UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI **INGEGNERIA**

Corso di Laurea Triennale in
Ingegneria Informatica e dell'Automazione

emotionAlBERTo: sviluppo di un sistema per il riconoscimento di emozioni nel testo italiano tramite BERT

emotionAlBERTo: development of a system for emotion recognition through BERT in italian texts

Relatrice:
Prof.ssa **Claudia Diamantini**

Tesi di Laurea di:
Lisa Burini

Correlatore:
Dott. **Alex Mircoli**

A.A. **2020** / **2021**

INDICE

INDICE FIGURE

1 INTRODUZIONE

1.1 Contesto

Il progetto si colloca nell'ambito dei Big Data e dell'Emotion Recognition, in particolare viene trattato il riconoscimento di emozioni nel testo italiano.
Negli ultimi anni, grazie ad Internet e ai Social Network, i dati generati in rete sono aumentati esponenzialmente. Questi enormi volumi di dati, denominati Big Data, e differenti per fonte e formato, sono divenuti indispensabili in molteplici ambiti, ad esempio produzione e marketing, analisi dei mercati finanziari, diagnostica clinica e comunicazione politica mirata.
L'Emotion Recognition è una tecnica di processing del linguaggio naturale utilizzata per classificare il testo con una certa emozione (es. gioia).
L'estrazione delle emozioni a partire dai testi è molto utile per le aziende, poiché aiuta a capire meglio le emozioni provate dai consumatori, rendendo più veloce ed accurato il processo decisionale aziendale. Poiché la maggior parte dei lavori in letteratura si occupano di Emotion Recognition per la lingua inglese, si è deciso di incentrare il lavoro di tesi sul testo italiano, per contribuire alla trattazione di queste tecniche anche per la lingua italiana.

1.2 Obiettivi

Gli obiettivi di questa tesi sono:

- costruzione di un dataset di tweets italiani validi al fine del riconoscimento di emozioni;
- sperimentazione di algoritmi di Deep Learning basati su Google BERT (Bidirectional Encoder Representations from Transformers);
- fine-tuning di BERT per la lingua italiana per la classificazione di emozioni.

1.3 Struttura della tesi

Il resto della tesi è organizzato come segue.

Nel secondo capitolo si introdurranno i concetti fondamentali utili alla comprensione del lavoro di tesi, si parlerà quindi di Machine Learning, Deep Learning, Reti Neurali e BERT. Si introdurrà inoltre il lavoro di ricerca "AlBERTo", su cui si basano i risultati di questa tesi. Dopo una breve presentazione del nostro lavoro, verranno esposte le tecniche di valutazione e di pre-processing utilizzate negli esperimenti.

Nel terzo capitolo verranno illustrati i due datasets impiegati nelle prove, di cui uno reperito in rete e uno costruito da noi.

Nel quarto capitolo verranno spiegati tutti gli esperimenti effettuati considerando le seguenti emozioni: gioia, fiducia, tristezza, rabbia, paura, disgusto. Nella prima fase viene costruito un modello utilizzando come training set il dataset "MultiEmotions-It" e come test set il dataset estratto da Twitter; mentre nella seconda fase si invertono il training set e il test set.

Il quinto capitolo è strutturato come il precedente, con la differenza che tutti gli esperimenti sono stati effettuati con sole quattro emozioni: gioia, tristezza, rabbia, paura.

Nel sesto capitolo si traggono le conclusioni di tutto il lavoro, analizzando e discutendo i risultati ottenuti.

2 STRUMENTI

2.1 Tecniche di Machine Learning

2.1.1 Background

2.1.1.1 Machine Learning

Con l'espressione "Machine Learning" ci si riferisce alla capacità di apprendimento automatico di una unità di elaborazione, la quale riesce ad apprendere un'attività attraverso una serie di esempi.

Il Machine Learning è quindi il processo di costruzione di sistemi che migliorano con l'esperienza acquisita tramite i dati. L'estrazione di informazioni dai dati si ottiene attraverso l'utilizzo di modelli probabilistici.

L'apprendimento automatico può essere supervisionato o no, nel primo caso si parla di "Supervised Learning", mentre nel secondo di "Unsupervised Learning". L'apprendimento supervisionato sfrutta dati di training etichettati e la funzione di output può essere continua o discreta; si parla quindi di regressione o classificazione. Nell'apprendimento non supervisionato, invece, i dati di training non sono etichettati e lo scopo è quello di riuscire ad ordinare i dati raggruppandoli in clusters. [1]

Le tecniche di Machine Learning possono essere utilizzate in una grande varietà di campi. Ad esempio, il Machine Learning si utilizza in medicina per la diagnosi del cancro; altri campi di applicazione interessanti sono il rilevamento delle frodi online e la predizione del traffico (**Figura I**).

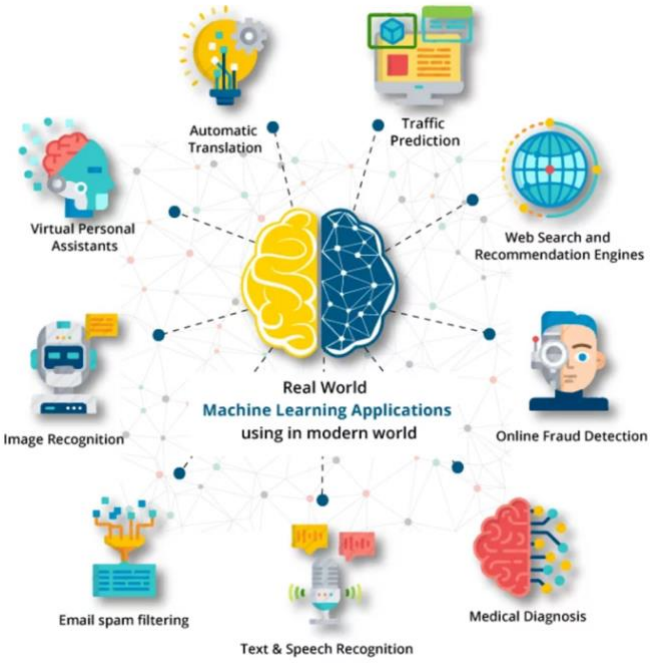


Figura I - Campi di applicazione per il Machine Learning

2.1.1.2 Deep Learning e Reti Neurali

Il Deep Learning è un sottoinsieme del Machine Learning, che a sua volta è un sottoinsieme dell'Intelligenza Artificiale (**Figura II**).

Sia gli algoritmi di Deep Learning che quelli del Machine Learning sono in grado di analizzare grandi quantità di dati, tuttavia varia il tipo di architettura utilizzata per l'analisi. Nello specifico, nel Deep Learning vengono proposte architetture alternative per le reti neurali artificiali [2], le quali sono algoritmi di Machine Learning che si ispirano ad alcune caratteristiche funzionali e cognitive del cervello umano.

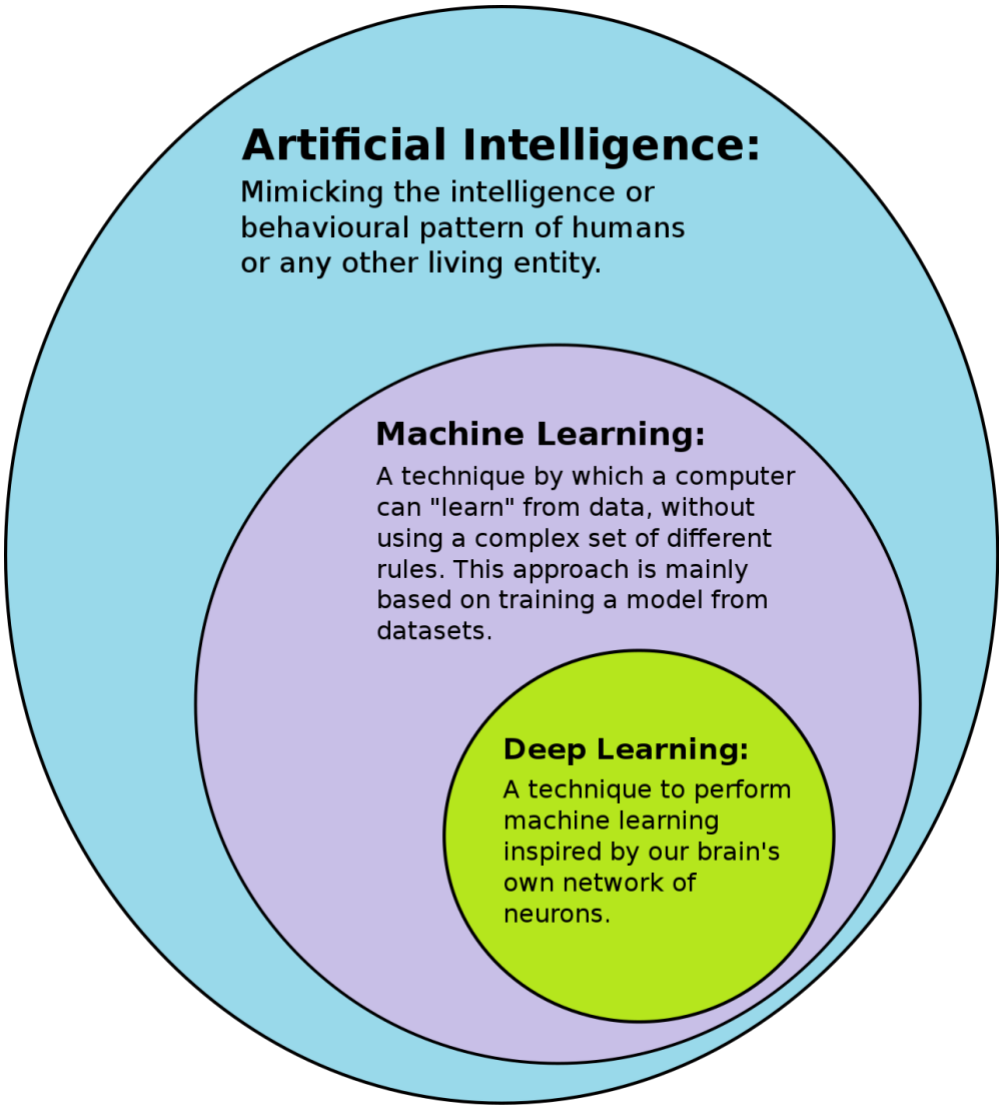


Figura II - Correlazione tra Artificial Intelligence, Machine Learning e Deep Learning

Le reti neurali artificiali sono divenute popolari negli ultimi vent'anni per svariate applicazioni, dalle predizioni finanziarie alla visione artificiale.

Esse sono dei modelli semplificati delle reti neurali biologiche, e vengono utilizzate con tecniche di training supervisionato. [3]

Queste reti sono composte da neuroni artificiali, i quali applicano alla media pesata degli ingressi una trasformazione, detta funzione di attivazione, che può essere lineare o non lineare (**Figura III**).

La tipologia di rete neurale più utilizzata in applicazioni pratiche consiste nella combinazione di molteplici strati nascosti di neuroni.

Durante il training di una rete neurale lo scopo è stimare i pesi che minimizzano l'errore tra le etichette del training set e i valori predetti dalla rete.

La stima dei pesi si ottiene con tecniche note di ottimizzazione, tra cui la più famosa è la "Back Propagation", che sfrutta la discesa del gradiente. [4]

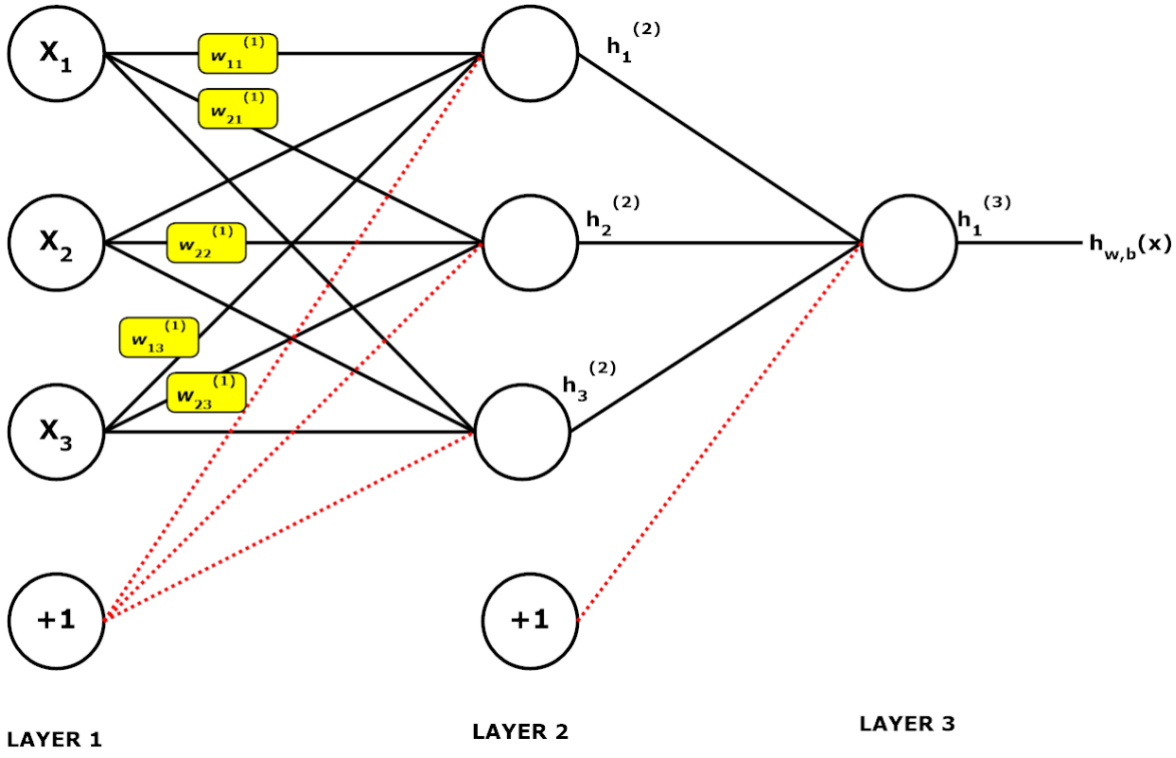


Figura III - Rappresentazione di una rete neurale artificiale

2.1.1.3 BERT

BERT, acronimo di Bidirectional Encoder Representations from Transformers, è un modello per la rappresentazione del linguaggio naturale prodotto da Google. Tramite BERT si è in grado di risolvere una grande varietà di tasks riguardanti il Natural Language Processing (NLP), come:

- Sentiment Analysis;
- Emotion Recognition;
- Text Summarization;
- Question Answering;
- Neural Machine Translation.

