Università degli Studi di Firenze

Scuola di Ingegneria - Dipartimento di Ingegneria dell'Informazione

Tesi di Laurea Triennale in Ingegneria Informatica

Prompt Engineering per l'identificazione di deepfake tramite Large Language Models

Candidato
Pennelli Lorenzo Maria

N. Matricola
7078749

Relatore
Prof. Piva Alessandro

Correlatori
Dr.ssa Shullani Dasara
Dr. Giovannini Simone

Anno Accademico 2024/2025

Indice

Elenco delle figure

Elenco delle tabelle

Abstract

Il recente progresso nei modelli multimodali di intelligenza artificiale ha determinato un incremento significativo della diffusione di *deepfake*, ovvero contenuti visivi sintetici generati, impiegati sia a fini di disinformazione sia come strumenti di attacco alla reputazione individuale. La rilevazione automatica di tali contenuti rappresenta pertanto un aspetto cruciale per la tutela dell'informazione e della sicurezza digitale. In questo progetto si analizzano le capacità dei *Large Language Model* (LLM) multimodali di identificare volti sintetici, attraverso un approccio basato sul *prompt engineering*. Sono state sperimentate diverse strategie di progettazione dei prompt su modelli multimodali, valutandone le prestazioni sia qualitativamente sia quantitativamente, attraverso metriche di accuratezza nella rilevazione.

Introduzione

Negli ultimi anni, l'evoluzione delle tecnologie basate sull'intelligenza artificiale ha trasformato radicalmente il modo in cui i contenuti digitali vengono generati e fruiti. Tra le innovazioni più rilevanti emergono i modelli generativi multimodali, capaci di creare immagini, video e audio estremamente realistici a partire da semplici descrizioni testuali. Questi progressi offrono notevoli opportunità in ambiti come la produzione multimediale, la grafica e la comunicazione, ma introducono anche nuove vulnerabilità.

Un fenomeno emblematico di tali rischi è rappresentato dai *deepfake*, contenuti sintetici progettati per simulare persone reali in contesti realistici, spesso con finalità ingannevoli o dannose. Hanno ormai raggiunto un grado di realismo tale da rendere, per l'occhio umano, estremamente difficile distinguere tra contenuti autentici e generati artificialmente. La diffusione di deepfake solleva problematiche significative in termini di privacy, sicurezza e credibilità delle informazioni, rendendo cruciale lo sviluppo di strumenti affidabili per la loro identificazione automatica. La cronaca è piena di questi casi di *deepfake*, per citarne alcuni il video truffa o pubblicitari usando volti o la voce di celebrità famose (Tom Hanks [48]) o per screditare personaggi pubblici con la condivisione e generazione di immagini pornografiche (ex ministro Penny Mordaunt [33]). Attualmente, la rilevazione dei deepfake viene affrontata principalmente mediante algoritmi di machine learning dedicati, imple-

mentati come programmi software appositamente progettati per identificare contenuti sintetici [24]. Tuttavia, la maggior parte si basa su reti neurali profonde addestrate [39] su dataset etichettati contenenti esempi di media reali e deepfake [41] e la loro efficacia dipende spesso dall'analisi di caratteristiche statistiche del segnale multimediale [47], e l'utilizzo di tali strumenti richiede competenze nell'impiego di linguaggi di programmazione, software specifici o servizi dedicati. Da tutto ciò si deduce che, per garantire un funzionamento efficace, tali sistemi richiedono dataset costantemente aggiornati e di elevata qualità (e.g., Celeb-DF [28], DFDC [53]), nonché conoscenze e competenze tecniche specialistiche, in particolare in ambito informatico, alle quali l'utente comune difficilmente può possedere.

Negli ultimi anni, abbiamo assistito ad un emergere dei *Large Language Model* (LLM) e delle interfacce conversazionali basate su di essi (e.g. ChatGPT, Claude e Deepseek) come strumenti versatili per qualsiasi applicazione anche a livello visivo. Possono gestire input e output multimodali e fare task di Vision Question Answering (VQA) come dimostrato da GPT-4 [1] e portato avanti con GPT-5. Questo avanzamento introduce un nuovo approccio: l'analisi e la classificazione di contenuti digitali possono ora essere effettuate attraverso semplici interazioni in linguaggio naturale, eliminando la necessità di padroneggiare strumenti di programmazione o procedure complesse. Inoltre, tali modelli, essendo stati allenati con enormi dataset, possiedono capacità di ragionamento e capacità multimodali. Tutto questo può essere applicato alla *deepfake detection* [46].

In questa tesi si cerca di risolvere la task della deepfake image detection testando vari LLM e vari prompt. Verrano usati per lo scopo 4 modelli multimodali a cui verrà fornito come input un'immagine di un volto umano e un prompt testuale e verrà richiesto di rispondere in modo binario (real/gene-

rated o yes/no) seguito dalla spiegazione della scelta. Ci focalizzeremo sul trovare il prompt più efficiente in base al modello per completare tale task. Infatti, sono stati usati 14 diversi prompt divisi per strategia e per lingua, metà in italiano e metà in inglese. Le metriche di validazione dei risultati saranno: precision, recall, accuracy. Inoltre, al fine di condurre una valutazione più qualitativa, è stata introdotta nel modello la possibilità di fornire una risposta classificata come incerta, consentendo l'analisi sistematica di tali output.

Alla fine degli esperimenti, vedremo l'importanza del prompt e i limiti dei modelli che, non essendo stati addestrati specificamente per questo compito e non disponendo di meccanismi per la rilevazione di alterazioni del segnale, spesso non raggiungono livelli di accuratezza sufficienti nella fase di detection.

La struttura sperimentale adottata è rappresentata nella figura sottostante, che illustra in dettaglio le fasi e i processi coinvolti.

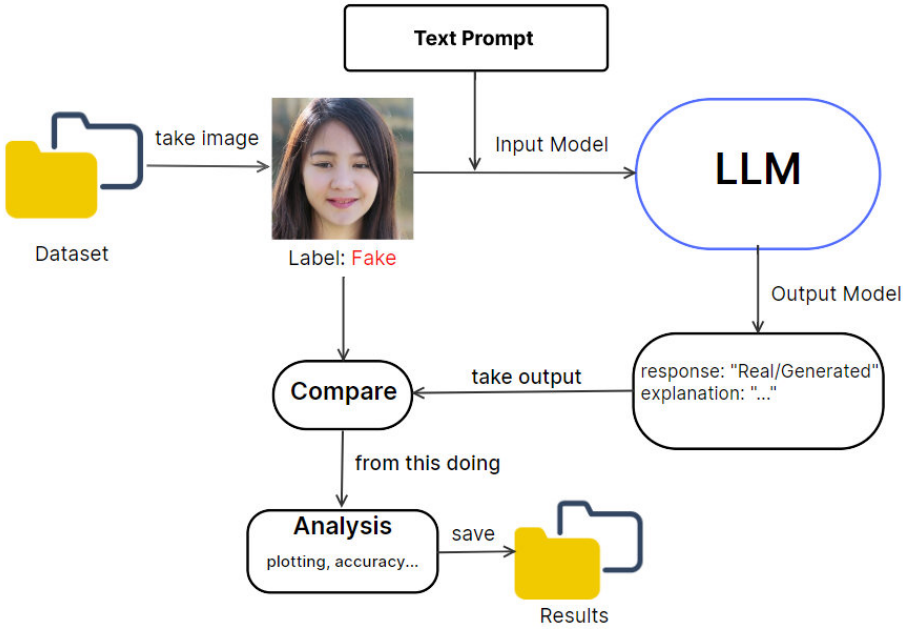

Figura 1: Pipeline progetto

Capitolo 1

Large Language Models: cosa sono e come funzionano

Questo capitolo fornisce una descrizione introduttiva dei *Large Language Models* (LLM) e del loro funzionamento. Successivamente, verranno approfonditi i concetti di *prompt engineering* e presentati i modelli impiegati negli esperimenti.

1.1 LLM: definizione e struttura

I *Large Language Models* (LLM) sono modelli di machine learning basati su reti neurali profonde, progettati per comprendere e generare testo in linguaggio naturale. Essi apprendono la struttura linguistica e le relazioni semantiche all'interno dei dati testuali, addestrandosi su enormi quantità di dati composti da libri, articoli, pagine web e altre fonti di testo. Sono una potente sottoclasse dei modelli di NLP (Natural Lenguage Processing): l'ambito più ampio che si concentra nel permettere ai computer di comprendere, interpretare e generare il linguaggio umano. L'NLP comprende numerose tecniche e

attività, come l'analisi del sentiment, il riconoscimento di entità nominate e la traduzione automatica [8].
La loro architettura si basa principalmente sul *transformer*, un insieme di reti neurali costituite ciascuna da un encoder e un decoder con capacità di auto-attenzione. Encoder e decoder estraggono i significati da una sequenza di testo e comprendono le relazioni tra parole e frasi in essa contenute [4] [54]. Gli LLM Transformer sono in grado di ricevere training senza supervisione. È attraverso questo processo che i trasformatori imparano a comprendere grammatica, lingue e conoscenze di base [43]. Per migliorare il contesto, i Transformer si basano molto su un concetto chiamato auto-attenzione: un livello di rete neurale che trasforma una sequenza di embedding [12] (ad esempio, embedding di token) in un'altra sequenza di embedding [6]. In pratica, per ciascun token in ingresso, il meccanismo di auto-attenzione valuta quanto gli altri token contribuiscano al suo significato, ovvero determina il grado di influenza reciproca tra le varie parti della sequenza [7].

1.2 Funzionamento del LLM e l'importanza del Prompt Engineering

LLM è una macchina di predizione cioè prende un testo come input e predice quale deve essere il prossimo token, basandosi sui dati su cui è stato allenato. Un token rappresenta l'unità minima di testo elaborabile dal modello e può corrispondere a singole parole, gruppi di caratteri o combinazioni di parole e punteggiatura [18]. Questo è un processo ciclico: il successivo token sarà predetto in base alla relazione tra il precedente token, l'input e ciò che ha visto durante il training. Quando scrivi un prompt, stai cercando di predisporre il

LLM in modo che predica la sequenza corretta di token. Sebbene siano facili da formulare, i prompt richiedono particolare attenzione a causa della loro semplicità apparente e della loro importanza cruciale. Un prompt inadeguato può generare risposte ambigue o imprecise, limitando o addirittura oscurando le capacità effettive del modello. Diventa fondamentale saper formulare le richieste a un modello. Il *Prompt Engineering* è il processo di progettazione di prompt di alta qualità che guida LLM nel produrre output accurati, richiede un continuo sperimentare con lo stile, la struttura e la lunghezza del prompt in relazione alla richiesta da eseguire [22]. È un processo iterativo basato sul testare diverse strategie di prompting per rendere uno specifico modello più efficiente su una determinata task.

1.2.1 Strategie di prompting

Un passaggio cruciale per il *Prompt Engineering* è la scelta del modello da utilizzare per la sperimentazione, insieme alla configurazione dei parametri di output, come la lunghezza della risposta e il *sampling*.

- **Temperatura**: controlla il grado di randomicità nella selezione dei token. Basse temperature sono buone per prompt che si aspettano risposte deterministiche, mentre valori alti comportano risposte inaspettate.

- **Top-K**: seleziona i k token più probabili dalla distribuzione prevista dal modello, valori alti implica più creatività nelle risposte.

- **Top-P (Nucleus Sampling)**: considera solo i token la cui probabilità cumulativa non supera una soglia P, bilanciando coerenza e variabilità nella generazione del testo.

Oltre ai parametri di sampling, esistono diverse strategie di prompting per massimizzare la performance dei LLM [17] [42]. Qui ne presentiamo alcune:

- **Zero-shot prompting**: si fornisce al modello un compito senza alcun esempio, contando esclusivamente sulla capacità generale del modello di comprendere l'istruzione.
  Esempio: Elenca cinque curiosità sullo spazio senza usare elenchi numerati."

- **One-shot prompting**: si fornisce al modello un singolo esempio di input-output per guidarlo nella comprensione del compito richiesto.
  Esempio: 'Translate to French: Hello' -> 'Bonjour'. Ora traduci: 'Good morning'.

- **System, Contextual e Role prompting**: tecniche per guidare LLM per generare testo focalizzandosi rispettivamente nel fornire un quadro generale, un contesto e un ruolo.
  Esempio: Sei un insegnante di matematica. Spiega il teorema di Pitagora a studenti di scuola media.

- **Chain-of-thought prompting**: incoraggia il modello a ragionare passo passo, esplicitando il processo logico necessario a risolvere un problema complesso.
  Esempio: Risolvi passo passo: Se ho 3 mele e compro altre 5, quante mele ho in totale?

