UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Confronto di metodi non supervisionati per la creazione di corpora di parafrasi da testi di lingua parlata

Candidato: *Lucia Pifferi*

Relatore: *Prof. Felice Dell'Orletta*

Correlatore: *Prof. Alessandro Lenci*

Anno Accademico 2018-2019

Ai miei genitori,
con immensa gratitudine.

Sommario

1. Introduzione

Il presente lavoro ha come argomento un progetto realizzato presso l'Istituto di Linguistica Computazionale del CNR di Pisa, finalizzato alla creazione di un corpus di parafrasi di notizie giornalistiche, partendo da un insieme di 116 trascrizioni radiofoniche. A tale scopo sono state elaborate diverse funzioni per il calcolo della similarità, partendo da quelle che utilizzano semplicemente il lessico senza alcuna competenza linguistica, sino ad arrivare a sfruttare la semantica distribuzionale, utilizzando il *word embedding*. I diversi metodi che verranno presentati sono automatici e *non supervisionati*, in quanto non sono stati utilizzati classificatori, ma funzioni di distanza.

Il lavoro si è articolato principalmente in quattro fasi. Nella prima fase, è stata effettuata l'estrazione delle notizie giornalistiche dalle trascrizioni radiofoniche. Nella seconda fase, si è proceduto con la pulizia dei dati ottenuti, che presentavano ancora le annotazioni circa la radio di provenienza (Radio 1, Radio 2, Radio 3, Rete 105, Radio Italia, Radio Radicale, Radio Vaticana), la tipologia comunicativa degli speaker (monologo, dialogo, telefonata, monologo a più voci, turno frammentato, esecutivo, programmato, semi improvvisato, spontaneo), gli speaker (professionista, esterno, maschio, femmina) e il genere di comunicazione riportata (pubblicità, annunci, letteratura, notizie, intrattenimento culturale, intrattenimento leggero). Al termine di questa seconda fase si sono così ottenuti i corpora che sono stati successivamente oggetto dell'allineamento. Questi primi due step appena citati, rientrano nel processo di *preprocessing* che verrà descritto in maniera più esaustiva nel terzo capitolo.

La fase successiva è consistita nell'elaborazione di diverse funzioni di allineamento sempre più raffinate, al fine di ad accoppiare notizie relative allo stesso argomento. In un primo momento è stata messa a punto una funzione, basata solamente sull'*overlap* di token, che utilizza l'*indice di Jaccard* come indice statistico. Solo in un secondo momento sono state implementate funzioni con crescente competenza linguistica, basate ad esempio sulla sovrapposizione dei lemmi delle *part of speech* rilevanti o sul *word embedding*. Tali funzioni verranno spiegate dettagliatamente nel capitolo quattro.

La quarta e ultima fase ha riguardato la valutazione e l'analisi dei risultati ottenuti dai diversi metodi di allineamento. Si è proceduti con un controllo – dato dalla lettura delle coppie di frasi allineate – degli *output* di ciascuna funzione di similarità, per decretare il metodo di allineamento migliore in termini di correttezza dell'accoppiamento delle frasi relative a uno stesso soggetto. In tal modo, si è individuata la funzione con minor margine di errore nell'allineamento delle frasi e si è potuta utilizzare per creare il nostro corpus di parafrasi. Nel quinto capitolo si traccerà in maniera completa questo processo di analisi, per poi giungere a una riflessione conclusiva nell'ultimo capitolo.

2. Stato dell'arte

Questo capitolo vuole offrire un quadro generale sulle nozioni e sugli studi precedenti relativi ai temi dell'allineamento tra testi e della semantica distribuzionale.

2.1 Allineamento tra testi

Il task dell'allineamento tra corpora consiste nell'associazione di contenuti relativi allo stesso argomento. L'allineamento può esser condotto a diversi livelli di testo e tra corpora che possono differire nella lingua, nella complessità, nel genere ecc. A seconda del diverso grado di specificità dei dati e del tipo di corpus da allineare, questo task è utile per scopi differenti. Si può fare una macro-distinzione tra l'allineamento di corpora multilingue e quello di corpora monolingue, entrambi condotti a livello di *sentence*.

