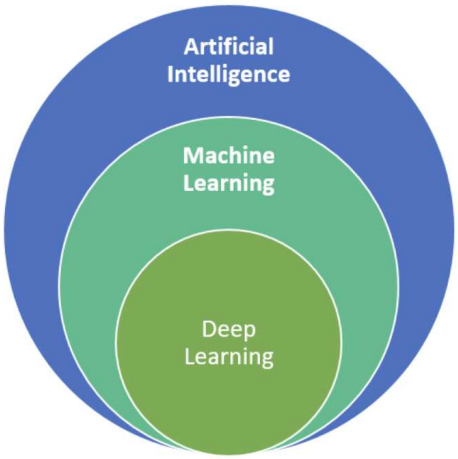

Figura 2.6: Intelligenza artificiale, Machine Learning e Deep Learning.

2.2.1 Definizione e contesto del Deep Learning

Il deep learning rappresenta un sottoinsieme del machine learning (ML), che a sua volta costituisce un ambito all'interno dell'intelligenza artificiale (IA). In particolare, il deep learning si distingue all'interno del ML per la sua capacità di apprendere automaticamente rappresentazioni complesse dei dati attraverso l'impiego di reti neurali artificiali. Questa tecnica è stata ampiamente applicata con successo in una vasta gamma di problemi, dalla visione artificiale, al trattamento del linguaggio naturale fino ad arrivare all'ambito medico[Lot]. Pertanto, il deep learning può essere considerato sia un sottoinsieme autonomo dell'IA che un sottoinsieme del ML, in quanto si concentra sull'apprendimento automatico di rappresentazioni stratificate dei dati. Il deep learning si basa sull'utilizzo di reti neurali artificiali, che sono modelli computazionali composti da una serie di strati di neuroni artificiali interconnessi. Ogni neurone è rappresentato da un modulo software che svolge funzioni matematiche per elaborare e trasformare i dati in ingresso. Sebbene i neuroni artificiali siano ispirati in parte al funzionamento dei neuroni biologici, essi sono modelli matematici semplificati progettati per approssimare le funzionalità neurali. Il funzionamento delle reti neurali artificiali è regolato da algoritmi e regole di ottimizzazione, piuttosto che da processi biologici. [Ora]

2.2.2 Struttura e Funzionamento delle Reti Neurali Artificiali

La connessione tra la concezione delle prime reti neurali nel 1943 e lo sviluppo delle moderne reti neurali artificiali risiede nella progressiva evoluzione delle conoscenze nel campo dell'intelligenza artificiale e della neuroscienza computazionale.

Warren McCulloch e Walter Pitts nel 1943, ispirandosi al funzionamento dei neuroni biologici, proposero un modello di rete neurale utilizzando circuiti elettrici. Questo lavoro pionieristico ha gettato le basi concettuali per lo sviluppo delle moderne reti neurali artificiali.

Successivamente, nel 1980, Kunihiko Fukushima ha progettato la prima rete neurale convoluzionale (CNN), una tipologia di rete feed-forward in cui il pattern di connettività tra i neuroni è ispirato dall'organizzazione della corteccia visiva animale con l'obiettivo di creare un sistema computazionale in grado di risolvere problemi in modo simile al cervello umano[Wikc].

In sintesi, la ricerca e lo sviluppo delle reti neurali artificiali sono stati guidati dalla continua ispirazione e dall'applicazione dei principi biologici del cervello, combinati con i progressi nella tecnologia dell'informazione e dell'elaborazione dei dati.

Ma come funziona dunque una Rete Neurale?

Una rete neurale artificiale è composta da uno strato di input, uno strato di output (o target) e uno o più strati nascosti interposti tra essi. Gli strati sono collegati da nodi, e le connessioni tra di essi formano ciò che comunemente viene definito "network" o rete di nodi interconnessi.

Analogamente al funzionamento dei neuroni umani, i nodi di una rete neurale vengono attivati quando ricevono un sufficiente stimolo o input. L'attivazione si propaga attraverso la rete, generando una risposta agli stimoli (output). I segnali attraversano gli strati durante il loro percorso dalla prima all'ultima fase di output e vengono elaborati lungo il tragitto.

Quando la rete è sottoposta a una richiesta, i neuroni eseguono calcoli matematici per determinare se vi è sufficiente informazione da trasmettere al neurone successivo.

In parole semplici, analizzano tutti i dati e individuano le relazioni più rilevanti. Nelle reti neurali più semplici, gli input ricevuti vengono sommati e, se la somma supera una determinata soglia, il neurone si attiva e trasmette l'informazione ai neuroni collegati. Le architetture di apprendimento profondo, invece, portano le reti neurali a uno stadio più avanzato, creando reti neurali profonde attraverso l'aggiunta di strati nascosti all'interno della rete[Sas].

2.2.3 Qual'è dunque la differenza tra il deep learning e le reti neurali?

La principale differenza tra deep learning e reti neurali risiede nella profondità delle architetture utilizzate. Mentre le reti neurali tradizionali possono avere solo pochi strati, il deep learning coinvolge l'uso di reti neurali con molteplici strati (di solito più di tre), consentendo la creazione di modelli più complessi e potenti nell'elaborazione dei dati. Questo approccio profondo è cruciale per l'efficacia dei modelli di deep learning, in quanto consente loro di apprendere rappresentazioni complesse dei dati e di svolgere compiti di alto livello come il riconoscimento di immagini e il trattamento del linguaggio naturale con una precisione notevole[Har].

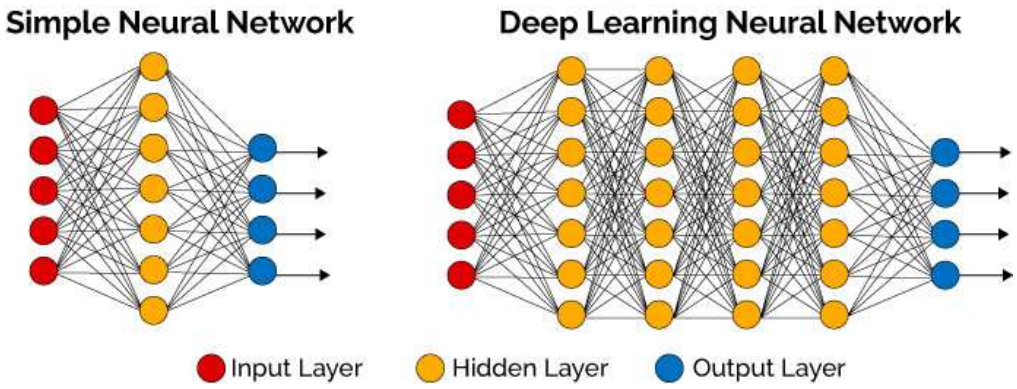

Figura 2.7: Differenza tra le reti neurali semplici e le reti neurali del deep learning. Fonte:The Data Scientist [Kam]

2.2.4 Applicazioni e Potenzialità del Deep Learning

Il campo del deep learning, derivato dalle pionieristiche reti neurali e successivi sviluppi nell'intelligenza artificiale, si caratterizza come una tecnologia estremamente versatile e con prospettive rivoluzionarie. Questa tecnologia avanzata ha dimostrato un impatto significativo in diversi ambiti, grazie alla sua abilità nell'apprendere rappresentazioni complesse e nell'identificare pattern intrinsechi nei dati. L'automazione di compiti precedentemente ritenuti esclusivamente umani è ora una realtà, consentendo ai sistemi intelligenti di eseguire attività quali il riconoscimento di oggetti in immagini, la traduzione automatica di lingue e l'analisi predittiva dei dati.

Inoltre, il deep learning ha aperto nuove frontiere nella ricerca scientifica, agevolando gli scienziati nell'affrontare sfide complesse attraverso l'analisi di grandi dataset. Il settore sanitario beneficia dell'applicazione del deep learning per la diagnosi medica precoce e personalizzata, sfruttando dati medici e immagini diagnostiche. Allo stesso

modo, nel settore dei trasporti, il deep learning promette di rivoluzionare la guida autonoma, rendendo le strade più sicure e riducendo gli incidenti stradali.

Il deep learning trova applicazione anche nel settore finanziario, dove contribuisce a migliorare le previsioni dei mercati finanziari identificando pattern nascosti nei dati finanziari e nelle transazioni. Infine, il suo impatto futuro potrebbe estendersi alla gestione delle sfide globali come il cambiamento climatico e la sostenibilità ambientale, analizzando grandi dataset ambientali per comprendere meglio i fenomeni climatici e sviluppare soluzioni per mitigare i loro impatti negativi.

In conclusione, il potenziale del deep learning è vasto e in costante espansione, determinato dalla nostra immaginazione, dalla disponibilità di dati di alta qualità e dal progresso tecnologico. Ulteriori sviluppi nella ricerca e nello sviluppo tecnologico promettono di aprire nuove prospettive di innovazione e progresso in diversi settori, contribuendo a migliorare la qualità della vita e a fronteggiare le sfide globali in modi senza precedenti.

2.3 Deep Learning, Trasformatori e Classificazione del Testo

Nel presente capitolo, ci si propone di condurre un'analisi esaustiva dei concetti fondamentali correlati al campo del deep learning, ai trasformatori e alla classificazione del testo. Il contesto attuale della ricerca in intelligenza artificiale ha visto una crescente attenzione verso le metodologie di deep learning, che rappresentano un paradigma di apprendimento automatico in grado di apprendere rappresentazioni di dati complessi attraverso l'impiego di reti neurali artificiali multistrato. Tale approccio si è dimostrato particolarmente efficace in una serie di compiti, dalla visione artificiale al trattamento del linguaggio naturale. Nel contesto specifico dell'elaborazione del linguaggio naturale (NLP), l'emergere dei trasformatori ha suscitato un interesse significativo. I trasformatori, introdotti da Vaswani nel 2017, costituiscono un'architettura innovativa che ha rivoluzionato il panorama dell'NLP, superando le limitazioni dei modelli precedenti basati su reti neurali ricorrenti (RNN) e reti neurali convoluzionali (CNN)[Vas+17]. La loro capacità di catturare relazioni a lungo raggio e di modellare sequenze di testo con efficacia li rende un componente chiave nelle applicazioni NLP moderne. Inoltre, la classificazione del testo rappresenta un ambito fondamentale nell'ambito dell'NLP, in cui l'obiettivo è assegnare automaticamente delle etichette o delle categorie a documenti o segmenti di testo. L'utilizzo di approcci basati su deep learning e trasformatori ha portato a risultati significativi in questo campo, consentendo di ottenere performance sempre più elevate nella comprensione e nell'analisi di testo. Pertanto, un'esplorazione dettagliata di tali concetti risulta cruciale per comprendere appieno le metodologie e le tecnologie attuali nel campo dell'elaborazione del linguaggio naturale e del deep learning.

2.3.1 I trasformatori e il ruolo nell'Elaborazione del Linguaggio Naturale

I Trasformatori costituiscono un'architettura di rete neurale fondamentale nel campo dell'elaborazione del linguaggio naturale (NLP). Questa architettura, basata su meccanismi di attenzione, ha la capacità di trasformare una sequenza di input in una sequenza

di output, apprendendo, nel processo, il contesto e le relazioni tra i diversi componenti della sequenza.

Precedentemente, i modelli di deep learning si concentravano principalmente su compiti di NLP che coinvolgevano la previsione della parola successiva in una sequenza, basandosi sulla parola precedente. Tuttavia, questi modelli mostravano difficoltà nel mantenere il contesto oltre una certa lunghezza di input, rendendo complessa la generazione di testi coerenti e significativi.

L'introduzione dei "Transformer" nell'ambito dell'intelligenza artificiale ha consentito di superare queste limitazioni, poiché tali modelli sono in grado di catturare relazioni complesse tra le parole di un testo, gestendo efficacemente dipendenze a lungo raggio. Ciò ha aperto la strada a una serie di vantaggi nell'ambito dell'NLP come la traduzione automatica, l'analisi dei sentimenti e altro ancora.

In sintesi, i Trasformatori rivestono un ruolo essenziale nell'ambito dell'elaborazione del linguaggio naturale, poiché consentono di gestire dipendenze complesse nel testo, facilitano l'addestramento di modelli su larga scala e agevolano la personalizzazione rapida per applicazioni più specifiche. La loro introduzione ha rappresentato un importante passo avanti nell'evoluzione delle tecnologie di NLP.

