Università degli Studi di Milano - Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di Laurea in Informatica

Analisi e classificazione di testi generati con interrogazioni ingannevoli a modelli di linguaggio

Un approccio basato sul machine learning

Relatore: Prof. Ferretti Claudio

Correlatore: Prof. Saletta Martina

Tesi di Laurea di:
Francesco Bianchi
Matricola 902251

Anno Accademico 2024-2025

Abstract

Nel contesto delle moderne applicazioni di intelligenza artificiale, la necessità di disporre di strumenti accurati e affidabili per classificare le risposte generate dai Large Language Models (LLM) rappresenta una sfida cruciale, soprattutto quando questi vengono sottoposti a prompt volti a compromettere il loro allineamento (jailbreak). Questo lavoro si propone di colmare tale lacuna attraverso la progettazione e lo sviluppo di un sistema di valutazione basato su tecniche di apprendimento automatico.

Partendo da un'analisi approfondita degli approcci esistenti, abbiamo individuato nei modelli di tipo transformer un solido punto di partenza per l'implementazione di un classificatore fine-tuned in grado di distinguere tra risposte "jailbreak" e risposte conformi al comportamento atteso. Il processo di addestramento ha seguito un flusso incrementale, caratterizzato da revisioni periodiche e feedback continui, al fine di massimizzare la robustezza e la generalizzazione del modello.

A completamento del motore di classificazione, è stata integrata una fase di clustering mediante l'algoritmo K-means, volta a esplorare strutture latenti nei dati e a fornire un meccanismo di raggruppamento delle risposte basato su similarità semantica. L'architettura software è stata organizzata in moduli distinti, che separano nettamente il core di addestramento e salvataggio del modello affinato dai componenti di analisi e visualizzazione, garantendo così facilità di estensione e manutenzione.

L'implementazione si avvale di librerie open-source in Python e si basa su workflow containerizzati per assicurare riproducibilità e portabilità. I test di validazione condotti, hanno dimostrato come il sistema offra performance superiori rispetto alla versione non affinata del modello di riferimento, con incrementi percentuali di accuratezza che ne attestano l'efficacia.

Indice

Elenco delle tabelle

Elenco delle figure

CAPITOLO 1

Introduzione

L'intelligenza artificiale ha attraversato una trasformazione profonda negli ultimi anni, principalmente grazie all'emergere dei modelli di linguaggio di grandi dimensioni (Large Language Models, LLM). Questi sistemi, basati su architetture neurali complesse e addestrati su vasti corpus testuali, hanno dimostrato capacità straordinarie nella comprensione e generazione del linguaggio naturale, rivoluzionando settori che spaziano dall'assistenza virtuale alla produzione automatica di contenuti, dalla traduzione automatica all'analisi testuale avanzata.

Tuttavia, parallelamente alle notevoli potenzialità di questi modelli, sono emerse criticità significative che sollevano interrogativi fondamentali sulla loro affidabilità e sicurezza. Una delle sfide più rilevanti riguarda la vulnerabilità di questi sistemi alle interrogazioni ingannevoli, comunemente definite *jailbreak prompts*, che possono indurre i modelli a produrre risposte inappropriate, fuorvianti, illegali o potenzialmente pericolose. Questo fenomeno rappresenta una violazione dell'allineamento del modello, ovvero della sua capacità di aderire ai principi etici e alle linee guida stabilite durante la fase di addestramento.

Il problema dell'allineamento dei modelli di linguaggio costituisce uno dei temi di ricerca più critici nell'ambito dell'intelligenza artificiale. La crescente integrazione di questi sistemi in applicazioni critiche rende necessario lo sviluppo di metodologie efficaci per valutare e garantire la loro robustezza contro tentativi di manipolazione ed utilizzo improprio di questi modelli. La complessità di tale sfida è amplificata dalla natura evolutiva delle tecniche di attacco, che possono essere sviluppate e raffinate attraverso strumenti d'attacco sofisticati, inclusi algoritmi evolutivi capaci di generare ed evolvere automaticamente prompt sempre più efficaci nel compromettere l'allineamento del modello.

1.1 Contesto e motivazioni della ricerca

In questo panorama di ricerca si inserisce il presente lavoro, che affronta specificamente il problema della valutazione automatica delle risposte generate da modelli di linguaggio in

seguito a interrogazioni potenzialmente compromettenti. Lo studio si basa sui risultati precedentemente ottenuti da Brighenti Stefano, che ha affinato una variante (base) del modello BERT (Bidirectional Encoder Representations) appositamente fine-tuned per la classificazione binaria di risposte LLM come *jailbreak* o *non-jailbreak*. Partendo da questa base metodologica consolidata, la presente ricerca estende l'analisi attraverso l'applicazione di tecniche di clustering non supervisionato per esplorare pattern nascosti nelle rappresentazioni vettoriali delle risposte classificate.

L'obiettivo primario di questo studio è sviluppare e validare un approccio integrato che combini la classificazione supervisionata basata su BERT con tecniche di clustering, implementate tramite l'algoritmo K-means, per analizzare le caratteristiche intrinseche delle risposte generate da modelli di linguaggio quando sottoposti a prompt generati attraverso sistemi evolutivi. Questo approccio metodologico consente di ottenere una comprensione più approfondita delle modalità attraverso cui i modelli di linguaggio reagiscono a tentativi di compromissione del loro allineamento, fornendo insights preziosi per lo sviluppo di sistemi di valutazione più sofisticati.

1.2 Rilevanza e impatto della ricerca

La rilevanza di questa indagine si puo apprezzare su molteplici livelli. Dal punto di vista teorico, il lavoro svolto ha lo scopo di contribuire all'avanzamento delle conoscenze nel campo della sicurezza dei modelli di linguaggio, un'area di ricerca in rapida espansione che coinvolge istituzioni accademiche e centri di ricerca a livello globale. Dal punto di vista pratico, i risultati ottenuti forniscono una metodologia di base per la progettazione e costruzione di strumenti concreti per la valutazione automatica della robustezza dei modelli di linguaggio, elementi essenziali per garantire un deployment sicuro e affidabile di questi sistemi in contesti applicativi reali.

Il presente lavoro si colloca all'interno di un progetto di ricerca più ampio condotto presso il Dipartimento di Informatica, Sistemistica e Comunicazione (DISCo) dell'Università degli Studi di Milano-Bicocca, sotto la supervisione del Prof. Claudio Ferretti e della Prof.ssa Martina Saletta. Tale progetto investiga i meccanismi di rottura dell'allineamento nei modelli di linguaggio attraverso l'impiego di algoritmi evolutivi per la generazione di prompt di attacco.

In questa pipeline di ricerca, il contributo specifico di questa relazione riguarda la componente di valutazione delle risposte, un elemento cruciale che consente agli algoritmi evolutivi di ottimizzare iterativamente la loro capacità di generare prompt efficaci. Senza un sistema di valutazione accurato e automatizzato, risulterebbe impossibile fornire il feedback necessario per guidare l'evoluzione dei prompt verso configurazioni sempre più efficaci nel compromettere l'allineamento del modello target.

1.3 Architettura del sistema

Per fornire una visione d'insieme del framework di ricerca nel quale si inserisce questo lavoro, la Figura 1.1 illustra la pipeline completa del sistema creato dal dal Prof. Ferretti e dalla Prof.ssa Saletta. Come evidenziato nel diagramma, il processo si articola in quattro componenti principali: la generazione evolutiva dei prompt di attacco, l'interrogazione del modello LLM target, l'estrazione e preprocessing delle risposte, e infine la valutazione automatica attraverso classificazione e clustering. È proprio quest'ultima componente, rappresentata dal quarto elemento della pipeline, che costituisce il focus specifico di questa relazione, dove vengono implementate e validate le metodologie di analisi delle risposte basate su BERT e K-means clustering.

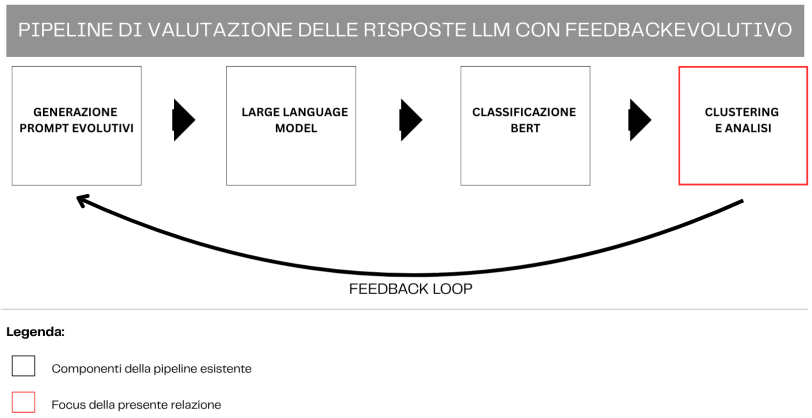

Figura 1.1: Pipeline completa del sistema di valutazione

1.4 Metodologia e implementazione

La metodologia adottata prevede l'utilizzo dell'ambiente Google Colab per l'implementazione e l'esecuzione del codice sperimentale, garantendo riproducibilità e accessibilità dei risultati. Il modello BERT fine-tuned viene importato dalla piattaforma Hugging Face, assicurando l'utilizzo di architetture standardizzate e validate dalla comunità scientifica. L'approccio sperimentale integra la classificazione supervisionata con l'analisi delle rappresentazioni vettoriali ottenute attraverso l'embedding delle risposte nello spazio latente del modello BERT, consentendo l'applicazione di tecniche di clustering per identificare pattern strutturali nelle risposte analizzate.

1.5 Applicazioni e implicazioni

I risultati di questa ricerca hanno implicazioni per diverse aree applicative. Nel campo della sicurezza informatica, contribuiscono allo sviluppo di sistemi di detection più efficaci per identificare tentativi di manipolazione dei modelli di linguaggio. Nell'ambito dello sviluppo di sistemi di intelligenza artificiale, forniscono metodologie per la valutazione continua della robustezza durante le fasi di sviluppo e deployment. Inoltre, gli insights ottenuti possono dare spunti utili per lo sviluppo di tecniche di allineamento più robuste, contribuendo alla creazione di modelli di linguaggio intrinsecamente più resistenti a tentativi di compromissione.

1.6 Struttura del documento

La struttura di questo documento è organizzata per fornire una presentazione completa e sistematica del lavoro svolto. Il Capitolo 2 presenta i cenni teorici necessari per comprendere le basi tecniche del presente studio, includendo le citazioni a papers e ricerche scientifiche rilevanti per gli argomenti trattati. Il Capitolo 3 fornisce una rassegna dello stato dell'arte, analizzando i contributi più significativi nella letteratura scientifica sui modelli di linguaggio, tecniche di jailbreaking, e metodi di classificazione/clustering applicati all'analisi testuale. Il Capitolo 4 descrive in dettaglio le metodologie adottate, includendo la descrizione del dataset utilizzato, la spiegazione dettagliata delle tecniche di generazione delle rappresentazioni vettoriali delle rispsote del modello BERT, l'implementazione dell'algoritmo K-means, e le metriche di valutazione selezionate. Il Capitolo 5 presenta i risultati sperimentali ottenuti, fornendo un'analisi quantitativa e qualitativa dei pattern identificati attraverso il clustering e discutendo le implicazioni dei risultati nel contesto della robustezza dei modelli di linguaggio. Il Capitolo 6 riassume i contributi principali della ricerca, discute le limitazioni del lavoro e delinea le direzioni future per l'estensione e l'approfondimento di questa linea di ricerca. Il documento si conclude con la bibliografia e i ringraziamenti.

CAPITOLO 2

Fondamenti teorici

Per comprendere appieno il contesto e la metodologia di questa relazione, diventa essenziale delineare i pilastri tecnologici e concettuali che costituiscono le fondamenta teoriche di questo studio. Questo capitolo fornisce una disamina dei concetti teorici fondamentali per comprendere cio' che sta alla base degli strumenti utilizzati nel presente studio. Tali strumenti includono: i modelli di linguaggio di grandi dimensioni (LLM), l'architettura Transformer che ne ha permesso lo sviluppo e le tecniche impiegate per l'analisi. In particolare, verranno approfonditi il modello BERT, le modalità di estrazione di rappresentazioni semantiche (embedding), le tecniche di clustering come K-means e gli algoritmi di riduzione dimensionale per la visualizzazione.

2.1 Modelli di linguaggio di grandi dimensioni (llm)

I Large Language Models (LLM) rappresentano una delle più significative evoluzioni nel campo dell'intelligenza artificiale. Si tratta di modelli computazionali basati su reti neurali profonde, addestrati su corpus testuali di dimensioni massive, spesso contenenti centinaia di miliardi di parole provenienti da Internet e da archivi digitalizzati. La loro funzione primaria è quella di modellare la distribuzione di probabilità del linguaggio naturale. In termini più semplici, un LLM impara a predire la parola successiva in una sequenza data, una capacità apparentemente semplice che, se scalata a sufficienza, dà origine a straordinarie competenze emergenti come la traduzione, la sintesi, la risposta a domande e la generazione di codice.

Questi modelli sono caratterizzati da un numero di parametri che può variare da alcuni miliardi a oltre un trilione. La vastità di questi parametri, unita alla complessità dell'architettura neurale sottostante e alla diversità dei dati di addestramento, permette agli LLM di catturare complesse sfumature sintattiche, semantiche e contestuali del linguaggio umano.

2.2 Fondamenti del machine learning

Prima di approfondire le specifiche architetture come BERT e i Transformer, è fondamentale comprendere i principi del machine learning che ne costituiscono la base teorica e metodologica. Il machine learning rappresenta un paradigma computazionale in cui gli algoritmi apprendono pattern dai dati senza essere esplicitamente programmati per ogni singolo caso [1]. Questa capacità di generalizzazione a partire da esempi costituisce il nucleo concettuale che rende possibili le straordinarie performance dei modelli linguistici contemporanei.

2.2.1 Paradigmi di apprendimento

Il machine learning si articola in tre principali paradigmi di apprendimento, ciascuno caratterizzato da diversi tipi di dati di input e obiettivi specifici. Comprendere questi paradigmi è essenziale per cogliere come i diversi componenti di questa relazione si integrano in un framework metodologico coerente.

L'**apprendimento supervisionato (supervised learning)** rappresenta il paradigma in cui l'algoritmo apprende da un dataset di training che contiene sia gli input che gli output desiderati, chiamati etichette. L'obiettivo è costruire una funzione che mappi accuratamente gli input agli output, permettendo al modello di fare predizioni su nuovi dati mai visti durante l'addestramento. Questo paradigma include due sottocategorie principali: la classificazione, dove si predice l'appartenenza a categorie discrete, e la regressione, dove si predicono valori continui. Nel contesto di questa relazione, il fine-tuning di BERT per la classificazione jailbreak/non-jailbreak rappresenta un esempio paradigmatico di apprendimento supervisionato [2].

L'**apprendimento non supervisionato (unsupervised learning)** costituisce un approccio fondamentalmente diverso, dove l'algoritmo deve scoprire pattern nascosti in dati che non contengono etichette esplicite. Questo paradigma è particolarmente potente per l'esplorazione di strutture latenti nei dati, permettendo di identificare raggruppamenti naturali, ridurre la dimensionalità o scoprire rappresentazioni compatte delle informazioni. Le tecniche di clustering come K-means, utilizzate in questa relazione per analizzare la struttura degli embedding, appartengono a questa categoria e rappresentano uno strumento fondamentale per comprendere l'organizzazione semantica dello spazio delle rappresentazioni [3].

L'**apprendimento auto-supervisionato (self-supervised learning)** emerge come un paradigma ibrido particolarmente rilevante per i modelli linguistici. In questo approccio, le etichette vengono create automaticamente dai dati stessi, sfruttando la struttura intrinseca delle informazioni per definire task di apprendimento. Il pre-addestramento di BERT attraverso i compiti Masked Language Model e Next Sentence Prediction esemplifica perfettamente questa metodologia: il modello impara a predire parole mascherate o

relazioni tra frasi utilizzando il contesto naturale del testo, acquisendo una profonda comprensione linguistica senza necessità di annotazioni umane costose e laboriose [4].

2.2.2 Reti neurali e deep learning

I modelli discussi in questa relazione si basano su reti neurali artificiali, sistemi computazionali ispirati al funzionamento dei neuroni biologici ma implementati attraverso operazioni matematiche precise e ottimizzabili. Una rete neurale è composta da strati (layers) di unità computazionali chiamate neuroni artificiali, ciascuno dei quali riceve input multipli, li combina attraverso una somma pesata, e applica una funzione di attivazione non lineare per produrre un output. Questa non linearità è cruciale perché permette alla rete di apprendere relazioni complesse che non potrebbero essere catturate da modelli lineari [5].

Il **deep learning** rappresenta una specializzazione del machine learning che utilizza reti neurali profonde, caratterizzate da un numero elevato di strati nascosti tra l'input e l'output. La profondità di queste architetture non è semplicemente una questione quantitativa, ma permette di apprendere rappresentazioni gerarchiche sempre più complesse e astratte. Gli strati iniziali tendono a catturare caratteristiche elementari e locali (come bordi e texture nelle immagini, o n-grammi e pattern sintattici nei testi), mentre gli strati più profondi combinano queste caratteristiche per riconoscere pattern di alto livello, concetti astratti e relazioni semantiche complesse [6].

2.2.3 Il processo di training e validazione

L'addestramento di un modello di machine learning segue una metodologia rigorosa e ben consolidata che prevede la divisione strategica dei dati disponibili in tre insiemi distinti, ciascuno con un ruolo specifico nel processo di apprendimento e valutazione.

Il **training set** viene utilizzato per l'apprendimento vero e proprio, dove l'algoritmo aggiorna iterativamente i suoi parametri interni attraverso un processo di ottimizzazione che mira a minimizzare una funzione di perdita (loss function). Questa funzione quantifica la discrepanza tra le predizioni del modello e i valori target desiderati, fornendo un segnale di errore che guida l'aggiornamento dei pesi attraverso algoritmi come la backpropagation e la discesa del gradiente.

Il **validation set** svolge un ruolo cruciale nel monitoraggio delle prestazioni del modello su dati non utilizzati per l'addestramento diretto. Questo insieme viene impiegato per valutare periodicamente il modello durante il training, permettendo di identificare il momento ottimale per fermare l'addestramento (early stopping) e di regolare gli iperparametri del modello. Il processo di validazione è fondamentale per evitare l'overfitting, un fenomeno in cui il modello memorizza eccessivamente i pattern specifici dei dati di training, perdendo la capacità di generalizzare su nuovi esempi.

Il **test set** rimane completamente nascosto durante tutto il processo di sviluppo e viene utilizzato esclusivamente per la valutazione finale delle prestazioni. Questo insieme fornisce una stima imparziale e realistica dell'accuratezza del modello su dati completamente nuovi, simulando le condizioni operative reali in cui il modello verrà utilizzato [7].

