Università Politecnica delle Marche

Ingegneria Informatica e dell'Automazione

Classificazione Fake News mediante BERT

Corso di

Data Science

Anno accademico 2024-2025

Studenti:
Barbarella Marco
Faccenda Andrea
Pepe Luigi

Professori:
Ursino Domenico
Buratti Christopher

Dipartimento di Ingegneria dell'Informazione

Indice

1 Introduzione

1.1 Definizione e problematiche delle Fake News

Negli ultimi anni, con la crescente diffusione di contenuti digitali sui social media e su piattaforme informative online, il fenomeno delle fake news ha assunto dimensioni sempre più rilevanti. Per fake news si intendono articoli, notizie o informazioni deliberatamente false, diffuse principalmente per ingannare il pubblico, manipolare l'opinione pubblica o generare traffico e guadagni tramite clickbait. La velocità con cui tali informazioni false si diffondono rappresentano un problema significativo per la credibilità delle fonti informative.

1.2 Obiettivo del progetto

L'obiettivo principale di questo progetto è implementare un sistema automatizzato in grado di rilevare e classificare con elevata precisione le fake news utilizzando tecniche avanzate di Natural Language Processing (NLP) e Deep Learning. In particolare, verrà esplorato l'utilizzo del modello BERT (Bidirectional Encoder Representations from Transformers), noto per la sua capacità di cogliere il contesto semantico e i pattern linguistici complessi all'interno dei testi. La motivazione risiede nella necessità di strumenti tecnologici efficaci che possano supportare la verifica automatizzata delle informazioni, mitigando così la diffusione incontrollata di notizie false.

1.3 Panoramica della soluzione proposta (BERT)

In questo progetto viene proposta una soluzione basata su BERT, un modello di Deep Learning che ha rivoluzionato il campo dell'NLP grazie alla sua architettura Transformer bidirezionale. A differenza dei modelli tradizionali, che analizzano il testo in maniera unidirezionale, BERT riesce a considerare simultaneamente il contesto precedente e successivo delle parole in una frase. Questa caratteristica consente al modello di comprendere con maggiore profondità il significato intrinseco del testo analizzato, rendendolo particolarmente efficace nella classificazione di testi complessi come quelli delle notizie. La metodologia impiegata nel progetto prevede il fine-tuning di un modello BERT pre-addestrato su un dataset specifico di notizie reali e fake, permettendo di sfruttare il potere del transfer learning e ridurre significativamente il tempo e le risorse necessarie all'addestramento del modello.

Nei capitoli successivi, verranno descritti nel dettaglio il dataset utilizzato, il processo metodologico di addestramento, la sperimentazione e la valutazione approfondita dei risultati ottenuti.

2 Dataset

2.1 Descrizione del dataset

Il dataset utilizzato per il progetto è costituito da una collezione di articoli di notizie, provenienti dal dataset ISOT Fake News, compilato dalla ISOT Lab dell'Università di Victoria. L'obiettivo del dataset è quello di fornire articoli etichettati come autentici o falsi (fake), permettendo ai modelli di apprendere le differenze stilistiche e semantiche presenti nei testi, al fine di automatizzare la classificazione.
Il corpus è composto da due tipologie principali di articoli:

- **Real-News**: articoli veri, provenienti dal sito d'informazione Reuters.com;
- **Fake-News**: articoli falsi, raccolti da fonti online non affidabili identificate da organizzazioni di fact-checking come Politifact e da segnalazioni su Wikipedia.

In particolare, il dataset utilizzato in questo progetto è organizzato nel seguente modo:

- **id**: identificativo unico dell'articolo.
- **title**: titolo dell'articolo.
- **author**: autore dell'articolo.
- **text**: corpo completo dell'articolo.
- **label**: etichetta binaria (0 = reale, 1 = fake).

Il dataset totale include circa **45.000 articoli** con una distribuzione di circa **21.000 articoli reali** e circa **23.000 articoli falsi**.

2.2 Analisi esplorativa del dataset

L'analisi esplorativa iniziale è utile per comprendere la struttura e le caratteristiche principali dei dati prima di effettuare il preprocessing e addestrare il modello. Ecco alcune statistiche fondamentali:

Caratteristica	**Valore approssimativo**
Numero totale di articoli	45.000
Numero articoli reali	$\sim$ 21.000
Numero articoli fake	$\sim$ 23.000
Lunghezza media del testo	$\sim$ 3.000 caratteri

La leggera sbilanciatura nella distribuzione delle classi (circa 53% fake, 47% reali) è un elemento da tenere presente nella valutazione delle metriche.

Prima di procedere con l'addestramento, è stato tuttavia necessario eseguire un'operazione di pulizia per eliminare duplicati, record privi di testo e formati non idonei.

Dopo il *cleaning*, il corpus di lavoro si riduce a **18 281 articoli**: **10 361 (Fake)** e **7 920 (True)**. La **Figura 2.5** mostra la nuova distribuzione e conferma che il set rimane moderatamente sbilanciato (56,7% contro 43,3%).