Infine, l'introduzione dei Transformer ha stimolato significativi progressi nella ricerca nell'NLP, portando a importanti innovazioni e avanzamenti tecnologici nel settore. La loro presenza continua a plasmare il futuro dell'elaborazione del linguaggio naturale, aprendo nuove prospettive e sfide nel campo della comprensione e generazione del linguaggio umano da parte delle macchine.

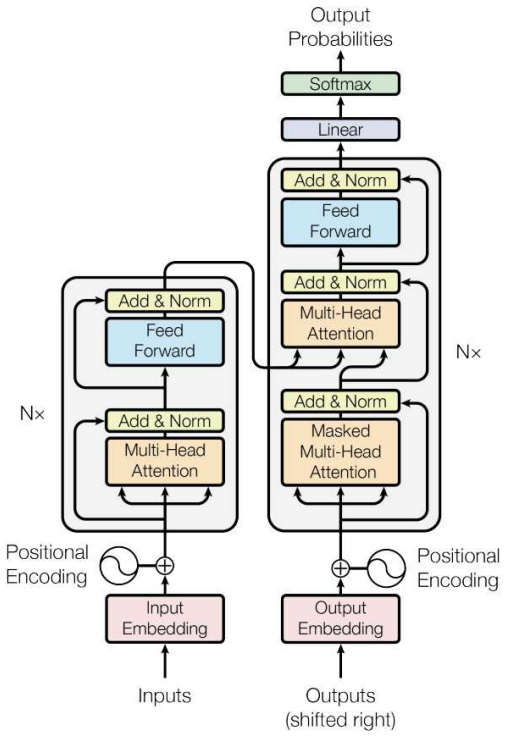

Figura 2.8: Architettura del modello di un Trasformatore - Fonte: Google Research [Vas+17]

2.3.2 Perché è importante la classificazione del testo in un contesto NLP?

La classificazione del testo, conosciuta anche come text tagging o text categorization, rappresenta il processo mediante il quale il testo viene organizzato in gruppi distinti. Questa tecnica di assegnamento è diventata sempre più diffusa in quanto consente di predire la categoria di un documento testuale senza la necessità di una sua lettura diretta[Mon].

La classificazione del testo può essere condotta utilizzando sia algoritmi di classificazione supervisionati che non supervisionati, a seconda delle caratteristiche del dataset e degli obiettivi dell'analisi[Del].

- **Classificazione Supervised**: In questo tipo di approccio, l'algoritmo viene addestrato su un dataset etichettato, in cui ciascun esempio è associato a una classe, un'etichetta o un dominio di appartenenza. L'algoritmo apprende dai dati di input e dalle relative etichette, cercando di creare un modello in grado di generalizzare e di classificare correttamente nuovi dati in input. L'obiettivo primario è quello di minimizzare l'errore tra le previsioni del modello e le etichette reali.

- **Classificazione Unsupervised**: In contrasto, in questo caso l'algoritmo opera su un dataset non etichettato, nel quale non sono disponibili informazioni sulla classe di appartenenza. L'obiettivo di questo tipo di algoritmo è individuare una struttura o un pattern intrinseco nei dati, senza la guida di etichette esterne.

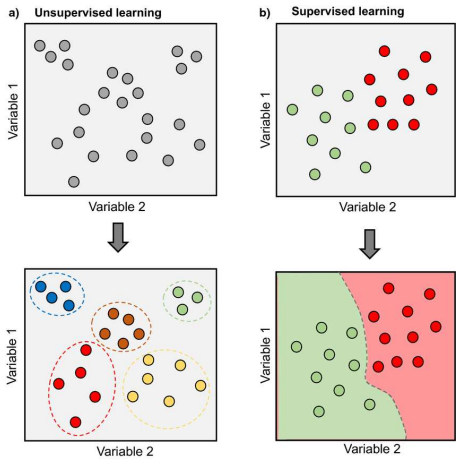

Figura 2.9: Supervised and unsupervised. Fonte: Researchgate [JM21]

La scelta tra le due modalità di classificazione dipende principalmente dagli obiettivi dell'analisi e dalle caratteristiche dei dati a disposizione. Se si dispone di dati etichettati e l'obiettivo è una classificazione accurata, allora si preferisce l'approccio supervisionato. Invece, se non si hanno etichette a disposizione o se si desidera individuare pattern intrinseci nei dati, allora l'approccio non supervisionato è preferibile.

2.4 Limiti e Considerazioni del Deep Learning

Nel campo dell'intelligenza artificiale, si ricerca sempre di più di emulare l'intelligenza delle macchine a quelle degli esseri umani. Tuttavia, si notano significative discrepanze tra l'approccio del deep learning e la vera natura dell'intelligenza umana secondo la filosofia dell'informazione. Quest'ultima definisce l'intelligenza come l'attività di un soggetto nell'acquisire, elaborare, creare, sviluppare, utilizzare e ottenere informazioni. L'intelligenza emerge da un processo informativo che si evolve dai livelli informativi più bassi, fino a generare nuove informazioni, coinvolgendo sia una progressione gerarchica dall'alto verso il basso, sia un'inibizione dall'alto verso il basso [TW22].

Nonostante i miglioramenti dei parametri durante il processo di apprendimento delle reti neurali, questi rimangono statici e predefiniti, privi della dinamicità e della trasformazione temporale necessaria per l'auto-organizzazione [Flo16].

Un'altra sfida del deep learning è la vasta quantità di dati richiesta per addestrare e successivamente validare il modello. Non tutti i dati sono ugualmente utili poiché potrebbero essere incompleti, rumorosi, incosistenti o fuorvianti, portando a risultati distorti.

Inoltre, il deep learning richiede notevoli risorse computazionali e tempo per l'allenamento e il rilascio dei modelli, poiché possono arrivare ad avere milioni o miliardi di parametri, rendendoli difficili da gestire in memoria e avviare su dispositivi.

In conclusione, l'approccio attuale del deep learning non è sufficiente per raggiungere l'obiettivo originale dell'intelligenza artificiale di creare sistemi informativi che manifestino un'intelligenza simile a quella umana, poiché mancano delle proprietà fondamentali della natura dell'informazione umana.

3. Metodologia

Nel contesto di questo capitolo, si fornirà una panoramica dettagliata delle fasi cruciali del nostro approccio ai dati. In particolare, si analizzeranno i dati utilizzati, la progettazione del Dataset personalizzato, l'architettura del modello per la classificazione e le procedure adottate per l'addestramento e la valutazione del modello.

Il nucleo concettuale alla base di questa ricerca è rappresentato dall'impiego di un modello di classificazione supervisionata. Tale scelta è motivata dalla natura dei nostri dati, i quali sono stati originariamente etichettati con il dominio di provenienza. L'obiettivo principale è dunque, quello di addestrare il modello utilizzando dati accurati e coerenti, al fine di ottenere una classificazione il più possibile precisa e affidabile.

L'utilizzo di un approccio di classificazione supervisionata garantisce che il modello sia istruito su un insieme di dati pre-etichettati, consentendo di apprendere dai dati di addestramento e di generalizzare correttamente su nuovi dati. Questo approccio si traduce in una maggiore capacità del modello di classificare in modo coerente i dati non visti durante il processo di addestramento, rappresentando così un fondamento solido per la nostra analisi e le nostre previsioni.

3.1 Descrizione dei Dati

I dati utilizzati nello studio derivano da modelli BPMN (Business Process Model and Notation), che consistono in diagrammi creati mediante un insieme predeterminato di elementi grafici. Questi diagrammi, ampiamente adottati da professionisti, rappresentano un metodo di modellazione dei flussi di lavoro che dettaglia le fasi di un processo pianificato, offrendo una visualizzazione chiara delle attività sequenziali e dei flussi informativi coinvolti nel completamento del processo.

3.1.1 Provenienza dei Dati

La fonte primaria dei dati utilizzati in questo studio è costituita da una varietà di collezioni online di modelli BPMN [Com+24]. I dati sono stati raccolti da sette diverse fonti, ciascuna contenente una vasta gamma di modelli BPMN, che includono:

1. **BIT Process Library**: Una collezione di 850 modelli resa disponibile dal gruppo IBM per la definizione e la validazione di un approccio per verificare le proprietà dei processi aziendali.

2. **Camunda BPMN**: Una raccolta di 3.739 modelli creati durante sessioni di formazione BPMN.

3. **eCH-BPM**: Una collezione di 117 modelli progettati da specialisti BPM e persone che lavorano in municipi pubblici svizzeri e agenzie federali.

4. **GenMyModel BPMN**: Una raccolta di 12.172 modelli derivati dalla piattaforma GenMyModel.

5. **GitHub BPMN**: Una collezione di 17.314 modelli disponibili pubblicamente su GitHub, recuperati attraverso un approccio di mining sistematico.

6. **RePROSitory BPMN**: Una raccolta di 593 modelli caricati sulla piattaforma RePROSitory.

7. **BPMAI**: Una collezione di 19.715 modelli derivati principalmente da attività di modellazione svolte dagli studenti all'interno di istituzioni accademiche e resi disponibili dall'iniziativa accademica BPM.

3.1.2 Metodo di Estrazione dei Dati

I dati utilizzati nella presente analisi sono stati estratti direttamente da un paper accademico redatto da membri di questa istituzione universitaria [Com+24]. Il paper in questione è intitolato A "*Study on the Usage of the BPMN Notation for Designing Process Collaboration, Choreography, and Conversation Models*" e rappresenta una fonte autorevole all'interno del contesto accademico.

3.1.3 Preparazione dei Dati

L'analisi si basa sui dati estratti da tali modelli BPMN, esportati nel formato .CSV per agevolare un'esame dettagliato degli elementi e dei simboli presenti nei diagrammi. Questo formato tabellare fornisce una panoramica accurata e strutturata delle componenti del processo, facilitando ulteriori analisi e trattamenti statistici.

In seguito, è stata condotta una fase di pulizia dei dati al fine di rimuovere marcatori della traduzione dei modelli BPMN in file .CSV, informazioni fuorvianti, incomplete o etichettate in modo improprio. Tale processo di pulizia è stato cruciale per garantire l'affidabilità e la coerenza dei dati analizzati, evitando che il modello di apprendimento automatico assimilasse informazioni erronee.

BIT	['s00001551', 's00001551', 's00001551']
BIT	['s00005840', 's00005840', 's00005840']
BIT	['s00006360', 's00006360', 's00006360']
BIT	['s00006385', 's00006385', 's00006385']
BIT	['s00006543', 's00006544']
BIT	['s00007121', 's00007121']
BIT	['s00001419', 's0001419']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'R']
BPMAI	['A', 'B', 'B', 'DefaultInputSet', 'DefaultOutputSet']
BPMAI	['A', 'B', 'C', 'D', 'E', 'E', 'F', 'F']
BPMAI	['Writefinancialreport']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O']
BPMAI	['A', 'B', 'C', 'C', 'D', 'DefaultInputSet', 'DefaultInputSet', 'DefaultOutputSet', 'DefaultOutputSet', 'E', 'F', 'F', 'G']
BPMAI	['A', 'A', 'B', 'C', 'D', 'E', 'E', 'F', 'F', 'G', 'G', 'H']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'K', 'L', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L']
BPMAI	['A', 'B', 'C', 'D', 'E']
BPMAI	['A', 'B', 'C', 'D', 'E', 'F', 'G']
BPMAI	['A', 'B', 'C', 'D']
BPMAI	['A', 'AA', 'AB', 'AC', 'AD', 'AE', 'AF', 'AF', 'AG', 'AH', 'AI', 'AJ', 'AK', 'AL', 'AM', 'AN', 'AO', 'AP', 'AQ', 'AR', 'AS', 'AT', 'AU', 'AU', 'AV', 'AW', 'AX', 'AY', 'AZ', 'B', 'BA', 'BB', 'BC', 'BD', 'BE', 'BF', 'BG', 'BH', 'BI', 'BJ', 'BK', 'BL', 'BM', 'BN', 'BO', 'BP', 'BQ', 'BR', 'BS', 'BT', 'BU', 'BV', 'BW', 'BX', 'BY', 'BZ', 'C', 'CA', 'CB', 'D', 'E', '
BPMAI	['']

Figura 3.1: Esempio di modelli inconsistenti.

