Università degli studi di
Milano-Bicocca

Advanced Machine Learning

Final Project

Toxic Comment Classification

Authors:
Simone Monti - 807994 - s.monti21@campus.unimib.it
Vittorio Maggio - 817034 - v.maggio5@campus.unimib.it

22 gennaio 2021

Sommario

L'obiettivo di questo paper è quello di presentare un'analisi completa di un approccio di deep learning al fine di individuare e valutare dei modelli per classificare commenti del web ritenuti 'Tossici', in riferimento alla sfida *Kaggle*: *Toxic Comment Classification Challenge*. Nel corso della trattazione sarà fatta un'analisi esplorativa del dataset, saranno poi poi costruiti diversi tipi di reti neurali, quali Feedforward Neural Networks (FNN), Convolutional Neural Networks e Recurrent neural networks (RNN). Per ogni categoria di rete neurale saranno presentate diverse architetture, così da poter selezionare la migliore per ogni tipologia (sulla base delle performance ottenute sul validation set) e infine effettuare un confronto fra esse.

1 Introduzione

Negli ultimi anni i social networks e le comunità online hanno visto una larga crescita di interazioni al loro interno: si va dalla condivisione delle proprie esperienze personali alle discussioni politiche. Questo ha fatto si che essi abbiano assunto un ruolo sempre più centrale all'interno della vita sociale delle persone. Purtroppo però, i social non si sono rivelati essere solo luoghi di condivisione di idee e pensieri, ma anche luoghi in cui sono sempre più frequenti commenti tossici, d'odio, di minacce e di insulti. Questo sta portando non solo problemi di convivenza all'interno di queste comunità ma anche problemi più seri, di natura psicologica, alle vittime di tali commenti. L'individuazione e la gestione di questi commenti, dunque, è diventata fondamentale.

L'obiettivo proposto dalla challange di Kaggle *Toxic Comment Classification Challenge* [1] è quello di classificare correttamente ciascun commento in una o più categorie (tra cui per esempio *minacce, oscenità, insulti,* e *odio razziale*) in base al tipo di tossicità presente. Da notare che un commento può appartenere a più categorie contemporaneamente come può non presentare nessuna di queste e quindi essere esente dalla classificazione.

Dopo un'iniziale analisi esplorativa del dataset si è passati alla fase di preprocessing del testo. L'attenzione poi è passata sull'addestramento di più modelli appartenenti a 3 diverse categorie di reti neurali: Feedforward Neural Network(5), Convolutional Neural Network(5) e Recurrent Neural Network(3) [1].

[1] fra parentesi è possibile vedere il numero di modelli confrontati per ciascuna categoria

Per ogni tipologia di rete si è stabilito quale architettura fra quelle proposte avesse le performance migliori, così da ottenere un modello per ogni tipologia di rete neurale e poter effettuare un confronto fra di esse.

2 Dataset

Kaggle fornisce un ricco dataset di training (esente da missing value) formato da commenti presi dal web (nello specifico commenti tratti da Wikipedia), categorizzati manualmente in sei diverse label binarie in base all'abuso presente: *toxic, severe toxic, oscene, threat, insult e identity hate*. Ciascuno di essi può appartenere a più categorie contemporaneamente (oppure nessuna di esse). Il dataset di training è formato da 159571 commenti e nella prossima immagine è possibile osservare il numero di elementi positivi e negativi per ciascuna label.

