UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

Deep Learning
A.A. 2020-2021
Prof. Ivan Serina
Luca Putelli

BioCreative VII (DrugProt track)

Stefano Trerotola (716197)

Introduzione

Questo documento riporta approcci e risultati della mia partecipazione alla sfida BioCreative VII con traccia DrugProt[1]: un compito di analisi testuale al fine di rilevare informazioni sulle interazioni tra chimici e geni.

L'utilizzo di BERT, nella variante riaddestrata su letteratura biomedica e clinica, ha permesso di ottenere risultati soddisfacenti ma non particolarmente elevati: la metrica recall è infatti generalmente buona (con valori attorno all'75-90%) per quelle classi ben rappresentate nel training set fornito, mentre è pessima per quelle classi mal rappresentate.

[1] Descrizione della sfida: https://biocreative.bioinformatics.udel.edu/tasks/biocreative-vii/track-1/

Indice

1 BlueBERT

BlueBERT [2] è il modello di deep learning scelto per affrontare il compito descritto precedentemente. L'acronimo si espande in Biomedical Language Understanding Evaluation - Bidirectional Encoder Representations from Transformers: la prima parte si riferisce al fatto che BlueBERT è un riaddestramento dell'architettura BERT su letteratura biomedica e clinica. Esistono, infatti, svariate specializzazioni di BERT (a sua volta addestrato su testi da Wikipedia e BookCorpus) in ambiti scientifici e medici. Nello specifico, l'addestramento è stato fatto su dieci dataset basati su cinque compiti differenti (somiglianza frasi, riconoscimento entità, classificazione documenti, inferenza ed estrazione relazioni – il compito richiesto da BioCreative VII). Il modello di partenza che ho scelto è BlueBERT-Base, uncased, addestrato su PubMed (e non su MIMIC-III). Ho scelto la versione base in quanto più rapida da addestrare e più rapida nel produrre risultati, ho scelto la versione addestrata solo su PubMed in quanto i risultati esposti nella seguente tabella suggeriscono migliori prestazioni in questa versione (osservare la linea relativa a ChemProt, compito simile a DrugProt):

Task	Metrics	SOTA*	ELMo	BioBERT	BlueBERT			
					Base (P)	Base (P+M)	Large (P)	Large (P+M)
MedSTS	Pearson	83.6	68.6	84.5	84.5	**84.8**	84.6	83.2
BIOSSES	Pearson	84.8	60.2	82.7	89.3	**91.6**	86.3	75.1
BC5CDR-disease	F	84.1	83.9	85.9	**86.6**	85.4	82.9	83.8
BC5CDR-chemical	F	93.3	91.5	93.0	**93.5**	92.4	91.7	91.1
ShARe/CLEFE	F	70.0	75.6	72.8	75.4	**77.1**	72.7	74.4
DDI	F	72.9	78.9	78.8	78.1	79.4	**79.9**	76.3
ChemProt	F	64.1	66.6	71.3	72.5	69.2	**74.4**	65.1
i2b2	F	73.7	71.2	72.2	74.4	**76.4**	73.3	73.9
HoC	F	81.5	80.0	82.9	85.3	83.1	**87.3**	85.3
MedNLI	acc	73.5	71.4	80.5	82.2	**84.0**	81.5	83.8
Total			78.8	80.5	82.2	**82.3**	81.5	79.2

[2] BlueBERT: https://medium.com/@manasmohanty/ncbi-bluebert-ncbi-bert-using-tensorflow-weights-with-huggingface-transformers-15a7ec27fc3d

Riguardo a BERT[3], in breve, è una rete neurale bidirezionale basata sul concetto di attenzione specializzata in rappresentazione del linguaggio naturale. La rete è di libero uso ma di proprietà di Google.

A differenza di molti precedenti approcci all'interpretazione del linguaggio naturale, l'analisi del testo non viene effettuata in maniera sequenziale (token dopo token) sfruttando cicli (ricorrenze) all'interno dell'architettura per catturare input di lunghezza variabile: infatti, la frase in ingresso viene spezzata in token (rappresentazioni vettoriali di dimensione fissa) e processata interamente dalla rete BERT (che può essere visto come un affinamento della precedente architettura Transformers). L'aggettivo bidirezionale non è dunque da intendersi come descrizione di una rete ricorrente che legge la frase in avanti e successivamente all'indietro, ma come una puntualizzazione del fatto che la rappresentazione di ogni parola (token) dipende dalla sua posizione nella frase e da quella delle restanti parole, in entrambe le direzioni di lettura.

BERT non richiede necessariamente la classificazione del testo sul quale andrà ad addestrarsi (compito oneroso in quanto da essere svolto da esseri umani), pertanto è stato possibile addestrarlo su grandi moli di testo. Tale addestramento "non supervisionato" consiste nel trovare quali siano parole che vengono casualmente mascherate in una frase.

