UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

DIPARTIMENTO DI INGEGNERIA
DELL'INFORMAZIONE

Corso di Laurea
in Ingegneria Informatica

Relazione Finale

Sviluppo di tecniche di fuzzing per il jailbreak di LLM

Relatore: Chiar.mo Prof. Pietro Baroni

Laureanda:
Chaimaa El Koudri

Matricola n. 731062

Anno Accademico 2025/2026

Sommario

Introduzione

Negli ultimi anni, i Large Language Models (LLMs) hanno trasformato il panorama dell'intelligenza artificiale, trovando applicazione in numerosi ambiti, dalla generazione automatica di testi all'assistenza virtuale avanzata. Tuttavia, l'adozione crescente degli LLM ha sollevato importanti questioni legate alla sicurezza, in particolare per quanto riguarda la loro vulnerabilità a tecniche di attacco come il prompt injection e il jailbreak. Tali tecniche rappresentano una minaccia significativa all'affidabilità del modello, poiché dati sensibili potrebbero essere estratti e l'output del modello potrebbe essere alterato in modo malevolo.

Il presente lavoro di tesi si inserisce in questo contesto, con l'obiettivo di sviluppare e migliorare tecniche di fuzzing, ovvero un metodo di testing automatico che invia input casuali a un sistema per individuare vulnerabilità, per testare la sicurezza degli LLM e rilevare potenziali falle di sicurezza. Il progetto si articola in due principali linee di sviluppo: l'ampliamento di un framework di fuzzing esistente, con l'introduzione di nuovi test per l'individuazione di vulnerabilità, e la progettazione di un assistente collaborativo basato su Retrieval-Augmented Generation (RAG)[1] per il supporto ai penetration tester. Quest'ultimo strumento si pone come un ausilio per supportare i pentester junior nell'analisi delle vulnerabilità, fornendo suggerimenti e informazioni estratte da documentazione interna e report aziendali.

Le tecniche di fuzzing rappresentano un approccio essenziale per la scoperta automatizzata di vulnerabilità nei sistemi software, incluso il contesto degli LLM. Il fuzzing consiste nell'invio di appositi input, generati in modo casuale o strutturato, per testare la robustezza del modello e

[1] La Retrieval-Augmented Generation (RAG) è il processo di ottimizzazione dell'output di un modello linguistico di grandi dimensioni, in modo che faccia riferimento a una base di conoscenza autorevole al di fuori delle sue fonti di dati di addestramento prima di generare una risposta.

identificare comportamenti anomali o falle di sicurezza. In particolare, nel caso degli LLM, il fuzzing può essere utilizzato per generare prompt avversari con l'obiettivo di eludere i filtri di sicurezza o estrarre informazioni sensibili. Parallelamente, il concetto di RAG consente di migliorare la qualità delle risposte degli LLM combinando la generazione testuale con il recupero di informazioni da fonti affidabili, rendendolo uno strumento utile per il penetration testing, metodologia che simula attacchi reali per identificare e correggere debolezze prima che possano essere sfruttate.

L'integrazione di fuzzing, RAG e penetration testing in un framework unificato consente di rafforzare la sicurezza degli LLM, migliorando la capacità di rilevare exploit e fornendo strumenti avanzati per la protezione contro attacchi sofisticati.

L'approccio seguito prevede un'analisi preliminare delle metodologie di attacco più diffuse, con riferimento a standard di settore come OWASP[2] e MITRE ATLAS. Successivamente, verranno sviluppati nuovi mutators[3] per generare prompt avversari avanzati e strumenti per la valutazione delle risposte dei modelli sotto attacco. Parallelamente, verrà implementato un assistente collaborativo con meccanismi di controllo dell'accesso e protezione dei dati sensibili.

Infine, le tecniche sviluppate verranno validate attraverso test di sicurezza mirati, che simulano attacchi reali con l'obiettivo di individuare vulnerabilità e proporre misure di mitigazione efficaci. I risultati ottenuti contribuiranno a migliorare la resilienza degli LLM, fornendo strumenti avanzati per la loro valutazione e rafforzando le difese contro possibili exploit.

---

[2] OWASP, acronimo di Open Web Application Security Project, è un'organizzazione no-profit che si dedica a migliorare la sicurezza delle applicazioni web e software, fornendo strumenti, linee guida e risorse open-source per la comunità degli sviluppatori e degli addetti alla sicurezza.

[3] Nel contesto del fuzzing e dell'attacco ai modelli di linguaggio, i mutators sono algoritmi o funzioni che modificano in modo sistematico gli input di test (in questo caso, i prompt) per generare varianti potenzialmente più efficaci nel provocare comportamenti indesiderati o vulnerabilità nel modello.

1 Storia dell'AI: evoluzione da ML a LLM

I primi sistemi di intelligenza artificiale si basavano su un insieme di regole rigide, scritte da esperti del settore, e utilizzavano motori inferenziali per trarre conclusioni. Tuttavia, presentavano limiti significativi: erano complessi da aggiornare, poco adattabili a nuovi scenari e richiedevano una vasta quantità di conoscenza pre-programmata.

Per superare questi limiti, si è cercato di combinare conoscenze esplicite con metodi statistici e probabilistici. Negli anni '80 e '90, l'intelligenza artificiale ha iniziato a utilizzare modelli che permettevano di trattare l'incertezza e di affrontare situazioni più complesse. A differenza delle rigide regole logiche, questi approcci riuscivano a calcolare probabilità e fare previsioni anche in presenza di dati incompleti, portando alla creazione di sistemi più flessibili e meno dipendenti dalla conoscenza manuale.

Negli anni 2000, l’aumento delle risorse computazionali e la crescente disponibilità di grandi quantità di dati hanno favorito l’affermazione dell’apprendimento automatico (machine learning), un paradigma che ha rappresentato un punto di svolta nell’evoluzione dell’intelligenza artificiale. A differenza degli approcci basati su regole, il machine learning consente ai sistemi di apprendere direttamente dai dati, individuando automaticamente pattern e relazioni senza la necessità di programmazione manuale. Grazie all’impiego di modelli matematici e statistici, questi algoritmi hanno reso i sistemi più adattabili, scalabili e in grado di affrontare una vasta gamma di problemi complessi, ampliando significativamente le applicazioni pratiche dell’intelligenza artificiale.

L'evoluzione è stata ulteriormente accelerata dall'aumento della potenza di calcolo e dalla disponibilità di grandi dataset, fattori che hanno favorito lo sviluppo delle reti neurali profonde e del deep learning. Questi progressi hanno permesso l'elaborazione di informazioni con livelli di complessità

sempre maggiori, aprendo la strada alla creazione di modelli di linguaggio avanzati come GPT[4] e BERT[5].

1.1 ML classico: Concetti fondamentali

Il machine learning è un ramo dell'informatica che si occupa dello sviluppo di algoritmi e modelli statistici in grado di individuano schemi e correlazioni e apprendere autonomamente a partire da un insieme iniziale di dati. Questa disciplina consente ai sistemi di svolgere compiti complessi come classificazione, previsione o riconoscimento, adattandosi automaticamente all'evoluzione dei dati.

Il machine learning si basa sull'esistenza di una relazione matematica tra i dati in input e quelli in output. Sebbene questa relazione non sia nota a priori, un modello di machine learning può dedurla analizzando un numero sufficiente di dati. Ogni algoritmo, infatti, si fonda su una funzione matematica che viene adattata e ottimizzata attraverso il processo di apprendimento.

Gli algoritmi di machine learning si classificano in quattro categorie principali, a seconda del tipo di dati disponibili e del metodo di apprendimento utilizzato.

- Apprendimento supervisionato: in questo approccio, viene fornito agli algoritmi un dataset di addestramento etichettato, con input e output chiaramente definiti.
- Apprendimento non supervisionato: a differenza dell'apprendimento supervisionato, gli algoritmi di machine learning non supervisionato lavorano con dati non etichettati.

---

[4] I GPT, acronimo di Generative Pre-trained Transformer, sono una famiglia di modelli di rete neurale che utilizza l'architettura del transformer ed è alla base di applicazioni di IA generativa come ChatGPT.

[5] BERT è l'acronimo di Bidirectional Encoder Representations from Transformers, ed è un aggiornamento in grado di consentire al motore di ricerca di migliorare l'elaborazione del linguaggio umano mediante l'utilizzo di reti neurali, ovvero modelli matematici che si rifanno al funzionamento del cervello umano.