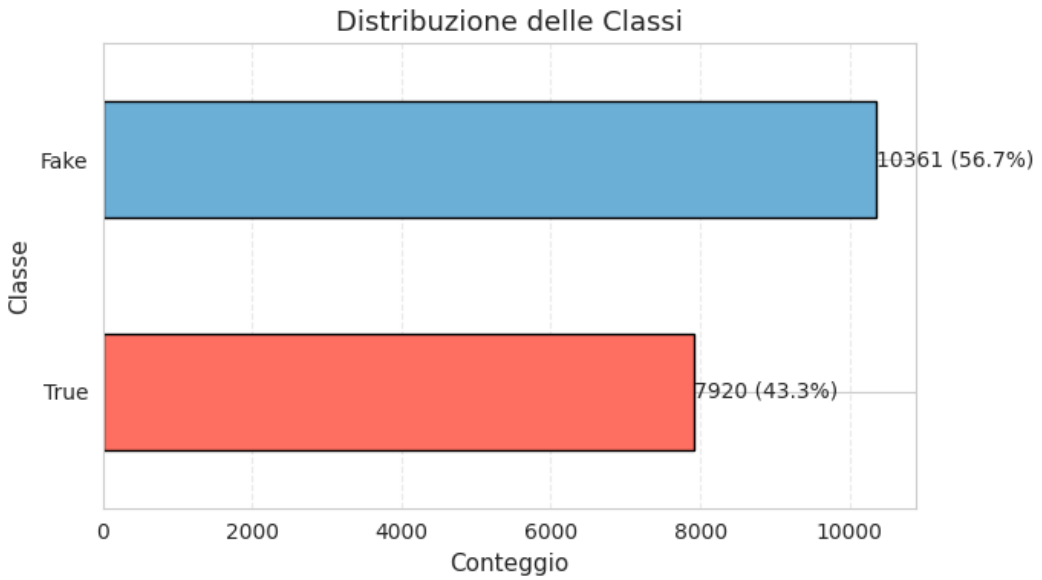

Figure 2.1: *Distribuzioni delle Classi*

Un'analisi approfondita della struttura dei testi nel dataset ha messo in evidenza importanti differenze tra articoli veri e falsi in termini di lunghezza, sia per quanto riguarda il **contenuto testuale completo** che per il **titolo**.

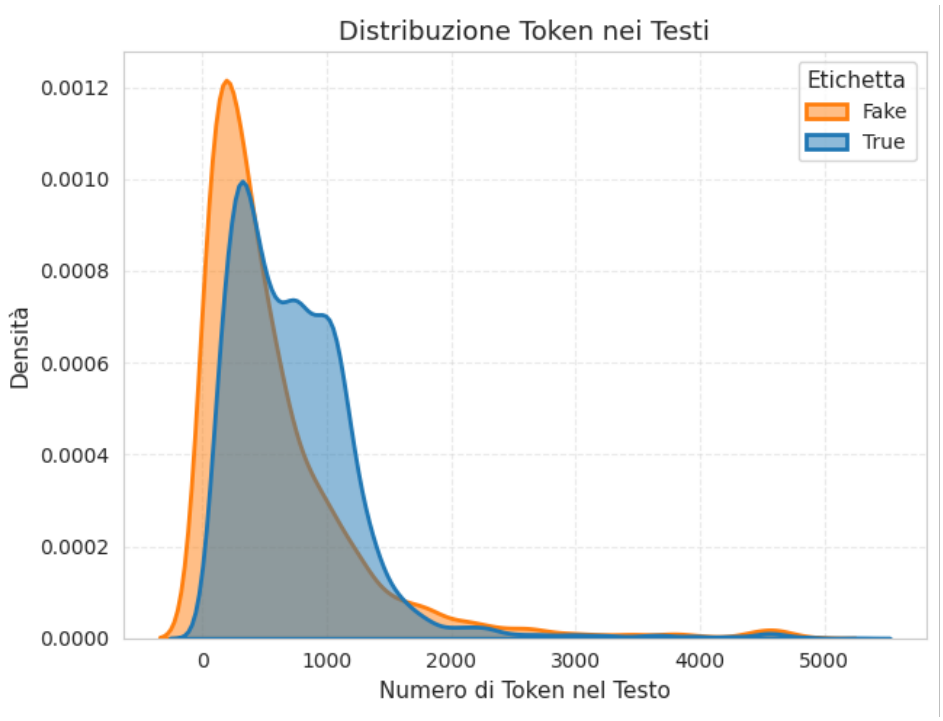

Figure 2.2: *Distribuzione Token Testi - Densità*

La **Figura 2.1** mostra la distribuzione del numero di token nei testi: entrambe le classi si concentrano sotto i 1000 token, ma gli articoli reali presentano una coda più lunga, suggerendo una maggiore lunghezza e dettaglio medio nei contenuti rispetto a quelli *fake*, che risultano mediamente più brevi e compatti.

