UNIVERSITÀ DI PISA

Relazione progetto di Text Analytics

Twitter US Airline

Biviano Matteo
Currao Federica
Racioppa Arianna

Anno Accademico 2020/2021

Indice

1 Introduzione

Twitter è un sistema di "microblogging" che consente di inviare e ricevere brevi posts chiamati *tweets*. Questo sistema rappresenta quindi, per le imprese, una buona fonte di feedback dei clienti, utili a migliorare i propri prodotti e servizi, attraverso anche tecniche di **Sentiment Analysis**. La Sentiment Analysis è un tipo di problema di elaborazione del linguaggio naturale che determina il sentimento, l'opinione o l'emozione di un testo. Spesso applicato per estrarre intuizioni dei sentimenti dei clienti che utilizzano un prodotto o servizio, utile per le aziende che vogliono monitorare il sentimento pubblico dei loro marchi. L'obiettivo di questo progetto è confrontare diversi metodi di etichettatura (debole e non) per implementare e valutare *Sentiment Classifiers*.
Dopo una fase preliminare di analisi e preprocessing dei dati, presente in Sezione [2 - 3], viene mostrata in Sezione [4] una rappresentazione visiva utile per ottenere informazioni dai dati di testo non strutturati. É possibile esaminare interattivamente queste analisi utilizzando la web app discussa in Sezione [5]. I sentimenti originali dei dati sono stati confrontati, in Sezione [6], con i sentimenti ottenuti da due metodologie di *weak labeling*: **VADER** e **TextBlob**. Successivamente, sono stati testati diversi classificatori divisibili in tre gruppi: **SimpleClassifiers** (*DecisionTree*, *KNearestNeighbor*, *NaiveBayes*, *LinearSVC*, *SVC*); **LSTM** (con embeddings *GloVe* e senza); **BERT**.

2 Data Understanding

Il dataset utilizzato per questo studio è lo US Twitter Airline, il quale consiste in una collezione di 14640 tweets, scritti da 7701 utenti e riguardanti le 6 principali compagnie aree statunitensi: *United*, *US Airways*, *American*, *Southwest*, *Delta* e *Virgin America*. I tweet sono stati raccolti dal 16 al 24 Febbraio 2015, periodo nel quale ai clienti è stato chiesto di etichettare i tweet come *positivi*, *negativi* e *neutri* in base alla polarità del sentimento, definendo, in caso di valutazione negativa, la motivazione della stessa (ad esempio *Late Flight* o *Rude service*). Il dataset contiene 20 features riguardanti informazioni sul cliente, sulla pubblicazione del tweet e sul sentimento espresso (informazioni tipiche dei progetti di *crowdsourcing*). Di queste, ai fini dello studio, sono state utilizzate le 4 seguenti:
- **text**: il testo del tweet;
- **airline**: nome della compagnia cui fa riferimento il tweet;
- **airline_sentiment**: il sentimento del tweet espresso dal cliente (*positive*, *negative*, *neutral*);
- **negativereason**: motivazione della valutazione negativa data alla compagnia.