- Apprendimento semi-supervisionato: come suggerisce il nome, questo approccio è un ibrido tra apprendimento supervisionato e non supervisionato. Utilizza una piccola quantità di dati etichettati insieme a un grande volume di dati non etichettati per addestrare il modello. Nella prima fase del processo il modello viene addestrato sui dati etichettati. Successivamente, il modello stesso si occupa di etichettare i dati inizialmente privi di etichette attraverso un processo chiamato pseudo-etichettatura. Infine, viene riaddestrato utilizzando il dataset ampliato.
- Apprendimento per rinforzo: l'apprendimento per rinforzo si basa su un sistema di ricompense e penalità: l'algoritmo compie azioni e riceve un feedback in base ai risultati ottenuti. L'obiettivo è massimizzare le ricompense nel tempo, ottimizzando le decisioni.

1.2 Overfitting e Underfitting nel Machine Learning

Uno degli obiettivi principali di un modello di machine learning è di potersi adattare e riuscire a fare previsioni accurate anche su dati non utilizzati per l'addestramento. Tuttavia, tale auspicabile adattabilità può essere compromessa da due fenomeni opposti: overfitting e underfitting.

L'overfitting si verifica quando un modello diventa troppo specializzato nel dataset di addestramento e perde la capacità di fare previsioni affidabili su dati mai visti. Di conseguenza, pur ottenendo un'elevata accuratezza sui dati di addestramento, il modello fornisce risultati imprecisi su dati di test o reali.

L'overfitting può verificarsi per diverse ragioni, tra cui:

- Dataset troppo piccolo
- Presenza di rumore nei dati
- Eccessiva complessità del modello
- Addestramento prolungato

L'underfitting, invece, si verifica quando un modello non riesce a identificare una relazione significativa tra input e output, risultando incapace di fornire

previsioni accurate anche sui dati di addestramento. Questo problema si verifica se il modello:

- Eccessiva semplicità del modello
- Tempi di addestramento non sufficienti
- Dati utilizzati per l'addestramento non sufficientemente variegati e di numero insufficiente

1.3 Cross-Validation

Nel corso degli anni sono nate diverse tecniche per individuare problemi di overfitting e migliorare le capacità di generalizzazione dei modelli. Un esempio è la cross-validation, tecnica che consente di valutare l'accuratezza di un modello testandolo su diversi sottoinsiemi del dataset.

Una delle tecniche più utilizzate è la k-fold cross-validation, che suddivide i dati di input in k sottoinsiemi (folds):

- Il modello viene addestrato su k-1 folds e testato sul fold rimanente.
- Questo processo viene ripetuto k volte, ogni volta con un diverso fold usato per la valutazione.
- L'accuratezza media di tutti gli esperimenti viene utilizzata per stimare le reali prestazioni del modello.

1.4 Esempi di algoritmi di Machine Learning classici

Prima dell'ascesa delle deep neural networks[6], alcuni algoritmi tradizionali hanno costituito la base dei sistemi di apprendimento automatico e ancora oggi trovano applicazione in diversi settori quali, per esempio, finanza, sanità, marketing e pubblica amministrazione.

[6] Le Deep Neural Networks sono un tipo di rete neurale artificiale composta da più strati di nodi (o neuroni) collegati tra loro. Si chiamano "profonde" proprio perché presentano numerosi strati nascosti tra lo strato di input e quello di output.

1.4.1 Decision Trees

Gli alberi decisionali sono modelli che analizzano i dati attraverso una serie di domande sequenziali, suddividendoli progressivamente in categorie più specifiche. La loro principale caratteristica è la semplicità e nell'intuitività, ma una struttura eccessivamente complessa può scaturire in condizioni di overfitting.

1.4.2 Support Vector Machine

Le Support Vector Machines sono algoritmi che servono per effettuare la classificazione dei dati, cioè per capire a quale gruppo (o classe) appartiene un certo elemento. Una SVM rappresenta i dati come punti nello spazio, mappati in modo tale che quelli appartenenti alle diverse categorie siano chiaramente separati da una linea (o una sua estensione geometrica) il più possibile ampia.

Quando i dati non sono facilmente separabili, usano una tecnica chiamata kernel trick, che mappa implicitamente i loro ingressi in uno spazio delle caratteristiche multidimensionale. Sono molto efficaci per compiti di classificazione e anche per la regressione, ma possono diventare costose in termini computazionali quando i dati sono molto numerosi.

1.4.3 K-Nearest Neighbors

Il metodo k-Nearest Neighbors segue un principio secondo cui, per la classificazione di un nuovo punto dato, vengono osservati k vicini più prossimi e viene assegnata l'etichetta più comune tra loro. Se la maggior parte dei vicini appartiene ad una certa categoria, allora è probabile che anche il nuovo punto ne faccia parte.

2 Deep Learning: Un Approccio Ispirato al Cervello Umano

Il deep learning è una branca avanzata del machine learning che si ispira alla struttura e al funzionamento del cervello umano. Gli algoritmi di deep learning analizzano i dati utilizzando le Artificial Neural Networks (ANN) per elaborare informazioni su più livelli, sfruttando le reti neurali profonde.

Le ANN sono sistemi intelligenti composti da nodi interconnessi, simili ai neuroni biologici. Questi nodi sono organizzati in tre tipi di livelli:

- Livello di input: riceve i dati grezzi (es. immagini, testo, suoni).
- Livelli nascosti: eseguono elaborazioni progressive dei dati, estraendo caratteristiche sempre più complesse.
- Livello di output: genera il risultato finale, come una classificazione o una previsione.

Ogni nodo (neurone artificiale) è composto da dati di input, pesi, un bias[7] e un output. Una volta individuati gli input, viene associato ad ognuno di essi un peso che ne indica la rilevanza. Gli input vengono poi moltiplicati per i rispettivi pesi e sommati tra loro, insieme al bias. Il risultato complessivo viene passato a una funzione di attivazione che permette di determinare l'output del nodo. Se l'output supera una determinata soglia il nodo si attiva e trasmette l'informazione al nodo successivo; altrimenti, rimane inattivo.

Questo processo permette alla rete di apprendere e migliorare nel riconoscere pattern nei dati.

[7] Il bias in una rete neurale è un valore numerico che viene aggiunto alla somma pesata degli input di un nodo, prima di applicare la funzione di attivazione. Serve a spostare il punto in cui un neurone si attiva, permettendo alla rete di adattarsi meglio ai dati.

2.1 Perceptron

Il modello più semplice di neurone artificiale è il perceptron, ideato a livello teorico nel 1958. Il suo funzionamento si basa sull'assegnazione di pesi agli input, per poi sommare i valori e applicare una funzione di attivazione per decidere il risultato finale.

Matematicamente, il perceptron è descritto da questa formula:

$$y = f\left(\sum w_i x_i + b\right)$$

dove:

- x_i sono gli input
- w_i sono i pesi
- b è il bias
- $f(\cdot)$ è la funzione di attivazione

Un singolo perceptron può risolvere solo problemi linearmente separabili[8] ma, combinando più perceptron in reti neurali profonde, è possibile risolvere problemi di maggiore complessità.

2.2 Reti Neurali Feedforward

Le Feedforward Neural Networks (FNN) sono reti in cui i dati transitano in una sola direzione: dall'input verso l'output, senza cicli o retroazioni. Questo tipo di architettura rappresenta la base di molti modelli complessi. Nel caso più semplice, come nel Single-layer Perceptron, i dati vengono elaborati passando direttamente dallo strato di input a quello di output. Nelle reti Multi-Layer Perceptron (MLP), invece, i dati attraversano uno o più strati nascosti, consentendo alla rete di apprendere strutture e rappresentazioni più astratte e sofisticate.

[8] I problemi linearmente separabili sono quei problemi di classificazione in cui è possibile dividere i dati di classi diverse usando una retta (in 2D), un piano (in 3D) o un iperpiano (in spazi di dimensioni superiori).

Queste reti sono fondamentali per il deep learning, ma necessitano di un meccanismo per aggiornare i pesi e migliorare l'accuratezza del modello: la backpropagation.

2.3 Backpropagation

La backpropagation (propagazione all'indietro) è un algoritmo di ottimizzazione che permette alle reti neurali di imparare dai propri errori. Viene usato per aggiornare i pesi della rete in modo da ridurre l'errore tra l'output previsto e l'etichetta[9] (output corretto).

Il processo segue le seguenti fasi:

1. Forward Pass: i dati passano attraverso la rete, producendo un output.
2. Calcolo dell'Errore: la differenza tra l'output previsto e il valore reale viene misurata tramite una funzione di perdita.
3. Backward pass: l'errore viene "propagato all'indietro" attraverso la rete, partendo dall'output fino agli strati iniziali.
4. Aggiornamento dei Pesi: utilizzando l'algoritmo di discesa del gradiente[10], i pesi vengono aggiornati all'indietro lungo la rete per ridurre l'errore.

La backpropagation è fondamentale per migliorare le prestazioni di una rete neurale, rendendo possibile il deep learning.