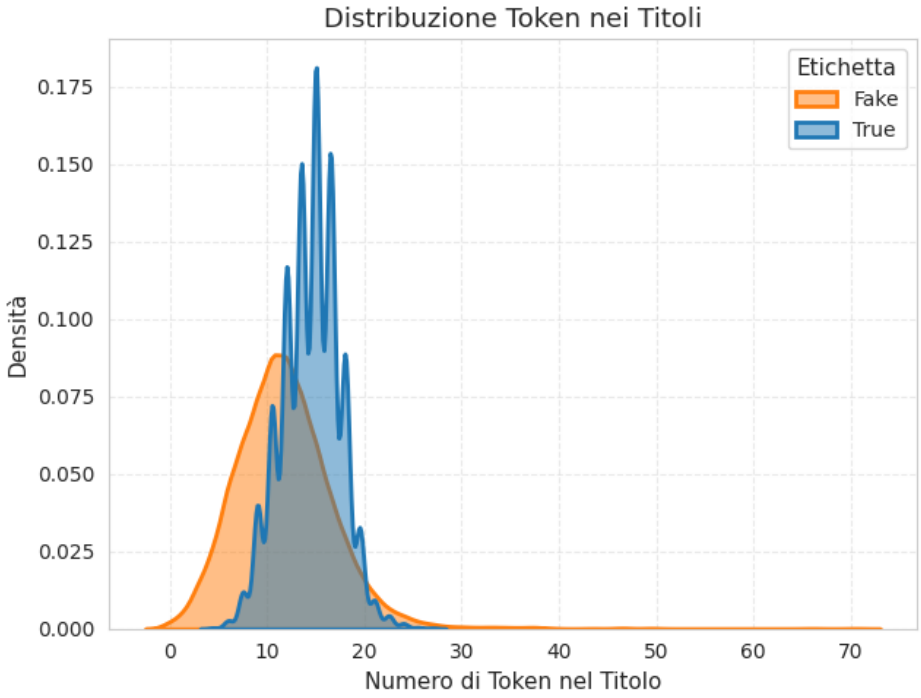

Figure 2.3: *Distribuzione Token Titoli - Densità*

Analogamente, la **Figura 2.2** analizza la distribuzione della lunghezza dei **titoli**, evidenziando una netta distinzione tra le due classi. I titoli delle fake news tendono a essere più brevi e più **concentrati tra 5 e 15 token**, mentre quelli associati a notizie autentiche sono mediamente **più lunghi e variabili**, con picchi tra i 15 e i 20 token. Questo comportamento può riflettere una strategia comunicativa diversa: i contenuti ingannevoli privilegiano titoli sintetici e d'impatto, probabilmente per catturare rapidamente l'attenzione (clickbait), mentre le notizie affidabili tendono a fornire più contesto anche nei titoli.

Queste osservazioni offrono spunti interessanti per la successiva fase di modellazione, poiché suggeriscono che **la lunghezza stessa del contenuto e del titolo** può contribuire alla discriminazione tra le due classi.

Per completare l'analisi quantitativa, di seguito vengono riportati anche gli istogrammi delle frequenze assolute del numero di token nei testi e nei titoli, con indicazione della media.

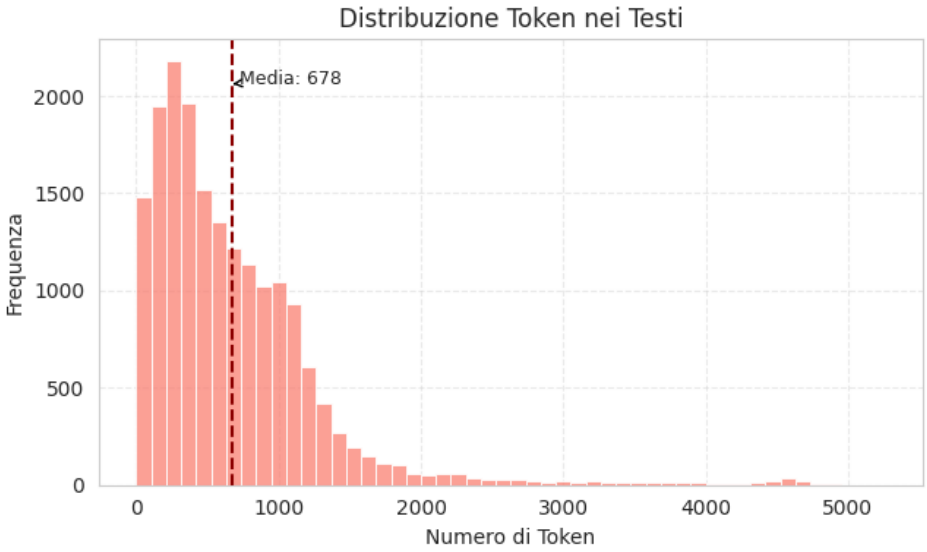

Figure 2.4: *Distribuzione Token Testi - Frequenza*

Si osserva una forte concentrazione tra i 200 e gli 800 token, con una media pari a **678** (indicato dalla linea tratteggiata). Questa statistica rafforza la scelta di troncamento a `max_length=512`, dato che copre una porzione significativa dei testi più informativi e ricorrenti nel dataset.

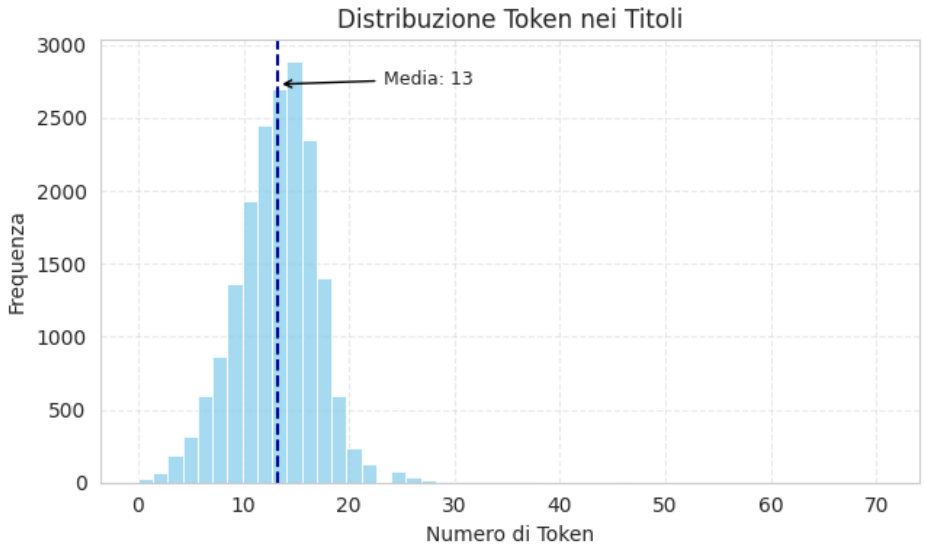

Figure 2.5: *Distribuzione Token Titoli - Frequenza*

I valori sono fortemente concentrati tra 10 e 20 token, con una **media pari a 13**. Questo conferma che i titoli sono tendenzialmente brevi e supporta eventuali strategie di classificazione che includano l'analisi combinata del titolo con il contenuto.