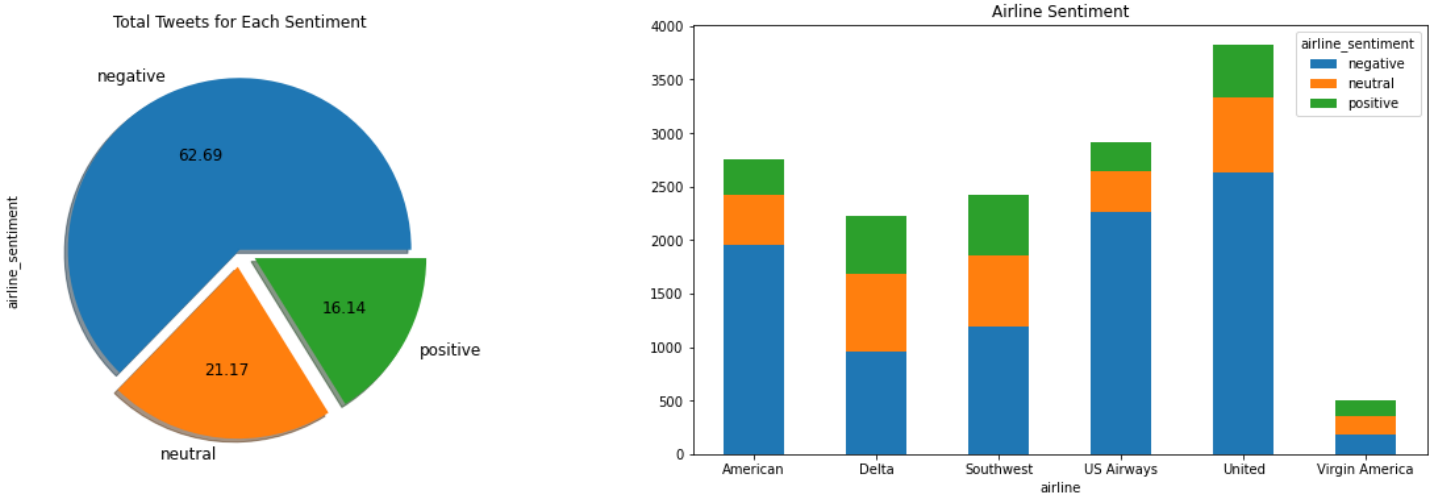


Figura 1: Distribuzione del sentimento

Come visibile in Figura [1] (a sinistra), il dataset risulta sbilanciato rispetto al sentimento riportato, infatti circa il 63% dei tweet corrispondono a valutazioni negative (pari a 9178 tweet). L'elevato numero di valutazioni negative è probabilmente dovuto al fatto che le persone generalmente utilizzano i social network per trasmettere le proprie opinioni insoddisfacenti riguardo un prodotto o servizio. A destra è possibile notare come il maggior numero di tweet negativi riguarda la compagnia *United*, la quale ha anche il numero maggiore di tweet.
Non tutti i tweets valutati come *negativi* contengono una motivazione della valutazione, infatti circa il 13% delle motivazioni non sono state specificate (identificabili dalla dicitura *Can't Tell* espressa dal cliente). Le tre motivazioni più frequenti sono risultate essere *Customer Service Issues*, *Late Flight* e *Cancelled Flight*. In particolare, *Customer Service Issues* è risultata essere la "negative reason" principale per le compagnie US Airways, United, American, Southwest e Virgin America; mentre *Late Flight* è il problema più frequente riportato nei tweets associati alla compagnia Delta. Virgin America ha il minor numero di tweet e, di conseguenza, anche di opinioni negative. Utilizzando il framework presentato in Sezione [5] è possibile ripercorrere agevolmente le analisi precedentemente descritte e fare esplorazioni più dettagliate come "Esiste una relazione tra sentimenti negativi e la data in cui è stato pubblicato il tweet?". Ad esempio, è interessante notare che American ha avuto un improvviso aumento di tweet negativi il 22 Febbraio 2015, successivamente dimezzati dopo solo due giorni

3 Data Preprocessing

In questa sezione vengono analizzate le metodologie applicate per preprocessare i dati, in modo da migliorare l'efficienza dei modelli considerati nel progetto. I procedimenti eseguiti in questa fase, riportati nel grafo in Figura [2], prendono in considerazione quanto analizzato in [5]. Twitter offre la possibilità di includere all'interno dei tweet riferimenti ad altri utenti, ad url o hashtag, i quali sono stati sostituiti come riportato in Figura [2]. Sono stati, inoltre, sostituiti gli emoji, i numeri e la punteggiatura rimossa. Il testo è stato quindi normalizzato convertendo tutti i caratteri in minuscolo, rimuovendo le *stopwords* e le *shortwords* (parole lunghe 1 carattere). Infine, il testo è stato lemmatizzato. In particolare, la lemmatizzazione è stata preferita allo *stemming* poichè quest'ultimo, non controllando la validità della radice prodotta, avrebbe generato in alcuni casi parole prive di significato per la lingua considerata [2].

