# POLITECNICO DI TORINO

**Laurea Magistrale in Ingegneria Informatica**


**Tesi di Laurea Magistrale**

# Cross-Lingual Sentiment Analysis

**Relatori**

**Prof. Luca CAGLIERO**

**Prof. Paolo GARZA**

**Dott. Flavio GIOBERGIA**

**Candidato**

**Lorenzo MONTORO**

**Dicembre 2020**

# Sommario

I dati disponibili su internet, e soprattutto le informazioni in essi contenute, sono già numerosissimi ed in costante aumento. Questi dati, che possono trovarsi rappresentati in diversi formati, dai testi alle immagini e i video, sono una miniera di informazioni che ricoprono fondamentale importanza per le aziende di tutto il mondo. Si stima però che circa l'80% di questi dati non sia strutturato, cioè non abbia una rappresentazione ordinata (un database rappresenta dati strutturati per esempio), e siano solitamente testi, immagini, audio e video che necessiterebbero dell'intervento umano per poter essere compresi. La Sentiment Analysis è un'analisi dei testi che sfruttando algoritmi soliti del Natural Language Processing si proprone di estrapolare il sentiment (positivo, negativo o neutro) espresso nel testo analizzato (che può essere una recensione, un tweet, una notizia ecc.), automatizzando quindi l'operazione di estrazione di informazioni da dati non strutturati. In particolare in questa tesi si analizza il problema di espandere la Sentiment Analysis anche ad altre lingue oltre l'inglese, per il quale i modelli creati fino ad ora già ottengono ottimi risultati, mentre faticano maggiormente con altre lingue per cui i dati disponibili sono numericamente e qualitativamente inferiori. Proprio in questa prospettiva si parla di Transfer Learning, cioè la realizzazione di un modello basato sulla lingua inglese, capace poi di generalizzare su dati provenienti da altri contesti rispetto al contesto di appartenenza dei dati usati per il training del modello, e su altre lingue che rappresentano una sfida ancor più complessa.

# Ringraziamenti

Un sentito ringraziamento ai miei familiari, ai miei coinquilini Salvatore, Alessandro, Cosmo, e in particolare Luca, per il supporto nella stesura di questa tesi; a Flavio per la pazienza con cui mi ha seguito per tutto il tempo necessario, e i professori Cagliero e Garza per avermi permesso di lavorare a un progetto del loro gruppo di ricerca.

# Indice

# Elenco delle tabelle

# Elenco delle figure

# Acronimi

**IA**
Intelligenza Artificiale

**NLP**
Natural Language Processing

**SA**
Sentiment Analysis

**ML**
Machine Learning

**CNN**
Convolutional Neural Network

**DC-CNN**
Dual-Channel Convolutional Neural Network

**DDM**
Dong e De Melo

**KNN**
K-Nearest Neighbors

**GD**
Gradient Descent

**LR**
Linear Regressor

**RF**

Random Forest

**MLP**

Multilayer Perceptron

**LSTM**

Long Short-Term Memory

# Capitolo 1

# Introduzione

L'Analisi del Sentiment, o Sentiment Analysis, è un campo del Natural Language Processing (NLP) che si pone l'obiettivo di analizzare un testo per identificare e classificare l'informazione in esso contenuta e ricopre un sempre crescente interesse nella comunità scientifica e nelle aziende di tutto il mondo. Con una ingente quantità di dati disponibili sul web, la possibilità di estrarre informazioni dai testi presenti in siti internet, quali siti di recensioni, forum, blog e social media, è di fondamentale importanza ad esempio per poter valutare in modo automatizzato l'opinione delle persone riguardo a un determinato marchio (brand perception), un prodotto, un servizio, il consenso politico e qualsiasi altro argomento su cui una persona possa esprimere una opinione. In generale la Sentiment Analysis estrae da un testo la sua polarità (opinione positiva, neutra o negativa), ma può aggiungere anche altre informazioni come l'oggetto di cui si parla e la persona che esprime l'opinione.

Si stima che circa l'80-90% dei dati reperibili in rete non sia strutturato, cioè non inseribile in strutture ben definite come ad esempio le tabelle di un database. Questi dati non strutturati sono testi (email, documenti, articoli, social media, chat ecc.), immagini, video, audio, che come tali non hanno una struttura ordinata che li organizzi e per cui sarebbe necessario l'intervento di una persona per estrapolare da essi le informazioni essenziali e inserire queste informazioni in strutture di facile interpretazione non solo per le persone, ma anche e soprattutto per software di analisi dei dati. Questa operazione di analisi e organizzazione dovrebbe quindi essere realizzata da una persona, rendendo l'operazione lenta e poco adatta a mantenere il passo con cui i dati vengono generati da centinaia di milioni di utenti collegati alla rete internet. Per questo motivo modelli di Intelligenza Artificiale (IA) che sappiano estrapolare autonomamente informazioni da dati non strutturati sono sempre più importanti poiché automatizzano i processi aziendali risparmiando ore di elaborazione manuale dei dati e rendendo i team più efficienti.

I modelli che vengono realizzati per la Sentiment Analysis, come detto, sono

solitamente usati per riconoscere la polarità del testo, ma possono essere usati anche per estrarre altre informazioni, come ad esempio le emozioni (rabbia, felicità, tristezza ecc.) o le intenzioni di chi ha scritto (interessato, non interessato ecc.).

Tra le specializzazioni più usate in ambito di Sentiment Analysis troviamo la Fine-grained Sentiment Analysis, l'Emotion Detection, la Aspect-based Sentiment Analysis e la Cross-lingual Sentiment Analysis.

Con Fine-grained Sentiment Analysis si intende una analisi del sentiment realizzata con un modello capace di riconoscere più livelli di polarità dei testi in esame, ad esempio espandendo le categorie riconosciute in molto positivo, positivo, neutro, negativo, molto negativo, cercando di interpretare il testo con la stessa scala di valori delle cinque stelle spesso utilizzata per le recensioni.

Emotion Detection, come è facile intuire dal nome, è il riconoscimento automatizzato delle emozioni espresse in un testo, mentre la Aspect-based Sentiment Analysis è una analisi capace non solo di riconoscere la polarità, ma anche gli aspetti o le funzionalità alle quali il sentiment si riferisce.

Infine per Cross-lingual Sentiment Analysis si intende una analisi che riconosca la polarità in testi scritti in diverse lingue. Proprio quest'ultimo caso è quello che verrà approfondito in questa tesi, dato che i risultati della Sentiment Analysis sulla lingua inglese sono già ottimi, mentre si fatica maggiormente con altre lingue per via della minore quantità e qualità dei dati disponibili.

Alcuni dei vantaggi della Sentiment Analysis sono:

- Scalabilità
- Possibilità di analisi in tempo reale
- Uso di criteri coerenti

Con Scalabilità si intende la possibilità offerta dalla Sentiment Analysis di elaborare dati su larga scala in modo efficiente ed economico. Un modello realizzato per la Sentiment Analysis è inoltre capace di analizzare un flusso di dati praticamente in tempo reale, dando la possibilità ad una azienda di prendere decisioni in maniera repentina. Infine i metodi utilizzati per la Sentiment Analysis permettono di eliminare la componente soggettiva nella valutazione delle informazioni. Infatti, c'è la possibilità che in alcuni casi persone diverse possano interpretare il sentiment di uno stesso testo in modo diverso, a seconda delle loro personali esperienze. L'uso di intelligenze artificiali permette invece di affidarsi a criteri di valutazione chiari e fissi, che, anche se non infallibili, permettono comunque di ridurre gli errori e migliorare la coerenza dei dati raccolti.

Per la Sentiment Analysis vengono utilizzati metodi e algoritmi derivati dal Natural Language Processing, che possono essere suddivisi in tre principali tipologie:

- Rule-based

- Automatico
- Ibrido

Per Rule-based si intendono sistemi che utilizzano un insieme di regole realizzate a mano per identificare il sentiment di un testo. Un esempio molto semplice di questa tipologia di algoritmi è creare una lista di parole considerate positive e una lista di parole considerate negative, prendere il testo in esame e contare quante parole presenti nel testo appartengono alla lista positiva e quante invece a quella negativa. Il sentiment assegnato al testo sarà deciso dal gruppo più numeroso: se nel testo risulteranno essere presenti più parole positive, il testo sarà etichettato con un sentiment positivo, altrimenti sarà etichettato con un sentiment negativo. Questo tipo di approccio manuale è molto basilare dato che non tiene in considerazione la sequenza delle parole e non è in grado di adattarsi autonomamente ai diversi ambiti a cui il testo potrebbe appartenere (una stessa parola usata in due contesti diversi potrebbe avere un significato, in termini di sentiment, diverso). Ovviamente più casi si riescono a prevedere con nuove regole, maggiori sono le probabilità che il sistema classifichi correttamente i dati, a costo però di una sempre crescente complessità e di possibili incongruenze con i risultati ottenuti in precedenza.

L'approccio Automatico prevede invece l'uso di tecniche di Machine Learning (ML), evitando quindi regole decise manualmente dallo sviluppatore. In questo caso la Sentiment Analysis è considerata un problema di classificazione, dove il classificatore (ad esempio una rete neurale) riceve un testo e ne definisce la categoria di appartenenza (per esempio positivo, negativo, neutro). Le tecniche di ML prevedono due processi che riguardano il modello scelto per la classificazione: Training e Prediction. Durante il processo di Training, il modello impara ad associare un input (in questo caso un testo) al suo corrispondente output (ad esempio alle categorie positivo, negativo e neutro), utilizzando degli esempi da cui imparare, cioè una lista di testi già classificati. Il processo di Prediction è invece la fase di effettivo utilizzo del modello generato, in cui testi non classificati vengono forniti al modello che si occupa di prevedere la categoria di appartenenza fornendo quindi i risultati a cui si è interessati. I testi devono essere trasformati prima di poter essere utilizzati, solitamente con un processo di vettorizzazione come ad esempio bag-of-words, oppure trasformando ogni parola del testo in un vettore di numeri che le rappresenti e che permetta a due parole con un significato simile di avere una rappresentazione numerica simile. Questi vettori numerici che rappresentano una parola sono chiamati Word Embedding. Gli algoritmi usati per la classificazione sono diversi, si passa da modelli Shallow come Naive Bayes, Linear Regression e Support Vector Machines, a modelli Deep come le Reti Neurali Artificiali.

Infine, l'approccio ibrido cerca di combinare gli aspetti positivi delle tecniche rule-based e automatiche in un unico sistema.

Come già detto in precedenza gli usi della Sentiment Analysis sono molteplici, come il monitoraggio dei social media per poter ad esempio valutare il cambiamento del sentiment dei clienti durante il tempo, capire il sentiment nei confronti di un prodotto o servizio, tenere traccia dei trend politici, conoscere la reputazione dei propri competitor, effettuare il cosiddetto Brand Monitoring, che consiste nel conoscere le opinioni delle persone riguardo un marchio attraverso non solo i social media, ma anche a notizie, blog, forum, recensioni ecc. Analisi di questo tipo permettono anche di classificare la reputazione del proprio marchio in base alle caratteristiche demografiche degli utenti, rendendo più facile capire dove e in che modo intervenire per migliorare la percezione del proprio brand, ad esempio organizzando il marketing sulla base delle informazioni estratte dalla Sentiment Analysis.

L'interesse della comunità scientifica e delle aziende per questo campo del NLP è sì dovuto all'importanza che ricopre per migliorare le scelte strategiche e l'efficienza di una compagnia, ma anche ai margini di miglioramento di questo tipo di analisi. La Sentiment Analysis è infatti molto complessa poiché giudicare il sentimento espresso in un testo non è affare semplice neanche per una persona e lascia quindi spazio a miglioramenti anche sostanziali nell'accuratezza con cui i testi vengono classificati. La situazione si complica ancora di più quando si prendono in considerazione lingue diverse dall'inglese, dato che la quantità di dati disponibili che possono essere usati per generare i modelli dei classificatori ha un impatto importante sulla qualità stessa dell'analisi, e spesso per le altre lingue scarseggiano i dataset da utilizzare. Questa è la motivazione per cui si parla di Cross-lingual Sentiment Analysis, cioè quando ci si focalizza sulla creazione di modelli capaci di categorizzare i testi anche in lingue diverse dall'inglese, argomento che verrà approfondito in questa tesi tramite la proposta di una soluzione che preveda l'uso di word embedding e sentiment embedding, e di una architettura di rete *deep*, che permetta di migliorare lo stato dell'arte in termini di accuratezza e scalabilità, ad esempio dimostrando la capacità di funzionare anche in contesti diversi da quelli dei dati utilizzati per la creazione del modello. Un altro degli obiettivi sarà quello di verificare se un modello così realizzato possa essere utilizzato per il Transfer Learning. Sarebbe infatti molto utile poter disporre di un modello il cui training è stato effettuato su una lingua (l'inglese) per cui la disponibilità di risorse è molto ampia, e che sia poi riutilizzabile per altre lingue per le quali le risorse disponibili sono inferiori.

In sintesi, si riportano le principali differenze metodologiche introdotte con questa tesi: eliminazione del Gradient Descent come algoritmo per la generazione di sentiment embedding, esplorazione dei vicini nello spazio degli embedding in un contesto multilingua, esplorazione di varianti o nuove architetture al posto della rete DC-CNN proposta da Dong e De Melo, e infine, valutare la possibilità di effettuare Transfer Learning.

# Capitolo 2

# Analisi dello stato dell'arte

In questo capitolo verranno presentati alcuni lavori correlati a questo progetto di tesi che propongono soluzioni diverse o che rappresentano l'attuale stato dell'arte. In particolare saranno riportati i dettagli di due pubblicazioni, di cui una realizzata dal Politecnico di Torino, che hanno funzionato da fondamenta per l'intera tesi e che saranno approfondite nelle due sezioni dedicate.

## 2.1 Cross-Lingual Propagation for Deep Sentiment Analysis

Questa pubblicazione [1] appartiene a Xin Dong e Gerard De Melo dell'università americana Rutgers, nello stato del New Jersey. I punti salienti saranno riassunti in questa sezione, data l'importanza che ricopre per il lavoro effettuato. Il punto di partenza della tesi è infatti la replica dei loro esperimenti e della rete neurale da loro proposta. In [1] Dong e De Melo propongono una possibile soluzione al problema della Sentiment Analysis multilingua, utilizzando come modello una rete neurale da loro progettata e chiamata *Dual-Channel Convolutional Neural Network* perché caratterizzata da una struttura a due canali (uno che gestisce i word embedding e uno che gestisce i sentiment embedding) e proponendo un approccio utile a generare embedding per diverse lingue, partendo dai dati disponibili in lingua inglese.