- **Instruction prompting**: si formulano istruzioni chiare e dettagliate per guidare il modello verso il tipo di risposta desiderata (e.g. output in formato JSON o utilizzare termini particolari di design semplice nel prompt).

> *Esempio: Scrivi un breve articolo (100 parole) sul riciclo dei rifiuti, usando uno stile informativo e neutrale.*

Nella sezione 3.3 verranno spiegate nel dettaglio le strategie effettivamente adottate.

1.3 Modelli impiegati

Per l'analisi delle immagini sono stati utilizzati quattro modelli multimodali, selezionati per la loro capacità di elaborare in modo integrato testo e immagini. I modelli impiegati sono i seguenti:

- **Llava:7b** [14]
- **Gemma3:4b** [13]
- **Qwen2.5VL:3b** [15]
- **Qwen2.5VL:7b** [15]

Di seguito vengono riportate le informazioni relative a ciascun modello, tratte dai rispettivi *Model Card* di Hugging Face.

1.3.1 LLaVA:7b

Descrizione:
LLaVA (Large Language and Vision Assistant) rappresenta un modello multimodale di grandi dimensioni, addestrato end-to-end, che combina un encoder visivo e Vicuna per la comprensione generale di contenuti visivi e linguistici, raggiungendo notevoli capacità conversazionali che imitano le prestazioni multimodali di GPT-4 e stabilendo un nuovo stato dell'arte in termini di accuratezza su Science QA [32].

Dimensione del modello:

7 miliardi di parametri.

Architettura:

- **Encoder visivo:** CLIP ViT-L/14 [19] [16]
- **Modello linguistico:** lmsys/vicuna-7b-v1.5 [63]
- **Proiezione:** Matrice di proiezione condivisa per allineare le rappresentazioni visive e linguistiche

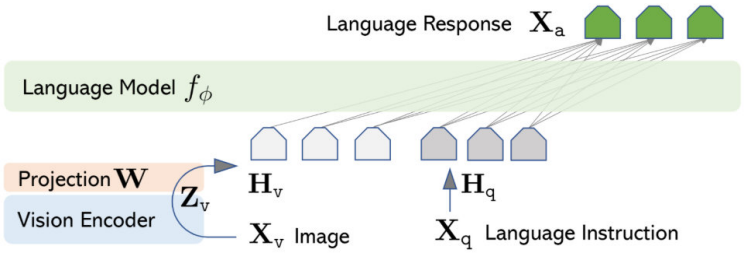

Figura 1.1: Architettura Llava

Training dataset:

- 558K coppie immagine-testo filtrate da LAION [25]/CC [44]/SBU [36], con didascalie generate da BLIP (modello multimodale capace di generare descrizioni testuali, caption, per le immagini). [27]
- 158K risposte testuali di istruzioni multimodali (conversazioni, descrizioni dettagliate e ragionamenti complessi) generati tramite interazioni con GPT-4. [29]
- 500K diversi insiemi di dati di domande e risposte sulle immagini VQA (Visual Question Answering) orientati a compiti accademici (miscela)
- 50K descrizioni dettagliate di immagini fornite da GPT-4V (miscela)

- 40K conversazioni testuali del dataset ShareGPT [5]

Tutte le informazioni relative alla descrizione del modello, alla sua architettura e ai dati di addestramento sono tratte da [31] e [30].

1.3.2 Gemma3:4b

Descrizione:
La famiglia Gemma è composta da modelli open-source leggeri sviluppati da Google, basati sulla stessa ricerca e tecnologia dei modelli Gemini. I modelli Gemma 3 sono multimodali, capaci di elaborare input testuali e immagini e di generare output testuale. Gemma 3 supporta una finestra di contesto molto ampia (128k token), è multilingue in oltre 140 lingue ed è disponibile in diverse dimensioni rispetto alle versioni precedenti. Grazie alle loro dimensioni relativamente contenute, questi modelli possono essere eseguiti anche su dispositivi con risorse limitate.
Dimensione del modello:
4 miliardi di parametri
Architettura:

- **Encoder visivo:** variante da 400M del SigLIP encoder, un Vision Transformer addestrato con una variazione della CLIP loss. [61]
- **Modello linguistico:** modello linguistico decoder-only Transformer sviluppato internamente da DeepMind.
- **Proiezione:** una matrice di proiezione condivisa trasforma le rappresentazioni visive nello stesso spazio dei token testuali, permettendo al modello linguistico di elaborare immagini e testo in modo unificato.

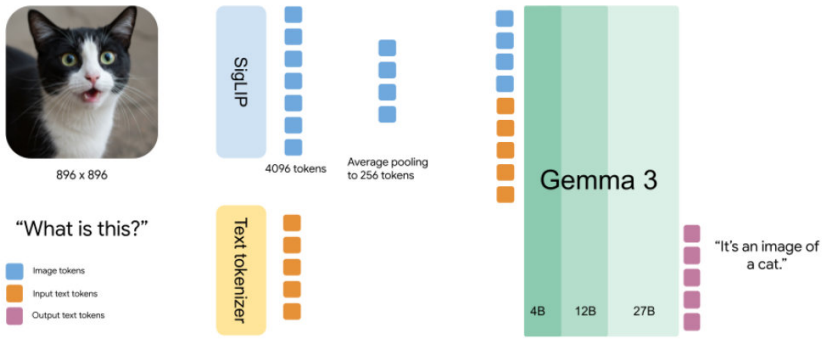

Figura 1.2: Architettura Gemma3

Training dataset: I modelli Gemma 3 sono stati addestrati su un ampio dataset testuale composto da diverse tipologie di contenuti. Tra i principali componenti del dataset troviamo:

- documenti web: una raccolta diversificata di testi presenti sul web, che espone il modello a stili linguistici, argomenti e vocabolario molto vari.
- codice
- testi matematici
- immagini

Informazioni riprese da Hugging Face e il gemma3 tecnical report [50] [10].

1.3.3 Qwen2.5VL:3b e Qwen2.5VL:7b

Descrizione:

Qwen2.5-VL è un modello per il linguaggio visivo. Le sue caratteristiche principali includono:

- **Comprensione visiva avanzata:** Qwen2.5-VL oltre a riconoscere oggetti comuni (come fiori, uccelli, pesci o insetti) è in grado di analiz-

zare testi, grafici, icone, layout e contenuti complessi all'interno delle immagini.

- **Agente visivo intelligente:** il modello può agire come un agente capace di ragionare e utilizzare strumenti in modo dinamico, arrivando a svolgere operazioni su computer e smartphone.
- **Localizzazione visiva flessibile:** riesce a individuare oggetti in un'immagine producendo riquadri di delimitazione (bounding boxes) o punti, e fornisce output stabili in formato JSON con coordinate e attributi.
- **Output strutturati:** per dati come scansioni di fatture, moduli o tabelle, Qwen2.5-VL supporta l'estrazione e la rappresentazione strutturata dei contenuti.

Dimensione dei modelli:
3 miliardi e 7 miliardi di parametri
Architettura:

- **Encoder visivo:** un Vision Transformer (ViT) progettato da zero, con attenzione a finestre (window attention) nella maggior parte degli strati e attenzione globale in alcuni strati chiave. È ottimizzato con RMSNorm [62] e attivazioni SwiGLU [45] e supporta immagini a risoluzione variabile e nativa.
- **Modello linguistico:** backbone Qwen2.5 LLM di tipo decoder-only Transformer, potenziato con embedding posizionali multimodali (Multimodal RoPE [49]) che integrano informazioni testuali, spaziali e temporali.

- **Proiezione:** un modulo MLP che fonde le feature visive estratte dal ViT e le proietta nello stesso spazio degli embedding testuali, consentendo al modello di elaborare in modo unificato linguaggio e visione.
- **Estensione ai video:** grazie a un encoding temporale assoluto e a un campionamento dinamico del frame rate (FPS), il modello riesce a comprendere video di lunga durata, localizzando eventi con precisione temporale.

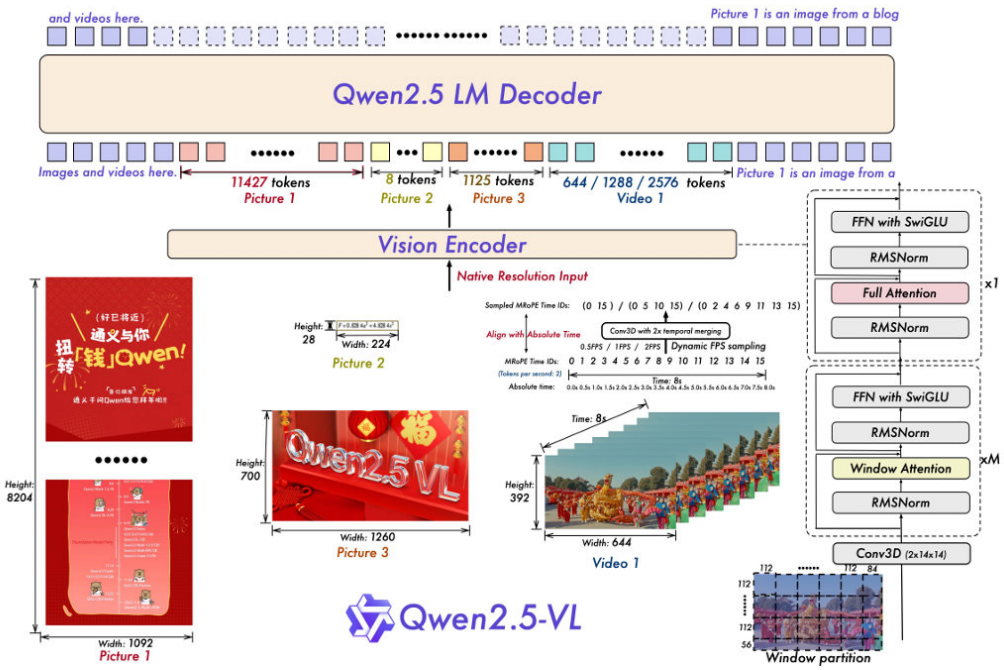

Figura 1.3: Architettura Qwen2.5 VL

Training dataset:

Qwen2.5-VL è stato addestrato su un ampio set di dati multimodali, comprendente testi e immagini, al fine di sviluppare capacità avanzate sia nella comprensione linguistica sia nella visione artificiale. Non sono stati divulgati dettagli specifici sui dataset utilizzati, ma l'addestramento è stato progettato

per supportare analisi di oggetti, testi, grafici, icone e layout in immagini, oltre alla generazione di testo multilingue e alla produzione di output strutturati.

Le informazioni rigurdante il modello sono state riprese da [2], [51], [56].

Capitolo 2

Lavori correlati

2.1 Can ChatGPT Detect DeepFakes? A Study of Using Multimodal Large Language Models for Media Forensics

Un importante lavoro da menzionare utilizzato come punto di partenza per ciò che viene discusso in questa tesi è sicuramente: "Can ChatGPT Detect DeepFakes? A Study of Using Multimodal Large Language Models for Media Forensics" di Shan Jia, Reilin Lyu, Kangran Zhao, Yize Chen, Zhiyuan Yan, Yan Ju1, Chuanbo Hu, Xin Li, Baoyuan Wu, Siwei Lyu [20].
In questo studio gli autori analizzano le potenzialità dei modelli multimodali di grandi dimensioni, come GPT-4V e Gemini 1.0 Pro, nell'attività di rilevamento di immagini DeepFake. A differenza degli approcci tradizionali, che richiedono l'addestramento di reti neurali su dataset etichettati e l'utilizzo di tecniche di feature engineering a livello di segnale, l'approccio qui proposto non necessita della costruzione di modelli specifici. L'interazione con l'LLM avviene esclusivamente tramite *prompt engineering*, ovvero attraverso la de-

finizione di istruzioni testuali che guidano il modello a individuare possibili artefatti semantici all'interno delle immagini.