La pulizia dei dati riveste un ruolo fondamentale in qualsiasi analisi dei dati, poiché dati grezzi contenenti errori o informazioni inconsistenti potrebbero influenzare la va-

lidità dei risultati ottenuti. Questo processo è stato eseguito mediante un'analisi dettagliata dei dati e l'applicazione di tecniche di pre-elaborazione.

E' stata eseguita a rimozione di stringhe di separazione:

Figura 3.2: Un esempio dell'eliminazione di una stringa di separazione.

Successivamente, è stata effettuata l'eliminazione di duplicati o stringhe prive di significato, la selezione di etichette con un numero adeguato di modelli per lo studio, la rimozione di tutte le stringhe di lingua non inglese ed in fine la separazione delle stringhe in parole:

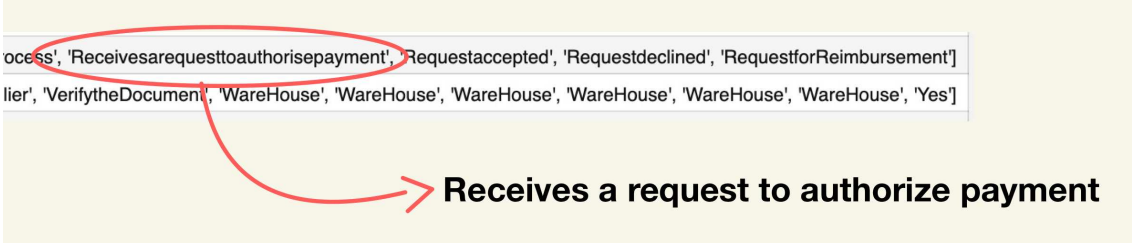

Figura 3.3: Un esempio di separazione di una stringa.

3.1.4 Limitazioni e Problemi riscontrati

Nel corso della preparazione del dataset, abbiamo dovuto confrontarci con diverse limitazioni e criticità che hanno influenzato considerevolmente la direzione della nostra analisi. In particolare, la fase di pulizia dei dati ha evidenziato alcune sfide cruciali.

1. **Eliminazione delle Discrepanze e dei Dati Fuorvianti**: La rimozione di dati inconsistenti, criptati o fuorvianti, seppur necessaria per garantire affidabilità e coerenza al dataset, ha comportato una significativa riduzione della sua varietà, con un conseguente ridimensionamento considerevole.

2. **Riduzione della Varianza dei Dati**: Al termine della pulizia, il dataset si è ridotto a includere solamente due domini distinti. Il dominio *eCH* è stato scartato a causa dell'insufficienza di dati per un'analisi accurata, mentre i dati del dominio *BIT* sono stati esclusi per l'utilizzo di una notazione di etichettatura criptata che ne ha causato la completa perdita dopo il primo processo di pulizia. Tale limitazione ha sollevato dubbi sulla validità e l'accuratezza di un'analisi di classificazione basata su un numero così esiguo di categorie

3. **Rischio di Banalizzazione nell'Analisi dei Dati**: La riduzione dei domini a soli due rischiava di portare ad un'analisi banale, in quanto la distinzione tra

due sole categorie potrebbe non essere sufficientemente dettagliata per cogliere le sfumature presenti nei dati.

3.2 Esplorazione di un Approccio Differente a Fronte delle Sfide Rilevate

Di fronte a queste sfide, abbiamo considerato un approccio innovativo per affrontare la complessità dei nostri dati e migliorare l'accuratezza della classificazione. Invece di limitarci a una classificazione tradizionale basata su due sole categorie, abbiamo esplorato l'opportunità di integrare diverse intelligenze artificiali per arricchire il processo decisionale.

Utilizzo di Intelligenze Artificiali Complementari Abbiamo proposto l'utilizzo congiunto di diverse intelligenze artificiali, tra cui ChatGPT, Gemini e Copilot, al fine di arricchire il nostro dataset con parole chiave e concetti rilevanti in diversi ambiti, come manufacturing, healthcare, logistics, education e public administration. Questo approccio ci consente di ampliare la comprensione dei dati, incorporando una vasta gamma di termini e concetti specifici di settore.

Vantaggi dell'Approccio Integrato:

- **Rappresentazione Multidimensionale dei Dati**: Integrando le intelligenze artificiali, possiamo ottenere una rappresentazione più completa e multidimensionale dei nostri dati, considerando una varietà di prospettive e contesti.
- **Aumento della Precisione nella Classificazione**: Aumento della Precisione nella Classificazione: Incorporando parole chiave e concetti specifici di settore, miglioriamo la precisione della nostra classificazione, consentendo al modello di distinguere in modo più accurato tra le diverse categorie.
- **Affinamento delle Analisi e delle Previsioni**: L'approccio integrato ci permette di affinare le nostre analisi e previsioni, considerando una gamma più ampia di informazioni e contesti rilevanti.

In conclusione, l'adozione di un approccio integrato basato su diverse intelligenze artificiali rappresenta una risposta alternativa alle sfide riscontrate nel nostro studio, consentendoci di affrontare in modo più completo e accurato la complessità dei nostri dati e di ottenere risultati più robusti e affidabili.

3.2.1 ChatGPT, Gemini e Copilot

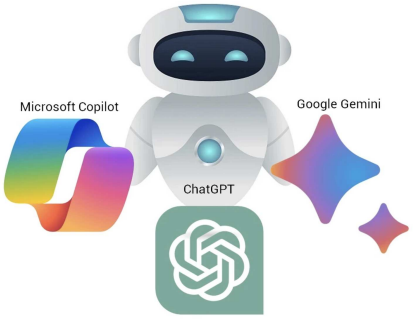

Figura 3.4: ChatGPT vs Gemini vs Copilot.

Cosa sono ChatGPT, Gemini e Copilot? Sono innanzitutto applicazioni software o interfacce web pensate per imitare la conversazione umana, usano l'intelligenza artificiale generativa (GAI), ovvero un tipo di intelligenza artificiale in grado di generare testo, immagini, video, musica o altre tipologie di media in risposta a delle richieste dette prompt, per permettere all'utente di mantenere una conversazione in un linguaggio naturale [Wika].

Alla fine del 2022, questa tecnologia ha preso sempre piede nel mondo comune, grazie proprio alla popolarità della tecnologia di OpenAI chiamata ChatGPT, seguita poi, successivamente, dalle tecnologie di Microsoft e Google, Copilot e Gemini.

Ognuna delle tecnologie sopra elencate è abilite di performare una varità estesa di differenti task, ma in maniera differente. In un mondo digitale sempre più veloce, questi strumenti offrono funzionalità uniche per aiutare gli utenti in una varietà di compiti. Esaminiamo quindi le differenze chiave tra questi assistenti:

1. **OpenAI ChatGPT**: Un Chatbot alimentato dall'IA sviluppato da OpenAI. Progettato per impegnare gli utenti in conversazioni naturali e coinvolgenti, utilizza il modello GPT (Generative Pre-trained Transformer) noto per la sua capacità di generare testo simile a quello umano in base all'input fornito dall'utente. ChatGPT è uno strumento versatile che può essere utilizzato per una varietà di scopi, tra cui supporto clienti, traduzione di lingue e creazione di contenuti.

2. **Microsoft Copilot**: Invece, è un assistente sviluppato da Microsoft per assistere gli sviluppatori software nella scrittura di codice in modo più efficiente. Utilizza il modello Codex, addestrato su un ampio dataset di repository di codice pubblicamente disponibili, per fornire suggerimenti e completamenti intelligenti del codice. Copilot può aiutare gli sviluppatori a scrivere codice più velocemente e con meno errori, rendendolo uno strumento prezioso per chiunque lavori nel campo dello sviluppo software.

3. **Google Gemini**: Un assistente sviluppato da Google per aiutare gli utenti a scoprire rapidamente nuovi interessi e attività. Utilizza una combinazione di apprendimento automatico e elaborazione del linguaggio naturale per fornire raccomandazioni personalizzate basate sulle preferenze e il comportamento dell'utente. Gemini può essere utilizzato per trovare nuovi ristoranti, film, libri e altro ancora, rendendolo uno strumento versatile per chiunque desideri ampliare i propri orizzonti.

Le differenze principali tra ChatGPT, Microsoft Copilot e Google Gemini possono essere riassunte come segue:

1. **Funzionalità Primaria**:
   - **ChatGPT**: Conversazioni coinvolgenti e generazione di testo.
   - **Microsoft Copilot**: Assistenza nella scrittura del codice.
   - **Google Gemini**: Raccomandazioni personalizzate per interessi e attività.
2. **Pubblico di Riferimento**:
   - **ChatGPT**: Aziende che desiderano automatizzare il supporto clienti o la creazione di contenuti.
   - **Microsoft Copilot**: Sviluppatori software alla ricerca di un aiuto nella scrittura del codice.
   - **Google Gemini**: Utenti alla ricerca di nuovi interessi e attività.
3. **Casi d'Uso**:
   - **ChatGPT**: Supporto clienti, Chatbot, creazione di contenuti.
   - **Microsoft Copilot**: Scrittura del codice, suggerimenti di completamento, correzione degli errori.
   - **Google Gemini**: Raccomandazioni personalizzate per ristoranti, film, libri e altro.
4. **Dati di Addestramento**:
   - **ChatGPT**: Dataset diversificato di linguaggio umano.
   - **Microsoft Copilot**: Dataset di repository di codice pubblicamente disponibili.
   - **Google Gemini**: Apprendimento automatico e comportamento dell'utente.
5. **Integrazione**:
   - **ChatGPT**: Integrazione semplice in piattaforme o applicazioni di chat esistenti.
   - **Microsoft Copilot**: Integrazione in Visual Studio Code e GitHub.
   - **Google Gemini**: Integrazione nel motore di ricerca Google e altre applicazioni Google.

In conclusione, ChatGPT, Microsoft Copilot e Google Gemini offrono funzionalità uniche, scegliendo l'assistente giusto in base alle esigenze, è possibile ottenere un supporto più efficace alle attività quotidiane di ogni genere [Ste; Woo].

3.2.2 Non-Chatbots

Nel panorama delle Intelligenze Artificiali, numerosi Large Language Models (LLM) sono impiegati per l'analisi e la generazione di testo. Tra essi, i Chatbot rappresentano uno strumento fondamentale per le interazioni conversazionali, e la gestione della cronologia dei messaggi assume un ruolo cruciale. Ad esempio, nell'ambito di sistemi come ChatGPT, ogni nuovo messaggio inviato richiede la consultazione dell'intera conversazione precedente, al fine di mantenere un contesto adeguato, poiché tali modelli non dispongono di una memoria autentica. Questa "memoria" virtuale della conversazione costituisce l'elemento distintivo principale rispetto ai sistemi non-Chatbot.

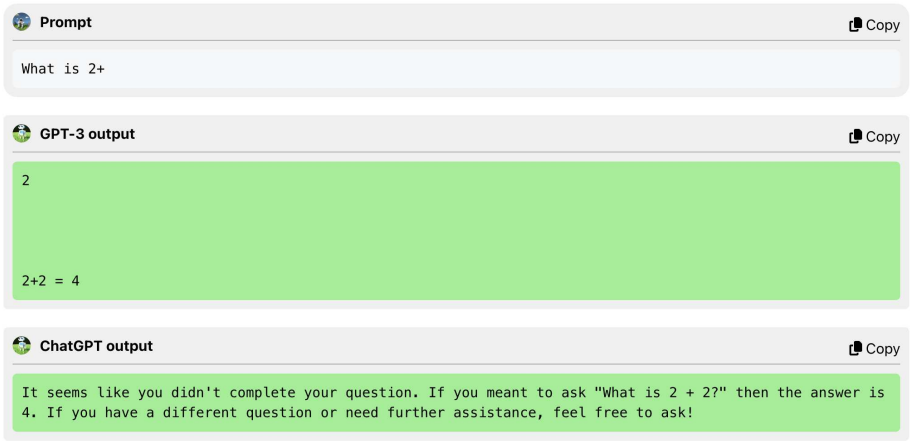

Figura 3.5: Risposte di GPT-3 and ChatGPT

Tuttavia, la distinzione tra ChatGPT e altri modelli di GPT va oltre il semplice aspetto conversazionale. Mentre GPT è un termine generico utilizzato per indicare i modelli generativi pre-addestrati, ChatGPT si focalizza specificamente sull'integrazione delle capacità di dialogo in applicazioni e piattaforme esistenti [Lea]. Ad esempio, OpenAI ha sviluppato un'API dedicata a ChatGPT, utilizzata da servizi come Instagram, Snap, Quizlet e Instacart per arricchire le loro funzionalità con l'intelligenza artificiale. Allo stesso modo, aziende come Google e Microsoft hanno introdotto rispettivamente i loro modelli ChatGPT-like, come Bard e Copilot, per offrire funzionalità di dialogo e assistenza nella scrittura di codice.