L'ottimizzatore impiegato è Adam, che, tuttavia, richiede un gran numero di vettori temporanei e quindi contribuisce a rendere BERT un modello particolarmente pesante nell'uso e nell'addestramento senza TPU. BERT viene fornito in due dimensioni:

- **BERT-Base**: 12-layer, 768-hidden, 12-heads, 110M parameters
- **BERT-Large**: 24-layer, 1024-hidden, 16-heads, 340M parameters

[3] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding 1810.04805.pdf (arxiv.org)

2 Preprocessing Dataset ed esecuzione

Il dataset di BioCreative VII si compone di tre file:

- Il file “abstracts” contiene titoli e testi su cui si basa il lavoro di estrazione delle relazioni. La lunghezza media di questi testi è 2148 caratteri.
- Il file “entities” contiene i riferimenti alle entità mediche citate in ogni abstract. Riferendo un abstract tramite il suo id, numera le entità con indice Tx, dove x è un indice interno all’abstract, viene riportato il nome completo dell’entità, la posizione nel testo (numero carattere iniziale e finale) e il tipo (CHEMICAL o GENE).
- Il file “relations” contiene la lista delle relazioni positive tra chemicals e genes. Riferendo un abstract per id, e le due entità (utilizzando l’indice Tx presente nel file entities), definisce il tipo di relazione tra le tredici presenti (quattordici considerando la classe negativa, aggiunta successivamente): queste sono le classi che il modello costruito dovrà apprendere a predire.

Viene fornito sia un dataset di Training (17288 relazioni positive su 3500 abstract) che uno di Development (3765 relazioni positive su 750 abstract). Il file entities di training distingue tra GENE-Y e GENE-N, mentre il file entities di development considera entrambe indistintamente come GENE. Vista questa differenza, e preso atto che BioCreative fornirà il test set in formato GENE, ho scelto di ignorare questa differenza unendo anche in fase di addestramento le classi Y e N (operazione eseguita con editor di testo comune).

2.1 Creazione elementi negativi

Il primo passo di elaborazione del dataset consiste nell’arricchimento del training e validation set con relazioni negative. Non avendo un criterio per costruire queste istanze utilizzando conoscenze di dominio considero come interazioni negative tutte le possibili relazioni non dichiarate come positive nell’elenco delle relazioni. Questa è probabilmente una scelta che ha effetti negativi sull’apprendimento in quanto non vi è alcuna garanzia che tutte le possibili relazioni positive tra chimici e geni siano state rilevate e scritte nell’elenco relazioni.

Lo script falseexample.py riscrive il file delle relazioni ricopiando quelle positive ed aggiungendo le rimanenti combinazioni classificandole come FALSE. Da notare che, sebbene non evidente dalla struttura del file relations (la scelta di chiamare Arg1 ed Arg2 al posto di Chemical e Gene non è d’aiuto), sono possibili solo relazioni tra chimico e gene, non sono quindi possibili relazioni tra entità dello stesso tipo.

Rilevo come le frasi negative siano, all'incirca, trenta volte in numero quelle positive. L'approccio scelto è addestrare la rete con circa pari numero di frasi positive e negative, e fornire la validazione (e dunque i risultati al successivo capitolo) considerando tutte le frasi negative. Per seguire questo approccio:

- Eseguire lo script con parametri train, no: questo produrrà il file di relazioni di addestramento. Tale file verrà nominato relations2.tsv, ed andrà sostituito al file relations nella cartella training.
- Eseguire lo script con parametri dev, si: questo produrrà il file di relazioni di valutazione, che potrà essere effettuata a fine addestramento. Pertanto occorrerà rinominare il file generato (sempre di nome relations2.tsv) in test.tsv

2.2 Costruzione training e test set per BlueBERT

BlueBERT si attende come input:

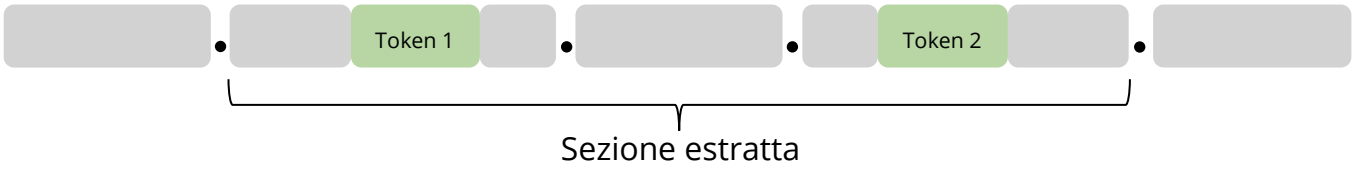
- Un indice, costruito come indice di abstract concatenato con indice delle due entità
- Il frammento di abstract che dovrebbe contenere le due entità della relazione
- La classe obbiettivo

Il file di addestramento (come anche quello di validazione o di test) ha quindi la stessa cardinalità del file delle relazioni.