Per far sì che BERT riesca a svolgere questi compiti specifici, vanno considerate due fasi: la fase di pre-training e quella di fine-tuning.
Durante la fase di pre-training il modello è allenato su dati non etichettati, mentre per la fase di fine-tuning il modello BERT è inizializzato con i parametri pre-allenati, quindi viene fatto il fine-tuning di tutti questi parametri utilizzando dati etichettati al fine di ottenere un modello fine-tuned per un task specifico (**Figura IV**). [5]

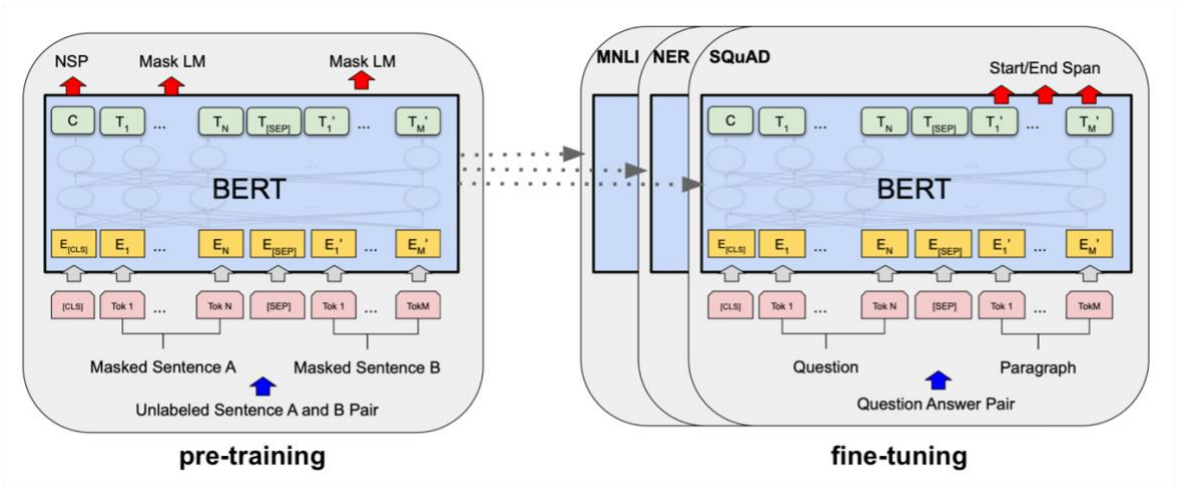


Figura IV - Fasi di pre-training e fine-tuning di BERT [5]

2.1.1.4 AlBERTo

Il progetto "AlBERTo" è un lavoro di ricerca ideato in Italia, volto a costruire un modello equivalente di BERT per il processing del linguaggio italiano.

BERT si è dimostrato il miglior modello per il processing del linguaggio naturale, pertanto è stato deciso di allenare BERT per la lingua italiana al fine di ottenere un modello fine-tuned di BERT: AlBERTo.

In particolare, AlBERTo è specializzato per il processing del linguaggio italiano utilizzato nei Social Network, specificatamente in Twitter.

Il modello AlBERTo è basato sul software BERT distribuito attraverso GitHub da Devlin et al. (2019) con il sostegno di Google.

Al fine di ottenere AlBERTo, BERT è stato allenato con testo italiano estratto da Twitter, contenente anche i caratteri tipici dei Social Media come le emoji, i link, gli hashtag e le mentions.

Per l'implementazione di AlBERTo si è utilizzata la strategia di apprendimento "Masked Learning", ma senza considerare la strategia "Next Following Sentence" (**Figura V**). Questo è un aspetto cruciale dello sviluppo di AlBERTo, poiché ne consegue che questo modello non è adatto al task del "Question Answering", dove quest'ultima caratteristica è essenziale.

Il modello AlBERTo è però perfettamente idoneo per attività di classificazione. [6]

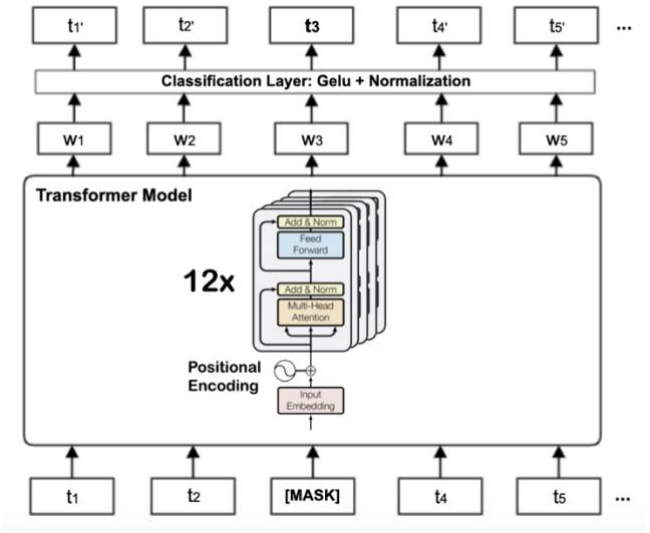


Figura V - Strategia di apprendimento di BERT e AlBERTo [6]

Il dataset utilizzato per allenare BERT con il testo italiano è TWITA, una grande raccolta di tweets in lingua italiana collezionata a partire da febbraio 2012 tramite le API ufficiali di Twitter. Per il fine-tuning di BERT sono stati selezionati 200 milioni di tweets rimuovendo i re-tweets, per un totale di 191GB di dati grezzi.
Il pre-processing è stato effettuato attraverso le librerie Ekphrasis e SentencePiece al fine di normalizzare: URL, email, mentions, orario, date, numeri di telefono, numeri ed emoticons.

Il training è stato eseguito sulla piattaforma Google Collaborative Environment (Colab) e ci sono volute circa 50 ore per il completamento della costruzione di AlBERTo (**Figura VI**). [6]

```
bert_base_config = {
  "attention_probs_dropout_prob": 0.1,
  "directionality": "bidi",
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "max_position_embeddings": 512,
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "type_vocab_size": 2,
  "vocab_size": 128000
}
```

```
# Input data pipeline config
TRAIN_BATCH_SIZE = 128
MAX_PREDICTIONS = 20
MAX_SEQ_LENGTH = 128
MASKED_LM_PROB = 0.15

# Training procedure config
EVAL_BATCH_SIZE = 64
LEARNING_RATE = 2e-5
TRAIN_STEPS = 1000000
SAVE_CHECKPOINTS_STEPS = 2500
NUM_TPU_CORES = 8
```

Figura VI - Configurazione per la fase di learning [6]

Infine, AlBERTo è stato valutato sul task della Sentiment Analysis per il linguaggio italiano. Il dataset utilizzato per la valutazione è stato rilasciato per l'evento SENTIPOLC16, i cui tweets sono differenti da quelli utilizzati per il training di AlBERTo.
Sono stati valutati tre tasks specifici:

- Subjectivity Classification: il sistema deve decidere se un testo è soggettivo o oggettivo;
- Polarity Classification: il sistema deve decidere se un testo è positivo o negativo;
- Irony Detection: il sistema deve decidere se un testo esprime ironia o no.

In totale sono stati utilizzati 7410 tweets per il training e 2000 per il test.
I dati di training e di test sono etichettati con quattro campi: "subj", "pos", "neg", "iro", per descrivere se la frase è soggettiva, positiva, negativa o ironica.
Per ognuna di queste classi, la frase è etichettata con "1" se viene soddisfatta la proprietà, altrimenti l'etichetta è "0".
Per ogni task di classificazione, la valutazione è stata svolta calcolando la *precision*, la *recall* e l'*F1 score* considerando prima "0" come valore positivo e poi "1" [1] (**Figura VII**). [6]

	Prec. 0	**Rec. 0**	**F1. 0**
Subjectivity	0.6838	0.8058	0.7398
Polarity Pos.	0.9262	0.8301	0.8755
Polarity Neg.	0.7537	0.9179	0.8277
Irony	0.9001	0.9853	0.9408
	Prec. 1	**Rec. 1**	**F1 . 1**
Subjectivity	0.8857	0.8015	0.8415
Polarity Pos.	0.5818	0.5314	0.5554
Polarity Neg.	0.7988	0.5208	0.6305
Irony	0.6176	0.1787	0.2772

Figura VII - Valutazione di AlBERTo su tasks di classificazione [6]

[1] La *precision*, la *recall* e l'*F1 score* sono metriche di valutazione calcolate a partire dalle predizioni True Positive, False Positive, False Negative, True Negative. Se si considera "0" come valore positivo, si indicano con TP i valori predetti con "0" e realmente classificati con "0", mentre con TN i valori predetti con "1" e realmente classificati con "1". Viceversa se si considera "1" come valore positivo. Tali metriche di valutazione verranno illustrate più nel dettaglio al paragrafo 2.2.1.

2.1.2 emotionAlBERTo

Il lavoro di questa tesi ha come punto di partenza il progetto di ricerca AlBERTo, di cui abbiamo implementato il fine-tuning con lo scopo di ottenere un modello specializzato nel task dell'Emotion Recognition per il testo italiano.
Il nome che abbiamo deciso di dare a questo progetto è "emotionAlBERTo", per rimarcare il fatto che si appoggia su AlBERTo e quindi a sua volta su BERT, e per mettere in evidenza la specializzazione del modello nel riconoscimento delle emozioni.
Si è deciso di incentrare la tesi sull'Emotion Recognition per ampliare le funzionalità di AlBERTo, in quanto esso è stato valutato solamente per il compito della Sentiment Analysis. Per questo motivo, si è allenato AlBERTo nella classificazione delle seguenti emozioni espresse nel testo: gioia, fiducia, tristezza, rabbia, paura, disgusto.

L'architettura di BERT utilizzata per la creazione di AlBERTo, e quindi anche per il lavoro di tesi, è $BERT_{BASE}$.
Questa versione di BERT consiste di 12 layers, 768 hidden units e 110M di parametri allenabili (**Figura VIII**). [5]

BERT Models	H = 128	H = 256	H = 512	H =768	H = 1024
L = 2	BERT-Tiny	--	--	--	--
L = 4	--	BERT-Mini	BERT-Small	--	--
L = 8	--	--	BERT-Medium	--	--
L = 12	--	--	--	BERT-Base	--
L = 24	--	--	--	--	BERT-Large

Figura VIII - Configurazioni di BERT

2.2 Tecniche di valutazione

I modelli definiti tramite algoritmi di Machine Learning possono portare a classificazioni più o meno corrette. Per valutare la bontà di un classificatore la misura fondamentale è quella di accuratezza (o il suo complemento, l'errore di classificazione), che in maniera semplice può essere definita come la percentuale di volte in cui la classe predetta e quella vera coincidono. A partire da questa idea generale, sono state proposte in letteratura una serie di metodologie e grandezze più dettagliate per una misurazione più realistica dell'accuratezza.

2.2.1 Metriche di valutazione

Per valutare le prestazioni dei classificatori durante gli esperimenti si è estratta ogni volta la matrice di confusione, che permette di visualizzare la classificazione predetta e la classificazione reale. Tutte le matrici di confusione utilizzate per mostrare i risultati di questa tesi contengono i valori reali nelle righe e i valori predetti nelle colonne. In questa particolare matrice, tutte le predizioni corrette si trovano sulla diagonale principale, mentre tutti i valori al di fuori della diagonale rappresentano le predizioni errate (**Figura IX**). [7]

	1	2	...	j	...	k
1	n_{11}	n_{12}	...	n_{1j}	...	n_{1k}
2	n_{21}	n_{22}	...	n_{2j}	...	n_{2k}
...	...	...	...	...	...	...
j	n_{j1}	n_{j2}	...	n_{jj}	...	n_{jk}
...	...	...	...	...	...	...
k	n_{k1}	n_{k2}	...	n_{kj}	...	n_{kk}

Figura IX - Matrice di confusione [7]

I termini della matrice di confusione vengono identificati come:

- **True Positive** (TP): è la somma dei valori nella diagonale principale;
- **True Negative** (TN): per ogni classe è la somma di tutti i valori della matrice di confusione escludendo quelli nella riga e nella colonna corrispondenti alla classe considerata;

- **False Positive** (FP): per ogni classe è la somma dei valori nella colonna della classe considerata, escluso il valore vero positivo;
- **False Negative** (FN): è la somma di tutti i valori nella riga della classe considerata, escluso il valore vero positivo.

Dalla matrice di confusione è possibile ricavare le seguenti metriche:

- *accuracy;*
- *precision;*
- *recall;*
- *F1 score.*

L'*accuracy* è l'accuratezza del modello, rappresenta quindi il rapporto tra i valori predetti correttamente e il totale dei valori predetti. Si definisce con *error rate* il suo complemento, il quale rappresenta la percentuale dei valori predetti in modo non corretto.

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

$$error\ rate = 1 - accuracy$$

La *precision* è definita come il rapporto tra i veri positivi e la somma dei valori predetti come positivi.

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

La *recall* esprime la percentuale dei valori predetti correttamente rapportati al numero totale di valori realmente appartenenti ad una determinata classe.

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

La metrica *F1 score* si ottiene attraverso la media armonica di *precision* e *recall.* [7]

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

2.2.2 Stratified 10-Fold Cross Validation

Per ogni esperimento si è scelto di valutare il modello tramite la Stratified 10-Fold Cross Validation. Segue una breve presentazione dei processi di valutazione possibili per motivare la scelta di questa tecnica.

Per la valutazione di un modello si potrebbe usare la procedura denominata "Holdout", la quale consiste nel suddividere il dataset in due parti, una per il training e una per il testing. È comune utilizzare due terzi dei dati per il training e un terzo per il testing. Il problema di questa metodologia è che il campione di dati utilizzato per il training o per il testing potrebbe non essere rappresentativo. La suddivisione è rappresentativa se sia il training set, che il test set, contengono ogni classe dell'intero dataset nelle giuste proporzioni. Per ovviare al problema del campione rappresentativo si utilizza una procedura denominata "Stratification": la tecnica di valutazione risultante è la "Stratified Holdout".

Una strada più efficace per mitigare ogni bias causato dalla particolare partizione scelta per l'Holdout è di ripetere l'intero processo di suddivisione in training e test set più volte, sempre mantenendo la stratificazione. Si parla quindi di Stratified K-Fold Cross Validation, dove K indica in quante parti uguali viene suddiviso il dataset (**Figura X**). La procedura standard prevede di dividere casualmente i dati in dieci parti, mantenendo le classi rappresentate con le stesse proporzioni che hanno nell'intero dataset. Quindi il modello verrà allenato e testato per dieci volte, la valutazione finale sarà una media dei risultati ottenuti. [8]

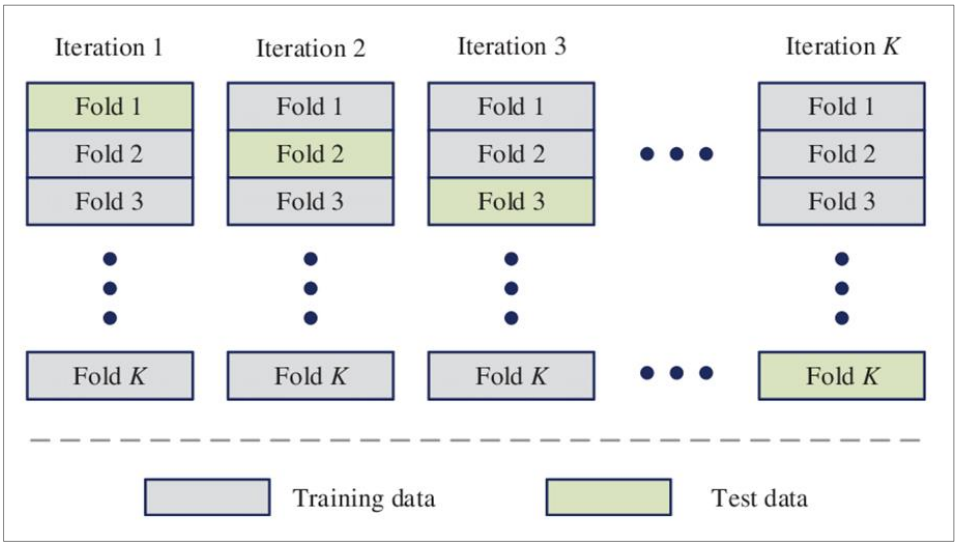


Figura X - K-Fold Cross Validation [11]

2.3 Pre-processing con Oversampling e Undersampling

Durante gli esperimenti sono stati utilizzati dataset sbilanciati, per questo si è ricorso alle tecniche di Oversampling e Undersampling.

