UNIVERSITÀ DEGLI STUDI DI ROMA TOR VERGATA

Macroarea di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea in Informatica

A.A. 2020/2021

Tesi di Laurea Triennale

Sistemi di Authorship Attribution basati su Deep Learning

Relatore

Prof. Fabio Massimo Zanzotto

Candidato

Federico Ranaldi

Abstract

Lo scopo di questo lavoro è quello di proporre dei sistemi di Authorship Attribution che si servono di modelli di apprendimento automatico.
L'Authorship Attribution è l'attività che consiste nell'identificare l'autore di un testo. Un sistema che svolge questa attività in maniera automatica si rivela utile alla risoluzione di problemi che appartengono a svariati campi(letterario,musicale,giuridico,...).
In particolare in questo lavoro verrà presentata un'applicazione nella quale si vuole riconoscere l'identità di un individuo che partecipa ad un forum tematico presente nel Dark Web.
In scenari come questi un sistema di Authorship Attribution può favorire la prevenzione contro le minacce alla sicurezza della collettività.
Negli ultimi anni il Deep Learning ha trovato terreno fertile nella potenza computazionale dei nuovi dispositivi hardware e nella crescente disponibilità di dati digitalizzati, meglio noti come Big Data.
Fra i modelli di apprendimento automatico che sfruttano le più recenti novità tecnologiche ci sono senza dubbio le reti neurali.
In questa tesi si vogliono studiare diverse implementazioni del processo di Author Attribution con l'utilizzo delle Feed-Forward-Neural-Networks e degli Encoders basati su Transformers.
Inoltre verranno discusse e valutate alcune tecniche di rappresentazione numerica dei dati, necessarie affinché questi possano essere processati da un modello di apprendimento automatico.

Contents

Introduzione

Il problema di riconoscere l'autore di un determinato testo si presenta in moltissime situazioni.

Riuscire a dedurre l'identità di colui che ha usato questa forma di comunicazione per danneggiare uno o più individui, può rappresentare una svolta nelle indagini delle forze dell'ordine.Si consideri un messaggio ostile rivolto alla vittima di un reato oppure che inneggia ad atti terroristici. In dinamiche ricorrenti e attuali come queste, l'attività di Authorship Attribution arriva a fornire supporto alla sicurezza di un paese.La sua importanza incide anche su questioni che caratterizzano altri ambiti da molto più tempo di quanto si possa pensare.

Infatti sin dall'antichità i precursori della filologia si occupavano di studiare in maniera analitica tutti gli aspetti concernenti i testi scritti che trattavano diverse tematiche (politiche,storiche,filosofiche,letterarie o scientifiche...).

Specialmente nella letteratura, l'analisi approfondita dei contenuti e dello stile di alcuni testi privi di firma che sono stati tramandati nei secoli, ha portato ad interessanti ipotesi sui loro autori e talvolta alla loro conferma.

L'intento di questo lavoro è quello di descrivere il processo di sviluppo di un sistema automatico, che sfruttando i principi e le tecniche del Machine Learning effettua Authorship Attribution.

Più precisamente questo sistema è basato sui metodi di Deep Learning ,una sottobranca del Machine Learning che viene implementata con le Neural Networks conosciute anche come Reti Neurali Artificiali (o semplicemente Reti Neurali).
Il modello di Deep Learning che si vuole implementare sono le reti neurali di tipo Feed Forward. Per funzionare ,le reti neurali, hanno bisogno di un certo tipo di dati che devono essere trattati in maniera tale che queste possano apprenderli ed in un secondo momento analizzarli per effettuare una previsione.
Dunque è importante descrivere la struttura e il contenuto dei dati nel corso della costruzione del sistema.
Infatti affinché si possa estrarre conoscenza dai dati , è necessario che questi abbiano una struttura e un linguaggio che ne esprime il contenuto comprensibili per chi li deve processare.Ed è ben noto che una macchina e un umano non ottengono informazioni a partire dallo stesso tipo di dati.
Verranno quindi descritte nella trattazione varie tecniche di rappresentazione di dati testuali.
Infine si vedranno delle applicazioni ,di quanto implementato, a diversi casi reali utilizzando dati strutturati presenti nel Web ed eventualmente modificati.
Con quest'ultima parte si vuole vedere in senso pratico l'utilità del sistema e darne una valutazione sul suo funzionamento per un problema specifico.

Background

In molti lavori relativi alla Text Analysis è stato affrontato il problema dell'Authorship Attribution.

In [17],gli autori propongono di effettuare Authorship Attribution su un insieme di testi scritti in lingua russa.I sistemi realizzati in questo lavoro prevedevano l'utilizzo di modelli di apprendimento come "SVM"(Support Vector Machines),"LSTM"(Long Short Term Memory) e "CNN with attention"(Convolutional Neural Networks con il meccanismo di "attention").Addestrando questi modelli con un dataset di 10 mila testi e 5 autori, i risultati ottenuti sono stati:

- 85% di accuracy usando SVM
- 59% di accuracy usando LSTM
- 75% di accuracy usando CNN with attention

In [20],viene proposto un metodo che sfrutta le tecniche di Data-Preprocessing e Machine Learning per effettuare Authorship Attribution su un insieme di recensioni.Il dataset su cui effettuare il task è costituito da circa 800 mila recensioni effettuate da circa 4 mila autori,dove ognuno di questi ne ha lasciate almeno 100.

I risultati ottenuti utilizzando degli algoritmi di Machine Learning che effettuano classificazione e tecniche basiche di Data-Preprocessing(come stemming e lemmatization) sono:

- 47.1% di accuracy usando MNB("Multinomial Naive Bayes")
- 84.2% di accuracy usando SVM
- 80.7% di accuracy usando ME("Maximum Entropy")

In [13] ,viene effettuata Authorship Attribution su 3 datasets ,ognuno dei quali contiene i testi di 5 autori.I 3 datasets sono ricavati rispettivamente dai noti "Blog","Enron Email" e "IMDb".
In quest'ultimo lavoro gli autori propongono l'utilizzo di una versione customizzata del Transformer "Bert" che chiamano "BertAA" per effettuare feature-extraction.La classificazione viene eseguita sulle features ricavate da BertAA utilizzando come modello di apprendimento "LR"(Logistic Regression).
Lo stato dell'arte sui più recenti lavori ha mostrato che tra i migliori modelli da implementare per la realizzazione di un sistema di Authorship Attribution ci sono sicuramente le SVM per la classificazione e Transformers come Bert per la Feature-Extraction.

Il processo di un sistema basato su Machine Learning

Una volta definito il problema che si vuole risolvere e scelto un insieme di dati strutturati che lo caratterizzano si procede con la realizzazione del sistema.È importante infatti capire di fronte a che tipo di problema ci si trova al fine di scegliere l'approccio corretto e quindi implementare i modelli più adatti.
Il processo di funzionamento di un sistema basato su Machine Learning segue in linea di massima le seguenti fasi:

1. Preprocessing dei Dati
2. Vettorizzazione dei Dati
3. Splitting dei dati
4. Implementazione di un modello di apprendimento
5. Addestramento del modello
6. Test del modello
7. Valutazione del modello

Ognuna di queste fasi è implementata all'interno di un sistema utilizzando diversi metodi in base al compito che si deve svolgere.
A seconda del tipo problema e dei dati a disposizione si sceglie un modello di apprendimento rispetto a un altro così come le tecniche di preprocessing e di Vettorizzazione.
In questo lavoro, per la realizzazione di alcuni sistemi si scelgono le Feed Forward Neural Networks come struttura del modello di apprendimento.Mentre per quel che riguarda il Preprocessing e la Vettorizzazione vengono descritte e confrontate diverse tecniche.
Il Machine Learning è una branca dell'Intelligenza Artificiale che si basa fondamentalmente su tecniche statistica applicate ai dati.Dunque un sistema di Machine Learning che effettua un determinato task si basa fortemente sui dati a disposizione relativi ad esso.
Fra i più importanti aspetti che riguardano la progettazione di sistemi di Machine Learning ci sono i dati e tutto ciò a cui essi sono legati.Infatti, come si vedrà in seguito,el realizzare un sistema si attribuisce molta importanza alla provenienza,alla tipologia e alla quantità dei dati a disposizione.