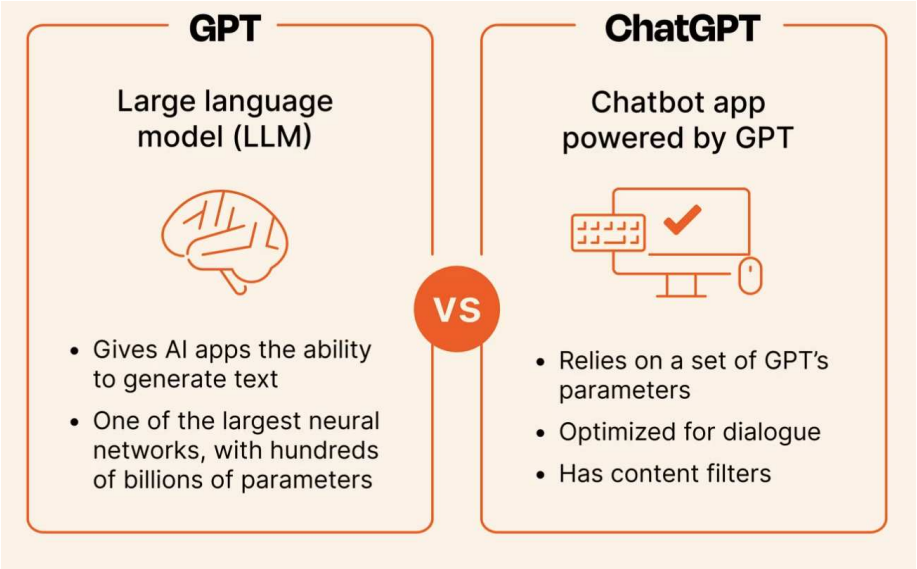

Figura 3.6: Speigazione di GPT-3 and ChatGPT

GPT, pertanto, rappresenta un linguaggio di modellazione piuttosto che un'applicazione definita, soggetto all'essere essere plasmato per soddisfare una vasta gamma di esigenze. Costituisce il cervello alla base dei Chatbot, consentendo loro di adattarsi e rispondere in modo flessibile a diversi contesti e richieste.

3.2.3 Cosa sono e come effettuare Prompt efficaci

I Prompt sono istruzioni o richieste che noi effettuiamo per interfacciarci con un'intelligenza artificiale da cui vogliamo una risposta. Un prompt è costituito da una serie di parole chiave e frasi destinate a stimolare una risposta da parte del Chatbot. I prompt sono dunque, ciò che determinano l'output del Chatbot, è dunque essenziale progettare ed eseguire prompt efficaci in modo da ottenere l'output che ci aspettiamo, migliore è la richiesta e migliore sarà la loro risposta. Ogni volta che parli con un Chatbot, lui impara a generare risposte più appropriate per te. Questo è il motivo per cui è importante fornire il messaggio giusto.

Ecco come è possibile farlo:

1. **Fornisci un contesto dettagliato**: I prompt migliori devono sempre includere una descrizione dettagliata del contesto di utilizzo, i Chatbot offrono spesso spazi per inserire le tue informazioni come "Prodotto X". Ma non adatta la risposta al tuo marchio, prodotto e pubblico target. Essendo più specifici e dettagliati si riesce ad ottenere dal Chatbot esattamente ciò di cui hai bisogno.

2. **Fonisci contenuti e materale di riferimento**: Molti Chatbot non posso navigare il Web, dunque, se hai bisogno di materiale o informazioni specifiche, dovresti fornirgli il contenuto della pagine a cui stai facendo riferimento.

3. **Fornisci chiaramente l'argomento o l'attività che desideri**: Come abbiamo visto nei punti precedenti, la specificità è ciò che lo rende efficace. Specificare l'attività o l'argomento che desideriamo che il Chatbot affronti. Ciò aiuta a focalizzare la risposta ed evitare deviazioni o risultati ambigui.

   Invece di un suggerimento generico come "Parla di strategie di marketing", sii specifico.

   Ad esempio, "Spiega tre strategie di marketing digitale efficaci per le piccole imprese" o "Fornisci suggerimenti per ottimizzare le campagne Google Ads".

4. **Imposta dei parametri**: Nell'ambito della specificazione dei vincoli per ottenere una risposta da un sistema di Chatbot, è essenziale stabilire chiaramente i parametri entro cui la risposta deve essere formulata. Questi vincoli possono riguardare diversi aspetti, tra cui il numero di parole, caratteri e frasi desiderati, nonché la quantità e la diversità delle risposte richieste.

   Ad esempio, nel contesto di una richiesta di generazione di titoli, è possibile definire il numero preciso di caratteri necessari per ciascun titolo, specificare il numero di titoli distinti richiesti e persino indicare punti di vista specifici da cui si desidera ricevere le risposte.

   Tale delineazione dei vincoli consente di ottenere risposte mirate e pertinenti alle esigenze dell'utente, garantendo al contempo un'adeguata varietà e completezza nell'output fornito dal sistema di Chatbot [Tea].

5. **Specifica il formato**: Fornisci al Chatbot indicazioni chiare e dettagliate riguardo al formato desiderato per la risposta. Questo potrebbe includere specifiche come il formato del testo, come una lista puntata o un formato Excel, oppure la richiesta di un racconto breve o dettagliato.

   L'aggiunta di queste informazioni al prompt permette al Chatbot di elaborare una risposta quanto più possibile utile alle esigenze dell'utente, garantendo una maggiore pertinenza e completezza nel risultato finale.

3.2.4 Cosa succede quando un Prompt non funziona

Nonostante le direttive precedentemente elencate, il prompt potrebbe risultare corretto ma il Chatbot non rispondere come dediseriamo, possiamo modificare il messaggio originale, talvolta chiedendo di essere più esaustivi o coincisi. Oppure, ponendo nuovamente la domanda utilizzando una formulazione diversa per ottenere risultati migliori.

Questo processo di perfezionamento del nostro prompt nel tempo è noto come ingegneria del prompt. Non scriverai mai il messaggio perfetto al primo tentativo, quindi è importante diventare bravo a perfezionarlo. Essere bravi nel prompt engineering deriva principalmente da molta pratica (prove ed errori). Il resto degli articoli di questa sezione presenteranno le diverse strategie di prompt che è possibile utilizzare nel processo di prompt engineering. È un sistema imperfetto e più il messaggio è complesso, meno è probabile che tu ottenga ciò che desideri [Pak].

Ma perchè alcune volte il Chatbot risponde in maniera errata?

La risposta errata fornita dai Chatbot può derivare da diversi fattori influenti, tra cui l'ambiguità e la specificità dei prompt, nonché le conoscenze presupposte. L'efficacia dei prompt risiede nella loro precisione e orientamento all'azione, che generano risultati più accurati e mirati. Tuttavia, quando si tratta di problemi matematici, i Chatbot possono mostrare delle limitazioni, poiché la risoluzione di tali questioni rappresenta un noto punto debole per i Large Language Models (LLM).

Sebbene i Chatbot come ChatGPT possano apparire capaci di affrontare una vasta gamma di problematiche, spesso falliscono nell'affrontare problemi matematici più o meno complessi. Questa incapacità di risolvere equazioni matematiche può risultare paradossale, considerando il successo delle calcolatrici e dei computer nel risolvere tali problemi da decenni.

Questo paradosso offre un'opportunità per riflettere sul cambiamento fondamentale nel modo in cui la tecnologia ha evoluto le nostre interazioni con i computer. Ad esempio, comunemente ci riferiamo agli individui come "persone matematiche" o "creative", attribuendo loro aspettative specifiche in base a tali designazioni [Pak].

La tecnologia come ChatGPT rappresenta un'anteprima di come potrebbe essere se i computer assumessero il ruolo di "creatività". Tuttavia, comprendere le capacità matematiche di ChatGPT richiede un approccio diverso. Analogamente a chiedere assistenza matematica a una persona con inclinazioni creative, si potrebbero ottenere spiegazioni comprensibili, ma non necessariamente risposte esatte.

Se si utilizza ChatGPT per l'assistenza matematica, è consigliabile richiedere istruzioni su come risolvere un problema piuttosto che affidarsi esclusivamente alla risposta finale [Lea]. È importante notare che, sebbene ChatGPT possa fornire formule corrette, potrebbe non garantire sempre risposte accurate dopo il segno di uguale. Pertanto, è consigliabile utilizzare ChatGPT con cautela per l'assistenza matematica e verificare sempre le risposte finali con strumenti tradizionali come una calcolatrice.

3.2.5 Generazione dei nuovi dati

Nel processo di generazione dei nuovi dati mediante l'impiego dei tre Chatbot selezionati - Google Gemini, OpenAI ChatGPT e Microsoft Copilot - ho individuato significative disparità tra le diverse tecnologie.

```
Create 5 lists (200 English words each) separated by commas, of the most used English words in the fields of Education, Manufacturing, Public Administration, Healthcare, and Logistics. The list should not have repeated words and should be singular nouns, not synonyms or antonyms
```

Figura 3.7: Prompt ideato per i 3 diversi Chatbot.

Dopo un'iterazione di perfezionamento del prompt, ho identificato quello che si dimostrava più efficace nel mio caso e ho quindi deciso di utilizzare lo stesso identico Prompt per tutti e tre i Chatbot.

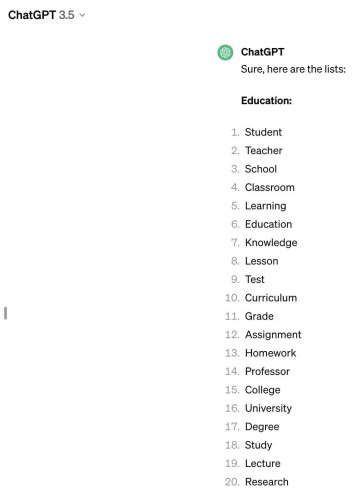

Figura 3.8: La risposta di ChatGPT al mio prompt

Successivamente avendo ottimizzato il prompt direttamente su ChatGPT ho avuto un buon riscontro, ha generato una buona lista di parole per ogni ambito richiesto, anche se ha optato per una lista numerata e non una lista divisa da virgole. Avendo ottenuto un feedback positivo, ho sottoposto lo stesso prompt agli altri due Chatbot.

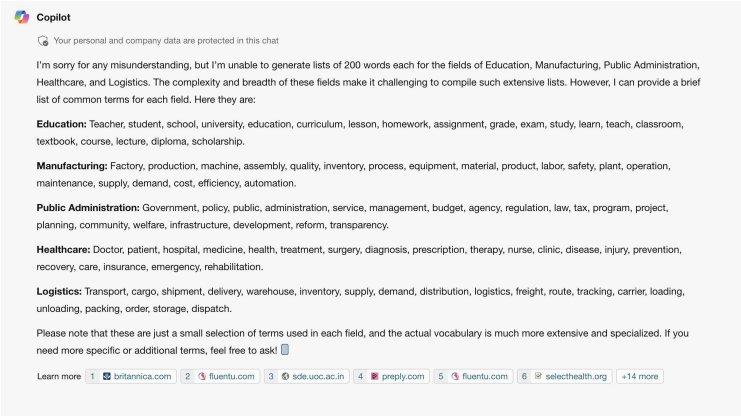

Figura 3.9: La risposta di Copilot al mio prompt

Tuttavia, l'esperienza con Copilot, la tecnologia sviluppata da Microsoft, ha presentato delle criticità evidenti. Come evidenziato, Copilot non è stato in grado di generare una lista di 200 parole per ogni ambito richiesto, costringendomi a adattare il prompt specificatamente per Copilot, in modo da ottenere una lista di parole chiave per ciascun ambito. Dopo diversi tentativi e iterazioni, sono riuscito a ottenere una lista, seppur di dimensioni variabili, per ciascun ambito.