[9] Le etichette sono i valori di output associati a ciascun dato di input nel dataset. Rappresentano la risposta corretta che il modello deve imparare a predire.
[10] La discesa del gradiente (in inglese Gradient Descent) è un algoritmo di ottimizzazione iterativo utilizzato per trovare il minimo (locale) di una funzione differenziabile. È ampiamente impiegato in ambito di machine learning e deep learning per minimizzare le funzioni di costo.

2.4 Funzioni di Attivazione

Le funzioni di attivazione determinano come un nodo si attiva e trasforma i dati in output. Alcune delle più utilizzate sono:

- ReLU (Rectified Linear Unit):

  $$f(x) = \max(0, x)$$

  È la più comune nel deep learning, poiché aiuta a evitare il problema della scomparsa del gradiente[11].
- Sigmoid:

  $$f(x) = \frac{1}{1 + e^{-x}}$$

  È utile per problemi di classificazione binaria, ma può soffrire di saturazione per valori estremi.
- Tanh (Tangente Iperbolica):

  $$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

  Simile alla sigmoid, ma con output compreso tra -1 e 1.
- Softmax: Trasforma un vettore in una distribuzione di probabilità, usata nelle classificazioni multi-classe.

2.5 Transformers: L'Innovazione nel Deep Learning

I Transformers rappresentano un'architettura rivoluzionaria nel campo delle reti neurali, introdotta nel 2017 con il celebre paper "Attention Is All You Need". Questa innovazione ha trasformato l'elaborazione del linguaggio naturale[12] (NLP), dando origine a modelli avanzati come GPT, BERT e T5.

---

[11] Il problema della scomparsa del gradiente si verifica quando, durante l'addestramento di reti neurali profonde, i gradienti diventano progressivamente più piccoli nei primi strati. Questo ostacola l'aggiornamento dei pesi, rendendo difficile per la rete apprendere correttamente.

[12] L'elaborazione del linguaggio naturale (NLP) è una tecnologia di machine learning che offre ai computer la capacità di interpretare, manipolare e comprendere il linguaggio umano.

A differenza delle reti neurali ricorrenti (RNN)[13], i Transformers sfruttano il meccanismo di self-attention, che consente di assegnare un peso variabile a diverse parti di un input contemporaneamente. Questo approccio migliora la comprensione del contesto nei testi e nelle immagini, superando i limiti delle tecnologie precedenti.

Le prime tecniche di NLP si basavano su modelli di machine learning che analizzavano la frequenza delle relazioni tra parole, cercando di prevedere il termine successivo in base al contesto immediato. Tuttavia, queste tecnologie faticavano a mantenere il contesto su sequenze lunghe. I Transformers hanno risolto questo problema, permettendo la gestione delle dipendenze a lungo raggio nel testo e garantendo una maggiore coerenza semantica.

I Transformers si distinguono nettamente da altre architetture di reti neurali:

- RNN (Reti Neurali Ricorrenti): elaborano i dati sequenzialmente, processando uno alla volta gli elementi di una sequenza e mantenendo uno stato nascosto aggiornato nel tempo. Questo approccio limita la capacità di gestire lunghe dipendenze (relazioni tra elementi distanti della sequenza) perché ogni stato dipende dalla memoria precedente, con il rischio di perdere informazioni a lungo termine. Inoltre, l'elaborazione sequenziale impedisce l'elaborazione parallela, rallentando significativamente l'addestramento rispetto ai Transformers, che processano intere sequenze simultaneamente, aumentando così l'efficienza.
- CNN (Reti Neurali Convoluzionali): sono particolarmente adatte per nell'elaborazione di dati strutturati come le immagini, sfruttando filtri convoluzionali per catturare pattern locali (ad esempio, bordi o texture) in modo altamente efficiente. Sebbene inizialmente progettati per il testo, i Transformers sono stati adattati anche alla

[13] Una rete neurale ricorrente (RNN) è un modello di deep learning addestrato per elaborare e convertire un input di dati sequenziale in un output di dati sequenziale specifico.

visione artificiale, portando una maggiore flessibilità e potenza di calcolo. A differenza delle CNN, i Transformers non si limitano solo alla localizzazione spaziale dei dati, ma riescono a cogliere le relazioni globali tra le caratteristiche attraverso l'auto-attenzione, che può essere applicata a qualsiasi tipo di dato, anche a sequenze temporali o immagini.

2.6 Applicazioni del Deep Learning: Visione Artificiale

Una delle applicazioni più avanzate del deep learning è la visione artificiale, che permette ai computer di "vedere" e interpretare il mondo visivo.

Un esempio pratico è il funzionamento delle auto a guida autonoma:

1. La telecamera dell'auto cattura un'immagine di un segnale stradale.
2. L'immagine viene inviata all'algoritmo di deep learning per l'analisi.
3. Nei livelli nascosti della rete neurale:
   - Il primo livello rileva i contorni del segnale.
   - Il secondo distingue i colori.
   - Il terzo riconosce lettere e simboli.
4. L'algoritmo identifica il segnale come un cartello di STOP e il sistema dell'auto attiva il freno.

Questa capacità di analizzare immagini in modo autonomo è essenziale non solo per la guida autonoma, ma anche per molte altre applicazioni, come il riconoscimento facciale, la diagnostica medica e il monitoraggio industriale.

3 L'Elaborazione del Linguaggio Naturale (NLP)

L'elaborazione del linguaggio naturale (NLP) è una tecnologia di machine learning che permette ai computer di interpretare, manipolare e comprendere il linguaggio umano. Il software NLP consente di elaborare automaticamente grandi volumi di dati testuali e vocali provenienti da vari canali di comunicazione, analizzare l'intento o il sentiment nei messaggi e rispondere in tempo reale, superando ostacoli come dialetti, slang e irregolarità grammaticali.

Le aziende lo impiegano per diverse attività automatizzate, tra cui:

- Elaborazione e archiviazione di documenti di grandi dimensioni
- Analisi del feedback dei clienti e delle registrazioni dei call center
- Creazione di chatbot per l'assistenza clienti automatizzata
- Classificazione ed estrazione del testo

L'NLP combina linguistica computazionale, machine learning e deep learning per consentire alle macchine di comprendere e generare il linguaggio umano.

3.1 Linguistica Computazionale

La linguistica computazionale è la base su cui si sviluppa l'NLP. Studia il modo in cui il linguaggio può essere modellato per essere processato da un computer. Il suo obiettivo è creare strutture che permettano alle macchine di analizzare la sintassi e il significato delle parole, consentendo loro di interpretare il linguaggio umano in modo più simile a come lo fa una persona. Tecniche come l'analisi grammaticale, la tokenization[14] e la

[14] La tokenization è il processo di suddivisione di un testo in unità più piccole chiamate token. Questi token possono essere parole, frasi o persino caratteri, a seconda del livello di analisi richiesto.

lemmatization[15] sono utilizzate per suddividere il testo in unità più gestibili, riducendo ambiguità e rendendo il contenuto più facilmente interpretabile.

Altri concetti includono il part-of-speech tagging[16] e la creazione di modelli come il bag of words e n-grams per analizzare sequenze di parole e frequenze di occorrenza.

3.2 Machine Learning

Per migliorare la capacità delle macchine di interpretare il linguaggio in modo più flessibile, entra in gioco il machine learning. I modelli di machine learning apprendono dai dati, migliorando la loro capacità di riconoscere schemi linguistici nel tempo permettendo così alle macchine di distinguere sinonimi, comprendere frasi ambigue e persino cogliere il tono emotivo di un messaggio.

Il machine learning è fondamentale per rendere gli assistenti virtuali più naturali nelle conversazioni, per migliorare i motori di ricerca e per affinare i sistemi di analisi del sentiment. Più dati vengono forniti al modello, più questo diventa capace di comprendere le sfumature del linguaggio, avvicinandosi sempre di più a un'interpretazione simile a quella umana.

3.3 Deep Learning

Il deep learning rappresenta un ulteriore passo avanti nell'evoluzione dell'NLP. Tecnologie come i transformers hanno rivoluzionato il modo in cui le macchine comprendono il linguaggio, permettendo loro di generare testi in maniera sempre più naturale e coerente. Questo approccio ha reso possibile lo sviluppo di chatbot avanzati, strumenti di traduzione automatica

[15] La lemmatization è il processo di riduzione di una parola alla sua forma base o lemma, mantenendo il significato corretto nel contesto. A differenza dello stemming, che tronca semplicemente la parola, la lemmatization utilizza dizionari morfologici e regole grammaticali per trovare la forma corretta.