Il problema dell'Authorship Attribution

Il problema dell'Authorship Attribution consiste nell'attribuire ad un determinato autore un testo scritto in linguaggio naturale proveniente da diverse fonti(libri di qualsiasi genere letterario,pagine Web,Social Media...).
Nel Machine Learning va spiegato chiaramente e formalmente come è fatta un'istanza del problema che si vuole risolvere.

4.1 Definizione formale del problema

Dunque,dato un insieme di testi $T = \{t_1, t_2, \ldots, t_n\}$, un insieme di autori $A = \{a_1, a_2, \ldots, a_m\}$ con $m \leq n$.
Sia $T' \subset T$,dove $T' = \{t_1, t_2, \ldots, t_k\}$, un insieme di testi per cui gli autori sono conosciuti,cioè esiste un insieme di coppie $D = \{(t_i, a_j)\}_{i=1}^{k}$.
Quindi esiste una funzione $g\colon \mathbf{T'} \longrightarrow \mathbf{A}$, dove $T' = \{t_1, t_2, \ldots, t_k\}$ che assegna un autore ad ogni testo appartenente al sottinsieme T'.
L'obbiettivo è quello di trovare una funzione $f\colon \mathbf{T''} \longrightarrow \mathbf{A}$ con $T'' = \{t_{k+1}, t2, \ldots, t_n\}$ cioè $T'' \subset T$ tale che $T' \cup T'' = T$ .Quindi si vuole una funzione che assegni ad ogni testo dell'insieme T'' un autore dell'insieme A.

4.2 Definizione del problema nel ML

Per come è stato formalizzato il problema si inserisce fra quelli affrontati Supervised Machine Learning.Il Supervised Machine Learning (o Apprendimento Supervisionato) è un tipo di Machine Learning che si occupa di risolvere problemi per i quali si ha un training set composto da dati etichettati e un test set composto da dati non etichettati.Nel training set si hanno a disposizione un'insieme di coppie composte da dato ed etichetta che possono essere viste come input e output,mentre nel test set si hanno solamente i dati che possono essere visti come singoli input.
Il Machine Learning Supervisionato racchiude l'insieme di algoritmi che si addestrano con le coppie dato-classe.Nella fase di apprendimento questi algoritmi cercano di cogliere le relazioni che intercorrono tra queste coppie per poi stabilire quali siano le etichette dei dati presenti nel test set.
L'Authorship Attribution è un chiaro esempio di problema di cui si può occupare il Machine Learning Supervisionato.Infatti i dati di cui si conoscono le etichette possono essere visti come i testi per cui è stato assegnato l'autore cioè che fanno parte dell'insieme T' ed i dati non etichettati come i testi dell'insieme T''.
Nel problema in questione il training set è rappresentato dall'insieme di coppie D e il test set è rappresentato dall'insieme di testi T''.

4.3 Il Machine Learning Supervisionato

Il Machine Learning si suddivide a sua volta in due categorie in base al tipo di problema e al tipo di dato che si sta trattando e al tipo di output che l'algoritmo dovrà predire.

4.3.1 Regressione

Negli algoritmi di regressione l'output da predire è un valore numerico continuo.Più precisamente l'output di una predizione può assumere un valore reale all'interno di un intervallo.

Questi algoritmi sono applicati a problemi come la previsione della temperatura nei giorni a seguire, la stima del numero di contagiati in una pandemia o del prezzo delle azioni di una società.

4.3.2 Classificazione

Nella classificazione si vuole attribuire ad un dato un'etichetta appartenente ad un insieme finito di cui si è conosciuto il contenuto in fase di addestramento.Infatti quando si ha a che fare con problemi di questo tipo le etichette vengono anche chiamate classi.

Esempi di classificazione sono il riconoscimento di email spam,il Fake News Detection (riconoscimento di notizie false) o di contenuti malevoli all'interno di un blog.

Fra i problemi di classificazione,in particolare di Text Classificaton, si può considerare anche l'Authorship Attribution per come è stato definito.

4.3.3 Metodi di Classificazione

I più noti algoritmi di classificazione sono Naive Bayes,gli Alberi Decisionali,Logistic Regression,le SVM(Support Vector Machine) e molti altri.

Le tecnologie più recenti hanno favorito l'utilizzo di reti neurali per la classificazione perché effettuano in modo parallelizzato un'analisi più elaborata dei dati che consente in alcuni casi di individuare le relazioni più complesse che ci sono tra essi.

Preprocessing del testo

In questo capitolo si trattano alcune tra le varie tecniche di preprocessing dei dati di cui si fa uso nei sistemi di Authorship Attribution proposti in seguito.Queste tecniche sono proprie dei metodi utilizzati nel Natural Language Processing.
Il NLP(o Elaborazione del Linguaggio Naturale) è il processo che si occupa di trattare in maniera automatica il linguaggio naturale, cercando di fronteggiare i problemi che lo caratterizzano.
In virtù del fatto che è naturale, il linguaggio non presenta regolarità e strutture sintattiche facili da trattare in maniera automatica.Inoltre una macchina non estrae conoscenza da un testo allo stesso modo con cui lo fa un essere umano.
Pertanto il testo scritto, che in questo caso rappresenta i dati, va modificato(attività di preprocessing) in maniera tale da essere reso più facilmente processabile dal modello di apprendimento automatico.
Ogni tecnica di preprocessing consiste nel modificare i dati aggiungendo le informazioni utili ed eliminando quelle superflue.Lo scopo di queste modifiche è quello di favorire l'apprendimento del modello fornendogli dati di maggiore qualità.
In questo capitolo sono descritte alcune tecniche di preprocessing che verranno poi utilizzate dai sistemi di Authorship Attribution che si vogliono proporre.L'ordine con cui sono discusse non si rispecchia esattamente con quello con cui sono messe in atto.Non esiste un preprocessing migliore degli altri perché l'adeguatezza di alcune

tecniche rispetto ad altre dipende dal singolo task e da come sono fatti i dati a disposizione.
Infatti i sistemi di Authorship Attribution discussi in seguito si distingueranno tra di loro per come le scelte implementative sul preprocessing sui dati a disposizione e su altre fasi del processo.