L'importante contributo di questa pubblicazione è l'algoritmo di propagazione, da loro definito cross-lingual, capace quindi di funzionare su più lingue e di generare risultati partendo da una lingua sorgente e arrivando a una lingua target diversa, attraverso cui sfruttare la grandissima quantità di dati disponibili in lingua inglese per risolvere il problema della mancanza e della difficoltà di estrazione dei dati in altre lingue. Per mancanza di dati non si intende una scarsità in senso assoluto di testi disponibili in rete in lingue che non siano l'inglese, dato che è facile rendersi conto semplicemente navigando su internet che ciò non sarebbe vero, bensì la

mancanza di tanti dati già raccolti e organizzati, pronti per essere usati per la realizzazione di un modello. Le tecniche di Deep Learning richiedono infatti molti dati per poter generare modelli affidabili, e l'inglese è la lingua su cui è dunque più facile reperire una grande quantità di testi già pronti e annotati per essere utilizzati nella fase di training delle reti neurali. L'annotazione dei dati, corrisponde nel nostro caso ad assegnare a ogni testo il suo sentiment, da cui la rete neurale (o un altro modello di ML) possa imparare a classificare correttamente i testi. Dong e De Melo propongono quindi un metodo per proiettare le informazioni del sentiment su più lingue, e di organizzare queste informazioni in embedding (vettori di numeri) che catturino le proprietà del sentiment lungo diverse dimensioni e permettano al modello di adattarsi a differenti domini. Con adattarsi a differenti domini si intende la possibilità del modello di gestire testi che trattino argomenti diversi, come ad esempio recensioni di alberghi, recensioni di elettrodomestici, recensioni di film, messaggi pubblicati su piattaforme social, notizie di politica, di economia ecc. Come detto, Dong e De Melo hanno utilizzato due tipi di embedding, i word embedding e i sentiment embedding. Entrambi sono una forma di rappresentazione vettoriale delle parole, ma i primi si concentrano sulla semantica della parola, mentre i secondi si prefiggono di rappresentare numericamente il sentiment espresso dalla parola su un'ampia gamma di contesti (o domini). Per generare i Word embeddings si possono utilizzare diversi metodi, ma quello oggi più usato, diventando di fatto uno standard, è la tecnica Word2vec (Mikolov et al. [2]) che trasforma una parola in un vettore che ha dimensione 300. Questo tipo di trasformazione si concentra sulla semantica della parola, mentre l'obiettivo che i due autori si sono posti è quello di realizzare degli embedding che catturassero il sentiment espresso da una parola. La tecnica usata è stata quella di raccogliere i sentiment embedding per parole inglesi, e poi derivare i sentiment embedding per altre lingue usando un algoritmo di propagazione basato su un grafo. I metodi usati per generare gli embeddings del sentiment in inglese sono stati diversi e sono spiegati di seguito:

**Sentiment Lexicons**, anche se caratterizzati da alcune limitazioni, il loro utilizzo per l'analisi del sentiment rimane molto diffuso. Uno dei loro punti di forza è la possibilità di comportarsi correttamente anche se usati in più domini, e per questo sono stati presi in considerazione come forma base di rappresentazione vettoriale di parole inglesi. In particolare Dong e De Melo si sono basati su un sentiment lexicon chiamato VADER (Hutto e Gilbert [3]), di cui i punteggi di polarità assegnati a ogni parola sono stati visti come componenti di un vettore monodimensionale.

**Domain-Specific Lexicon Induction**, cioè adattare i lexicon per essere specifici in base al dominio di utilizzo. I sentiment lexicon generici, infatti, non tengono in considerazione che una stessa parola, usata in contesti differenti, potrebbe esprimere un sentiment diverso. A questo scopo sono stati usati i dati raccolti in SocialSent dallo Stanford NLP group (Hamilton et al. [4]). Il loro studio ha

prodotto punteggi specifici per ogni dominio delle 250 sottocomunità di Reddit prese in considerazione. Questi lexicon, se presi tutti insieme, possono essere usati per generare embedding di 250 dimensioni, che riflettano la distribuzione della polarità del sentiment di una parola su molti contesti differenti.

**Transfer Learning** è l'ultimo dei metodi utilizzato per generare sentiment embedding, e fa affidamento ad un approccio di apprendimento supervisionato basato su dati di training annotati. Da questi dati caratterizzati da $n$ task di classificazione binaria del sentiment, si generano $n$ modelli corrispondenti. Da questi modelli si estraggono i pesi delle feature delle parole che sono collegati a specifici risultati della classificazione. In particolare Dong e De Melo hanno eseguito il training di $n$ modelli lineari del tipo:

$$f_i(x) = w_i^T x + b_i \tag{2.1}$$

dove $i = 1, ..., n$ e rappresenta i task. Dopodiché a ogni indice $j$ di una parola nel vocabolario viene assegnato un nuovo word vector (o sentiment embedding) della forma $(w_{1,j}, ..., w_{n,j})$ che incorpora i coefficienti lineari di quella parola lungo gli $n$ modelli lineari. In questo caso i sentiment embedding generati hanno dimensione 26.

Una volta ottenuti i sentiment embedding in lingua inglese utilizzando uno dei tre metodi appena esposti, si procede con la fase di **Cross-Lingual Induction**, cioè con la generazione dei sentiment embeddings in altre lingue. Questo processo è soddisfatto propagando i pesi (weights) attraverso parole in diverse lingue. Utilizzando la stessa notazione usata da Dong e De Melo, si inizia con un vocabolario $V_0 \subset V$ che è un sottoinsieme in lingua inglese del vocabolario multilingua $V$. Per ogni $x \in V_0$, si prende come input il corrispondente vettore $\tilde{v}_x \in \mathbb{R}^n$ ottenuto con uno dei tre metodi elencati in precedenza. L'obiettivo di questa fase, che rappresenta insieme alla struttura della DC-CNN il grande contributo di Dong e De Melo, è quello di generare gli embeddings $v$ per tutti gli $x \in V$. Si assume quindi un Translation Lexicon $T_L = \{(x_1, x_1^{'}, w_1), ..., (x_m, x_m^{'}, w_m)\}$ che fornisce l'evidenza di una relazione semantica tra parole in $V$ con pesi $w_i$. $T_L$ non contiene solo un mapping uno a uno tra parole che siano l'esatta traduzione l'una dell'altra nelle diverse lingue, bensì vengono presi in considerazione i collegamenti tra parole che abbiano un significato simile, tenendo in considerazione anche la sinonimia, cioè parole dal significato simile, e la polisemia, cioè la possibilità di una parola di avere significati diversi in base al contesto. Inoltre $T_L$ può contenere collegamenti monolingua, che sono stati usati per incorporare collegamenti tra sinonimi, varianti ortografiche e parole correlate semanticamente o etimologicamente. L'obbiettivo del training sui dati è quello di minimizzare il seguente valore:

$$-\sum_{x \in V} v_x^T \left[ \frac{1}{\sum_{(x,x',w) \in T_L} w} \sum_{(x,x',w) \in T_L} w v_{x'} \right] + C \sum_{x \in V_0} \|v_x - \tilde{v}_x\|_2 \tag{2.2}$$

La prima parte della formula verifica che i sentiment embeddings delle parole siano in accordo con i sentiment embeddings delle parole a esse collegate in termini di prodotto scalare, mentre la seconda parte della formula assicura che la deviazione dal word vector iniziale $\tilde{v}_x$ sia minima. In questo modo, le parole contenute nel vocabolario iniziale riceveranno vettori $v$ che non si discostano significativamente dall'originale $\tilde{v}_x$. Le nuove parole, al contrario, sono limitate ad avere vettori simili a quelli delle parole a loro vicine nel grafo. Per ottimizzare il processo, sono stati pre-inizializzati $v_x = \tilde{v}_x$ per tutti gli $x \in V_0$, e poi sono stati usati gli step della stochastic gradient descent. Il risultato è che il segnale del sentiment vector si propaga gradualmente dalle parole nel vocabolario originale alle altre parole nel lexicon.
Come detto, l'altra importante proposta fornita da Dong e De Melo è la struttura della DC-CNN, che però sarà approfondita nel capitolo 3 dato che la sua replica rappresenta uno dei contributi principali di questa tesi.

## 2.2 Cross-Lingual Propagation of Sentiment Information Based on Bilingual Vector Space Alignment

Questa pubblicazione [5] è stata invece realizzata al Politecnico di Torino e ha come obiettivo migliorare le metodologie utilizzate per generare i sentiment embeddings. Lo spunto per la ricerca condotta in questa pubblicazione è stato il lavoro svolto da Dong e De Melo per quanto riguarda la realizzazione di sentiment embeddings per più lingue a partire dai sentiment embeddings di una lingua sorgente (inglese). La soluzione proposta è quella di collegare indirettamente le parole correlate semanticamente in base alla loro similarità in uno spazio bilingue di word embeddings.

All'interno del **Bilingual embedding space** ogni parola in un dizionario è mappata a un vettore nello spazio latente. Si utilizzano i word embeddings così come generati da fastText [6], una estensione del già citato Word2Vec [2], che fornisce una rappresentazione vettoriale più efficace incorporando sotto-parole nel dizionario di input. Grazie a questa peculiarità, è possibile combinare i vettori associati alle sotto-parole per generare gli embeddings di nuove parole che non sono ancora presenti all'interno del dizionario. Le rappresentazioni vettoriali del testo sono realizzate separatamente per ogni lingua, usando architetture per Deep Learning. Per poter collegare parole di lingue diverse, i modelli di ogni lingua devono essere prima allineati. Un discreto numero di modelli allineati pre-allenati sono già disponibili, dando la possibilità di usare vettori multilingua general-purpose, senza doverli sottoporre nuovamente a una fase di training.

**Strategia di propagazione del Sentiment**. Di seguito viene riportata la strategia proposta da questa pubblicazione, utilizzando la stessa notazione. Sia $E_o$ lo spazio degli embedding generati con fastText nella lingua originale (inglese), e $E_T$ lo spazio degli embedding allineati nella lingua target (es. italiano, francese, tedesco ecc.). Ogni parola $x$ nella lingua di origine ha un corrispettivo vettore $v_x^{eo}$ in $E_o$. Grazie allo spazio dei vettori bilingue allineati, si può proiettare $v_x^{eo}$ nello spazio dei vettori target per ottenere il corrispondente vettore target $v_x^{et}$ in $E_T$. Bisogna però notare che il vettore così ottenuto nello spazio target, non necessariamente corrisponde a una parola esistente. Si sfruttano le somiglianze tra parole nello spazio latente per propagare le informazioni riguardanti il sentiment, in particolare, considerando $v_x^s$ come il sentiment vector di una parola $x \in V_0$ , l'obiettivo è propagare le informazioni riguardanti il sentiment ad altre parole in $V/V_0$. Questo obiettivo si raggiunge in due step: per prima cosa si crea un grafo di parole che rappresenti le principali coppie di parole in base alla similarità; il secondo step consiste nel propagare i punteggi del sentiment alle altre parole, utilizzando Gradient Descent.

La creazione del grafo di parole è un'altra delle questioni che deve essere approfondita. Il grafo $G = (V, E)$ è un grafo pesato indiretto, che connette coppie di parole in $V$. Gli archi in $E$ sono triplette della forma $\left(x, x^{'}, w_{xx'}\right)$, dove $x, x^{'} \in V$ sono i nodi collegati, e $w_{xx'}$ è il peso dell'arco che li collega. Per ogni parola $x \in V$ si esplorano i vicini del vettore $v_x^{et}$ nello spazio latente target, per trovare le parole vicine che sono semanticamente più correlate a $x$. In particolare si guarda ai $K$ vettori più vicini (dove $K$ è un parametro che deve essere specificato dall'utente) che corrispondono alle parole della lingua target che sono più vicine a $v_x^{et}$, e si selezionano tali parole. Dato l'insieme $NN_x$ delle parole più vicine a $x$, si crea un arco pesato $e \in E$ che colleghi ogni $x^{'} \in NN_x$ a $x$. Il peso dell'arco che collega le parole $x$ e $x'$ indicano la similarità della coppia di parole nello spazio latente ed è calcolato usando la similarità del coseno (cosine similarity). Per evitare di introdurre relazioni non affidabili tra parole, vengono filtrati gli archi il cui peso è inferiore a una determinata soglia $\alpha$.

Il Gradient Descent è stato usato per propagare le informazioni sul sentiment attraverso il grafo di parole. Questo processo di propagazione deve preservare i valori dei vettori presenti nel vocabolario iniziale $V_0$, e allo stesso tempo garantire un alto livello di similarità tra i sentiment vector delle parole collegate. La funzione della loss utilizzata per ottenere questi due obiettivi è diversa da quella proposta da [1] ed è così definita:

$$C \cdot \sum_{x \in V_0} \|v_x - \tilde{v}_x\|_2 - \sum_{x \in V} \left\| v_x - \left[ \frac{1}{\sum_{\left(x,x',w_{xx'}\right) \in E} w_{xx'}} \cdot \sum_{\left(x,x',w_{xx'}\right) \in E} w_{xx'} v_{x'} \right] \right\|_2 \quad (2.3)$$

dove, per ogni parola $x$ nel vocabolario iniziale $V_0$, il primo termine minimizza

la deviazione dal suo sentiment embedding iniziale $\tilde{v}_x$, mentre il secondo termine minimizza la deviazione dai sentiment vectors delle parole vicine, rappresentate come parole collegate all'interno del grafo. In questo modo, le parole contenute nel vocabolario iniziale mantengono con una buona approssimazione la loro rappresentazione vettoriale originale, mentre le nuove parole ottengono dei punteggi della polarità del sentiment simili a quelli delle parole vicine nello spazio latente target.