Un dataset è sbilanciato se il numero di esempi per ogni classe non ha distribuzione uniforme. In tal caso, il processo di training può essere distorto poiché il modello tenderà a focalizzarsi sulla classe prevalente ignorando gli eventi più rari.

Per risolvere il problema si possono applicare tecniche di campionamento, le quali effettuano un lavoro di pre-processing sui dati, fornendo una distribuzione bilanciata tra le classi.

Le tecniche di campionamento più comuni sono il Random Oversampling e il Random Undersampling. Il primo è un metodo che mira a bilanciare la distribuzione delle classi attraverso la replicazione casuale degli esempi appartenenti alle classi minoritarie. Invece, il Random Undersampling seleziona degli esempi dalla classe maggioritaria, sempre con il fine di rendere la distribuzione delle classi uniforme (**Figura XI**).

Entrambe le metodologie presentano controindicazioni, infatti l'Oversampling aumenta la probabilità di *overfitting*[2], mentre con l'Undersampling si rischia di escludere dal dataset esempi significativi. [7]

Figura XI - Oversampling e Undersampling [12]

[2] Si parla di *overfitting* quando l'algoritmo di apprendimento è in grado di predire correttamente tutti gli esempi utilizzati per il training, ma non è in grado di generalizzare la predizione a nuovi esempi.

3 Datasets

In tutti gli esperimenti sono stati utilizzati due datasets, di cui uno reperito online e uno costruito da noi. Di seguito verrà descritta la struttura e la composizione di questi datasets.

3.1 Dataset MultiEmotions-It

Il dataset MultiEmotions-It è stato reso pubblico online nel 2020 con lo scopo di fornire una nuova risorsa di dati per la lingua italiana. Questo insieme di dati è stato costruito raccogliendo più di 3000 commenti sotto video musicali e video pubblicitari su YouTube e Facebook. Ad ogni commento è stata associata manualmente una o più emozioni tra: gioia, fiducia, tristezza, rabbia, paura, disgusto, trepidazione e sorpresa. [9]
Per questo progetto di tesi si è deciso di non considerare le emozioni trepidazione e sorpresa, e di eliminare tutti i commenti che esprimevano più emozioni.
Durante i test svolti abbiamo considerato sia sei che quattro emozioni.
Per gli esperimenti con sei emozioni si sono quindi considerate: gioia, fiducia, tristezza, rabbia, paura e disgusto; il dataset risultante è distribuito come segue (**Figura XII**).

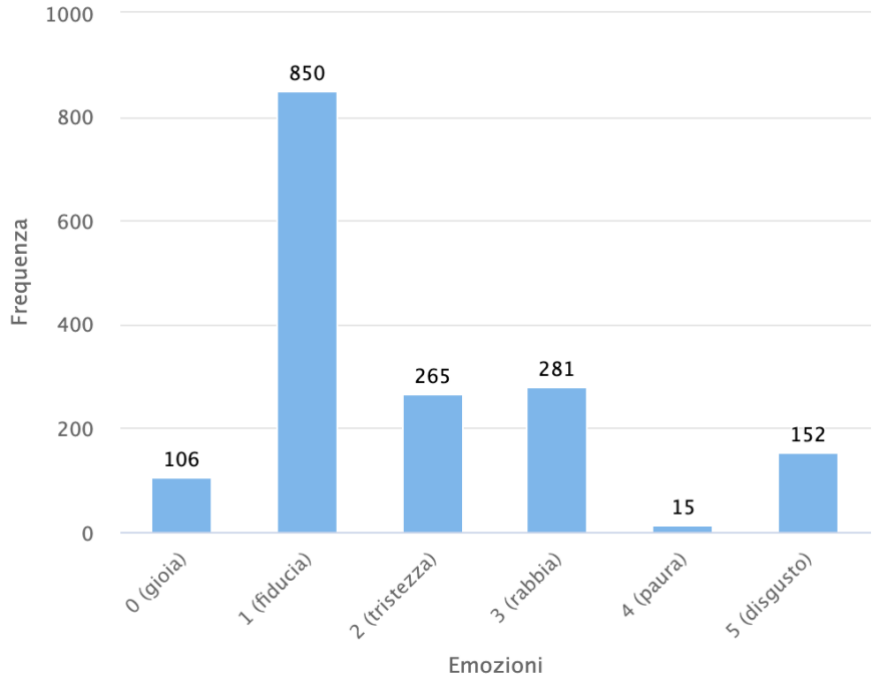


Figura XII - Distribuzione dataset MultiEmotions-It con sei emozioni

Per gli esperimenti con quattro emozioni si sono considerate solamente: gioia, tristezza, rabbia e paura. Si noti che la distribuzione dei dati differisce da quella precedente poiché i commenti che esprimono più emozioni, tra cui fiducia e disgusto (es. rabbia-disgusto, gioia-fiducia), ora non vengono eliminati in quanto esprimono una sola emozione tra quelle considerate (**Figura XIII**).

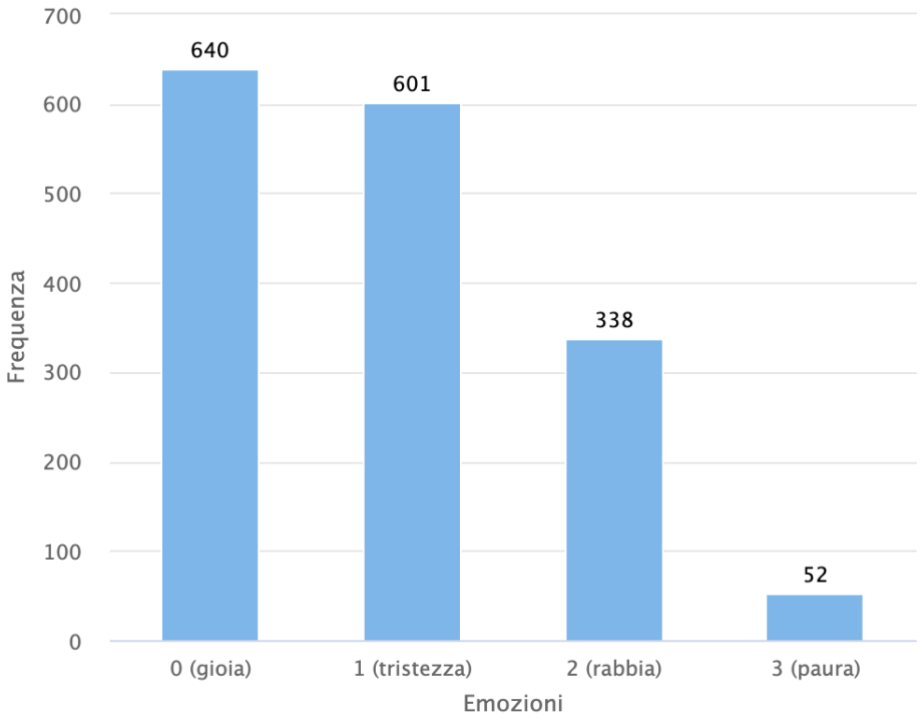


Figura XIII - Distribuzione dataset MultiEmotions-It con quattro emozioni

3.2 Twitter Dataset

Per estendere la sperimentazione serviva un altro dataset con testi italiani estratti dai Social Media annotati con le sei emozioni sopra citate, ma in rete non sono reperibili altri datasets italiani per l'Emotion Recognition. Si è quindi deciso di procedere con l'estrazione di un dataset da Twitter e di annotarlo in modo automatico attraverso le emoji. È stato utilizzato il software RapidMiner per l'estrazione dei dati tramite le API di Twitter. Sono stati estratti 2000 tweets per ogni emozione utilizzando come query di ricerca l'emoji associata all'emozione considerata. La corrispondenza scelta tra emoji ed emozioni è la seguente:

GIOIA 💘😍
FIDUCIA 👏
TRISTEZZA 😔😢
RABBIA 😠😡
PAURA 😰
DISGUSTO 🤮🤢

Durante il processo di estrazione dei dati è stato effettuato il pre-processing con lo scopo di ottenere un dataset più pulito.
Attraverso l'uso delle seguenti espressioni regolari, queste parti sono state selezionate ed eliminate:

Mentions
```
\@[a-zA-Z0-9_:ìàòéè/\.!"#$%&'()*+,./:;<=>?@\[\\\]_`{|}~-]*
```

Link
```
\bhttp[a-zA-Z0-9_:ìàòéè/\.!"#$%&'()*+,./:;<=>?@\[\\\]_`{|}~
-]*
```

Hashtag
```
\#[a-zA-Z0-9_:ìàòéè/\.!"#$%&'()*+,./:;<=>?@\[\\\]_`{|}~-]*
```

RT
```
\b(RT )
```

Spazi o tab all'inizio di un tweet
```
^[ \t]*
```

Spazi o tab alla fine di un tweet
```
[ \t]*$
```

Infine, attraverso il blocco apposito "Remove Duplicates", sono stati rimossi tutti i tweets duplicati. Con il blocco "Select Attributes" sono state selezionate le colonne "id" e "testo". Attraverso "Generate Attribute" è stata generata la colonna "emozione" per procedere con l'annotazione dei testi: tutto il processo viene ripetuto per ogni emoji e quindi a seconda della query tutti i testi estratti verranno annotati con "0", "1", "2", "3", "4" o "5" (**Figura XIV**).

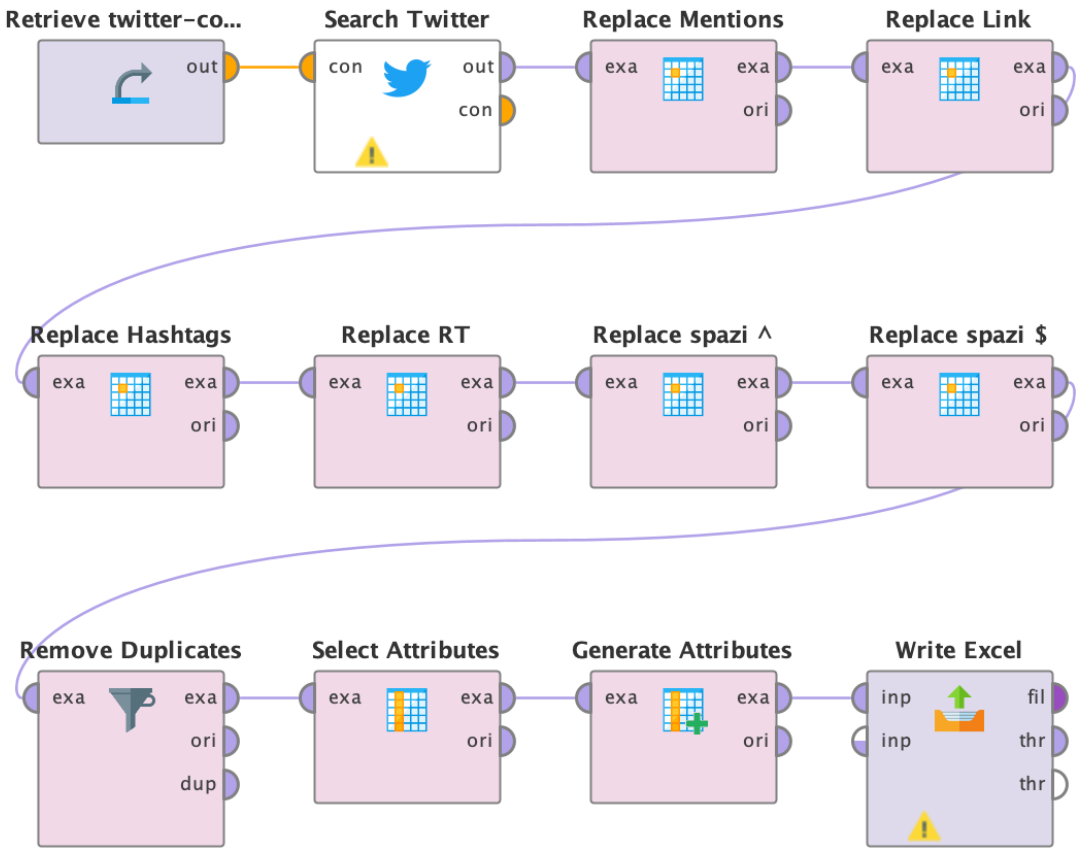


Figura XIV - Processo RapidMiner

Tutti i dataset estratti sono stati poi uniti: il dataset finale risulta distribuito come segue per le emozioni gioia, fiducia, tristezza, rabbia, paura e disgusto (**Figura XV**). Si noti che per ogni emozione si hanno meno di 2000 tweets a causa della rimozione dei duplicati.

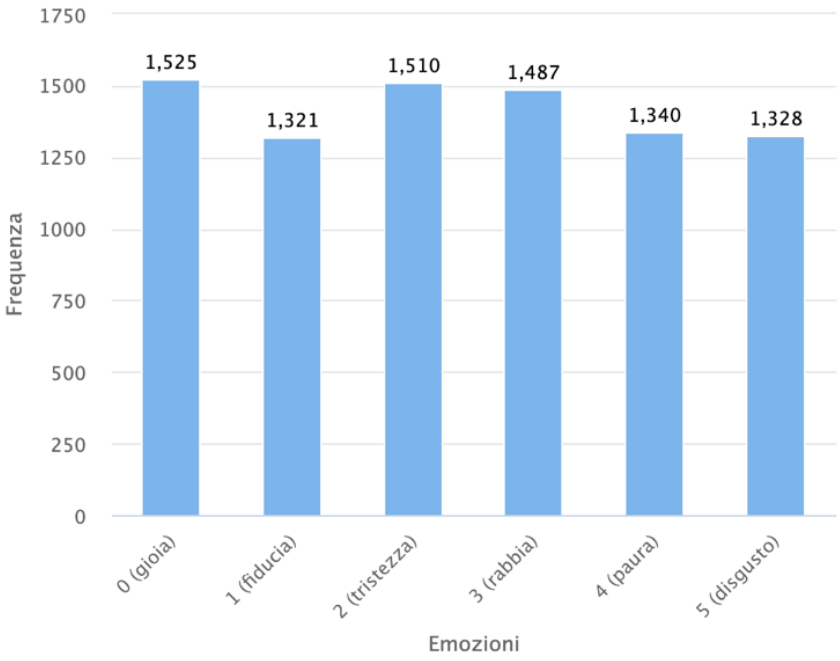


Figura XV - Distribuzione Twitter Dataset con sei emozioni

Analizzando manualmente tutti i tweets estratti è stata effettuata una pulizia manuale per ottenere un dataset di qualità maggiore e più valido ai fini del riconoscimento di emozioni. Sono stati quindi eliminati i tweets non in italiano e quelli non corrispondenti all'emozione associata all'emoji: il dataset risultante è distribuito come segue (**Figura XVI**).