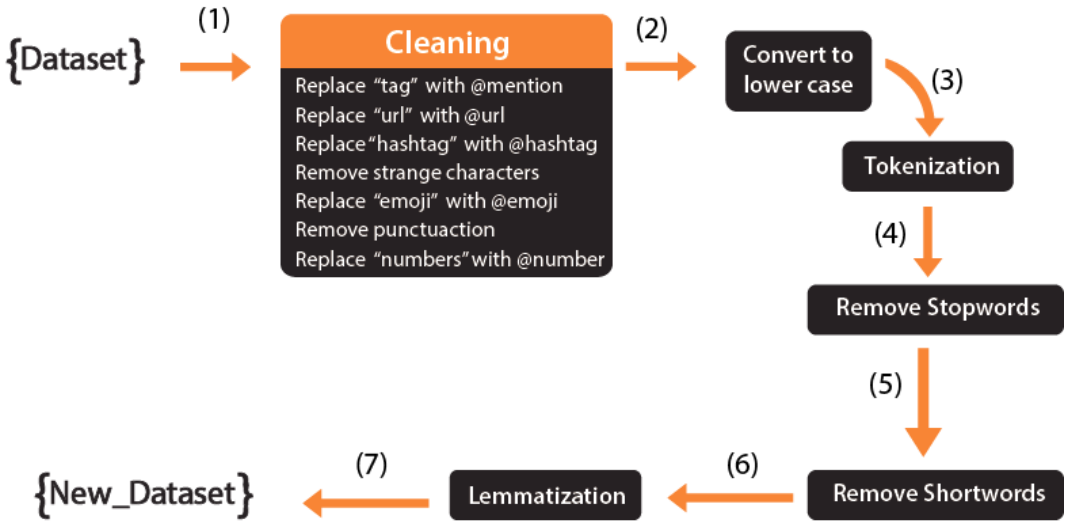


Figura 2: Data Preprocessing

4 Word Cloud

Dopo aver preprocessato i dati, è stato utilizzato il metodo **Word Cloud** per comprendere meglio la differenza di sentimenti per i tweet positivi, negativi e neutrali, permettendo di vedere quanto bene sono distribuiti i sentimenti nei dati. Una WordCloud è una visualizzazione in cui le parole più frequenti appaiono con dimensioni più grandi rispetto alle parole meno frequenti. Può risultare un'opzione eccellente per aiutare a interpretare visivamente il testo ed è utile per ottenere rapidamente informazioni sugli elementi più importanti in un dato testo. Viene utilizzato nel campo dei social media, dove vi è la necessità sempre crescente di analizzare le enormi quantità di testi. In questo caso è stata considerata una versione dei dati priva degli elementi introdotti in Sezione [3], i quali risultano tra le keywords più frequenti, le cui occorrenze sono:

- @mention: 16292;
- @number: 5694;
- @hashtag: 3522;
- @url: 1211;
- @emoji: 1166.

In Figura [3], è possibile notare (a sinistra) come per i tweets positivi risaltano le parole: *thank*, *great*, *love*, *awesome*, in contrasto con i negativi (al centro) le cui parole sono: *help*, *delay*, *cancel flightled*, seppur meno evidenti.