### 2.2.1 Generazione di nuovi Sentiment Embedding

Collegato a [5] è il lavoro di generazione di sentiment embedding con l'ausilio di nuovi algoritmi al posto di Gradient Descent. I sentiment embedding generati con questi algoritmi vanno ad espandere il lavoro svolto in [5], essendone la naturale evoluzione, e rappresentano un componente fondamentale per l'intero lavoro di tesi. Gli algoritmi utilizzati per generare i nuovi sentiment embedding sono:

- K-Nearest Neighbors
- Linear Regressor
- Random Forest
- Multilayer Perceptron

Alcuni di questi sono stati usati con diversi parametri, come nel caso di K-Nearest Neighbors che ha tre diverse versioni, Random Forest che è stato usato sia con 100 che con 250 alberi di decisione, e Multilayer Perceptron che è anch'esso stato utilizzato in diverse varianti, aumentando così il numero di diverse soluzioni adottate per generare nuovi sentiment embeddings.

## 2.3 Altri lavori correlati

In questa sezione vengono presentati alcuni lavori correlati al lavoro di tesi, che trattano dunque lo stesso argomento (Cross-Lingual Sentiment Analysis) proponendo soluzioni differenti, ma che a differenza delle pubblicazioni presentate in dettaglio nelle sezioni precedenti non hanno rappresentato uno spunto per le soluzioni proposte in questa tesi.

### 2.3.1 Cross-Lingual Sentiment Analysis Without (Good) Translation

In questo articolo [7] viene proposta una possibile soluzione per estendere la sentiment analysis a lingue per le quali i dati disponibili sono molto carenti. Si

propone infatti un metodo che si distingua dalle due metodologie più utilizzate per trasferire le informazioni possedute sulla lingua inglese ad altre lingue. I metodi solitamente usati per questo tipo di analisi in diverse lingue rientrano infatti in due categorie: i) quelli che si basano sull'uso di Word Embedding bilingue, quindi usando embedding pre-trained, e ii) quelli che utilizzano sistemi di traduzione (Machine Translation) per ottenere input allineati per estrarre le features che funzionano su entrambe le lingue. Il problema di queste due metodologie è che molte lingue non hanno un sistema di machine translation esistente, e anche raccogliere i dati necessari per i word embedding può essere molto dispendioso. Per questo motivo in [7] viene proposto un metodo efficace e poco costoso per effettuare la classificazione del sentiment cross-lingual, dimostrando come con pochi dati di training si possano ottenere buoni risultati in diversi contesti. Questo approccio si basa sul metodo della matrice di traduzione (translation matrix) nello spazio dei vettori, come presentato in [8], il quale prevede la creazione di una matrice capace di tradurre dallo spazio vettoriale di una lingua sorgente allo spazio vettoriale di una lingua target. Questo metodo prevede che siano disponibili coppie di parole e le loro rispettive rappresentazioni vettoriali. In particolare, sono date $j$ coppie di parole, $\{x_i, z_i\}_{i=1}^{j}$ dove $x_i \in \mathbb{R}^n$ è un vettore della lingua sorgente della parola $i$, e $z_i \in \mathbb{R}^m$ è la rappresentazione vettoriale della corrispettiva parola tradotta nella lingua target. Si vuole trovare una matrice $W$ per la quale $Wx_i$ approssimi $z_i$. Questa matrice si trova risolvendo il seguente problema di ottimizzazione:

$$\min_{W} \sum_{i=1}^{j} \|Wx_i - z_i\|^2 \tag{2.4}$$

Per tradurre una parola dalla lingua sorgente a quella target, si pone $z = Wx$, e si cerca la parola più vicina nello spazio di quella lingua, usando la similarità del coseno per misurare la distanza.

Questo metodo si basa sull'osservazione che il sentiment è preservato anche in caso di una traduzione poco precisa. Gli autori di questo lavoro hanno infatti notato che un classificatore di sentiment allenato solo con word vector inglesi funziona bene con parole di lingue mai viste prima tradotte nello spazio vettoriale della lingua inglese tramite la matrice ottenuta con il metodo spiegato in precedenza, anche quando le traduzioni non sono ottimali. Sono stati usati 2000, 4500 e 8500 coppie di parole per i test, numeri che sono semplici da raccogliere per una persona anche con lingue per cui c'è carenza di dati a disposizione. La classificazione (positiva o negativa), una volta ottenuti i vettori "tradotti" per mezzo della matrice di traduzione, viene effettuata tramite l'uso di una Support Vector Machine (SVM).

### 2.3.2 Experiments in Cross-Lingual Sentiment Analysis in Discussion Forums

In questo lavoro [9] si propone una classificazione supervisionata di post pubblicati su forum francesi facendo uso di alcune funzionalità come il POS (part-of-speech) tagging. Per effettuare la sentiment analysis di questi testi francesi sono state proposte tre categorie di caratteristiche del testo: lessicale, morfo-sintattica, semantica. Le caratteristiche lessicali e morfo-sintattiche sono state formulate considerando ogni singola parola, mentre le caratteristiche semantiche sono state generate a livello di frasi e dell'intero post. Le parole vengono utilizzate come unigram, e l'uso del POS tagging (come ad esempio distinguere gli aggettivi dai nomi) serve per aumentare le caratteristiche degli unigram con informazioni aggiuntive che permettano ad esempio di distinguere parole che siano molto simili nello spelling, ma che abbiano una polarità del sentiment diversa. Per estrarre la polarità delle parole è stato usato SentiWordNet [10], con il quale sono state calcolate anche le polarità per ogni POS tag. SentiWordNet assegna a ogni gruppo di sinonimi (synset) tre punteggi: positività, negatività e obiettività. Sono state aggiunte anche due features che contengono un valore binario per la positività e negatività di ogni categoria di POS (aggettivi, avverbi, nomi e verbi). Questo valore binario è stato definito in base alla somma dei punteggi estratti da SentiWordNet: se il valore maggiore è risultato essere positivo, la feature positiva è impostata a 1 e quella negativa a null, altrimenti vale il ragionamento inverso. SentiWordNet si riferisce però a parole inglesi, e quindi è stato necessario tradurre i testi francesi in inglese prima di procedere alla definizione del sentiment. Il problema che questa pubblicazione cerca di risolvere, è la corretta selezione del gruppo di sinonimi a cui una parola tradotta appartiene. SentiWordNet assegna infatti i punteggi di polarità organizzando le parole in synset; la traduzione dal francese all'inglese può però far ricadere una parola nel gruppo sbagliato di sinonimi (data la possibilità di una parola di avere significati diversi in base al contesto) rendendo poi l'assegnazione della polarità imprecisa. A questo scopo, oltre a tradurre i testi francesi in inglese, è stato usato EuroWordNet che contiene i gruppi di sinonimi allineati per diverse lingue, così da poter passare correttamente da un synset di una lingua, al corrispettivo synset di un'altra lingua. In questo lavoro EuroWordNet è stato ovviamente utilizzato per passare correttamente dai synset francesi a quelli inglesi, per poi procedere all'estrazione del sentiment usando i punteggi di SentiWordNet che come detto si riferiscono solo alle parole inglesi. Il modello utilizzato in questo caso, così come nel lavoro presentato nella sezione precedente, è una Support Vector Machine.

### 2.3.3 Cross-Lingual and Low-Resource Sentiment Analysis

Questa tesi di dottorato [11] propone un modello e dei metodi di raccolta e preparazione dei dati utili a realizzare sentiment analysis su lingue che dispongono di pochissime risorse reperibili in rete. L'obiettivo di questo lavoro è infatti la realizzazione di un modello con il quale sia possibile effettuare transfer learning, passando da una lingua per cui i dati sono molto abbondanti (come sempre, l'inglese) a una lingua per cui le risorse a disposizione sono scarse. A questo scopo vengono raccolte risorse bilingue (come ad esempio word embedding bilingue) che servono per fare da ponte tra la lingua sorgente e la lingua target, rendendo possibile la realizzazione di un modello il cui training sia stato effettuato solo su dataset della lingua sorgente. In questo modo si trasportano le informazioni sul sentiment da una lingua con tante risorse, a una lingua con poche risorse disponibili. La sfida più importante quando si implementano sistemi per trasferire informazioni tra più lingue è quella di realizzare un modello capace di generalizzare le caratteristiche imparate dalla lingua sorgente. Spesso infatti, le caratteristiche lessicali imparate da una lingua non sono generalizzabili alle altre lingua, rendendo il transfer learning molto complesso da attuare. Per risolvere questo problema, l'autore prende in considerazione alcune tecniche per rappresentare le features sulle due lingue in esame (sorgente e target): l'uso di word embedding cross-lingual e pre-trained, l'uso di sentiment embedding bilingue e la realizzazione di un lessico per la lingua target per poter aggiornare i pesi degli embedding bilingue.

Il modello utilizzato è un'architettura di deep learning composta prevalentemente da une rete Long Short-Term Memory (LSTM) bidirezionale. L'input è rappresentato da una sequenza di parole e viene fornito a due layer distinti: una LSTM bidirezionale che codifica l'input in una rappresentazione basata sulla sequenza dei dati, e un layer di average pooling, che media le features considerando tutte le parole di input. La LSTM bidirezionale cattura la sequenza di parole in ingresso, mentre il layer di average pooling serve per gestire situazioni per cui la lingua sorgente e la lingua target presentano differenze nell'ordine in cui sono usate le parole o differenze nella struttura di una frase. Questi due layer sono infine concatenati e forniti in input all'ultimo layer dell'architettura caratterizzato dall'uso di *softmax* come funzione di attivazione, che fornisce in output le probabilità per ognuna delle tre classi (positivo, negativo, neutro) da cui viene scelta l'etichetta del testo di input prendendo la classe con la probabilità più alta.

## 2.4 Confronto riassuntivo tra metodi proposti e lavori correlati

Di seguito sono riassunte le principali differenze tra le metodologie utilizzate in questo lavoro di tesi e i lavori correlati che sono stati presentati in questo capitolo. In particolare i confronti più interessanti sono quelli con i lavori che rappresentano il punto di partenza di questa tesi, cioè [1] e [5].

**Differenze con metodi proposti da Dong e De Melo [1]:**

- utilizzo di nuovi algoritmi per la generazione di sentiment embedding, abbandonando Gradient Descent usato da Dong e De Melo perché non garantisce di trovare un ottimo globale e richiede molto tempo per l'elaborazione;
- aggiunta di nuovi dataset che abbiano dimensioni maggiori rispetto a quelli usati da Dong e De Melo (i loro dataset hanno dimensioni limitate);
- valutazione della portabilità di un modello specifico in un contesto cross-lingual (Transfer Learning);
- proposta di una nuova architettura di rete (LSTM) al posto della DC-CNN, per poter utilizzare così una rete più semplice e meno profonda caratterizzata dalla capacità di mantenere una memoria dei dati analizzati.

**Differenze con metodi proposti da Giobergia et al. [5]:**

- utilizzo di nuovi algoritmi per la generazione di sentiment embedding;
- aggiunta di dataset appartenenti ad un contesto diverso dalle recensioni (nello specifico Twitter);
- uso di modelli di deep learning per la classificazione del sentiment.

**Differenze con metodi proposti da Abdalla e Hirst [7]:**

- uso di sentiment embedding insieme ai word embedding;
- il classificatore usato nella tesi è una deep neural network e non un modello shallow come SVM;

**Differenze con metodi proposti da Ghorbel [9]:**

- uso di embedding per rappresentare in forma vettoriale le parole;
- supporto di più lingue, e generazione di embeddig per ognuna delle lingue al posto dell'uso di machine translation;

- rete neurale deep per la classificazione dei dati di input al posto del modello shallow (SVM) usato nel lavoro di Ghorbel.

**Differenze con metodi proposti da Farra [11]:**

- in questa tesi si è cercato di adattare il modello proposto da Dong e De Melo nel tentativo di rendere possibile il transfer learning, mentre nel lavoro di Farra il modello proposto (una rete LSTM bidirezionale al posto delle reti DC-CNN e LSTM classica usati per la tesi) è stato pensato appositamente per il transfer learning;

# Capitolo 3

# Soluzioni Proposte

## 3.1 Sperimentazioni proposte

### 3.1.1 Implementazione Dual-Channel Convolutional Neural Network

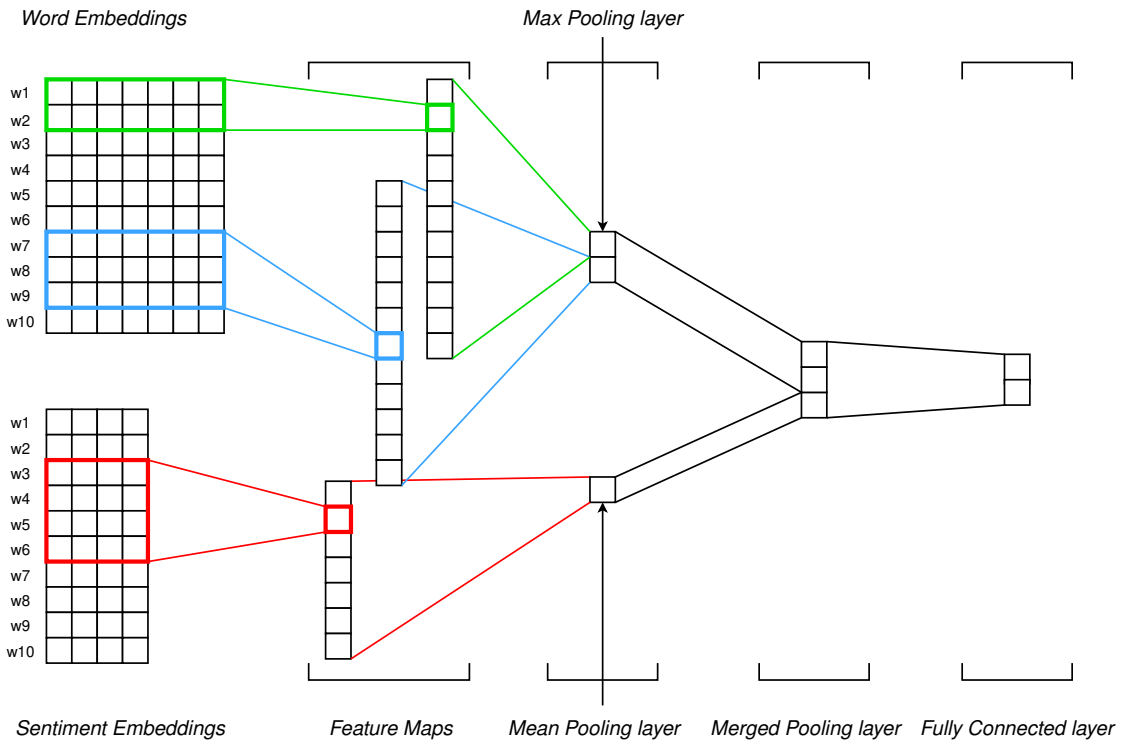


**Figura 3.1:** Struttura della Dual-Channel Convolutional Neural Network

Il primo passo, e anche il più importante, è stato ricostruire la rete definita da Dong e De Melo *Dual-Channel Convolutional Neural Network*, così chiamata perché caratterizzata da due canali, e dall'uso di layer convoluzionali. Questa rete, proposta in [1], è stata in grado di ottenere i migliori risultati al confronto con altre soluzioni, e per questo la sua replica è importante per poter valutare i sentiment embedding generati con il metodo presentato in [5], e i suoi sviluppi come spiegato nella sezione 2.2.1, utilizzando una rete neurale che rappresenti lo stato dell'arte. Purtroppo il codice sorgente della DC-CNN non è stato fornito da Dong e De Melo e per questo motivo la rete deve essere replicata utilizzando le informazioni contenute in [1]. Come detto, l'architettura della rete è caratterizzata da due channel: uno gestisce i *word embedding*, l'altro gestisce i *sentiment embedding*. Ricordiamo che gli embedding sono una rappresentazione numerica vettoriale di una parola, ed in particolare i word embedding rappresentano il significato di una parola, mentre i sentiment embedding rappresentano le informazioni sulla polarità (positivo o negativo) della parola stessa. Dato l'obiettivo di replicare esattamente la DC-CNN proposta in [1], anche tutti i parametri della rete saranno quelli scelti da Dong e De Melo.