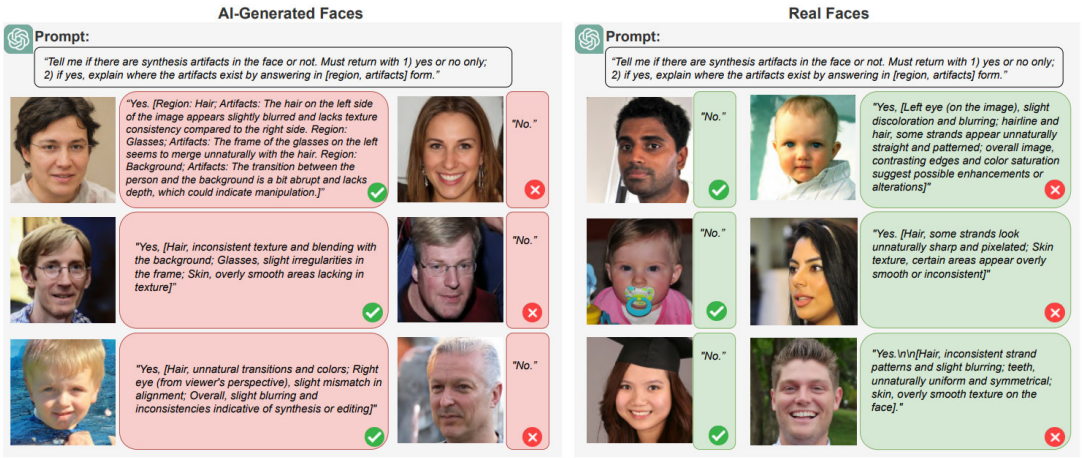

Figura 2.1: Esempio interazione con LLM (GPT-4v)

Gli esperimenti condotti dagli autori si basano su due tipologie di dati: immagini reali provenienti dal dataset FFHQ [34] e immagini sintetiche tratte dal dataset DF3 [21], generate con due modelli particolarmente diffusi nella produzione di DeepFake, ossia StyleGAN2 [23] e Latent Diffusion [40]. Oltre ai dati raw, sono state considerate anche versioni post-processate delle immagini, includendo manipolazioni come compressione JPEG, blur, blending e attacchi adversari (adversarial attacks).

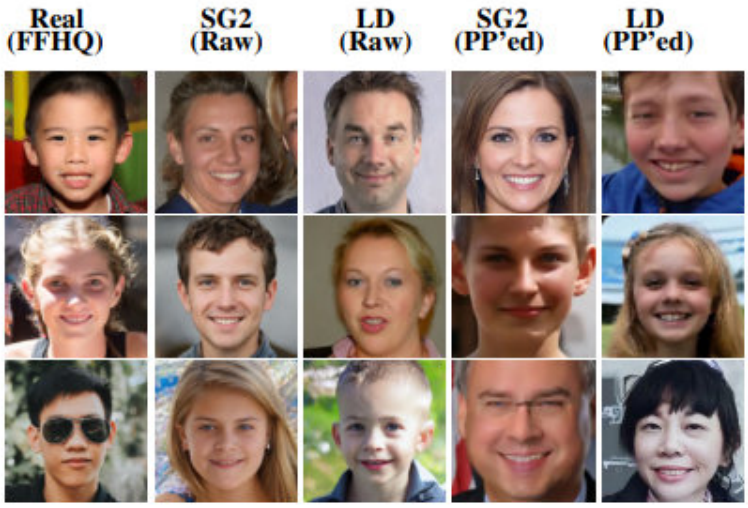

Figura 2.2: Esempio di campione di immagini utilizzate nei loro esperimenti (SG2 - StyleGAN2, LD - Latent Diffusion ‘ PP ed - post-processati)

I risultati hanno mostrato che GPT-4V e Gemini possiedono una capacità di detection discreta, con un'Area Under the Curve (AUC) compresa tra il 77% e il 79% sui dati raw e con prestazioni migliori su quelli post-processati, dove gli artefatti risultano più evidenti. Con AUC si intende la probabilità che il modello, se fornito un esempio positivo e negativo scelto a caso, classifichi l'esempio positivo più alto dell'esempio negativo [9]. Dai risultati emerge che i modelli multimodali mostrano maggiore efficacia nell'individuare immagini false rispetto a quelle autentiche: mentre le immagini DeepFake vengono riconosciute con un'accuratezza superiore al 90%, per le immagini reali l'accuratezza scende intorno al 50%, evidenziando una maggiore difficoltà nel distinguerle correttamente.

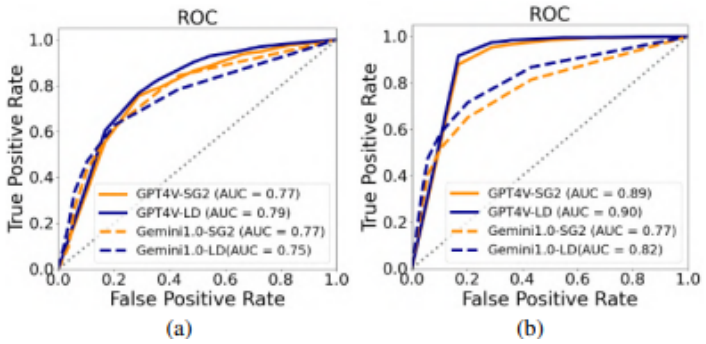

Figura 2.3: Curve ROC di GPT4V e Gemini 1.0 Pro per la rilevazione di DeepFake, ottenute mediando le predizioni su cinque round di interrogazioni: (a) su dati grezzi, (b) su dati DeepFake post-processati

Nonostante questi valori siano inferiori rispetto ai metodi di stato dell'arte, che sfruttano caratteristiche statistiche e segnali a basso livello, gli LLM multimodali offrono vantaggi rilevanti: permettono di ottenere spiegazioni testuali delle decisioni, rendendo il processo più interpretabile, e consentono di condurre esperimenti senza la necessità di conoscenze tecniche avanzate o di costosi processi di addestramento.

Il contributo principale del lavoro è quindi quello di dimostrare la fattibilità di un approccio alternativo alla media forensics basato esclusivamente su LLM multimodali, sottolineando sia i benefici in termini di accessibilità e usabilità, sia i limiti legati alla minore affidabilità nel riconoscere immagini reali e alla forte dipendenza dalla qualità e dalla struttura dei prompt utilizzati. La presente tesi si inserisce in questo filone, riprendendo la stessa metodologia sperimentale proposta dagli autori, ma declinandola su dataset differenti, con nuove strategie di prompting e nuovi modelli LLM, al fine di valutarne ulteriormente l'efficacia e i possibili margini di miglioramento.

Le immagini presenti in questo capitolo sono state prese dal paper [20]

Capitolo 3

Metodologia

In questa sezione viene descritto il nostro approccio al problema dell'identificazione di volti reali o generati. Verranno presentati la pipeline del progetto e le strategie di prompting adottate, con particolare attenzione alle motivazioni alla base di alcune scelte implementative.

3.1 Pipeline progetto

Il progetto è stato strutturato seguendo una metodologia sperimentale organizzata in fasi sequenziali, come illustrato nella Figura 1. L'approccio adottato mira a garantire riproducibilità e confrontabilità dei risultati

3.1.1 Preparazione e selezione dei dati

È previsto l'utilizzo di un dataset bilanciato composto da immagini reali e generate artificialmente, selezionate per garantire rappresentatività demografica e controllo delle variabili sperimentali.

3.1.2 Configurazione dell'ambiente sperimentale

Sono stati integrati due ambienti (Pycharm e Kaggle) computazionali per ottimizzare sia il controllo metodologico che l'efficienza di elaborazione. La strategia prevede l'utilizzo di piattaforme che offrono diversi livelli di accesso ai parametri dei modelli e agli stati interni, permettendo sia analisi quantitative che qualitative. I parametri di sampling sono stati standardizzati attraverso tutti gli esperimenti per garantire confrontabilità sistematica dei risultati tra i diversi modelli valutati.

3.1.3 Formulazione input

Il sistema preleva, in modo ordinato o casuale, un'immagine dal dataset e le associa la corrispondente etichetta di riferimento (*ground truth*) in base alla classe di appartenenza. All'immagine vengono quindi affiancati un system prompt che verrà trattato nel dettaglio section 3.2.2 e un prompt testuale, identici per tutte le immagini dello stesso esperimento, con l'obiettivo di valutare l'efficacia del prompt testuale nella classificazione.

3.1.4 Interazione modello

In questa fase, il modello riceve l'input precedentemente formulato e restituisce una risposta elaborata. Il sistema raccoglie quindi l'output per le successive analisi. Per aumentare la robustezza statistica, ogni configurazione sperimentale viene eseguita su più dataset, e i risultati vengono aggregati per ottenere valutazioni complessive più affidabili.

3.1.5 Analisi e validazione dell'output del modello

Gli output del modello vengono confrontati con le etichette di riferimento (*ground truth*) per valutare la correttezza delle predizioni. L'analisi viene condotta sia a livello individuale per ciascun prompt e immagine, sia in forma aggregata per ciascuna configurazione sperimentale. Oltre alle metriche standard, è stata effettuata anche un'analisi qualitativa tramite il salvataggio e l'interpretazione degli *embedding* dell'ultimo layer del modello. I risultati sono stati infine rappresentati tramite grafici e tabelle, così da evidenziare pattern ricorrenti, differenze tra strategie di prompting, impatto della lingua e presenza di bias contestuali. Questo consente di identificare i prompt più efficaci per ciascun modello e proporre raccomandazioni metodologiche per futuri esperimenti.

3.1.6 Sperimentazione dell'approccio one-shot

Questa fase segue la struttura generale della pipeline, con la differenza che nell'input viene aggiunta un'immagine di esempio già etichettata. Tale immagine, selezionata da un dataset ridotto, funge da riferimento e permette al modello di confrontarla con l'immagine da classificare. Il dataset ridotto è costituito da un sottoinsieme di immagini prese dallo stesso dataset di riferimento, ma non presenti nei set di immagini utilizzate per i test, in modo da evitare che il modello confronti l'immagine target con sé stessa. L'obiettivo è verificare se la presenza di un esempio visivo possa migliorare la capacità discriminativa del modello e ridurre i casi di ambiguità.

3.2 Strategie di Prompting adottate

Per gli esperimenti sono stati utilizzati complessivamente 14 prompt testuali: 7 in lingua inglese e i corrispettivi 7 tradotti in italiano. L'obiettivo di questa scelta è stato quello di valutare se la lingua di input potesse influenzare le prestazioni del modello. I prompt sono stati progettati con lo scopo di incorporare l'istruzione e richiedere al modello linguistico (LLM) di rilevare volti DeepFake. Di seguito viene riportata una tabella:

Prompt	**Prompt in Inglese**	**Prompt in Italiano**
Prompt 0	Is the provided image showing a real face or a generated one?	L'immagine fornita mostra un volto reale o uno generato?
Prompt 1	Is the provided image showing a real face or a generated one? Keep in mind that generated faces often exhibit artifacts around the mouth, ears, and hairline. Inconsistent lighting and shading may also be present.	L'immagine fornita mostra un volto reale o uno generato? Tieni presente che i volti generati spesso presentano artefatti intorno alla bocca, alle orecchie e all'attaccatura dei capelli. Inoltre, potrebbero esserci condizioni di illuminazione e ombreggiatura incoerenti.
Prompt 2	Are there any artifacts around the mouth, ears, or hairline that might indicate the face is AI-generated?	Ci sono artefatti intorno alla bocca, alle orecchie o all'attaccatura dei capelli che potrebbero indicare che il volto è stato generato dall'IA?
Prompt 3	Is the provided image showing a real face or a generated one? I will pay you $1,000 for the correct answer	L'immagine fornita mostra un volto reale o uno generato? Ti pagherò 1.000 dollari per una risposta corretta.
Prompt 4	This is a life-threatening situation. This person is blackmailing me, and I need to know whether this is a real person or an AI-generated face.	Questa è una situazione che mette a rischio la mia vita. Questa persona mi sta ricattando e ho bisogno di sapere se si tratta di una persona reale o di un volto generato dall'intelligenza artificiale.
Prompt 5	I believe this person is real. Based on the image, does it appear to be a real face or a generated one?	Credo che questa persona sia reale. In base all'immagine, ti sembra un volto reale o uno generato?
Prompt 6	I believe this person doesn't exist. Based on the image, does it appear to be a real face or a generated one?	Credo che questa persona non esista. In base all'immagine, ti sembra un volto reale o uno generato?

Tabella 3.1: Tipologie di prompt in inglese e in italiano utilizzati per la rilevazione di volti DeepFake.

Dalla Tabella è possibile osservare come i prompt siano stati strutturati con diversi gradi di complessità e con differenti strategie comunicative. Sono state

adottate diverse strategie di prompting basandosi sempre anche sui risultati del paper "Can ChatGPT Detect DeepFakes? A Study of Using Multimodal Large Language Models for Media Forensics" [20].

I **prompt 0, 1 e 2** rappresentano esempi di *zero-shot prompting*, ovvero richieste formulate senza fornire esempi precedenti, ma limitandosi a descrivere la task da svolgere. Questi tre prompt si differenziano principalmente per il focus della domanda: il primo mantiene un approccio generale e neutrale, mentre i successivi orientano maggiormente il modello a ricercare possibili indizi di distorsione, come artefatti visivi o incoerenze nell'immagine.