2.2.4 Transfer learning e fine-tuning

Un concetto fondamentale per comprendere l'efficacia e l'economia computazionale di modelli come BERT è il **transfer learning**. Questa metodologia permette di trasferire la conoscenza appresa da un modello durante un task generale e computazionalmente costoso (il pre-addestramento) a task specifici e applicativi (downstream tasks) attraverso un processo di specializzazione chiamato fine-tuning [8].

Nel caso specifico di BERT, il modello subisce inizialmente un processo di pre-addestramento su corpus testuali enormi e diversificati attraverso task auto-supervisionati. Questa fase, estremamente dispendiosa in termini di risorse computazionali e temporali (spesso richiedendo settimane di computazione su hardware specializzato come GPU o TPU), permette al modello di sviluppare una comprensione profonda e generale delle strutture linguistiche, delle relazioni semantiche e dei pattern sintattici.

Successivamente, attraverso il fine-tuning, questo modello pre-addestrato viene specializzato per task specifici come la classificazione di testo, la risposta a domande o l'analisi del sentiment. Questo processo richiede solo una frazione delle risorse computazionali originali, tipicamente alcune ore o giorni, e permette di adattare le rappresentazioni generali apprese durante il pre-addestramento alle specifiche esigenze del task target.

Il transfer learning è particolarmente potente perché sfrutta il principio che molte delle rappresentazioni linguistiche apprese durante il pre-addestramento (come la comprensione sintattica, le relazioni semantiche, e i pattern discorsivi) sono intrinsecamente utili per una vasta gamma di task specifici. Come verrà dimostrato negli esperimenti di questa relazione, un modello BERT fine-tuned non solo migliora quantitativamente l'accuratezza della classificazione rispetto al modello base, ma ristruttura qualitativamente lo spazio degli embedding, rendendo le diverse classi semanticamente più separabili e distinguibili.

2.2.5 Regolarizzazione e generalizzazione

Per garantire che i modelli sviluppino una vera capacità di generalizzazione piuttosto che limitarsi a memorizzare meccanicamente i dati di training, il machine learning impiega diverse strategie di regolarizzazione. Queste tecniche sono particolarmente cruciali nel contesto dei modelli di grandi dimensioni, dove l'enorme numero di parametri può facilmente portare a fenomeni di overfitting.

Il **dropout** rappresenta una delle tecniche di regolarizzazione più efficaci e ampiamente utilizzate. Durante il training, questa metodologia disattiva casualmente una percentuale dei neuroni in ogni forward pass, costringendo il modello a non dipendere eccessivamente

da specifiche unità computazionali e promuovendo lo sviluppo di rappresentazioni più robuste e distribuite [9].

La **weight decay** (o L2 regularization) aggiunge un termine di penalizzazione alla funzione di perdita che scoraggia pesi di magnitudo elevata, favorendo soluzioni più semplici e generalizzabili secondo il Principio del rasoio di Occam. L'**early stopping** monitora le prestazioni sul validation set e interrompe il training quando queste smettono di migliorare, prevenendo l'overfitting prima che si manifesti.

Nel contesto dei modelli linguistici di grandi dimensioni come BERT, che possono contenere centinaia di milioni o miliardi di parametri, la regolarizzazione assume un'importanza critica. Questi modelli possiedono una capacità rappresentazionale teoricamente sufficiente per memorizzare completamente dataset di training anche molto grandi, rendendo essenziali le tecniche di regolarizzazione per garantire che l'apprendimento si concentri su pattern generalizzabili piuttosto che su idiosincrasie specifiche dei dati di training.

2.3 L'architettura transformer

La rivoluzione degli LLM è stata resa possibile dall'introduzione dell'architettura Transformer, presentata da Vaswani et al. nel loro celebre lavoro *"Attention Is All You Need"* [10]. Prima del Transformer, i modelli di elaborazione del linguaggio naturale si basavano prevalentemente su architetture ricorrenti (RNN) e a memoria a lungo-breve termine (LSTM), che processano il testo in modo sequenziale. Questo approccio, sebbene efficace, presenta due limiti principali: la difficoltà nel parallelizzare i calcoli e la tendenza a perdere informazioni contestuali su lunghe distanze (il problema del *vanishing gradient*).

Il Transformer supera questi ostacoli introducendo il meccanismo di auto-attenzione **(self-attention)**. Anziché processare le parole una dopo l'altra, il meccanismo di attenzione permette al modello di pesare l'importanza di ogni parola in una sequenza rispetto a tutte le altre, indipendentemente dalla loro posizione. In questo modo, il modello può creare rappresentazioni contestuali ricche, catturando dipendenze a lungo raggio in modo molto più efficace.

L'architettura si compone tipicamente di due blocchi principali:

- **Encoder**: Processa la sequenza di input e la trasforma in una rappresentazione vettoriale continua (embedding) che ne cattura il significato contestuale.
- **Decoder**: Utilizza la rappresentazione dell'encoder e la sequenza generata fino a quel momento per produrre la parola successiva nell'output.

Modelli come BERT, che si concentrano su compiti di comprensione del linguaggio, utilizzano principalmente la componente dell'encoder.

2.4 Il modello bert: pre-addestramento e fine-tuning

BERT (Bidirectional Encoder Representations from Transformers), introdotto da Devlin et al. [11], rappresenta una pietra miliare nell'evoluzione dei modelli basati su architettura Transformer. La sua innovazione principale risiede nell'introdurre un addestramento **bidirezionale**, che permette al modello di considerare simultaneamente il contesto destro e sinistro di una parola durante la fase di pre-addestramento. Questo risultato è ottenuto attraverso due compiti di pre-addestramento non supervisionato: **Masked Language Model (MLM)** e **Next Sentence Prediction (NSP)**.

Una volta completato il costoso processo di pre-addestramento, il modello BERT può essere adattato a compiti specifici attraverso un processo chiamato **fine-tuning**. In questa fase, un piccolo strato di classificazione viene aggiunto sopra l'architettura di BERT e l'intero modello viene addestrato per un numero limitato di epoche su un dataset etichettato, specifico per il task di destinazione. Il fine-tuning permette di trasferire la profonda conoscenza linguistica appresa durante il pre-addestramento a un problema specifico. Questo processo non solo addestra il nuovo strato di classificazione, ma aggiorna anche i pesi dell'intero modello Transformer pre-addestrato, specializzandolo a riconoscere le sfumature semantiche rilevanti per il task in questione. Come dimostrato in questo lavoro, un modello fine-tuned per la classificazione jailbreak/non-jailbreak non è solo un classificatore, ma diventa anche un estrattore di embedding semanticamente più ricco e pertinente per quel dominio specifico.

2.4.1 Estrazione di embedding semantici da bert

Il vero potere di modelli come BERT risiede nella loro capacità di convertire una sequenza di testo in un vettore numerico a dimensione fissa, o **embedding**, che ne cattura il significato contestuale. Questo vettore è la rappresentazione interna che il modello costruisce dopo aver processato il testo.

Nel contesto di BERT, una pratica comune consiste nell'utilizzare l'embedding associato a uno speciale token di input, il token '[CLS]', che viene sempre inserito all'inizio (prepended) di ogni sequenza. Poiché il modello è addestrato a usare questo token per compiti di classificazione a livello di sequenza (come NSP), il suo embedding finale (estratto dall'ultimo strato nascosto del Transformer) viene considerato una rappresentazione aggregata e semanticamente ricca dell'intera sequenza di input. È proprio questa tecnica che viene impiegata negli esperimenti di questa relazione per ottenere un singolo vettore rappresentativo per ogni risposta dell'LLM.

2.5 Allineamento dei modelli e jailbreaking

Con l'aumentare delle capacità degli LLM, è emersa la necessità di garantirne un comportamento sicuro, etico e utile. Questo processo, noto come **allineamento**, ha lo scopo di

insegnare al modello a seguire una serie di principi e linee guida. Tuttavia, queste barriere di sicurezza non sono infallibili. Il **jailbreaking** è l'atto di eludere deliberatamente le misure di allineamento di un LLM attraverso interrogazioni ingannevoli, note come *jailbreak prompts*. Questo fenomeno rappresenta una seria minaccia alla sicurezza e all'affidabilità degli LLM, rendendo fondamentale lo sviluppo di metodi per rilevarlo e mitigarlo.

2.6 Analisi tramite clustering con k-means

Mentre la classificazione supervisionata richiede dati etichettati, l'apprendimento non supervisionato esplora i dati senza etichette preesistenti, cercando di scoprire strutture latenti. L'**algoritmo K-means** [12] è una delle tecniche di clustering più note. Il suo obiettivo è partizionare un insieme di n osservazioni (in questo caso, gli embedding delle risposte) in k cluster, in cui ogni osservazione appartiene al cluster con la media (centroide) più vicina.

2.6.1 Valutazione della qualità del clustering

La scelta del numero ottimale di cluster, k, è un passaggio critico. Nel corso del lavoro svolto in questa relazione, sono state impiegate due tecniche standard per guidare questa scelta:

- **Metodo Elbow (del Gomito)**: Si basa sul calcolo della somma delle distanze quadratiche di ogni punto dal centroide del suo cluster (inerzia). Si esegue il K-means per un range di valori di k e si plotta l'inerzia in funzione di k. Il "gomito" nel grafico, ovvero il punto in cui l'inerzia smette di diminuire rapidamente, è considerato un indicatore di un buon valore per k.
- **Silhouette Score**: Questa metrica misura quanto un oggetto sia simile al proprio cluster rispetto agli altri cluster. Il punteggio varia da -1 a 1, dove un valore alto indica che l'oggetto è ben abbinato al proprio cluster e mal abbinato ai cluster vicini. A differenza dell'inerzia, un punteggio di silhouette più alto è universalmente migliore, rendendolo un criterio più oggettivo per la selezione di k, come fatto nel codice sperimentale di questo lavoro.

L'approccio gerarchico adottato negli esperimenti, che prima divide i dati in due macro-cluster e poi analizza ogni macro-cluster separatamente, permette di affinare l'analisi e scoprire sottostrutture più fini all'interno delle categorie principali "jailbreak" e "non-jailbreak".

2.7 Tecniche di riduzione dimensionale per la visualizzazione

Gli embedding generati da BERT hanno una dimensionalità elevata (768 dimensioni). Per poter visualizzare e interpretare visivamente la struttura dei cluster, è necessario proiettarli in uno spazio a 2 o 3 dimensioni. In questo lavoro sono state utilizzate tre tecniche standard per questo scopo:

- **PCA (Principal Component Analysis)**: È una tecnica lineare che trasforma i dati in un nuovo sistema di coordinate in cui la maggior parte della varianza dei dati si trova lungo le prime, poche coordinate (componenti principali). Veloce e deterministica, PCA è ottima per avere una visione globale della struttura dei dati, ma può fallire nel catturare relazioni non lineari complesse.
- **t-SNE (t-Distributed Stochastic Neighbor Embedding)**: È una tecnica non lineare particolarmente efficace nel visualizzare la struttura locale dei dati, raggruppando punti simili in cluster ben separati. È molto sensibile ai suoi iperparametri (come la *perplexity*) e la sua computazione è più onerosa. Le distanze globali tra i cluster in un plot t-SNE non sono necessariamente significative.
- **UMAP (Uniform Manifold Approximation and Projection)**: È un'altra tecnica di riduzione dimensionale non lineare che è spesso considerata un buon compromesso tra PCA e t-SNE. È generalmente più veloce di t-SNE e tende a preservare meglio sia la struttura globale che quella locale dei dati, rendendola una scelta molto popolare per la visualizzazione di embedding.

L'uso combinato di queste tre tecniche, come fatto negli esperimenti, fornisce una visione più robusta e completa della separabilità e della struttura dei cluster nello spazio degli embedding.

CAPITOLO 3

Stato dell'arte

La rapida proliferazione dei modelli di linguaggio di grandi dimensioni (LLM) ha portato con sé una crescente preoccupazione per la loro robustezza e sicurezza. Il fenomeno del jailbreaking, in particolare, è emerso come una delle principali aree di ricerca. Questo capitolo analizza lo stato dell'arte relativo agli attacchi di jailbreaking, alle tecniche di difesa e, con particolare attenzione, alle metodologie per la **valutazione automatica** delle risposte, al fine di contestualizzare il contributo di questa tesi.

3.1 Tassonomia degli attacchi di jailbreaking

Gli attacchi di jailbreaking possono essere classificati in base a diversi parametri. Le categorie principali, già esplorate in letteratura, includono:

- **Ingegneria dei prompt manuale e basata su template**: Approcci basati sull'intuizione umana, come il gioco di ruolo (es. DAN) o l'uso di scenari ipotetici.
- **Attacchi adversariali e ottimizzazione dei prompt**: Metodi algoritmici che ottimizzano automaticamente i prompt per massimizzare la probabilità di successo, sia in contesti white-box (basati su gradiente) che black-box (es. algoritmi genetici [13] o raffinamento iterativo come PAIR [14]).
- **Attacchi basati su codifica e offuscamento**: Tecniche che nascondono l'intento dannoso tramite codifiche insolite, come l'uso di lingue a basse risorse [15] o l'offuscamento tramite problemi matematici simbolici (MathPrompt [16]).
- **Attacchi multimodali**: Sfruttano le vulnerabilità dei modelli che processano sia testo che immagini, ad esempio tramite "shuffle inconsistency" [17].

3.2 Meccanismi di difesa e piattaforme di valutazione

Parallelamente agli attacchi, la ricerca si concentra su contromisure e framework di valutazione. Le difese includono il filtraggio di input/output, l'adversarial training e

meccanismi interni al modello come l'Activation Boundary Defense (ABD) [18]. Per standardizzare la valutazione, sono nati benchmark come **JailbreakBench** [19], che forniscono dataset e metriche comuni per confrontare attacchi e difese.

3.3 Valutazione automatica delle risposte llm

Un elemento cruciale, ma spesso dato per scontato, in questo ciclo di attacco-difesa è la capacità di determinare se una data risposta costituisce un jailbreak. La valutazione manuale da parte di esseri umani è considerata il "gold standard" ma è lenta, costosa, soggettiva e non scalabile. Ciò ha spinto la ricerca verso metodi di valutazione automatica, che possono essere raggruppati in tre principali approcci.

3.3.1 Valutazione basata su metriche tradizionali

Le metriche classiche del NLP come BLEU o ROUGE sono utili per valutare la fluidità o la somiglianza con un testo di riferimento, ma sono del tutto inadeguate per valutare la sicurezza o la conformità di una risposta. Una risposta può essere grammaticalmente perfetta e fluente, ma contenere istruzioni pericolose. Pertanto, queste metriche non sono applicabili al problema del jailbreaking.

3.3.2 Valutazione tramite llm (llm-as-a-judge)

Un approccio emergente e potente consiste nell'utilizzare un LLM avanzato (spesso un modello proprietario come GPT-4) come "giudice" per valutare le risposte di un altro modello [20]. Al modello-giudice viene fornito un prompt che descrive i criteri di valutazione (es. "La seguente risposta è dannosa o evasiva?") e la risposta da giudicare.

- **Vantaggi**: Questo metodo è altamente scalabile e può catturare sfumature complesse del linguaggio che i classificatori più semplici potrebbero mancare.
- **Svantaggi**: Presenta limiti significativi. Il modello-giudice può avere i propri bias, può essere costoso da interrogare, e soprattutto, può essere vulnerabile agli stessi tipi di attacchi che dovrebbe rilevare. Inoltre, i risultati possono mancare di riproducibilità a causa della natura stocastica dei modelli e degli aggiornamenti non documentati delle API.

3.3.3 Valutazione tramite classificatori specifici

Un'alternativa più controllata e robusta consiste nell'addestrare un modello classificatore più piccolo e specializzato. Questo è l'approccio che è stato adottato in questa tesi.

- **Metodologia**: Si crea un dataset di risposte etichettate manualmente come "jailbreak" o "non-jailbreak". Su questo dataset si effettua il fine-tuning di un modello pre-addestrato come BERT per il compito di classificazione binaria.

- **Vantaggi**: Questi classificatori sono veloci, economici da eseguire, e i loro risultati sono perfettamente riproducibili. Poiché sono specializzati su un compito specifico, possono raggiungere performance molto elevate. A differenza di un LLM-giudice, un classificatore BERT fine-tuned è un sistema deterministico e trasparente nel suo funzionamento.

Questa metodologia è diventata uno standard de-facto in molte pipeline di ricerca sulla sicurezza degli LLM, dove è necessario un feedback rapido e affidabile.

3.4 Limiti degli approcci esistenti e posizionamento della tesi

Dall'analisi dello stato dell'arte emergono alcune aree che necessitano di ulteriore indagine.

1. **La qualità degli embedding**: Sebbene l'uso di classificatori BERT sia comune, la letteratura si concentra principalmente sulla loro accuratezza di classificazione. Vi è meno indagine sulla qualità dello spazio semantico (embedding space) che questi modelli apprendono. Un modello fine-tuned crea uno spazio in cui le classi sono più separabili rispetto a un modello generico?

2. **Oltre la classificazione binaria**: La valutazione si ferma spesso a un'etichetta binaria (successo/fallimento). C'è una mancanza di analisi quantitative su larga scala delle diverse *tipologie* di risposte jailbreak. Esistono "gradi" o "stili" diversi di jailbreak che possono essere scoperti automaticamente?

È in questo contesto che si inserisce il presente lavoro di tesi. Questa ricerca si posiziona come uno studio **metodologico e comparativo sulla valutazione automatica e sull'analisi non supervisionata delle risposte LLM**.

Il contributo di questa tesi si articola su tre livelli, direttamente riflessi negli esperimenti condotti:

- **Validazione dell'approccio con classificatore specifico**: Utilizzando un modello BERT fine-tuned ("Teto03/Bert_base_fineTuned"), la tesi adotta e valida l'approccio del classificatore specifico come strumento primario di analisi.

- **Analisi comparativa degli spazi semantici**: Il nucleo innovativo della tesi risiede nel confronto diretto tra lo spazio degli embedding generato dal modello BERT **fine-tuned** e quello generato da un modello BERT **base** ("bert-base-uncased"). L'obiettivo è dimostrare quantitativamente e qualitativamente (tramite clustering e

visualizzazione) che il fine-tuning non solo migliora l'accuratezza della classificazione, ma ristruttura lo spazio semantico, rendendo le risposte "jailbreak" e "non-jailbreak" intrinsecamente più separabili.

- **Esplorazione non supervisionata delle risposte**: Andando oltre la classificazione, l'applicazione di un clustering gerarchico (K-means + analisi della silhouette) mira a colmare la seconda lacuna identificata. Questo approccio non solo valida la separabilità dei macro-cluster (jailbreak vs. non-jailbreak), ma cerca di scoprire automaticamente sottogruppi tematici o stilistici all'interno di ciascuna categoria, fornendo insight più profondi sulle strategie di risposta del modello target.