5.1 Tokenizzazione

Un testo per essere processato ha bisogno di essere scomposto in più parti. Solitamente queste parti o tokens sono rappresentate dalle parole o dalle frasi in un testo.
All'atto pratico la tokenizzazione viene effettuata seguendo delle regole o tenendo conto di un dizionario che contiene al suo interno tokens esistenti.
Esempio di tokenizzazione per parole:
"He ate so much there was nothing left"
$\rightarrow$ ["He","ate","so","$much$","$there$","was","$nothing$","$left$"]

5.2 Data Cleaning

I dati digitalizzati e strutturati potrebbero provenire da diverse fonti come ad esempio corpora letterari,contenuti di pagine web,contenuti di social networks ed essendo in alcuni casi prodotti da esseri umani potrebbero contenere errori(simboli inseriti a caso come ad esempio virgolette che non racchiudiono nulla,spazi vuoti...),simboli non leggibili oppure simboli che non rappresentano informazioni importanti ai fini dell'apprendimento del modello.

5.2.1 Rimozione dei rumori

I rumori(noises) sono quei simboli presenti nel testo che non favoriscono in modo significativo l'apprendimento.

Con questa tecnica si usano delle liste di rumori noti e frequenti nei testi per pulirne il contenuto.

Esempio:

"<pre>My Bonnie lies over the ocean ... Oh, bring back my Bonnie to me.</pre>" →

"My Bonnie lies over the ocean Oh bring back my Bonnie to me"

5.2.2 Rimozione delle stopwords

Vengono rimosse da un testo le parole non particolarmente significative.Questa operazione va effettuata in maniera molto delicata poiché la rimozione di una parola di uso molto frequente (come un pronome personale) non sempre favorisce l'apprendimento automatico.

Esempio:

"This is a sample sentence showing off the stop words filtration" →

"This sample sentence showing stop words filtration"

5.2.3 Lowercase Conversion

Per le convenzioni della scrittura dopo alcuni tipi di punteggiatura le parole vengono scritte in maiuscolo.Nel caso in cui le parole rappresentino nomi comuni ha senso trasformarle tutte in minuscolo.Invece nel caso opposto trasformare un nome proprio in minuscolo potrebbe causare una perdita significativa di informazioni.

5.3 Lemmatizazione

Viene effettuata per portare le parole all'interno di un corpus nella forma base o lemma.La forma base di una parola è anch'essa una parola(di senso compiuto) e solitamente si trova all'interno dei dizionari.Essendo una tecnica di rimozione delle informazioni potrebbe sfavorire l'apprendimento qualora queste non fossero superflue.
Esempio:

$am,\ are,\ is \rightarrow be$

$mice,mousey,mouser \rightarrow mouse$

5.4 Stemming

Viene effettuata per trasformare le parole all'interno del corpus nelle loro radici.Come la Lemmatizzazione questa tecnica di preprocessing può portare alla perdita di informazioni.Inoltre le parole a cui porta lo stemming non sempre sono di senso compiuto.
Esempi:

$studies,\ study,\ studied \rightarrow stud$

$cats,catting \rightarrow cat$

Talvolta Lemmattizazione e Stemming coincidono(come in quest'ultimo esempio).

5.5 Data Annotation

L'insieme di tecniche di annotazione dei dati hanno come utilità quella di aggiungere informazioni al testo.Esistono molti modi di aggiungere informazioni ai dati a disposizione che dipendono fondamentalmente da cosa essi rappresentano(testi,immagini...).
L'annotazione dei testi favorisce l'apprendimento risolvendo problemi tipici del linguaggio naturale come la polisemia.

5.5.1 Pos-Tagging

Il Pos-Tagging(Part-Of-Speech tagging) effettua una pseudo analisi grammaticale di un testo e quindi attribuisce ad ogni parola all'interno di esso un tag(etichetta) che ne identifica il ruolo all'interno del discorso.

Per effettuare il Pos-Tagging si utilizzano algoritmi che possono basarsi su tecniche stocastiche oppure metodi di Machine Learning(quindi modelli preaddestrati).

Esempio:

"John likes the blue house at the end of the street"

('John', 'NNP'), ('likes', 'VBZ'), ('the', 'DT'), ('blue', 'JJ'), ('house', 'NN'),

('at', 'IN'), ('the', 'DT'), ('end', 'NN'), ('of', 'IN'), ('the', 'DT'), ('street', 'NN')

dove: NNP="nome proprio",VBZ="verbo,3pers,sing",DT="articolo determinativo",

JJ="aggettivo",NN="nome comune,sing",IN="preposizione"

5.5.2 Named-Entity-Recognition (NER)

Attribuire ad una parola o ad un immagine un'entità aiuta il modello di apprendimento a risolvere i problemi di ambiguità distinguendo un contesto rispetto all'altro.Con la NER ad ogni token viene assegnata un'etichetta che ne rappresenta la categoria.

Sostanzialmente questa tecnica di Data-Annotation effettua una classificazione poiché attribuisce ad un token un'etichetta facente parte di un insieme di categorie.

Esempio:

"Apple is looking at buying U.K. startup for $1 billion"

("Apple","ORG"), ("U.K.","GPE"), ("$1 billion","MONEY")

dove: ORG="organizzazione",GPE="entità geopolitica",MONEY="quantità di denaro"

5.6 Bilanciamento dei dati

Non è esattamente una tecnica di preprocessing anche se si rivela molto utile in senso pratico.Può essere utilizzata quando si ha a che fare con un insieme di dati non bilanciato.Ad esempio nel caso dell'Authorship Attribution si potrebbero avere dei training set contenenti delle quantità di testi per autore non equilibrate.
Uno dei modi di far fronte a questa circostanza potrebbe essere quello di eseguire Data Augmentation inserendo arbitrariamente nel training set una serie di dati reali o fittizi in maniera da renderlo bilanciato(oversampling).Solitamente i dati inseriti sono copie leggermente modificate di quelli già presenti.
La soluzione opposta invece consiste nell'eliminare i dati che rendono sbilanciato il dataset(undersampling).

Tecniche di rappresentazione dati

Come per l'estrazione di conoscenza, il modo in cui le macchine visualizzano i dati è diverso dal modo in cui lo fa l'uomo.Di conseguenza è necessario fornire una loro rappresentazione che li renda comprensibili ai modelli.

Affinché possano essere visualizzati e compresi dalle macchine, ai dati viene conferito un formato numerico.

Fondamentalmente per rappresentare un dato si effettua una codifica dei suoi contenuti che porta ad un vettore di numeri.Tale processo è chiamato impropriamente Vettorizzazione (o Vettorializzazione).

Dunque le tecniche di Vettorizzazione sono operazioni volte a trasformare i dati dal loro tipo originale(stringhe,pixel...) ad un tipo numerico che può quindi essere elaborato dai modelli di Machine Learning.In questa fase può verificarsi una perdita di informazioni in quantità significative.

Nel problema che si vuole affrontare con questo lavoro, i dati sono rappresentati da discorsi in forma testuale che vengono trasformati in liste di parole(o token) in fase di Preprocessing.Oltre alle informazioni che danno le singole parole, bisogna tener presente anche della loro posizione all'interno delle frasi,del loro numero,delle loro occorrenze in un singolo dato e nell'intero dataset.

Qui vengono discusse e confrontate le tecniche di Vettorizzazione che sono utilizzate nei sistemi di Authorship Attribution descritti in seguito.