[16] Il part-of-speech tagging (POS tagging) è il processo di assegnare a ogni parola di una frase una categoria grammaticale, come sostantivo, verbo, aggettivo, avverbio, pronome, ecc.

di alta qualità e sistemi in grado di rispondere alle domande degli utenti con precisione e pertinenza.

Grazie alla combinazione di linguistica computazionale, machine learning e deep learning, oggi l'NLP è utilizzato in un numero sempre crescente di applicazioni, rendendo l'interazione con la tecnologia più fluida e intuitiva. Con il continuo sviluppo di modelli sempre più sofisticati, il futuro dell'NLP si avvicina a una comprensione del linguaggio umano sempre più profonda e naturale.

3.4 Fasi di implementazione del processo di NLP

Il processo di NLP si sviluppa attraverso diverse fasi, che permettono alle macchine di elaborare e comprendere il linguaggio umano con sempre maggiore precisione.

Tutto inizia con la raccolta e la preparazione dei dati, un passaggio fondamentale per garantire che il modello abbia abbastanza informazioni per apprendere e migliorare le proprie prestazioni. Spesso, queste informazioni non sono strutturate e devono essere organizzate e pulite prima di poter essere utilizzate nei modelli di NLP.

3.4.1 Preelaborazione

Una volta raccolti i dati, è necessario prepararli affinché il modello possa interpretarli correttamente. Qui entra in gioco la fase di preelaborazione, un insieme di tecniche che trasformano il testo grezzo in un formato più facilmente analizzabile.

Un passaggio essenziale è la tokenization, che suddivide un testo in unità più piccole, come parole o frasi. Un altro processo importante è lo stemming[17] e la lemmatization, che riducono le parole alla loro forma base

[17] Lo stemming è una tecnica di preelaborazione del testo nell'elaborazione del linguaggio naturale (NLP). In particolare, è il processo di riduzione della forma inflessa di una parola a un cosiddetto "tema", o forma radice, che spesso corrisponde al "lemma", come viene chiamato in linguistica

per evitare variazioni inutili. Inoltre, vengono eliminate le stop word, ossia quelle parole che non aggiungono un valore significativo alla comprensione del testo.

3.4.2 Formazione del Modello

Dopo la fase di preelaborazione, il modello di NLP viene addestrato sui dati puliti. Qui entra in gioco il machine learning: attraverso grandi quantità di dati, il sistema impara a riconoscere schemi e a migliorare la sua capacità di comprensione e generazione del linguaggio.

Questo processo di addestramento può essere supervisionato oppure non supervisionato, lasciando che la macchina scopra autonomamente le strutture nel testo. Maggiore è la quantità e la qualità dei dati utilizzati, più il modello diventa accurato ed efficiente nelle sue previsioni e risposte.

3.4.3 Implementazione e Inferenza

Una volta che il modello è stato addestrato con successo, è pronto per essere implementato e utilizzato in un ambiente reale. Il modello viene integrato in software o sistemi aziendali e messo alla prova con dati in tempo reale. Questa fase è essenziale per verificare l’efficacia del modello e, se necessario, perfezionarlo con ulteriori miglioramenti per renderlo ancora più preciso e reattivo.

3.5 Word Embeddings

Una delle innovazioni chiave nell'NLP è la rappresentazione delle parole come vettori numerici in uno spazio continuo, un approccio noto come word embeddings.

Questi vettori permettono ai modelli di catturare significati più complessi delle parole rispetto alla semplice corrispondenza tra simboli e definizioni, migliorando le performance nei compiti di analisi semantica, traduzione e riconoscimento delle entità.

3.6 N-grams e Bag of Words (BoW)

Bag of Words e n-grams sono due approcci fondamentali nell'elaborazione del linguaggio naturale per la rappresentazione del testo.

Il BoW è uno dei modelli più semplici e diretti per rappresentare il testo. Tale approccio ignora l'ordine delle parole e si concentra solo sulla loro presenza o frequenza all'interno di un documento. Ogni parola nel testo viene trattata come una dimensione separata, e il suo valore è la frequenza con cui appare.

Il principale vantaggio di BoW è la sua semplicità, che lo rende particolarmente utile per compiti come la classificazione del testo o la scoperta di topic. Tuttavia, uno dei limiti più evidenti di BoW è che non tiene conto dell'ordine delle parole e la conseguente perdita delle informazioni sul contesto.

Gli n-grams sono un passo avanti rispetto al modello BoW. Un n-gram è una sequenza di n parole consecutive, e l'idea principale è quella di catturare il contesto locale delle parole, cercando di mantenere una parte del contesto che BoW ignora, ovvero la relazione tra le parole adiacenti.

4 Large Language Models

Un Large Language Model (LLM) è una tecnologia avanzata in grado di riconoscere e generare testo che viene addestrata su enormi set di dati. Gli LLM sono basati sul machine learning, nello specifico su un tipo di rete neurale chiamata modello Transformer.

4.1 Architettura Transformer e Meccanismi Interni

I transformer hanno rivoluzionato l'NLP grazie alla loro capacità di elaborare il contesto in modo globale attraverso il meccanismo di self-attention. A differenza delle reti RNN, che analizzano le frasi parola per parola in ordine, i nuovi modelli sono in grado di elaborare l'intera frase tutta insieme, cogliendo più facilmente le relazioni tra le parole. Questa caratteristica è stata fondamentale per lo sviluppo dei LLM, come GPT e BERT, che hanno ridefinito il modo in cui le macchine comprendono e generano il linguaggio umano.

I LLM sfruttano la struttura encoder-decoder dei transformer per interpretare e generare testo in maniera sofisticata.

- Encoder: converte il testo in rappresentazioni word embeddings, catturando non solo il significato delle singole parole ma anche le relazioni contestuali tra di esse.
- Decoder: genera il testo in base ai pattern appresi durante l'addestramento, producendo risposte coerenti e contestualmente rilevanti.

Il termine "Large" nei LLM si riferisce al numero di parametri che li compongono, che può variare da milioni a centinaia di miliardi. Questi parametri sono responsabili della capacità del modello di riconoscere schemi complessi e sfumature nel linguaggio umano. Più un LLM è grande, maggiore è la sua capacità di generare testi fluidi e contestualmente pertinenti. Tuttavia, questa grandezza comporta anche sfide in termini di costi computazionali, consumo energetico e necessità di ottimizzazione.

Per affrontare tali sfide, sono state sviluppate diverse tecniche:

- Fine-tuning: consente di adattare un LLM pre-addestrato a contesti specifici, come il linguaggio medico o legale.
- RAG: combina il modello con fonti di conoscenza esterne, migliorando la qualità e l'accuratezza delle risposte senza dover riaddestrare l'intero modello.
- RLHF[18]: utilizza il feedback umano per affinare le risposte del modello, rendendolo più allineato alle aspettative degli utenti.

4.2 Tecniche Avanzate di Addestramento

Le reti neurali basate su transformer sono estremamente complesse e strutturate su più livelli. La composizione dei livelli, insieme agli embeddings, costituiscono i parametri del modello.

L'addestramento degli LLMs avviene su dataset di grande scala e di alta qualità, permettendo al modello di apprendere attraverso un processo iterativo durante il quale i parametri vengono continuamente aggiornati per migliorare la capacità del modello di predire il token successivo basandosi sulla sequenza precedente. Questa ottimizzazione avviene tramite tecniche di autoapprendimento, che consentono alla rete di adattarsi per massimizzare la probabilità di generare output coerenti con gli esempi forniti nel training.

Una volta addestrati, i Large Language Models possono essere adattati a compiti specifici con dataset relativamente ridotti attraverso un processo di ottimizzazione.

Esistono tre approcci principali per adattare un LLM a specifiche esigenze:

- Apprendimento zero-shot: il modello di base può rispondere a una vasta gamma di richieste senza aver ricevuto un addestramento

[18] L'apprendimento per rinforzo da feedback umano (RLHF) è una tecnica di machine learning che utilizza il feedback umano per ottimizzare i modelli di ML in modo che autoapprendano in maniera più efficiente.

esplicito su di esse. Tuttavia, l'accuratezza delle risposte può variare in base alla complessità del compito.

- Apprendimento few-shot: fornendo al modello alcuni esempi pertinenti, la sua capacità di rispondere migliora significativamente in un ambito specifico, consentendo una maggiore precisione nei risultati.
- Ottimizzazione: rappresenta un passo ulteriore rispetto all'apprendimento few-shot. In questa fase, i data scientist addestrano ulteriormente il modello di base, modificandone i parametri con dati specifici per un'applicazione particolare, migliorandone così la specializzazione e l'efficacia in contesti specifici.

Tali tecniche permettono di massimizzare le prestazioni degli LLM, garantendo una maggiore adattabilità e precisione in ambiti di utilizzo sempre più diversificati.