Il compito dello script preprocessor.py è quindi di, per ogni relazione dal file relations:

- rilevare le entità citate trovandone nome, tipo e posizione nel file entità
- sostituire esse all'interno dell'abstract di riferimento (il loro nome sarà sostituito con la rispettiva tipologia, in formato @CHEMICAL$ e @GENE$)
- assegnare la classe corretta, prendendola dal file relations

Dato che BERT non può accettare frasi di lunghezza illimitata come input, viene trascritta solo la porzione di abstract che include le due entità in gioco nella relazione: nello specifico, viene estratto tutto il testo compreso fra le due entità, tutto il testo che precede la prima entità da inizio frase, e il testo che va dalla seconda entità a fine della frase in cui compare.



Si spera che ciò sia sufficiente a non sforare la lunghezza massima dopo la tokenizzazione interna al pre-processatore di BERT, altrimenti il significato dell'input formato ne

risulterebbe compromesso. Ricordo come BERT, di default, accetti come input solo frasi contenenti esattamente 128 token (ad una parola possono corrispondere uno o più token), le frasi più corte vengono allungate con del padding. La formazione della stringa di input illustrata è del tutto arbitraria e dettata unicamente dall'intuitività dell'informare la rete del testo che circonda chimico e gene in analisi.

Il programma preprocessor.py è configurabile con parametri:

- è richiesto specificare su quale dataset lavorare, scegliendo tra train, dev o test
- è richiesto quale procedura avviare: le procedure 0 ed 1 eseguono quanto descritto fin ora, la procedura 2, invece, si occupa di creare il file relations partendo dal file entities (necessario per produrre le sottomissioni dal test set, come illustrato in seguito)
- se scelte le procedure 0 o 1, che sono tra loro equipollenti, viene data la possibilità di dividere l'output tra due file train.tsv (due terzi delle istanze) e test.tsv (il restante terzo): ciò può essere comodo per eseguire esperimenti mantenendo i tempi di addestramento e predizione brevi.

2.3 Addestramento e produzione predizioni

Lo script chiama.py si occupa di semplificare l'interazione con l'esecutore run_bluebert.py. Ciò nonostante, alcuni parametri come learning rate e batch size sono da cambiare manualmente nello script bluebert/bluebert/run_bluebert.py (sono definiti a inizio file come parametri di Tensorflow). L'interazione con lo script è intuitiva. Segnalo che questo, come tutti gli script va eseguito dalla cartella radice del progetto. Prima di addestrate sarebbe, inoltre, buona norma ripristinare il modello BlueBERT originale, in quanto non mi è chiaro se l'addestramento intacchi anche la copia presente nella cartella bluebert. Il file bluebert/reset.sh automatizza l'operazione se una copia viene mantenuta in bluebert/vanilla.

2.4 Valutazione

Il file val.py si occupa di produrre una valutazione delle performance del modello addestrato sfruttando il classification report di sklearn. Le valutazioni vengono effettuate considerando come classe identificata dal modello quella con la maggiore probabilità in uscita dalla softmax (l'output delle predizioni di BlueBERT è appunto questo) nel file output/test_results.tsv, e considerando come classe vera quella presente nel file test.tsv. Il