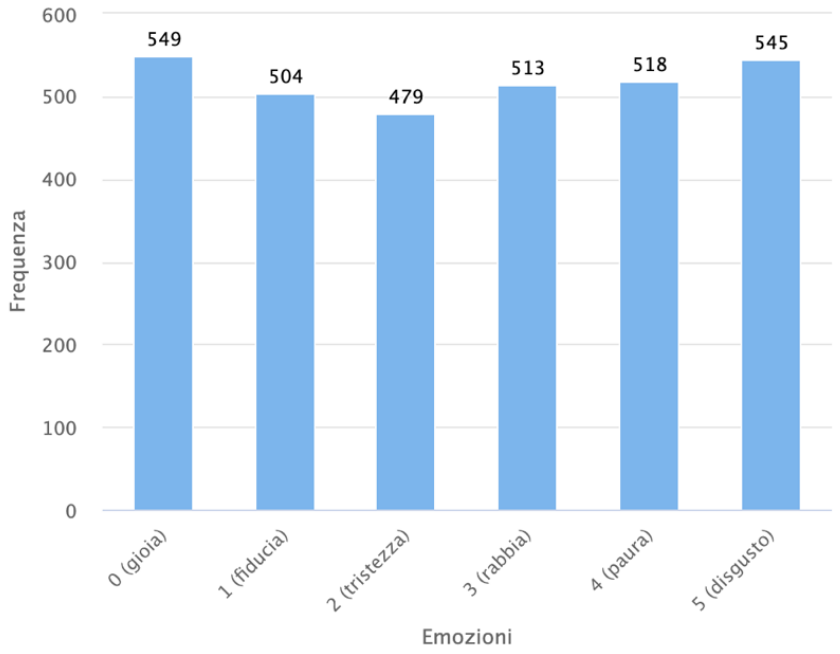


Figura XVI - Distribuzione Twitter Dataset pulito manualmente con sei emozioni

Considerando solo quattro emozioni, si ottengono datasets (**Figura XVII** e **Figura XVIII**) distribuiti come quelli sopra illustrati, ma con le sole etichette "gioia", "tristezza", "rabbia" e "paura".

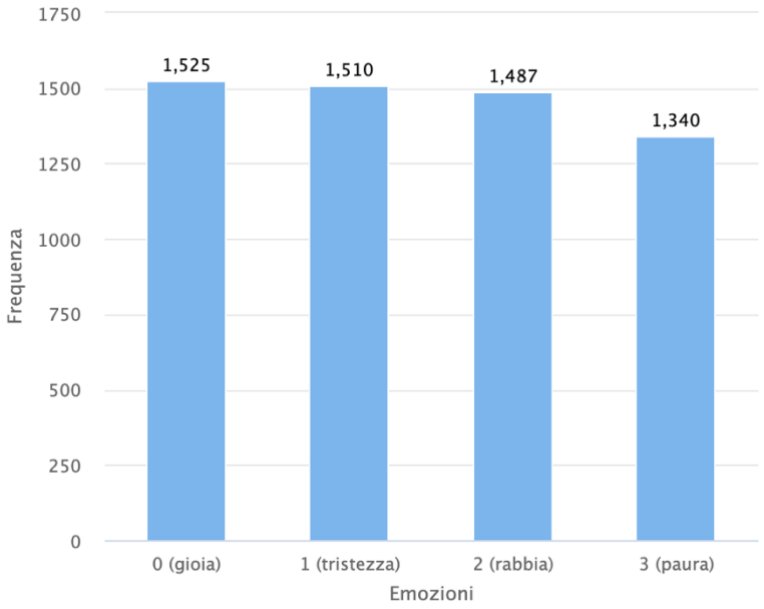


Figura XVII - Distribuzione Twitter Dataset con quattro emozioni

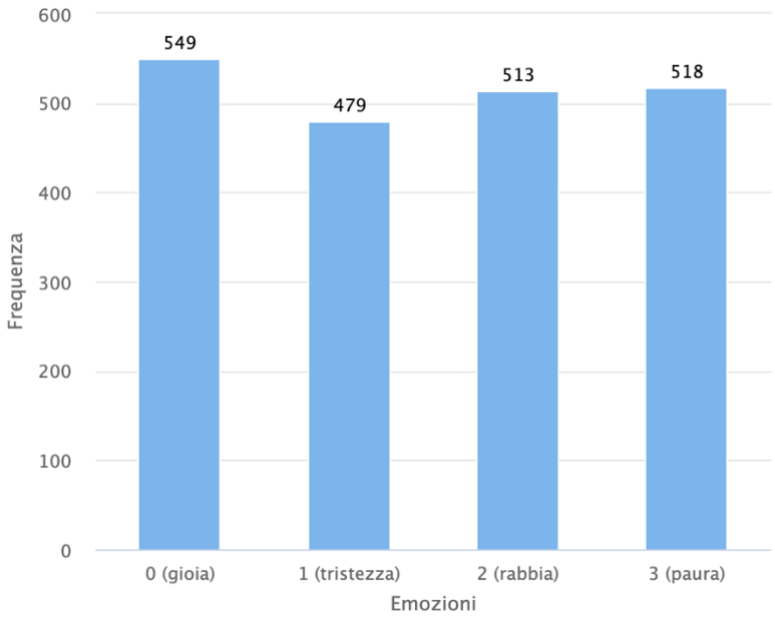


Figura XVIII - Distribuzione Twitter Dataset pulito manualmente con quattro emozioni

4 ESPERIMENTI CON SEI EMOZIONI

4.1 Costruzione modello con dataset MultiEmotions-It

Di seguito verranno spiegati gli esperimenti effettuati per ottenere il miglior modello classificatore di sei emozioni costruito con il dataset MultiEmotions-It. Saranno inoltre illustrati i risultati ottenuti testando il miglior modello ottenuto con il dataset estratto da Twitter.

4.1.1 Valutazione con pre-processing classico

Il primo esperimento effettuato prende in considerazione il dataset MultiEmotions-It con pre-processing "classico", il che significa che i dati sono stati elaborati come spiegato in precedenza, eliminando tutti i testi esprimenti due o più emozioni e quelli non classificati con nessuna delle emozioni considerate (**Figura XIX** e **Figura XX**).

```
# ELIMINAZIONE DELLE RIGHE MULTIEMOTIONS
text_multiemotions = 0
for row in italian_dataset.iterrows():
  ones = 0
  for emotion in row[1].iloc[[1,2,3,4,5,6]]:
    if emotion == 1:
      ones = ones + 1
  if ones >= 2: # multiemotions -> elimino riga
    print(row[1])
    text_multiemotions = text_multiemotions + 1
    italian_dataset.drop(row[0], inplace=True)
```

Figura XIX - Codice utilizzato per eliminare i testi multiemotions

```
# ELIMINAZIONE DEI TESTI NON CLASSIFICATI
text_not_classified = 0
for row in italian_dataset.iterrows():
  ones = 0
  for emotion in row[1].iloc[[1,2,3,4,5,6]]:
    if emotion == 1:
      ones = ones + 1
  if ones == 0: # not classified -> elimino riga
    print(row[1])
    text_not_classified = text_not_classified + 1
    italian_dataset.drop(row[0], inplace=True)
```

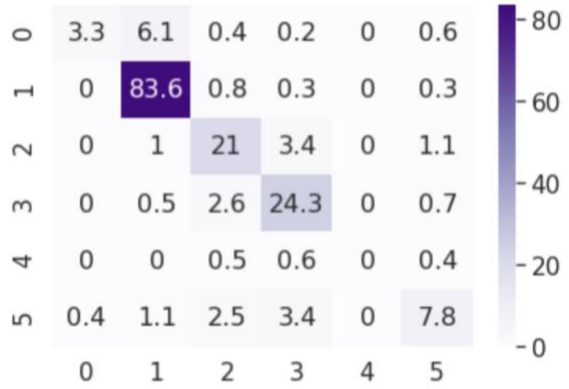
Figura XX - Codice per eliminare i testi non classificati

Il dataset MultiEmotions-It ottenuto dopo il pre-processing contiene 1669 testi classificati (**Figura XXI**).

```
[1669 rows x 7 columns]
I testi che esprimono gioia sono: 106
I testi che esprimono fiducia sono: 850
I testi che esprimono tristezza sono: 265
I testi che esprimono rabbia sono: 281
I testi che esprimono paura sono: 15
I testi che esprimono disgusto sono: 152
```

Figura XXI - Numero di testi per emozione nel dataset MultiEmotions-It

Il dataset così composto è stato utilizzato per il fine-tuning di AlBERTo per il riconoscimento di emozioni, quindi il modello risultante è stato valutato attraverso la Stratified 10-Fold Cross Validation (**Figura XXII**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.891892	0.905742	0.755396	0.754658	0.0	0.715596
Recall	0.311321	0.983529	0.792453	0.864769	0.0	0.513158

```
Accuratezza: 0.8388175456316282
F1 score: 0.5932166857578924
```

Figura XXII - Valutazione tramite Stratified 10-Fold Cross Validation

È facile osservare che le classi meno riconosciute sono quelle con meno esempi nel dataset, ovvero: disgusto, gioia e paura. In particolare, la paura non viene mai predetta dal modello. I successivi esperimenti sono stati svolti per provare a risolvere il problema dei dati estremamente sbilanciati e con troppe poche occorrenze per alcune emozioni.

4.1.2 Valutazione con pre-processing alternativo

Prima di sperimentare tecniche classiche di campionamento come l'Oversampling e l'Undersampling, ho voluto effettuare un pre-processing alternativo del dataset MultiEmotions-It. Vista l'importanza di avere più testi possibili per ogni emozione, ho pensato si potesse agire sui testi multiemotions, con lo scopo di eliminare meno testi dal dataset. Ho quindi proceduto così nel pre-processing del dataset:

1. eliminazione dei testi non classificati;
2. eliminazione dei testi classificati con tre o più emozioni (**Figura XXIII**);
3. classificazione dei testi esprimenti due emozioni con l'emozione meno presente nel dataset. Questo è stato fatto per tutte le coppie di emozioni tranne per quelle discordanti (gioia-tristezza, gioia-rabbia, gioia-paura) (**Figura XXIV** e **Figura XXV**);
4. eliminazione dei testi rimasti esprimenti due emozioni, che corrispondono a quelli classificati con due emozioni discordanti;
5. creazione del dataset "alternativo" (**Figura XXVI**).

```
# ELIMINO TESTI CLASSIFICATI CON 3 EMOZIONI O PIU'
text_multiemotions = 0
for row in italian_dataset.iterrows():
  ones = 0
  for emotion in row[1].iloc[[1,2,3,4,5,6]]:
    if emotion == 1:
      ones = ones + 1
  if ones >= 3: # multiemotions almeno 3 emozioni -> elimino riga
    print(row[1])
    text_multiemotions = text_multiemotions + 1
    italian_dataset.drop(row[0], inplace=True)
```

Figura XXIII - Codice per eliminare testi esprimenti tre o più emozioni

```
[2671 rows x 7 columns]
I testi che esprimono gioia sono: 647
I testi che esprimono fiducia sono: 1757
I testi che esprimono tristezza sono: 623
I testi che esprimono rabbia sono: 347
I testi che esprimono paura sono: 66
I testi che esprimono disgusto sono: 233

gioia_fiducia = 532
gioia_tristezza = 2
gioia_rabbia = 4
gioia_paura = 1
gioia_disgusto = 2

fiducia_tristezza = 332
fiducia_rabbia = 24
fiducia_paura = 6
fiducia_disgusto = 13

tristezza_rabbia = 7
tristezza_paura = 13
tristezza_disgusto = 4

rabbia_paura = 0
rabbia_disgusto = 31

paura_disgusto = 31
```

Figura XXIV - Numero testi totali e multiemotions PRIMA della classificazione dei testi multiemotions con la classe minoritaria

```
[2671 rows x 7 columns]
I testi che esprimono gioia sono: 647
I testi che esprimono fiducia sono: 850
I testi che esprimono tristezza sono: 599
I testi che esprimono rabbia sono: 316
I testi che esprimono paura sono: 66
I testi che esprimono disgusto sono: 202

gioia_fiducia = 0
gioia_tristezza = 2
gioia_rabbia = 4
gioia_paura = 1
gioia_disgusto = 2

fiducia_tristezza = 0
fiducia_rabbia = 0
fiducia_paura = 0
fiducia_disgusto = 0

tristezza_rabbia = 0
tristezza_paura = 0
tristezza_disgusto = 0

rabbia_paura = 0
rabbia_disgusto = 0

paura_disgusto = 0
```

Figura XXV - Numero testi totali e multiemotions DOPO il pre-processing alternativo

```python
# CREAZIONE NUOVO DATASET DI DUE COLONNE CON ETICHETTA EMOZIONE (testo, emozione)
rows_list = []

for row in italian_dataset.iterrows():
  i = 1
  dictionary = {
    "testo": "",
    "emozione": "",
  }
  for emotion in row[1].iloc[[1,2,3,4,5,6]]:
    if emotion == 1 and i == 1:
      label = "0"                    # gioia 0
    elif emotion == 1 and i == 2:
      label = "1"                    # fiducia 1
    elif emotion == 1 and i == 3:
      label = "2"                    # tristezza 2
    elif emotion == 1 and i == 4:
      label = "3"                    # rabbia 3
    elif emotion == 1 and i == 5:
      label = "4"                    # paura 4
    elif emotion == 1 and i == 6:
      label = "5"                    # disgusto 5
    i = i + 1
  dictionary["testo"] = row[1].iloc[0]
  dictionary["emozione"] = label
  rows_list.append(dictionary)

print(rows_list)
new_dataset = pd.DataFrame(rows_list)
print(new_dataset)


# salvo il nuovo dataset in google storage
new_dataset.to_csv('gs://bucket-alberto/dataset_alternativo.csv', sep=';', encoding='UTF-8')
```

Figura XXVI - Codice per la creazione del dataset "alternativo"

Ho ottenuto quindi un dataset finale, per le sei emozioni gioia, fiducia, tristezza, rabbia, paura e disgusto, distribuito come segue (**Figura XXVII**).

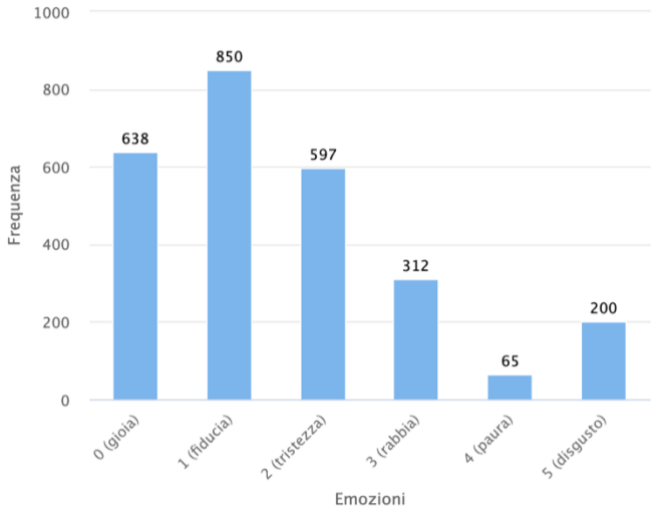
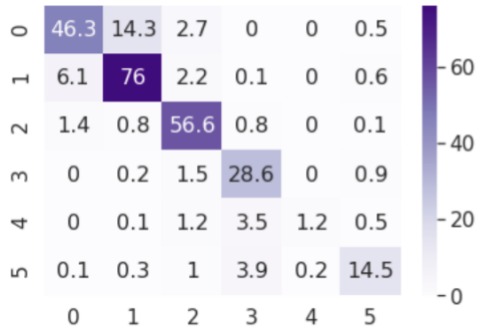


Figura XXVII - Distribuzione dataset "alternativo"

Si noti che rispetto al pre-processing "classico", si sono ottenuti 532 testi in più di gioia, 332 testi in più di tristezza, 31 testi in più di rabbia, 50 testi in più di paura e 48 testi in più di disgusto.