Tra i diversi obiettivi dell'allineamento tra corpora multilingue, vi è la creazione di corpora allineati per l'addestramento di sistemi di *machine traslation*. Infatti, proprio nel contesto della traduzione automatica statistica è stato affrontato per la prima volta il problema dell'allineamento delle frasi. Nel 1993, Gale e Church hanno proposto un algoritmo di programmazione dinamica per l'allineamento a livello di frase nelle traduzioni. Tale algoritmo teneva conto di due fattori: il fatto che la lunghezza delle frasi tradotte corrispondesse approssimativamente alla lunghezza delle frasi originali e il fatto che la sequenza delle frasi nel testo tradotto corrispondesse ampiamente all'ordine originale delle frasi. Con questo approccio semplice hanno raggiunto un alto grado di precisione.

L'allineamento di corpora monolingue è invece necessario al fine di creare *training set* per addestrare sistemi di semplificazione di testi o sistemi di riscrittura di riassunti o parafrasi.
Inizialmente gran parte del lavoro si è concentrata sull'allineamento tra i riassunti e i testi originali. Nel 2002, Jing ha presentato un algoritmo che allineava stringhe di parole a parti del testo originale utilizzando un modello di Markov nascosto. Sempre nell'ambito del miglioramento di sistemi di riscrittura dei testi, rientra il lavoro del 2003 condotto da Barzilay ed Elhadad. Tale studio, ha dimostrato l'efficacia dell'uso

del contesto del documento per l'allineamento del testo. Nelken e Shieber, nel 2006, hanno migliorato il lavoro di Barzilay ed Elhadad pur ribadendo l'importanza del contesto. La loro novità è stata l'introduzione di un nuovo algoritmo di allineamento di frasi monolingue, che combinasse il punteggio TF*IDF[1] (trasformato in una distribuzione di probabilità usando la regressione logistica) come misura di similarità per l'allineamento delle frasi, con un algoritmo di programmazione dinamica di allineamento globale. Questo nuovo approccio ha fornito una soluzione più semplice e robusta con un netto miglioramento dell'accuratezza rispetto ai sistemi esistenti fino a quel momento.

Relativamente allo sviluppo di sistemi di semplificazione di testi, Bott e Saggion, nel 2011, hanno elaborato un algoritmo di allineamento non supervisionato per la costruzione di un corpus parallelo costituito da brevi testi di notizie in spagnolo e la loro controparte semplificata. Se si considerano le persone con handicap linguistici o, più semplicemente, coloro che vogliono imparare la lingua o che hanno esigenze di lettura e comprensione particolari, la percentuale di necessità di semplificazione di testi è stimata attorno al 25% della popolazione (Bott e Saggion, 2011), quindi è di grande importanza lo sviluppo di metodi e strumenti per affrontare questo problema. Tale task, che ha come obiettivo la trasformazione di testi in equivalenti meno complessi nel vocabolario e nella forma, mira a ridurre gli sforzi della semplificazione umana e, allo stesso tempo, permette una elaborazione dei testi più efficiente da parte dei diversi processori di elaborazione del linguaggio naturale (come ad esempio i parser).

Sempre nel 2011, data la crescente attenzione al tema, Coster e Kauchak hanno analizzato un nuovo set di dati derivato dall'allineamento tra la versione in inglese standard e la versione in inglese semplificato di Wikipedia. Si ricorda tale studio perché i dati analizzati sono di un ordine di grandezza maggiore rispetto a qualsiasi

[1] La funzione di peso **tf-idf** (*term frequency–inverse document frequency*) è una funzione utilizzata in information retrieval per misurare l'importanza di un termine rispetto ad un documento o ad una collezione di documenti. Tale funzione aumenta proporzionalmente al numero di volte che il termine è contenuto nel documento, ma cresce in maniera inversamente proporzionale con la frequenza del termine nella collezione.

altro testo precedentemente analizzato per la semplificazione delle frasi e contengono l'intera gamma di operazioni di semplificazione possibili (tra cui la riformulazione, il riordino, l'inserimento e la cancellazione).