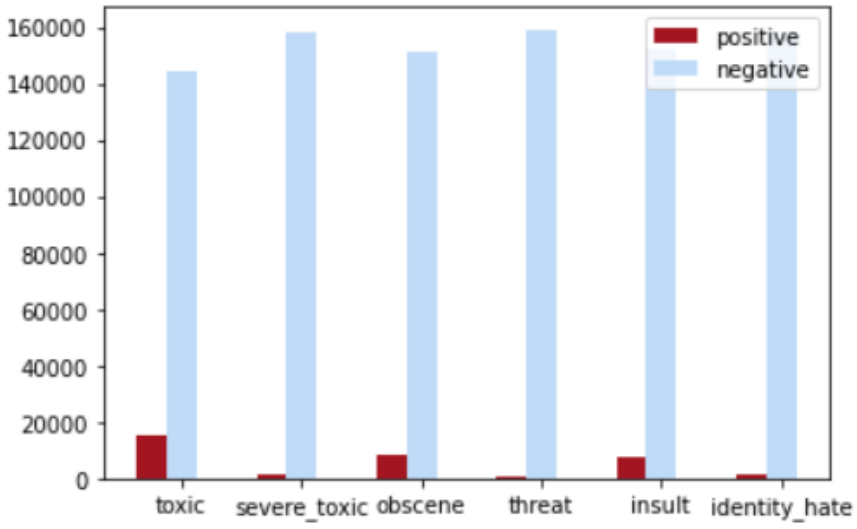

Figura 1: confronto presenza/assenza di tossicità

Data la possibilità di un commento di essere categorizzato in diverse classi contemporaneamente abbiamo voluto individuare la possibile presenza di relazioni fra queste ultime. La matrice di correlazione (Figura 2) evidenzia:

- (sorprendentemente) una bassa correlazione fra *toxic* e *severe toxic*,
- una forte correlazione fra le categorie *insult* e *obscene* (Figura 3),
- una correlazione fra *toxic* e rispettivamente *insult* e *obscene*.

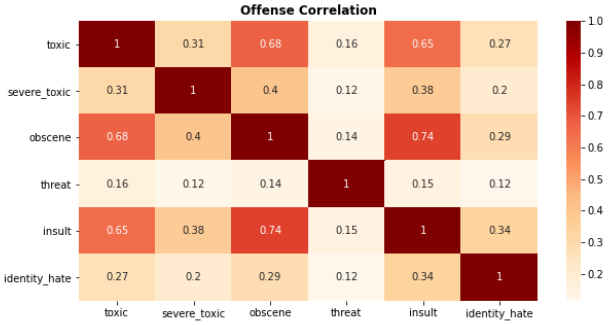

Figura 2: matrice di correlazione

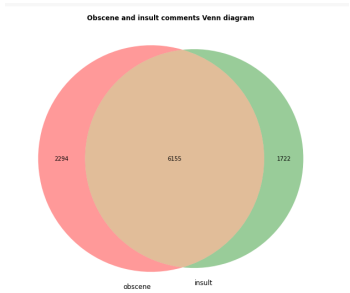

Figura 3: diagramma di Venn tra obscene e insult

È stato interessante individuare le parole che, effettivamente, discriminassero un commento ad appartenere ad una categoria, per far ciò è stato utilizzato WordCloud, un tool di visualizzazione che permette di evidenziare le parole più frequenti all'interno dei testi, applicandolo ai commenti di ciascuna categoria. Prima del suo utilizzo a ciascun commento sono state applicate le operazioni di tokenizzazione, conversione in lower case, rimozione delle stopword e stemming.

Figura 4: identity hate

Figura 5: obscene

Figura 6: insult

Figura 7: threat

Figura 8: severe toxic

Figura 9: toxic

Si nota come nelle classi correlate ci siano in evidenza parole simili all'interno del proprio diagramma WordCloud.

Ultimo step della fase di data exploration consiste nel calcolo della distribuzione della lunghezza dei commenti. Questo step sarà fondamentale, come vedremo nel prossimo capitolo, per individuare la dimensione ottimale dell'embedding utilizzato per la rappresentazione del testo.

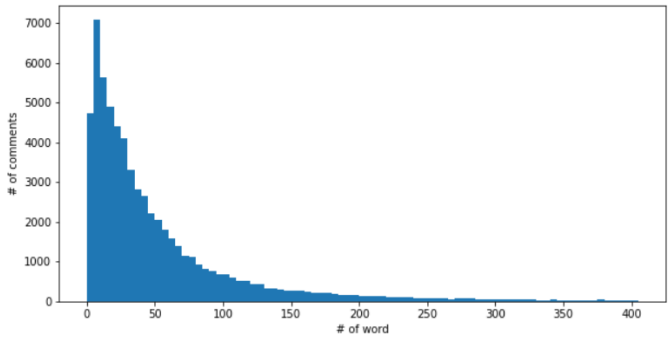

Figura 10: distribuzione della lunghezza dei commenti

Dall'istogramma (Figura 10) si può vedere come la stragrande maggioranza dei commenti abbia una lunghezza inferiore alle 200 parole.