Figura 3: WordCloud: Positive - Negative - Neutral

É stata, inoltre, analizzata la frequenza dei bigrammi, ottenendo come più frequenti *customer service* (564 occorrenze) e *cancel flightled* (487 occorrenze). É stata anche osservata la frequenza per gli 8 topics presenti nelle *negative reason* riportate dagli utenti, escludendo i tweet per i quali non è stata riportata una ragione specifica attraverso la dichiarazione *Can't Tell*. I 3 più frequenti per ogni topic sono:

Negative Reason	Top 3 bigram (number of occurrences)		
Late Flight	late flight (139)	flight delay (128)	delay flight (75)
Flight Attendant Complaints	flight attendant (45)	gate agent (37)	check bag (10)
Customer Service Issue	customer service (402)	hold hour (99)	call back (96)
Flight Booking Problems	book problem (33)	book flight (31)	flight book (27)
Cancelled Flight	cancel flightled (355)	cancel flight (241)	flight cancel (188)
Longlines	still wait (7)	people line (6)	customer service (6)
Bad Flight	ive ever (10)	pay extra (8)	wifi flight (8)
Damaged Luggage	baggage claim (3)	destroy luggage (2)	throw bag (2)
Lost Luggage	lose bag (44)	lose luggage (29)	baggage claim (22)

5 Dashboard

Per poter esaminare interattivamente le analisi effettuate nelle prime sezioni di questo report, è stata creata una web app utilizzando il framework Streamlit, tramite il quale è possibile, infatti, creare un'interfaccia utente utile per visualizzare in modo efficace i key performance indicators dei dati. Inoltre, il framework viene fornito con il server Web integrato e consente di eseguire la distribuzione nel container Docker.

5.1 Esecuzione del framework

É possibile eseguire manualmente il framework attraverso questa serie di comandi:

```
$ pip install streamlit
$ cd Dashboard
$ streamlit run dashboard.py
```

Dopo aver effettuato il *run*, il server localhost si aprirà automaticamente nel browser. All'interno del file *requirements.txt* sono state specificate le versioni delle librerie utilizzate, con le quali è stato sviluppato il framework.
Grazie all'uso del servizio di deployment del framwork, è possibile, inoltre, visualizzare la dashboard online su TweetsAirline-Dashboard.

6 Weak Labeling

La tecnica del weak labeling ha lo scopo di ridurre i costi e aumentare l'efficienza degli sforzi umani impiegati nell'etichettatura manuale dei dati. L'inconveniente principale di questo approccio è che il modello si preoccupa delle singole parole ed ignora il contesto in cui vengono utilizzate. Sono stati quindi confrontati i risultati di due metodi di *weak labeling*, Vader e TextBlob, rispetto all'etichettatura umana riportata nel dataset.

6.1 Metodologie utilizzate

VADER (Valence Aware Dictionary and Sentiment Reasoner) è uno strumento di sentiment analysis rule-based [1] che risulta essere particolamente adatto ad analisi su sentimenti espressi nei social media. Questo strumento utilizza un elenco di features lessicali che sono etichettate come positive, negative o neutre in base al loro orientamento semantico per calcolare il sentimento del testo. VADER, applicato al tweet, produce quattro metriche *Positive*, *Negative*, *Neutral* (rappresentanti la proporzione del testo che rientra in quelle categorie) e *Compound*. Il Compound viene calcolato sommando i punteggi di valenza di ogni parola nel lessico, variando tra -1 ed 1. Questa è la metrica più utile se si desidera una singola misura unidimensionale del sentimento per una determinata frase.
Un altro strumento di sentiment analysis è **TextBlob** [3] il quale restituisce per ogni frase due proprietà:
- La polarità, compresa tra [-1, 1], dove -1 indica sentiment negativo e +1 sentiment positivo;
- La soggettività, compresa tra [0, 1]. Più piccolo è il valore più è probabile che la frase sia un'opinione. Maggiore è il valore, più è probabile che la frase rappresenti un'informazione "fattuale".

Textblob, inoltre, ignora le parole che non conosce, considerarando le parole e le frasi a cui può assegnare la polarità e calcolare le medie per ottenere il punteggio finale.

6.2 Confronto risultati

I modelli sono stati applicati sia ai dati originali sia al testo preprocessato (ottenuto in Sezione [3]) al fine di valutare eventuali miglioramenti. In Figure [4 - 5] vengono riportati i risultati ottenuti per due soglie di threshold **0.05** e **0.10** e le metriche *precision*, *recall* ed *f1-score* per la classe minoritaria (*positive*).