Per quanto riguarda l'italiano, nel 2016, è stato realizzato PaCCSS-IT (**Pa**rallel **C**orpus of **C**omplex–**S**imple Aligned **S**entences for **IT**alian), un corpus parallelo composto da frasi semplici e complesse allineate tra loro. Per costruire la risorsa è stato sviluppato un nuovo metodo in grado di acquisire automaticamente un corpus di frasi accoppiate semplici-complesse. Tale approccio è adatto anche per i linguaggi con meno risorse in quanto richiede un vasto quantitativo di testi che possono essere facilmente estratti dal Web. Ad oggi, ad eccezione dell'inglese, PaCCSS-IT è il corpus più grande di frasi semplici e complesse allineate.

2.2 Semantica distribuzionale

Nei modelli di semantica distribuzionale il significato delle parole è rappresentato attraverso l'analisi statistica dei contesti linguistici in cui le parole ricorrono. Due lessemi sono tanto più simili quanto più sono simili le loro distribuzioni contestuali costruite con informazioni estratte da corpora testuali. Il significato delle parole è dunque una proprietà che emerge dall'uso di queste nei diversi contesti linguistici.
Sia in Linguistica Computazionale che nelle Scienze Cognitive, l'approccio distribuzionale all'analisi semantica è stato oggetto di intensa ricerca tanto che è diventato un paradigma centrale nel Trattamento Automatico della Lingua (TAL).
La semantica distribuzionale ha raggiunto oggi una sua maturità grazie alla disponibilità sempre crescente di dati testuali e una maggiore potenza di calcolo. Infatti oggigiorno sono disponibili molti algoritmi per la costruzione di spazi semantici distribuzionali e per la loro valutazione. Nell'ultimo periodo, gli sforzi della ricerca si sono concentrati sullo sviluppo di metodi di ottimizzazione dei modelli distribuzionali per essere poi applicati a corpora di grandi dimensioni e sullo studio e valutazione dei parametri significativi per le rappresentazioni semantiche costruite da tali modelli distribuzionali. Rimangono comunque oggetto di studio e approfondimento metodi, sia per giungere a una migliore comprensione del tipo di informazioni estratte dai dati linguistici, sia per una più ampia applicazione alla modellazione di nuovi fenomeni semantici.

2.2.1 Ipotesi Distribuzionale

Alla base della semantica distribuzionale, vi è l'Ipotesi Distribuzionale, secondo cui due parole sono tanto più semanticamente simili quanto più tendono a ricorrere in contesti simili (Miller e Charles, 1991). Il pioniere di tale ipotesi è ritenuto Zelling S. Harris (Harris, 1954) che considerava tale metodologia distribuzionale come l'unico approccio scientifico per lo studio del significato linguistico. Nei lavori più recenti, Harris raccoglie e analizza le relazioni di dipendenza sintattica, espresse in termini di operatori e argomenti (Harris, 1991), per proporre un metodo in grado di classificare le parole sulla base dei loro contesti. Dagli inizi degli anni Sessanta molte implementazioni dell'Ipotesi Distribuzionale sono state utilizzate per costruire automaticamente *thesauri* (G. Grefenstette, 1994), dizionari privi di definizioni organizzati per campi semantici. Il *vector space model* nell'*Information Retrieval* (Salton e altri, 1975) ha contribuito allo sviluppo della semantica distribuzionale, in quanto ha consentito di migliorare la metodologia originale di Harris riguardo sia alla natura dei dati che alla loro formalizzazione matematica, accelerando la diffusione del paradigma distribuzionale in linguistica computazionale. Negli ultimi venti anni, l'approccio distribuzionale è diventato il paradigma semantico di riferimento nel Trattamento Automatico del Linguaggio grazie alla possibilità di esser applicato su ampia scala in corpora di grandi dimensioni.

2.2.2 I modelli semantici distribuzionali

Nei modelli distribuzionali ogni parola è rappresentata come un vettore costruito a partire dalla sua distribuzione nei contesti linguistici. La similarità tra le parole è considerata la distanza geometrica tra questi vettori.
La costruzione dei modelli semantici distribuzionali si articola tipicamente in quattro fasi:

1. Vengono raccolti e contati i contesti di ciascuna parola target al fine di generare una matrice di co-occorrenza;
2. La frequenza di ogni parola viene trasformata in un peso statistico che riflette l'importanza del contesto;

3. Siccome la matrice ottenuta è molto grande e sparsa (la maggior parte delle sue entrate è zero), vengono applicate tecniche matematiche per ridurre il numero delle sue dimensioni;
4. La similarità semantica delle parole target viene misurata attraverso la similarità dei corrispondenti vettori riga nella matrice.