3 L'approccio metodologico

L'obiettivo della trattazione, come anticipato, è quello di individuare il miglior modello fra quelli proposti per ogni tipologia di rete neurale e effettuare un confronto fra questi. L'immagine seguente (Figura 11) rappresenta la pipeline seguita.

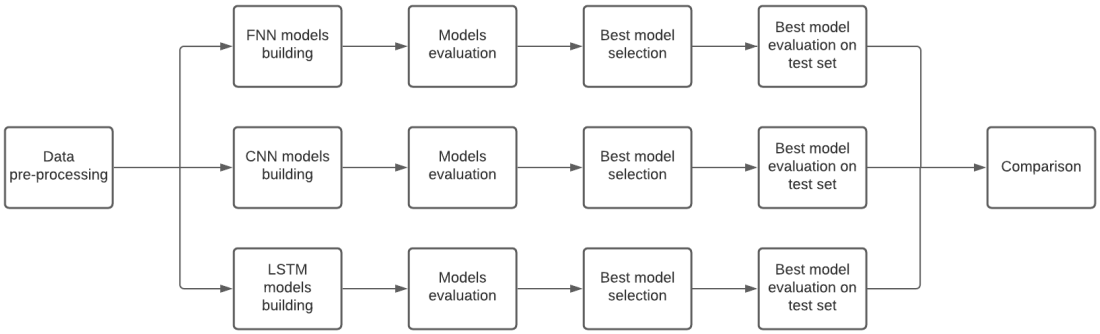

Figura 11: flowchart del processo

Nella fase di pre-processing è stata effettuata la tokenizzazione del testo ed è stato eseguito il padding per ottenere liste di token di lunghezza omogenea. Essa è stata settata pari a 200 in modo tale che la stragrande maggioranza dei commenti potesse essere considerata nella totalità del dettaglio (Figura 10). Questa fase di pre-processing è stata necessaria per addestrare il layer di embedding per la rappresentazione del testo, presente in ogni modello proposto.

Sono stati selezionati più modelli, appartenenti a tre differenti categorie di reti neurali, facendoli variare per numero di neuroni e di layer. Nell'elenco sottostante viene descritta l'architettura di ciascun modello, divisi per tiplogia di rete:

1. **Forward Neural Network**

   - **Model 1**: un layer Embedding (input_dim: 200, output_dim=128), un layer GlobalMaxPool1D, un layer Dense formato da 20 neuroni e con Relu come activation function, un layer Dropout (con rate uguale a 0.3), un layer Dense formato da 6 neuroni con funzione Sigmoid come activation function (avendo come output 6 differenti valori binari).
     Totale parametri addestrabili: 2,562,706;
   - **Model 2**: come il Modello 1 ma con 30 neuroni nel primo layer di tipo Dense.
     Totale parametri addestrabili: 2,564,056;
   - **Model 3**: come il Modello 1 ma con 40 neuroni nel primo layer di tipo Dense.
     Totale parmetri addestrabili: 2,565,406;

- **Model 4**: come il Modello 1 ma con 50 neuroni nel primo layer di tipo Dense.
  Totale parametri addestrabili: 2,566,756;
- **Model 5**:un layer Embedding, un layer GlobalMaxPool1D, un layer Dense formato da 40 neuroni e con Relu come activation function, un layer Dropout (con rate uguale a 0.3), un layer Dense formato da 20 neuroni e con Relu come activation function, un layer Dropout (con rate uguale a 0.3), un layer Dense formato da 6 neuroni con funzione Sigmoid come activation function (avendo come output 6 differenti valori binari).
  Totale parametri addestrabili: 2,566,106.