	Original text		Preprocessed text	
Threshold	0.05	0.10	0.05	0.10
Accuracy	0.543	0.541	0.497	0.466
Precision	0.337	0.349	0.337	0.332
Recall	0.871	0.869	0.871	0.849
F1 score	0.486	0.498	0.486	0.477

Figura 4: VADER

	Original text		Preprocessed text	
Threshold	0.05	0.10	0.05	0.10
Accuracy	0.453	0.436	0.396	0.382
Precision	0.342	0.358	0.293	0.305
Recall	0.759	0.741	0.601	0.588
F1 score	0.471	0.483	0.394	0.402

Figura 5: TextBlob

É possibile notare come entrambi i modelli hanno un'accuracy inferiore al modello triviale, infatti, predicendo ogni tweet con sentimento negativo si avrebbe un'accuracy del 62% a causa dello sbilanciamento della classe. In particolare, l'accuracy risulta sempre più alta sul testo originale rispetto al testo preprocessato (non risultando una metodologia utile in questo contesto). Tra i due metodi, VADER ha riportato risultati leggermente superiori a TextBlob per tutte le metriche.

7 Sentiment Analysis

La parte di *Sentiment Analysis* del progetto è stata effettuata sia prendendo in considerazione classificatori base che metodologie più complesse quali *LSTM* (con embedding *GloVe* e senza) e *BERT*. Per poter valutare le prestazioni dei modelli il dataset è stato splittato in training e test (secondo la regola 70-30, mantenendo la proporzione di target nei due insiemi per non indurre un bias nel modello) e la variabile target è stata trasformata, tramite encoding, in variabile numerica. L'analisi è stata effettuata sia considerando il target originale sia escludendo i tweet neutrali effettuando classificazione binaria. Dalla classificazione binaria sono stati ottenuti, chiaramente, risultati complessivamente superiori a causa della difficoltà (nel caso di classificazione multiclasse) da parte dei classificatori di riconoscere la classe "neutral" per l'assenza di termini che oggettivamente si riferiscono a questo sentimento.

7.1 Simple Classifiers

In questa fase di analisi sono stati testati alcuni algoritmi tradizionali tra cui *Naive Bayes*, *DecisionTree*, *KNearestNeighbor* ed *SVM*. I dati sono stati preprocessati attraverso la seguente pipeline di eventi:
- CountVectorizer, con cui il testo è stato vettorizzato per trasformarlo in una matrice document-term.
- SelectKBest, con il quale è stata effettuata la feature selection in base alle statistiche del chi-quadro (misura della dipendenza tra le variabili).
- TfidfTransformer, per trasformare la matrice in una rappresentazione normalizzata di valori TF/IDF.

É stata effettuata una fase preliminare di validation, in modo da scegliere gli iperparametri che massimizzassero la **f1_macro** per garantire un buon equilibrio tra precision e recall. Questo perchè, a causa dello sbilanciamento del dataset usare metriche come l'accuracy avrebbe potuto comportare ottimi valori, ma modelli privi di informazioni.

In questo contesto è stato utilizzato l'algoritmo **GridSearchCV**, nel quale il numero di fold è stato ottenuto tramite una **Stratified 3-Fold Cross Validation**. In Figura [6] vengono riportati i risultati ottenuti per ogni classificatore rispetto alle medie *macro* di ogni metrica. Le performance ottenute dai classificatori (di cui *SVC* è risultato migliore) sono buone, ma non ottime, ciò a dimostrare quanto il compito di inferenza del sentimento è un compito molto difficile, poichè richiede ai modelli di comprendere il contesto sottostante delle frasi. In particolare, per ogni classificatore sono state ottenute performance inferiori per quanto riguarda i tweet con sentimento neutrale, come da aspettative. In Figura [7] vengono riportati i risultati di classificazione binaria, dai quali è possibile notare un incremento consistente di prestazioni per tutti i classificatori. Come nella prova precedente, anche in questo caso l'*SVC* è risultato il migliore in termini di performance.