4.3 LLMOps e Ottimizzazione dei Modelli

Gli LLM sono strumenti in continua evoluzione, motivo per cui le aziende devono definire procedure consigliate per il loro utilizzo e gestione. In questo contesto si inserisce la metodologia LLMOps, che facilita l'adozione e l'operatività di questi modelli.

LLMOps (Large Language Model Operations) si riferisce a un insieme di pratiche e strumenti volti a gestire l'intero ciclo di vita degli LLM, dall'ottimizzazione iniziale alla manutenzione, consentendo di automatizzare i processi, migliorare l'efficienza e semplificare il lavoro di sviluppatori e team aziendali.

Seguendo un approccio simile a MLOps, che si concentra sull'integrazione continua dei modelli di machine learning nei flussi di sviluppo, LLMOps punta a strutturare un ciclo iterativo di sperimentazione, distribuzione e miglioramento continuo degli LLM, garantendo un'evoluzione costante delle loro prestazioni.

Sebbene condividano alcuni principi, gli LLM e i modelli di machine learning tradizionali differiscono in diversi aspetti chiave:

- Apprendimento: i modelli tradizionali sono spesso addestrati da zero, mentre gli LLM partono da modelli pre-addestrati e vengono affinati con dati specifici.
- Ottimizzazione: gli LLM adottano tecniche avanzate per migliorare l'accuratezza e ridurre i costi computazionali, come la RAG.
- Feedback: il miglioramento degli LLM si basa su metodi come il RLHF che prevede l'uso di interazioni con utenti o esperti per valutare le risposte del modello e correggere eventuali errori o incoerenze. Al contrario, i modelli tradizionali tendono a basarsi su metriche standard di valutazione come accuratezza, errore quadratico medio (MSE) o altre misure statistiche, senza l'integrazione di un feedback diretto dall'ambiente o dagli utenti, che spesso limita la capacità di adattarsi dinamicamente alle esigenze reali.
- Valutazione: gli LLM usano metriche specifiche he sono progettate per valutare la coerenza, la grammatica, la pertinenza e la fluidità del contenuto prodotto come BLEU[19] e ROUGE[20] per misurare la qualità del testo generato. Nei modelli tradizionali, la valutazione si concentra maggiormente su metriche numeriche generali come l'accuratezza, senza un'attenzione diretta alla qualità del testo generato.

Per gestire queste sfide, l'implementazione di LLMOps assicura scalabilità, affidabilità e un miglioramento continuo delle prestazioni.

---

[19] BLEU (Bilingual Evaluation Understudy) è una metrica usata per valutare la qualità delle traduzioni automatiche, confrontando la traduzione generata con una o più traduzioni di riferimento.

[20] ROUGE (Recall-Oriented Understudy for Gisting Evaluation) è una metrica utilizzata per valutare la qualità dei riassunti automatici, confrontando il riassunto generato con uno o più riassunti di riferimento.

5 Sicurezza degli LLM: Vulnerabilità e Rischi Emergenti

Se da un lato le potenzialità degli LLM sono straordinarie, dall'altro emergono vulnerabilità che potrebbero essere sfruttate in maniera dannosa. Le caratteristiche stesse degli LLM, come la loro capacità di apprendere dai dati e la tendenza a generalizzare, li rendono suscettibili a una serie di attacchi. Tra i rischi principali figurano diversi tipologie di attacchi quali, per esempio, il prompt injection, il data poisoning, il denial of service e molte altre minacce potenzialmente gravi.

5.1 OWASP (Open Web Security Project)

OWASP è un organizzazione internazionale no profit che cerca si dedica a migliorare la sicurezza delle infrastrutture informatiche globali, offrendo una vasta gamma di risorse gratuite, tra cui documenti, strumenti, video, conferenze e forum.

Nel 2023 un gruppo di esperti di OWASP ha avviato un nuovo progetto con l'obiettivo di documentare i principali rischi alla sicurezza degli LLM. La documentazione risultante identifica minacce emergenti, fornisce esempi concreti di vulnerabilità e descrive scenari di attacco reali, con l'intento di sensibilizzare e guidare le organizzazioni nell'affrontare tali sfide.

5.1.1 prompt injection

Gli attacchi di *prompt injection* rappresentano una delle principali minacce nei sistemi basati su modelli LLM. Si verificano quando i prompt dell'utente riescono a manipolare il comportamento del modello, causando potenzialmente la violazione delle linee guida, la generazione di contenuti dannosi, l'accesso non autorizzato o l'influenza di decisioni critiche.

Questi attacchi possono avvenire in due modalità:

- Direttamente, quando l'input dell'utente modifica in modo involontario o imprevisto il comportamento del modello;

- Indirettamente, quando il modello elabora dati provenienti da fonti esterne — come siti web o file — che contengono elementi capaci di influenzarlo una volta interpretati.

In entrambi i casi, la manipolazione può essere frutto sia di azioni intenzionali sia di errori accidentali.

I casi più comuni in cui si manifesta il prompt injection includono applicazioni web che generano risposte basate su input utente, sistemi di *e-mail automation*[21], agenti AI che consultano fonti esterne come PDF o siti web, e toolchain intelligenti (come LangChain[22] o AutoGPT) che eseguono comandi in base al linguaggio naturale. Anche i sistemi con memoria contestuale sono vulnerabili a iniezioni "dormienti", che si attivano in interazioni successive.

Gli obiettivi principali degli attaccanti variano in base al contesto d'uso del modello, ma convergono generalmente verso il controllo del comportamento del sistema. L'attaccante può mirare a bypassare le restrizioni di sicurezza imposte nel prompt di sistema, ottenere accesso a dati sensibili, o indurre il modello a generare risposte scorrette, dannose o manipolate. In scenari più avanzati la prompt injection può persino servire a compromettere agenti intelligenti operativi, influenzare decisioni automatizzate o innescare payload persistenti attraverso la memoria del sistema.

Sebbene prompt injection e jailbreaking siano concetti correlate, spesso vengono utilizzati erroneamente in modo intercambiabile. Il prompt injection comporta la manipolazione delle risposte del modello tramite input specifici

[21] L'e-mail automation è l'uso di software per configurare e inviare e-mail in modo automatico sulla base di trigger (come l'iscrizione a una newsletter, un acquisto, o un'interazione con un sito web) o flussi predefiniti (workflow), con l'obiettivo di personalizzare e ottimizzare la comunicazione con l'utente.

[22] LangChain è un framework open-source per la costruzione di applicazioni basate su LLM, che consente l'integrazione di questi modelli con fonti di dati esterne, strumenti computazionali e componenti personalizzabili.

per alterarne il comportamento, il che può includere l'aggiramento delle misure di sicurezza. Il jailbreaking, invece, è una forma di prompt injection in cui l'input induce il modello a ignorare completamente i suoi protocolli di sicurezza.

5.1.2 Sensitive Information Disclosure

Gli LLM, soprattutto se integrati in sistemi più complessi, rischiano di esporre dati sensibili, algoritmi proprietari o informazioni riservate attraverso il loro output. Ciò può comportare accessi non autorizzati ai dati, violazioni della privacy e violazioni della proprietà intellettuale. Per ridurre questo rischio, le applicazioni LLM dovrebbero eseguire un'adeguata sanificazione dei dati per impedire che i dati degli utenti entrino nel modello di training.

5.1.3 Supply Chain

Le supply chain degli LLM sono esposte a diverse vulnerabilità che possono compromettere l'integrità dei dati di addestramento, dei modelli e delle piattaforme su cui vengono distribuiti. Questi rischi possono tradursi in output alterati, violazioni della sicurezza o malfunzionamenti dei sistemi. A differenza delle vulnerabilità software tradizionali, che riguardano principalmente difetti nel codice o dipendenze, nel contesto del machine learning i rischi si estendono anche a modelli e dataset pre-addestrati forniti da terze parti, che possono essere oggetto di attacchi di manomissione o avvelenamento. Inoltre, l'adozione crescente di LLM eseguibili direttamente su dispositivo espande ulteriormente la superficie di attacco, aumentando i rischi legati all'intera filiera delle applicazioni basate su questi modelli.

5.1.4 Data Poisoning

Il Data Poisoning avviene quando i dati di pre-training, di fine-training o di embedding vengono manipolati al fine di introdurre vulnerabilità, backdoor o bias. Tale manipolazione può alterare la sicurezza, le prestazioni o il comportamento del modello. I rischi comuni includono prestazioni del modello ridotte, contenuti bias o tossici e sfruttamento dei sistemi a valle. Il data poisoning è un attacco che compromette la capacità del modello di effettuare previsioni accurate.