Il canale dei word embedding è così composto:

- Input layer
- Convolutional layer
- Dropout
- Max pooling

In particolare, l' Input layer accetta in input una matrice di dimensione $w \times d$, dove $w$ è il numero di parole contenute nella frase di input (che può essere una recensione, un tweet, o qualsiasi altro testo) e $d$ è la dimensionalità degli embeddings. Nel nostro caso, i word embeddings sono quelli generati con fastText e hanno quindi una dimensione $d = 300$. La dimensione della matrice di input deve essere fissa. Se questo è vero per quanto riguarda il valore di $d$, ovvero la dimensione dei word embedding che rappresentano le colonne della nostra matrice di input, altrettanto non si può dire per $w$, cioè il numero di parole contenute nel testo di input, corrispondente al numero di righe della matrice. Ogni testo, infatti, contiene potenzialmente un numero diverso di parole. Per risolvere questo problema, prima di iniziare la fase di training, si cerca il testo che contiene più parole (o token, visto che il testo originale viene prima suddiviso in token e ripulito dai caratteri inutili), e tutti gli altri testi che risultino essere più corti, vengono portati a questa lunghezza massima applicando del padding a zero. Prima del Convolutional layer, viene applicato uno zero padding all'inizio e alla fine del testo (che corrisponde ad aggiungere delle parole "nulle") per poter effettuare una *wide convolution*, cioè per

fare in modo che il filtro del layer convoluzionale che "scorre" sugli embedding, passi sopra a tutti gli embedding con ognuna delle sue righe. Il padding da applicare deve avere quindi dimensione uguale alla dimensione del filtro meno uno. Nel canale delle word vengono applicati contemporaneamente 3 filtri di dimensioni 3,4,5 (riferito sempre al numero di righe). Il numero di feature maps generate dalla convoluzione è impostato a 100 (per ogni filtro usato). Si applica poi il Dropout (posto a 0.5 come suggerito da Dong e De Melo), e infine si calcola il Max Pooling prendendo in considerazione tutta la lunghezza della lista ottenuta dal Convolutional layer.

La struttura del canale dei sentiment embedding è invece leggermente diversa:

- Input layer
- Convolutional layer
- Dropout
- Mean pooling

L'Input layer del channel dedicato ai sentiment embedding funziona esattamente come l'Input layer del channel dei word embedding, ma in questo caso la matrice di input ha dimensioni $w \times d'$, dove $w$ indica esattamente come prima il numero di parole (token) contenute nel testo di input più lungo (anche in questo caso si procede al padding a zero per portare i testi più corti alla stessa lunghezza di quello che contiene più parole così da avere le dimensioni della matrice di input fisse), e $d'$ è la dimensione dei sentiment embeddings, nel nostro caso 26. Su questa matrice si effettua una *narrow convolution*, e quindi non viene utilizzato padding per far scorrere il filtro su tutte le righe di input. Il Convolutional layer in questo caso usa un solo filtro, la cui dimensione può variare a seconda della lingua. In particolare, Dong e De Melo propongono un filtro di dimensione 5 per tutte le lingue eccetto il tedesco per cui hanno usato un filtro di dimensione 20. Anche in questo caso il numero di feature maps generate è impostato a 100. Segue poi il Dropout (sempre impostato a 0.5) e infine si applica un layer di Mean Pooling (anche detto Average Pooling) che viene applicato sull'intero tensore risultante dal layer di convoluzione, e che, insieme al numero di filtri usati per la convoluzione, rappresenta l'altra importante differenza con il channel dei word embedding. Il motivo di quest'ultima differenza, secondo Dong e De Melo, è da ricercarsi nel tentativo di catturare la polarità media del sentiment e non il sentiment più marcato che sarebbe stato invece catturato dal Max Pooling.

I risultati ottenuti in uscita dai due layer di pooling (corrispondenti al channel di word embedding e al channel di sentiment embedding) vengono uniti nel Merged Pooling layer, che rappresenta il punto di incontro dei due channel. Dal Merged Pooling layer si passa infine a un Fully Connected layer con 2 neuroni, che si occupa di fornire la classificazione finale (positivo o negativo), sfruttando Softmax come

funzione di attivazione. Entrambi i canali usano invece *ReLU* come funzione di attivazione del Convolutional layer.

La loss function utilizzata per questa rete è la funzione *cross-entropy*, così definita:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c\in C} y_{i,c} \ln \hat{y}_{i,c} \tag{3.1}$$

dove $n$ è il numero di esempi di training, $C$ è l'insieme delle classi (due, positivo e negativo), $y_{i,c}$ sono le etichette della ground truth, e $\hat{y}_{i,c}$ sono le probabilità della classificazione effettuata dal modello. La fase di training del modello è stata effettuata utilizzando l'ottimizzatore Adam [12], che permette di ottenere un modello più robusto con diversi dataset.

### 3.1.2 Specifiche dei Dataset utilizzati

I dataset forniti da Dong e De Melo sono multilingua, e coprono 8 differenti lingue (Ceco (cs), Francese (fr), Inglese (en), Italiano (it), Olandese (nl), Russo (ru), Spagnolo (es), e Tedesco (de) ).

| **Dataset** | **Cardinalità** | **Positive** | **Negative** |
|---|---|---|---|
| cs | 2,458 | 1,660 | 798 |
| de | 2,407 | 1,839 | 568 |
| en | 5,949 | 3,622 | 2,327 |
| es | 2,951 | 2,367 | 584 |
| fr | 3,912 | 2,080 | 1,832 |
| it | 3,559 | 2,867 | 692 |
| nl | 1,892 | 1,232 | 660 |
| ru | 3,414 | 2,500 | 914 |

**Tabella 3.1:** Cardinalità e distribuzione delle classi per ognuno dei dataset presentati in [1]

Questi dataset sono stati raccolti da Dong e De Melo, e sono formati da recensioni che hanno diverse origini:

- Stanford Sentiment Treebank [13], che caratterizza uno dei due dataset inglesi (questo non è stato usato nei nostri test) e che racchiude recensioni di film prese dal sito Rotten Tomatoes.
- Amazon Fine Food Reviews [14], che è un insieme di 568,454 recensioni pubblicate su Amazon, di cui una parte è stata selezionata e pre-processata, per creare il secondo dataset inglese (quello che abbiamo usato per i nostri test). La fase di pre-processing effettuato da Dong e De Melo è consistito

nella traduzione dei punteggi (su una scala di 5 stelle) in etichette (positivo, negativo). A tutte le recensioni con 1 o 2 stelle è stata assegnata l'etichetta negativa, alle recensioni con 4 o 5 stelle quella positiva. Le recensioni con 3 stelle, indicando una recensione neutra, sono state scartate.

- TripAdvisor, sito molto conosciuto che raccoglie recensioni di alberghi, ristoranti, attrazioni turistiche ecc. Il sito è disponibile in molte lingue, per questo è stato usato come fonte per i dataset in lingua tedesca, russa, italiana e ceca. Anche in questo caso è stato eseguito un lavoro di pre-processing dei dati che ha seguito la stessa logica di quanto fatto con le recensioni raccolte da Amazon. Questi dataset sono stati raccolti direttamente da Dong e De Melo [1].
- Il dataset AlloCine [15], che contiene recensioni di serie TV francesi, ed è quindi stato utilizzato per creare il dataset francese.
- Il dataset SemEval-2016 Task 5 [16], che contiene recensioni di ristoranti in lingua spagnola e tedesca.

A questi dataset, abbiamo poi aggiunto due nuovi dataset Italiani ($IT_1$ e $IT_2$) che contengono solo recensioni in italiano e sono caratterizzati da una cardinalità molto maggiore rispetto ai dataset forniti da Dong e De Melo, e cinque dataset multilingua ricavati da Twitter che permettono di testare il modello usando dati piuttosto diversi dalle recensioni contenute in tutti gli altri dataset. I particolari di questi nuovi dataset (sia quelli italiani, sia quelli contenenti tweet) sono riportati nelle due sezioni dedicate di questo capitolo.

### 3.1.3 Replicabilità dei risultati ottenuti da Dong e De Melo

Dopo aver ricostruito la rete DC-CNN proposta in [1], il passo successivo è stato validare il lavoro svolto cercando di replicare i risultati ottenuti da Dong e De Melo. A questo scopo sono stati usati gli stessi embedding (sia word che sentiment) e gli stessi dataset da loro utilizzati.

Prima di procedere con le canoniche fasi di training e test, le recensioni (in questo caso i dataset sono tutti formati da recensioni) vengono caricate nel loro formato originario testuale e subiscono poi un processo di trasformazione per poter essere utilizzate nella DC-CNN. Tale processo comincia trasformando le recensioni in liste di token (cioè parole ripulite dalla punteggiatura e da altri caratteri inutili), e ottenendo le etichette relative a ognuna delle recensioni presenti nel dataset (che sono espresse in formato testuale, "neg" per le recensioni negative e "pos" per le recensioni positive). Le etichette vengono convertite in interi, 0 se la recensione è negativa, 1 se è positiva. Si prosegue con l'ultimo processo necessario da applicare ai dati di input, cioè la trasformazione dei token delle recensioni, nei rispettivi

word e sentiment embeddings. Gli embeddings sono contenuti in un dizionario che mappa una parola (token) al suo corrispondente vettore. Nel caso in cui una parola non sia contenuta nel dizionario degli embeddings, viene inserito un embedding di valori pari a zero. I dati di input sono divisi in due gruppi, il training set e il test set, con una distribuzione di 80% training set e 20% test set. Il validation set non è stato preso in considerazione dato che il lavoro di tuning dei parametri della rete è già stato svolto in [1].

A questo punto si procede con la fase di training del modello nel tentativo di replicare i risultati ottenuti da Dong e De Melo, e per questo motivo i parametri della rete usati sono stati quelli riportati in [1]. In particolare, i dati vengono processati nella fase di trainig con batch di 50 recensioni, mentre il Learning Rate (l'unico parametro di rete di cui non si è ancora parlato), assume due valori a seconda della lingua del dataset in uso. I Learning Rate $\alpha$ usati sono riportati nella seguente tabella.

| | cs | de | en | es | fr | it | nl | ru |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | 0.001 | 0.0004 | 0.0004 | 0.001 | 0.0004 | 0.0004 | 0.001 | 0.0004 |

**Tabella 3.2:** Riepilogo Learning Rate $\alpha$ usati in base al dataset

Per effettuare il training cercando di ottenere il modello migliore, si è utilizzata la tecnica di Early Stopping, vale a dire che il training è stato impostato per eseguire un elevato numero di epoche, ma con la possibilità di fermarsi anticipatamente in base ai valori della loss. Infatti, nel caso in cui la loss sul training set non diminuisca di più di 0.01 per 3 epoche consecutive, la fase di training viene terminata. Le epoche massime impostate sono pari a 50, numero abbastanza grande da permettere all'Early Stopping di funzionare correttamente con tutti i dataset. In generale le epoche in cui si sono ottenuti i risultati migliori variano dalla epoca 15 alla 27 a seconda del dataset in uso. Ovviamente in questo caso i sentiment embedding usati sono stati quelli generati da Dong e De Melo con il metodo presentato in [1]. Si è proceduto in fine alla fase di test del modello ottenuto e alla raccolta dei risultati (in termini di accuracy, macro f1, precision, recall). I risultati, così come anche per tutti i test successivi, sono raccolti e commentati nel Capitolo 4.

Una volta replicati i test effettuati da Dong e De Melo e verificato che la rete si comporta come atteso, si è proceduto a nuovi test che si concentrassero sull'uso di nuovi sentiment embedding. Il primo di questi test è stato realizzato usando nella fase di training (e di test) i sentiment embedding ottenuti con il lavoro svolto in [5], di cui si è parlato in dettaglio nella sezione dedicata del Capitolo 2. Tutto il resto dei parametri, della struttura della rete, e dei dataset è rimasto invariato, in modo da poter valutare solo il contributo dei nuovi embedding. Si è proceduto poi a provare tutti i nuovi sentiment embedding di cui si è parlato nella sezione

2.2.1 del Capitolo 2, sempre mantenendo invariati i parametri di rete. Questi nuovi sentiment embeddings, come detto, sono stati generati sfruttando diversi algoritmi e sono stati generati per tutte le lingue escluso l'inglese. I primi ad essere stati utilizzati sono quelli ottenuti con *K-Nearest Neighbors*, dove è stato posto $K = 10$, e sono state generate anche due varianti definite *Euclidean* e $Uniform$ dove sono stati modificati alcuni parametri del regressore mantenendo $K = 10$. Sono poi stati testati gli embedding ottenuti con un *Regressore Lineare (Linear Regressor LR)*, quelli ottenuti con l'algoritmo *Random Forest* che è stato utilizzato per generare due sentiment embedding per ogni lingua, usando 100 e 250 alberi di decisione, e infine sono stati utilizzati i sentiment embedding ottenuti con il *Multilayer Perceptron* (MLP), con cui sono stati generati quattro sentiment embedding per ogni lingua (MLP, MLP con un hidden layer da 500 unità, MLP con 500 e 1000 unità nell'hidden layer con applicato min-max scaling sull'input).

