UNIVERSITÀ DEGLI STUDI DI SALERNO

Progetto di Intelligenza Artificiale

Studente	**Matricola**
Scaparra Daniele Pio	0512116260
Fasolino Pietro	0512116473
Vitulano Antonio	0512116776

Link alla repository GitHub:
https://github.com/dscap02/EmotionsReleave

Documento di Modeling e Evalution

Indice

1 Introduzione

Adesso trattiamo le due fasi successive a quelle di analisi e preparazione dati, cioè le fasi di **modeling** ed **evaluation**.

Con il **modeling**, ci concentriamo ad ideare l'algoritmo per risolvere il problema in un determinato contesto, in questa fase ci rendiamo conto se le opportune operazioni eseguite precedentemente sui dati risultano utili nella stesura di un modello. Dopo aver sviluppato il modello, si passa alla fase di addestramento, in cui si configurano i suoi parametri, si addestra e si descrivono i suoi risultati. Con l'**evaluation**, andiamo a valutare in base ai risultati ottenuti nella fase di addestramento, se rispecchiano gli obiettivi prefissati (**business goal**) durante la prima fase, per controllare la consistenza e solidità della soluzione sviluppata.

2 Struttura del documento

Questo documento è stato strutturato in maniera particolare, perché dato che siamo più persone a lavorare sul progetto, abbiamo deciso che un membro del gruppo doveva lavorare su un proprio modello in modo tale da far vedere diversi algoritmi di intelligenza artificiale, e in seguito di scegliere il migliore. Quindi, il documento è strutturato nel seguente modo: Prima è presente il modello 1, in seguito il modello 2 e infine nella parte di evaluation spieghiamo qual è stata la nostra scelta.

3 Metodo 1

La selezione degli algoritmi di machine learning è stata effettuata considerando la natura dei dati e il tipo di output richiesto. Data la necessità di classificare testi in 28 categorie emozionali distinte, si è optato per modelli basati su deep learning, in particolare reti neurali transformer come BERT, data la loro comprovata efficacia nel Natural Language Processing (NLP). Abbiamo scelto BERT perché è in grado di capire il contesto. Di seguito andremo a spiegare come funziona BERT.

4 BERT

BERT (Bidirectional Encoder Representations from Transformers) è un modello di deep learning basato sull'architettura Transformer, progettato per comprendere il significato del testo analizzandolo in modo bidirezionale. Questo significa che, a differenza dei modelli tradizionali di NLP che analizzano il testo in modo unidirezionale (da sinistra a destra o da destra a sinistra), BERT considera l'intero contesto della frase, sia prima che dopo ogni parola. BERT si basa su diversi passaggi fondamentali che gli permettono di elaborare e comprendere il testo in maniera efficace:

4.1 Tokenizzazione con WordPiece

Il testo viene suddiviso in sottounità (subwords) tramite la tecnica *WordPiece*, che spezza le parole meno comuni in segmenti più piccoli. Ad esempio, la parola 'unhappiness'

potrebbe essere suddivisa in `["un", "happiness"]`, consentendo una gestione efficiente di parole non comuni.

4.2 Embedding contestuali

Ogni token del testo viene convertito in una rappresentazione numerica (*embedding*) che tiene conto del contesto bidirezionale, ovvero di tutte le parole circostanti.

4.3 Pre-training su due compiti principali

BERT viene pre-addestrato su enormi quantità di testo utilizzando due obiettivi principali:

4.3.1 Masked Language Model (MLM)

Durante l'addestramento, una parte delle parole nella frase (tipicamente il 15%) viene sostituita da un token speciale `[MASK]`, e il modello deve prevedere la parola corretta basandosi sul contesto.

- **Esempio:**
  - Input: 'I love `[MASK]` food.'
  - Output previsto: 'I love Italian food.'

4.3.2 Next Sentence Prediction (NSP)

Al modello vengono date due frasi, e deve determinare se la seconda frase segue logicamente la prima.

- **Esempio:**
  - Frase A: 'I went to the store.'
  - Frase B: 'I bought some milk.'
  - Output previsto: 'True' (le frasi sono collegate logicamente).

4.4 Fine-tuning su compiti specifici

Dopo il pre-training su un grande corpus di dati generali (ad esempio Wikipedia e libri), BERT può essere adattato a compiti specifici (come la classificazione delle emozioni) mediante un ulteriore addestramento su dataset più piccoli e specifici, aggiornando i suoi pesi in base ai nuovi esempi.

5 Addestramento

L'addestramento del modello è stato condotto utilizzando il dataset preparato, con l'obiettivo di ottimizzare i parametri chiave e valutare le prestazioni su differenti configurazioni. Per garantire la migliore accuratezza possibile, è stato adottato un approccio iterativo, testando diverse combinazioni di parametri e monitorando le prestazioni in ogni fase.

5.1 Preparazione dei dati

Il dataset è stato suddiviso in due parti principali:

- **Training set (80%)**: utilizzato per addestrare il modello, permettendogli di apprendere le caratteristiche dei dati.
- **Test set (20%)**: utilizzato per valutare le prestazioni del modello su dati mai visti, verificando la sua capacità di generalizzazione.

Inoltre, i testi sono stati preprocessati e tokenizzati utilizzando il tokenizer di BERT.

5.2 Processo di addestramento

Durante l'addestramento, il modello ha iterato attraverso il dataset di training in più epoche. Ad ogni iterazione, il modello ha calcolato la perdita (*loss*) confrontando le previsioni con le etichette reali, e ha aggiornato i suoi pesi utilizzando la discesa del gradiente.