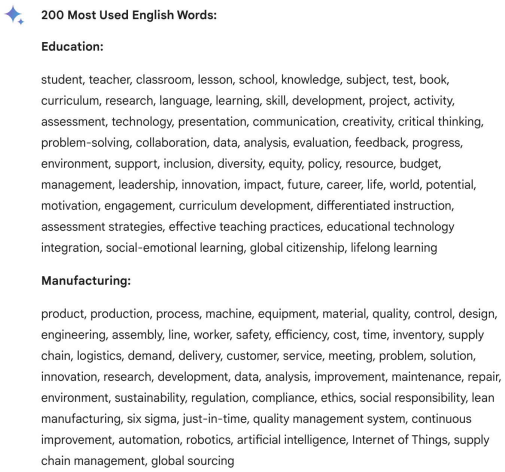

Figura 3.10: La risposta di Gemini al mio prompt

Invece, nonostante il prompt iniziale sia stato perfezionato su ChatGPT, l'esperienza con Gemini, la tecnologia di Google, si è rivelata altamente soddisfacente. Gemini ha generato una lista di parole uniche per ciascun ambito richiesto, confermando la sua efficacia nell'ambito della generazione di nuovi dati.

3.2.6 Preparazione dei dati

Durante questa fase, dopo aver precedentemente eseguito l'operazione di pulizia e preparazione dei dati per il modello precedente, ci siamo concentrati sulla creazione di diversi file .csv strutturati secondo lo stesso schema dei dati originali.

Questi file contengono un dataframe composto da due colonne: "CollectionName", che indica il dominio di appartenenza, e "Labels", che rappresenta l'insieme delle parole generate dai tre diversi Chatbot.

È importante notare che in questa fase sono stati introdotti nuovi domini di appartenenza per i dati di training, quali Manufacturing, Logistics, Public Administration, Healthcare e Education. Lo scopo di questa aggiunta è quello di studiare una classificazione addestrata su questi nuovi domini al fine di poterla successivamente applicare ai domini provenienti da BPMAI e Camunda, cercando così delle correlazioni tra di essi.

Al fine di analizzare in modo esaustivo le differenze tra i Chatbot, abbiamo esaminato i risultati sia utilizzando l'insieme combinato di risposte generate da ChatGPT, Gemini e Copilot, sia considerando i risultati ottenuti singolarmente.

4. Tecniche e Strumenti Utilizzati

Dopo aver delineato la metodologia e i principi teorici nel capitolo precedente, è fondamentale ora esaminare le tecniche e gli strumenti pratici adottati per implementare tali concetti nell'ambito dell'analisi dei dati.

In questo capitolo, ci concentreremo sulla selezione e sull'utilizzo dei metodi e degli strumenti che hanno permesso di tradurre la teoria in pratica. Questa selezione è stata guidata da una valutazione approfondita delle esigenze specifiche della ricerca e delle caratteristiche dei dati disponibili. Sono stati presi in considerazione i vantaggi e gli svantaggi di diverse opzioni, al fine di scegliere le risorse più adeguate per affrontare le sfide poste dalla ricerca.

Esamineremo quindi la progettazione e l'implementazione di un Dataset personalizzato, il cui ruolo è cruciale nell'alimentare il modello di classificazione. Successivamente, analizzeremo il funzionamento del DataLoader e il suo contributo durante le fasi di addestramento e validazione del modello.

Approfondiremo anche l'architettura del modello utilizzato per la classificazione, discutendo le modalità di addestramento, la validazione e la sua interpretazione dei risultati ottenuti.

Infine, esploreremo l'implementazione di un'interfaccia web per facilitare l'interazione con il modello e l'analisi dei suoi output.

Questo capitolo mira a fornire una panoramica completa delle scelte metodologiche adottate, illustrando come tali decisioni abbiano influenzato il processo di analisi e i risultati finali della ricerca.

4.1 L'idea su come utilizzare GPT-2 per la classificazione di un testo

GPT-2 (Generative Pre-trained Trasformer 2) rappresenta un notevole esempio di Large Langue Model (LLM) sviluppato da OpenAI. La sua capacità generale è quella di prevedere con precisione l'elemento successivo in una sequenza, ciò gli permette di tradurre testi, rispondere a domande su un argomento specifico, riassumere informazioni da testi più estesi e generare output di testo che, a volte, risulta indistinguibile da quello umano [Wikb].

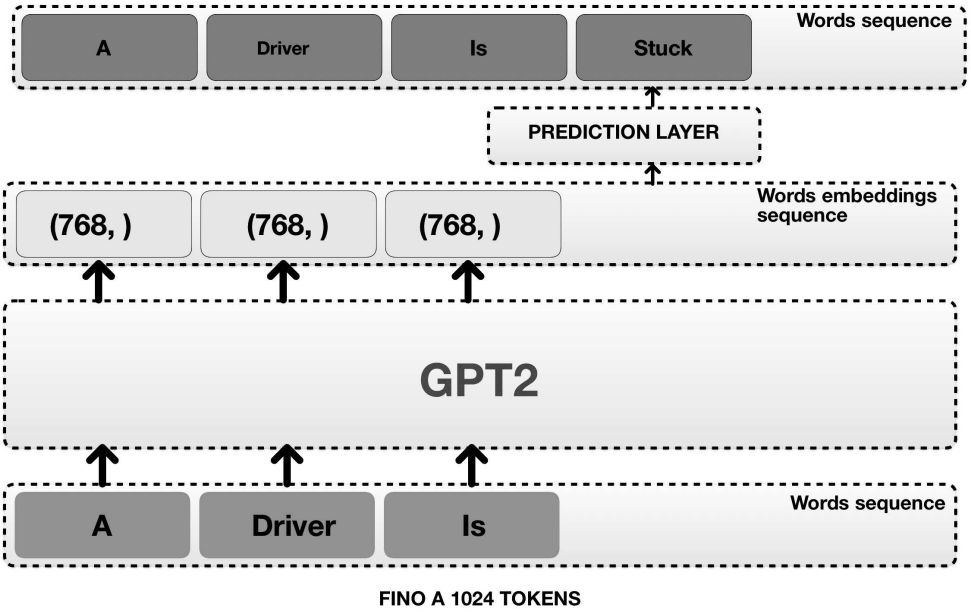

Figura 4.1: Processo di predizione dell'elemento successivo con GPT-2.

Vediamo dunque come utilizzarlo per fare predizioni

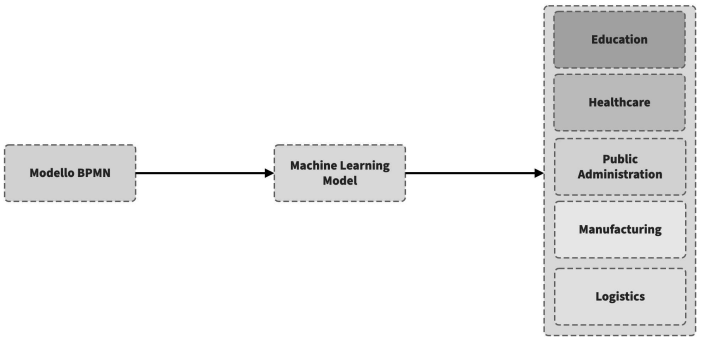

Figura 4.2: L'idea di GPT-2 per la classificazione.

Poiché GPT2 è un transformer decoder, un'architettura che sfrutta l'autoattenzione per catturare le relazioni a lungo termine all'interno di un testo [Hoq]. In particolare,

durante il processo di predizione, GPT-2 utilizza l'ultimo token della sequenza di input per generare previsioni sul token successivo che dovrebbe seguire l'input. Ciò significa che l'ultimo token della sequenza di input contiene tutte le informazioni necessarie nella previsione successiva.

Tenendo presente questo, possiamo utilizzare tali informazioni per fare una previsione in un'attività di classificazione anziché in un'attività di generazione. In questa modalità, anziche generare testo successivo, il modello utilizza le informazioni contenute nell'ultimo token della sequenza per effettuare previsioni in base a classi di appartenenza predefinite.

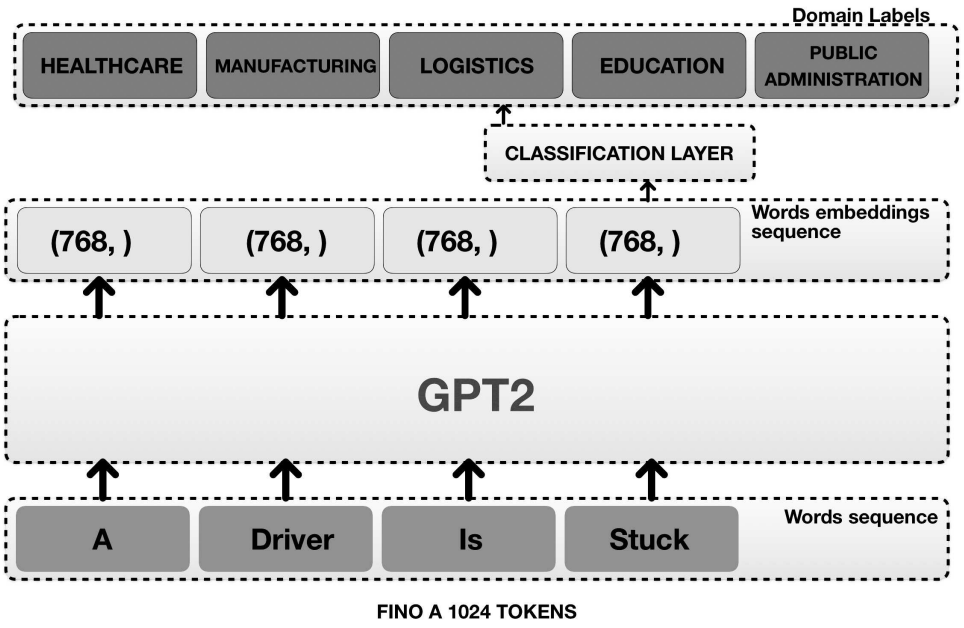

Figura 4.3: L'idea di GPT-2 per la classificazione in dettaglio.

Nel prosieguo di questo capitolo, approfondiremo il processo di utilizzo di GPT-2 per affrontare specificamente compiti di classificazione del testo, esaminando le metodologie e le pratiche ottimali per ottenere risultati efficaci e accurati [Wikb].

4.1.1 Come utilizziamo *GPT2ForSequenceClassification* per la classificazione

Come spiegato precedentemente l'idea principale è quella di usare l'ultimo token della sequenza in input, token che continene tutte le informazioni necessarie, per generare un'ultimo token che rappresenza la previsione su una classe di appartenenza.

Ecco una panoramica di come dovrebbe funzionare il processo:

1. **Addestramento del Modello**:
    - **Preparazione dei dati di addestramento**: Preparazione di una quantità di testi etichettati con le rispettive classi di classificazione. Assicurandoci che i dati siano bilanciati e rappresentativi della distribuzione reale.
    - **Preprocessing dei dati**: Pulizia dei dati, tokenizzazione e creazione di embedding, ecc..., in base alle esigenze del modello.

- **Configurazione del modello**: Configurazione dei parametri del modello GPT-2ForSequenceClassification, come ad esempio la dimensione dell'embedding, numero di layers, dimensione dei batch, numero di epoche, tasso di apprendimento ecc.
- **Addestramento del modello**: Utilizzo dei dati di addestramento per addestrare il modello. Durante l'addestramento, il modello impara a predirre la classe di classificazione corretta per una data sequenza di testo.

2. **Fine-Tuning**:

   - **Scelta di un modello Pre-Addestrato**: C'è un significativo vantaggio nell'uso di un modello pre-addestrato. Riduce il costo computazionale e ti permette di utilizzare modelli all'avanguardia senza doverne addestrare uno da zero, come GPT-2PreTrained come punto di partenza. Questo modello può essere addestrato su un ampio spettro di testi non etichettati e può avere una conoscenza linguistica generale.
   - **Fine-Tuning del Modello**: Adattamento e addestramento del modello pre-addestrato ai dati specifici del tuo studio, nel problema di classificazione di sequenze in input. Durante il fine-tuning, i pesi del modello vengono aggiornati, utilizzando dei dati etichettati specifici per migliorare le prestazioni del modello sul task di classificazione.