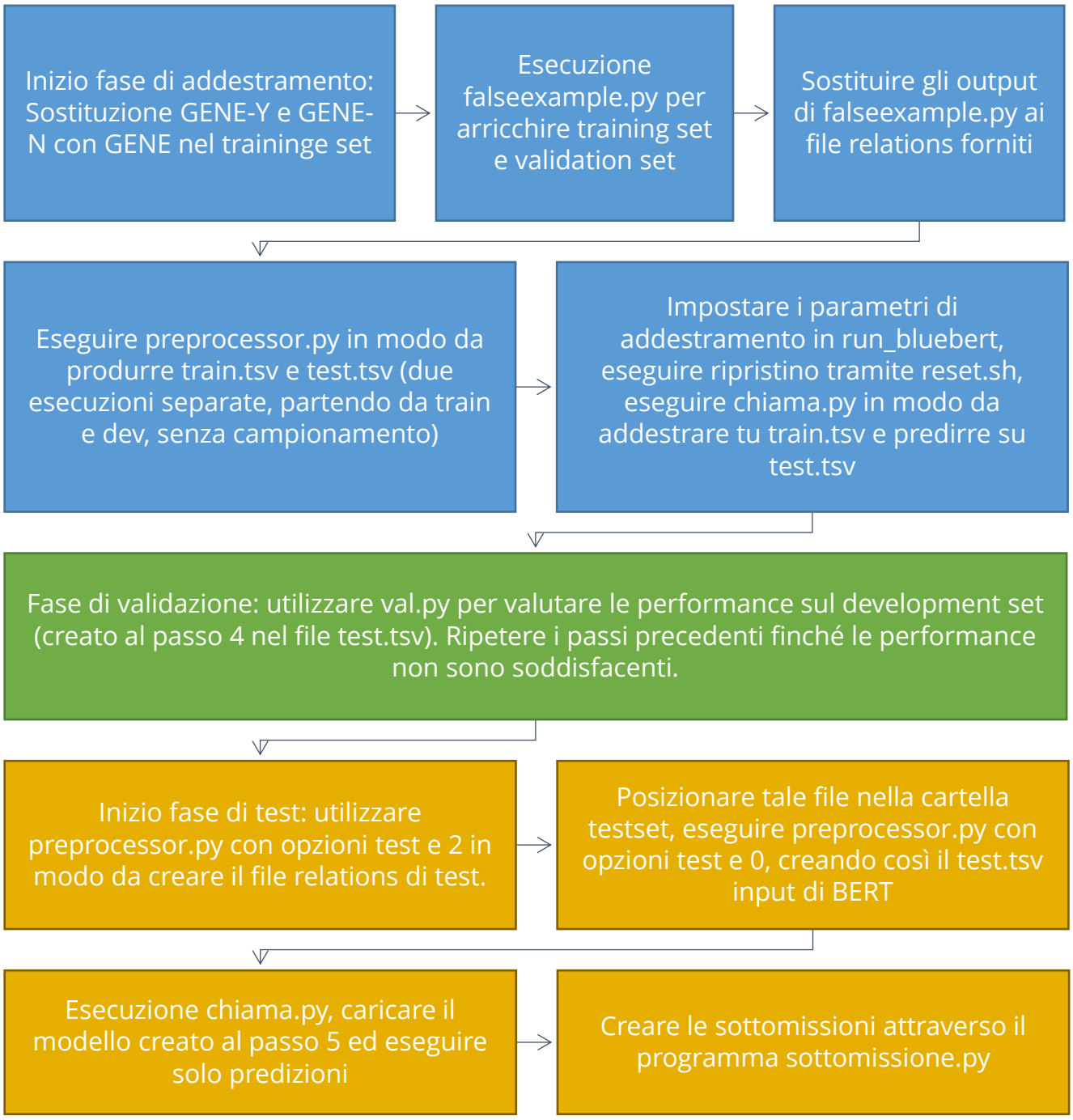
programma permette, inoltre, di salvare l'output di sklearn in un file csv, funzione utile per la successiva creazione dei grafici in questa relazione.

2.5 Generare sottomissione DrugProt

Il test set fornito da DrugProt VII non contiene il file relations, che è, appunto, ciò che ci si aspetta venga prodotto a seguito del lavoro di sottomissione. Tuttavia, per come è strutturato il codice illustrato fin ora, per generare i file di addestramento e di test da dare in input a BlueBERT serve un file di relazioni. Il programma 2 del file preprocessor.py si occupa quindi di ricreare artificialmente il file relations a partire dal file entità fornito, utilizzando come classe per ciascuna relazione un "?". Una volta costruito questo file relations e posizionato nella cartella testset è possibile richiamare preprocessor.py con opzioni test e 0 al fine di creare il file test.tsv dal quale generare le predizioni.

Una volta generate le predizioni con una rete precedentemente addestrata è necessario comporre la sottomissione nel formato atteso, dunque nel formato di relations. Il programma sottomissione.py prende in input le predizioni generate in output/test_results.tsv e il file di predizioni test.tsv, dunque compone nel file sottomissioni.tsv la sottomissione nel classico formato di relations.

Per ricapitolare, questo è il flusso di esecuzione:



3 Risultati

I risultati sono basati sulla predizione del development set arricchito con le frasi corrispondenti a relazioni negative, come precedentemente illustrato.

I principali parametri sui quali è possibile operare:

- Il numero di epoche, che ho fatto variare tra uno e dieci
- La dimensione dell'input alla rete in addestramento (batch_size), che ho fatto variare tra 16 e 64
- Il fattore di apprendimento (learning rate), che ho fatto variare attorno al valore di default di 5×10^{-5}. Ricordo che BERT utilizza Adam come ottimizzatore della discesa del gradiente.
- La percentuale di frasi con classe negativa da aggiungere dal training set.

Dove non diversamente specificato i risultati riportati derivano da addestramenti sull'intero training set arricchito con elementi di classe FALSE campionati con probabilità 1/30. Le metriche sono sempre calcolate sul development set arricchito con tutte le possibili frasi di classe FALSE.