### 3.1.4 Nuovi dataset in lingua italiana

Nel tentativo di ottenere risultati dalla qualità ancora migliore ed espandere i dati a disposizione per la sentiment analysis, abbiamo utilizzato due nuovi dataset italiani chiamati $IT_1$ e $IT_2$. Questi due dataset sono preferibili a quelli usati da Dong e De Melo per due ragioni: la prima, è che questi due dataset sono meglio bilanciati, cioè il numero di recensioni positive è uguale al numero di recensioni negative (50% e 50%); la seconda, è che questi dataset sono sensibilmente più grandi, cioè contengono molte più recensioni rispetto ai dataset usati da Dong e De Melo. Nella tabella seguente sono riportate le cardinalità dei due nuovi dataset.

| **Dataset** | **Cardinalità** | **Positivi** | **Negativi** |
|---|---|---|---|
| $IT_1$ | 10,024 | 5,012 | 5,012 |
| $IT_2$ | 13,888 | 6,942 | 6,946 |

**Tabella 3.3:** Cardinalità dei due nuovi dataset Italiani

I due dataset sono stati creati raccogliendo recensioni di alloggi di diverse città italiane sul sito Airbnb. I test eseguiti su questi due dataset hanno ricalcato esattamente quanto fatto con i dataset originali di Dong e De Melo, quindi provando tutti i diversi sentiment embedding e utilizzando i parametri della rete DC-CNN riportati in [1].

### 3.1.5 Dataset multilingua Twitter

Oltre ai dataset contenenti recensioni, cioè i dataset usati da Dong e De Melo e i due nuovi dataset italiani raccolti da Airbnb, abbiamo deciso di mettere alla

prova la DC-CNN con testi che non fossero recensioni per testare la capacità degli embeddings di generalizzare le loro informazioni, utilizzando cinque dataset contenenti tweet in quattro diverse lingue riguardanti i mondiali di calcio del 2018. I dataset sono disponibili pubblicamente su Github[1] e comprendono nella loro interezza circa 8 milioni di tweet. Di questi 8 milioni, 6 sono in lingua inglese, che non sono però stati presi in considerazione nei test effettuati. I restanti 2 milioni di tweet in lingua francese, italiana, spagnola e tedesca sono a loro volta divisi in due sotto-dataset, uno dei quali, per ognuna di queste quattro lingue, è stato usato con la nostra rete neurale. Il tedesco è l'unica lingua per cui abbiamo usato entrambi i dataset disponibili (*players*, contenente tweet riguardanti i giocatori, e *teams*, contenente tweet riguardanti le squadre). Per il francese abbiamo usato il dataset teams, così come per lo spagnolo, mentre per l'italiano si è usato il dataset players. Questi dataset contengono per ogni riga un tweet, accompagnato dal corrispondente sentiment, che può essere di 4 tipi: Positivo, Negativo, Neutrale, Misto. Dato che la nostra rete prevede una classificazione binaria, abbiamo tenuto solo i tweet che avessero un sentiment Positivo o Negativo. I tweet sono stati raccolti nelle loro lingue originali, ma per essere etichettate (Positive, Negative, Neutre o Miste) si è utilizzato AWS Comprehend, che essendo però un sistema automatizzato non garantisce la stessa affidabilità che garantirebbe una persona nell'assegnazione dei sentiment a ogni tweet. Nel caso di francese e spagnolo, i tweet sono stati prima tradotti in inglese con Google Translate (che aggiunge un ulteriore limite all'affidabilità dell'assegnazione delle etichette) e poi forniti ad AWS Comprehend per essere etichettati. I test eseguiti su questi dataset hanno ricalcato quanto fatto con i dataset usati in precedenza. Sfortunatamente non ci sono pubblicazioni che abbiano effettuato questi tipi di test su questi tweet e non è quindi possibile confrontare i risultati da noi ottenuti con quelli ottenuti da altri.

## 3.2 Nuove metodologie proposte

### 3.2.1 Transfer Learning

Fino a questo punto si sono sempre creati dei modelli specifici per ogni lingua (e quindi per ogni dataset), rendendo necessario ripetere le operazioni di training per ognuna delle lingue prese in esame. Abbiamo dunque deciso di provare a effettuare Transfer Learning, allenando un unico modello su un dataset di buona qualità e di grandi dimensioni, per poi testarlo con i dataset di altre lingue. Il dataset usato per realizzare il modello è $IT_1$, uno dei due nuovi dataset italiani, che come detto è caratterizzato da un buon numero di recensioni (nettamente più numerose rispetto

[1] https://github.com/charlesmalafosse/open-dataset-for-sentiment-analysis

ai dataset forniti da Dong e De Melo) ed è perfettamente bilanciato. I sentiment embedding che sono stati usati sono invece quelli generati con K-Nearest Neighbors, sia per creare il modello, sia per la successiva fase di test. Una volta terminata la fase di training del modello, si è proceduto con la fase di test per valutare con quale accuratezza questo modello sapesse etichettare i dataset non italiani. Questa soluzione, che purtroppo non ha fornito i risultati sperati (come si può verificare guardando i dati raccolti nell'apposita sezione del Capitolo 4), avrebbe permesso di risparmiare molto tempo, potendosi concentrare sulla realizzazione di un solo modello su cui condurre esperimenti basato sui tantissimi dati disponibili in inglese, potendo poi contare sulla possibilità di utilizzarlo in modo generalizzato su più lingue. L'attenzione a quel punto si sarebbe potuta rivolgere alla generazione di nuovi word e sentiment embeddings, o alla definizione di nuove reti neurali.

### 3.2.2 Raccolta statistiche su word e sentiment embedding

In questa sezione sono riportate alcune statistiche che sono state raccolte riguardo il numero di parole che abbiano il corrispettivo word embedding e sentiment embedding. Come spiegato nella sezione dedicata alla replica dei risultati ottenuti da Dong e De Melo in cui si è spiegato in quale modo è organizzata la fase di pre-processing dei dati, non tutte le parole che si possono trovare all'interno dei dataset hanno il corrispondente embedding. Queste parole, per cui gli embedding non sono stati generati, sono trasformate in vettori nulli. L'obiettivo di questa analisi statistica è capire quante parole non abbiano un embedding, e se in qualche modo questa mancanza possa avere un impatto sulla capacità del modello di classificare i dati di input. Nelle tabella seguente sono riportate le statistiche raccolte, così organizzate:

per ogni dataset contenente recensioni (quindi dataset forniti da Dong e De Melo e i due nuovi dataset italiani) sono state calcolate il numero di parole (token) contenute, il numero di word embedding non trovati e il relativo valore percentuale rispetto al totale, e il numero di sentiment embedding non trovati, anch'essi accompagnati dal valore percentuale rispetto al totale. Nel caso dei sentiment embedding, il conteggio è stato ripetuto per ogni tipologia di embedding utilizzata. In particolare quelli generati con altri algoritmi rispetto a quanto fatto in [1] e [5] hanno avuto gli stessi risultati e sono quindi stati accorpati nella stessa riga della tabella (Altri SE).

| | **en** | | **es** | | **nl** | | **ru** | |
|---|---|---|---|---|---|---|---|---|
| **token tot** | 246,415 | | 42,454 | | 28,072 | | 179,603 | |
| **WE** | 6,525 | 2.65% | 2,316 | 5.46% | 1,189 | 4.24% | 6,238 | 3.47% |
| **SE DDM** | 10,713 | 4.35% | 5,876 | 13.84% | 5,911 | 21.06% | 36,290 | 20.21% |
| **SE GD** | 7,546 | 3.06% | 15,942 | 37.55% | 14,605 | 52.03% | 122,970 | 68.47% |
| **Altri SE** | | | 2,316 | 5.46% | 1,189 | 4.24% | 6,238 | 3.47% |

| | **de** | | **cs** | | **it** | | **fr** | |
|---|---|---|---|---|---|---|---|---|
| **token tot** | 119,825 | | 84,132 | | 162,085 | | 144,107 | |
| **WE** | 3,325 | 2.77% | 5,882 | 6.99% | 5,530 | 3.41% | 11,520 | 7.99% |
| **SE DDM** | 34,008 | 28.38% | 35,229 | 41.87% | 24,157 | 14.90% | 21,347 | 14.81% |
| **SE GD** | 61,944 | 51.70% | 56,890 | 67.62% | 70,049 | 43.22% | 78,823 | 54.70% |
| **Altri SE** | 3,325 | 2.77% | 5,882 | 6.99% | 5,530 | 3.41% | 11,520 | 7.99% |

| | **$IT_1$** | | **$IT_2$** | |
|---|---|---|---|---|
| **token tot** | 1,167,711 | | 1,892,690 | |
| **WE** | 44,157 | 3.78% | 78,403 | 4.14% |
| **SE DDM** | 169,405 | 14.51% | 268,073 | 14.16% |
| **SE GD** | 538,815 | 46.14% | 838,485 | 44.30% |
| **Altri SE** | 44,157 | 3.78% | 78,403 | 4.14% |

**Tabella 3.4:** Statistiche riguardanti il numero di parole contenute in ogni dataset di recensioni, il cui word o sentiment embedding non è presente nel dizionario degli embeddings. WE (word embedding non trovati), SE DDM (sentiment embedding non trovati tra quelli forniti da Dong e De Melo), SE GD (sentiment embedding non trovati tra quelli generati con il metodo presentato in [5]), Altri SE (sentiment embedding non trovati prendendo in considerazione quelli generati con KNN, LR, RF, MLP)

Valutando i risultati ottenuti con la rete DC-CNN (che sono riportati nel Capitolo 4) si può vedere come la copertura offerta da parte degli embedding (cioè per quante parola esista il corrispondente embedding) non influisca sui risultati ottenuti. Eppure dalla tabella 3.4 si vede chiaramente che le differenze possono essere anche piuttosto marcate in termini di sentiment embedding. Non è possibile quindi stabilire una relazione certa tra il numero di embedding a disposizione e la capacità del modello di predire correttamente la categoria del testo. Si possono avanzare due tipi di ragionamento: da una parte, un numero maggiore di embedding, che dia la possibilità di coprire tutte le parole presenti nei dataset, permetterebbe di non perdere alcuna informazione, aiutando il modello a funzionare ancora meglio; dall'altra parte, bisogna considerare che non tutte le parole esprimono un sentiment, anzi, la maggior parte potremmo definirle parole neutre, che si limitano a indicare oggetti o situazioni. Nella sentiment analysis ciò che aiuta a definire il sentiment di un testo sono quelle parole che portino nel loro significato una opinione di chi le usa. In questo senso, avere un numero limitato di embedding, ma che coprano tutte quelle parole alle quali sia possibile assegnare delle informazioni in termini di sentiment, si può definire una ottimizzazione. Sarebbe infatti inutile avere a disposizione gli embedding di parole il cui contributo alla definizione del sentiment di un testo sia irrilevante.

### 3.2.3 Implementazione rete LSTM

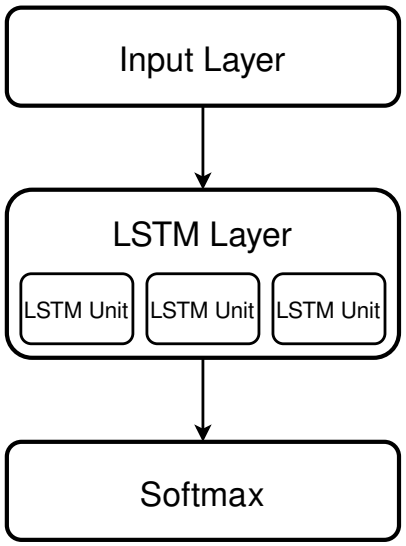


**Figura 3.2:** Architettura della rete Long Short-Term Memory

Gli ultimi test sono stati effettuati utilizzando una rete Long Short-Term Memory (LSTM) [17] al posto della DC-CNN utilizzata per tutte le prove precedenti. La rete LSTM è una evoluzione delle reti RNN (Recurrent Neural Network) caratterizzata dalla capacità di imparare dipendenze tra dati temporalmente distanti tra loro. Con le altre reti neurali, ogni dato di input viene gestito come un'entità separata, ma non sempre questo approccio può definirsi corretto. Nel caso specifico di un testo, come nel nostro caso, ogni parola non può essere correttamente interpretata se presa in esame singolarmente; il suo significato può essere compreso solo tenendo in considerazione l'intera frase in cui la parola è inserita. Una rete LSTM permette di analizzare un testo, tenendo in considerazione proprio questo rapporto tra parole, sfruttando la memoria di cui è dotata per memorizzare i dati ricevuti in precedenza. Scendendo maggiormente nei dettagli, una rete LSTM è composta da una catena di moduli (chiamati hidden units o recurrent units), ognuno dei quali consiste di tre gate (Forget Gate, Input Gate e Output Gate). Il componente chiave di una rete LSTM è il *cell state vector*, una sorta di nastro trasportatore che attraversa tutti i moduli della rete, e che rappresenta la memoria della rete stessa. I tre gate nominati in precedenza si occupano di decidere quali informazioni aggiungere o eliminare dal *cell state vector*. In particolare:

- Il Forget Gate decide quali dati eliminare dalla memoria, e quanto il *cell state vector* dovrebbe ricordare.

- L'Input Gate decide quali informazioni contenute nei dati di input attuali devono essere aggiunti alla memoria della rete
- L'Output Gate decide quali informazioni contenute nel *cell state vector* comprendere nell'output della rete.