3. **Inferenza**:

   - **Preparazione dei dati di inferenza**: Preparazione del testo in input che desideriamo classificare utilizzando il modello fine-tuned. Assicurandoci che il testo sia formattato correttamente e che sia coerente con i dati utilizzati per il fine-tuning.
   - **Esecuzione dell'inferenza**: Utilizzo del modello fine-tuned per classificare il testo in input. Passiamo il modello e otteniamo le previsioni della classe di classificazione associata al testo.
   - **Interpretazione dei risultati**: Valutazione delle previsioni ottenute dal modello per determinare l'accuratezza e l'affidabilità della classificazione. Analizzare le previsioni errate per identiciare eventuali epttern o problemi nel modello.

4. **Valutazione delle Prestazioni**:

   - **Metriche di valutazione**: Utilizzo di metriche appropriate per la valutazione delle prestazioni del modello.
   - **Valutazione dei logits**: Utilizziamo i logits in output dalla rete neurale per studiare le prestazioni del modello ed identificare eventuali problematiche.
   - **Confronto con altri modelli**: Confronto delle prestazioni del modello fine-tuned con altri modello di classificazione sequenziale, sia modelli rule-based che modelli tradizionali.
   - **Analisi degli errori**: Analisi degli errori del modello per identiicare le aree in cui il modello può migliorare. Questo potrebbe includere l'analisi dei falsi positivi, falsi negativi e ambiguità nel testo di input.

Tuttavia dato che i due domini di etichettatura, per il training e per il validation, sono diversi e non esiste un'equivalenza diretta tra le etichette dei due domini, l'uso diretto di metriche come l'Accuratezza e l'F1-score potrebbe non essere appropriata.

Quando i due domini di etichettatura sono diversi, è necessario, dunque, adottare un'approccio creativo e flessibile alla valutazione delle prestazioni del modello, combinando valutazioni qualitative e quantitative per ottenere una comprensione completa delle sue capacità e delle sue limitazioni.

4.2 Progettazione del Dataset Personalizzato

Il Dataset personalizzato è stato progettato per adattarsi alle esigenze specifiche dello studio in corso. È stato creato un Dataset denominato *BPMNDomainDataset* e *BPMN-Dataset*, implementato attraverso la classe Dataset di PyTorch. Questo Dataset è stato strutturato per gestire dati tabulari, essendo essenziale per il progetto e i requisiti dei transformers HuggingFace.

La creazione del Dataset personalizzato coinvolge diverse fasi. Inizialmente, si accerta la validità del percorso fornito, verificando che sia un file valido. Successivamente, i dati vengono caricati da un file CSV utilizzando la libreria Pandas, con particolare attenzione all'encoding per garantire la corretta interpretazione dei caratteri.

```
class BPMNDomainDataset(Dataset):
    def __init__(self, path):
        if not os.path.isfile(path):
            raise ValueError('Invalid `path` variable! Needs to be a file')

        self.df = pd.read_csv(path, sep=';', engine='python', encoding=get_file_encoding(path))
        self.descriptions = self.df['Labels'].to_list()
        self.domains = self.df['CollectionName'].to_list()
        self.flattened_domains = [label for sublist in self.domains for label in sublist.split(',')]
        self.n_examples = len(self.descriptions)

    def __len__(self):
        return self.n_examples

    def __getitem__(self, item):
        return {"text": self.descriptions[item], "label": self.flattened_domains[item]}
```

Figura 4.4: Creazione del Dataset PyTorch.

All'interno del Dataset, ogni riga rappresenta un esempio e contiene una descrizione testuale e l'etichetta associata. Le descrizioni testuali sono estratte dalla colonna '*Labels*', mentre le etichette sono tratte dalla colonna '*CollectionName*'. Nel caso delle etichette, esse possono essere composte da più categorie separate da virgole, e sono state quindi scomposte in una lista per facilitare la gestione.

In aggiunta alla creazione del Dataset principale, è stato sviluppato un metodo per gestire gli esempi di validazione privi di etichettatura o per i quali non si desiderava utilizzare l'etichetta associata. Questo è stato necessario per garantire una gestione completa e accurata dei dati di validazione.

```
class BPMNDataset(Dataset):
    def __init__(self, path):
        if not os.path.isfile(path):
            raise ValueError('Invalid `path` variable! Needs to be a file')

        self.df = pd.read_csv(path, sep=';', engine='python', encoding=get_file_encoding(path))
        self.descriptions = self.df['Labels'].to_list()
        self.n_examples = len(self.descriptions)

    def __len__(self):
        return self.n_examples

    def __getitem__(self, item):
        return {'text': self.descriptions[item]}
```

Figura 4.5: Creazione del Dataset PyTorch per esempi senza etichette.

4.3 Come collaborano Dataset e Dataloader

Nel contesto dell'addestramento e della gestione dei modelli di machine learning, l'utilizzo di Dataset e dataloader gioca un ruolo fondamentale nella preparazione e nell'elaborazione dei dati, specialmente nell'ecosistema di PyTorch. In questa sottosezione, si esplorerà il ruolo e il funzionamento di Dataset e dataloader nel contesto del progetto in esame.

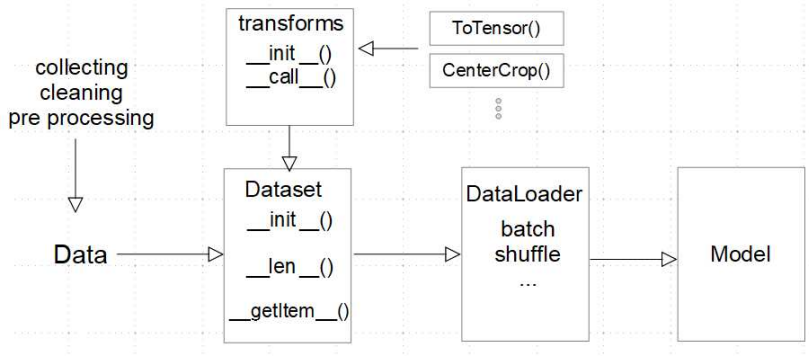

Figura 4.6: L'utilizzo dei Dataset e dataloader in un modello ML.

I Dataset rappresentano la fonte primaria di dati utilizzati per addestrare e valutare i modelli di machine learning. In PyTorch, la classe Dataset definisce un'interfaccia standard per i Dataset e fornisce metodi per accedere ai dati e alle etichette. Nel nostro caso, i Dataset personalizzati *BPMNDomainDataset* e *BPMNDataset* sono stati progettati per ospitare le descrizioni testuali e, eventualmente, le relative etichette associate, caricando i dati da file CSV, fornendo così i dati necessari per l'addestramento del modello GPT-2. Ogni esempio nel Dataset è rappresentato come una coppia testo-etichetta, che costituisce l'unità di base per l'elaborazione dei dati.

4.3.1 Dataloader per l'Addestramento

I dataloader fungono da interfaccia tra il Dataset e il modello di machine learning. Questi oggetti sono responsabili della gestione dell'accesso efficiente ai dati durante l'addestramento del modello, suddividendo il Dataset in mini-batch di dimensioni gestibili e fornendo tali mini-batch al modello per l'elaborazione. Nel nostro caso, il dataloader di addestramento si occupa di distribuire i dati di addestramento in mini-batch. Prima che i dati vengano passati al modello, essi sono pre-processati utilizzando il collator *Gpt2ClassificationCollatorDomain*, che si occupa di tokenizzare il testo e codificare le etichette.

4.3.2 Dataloader per la Validazione

Analogamente al dataloader per l'addestramento, viene utilizzato un dataloader specifico anche per la fase di validazione. In questa fase, i dati vengono utilizzati per valutare le performance del modello. Tuttavia, non è necessario utilizzare un collator per la codifica delle etichette in questa fase, poiché i dati di validazione non vengono utilizzati per l'addestramento e quindi non richiedono l'encoding delle etichette.

4.3.3 L'utilizzo del Collator per agevolare l'utilizzo di GPT-2

Le classi *Gpt2ClassificationCollatorDomain* e *Gpt2ClassificationCollator* sono state implementate per facilitare il processo di tokenizzazione e preparazione dei dati per l'addestramento del modello GPT-2. Questi collator si occupano di tokenizzare i testi utilizzando il tokenizer associato al modello, gestendo anche la codifica delle etichette in numeri interi utilizzando un encoder specifico. I testi tokenizzati vengono poi adeguatamente formattati e preparati per essere passati al modello.

Ci sono due parti principali nella classe collator:

- ***init()***: Dove inizializziamo il tokenizer che intendiamo usare, come codifichiamo le etichette e se dobbiamo impostare la lunghezza della sequenza in input ad un valore differente.
- ***call()***: Usata come una funzione collatore che prende in input un insieme di dati. Questa funziona ritorna un oggetto con il formato adatto per essere dato in pasto al nostro modello. Fortunatamente il tokenizer effettua ciò per noi e ritorna un dizionario di variabili pronte per essere date in pasto al nostro modello.

```python
class Gpt2ClassificationCollatorDomain(object):
    def __init__(self,
                 use_tokenizer,
                 labels_encoder,
                 max_sequence_len=None):
        self.use_tokenizer = use_tokenizer
        self.max_sequence_len = use_tokenizer.model_max_length if max_sequence_len is None else max_sequence_len
        self.labels_encoder = labels_encoder

    def __call__(self, sequences):
        texts = [sequence.get('text', None) for sequence in sequences]
        labels = [sequence.get('label', None) for sequence in sequences]
        label_ids = [self.labels_encoder[label] for label in labels]
        inputs = self.use_tokenizer(text=texts, return_tensors="pt", padding=True, truncation=True,  max_length=self.max_sequence_len)
        inputs['labels'] = torch.tensor(label_ids)
        return inputs
```

Figura 4.7: Collator e il suo utilizzo.

4.4 L'Architettura del modello per la classificazione

In questa sezione, esamineremo l'architettura del modello utilizzato per la classificazione basata su GPT-2. Saranno analizzati i dettagli relativi al processo di addestramento e validazione del modello, delineando le strategie adottate per massimizzarne le prestazioni e garantirne la robustezza.

4.4.1 L'addestramento

Nella fase di addestramento del modello per la classificazione, utilizziamo il Dataset personalizzato precedentemente creato. Questo Dataset è stato progettato per adattarsi alle specifiche esigenze della nostra ricerca e contiene una vasta gamma di esempi di testo associati alle rispettive etichette di classificazione. Utilizzando il DataLoader, carichiamo i dati in batch e iteriamo su di essi mentre il modello apprende le relazioni tra le sequenze di testo e le rispettive etichette di classificazione.

```python
def train(model, dataloader, optimizer, scheduler, device):
    predictions_labels = []
    true_labels = []
    total_loss = 0
    model.train()


    # Utilizzo tqdm per visualizzare una barra di avanzamento mentre itero sui batch
    for batch in tqdm(dataloader, total=len(dataloader)):
        true_labels += batch['labels'].numpy().flatten().tolist()
        batch = {k:v.type(torch.long).to(device) for k,v in batch.items()}
        optimizer.zero_grad()
        outputs = model(**batch)
        loss, logits = outputs[:2]
        total_loss += loss.item()
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), 1.0)
        optimizer.step()
        scheduler.step()
        logits = logits.detach().cpu().numpy()
        predictions_labels += logits.argmax(axis=-1).flatten().tolist()

    avg_epoch_loss = total_loss / len(dataloader)

    return true_labels, predictions_labels
```

Figura 4.8: Codice per il training.

Durante l'addestramento, il modello viene ottimizzato utilizzando l'algoritmo di discesa del gradiente stocastico (SGD) con un'implementazione personalizzata dell'ottimizzatore. Questo ci consente di regolare i pesi del modello in modo da minimizzare la discrepanza tra le previsioni del modello e i valori reali dell'obiettivo (ground truth). Ogni batch di dati viene propagato attraverso il modello e calcolata la perdita, che rappresenta la discrepanza tra le previsioni del modello e i valori reali dell'obiettivo. Questa perdita viene quindi utilizzata per calcolare i gradienti e aggiornare i pesi del modello in modo appropriato.