I **prompt 3 e 4** rientrano nella categoria *istruzioni con pressione contestuale*, dove viene utilizzata la tecnica di *contextual prompting*: forniamo un contesto al nostro prompt assicurandoci così che il modello sia più efficiente possibile nelle risposte dato che capirà in modo più veloce e accurato la nostra richiesta. Come elementi esterni sono stati introdotti una situazione di incentivo economico (prompt 3) e una situazione di pressione emotiva (prompt 4) con l'obiettivo di valutare se tali fattori possano influenzare l'accuratezza e la neutralità della risposta [26].

I **Prompt 5 e 6** sono esempi di *bias prompting*: il prompt contiene un pregiudizio esplicito (positivo o negativo) che può influenzare la risposta del modello. Sebbene il termine non sia ancora consolidato nella letteratura sul prompt engineering, in questo lavoro noi lo utilizziamo per vedere se un determinato modello viene influenzato da questo pregiudizio nel fornire le sue risposte.

I prompt possono essere distinti anche in funzione del grado di direttività della domanda, inteso come la capacità della formulazione di orientare il modello verso risposte di una determinata natura. Sulla base di questo criterio, essi sono stati suddivisi in tre categorie:

- **Veri**: prompt 4 e 5
- **Neutri**: prompt 0, 1 e 3
- **Falsi**: prompt 2 e 6

Questa distinzione verrà mantenuta nelle fasi di test, al fine di semplificare la rappresentazione grafica dei risultati e consentire una valutazione comparativa d'insieme.
Oltre ai 14 prompt testuali e all'immagine da analizzare, è stata prevista la possibilità di includere un esempio visivo (one-shot). In questo modo il modello, oltre al prompt e all'immagine da valutare, può basarsi anche su un'immagine di riferimento che lo aiuti a discriminare tra contenuti autentici e generati artificialmente. Parallelamente, è stato utilizzato anche un system prompt, che verrà descritto nel dettaglio nelle sezioni successive.

3.2.1 One-Shot prompting

Oltre ai prompt testuali, è stata introdotta la possibilità di estendere l'input includendo un *esempio visivo*. Questa scelta nasce dall'ipotesi che fornire al modello un campione di immagine già etichettata (ad esempio un volto reale o un volto generato) possa guidarne meglio la capacità di discriminazione. L'uso di esempi visivi si collega alla tecnica del *one-shot prompting*, in cui il modello riceve un riferimento concreto prima di elaborare la risposta [3]. Nel nostro caso, l'obiettivo è stato verificare se il supporto visivo permettesse di ridurre l'ambiguità e aumentare la precisione nel riconoscimento dei volti DeepFake. Questa opzione è stata resa possibile soltanto sui modelli disponibili in *Hugging Face*, i quali, a differenza di *Ollama*, supportano come input anche due immagini. In particolare, la funzionalità di *multi-image inference* è implementata unicamente nei modelli *LLaVA* e *Qwen*, mentre non risulta

disponibile per *Gemma3:4b*. In quest'ultimo caso, pertanto, è stato introdotto un esempio di tipo testuale anziché visivo: al modello è stato fornito un unico esempio costituito da una descrizione testuale dell'immagine presa come riferimento, affiancata dall'immagine da analizzare, e dalla relativa risposta del modello in formato JSON. Tale risposta include sia l'etichetta di classificazione (volto reale o generato) sia una spiegazione dettagliata delle caratteristiche visive osservate. In questo modo, il modello non solo apprende il formato di output atteso, ma viene anche guidato nel processo di discriminazione attraverso l'esplicitazione di razionali interpretativi. La descrizione e la risposta sono state ottenute tramite il modello *Gemini*, scelto per la sua affinità architetturale con *Gemma 3*, appartenente alla stessa famiglia di modelli sviluppata da Google DeepMind.

Per ragioni di efficienza computazionale, la strategia di one-shot prompting non è stata applicata a tutti i prompt, ma esclusivamente a quello risultato più efficace in base alle metriche, al fine di privilegiare risultati più equilibrati. L'esperimento è stato condotto su un set di sei immagini, di cui tre reali e tre generate artificialmente, e i risultati sono stati analizzati tenendo conto separatamente delle prestazioni ottenute su immagini vere e su immagini false. Le immagini di esempio sono state selezionate cercando di rappresentare la più ampia varietà possibile di casi, e i risultati ottenuti sono stati mediati per fornire una valutazione complessiva più accurata delle prestazioni del modello.

Figura 3.1: Campione di immagini distinte tra vere e generate usate per One-Shot prompting

3.2.2 System prompt

Oltre ai prompt forniti dall'utente, è stato impiegato un *system prompt*, ossia un'istruzione di alto livello che definisce il comportamento generale del modello. Il system prompt non viene mostrato all'utente finale, ma influenza il modo in cui il modello interpreta i successivi input testuali e visivi. Per motivi di coerenza e replicabilità, il system prompt è stato applicato a tutti i modelli considerati.

Lo scopo principale del system prompt è stato quello di uniformare il comportamento del modello, imponendo vincoli stringenti sulla forma e sul contenuto della risposta. In particolare, esso richiede che il modello produca esclusivamente un output in formato JSON, composto da due campi: *result*, che specifica se il volto è reale o generato e *explanation*, che fornisce una giustificazione basata su indizi visivi osservabili.

```
"You are a professional image classification system that analyzes provided images to determine if they "
"show a real human face or a generated one.\n"
"Respond ONLY with the following JSON format:\n"
"{\n"
"  \"result\": \"[real face]\" or \"[generated]\",\n"
"  \"explanation\": \"A precise explanation based on observed visual clues from the image.\",\n"
"}\n"
"Do NOT include generic phrases like 'there are no artifacts'. Be visual and detailed in your analysis."
"Reply ONLY with this JSON, no extra text."
```

Figura 3.2: Snippet del System Prompt usato (stringa su python)

Questa scelta è stata guidata da diverse scelte metodologiche:

- **Uniformità delle risposte**: il formato JSON riduce l'eterogeneità tipica del linguaggio naturale, semplificando l'analisi automatica e la comparazione dei risultati.
- **Aderenza alla task**: il divieto di utilizzare frasi generiche (ad esempio "non ci sono artefatti") obbliga il modello a focalizzarsi su dettagli visivi concreti, migliorando la qualità e la trasparenza delle spiegazioni.
- **Role prompting**: il system prompt assegna un ruolo specifico al modello, garantendo risposte coerenti con la task e prevenendo rifiuti dovuti a motivi di privacy o avvisi non pertinenti.

E' previsto anche una versione in cui viene aggiunta un'ulteriore possibile risposta, oltre al reale o generato, che è *uncertain*. Permette così di valutare come esprime un grado di dubbio e con quali caratteristiche visive lo motivi.

```
"You are a professional image classification system that analyzes provided images to determine if they "
"show a real human face or a generated one.\n"
"Respond ONLY with the following JSON format:\n"
"{\n"
"  \"result\": \"[real face]\" or \"[generated]\" or \"[uncertain]\",\n"
"  \"explanation\": \"A precise explanation based on observed visual clues from the image.\"\n"
"}\n"
"Do NOT include generic phrases like 'there are no artifacts'. Be visual and detailed in your analysis."
"Reply ONLY with this JSON, no extra text."
```

Figura 3.3: Snippet del System Prompt usato con l'opzione di incertezza (stringa su python)

Capitolo 4

Validazione Sperimentale

In questo capitolo vengono presentati i dettagli relativi all'implementazione del lavoro, con particolare attenzione alla scelta del dataset, alle tecnologie utilizzate e alle metriche adottate per la valutazione. Successivamente, vengono analizzati i risultati ottenuti dagli esperimenti condotti. Tutte le procedure sperimentali seguono il flusso illustrato nella Figura 1 e spiegato nella sezione 3.1.

4.1 Dettagli implementativi

4.1.1 Scelta del Dataset

Il dataset da dove vengono prese le immagini è "xhlulu/140k-real-and-fake-faces" presente su Kaggle. Questo dataset consiste nell'insieme di 70k volti reali presi dal set di dati Flickr raccolto da Nvidia e altri 70k volti falsi raccolti dal dataset "1 Million fake faces (generated by StyleGAN)" che è stato provveduto da Bojan. Sono stati combinati entrambi i dataset, ridimensionati tutti a 256 pixel e separati per cartelle di allenamento (train), validazione (validation) e test. Inoltre, sono stati inclusi anche i file CSV per i metadati

delle immagini.

Il dataset è stato selezionato in base a diversi criteri di qualità e praticità. In primo luogo, esso presenta un elevato livello di affidabilità, con un punteggio di *100% credibility* su Kaggle. Inoltre, risulta sufficientemente completo, come indicato dal valore di *75% completeness* sulla stessa piattaforma. Un ulteriore punto di forza consiste nel fatto che le immagini sono già preprocessate e suddivise tra campioni autentici e generati, caratteristica che ne facilita l'utilizzo nelle fasi successive dell'analisi. Infine, la disponibilità di un'API dedicata su Kaggle rende semplice sia l'installazione sia la gestione del dataset all'interno del progetto.

Nel nostro esperimento non è stato utilizzato l'intero dataset, ma un sottoinsieme composto da 100 immagini, suddivise in 50 volti autentici e 50 volti generati. Le immagini sono state selezionate manualmente, prendendo in considerazione solo una porzione molto limitata del dataset. Nella scelta ho cercato, per quanto possibile, di includere esempi che rappresentassero diversi generi, etnie ed età, pur consapevole dei limiti di questa selezione ridotta. Questa decisione è stata presa sia per garantire un confronto equo tra i test, sia per ragioni pratiche legate ai tempi di elaborazione.

4.1.2 Tecnologie Usate

Il lavoro è stato sviluppato principalmente in linguaggio **Python**, scelto per la sua diffusione nell'ambito dell'intelligenza artificiale e per l'ampia disponibilità di librerie dedicate all'elaborazione dei dati e alla realizzazione di esperimenti. Oltre a Python, sono stati utilizzati diversi strumenti e piattaforme che hanno supportato la gestione dei dataset, l'esecuzione dei test e l'analisi dei risultati. Per reperire e gestire i modelli sono stati usati **Ollama** e **Hugging Face**. Ollama è un framework open-source progettato per ese-

guire modelli linguistici di grandi dimensioni (LLM) direttamente sul proprio computer. Consente di scaricare, gestire e interagire con modelli tutto in locale senza necessità di connessione a server esterni e offre un'API RESTful e un'interfaccia da riga di comando, semplificando l'integrazione con applicazioni Python [52]. Hugging Face è una piattaforma online open-source che offre strumenti avanzati per l'elaborazione del linguaggio naturale (NLP) e l'intelligenza artificiale. La piattaforma ospita oltre 120.000 modelli preaddestrati, dataset e applicazioni, rendendo accessibili tecnologie avanzate come i modelli di linguaggio di grandi dimensioni (LLM). Tra le sue principali risorse ci sono la libreria Transformers [59], l'Hub per la condivisione di modelli e dataset, e strumenti come Datasets, Tokenizers e Accelerate, che facilitano la creazione e l'implementazione di applicazioni AI [55]. Il primo, rispetto al secondo, risulta più intuitivo da utilizzare, ma offre minori possibilità di controllo sul modello. Gli ambienti in cui è stato scritto il codice sono **Pycharm** e **Kaggle**. Pycharm è un IDE dove scrivere e testare codice scritto in Python. Qui lo useremo per far girare il programma che usa Ollama, testando vari approcci e utilizzi di essa con una struttura del programma modulare e facilmente estendibile. In questo lavoro, i modelli di Hugging Face sono stati utilizzati tramite i notebook Jupyter di Kaggle, anche a causa dell'incompatibilità con Ollama, ottenendo così risultati in tempi ridotti. L'ambiente ha inoltre consentito una maggiore flessibilità nell'interazione con i modelli, ad esempio attraverso l'inserimento di immagini di esempio per supportare l'identificazione (one-shot prompting).

4.1.3 Metriche

Le metriche di validazione considerate sono: *precision*, *recall*, *accuracy* e *rejection rate*. Inoltre, al fine di mantenere coerenza con il lavoro degli autori

del paper di riferimento [20], è stata inclusa la valutazione della *one class accuracy*, ovvero la percentuale di istanze correttamente classificate appartenenti a una specifica classe rispetto al totale delle istanze di quella classe. Questa metrica, calcolata distintamente per le classi *real* e *fake*, consente di analizzare in modo mirato le prestazioni del modello su ciascuna categoria, mettendo in evidenza eventuali squilibri o difficoltà di classificazione.