Per rappresentare il contenuto lessicale nella semantica distribuzionale lo strumento principale di rappresentazione matematica sono i *vettori*. Con *vettore* si intende una lista ordinata di numeri reali ($v_1,\ldots, v_n$), in cui v_i è l'i-esima componente del vettore. Ciascun vettore può essere posizionato su un sistema di assi cartesiani (Figura 1).

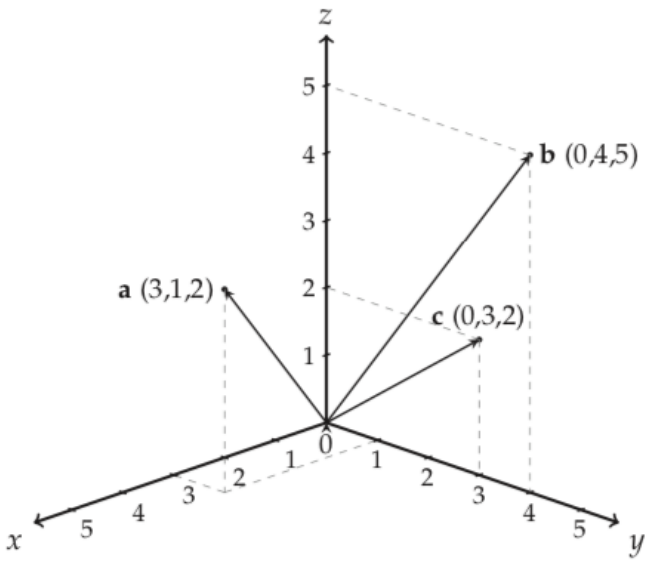

Figura 1. Vettori in uno spazio tridimensionale

I vettori **a**, **b**, **c** sono rappresentati come frecce che congiungono nello spazio un punto origine – ossia l'incrocio degli assi cartesiani - a un punto finale, le cui coordinate corrispondono alle componenti del vettore. In questo modo, i modelli distribuzionali permettono di rappresentare geometricamente il lessico come uno spazio vettoriale semantico.

Per meglio illustrare il concetto di similarità distribuzionale si riporta l'esempio di Lenci (Lenci, 2014). Supponendo di aver contato quante volte i token *auto*, *gatto*, *cane* e *camion* co-occorrono in un corpus con i verbi *mangiare*, *guidare* e *correre*, si ottengono le seguenti distribuzioni di frequenza:

Tabella 1. Distribuzioni di frequenza

	mangiare	guidare	correre
auto	0	3	2
cane	3	0	4
camion	0	2	3

Si possono dunque riportare i seguenti vettori distribuzionali su un piano cartesiano:

auto = (0, 3, 2)
gatto = (4, 0, 3)
cane = (3, 0, 4)
camion = (0, 2, 3)

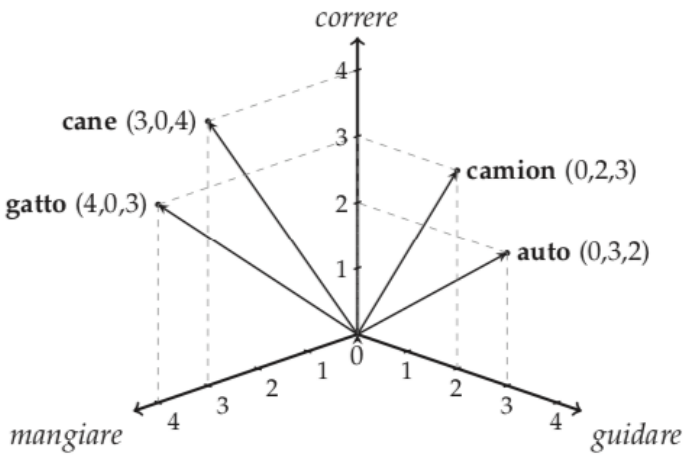

Figura 2. Vettori distribuzionali

L'asse x è etichettato con *mangiare*, l'asse y con *guidare* e l'asse z con *correre*. Infatti, la prima componente di ciascun vettore è la frequenza di co-occorrenza con *mangiare*, la seconda con *guidare* e la terza con *correre*. Se il valore è 0, significa che quel token non ricorre mai in quel contesto.