6.1 Bag-of-words

Uno dei metodi più semplici di rappresentare dati testuali sui quali è stata fatta una "tokenizzazione per parole" è il BOW (Bag-of-words trad. "Borsa di Parole").
Questo metodo vede ogni singolo dato come una cesta di parole(bag of words) senza dare importanza all'ordine con cui queste compaiono.
Per prima cosa il modello individua le parole diverse all'interno di tutti i dati di cui si vuole fornire una rappresentazione numerica.Viene quindi individuato l'insieme $F = \{f_1, f_2, \ldots, f_n\}$ di parole dette "features" privo di ripetizioni.
Sia W la cardinalità dell'insieme F ossia il numero di parole diverse(features) trovate all'interno delle frasi.
Il modello procederà ad assegnare univocamente ad ogni feature $f_{i=0}^{n}$ un intero $v \in [0, 1, ..., W-1]$.Ciò può essere visto più formalmente come l'applicazione della funzione iniettiva w: $\mathbf{F} \longrightarrow [\mathbf{0}, \mathbf{1}, \mathbf{2}...\mathbf{W-1}]$,il cui codominio è un sottinsieme dei numeri naturali.
Il modello verifica le occorrenze di ognuna delle features all'interno di ciascun dato.
Sia δ un dato(quindi una sequenza di parole),sia f una feature e sia Δ il risultato di BOW applicato a δ (quindi Δ è un vettore contenente numeri di dimensione W-1).
Per il funzionamento di BOW si ha che, se la feature f compare all'interno di δ un numero pari a k volte ,con $k \in \mathbb{N}$, allora in $\Delta[w(f)]$ verrà scritto k,dove $w(f)$ è il valore assegnato a f dalla funzione v.
Si consideri per esempio l'insieme di dati:

δ1: *"There used to be Stone Age"*

δ2: *"There used to be Bronze Age"*

δ3: *"There used to be Iron Age"*

$\delta 4$: *"There was Age of Revolution"*

$\delta 5$: *"Now it is Digital Age"*

L'insieme di features F è :

$F = \{$"$There$","was","to","be","$used$","$Stone$","$Bronze$","$Iron$","$Revolution$",

"$Digital$","Age","of","Now","it","is"$\}$

L'insieme delle assegnazioni fatte dalla funzione w é :

$\{$"age" : 0,"be" : 1,"$bronze$" : 2,"$digital$" : 3,"$iron$" : 4,

"is" : 5,"it" : 6,"now" : 7,"of" : 8,"$revolution$" : 9,"$stone$" : 10,

"$there$" : 11,"to" : 12,"$used$" : 13,"was" : 14$\}$

L'insieme delle assegnazioni fatto da w è anche detto vocabolario.

Il risultato della vettorizzazione con BOW applicato ai dati $\delta 1, \delta 2, \delta 3, \delta 4$ è:

$\Delta 1 :\ [1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0]$

$\Delta 2 :\ [1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0]$

$\Delta 3 :\ [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1]$

$\Delta 4 :\ [1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]$

I risultati di BOW non sono altro che vettori o array di numeri che mostrano le occorrenze di ogni feature in ogni dato.

Si osserva infatti che in corrispondenza di determinati indici dei vettori ci sono le occorrenze di quella feature nel dato che rappresentano.

Il Bag-Of-Words è una tecnica molto semplice e facile da mettere in pratica.

La sua semplicità però viene pagata con la perdita di informazioni molto importanti tipiche dei dati testuali come l'ordine delle parole che non viene codificato.

Inoltre considerando ogni dato come una cesta di parole, questo metodo attribuisce ad ognuna di esse la stessa importanza.

6.2 Term frequency-inverse document frequency based

Oltre alla perdità dell'informazione relativa all'ordine ,il primo metodo di vettorizzazione descritto non riporta quali sono le parole che caratterizzano un testo.Infatti all'interno di un discorso esiste una gerarchia tra le parole più significative rispetto all'argomento che si sta trattando.

Una vettorizzazione che tiene conto dell'importanza delle parole all'interno di un insieme di dati testuali può essere quella basata su quanto riporta il calcolo della funzione Tf-idf ("Term frequency-inverse document frequency") applicata all'insieme dei dati a disposizione.

Siccome questa tecnica di vettorizzazione si applica esclusivamente ai testi, in seguito si parlerà di "termini"(anziché parole),di "documenti"(anziché dati) e di "collezione di documenti"(anziché insieme di dati).

Tf-idf[16] è una funzione di peso usata in ambito statistico per valutare la specificità di un termine all'interno di un documento facente parte di una collezione.

Questa funzione attribuisce ad ogni termine un valore ricavato dal calcolo di due metriche:

- TF(Term Frequency trad. Frequenza del termine)
- IDF(Inverse Document Frequency trad. Frequenza nel documento inversa)

Sia $T = \{t_1, t_2, \ldots, t_m\}$ l'insieme di tutti i termini usati nella collezione di documenti $D = \{d_1, d_2, \ldots, d_n\}$ dove ogni documento $d_i = [..., t^i, ...]$ per $i = 1....n$ e $\in T$ è una combinazione di termini $t^i \in T$.

Dato un termine t^i corrispondente a un termine del documento $d_i \in D$:

- TF calcola la frequenza del termine nel suo documento
- IDF calcola l'inverso della frequenza del termine nella collezione dei documenti

Il calcolo di quanto esprimono TF ed IDF è rappresentato con diverse formule.In questo caso sono riportate le formule[18] con cui il metodo è implementato nella realizzazione del sistema.

$TF(t^i, d_i) = \frac{\#t^i}{|d_i|}$, dove:

$\#t^i$ è il numero di occorrenze del termine t^i nel documento d_i

$|d_i|$ è il numero di termini presenti nel documento d_i

$IDF(t^i, d_i, D) = \log_e \left[\frac{n}{DF(t^i, D)}\right] + 1$, dove:

n è il numero di documenti in D

$DF(t^i)$ è il numero di documenti in cui t^i occorre

Tf-idf calcola per ogni termine t^i per $i = 1, ..., n$ presente in un documento d_i per $i = 1...n$:

$TF - IDF(t^i, d_i, D) = TF(t^i, d_i) * IDF(t^i, d_i, D)$

Dunque l'idea di questo metodo è quella di valutare l'importanza di un termine pesando la sua frequenza nel documento e con la sua frequenza nell'intera collezione. Dato un termine molto frequente in un documento, questo è considerato importante se si presenta raramente nel resto della collezione.Ciò si rispecchia con la realtà in quanto, è molto probabile che, un termine molto utilizzato sia in un documento che negli altri, non sia particolarmente specifco.

Nella vettorizzazione basata su questo metodo, per ogni documento si crea un vettore, le cui componenti rappresentano i valori ritornati dalla funzione Tf-idf applicata a ogni termine.

Esempio:

Sia $D = [$*"It is going to rain today"*, *"Today I am going outside"*, *"I am going to watch tv"*$]$ una collezione di documenti che in seguito verranno indicati rispettivamente con d_1,d_2,d_3 e sia $T = \{$"am","$going$","i","is","it","$outside$","$rain$","to","$today$",tv,"$watch$"$\}$ l'insieme dei termini presenti in D.