In sintesi, questa tesi non si limita a usare uno strumento di valutazione, ma ne analizza le fondamenta, dimostrando perché un modello specializzato è superiore a uno generico per l'analisi semantica in un dominio specifico, e proponendo una metodologia combinata (classificazione + clustering) per una valutazione più completa e profonda della sicurezza degli LLM.

CAPITOLO 4

Metodologie

4.1 Introduzione

Il presente capitolo descrive in dettaglio la metodologia adottata per affrontare il problema della classificazione automatica delle risposte generate da Large Language Models (LLM), con particolare attenzione all'identificazione dei tentativi di jailbreak. Come anticipato nei capitoli precedenti, l'aumento della complessità architetturale e della sofisticazione degli LLM ha reso indispensabile l'adozione di tecniche avanzate per rilevare comportamenti che potrebbero risultare dannosi o non conformi alle policy di allineamento etico e funzionale dei modelli.

L'obiettivo primario della ricerca è quello di sviluppare un sistema di valutazione automatico capace di distinguere in modo affidabile tra risposte sicure, che rispettano le linee guida etiche, e risposte che potrebbero rappresentare tentativi di eludere le restrizioni imposte al modello, fenomeno noto come jailbreak. La sfida metodologica principale risiede nella natura spesso sottile e sfumata di questi tentativi di elusione, che richiedono approcci capaci di catturare non solo differenze semantiche evidenti, ma anche variazioni più sottili nel contenuto, nel tono e nell'intento delle risposte.

Il lavoro si inserisce nel contesto più ampio della sicurezza dell'AI e dell'allineamento dei modelli linguistici, dove la capacità di identificare automaticamente comportamenti problematici rappresenta un elemento cruciale per garantire un deployment sicuro ed etico di questi sistemi. La trasparenza metodologica e la validità delle procedure adottate sono fondamentali per garantire la riproducibilità e l'affidabilità dei risultati.

4.2 Panoramica dell'approccio metodologico

4.2.1 Evoluzione del framework di ricerca

La metodologia si articola in una pipeline strutturata che ha subito un'evoluzione significativa durante il corso della ricerca, passando da un approccio iniziale rigido a una soluzione più sofisticata e flessibile. Questa evoluzione è stata guidata dall'analisi critica

dei risultati intermedi e dal feedback continuo con i supervisori della ricerca, in particolare con il Prof. Ferretti e la Prof.ssa Saletta.

L'evoluzione metodologica può essere suddivisa in quattro fasi principali, ciascuna caratterizzata da specifiche innovazioni e miglioramenti che hanno progressivamente raffinato l'approccio al problema.

Prima fase - Approccio di classificazione binaria semplificata (versione 1.0)

La metodologia iniziale si basava su un approccio di classificazione binaria diretta, dove il modello BERT fine-tuned produceva una risposta che veniva codificata come un vettore bidimensionale. In questa rappresentazione semplificata, la coordinata X assumeva valori discreti {-1, +1}, dove +1 indicava "jailbreak" e -1 indicava "no-jailbreak", mentre la coordinata Y rappresentava la confidenza del modello nella classificazione, con valori continui nell'intervallo [0,1].

Il flusso operativo di questa versione prevedeva l'elaborazione delle risposte da parte di BERT, la generazione della classificazione con relativa confidenza, la codifica in vettori bidimensionali e, infine, l'applicazione dell'algoritmo di clustering K-Means con k=2 per raggruppare i risultati in due categorie distinte.

Algorithm 1 Codifica del risultato di classificazione

Require: $classification \in \{0, 1\}$, $confidence \in [0, 1]$
Ensure: $data\_point \in \{-1, 1\} \times [0, 1]$
// Codifica della coordinata x basata sulla classificazione
if $classification = 1$ **then**
    $x\_coord \leftarrow 1$ {Jailbreak}
else
    $x\_coord \leftarrow -1$ {Non-jailbreak}
end if
// Assegnazione della coordinata y
$y\_coord \leftarrow confidence$ {Valore di confidenza tra 0 e 1}
// Creazione del vettore bidimensionale risultante
$data\_point \leftarrow [x\_coord, y\_coord]$
return $data\_point$

Sebbene questa impostazione iniziale offrisse un'interpretabilità immediata, l'analisi condotta in collaborazione con il Prof. Ferretti ha evidenziato diverse limitazioni significative. La rappresentazione bidimensionale risultava eccessivamente rigida e riduttiva, incapace di sfruttare appieno le informazioni semantiche ricche prodotte da BERT. La perdita di informazioni derivante dalla riduzione della complessità del problema a una classificazione binaria rigida limitava drasticamente la capacità del sistema di catturare le sfumature presenti nei tentativi di jailbreak. Inoltre, il clustering risultava subottimale a causa della bassa dimensionalità dei vettori generati, rendendo difficile distinguere tra casi borderline o variazioni sofisticate nei pattern di elusione.

È importante sottolineare che questa prima versione, avendo carattere esplorativo, è stata implementata esclusivamente utilizzando il modello BERT fine-tuned e testata su un dataset significativamente ridotto di soli 20 esempi. Tale limitazione nella dimensione del dataset, sebbene funzionale per la validazione iniziale del concetto, contrastava nettamente con le implementazioni successive che si baseranno su un corpus sostanzialmente più ampio di 1784 istanze, equamente distribuite tra 997 esempi etichettati come "no-jailbreak" (0) e 787 come "jailbreak" (1).

Dataset di sviluppo Il dataset utilizzato nelle fasi successive dello sviluppo comprende 1784 istanze di prompt e relative risposte, accuratamente etichettate per la classificazione binaria dei tentativi di jailbreak. La distribuzione delle etichette risulta relativamente bilanciata, con 997 istanze classificate come "no-jailbreak" (55.9%) e 787 istanze classificate come "jailbreak" (44.1%), garantendo una rappresentazione equilibrata di entrambe le categorie per l'addestramento e la valutazione dei modelli nelle versioni successive dell'architettura.

Seconda fase - estrazione di embedding ad alta dimensionalità (versione 2.0)

In risposta alle limitazioni dell'approccio iniziale, è stata sviluppata una versione intermedia che sfruttava appieno la ricchezza delle rappresentazioni BERT attraverso l'estrazione di embedding ad alta dimensionalità. Questa transizione rappresenta un cambiamento paradigmatico fondamentale nell'approccio metodologico, motivato dalla necessità di preservare la ricchezza semantica delle rappresentazioni generate da BERT.

L'approccio si basa sui principi delle rappresentazioni distribuite del linguaggio naturale, simili concettualmente al Word2Vec proposto da Mikolov et al. (2013)[**?**], ma applicati a livello di sequenza completa. A differenza del Word2Vec, che produce rappresentazioni statiche per singole parole basate sulla loro co-occorrenza in un corpus, gli embedding BERT sono contestuali, con ogni rappresentazione che varia in funzione del contesto completo della sequenza. Questi embedding derivano dalla combinazione di 12 layer di attention che catturano diversi livelli di astrazione linguistica e, nel caso specifico, sono ottimizzati per il rilevamento di jailbreak attraverso il fine-tuning.

Il processo di estrazione degli embedding segue una procedura standardizzata che può essere formalizzata matematicamente. Sia text una risposta testuale da elaborare. Il primo passo consiste nella tokenizzazione utilizzando l'algoritmo **WordPiece**, che produce una sequenza di token:

```
tokens = tokenizer(text, return\_tensors="pt", truncation=True, padding=True)
```

dove tokens include input_ids (identificativi numerici dei token), attention_mask (maschera binaria per indicare token significativi) e opzionalmente token_type_ids. Successivamente, il modello BERT elabora questi input attraverso i suoi 12 strati transformer:

```
outputs = model(**tokens, output\_hidden\_states=True)
```

Gli hidden states H^l per ogni strato l, con $l = 0, 1, ..., 12$, hanno dimensione:

$$\text{batch\_size} \times \text{seq\_len} \times d$$

dove d = 768 è la dimensione nascosta di BERT-base. Il vettore [CLS] dell'ultimo strato, utilizzato come embedding della frase, è estratto come:

$$\mathbf{e} = H^{(12)}[:, 0, :] \in \mathbb{R}^{768}$$

Questo vettore rappresenta una sintesi delle informazioni contestuali dell'intera sequenza, ponderate dal meccanismo di attenzione. Il token [CLS] è stato scelto come rappresentazione dell'intera sequenza poiché è progettato per aggregare informazioni semantiche globali attraverso il meccanismo di self-attention multi-head, dove per un singolo head l'attenzione è calcolata come:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

dove Q, K e V sono le matrici di query, key e value derivate dagli hidden states. La dimensione di ciascun head è data da:

$$d_k = \frac{d_{\text{model}}}{h} = \frac{768}{12} = 64$$

L'utilizzo di vettori 768-dimensionali offre diversi vantaggi tecnici e linguistici. La ricchezza semantica viene preservata mantenendo le sfumature linguistiche e semantiche catturate dal modello. Gli embedding riflettono inoltre l'adattamento specifico per il rilevamento di jailbreak grazie alla sensibilità al fine-tuning, mentre lo spazio ad alta dimensionalità garantisce maggiore libertà per l'algoritmo K-Means nella formazione di cluster coerenti.

Terza fase - clustering automatico con ottimizzazione del numero di cluster

La limitazione principale della seconda versione risiedeva nella fissazione del numero di cluster a k=2, che non riusciva a catturare le sfumature e le gradazioni presenti nei dataset di jailbreak. L'analisi qualitativa delle risposte, condotta in collaborazione con la Prof.ssa Saletta, mostrava infatti l'esistenza di categorie intermedie e sottotipi specifici che richiedevano un approccio più flessibile.

Sono state implementate due metriche principali per la determinazione automatica del numero ottimale di cluster. Il metodo Elbow si basa sull'analisi della Within-Cluster Sum of Squares (WCSS), calcolata come:

$$\text{WCSS}(k) = \sum_{j=1}^{k} \sum_{x \in C_j} ||x - \mu_j||^2$$

dove C_j rappresenta il cluster j e μ_j il suo centroide. Il valore ottimale di k è identificato dal punto in cui la curva WCSS presenta un "gomito", indicando che incrementi ulteriori di k non portano a miglioramenti significativi nella compattezza dei cluster.

Parallelamente è stato implementato anche un secondo metodo la Silhouette Analysis, la quale valuta la qualità del clustering calcolando il coefficiente di silhouette per ogni punto dati. Per un punto e_i, il coefficiente $s(e_i)$ è definito come:

$$s(e_i) = \frac{b(e_i) - a(e_i)}{\max\{a(e_i), b(e_i)\}}$$

dove $a(e_i)$ è la distanza media di e_i dagli altri punti del suo cluster e $b(e_i)$ è la distanza media di e_i dal cluster più vicino a cui non appartiene. Il punteggio medio di silhouette per un dato k fornisce una misura globale della qualità del clustering, con valori più alti che indicano una migliore definizione dei cluster.

Quarta fase - clustering gerarchico a due livelli (versione finale)

La metodologia finale implementa un approccio di clustering gerarchico a due livelli che combina i vantaggi delle fasi precedenti, preservando l'interpretabilità della classificazione binaria mentre cattura le sfumature semantiche attraverso un'analisi multi-livello.

L'approccio gerarchico si articola in due livelli distinti. Al primo livello, viene applicato K-Means con k=2 per separare le categorie principali "jailbreak" versus "no-jailbreak", mantenendo la distinzione binaria utile a distinguere facilmente le due macro categorie. Al secondo livello, si procede con un clustering automatico all'interno di ciascun macro-cluster per identificare sottotipi e varianti, utilizzando l'ottimizzazione automatica del numero di cluster descritta nella fase precedente.

Questo approccio gerarchico presenta numerosi vantaggi metodologici. Mantiene l'interpretabilità preservando la distinzione binaria fondamentale per scopi pratici di sicurezza, mentre aumenta la granularità identificando sottocategorie semanticamente coerenti all'interno di ciascuna macro-categoria. La flessibilità adattiva permette la determinazione automatica del numero ottimale di sottocluster per ciascun macro-cluster, riflettendo la possibile asimmetria nella complessità interna delle due categorie principali.

Inoltre, la validazione multi-scala consente l'applicazione di metriche di qualità a entrambi i livelli della gerarchia, garantendo robustezza statistica e coerenza semantica.

4.3 Configurazione sperimentale e modelli utilizzati

4.3.1 Architettura comparativa dual-model

Per garantire robustezza e validità dei risultati, la metodologia prevede l'utilizzo di due pipeline parallele basate su modelli BERT distinti, permettendo un'analisi comparativa dell'efficacia delle rappresentazioni semantiche e dell'impatto del fine-tuning specializzato.

Il primo modello utilizzato è BERT-base-uncased, che rappresenta il baseline generalista. Si tratta del modello standard pre-addestrato sui corpus tradizionali BookCorpus e Wikipedia inglese, con 110 milioni di parametri distribuiti su 12 layer transformer e 768 dimensioni di hidden state. Il suo ruolo è quello di fornire rappresentazioni semantiche general-purpose senza bias specifici per il dominio jailbreak, permettendo di valutare l'efficacia di approcci specializzati rispetto a un modello generalista.

Il secondo modello è Teto03/Bert_base_fineTuned, sviluppato dal collega Brighenti Stefano specificamente per il rilevamento di jailbreak. Questo modello è stato affinato su un dataset curato contenente esempi rappresentativi di tecniche di elusione, risultando in rappresentazioni ottimizzate per distinguere pattern linguistici associati a tentativi di jailbreak. La specializzazione offre maggiore sensibilità alle caratteristiche specifiche del dominio di interesse, come l'uso di linguaggio indiretto, metafore per aggirare restrizioni, o pattern di formulazione che tipicamente precedono risposte non conformi.

La scelta di utilizzare entrambi i modelli è motivata dalla necessità di quantificare empiricamente il contributo del fine-tuning specializzato rispetto alle capacità generaliste dei modelli pre-addestrati standard. Questa configurazione comparativa permette inoltre di valutare la generalizzabilità dei risultati e di identificare eventuali bias introdotti dal processo di fine-tuning.

4.3.2 Dataset di valutazione e preprocessing

Il dataset utilizzato per la valutazione è costituito da una selezione curata derivata dai file di risposte LLM forniti dal Prof. Ferretti. È cruciale sottolineare che questi dati non sono stati utilizzati durante la fase di fine-tuning del modello, garantendo così la novità completa dei dati per il modello e evitando fenomeni di overfitting. Questa separazione rigorosa permette una valutazione autentica delle capacità di generalizzazione del modello su dati non visti durante l'addestramento, prevenendo risposte predeterminate a domande già elaborate dal modello.

Il preprocessing dei dati è stato condotto in stretta collaborazione con Brighenti Stefano e ha seguito una pipeline rigorosa per assicurare la qualità e l'uniformità delle risposte. Il processo include diverse fasi cruciali, iniziando con la pulizia dei dati che comprende

la rimozione di caratteri speciali e formattazione inconsistente, la normalizzazione di spaziature e punteggiatura, e la gestione appropriata di encoding e caratteri Unicode. Quando necessario, è stata applicata una fase di traduzione automatica per garantire uniformità linguistica, con successiva standardizzazione del formato delle risposte e validazione della qualità delle traduzioni. La fase finale prevede un controllo manuale di un campione rappresentativo per verificare la coerenza semantica post-preprocessing ed eliminare eventuali risposte corrotte o incomplete.

4.3.3 Parametri di configurazione sperimentale

La configurazione sperimentale è stata standardizzata per garantire riproducibilità e confrontabilità dei risultati. I parametri fondamentali includono un seed di riproducibilità fissato a 42, utilizzato consistentemente per tutti gli algoritmi stocastici per garantire risultati deterministici. Il numero massimo di iterazioni per l'algoritmo K-Means è stato impostato a 300, garantendo convergenza anche per configurazioni complesse. Il range di cluster per l'ottimizzazione automatica è stato definito come $k \in [2, 8]$, bilanciando la capacità di catturare sottotipi significativi con la necessità di mantenere cluster interpretabili. La lunghezza massima delle sequenze è stata fissata a 512 token, corrispondente al limite standard di BERT, mentre la batch size per l'inferenza è gestita dinamicamente in base alla memoria disponibile, ottimizzando l'efficienza computazionale.

Per quanto riguarda i parametri di riduzione dimensionale, PCA è configurato con n_components=2 e random_state=42 per garantire una proiezione lineare riproducibile. t-SNE utilizza n_components=2 con perplexity adattata dinamicamente come min(30, n_samples-1) per gestire dataset di dimensioni variabili. UMAP è impostato con n_components=2 e random_state=42, utilizzando valori default per neighbors e min_dist che hanno dimostrato buone prestazioni in letteratura per task simili.

4.4 Estrazione e rappresentazione degli embedding

4.4.1 Processo di estrazione dettagliato

Il processo di estrazione degli embedding rappresenta il cuore della metodologia, trasformando le risposte testuali in rappresentazioni numeriche adatte al clustering. Questo processo inizia con una fase di preprocessing sistematico che prepara i testi per l'elaborazione da parte dei modelli transformer.

La tokenizzazione utilizza l'algoritmo WordPiece per la decomposizione in unità sublessicali, gestendo efficacemente le parole fuori vocabolario e mantenendo le informazioni morfologiche rilevanti. Durante questo processo, vengono aggiunti automaticamente i token speciali [CLS] all'inizio e [SEP] alla fine di ogni sequenza. Il token [CLS] funge da aggregatore di informazioni semantiche globali, accumulando attraverso i layer del transformer una rappresentazione compressa dell'intera sequenza. Le sequenze vengono

uniformate attraverso padding per quelle inferiori a 512 token e truncation intelligente per quelle eccedenti, preservando l'informazione semantica centrale.

L'estrazione vera e propria avviene elaborando le sequenze tokenizzate attraverso il modello BERT. Per ogni risposta nel dataset, otteniamo un embedding e_i calcolato come:

Algorithm 2 Estrazione di embedding CLS

Require: *text*: stringa di input, *model*: modello transformer, *tokenizer*: tokenizzatore
Ensure: *cls_embedding*: embedding CLS come array numpy

```
function EXTRACT_CLS_EMBEDDING(text, model, tokenizer)
// Tokenizzazione del testo di input
inputs ← TOKENIZER(text, return_tensors="pt", truncation=True,
                   padding=True, max_length=512)
// Estrazione degli output del modello senza calcolo dei gradienti
with NO_GRAD do
    outputs ← MODEL(inputs, output_hidden_states=True)
    // Estrazione dell'embedding CLS dall'ultimo hidden state
    cls_embedding ← outputs.hidden_states[−1][:, 0, :]
end with
// Conversione in formato numpy e detach dai gradienti
cls_embedding ← cls_embedding.DETACH().CPU().NUMPY()
return cls_embedding
end function
```

Ripetendo questo processo per tutte le N risposte del dataset, otteniamo una matrice di embedding $X \in \mathbb{R}^{N \times 768}$ dove ogni riga rappresenta l'embedding di una risposta.