Una volta riportati i vettori sugli assi, si può calcolare la similarità distribuzionale come il coseno dell'angolo θ tra due vettori:

$$sim_{cos}(\boldsymbol{u}, \boldsymbol{v}) = \cos(\theta) = \frac{\boldsymbol{u} \cdot \boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2}\sqrt{\sum_{i=1}^{n} v_i^2}}$$

Tabella 2. Calcolo della similarità tra vettori

sim_{cos}	auto	gatto	cane	camion
auto	1			
gatto	0.33	1		
cane	0.44	0.96	1	
camion	0.92	0.50	0.66	1

Il coseno è la funzione che permette di calcolare la similarità tra vettori in termini geometrici, sfruttando la loro vicinanza nello spazio. Il valore massimo di similarità è dato quando due vettori sono geometricamente allineati sulla stessa linea e puntano nella medesima direzione, l'angolo tra di loro misura 0 gradi e il coseno è 1. Il caso opposto è dato quando due vettori sono ortogonali, il loro angolo è vicino a 90 gradi e il coseno è 0 (e dunque si ha assenza di similarità). In generale si può sostenere che *più piccolo è l'angolo tra due vettori, maggiore è il coseno e la similarità tra questi*. Infatti, i token che risultano più simili tra di loro sono *cane* e *gatto*, i cui vettori formano un angolo minore rispetto agli altri vettori.

2.2.3 Struttura dei modelli semantici distribuzionali

I modelli distribuzionali possono essere realizzati in diversi modi anche in relazione a diversi parametri in ogni fase del processo di costruzione.

1. Uno di questi parametri è *la dimensione del corpus che viene analizzato*. Mentre negli anni Novanta venivano utilizzati corpora di medie dimensioni, oggigiorno vi è la tendenza a usare corpora sempre più grandi per aumentare la copertura delle risorse lessicali distribuzionali riducendo allo stesso tempo la sparsità dei dati, che può avere effetti negativi sulla qualità degli spazi semantici.
2. Un altro parametro cruciale è la *definizione dei contesti*. Tipicamente vengono utilizzati tre tipi di contesti linguistici:

a. Nei *modelli basati su documenti* (*document models*), le parole sono considerate simili se appaiono negli stessi documenti o negli stessi paragrafi;
b. Nei *modelli basati sulle parole* (*word models*), viene considerata la finestra di parole che si presentano intorno ai lessemi target (Lund, Burgess, 1997; Sahlgren, 2008; Ferret, 2013). In tali modelli vi è un parametro aggiuntivo dato dalla dimensione della finestra per la selezione delle parole contesto;
c. Nei *modelli sintattici* (*syntactic models*) vengono utilizzati come contesti le relazioni di dipendenza sintattica delle parole target (Curran, 2004; Padò, Lapata, 2007; Baroni, Lenci, 2010).

3. Tra i parametri abbiamo anche *i pesi statistici dei contesti* e le *misure di similarità dei vettori*. Esistono diverse possibilità per entrambi, ma vengono attualmente più utilizzati la *Positive Pointwise Mutual Information* come peso statistico per calcolare l'importanza dei contesti e il coseno come misura di similarità semantica.
4. Infine vi sono diverse *tecniche di riduzione delle dimensioni dei vettori* per limitare la complessità computazionale. Le dimensioni del vettore corrispondono a ciascun contesto specifico in cui la parola target è osservata.

Attualmente molte ricerche sono finalizzate alla comprensione dell'impatto di questi parametri sulle performance dei modelli distribuzionali. Tra gli ultimi studi si citano quelli di Lapesa, Evert (2014) e Kiela, Clark (2014) che esaminano la vasta gamma di parametri elencati per individuare le configurazioni migliori per i modelli distribuzionali.

2.2.4 Count models e prediction models

I modelli descritti finora sono definiti *count models* in quanto si basano sul conteggio delle frequenze delle parole nei testi. Vi è però anche una nuova famiglia di modelli distribuzionali – i *prediction models* - basata su una tecnica di predizione: vengono sfruttati algoritmi neurali che creano direttamente rappresentazioni distribuzionali dense e a bassa dimensionalità, imparando a predire in maniera ottimale i contesti di una parola target (Mikolov e altri, 2013). Le rappresentazioni costruite da tali modelli

vengono chiamate *embedding*, in quanto le parole sono incassate (*embedded*) entro uno spazio lineare a bassa dimensionalità formato da *feature* latenti.