4.4.2 La validazione

La fase di validazione del modello è essenziale per valutare le prestazioni del modello su dati non visti durante l'addestramento. Utilizzando un secondo Dataset, denominato *valid_Dataset*, valutiamo le prestazioni del modello su nuovi esempi di testo. Come nella fase di addestramento, utilizziamo il DataLoader per caricare i dati in batch e iteriamo su di essi mentre il modello genera previsioni per ciascun esempio di testo.

```python
def validation(dataloader, device_, model):
    predictions_labels = []

    model.eval()

    for batch in tqdm(dataloader, total=len(dataloader)):
        batch = {k: v.to(device_) for k, v in batch.items()}
        with torch.no_grad():
            outputs = model(**batch)
            logits = outputs.logits.detach().cpu().numpy()
            predictions_labels.extend(logits.argmax(axis=-1).tolist())

    return predictions_labels
```

Figura 4.9: Codice per il validation.

Durante la fase di validazione, il modello viene impostato in modalità di valutazione (*model.eval()*), il che significa che i gradienti non vengono calcolati e i pesi del modello non vengono aggiornati. Invece, il modello genera previsioni per ciascun esempio di testo nel Dataset di validazione e queste previsioni vengono confrontate con le etichette di classificazione reali per valutare le prestazioni del modello. Questo ci consente di

determinare l'accuratezza del modello su dati non visti e di identificare eventuali aree in cui il modello potrebbe migliorare. listings xcolor

5. Risultati Sperimentali

Nel presente capitolo si propone di esporre e analizzare i risultati emersi dall'esecuzione degli esperimenti condotti nel contesto di questa ricerca. L'obiettivo principale è fornire un'interpretazione dettagliata delle osservazioni derivanti dall'analisi dei dati e delle metriche di valutazione utilizzate.

Durante l'esame dei risultati, sarà dedicata particolare attenzione alle diverse combinazioni sperimentali adottate. Questo approccio consente di visualizzare l'efficacia del modello da molteplici prospettive, consentendo di ottenere una panoramica completa delle sue capacità.

Mediante un'analisi approfondita delle varie combinazioni, sarà possibile trarre conclusioni significative e informate sulle prestazioni del modello e sulle eventuali aree di miglioramento.

5.1 Comparazione dei risultati con i dati iniziali

I risultati ottenuti attraverso diverse combinazioni di file di addestramento hanno fornito interessanti osservazioni sul comportamento del modello. In particolare, il modello è stato allenato utilizzando sia le parole derivanti dai tre Chatbot (ChatGPT, Gemini e CoPilot) contemporaneamente, sia separatamente, con l'obiettivo di analizzare l'impatto delle parole di ciascun Chatbot sulla classificazione delle etichette relative ai modelli BPMN.

Nel primo caso, in cui il modello è stato addestrato utilizzando parole derivanti da tutti e tre i Chatbot contemporaneamente, si è notata una predominanza nella classificazione come "logistics" sia per i modelli provenienti da Camunda che per quelli derivanti da BPMAI.

	BPMAI	Camunda
Logistics	4302	366
Public Administration	59	5
Manufacturing	24	0
Education	146	12
Healthcare	9	1

Tabella 5.1: Distribuzione delle etichette BPMAI e Camunda sulla combinazione di 3 Chatbot contemporaneamente.

Nel secondo caso, in cui il modello è stato addestrato utilizzando parole derivanti da ciascun Chatbot singolarmente, sono emerse diverse osservazioni:

- **ChatGPT:** Per quanto riguarda le parole derivate da ChatGPT, sia i modelli derivanti da Camunda che da BPMAI sono stati prevalentemente classificati come "logistics", mentre una minor parte è stata classificata come "education".

	BPMAI	Camunda
Logistics	3710	318
Public Administration	160	6
Manufacturing	15	3
Education	642	56
Healthcare	13	1

Tabella 5.2: Distribuzione delle etichette BPMAI e Camunda su parole derivanti da ChatGPT

- **Gemini:** Per le parole derivate da Gemini, sia i modelli derivanti da Camunda che da BPMAI sono stati principalmente classificati come "logistics", mentre una minor parte è stata classificata come "manufacturing".

	BPMAI	Camunda
Logistics	3856	372
Public Administration	84	5
Manufacturing	364	6
Education	128	0
Healthcare	108	1

Tabella 5.3: Distribuzione delle etichette BPMAI e Camunda su parole derivanti da Gemini

- **Copilot:** Per le parole derivate da CoPilot, sia i modelli derivanti da Camunda che da BPMAI sono stati principalmente classificati come "logistics", con una minor parte classificata come "manufacturing".

	BPMAI	Camunda
Logistics	4009	373
Public Administration	21	2
Manufacturing	357	6
Education	42	0
Healthcare	111	3

Tabella 5.4: Distribuzione delle etichette BPMAI e Camunda su parole derivanti da Copilot

Queste osservazioni indicano una certa coerenza nei risultati tra i diversi modelli derivanti dai vari Chatbot, con una tendenza predominante verso la classificazione come "logistics" nelle diverse condizioni sperimentali.

5.1.1 Intrepretazione dei risultati

Dai risultati presentati nelle tabelle, emerge chiaramente una predominanza nella classificazione delle etichette dei modelli BPMN come "logistics". Questa tendenza è osservata in diverse condizioni sperimentali, sia quando il modello è addestrato utilizzando le parole derivanti da tutti e tre i Chatbot contemporaneamente, sia quando è addestrato con parole derivanti da ciascun Chatbot singolarmente. Tuttavia, si notano variazioni nelle classificazioni secondarie tra i modelli derivanti dai diversi Chatbot, suggerendo che le parole specifiche a ciascun Chatbot possono influenzare le etichette dei modelli in modo differente. Nonostante queste variazioni, esiste una certa coerenza nei risultati tra i diversi modelli derivati dai vari Chatbot, indicando la presenza di tendenze comuni nei dati di addestramento o nei modelli stessi. Queste osservazioni evidenziano l'importanza di considerare l'impatto delle parole derivanti dai Chatbot sull'addestramento e sulle prestazioni dei modelli BPMN, e suggeriscono possibili direzioni per future indagini e analisi approfondite.

5.1.2 Studio dei Logits

Per ottenere una comprensione più approfondita delle previsioni del modello e valutarne la fiducia nelle varie classificazioni, è stato condotto un'analisi dei logits. I logits forniscono una misura della probabilità associata a ciascuna classe di etichette, consentendo di valutare la coerenza e l'affidabilità delle previsioni del modello. L'analisi dei logits offre quindi un'opportunità per esaminare in dettaglio le decisioni del modello e identificare eventuali ambiguità o incertezze nelle sue previsioni. Per facilitare questa analisi, è stato sviluppato un nuovo notebook che consente di fornire in input al modello un file contenente testo da analizzare e di ottenere in output i logits corrispondenti. Questo approccio consente di valutare singolarmente le previsioni del modello BPMN, consentendo una valutazione più approfondita delle prestazioni e delle tendenze del modello su diverse classificazioni.

5.2 Sviluppo di un'Interfaccia Web Interattiva per la Classificazione di Modelli BPMN

In questo paragrafo, descriveremo la creazione di un piccolo Front-End, questa integrazione rappresenta un elemento fondamentale per migliorare l'accessibilità e l'utilizzo pratico del sistema di classificazione dei modelli BPMN. Un Front-End ben progettato consente agli utenti finali di interagire in modo intuitivo con il sistema, facilitando la navigazione tra le funzionalità offerte e semplificando il processo di inserimento e visualizzazione dei dati. Questo non solo aumenta l'efficienza nell'utilizzo del sistema, ma anche riduce la necessità di competenze tecniche specializzate da parte degli utenti.

È trovare il codice sorgente del Front-End su GitHub al seguente link: GitHub.

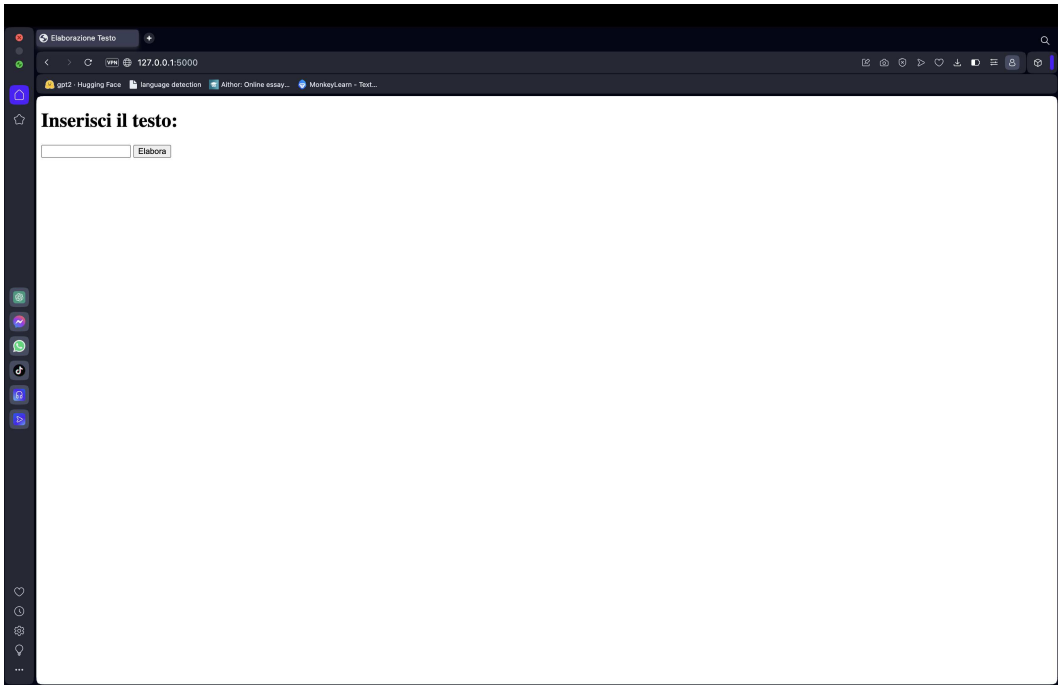

Figura 5.1: Pagina iniziale Front-End.

5.2.1 Metodologia e Strumenti per lo Sviluppo dell'Interfaccia Web con Flask

Per creare l'Interfaccia Web, abbiamo utilizzato Flask insieme ad un file HTML per la struttura della pagina web. L'Interfaccia Web è composto principalmente da due file: il file Python che contiene la logica dell'applicazione Flask e il file HTML che definisce la struttura della pagina web.

Nel file Python, abbiamo definito una funzione index che gestisce la richiesta HTTP GET per la homepage dell'applicazione. Quando un utente accede alla homepage, viene visualizzata una pagina HTML con un modulo per l'inserimento del testo e un pulsante per l'invio del modulo. Quando l'utente invia il modulo, il testo inserito viene elaborato utilizzando una funzione specifica che utilizza i logits.

I logits sono le uscite non normalizzate di un modello di classificazione. In questo caso, abbiamo utilizzato un modello di classificazione GPT-2 per elaborare il testo inserito dagli utenti. I logits rappresentano le "probabilità" che il modello assegna a ciascuna classe di output. Utilizzando i logits, siamo in grado di ottenere una previsione delle etichette per il testo inserito.

Figura 5.2: Stampa dei Logits.

Per gestire le richieste HTTP e generare dinamicamente le pagine web, abbiamo utilizzato il modulo Flask di Python. Flask semplifica notevolmente la gestione delle richieste HTTP e l'integrazione di Python con HTML per la creazione di frontend dinamici.

Complessivamente, il frontend creato con Flask offre un'interfaccia utente semplice e intuitiva per l'elaborazione del testo. Utilizzando Flask insieme a HTML e CSS, siamo riusciti a creare un Front-End funzionale per l'applicazione web.

5.2.2 Realizzazione di un eseguibile a partire da un notebook Jupyter

Per trasformare il codice sviluppato in un notebook Jupyter in un'applicazione Python eseguibile, abbiamo adottato un processo che sfrutta l'utilità `nbconvert` fornita da Jupyter Notebook.

```
1 jupyter nbconvert --to script 2.1.1 GPT_Study_AIs_percentages-FRONT_END.
    ipynb
```

Codice 5.1: Comando python per nbconvert.