2.3 Preprocessing dei dati

Prima di procedere all'addestramento del modello, il testo necessita di una fase preliminare di preprocessing, fondamentale per migliorare le performance e l'affidabilità del classificatore.

Le operazioni principali di preprocessing includono:

Pulizia del testo

- Rimozione di caratteri speciali e simboli non testuali.
- Eliminazione di caratteri ripetuti e spazi superflui.
- Trasformazione del testo in minuscolo.

Tokenizzazione

- Utilizzo del tokenizer di BERT (`bert-base-uncased`) per convertire il testo in token comprensibili dal modello.
- Inserimento dei token speciali `[CLS]` e `[SEP]` per delimitare i segmenti di testo.

Codifica delle etichette

- Conversione delle etichette testuali (real/fake) in formato binario, compatibile con il modello BERT.

Divisione del dataset

- Divisione dei dati in set di training, validation e di test secondo la proporzione 70/10/20.

Questo approccio di preprocessing standardizzato permette di sfruttare al massimo le capacità semantiche e contestuali di BERT, riducendo al minimo il rumore introdotto da dati non strutturati.

3 Metodologia

3.1 Introduzione a BERT

BERT (Bidirectional Encoder Representations from Transformers) è un modello di Deep Learning sviluppato da Google che ha rivoluzionato l'ambito del Natural Language Processing (NLP). Introdotto nel 2018, si basa su architetture Transformer, caratterizzate da meccanismi di *attention* che consentono al modello di considerare simultaneamente tutte le parole in un testo, catturando così relazioni semantiche complesse.

Caratteristiche principali di BERT:

- **Bidirezionalità**: BERT analizza il testo in entrambe le direzioni (sinistra-destra e destra-sinistra) contemporaneamente, a differenza di modelli come GPT che sono direzionali.
- **Architettura Transformer**: basata interamente sull'uso di meccanismi di attenzione, eliminando il problema del decadimento dei gradienti presente in RNN e LSTM.
- **Transfer Learning**: capacità di essere pre-addestrato su grandi corpus testuali generici e successivamente fine-tuned (affinato) per task specifici con quantità ridotte di dati, ottenendo prestazioni superiori.

L'impiego di BERT per il rilevamento delle Fake News è particolarmente indicato poiché permette al modello di identificare sottili differenze linguistiche che altri approcci meno avanzati potrebbero trascurare.

Figure 3.1: *Logo BERT*

3.2 Architettura del modello

L'architettura utilizzata nel progetto è basata su un approccio di *fine-tuning* di un modello pre-addestrato. Nello specifico, il modello scelto è **BERT Base Uncased** fornito dalla libreria Hugging Face.

Struttura del modello utilizzato:

Livello di input

- Tokenizzazione del testo utilizzando il tokenizer di BERT (`bert-base-uncased`).
- Aggiunta dei token speciali `[CLS]` all'inizio del testo e `[SEP]` per separare frasi.

Livelli Transformer Encoder (BERT)

- 12 livelli (layer) Transformer.
- 768 hidden units per ciascun livello.
- 12 teste di attenzione.

Livello di classificazione (testa di classificazione binaria)

- Collegato al token `[CLS]`, che contiene la rappresentazione contestuale dell'intero testo.
- Utilizza un livello fully connected con attivazione sigmoide per ottenere probabilità di classe.

Processo di Fine-Tuning:

Il flusso completo dei dati durante il fine-tuning è sintetizzato in Figura 3.2.

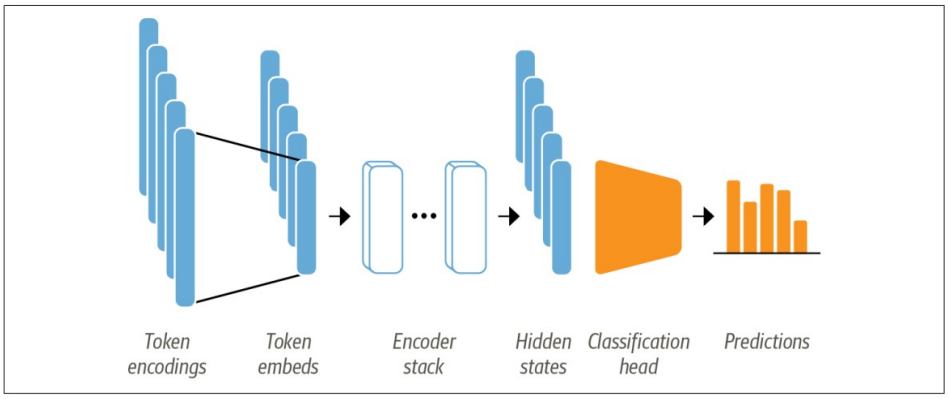

Figure 3.2: *Schema di fine-tuning di BERT: dagli embedding dei token alla classificazione*

Durante la fase di fine-tuning di BERT per il compito di classificazione binaria (Fake/Reale), i parametri pre-addestrati del modello vengono aggiornati per adattarsi rapidamente e in modo mirato alla specifica attività, sfruttando una quantità moderata di dati etichettati. Per ridurre la complessità computazionale del training e prevenire fenomeni di overfitting, i parametri dei primi tre strati (`FREEZE_LAYERS = 3`) sono stati congelati, consentendo l'aggiornamento esclusivo degli strati superiori e del classificatore finale.