	Accuracy	Precision	Recall	F1
NaiveBayes	0.76	0.75	0.61	0.65
KNN	0.71	0.66	0.61	0.63
DecisionTree	0.70	0.63	0.60	0.62
LinearSVC	0.78	0.73	0.68	0.70
SVC	0.78	0.74	0.67	0.70

Figura 6: SimpleClassifier: risultati target multiclasse

	Accuracy	Precision	Recall	F1
NaiveBayes	0.90	0.87	0.80	0.83
KNN	0.88	0.83	0.79	0.81
DecisionTree	0.88	0.82	0.79	0.80
LinearSVC	0.92	0.89	0.85	0.86
SVC	0.91	0.88	0.84	0.86

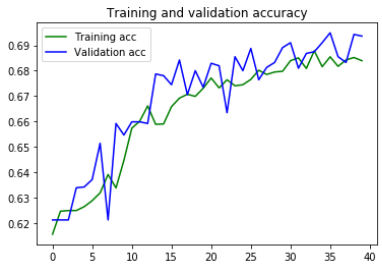
Figura 7: SimpleClassifier: risultati target binario

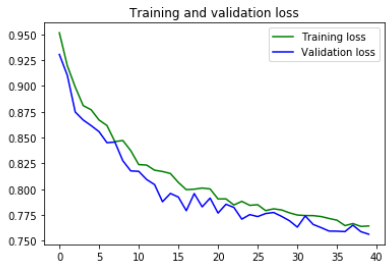
7.2 LSTM

Un'unità LSTM (*Long Short-Term Memory*) è un'estensione di una RNN (*Recurrent Neural Network*). Una rete neurale ricorrente prende in input non solo lo stato corrente, ma anche quanto percepito in precedenza, ed è progettata per riconoscere schemi in dati quali testo, dati numerici di serie temporali, ecc.
In questa fase sono state testate diverse configurazioni di LSTM (con embedding standard e con embedding GloVe) sia per il target originale che per la versione binaria. In particolare, è stato notato che modelli più complessi portavano a maggior overfitting. É stato quindi preferito considerare modelli più semplici eseguiti su 40 epoche.

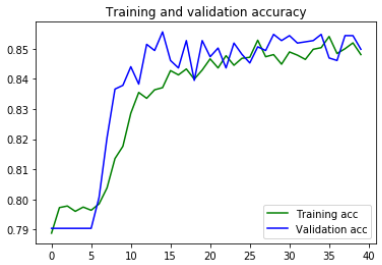
7.3 LSTM con embedding Standard

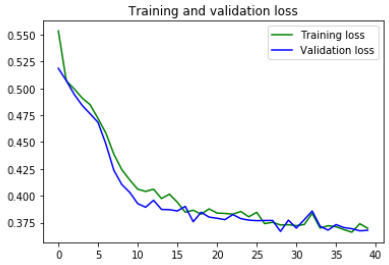
Il modello testato è stato creato utilizzando un singolo livello core di tipo LSTM con 64 unità. In Figura [8] vengono riportati i risultati ottenuti con questo modello sia per il target multivariato (in alto) che per il target binario (in basso). É possibile notare come in entrambi i casi le curve della loss (per training e validation) tendono a coincidere, diminuendo relativamente alla stessa velocità. In particolare, nel caso di classificazione binaria, la differenza di andamento dell'accuracy per training e validation tende a ridursi dopo le prime 15 epoche.