La rete LSTM è stata usata come blocco principale all'interno di una struttura che comprende un layer di Input e un fully connected layer di output nel quale la funzione di attivazione *softmax* determina la categoria di appartenenza (positivo, negativo) del testo fornito in ingresso.

I test sono stati effettuati con i dataset multilingua forniti da Dong e De Melo e con i dataset di Twitter, utilizzando tutti i diversi sentiment embedding utilizzati anche per le prove eseguite in precedenza. In questo caso però, data la nuova struttura del modello in uso, il formato dei dati in ingresso e i parametri della rete differiscono rispetto alla rete DC-CNN. I testi di input (recensioni o tweet) anche in questo caso sono stati trasformati nei rispettivi embedding, ma questa volta il vettore con cui ogni parola è stata rappresentata è stato creato concatenando i relativi word embedding e sentiment embedding. La dimensionalità di ognuno di questi vettori è quindi 326 (300 di word embedding + 26 di sentiment embedding). La rete LSTM è caratterizzata da diversi parametri regolabili, tra i quali è di particolare importanza il numero di *recursive units* contenute nel layer LSTM, numero che corrisponde anche alla dimensione dell'output del layer stesso. Nei test eseguiti questo valore (che dunque indica anche la dimensionalità del vettore di output generato dalla rete LSTM) è stato impostato a 50. Questo parametro è stato scelto effettuando alcuni rapidi test con diversi valori tra quelli più usati in altri lavori, come in [18], ed è risultato ottenere risultati leggermente migliori. Come verrà spiegato nel capitolo riservato ai risultati, la selezione di questo parametro è stata frutto di test molto grossolani, utili ad ottenere dei primi risultati per verificare le capacità della rete, ma che necessita di ulteriori affinamenti con una selezione del parametro sulla base di test più approfonditi.

# Capitolo 4

# Risultati e Discussione

In questo capitolo sono riportate le tabelle contenenti i risultati ottenuti, suddivisi in base ai Dataset usati. Nelle ultime sezione del capitolo sono invece riportati i risultati ottenuti con una prova di Transfer Learning, e con la rete LSTM.

## 4.1 Obiettivi della Sperimentazione

Gli obiettivi che si vogliono raggiungere con questa tesi sono riportati nel seguente elenco:

- replica della rete Dual-Channel Convolutional Neural Network e dei risultati in termini di accuratezza ottenuti da Dong e De Melo in [1];
- miglioramento dei risultati di accuratezza tramite l'uso di sentiment embedding generati con algoritmi diversi da quanto proposto in [1];
- aggiunta di nuovi dataset che siano migliori di quelli forniti da Dong e De Melo, o che riguardino nuovi contesti (che non siano cioè composti da recensioni), così da analizzare la scalabilità del modello e degli embedding;
- realizzazione di un modello per il Transfer Learning;
- verifica delle capacità di classificazione di architetture di rete diverse dalla DC-CNN proposta da Dong e De Melo;

## 4.2 Design della Sperimentazione

Per eseguire tutti i test è stata utilizzata una macchina equipaggiata con Intel® Xeon® X5650, dotata di 32 GB di RAM e sistema operativo Ubuntu 18.04.1 LTS.

I dati contenuti nei dataset sono stati divisi in *train set* e *test set* con una proporzione rispettivamente dell'80% e del 20%, mentre non si è fatto uso di alcun tipo di validation set poiché i migliori parametri della rete sono già stati definiti da Dong e De Melo per quanto riguarda la rete DC-CNN, mentre per i test eseguiti con la rete LSTM come già detto nella sezione dedicata non è stata effettuata una validazione dei parametri scelti. La fase di training è stata eseguita per tutti i test utilizzando la tecnica di *early stopping* per decidere dopo quante epoche terminare questa fase. L'*early stopping* è stato impostato per terminare la fase di training quando la loss smette di decrescere per tre epoche successive, considerando il valore 0.01 come soglia della decrescita (cioè la loss viene considerata stabile se non diminuisce tra un'epoca e le successive di almeno 0.01).

Il codice utilizzato è stato scritto interamente da noi, facendo uso di due librerie per implementare alcune funzionalità: *Keras*, che è stato utilizzato per definire la struttura della rete (sia DC-CNN che LSTM), utilizzando TensorFlow come backend, e *scikit-learn*.

Il tempo richiesto per l'esecuzione di un'epoca della fase di training usando la rete DC-CNN corrisponde a circa 10 secondi con i dataset multilingua di Dong e De Melo, qualche decina di secondi (circa 50) con i dataset Twitter, e tra i 6 e gli 11 minuti con i nuovi dataset italiani. La rete LSTM è, come atteso, leggermente più veloce, richiedendo circa 6 secondi con i dataset di Dong e De Melo, e circa 30 secondi con i dataset Twitter.

## 4.3 Risultati Dataset multilingua di Dong e De Melo

In questa sezione saranno presentati i risultati riguardanti i dataset forniti da Dong e De Melo. Dato l'obiettivo di replicare [1], nella tabella sottostante è riportato un confronto tra i risultati ottenuti da Dong e De Melo, e quelli ottenuti da noi con la stessa struttura di rete e stessi embedding.

| | **en** | **es** | **nl** | **ru** | **de** | **cs** |
|---|---|---|---|---|---|---|
| **DDM** | 0.8949 | 0.8593 | 0.7930 | 0.9326 | 0.9231 | 0.9369 |
| **Nostri risultati** | 0.9008 | 0.8477 | 0.8391 | 0.9356 | 0.9398 | 0.9248 |
| **Differenza** | -0.59 % | 1.16 % | -4.61 % | -0.3 % | -1.67 % | 1.21 % |
| | **it** | **fr** | | | | |
| **DDM** | 0.9648 | 0.9297 | | | | |
| **Nostri risultati** | 0.9663 | 0.9068 | | | | |
| **Differenza** | -0.15 % | 2.29 % | | | | |

**Tabella 4.1:** Confronto tra i risultati da noi ottenuti e quelli di Dong e De Melo, a parità di dataset (dataset multilingua usati in [1]) e struttura di rete (DC-CNN).

La metrica utilizzata è ovviamente la stessa usata da Dong e De Melo, cioè l'accuratezza. L'ultima riga della tabella riporta la differenza in termini di percentuale tra i due risultati; un valore negativo significa che i risultati da noi ottenuti sono migliori, se positivo vale il ragionamento opposto. Le differenze sono minime per la maggior parte dei dataset, con l'esclusione dell'olandese e del francese. Per tutti gli altri dataset le differenze sono inferiori al 2%, motivo per cui possiamo ritenerci soddisfatti della nostra implementazione della rete DC-CNN.

Una volta confermata la nostra implementazione della rete DC-CNN, come già spiegato nel Capitolo 3, si è proceduto a provare tutti i diversi sentiment embedding, frutto del lavoro di [5] e dell'uso di altri algoritmi non usati in precedenza. Nelle tabelle che seguono, a ogni colonna corrisponde un dataset (e quindi una lingua) mentre a ogni riga corrispondono sentiment embedding diversi. Saranno presentate due tabelle, una con i valori di Accuratezza e l'altra con i valori Macro f1-score. Come è convenzione fare, i risultati migliori per ogni dataset sono stati sottolineati in grassetto. I dataset forniti da Dong e De Melo, oltre a essere abbastanza piccoli in termini di numero di recensioni contenute, sono molto sbilanciati, con un numero di recensioni positive molto superiore alle recensioni negative. Questo sbilanciamento ha delle ricadute sull'affidabilità delle misurazioni effettuate; per esempio, in un dataset composto dal 90% di recensioni positive, e per il restante 10% da recensioni negative, basterebbe un modello che classifichi tutte le recensioni come positive per ottenere una accuratezza del 90%. Nonostante l'elevato punteggio di accuratezza, il modello non sarebbe però in grado di riconoscere correttamente il sentiment di una recensione negativa e quindi in condizioni realistiche (cioè con una distribuzione di recensioni positive e negative più equa) non otterrebbe buoni risultati. Per questo motivo si è usata anche un'altra misurazione, il Macro F1-score, che pesa diversamente il contributo delle due classi (positiva e negativa) per tenere conto della differenza in termini di frequenza nei dataset analizzati. Il Macro F1-score è così definito:

$$\text{Macro F1-score} = \frac{1}{N}\sum_{i=0}^{N}\text{F1-score}_i \tag{4.1}$$

dove $i$ è l'indice dell'$i$-esima classe, $N$ è il numero totale di classi e l'F1-score è uguale a:

$$\text{F1-score} = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.2}$$

dove *precision* è il numero di veri positivi diviso per il numero di tutti i risultati positivi e *recall* è il numero di veri positivi diviso il numero di veri positivi più falsi negativi. Nonostante questa metrica non possa risolvere un problema che è insito nei dataset utilizzati, può comunque aiutare nel comprendere meglio il comportamento del modello.

| | **en** | **es** | **nl** | **ru** | **de** | **cs** |
|---|---|---|---|---|---|---|
| **DDM** | 0.9008 | **0.8477** | 0.8391 | 0.9356 | 0.9398 | 0.9248 |
| **GD** | **0.9034** | 0.8206 | 0.8311 | 0.9414 | 0.9378 | 0.9228 |
| **KNN 10** | | 0.8223 | 0.8364 | 0.9370 | 0.9336 | 0.9228 |
| **KNN Euclid** | | 0.8274 | 0.8364 | 0.9327 | 0.9357 | 0.9187 |
| **KNN Unif** | | 0.8426 | 0.8338 | 0.9385 | 0.9336 | 0.9167 |
| **LR** | | 0.8376 | 0.8391 | 0.9341 | **0.9440** | **0.9309** |
| **RF 100** | | 0.8291 | 0.8417 | 0.9385 | 0.9398 | 0.9228 |
| **RF 250** | | 0.8156 | **0.8496** | 0.9400 | 0.9357 | 0.9289 |
| **MLP** | | 0.8409 | 0.8391 | **0.9444** | 0.9336 | 0.9146 |
| **MLP 500** | | 0.8342 | 0.8153 | 0.9414 | 0.9253 | 0.9146 |
| **MLP 500 sc** | | 0.8409 | 0.8470 | 0.9341 | 0.9378 | 0.9187 |
| **MLP 1000 sc** | | 0.8291 | 0.8311 | 0.9341 | 0.9378 | 0.9207 |

| | **it** | **fr** |
|---|---|---|
| **DDM** | 0.9663 | 0.9068 |
| **GD** | **0.9747** | 0.9029 |
| **KNN 10** | 0.9677 | 0.9055 |
| **KNN Euclid** | 0.9691 | 0.9080 |
| **KNN Unif** | 0.9649 | **0.9093** |
| **LR** | 0.9705 | 0.9042 |
| **RF 100** | 0.9691 | 0.8927 |
| **RF 250** | 0.9705 | 0.9029 |
| **MLP** | 0.9719 | 0.9029 |
| **MLP 500** | 0.9607 | 0.9017 |
| **MLP 500 sc** | 0.9677 | 0.9029 |
| **MLP 1000 sc** | 0.9677 | 0.9004 |

**Tabella 4.2:** Accuratezza ottenuta con i Dataset multilingua di Dong e De Melo: Inglese (en), Spagnolo (es), Olandese (nl), Russo (ru), Tedesco (de), Ceco (cs), Italiano (it), Francese (fr). I sentiment embedding usati sono: generati da Dong e De Melo in [1] (DDM), generati con Gradient Descent secondo il lavoro [5] (GD), generati con l'algoritmo K-Nearest Neighbors con k=10 (KNN 10, KNN Euclid, KNN Unif), generati con un regressore lineare (LR), generati con l'algoritmo Random Forest con 100 alberi di decisione (RF 100) o 250 alberi di decisione (RF 250), generati con Multilayer Perceptron (MLP), con 500 unità nell'hidden layer (MLP 500), con 500 unità nell'hidden layer e min-max scaling applicato all'input (MLP 500 sc), con 1000 unità nell'hidden layer e min-max scaling applicato all'input (MLP 1000 sc).

Valutando i risultati in termini di accuratezza della classificazione effettuata dal modello, si può notare come i risultati migliori non siano stati ottenuti con uno specifico algoritmo di generazione dei sentiment embedding, neanche considerandone

le varie famiglie di appartenenza (ad esempio accorpando tutti gli embedding generati con KNN o MLP). Ogni dataset sembra preferire embedding diversi, anche se le differenze sono comunque minime. L'aspetto positivo di questa analisi è che in generale i risultati migliori sono stati ottenuti utilizzando sentiment embedding generati con tecniche diverse rispetto a quanto fatto da Dong e De Melo, e che quindi il lavoro svolto in [5] si è rivelato molto valido anche con l'uso della rete DC-CNN. L'unica eccezione è rappresentata dal dataset spagnolo, che ha ottenuto il risultato migliore utilizzando i sentiment embedding generati in [1].

| | **en** | **es** | **nl** | **ru** | **de** | **cs** |
|---|---|---|---|---|---|---|
| **DDM** | 0.8970 | 0.7571 | 0.8270 | 0.9145 | 0.9164 | 0.9133 |
| **GD** | **0.8986** | 0.7456 | 0.8105 | 0.9240 | 0.9138 | 0.9096 |
| **KNN 10** | | 0.7460 | 0.8217 | 0.9179 | 0.9081 | 0.9120 |
| **KNN Euclid** | | 0.7499 | 0.8198 | 0.9110 | 0.9107 | 0.9055 |
| **KNN Unif** | | 0.7634 | 0.8185 | 0.9193 | 0.9086 | 0.9022 |
| **LR** | | 0.7338 | 0.8248 | 0.9127 | **0.9222** | **0.9194** |
| **RF 100** | | 0.7335 | 0.8285 | 0.9193 | 0.9179 | 0.9105 |
| **RF 250** | | 0.7538 | **0.8406** | 0.9214 | 0.9066 | 0.9188 |
| **MLP** | | **0.7657** | 0.8270 | **0.9278** | 0.9058 | 0.9034 |
| **MLP 500** | | 0.7485 | 0.7999 | 0.9235 | 0.8972 | 0.9024 |
| **MLP 500 sc** | | 0.7616 | 0.8376 | 0.9140 | 0.9153 | 0.9061 |
| **MLP 1000 sc** | | 0.7475 | 0.8171 | 0.9134 | 0.9105 | 0.9086 |

| | **it** | **fr** |
|---|---|---|
| **DDM** | 0.9464 | 0.9061 |
| **GD** | **0.9593** | 0.9026 |
| **KNN 10** | 0.9470 | 0.9049 |
| **KNN Euclid** | 0.9497 | 0.9075 |
| **KNN Unif** | 0.9437 | **0.9088** |
| **LR** | 0.9527 | 0.9038 |
| **RF 100** | 0.9508 | 0.8925 |
| **RF 250** | 0.9521 | 0.9023 |
| **MLP** | 0.9551 | 0.9020 |
| **MLP 500** | 0.9364 | 0.9009 |
| **MLP 500 sc** | 0.9479 | 0.9023 |
| **MLP 1000 sc** | 0.9482 | 0.8997 |

**Tabella 4.3:** Macro F1-score ottenuta con i Dataset multilingua di Dong e De Melo: Inglese (en), Spagnolo (es), Olandese (nl), Russo (ru), Tedesco (de), Ceco (cs), Italiano (it), Francese (fr). Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

Anche i risultati ottenuti in termini di Macro f1-score sono praticamente identici a quelli ottenuti misurando l'accuratezza se consideriamo quali sentiment embedding

abbiano funzionato meglio a seconda del dataset considerato. Solo il dataset spagnolo ha subito una variazione in questo senso, ottenendo il risultato migliore con un algoritmo diverso rispetto a quanto fatto considerando l'accuratezza. I valori numerici, che anche nel caso del Macro F1-score così come con l'accuratezza, sono compresi tra 0 e 1, dove un valore maggiore è indice di un risultato migliore, sono leggermente inferiori rispetto ai rispettivi valori di accuratezza. Questa discrepanza, seppur piccola, sottolinea come il modello non sia in grado di classificare con la stessa precisione entrambe le categorie (positivo e negativo) ma che fatichi maggiormente con le recensioni negative di cui dispone di un numero inferiore di esempi da cui imparare in fase di training.