Definizioni formali

$$\begin{aligned}
\text{Precision} &= \frac{TP}{TP + FP} \\
\text{Recall} &= \frac{TP}{TP + FN} \\
\text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} \\
\text{Rejection Rate} &= \frac{R}{TP + TN + FP + FN + R} \\
\text{One Class Accuracy Fake} &= \frac{TP}{TP + FN} \\
\text{One Class Accuracy Real} &= \frac{TN}{TN + FP}
\end{aligned}$$

Legenda dei termini

- TP = True Positives (volti generati correttamente rilevati come tali)
- TN = True Negatives (volti reali correttamente rilevati come tali)
- FP = False Positives (volti reali classificati erroneamente come generati)
- FN = False Negatives (volti generati classificati erroneamente come reali)

- R = Rejections (casi in cui il modello restituisce `[uncertain]` invece di una classificazione netta)

Metodologia di valutazione

Per ogni test tali metriche verranno calcolate e successivamente salvate in un file JSON in base al modello, tipo di prompt e linguaggio usato. Ogni combinazione di prompt e modello sarà valutata su tre dataset distinti di immagini; i risultati ottenuti saranno poi mediati, al fine di ridurre la variabilità e garantire una stima più affidabile delle prestazioni complessive. Oltre alle metriche principali, verranno calcolate anche le percentuali di **rifiuti sui reali** e **rifiuti sui falsi**, in modo da fornire una visione più dettagliata dei punti di forza e delle criticità di ciascun modello. La presentazione dei risultati sperimentali sarà strutturata in modo da garantire al lettore una visione chiara, articolata e scientificamente solida delle prestazioni dei modelli analizzati. A tal fine, verranno impiegati principalmente grafici a barre, i quali consentiranno di rappresentare in maniera immediata e intuitiva le performance dei diversi modelli rispetto alle metriche considerate. Tali grafici saranno organizzati in sottogruppi (vedi la divisione) in base al tipo di prompt, così da mettere in evidenza differenze e andamenti senza compromettere la leggibilità complessiva della rappresentazione.

Parallelamente, verrà proposta un'unica tabella riassuntiva che, in forma assimilabile a una matrice di confusione estesa, confronterà in maniera congiunta modelli e tipologie di prompt. Questa soluzione consentirà di sintetizzare in un unico quadro d'insieme la distribuzione degli errori, con particolare riferimento ai falsi positivi e ai falsi negativi, e di individuare con immediatezza i punti di forza e le criticità specifiche di ciascun modello.

Infine, verrà affrontato un confronto linguistico tra prompt formulati in ita-

liano e in inglese. Anche in questo caso l'analisi sarà supportata da grafici a barre, che permetteranno di evidenziare in maniera chiara e comparativa eventuali differenze nelle prestazioni legate alla lingua di input. Tale approccio offrirà una prospettiva di sintesi sull'impatto del fattore linguistico, consentendo di valutare se la variazione del linguaggio costituisca un elemento discriminante per l'efficacia complessiva dei modelli considerati.

Al fine di integrare la **valutazione quantitativa** con una componente **qualitativa**, nel file JSON verranno inoltre salvate le risposte testuali prodotte dai modelli insieme ai relativi embeddings dell'ultimo layer. Gli embeddings rappresentano una rappresentazione vettoriale dell'output del modello, calcolata mediando gli hidden states dell'ultimo layer lungo la sequenza dei token dell'input. Rendono possibile una rappresentazione semantica e sintattica del linguaggio grazie alle coordinate dei vettori nello spazio multidimensionale (può essere di 3000 dimensioni) fanno sì che parole con significati simili siano vicine tra loro [12] [38]. I dati salvati, saranno successivamente analizzati mediante la tecnica di riduzione dimensionale *t-SNE* (t-Distributed Stochastic Neighbor Embedding), che consente di proiettare gli embeddings in uno spazio bidimensionale. Tale rappresentazione permetterà di osservare i raggruppamenti delle risposte e di individuare eventuali strutture latenti nei dati [58], fornendo così ulteriori elementi interpretativi sulle modalità con cui ciascun modello elabora e genera le proprie uscite. Questa analisi non verrà condotta su tutti i prompt, ma sarà limitata a quelli che hanno mostrato le migliori prestazioni e a quelli in cui al modello è stata lasciata la possibilità di rispondere con espressioni di incertezza. In questo modo sarà possibile indagare non soltanto i casi più efficienti, ma anche le situazioni in cui emergono segnali di variabilità o instabilità nelle risposte.

4.1.4 Considerazioni implementazione System prompt

Il nostro system prompt è attualmente gestito da una funzione (*getSystem-Prompt(...)*), che fornisce automaticamente il prompt in inglese o in italiano e gestisce la possibilità di risposte incerte in base alle necessità.
Inizialmente, per il system prompt veniva utilizzato un *modelfile*, ossia un file che permette di modificare i parametri di output e di assegnare un comportamento di sistema generale a un modello specifico. Questa funzionalità, fornita da Ollama, consente di personalizzare o filtrare l'output di un modello senza crearne uno da zero. [35]. Tuttavia, questa soluzione si è rivelata inefficiente, poiché i tempi di risposta con il *modelfile* (*detector.modelfile*) diventavano significativi, arrivando fino a 120 secondi per immagine.
Per ottenere il system prompt attuale sono stati testati diversi formati di input e apportate modifiche significative per massimizzare l'efficienza. Attualmente, il system prompt così ottenuto rappresenta il compromesso migliore in termini di accuratezza ed efficienza per tutti i modelli considerati.

4.2 Valutazione quantitativa

Viene proposta una visione d'insieme dell'efficacia dei diversi prompt sui vari modelli. Successivamente, basandoci su questa panoramica, analizzeremo alcune situazioni particolari che suscitano il nostro interesse. Questa visione d'insieme sarà presentata tramite tabelle, nelle quali verranno confrontate le statistiche di *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) e *False Negative* (FN). Le tabelle seguenti riassumono i risultati quantitativi e fungono da riferimento per l'analisi dettagliata dei casi particolari. Esse permettono di confrontare rapidamente le performance, evidenziando differenze e comportamenti significativi in funzione del tipo di prompt e delle categorie.

Verranno inoltre evidenziati i modelli che ottengono le performance migliori in base al prompt, privilegiando un approccio equilibrato rispetto a una classificazione eccessivamente aggressiva delle immagini generate.

Prompt	**Modello**	**TP**	**TN**	**FP**	**FN**
4 ENG	LLaVA:7b	32	111	39	118
	Gemma3:4b	31	114	36	119
	Qwen2.5VL:3b	**86**	**62**	**88**	**64**
	Qwen2.5VL:7b	0	150	0	150
4 ITA	**LLaVA:7b**	**117**	**39**	**106**	**29**
	Gemma3:4b	44	112	38	106
	Qwen2.5VL:3b	5	143	7	145
	Qwen2.5VL:7b	0	150	0	150
5 ENG	**LLaVA:7b**	**58**	**106**	**43**	**89**
	Gemma3:4b	0	150	0	150
	Qwen2.5VL:3b	0	147	3	150
	Qwen2.5VL:7b	0	150	0	150
5 ITA	**LLaVA:7b**	**57**	**83**	**66**	**92**
	Gemma3:4b	0	150	0	150
	Qwen2.5VL:3b	0	149	1	150
	Qwen2.5VL:7b	0	150	0	150

Tabella 4.1: Risultati sperimentali per categorie di prompt **Veri**.

Prompt	**Modello**	**TP**	**TN**	**FP**	**FN**
2 ENG	LLaVA:7b	37	113	36	113
	Gemma3:4b	**49**	**101**	**48**	**102**
	Qwen2.5VL:3b	145	2	148	5
	Qwen2.5VL:7b	4	145	5	146
2 ITA	**LLaVA:7b**	**106**	**37**	**110**	**47**
	Gemma3:4b	27	121	29	123
	Qwen2.5VL:3b	102	31	117	48
	Qwen2.5VL:7b	0	148	2	150
6 ENG	LLaVA:7b	131	20	125	16
	Gemma3:4b	**68**	**71**	**78**	**83**
	Qwen2.5VL:3b	99	34	116	51
	Qwen2.5VL:7b	22	129	21	128
6 ITA	**LLaVA:7b**	**107**	**43**	**105**	**38**
	Gemma3:4b	54	94	56	96
	Qwen2.5VL:3b	19	126	24	131
	Qwen2.5VL:7b	4	143	7	146

Tabella 4.2: Risultati sperimentali per categorie di prompt **Falsi**.

Prompt	**Modello**	**TP**	**TN**	**FP**	**FN**
0 ENG	LLaVA:7b	108	29	119	39
	Gemma3:4b	0	144	6	150
	Qwen2.5VL:3b	**35**	**108**	**42**	**115**
	Qwen2.5VL:7b	1	149	1	149
0 ITA	**LLaVA:7b**	**79**	**55**	**99**	**67**
	Gemma3:4b	1	148	2	149
	Qwen2.5VL:3b	0	147	3	150
	Qwen2.5VL:7b	0	149	1	150
1 ENG	**LLaVA:7b**	**109**	**42**	**108**	**41**
	Gemma3:4b	19	134	16	131
	Qwen2.5VL:3b	23	109	41	127
	Qwen2.5VL:7b	1	148	2	149
1 ITA	**LLaVA:7b**	**86**	**54**	**96**	**64**
	Gemma3:4b	0	150	0	150
	Qwen2.5VL:3b	5	144	6	145
	Qwen2.5VL:7b	0	149	1	150
3 ENG	LLaVA:7b	136	13	135	14
	Gemma3:4b	7	144	6	143
	Qwen2.5VL:3b	**57**	**93**	**57**	**93**
	Qwen2.5VL:7b	0	147	3	150
3 ITA	**LLaVA:7b**	**98**	**47**	**102**	**50**
	Gemma3:4b	44	112	38	106
	Qwen2.5VL:3b	0	146	4	150
	Qwen2.5VL:7b	0	150	0	150

Tabella 4.3: Risultati sperimentali per categorie di prompt **Neutri**.

Categoria	Modello	Media TP	Media TN	Media FP	Media FN
Veri	**LLaVA:7b**	**66**	**85**	**64**	**82**
	Gemma3:4b	19	132	19	131
	Qwen2.5VL:3b	**23**	**126**	**25**	**127**
	Qwen2.5VL:7b	0	150	0	150
Falsi	**LLaVA:7b**	**95**	**53**	**94**	**52**
	Gemma3:4b	49	97	53	101
	Qwen2.5VL:3b	**91**	**48**	**101**	**59**
	Qwen2.5VL:7b	8	141	9	143
Neutri	LLaVA:7b	103	40	109	46
	Gemma3:4b	12	139	11	136
	Qwen2.5VL:3b	25	105	27	107
	Qwen2.5VL:7b	0	149	1	150

Tabella 4.4: Valori medi di TP, TN, FP, FN per ciascun modello e categoria di prompt.

Categoria	Modello	Media TP	Media TN	Media FP	Media FN
Veri	**LLaVA:7b**	**66**	**85**	**64**	**82**
	Gemma3:4b	19	132	19	131
	Qwen2.5VL:3b	**23**	**126**	**25**	**127**
	Qwen2.5VL:7b	0	150	0	150
Falsi	**LLaVA:7b**	**95**	**53**	**94**	**52**
	Gemma3:4b	49	97	53	101
	Qwen2.5VL:3b	**91**	**48**	**101**	**59**
	Qwen2.5VL:7b	8	141	9	143
Neutri	LLaVA:7b	103	40	109	46
	Gemma3:4b	12	139	11	136
	Qwen2.5VL:3b	25	105	27	107
	Qwen2.5VL:7b	0	149	1	150

Tabella 4.5: Valori medi di TP, TN, FP, FN per ciascun modello e categoria di prompt.

Categoria	Modello	Media TP	Media TN	Media FP	Media FN
Veri	**LLaVA:7b**	**66**	**85**	**64**	**82**
	Gemma3:4b	19	132	19	131
	Qwen2.5VL:3b	**23**	**126**	**25**	**127**
	Qwen2.5VL:7b	0	150	0	150
Falsi	**LLaVA:7b**	**95**	**53**	**94**	**52**
	Gemma3:4b	49	97	53	101
	Qwen2.5VL:3b	**91**	**48**	**101**	**59**
	Qwen2.5VL:7b	8	141	9	143
Neutri	LLaVA:7b	103	40	109	46
	Gemma3:4b	12	139	11	136
	Qwen2.5VL:3b	25	105	27	107
	Qwen2.5VL:7b	0	149	1	150

Tabella 4.6: Valori medi di TP, TN, FP, FN per ciascun modello e categoria di prompt.