2. **Convolutional Neural Network**

- **Model 1**: un layer Embedding (input_dim: 200, output_dim=128), un layer SpatialDropout1D (con rate uguale 0.3), un layer Conv1D con 20 filtri da 4x1 e Relu come activation function, un layer BatchNormalization, un layer GlobalMaxPool1d, un layer Dropout (con rate uguale 0.3), un layer Dense formato da 20 neuroni e con Relu con activation function, un Dense layer con 6 neuroni e con sigmoid come activation function (avendo come output 6 differenti valori binari).
  Totale parametri addestrabili: 2,570,846;
- **Model 2**: uguale al Modello 1 ma il layer di tipo Conv1D ha 30 filtri di dimensione 4x1.
  Totale parametri addestrabili: 2,576,196;
- **Model 3**: uguale al Modello 1 ma il layer di tipo Conv1D ha 40 filtri di dimensione 4x1.
  Totale parametri addestrabili: 2,581,546;
- **Model 4**: uguale al Modello 1 ma il layer di tipo Conv1D ha 50 filtri di dimensione 4x1.
  Totale parametri addestrabili: 2,586,896;
- **Model 5**: uguale al Modello 1 ma il layer di tipo Conv1D ha 60 filtri di dimensione 4x1.
  Totale parametri addestrabili: 2,592,246.

3. **Recurrent Neural Network**

- **Smaller bidirectional**: un layer Embedding (input_dim: 200, output_dim=128), un layer SpatialDropout1D (con rate uguale a 0.3), un

layer LSTM bidirezionale con 25 neuroni e con Tanh come activation function e Sigmoid come recurrent activation function, un layer BatchNormalization, un layer GlobalMaxPool1d, un layer Dropout (con rate uguale a 0.3), un layer Dense con 15 neuroni e Relu come actiovation function, un layer Dense con 6 neuroni e sigmoid come activation function (avendo come output 6 differenti valori binari).
Totali parametri addestrabili: 2,591,761;

- **Bigger bidirectional**: uguale al modello 11 ma con il layer LSTM bidirezionale formato da 50 neuroni e con il primo Dense layer formato da 20 neuroni.
  Totale parametri addestrabili: 2,633,946;
- **Standard LSTM**: un layer Embedding, un layer SpatialDropout1D (con rate uguale a 0.3), un layer LSTM con 25 neuroni e con Tanh come activation function e Sigmoid come recurrent activation function, un layer BatchNormalization, un layer GlobalMaxPool1d, un layer Dropout (con rate uguale a 0.3), un layer Dense con 15 neuroni e Relu come actiovation function, un layer Dense con 6 neuroni e sigmoid come activation function (avendo come output 6 differenti valori binari).
  Totale parametri addestrabili: 2,575,936.

Per effettuare il training e la valutazione dei modelli abbiamo utilizzato la tecnica della 10-CrossValidation con i seguenti iperparametri:

- epoche: 10;
- batch size: 256;
- validation split: 0.1 (1 fold della Crossvalidation);
- optimizer Adam(con rate pari a 0.01);
- loss: BinaryCrossEntropy.

Per prevenire un possibile overfitting, è stata utilizzata la tecnica di regolarizzazione dell'early stopping (oltre a quella del Dropout, precedentemente indicata) su ogni modello, con i seguenti parametri:

- monitor: validation loss;
- patience: 2;

- mode: auto.

I modelli sono stati valutati sulla metrica F1 Score, implementata ed adattata per essere calcolata sul validation set direttamente durante l'addestramento in quanto non presente nelle metriche base fornite da Keras. È stato calcolato il suo intervallo di confidenza al 95% dagli importi ottenuti dalla crossvalidation. Gli intervalli sono stati successivamente confrontati ed è stata scelta la rete neurale con le migliori performance per ciascuna categoria.

Nell'ultimo step della pipeline i tre modelli finali sono stati nuovamente addestrati sul training set (applicando lo split train-validation) per poi essere valutati sul test set. Iperparametri utilizzati:

- epoche: 10;
- batch size: 256;
- validation split: 0.2;
- optimizer Adam(con rate pari a 0.01);
- loss: BinaryCrossEntropy.