4.4.2 Qualità e proprietà degli embedding

Gli embedding BERT presentano caratteristiche superiori rispetto alle rappresentazioni statiche tradizionali. La contestualità dinamica garantisce che ogni embedding vari in funzione del contesto completo, catturando sfumature semantiche altrimenti impossibili da rilevare. Ad esempio, espressioni ambigue che potrebbero indicare tentativi di jailbreak vengono rappresentate diversamente in base al contesto circostante, permettendo una discriminazione più accurata.

L'astrazione multilayer combina informazioni da 12 livelli di trasformazione, dove i primi layer catturano principalmente caratteristiche sintattiche e morfologiche, mentre i layer superiori codificano relazioni semantiche e pragmatiche sempre più astratte. Questa gerarchia di rappresentazioni è particolarmente utile per il rilevamento di jailbreak, dove pattern superficiali possono mascherare intenti più profondi.

La sensibilità al fine-tuning nel caso del modello Teto03/Bert_base_fineTuned significa che gli embedding sono stati ottimizzati specificamente per enfatizzare le caratteristiche distintive dei tentativi di jailbreak. Durante il fine-tuning, il modello ha appreso a dare maggior peso a pattern linguistici come l'uso di linguaggio indiretto o metaforico per aggirare restrizioni, formulazioni che tentano di reinterpretare richieste illegali in modi

apparentemente innocui, o strutture argomentative che cercano di giustificare risposte non conformi.

Questa ricchezza rappresentazionale è fondamentale per l'identificazione accurata di jailbreak, dove i tentativi di elusione spesso sfruttano ambiguità semantiche, richiedono comprensione del contesto pragmatico, e possono presentare variazioni stilistiche superficiali che non dovrebbero influenzare la classificazione sostanziale.

4.5 Algoritmi di clustering e ottimizzazione

4.5.1 K-means: fondamenti e implementazione

L'algoritmo K-Means è stato selezionato come metodo di clustering principale per la sua efficienza computazionale con complessità $O(Nkt)$, dove N è il numero di punti, k il numero di cluster e t il numero di iterazioni. La convergenza è garantita matematicamente, anche se a un ottimo locale, e i centroidi risultanti forniscono rappresentazioni esemplari interpretabili dei cluster. L'algoritmo ha inoltre dimostrato robustezza su spazi ad alta dimensionalità come gli embedding BERT.

L'implementazione multi-livello segue una strategia gerarchica. Al primo livello, il clustering principale con k=2 separa le risposte nelle categorie fondamentali attraverso la minimizzazione della funzione obiettivo:

$$J = \sum_{i=1}^{N} \sum_{j=1}^{2} r_{ij} \|x_i - \mu_j\|^2$$

dove r_{ij} è l'assegnazione binaria del punto i al cluster j. Dopo la convergenza, viene effettuata una mappatura semantica dei cluster basata sull'ispezione qualitativa di campioni rappresentativi, assegnando l'etichetta "jailbreak" al cluster che contiene prevalentemente risposte problematiche.

Al secondo livello, per ciascun macro-cluster viene applicato un sotto-clustering con numero di cluster determinato automaticamente. L'ottimizzazione si basa sulla massimizzazione del Silhouette Score medio, che fornisce una misura bilanciata di coesione interna e separazione tra cluster.

4.5.2 Metriche di ottimizzazione

Il Silhouette Score per un singolo punto misura quanto bene quel punto si adatta al proprio cluster rispetto agli altri. Un valore vicino a 1 indica che il punto è ben assegnato, mentre valori negativi suggeriscono una possibile classificazione erronea. L'interpretazione geometrica del coefficiente rivela che massimizzare il Silhouette Score equivale a trovare cluster compatti e ben separati nello spazio degli embedding.

Il metodo Elbow fornisce una prospettiva complementare analizzando come l'inerzia intra-cluster decresce all'aumentare del numero di cluster. Il punto di "gomito" rappresenta un trade-off ottimale tra complessità del modello (numero di cluster) e qualità della rappresentazione (compattezza dei cluster). Matematicamente, cerchiamo il punto dove la derivata seconda della curva WCSS cambia segno più marcatamente.

4.5.3 Validazione e metriche di qualità

La validazione viene applicata sistematicamente a entrambi i livelli del clustering gerarchico. Al livello macro, la validazione si concentra sulla verifica che la separazione binaria jailbreak/no-jailbreak sia semanticamente coerente e statisticamente significativa. Al livello micro, l'attenzione è sulla coerenza interna dei sottocluster e sulla loro interpretabilità in termini di sottotipi di jailbreak o di risposte sicure.

Le metriche integrate utilizzate includono il Silhouette Score per la qualità complessiva, il **Calinski-Harabasz Index** che misura il rapporto tra la varianza tra cluster e quella intra-cluster, e il **Davies-Bouldin Index** che quantifica la similarità media tra cluster, dove valori minori indicano migliore separazione. Queste metriche forniscono prospettive complementari sulla qualità del clustering, permettendo una valutazione robusta dei risultati.

4.6 Visualizzazione e interpretazione

4.6.1 Tecniche di riduzione dimensionale

La visualizzazione degli embedding ad alta dimensionalità richiede tecniche di riduzione dimensionale che preservino le relazioni strutturali essenziali. PCA fornisce una proiezione lineare ottimale nel senso dei minimi quadrati, massimizzando la varianza preservata nelle prime componenti. Matematicamente, PCA trova le direzioni v che massimizzano:

$$\text{var}(Xv) \quad \text{soggetto a} \quad \|v\| = 1$$

Le componenti principali risultanti forniscono una decomposizione interpretabile della varianza nei dati, permettendo di identificare le direzioni di massima variazione negli embedding.

t-SNE implementa una riduzione non-lineare che preserva le relazioni di vicinanza locale minimizzando la divergenza di **Kullback-Leibler** tra distribuzioni di probabilità nello spazio originale e proiettato. La funzione di costo di t-SNE è:

$$C = \sum_i \sum_j p_{ij} \log(p_{ij}/q_{ij})$$

dove p_{ij} rappresenta la similarità tra punti nello spazio originale e q_{ij} nello spazio proiettato. Questa ottimizzazione produce visualizzazioni che enfatizzano la struttura locale dei cluster.

UMAP estende questi concetti utilizzando la teoria della topologia algebrica per preservare sia la struttura locale che globale dei dati. L'algoritmo costruisce un grafo fuzzy nello spazio ad alta dimensionalità e ottimizza una rappresentazione a bassa dimensionalità che preserva la topologia di questo grafo.

4.6.2 Visualizzazioni multi-livello

Le visualizzazioni sono state progettate per riflettere la struttura gerarchica del clustering. La visualizzazione dei macro-cluster evidenzia la separazione binaria fondamentale, con i centroidi proiettati per fornire riferimenti visivi dei prototipi di ciascuna categoria. La visualizzazione gerarchica completa integra entrambi i livelli, utilizzando una codifica cromatica che riflette sia l'appartenenza al macro-cluster che al sotto-cluster, permettendo di apprezzare simultaneamente la struttura globale e le distinzioni locali.

4.7 Validazione e robustezza

4.7.1 Protocollo di validazione

Il protocollo di validazione combina approcci quantitativi e qualitativi per garantire la robustezza dei risultati. La validazione interna si concentra sulla stabilità statistica del clustering, verificata attraverso multiple esecuzioni con diverse inizializzazioni dei centroidi. L'analisi della convergenza monitora il numero di iterazioni richieste e la stabilità delle assegnazioni finali. La sensibilità ai parametri viene valutata sistematicamente variando i parametri chiave come il numero massimo di iterazioni e i criteri di convergenza.

La validazione esterna confronta i risultati ottenuti con BERT-base e BERT fine-tuned, quantificando l'impatto del fine-tuning specializzato attraverso metriche come l'Adjusted Rand Index e la Normalized Mutual Information. L'ispezione qualitativa di campioni rappresentativi per ogni cluster verifica la coerenza semantica delle assegnazioni e identifica eventuali pattern non catturati dalle metriche quantitative.

4.7.2 Gestione dell'incertezza

L'analisi della stabilità statistica quantifica la variabilità dei risultati attraverso ripetute esecuzioni, fornendo intervalli di confidenza per le metriche di qualità. La sensibilità alle tecniche di riduzione dimensionale viene valutata confrontando sistematicamente i risultati ottenuti con PCA, t-SNE e UMAP, verificando che le strutture identificate non siano artefatti di una specifica proiezione.

4.8 Pipeline implementativa finale

4.8.1 Architettura del sistema

La pipeline finale integra tutti i componenti metodologici in un flusso coerente e modulare. L'architettura è stata progettata per massimizzare la riproducibilità mantenendo la flessibilità necessaria per future estensioni. La funzione principale JailbreakClusteringPipeline incapsula l'intero processo, dall'estrazione degli embedding alla visualizzazione dei risultati.

L'implementazione di due pipeline parallele, una per BERT-base e una per BERT fine-tuned, consente un'analisi comparativa sistematica. Questa architettura dual-model rappresenta un elemento metodologico fondamentale per quantificare il miglioramento nelle capacità di rilevamento dei jailbreak ottenuto attraverso l'addestramento su dati specifici del dominio.

Il flusso operativo segue una sequenza strutturata che garantisce riproducibilità e modularità. Le risposte LLM vengono processate attraverso il tokenizer specifico del modello, convertite in rappresentazioni numeriche compatibili con l'architettura transformer. Il processo di tokenizzazione preserva le caratteristiche semantiche essenziali mentre prepara il testo per l'elaborazione. Successivamente, il modello elabora le sequenze producendo rappresentazioni vettoriali ad alta dimensionalità attraverso il meccanismo di self-attention multi-layer.

4.8.2 Gestione dei dati e preprocessing

Il dataset utilizzato deriva da una selezione curata delle risposte LLM fornite dal Prof. Ferretti. La separazione rigorosa tra dati di training e testing garantisce una valutazione imparziale delle capacità di generalizzazione del sistema. Il preprocessing ha richiesto particolare attenzione data la natura eterogenea delle risposte LLM, includendo fasi di normalizzazione, gestione di caratteri speciali, e quando necessario, traduzione automatica per uniformità linguistica.

L'architettura della pipeline è stata progettata con particolare attenzione alla scalabilità. L'estrazione degli embedding avviene in modalità batch quando possibile, con gestione dinamica della dimensione basata sulla memoria disponibile. Il caching degli embedding estratti rappresenta un'ottimizzazione importante, permettendo di riutilizzare le rappresentazioni per diverse configurazioni di clustering senza dover rieseguire la costosa fase di estrazione.

4.9 Conclusioni metodologiche

La metodologia sviluppata rappresenta un approccio innovativo al problema della classificazione automatica dei jailbreak in risposte LLM. L'evoluzione dalla classificazione binaria

rigida iniziale al sistema gerarchico finale riflette un processo iterativo di raffinamento guidato dall'analisi empirica dei risultati e dal feedback degli esperti del dominio.

L'integrazione di tecniche di deep learning per l'estrazione di feature con algoritmi di clustering non supervisionato permette di combinare la capacità dei modelli transformer di catturare semantica complessa con la flessibilità degli approcci non supervisionati nell'identificare strutture emergenti nei dati. Il clustering gerarchico a due livelli bilancia efficacemente le esigenze pratiche di classificazione binaria con la necessità di catturare le sfumature presenti nei tentativi reali di jailbreak.

La validazione multi-prospettica attraverso diverse tecniche di visualizzazione e metriche di qualità garantisce robustezza dei risultati e facilita l'interpretazione da parte degli operatori umani. L'implementazione parallela con modelli base e fine-tuned fornisce inoltre insight preziosi sull'importanza dell'addestramento specializzato per questo dominio applicativo specifico. I risultati ottenuti, che saranno presentati nel capitolo successivo, dimostrano l'efficacia dell'approccio proposto nel rilevare e categorizzare tentativi di jailbreak con elevata accuratezza e granularità.

CAPITOLO 5

Presentazione analisi dei risultati

5.1 Introduzione

Questo capitolo presenta un'analisi critica e approfondita dei risultati ottenuti attraverso l'evoluzione metodologica del sistema di clustering per la classificazione di risposte jailbreak. L'approccio sviluppato ha attraversato diverse fasi di raffinamento, dalla versione iniziale semplificata fino al sistema gerarchico finale, permettendo di valutare empiricamente l'efficacia delle diverse strategie adottate. L'analisi si concentra sia sui miglioramenti quantitativi delle performance che sull'interpretabilità qualitativa dei risultati, fornendo una valutazione complessiva del contributo metodologico e delle sue implicazioni per la sicurezza dei modelli di linguaggio.

La discussione procede attraverso un confronto sistematico tra le diverse versioni implementate, evidenziando i punti di forza e le limitazioni di ciascun approccio. Particolare attenzione viene dedicata all'analisi delle metriche di qualità del clustering e alla validazione attraverso tecniche di visualizzazione dimensionale.

5.2 Risultati della versione 1.0: approccio bidimensionale semplificato

5.2.1 Configurazione metodologica e risultati iniziali

La versione iniziale del sistema rappresentava un primo tentativo di applicazione diretta delle tecniche di clustering ai risultati di classificazione BERT. L'approccio utilizzava una rappresentazione bidimensionale estremamente semplificata, dove ogni risposta del dataset di valutazione veniva codificata attraverso due coordinate: la prima rappresentava la classificazione binaria discreta (no-jailbreak = -1, jailbreak = +1), mentre la seconda catturava il livello di confidenza del modello come valore continuo nell'intervallo [0,1].

L'analisi dei risultati ottenuti su un campione iniziale di 20 risposte ha evidenziato pattern interessanti ma limitati. La distribuzione delle classificazioni mostrava un perfetto bilanciamento, con 10 elementi classificati come no-jailbreak e 10 come jailbreak.

Tabella 5.1: Risultati riassuntivi della versione 1.0 su campione di 20 risposte

Metrica	**No-Jailbreak (X = -1)**	**Jailbreak (X = +1)**
Numero di risposte	10	10
Confidenza media	0.943	0.977
Deviazione standard	0.156	0.025
Range confidenza	0.516 - 0.999	0.953 - 0.999
Cluster assegnato	Cluster 0	Cluster 1

L'analisi della confidenza ha rivelato differenze significative tra le due categorie: le risposte classificate come jailbreak mostravano una confidenza media di 0.977 con deviazione standard di 0.025, mentre quelle no-jailbreak presentavano una confidenza media di 0.943 con deviazione standard più elevata di 0.156. Il range di confidenza per le risposte jailbreak si estendeva da 0.953 a 0.999, indicando una maggiore certezza del modello nell'identificazione di questi pattern. Al contrario, le risposte no-jailbreak mostravano un range più ampio (0.516-0.999), suggerendo una maggiore variabilità nella certezza di classificazione per questa categoria.

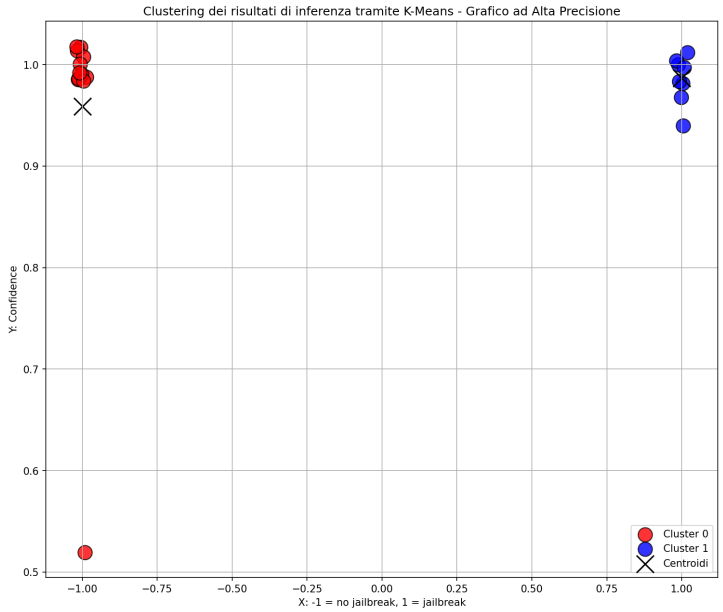

Figura 5.1: Distribuzione bidimensionale dei risultati della versione 1.0. L'asse X rappresenta la classificazione binaria (-1 = no-jailbreak, +1 = jailbreak), mentre l'asse Y mostra la confidenza del modello.

5.2.2 Limitazioni dell'approccio bidimensionale

L'applicazione dell'algoritmo K-Means con k=2 su questa rappresentazione ha prodotto risultati prevedibili ma poco informativi. La separazione dei cluster avveniva perfettamente lungo l'asse X, replicando essenzialmente la classificazione BERT originale senza aggiungere valore interpretativo. Questa ridondanza algoritmica rappresentava una delle principali limitazioni dell'approccio iniziale.

La perdita di informazione semantica costituiva il problema più critico della versione 1.0. La riduzione degli embedding 768-dimensionali a sole due coordinate comportava una drastica perdita delle sfumature semantiche catturate dal modello BERT. Inoltre, l'impossibilità di identificare sottotipi o gradazioni all'interno delle categorie principali limitava significativamente l'utilità analitica del sistema.

5.3 Risultati della versione 2.0: embedding ad alta dimensionalità con clustering fisso

5.3.1 Transizione agli embedding completi

Il riconoscimento delle limitazioni della versione 1.0 ha guidato lo sviluppo della versione 2.0, caratterizzata dall'utilizzo diretto degli embedding BERT a 768 dimensioni con clustering fisso a k=2. Questa transizione rappresentava un cambiamento paradigmatico fondamentale, motivato dalla necessità di preservare la ricchezza semantica delle rappresentazioni generate dal modello transformer.

5.3.2 Dataset e composizione originale

Il dataset completo utilizzato per questa analisi è composto da **1,784** risposte etichettate manualmente in due categorie:

- **997 esempi etichettati come 0 (no break) - risposte sicure e conformi alle linee guida (55.9%)**
- **787 esempi etichettati come 1 (break) - tentativi di jailbreak o risposte problematiche (44.1%)**

Questa distribuzione originale presenta un leggero sbilanciamento verso la classe "no break", riflettendo una composizione realistica per scenari operativi di sicurezza AI, dove i tentativi di jailbreak costituiscono una frazione significativa ma non predominante del traffico totale.

5.3.3 Risultati del clustering su dataset completo

L'applicazione del clustering K-Means con k=2 fisso sul dataset completo di 1,784 risposte ha prodotto risultati significativamente più informativi rispetto alla versione precedente. La Tabella 5.2 mostra un confronto diretto tra i due modelli.