Utilizzando questo dataset è stato quindi effettuato il fine-tuning di AlBERTo e il modello risultante è stato valutato tramite la Stratified 10-Fold Cross Validation (**Figura XXVIII**). Questa metodologia ha portato a dei risultati migliori, ma trattandosi di una tecnica sperimentale non validata, si è deciso di continuare gli esperimenti con tecniche note e più valide per il pre-processing del dataset.



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.858998	0.828790	0.868098	0.775068	0.857143	0.847953
Recall	0.725705	0.894118	0.948074	0.916667	0.184615	0.725000

```
Accuratezza: 0.8385739629973811
F1 score: 0.7423745817314076
```

Figura XXVIII - Valutazione tramite Stratified 10-Fold Cross Validation con dataset "alternativo"

4.1.3 Valutazione con Oversampling e Undersampling

Con l'utilizzo delle tecniche di campionamento di Oversampling e Undersampling sono stati effettuati quattro esperimenti diversi. Per ognuno si è gestito il dataset MultiEmotions-It differentemente e poi si è fatto il fine-tuning di AlBERTo con il dataset ottenuto. Ogni prova è stata valutata attraverso la Stratified 10-Fold Cross Validation: per semplificare la lettura di questo elaborato le prove verranno presentate in ordine di bontà dei risultati. Si noti che l'Oversampling e l'Undersampling sono stati applicati ad ogni split della 10-Fold Cross Validation solamente alla porzione di dati utilizzata come training set.

4.1.3.1 Oversampling su tutte le classi minoritarie

La tecnica di Random Oversampling è stata applicata a tutte le classi tranne la maggioritaria, per cui tutti i testi tranne quelli esprimenti fiducia sono stati duplicati in modo randomico al fine di ottenere un dataset bilanciato (**Figura XXIX**).

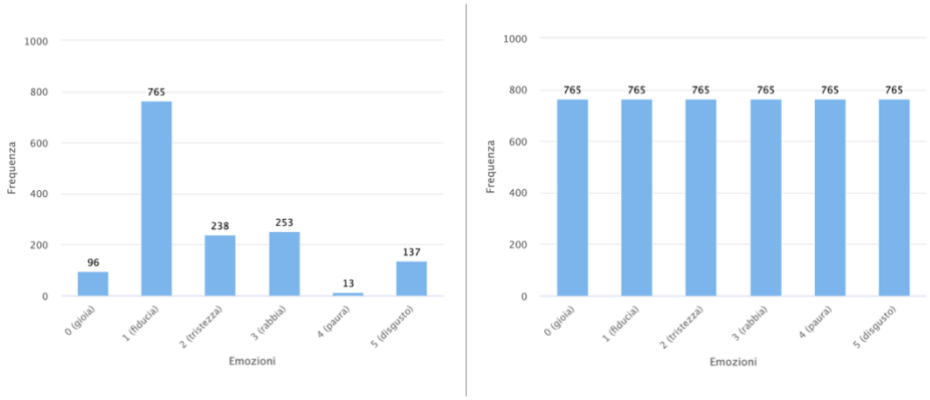
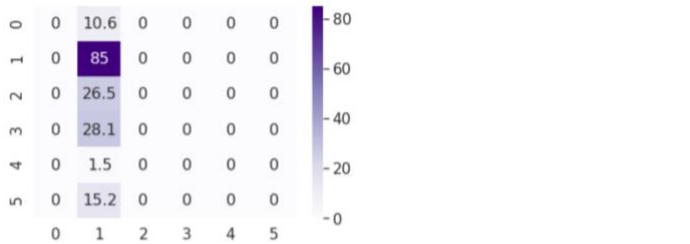


Figura XXIX - Dataset MultiEmotions-It PRIMA e DOPO l'Oversampling su tutte le classi tranne la maggioritaria

I risultati ottenuti attraverso la Stratified 10-Fold Cross Validation mostrano un mal funzionamento del modello allenato con questo dataset: il test set viene interamente classificato con la classe maggioritaria del dataset MultiEmotions-It originale (**Figura XXX**). Una causa di questo fenomeno potrebbe essere l'overfitting.



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.0	0.509287	0.0	0.0	0.0	0.0
Recall	0.0	1.000000	0.0	0.0	0.0	0.0

Accuratezza: 0.5092886516124377
F1 score: 0.11247865680136596

Figura XXX - Valutazione tramite Stratified 10-Fold Cross Validation con Oversampling su tutte le classi minoritarie

4.1.3.2 Bilanciamento dataset con Oversampling e Undersampling

Visti i risultati precedenti, si è provato a mitigare l'effetto dell'Oversampling fissando un threshold a 300: valore intermedio tra il numero di occorrenze della classe maggioritaria e quelle della classe minoritaria. Si è applicato quindi l'Oversampling a tutte le classi con meno di 300 occorrenze e l'Undersampling alla classe "fiducia". Il dataset finale è quindi bilanciato, con un totale di 300 occorrenze per ogni emozione (**Figura XXXI**).

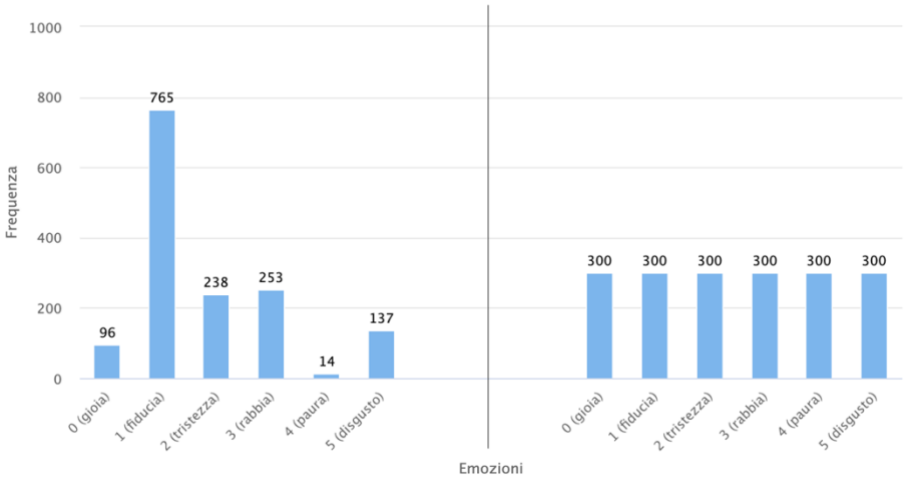
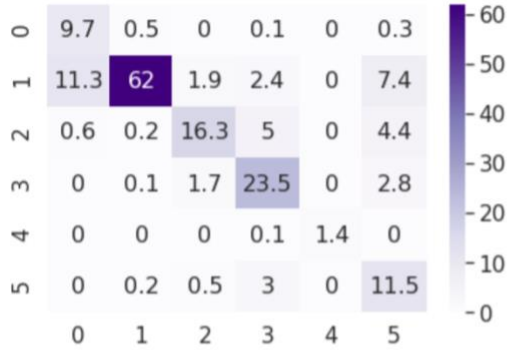


Figura XXXI - Dataset MultiEmotions-It PRIMA e DOPO l'Oversampling e l'Undersampling con threshold fissato a 300

I risultati ottenuti con la Stratified 10-Fold Cross Validation sono mostrati di seguito (**Figura XXXII**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.449074	0.984127	0.799020	0.689150	1.000000	0.435606
Recall	0.915094	0.729412	0.615094	0.836299	0.933333	0.756579

```
Accuratezza: 0.7453574778154535
F1 score: 0.7241323475150108
```

Figura XXXII - Valutazione tramite Stratified 10-Fold Cross Validation con Oversampling e Undersampling

4.1.3.3 Oversampling sulla classe minoritaria (paura)

In questa prova è stato effettuato il Random Oversampling unicamente sulla classe minoritaria: la paura. Essendo però il dataset MultiEmotions-It molto sbilanciato, si è scelto di non portare le occorrenze della paura al numero di occorrenze della classe maggioritaria, ma si è fissato "100" come threshold (**Figura XXXIII**).

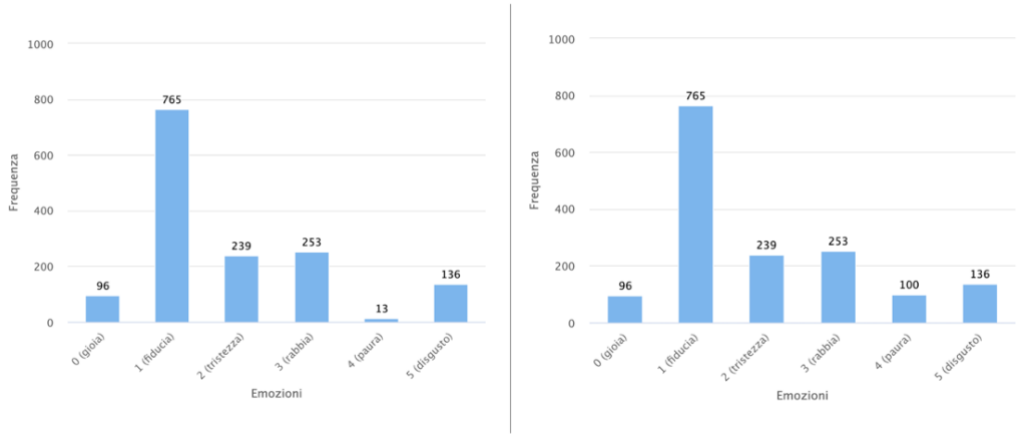
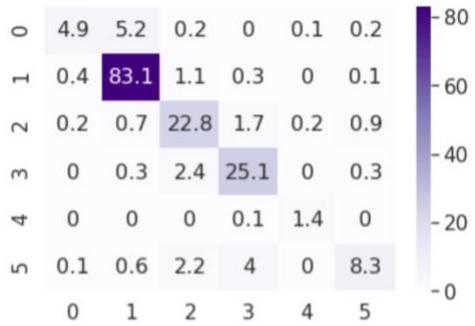


Figura XXXIII - Dataset MultiEmotions-It PRIMA e DOPO l'Oversampling solo sulla classe minoritaria

È evidente che i risultati sono migliori: la paura viene riconosciuta dal classificatore e quasi mai gli altri testi vengono predetti erroneamente come "paura". Rimangono però dei valori bassi di *recall* per la gioia e il disgusto (**Figura XXXIV**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.875000	0.924360	0.794425	0.804487	0.823529	0.846939
Recall	0.462264	0.977647	0.860377	0.893238	0.933333	0.546053

```
Accuratezza: 0.8723649087367434
F1 score: 0.7892888983039306
```

Figura XXXIV - Valutazione tramite Stratified 10-Fold Cross Validation con Oversampling sulla classe minoritaria

4.1.3.4 Oversampling sulle tre classi minoritarie (paura, gioia, disgusto)

Come appena evidenziato, il valore di *F1 score* è minore dell'*accuracy* poiché il dataset MultiEmotions-It è molto sbilanciato, in particolare si hanno valori bassi di *recall* per le tre classi meno presenti nel dataset: paura, gioia e disgusto. Ho provato, dunque, ad applicare il Random Oversampling a queste tre classi, essendo quelle meno riconosciute dal modello. Quando si splitta il dataset in training set e test set, si nota che le tre classi più presenti nel training set sono la fiducia con 765 occorrenze, la rabbia con 252 e la tristezza con 239. Ho quindi deciso di non effettuare l'Oversampling portando le occorrenze di paura, gioia e disgusto al numero di occorrenza della classe maggioritaria, ma ho fissato un threshold a 200 (**Figura XXXV**).

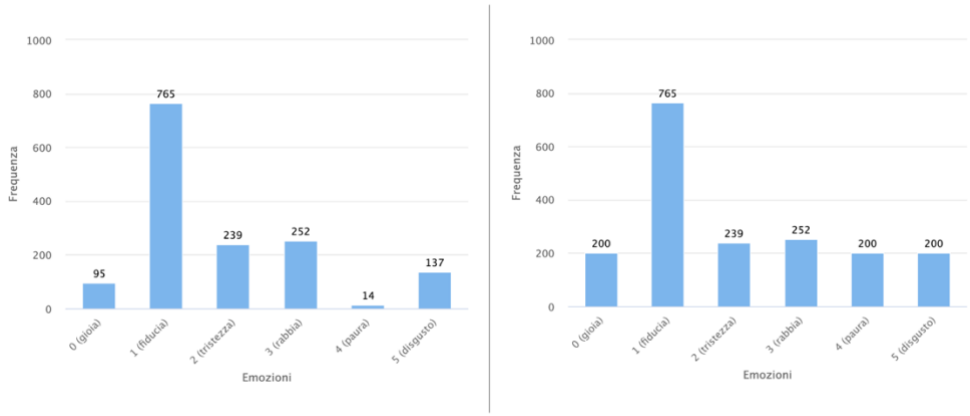
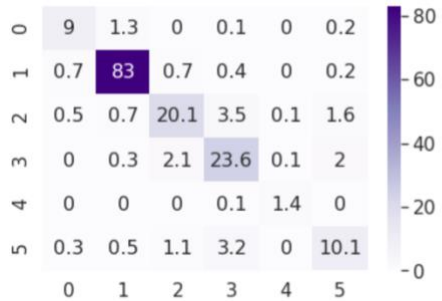


Figura XXXV - Dataset MultiEmotions-It PRIMA e DOPO l'Oversampling sulle tre classi minoritarie (paura, gioia, disgusto)

Con questa metodologia si sono ottenuti i risultati di valutazione migliori (**Figura XXXVI**), per questo si è deciso di proseguire con la costruzione del modello attraverso tale tecnica.



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.857143	0.967366	0.837500	0.763754	0.875000	0.716312
Recall	0.849057	0.976471	0.758491	0.839858	0.933333	0.664474

```
Accuratezza: 0.8819637832768198
F1 score: 0.8260620149249046
```

Figura XXXVI - Valutazione tramite Stratified 10-Fold Cross Validation con Oversampling sulle tre classi minoritarie (paura, gioia, disgusto)

4.1.4 Model Building

Una volta effettuati gli esperimenti utilizzando il dataset MultiEmotions-It e valutata la tecnica di campionamento migliore, ho proceduto con la costruzione del modello. Ho quindi scelto come training set l'intero dataset MultiEmotions-It (non più partizionato in training e test set, poiché in questa fase non viene utilizzata la 10-Fold Cross Validation), e applicato il Random Oversampling alle tre classi minoritarie "paura", "gioia" e "disgusto" con threshold fissato a 200 (**Figura XXXVII**).