Anche in questo caso è stata applicata la tecnina di regolarizzazione dell'early stopping.

4 Risultati e valutazioni

Di seguito sono riportati i risultati ottenuti dai diversi modelli applicando la 10-fold cross-validation e considerando un intervallo di confidenza del 95% sulla metrica F1:

Feedforward neural network:

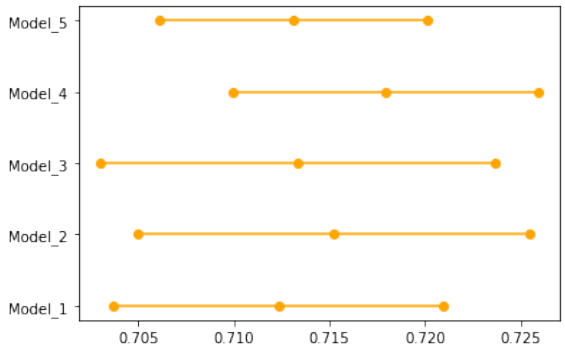

Figura 12: FNN: 10-CV, F1-score, C.I.: 95%

	Min	Mean	Max
Model_1	0.703	0.712	0.72
Model_2	0.704	0.715	0.725
Model_3	0.703	0.713	0.723
Model_4	0.709	0.717	0.725
Model_5	0.706	0.713	0.72

Tabella 1: FNN: 10-CV, F1-score, C.I.: 95%

Convolutional neural network:

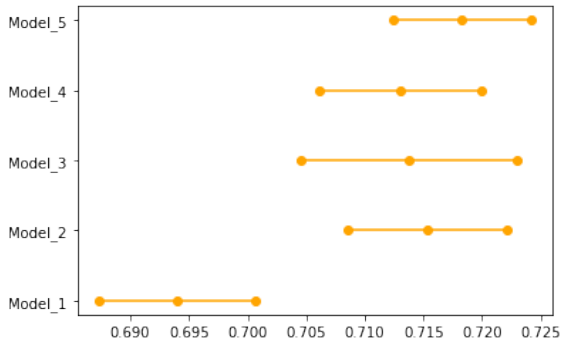

Figura 13: CNN: 10-CV, F1-score, C.I.: 95%

	Min	Mean	Max
Model_1	0.687	0.693	0.700
Model_2	0.708	0.715	0.722
Model_3	0.704	0.713	0.723
Model_4	0.706	0.713	0.719
Model_5	0.712	0.718	0.724

Tabella 2: CNN: 10-CV, F1-score, C.I.: 95%

Recurrent neural network:

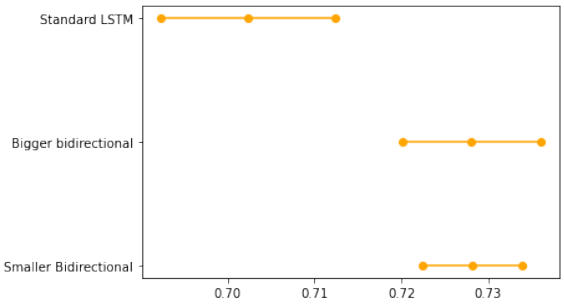

Figura 14: RNN: 10-CV, F1-score, C.I.: 95%

	Min	**Mean**	**Max**
Smaller bid.	0.722	0.728	0.733
Bigger bid.	0.720	0.728	0.736
st. LSTM	0.692	0.702	0.712

Tabella 3: RNN: 10-CV, F1-score, C.I.: 95%

I modelli migliori per ogni tipo di rete neurale sono risultati essere:

- Feedforward neural network: Model 4;
- Convolutional neural network: Model 5;
- Recurrent neural network: Smaller bidirectional.