Nonostante alcuni esperimenti abbiano mostrato che i *prediction models* superano i *count models* in diversi task (Baroni, Dinu e altri, 2014), per ora i due approcci non differiscono in modo rilevante relativamente al significato che sono in grado di catturare. A tutti gli effetti sono due modi alternativi di costruire rappresentazioni distribuzionali.

3. I corpora

Il seguente capitolo intende focalizzarsi sulle risorse testuali utilizzate in questo studio: i corpora. In prima istanza si procederà con una descrizione generale della struttura dei dati, successivamente si illustrerà la fase di *preprocessing* preparativa all'analisi vera e propria.

Per questo lavoro sono state utilizzate trascrizioni radiofoniche del 1995 e del 2003. I testi del 1995 sono stati tratti dal corpus *LIR* (*Lessico Italiano Radiofonico*) che è il risultato di una ricerca pluriennale svolta presso il Centro di Grammatica Italiana dell'Accademia della Crusca, in collaborazione con il Dipartimento di Italianistica dell'Università di Firenze e la Scuola Normale Superiore di Pisa (Maraschio, 2004). Il *LIR* è un corpus di parlato radiofonico, registrato nel 1995 da diverse radio a diffusione nazionale, scelte sulla base dell'audience e della loro importanza storico-culturale. Sono state registrate 12 ore al giorno per ogni radio (dalle 7 alle 19) e sono stati fatti prelievi a scacchiera per una settimana, per un totale di 108 ore di registrazione, in modo da creare un corpus rappresentativo della grande varietà di programmazione radiofonica e di plurilinguismo che ogni giorno viene mandato in onda. Una volta eliminata la musica, sono rimaste 64 ore di parlato, distribuite in modo diverso nelle varie radio.
L'insieme dei testi è stato marcato in base a parametri diversi, come l'*emittente*, il *genere*, lo *speaker*, la *tipologia comunicativa* e a fenomeni tipici del parlato, come le *autocorrezioni*, i *troncamenti*, le *sovrapposizioni*, le *esitazioni* e le *pause*.
I testi del 2003 considerati nello studio sono un'espansione del corpus *LIR*.

3.1 Descrizione dei corpora

I dati, oggetto dell'analisi, sono la trascrizione di 116 trasmissioni radiofoniche relative a sette stazioni radio differenti: Radio Italia (IR), Rete 105 (R105), Radio 1 (RAI 1), Radio 2 (RAI 2), Radio 3 (RAI 3), Radio Radicale (RR), Radio Vaticana (RV).
I programmi radiofonici trascritti sono stati trasmessi in quattro giornate del 1995 e in cinque del 2003, rispettivamente il 23 maggio, 25 maggio, 27 maggio e 29 maggio per

quanto riguarda il 1995 e il 13 maggio, 15 maggio, 17 maggio e 18 maggio per quanto riguarda il 2003.
Di 116 trascrizioni, 81 sono relative al 1995 e 35 al 2003. Più specificatamente, per il 1995, sono presenti 12 file *.txt* per ogni stazione radiofonica (eccetto Radio Vaticana per cui sono presenti solamente 9 file). Per quanto concerne, invece, il 2003, si hanno 12 file *.txt* per Radio 1 e Radio 3 e 11 per Radio 2.

Nelle seguenti tabelle (v. tab. 3 e tab. 4), per ogni giornata analizzata, sono indicate le fasce orarie di ogni trascrizione radiofonica delle stazioni prese in esame.

Tabella 3. Tabella riassuntiva circa le informazioni delle trascrizioni del 1995

Data / Radio	23.5.1995	25.5.1995	27.5.1995	29.5.1995	Numero file
Radio Italia	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12	h. 12-13		
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
R105	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12		h. 12-13	
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18		h. 18-19	
RAI 1	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12	h. 12-13		
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
RAI 2	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12	h. 12-13		
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
RAI 3	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12	h. 12-13		
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
Radio Radicale	h. 7-8	h. 8-9	h. 9-10		12
	h. 10-11	h. 11-12	h. 12-13		
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
Radio Vaticana	h. 7-8				9
	h. 10-11	h. 11-12			
	h. 13-14	h. 14-15	h. 15-16		
	h. 16-17	h. 17-18	h. 18-19		
Totale					81