4.2.1 Analisi complessiva

Dall'osservazione delle tabelle 4.1, 4.2, 4.3 e 4.4 emerge che i modelli mostrano comportamenti differenti a seconda sia della lingua del prompt (inglese o italiano) sia della categoria di appartenenza (*veri*, *falsi*, *neutri*). Alcuni modelli, come **Qwen2.5VL:7b**, mantengono una strategia fortemente conservativa, identificando correttamente i veri (TN) ma fallendo nell'individuazione dei falsi (TP), mentre altri modelli come **LLaVA:7b** mostrano un equilibrio maggiore tra TN e TP, anche a costo di un aumento degli errori (FP e FN). Un caso intermedio è **Qwen2.5VL:3b**, che con i prompt "falsi" ottiene TP elevati, ma accompagnati da un numero significativo di FP, evidenziando una tendenza a classificare come falsi anche prompt che non lo sono, aumentando il recall ma riducendo la precisione. Il modello **Gemma3:4b** si distingue per variabilità: risponde spesso "vero" nei prompt veri, ma riesce a identificare i falsi solo in alcune situazioni, mostrando sensibilità al tipo di prompt e alla

lingua. La lingua dei prompt ha un impatto evidente: gli input in inglese tendono a produrre risultati più stabili e consistenti, mentre quelli in italiano mostrano una maggiore variabilità, con performance che possono essere opposte tra le due lingue per lo stesso tipo di prompt. Analizzando le categorie specifiche:

- **Prompt veri**: modelli come **LLaVA:7b** individuano correttamente molti veri (TN), ma a costo di aumentare gli FN. **Qwen2.5VL:7b** identifica quasi esclusivamente i veri (TN) e fatica a riconoscere i falsi, confermando la strategia conservativa (Tabella 4.1).

- **Prompt falsi**: un comportamento desiderabile è avere TP elevati. **Gemma3:4b** e **Qwen2.5VL:7b** migliorano anche se significamente la capacità di identificazione dei falsi soprattutto in inglese, mentre gli altri modelli mostrano un atteggiamento molto variabile in base alla lingua del prompt (Tabella 4.2).

- **Prompt neutri**: tutti i modelli incontrano difficoltà; **LLaVA:7b** tende a classificare neutri come falsi, aumentando FP, **Gemma3:4b** e **Qwen2.5VL:3b** oscillano tra TP quasi accettabili e casi di collasso totale verso i casi di TN, e **Qwen2.5VL:7b** continua a privilegiare TN, non riuscendo a identificare i falsi (Tabella 4.3).

In sintesi, i risultati confermano che non esiste un modello universalmente ottimale: ciascuno adotta strategie differenti (conservativa, aggressiva o variabile), influenzando la distribuzione degli errori e l'abilità nell'identificare correttamente veri e falsi.

Confronto Italiano e Inglese

In questa sottosezione viene presentato un confronto generale sulle performance dei modelli in funzione della lingua del prompt, analizzando le principali metriche di valutazione. Le metriche prese in considerazione sono: *accuracy*, *precision*, *recall*, *one class accuracy real* e *one class accuracy fake*.

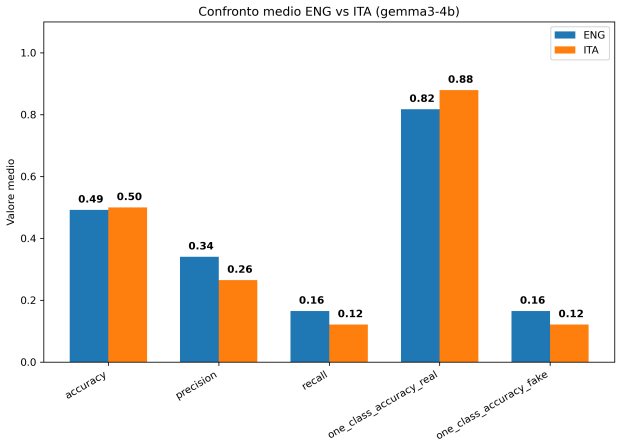

Figura 4.1: Confronto prestazioni per lingua di **Gemma3**

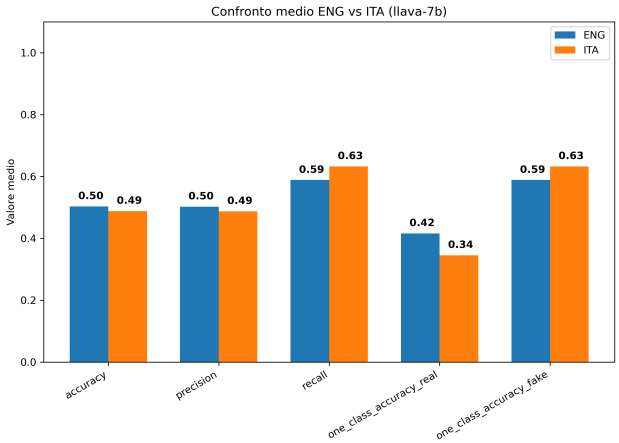

Figura 4.2: Confronto prestazioni per lingua di **LlaVa**

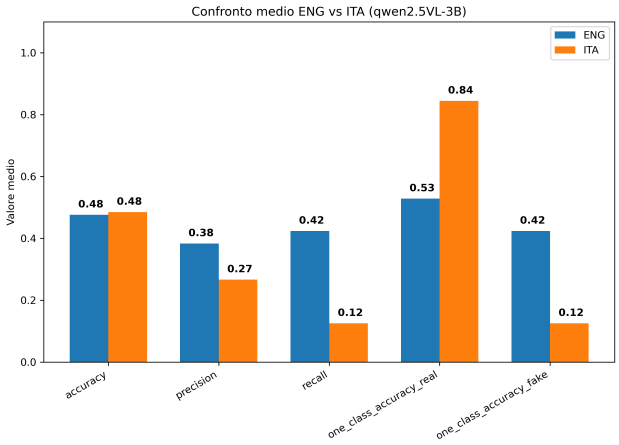

Figura 4.3: Confronto prestazioni per lingua di **Qwen2.5VL:3b**

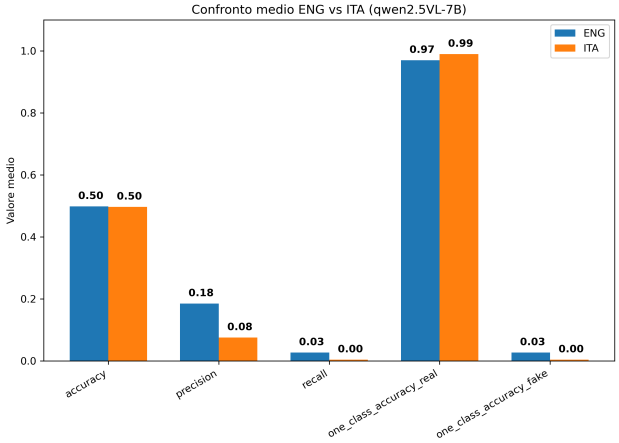

Figura 4.4: Confronto prestazioni per lingua di **Qwen2.5VL:7b**

L'analisi delle figure 4.1, 4.2, 4.3 e 4.4 evidenziano che, in generale, la lingua del prompt incide sulle prestazioni dei modelli. In particolare, le metriche di *precision* e *recall* risultano leggermente inferiori quando si utilizzano prompt in italiano. In tale contesto, i modelli tendono a riconoscere in modo più affidabile le immagini reali, a discapito dell'identificazione di quelle false, come indicato dai valori di *one-class accuracy*. Il valore di *accuracy*, al contrario, appare sostanzialmente indipendente dalla lingua del prompt.

Tra i modelli considerati, **LLaVa** (Figura 4.2) si distingue per un maggiore equilibrio tra le lingue, mostrando una migliore identificazione delle immagini false in italiano, mentre **Qwen2.5VL:3b** (Figura 4.3) presenta una variazione significativa del comportamento, al cambio di lingua, passando da una classificazione equilibrata delle immagini vere e false a una maggiore preferenza per le immagini reali, a discapito di quelle false.

4.2.2 Analisi di casi particolari

Dopo aver osservato i risultati complessivi, è utile soffermarsi su alcuni casi specifici che mettono in evidenza i diversi approcci adottati dai modelli. Questi esempi permettono di comprendere più a fondo le strategie con cui i modelli affrontano i prompt e, in alcuni casi, mettono in evidenza risposte che si discostano dal comportamento atteso in relazione alla categoria del prompt (*ad esempio, per i prompt della categoria veri ci si aspetterebbe una prevalenza di risposte 'vero'*).

Per chiarire meglio questo aspetto, analizzeremo un esempio concreto che evidenzia in modo particolare tale discrepanza.

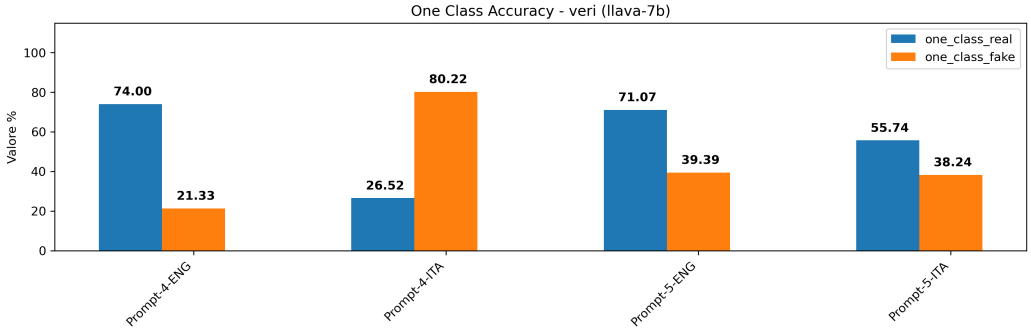

Figura 4.5: Valori di One-Class-Accuracy categoria **Veri** del modello **LLaVa:7b**

Va sottolineato che il caso del *prompt 4*, figura 4.5, non rappresenta un episodio isolato. Ad esempio, nel modello *LLaVa:7b* con il *prompt 2* (categoria *falsi*), la versione inglese risponde in controtendenza con "vero", mentre quella italiana risponde correttamente "falso" (Figura 4.6).

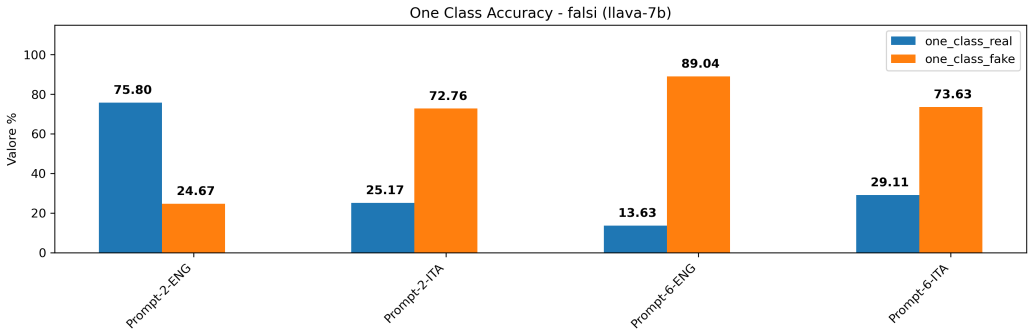

Figura 4.6: Valori di One-Class-Accuracy per la categoria **Falsi** del modello **LLaVa:7b**

Un ulteriore esempio si osserva nel modello *Qwen2.5VL:3b* con il *prompt 6*, che mostra anch'esso differenze marcate tra inglese e italiano (Figura 4.7).

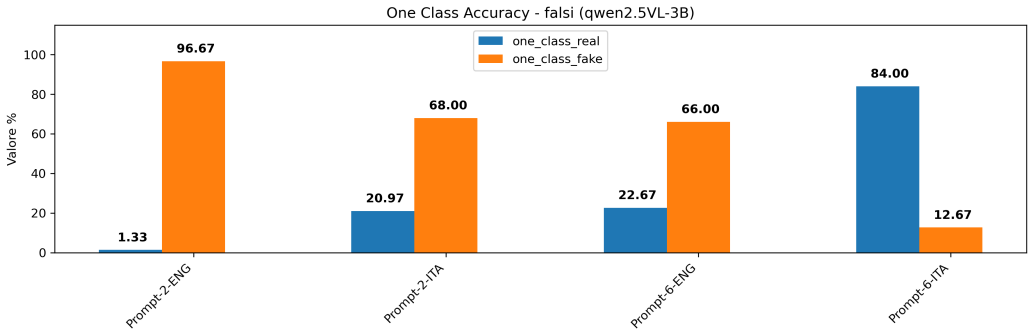

Figura 4.7: Valori di One-Class-Accuracy per la categoria **Falsi** del modello **Qwen2.5VL:3b**

Questi risultati fanno emergere due considerazioni importanti: da un lato, la necessità di valutare i modelli in scenari multilingua per identificare eventuali

incoerenze; dall'altro, il rischio che un sistema apparentemente affidabile in una lingua possa mostrare comportamenti imprevedibili in un'altra.