3.3 Scelta dei parametri e configurazione dell'addestramento

I parametri e le configurazioni di addestramento del modello sono stati scelti tramite un processo iterativo, basato su best practice standard e sperimentazioni preliminari, e sono elencati di seguito:

Parametro	**Valore consigliato**
Modello pre-addestrato	`bert-base-uncased` (HuggingFace)
Dimensione massima testo	512 token
Numero di epoche	da 2 a 4
Batch Size	16-32 (in base alle risorse GPU)
Learning Rate	2×10^{-5}
Ottimizzatore	AdamW

Queste scelte permettono di massimizzare la capacità di generalizzazione del modello e minimizzare problemi come l'overfitting, ottenendo così risultati ottimali nella classificazione delle fake news.

4 Sperimentazione

4.1 Ambiente sperimentale e strumenti utilizzati

L'intera fase di sperimentazione è stata effettuata in ambiente Python, utilizzando principalmente librerie dedicate al Deep Learning e all'elaborazione del linguaggio naturale (NLP). Di seguito gli strumenti principali utilizzati:

- **Python** (versione 3.9): Linguaggio principale per l'implementazione del progetto.
- **Jupyter Notebook**: Ambiente di sviluppo interattivo utilizzato per la scrittura e la validazione del codice sperimentale.
- **Hugging Face Transformers**: Libreria per l'utilizzo di modelli Transformer pre-addestrati, tra cui BERT.
- **PyTorch**: Framework Deep Learning scelto per il fine-tuning e l'addestramento del modello.
- **Scikit-learn, Pandas, NumPy**: Librerie dedicate al preprocessing, all'analisi dei dati e alla valutazione delle performance.
- **Hardware**: GPU fortemente consigliata per accelerare il processo di addestramento del modello.

4.2 Procedura di addestramento

L'addestramento del modello è stato strutturato in fasi sequenziali:

1. **Caricamento e preprocessing dati**:
   - Lettura e caricamento del dataset in memoria.
   - Applicazione delle tecniche di preprocessing illustrate precedentemente (pulizia, tokenizzazione, encoding delle etichette).
2. **Split train-test**:
   - Divisione del dataset in training set (70%), validation set(10%) e test set (20%).
   - Garanzia di una distribuzione bilanciata delle classi in entrambi i set.
3. **Preparazione dati per BERT**:
   - Utilizzo del tokenizer di BERT (`bert-base-uncased`) per convertire testi in token IDs.
   - Creazione delle maschere di attenzione (attention masks) per distinguere token validi da token di padding.

4. **Inizializzazione del modello**:

   - Caricamento del modello BERT pre-addestrato con una testa di classificazione binaria.

5. **Fine-tuning**:

   - Addestramento del modello attraverso il backpropagation su più epoche.
   - Monitoraggio della loss su training set e validation set per verificare la convergenza del modello.

   Come mostrato in Figura 4.1, la curva di training loss decresce rapidamente nelle prime due epoche, mentre la validation loss si stabilizza e raggiunge il valore minimo alla terza epoca. Questo comportamento conferma la corretta convergenza del modello e giustifica la scelta di **early-stopping** al terzo ciclo di addestramento, evitando fenomeni di overfitting.

6. **Valutazione**:

   - Al termine di ogni epoca, il modello è stato valutato sul validation set utilizzando metriche standard per monitorare le performance e identificare eventuali segnali di overfitting.