Tabella 4. Tabella riassuntiva circa le informazioni delle trascrizioni del 2003

Data / Radio	13.5.2003	14.5.2003	15.5.2003	17.5.2003	18.5.2003	Numero file
RAI 1	h. 7- 8		h. 8-9	h. 12-13	h. 15-16	12
			h. 9-10	h. 14-15		
			h. 10-11	h. 16-17		
			h. 11-12	h. 17-18		
				h. 18-19		
RAI 2	h. 7-8		h. 8-9	h. 9-10		11
	h. 10-11		h. 11-12	h. 12-13		
	h. 13-14		h. 14-15	h. 15-16		
			h. 17-18	h.18-19		
RAI 3	h. 7-8	h. 14-15	h. 8-9	h. 9-10		12
	h. 10-11		h.11-12	h. 12-13		
	h. 13-14			h. 15-16		
	h. 16-17		h. 17-18	h. 18-19		
Totale						35

Inoltre, ciascuna trascrizione presenta già delle annotazioni circa la radio di provenienza, lo speaker, la tipologia comunicativa e il genere di comunicazione (v. fig. 3).

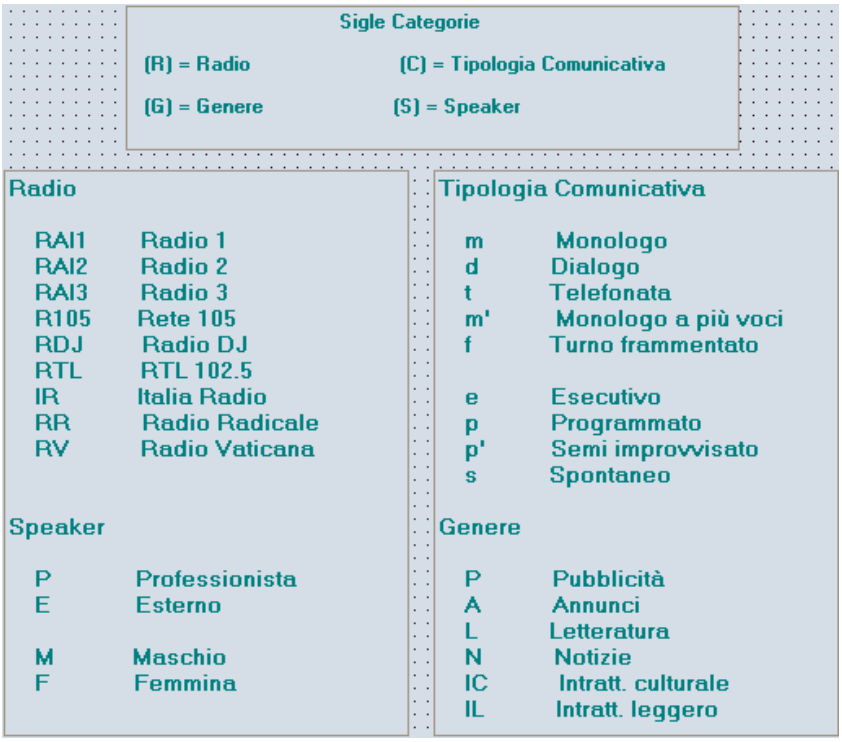

Figura 3. Legenda delle annotazioni

Di seguito si riporta un frammento di testo annotato a titolo esemplificativo:

```
\(R)RAI1\
[RAI1; 13.5.2003; h. 7-8; "Giornale radio GR1" ; conducono Ruggero
Po = 1pm e Simona Petracca = 2pf (…)]
(…)
\(G)N\\(C)m'e\
\(S)1pm\ [Po] terrore a Riyadh / esplosioni
&«R1_0708.wav:30129»contro obbiettivi statunitensi/ nella
&«R1_0708.wav:32133»capitale saudita / dove è atteso oggi /
&«R1_0708.wav:34146»Colin Powell / ancora imprecisato il
&«R1_0708.wav:36146»numero dei morti / decine i feriti //
&«R1_0708.wav:38161»
\(S)2pf\ [Petracca] Medio Oriente / nonostante le
&«R1_0708.wav:40170»nuove violenze nei territori / tra
&«R1_0708.wav:42166»Sharon e Abu Mazen / prove di dialogo /
&«R1_0708.wav:44179»all'indomani / della missione del segretario
di &«R1_0708.wav:46179»stato americano //
```