Occorrenze di ogni termine nella collezione:

termine	occorenze in D
am	2
going	3
i	2
is	1
it	1
outside	1
rain	1
to	2
today	2
tv	1
watch	1

Calcolo di Tf per ogni termine in ogni documento:

termine/documento	TF in d_1	TF in d_2	TF in d_3
am	0	0.2	0.16
going	0.16	0.2	0.16
i	0	0.2	0.16
is	0.16	0	0
it	0.16	0	0
outside	0	0.2	0
rain	0.16	0	0
to	0.16	0	0.16
today	0.16	0.2	0
tv	0	0	0.16
watch	0	0	0.16

Calcolo di Idf per ogni termine nella collezione:

termine/documento	IDF in D
am	1.4
going	1
i	1.4
is	2
it	2
outside	2
rain	2
to	1.4
today	1.4
tv	2
watch	2

Calcolo di Tf-idf per ogni documento:

termine/documento	TF-IDF in d_1	TF-IDF in d_2	TF-IDF in d_3
am	0	0.28	0.22
going	0.16	0.2	0.16
i	0	0.28	0.22
is	0.32	0	0
it	0.32	0	0
outside	0	0.4	0
rain	0.32	0	0
to	0.22	0	0.22
today	0.22	0.28	0
tv	0	0	0.32
watch	0	0	0.32

Si consideri il vocabolario che assegna ad ogni termine un indice all'interno del vettore con la funzione iniettiva w(descritta in cap.6,sec.6.1) avente in questo caso come dominio l'insieme T e come codominio $W = [0, 1, 2...|T| - 1]$ tale che $W \subset \mathbb{N}$:

$V = \{"am" : 0, "going" : 1, "i" : 2, "is" : 3, "it" : 4, "outside" : 5,$

$"rain" : 6, "to" : 7, "today" : 8, "tv" : 9, "watch" : 10\}$

La rappresentazione in forma numerica di d_1,d_2,d_3 è :

d_1 : $[0, 0.16, 0, 0.32, 0.32, 0, 0.32, 0.22, 0.22, 0, 0]$

d_2 : $[0.28, 0.2, 0.28, 0, 0, 0.4, 0, 0, 0.28, 0, 0]$

d_3 : $[0.22, 0.16, 0.22, 0, 0, 0, 0, 0.22, 0, 0.32, 0.32]$

Si osserva che a termini come "going", nei documenti in cui sono presenti ,è stato attribuito un valore più basso essendo molto frequenti nell'intera collezione.Invece a termini come "rain", nei documenti in cui sono presenti ,è stato assegnato un valore più alto essendo poco frequenti nel resto della collezione.
Fra i termini a cui è stato assegnato un valore più alto c'è "it",un termine di uso molto comune, presente nel primo documento.Essendo quest'ultimo un pronome personale, ci si aspetta che gli venga attribuita una bassa rilevanza.Facendo riferimento a quest'ultimo esempio, si può affermare che è conveniente eliminare(in fase di Preprocessing) dai documenti i termini di uso comune(stopwords come preposizioni,pronomi,congiunzioni...) affinché la vettorizzazione basata su Tf-idf fornisca una migliore rappresentazione dei dati.

6.3 Transformer based

La vettorizzazione basata su Tf-idf fornisce in molti casi una codifica delle informazioni contenute nei dati migliore di BOW,ma come quest'ultima non tiene conto dell'ordine delle parole.
Altre informazioni nascoste all'interno di un testo che non sono facili da codificare in formato numerico sono i legami tra le parole e tra le frasi.Accade nel linguaggio

naturale che, all'interno di molti tipi di testo,due o più parole distanti, in realtà siano semanticamente collegate.Lo stesso fenomeno si verifica per le frasi nel caso in cui un autore faccia utilizzo di digressioni.
Altre contingenze tipiche del linguaggio naturale che rendono difficile la codifica di tutte informazioni contenute in un testo sono :

- Parti del discorso mancanti che vengono sottintese(come il soggetto)
- Parole polisemiche
- Espressioni idiomatiche
- Figure retoriche
- Abbreviazioni
- Eccezioni di vario genere

Un metodo recente di trasformare i dati dal formato originale a quello numerico consiste nell'utilizzo di alcuni modelli di Deep Learning basati su Transformers.
Le tecniche di vettorizzazione descritte finora seguono nell'esecuzione una serie di operazioni prefissate.
I ricercatori di Google hanno proposto in un articolo [15] l'utilizzo di BERT (Bidirectional Encoder Representations from Transformers),un modello(chiamato anche "Encoder") basato su Transformers per la codifica delle informazioni contenute in un testo.
I Transformers sono una categoria di modelli di Deep Learning introdotti in "Attention Is All You Need"[19] caratterizzati dal meccanismo della "self-attention".
Gli Encoder basati su Transformers come BERT sono utilizzati per il "transfer-learning",un processo che consiste nell'implementare all'interno di un sistema, un modello preaddestrato, fornedogli dati relativi al task in questione.

Gli Encoder che vengono utilizzati nei sistemi di Authorship Attribution proposti in seguito, sono accomunati dal fatto che sfruttano i vantaggi delle architetture dei transformers e dal fatto che possiedono conoscenza acquisita grazie ad un loro pre-training su grandi quantità di dati generici.
Dunque alcuni sistemi basati su deep learning che svolgono task di classificazione di testi,implementano gli encoder e li addestrano su dei dati specifici.Quest'ultima procedura è anche chiamata "fine-tuning"(espressione che indica l'utilizzo di transfer-learning per un determinato task).
Esempi di encoder basati su transformers son:

- BERT(Bidirectional Encoder Representations from Transformers)
- XLNET [22]
- ELECTRA [12]
- ERNIE(Enhanced Language Representation with Informative Entities) [21]

In particolare BERT è un modello in grado di leggere i dati secondo qualsiasi ordine a differenza dei modelli tradizionali che lo fanno in maniera sequenziale("da sinistra a destra").Ciò consente di capire i contesti delle parole all'interno di un testo e di rimediare a problemi del linguaggio naturale come la digressione.
A favorire BERT nel processo di encoding delle informazioni è la conoscenza sul linguaggio naturale che già possiede grazie al preaddestramento svolto sul dataset di Wikipedia in inglese e su "Brown Corpus" [14].
I tasks per cui BERT è stato (pre)addestrato sono :

- Masked Language Modeling : data una frase in cui è stata nascosta una parola("mascherata"), questo task consiste nel predire quella parola a seconda del contesto(dato dalle

parole che la circondano).

Esempio:

INPUT:"I have watched this [MASK] and it was awesome."

OUTPUT:"I have watched this movie and it was awesome."←la parola predetta è "movie"

- Next Sentence Prediction : date due frasi il task consiste nel prevedere se la seconda è il seguito della prima.

  Esempio1:

  INPUT: A="Jim went to the store.",B="He bought a pair of shoes."

  OUTPUT: B segue A← istanza SÌ di NSP

  Esempio2:

  INPUT: A="Peter cooked a meal.",B="The garage was empty."

  OUTPUT: B NON segue A← istanza NO di NSP

Modelli di Deep Learning

In questo capitolo si trattano i modelli di apprendimento implementati nei sistemi di Machine Learning per l'Authorship Attribution discussi successivamente.
Una delle possibili implementazioni del Machine Learning è data dalle Reti Neurali.
In letteratura si parla di Deep Learning quando ci si riferisce a modelli di apprendimento che in qualche modo si ispirano alla struttura e vagamente al funzionamento del cervello biologico.
Questi modelli sono le Reti Neurali Artificiali che a loro volta si suddividono in diverse tipologie a seconda della loro struttura e del loro funzionamento.
Il motivo per cui esistono più tipi di reti neurali con caratteristiche diverse è legato alle categorie di problemi per cui queste si adattano.