In generale, considerando entrambe le metriche (Accuratezza e Macro F1-score), i risultati si possono considerare molto buoni, vicini e spesso superiori ai risultati di Dong e De Melo che rappresentano lo stato dell'arte.

## 4.4 Risultati Dataset Italiani

In questa sezione sono presentate le stesse due tabelle presenti nella sezione precedente, ma con i risultati ottenuti utilizzando i due Dataset in lingua Italiana realizzati al Politecnico di Torino.

| | **IT$_1$** | **IT$_2$** |
|---|---|---|
| **DDM** | 0.9596 | 0.9615 |
| **GD** | 0.9621 | 0.9654 |
| **KNN 10** | 0.9611 | 0.9611 |
| **KNN Euclid** | **0.9631** | 0.9629 |
| **KNN Unif** | 0.9626 | **0.9665** |
| **LR** | 0.9561 | 0.9622 |
| **RF 100** | 0.9616 | 0.9553 |
| **RF 250** | 0.9601 | 0.9618 |
| **MLP** | 0.9606 | 0.9651 |
| **MLP 500** | 0.9601 | 0.9604 |
| **MLP 500 sc** | 0.9626 | 0.9658 |
| **MLP 1000 sc** | 0.9626 | 0.9636 |

**Tabella 4.4:** Accuratezza ottenuta con i nuovi dataset italiani. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

| | $\mathbf{IT}_1$ | $\mathbf{IT}_2$ |
|---|---|---|
| **DDM** | 0.9596 | 0.9615 |
| **GD** | 0.9621 | 0.9654 |
| **KNN 10** | 0.9611 | 0.9611 |
| **KNN Euclid** | **0.9631** | 0.9629 |
| **KNN Unif** | 0.9626 | **0.9665** |
| **LR** | 0.9561 | 0.9622 |
| **RF 100** | 0.9616 | 0.9553 |
| **RF 250** | 0.9601 | 0.9618 |
| **MLP** | 0.9606 | 0.9651 |
| **MLP 500** | 0.9601 | 0.9604 |
| **MLP 500 sc** | 0.9626 | 0.9658 |
| **MLP 1000 sc** | 0.9626 | 0.9636 |

**Tabella 4.5:** Macro F1-score ottenuto con i nuovi dataset italiani. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

Con questi dataset, che come detto sono perfettamente bilanciati, si può notare come i risultati in termini di accuratezza e macro F1-score siano identici. I valori ottenuti sono ottimi (96% di accuratezza per entrambi i dataset). Se confrontato con i risultati ottenuti con il dataset italiano di Dong e De Melo (considerando i rispettivi risultati migliori), in termini di accuratezza il risultato è inferiore di circa un punto percentuale, mentre il risultato del macro F1-score è migliore ancora una volta di circa un punto percentuale. Possiamo dire quindi che i risultati sono paragonabili in termini di puri valori numerici, ma ricordando che questi due nuovi dataset italiani contengono molte più recensioni e sono bilanciati nella frequenza con cui le due classi (positiva e negativa) compaiono al loro interno, sono sicuramente preferibili al dataset fornito da Dong e De Melo. Il modello allenato sui nuovi dataset ha dimostrato infatti di essere più robusto nella classificazione di recensioni negative, testimoniato da un punteggio macro F1 migliore.

## 4.5 Risultati Dataset Twitter

In questa sezione sono riportati i risultati ottenuti con i dataset che raccolgono tweet, la cui composizione è stata spiegata nel dettaglio nel corso del Capitolo 3. Anche in questo caso i valori riportati si riferiscono prima all'accuratezza della classificazione, e poi al macro F1-score.

| | **de-players** | **de-teams** | **es-teams** | **fr-teams** | **it-players** |
|---|---|---|---|---|---|
| **DDM** | 0.8983 | 0.9378 | 0.9114 | 0.8916 | 0.9329 |
| **GD** | 0.8945 | 0.9359 | **0.9210** | 0.8947 | 0.9314 |
| **KNN 10** | 0.8977 | 0.9386 | 0.9127 | 0.8947 | **0.9352** |
| **KNN Eucl** | 0.8988 | **0.9414** | 0.9127 | **0.9032** | 0.9330 |
| **KNN Unif** | 0.8961 | 0.9370 | 0.9156 | 0.8990 | 0.9337 |
| **LR** | 0.8943 | 0.9370 | 0.9202 | 0.8959 | 0.9334 |
| **RF 100** | 0.8959 | 0.9351 | 0.9114 | 0.8959 | 0.9316 |
| **RF 250** | 0.8959 | 0.9367 | 0.9119 | 0.8959 | 0.9339 |
| **MLP** | 0.8935 | 0.9356 | 0.9095 | 0.8965 | 0.9312 |
| **MLP 500** | **0.8993** | 0.9329 | 0.9139 | 0.8965 | 0.9311 |
| **MLP 500 sc** | 0.8945 | 0.9331 | 0.9175 | 0.8947 | 0.9241 |
| **MLP 1000 sc** | 0.8818 | 0.9381 | 0.9178 | 0.8990 | 0.9266 |

**Tabella 4.6:** Accuratezza ottenuta con dataset Twitter. teams: tweet riguardanti le squadre, players: tweet riguardanti i giocatori, de: tweet in tedesco, es: tweet in spagnolo, fr: tweet in francese, it: tweet in italiano. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

| | **de-players** | **de-teams** | **es-teams** | **fr-teams** | **it-players** |
|---|---|---|---|---|---|
| **DDM** | 0.8885 | 0.9245 | 0.9113 | 0.8565 | 0.9044 |
| **GD** | 0.8864 | 0.9224 | **0.9207** | 0.8579 | 0.8992 |
| **KNN 10** | 0.8873 | 0.9245 | 0.9120 | 0.8606 | **0.9067** |
| **KNN Eucl** | **0.8908** | **0.9273** | 0.9121 | **0.8662** | 0.9022 |
| **KNN Unif** | 0.8874 | 0.9238 | 0.9150 | 0.8628 | 0.9045 |
| **LR** | 0.8866 | 0.9226 | 0.9199 | 0.8503 | 0.9034 |
| **RF 100** | 0.8863 | 0.9214 | 0.9110 | 0.8512 | 0.9021 |
| **RF 250** | 0.8871 | 0.9214 | 0.9118 | 0.8571 | 0.9013 |
| **MLP** | 0.8816 | 0.9213 | 0.9086 | 0.8511 | 0.9006 |
| **MLP 500** | 0.8886 | 0.9192 | 0.9135 | 0.8581 | 0.8987 |
| **MLP 500 sc** | 0.8834 | 0.9200 | 0.9170 | 0.8595 | 0.8963 |
| **MLP 1000 sc** | 0.8752 | 0.9251 | 0.9171 | 0.8609 | 0.8981 |

**Tabella 4.7:** Macro F1-score ottenuto con dataset Twitter. teams: tweet riguardanti le squadre, players: tweet riguardanti i giocatori, de: tweet in tedesco, es: tweet in spagnolo, fr: tweet in francese, it: tweet in italiano. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

I dataset di Twitter sono un ottimo banco di prova per verificare la capacità della rete e degli embedding di generalizzare su contesti diversi. La struttura, in particolare la lunghezza, e i contenuti di un tweet si differenziano moltissimo dalle recensioni che sono state utilizzate per tutti gli altri dataset considerati fino a

questo punto. Purtroppo non esistono altri lavori che abbiano utilizzato lo stesso modello e gli stessi embedding con cui confrontare i nostri risultati così come fatto con Dong e De Melo. Nonostante questo, i risultati si possono definire molto interessanti; i valori di accuratezza e macro F1-score non sono così lontani da quelli ottenuti con dataset di recensioni, confermando quindi la bontà del modello e degli embedding di fornire ottimi risultati anche in contesti diversi. Anche questi dataset sono piuttosto sbilanciati, contenendo molti più tweet positivi di quelli negativi, e per questo è possibile notare delle piccole variazioni tra i valori di accuratezza e macro F1-score. In questo caso però, a differenza di quanto avvenuto con i dataset di Dong e De Melo, è possibile definire dei sentiment embedding che in generale abbiano dato i risultati migliori, ancor più evidente guardando la tabella contenente i macro F1-score. Questi sentiment embedding sono stati generati con l'algoritmo K-Nearest Neighbors, che è lo stesso algoritmo con cui si sono ottenuti i risultati migliori anche per i nuovi dataset italiani.

## 4.6 Risultati ottenuti con Transfer Learning

Di seguito sono riportati i valori di accuratezza e macro F1 ottenuti con una prova di Transfer Learning.

| | **es** | **nl** | **ru** | **de** | **cs** | **it** | **fr** |
|---|---|---|---|---|---|---|---|
| **Accuracy** | 0.5685 | 0.6121 | 0.6296 | 0.6680 | 0.7053 | 0.9522 | 0.6398 |
| **Macro F1** | 0.5500 | 0.6113 | 0.6263 | 0.6543 | 0.7044 | 0.9282 | 0.6087 |

**Tabella 4.8:** Risultati ottenuti con Transfer Learning, usando i Sentiment Embedding generati con K-Nearest Neighbors.

Basta paragonare questi risultati con quelli delle tabelle 4.2 e 4.3 per capire che siano risultati molto deludenti e ben lontani da quanto è possibile ottenere utilizzando un modello specifico per ogni lingua. L'unico dataset che ha mantenuto il valore di accuratezza atteso è stato quello italiano, dovuto al fatto che come spiegato nel Capitolo 3 il dataset utilizzato per la fase di training del modello sia stato $IT_1$ e quindi anch'esso in lingua italiana. Da questo test ci si aspettavano risultati migliori, considerando che gli embedding delle diverse lingue dovrebbero essere allineati. Evidentemente i word embedding pre-trained di cui si è fatto uso non sono perfettamente allineati, da qui la discrepanza tra i risultati ottenuti, e i risultati che ci si aspettava (vicini cioè a quanto ottenuto con un modello adattato per ogni lingua).

## 4.7 Risultati ottenuti con rete LSTM

In questa sezione sono riportati i risultati ottenuti utilizzando una rete LSTM al posto della DC-CNN usata in tutti i test precedenti.

### 4.7.1 Risultati con dataset multilingua di Dong e De Melo

Nelle tabelle seguenti, ricalcando quanto fatto fino a questo punto con tutti i test precedenti, sono riportati i valori di accuratezza e macro F1-score ottenuti con i dataset di Dong e De Melo e i diversi sentiment embedding. Per questi test sono stati usati i seguenti Learning Rate:

- Inglese, Italiano, Tedesco e Russo con Learning Rate = 0.0004
- Ceco con Learning Rate = 0.001
- Francese con Learning Rate = 0.0008 (tranne con sentiment embeddings KNN Uniform dove Learning Rate = 0.0005)

Da notare che per i dataset in lingua olandese e spagnola non è stato possibile trovare un valore di Learning Rate che permettesse al modello di imparare dai dati di training e fornire risultati accettabili.

| | **en** | **ru** | **de** | **cs** | **it** | **fr** |
|---|---|---|---|---|---|---|
| **DDM** | 0.8723 | 0.9283 | 0.9087 | 0.9045 | 0.9593 | 0.8429 |
| **GD** | **0.8790** | 0.9268 | 0.8776 | 0.8902 | 0.9466 | 0.8084 |
| **KNN 10** | | 0.9209 | 0.9212 | 0.8902 | 0.9537 | 0.8263 |
| **KNN Euclid** | | 0.9253 | **0.9253** | **0.9085** | 0.9565 | 0.8314 |
| **KNN Unif** | | 0.8902 | 0.9191 | 0.8862 | 0.9480 | 0.8340 |
| **LR** | | 0.9195 | 0.8983 | 0.8801 | 0.9579 | 0.8148 |
| **RF 100** | | 0.9195 | 0.9025 | 0.8801 | 0.9424 | 0.8544 |
| **RF 250** | | 0.9195 | 0.8983 | 0.8841 | 0.9424 | 0.7816 |
| **MLP** | | **0.9327** | 0.9066 | 0.8760 | **0.9607** | **0.8608** |
| **MLP 500** | | 0.9312 | 0.9046 | 0.9024 | 0.9508 | 0.8544 |
| **MLP 500 sc** | | 0.8243 | 0.9066 | 0.8821 | 0.9579 | 0.8238 |
| **MLP 1000 sc** | | 0.9048 | 0.9046 | 0.8963 | 0.9522 | 0.8186 |