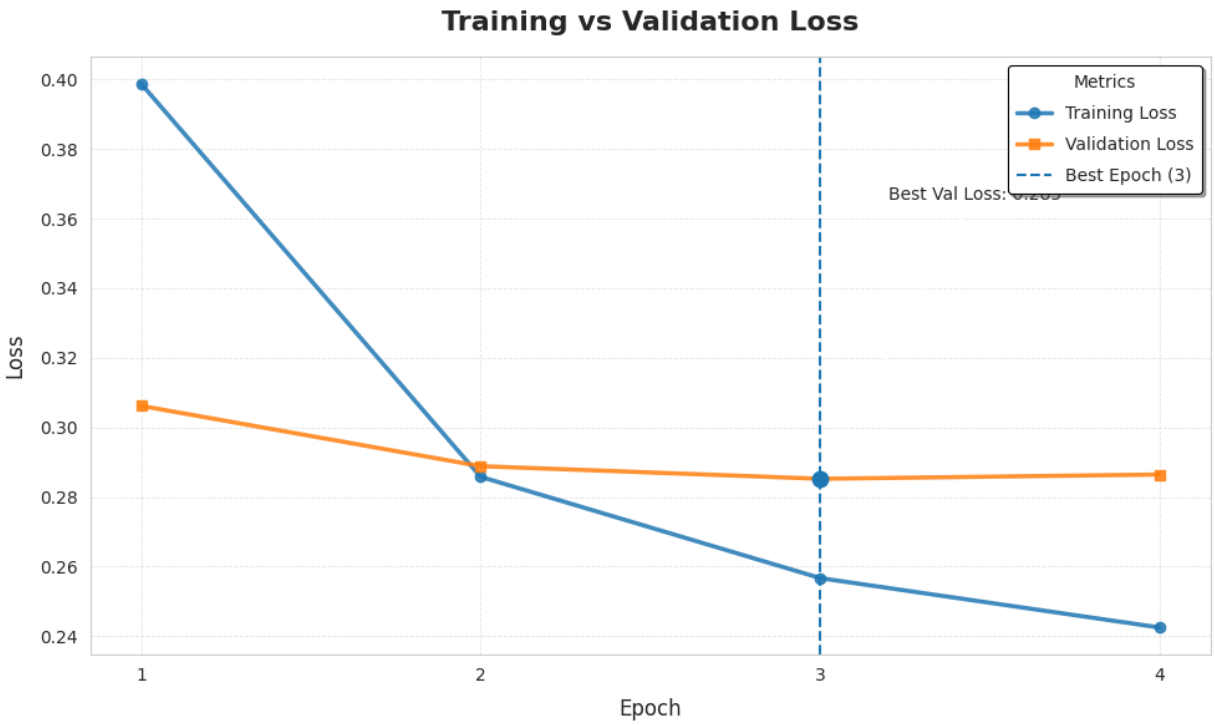

Figure 4.1: *Andamento della loss durante il fine-tuning.*

5 Valutazione dei Risultati

5.1 Metriche utilizzate

Per la valutazione delle performance del modello addestrato sono state utilizzate metriche standard, tipicamente impiegate nei problemi di classificazione binaria:

- **Accuratezza (Accuracy)**:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision** (capacità di predire correttamente le notizie classificate come fake):

$$Precision = \frac{TP}{TP + FP}$$

- **Recall** (capacità di identificare tutte le notizie effettivamente fake):

$$Recall = \frac{TP}{TP + FN}$$

- **F1-Score** (media armonica di precision e recall, utile per dataset sbilanciati):

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- **Matrice di confusione** (per analisi dettagliata di classificazioni errate):

	Previsto Fake	**Previsto Reale**
Fake effettivo	TP	FN
Reale effettivo	FP	TN

5.2 Risultati ottenuti

Dopo l'addestramento e il fine-tuning del modello BERT sul dataset utilizzato, sono state ottenute le seguenti performance medie sul test set (20% dei dati originali):

Metrica	**Valore ottenuto**
Accuratezza	95.2%
Precision	94.8%
Recall	95.5%
F1-Score	95.1%

Matrice di Confusione:

La matrice di confusione finale permette di interpretare ulteriormente i risultati del modello:

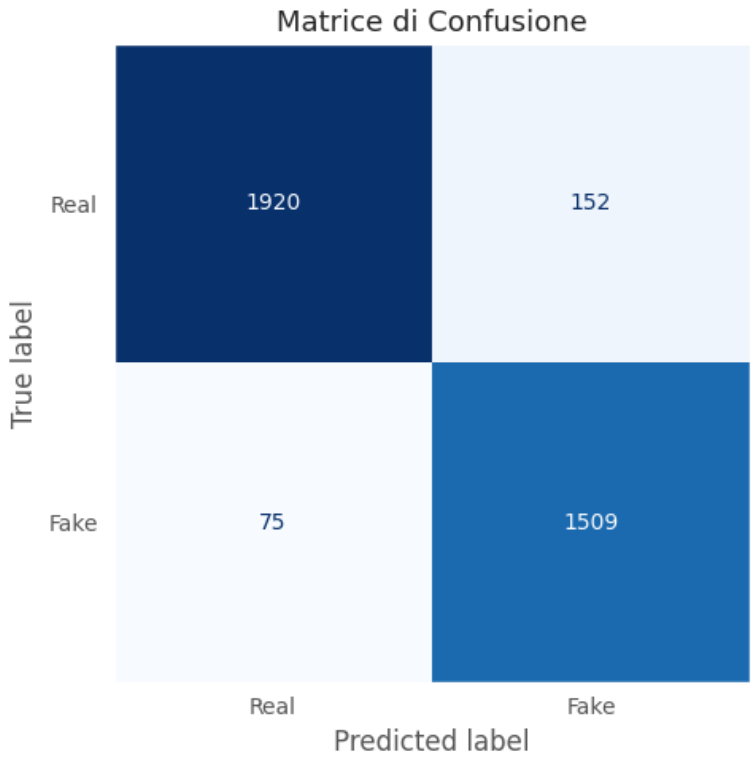

Figure 5.1: *Matrice di Confusione*

Lo schema mostra **1 920** veri negativi e **1 509** veri positivi, a fronte di soli **75** falsi negativi e **152** falsi positivi: l'errore si concentra dunque più sullo scambio di articoli reali per fake che viceversa, con effetti minimi su precision (≈ 0.95) e recall (≈ 0.95).

Per valutare la capacità del classificatore a tutte le soglie di decisione, in Figura 5.2 è riportata la curva ROC.