7.1 Reti Neurali Feed Forward

Per i sistemi di Classificazione si fa spesso uso di Feed-Forward Neural Networks(trad. Reti Neurali Feed Forward;in seguito verranno chiamate FFNN).
Le FFNN sono la più semplice tipologia di reti neurali e costituiscono una base per la realizzazione di modelli più complessi.
In seguito vengono descritte in maniera molto superficiale la struttura e il funzionamento che le caratterizzano.

7.1.1 Struttura

Dal punto di vista struttrale una FFNN può essere vista come un grafo diretto,aciclico e pesato $G = (V, E)$ dove l'insieme dei nodi $V = \{n_1..., n_n\}$ rappresenta l'insieme dei neuroni e l'insieme degli archi $E = \{c_1..., c_m\}$ rappresenta le connessioni tra di essi.Inoltre c'è una funzione $w\colon \mathbf{E} \longrightarrow \mathbb{R}$ che inizialmente assegna ad ogni connessione un peso.

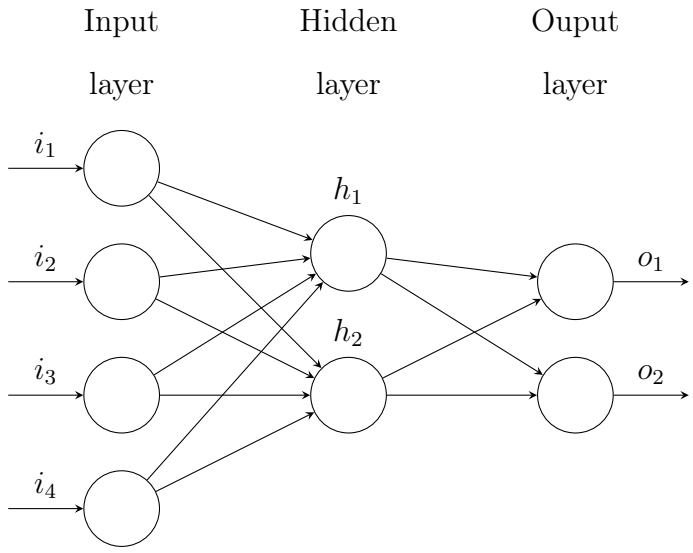
I neuroni sono partizionati in una serie di strati o layer $D_1, ..., D_k$ dove $k >= 2$ tali che $\bigcup_{i=1}^{k} D_i = V$ e $\forall x, y \ \in D_i, \ \nexists e \in E$ tale che $e = (x, y)$. Letteralmente i layer sono dei sottinsiemi di neuroni non collegati tra di loro.

D_1 è definito "input layer"(strato di input) e D_k è definito "output layer"(strato di output).Se esistono $D_2, D_3, ..., D_{k-1}$ sono detti hidden layers o strati nascosti.

Inoltre ogni coppia di layers adiacenti forma un grafo bipartito.Cioè dati due layers adiacenti (D_j, D_{j+1}) per $j = 1, ..., k-1$ vale che per ogni nodo $n' \in D_j$ e per ogni nodo $n'' \in D_{j+1}$, $\exists e \in E$ tale che $e = (n', n'')$.Letteralmente per ogni nodo in D_j e per ogni nodo in D_{j+1} esiste un arco diretto che li collega.Si sottolinea che la direzione degli archi è sempre rivolta verso destra cioè da D_j a D_{j+1}.

Una FFNN si distingue strutturalmente dalle dalle altre reti neurali per l'assenza di connessioni(archi) tra layers non adiacenti o non consecutivi.

Esempio:



FFNN in cui:

- l'insieme dei neuroni è : $V = \{i_1, i_2, i_3, i_4, h_1, h_2, o_1, o_2\}$.

- I layers sono : D_1, D_2, D_3.

- Input layer : $D_1 = \{i_1, i_2, i_3, i_4\}$.

- Hidden layer : $D_2 = \{h_1, h_2\}$.

- Output layer : $D_3 = \{o_1, o_2\}$.

  Oss: non ci sono connessioni tra layers non adiacenti

  Oss: non ci sono connessioni tra layers non consecutivi

NOTA : per ora si trascura il significato delle frecce entranti nei neuroni dell'input layer e delle frecce uscenti dall'ouput layer.Rispetto alla semplice topologia di una rete neurale si può immaginare che i nodi dell'input layer siano "nodi sorgente" e i nodi dell'output layer siano "nodi pozzo".

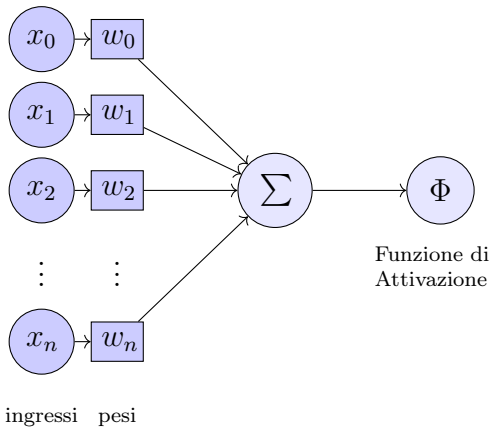
7.1.2 Il percettrone

Dal punto di vista del funzionamento una rete neurale può essere vista come un insieme di percettroni interconnessi.

Il modello più semplice di Deep Learning è il Percettrone introdotto da Frank Rosenblatt nel 1958.

Un percettrone può essere descritto come una rete neurale a singolo neurone,dove Input Layer e Output Layer coincidono.Siccome una FFNN è costituita da un insieme di percettroni collegati tra di loro viene anche chiamata "Multilayer Perceptron"(MLP cioè rete di percettroni a più strati).

Un percettrone è caratterizzato da avere un neurone che accetta dei valori in ingresso,un insieme di connessioni pesate in uscita,un nucleo di calcolo Σ e una funzione di attivazione Φ.



Il percettrone riceve un insieme di valori in ingresso(nell'esempio precedente le frecce entranti nell'input layer rappresentano dei valori in ingresso).

Siano $x_0, x_1, x_2..., x_n$ con $x_i \in \mathbb{R}$ per $i = 0, 1, 2, ..., n$ dei valori che rappresentano i dati in input e siano $w_0, w_1, w_2..., w_n$ i pesi relativi ad ogni dato di input.

Allora il nucleo di calcolo Σ effettua la somma di ogni dato x_i moltiplicato per il rispettivo peso w_i per $i = 0, 1, 2, ..., n$.

Formalmente il nucleo effettua la sommatoria :

$$\sum_{i=0}^{n} x_i * w_i$$

Il risultato di questa sommatoria viene passato in input alla funzione di attivazione(scelta in fase di implementazione) Φ che calcola l'output del percettrone.
In una rete a più strati, l'output di un percettrone può influenzare sull'input di un altro percettrone qualora questi fossero collegati.

7.1.3 Addestramento

Come ogni modello di Machine Learning una FFNN attraversa due fasi : training e test.
Nella fase di training(o addestramento) la rete prende in ingresso tramite il suo input layer un insieme di dati in formato numerico,il training set.È importante ricordare che all'interno di un training set relativo un task di apprendimento supervisionato per ogni singolo dato c'è anche la sua etichetta o classe.Pertanto ogni percettrone di cui è composto l'input layer fornirà in qualche modo il risultato della propria funzione di attivazione come input a ogni percettrone dello strato successivo.Infine nell'output layer ci finiscono i risultati calcolati dai percettroni più interni che dipendono dai dati(che sono valori numerici) e dai pesi che li collegano.
L'output elaborato o predetto dalla rete viene confrontato con quello desiderato e si stima una *Loss Function*.
La *Loss Function*(trad: funzione di perdita) si usa nell'apprendimento supervisionato, per calcolare quanto un valore predetto si discosta da quello reale, cioè l'errore commesso nell'effettuare la predizione.In base ai valori della Loss Function calcolata sulla predizione di un insieme di dati, la rete valuta se cambiare i pesi delle connessioni tra i vari percettroni.