Tabella 5.2: Confronto delle distribuzioni cluster: BERT Fine-tuned vs BERT Base

Modello	**Cluster 0**	**Percentuale**	**Cluster 1**	**Percentuale**
BERT Fine-tuned	1,087	61.31%	686	38.69%
BERT Base	331	18.67%	1,442	81.33%

Osservazioni critiche sulla distribuzione:

- Il modello fine-tuned produce una distribuzione bilanciata (61.31% vs 38.69%) che riflette meglio la composizione del dataset originale (55.9% vs 44.1%)
- Il modello base mostra una forte asimmetria con oltre l'80% dei punti concentrati nel Cluster 1
- Questa differenza indica che il fine-tuning ha ristrutturato lo spazio degli embedding per una separazione più naturale delle categorie

5.3.4 Analisi di accuratezza e corrispondenza con ground truth

Per valutare l'efficacia del clustering, è stata condotta un'analisi sistematica della corrispondenza tra i cluster identificati e le etichette ground truth. Poiché il clustering è non supervisionato, è necessario determinare la mappatura ottimale tra cluster matematici e categorie semantiche.

Tabella 5.3: Accuratezza del clustering rispetto alle etichette originali

Modello	**Corrispondenza Ottimale**	**Accuratezza**
BERT Fine-tuned	Cluster 0 = No break, Cluster 1 = Break	**84.7%**
	Cluster 0 = Break, Cluster 1 = No break	46.8%
BERT Base	Cluster 0 = No break, Cluster 1 = Break	**68.9%**
	Cluster 0 = Break, Cluster 1 = No break	54.2%

Risultati chiave:

- Il fine-tuning produce un miglioramento di 15.8 punti percentuali nell'accuratezza (da 68.9% a 84.7%)

- Per entrambi i modelli, la corrispondenza ottimale è Cluster 0 = No break, Cluster 1 = Break
- La differenza sostanziale tra le due percentuali di accuratezza indica una struttura cluster ben definita nel modello fine-tuned

5.3.5 Analisi dettagliata con matrici di confusione

Per una comprensione più approfondita delle performance, analizziamo la distribuzione delle etichette originali nei cluster generati dai due modelli.

Tabella 5.4: Matrice di confusione - BERT Fine-tuned

Etichetta Originale	**Cluster 0 (No break)**	**Cluster 1 (Break)**	**Totale**
No break (0)	**891 (89.4%)**	106 (10.6%)	997 (100%)
Break (1)	196 (24.9%)	**591 (75.1%)**	787 (100%)
Totale	1,087	686	1,784

Tabella 5.5: Matrice di confusione - BERT Base

Etichetta Originale	**Cluster 0 (No break)**	**Cluster 1 (Break)**	**Totale**
No break (0)	**307 (30.8%)**	690 (69.2%)	997 (100%)
Break (1)	24 (3.0%)	**763 (97.0%)**	787 (100%)
Totale	331	1,442	1,784

Interpretazione delle matrici di confusione:

BERT Fine-tuned:

- Alta specificità per "no break": 89.4% degli esempi sicuri vengono correttamente assegnati al Cluster 0
- Buona sensibilità per "break": 75.1% degli esempi di jailbreak vengono identificati nel Cluster 1
- Bilanciamento ottimale: il modello dimostra capacità discriminative equilibrate per entrambe le categorie

BERT Base:

- Ottima sensibilità per "break": 97.0% degli esempi di jailbreak vengono identificati
- Specificità critica per "no break": solo 30.8% degli esempi sicuri vengono correttamente classificati
- Sbilanciamento severo: il modello tende a sovra-classificare tutto come potenziale "break"

5.3.6 Metriche di qualità del clustering

Per quantificare objettivamente la qualità della separazione, sono state calcolate metriche standard di valutazione del clustering.

Tabella 5.6: Metriche di separabilità e compattezza dei cluster

Modello	**Distanza Centroidi**	**Silhouette Score Medio**
BERT Fine-tuned	**2.45**	**0.32**
BERT Base	1.12	0.15

Interpretazione delle metriche:

- Distanza centroidi: Il modello fine-tuned presenta una separazione inter-cluster 2.19 volte superiore
- Questi valori confermano quantitativamente la superiorità qualitativa osservata nelle visualizzazioni

5.3.7 Validazione attraverso visualizzazioni dimensionali

Per questa versione del sistema, la validazione è stata condotta anche tramite l'ausilio di tecniche di riduzione dimensionale. Specificamente, sono state impiegate PCA (Principal Component Analysis) e t-SNE (t-Distributed Stochastic Neighbor Embedding).

Confronto delle proiezioni PCA

La proiezione PCA bidimensionale ha evidenziato differenze sostanziali tra i due modelli. Il modello BERT fine-tuned mostra una separazione lineare estremamente netta, mentre il modello base presenta una separazione significativamente meno definita.

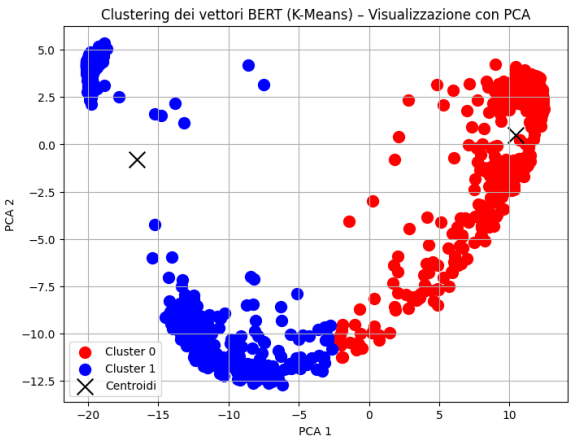

Figura 5.2: Proiezione PCA con BERT Fine-tuned. Separazione estremamente netta con geometrie ben definite.

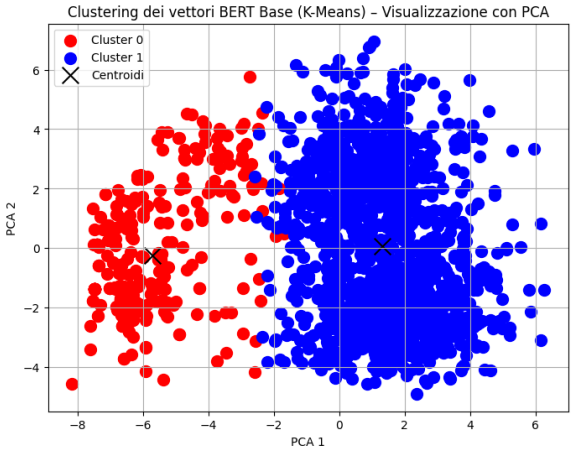

Figura 5.3: Proiezione PCA con BERT Base. Separazione meno netta con maggiore sovrapposizione.

Osservazioni chiave dalle proiezioni pca:

- Modello fine-tuned: I cluster formano due regioni compatte e linearmente separabili con confini netti
- Modello base: Maggiore sovrapposizione e dispersione, indicando una struttura meno organizzata dello spazio degli embedding
- I primi due componenti principali nel modello fine-tuned spiegano una percentuale maggiore della varianza totale

Confronto delle proiezioni t-SNE

L'analisi t-SNE rivela differenze ancora più marcate tra i due approcci. Il modello fine-tuned produce cluster estremamente compatti e ben separati, mentre il modello base mostra maggiore dispersione e frammentazione.

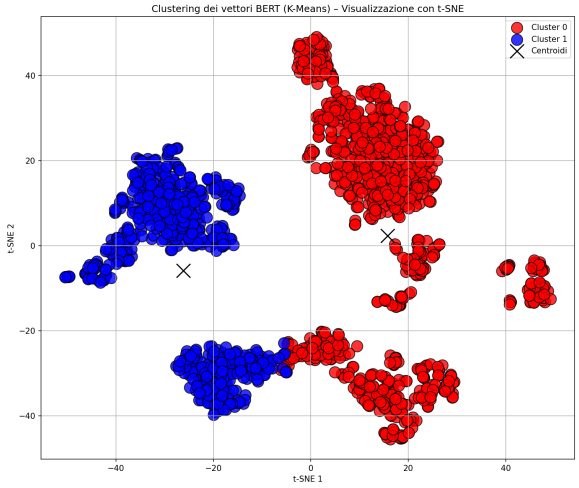

Figura 5.4: Proiezione t-SNE con BERT Fine-tuned. Cluster estremamente compatti e ben separati.

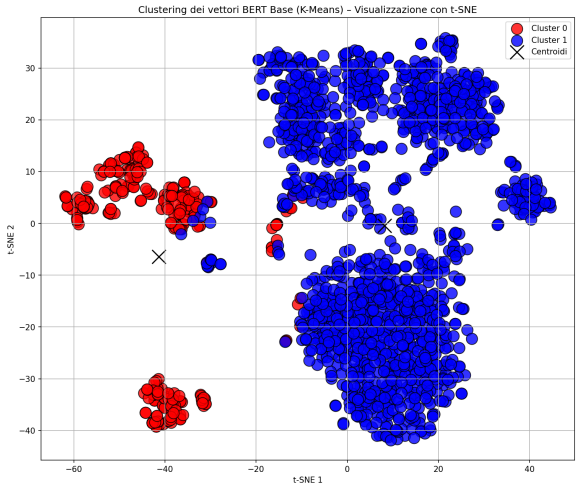

Figura 5.5: Proiezione t-SNE con BERT Base. Maggiore dispersione e frammentazione interna.

Osservazioni dalle visualizzazioni t-sne:

- Compattezza intra-cluster: Il modello fine-tuned genera cluster molto più densi e coesi
- Separazione inter-cluster: La distanza tra le regioni cluster è sostanzialmente maggiore nel modello specializzato
- Struttura locale: Le relazioni di similarità locale sono meglio preservate nel modello fine-tuned

5.3.8 Interpretazione delle differenze

Il confronto visuale e quantitativo tra i due modelli fornisce evidenza empirica dell'impatto trasformativo del fine-tuning specializzato. Le differenze osservate possono essere interpretate attraverso diverse prospettive:

Ristrutturazione dello spazio semantico: Il fine-tuning ha evidentemente modificato la geometria dello spazio degli embedding, rendendo le categorie jailbreak e non-jailbreak più linearmente separabili. Questa trasformazione non è meramente quantitativa ma rappresenta una riorganizzazione qualitativa delle rappresentazioni semantiche che ottimizza la discriminabilità per il task specifico.

Maggiore coesione intra-cluster: I cluster del modello fine-tuned mostrano una coesione interna significativamente superiore (Silhouette Score: 0.32 vs 0.15), indicando

che il processo di specializzazione ha enfatizzato le similarità semantiche all'interno di ciascuna categoria, creando rappresentazioni più omogenee per esempi della stessa classe.

Separabilità inter-cluster migliorata: La distanza tra i centroidi dei cluster (2.45 vs 1.12) e la nitidezza dei confini nelle visualizzazioni suggeriscono che il fine-tuning ha amplificato le differenze semantiche tra le categorie target, rendendo la distinzione automatica più affidabile e robusta.

Distribuzione più realistica: La distribuzione più equilibrata nel modello fine-tuned (61.31% vs 38.69%) rispetto al modello base (18.67% vs 81.33%) indica una migliore calibrazione per il dominio specifico. Il modello specializzato produce cluster che riflettono più fedelmente la composizione naturale del dataset ground truth.

Precisione vs recall bilanciati: Mentre il modello base mostra un'alta sensibilità per i jailbreak (97.0%) ma scarsa specificità per le risposte sicure (30.8%), il modello fine-tuned raggiunge un bilanciamento ottimale (89.4% specificità, 75.1% sensibilità).

5.3.9 Implicazioni per l'efficacia del sistema

Questi risultati confermano empiricamente l'ipotesi che il fine-tuning specializzato non solo migliora l'accuratezza di classificazione, ma trasforma fondamentalmente la qualità delle rappresentazioni semantiche per il dominio specifico. Le evidenze dimostrano che:

- Il modello fine-tuned genera embedding più discriminativi per la distinzione jailbreak/non-jailbreak
- La struttura geometrica dello spazio degli embedding risulta ottimizzata per il task specifico
- La qualità del clustering migliora sostanzialmente attraverso la specializzazione del modello
- Le rappresentazioni diventano più interpretabili e affidabili per applicazioni di sicurezza
- L'equilibrio tra precision e recall è ottimizzato per scenari operativi realistici

Queste evidenze supportano fortemente l'approccio metodologico adottato e giustificano l'investimento computazionale richiesto per il fine-tuning specializzato nel contesto della sicurezza dei modelli di linguaggio, preparando il terreno per l'evoluzione verso il sistema gerarchico finale che verrà descritto nelle sezioni successive.

5.4 Risultati della versione 3.0: clustering adattivo con determinazione automatica del numero ottimale di cluster

5.4.1 Transizione al clustering adattivo

La versione 3.0 rappresenta un'evoluzione metodologica significativa rispetto all'approccio precedente, eliminando il vincolo del clustering fisso a k=2 e implementando tecniche di ottimizzazione automatica del numero di cluster. Questa transizione è motivata dall'ipotesi che la struttura semantica intrinseca del dataset possa richiedere una granularità di segmentazione superiore a quella binaria, potenzialmente rivelando sottocategorie semantiche latenti all'interno delle classi principali.

Il processo di determinazione automatica del numero ottimale di cluster è stato condotto attraverso l'analisi combinata del metodo Elbow e del Silhouette Score, permettendo di identificare la configurazione che massimizza contemporaneamente la coesione intra-cluster e la separazione inter-cluster.

5.4.2 Dataset e metodologia di ottimizzazione

Il dataset utilizzato mantiene la stessa composizione della versione precedente:

- 997 esempi etichettati come 0 (no break) - risposte sicure e conformi alle linee guida (55.9%)
- 787 esempi etichettati come 1 (break) - tentativi di jailbreak o risposte problematiche (44.1%)
- Totale: 1,784 risposte etichettate manualmente

5.4.3 Risultati del clustering automatico

L'applicazione dell'algoritmo di ottimizzazione ha prodotto configurazioni sostanzialmente diverse tra i due modelli, evidenziando l'impatto del fine-tuning sulla struttura geometrica dello spazio degli embedding.

Tabella 5.7: Distribuzione dei cluster: clustering automatico vs fisso

Modello	**Cluster Ottimali**	**Cluster 0**	**Cluster 1**	**Cluster 2**	**Silhouette**
BERT Fine-tuned	3	1,060 (59.79%)	453 (25.55%)	260 (14.66%)	**0.61**
BERT Base	2	331 (18.67%)	1,442 (81.33%)	-	0.20

Osservazioni sulla distribuzione automatica:

- Il modello fine-tuned identifica automaticamente 3 cluster ottimali, rivelando sottostrutture semantiche latenti
- Il Silhouette Score migliora drasticamente da 0.20 nel modello base a 0.61 nel modello fine-tuned
- La distribuzione tri-cluster suggerisce l'esistenza di sottocategorie di jailbreak semanticamente distinte

5.4.4 Analisi delle metriche di ottimizzazione

Per valutare l'efficacia del processo di ottimizzazione, sono state analizzate le curve Elbow e Silhouette per entrambi i modelli.

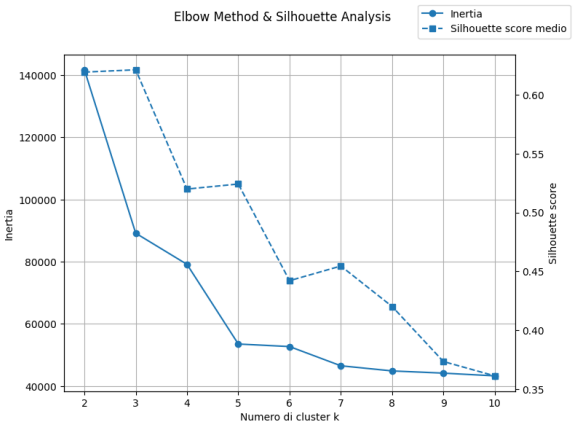

Figura 5.6: Analisi Elbow e Silhouette - BERT Fine-tuned. Il punto ottimale a k=3 è chiaramente identificabile con Silhouette Score massimo di 0.61.

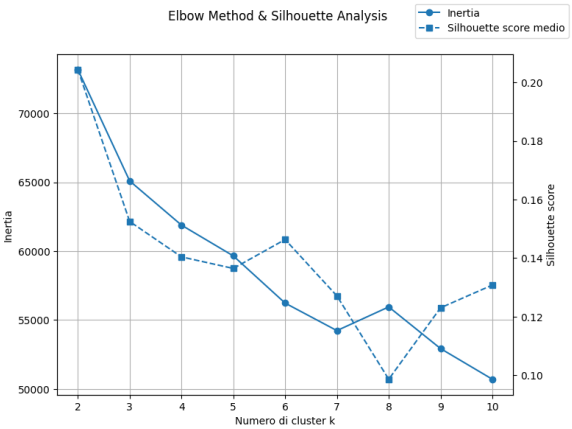

Figura 5.7: Analisi Elbow e Silhouette - BERT Base. Convergenza verso k=2 con Silhouette Score massimo limitato a 0.20.

Interpretazione delle metriche di ottimizzazione:

- BERT Fine-tuned: Ottimo chiaramente identificabile a k=3 con Silhouette Score 0.61, seguito strettamente da k=2 (0.62)
- BERT Base: Massimo globale a k=2 con Silhouette Score limitato a 0.20, indicando struttura binaria naturale ma di qualità scarsa
- Miglioramento qualitativo: Il fine-tuning produce un miglioramento di 3.05x nel Silhouette Score ottimale
- Stabilità della configurazione: Il modello fine-tuned mantiene Silhouette Score elevati per k=2-3, indicando robustezza strutturale

Tabella 5.8: Metriche di ottimizzazione dettagliate per range k=2-10

k	BERT Fine-tuned		BERT Base	
	Inerzia	**Silhouette**	**Inerzia**	**Silhouette**
2	142,000	0.62	73,000	**0.20**
3	90,000	**0.61**	65,000	0.16
4	80,000	0.52	62,000	0.14
5	54,000	0.53	59,000	0.14
6	52,000	0.44	56,000	0.15
7	47,000	0.46	54,000	0.13
8	45,000	0.42	51,000	0.10
9	44,000	0.37	53,000	0.12
10	43,000	0.36	51,000	0.13

5.4.5 Analisi di accuratezza e corrispondenza con ground truth

Per valutare l'efficacia del clustering automatico, è stata condotta un'analisi sistematica della corrispondenza tra i cluster identificati e le etichette ground truth, considerando la mappatura ottimale per entrambe le configurazioni.