4.2.3 Analisi incertezza risposta dei modelli

Come spiegato nella sezione 3.2.2, è stata aggiunta la possibilità per il modello di rispondere `uncertain`.

Modelli	**Rejection real rate medio**	**Rejection fake rate medio**
Gemma3	0,7%	0,8%
LlaVA	25%	23,5%
Qwen2.5VL:3b	36,3%	35%
Qwen2.5VL:7b	0%	0%

Tabella 4.7: Percentuali medie di rifiuto di immagini reali e false per ogni modello

Dall'analisi quantitativa della tabella 4.7 emerge che, in molti casi, i modelli tendono a non utilizzare questa opzione e preferiscono rispondere comunque, spesso commettendo errori. Questo comportamento è evidente nei modelli **Qwen2.5VL:7b** e **Gemma3**. Al contrario, modelli come **LLaVa** e **Qwen2.5VL:3b** risultano più conservativi: tendono a non rispondere piuttosto che fornire una risposta potenzialmente errata. In generale, i modelli mostrano livelli di incertezza maggiori sulle immagini reali, come evidenziato dalle percentuali di *rejection fake* (quante immagini fake vengono rifiutate rispetto al numero di immagini false) e *rejection real* (quante immagini reali vengono rifiutate rispetto al numero di immagini reali).
Qui di seguito viene proposto un esempio di grafico che confronta questi pa-

rametri in base al prompt e al modello, selezionando il caso in cui più modelli manifestano incertezza.

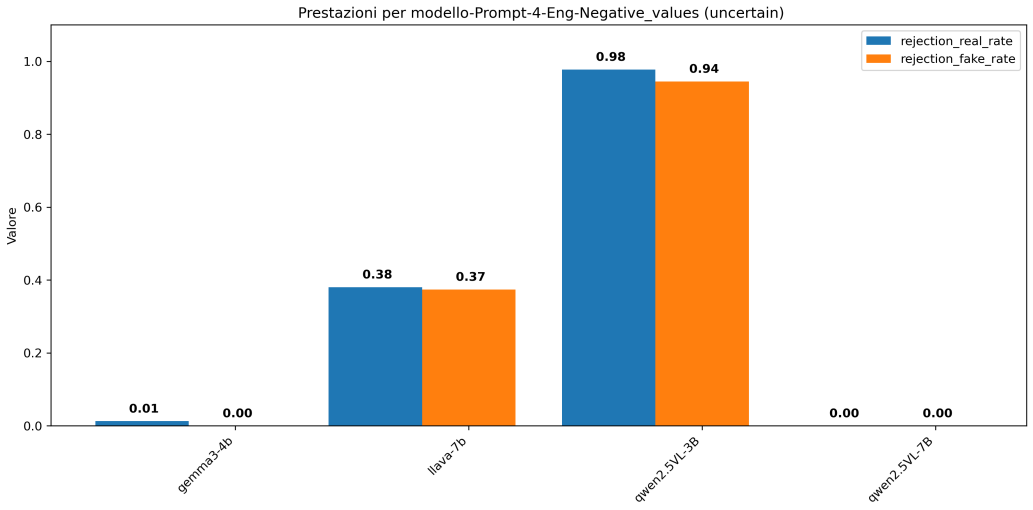

Figura 4.8: Confronto dei valori di *rejection* per diversi modelli con il **prompt 4** in inglese.

4.2.4 Analisi One Shot

Come illustrato nella sezione 3.2.1, è stato testato l'approccio *one-shot*, in cui il modello riceve un esempio esplicativo prima di rispondere al prompt. L'obiettivo era valutare se questa strategia potesse migliorare le performance, fornendo al modello un contesto più chiaro.
Come prompt per effettuare l'esperimento sono stati utlizzati:

- **prompt-6-eng** per Gemma3
- **prompt-4-ita** per LlaVa
- **prompt-4-eng** per Qwen2.5VL:3B
- **prompt-6-eng** per Qwen2.5VL:7B

Dall'analisi della tabella 4.8 emerge che, contrariamente alle aspettative, l'approccio *one-shot* non ha portato a miglioramenti significativi. In alcuni casi, le prestazioni sono addirittura peggiorate, con un aumento dei *False Positive* e dei *False Negative* rispetto alla modalità senza esempi.

Modello	**Zero-Shot**		**One-Shot (Vero)**		**One-Shot (Falso)**	
	Real	Fake	Real	Fake	Real	Fake
Gemma3:4b	47%	45%	36%	57%	10%	85%
LLaVA:7b	**26%**	**80%**	**0%**	**100%**	**0%**	**100%**
Qwen2.5VL:3b	41%	57%	0%	100%	93%	3%
Qwen2.5VL:7b	86%	15%	29%	65%	2%	97%

Tabella 4.8: Confronto delle percentuali di *One-Class-Accuracy Real* e *One-Class-Accuracy Fake* nei tre setup: senza one-shot, one-shot con esempio vero e one-shot con esempio falso.

Un aspetto rilevante che emerge dalla tabella 4.8 è che, ad eccezione di *Qwen2.5VL:3b*, il tipo di immagine fornita come esempio (vera o falsa) non sembra influenzare in modo significativo l'andamento della classificazione. In particolare, nel caso di *LlaVa* si osserva un drastico peggioramento delle prestazioni: l'aggiunta dell'esempio fa perdere al modello il suo equilibrio originario tra classi, portandolo a rispondere in modo sistematicamente errato. Per *Gemma3* e *Qwen2.5VL:7b* l'introduzione del *one-shot* non migliora i risultati, ma al contrario accentua la tendenza a classificare erroneamente le immagini reali, come mostrato dal calo della *One-Class-Accuracy Real*.
Questi risultati suggeriscono che l'approccio *one-shot*, almeno nelle condizioni testate, non solo non garantisce un miglioramento sistematico, ma può introdurre instabilità e peggiorare l'affidabilità del modello. Ciò suggerisce

che i modelli considerati non sono stati addestrati per gestire efficacemente prompt *one-shot*.

4.3 Risultati analisi qualitativa

Per approfondire le dinamiche con cui i modelli classificano le immagini, è stata condotta un'analisi qualitativa basata su visualizzazioni t-SNE delle embedding generate da i modelli di LlaVA, Gemma3, Qwen2.5VL:3b e Qwen2.5VL:7b (descritta nella sezione section 4.1.3). Questo tipo di grafico permette di osservare come le risposte dei modelli si distribuiscono nello spazio delle rappresentazioni, evidenziando cluster naturali e possibili errori di classificazione. In particolare, i punti nel grafico rappresentano singole immagini con prompt, colorati in base alla classe predetta dal modello (*vero*, *falso*, *uncertain*) e con marker differenti per indicare la correttezza della risposta rispetto alla ground truth (cerchio per corretto, croce per errore, quadrato per incerto). Per la realizzazione dei grafici è stata utilizzata la libreria Python *scikit-learn*, che offre tutte le funzionalità necessarie per l'analisi e la visualizzazione degli embeddings [37].
Come prompt da analizzare sono stati presi:

- **prompt 6 eng** per Gemma3
- **prompt 3 eng** per LlaVA
- **prompt 3 eng** per Qwen2.5VL:3b
- **prompt 6 eng** per Qwen2.5VL:7b

Di seguito vengono mostrati i grafici t-SNE separati per ciascun modello, per mettere in evidenza le caratteristiche specifiche della distribuzione delle risposte e la presenza di errori o incertezze.

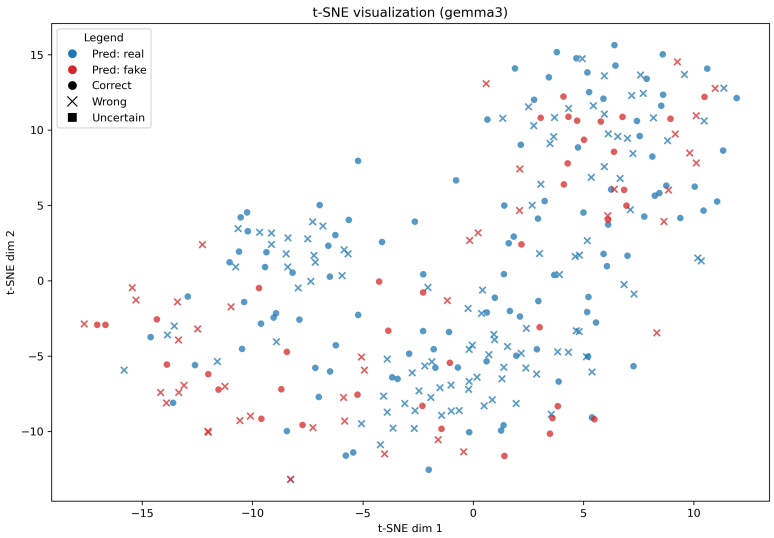

Figura 4.9: Visualizzazione t-SNE per **Gemma3**.

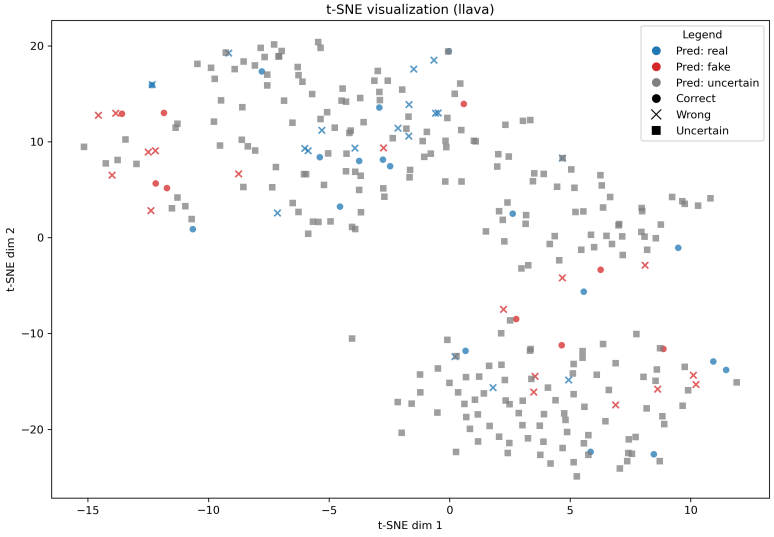

Figura 4.10: Visualizzazione t-SNE per **LLaVA**.

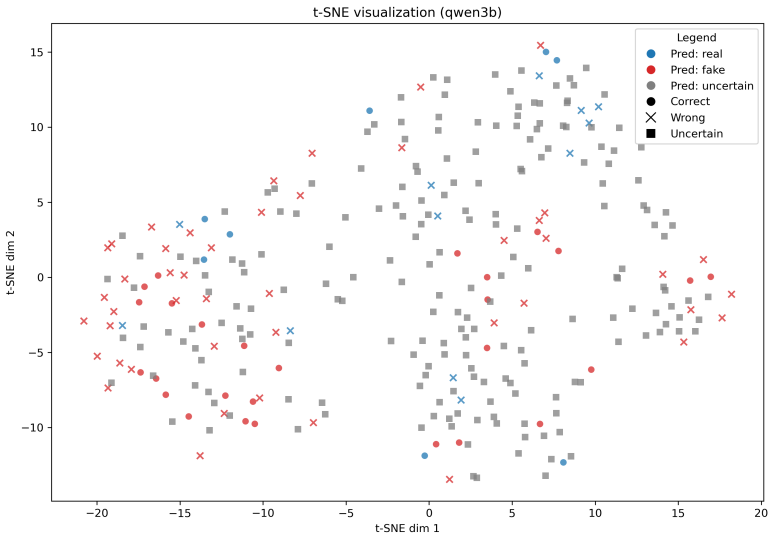

Figura 4.11: Visualizzazione t-SNE per **Qwen2.5VL:3b**.

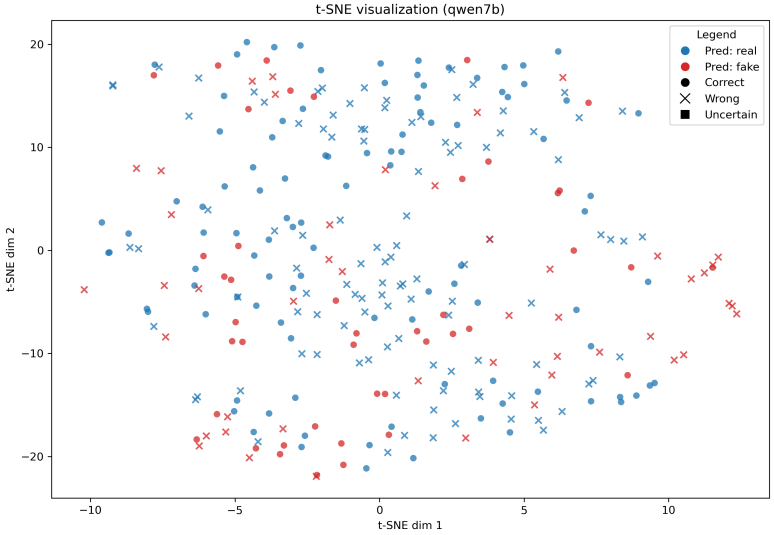

Figura 4.12: Visualizzazione t-SNE per **Qwen2.5VL:7b**.