**Tabella 4.9:** Accuratezza ottenuta con i Dataset multilingua di Dong e De Melo: Inglese (en), Russo (ru), Tedesco (de), Ceco (cs), Italiano (it), Francese (fr). Rete LSTM con 50 hidden units. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

| | en | ru | de | cs | it | fr |
|---|---|---|---|---|---|---|
| **DDM** | 0.8671 | 0.9078 | 0.8758 | 0.8879 | 0.9324 | 0.8425 |
| **GD** | **0.8731** | 0.9036 | 0.8027 | 0.8715 | 0.9177 | 0.7998 |
| **KNN 10** | | 0.8959 | 0.8874 | 0.8728 | 0.9284 | 0.8257 |
| **KNN Euclid** | | 0.9019 | **0.8947** | **0.8935** | 0.9313 | 0.8302 |
| **KNN Uniform** | | 0.8410 | 0.8848 | 0.8625 | 0.9193 | 0.8331 |
| **LR** | | 0.8938 | 0.8543 | 0.8669 | 0.9330 | 0.8148 |
| **RF 100** | | 0.9012 | 0.8512 | 0.8598 | 0.9136 | 0.8533 |
| **RF 250** | | 0.8942 | 0.8425 | 0.8615 | 0.9132 | 0.7667 |
| **MLP** | | **0.9145** | 0.8586 | 0.8631 | **0.9345** | **0.8589** |
| **MLP 500** | | 0.9142 | 0.8738 | 0.8874 | 0.9220 | 0.8535 |
| **MLP 500 sc** | | 0.8076 | 0.8635 | 0.8674 | 0.9286 | 0.8217 |
| **MLP 1000 sc** | | 0.8832 | 0.8580 | 0.8809 | 0.9252 | 0.8134 |

**Tabella 4.10:** Macro F1-score ottenuto con i Dataset multilingua di Dong e De Melo: Inglese (en), Russo (ru), Tedesco (de), Ceco (cs), Italiano (it), Francese (fr). Rete LSTM con 50 hidden units. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

Così come avvenuto con la rete DC-CNN non è possibile definire dei sentiment embedding che forniscano i risultati migliori con tutti, o la maggior parte, dei dataset, anche se si può notare come i sentiment embedding generati con l'algoritmo MLP permettano di ottenere i risultati migliori su ben tre dataset. I valori non sono molto lontani, seppur inferiori, a quelli ottenuti con l'architettura di rete proposta da Dong e De Melo, e questo si può ritenere un risultato interessante. Bisogna infatti considerare che i risultati ottenuti con la rete DC-CNN, hanno fatto uso di parametri di rete frutto di un lavoro di fine-tuning svolto proprio dagli ideatori della rete stessa; altrettanto non si può dire di questi risultati ottenuti con la rete LSTM dove i parametri della rete non hanno ricevuto alcun lavoro di rifinitura, lasciando quindi spazio a possibili, e anche sensibili, miglioramenti.

### 4.7.2 Risultati con dataset Twitter

Gli stessi test sono stati svolti anche con i dataset di Twitter per verificare se la rete fosse adatta a gestire anche testi diversi dalle recensioni. In questo caso il Learning Rate è stato impostato a 0.0004 come fatto con la maggior parte delle lingue dei dataset di Dong e De Melo, oltre ad essere il valore di learning rate più usato anche per la rete DC-CNN, come suggerito in [1]. Solo nel caso del dataset spagnolo *es-teams* quando sono stati usati i sentiment embedding *KNN Uniform* è stato necessario usare un learning rate diverso a causa della difficoltà della rete ad imparare con il valore impostato per tutti gli altri casi. In questo specifico caso si

è quindi usato Learning Rate = 0.0002, frutto di una rapida ricerca manuale tra possibili valori nell'intorno del valore utilizzato per gli altri casi.

| | **de-players** | **de-teams** | **es-teams** | **fr-teams** | **it-players** |
|---|---|---|---|---|---|
| **DDM** | 0.8551 | 0.9108 | 0.8917 | 0.8579 | 0.9169 |
| **GD** | 0.8772 | 0.9081 | 0.8958 | 0.8634 | 0.9234 |
| **KNN 10** | 0.8748 | 0.9136 | 0.8941 | 0.8714 | 0.9256 |
| **KNN Eucl** | 0.8804 | **0.9246** | 0.8978 | 0.8763 | 0.9234 |
| **KNN Unif** | **0.8866** | 0.9238 | 0.8817 | 0.8671 | 0.9237 |
| **LR** | 0.8751 | 0.8996 | 0.8851 | 0.8634 | 0.9211 |
| **RF 100** | 0.8644 | 0.9056 | 0.8890 | 0.8634 | 0.9198 |
| **RF 250** | 0.8698 | 0.9064 | 0.8897 | 0.8641 | 0.9254 |
| **MLP** | 0.8796 | 0.9169 | 0.8936 | 0.8702 | 0.9193 |
| **MLP 500** | 0.8751 | 0.9097 | **0.8992** | **0.8892** | 0.9023 |
| **MLP 500 sc** | 0.8751 | 0.9216 | 0.8919 | 0.8696 | 0.9163 |
| **MLP 1000 sc** | 0.8626 | 0.9023 | 0.8773 | 0.8659 | **0.9274** |

**Tabella 4.11:** Accuratezza ottenuta con i dataset Twitter, e rete LSTM con 50 hidden units. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

| | **de-players** | **de-teams** | **es-teams** | **fr-teams** | **it-players** |
|---|---|---|---|---|---|
| **DDM** | 0.8462 | 0.8849 | 0.8911 | 0.8105 | 0.8762 |
| **GD** | 0.8606 | 0.8883 | 0.8953 | 0.8241 | 0.8856 |
| **KNN 10** | 0.8603 | 0.8971 | 0.8940 | 0.8206 | 0.8859 |
| **KNN Eucl** | 0.8689 | **0.9081** | 0.8969 | 0.8258 | 0.8828 |
| **KNN Unif** | **0.8736** | 0.9047 | 0.8816 | 0.8076 | 0.8860 |
| **LR** | 0.8632 | 0.8754 | 0.8843 | 0.8007 | 0.8854 |
| **RF 100** | 0.8544 | 0.8832 | 0.8883 | 0.8223 | 0.8752 |
| **RF 250** | 0.8572 | 0.8835 | 0.8889 | 0.8026 | 0.8879 |
| **MLP** | 0.8701 | 0.8940 | 0.8932 | 0.8172 | 0.8739 |
| **MLP 500** | 0.8656 | 0.8913 | **0.8987** | **0.8452** | 0.8664 |
| **MLP 500 sc** | 0.8665 | 0.9049 | 0.8909 | 0.8155 | 0.8783 |
| **MLP 1000 sc** | 0.8532 | 0.8856 | 0.8760 | 0.8127 | **0.8931** |

**Tabella 4.12:** Macro F1-score ottenuto con dataset Twitter e rete LSTM con 50 hidden units. Per la spiegazione dei nomi dei sentiment embedding fare riferimento alla didascalia della tabella 4.2

Osservando i risultati si nota come due "famiglie" di sentiment embedding abbiano ottenuto i risultati migliori: *K-Nearest Neighbors* e *Multilayer Perceptron*. Anche in questo caso, così come accaduto con i dataset di Dong e De Melo, i valori

di accuratezza e macro F1 non sono molto distanti da quelli ottenuti con la rete DC-CNN e anche in questo caso vale lo stesso discorso fatto in precedenza sui parametri della rete LSTM. Con un lavoro di tuning di questi parametri è possibile migliorare i risultati ottenuti, possibilmente portando la rete LSTM a competere con la DC-CNN.

# Capitolo 5

# Sviluppi Futuri

Come detto già nel capitolo introduttivo, la Sentiment Analysis è un campo in continua evoluzione, per cui le possibilità di sviluppo per il futuro sono molteplici. Guardando in particolare al lavoro correlato e svolto in questa tesi, alcune di queste possibilità possono ricercarsi in:

- miglioramento della rete *Dual-Channel Convolutional Neural Network*, apportando modifiche alla struttura della rete così come proposta da Dong e De Melo;
- uso di nuovi modelli di Deep Learning per sostituire la DC-CNN;
- ricerca dei parametri di rete migliori per il modello *Long Short-Term Memory*, attraverso un processo di fine-tuning, con cui poter realizzare un confronto con i risultati della DC-CNN a parità di parametri ottimali per ciascuna delle reti;
- generazione di nuovi embedding, con l'uso di nuovi algoritmi o l'ideazione di nuove metodologie di generazione, per rappresentare in formato vettoriale più informazioni possibili utili alla definizione del sentiment di un testo. In particolare può essere di grande interesse approfondire la qualità degli attuali word embedding e ottenerne versioni ben allineate che permettano la realizzazione di un modello per effettuare transfer learning, così da poter utilizzare un solo modello per molte lingue, sfruttando l'enorme disponibilità di dati di una determinata lingua (normalmente l'inglese) per allenare il modello in questione;
- generazione di embedding per nuove lingue; uno degli obiettivi principali di questo lavoro di tesi è portare la Sentiment Analysis ad ottenere ottimi risultati anche con lingue diverse dall'inglese. Anche se sono stati utilizzati molti dataset in lingue diverse (in particolare 8 lingue in totale), ci sono

ancora molte altre lingue che possono essere prese in considerazione e alle quali molte aziende potrebbero essere interessate, come ad esempio il cinese, giapponese, portoghese, arabo ecc. e che rappresentano una grande porzione della popolazione mondiale. Ovviamente, maggiore è la copertura, in termini di lingue supportate, maggiore è l'utilità che questo tipo di analisi può garantire.

# Capitolo 6

# Conclusione

Con il lavoro svolto per questa tesi si è cercato di replicare la rete Dual-Channel Convolutional Neural Network che attualmente rappresenta lo stato dell'arte per la Sentiment Analysis multilingua, potendo così procedere a verificare la qualità dei sentiment embedding generati con il metodo presentato in [5]. Si sono inoltre aggiunti due dataset italiani di ottima qualità e cinque nuovi dataset contenenti tweet, che hanno permesso di mettere alla prova il modello e gli embedding con testi che non siano recensioni. I risultati ottenuti in tutti questi casi si possono dire molto buoni, soprattutto essendo riusciti spesso a superare quanto fatto da Dong e De Melo a parità di dataset e modello usato, anche se con differenze minime. I risultati ottenuti con la rete *Long Short-Term Memory* infine, dimostrano che altri modelli possono ottenere risultati che si avvicinino molto a quelli ottenuti con la rete DC-CNN, indicando una delle molteplici vie che si possono seguire alla ricerca di risultati ancora migliori rispetto a quelli fino ad ora ottenuti.

# Bibliografia


[1] Xin Dong e Gerard De Melo. «Cross-lingual propagation for deep sentiment analysis». English (US). In: *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*. AAAI press, gen. 2018, pp. 5771–5778 (cit. alle pp. 5, 9, 14, 17, 19–22, 24, 28, 29, 31, 32, 38).

[2] Tomas Mikolov, Kai Chen, Greg Corrado e Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: `1301.3781 [cs.CL]` (cit. alle pp. 6, 8).

[3] C. Hutto e E. Gilbert. «VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text». In: *ICWSM*. 2014 (cit. a p. 6).

[4] William L. Hamilton, Kevin Clark, Jure Leskovec e Dan Jurafsky. «Inducing Domain-Specific Sentiment Lexicons from Unlabeled Corpora». In: *Proceedings of EMNLP*. 2016 (cit. a p. 6).

[5] Flavio Giobergia, Luca Cagliero, Paolo Garza e Elena Baralis. «Cross-Lingual Propagation of Sentiment Information Based on Bilingual Vector Space Alignment». In: *EDBT/ICDT Workshops* (2020) (cit. alle pp. 8, 10, 14, 17, 21, 24, 25, 30–32, 43).

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin e Tomas Mikolov. «Enriching Word Vectors with Subword Information». In: *arXiv preprint arXiv:1607.04606* (2016) (cit. a p. 8).

[7] Mohamed Abdalla e Graeme Hirst. «Cross-lingual sentiment analysis without (good) translation». In: *arXiv preprint arXiv:1707.01626* (2017) (cit. alle pp. 10, 11, 14).

[8] Tomas Mikolov, Quoc V. Le e Ilya Sutskever. *Exploiting Similarities among Languages for Machine Translation*. 2013. arXiv: `1309.4168 [cs.CL]` (cit. a p. 11).

[9] Hatem Ghorbel. «Experiments in cross-lingual sentiment analysis in discussion forums». In: *International Conference on Social Informatics*. Springer. 2012, pp. 138–151 (cit. alle pp. 12, 14).

[10] Stefano Baccianella, Andrea Esuli e Fabrizio Sebastiani. «Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining.» In: *Lrec*. Vol. 10. 2010. 2010, pp. 2200–2204 (cit. a p. 12).

[11] Noura Farra. «Cross-Lingual and Low-Resource Sentiment Analysis». Tesi di dott. Columbia University, 2019 (cit. alle pp. 13, 15).

[12] Diederik P. Kingma e Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG] (cit. a p. 19).

[13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng e Christopher Potts. «Recursive deep models for semantic compositionality over a sentiment treebank». In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, pp. 1631–1642 (cit. a p. 19).

[14] Julian McAuley e Jure Leskovec. *From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews*. 2013. arXiv: 1303.4402 [cs.SI] (cit. a p. 19).

[15] Theophile Blard. *french-sentiment-analysis-with-bert*. urlhttps://github.com/TheophileBlard/french-sentiment-analysis-with-bert. 2020 (cit. a p. 20).

[16] Maria Pontiki et al. «Semeval-2016 task 5: Aspect based sentiment analysis». In: *10th International Workshop on Semantic Evaluation (SemEval 2016)*. 2016 (cit. a p. 20).

[17] Sepp Hochreiter e Jürgen Schmidhuber. «Long Short-term Memory». In: *Neural computation* 9 (dic. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735 (cit. a p. 26).

[18] Nils Reimers e Iryna Gurevych. «Optimal hyperparameters for deep lstm-networks for sequence labeling tasks». In: *arXiv preprint arXiv:1707.06799* (2017) (cit. a p. 27).