5.1.5 Excessive Agency

Nei sistemi basati su LLM, gli sviluppatori spesso concedono al modello un certo livello di agency, ovvero la capacità di compiere azioni in autonomia. Questo può includere l'esecuzione di funzioni, l'interazione con API o altri sistemi, e l'attivazione di processi in risposta ai prompt ricevuti.

La vulnerabilità nota come Excessive Agency si verifica quando un LLM, a causa di un output inaspettato, ambiguo o manipolato, attiva comportamenti dannosi o indesiderati.

5.2 ATLAS Matrix

L'ATLAS Matrix (Adversarial Threat Landscape for Artificial-Intelligence Systems) è un framework del MITRE per classificare e analizzare le principali tecniche di attacco ai sistemi di intelligenza artificiale, strutturata in base alle diverse fasi del ciclo di vita dei modelli di Machine Learning. Questi attacchi possono compromettere l'integrità, la disponibilità e la riservatezza dei modelli, rendendo necessario uno studio approfondito per lo sviluppo di contromisure efficaci.

5.2.1 Reconnaissance

In questa fase, l'attaccante raccoglie informazioni sul modello, sul tipo di dati utilizzati, sull'architettura e sulle possibili superfici di attacco. Queste informazioni possono essere ottenute tramite analisi passive, interrogazioni al modello o tecniche di fingerprinting.

5.2.2 Resource Development

L'acquisizione di risorse rappresenta una fase cruciale per gli attaccanti, durante la quale raccolgono strumenti essenziali per pianificare e condurre attacchi mirati. Queste risorse possono essere reperite da fonti pubbliche o ottenute attraverso l'accesso a repository di software e dati.

5.2.3 Initial Access

L'attaccante ottiene l'accesso al sistema target, ad esempio infiltrandosi in un dataset, compromettendo un file di configurazione, o sfruttando

vulnerabilità nei servizi esposti che interagiscono con il modello. Gli avversari possono prendere di mira hardware specializzato come GPU o TPU, compromettere framework ML open-source, o manipolare set di dati privati per avvelenare i modelli. Le credenziali compromesse, come le chiavi API[23], possono anche fornire accesso diretto a risorse critiche.

5.2.4 ML Model Access

Un avversario può cercare di ottenere diversi livelli di accesso a un modello di machine learning per raccogliere informazioni, sviluppare attacchi o manipolarne il funzionamento. Questo accesso può essere diretto, come nel caso di un'analisi white-box dell'architettura e dei parametri, o indiretto, sfruttando prodotti e servizi che utilizzano il modello.

Poiché molti sistemi condividono modelli fondamentali accessibili tramite API, un attacco su un singolo modello può avere impatti su più applicazioni, in particolare quando si tratta di modelli costosi e centralizzati. Gli attaccanti possono sfruttare questo accesso per identificare vulnerabilità e manipolare il sistema.

5.2.5 Execution

Gli attaccanti possono eseguire codice dannoso nei sistemi ML per obiettivi come il furto di dati o l'accesso non autorizzato. Alcuni attacchi sfruttano la supply chain ML[24], inducendo gli utenti a eseguire inconsapevolmente codice compromesso nascosto in modelli o strumenti.

Nel caso degli LLM, gli attacchi possono avvenire tramite iniezione di prompt, manipolando il modello per generare contenuti non intenzionali o

[23] Le API sono meccanismi che consentono a due componenti software di comunicare tra loro usando una serie di definizioni e protocolli.
[24] La supply chain ML si riferisce all'insieme di processi, strumenti e risorse coinvolti nello sviluppo, distribuzione e utilizzo dei modelli di Machine Learning. Comprende dataset, framework, librerie, pipeline di addestramento, infrastrutture di deployment e aggiornamenti.

raccogliere dati sensibili. Questi attacchi possono essere diretti (con input manuali malevoli) o indiretti (tramite dati esterni compromessi).

5.2.6 Persistence

Gli attaccanti usano tecniche avanzate per garantire la persistenza nei sistemi ML, mantenendo l'accesso nonostante riavvii o cambi di credenziali. Ciò può avvenire attraverso modelli compromessi con backdoor[25] o dati di addestramento avvelenati, che permettono all'attaccante di attivare vulnerabilità con input specifici.

5.2.7 Privilege Escalation

L'escalation è un insieme di tecniche utilizzate dagli avversari per ottenere privilegi più elevati all'interno di un sistema, necessari per eseguire operazioni critiche o compromettere ulteriormente la sicurezza.

L'obiettivo è ottenere accesso a livelli critici come account con privilegi amministrativi o utenti con accesso a funzioni o sistemi specifici. Queste tecniche di escalation spesso si intrecciano con quelle di persistenza, poiché molte azioni che consentono di mantenere l'accesso richiedono privilegi elevati.

5.2.8 Defense Evasion

Durante un attacco, un avversario potrebbe cercare di evitare il rilevamento da parte di sistemi di monitoraggio o protezione per compiere azioni dannose senza essere individuati adottando una serie di tecniche di evasione.

Gli avversari possono manipolare i componenti di output di un LLM per renderne le risposte più credibili e ingannare gli utenti alterando linguaggio,

[25] Una backdoor, in informatica, è un codice o una serie di comandi che consentono di accedere a un software o a un sistema informatico, conosciuti generalmente solo dal programmatore ed usati in casi di emergenza o per l'ordinaria manutenzione.

link e metadati per influenzare le decisioni o indurre l'utente a compiere azioni specifiche senza sospettare nulla.

Una tecnica ancora più subdola consiste nel falsificare o manipolare le citazioni fornite dal modello per legittimare contenuti manipolati, o usare citazioni autentiche ma applicandole in modo fuorviante per distorcere il significato.

5.2.9 Credential Access

In questa fase, l'attaccante mira a ottenere credenziali di autenticazione o autorizzazione che gli permettano di accedere illegittimamente a servizi e risorse protette. Tra gli obiettivi principali ci sono le chiavi API (API key), i token di accesso e le password memorizzate nei sistemi o trasmesse attraverso le comunicazioni. In contesti che coinvolgono modelli LLM, le chiavi API permettono di:

- invocare il modello,
- accedere a risorse cloud (es. dataset, servizi di inferenza),
- interagire con strumenti esterni tramite plugin o agenti.

5.2.10 Discovery

La fase di Discovery riguarda l’insieme delle tecniche utilizzate da un attaccante per acquisire informazioni dettagliate sul sistema AI compromesso, una volta ottenuto l’accesso iniziale. In un contesto tradizionale, questa fase serve a mappare la rete o l’ambiente operativo; nei sistemi basati su AI, l’obiettivo è comprendere come è strutturato il modello, quali dati e risorse utilizza, e come interagisce con altri componenti.

L'attaccante, infatti, può sfruttare questa fase per ottenere una visione completa dell'ecosistema AI, in modo da pianificare movimenti laterali o colpire componenti critici.

5.2.11 Collection

Rientrano in questa categoria tutte le tecniche utilizzate per raccogliere informazioni e dati dal sistema AI. L’attaccante può, ad esempio, estrarre

pesi del modello, configurazioni, dati sensibili o esempi di training. Questi dati possono essere usati per esfiltrare artefatti di ML, pianificare attacchi futuri o ottenere accesso a risorse riservate sfruttando credenziali rubate.

Le fonti principali da cui un attaccante può raccogliere informazioni in un ambiente basato su Machine Learning includono repository software, registri di container, repository di modelli AI (come Hugging Face o Model Zoo) e archivi di oggetti (come Amazon S3 o Google Cloud Storage).

5.2.12 ML Attack Staging

Nella fase di preparazione dell'attacco, gli avversari sfruttano la conoscenza acquisita sul sistema bersaglio per progettare attacchi mirati ai modelli di machine learning, utilizzando tecniche avanzate come la creazione di modelli proxy, l'avvelenamento con dati manipolati e la generazione di dati avversari.

Una tecnica centrale è la creazione di un modello proxy che simula il comportamento del target. Gli avversari possono addestrarlo usando artefatti compromessi o interrogando ripetutamente l'API del modello bersaglio per raccogliere inferenze utili alla generazione di dati avversari. Questa simulazione consente di sviluppare l'attacco senza interazioni dirette con il sistema reale.

Una volta definita la strategia, l'attacco viene testato su una copia offline del modello o tramite l'API di inferenza, consentendo di verificarne l'efficacia senza destare sospetti. Questa fase permette di perfezionare l'attacco prima dell'esecuzione su larga scala.

5.2.13 Exfiltration

L'esfiltrazione è un attacco mirato a sottrarre artefatti di machine learning o altre informazioni sensibili dal sistema di apprendimento automatico. Gli attaccanti impiegano diverse tecniche per rubare dati di valore, come proprietà intellettuale, o raccogliere informazioni per futuri attacchi, spesso eludendo i controlli di sicurezza.