Questi modelli sono stati quindi valutati sul test set, per ogni modello sono riportati matrice di confusione e relative misure di performance:

Feedforward neural network Model 4, evaluation on test set:

Actual \ Predicted	Toxic 0	Toxic 1
0	53948	3939
1	1134	4956

Actual \ Predicted	Severe toxic 0	Severe toxic 1
0	63531	79
1	340	27

Actual \ Predicted	Obscene 0	Obscene 1
0	58718	1568
1	957	2734

Actual \ Predicted	Threat 0	Threat 1
0	63728	38
1	188	23

Actual \ Predicted	Insult 0	Insult 1
0	59246	1304
1	1236	2191

Actual \ Predicted	Identity hate 0	Identity hate 1
0	63185	80
1	493	219

Tabella 4: Matrici di confusione FNN Model 4 (test set)

	Precision	Recall	F1-score	support
Toxic	0.59	0.79	0.67	6090
Severe_toxic	0.39	0.32	0.35	367
Obscene	0.67	0.72	0.69	3691
Threat	0.37	0.10	0.16	211
Insult	0.63	0.63	0.63	3427
Identity_hate	0.75	0.12	0.21	712
Micro avg	0.61	0.68	0.64	14498

Loss: 0.072
Accuracy: 0.995

Tabella 5: Perf. report FNN Model 4 (test set)

Convolutional neural network Model 5, evaluation on test set:

Actual \ Predicted	Toxic 0	Toxic 1
0	54088	3799
1	1260	4830

Actual \ Predicted	Severe toxic 0	Severe toxic 1
0	63269	341
1	227	140

Actual \ Predicted	Obscene 0	Obscene 1
0	58488	1798
1	960	2731

Actual \ Predicted	Threat 0	Threat 1
0	63766	0
1	211	0

Actual \ Predicted	Insult 0	Insult 1
0	58808	1742
1	1223	2204

Actual \ Predicted	Identity hate 0	Identity hate 1
0	63255	10
1	702	10

Tabella 6: Matrici di confusione CNN Model 5 (test set)

	Precision	Recall	F1-score	support
Toxic	0.56	0.79	0.65	6090
Severe_toxic	0.29	0.38	0.33	367
Obscene	0.60	0.74	0.66	3691
Threat	1.00	0.00	0.00	211
Insult	0.56	0.64	0.60	3427
Identity_hate	0.50	0.01	0.03	712
Micro avg	0.56	0.68	0.62	14498

Loss: 0.079
Accuracy: 0.978

Tabella 7: Perf. report CNN Model 5 (test set)

Recurrent neural network, evaluation on test set:

Actual \ Predicted	Toxic	0	1
	0	53875	4013
	1	1048	5042

Actual \ Predicted	Severe toxic	0	1
	0	63590	21
	1	344	23

Actual \ Predicted	Obscene	0	1
	0	59127	1160
	1	1119	2572

Actual \ Predicted	Threat	0	1
	0	63767	0
	1	211	0

Actual \ Predicted	Insult	0	1
	0	59379	1172
	1	1373	2054

Actual \ Predicted	Identity hate	0	1
	0	63266	0
	1	712	0

Tabella 8: Matrici di confusione RNN Smaller bidirectional (test set)

	Precision	Recall	F1-score	support
Toxic	0.56	0.83	0.67	6090
Severe_toxic	0.52	0.06	0.11	367
Obscene	0.69	0.70	0.69	3691
Threat	1.00	0.00	0.00	211
Insult	0.64	0.60	0.62	3427
Identity_hate	1.00	0.00	0.00	712
Micro avg	0.57	0.71	0.63	14498

Loss: 0.073
Accuracy: 0.998

Tabella 9: Perf. report RNN Smaller bidirectional (test set)

5 Discussione dei risultati

Per valutare le performance dei modelli precedentemente descritti, considerare come metrica principale la misura di accuracy non sarebbe risultato sufficiente, a causa del dataset fortemente sbilanciato (Figura 1): il suo valore risulterebbe fortemente biased. Si è quindi, utilizzata la misura di F1-score che viene interpretata come la media armonica di precision e recall.