L'obiettivo è quello di modificare i pesi(che inizialmente vengono scelti casualmente) in maniera tale da minimizzare il valore della Loss Function.Per effettuare questo aggiustamento in maniera opportuna si usano degli algoritmi di ottimizzazione specifici. Gli algoritmi che si occupano di minimizzare la Loss Function si basano sulla tecnica di Discesa del Gradiente.
Dunque ogni volta che viene calcolato un output dalla rete vengono aggiornati i pesi di tutte le connessioni(Backpropagation) in base agli errori commessi.

7.1.4 Test e Valutazione

Nella fase di test la rete prende in input un insieme di dati in formato numerico detto test set.A differenza del training set, per ogni singolo dato del test non c'è più la classe di appartenenza.In questa fase è la rete ad occuparsi di predire la classe di ogni dato fissati i pesi delle sue connessioni.
Le performances del modello possono essere valutate con diverse metriche che si basano fondamentalmente sulla correttezza delle predizioni effettuate.
Una tra le più semplici è l'accuracy che misura il numero di predizioni effettuate correttamente sul numero di predizioni totali.

$$Accuracy = \frac{Numero\ di\ predizioni\ corrette}{Numero\ di\ predizioni\ effettuate}$$

Esistono altre metriche per la valutazione di modelli che effettuano classificazione che ne descrivono le prestazioni tenendo conto di altri fattori.Ad esempio *F1 score* tiene conto della distribuzione dei dati del training set tra le varie classi.

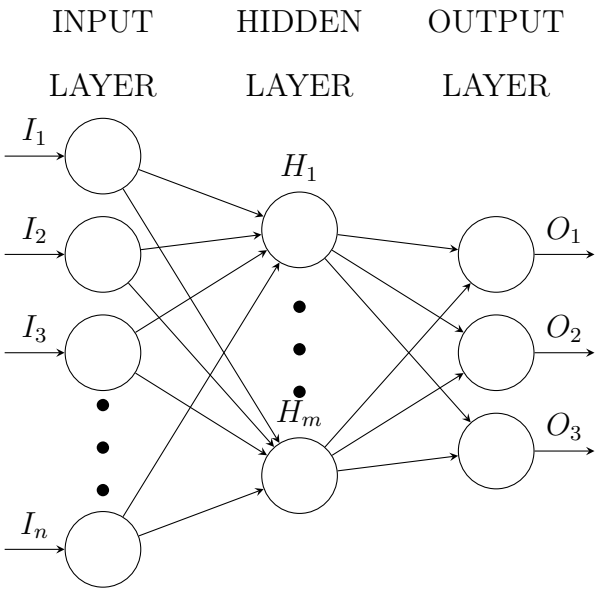
Applicazioni

Un sistema automatico e intelligente di Authorship Attribution può tornare utile in una notevole varietà di ambiti.
In seguito viene presentato e discusso un caso d'uso di più sistemi di Authorship Attribution che si basano sulle tecniche descritte precedentemente.
Il dataset utilizzato per effettuare training e test è stato costruito a partire da contenuti preesistenti trovati nel Web.Il training set con cui vengono addestrati i modelli costituisce il 70% dell'intero dataset mentre il test set il restante 30%.

8.1 Sistemi di Authorship Attribution

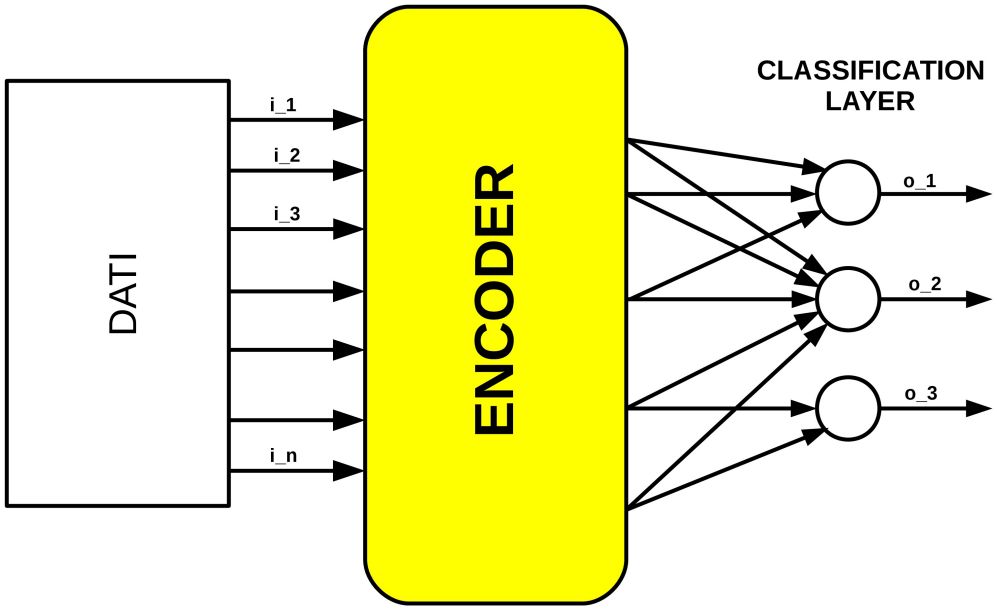
I sistemi di Authorship Attribution realizzati si suddividono in due categorie in base ai modelli di apprendimento che sono stati implementati e al tipo di preprocessing e vettorizzazione effettuati.In particolare nella prima categoria di sistemi sono state utilizzate tecniche di preprocessing e vettorizzazione tradizionali insieme alle FFNN a tre strati(quindi con un solo hidden layer).Nella seconda categoria di sistemi invece, sono stati utilizzati gli Encoders basati su transformers in combinazione con un singolo strato di percettroni.

Un esempio di modello utilizzato per un task in cui le classi da predire sono 3 è il seguente:



In questa rete neurale di tipo Feed Forward, l'input layer è formato da un numero di neuroni corrispondente al numero caratteristiche(o features) risultanti dalla rappresentazione numerica di un singolo dato.Mentre, come si può notare, l'output layer è formato da 3 neuroni in quanto il numero di classi che si può predire è 3.