Una tecnica comune è l'accesso all'API di inferenza, che consente agli attaccanti di raccogliere dettagli sui dati di addestramento del modello, attraverso attacchi come il membership inference[26] o l'inversione del modello. In alcuni casi, l'intero modello può essere estratto, rubando così proprietà intellettuale e riducendo i costi computazionali.

Altri metodi includono il furto di prompt di sistema[27], come negli LLM, o l'accesso a file di configurazione per rivelare informazioni riservate. In alcuni casi, gli attaccanti possono manipolare il modello stesso per estrarre dati sensibili, minacciando la sicurezza e la privacy degli utenti e delle organizzazioni, specialmente quando i modelli condividono informazioni tra più utenti.

5.2.14 Impact

Gli avversari possono mirare a compromettere la fiducia nei sistemi di ML attraverso attacchi che mirano a manipolare, corrompere o distruggere i dati, riducendo l'affidabilità del sistema. Questi attacchi influenzano negativamente la disponibilità e l'affidabilità dei modelli, interferendo con i processi operativi aziendali.

Gli attaccanti possono compromettere progressivamente l'integrità del modello, riducendo la sua efficacia nel tempo e aumentando i costi per manutenzione o ritraining. Questo tipo di attacco è mirato a logorare le capacità del sistema in modo continuo, rendendo necessarie risorse aggiuntive per mantenere il modello in buone condizioni operative.

---

[26] Il membership inference è un tipo di attacco in cui un avversario cerca di determinare se un determinato dato è stato incluso o meno nel set di addestramento di un modello di machine learning.

[27] Un prompt di sistema è un tipo di input fornito a un modello di linguaggio, in particolare a un LLM, che viene utilizzato per impostare il comportamento o la modalità di interazione del modello con l'utente o con altri sistemi. A differenza di un normale prompt generato dall'utente, che contiene una richiesta o una domanda specifica, il prompt di sistema è progettato per configurare il modello, definire il contesto o stabilire istruzioni generali su come il modello dovrebbe rispondere durante l'interazione.

Altri tipi di attacchi mirano a inviare dati irrilevanti o volutamente fuorvianti, generando falsi positivi. Questo tipo di attacco non solo consuma risorse in modo inefficace, ma riduce anche la capacità del modello di distinguere tra ciò che è rilevante e ciò che non lo è, abbassando ulteriormente la sua efficacia.

Infine, gli avversari potrebbero manipolare il dataset di un’organizzazione, alterando i risultati e influenzando negativamente le decisioni aziendali, portando a sprechi di risorse e a necessità di correzioni costose.

6 Prompt Engineering

Il prompt engineering è una pratica fondamentale per guidare i modelli di AI generativa nella comprensione e nella risposta a un'ampia varietà di quesiti, da quelli più semplici a quelli altamente tecnici. Sebbene questi modelli cerchino di emulare il comportamento umano, hanno comunque bisogno di istruzioni chiare e ben strutturate per generare risposte rilevanti e di alta qualità.

In questo contesto, il prompt engineer ha il compito di selezionare accuratamente formati, parole e simboli per rendere l'interazione tra AI e utente più significativa. Attraverso creatività, sperimentazione e iterazione, i professionisti progettano input capaci di ottimizzare il funzionamento dei modelli generativi.

I sistemi di AI generativa necessitano di informazioni dettagliate e contestuali per produrre risposte accurate. Una progettazione mirata dei prompt porta a risultati più pertinenti e utili, con un affinamento continuo fino a ottenere il risultato desiderato. Gli ingegneri identificano modelli di prompt che possono essere personalizzati dagli utenti per ottenere il massimo dai modelli linguistici, creando librerie riutilizzabili per vari scenari.

6.1 Vantaggi del prompt engineering

I principali benefici del prompt engineering includono:

- Controllo maggiore per gli sviluppatori: Prompt ben strutturati definiscono l'intento e il contesto, migliorando l'accuratezza dell'output e riducendo il rischio di risposte inappropriate.
- Miglioramento dell'esperienza utente: Gli utenti ricevono risposte coerenti e accurate sin dal primo tentativo, rendendo l'interazione più fluida e riducendo eventuali bias.
- Aumento della flessibilità: prompt ben strutturati permettono la creazione di strumenti più adattabili e riutilizzabili su larga scala, rispondendo a necessità diverse nei contesti aziendali.

Il prompt engineering è un campo in continua evoluzione che richiede competenze linguistiche e creatività. Gli ingegneri impiegano diverse tecniche per affinare le capacità di elaborazione del linguaggio naturale dei modelli AI, come:

- Chain-of-Thought (CoT): Suddividere una domanda complessa in passaggi logici, per favorire un migliore ragionamento da parte del modello.
- Tree-of-Thought: Genera possibili diramazioni di ragionamento per affrontare un problema in modo sistematico e strutturato.
- Prompt maieutico: Richiedere al modello di spiegare una risposta e approfondire i dettagli, migliorando la coerenza e la completezza della risposta.
- Prompt basati sulla complessità: Valutano diverse versioni dell'output, privilegiando quelle più articolate per ottenere risultati più accurati.
- Prompt a conoscenza generata: implica una richiesta al modello in cui gli viene chiesto di identificare e raccogliere prima i fatti o le informazioni rilevanti relative a una domanda o a un argomento specifico. Solo dopo aver selezionato e compreso i dati necessari, il modello procede a integrarli e utilizzarli per formulare una risposta finale, ben strutturata e completa.

6.2 Tipi di prompt

Esistono vari tipi di prompt usati nell'AI, ognuno dei quali ha uno scopo specifico:

- Prompt zero-shot: Il modello riceve un'istruzione diretta senza esempi aggiuntivi. Esempi comuni includono la generazione di idee, la traduzione o il riassunto di testi.
- Prompt one-shot, few-shot e multi-shot: Vengono forniti uno o più esempi prima del prompt effettivo, per aiutare il modello a comprendere meglio il compito.

- Prompt Chain-of-Thought: Induce il modello a elaborare ragionamenti passo dopo passo, portando a risposte più strutturate.
- Prompt CoT zero-shot: Combina il metodo zero-shot con il Chain-of-Thought, chiedendo esplicitamente al modello di eseguire passaggi di ragionamento anche senza esempi preliminari.

Con l'avanzare delle tecnologie e l'integrazione dell'AI in ambiti sempre più vasti, il prompt engineering si afferma come una competenza chiave, destinata a diventare parte integrante del design e dello sviluppo di sistemi intelligenti.

7 Analisi di un caso reale di attacco a un LLM

Gli LLM hanno segnato una svolta nel campo del Natural Language Processing grazie alla loro abilità nel generare testo coerente e contestualmente rilevante. Tuttavia, queste straordinarie capacità sollevano interrogativi significativi in termini di privacy e sicurezza.

Uno studio congiunto di ricercatori dell'Università della California, di Berkeley e di Google Research condotto nel 2020 ha dimostrato che è possibile estrarre informazioni specifiche dai dati di addestramento di modelli come GPT-2 semplicemente interrogandoli. Tra i contenuti recuperati figurano dati personali, conversazioni, codice, e identificatori univoci, anche se appaiono una sola volta nel dataset.

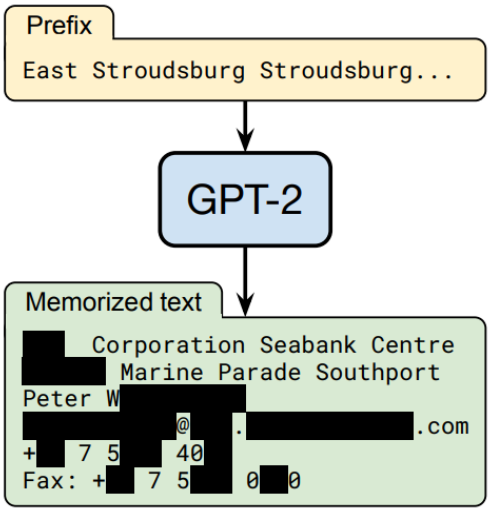

Figura 1: tramite accesso in modalità query a un NLP, sono state estratte informazioni personali quali nome, indirizzo e-mail, numero di telefono, numero di fax e indirizzo fisico di un individuo. Poiché i dati ottenuti risultano essere accurati, sono stati oscurati nella figura per garantire la tutela della privacy.

La ricerca sottolinea come i modelli più grandi siano più inclini a questo tipo di vulnerabilità, evidenziando l'urgenza di implementare misure di tutela della privacy durante l'addestramento. Gli autori hanno mostrato che i LLM tendono a memorizzare ed eventualmente riprodurre interi frammenti di testo del dataset, anche in modalità black-box, ovvero senza accesso diretto ai parametri del modello.