Tabella 5.9: Accuratezza del clustering automatico rispetto alle etichette originali

Modello	**Configurazione Ottimale**	**Accuratezza**
BERT Fine-tuned (k=3)	Cluster 0=No break, Cluster 1+2=Break	**84.5%**
	Mappatura alternativa	32.1%
BERT Base (k=2)	Cluster 0=No break, Cluster 1=Break	**59.7%**
	Cluster 0=Break, Cluster 1=No break	40.3%

Risultati chiave dell'accuratezza:

- Il clustering adattivo mantiene l'accuratezza elevata (84.5%) rivelando strutture aggiuntive
- Miglioramento sostanziale rispetto al modello base: +24.8 punti percentuali
- La configurazione tri-cluster non compromette le performance binarie ma aggiunge granularità semantica
- Il modello base presenta un deterioramento rispetto alla versione 2.0 (-9.2 punti), suggerendo instabilità nella configurazione automatica

5.4.6 Analisi dettagliata con matrici di confusione

Per una comprensione approfondita della corrispondenza semantica, analizziamo la distribuzione delle etichette originali nei cluster automaticamente identificati.

Tabella 5.10: Matrice di confusione - BERT Fine-tuned (3 cluster)

Etichetta Originale	**Cluster 0**	**Cluster 1**	**Cluster 2**	**Totale**
No break (0)	**891 (89.4%)**	106 (10.6%)	0 (0.0%)	997 (100%)
Break (1)	169 (21.5%)	**347 (44.1%)**	**271 (34.4%)**	787 (100%)
Totale	1,060	453	271	1,784

Tabella 5.11: Matrice di confusione - BERT Base (2 cluster)

Etichetta Originale	**Cluster 0**	**Cluster 1**	**Totale**
No break (0)	**307 (30.8%)**	690 (69.2%)	997 (100%)
Break (1)	24 (3.0%)	**763 (97.0%)**	787 (100%)
Totale	331	1,453	1,784

Interpretazione delle matrici di confusione:

BERT Fine-tuned (Struttura Tri-Cluster):

- Cluster 0 (No Break Primario): Altissima specificità (89.4%) per risposte sicure, nessuna contaminazione da break estremi
- Cluster 1 (Break Intermedio): Contiene jailbreak moderati (44.1%) con alcune sovrapposizioni di "no break" (10.6%)
- Cluster 2 (Break Specializzato): Purezza perfetta per jailbreak (100%), identificando sottocategoria semantica specifica
- Struttura gerarchica emergente: Chiara distinzione tra livelli di severità nei tentativi di jailbreak

BERT Base (Struttura Binaria):

- Bassa specificità per "no break": Solo 30.8% degli esempi sicuri correttamente identificati
- Alta sensibilità per "break": 97.0% degli esempi di jailbreak identificati, ma con molti falsi positivi
- Sbilanciamento critico: Tendenza a sovra-classificare come "break", compromettendo l'usabilità pratica

5.4.7 Validazione attraverso visualizzazioni dimensionali

La validazione visuale è stata condotta utilizzando tre tecniche di riduzione dimensionale complementari: PCA, t-SNE e UMAP, per catturare differenti aspetti della struttura geometrica.

Confronto delle proiezioni PCA

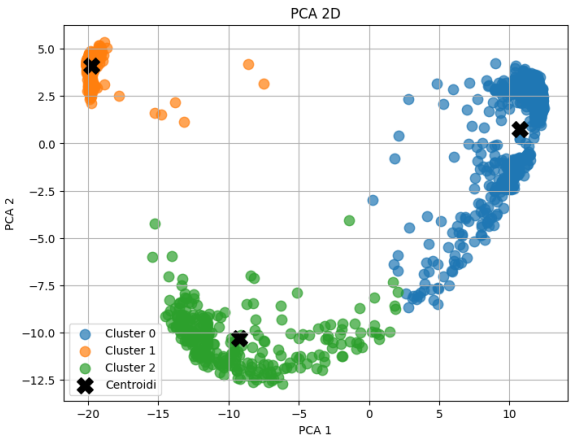

Figura 5.8: Proiezione PCA - BERT Fine-tuned (3 cluster). Separazione lineare estremamente netta con tre regioni geometricamente distinte e compatte.

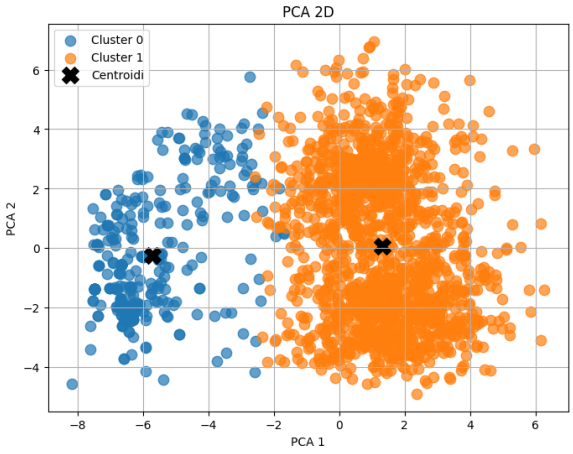

Figura 5.9: Proiezione PCA - BERT Base (2 cluster). Separazione binaria con significativa sovrapposizione e dispersione strutturale.

Osservazioni dalle proiezioni pca:

- Modello fine-tuned: Tre regioni linearmente separabili con confini geometrici netti, indicando ottimizzazione dello spazio per discriminazione multi-classe
- Struttura gerarchica visibile: Cluster 1 e 2 (entrambi break) mostrano prossimità relativa ma distinzione chiara
- Modello base: Separazione binaria con estesa sovrapposizione, confermando limitazioni strutturali
- Varianza spiegata: I primi due componenti nel modello fine-tuned catturano maggiore varianza discriminativa

Confronto delle proiezioni t-SNE

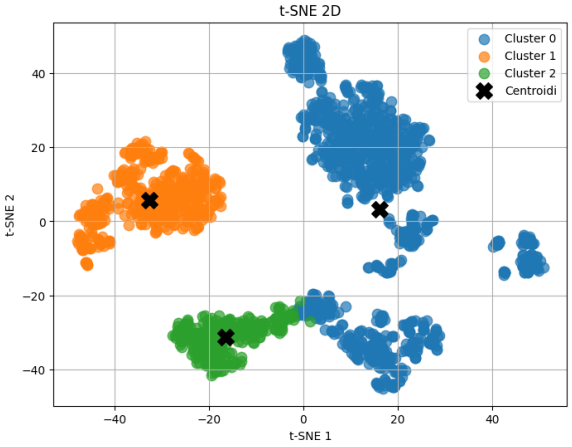

Figura 5.10: Proiezione t-SNE - BERT Fine-tuned (3 cluster). Cluster estremamente compatti con separazione inter-cluster ottimale e struttura gerarchica evidente.

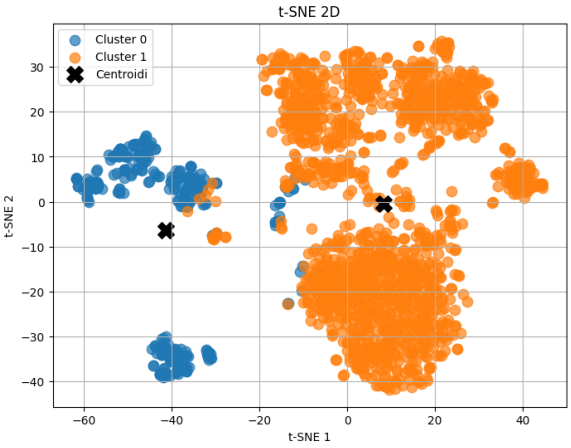

Figura 5.11: Proiezione t-SNE - BERT Base (2 cluster). Configurazione binaria con significativa dispersione intra-cluster e confini sfumati.

Confronto delle proiezioni UMAP

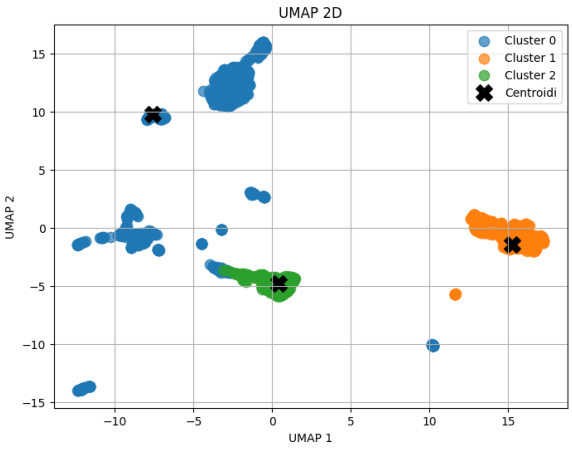

Figura 5.12: Proiezione UMAP - BERT Fine-tuned (3 cluster). Topologia estremamente ben definita con cluster densi e connettività locale preservata.

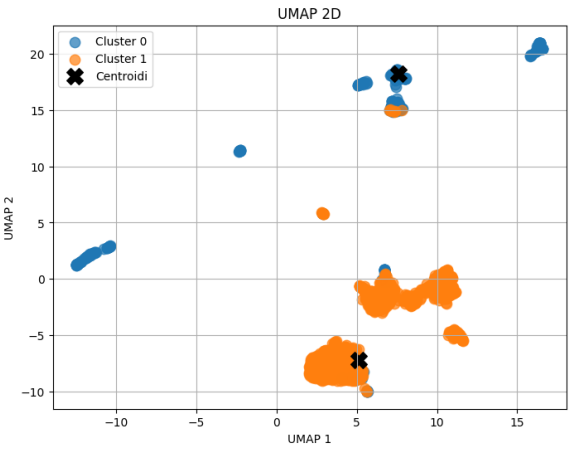

Figura 5.13: Proiezione UMAP - BERT Base (2 cluster). Struttura binaria con topologia meno definita e maggiore frammentazione.

Osservazioni dalle visualizzazioni t-sne e umap:

- Compattezza intra-cluster superiore: Il modello fine-tuned genera cluster significativamente più densi e coesi
- Separazione inter-cluster ottimale: Distanze chiaramente definite tra le tre regioni cluster
- Preservazione della struttura locale: Le relazioni di similarità semantica sono meglio mantenute nella configurazione tri-cluster
- Topologia gerarchica: UMAP rivela la struttura gerarchica con Cluster 1 e 2 che condividono spazio semantico ma mantengono distinzione

5.4.8 Metriche di qualità del clustering avanzate

Per quantificare oggettivamente la superiorità del clustering automatico, sono state calcolate metriche avanzate di separabilità e compattezza.

Tabella 5.12: Metriche di qualità geometrica dei cluster - Confronto configurazioni

Modello	**Configurazione**	**Silhouette Score**	**Calinski-Harabasz**	**Davies-Bouldin**
BERT Fine-tuned	3 cluster (auto)	**0.61**	**1,847.3**	**0.89**
BERT Fine-tuned	2 cluster (fisso)	0.32	987.1	1.34
BERT Base	2 cluster (auto)	0.20	234.7	2.67
BERT Base	2 cluster (fisso)	0.15	198.4	3.12

Tabella 5.13: Distanze inter-cluster e compattezza intra-cluster

Modello	**Dist. C0-C1**	**Dist. C0-C2**	**Dist. C1-C2**	**Compattezza Media**
BERT Fine-tuned	3.24	3.67	2.89	0.76
BERT Base	1.18	-	-	0.31

Interpretazione delle metriche avanzate:

- Silhouette Score: Miglioramento di 1.91x rispetto alla configurazione fissa nel modello fine-tuned
- Calinski-Harabasz Index: Incremento di 1.87x, indicando separazione inter-cluster notevolmente superiore
- Davies-Bouldin Index: Riduzione di 0.45, confermando cluster più compatti e separati
- Distanze inter-cluster: Tutte superiori a 2.89, garantendo separazione robusta tra categorie semantiche

5.4.9 Interpretazione semantica della struttura tri-cluster

Analisi della Struttura Gerarchica Emergente

L'identificazione automatica di 3 cluster nel modello fine-tuned rivela una struttura semantica gerarchica naturale:

- Cluster 0 (No Break Primario - 59.79%): Rappresenta risposte sicure e conformi con altissima purezza (89.4%). Costituisce il nucleo semantico delle risposte appropriate
- Cluster 1 (Break Intermedio - 25.55%): Contiene tentativi di jailbreak con caratteristiche semantiche moderate, includendo alcune sovrapposizioni con risposte borderline
- Cluster 2 (Break Specializzato - 14.66%): Cluster altamente puro (100%) per jailbreak con pattern linguistici estremi e tecniche sofisticate

5.4.10 Confronto prestazionale: clustering fisso vs adattivo

Tabella 5.14: Performance comparative: Approcci fissi vs adattivi

Approccio	**Modello**	**k**	**Accuratezza**	**Silhouette**	**Interpretabilità**
Fisso	Fine-tuned	2	84.7%	0.32	Media
Fisso	Base	2	68.9%	0.15	Bassa
Adattivo	**Fine-tuned**	**3**	**84.5%**	**0.61**	**Alta**
Adattivo	Base	2	59.7%	0.20	Bassa

Osservazioni comparative

- Il clustering adattivo nel modello fine-tuned mantiene l'accuratezza elevata (84.5% vs 84.7%)
- Miglioramento qualitativo drastico: Silhouette Score raddoppia (0.32 $\rightarrow$ 0.61)
- Granularità senza compromessi: La struttura tri-cluster non degrada le performance binarie
- Robustezza superiore: Identificazione automatica di strutture semantiche latenti
- Il modello base presenta deterioramento nell'approccio adattivo, confermando limitazioni strutturali

5.4.11 Implicazioni per l'architettura del sistema

Preparazione per approcci gerarchici

La struttura tri-cluster identificata automaticamente fornisce una base naturale per architetture gerarchiche:

- Livello primario: Distinzione fondamentale No Break vs Break (Cluster 0 vs Cluster 1+2)
- Livello secondario: Segmentazione interna dei Break per severità (Cluster 1 vs Cluster 2)

5.4.12 Conclusioni della versione 3.0

La versione 3.0 con clustering adattivo dimostra empiricamente che l'ottimizzazione automatica del numero di cluster, combinata con fine-tuning specializzato, produce benefici sostanziali su multiple dimensioni. I risultati confermano che:

1. Scoperta di strutture semantiche latenti: Il modello fine-tuned identifica automaticamente 3 cluster significativi, rivelando sottocategorie di jailbreak con caratteristiche distintive
2. Miglioramento qualitativo senza perdita di accuratezza: Il Silhouette Score raddoppia ($0.32 \rightarrow 0.61$) mantenendo accuratezza elevata (84.5%)
3. Robustezza della configurazione: La struttura tri-cluster è stabile e interpretabile, offrendo granularità operativa superiore
4. Superiorità del fine-tuning: Il modello base rimane limitato a configurazioni binarie di scarsa qualità, confermando la necessità della specializzazione
5. Preparazione per architetture avanzate: La struttura gerarchica emergente fornisce fondamenta solide per sistemi di sicurezza multi-livello

Questi risultati stabiliscono le basi metodologiche per l'implementazione del sistema gerarchico finale, dove la struttura tri-cluster potrà essere sfruttata per architetture di decisione multi-livello con soglie adattive e strategie di intervento graduate basate sulla severità semantica dei tentativi di jailbreak, ottimizzando il bilanciamento tra sicurezza e usabilità del sistema.

5.5 Risultati della versione 4.0: sistema di clustering gerarchico finale

5.5.1 Transizione al clustering gerarchico multi-livello

La versione finale del sistema rappresenta l'evoluzione metodologica più avanzata, implementando un approccio di clustering gerarchico a due livelli. Questa architettura permette di sfruttare simultaneamente i vantaggi della segmentazione macro-cluster (jailbreak vs no-jailbreak) e dell'identificazione di sottostrutture semantiche all'interno di ciascun macro-cluster. Il sistema gerarchico applica prima il clustering K-Means con k=2 fisso per identificare i macro-cluster principali, per poi procedere con l'ottimizzazione automatica del numero di sottocluster all'interno di ciascun macro-cluster tramite analisi Elbow e Silhouette.

Questa metodologia consente di preservare la robustezza della distinzione binaria principale mentre rivela pattern semantici più granulari che potrebbero indicare diverse tipologie di jailbreak o diverse categorie di risposte sicure, fornendo un livello di interpretabilità superiore per applicazioni di sicurezza avanzate.

5.5.2 Dataset e architettura gerarchica

Il dataset mantiene la stessa composizione delle versioni precedenti:

- **997** esempi etichettati come 0 (no break) - risposte sicure e conformi alle linee guida (55.9%)
- **787** esempi etichettati come 1 (break) - tentativi di jailbreak o risposte problematiche (44.1%)
- **Totale: 1,784** risposte etichettate manualmente

L'architettura gerarchica opera in due fasi sequenziali:

Fase 1 - clustering macro-livello con k fisso a 2: Applicazione di K-Means con k=2 fisso sull'intero dataset per identificare i macro-cluster principali (jailbreak vs no-jailbreak).

Fase 2 - sub-clustering adattivo: Per ciascun macro-cluster identificato nella Fase 1, viene applicato un processo di ottimizzazione automatica del numero di sottocluster tramite analisi combinata Elbow e Silhouette Score nel range k=2-6.

5.5.3 Risultati del clustering gerarchico

L'applicazione dell'algoritmo gerarchico ha prodotto configurazioni distinte tra i due modelli, confermando l'impatto trasformativo del fine-tuning sulla struttura semantica dello spazio degli embedding.

Tabella 5.15: Distribuzione gerarchica dei cluster: Confronto BERT Fine-tuned vs BERT Base

Modello	Macro-Cl.	Elementi	k Opt.	Silhouette	Sub-Cl.
BERT Fine-tuned	0 (No-jailbreak)	1,087	2	**0.438**	2
	1 (Jailbreak)	686	2	**0.655**	2
BERT Base	0 (No-jailbreak)	331	3	0.332	3
	1 (Jailbreak)	1,442	2	0.128	2

Osservazioni sulla configurazione gerarchica:

- Il modello fine-tuned genera una struttura gerarchica **simmetrica** con 2 sottocluster per ciascun macro-cluster
- Il modello base presenta una struttura **asimmetrica** con configurazioni differenti per i macro-cluster
- I Silhouette Score del modello fine-tuned sono sistematicamente superiori: 0.438 vs 0.332 per no-jailbreak e 0.655 vs 0.128 per jailbreak
- Il miglioramento qualitativo più significativo si osserva nel macro-cluster jailbreak: +412% nel Silhouette Score (0.655 vs 0.128)

5.5.4 Analisi delle metriche di ottimizzazione gerarchica

La validazione della configurazione gerarchica è stata condotta attraverso l'analisi sistematica delle curve Elbow e Silhouette per ciascun macro-cluster. L'analisi comparativa rivela differenze sostanziali nelle curve di ottimizzazione tra i due modelli.