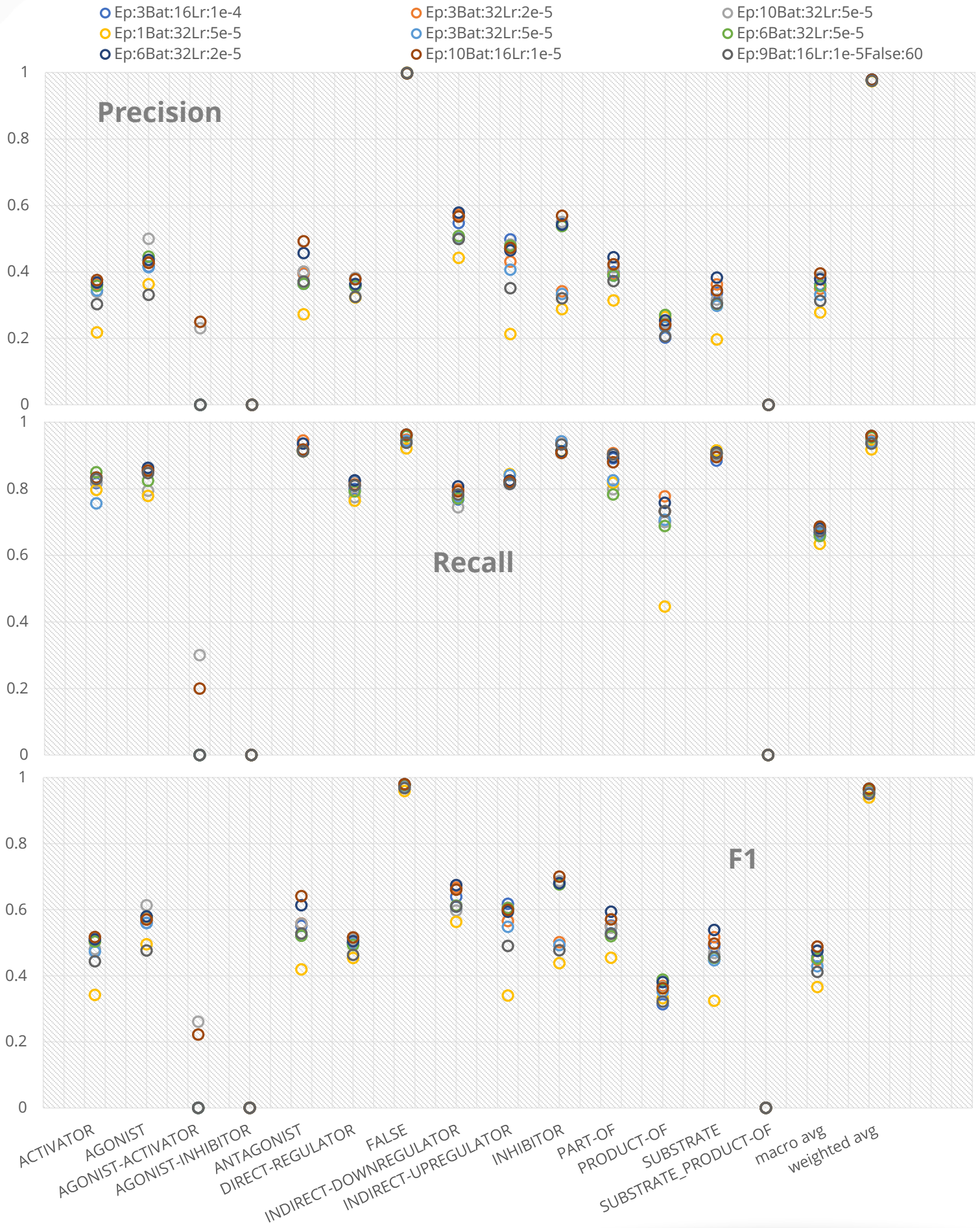
3.1 Analisi dei risultati

Dalle prove effettuate risulta come per ottenere buone performance sia necessario un buon numero di epoche.

Il parametro learning rate è quello più delicato in quanto si è osservato che tenerlo troppo elevato risulta disastroso: nell'esperimento con valore 3×10^{-4} (omesso nei grafici, ma riportato in seguito) la rete perde ogni capacità cognitiva, sintomo che il tasso di apprendimento troppo sostenuto ha cancellato il pre-addestramento di BERT piuttosto che raffinarlo.

La dimensione del batch non è risultato un parametro significativo negli esperimenti effettuati.

In definitiva, i migliori risultati si sono ottenuti con lunghi addestramenti con tasso di apprendimento basso.


Ep:3Bat:16Lr:1e-4
Ep:3Bat:32Lr:2e-5
Ep:10Bat:32Lr:5e-5
Ep:1Bat:32Lr:5e-5
Ep:3Bat:32Lr:5e-5
Ep:6Bat:32Lr:5e-5
Ep:6Bat:32Lr:2e-5
Ep:10Bat:16Lr:1e-5
Ep:9Bat:16Lr:1e-5False:60
Precision
Recall
F1
1
0.8
0.6
0.4
0.2
0
ACTIVATOR
AGONIST
AGONIST-ACTIVATOR
AGONIST-INHIBITOR
ANTAGONIST
DIRECT-REGULATOR
FALSE
INDIRECT-DOWNREGULATOR
INDIRECT-UPREGULATOR
INHIBITOR
PART-OF
PRODUCT-OF
SUBSTRATE
SUBSTRATE_PRODUCT-OF
macro avg
weighted avg