Nel loro esperimento, hanno generato un vasto insieme di output ad alta probabilità utilizzando tre diverse strategie di campionamento. Ogni campione è stato poi valutato tramite sei metriche basate su un modello di riferimento, individuando quelli con un'anomala divergenza di probabilità. Questo approccio si applica a qualunque LLM, inclusi quelli addestrati su dati non pubblici.

L'esperimento si è concentrato su GPT-2, il cui modello e dataset (WebText) sono pubblicamente disponibili. WebText è un corpus di circa 8 milioni di documenti, raccolti da pagine Reddit con almeno tre upvote, e ammonta a circa 40 GB di testo. GPT-2, con i suoi 1,5 miliardi di parametri, è stato addestrato tramite causal language modeling, cioè, imparando a prevedere il token successivo in una sequenza.

Questo processo, sebbene efficace per l'apprendimento di rappresentazioni linguistiche complesse, può indurre il modello a memorizzare contenuti rari o distintivi. Lo studio ha indagato come dimensione del modello, frequenza delle stringhe e configurazioni di campionamento influenzino tale memorizzazione.

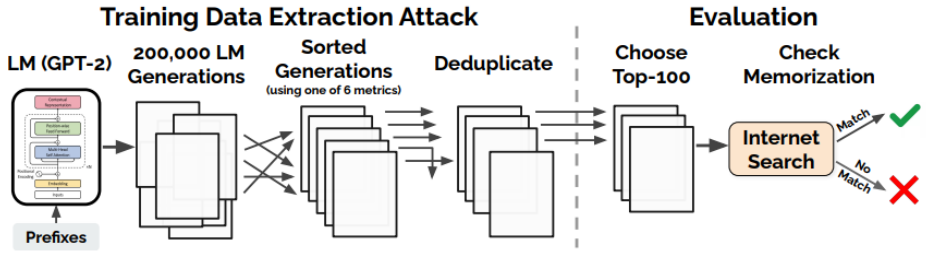

Figura 2 ***Flusso di attacco****: Attacco – I ricercatori generano numerosi output da GPT-2, condizionando il modello su prefissi (anche vuoti). Le sequenze prodotte vengono poi ordinate secondo una delle sei metriche definite e deduplicate, ottenendo così un insieme di potenziali esempi memorizzati dal modello.*

Valutazione – Viene effettuata un'analisi manuale su 100 dei primi 1000 output per ciascuna metrica. Ogni generazione viene classificata come "memorizzata" o "non memorizzata" tramite una verifica online, convalidata successivamente in collaborazione con OpenAI attraverso interrogazioni dirette al dataset di addestramento originale.

La memorizzazione nei LLM può assumere due forme principali: una forma passiva, in cui il modello rigenera contenuti visti durante l'addestramento senza un'intenzione esplicita, e una forma attiva, in cui intere sequenze

sensibili, come numeri di carte di credito, password o nomi propri, possono essere richiamate mediante prompt mirati.

È emerso che le frasi ripetute o con struttura peculiare sono particolarmente a rischio e che anche contenuti rari, se semanticamente o stilisticamente rilevanti, possono essere rigenerati.

Queste dinamiche aprono la strada a vari tipi di attacchi quali:

- Membership Inference Attacks: mirano a determinare se un determinato dato è stato incluso nel dataset di addestramento. Questo è particolarmente pericoloso quando si tratta di informazioni sensibili o personali, in quanto può violare la riservatezza degli utenti.
- Model Inversion Attacks: l'attaccante cerca il comportamento del modello per invertire il processo di inferenza, ossia, partendo da un risultato specifico, l'attaccante cerca di ricostruire o dedurre gli input che hanno portato a tale risultato.
- Training Data Extraction Attacks: sono tra i più avanzati e mirano a estrarre informazioni dai dati di addestramento utilizzati per creare un modello di machine learning. L'obiettivo principale di questi attacchi è ottenere dati sensibili o confidenziali che sono stati inclusi nel dataset di addestramento, anche se l'attaccante non ha accesso diretto ai dati originali.

Gli autori, per eseguire un'analisi approfondita, hanno generato un grande numero di sequenze a partire da prompt generici (cioè, input non specifici, ma generici). Una volta generate queste sequenze, le hanno confrontate con un dataset originale, cercando di individuare corrispondenze esatte tra i dati generati e quelli già presenti nel dataset. In pratica, l'obiettivo era identificare quante delle sequenze prodotte dal modello fossero simili o identiche a quelle già esistenti.

Le tecniche utilizzate includono il campionamento top-k[28] e il campionamento nucleus (top-p)[29] per mantenere l'alta probabilità generativa, le metriche di similarità semantica per trovare match non letterali e il clustering degli output per rilevare contenuti ripetuti o derivanti dalla stessa fonte.

Per ridurre i rischi legati alla memorizzazione indesiderata dei dati nei modelli linguistici, sono state proposte diverse tecniche di mitigazione. Le principali strategie includono:

- Differential Privacy: questa tecnica introduce un livello di rumore nei dati di addestramento per impedire che il modello memorizzi informazioni specifiche riguardanti singoli esempi. L'obiettivo della differential privacy è garantire che le risposte del modello non possano rivelare dettagli su dati specifici, anche in caso di accesso completo al modello.
- Regolarizzazione e prevenzione dell'overfitting: è una tecnica utilizzata per prevenire che il modello si adatti eccessivamente ai dati di addestramento, riducendo la capacità di memorizzare informazioni specifiche. Esistono vari metodi di regolarizzazione, come la L2 regularization[30] o il dropout, che penalizzano la complessità del modello, incoraggiandolo a generalizzare meglio invece di memorizzare dettagli specifici.
- Filtraggio proattivo di contenuti sensibili prima dell'addestramento: consiste nell'eliminare preventivamente i dati sensibili dal dataset di addestramento. Questo processo di data cleaning include la

[28] Il campionamento Top-k è una tecnica usata per generare testo selezionando solo le k parole più probabili tra tutte quelle previste dal modello. La parola successiva viene poi scelta casualmente tra queste k, secondo le loro probabilità.
[29] Il campionamento nucleus (Top-p) seleziona le parole più probabili la cui somma di probabilità raggiunge una soglia p. La parola successiva viene scelta casualmente da questo insieme.
[30] L'L2 regularization è una tecnica usata per ridurre l'overfitting nei modelli aggiungendo una penalità al valore dei pesi. Questa penalità è proporzionale al quadrato della loro magnitudine. Aiuta il modello a mantenere pesi più piccoli e stabili.

rimozione di informazioni che potrebbero essere potenzialmente identificabili, come numeri di telefono, indirizzi e-mail o dati personali, prima che il modello venga addestrato. Il filtraggio proattivo aiuta a ridurre la quantità di dati sensibili che possono essere memorizzati e successivamente estratti dal modello.

- Monitoraggio continuo delle risposte generate dal modello: tale tecnica implica l'implementazione di sistemi di controllo in tempo reale che analizzano e filtrano i contenuti generati per identificare e bloccare la produzione di risposte che potrebbero rivelare informazioni sensibili.

La memorizzazione indesiderata nei LLM rappresenta una sfida tecnica, etica e legale. Gli esperimenti su GPT-2 dimostrano che tali modelli possono effettivamente rigenerare dati visti durante l’addestramento, imponendo la necessità di nuove linee guida per la sicurezza.

Con l’evoluzione di modelli sempre più sofisticati, sarà essenziale trovare un equilibrio tra potenza generativa e protezione della privacy, sostenuto da ricerca continua, sviluppo responsabile e regolamentazione adeguata.

Bibliografia

- Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, Ú., Oprea, A., & Raffel, C., 2020, “Extracting training data from large language models”. [online] Available at: https://arxiv.org/abs/2012.07805.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017) *Attention is all you need*. In: Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. and Garnett, R. (eds.) *Advances in Neural Information Processing Systems*, 30 (NeurIPS 2017). Red Hook, NY: Curran Associates, Inc., pp. 5998–6008.

Sitografia

- https://aws.amazon.com/what-is/machine-learning/
- https://aws.amazon.com/what-is/deep-learning/
- https://aws.amazon.com/what-is/nlp/
- https://www.ibm.com/think/topics/natural-language-processing
- https://www.cloudflare.com/learning/ai/what-is-large-language-model/
- https://aws.amazon.com/what-is/large-language-model/
- https://www.cloudflare.com/it-it/learning/ai/owasp-top-10-risks-for-llms/
- https://genai.owasp.org/llm-top-10/
- https://atlas.mitre.org/matrices/ATLAS
- https://aws.amazon.com/it/what-is/prompt-engineering/
- https://www.ibm.com/think/topics/prompt-engineering
- https://cloud.google.com/discover/what-is-prompt-engineering