La seconda categoria di sistemi prevede che l'output di un Encoder venga preso in input(o decodificato) da un singolo layer che funge da classificatore.
La struttura descritta finora è la seguente:



8.2 Valutazione dei modelli

Essendo il dataset bilanciato la metrica con la quale sono stati valutati i modelli di apprendimento è l'accuracy.

$$Accuracy = \frac{Numero\ di\ predizioni\ corrette}{Numero\ di\ predizioni\ effettuate}$$

8.3 Dettagli implementativi

Per la realizzazione di questi sistemi, effettuata con il linguaggio python, si è fatto riferimento a diverse librerie tra cui le più importanti:

- Pytorch[1] : tutto ciò che concerne le reti neurali
- Pandas[2] : per le operazioni sui dataset
- Scikit-Learn[3] : per la vettorizzazione e per lo splitting dei dataset
- Numpy[4] : per le operazioni su dati vettorizzati
- NLTK[5] : per il preprocessing

Per il primo insieme di sistemi sono state scelte come *loss function* Cross-Entropy[6] ,come funzione di attivazione degli strati interni Relu[7],come funzione di attivazione per lo strato di output Softmax[8] e come algoritmo di aggiustamento dei pesi Stochastic Gradient Descent[9].
Invece per l'implementazione degli Encoders basati su Transformers all'interno dei sistemi di Authorship Attribution che ne prevedevano l'utilizzo si è fatto riferimento alla documentazione di Hugging Face[10].
I codici e il dataset dell'esperimento sono reperibili sul repository git:
`https://github.com/nexus126/authorship_attribution`

8.4 Forum Tematico

Per sperimentare il funzionamento di questi sistemi si è considerata una possibile applicazione che potrebbe rivelarsi una strategia utile per garantire la sicurezza di un paese.

Il task consiste nell'effettuare Authorship Attribution su messaggi scritti in lingua inglese(tranne piccole eccezioni) all'interno di un forum a tema religioso.

8.4.1 Dataset

Il dataset è stato creato a partire da un file messo a disposizione da una pagina Web [11].Questo file(`"Islamic_Network.txt"`) contiene dati relativi ad un forum religioso presente nel Dark Web, tra i quali ci sono messaggi,membri e gruppi di cui questi fanno parte.

Come riportato dagli autori della pagina, i partecipanti della discussione trattano diversi temi riguardanti l'Islam e alcuni di loro simpatizzano e supportano organizzazioni terroristiche.

Il contenuto del dataset creato a partire dal file è suddiviso nel seguente modo:

Partecipante	Numero di messaggi
AbuIlyaas	1436
Editor	1436
Helper	1436
Isam Rajab	1436
Khadijah UmmZakariyyah	1436
Prof	1436
Rose15	1436
caramel	1436
heartsofgreenbirds	1436
sasjamal	1436

NOTA : il dataset è stato costruito in maniera tale che fosse perfettamente bilanciato.

In seguito è mostrata una piccola porzione del dataset:

Testo	Autore
I didn't say you give a christmas present,...	Isam Rajab
Asalaam alaikum, I didn't mean not to be sill...	Khadijah UmmZakariyyah
I want to apologize to anyone whom I personal...	sasjamal
do you know where I could get them from sis fa...	heartsofgreenbirds

8.4.2 Risultati

I sistemi testati per questo task variano per le tecniche di Preprocessing,di Vettorizzazione e dei modelli di apprendimento usati.

Le accuracy medie(ottenute effettuando ogni esperimento per 3 volte) dei modelli di apprendimento sono misurate rispettivamente sui testi di 3 autori ,sui testi di 5 autori e sui testi di 10 autori.

Modelli di DL	Metodi	3Autori	5Autori	10Autori
FFNN con 1 Hidden Layer	BOW	84.1	73.2	58.3
	BOW+POS*	**84.9**	**84.5**	57.59
	TFIDF	83.7	84.2	57.8
	TFIDF+POS*	83.6	81.1	**58.9**
ENCODER + Classification Layer	BERT	62.1	47.6	32.7
	BERTmulti	40.2	29.1	23.3
	XLNET	45.7	31.7	19.9
	ELECTRA	61.4	45.7	32.1
	ERNIE	62.1	42.4	29.4

*prima del Pos-tagging è stata effettuata Lemmatizzazione

8.4.3 Osservazioni

Nei sistemi che implementano le FFNN con 1 hidden layer sono stati osservate le migliori performances in termini di accuracy.In particolare il sistema che effettua Pos-Tagging in fase di Preprocessing e Vettorizzazione Bag-Of-Words su un dataset di contenente i messaggi di 3 membri ottiene in fase di test l'84.9% di accuracy.Ciò significa che questo sistema riesce ad indovinare l'autore di un messaggio in quasi l'85% dei casi.

Il calcolo della funzione Tfidf per la rappresentazione dei dati in forma numerica non ha portato ad un miglioramento dell'accuracy così come il Pos-Tagging in fase di preprocessing .

I risultati dell'implementazione di Encoder basati su Tranformers non sono stati particolarmente soddisfacenti.

8.5 Conclusioni e Sviluppi Futuri

Grazie ai risultati dell'esperimento effettuato, si può affermare l'utilizzo di sistemi basati su Deep Learning può agevlare l'attività dell'Authorship Attribution.
Gli Encoders basati su Transformers hanno fornito risultati inferiori a quanto ci si aspettava nonostante questi portino a ottimi risultati in tasks di text classification.
Dunque sarà necessario individuare in maniera approfondita gli aspetti che hanno inciso maggiormente sulle performances riportate, in modo da poter valutare al meglio i sistemi progettati.
L'obbiettivo principale è quello di elaborare sistemi basati su Deep Learning che possano attuare al meglio l'attività dell'Authorship Attribution.
In particolare, visti i risultati poco soddfisfacenti negli esperimenti effettuati, si vuole approfondire il funzionamento degli Encoders implementati, per poter giungere a conclusioni interessanti e proporre nuovi sistemi che possano classificare i testi con più precisione.
Infine si potrebbero considerare tecniche di rappresentazione del testo più elaborate che non sono state usate finora,meglio note come Word Embedding , che portano alla codifica di una maggiore quantità di informazioni nascoste all'interno dei dati.

Bibliography

[1] https://pytorch.org/.

[2] https://pandas.pydata.org/.

[3] https://scikit-learn.org/stable/.

[4] https://numpy.org/.

[5] https://www.nltk.org/.

[6] https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html.

[7] httpshttps://pytorch.org/docs/stable/generated/torch.nn.ReLU.html.

[8] https://pytorch.org/docs/stable/generated/torch.nn.Softmax.html.

[9] https://pytorch.org/docs/stable/optim.html.

[10] https://huggingface.co/docs/transformers/index.

[11] https://www.azsecure-data.org/dark-web-forums.html.

[12] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555, 2020.

[13] Maël Fabien, Esau Villatoro-Tello, Petr Motlicek, and Shantipriya Parida. BertAA : BERT fine-tuning for authorship attribution. *Science Direct*, 2020.

[14] W. N. Francis and H. Kucera. Brown corpus manual, 1979. `http://icame.uib.no/brown/bcm.html`.

[15] Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. Bert: Pre-training of deep bidirectional transformers for language understanding. *Science Direct*, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers):4171–41861, 2019.

[16] Karen Spark Jones. A statistical interpretation of term specificity and its application in retrieval. *Science Direct*, Vol. 28 No. 1:11–21, 1972.

[17] Aleksandr Romanov, Anna Kurkutova, Alexander Shelupanov, and Anastasia Fedotova. Authorship identification of a russian-language text using support vector machine and deep neural networks. *Future Internet*, 2020.

[18] scikit-learn's developers. Tfidfvectorizer. `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html`.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 06 2017.

[20] Biveeken Vijayakumar and Muhammad Marwan Muhammad Fuad. A new method to identify short-text authors using combinations of machine learning and natural language processing techniques. *Science Direct*, 2019.

[21] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. 2019.

[22] Yiming Yang Jaime Carbonell Russ R. Salakhutdinov Quoc V. Le Zhilin Yang, Zihang Dai. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019.