3.1.1 Epoche: 1, Batch size 32, Learning rate: 5×10^{-5} (default)

	precision	recall	f1-score	support
ACTIVATOR	0.22	0.80	0.34	246
AGONIST	0.36	0.78	0.50	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.27	0.91	0.42	218
DIRECT-REGULATOR	0.32	0.76	0.45	458
FALSE	1.00	0.92	0.96	100969
INDIRECT-DOWNREGULATOR	0.44	0.77	0.56	332
INDIRECT-UPREGULATOR	0.21	0.84	0.34	302
INHIBITOR	0.29	0.91	0.44	1152
PART-OF	0.31	0.82	0.45	258
PRODUCT-OF	0.26	0.45	0.33	157
SUBSTRATE	0.20	0.92	0.32	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.92	104733
macro avg	0.28	0.63	0.37	104733
weighted avg	0.97	0.92	0.94	104733

3.1.2 Epoche: 3, Batch size: 32, Learning rate: 5×10^{-5} (default)

	precision	recall	f1-score	support
ACTIVATOR	0.34	0.76	0.47	246
AGONIST	0.41	0.86	0.56	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.36	0.92	0.52	218
DIRECT-REGULATOR	0.36	0.82	0.50	458
FALSE	1.00	0.94	0.97	100969
INDIRECT-DOWNREGULATOR	0.50	0.78	0.61	332
INDIRECT-UPREGULATOR	0.41	0.84	0.55	302
INHIBITOR	0.33	0.94	0.49	1152
PART-OF	0.39	0.83	0.53	258
PRODUCT-OF	0.23	0.71	0.35	157
SUBSTRATE	0.30	0.89	0.45	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.94	104733
macro avg	0.33	0.66	0.43	104733
weighted avg	0.98	0.94	0.95	104733

3.1.3 Epoche: 6, Batch size: 32, Learning rate: 5×10^{-5} (default)

	precision	recall	f1-score	support
ACTIVATOR	0.36	0.85	0.50	246
AGONIST	0.45	0.82	0.58	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.36	0.91	0.52	218
DIRECT-REGULATOR	0.36	0.79	0.49	458
FALSE	1.00	0.96	0.98	100969
INDIRECT-DOWNREGULATOR	0.51	0.77	0.61	332
INDIRECT-UPREGULATOR	0.48	0.82	0.60	302
INHIBITOR	0.54	0.91	0.68	1152
PART-OF	0.39	0.78	0.52	258
PRODUCT-OF	0.27	0.69	0.39	157
SUBSTRATE	0.30	0.91	0.45	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.95	104733
macro avg	0.36	0.66	0.45	104733
weighted avg	0.98	0.95	0.96	104733

3.1.4 Epoche: 10, Batch size: 32, Learning rate: 5×10^{-5} (default)

	precision	recall	f1-score	support
ACTIVATOR	0.34	0.82	0.48	246
AGONIST	0.50	0.79	0.61	131
AGONIST-ACTIVATOR	0.23	0.30	0.26	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.40	0.92	0.56	218
DIRECT-REGULATOR	0.38	0.78	0.51	458
FALSE	1.00	0.96	0.98	100969
INDIRECT-DOWNREGULATOR	0.50	0.74	0.60	332
INDIRECT-UPREGULATOR	0.48	0.81	0.61	302
INHIBITOR	0.55	0.91	0.69	1152
PART-OF	0.41	0.80	0.55	258
PRODUCT-OF	0.25	0.70	0.36	157
SUBSTRATE	0.32	0.90	0.47	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.96	104733
macro avg	0.38	0.67	0.48	104733
weighted avg	0.98	0.96	0.96	104733

3.1.5 Epoche: 6, Batch size: 64, Learning rate: 3×10^{-4}

	precision	recall	f1-score	support
ACTIVATOR	0.00	0.00	0.00	246
AGONIST	0.00	0.00	0.00	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.00	0.00	0.00	218
DIRECT-REGULATOR	0.00	0.00	0.00	458
FALSE	0.96	1.00	0.98	100969
INDIRECT-DOWNREGULATOR	0.00	0.00	0.00	332
INDIRECT-UPREGULATOR	0.00	0.00	0.00	302
INHIBITOR	0.00	0.00	0.00	1152
PART-OF	0.00	0.00	0.00	258
PRODUCT-OF	0.00	0.00	0.00	157
SUBSTRATE	0.00	0.00	0.00	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.96	104733
macro avg	0.07	0.07	0.07	104733
weighted avg	0.93	0.96	0.95	104733

3.1.6 Epoche: 3, Batch size: 16, Learning rate: 1×10^{-4}

	precision	recall	f1-score	support
ACTIVATOR	0.37	0.82	0.51	246
AGONIST	0.42	0.85	0.56	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.39	0.91	0.55	218
DIRECT-REGULATOR	0.36	0.80	0.50	458
FALSE	1.00	0.96	0.98	100969
INDIRECT-DOWNREGULATOR	0.55	0.77	0.64	332
INDIRECT-UPREGULATOR	0.50	0.81	0.62	302
INHIBITOR	0.54	0.91	0.68	1152
PART-OF	0.40	0.89	0.55	258
PRODUCT-OF	0.20	0.70	0.31	157
SUBSTRATE	0.34	0.88	0.49	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.96	104733
macro avg	0.36	0.67	0.46	104733
weighted avg	0.98	0.96	0.96	104733