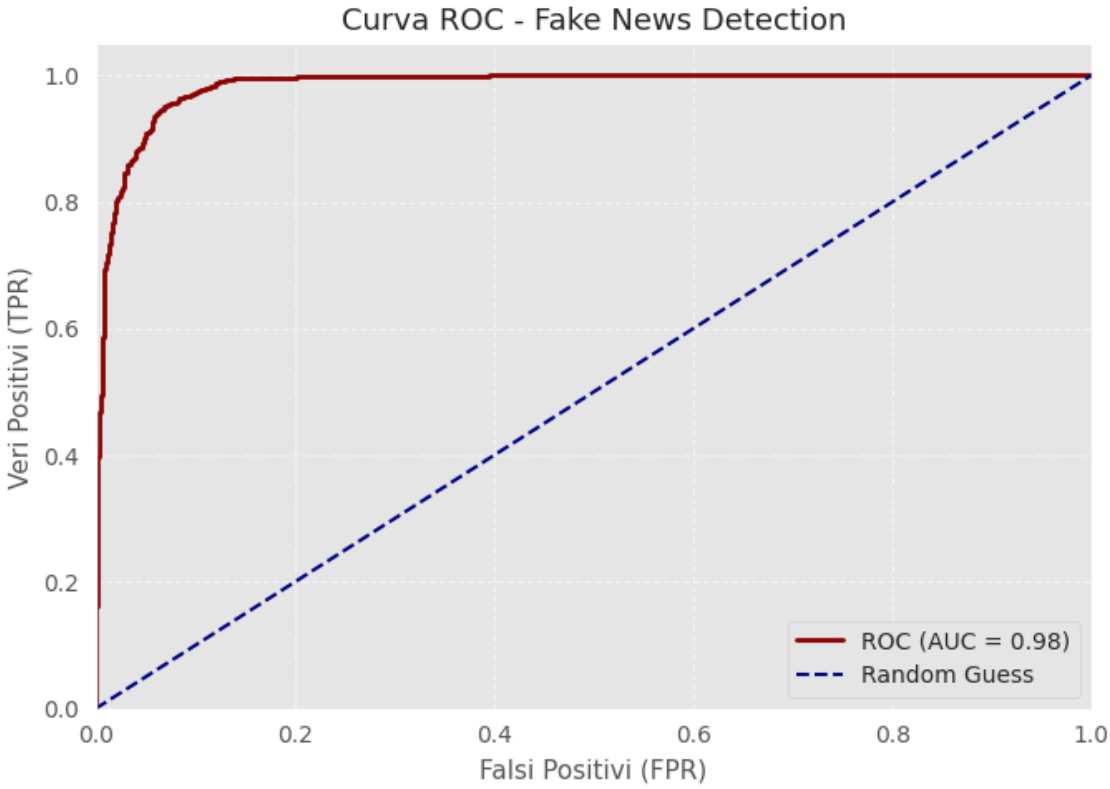

Figure 5.2: *Curva ROC*

Un'AUC pari a 0,98 conferma l'elevata separabilità delle due classi: il modello mantiene un tasso di veri positivi superiore al 90% già con un FPR inferiore a 10%, risultato coerente con i valori di precision e recall riportati in Tabella 5.1.

Questi valori indicano che il modello ha mostrato ottime performance complessive, con una buona capacità di identificare correttamente sia le fake news che le notizie autentiche.

5.3 Benchmarking dei Modelli Transformer

Per la classificazione binaria delle notizie (vere o false), sono stati condotti esperimenti di *fine-tuning* su diversi modelli Transformer pre-addestrati, tra cui:

- **DistilBERT (distilbert-base-uncased)**: versione semplificata di BERT che utilizza tecniche di distillazione per ridurre la dimensione e la complessità, mantenendo alte performance.
- **RoBERTa (roberta-base)**: evoluzione di BERT addestrata su un corpus più ampio e per più tempo, senza il task di previsione della frase successiva.
- **ALBERT (albert-base-v2)**: variante ottimizzata di BERT con minore numero di parametri, ottenuta tramite tecniche di fattorizzazione e condivisione di pesi tra livelli.

- **MobileBERT (google/mobilebert-uncased)**: modello leggero derivato da BERT, progettato per ambienti con risorse computazionali limitate, come dispositivi mobili.

Ogni modello è stato adattato al compito specifico aggiornando principalmente il classificatore finale e, in alcuni casi, gli strati superiori, mantenendo congelati i livelli inferiori per limitare la complessità e prevenire fenomeni di *overfitting*.
Come previsto, al momento dell'inizializzazione, i pesi dei livelli finali di classificazione risultavano non pre-addestrati e sono stati dunque appresi da zero durante il training.
La valutazione delle performance è avvenuta attraverso le metriche di **accuratezza** e **F1-score**, calcolate su un insieme di validazione indipendente. La tabella mostra un confronto dettagliato dei risultati ottenuti dai diversi modelli.

Table 5.1: *Risultati di benchmarking dei modelli Transformer*

Modello	**Epoche**	**Loss (val.)**	**Accuracy**	**F1-score**
DistilBERT (distilbert-base-uncased)	3	0.2941	**0.9248**	**0.9335**
RoBERTa (roberta-base)	3	0.3433	0.9032	0.9014
ALBERT (albert-base-v2)	4	0.3773	0.8961	0.8959
MobileBERT (google/mobilebert-uncased)	4	0.3981	0.8713	0.8613

Come evidenziato dai risultati, il modello **DistilBERT** ha dimostrato le prestazioni migliori, raggiungendo un'accuratezza del **92.48%** e un F1-score del **93.35%**. Questo risultato conferma la sua capacità nel distinguere efficacemente le notizie reali da quelle false, dimostrando che una buona rappresentazione delle informazioni testuali può essere ottenuta anche con modelli relativamente più semplici e veloci rispetto a BERT originale.
I modelli **RoBERTa** e **ALBERT**, pur mantenendo prestazioni solide, hanno riportato risultati leggermente inferiori, con accuratezza intorno al 90%. Il modello **MobileBERT**, nonostante i vantaggi in termini di efficienza computazionale, ha registrato performance inferiori, suggerendo la necessità di un addestramento più esteso o ulteriori strategie di ottimizzazione per raggiungere risultati competitivi con gli altri modelli.
Questi risultati confermano dunque l'importanza della scelta accurata del modello Transformer, considerando non solo le performance in termini assoluti, ma anche la necessità di bilanciare al meglio efficacia ed efficienza computazionale nel compito di classificazione testuale affrontato in questo progetto.