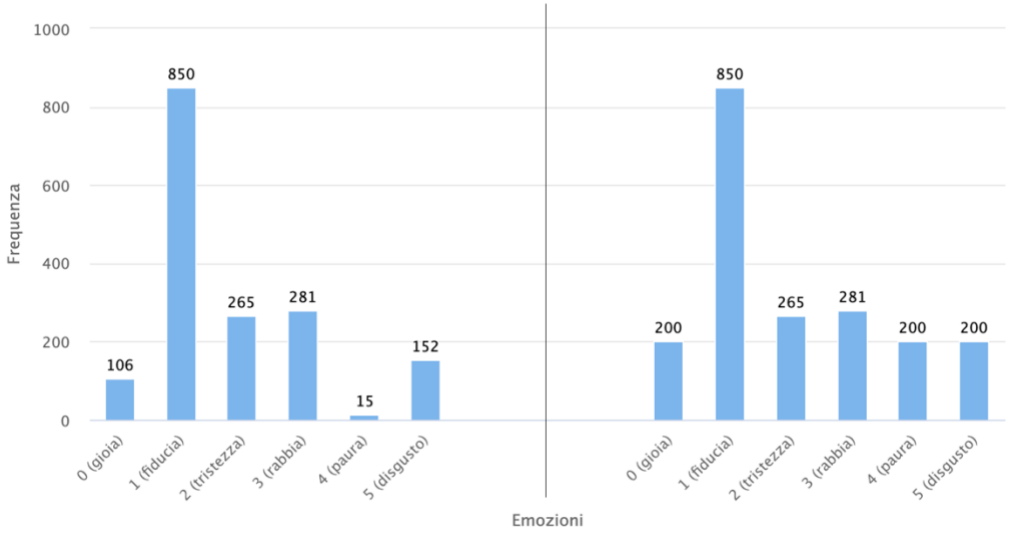


Figura XXXVII - Dataset di partenza e dataset ottenuto con l'Oversampling (training set)

Nella seguente figura (**Figura XXXVIII**) sono mostrati i parametri utilizzati per il fine-tuning di AlBERTo, effettuato utilizzando come training set il dataset appena illustrato.

```
#SET THE PARAMETERS
TRAIN_BATCH_SIZE = 512
PREDICT_BATCH_SIZE = 512
EVAL_BATCH_SIZE = 512
LEARNING_RATE = 2e-5
NUM_TRAIN_EPOCHS = 10.0
MAX_SEQ_LENGTH = 128
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 1000
SAVE_SUMMARY_STEPS = 500
```

Figura XXXVIII - Parametri utilizzati per il training

4.1.5 Test con Twitter Dataset

In questa fase si è voluto testare il modello allenato con il dataset MultiEmotions-It con un dataset differente, ovvero il dataset estratto da noi attraverso Twitter. I dataset considerati per il training e il test set sono entrambi composti da testi estratti dai Social Media, ma estrapolati in piattaforme e contesti diversi. La metodologia con cui è stato costruito il dataset estratto da Twitter è esposta nel paragrafo 3.2, ma per chiarezza riporto anche in questa sezione la distribuzione del dataset ottenuto che fungerà da test set. Il primo test è stato fatto con il Twitter Dataset completo, cioè quello contenente tutti i tweets estratti (**Figura XXXIX**).

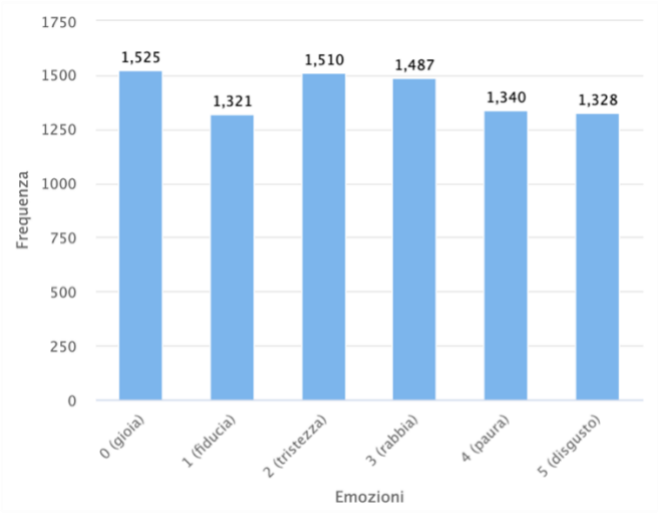
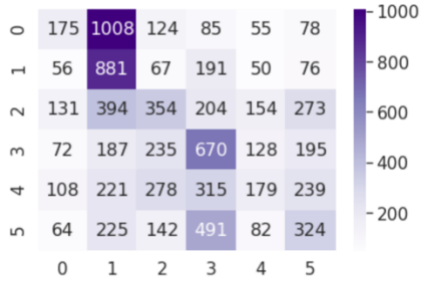


Figura XXXIX - Twitter Dataset completo (test set)

Di seguito sono riportati i risultati ottenuti (**Figura XL**). Dalle metriche di valutazione si nota che il modello non riesce a predire correttamente le emozioni espresse dai tweets.



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.288779	0.302126	0.295000	0.342536	0.276235	0.273418
Recall	0.114754	0.666919	0.234437	0.450572	0.133582	0.243976

```
Accuratezza: 0.30348960169192807
F1 score = 0.27808201922691916
```

Figura XL - Metriche di valutazione per il test del modello con il dataset estratto da Twitter

Per provare ad ottenere risultati migliori, è stato testato il modello con il dataset estratto da Twitter pulito manualmente. Il test set considerato è quindi composto come segue (**Figura XLI**).

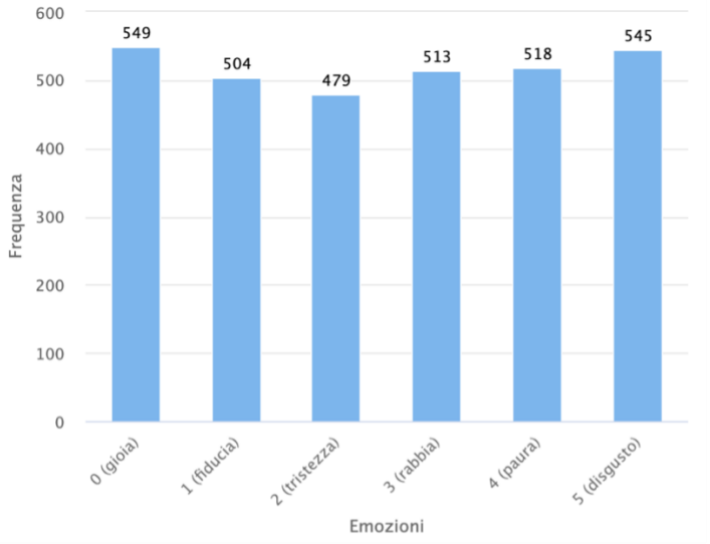
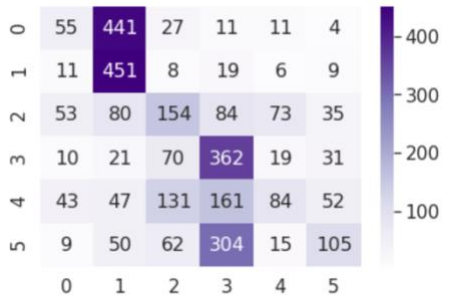


Figura XLI - Twitter Dataset pulito manualmente (test set)

Dalla valutazione mostrata di seguito si deduce che la predizione è leggermente migliore rispetto a quella precedente, probabilmente poiché questo test set contiene testi più significativi al fine del riconoscimento delle emozioni nel testo italiano. Dalla figura è evidente che la misclassificazione più frequente è tra le coppie di emozioni gioia/fiducia, rabbia/disgusto e tristezza/paura (**Figura XLII**). Per ridurre tale errore si è deciso di effettuare gli esperimenti senza considerare le emozioni "disgusto" e "fiducia"; questi verranno illustrati nel capitolo 5.



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.303867	0.413761	0.340708	0.384697	0.403846	0.444915
Recall	0.100182	0.894841	0.321503	0.705653	0.162162	0.192661

```
Accuratezza: 0.38963963963963966
F1 score = 0.3409352912892123
```

Figura XLII - Metriche di valutazione per il test del modello con il dataset estratto da Twitter pulito manualmente

4.2 Costruzione modello con Twitter Dataset

Tutti gli esperimenti appena mostrati sono stati replicati con lo stesso processo, ma invertendo il training e il test set. Quindi di seguito vengono presentati la costruzione del modello attraverso il dataset estratto da Twitter e il test di quest'ultimo con il dataset MultiEmotions-It.

4.2.1 Valutazione senza Oversampling

Durante questo primo esperimento si è provato a valutare il funzionamento del modello con la 10-Fold Cross Validation utilizzando il dataset estratto da Twitter pulito manualmente (**Figura XLIII**).

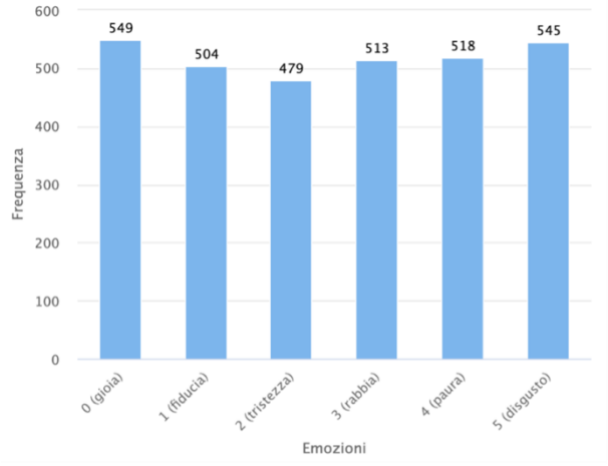
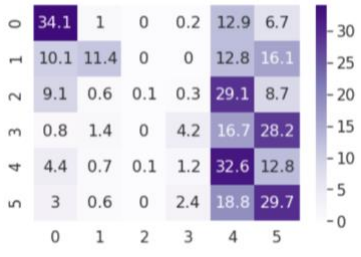


Figura XLIII - Twitter Dataset pulito manualmente

Dalla figura seguente si evince che il modello allenato con il 90% di questo dataset non riesce a predire in modo corretto il restante 10% del dataset utilizzato come test set durante la Stratified 10-Fold Cross Validation (**Figura XLIV**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.554472	0.726115	0.500000	0.506024	0.265256	0.290607
Recall	0.621129	0.226190	0.002088	0.081871	0.629344	0.544954

Accuratezza: 0.36069494865677837
F1 score: 0.30374707240801746

Figura XLIV - Valutazione tramite Stratified 10-Fold Cross Validation con Twitter Dataset pulito manualmente

4.2.2 Valutazione con Oversampling

Per provare a migliorare le prestazioni del modello allenato con il dataset estratto da Twitter si è utilizzata la tecnica di campionamento "Oversampling", ma anziché duplicare in modo randomico i testi di tutte le emozioni minoritarie fino a raggiungere il numero di occorrenze della classe maggioritaria, si è fissato il threshold a 1000. Il modello è stato quindi allenato con più dati, anche se duplicati, con lo scopo di far analizzare più volte la stessa frase all'algoritmo per rafforzare la capacità di classificare determinate parole chiave.
Come per gli esperimenti precedenti, l'Oversampling è stato applicato ad ogni split della Stratified 10-Fold Cross Validation solamente sul 90% del dataset che funge da training set (**Figura XLV**).

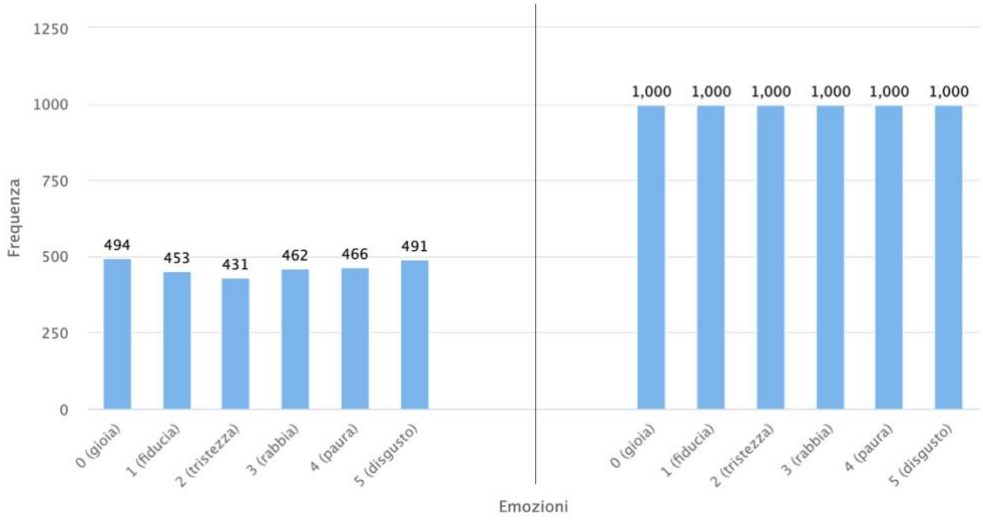
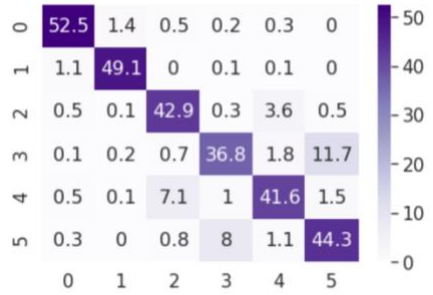


Figura XLV - Twitter Dataset PRIMA e DOPO l'Oversampling su tutte le emozioni

Così facendo si sono ottenute predizioni molto più accurate (**Figura XLVI**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.954545	0.964637	0.825000	0.793103	0.857732	0.763793
Recall	0.956284	0.974206	0.895616	0.717349	0.803089	0.812844

```
Accuratezza: 0.8597438025101131
F1 score: 0.8588373370440971
```

Figura XLVI - Valutazione tramite Stratified 10-Fold Cross Validation con Twitter Dataset pulito manualmente e Oversampling

4.2.3 Model Building

Per costruire il modello si è utilizzato il dataset estratto da Twitter pulito manualmente con la tecnica di Oversampling illustrata nel paragrafo precedente. Il training set risulta quindi essere distribuito come segue (**Figura XLVII**).

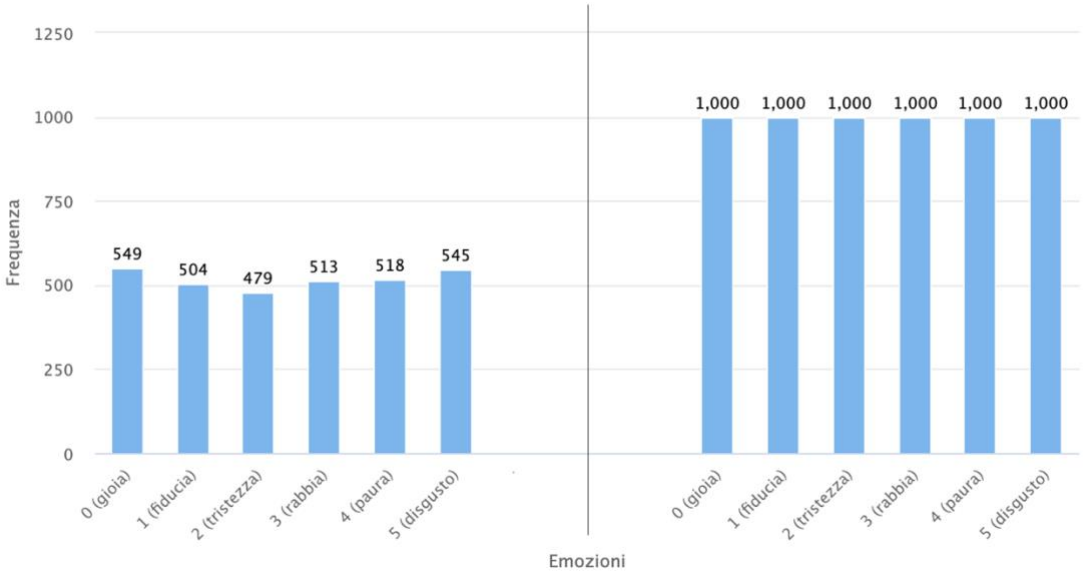


Figura XLVII - Dataset di partenza e dataset ottenuto con l'Oversampling su tutte le emozioni (training set)

Qui vengono mostrati i parametri utilizzati per il fine-tuning di AlBERTo (**Figura XLVIII**), effettuato utilizzando come training set il dataset appena illustrato.

```
#SET THE PARAMETERS
TRAIN_BATCH_SIZE = 512
PREDICT_BATCH_SIZE = 512
EVAL_BATCH_SIZE = 512
LEARNING_RATE = 2e-5
NUM_TRAIN_EPOCHS = 10.0
MAX_SEQ_LENGTH = 128
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 1000
SAVE_SUMMARY_STEPS = 500
```

Figura XLVIII - Parametri utilizzati per il training

4.2.4 Test con dataset MultiEmotions-It

Il modello ottenuto attraverso il fine-tuning di AlBERTo con il dataset estratto da Twitter è stato testato con il dataset MultiEmotions-It. La composizione di quest'ultimo è stata precedentemente illustrata, ma per maggiore chiarezza è riportata anche di seguito (**Figura XLIX**).