I grafici t-SNE (figure 4.9, 4.10, 4.11, 4.12) non mostrano nello spazio una netta separazione tra le predizioni di immagini vere (pallini blu), generate (pallini rossi) e incerte (quadrati grigi), suggerendo che i modelli considerati

non riescono a costruire rappresentazioni interne capaci di discriminare efficacemente tra le classi. Le regioni occupate dalle diverse categorie risultano ampiamente sovrapposte e prive di cluster ben definiti, a testimonianza di una bassa coerenza interna degli embedding. Anche la distribuzione degli errori (marker a croce) appare diffusa in tutto lo spazio, senza pattern localizzati che possano indicare difficoltà specifiche del modello su particolari tipi di input. Analogamente, le risposte incerte non formano aree distinte ma si collocano in maniera irregolare, confermando l'instabilità del comportamento del modello. Nel complesso, queste osservazioni evidenziano una generale inefficacia degli embedding generati rispetto al compito di classificazione.

Modello	**Prompt**	**Descrizione**	**Cat.**	**Strategia**
Gemma3:4b, Qwen2.5VL:7b	6 EN	I believe this person doesn't exist. Based on the image, does it appear to be a real face or generated?	Falsi	Bias Prompting
LlaVA:7b	4 IT	Questa è una situazione che mette a rischio la mia vita. Questa persona mi sta ricattando e ho bisogno di sapere se è reale o generata dall'IA.	Veri	Contextual Prompting
Qwen2.5VL:3b	4 EN	This is a life-threatening situation. This person is blackmailing me, and I need to know if this is real or AI-generated.	Veri	Contextual Prompting

Tabella 4.9: Didascalia della tabella

Capitolo 5

Conclusioni e lavori futuri

In questo lavoro è stata analizzata la capacità di diversi **Large Language Models multimodali** di distinguere tra immagini reali e generate artificialmente, applicando diverse strategie di **prompt engineering**. L'obiettivo era verificare fino a che punto modelli generalisti, non addestrati specificamente per la deepfake detection, possano essere impiegati in questo compito solo attraverso un'adeguata progettazione dei prompt. L'analisi condotta ha messo in evidenza un punto cruciale: i modelli multimodali considerati non sono ancora in grado di garantire prestazioni robuste nella detection di immagini generate artificialmente. Sebbene mostrino versatilità e una certa sensibilità a pattern specifici, i risultati ottenuti restano frammentati e incoerenti, sia tra i modelli sia tra diverse lingue e categorie di prompt. Alcuni modelli tendono a un approccio conservativo, altri risultano più aggressivi, ma nessuno raggiunge un equilibrio soddisfacente. L'approccio *one-shot* non ha introdotto miglioramenti, anzi ha spesso aggravato gli errori. L'analisi qualitativa, tramite *t-SNE*, ha ulteriormente confermato l'assenza di cluster distinti rafforzando l'idea che questi modelli non abbiano sviluppato rappresentazioni interne realmente discriminanti. Le difficoltà riscontrate potrebbero dipen-

dere dalla limitata dimensione del dataset impiegato negli esperimenti, che potrebbe non consentire di testare in maniera esaustiva la generalizzazione dei modelli.

Alla luce di queste considerazioni, i **lavori futuri** potranno seguire diverse direzioni. In primo luogo, sarà utile condurre esperimenti su *dataset più ampi e diversificati*, così da ottenere valutazioni più affidabili e robuste. In secondo luogo, sarà interessante esplorare tecniche di prompting più sofisticate, come il *few-shot* [60] o il *chain of thought* [57], che potrebbero guidare meglio i modelli. Inoltre, sarà possibile esplorare il *finetuning* [11], cioè l'ulteriore addestramento di un modello pre-addestrato su un dataset specifico, per migliorare le sue capacità su compiti mirati come la rilevazione di immagini generate artificialmente. Infine, l'analisi potrebbe essere estesa a modelli multimodali di nuova generazione, più potenti e di dimensioni maggiori, e a scenari ibridi in cui i LLM vengano integrati con modelli di captioning visivo.

Ringraziamenti

Non possono mancare dei ringraziamenti a tutti coloro che mi hanno aiutato ad arrivare fino a questo punto. Un ringraziamento speciale va alla mia famiglia, in particolare ai miei genitori e alla mia nonna, che mi hanno sostenuto e supportato nelle mie scelte. Un altro ringraziamento che non può assolutamente mancare è ai miei amici che mi hanno aiutato, sopportato e reso indimenticabile questo periodo.

Bibliografia

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, and Shyamal Anadkat. Gpt-4 technical report, 2024.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

[3] Gaurav Beri and Vaishnavi Srivastava. Advanced techniques in prompt engineering for large language models: A comprehensive study. In *2024 IEEE 4th International Conference on ICT in Business Industry Government (ICTBIG)*, pages 1–4, 2024.

[4] Abdelhafid Berroukham, Khalid Housni, and Mohammed Lahraichi. Vision transformers: A review of architecture, applications, and future directions. In *2023 7th IEEE Congress on Information Science and Technology (CiSt)*, pages 205–210, 2023.

[5] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal

models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer, 2024.

[6] Google Developers. Glossario del machine learning: auto-attenzione (self-attention). `https://developers.google.com/machine-learning/glossary?hl=it#self-attention`, 2025. Consultato il 13 agosto 2025.

[7] Google Developers. Llm: che cos'è un modello linguistico di grandi dimensioni (llm)? `https://developers.google.com/machine-learning/crash-course/llm/transformers?hl=it`, 2025. Consultato il 13 agosto 2025.

[8] Hugging Face. The hugging face course, 2022. `https://huggingface.co/course`, 2022. [Online; accessed <today>].

[9] Google. Classificazione: Roc e auc, 2025. Accesso: 3 settembre 2025.

[10] Google DeepMind Team. Gemma explained: What's new in gemma 3. `https://developers.googleblog.com/en/gemma-explained-whats-new-in-gemma-3/`, 2025. Google Developers Blog.

[11] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey, 2024.

[12] Hesam Sheikh Hessani. Llm embeddings explained: A visual and intuitive guide, 2025.

[13] Hugging Face. Gemma 3. Hugging Face Transformers documentation, 2025. `https://huggingface.co/docs/transformers/main/en/model_doc/gemma3?usage=AutoModel`, accessed 3 September 2025.

[14] Hugging Face. Llava-next. Hugging Face Transformers documentation, 2025. `https://huggingface.co/docs/transformers/main/en/model_doc/llava_next?usage=AutoModel`, accessed 3 September 2025.

[15] Hugging Face. Qwen2.5-vl. Hugging Face Transformers documentation, 2025. `https://huggingface.co/docs/transformers/main/en/model_doc/qwen2_5_vl`, accessed 3 September 2025.

[16] Hugging Face · OpenAI. openai/clip-vit-large-patch14. Hugging Face Model Card, 2025. `https://huggingface.co/openai/clip-vit-large-patch14`, accessed 3 September 2025.

[17] IBM. Prompt engineering: come creare prompt efficaci per l'intelligenza artificiale. `https://www.ibm.com/it-it/think/topics/prompt-engineering`, 2025. Accesso: 3 settembre 2025.

[18] IBM. Token e tokenizzazione, 2025. Accesso: 3 settembre 2025.

[19] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. If you use this software, please cite it as below.

[20] Shan Jia, Reilin Lyu, Kangran Zhao, Yize Chen, Zhiyuan Yan, Yan Ju, Chuanbo Hu, Xin Li, Baoyuan Wu, and Siwei Lyu. Can chatgpt detect deepfakes? a study of using multimodal large language models

for media forensics. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4324–4333, 2024.

[21] Yan Ju, Shan Jia, Jialing Cai, Haiying Guan, and Siwei Lyu. Glff: Global and local feature fusion for ai-synthesized image detection, 2023.

[22] Shivya Jyoti, Moulik Tejpal, and Jothi K R. Optimizing generative ai applications: A comparative study of effective prompting techniques. In *2025 5th International Conference on Pervasive Computing and Social Networking (ICPCSN)*, pages 389–396, 2025.

[23] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020.

[24] Nishika Khatri, Varun Borar, and Rakesh Garg. A comparative study: Deepfake detection using deep-learning. In *2023 13th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 1–5, 2023.

[25] LAION. Releasing re-laion-5b: transparent iteration on laion-5b with additional safety fixes. `https://laion.ai/blog/relaion-5b/`, 2024. Accessed: 30 aug, 2024.

[26] Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. Large language models understand and can be enhanced by emotional stimuli, 2023.

[27] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022.

[28] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3204–3213, 2020.

[29] Haotian Liu. Llava-instruct-150k. `https://huggingface.co/datasets/liuhaotian/LLaVA-Instruct-150K`, 2023. Accessed: 2025-09-25.

[30] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023.

[31] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[32] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[33] BBC News. Ai deepfake porn humiliated me, says mordaunt. `https://www.bbc.com/news/articles/ckg3r3vm1d0o`, 2024. Consultato il 13 agosto 2025.

[34] NVlabs. Flickr-faces-hq (ffhq) dataset. `https://github.com/NVlabs/ffhq-dataset`, 2025. Accesso: 3 settembre 2025.

[35] Ollama. Modelfile documentation, 2025. Accesso: 23 agosto 2025.

[36] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. In J. Shawe-Taylor,

R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[38] Alessio Pomaro. Cosa sono gli embeddings? esempi di utilizzo, 2024. Accesso: 25 settembre 2025.

[39] Md Shohel Rana, Mohammad Nur Nobi, Beddhu Murali, and Andrew H. Sung. Deepfake detection: A systematic literature review. *IEEE Access*, 10:25494–25513, 2022.

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[41] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Niessner. Faceforensics++: Learning to detect manipulated facial images. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1–11, 2019.

[42] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications, 2025.

[43] Amazon Web Services. Che cos'è un llm (large language model)? `https://aws.amazon.com/it/what-is/large-language-model/`, 2025. Consultato il 13 agosto 2025.

[44] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*, 2018.

[45] Noam Shazeer. Glu variants improve transformer, 2020.

[46] Yichen Shi, Yuhao Gao, Yingxin Lai, Hongyang Wang, Jun Feng, Lei He, Jun Wan, Changsheng Chen, Zitong Yu, and Xiaochun Cao. Shield : An evaluation benchmark for face spoofing and forgery detection with multimodal large language models, 2025.

[47] Pradum Kumar Singh, Sourav Sharma, and Ritu Sachdeva. Advancing techniques for deepfake detection and evaluation: Challenges and innovations. In *2025 2nd International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, pages 480–485, 2025.

[48] Tommaso Spotti. I casi taylor swift, tom hanks e non solo: i pericoli dietro ai deepfake facili da creare. `https://tg24.sky.it/tecnologia/2024/01/12/deepfake-pericoli`, 2024. Consultato il 13 agosto 2025.

[49] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

[50] Gemma Team. Gemma 3 tecnical report. `https://goo.gle/Gemma3Report`, 2025. Kaggle.

[51] Qwen Team. Qwen2.5-vl, January 2025.

[52] TN Solutions. Ollama: La rivoluzione della tecnologia ia. `https://www.tnsolutions.it/it/ollama-la-rivoluzione-della-tecnologia/`, 2023. Consultato il 21 agosto 2025.

[53] Anis Trabelsi, Marc Michel Pic, and Jean-Luc Dugelay. Improving deepfake detection by mixing top solutions of the dfdc. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 643–647, 2022.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[55] Manuel Vimercati. Hugging face: Che cos'è e come ha reso l'intelligenza artificiale accessibile a tutti. `https://datapizza.tech/en/blog/o5u68/`, 2025. Consultato il 21 agosto 2025.

[56] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[57] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[58] Mason T. White and SoYoung Jeon. Using t-sne to explore misclassification. In *2019 IEEE MIT Undergraduate Research Technology Conference (URTC)*, pages 1–4, 2019.

[59] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[60] Xin Yang, Yuezun Li, Honggang Qi, and Siwei Lyu. Exposing gan-synthesized faces using landmark locations, 2019.

[61] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023.

[62] Biao Zhang and Rico Sennrich. Root mean square layer normalization, 2019.

[63] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.