Siamo stati in grado di convertire il notebook Jupyter in un file di script Python. Questo processo ha trasformato tutto il codice presente nel notebook, inclusi sia il codice delle celle di codice che il testo delle celle di markdown, in un formato di script Python convenzionale.

Successivamente, per eseguire lo script Python appena creato come un'applicazione Python indipendente, abbiamo utilizzato il comando:

```
python 2.1.1 GPT_Study_AIs_percentages-FRONT_END.py
```

Codice 5.2: Comando per avviare l'eseguibile.

Questo comando ha avviato l'interprete Python e caricato il file di script Python specificato, eseguendo tutte le istruzioni presenti nel file. In tal modo, siamo stati in grado di trasformare il codice di un notebook Jupyter in un formato più convenzionale per la distribuzione e l'esecuzione, consentendo una maggiore facilità di utilizzo e di condivisione del codice sviluppato.

6. Discussione Finale

L'analisi condotta sui modelli BPMN mediante l'utilizzo di parole derivanti da diversi Chatbot ha rivelato risultati interessanti. L'approccio unico di questo studio ha permesso di esaminare da vicino l'impatto delle caratteristiche linguistiche specifiche di ogni Chatbot sull'addestramento e sulle prestazioni delle etichette provenienti dai modelli BPMN.

Uno dei punti più rilevanti di questo studio è la scoperta di una tendenza costante alla classificazione dei modelli BPMN come "logistics", indipendentemente dal Chatbot utilizzato per l'addestramento. Questo risultato sottolinea l'importanza delle parole derivanti dai Chatbot nella formazione dei modelli.

Tuttavia, ciò che rende unico questo studio è la sua capacità di analizzare non solo le classificazioni principali, ma anche le variazioni nelle classificazioni secondarie tra i modelli derivanti dai diversi chatbot. Questo approccio ha evidenziato la complessità delle dinamiche linguistiche e ha messo in luce l'importanza di considerare il contesto specifico di addestramento e le caratteristiche del testo utilizzato.

Inoltre, l'analisi dei logits ha offerto una prospettiva più dettagliata delle previsioni del modello, consentendo di valutare la coerenza e l'affidabilità delle sue classificazioni. Questo approfondimento ha sottolineato l'importanza di sviluppare metodologie più sofisticate per l'analisi e la classificazione dei modelli BPMN, al fine di cogliere le sottili sfumature.

In conclusione, questo studio ha dimostrato l'importanza di considerare attentamente il contesto di addestramento e le caratteristiche linguistiche specifiche dei chatbot nell'analisi e nell'interpretazione dei modelli BPMN. Le conclusioni emerse offrono spunti interessanti per future ricerche nell'ambito dell'intelligenza artificiale, suggerendo l'importanza di un approccio olistico che tenga conto delle complessità linguistiche e delle dinamiche interattive tra i modelli e i dati di addestramento.

6.1 Possibili sviluppi futuri

Basandoci sui risultati e sulle osservazioni emerse da questa analisi, ci sono diverse direzioni che potrebbero risultare interessanti e che potrebbero essere esplorate in futuri sviluppi di questa ricerca sull'intelligenza artificiale applicata ai processi aziendali e alla classificazione dei modelli BPMN.

Alcuni possibili sviluppi futuri includono:

- **Integrazione di un Front-end User-Friendly:** Uno sviluppo promettente sarebbe l'integrazione di un'interfaccia utente intuitiva e user-friendly per l'interazione con il sistema di classificazione dei modelli BPMN. Un front-end ben progettato potrebbe semplificare l'uso del sistema da parte degli utenti finali e consentire una migliore integrazione con gli ambienti aziendali esistenti.
- **Utilizzo di Modelli di Classificazione Diversi:** Esplorare l'utilizzo di modelli di classificazione diversi potrebbe portare a una migliore comprensione delle variazioni nei risultati e delle prestazioni del sistema. Ad esempio, l'implementazione di modelli basati su reti neurali ricorrenti o di modelli di trasformatori potrebbe consentire di catturare relazioni più complesse tra le parole e le etichette di classificazione.
- **Studio Approfondito degli Elementi BPMN:** Condurre uno studio approfondito sugli elementi specifici dei modelli BPMN potrebbe fornire una maggiore comprensione delle dinamiche interne dei processi aziendali. Questo potrebbe includere l'analisi delle relazioni tra gli elementi utilizzati nei processi e la loro correlazione con le etichette di classificazione.
- **Valutazione dell'Impatto Pratico:** Valutare l'efficacia e l'efficienza dell'utilizzo dei modelli di classificazione dei processi BPMN in contesti aziendali reali potrebbe fornire indicazioni preziose sulle loro potenzialità e limitazioni. Questo potrebbe includere studi di caso presso aziende attive in settori diversi per valutare l'applicabilità pratica dei modelli.

In conclusione, l'avanzamento della ricerca sull'intelligenza artificiale applicata ai processi aziendali e alla classificazione dei modelli BPMN richiederà un approccio diversificato e innovativo. Esplorare queste direzioni di ricerca potrebbe portare a progressi significativi nell'ambito della comprensione e dell'applicazione pratica dei modelli BPMN, con importanti implicazioni per l'ottimizzazione e l'automazione dei processi aziendali.

Bibliografia

[Ber] Berkeley. *What Is Machine Learning (ML)?* URL: https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/.

[Com+24] Ivan Compagnucci et al. «A Study on the Usage of the BPMN Notation for Designing Process Collaboration, Choreography, and Conversation Models». In: *Business & Information Systems Engineering* 66 (2024), 43–66.

[Del] Julianna Delua. *Supervised vs. Unsupervised Learning: What's the Difference?* URL: https://www.ibm.com/blog/supervised-vs-unsupervised-learning/.

[Flo16] Luciano Floridi. *Il Futuro Prossimo dell'Intelligenza Artificiale*. Youtube. 2016. URL: https://www.youtube.com/watch?v=0E1kNRpNW10.

[Har] Larry Hardesty. *Explained: Neural networks*. URL: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414.

[Hoq] Minhajul Hoque. *A Comprehensive Overview of Transformer-Based Models: Encoders, Decoders, and More*. URL: https://medium.com/@minh.hoque/a-comprehensive-overview-of-transformer-based-models-encoders-decoders-and-more-e9bc0644a4e5.

[JM21] Fleur Ponton Juliano Morimoto. «Attention is all you need». In: (2021).

[Kam] Stylianos Kampakis. *What Deep Learning Is And Isn't*. URL: https://thedatascientist.com/what-deep-learning-is-and-isnt/.

[Lea] LearnPrompting. *Chatbos*. URL: https://learnprompting.org/docs/basics/chatbot_basics.

[Lot] Mehdi Lotfinejad. *6 Most Common Deep Learning Applications*. URL: https://www.dataquest.io/blog/6-most-common-deep-learning-applications/.

[Mon] MonkeyLearn. *What is Text Classification?* URL: https://monkeylearn.com/what-is-text-classification/.

[Ora] Oracle. *Che cos'è il Deep Learning?* URL: https://www.oracle.com/it/artificial-intelligence/machine-learning/what-is-deep-learning/.

[Pak] Enoch Pakanati. *WHY YOUR CHATGPT PROMPTS AREN'T WORKING?* URL: https://www.semrush.com/blog/chatgpt-prompts/.

[Sas] Sas. *History of Neural Network*. URL: https://www.sas.com/sk_sk/insights/analytics/neural-networks.html#industries.

[Ste] Jakob Steinschaden. *Google Gemini vs. ChatGPT Plus vs. Microsoft Copilot*. URL: https://www.trendingtopics.eu/google-gemini-vs-chatgpt-plus-vs-microsoft-copilot-2/.

[Tak] Akash Takyar. *AI USE CASES APPLICATIONS ACROSS MAJOR INDUSTRIES*. URL: https://www.leewayhertz.com/ai-use-cases-and-applications/.

[Tea] Semrush Team. *195 ChatGPT Prompts (How to Write Your Own)*. URL: https://www.semrush.com/blog/chatgpt-prompts/.

[TW22] Ruiqi Jin Tianqi Wu. ≪The Limitations of Deep Learning in Achieving Real Artificial Intelligence≫. In: (2022).

[Vas+17] Ashish Vaswani et al. ≪Attention is all you need≫. In: (2017).

[Wika] Wikipedia. *Chatbot*. URL: https://en.wikipedia.org/wiki/Chatbot.

[Wikb] Wikipedia. *GPT-2*. URL: https://en.wikipedia.org/wiki/GPT-2.

[Wikc] Wikipedia. *Rete neurale convoluzionale*. URL: https://it.wikipedia.org/wiki/Rete_neurale_convoluzionale.

[Woo] Carroll Woodard. *ChatGPT vs Google Gemini vs Microsoft Copilot: What are the differences?* URL: https://www.linkedin.com/pulse/chatgpt-vs-google-gemini-microsoft-copilot-what-carroll-woodard-pduxc?trk=article-ssr-frontend-pulse_more-articles_related-content-card.

Ringraziamenti

Desidero esprimere la mia più profonda gratitudine a tutte le persone che hanno contribuito al completamento di questa tesi di laurea, ma sopratutto di questo tragitto che mi ha portato fino a quì. Senza il loro sostegno, probabilmente, non avrei mai portato a termine questo percorso accademico.

In primo luogo, desidero ringraziare il mio relatore e il mio correlatore, il Prof. Marco Piangerelli e il Dott. Ivan Compagnucci, per l'aiuto, il supporto e la pazienza nel corso di questo progetto.

Vorrei però fare un ringraziamento speciale per i miei familiari, che nonostante tutto, mi hanno dato questa grande opportunità, non del tutto scontata e ci terrei a ringraziare singolarmente ognuno di loro: I miei genitori, mamma Manuela e babbo Giovanni per essere stati sempre comprensivi e non avermi mai messo pressione anche nei momenti di difficoltà. Le mie due sorelle, Vanessa e Debora, per aver sopportato un fratello non facilmente gestibile e molto diverso da loro. Ed infine, i miei nonni, Giuliana, Adamo ed Elvira che son fortunato di avere ancora quì con me a sostenermi. Vorrei ringraziarli tutti per il loro amore incondizionato, il loro costante sostegno e il loro incoraggiamento nel corso degli anni. I loro sacrifici hanno reso possibile per me raggiungere questo importante obbiettivo.

Vorrei poi anche ringraziare i miei amici e i miei colleghi, per il loro sostegno morale e per i momenti di allegria condivisi insieme durante il percorso accademico. Le nostre esperienze e lo scambio di idee hanno arricchito significativamente questa esperienza sia sul piano sociale che su quello accademico e mi avete aiutato a superare ogni anno di questo percorso.

Desidero esprimere la mia gratitudine ai partecipanti della mia ricerca, nonchè i miei colleghi che spesso trovavo in aula studio, che hanno gentilmente dedicato il loro tempo e le loro esperienze per aiutarmi. Senza il loro contributo, questa tesi non sarebbe stata possibile.

Ancora una volta, però, vorrei ringraziare singolarmente tutte le persone che hanno giocato un ruolo importante in questo mio percorso, che sia stato un ruolo di arricchimento personale o di puro svago. Dal mio coinquilino Shine che ha alleggerito molte delle mie giornate pesanti e noiose con il suo particolare umorismo e che poi mi ha introdotto a quelli che sono attualmente i miei amici di Camerino: Paolo, Yuri, Ivan, Genny, Erica ed Hillary, che hanno supportato e sopportato tante delle mie giornate. Vorrei poi ringraziare tanti dei miei amici a distanza che mi hanno sempre sostenuto con belle parole ed infine i miei amici di casa che spesso ho trascurato ma a cui tengo particolarmente e che nonostante la mia assenza ci sono sempre.

Vorrei anche ringraziare tutti i miei colleghi con cui ho avuto l'opportunità di condividere progetti universitari, che indubbiamente, hanno reso possibile la mia presenza quì grazie alla collaborazione per il superamento degli obbiettivi accademici.

Avete tutti giocato un ruolo fondamentale in questo percorso accademico e vi sono profondamente grato. Questo traguardo non sarebbe raggiunto senza il sostegno di ognuno dei sopra citati e il loro incoraggiamento.

Infine, vorrei anche ringraziare UniCam per fornire le risorse necessarie per portare avanti questi progetti e per l'opportunità di crescita accademica e personale che mi ha offerto.

Grazie di cuore, Anthony Eleuteri.