3.1.7 Epoche: 3, Batch size: 32, Learning rate: 2×10^{-5}

	precision	recall	f1-score	support
ACTIVATOR	0.36	0.83	0.50	246
AGONIST	0.43	0.85	0.57	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.40	0.94	0.56	218
DIRECT-REGULATOR	0.36	0.82	0.50	458
FALSE	1.00	0.95	0.97	100969
INDIRECT-DOWNREGULATOR	0.57	0.80	0.67	332
INDIRECT-UPREGULATOR	0.43	0.82	0.57	302
INHIBITOR	0.34	0.94	0.50	1152
PART-OF	0.39	0.91	0.55	258
PRODUCT-OF	0.24	0.78	0.37	157
SUBSTRATE	0.36	0.90	0.52	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.95	104733
macro avg	0.35	0.68	0.45	104733
weighted avg	0.98	0.95	0.96	104733

3.1.8 Epoche: 3, Batch size: 32, Learning rate: 2×10^{-5}

	precision	recall	f1-score	support
ACTIVATOR	0.37	0.83	0.51	246
AGONIST	0.44	0.86	0.58	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.46	0.94	0.61	218
DIRECT-REGULATOR	0.36	0.83	0.51	458
FALSE	1.00	0.96	0.98	100969
INDIRECT-DOWNREGULATOR	0.58	0.81	0.67	332
INDIRECT-UPREGULATOR	0.46	0.82	0.59	302
INHIBITOR	0.54	0.91	0.68	1152
PART-OF	0.44	0.90	0.59	258
PRODUCT-OF	0.25	0.76	0.38	157
SUBSTRATE	0.38	0.91	0.54	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.96	104733
macro avg	0.38	0.68	0.48	104733
weighted avg	0.98	0.96	0.97	104733

3.1.9 Epoche: 10, Batch size: 16, Learning rate: 1×10^{-5}

	precision	recall	f1-score	support
ACTIVATOR	0.38	0.83	0.52	246
AGONIST	0.43	0.85	0.57	131
AGONIST-ACTIVATOR	0.25	0.20	0.22	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.49	0.92	0.64	218
DIRECT-REGULATOR	0.38	0.81	0.52	458
FALSE	1.00	0.96	0.98	100969
INDIRECT-DOWNREGULATOR	0.57	0.79	0.66	332
INDIRECT-UPREGULATOR	0.47	0.82	0.60	302
INHIBITOR	0.57	0.91	0.70	1152
PART-OF	0.42	0.88	0.57	258
PRODUCT-OF	0.24	0.73	0.36	157
SUBSTRATE	0.34	0.89	0.50	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.96	104733
macro avg	0.40	0.69	0.49	104733
weighted avg	0.98	0.96	0.97	104733