Per ogni tipologia di rete neurale si sono confrontati i modelli implementati utilizzando un intervallo di confidenza al 95% della metrica F1, calcolata sui risultati ottenuti dai rispettivi modelli sul validation set durante i diversi cicli di cross-validation.

- **FNN** *(Figura 12, Tabella 1)*: il modello feedforward che ha raggiunto prestazioni migliori è stato il *Model_4*: esso ha ottenuto valori di limete inferiore, media e limite superiore dell' F1-score più alti fra i modelli considerati. Si noti dai valori ottenuti dal *Model_5* come l'aggiunta di un layer al *Model_4* non abbia portato a un miglioramento delle performance. In generale nessun modello ha prevalso nettamente su di un altro: tutti i modelli hanno ottenuto prestazioni paragonabili.
- **CNN** *(Figura 13, Tabella 2)*: anche in questo caso i modelli hanno ottenuto performance molto simili fra loro (eccetto per il Model_1 che ha mostrato

performance nettamente inferiori), quello che ha ottenuto valori di F1 maggiori è stao il *Model_ 5*.

- **RNN** *(Figura 14, Tabella 3)*: la differenza fra delle performance di questi modelli non è dovuta alla dimensione della rete ma alla presenza di una feature: la bidirezionalità. Si può notare come il modello standard di LSTM non possa competere con le performance ottenute con i modelli bidirezionali, che presentano performance pressoché identiche ma abbiamo scelto il modello *Smaller bidirectional* per via del numero di parametri minore.

I tre modelli scelti (appartenenti ognuno a una tipologia di rete neurale diversa) sono stati quindi valutati sul test set. Come è possibile vedere dai risultati riportati nella sezione precedente (Tabelle: 5,7,9) non è presente una differenza significativa fra i 3 modelli considerati, il modello feedforward è quello che ha ottenuto il valore maggiore di F1-score pari a 0.64 (Tabella 5), a seguire il modello di Recurrent neural network ha ottenuto un punteggio di F1 pari a 0.63 e l'architettura della Convolutional neural network un punteggio di 0.62.

Quello che è possibile notare da tutti i modelli è la difficoltà a individuare i commenti appartenenti alle classi 'severe_toxic', 'threat' e 'identity_hate', questo perché come si nota dalla Figura 1, il numero di commenti appartenenti a queste categorie è significativamente minore rispetto alla numerosità delle altre categorie. Inoltre si noti come nonostante i valori mediocri ottenuti dalla metrica F1, tutti i modelli presentano un'accuratezza estremamente elevata, questo è riconducibile al fatto che, come notato in precedenza e come riportato dalla Figura 1, il dataset è fortemente sbilanciato: il numero di commenti contrassegnati come non tossici è significativamente maggiore rispetto ai commenti tossici.

Per un eventuale sviluppo futuro si potrebbe ovviare al problema dell'estremo sbilanciamento usando la tecniche di Data Augmentation [2], un ulteriore sviluppo futuro potrebbe consistere nel testare layers di embedding pre-addestrati come Word2Vec[3], Glove[4] o BERT[5].

6 Conclusione

In questo paper si è voluta affrontare una pipeline completa di un approccio di deep learning per risolvere la challenge di kaggle *Toxic Comment Classification*. Sono state analizzate reti modelli di reti neurali appartenenti a tipologie diverse, che hanno

avuto un comportamento simile fra loro: nessuna tipologia di rete è prevalsa su di un altra, tutte hanno ottenuto prestazioni mediocri. Come notato nella sezione precedente i problemi riscontrati dai modelli sono riconducibili allo sbilanciamento del dataset e alla scarsa numerosità di alcune categorie di commenti.

Riferimenti bibliografici

[1] Kaggle - Toxic Comment Classification
https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

[2] EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks
https://arxiv.org/abs/1901.11196

[3] Efficient Estimation of Word Representations in Vector Space
https://arxiv.org/pdf/1301.3781.pdf

[4] Glove: Global Vectors for Word Representation
https://www.researchgate.net/publication/284576917_Glove_Global_Vectors_for_Word_Representation

[5] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
https://arxiv.org/pdf/1810.04805.pdf