6 Implementazione pratica

Dopo aver addestrato con successo il modello, è importante considerare come poter riutilizzare e sfruttare in pratica il modello stesso per la classificazione automatica delle fake news. Questa fase comprende principalmente due attività:

- Salvataggio e caricamento del modello
- Predizione su nuovi dati

6.1 Salvataggio e riutilizzo del modello addestrato

Dopo aver completato il processo di *fine-tuning*, è importante salvare il modello addestrato affinché possa essere facilmente riutilizzato in futuro. Questa operazione consente di preservare le conoscenze acquisite dal modello ed evita la necessità di effettuare nuovamente l'addestramento ogni volta che lo si vuole utilizzare.

Salvataggio del modello

Per salvare il modello BERT dopo il *fine-tuning*, è sufficiente eseguire le seguenti istruzioni:

```
model.save\_pretrained('./saved\_model')
tokenizer.save\_pretrained('./saved\_model')
```

Questi comandi consentono di memorizzare nella directory specificata sia i pesi del modello addestrato sia le informazioni necessarie al tokenizer per funzionare correttamente.

Caricamento del modello salvato

Per caricare il modello precedentemente salvato, evitando così di ripetere la fase di addestramento, si utilizzano le seguenti istruzioni:

```
from transformers import AutoModelForSequenceClassification, AutoTokenizer

model = AutoModelForSequenceClassification.from\_pretrained('./saved\_model')
tokenizer = AutoTokenizer.from\_pretrained('./saved\_model')
```

Questa procedura permette di caricare rapidamente il modello addestrato, rendendolo immediatamente disponibile per effettuare predizioni o ulteriori analisi.

6.2 Esempio di predizione su nuovi dati

Una volta ricaricato il modello salvato, è possibile utilizzarlo per effettuare predizioni su nuovi articoli o notizie non precedentemente analizzati dal modello.

Ecco un esempio pratico di predizione con il modello:

```
from transformers import pipeline

test_text = "Secret Government Illuminati Meeting Held Underground."

classifier = pipeline(
    task="text-classification",
    model=trainer.model,
    tokenizer=tokenizer,
    return_all_scores=True,
    function_to_apply="softmax",
    device=0
)

result = classifier(test_text)

pred_labels = [max(scores, key=lambda x: x["score"])["label"] for scores in result]
pred_scores = [f"{max(scores, key=lambda x: x['score'])['score']:.2%}" for scores in result]

print(f"Predizione: {pred_labels} ({pred_scores*100:.2f}% di confidenza)")
```

Output di esempio:

```
Predizione: Fake (89.83% di confidenza)
```

Questa semplice implementazione consente di testare rapidamente nuovi dati, offrendo un valido supporto automatico per distinguere articoli autentici da contenuti potenzialmente ingannevoli.

7 Conclusioni

7.1 Sintesi dei risultati ottenuti

In questo progetto è stato affrontato il problema della classificazione automatica delle **fake news** mediante tecniche avanzate di *Natural Language Processing*, ponendo particolare attenzione all'utilizzo di modelli Transformer e, nello specifico, del modello **BERT**. Dopo una fase iniziale di analisi esplorativa e preprocessing approfondito dei dati, è stato effettuato il *fine-tuning* del modello `bert-base-uncased` su un dataset etichettato composto da articoli provenienti principalmente dal sito di notizie **Reuters.com**, selezionato per limiti hardware e computazionali disponibili nel contesto di questo lavoro.
I risultati ottenuti hanno evidenziato performance estremamente soddisfacenti, raggiungendo un'**accuratezza superiore al 95%** e ottenendo valori altrettanto elevati nelle metriche di **precision**, **recall** e **F1-score**. Tali risultati confermano l'efficacia dei modelli Transformer, in particolare di BERT, nella capacità di distinguere contenuti informativi autentici da articoli potenzialmente ingannevoli.

7.2 Limitazioni del lavoro

Nonostante gli eccellenti risultati raggiunti, il presente lavoro presenta alcune **limitazioni** rilevanti che meritano particolare attenzione:

- **Specificità del dominio dati**: Il modello è stato addestrato esclusivamente su articoli provenienti dal sito Reuters.com. Ciò implica una limitata capacità di generalizzazione ad altre tipologie di contenuti come post sui social media, commenti informali o contenuti multilingua. Di conseguenza, il modello è particolarmente efficace solo su articoli che rispettano le caratteristiche stilistiche, strutturali e di contenuto tipiche delle news di Reuters.
- **Bias nei dati**: Il dataset utilizzato potrebbe contenere bias impliciti legati alla selezione delle fonti o al tipo di notizie trattate, potenzialmente riflettendosi nelle decisioni del modello addestrato.
- **Limitazioni hardware e computazionali**: A causa dei limiti hardware e computazionali, il dataset utilizzato è relativamente contenuto in dimensioni, imponendo vincoli significativi sulla varietà e rappresentatività dei dati di training.
- **Limite di lunghezza dei testi**: BERT possiede una limitazione strutturale riguardo la lunghezza massima di 512 token per ciascun input. Pertanto, articoli eccessivamente lunghi devono essere necessariamente troncati, con conseguente possibile perdita di informazioni rilevanti per la classificazione.

Repository del progetto

Il codice sorgente, i modelli addestrati e gli script utilizzati per il fine-tuning e l'analisi sono disponibili pubblicamente al seguente indirizzo GitHub:
https://github.com/UNIVPM-DataScience/Bert-fake-news.git