Il processo è stato monitorato misurando la perdita su ogni epoca, con i seguenti risultati:

- **Epoca 1**: Perdita iniziale elevata, miglioramento progressivo.
- **Epoca 2**: La perdita è diminuita, segnalando un apprendimento efficace.
- **Epoca 3**: Ulteriore riduzione della perdita, senza segni evidenti di overfitting.

6 Metodo con applicazione di Naive Bayes

Una soluzione alternativa può essere l'utilizzo di un modello basato sul calcolo delle probabilità e applicazione del teorema di Bayes.

Come fase inziale viene eseguito un processo di normalizzazione del testo,utilizzando la libreria di nltk,che permette di tokenizzare il testo del messaggio ,rimuovendo punteggiatura e stop words,così in modo da ottenere un testo formattato senza possibilità di fattori che influenzano la valutazione del messaggio.

La formula principale sviluppata grazie anche all'applicazione del teorema di Bayes è la seguente:

- **Formula Naive Bayes:**

$$P(Emozione) = \frac{P(Emozione) * P(Parola|Emozione)}{P(Parola)}$$

L'idea è basata sul calcolo di probabilità legate alle frequenze,quindi vengono calcolate inzialmente la frequenza dell'emozione nel dataset,la frequenza della parola in base a quell'emozione,calcolata in base alla funzione descritta in seguito,e la frequenza della parola in relazione a tutto il dataset.

Poi è stata definita anche una funzione,come anticipato precedentemente, che permette di ottenere dal dataset,le parole di tutte le istanze così da associarle a quella determinata emozione,tutto ciò mappato in un vocabolario.

Successivamente sono state calcolate le varie probabilità,ecco le formule:

- **Probabilità dell'emozione**
  - probabilità calcolata mediante la frequenza dell'emozione sul train set,calcolata con l'utilizzo di una funzione secondaria;

$$P(Emozione) = \frac{frequenza\_emozione}{istanze\_dataset}$$

- **Probabilità della parola in base all'emozione**
  - probabilità calcolata in base a frequenza calcolata sul vocabolario mediante funzione, ottenuto con la funzione secondaria descritta precedentemente;

$$P(Parola \mid Emozione) = \frac{frequenza\_parola\_istanze\_emozione + 1}{totale\_parole\_istanze\_emozione + |frequenza\_totale\_par}$$

- **Probabilità della parola in base al dataset**
  - probabilità calcolata in base alla frequenza ottenuta mediante funzione definita che restituisce il numero di istanza totali contenenti quella determinata parola.

$$P(Parola) = \frac{frequenza\_parola}{totale\_parole\_dataset}$$

Di seguito viene riportata la funzione,che prende come parametri il messagio,già preprocessato,e l'emozione,che applica sulla base di queste probabilità definite,la formula principale definita precedentemente.In questa funzione si ottiene inzialmente il messaggio,da cui vengono ottenute le varie parole.

Si applica il calcolo della probabilità ,definita precedentemente,di quella parola legata a quell'emozione,e si moltiplicano e si salva la probabilità totale,derivante dalla moltiplicazione delle probabilità calcolate su tutte le parole.

Successivamente si calcola per ogni parola,la probabilità legata alla parola in relazione al dataset,ottenendo e salvando una probabilità finale derivante dalla moltiplicazione delle varie probabilità calcolate.

Come ultima probabilità si calcola quella relativa all'emozione,definita precedentemente,calcolata in relazione all'intero dataset e viene salvata.

Infine viene applicata la formula principale sulla base di queste probabilità calcolate,e viene restituita la probabilità dell'emozione in base al messaggio.

Per ogni messaggio viene calcolata la probabilità in relazione a ogni emozione e restituita quella che risulta essere la probabilità maggiore,in base a cui il modello predice l'emozione predominante.

Ci sono alcune probabilità che possono incidere molto sul calcolo della probabilità finale,come ad esempio la probabilità della parola in base all'emozione,che ad esempio una o più parole,del messaggio,riportano una frequenza elevata in base quell'emozione,ciò potrebbe già determinare una possibile predizione candidata come output per il modello.

Per quanto riguarda la fase di **addestramento**,abbiamo effettuato una divisione del dataset in due parti,utilizzando la libreria di **scikit-learn**,che ti permette velocemente di effettuare quest'operazione.Il **training set**,su cui il modello si addestra

e ricava le frequenze necessarie al calcolo delle probabilità e **test set**,per valutare la predizione del modello in base ai calcoli delle probabilità,che compongono quella finale, su nuovi messaggi.

7 Valutazione dei due modelli

Per valutare i modelli si è scelto di usare la metrica **accuracy**,che nel caso del primo modello è dell'84 percento,inoltre abbiamo scelto di valutare anche il **tempo di risposta** impiegato dai modelli,per quanto riguarda il primo modello utilizza un tempo di risposta ragionevole e basso,un aspetto fondamentale dato che i modelli devono analizzare chat composte da un certo numero di messaggi.

Per quanto riguardo il secondo modello,**l'accuracy** risulta eqivalente al 47 percento,come risultato non è poi così negativo,considerando che il modello è stato sviluppato da zero.In ogni modo sicuramente c'è bisgono di applicare dei miglioramenti,per renderlo ancora più efficiente e aumentarne l'accuratezza,poichè l'idea è stata sviluppata e poi non utilizzata.

Un possibile problema può derivare dal fatto che il modello utilizza molte risorse di calcolo,quando va a ricavare le varie probabilità e poi quella finale su ogni messaggio per ogni emozione,così da indurre a un **tempo di risposta** maggiore,ma non così elevato,rispetto al modello descritto precedentemente. Mettendo a confronto le due accuracy risultanti e tempi di risposta,sarebbe preferibile usare un modello più performante,sulla base di questo confronto si è scelto di usare il primo modello come soluzione principale.