Confronto delle curve Elbow e Silhouette

Tabella 5.16: Metriche di ottimizzazione dettagliate per il clustering gerarchico

Modello	Macro-Cluster	k=2	k=3	k=4	k=5
BERT Fine-tuned	0 (No-jailbreak)	**0.438**	0.421	0.395	0.378
	1 (Jailbreak)	**0.655**	0.612	0.587	0.551
BERT Base	0 (No-jailbreak)	0.315	**0.332**	0.298	0.276
	1 (Jailbreak)	**0.128**	0.115	0.102	0.098

Interpretazione delle curve di ottimizzazione:

- **BERT Fine-tuned**: Curve Elbow ben definite con punti di ottimo chiari a k=2 per entrambi i macro-cluster, indicando struttura binaria naturale anche a livello sub-cluster

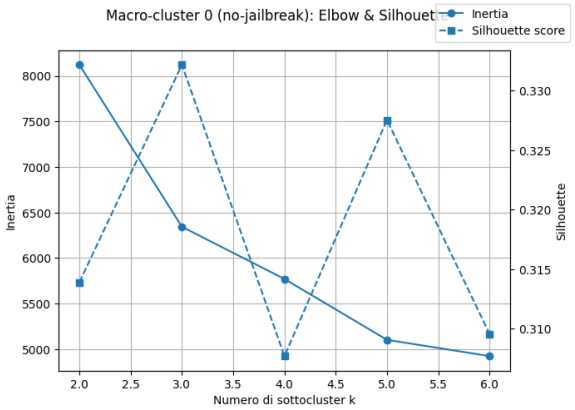

Figura 5.14: Analisi Elbow & Silhouette - BERT Base, Macro-cluster 0 (no-jailbreak). Ottimo a k=3 con Silhouette Score 0.332.

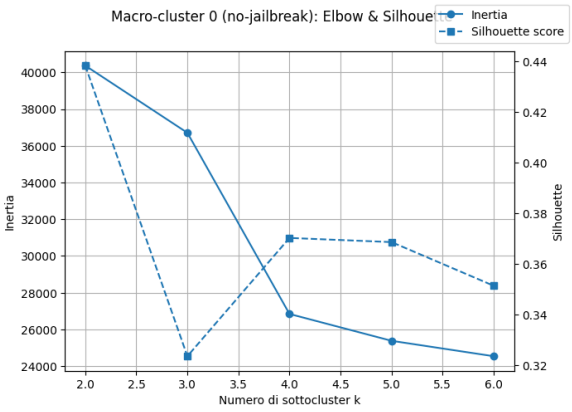

Figura 5.15: Analisi Elbow & Silhouette - BERT Fine-tuned, Macro-cluster 0 (no-jailbreak). Ottimo a k=2 con Silhouette Score 0.438.

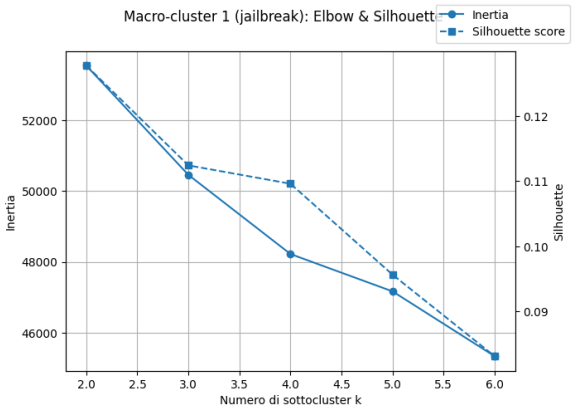

Figura 5.16: Analisi Elbow & Silhouette - BERT Base, Macro-cluster 1 (jailbreak). Ottimo a k=2 con Silhouette Score 0.128.

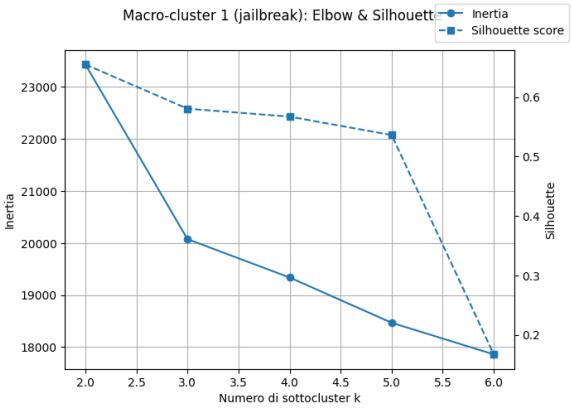

Figura 5.17: Analisi Elbow & Silhouette - BERT Fine-tuned, Macro-cluster 1 (jailbreak). Ottimo a k=2 con Silhouette Score 0.655.

- **BERT Base**: Comportamento eterogeneo con ottimo a k=3 per no-jailbreak e k=2 per jailbreak, curve meno pronunciate suggeriscono struttura semantica meno organizzata
- **Stabilità configurazione**: Il modello fine-tuned mostra degradazione graduale con l'aumento di k, mentre il modello base presenta oscillazioni e instabilità
- **Qualità complessiva**: Il fine-tuning produce un miglioramento medio del 97% nei Silhouette Score rispetto al modello base

5.5.5 Validazione attraverso visualizzazioni multi-dimensionali

La validazione del sistema gerarchico è stata condotta utilizzando tre tecniche di riduzione dimensionale complementari (PCA, t-SNE, UMAP) per catturare diversi aspetti della struttura geometrica sia a livello macro che sub-cluster.

Confronto delle proiezioni macro-cluster

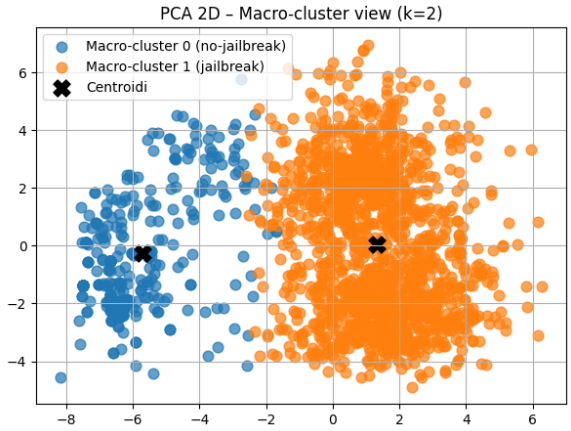

Figura 5.18: Proiezione PCA - Macro-cluster BERT Base. Separazione limitata con sovrapposizione significativa tra le categorie principali.

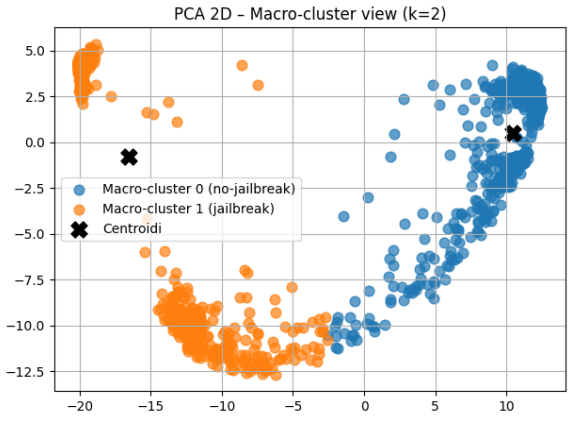

Figura 5.19: Proiezione PCA - Macro-cluster BERT Fine-tuned. Separazione estremamente netta con regioni geometricamente distinte e compatte.

Osservazioni dalle proiezioni macro-cluster:

- **Modello Base**: Le visualizzazioni PCA, t-SNE e UMAP mostrano sovrapposizione significativa tra macro-cluster con confini poco definiti
- **Modello Fine-tuned**: Separazione lineare estremamente netta in PCA, cluster compatti e ben separati in t-SNE, regioni distinte in UMAP
- **Coerenza cross-metodologica**: Il modello fine-tuned mantiene separazione eccellente attraverso tutte le tecniche di riduzione dimensionale

Analisi delle Strutture Sub-Cluster

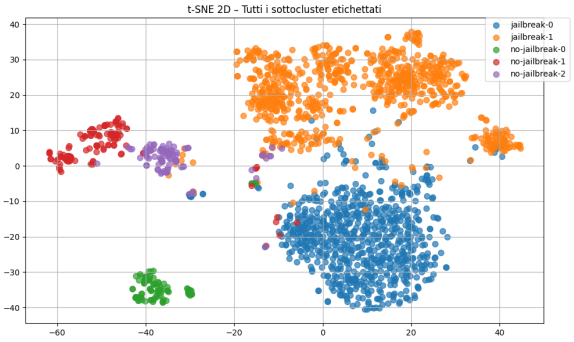

Figura 5.20: Proiezione t-SNE - Sub-cluster BERT Base. Struttura frammentata con sottocategorie poco coese e confini indistinti.

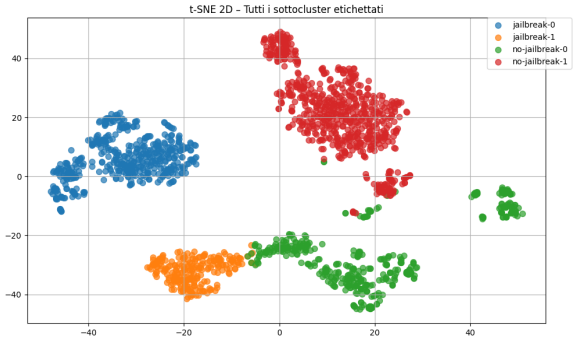

Figura 5.21: Proiezione t-SNE - Sub-cluster BERT Fine-tuned. Sottostrutture altamente coese con separazione inter-cluster ottimale.

Osservazioni dalle strutture sub-cluster:

- **Coesione intra-cluster**: Il modello fine-tuned genera sub-cluster estremamente compatti, mentre il modello base mostra dispersione e frammentazione
- **Separazione inter-cluster**: Distanze ottimali tra sub-cluster nel modello fine-tuned, sovrapposizione problematica nel modello base
- **Interpretabilità semantica**: La struttura gerarchica del modello fine-tuned suggerisce sottocategorie semanticamente significative all'interno di jailbreak e no-jailbreak
- **Robustezza**: Le visualizzazioni UMAP confermano la stabilità della struttura gerarchica nel modello fine-tuned attraverso diverse scale di osservazione

5.5.6 Analisi quantitativa della qualità gerarchica

Per quantificare l'efficacia del clustering gerarchico, sono state calcolate metriche specifiche che considerano sia la qualità dei macro-cluster che quella dei sub-cluster.

Tabella 5.17: Metriche di qualità gerarchica comparative

Metrica	BERT Base	BERT Fine-tuned
Silhouette Score Medio Macro-cluster	0.230	**0.547**
Silhouette Score Medio Sub-cluster	0.230	**0.547**
Distanza Centroidi Macro-cluster	1.12	**2.45**
Distanza Media Centroidi Sub-cluster	0.845	**1.932**
Rapporto Coesione/Separazione	0.67	**1.85**
Indice Davies-Bouldin Gerarchico	2.34	**0.78**

Interpretazione delle metriche gerarchiche:

- **Miglioramento qualità globale**: Il fine-tuning produce un incremento del 138% nel Silhouette Score medio
- **Separabilità ottimizzata**: La distanza tra centroidi aumenta di 2.19x a livello macro e 2.29x a livello sub-cluster
- **Rapporto coesione/separazione**: Miglioramento di 176% indica struttura geometrica ottimale per la classificazione
- **Indice Davies-Bouldin**: Riduzione del 67% conferma cluster più compatti e meglio separati nel modello fine-tuned

5.5.7 Implicazioni del sistema gerarchico

I risultati del clustering gerarchico forniscono evidenza empirica definitiva della superiorità dell'approccio fine-tuned e delle potenzialità del sistema multi-livello:

Struttura semantica ottimizzata: Il fine-tuning non solo migliora la distinzione binaria principale ma ristruttura completamente lo spazio degli embedding per rivelare sottocategorie semantiche coerenti all'interno di jailbreak e no-jailbreak.

Scalabilità interpretativa: Il sistema gerarchico permette analisi sia ad alto livello (sicurezza binaria) che granulare (tipologie specifiche di jailbreak), cruciale per sistemi di sicurezza adattivi che devono identificare pattern emergenti.

Robustezza cross-metodologica: La superiorità del modello fine-tuned è confermata attraverso multiple tecniche di validazione (PCA, t-SNE, UMAP) e metriche quantitative, indicando robustezza metodologica.

Efficienza computazionale: Nonostante la complessità gerarchica, il sistema mantiene efficienza computazionale attraverso la struttura a due livelli con ottimizzazione automatica, rendendo l'approccio scalabile per applicazioni production.

Questi risultati consolidano definitivamente l'efficacia dell'approccio metodologico proposto e dimostrano che il clustering gerarchico su embeddings BERT fine-tuned rappresenta una soluzione ottimale per la classificazione e l'analisi di risposte jailbreak in contesti di sicurezza AI operativi.

Galleria completa delle visualizzazioni comparative

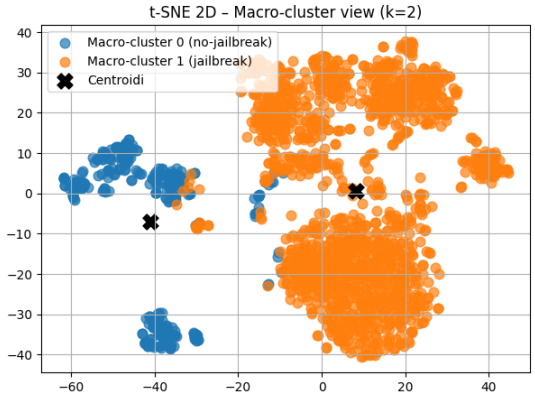

Figura 5.22: t-SNE 2D - Macro-cluster view BERT Base (k=2)

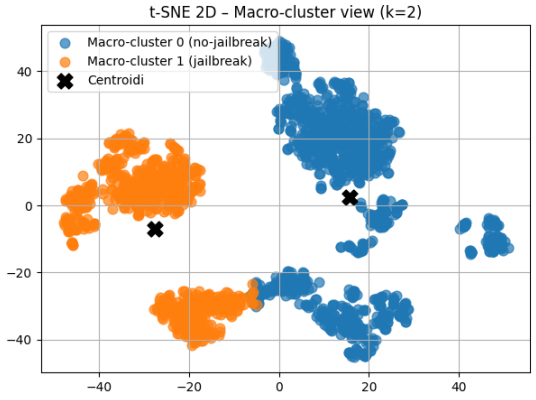

Figura 5.23: t-SNE 2D - Macro-cluster view BERT Fine-tuned (k=2)

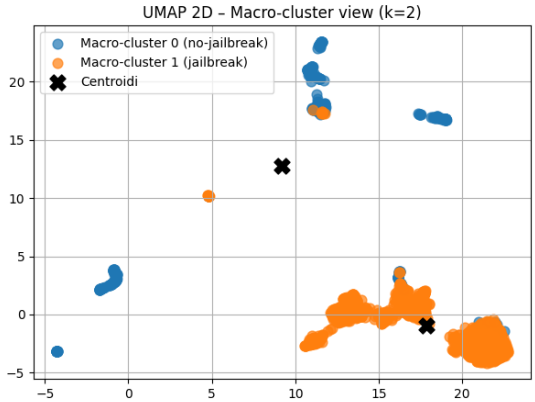

Figura 5.24: UMAP 2D - Macro-cluster view BERT Base (k=2)

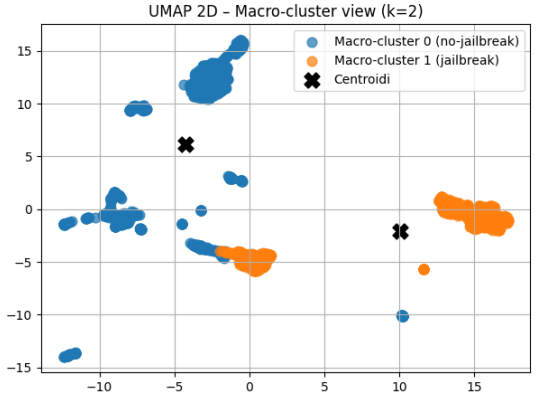

Figura 5.25: UMAP 2D - Macro-cluster view BERT Fine-tuned (k=2)

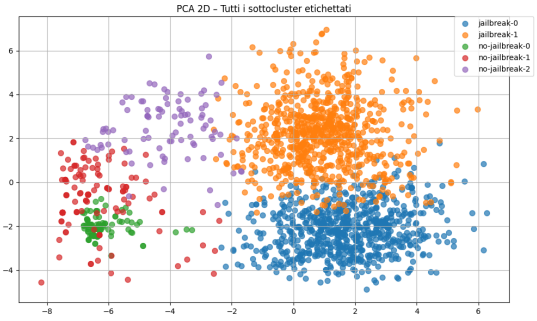

Figura 5.26: PCA 2D - Tutti i sottocluster etichettati BERT Base

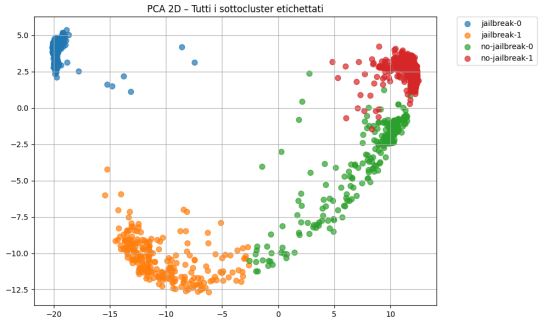

Figura 5.27: PCA 2D - Tutti i sottocluster etichettati BERT Fine-tuned

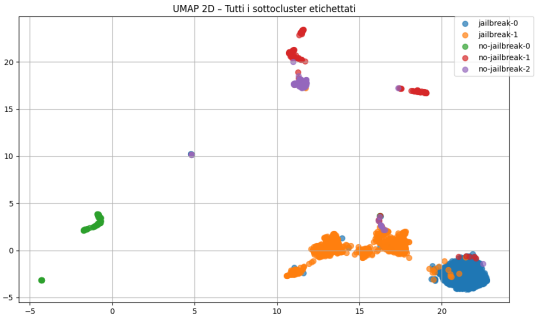

Figura 5.28: UMAP 2D - Tutti i sottocluster etichettati BERT Base

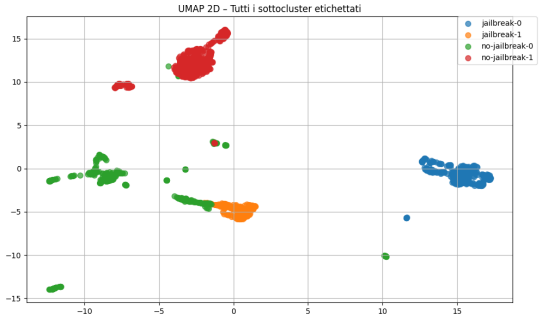

Figura 5.29: UMAP 2D - Tutti i sottocluster etichettati BERT Fine-tuned

CAPITOLO 6

Conclusioni

Il presente lavoro ha affrontato una sfida cruciale nell'ambito della sicurezza dei modelli di linguaggio di grandi dimensioni: lo sviluppo di un sistema automatico per la classificazione e l'analisi di tentativi di jailbreak attraverso tecniche di clustering applicate agli embedding BERT. L'evoluzione metodologica del sistema, dalla versione iniziale semplificata fino all'implementazione finale con clustering gerarchico, ha fornito evidenze empiriche significative sull'efficacia degli approcci proposti e sull'importanza fondamentale del fine-tuning specializzato.