3.1.10 1/60 FALSE, Epoche: 9, Batch size: 16, Learning rate: 1×10^{-5}

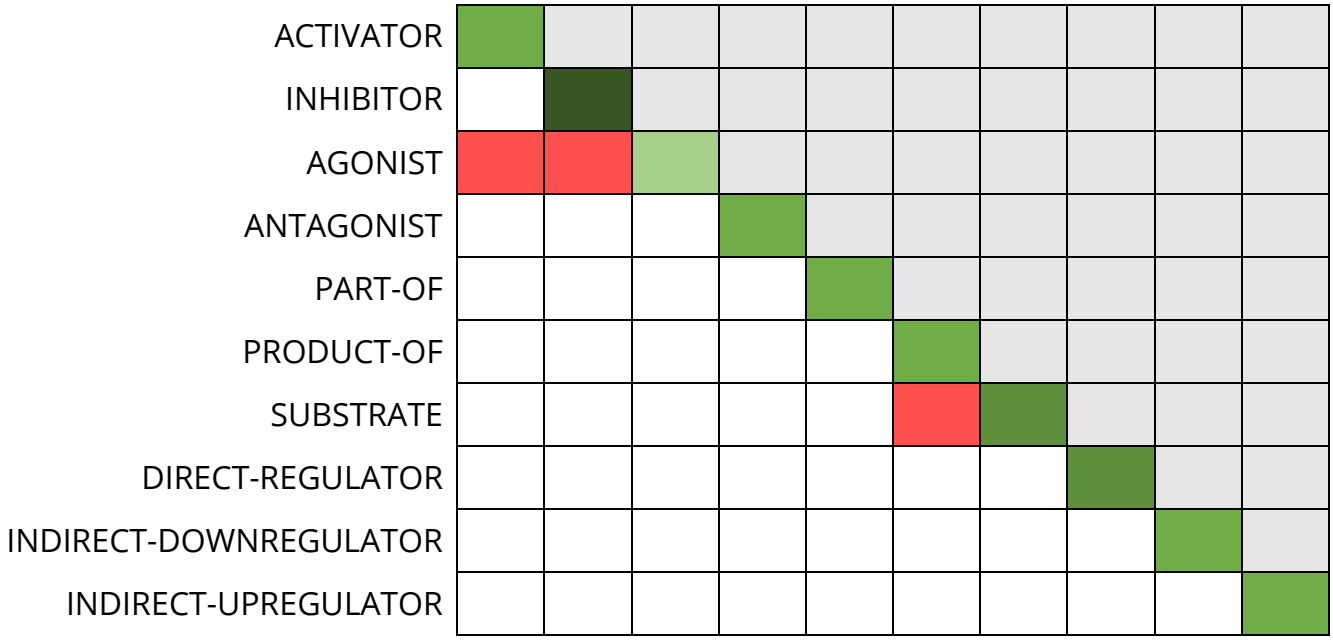
	precision	recall	f1-score	support
ACTIVATOR	0.30	0.83	0.44	246
AGONIST	0.33	0.85	0.48	131
AGONIST-ACTIVATOR	0.00	0.00	0.00	10
AGONIST-INHIBITOR	0.00	0.00	0.00	2
ANTAGONIST	0.37	0.91	0.53	218
DIRECT-REGULATOR	0.32	0.81	0.46	458
FALSE	1.00	0.94	0.97	100969
INDIRECT-DOWNREGULATOR	0.50	0.78	0.61	332
INDIRECT-UPREGULATOR	0.35	0.81	0.49	302
INHIBITOR	0.32	0.93	0.48	1152
PART-OF	0.37	0.90	0.53	258
PRODUCT-OF	0.21	0.73	0.32	157
SUBSTRATE	0.31	0.91	0.46	495
SUBSTRATE_PRODUCT-OF	0.00	0.00	0.00	3
accuracy			0.94	104733
macro avg	0.31	0.67	0.41	104733
weighted avg	0.98	0.94	0.95	104733

4 Conclusioni e futuri lavori

Futuri lavori potrebbero riguardare, oltre all'affinamento della tecnica impiegata basata su BlueBERT, o della elaborazione dei dati in ingresso al modello, anche al creare un modello generativo di quelle classi mal rappresentate nel training set, che possono essere considerate outlier.

Si nota, inoltre, che le tre classi meno rappresentate sono in realtà relative ad elementi che fanno parte di due classi contemporaneamente. Diversi approcci potrebbero essere impiegati alla soluzione di questo problema, uno di essi potrebbe essere quello di considerare questi elementi appartenenti a più classi ad avere etichetta target (dunque obbiettivo di uscita dalla softmax) valore di 0.5 per le due classi, eliminando dunque tre elementi da suddetta softmax di uscita. In fase di test e produzione risultati in cui la rete produce probabilità prossime a 0.5 per queste tre situazioni andrebbero rilevati e trattati separatamente.

Analizzando le uscite del modello attuale, si vede come, ad esempio, per la classe SUBSTRATE_PRODUCT-OF, mai predetta correttamente, essa viene spesso classificata come SUBSTRATE, e come classe seconda (con gran distacco, tuttavia) per probabilità PRODUCT-OF.



Un altro possibile lavoro potrebbe essere lo sviluppo di un ulteriore modello in grado di elaborare velocemente gli stessi input identificando le frasi che contengono molto probabilmente una relazione negativa: questo modello, che non può basarsi né su BERT né su altri derivati da Transformers, ma su tecnologie più leggere, sarebbe utile nelle situazioni in cui sia necessario ottenere risultati su grandi moli di dati velocemente. La rete neurale creata in questo compito verrebbe quindi interrogata solo sulle istanze non a priori etichettate come false.

In conclusione, è possibile affermare che il riaddestramento dell'architettura BlueBERT sul dataset DrugProt ha ottenuto discreti risultati, seppur non eccezionali. Lavorando in ambiente medico, probabilmente la metrica più significativa è il recall, e sicuramente i valori ottenuti non sono sufficienti per fare affidamento esclusivo alla rete addestrata, che può però essere usata come strumento automatico di supporto per lavori di ricerca.

Utilizzo

Il codice sviluppato è reperibile presso: Stefano-BS/BioCreativeVIIBlueBERT (github.com)

Tale sorgente contiene sia il codice da me sviluppato, sia quello necessario per l'utilizzo di BlueBERT. Vengono inclusi anche i dataset forniti, oltre ai dataset elaborati come illustrato alla sezione due. Non è presente, invece, il modello BlueBERT pre-addestrato (reperibile presso ncbi-nlp/bluebert: BlueBERT-Base, Uncased, PubMed (github.com)) o gli output degli addestramenti, predizioni e valutazioni, per ragioni di spazio. Segnalo, infine, che il contenuto della cartella bluebert/bert va sostituito a quello del pacchetto bert installabile tramite pip.