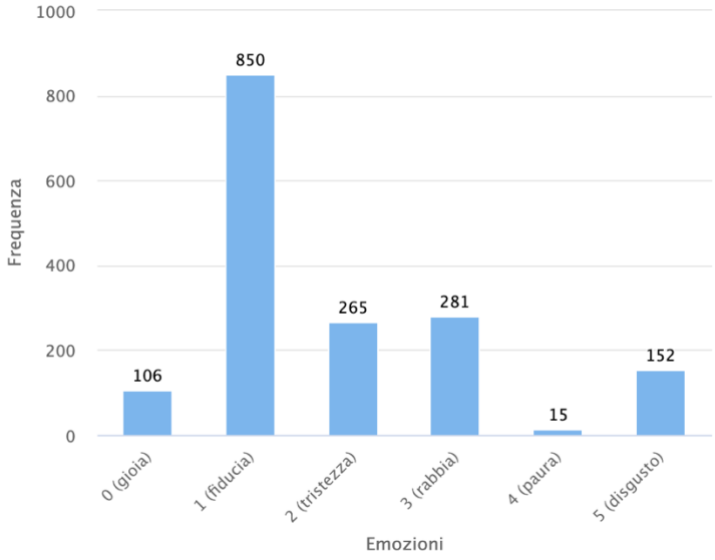
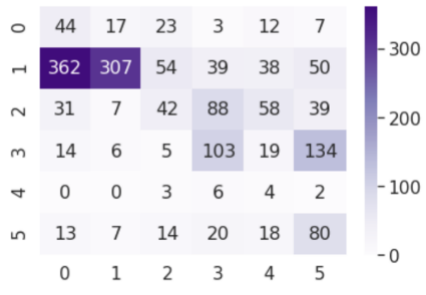


Figura XLIX - Dataset MultiEmotions-It (test set)

Le predizioni del test set non hanno portato ai risultati sperati. Confrontando questa valutazione con il test avente il training e il test set invertiti, il modello allenato con il dataset MultiEmotions-It e testato con il Twitter Dataset è stato in grado di predire in modo leggermente migliore le frasi del test set. Dall'immagine seguente si nota, come nel caso antecedente, che le coppie di emozioni maggiormente confuse sono gioia/fiducia, rabbia/disgusto e tristezza/paura (**Figura L**).



	0 (gioia)	1 (fiducia)	2 (tristezza)	3 (rabbia)	4 (paura)	5 (disgusto)
Precision	0.094828	0.892442	0.297872	0.397683	0.026846	0.256410
Recall	0.415094	0.361176	0.158491	0.366548	0.266667	0.526316

```
Accuratezza: 0.347513481126423
F1 score = 0.27510165467934555
```

Figura L - Metriche di valutazione per il test del modello con il dataset MultiEmotions-It

5 ESPERIMENTI CON QUATTRO EMOZIONI

In questo capitolo vengono illustrati tutti gli esperimenti svolti considerando solamente le emozioni: gioia, tristezza, rabbia, paura. L'ordine delle prove con quattro emozioni è lo stesso di quello esposto per sei emozioni: si procede costruendo il modello con il dataset MultiEmotions-It e testandolo con il Twitter Dataset; poi questi due dataset vengono invertiti per il training e il testing del modello.

5.1 Costruzione modello con dataset MultiEmotions-It

5.1.1 Valutazione con Oversampling su tutte le classi minoritarie

Per la valutazione del modello con la Stratified 10-Fold Cross Validation si è preso in considerazione il dataset MultiEmotions-It con quattro emozioni (**Figura LI**), costruito secondo la metodologia esposta nel paragrafo 3.1.

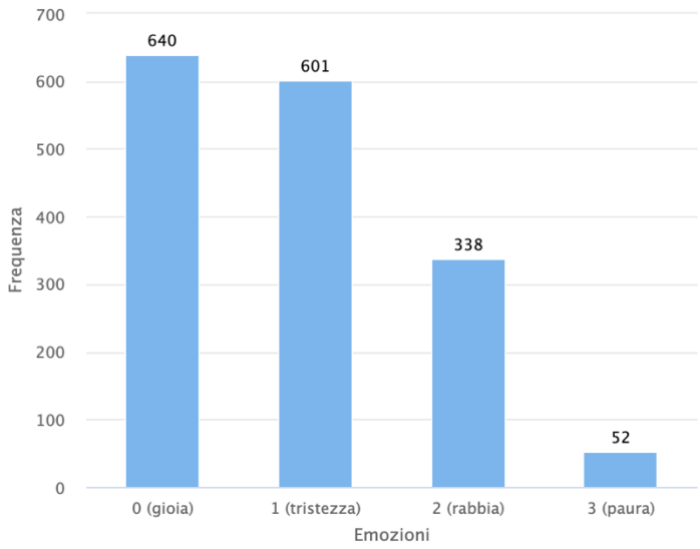


Figura LI - Distribuzione dataset MultiEmotions-It con quattro emozioni

Dal momento che tale dataset è molto sbilanciato, si è deciso di applicare l'Oversampling a tutte le classi, tranne la maggioritaria, nel training set che si ottiene ad ogni split del dataset durante la convalida incrociata (**Figura LII**).

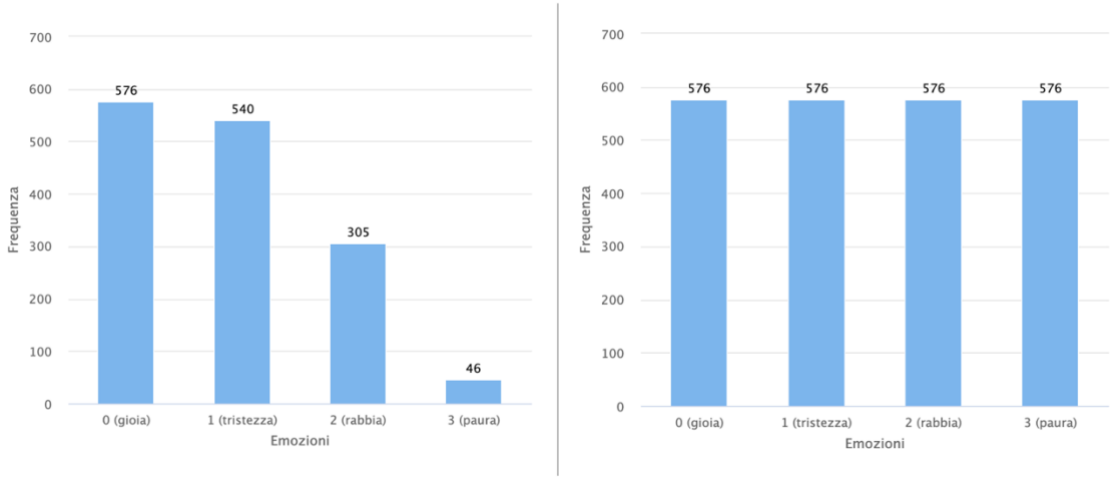
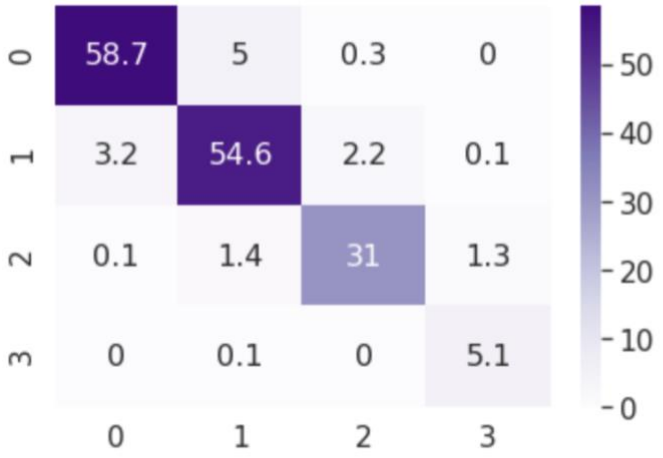


Figura LII - Dataset MultiEmotions-It PRIMA e DOPO l'Oversampling su tutte le classi tranne la maggioritaria

Con la valutazione del modello attraverso la Stratified 10-Fold Cross Validation si sono ottenuti ottimi risultati, con un'accuratezza e un *F1 score* maggiori di 0.9. Osservando la matrice di confusione (**Figura LIII**) si evidenzia che non è stato applicato l'Oversampling al test set, infatti esso risulta avere le stesse proporzioni del dataset MultiEmotions-It originale (**Figura LI**).



	0 (gioia)	1 (tristezza)	2 (rabbia)	3 (paura)
Precision	0.946774	0.893617	0.925373	0.784615
Recall	0.917188	0.908486	0.917160	0.980769

```
Accuratezza: 0.9161117761484364
F1 score: 0.90891786423257
```

Figura LIII - Valutazione tramite Stratified 10-Fold Cross Validation

5.1.2 Model Building

Avendo ottenuto buoni risultati attraverso il training del modello con il dataset MultiEmotions-It con Oversampling, si è deciso di replicare questa metodologia per la costruzione del modello da testare con il Twitter Dataset. L'intero dataset MultiEmotions-It con Oversampling su tutte le classi minoritarie è stato quindi considerato come training set **(Figura LIV**).

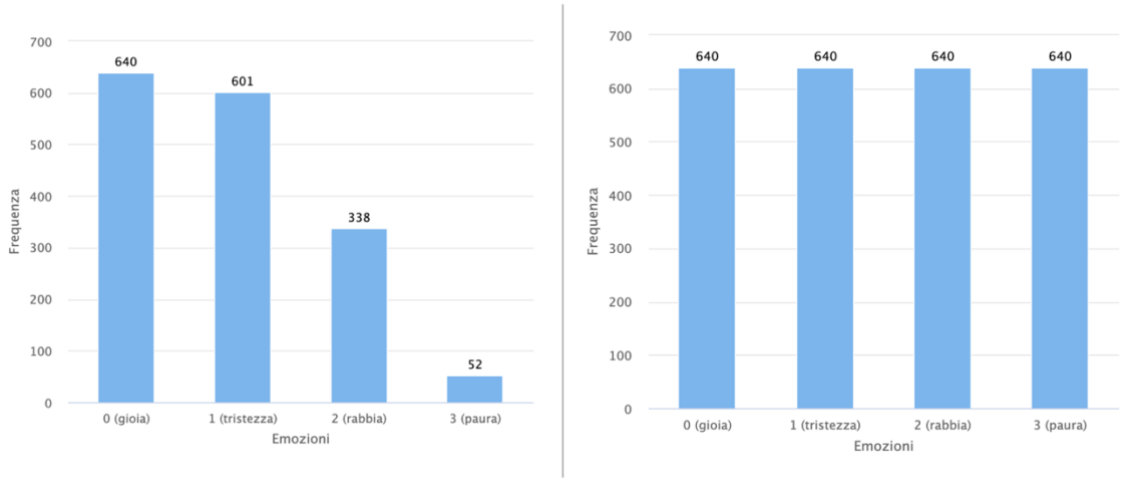


Figura LIV - Dataset di partenza e dataset ottenuto con l'Oversampling (training set)

Di seguito sono illustrati i parametri utilizzati per il fine-tuning di AlBERTo (**Figura LV**), effettuato utilizzando come training set il dataset sopra mostrato.

```
#SET THE PARAMETERS
TRAIN_BATCH_SIZE = 512
PREDICT_BATCH_SIZE = 512
EVAL_BATCH_SIZE = 512
LEARNING_RATE = 2e-5
NUM_TRAIN_EPOCHS = 10.0
MAX_SEQ_LENGTH = 128
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 1000
SAVE_SUMMARY_STEPS = 500
```

Figura LV - Parametri utilizzati per il training

5.1.3 Test con Twitter Dataset

Il modello costruito come appena descritto è stato testato con il dataset estratto da Twitter pulito manualmente per quattro emozioni (**Figura LVI**).

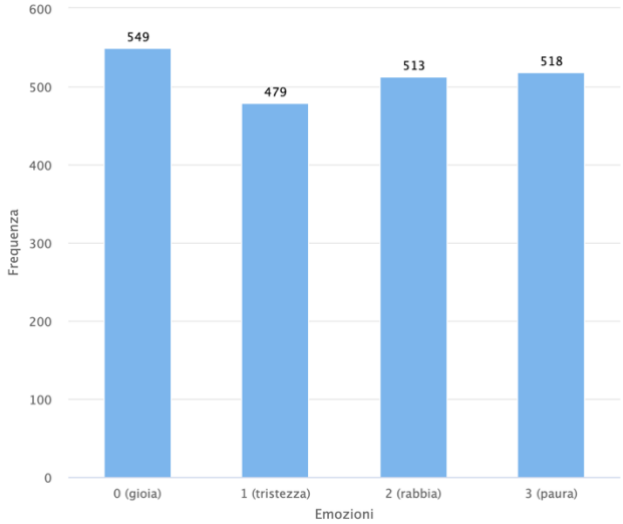
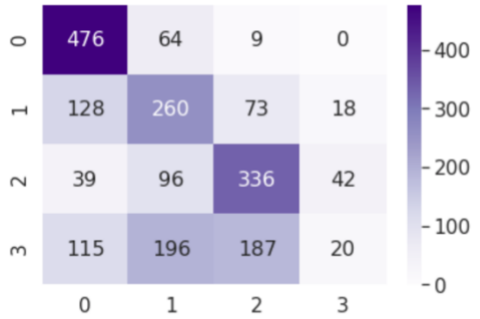


Figura LVI - Twitter Dataset pulito manualmente (test set)

In generale, i risultati ottenuti sono significativamente migliori rispetto alla valutazione con sei emozioni, in quanto l'accuratezza e l'*F1 score* sono aumentati di circa il 15%. Non considerando le emozioni "fiducia" e "disgusto" si è ovviato il problema della misclassificazione tra gioia/fiducia e rabbia/disgusto, per cui le frasi esprimenti gioia e rabbia sono predette in maniera più accurata dal modello. Si nota però, dalla matrice di confusione, che l'emozione "paura" non viene quasi mai riconosciuta e i testi classificati con questa emozione vengono predetti spesso come "tristezza" o "rabbia" (**Figura LVII**). Questo problema può essere causato dal fatto che il dataset MultiEmotions-It con cui si è allenato il modello contiene solamente 52 frasi esprimenti paura: anche se esse sono state replicate con l'Oversampling, le parole chiave che il modello riesce ad associare a questa emozione sono limitate.