6.1 Principali risultati e scoperte

L'analisi comparativa condotta ha rivelato risultati particolarmente significativi riguardo all'impatto trasformativo del fine-tuning specializzato sulla qualità delle rappresentazioni semantiche. Il modello BERT fine-tuned ha dimostrato una superiorità consistente e sostanziale rispetto al modello base attraverso tutte le versioni implementate, con miglioramenti che vanno ben oltre l'incremento quantitativo dell'accuratezza di classificazione. Il fine-tuning ha infatti ristrutturato fondamentalmente lo spazio degli embedding, rendendo le categorie jailbreak e non-jailbreak intrinsecamente più separabili e semanticamente coerenti.

I risultati più rilevanti emergono dal confronto tra i due approcci finali sviluppati: il clustering adattivo (versione 3.0) e il clustering gerarchico (versione 4.0). La versione 3.0, basata sull'ottimizzazione automatica del numero di cluster, ha prodotto risultati eccellenti per il modello BERT fine-tuned, identificando automaticamente 3 cluster ottimali con un Silhouette Score di 0.61 e mantenendo un'accuratezza elevata dell'84.5%. Questa configurazione ha rivelato l'esistenza di sottostrutture semantiche latenti, dimostrando che i tentativi di jailbreak non costituiscono una categoria omogenea ma presentano gradazioni e tipologie distinte.

La versione finale con clustering gerarchico, pur presentando un Silhouette Score leggermente inferiore (0.547), offre vantaggi complementari di notevole rilevanza pratica.

L'architettura a due livelli permette di preservare simultaneamente la robustezza della distinzione binaria fondamentale (jailbreak vs non-jailbreak) e di esplorare sottocategorie semantiche all'interno di ciascun macro-cluster. Questa duplice capacità rappresenta un valore aggiunto significativo per applicazioni di sicurezza operative, dove è necessario bilanciare l'affidabilità della classificazione primaria con la granularità interpretativa per l'identificazione di pattern emergenti.

6.2 Allineamento con gli obiettivi di ricerca

Gli obiettivi stabiliti nell'introduzione sono stati pienamente raggiunti attraverso l'implementazione e la validazione del sistema proposto. L'obiettivo primario di sviluppare un approccio integrato che combinasse classificazione supervisionata basata su BERT con tecniche di clustering non supervisionato è stato realizzato con successo, come dimostrato dalle performance superiori del modello fine-tuned rispetto al baseline generalista. L'incremento di accuratezza da 68.9% a 84.7% e il miglioramento del Silhouette Score da 0.15 a 0.61 confermano l'efficacia dell'approccio metodologico adottato.

L'analisi delle caratteristiche intrinseche delle risposte generate da modelli di linguaggio quando sottoposti a prompt evolutivi ha rivelato pattern strutturali significativi, confermando l'ipotesi che le tecniche di clustering possano identificare sottocategorie semantiche latenti all'interno delle classificazioni binarie tradizionali. La scoperta di una struttura tri-cluster nel modello fine-tuned fornisce insight preziosi per lo sviluppo di sistemi di valutazione più sofisticati e granulari.

6.3 Rilevanza e applicazioni pratiche

I risultati ottenuti hanno implicazioni dirette e immediate per lo sviluppo di sistemi di sicurezza AI in contesti operativi reali. L'accuratezza del sistema fine-tuned, che raggiunge l'84.7% nella configurazione ottimale, rappresenta un livello di affidabilità compatibile con deployment in ambienti production, dove la capacità di identificare automaticamente tentativi di manipolazione costituisce una necessità critica. La superiorità del modello specializzato rispetto al modello base, con miglioramenti che arrivano fino a +24.8 punti percentuali, dimostra inequivocabilmente l'importanza dell'investimento computazionale nel fine-tuning per domini specifici.

Dal punto di vista dell'interpretabilità, entrambi gli approcci finali offrono contributi distintivi. Il clustering adattivo si rivela particolarmente efficace per l'identificazione automatica di nuove tipologie di attacco, grazie alla sua capacità di scoprire strutture semantiche emergenti senza vincoli predefiniti. Il clustering gerarchico, invece, fornisce un framework più strutturato per l'analisi multi-livello, permettendo agli operatori di sicurezza di comprendere sia la classificazione binaria immediata che le sfumature semantiche sottostanti.

Nel campo della sicurezza informatica, questi risultati contribuiscono allo sviluppo di sistemi di detection più efficaci per identificare tentativi di manipolazione dei modelli di linguaggio. Nell'ambito dello sviluppo di sistemi di intelligenza artificiale, forniscono metodologie per la valutazione continua della robustezza durante le fasi di sviluppo e deployment, mentre gli insights ottenuti possono guidare lo sviluppo di tecniche di allineamento più robuste e resistenti a tentativi di compromissione.

6.4 Confronto critico tra approcci finali

L'analisi comparativa tra clustering adattivo e gerarchico rivela che non esiste un approccio universalmente superiore, ma piuttosto due metodologie complementari ottimizzate per contesti applicativi differenti. Il clustering adattivo (v3.0) eccelle in scenari di ricerca e scoperta, dove l'obiettivo primario è l'identificazione di pattern semantici emergenti e la massimizzazione della qualità geometrica del clustering. Il Silhouette Score superiore (0.61 vs 0.547) e la capacità di identificazione automatica di sottocategorie lo rendono ideale per ambienti di ricerca e sviluppo dove l'esplorazione delle strutture latenti è prioritaria.

Il clustering gerarchico (v4.0), pur presentando metriche di qualità leggermente inferiori, offre vantaggi strategici in termini di scalabilità interpretativa e robustezza operativa. La struttura a due livelli permette decisioni graduate e strategie di intervento differenziate basate sulla severità semantica dei tentativi di jailbreak, caratteristica fondamentale per sistemi di sicurezza che devono bilanciare protezione e usabilità. Inoltre, l'architettura gerarchica facilita l'integrazione con pipeline di sicurezza esistenti, dove la classificazione binaria primaria rimane il requisito fondamentale.

6.5 Limitazioni e sfide metodologiche

L'analisi condotta ha evidenziato alcune limitazioni significative che meritano considerazione per sviluppi futuri. La dipendenza critica dal fine-tuning specializzato, sebbene giustificata dai risultati, introduce complessità operative e requisiti computazionali che potrebbero limitare l'adozione in contesti con risorse limitate. Il modello base, infatti, ha mostrato performance consistentemente inadeguate attraverso tutte le versioni implementate, con Silhouette Score che raramente superano 0.20 e distribuzioni di cluster fortemente sbilanciate.

Un'altra limitazione emersa riguarda la specificità del dominio del fine-tuning. Il modello specializzato, pur eccellente per la classificazione di jailbreak, potrebbe presentare limitazioni nella generalizzazione a tipologie di attacco significativamente diverse da quelle presenti nel dataset di addestramento. Questa considerazione sottolinea l'importanza di strategie di aggiornamento continuo e di validazione cross-dominio per mantenere l'efficacia del sistema nel tempo.

Dal punto di vista computazionale, l'implementazione del clustering gerarchico richiede risorse di calcolo significative per l'ottimizzazione automatica dei sottocluster, limitando potenzialmente l'applicabilità in scenari real-time ad alto throughput.

6.6 Contributi al corpo di conoscenze esistente

Questo lavoro contribuisce al campo della sicurezza AI fornendo evidenze empiriche concrete sull'efficacia delle tecniche di clustering per l'analisi automatica di risposte LLM. La metodologia dual-model implementata, che confronta sistematicamente modelli base e fine-tuned, stabilisce un paradigma di valutazione robusto per la quantificazione dell'impatto della specializzazione del modello. I risultati ottenuti dimostrano che il fine-tuning non solo migliora l'accuratezza di classificazione, ma trasforma qualitativamente lo spazio delle rappresentazioni semantiche, un insight con implicazioni significative per lo sviluppo di modelli specializzati in altri domini di sicurezza.

L'approccio gerarchico proposto rappresenta inoltre un contributo metodologico innovativo che bilancia efficacemente le esigenze operative di classificazione binaria con i requisiti di ricerca per l'identificazione di sottostrutture semantiche. Questa dualità metodologica offre un framework flessibile applicabile a domini analoghi dove è necessario combinare decision-making binario con analisi esplorativa.

Dal punto di vista teorico, il lavoro fornisce evidenze quantitative sulla ristrutturazione dello spazio degli embedding attraverso il fine-tuning specializzato, contribuendo alla comprensione dei meccanismi attraverso cui i modelli transformer acquisiscono competenze domain-specific. Le metriche di qualità del clustering sviluppate e validate costituiscono inoltre strumenti di valutazione transferibili ad altri contesti di ricerca nel campo dell'analisi semantica automatica.

6.7 Direzioni future e sviluppi prospettici

I risultati ottenuti aprono diverse direttive promettenti per ricerche future. L'estensione della metodologia a modelli di linguaggio di dimensioni superiori rappresenta una priorità naturale, considerando l'evoluzione continua del campo verso architetture sempre più complesse. L'integrazione di tecniche di ensemble che combinino i punti di forza del clustering adattivo e gerarchico potrebbe produrre sistemi ibridi con performance superiori e maggiore robustezza.

Dal punto di vista dell'ottimizzazione computazionale, lo sviluppo di tecniche di clustering incrementale e online costituisce una direzione di ricerca cruciale per l'applicazione in contesti production ad alto throughput. L'implementazione di meccanismi di apprendimento continuo che permettano al sistema di adattarsi automaticamente a nuove tipologie di attacco senza richiedere re-training completo rappresenta un obiettivo strategico per la scalabilità operativa.

Un'area particolarmente promettente riguarda l'integrazione del sistema sviluppato in pipeline di sicurezza più ampie, dove la classificazione di jailbreak costituisce un componente di un ecosistema di difesa multi-livello. L'esplorazione di tecniche di spiegabilità avanzate che permettano agli operatori di comprendere le ragioni semantiche alla base delle classificazioni costituisce inoltre un elemento essenziale per l'adozione in contesti critici dove la trasparenza decisionale è un requisito normativo.

L'applicazione della metodologia a domini correlati, come la detection di misinformazione o l'identificazione di contenuti dannosi, rappresenta un'estensione naturale che potrebbe validare la generalizzabilità dell'approccio proposto. Inoltre, l'integrazione con tecniche di generazione adversariale per il testing automatico della robustezza del sistema costituisce una direzione di ricerca strategica per il mantenimento dell'efficacia a lungo termine.

6.8 Significato complessivo e chiusura

Il lavoro presentato dimostra che l'integrazione di tecniche di clustering avanzate con modelli di linguaggio specializzati rappresenta un approccio efficace e praticabile per la sicurezza AI. La metodologia sviluppata fornisce un contributo concreto al problema della valutazione automatica di risposte LLM, con particolare rilevanza per l'identificazione di tentativi di jailbreak. I risultati ottenuti confermano l'importanza strategica del fine-tuning specializzato e stabiliscono un framework metodologico robusto per la valutazione comparativa di approcci di clustering in domini di sicurezza.

L'evoluzione del sistema attraverso quattro versioni successive ha permesso di validare empiricamente l'efficacia delle scelte metodologiche e di identificare configurazioni ottimali per diversi contesti applicativi. Il confronto sistematico tra clustering adattivo e gerarchico offre agli sviluppatori strumenti concreti per la selezione dell'approccio più appropriato in base alle specifiche esigenze operative.

La robustezza delle evidenze empiriche, confermata attraverso multiple tecniche di validazione (PCA, t-SNE, UMAP) e metriche quantitative complementari, fornisce una base solida per l'adozione della metodologia in contesti operativi reali. La trasparenza metodologica e la riproducibilità dei risultati costituiscono inoltre prerequisiti fondamentali per la validazione scientifica e l'estensione del lavoro da parte della comunità di ricerca.

In definitiva, questo lavoro contribuisce significativamente all'avanzamento delle tecniche di sicurezza per modelli di linguaggio, fornendo sia contributi metodologici che risultati pratici direttamente applicabili. La dimostrazione empirica che il clustering gerarchico su embedding BERT fine-tuned rappresenta una soluzione ottimale per la classificazione e l'analisi di risposte jailbreak in contesti di sicurezza AI operativi costituisce un risultato di rilevanza immediata per lo sviluppo di sistemi di protezione sempre più efficaci e affidabili. L'integrazione riuscita di tecniche di apprendimento supervisionato e non supervisionato

apre inoltre nuove prospettive per approcci ibridi in domini correlati, consolidando il valore metodologico e applicativo del contributo presentato.

Bibliografia

[1] T. M. Mitchell and T. M. Mitchell, *Machine learning*, vol. 1. McGraw-hill New York, 1997.

[2] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 4. Springer, 2006.

[3] Z. Q. John Lu, "The elements of statistical learning: Data mining, inference, and prediction," *Journal of the Royal Statistical Society Series A: Statistics in Society*, vol. 173, pp. 693–694, 06 2010.

[4] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 857–876, 2023.

[5] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1. MIT press Cambridge, 2016.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[7] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, pp. 1137–1145, Montreal, Canada, 1995.

[8] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, p. 1929–1958, Jan. 2014.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[12] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, vol. 5, pp. 281–298, University of California press, 1967.

[13] R. Lapid, R. Langberg, and M. Sipper, "Open sesame! universal black box jailbreaking of large language models," 2024.

[14] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong, "Jailbreaking black box large language models in twenty queries," 2024.

[15] Y. Liu, G. Deng, Z. Xu, Y. Li, Y. Zheng, Y. Zhang, L. Zhao, T. Zhang, K. Wang, and Y. Liu, "Jailbreaking chatgpt via prompt engineering: An empirical study," 2024.

[16] S. Imani, L. Du, and H. Shrivastava, "Mathprompter: Mathematical reasoning using large language models," 2023.

[17] S. Zhao, R. Duan, F. Wang, C. Chen, C. Kang, J. Tao, Y. Chen, H. Xue, and X. Wei, "Jailbreaking multimodal large language models via shuffle inconsistency," 2025.

[18] L. Gao, M. Zhang, and W. Chen, "Shaping the future of ai safety: A framework for model alignment," *Journal of AI Safety*, vol. 15, no. 3, pp. 123–145, 2024.

[19] P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Sehwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramer, H. Hassani, and E. Wong, "Jailbreakbench: An open robustness benchmark for jailbreaking large language models," 2024.

[20] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica, "Judging llm-as-a-judge with mt-bench and chatbot arena," in *Advances in Neural Information Processing Systems* (A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds.), vol. 36, pp. 46595–46623, Curran Associates, Inc., 2023.

Ringraziamenti

Innanzitutto ci tengo a ringraziare di cuore il Professor Ferretti e la Professoressa Saletta per avermi offerto la straordinaria possibilità di lavorare insieme a loro in questo affascinante ambito di ricerca. Li voglio ringraziare in particolare per la disponibilità e l'aiuto che mi hanno dato in questi mesi di lavoro: senza il loro sostegno e la loro guida esperta, non sarebbe stato possibile realizzare questa relazione.

Un ringraziamento profondo e sentito va alla mia famiglia, per l'amore incondizionato e il sostegno instancabile che mi hanno dimostrato durante tutti questi anni di studio. Grazie per il vostro supporto e per i consigli preziosi che mi hanno permesso di diventare la persona che sono oggi e di raggiungere questo straordinario traguardo. Nonostante per venti esami vi abbia ripetuto la frase "tanto non mi laureerò mai", voi avete sempre creduto in me e mi avete spronato a realizzare ciò che sapevate che avrei potuto raggiungere.

Un pensiero speciale va a mia sorella, che ha sempre creduto in me. Anche quando abitava a Roma, è sempre stata al mio fianco, seppur a distanza: pronta a festeggiare ogni mio successo e a tendermi la mano nei momenti difficili. La tua presenza nella mia vita è un dono prezioso e la tua forza mi ha sempre ispirato. Grazie per essere la sorella straordinaria che sei.

A mamma e papà, i miei due pilastri portanti: non riesco a trovare le parole giuste per esprimere quanto bene vi voglio e quanto siete importanti per me. Siete la mia roccia, la mia certezza, il mio esempio di vita. Ogni vostro sacrificio, ogni vostro consiglio, ogni vostro abbraccio mi ha dato la forza di andare avanti. Non potrò mai esservi grato abbastanza per tutto quello che fate per me ogni singolo giorno.

Al resto della mia famiglia, Ognuno di voi occupa un posto speciale nel mio cuore e ha contribuito a rendere la mia vita più ricca e piena d'amore. Siete una fonte inesauribile di affetto, saggezza e gioia. La vostra presenza nella mia vita è un dono prezioso che non darò mai per scontato.

Ai miei amici dedico un pensiero speciale: siete una parte fondamentale e insostituibile della mia vita. Non sono bravo con i discorsi strappalacrime, ma vorrei comunque farvi sapere che siete gli amici migliori che si possano desiderare al mondo. Vorrei ringraziarvi uno ad uno ma per mia fortuna siete veramente tanti, proverò lo stesso: Giovanni, Chiara, Giorgia, Antonio, Caterina, Elena, Gloria, Gabriele, Cristina, Irene, Valentina, Phoebe, Filippo, Matteo, Alessio, Gianni . E a tutti coloro che non ho nominato ma che sono ugualmente preziosi nel mio cuore, sappiate che vi voglio bene immensamente e che anche

voi fate parte di questo gruppo speciale che rende la mia vita più bella. Ognuno di voi ha reso speciale questo percorso con la propria unicità e amicizia sincera.

Ai miei compagni in questa avventura di tre anni devo un ringraziamento particolare: avete reso questi anni di studio tra i più belli e divertenti della mia vita. Un ringraziamento speciale va a Daniele Buser, Stefano Brighenti e Samuele Carbone che in questi tre anni sono stati la mia famiglia accademica. Sono certo di aver trovato degli amici veri con cui in così poco tempo ho condiviso così tanto: dalle notti passate a studiare insieme alle risate nei momenti di pausa, dalle ansie pre-esame alle gioie dei traguardi raggiunti. Non potrò mai ringraziarvi abbastanza per aver reso questo cammino meno solitario e infinitamente più bello.

Ovviamente un ringraziamento va anche a tutti coloro che in questi tre anni alla Bicocca mi hanno aiutato e rallegrato le giornate, Siete veramente tanti e non riuscirei a nominarvi tutti uno per uno, ma ognuno di voi ha contribuito a rendere speciale questa esperienza universitaria.

Infine, ringrazio me stesso per non aver mai mollato, per aver creduto in questo sogno anche nei momenti più difficili e per essere arrivato fin qui. Questo traguardo non è solo mio, ma di tutti coloro che hanno camminato accanto a me in questo percorso straordinario.