Accuracy_Training	0.70
Accuracy_Test	0.68
Precision	0.64
Recall	0.50
F1	0.53



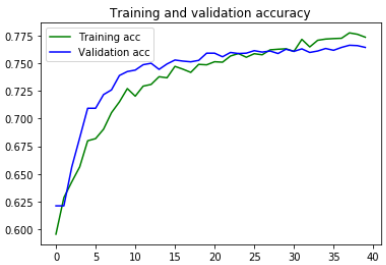


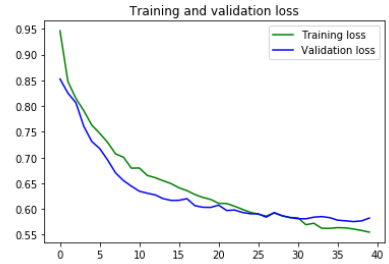
Accuracy_Training	0.86
Accuracy_Test	0.85
Precision	0.79
Recall	0.69
F1	0.72

Figura 8: LSTM standard: risultati

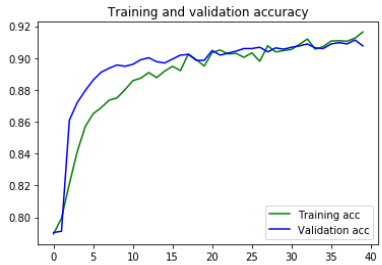
7.4 LSTM con embedding GloVe

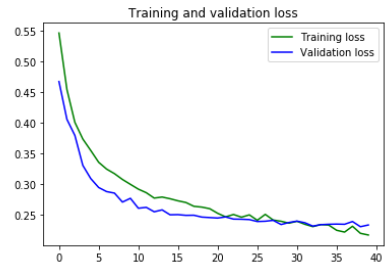
Al fine di aumentare l'accuratezza del modello, è stata testata una sua versione modificata, utilizzando un livello embedding preaddestrato con GloVe e un livello LSTM con 32 unità. I risultati ottenuti sono stati riportati in Figura [9]. Rispetto ai risultati della versione senza GloVe è possibile osservare non solo l'aumento di prestazioni ottenute, ma che i valori di accuratezza per training e validation tendono a coincidere a partire da un numero di epoche inferiori al caso precedente. Tuttavia, un lieve overfitting (sia in termini di accuracy che di conseguente loss) si può notare superate le 30 epoche.





Accuracy_Training	0.79
Accuracy_Test	0.76
Precision	0.71
Recall	0.65
F1	0.67





Accuracy_Training	0.92
Accuracy_Test	0.90
Precision	0.84
Recall	0.84
F1	0.84

Figura 9: LSTM GloVe: risultati

8 BERT

BERT (*Bidirectional Encoders Representations from Transformer*) è un modello di comprensione del linguaggio naturale di ultima generazione, della famiglia dei *Transformers*. A differenza dei modelli unidirezionali che leggono il testo in ingresso in sequenza (da sinistra a destra o da destra a sinistra), BERT fa uso di un encoder Transformer

che legge l'intera sequenza di parole contemporaneamente, pertanto è considerato bidirezionale. Questa caratteristica consente al modello di apprendere il contesto di una parola in base a tutto ciò che lo circonda (a sinistra e a destra della parola).

8.1 Pre-processing

I dati sono stati pre-elaborati con la stessa metodologia espressa in Sezione [3]. Tuttavia, per poter eseguire il modello, i dati necessitano di ulteriori elaborazioni, effettuate tramite l'uso della classe *BertTokenizer* fornita da HuggingFace. In particolare, è stato utilizzato il metodo **encode_plus**, il quale incapsula i seguenti passaggi:

1. Tokenizzazione delle frasi.
2. Inserimento all'inizio di ogni frase del token speciale [CLS]. Questo è l'incorporamento che verrà utilizzato dal classificatore, restituito come output del 12-esimo livello transformer.
3. Inserimento alla fine di ogni frase del token speciale [SEP].
4. Aggiunta di tokens di padding [PAD] per far raggiungere ad ogni frase la lunghezza massima.
5. Mapping dei token con gli indici del vocabolario *bert-base-uncased* del Tokenizer.
6. Creazione delle attention mask, le quali differenziano i token reali dai token di padding.

É stato decisio di utilizzare la lunghezza massima di 25 tokens poichè, analizzando la distribuzione presente in Figura [10], questo numero avrebbe evitato di troncare frasi, rimanendo comunque molto al di sotto della soglia massima di 512 tokens definita dal modello (quindi non appesantendo computazionalmente il modello).