	0 (gioia)	1 (tristezza)	2 (rabbia)	3 (paura)
Precision	0.627968	0.422078	0.555372	0.25000
Recall	0.867031	0.542797	0.654971	0.03861

```
Accuratezza: 0.5303545410393394
F1 score = 0.4678086095073952
```

Figura LVII - Valutazione tramite Stratified 10-Fold Cross Validation

5.2 Costruzione modello con Twitter Dataset

Il procedimento appena descritto è stato replicato utilizzando però come training set il dataset estratto da Twitter e come test set il dataset MultiEmotions-It, entrambi con quattro emozioni.

5.2.1 Valutazione con Twitter Dataset

Innanzitutto, è stato valutato il modello con il Twitter Dataset pulito manualmente per quattro emozioni (**Figura LVIII**) attraverso la tecnica di valutazione Stratified 10-Fold Cross Validation.

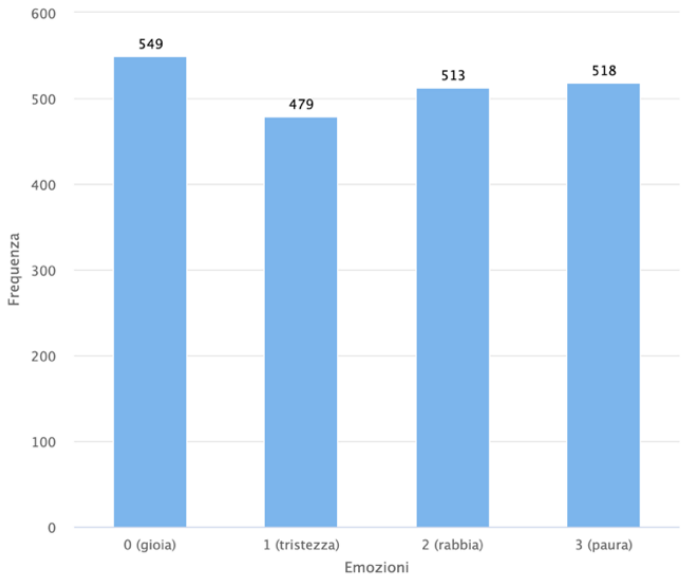
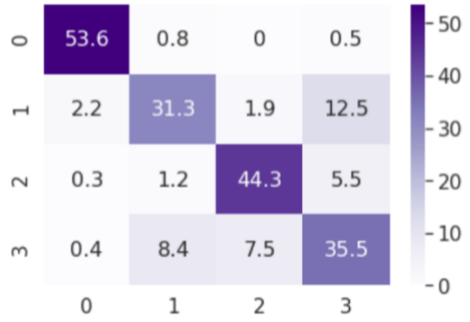


Figura LVIII - Twitter Dataset pulito manualmente con quattro emozioni

I risultati della validazione incrociata mostrano che le emozioni gioia e rabbia sono ben classificate, mentre la tristezza e la paura sono maggiormente confuse (**Figura LIX**). I valori di accuratezza e *F1 score* sono simili e si aggirano intorno allo 0.8. Questo primo esperimento è stato svolto senza effettuare l'Oversampling, in quanto il Twitter Dataset è già di per sé abbastanza bilanciato. Dal momento che attraverso la Stratified 10-Fold Cross Validation si sono ottenuti buoni risultati, non si sono effettuate altre valutazioni con tecniche di pre-processing sul dataset.



	0 (gioia)	1 (tristezza)	2 (rabbia)	3 (paura)
Precision	0.948673	0.750600	0.824953	0.657407
Recall	0.976321	0.653445	0.863548	0.685328

```
Accuratezza: 0.7999289604546531
F1 score: 0.7937729456276209
```

Figura LIX - Valutazione tramite Stratified 10-Fold Cross Validation con Twitter Dataset pulito manualmente con quattro emozioni

5.2.2 Model Building

Per la costruzione del modello si è utilizzato come training set il dataset mostrato nel paragrafo precedente (**Figura LVIII**).

I parametri utilizzati per il training al fine di effettuare il fine-tuning di AlBERTo sono i seguenti (**Figura LX**).

```
#SET THE PARAMETERS
TRAIN_BATCH_SIZE = 512
PREDICT_BATCH_SIZE = 512
EVAL_BATCH_SIZE = 512
LEARNING_RATE = 2e-5
NUM_TRAIN_EPOCHS = 10.0
MAX_SEQ_LENGTH = 128
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 1000
SAVE_SUMMARY_STEPS = 500
```

Figura LX - Parametri utilizzati per il training

5.2.3 Test con dataset MultiEmotions-It

Il test del modello è stato effettuato con il dataset MultiEmotions-It, distribuito come mostrato di seguito (**Figura LXI**).

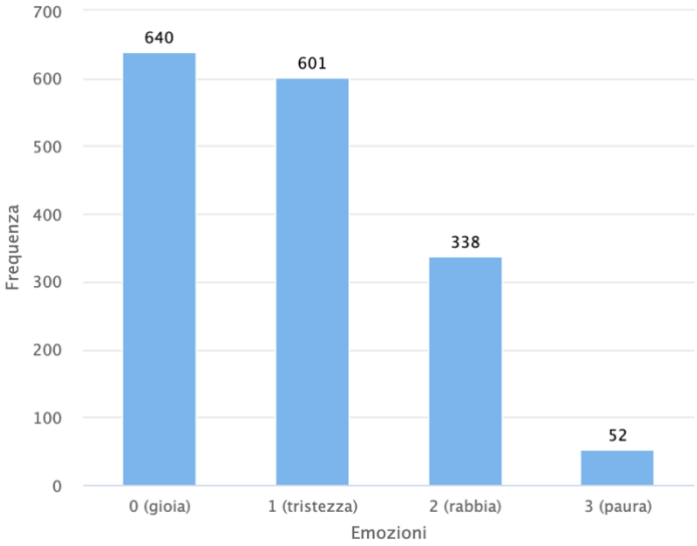
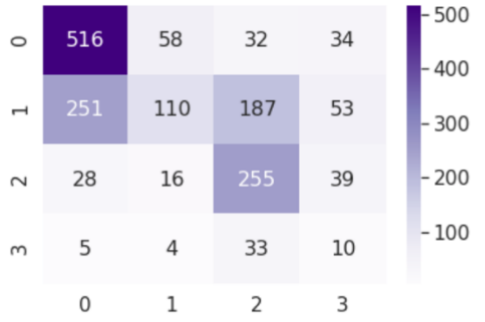


Figura LXI - Dataset MultiEmotions-It con quattro emozioni (test set)

Le predizioni del testing rispecchiano in parte i risultati ottenuti durante la validazione incrociata: la tristezza e la paura sono le emozioni classificate in modo peggiore. In particolare, dal basso valore di recall per tali emozioni si deduce che la maggior parte dei testi classificati con "paura" e "tristezza" sono predetti dal modello con l'emozione sbagliata (**Figura LXII**).



	0 (gioia)	1 (tristezza)	2 (rabbia)	3 (paura)
Precision	0.64500	0.585106	0.502959	0.073529
Recall	0.80625	0.183028	0.754438	0.192308

```
Accuratezza: 0.5462906192519926
F1 score = 0.42635847707373853
```

Figura LXII - Metriche di valutazione per il test del modello con il dataset MultiEmotions-It

6 DISCUSSIONE DEI RISULTATI

Gli esperimenti di questo lavoro di tesi si sono incentrati sul fine-tuning di AlBERTo, con lo scopo di ottenere un modello classificatore di emozioni per il testo italiano. In particolare, si sono utilizzate tecniche di pre-processing dei dati al fine di migliorare le prestazioni dei modelli ottenuti.
Sia per sei emozioni che per quattro, si è fatto il training con il dataset MultiEmotions-It, quindi il test con il Twitter Dataset, e viceversa. Di seguito verranno confrontati i risultati dei test.

La figura successiva (**Figura LXIII**) rappresenta i risultati ottenuti effettuando il fine-tuning di AlBERTo con il dataset MultiEmotions-It e testando il modello ottenuto con il Twitter Dataset pulito manualmente, per sei e per quattro emozioni.
Si nota che, considerando sei emozioni, le coppie maggiormente misclassificate sono gioia/fiducia, tristezza/paura e rabbia/disgusto. Per cui valutando solo quattro emozioni, la gioia e il disgusto vengono predette in modo più accurato, mentre rimane la confusione tra tristezza e paura. È inoltre interessante notare che i valori di *precision* e *recall* sono molto bassi per la paura in entrambi i casi. Questo è probabilmente causato dal fatto che il dataset MultiEmotions-It, con cui è stato allenato il modello, contenga pochi testi esprimenti paura e quindi il sistema non riesce a classificare questa emozione.

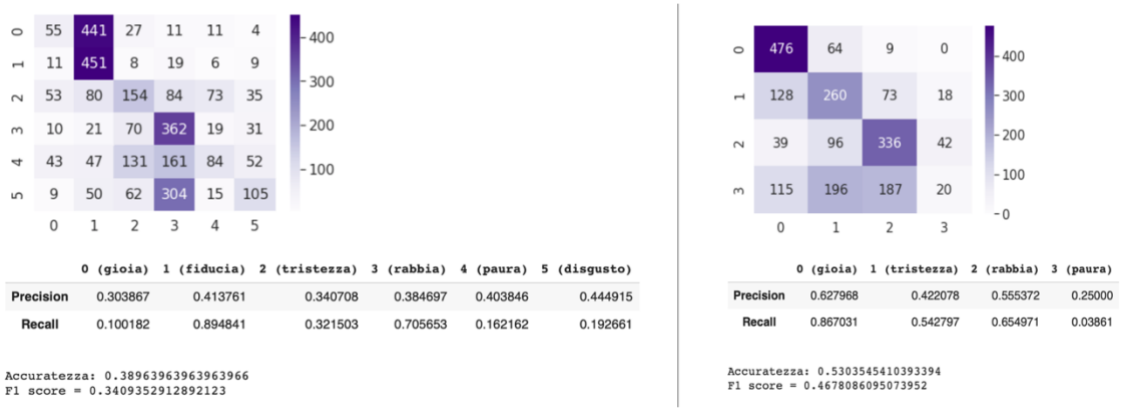


Figura LXIII - Modello testato con Twitter Dataset con sei e quattro emozioni

Di seguito (**Figura LXIV**) si confrontano i risultati per sei e quattro emozioni del modello di AlBERTo fine-tuned con il dataset estratto da Twitter e testato con il dataset MultiEmotions-It. Anche in questo caso si nota un netto miglioramento delle prestazioni nel caso di quattro emozioni.
Confrontando questi risultati con quelli precedenti (**Figura LXIII**), vediamo che il sistema costruito con il Twitter Dataset predice in modo leggermente meno accurato.

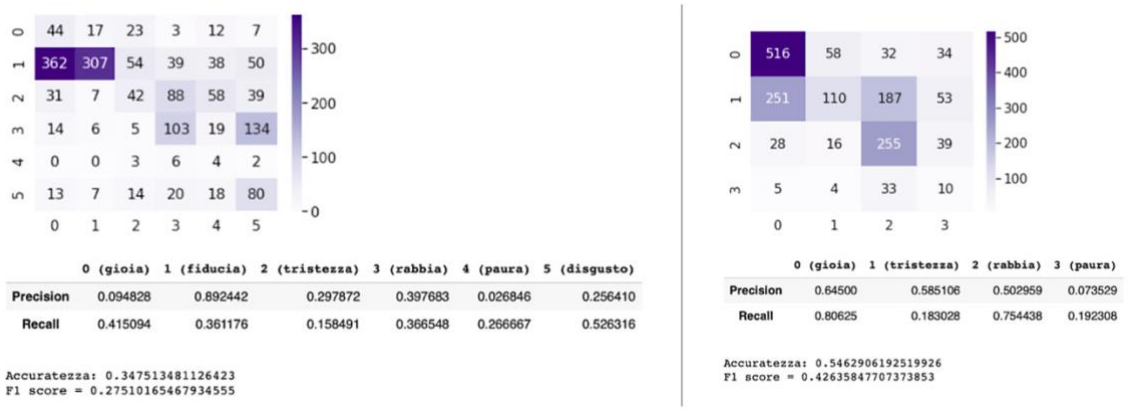


Figura LXIV - Modello testato con dataset MultiEmotions-It con sei e quattro emozioni

In generale, abbiamo visto che attraverso la tecnica di valutazione Stratified 10-Fold Cross Validation si sono sempre ottenuti ottimi risultati, mentre andando ad utilizzare un test set completamente diverso dal training set si ottiene massimo il 55% di accuratezza. Ne deduciamo che i modelli costruiti non sono in grado di generalizzare la classificazione a testi estrapolati da piattaforme e contesti diversi. Infatti, mentre il dataset MultiEmotions-It contiene commenti riferiti a video musicali e pubblicità su YouTube, il Twitter Dataset contiene tweets più generali.
Un altro fattore da tenere in considerazione che può aver contribuito ad aumentare *l'error rate*, è che il dataset estratto da Twitter è stato annotato in modo automatico attraverso le emoji, il che può aver generato annotazioni errate.

Questo progetto di tesi potrebbe essere esteso in futuro:

- ampliando il dataset MultiEmotions-It tramite l'estrazione di ulteriori commenti e l'annotazione manuale degli stessi, con lo scopo di ottenere un dataset più bilanciato;
- ripetendo gli esperimenti effettuando il fine-tuning di UmBERTo, anziché di AlBERTo: UmBERTo è un modello equivalente di BERT per il testo italiano e può essere specializzato per i tasks di classificazione come la Sentiment Analysis o l'Emotion Recognition. [10]

7 RINGRAZIAMENTI

Vorrei dedicare questo spazio del mio elaborato alle persone che hanno contribuito, con il loro supporto, alla realizzazione dello stesso.

Un sentito grazie alla mia relatrice Claudia Diamantini, senza la quale questo progetto di tesi non esisterebbe.

Un ringraziamento particolare va al mio correlatore, Alex Mircoli, per la sua infinita disponibilità.

Grazie ai miei compagni di corso Michelangelo e Simone, per aver condiviso con me ogni momento di questo percorso, riuscendo a farmi ridere anche nei momenti più bui.

Un immenso grazie a Mattia, per essere sempre stato al mio fianco.

In conclusione, ringrazio infinitamente la mia famiglia, che da sempre mi sostiene appoggiando ogni mia scelta.

BIBLIOGRAFIA

[1] Y. Zhang, New Advances in Machine Learning, InTech, 2010.

[2] L. M. John Paul Mueller, Deep Learning for dummies, John Wiley & Sons, 2019.

[3] P. L. B. Martin Anthony, Neural Network Learning, Cambridge University Press, 1999.

[4] P. Medici, 2017. [Online]. Available: http://www.ce.unipr.it/people/medici/geometry/node107.html.

[5] M.-W. C. K. L. K. T. Jacob Devlin, «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,» 2019.

[6] P. B. M. d. G. G. S. V. B. Marco Polignano, «ALBERTO: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets,» 2019.

[7] M. Morrelli, «Addestramento con Dataset Sbilanciati».

[8] E. F. M. A. H. Ian H. Witten, Data Mining - Practical Machine Learning Tools and Techniques, Elsevier, 2011.

[9] R. Sprugnoli, «MultiEmotions-It: a New Dataset for Opinion Polarity and Emotion Analysis for Italian,» 2020.

[10] D. N. D. H. Federico Bianchi, «FEEL-IT: Emotion and Sentiment Classification for the Italian Language,» 2021.

[11] M. L. S. H. Qiubing Ren, «Tectonic discrimination of olivine in basalt using data mining techniques based on major elements: a comparative study from multiple perspectives,» 2019.

[12] J. R. M. A. Roweida Mohammed, «Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results,» 2020.