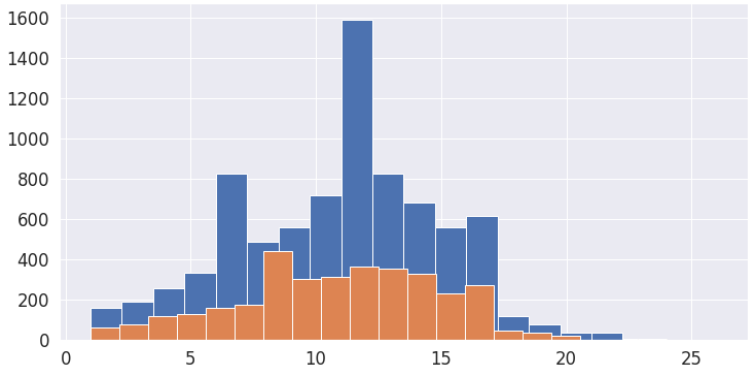


Figura 10: Distribuzione delle parole: training e test

Il seguente è un esempio di come vengono pre-elaborati i dati:

- Frase originale: "@mention ok may keep @number lose bag info longer trust bad way handle"
- Embedding: ['**[CLS]**', '@', 'mention', 'ok', 'may', 'keep', '@', 'number', 'lose', 'bag', 'info', 'longer', 'trust', 'bad', 'way', 'handle', '**[SEP]**']
- Mapping dopo padding: [**101**, 1030, 5254, 7929, 2089, 2562, 1030, 2193, 4558, 4524, 18558, 2936, 3404, 2919, 2126, 5047, **102**, 0, 0, 0, 0, 0, 0, 0, 0]
- Attention Mask: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]

Per il dataset è stato, inoltre, creato un iteratore utilizzando la classe *DataLoader*. A differenza di un ciclo *for*, con un iteratore non è necessario caricare in memoria l'intero dataset, permettendo di risparmiare sulla memoria durante la fase di addestramento. Il modello utilizzato è stato il modello BERT base (**BertForSequenceClassification**), il quale presenta un singolo strato lineare aggiunto in cima per poter effettuare la classificazione. Per la fase di validazione del modello è stata utilizzata la *validation loss* in quanto più precisa dell'*accuracy*, potendo cogliere

eventuali previsioni corrette con minore confidenza. Attraverso questa metrica è stato possibile capire come un numero maggiore di 2 epoche avrebbe comportato un eccessivo adattamento dei dati sul training (all'aumentare delle epoche infatti la loss sul training tendeva a diminuire, mentre quella sul validation ad aumentare).

8.2 Risultati

Il modello è stato testato, come nelle analisi precedenti, sia sul dataset con il target originale che sulla versione binaria. I risultati ottenuti, riportati in Figura [11], sono risultati leggermente migliori rispetto ai modelli testati in precedenza. Anche in questo caso, il classificatore non riesce ad etichettare correttamente i dati quando si considerano i tweets con sentimento *neutrale*.

	Negative	Positive
Accuracy	0.92	
Precision	0.95	0.81
Recall	0.95	0.80
F1	0.95	0.80

	Negative	Positive	Neutral
Accuracy	0.79		
Precision	0.85	0.75	0.63
Recall	0.90	0.73	0.52
F1	0.87	0.74	0.57

Figura 11: BERT: risultati

Riferimenti bibliografici

[1] Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, 2014.

[2] Anjali Ganesh Jivani et al. A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, 2(6):1930–1938, 2011.

[3] Steven Loria. textblob documentation. *Release 0.15*, 2, 2018.

[4] Peter D Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *arXiv preprint cs/0212032*, 2002.

[5] S Vijayarani, R Janani, et al. Text mining: open source tokenization tools-an analysis. *Advanced Computational Intelligence: An International Journal (ACII)*, 3(1):37–47, 2016.