Nella prima riga è indicata la radio di riferimento, ossia Radio 1. Successivamente si possono notare le informazioni relative al giorno, all'ora, al titolo del programma trasmesso e ai conduttori. A ciascun presentatore è assegnato un codice identificativo, espresso ogni volta che lo speaker prende parola (ad esempio: `\(S)1pm\` per indicare che la battuta riportata è stata pronunciata da Ruggero Po).
Il programma radiofonico vero e proprio è preceduto dall'indicazione del genere e della tipologia comunicativa, in questo caso si tratta di un *notiziario* - `(G)N` - e di un *monologo a più voci esecutivo* - `(C)m'e`. Ogni notizia è riportata su una singola riga.

3.2 Fase di preprocessing

Dato che lo scopo del lavoro è quello di allineare notizie giornalistiche, per prima cosa è stato necessario ricavare dai dati di partenza solamente le porzioni di testo relative alle notizie. Ciò è stato possibile sfruttando il fatto che, ogni programma radiofonico, è preceduto dall'annotazione circa il genere di appartenenza, per cui è bastato estrarre le porzioni di testo che iniziavano con `\(G)N` (ossia con l'indicazione del genere delle notizie). Come delimitatore inferiore, si è utilizzata, invece, la segnalazione di un nuovo genere.
Di seguito si riporta la funzione in *python* elaborata per tale scopo:

```
def estrai(G, file):
      x= ""
      #variabile per controllare se il giro prima si era interrotto
perché aveva trovato un \(G)+ Genere di interesse
      OkGiroPrima=False
      categoriaOK="\(G)"+G
```

```
for line in file:
      if line[0:5]==categoriaOK or OkGiroPrima:
            x = x + line
            for line1 in file:
                  if line1:
                        if not(line1[0:4]== "\(G)"):
                              x = x + line1
                        else:
                              OkGiroPrima=False
                              if line1[0:5]== categoriaOK:
                                    OkGiroPrima = True
                              break
return x
```

Codice 1. Funzione per estrarre le porzioni di testo relative al genere di interesse

Tale funzione ha come parametri il genere del programma radiofonico che si vuole estrarre e il file della trascrizione e restituisce le porzioni di testo relative al genere di interesse.

Una volta ottenuti i file contenenti le notizie, si è proceduti, tramite l'utilizzo di *espressioni regolari*, con la pulizia dalle annotazioni, in quanto non necessarie al fine dell'allineamento.

```
def pulisci(notizie):
      pulito = re.sub(r'«IR\d+-\w+\.wav:\d+»|«R\d+-
      \w+\.wav:\d+»|«R\d.\d+.?\.wav:\d+»|«RJ\d+-
      \w+\.wav:\d+»|«RR\d+-\w+\.wav:\d+»|«RL\d+-
      \w+\.wav:\d+»|«RV\d+-\w+\.wav:\d+»|«RV\d+-
      \w+\.WAV:\d+»|\[.*?\]|\/|&T[EADFCGL]|&&.*&&|#|&c|&C|{.+}|&|\\.
      +\\', '', notizie)
      pulito1 = re.sub(r'&_|&-|_', ' ', pulito)
      return pulito1
```

Codice 2. Funzione per la pulizia dei corpora

Si illustra una porzione di file prima della pulizia e dopo la pulizia come esempio:

```
\(G)N\\(C)m'e\
  (…)
\(S)2pf\  [Petracca]  lo  sport  /
calcio   /   notte   di   stelle
&«R1_0708.wav:76298»a  San  Siro  /
per        l'euroderby        che
&«R1_0708.wav:78309»vale       una
stagione    /    stasera    Inter
&«R1_0708.wav:80308»Milan   /   a
caccia     della     finale     di
&«R1_0708.wav:82310»&TE&CChampions
&_league&c / code per i biglietti /
tutto  &«R1_0708.wav:84306»esaurito
//
```

Parte di testo prima della pulizia

```
(…)

  lo sport  calcio  notte di stelle
a San Siro  per l'euroderby che vale
una stagione  stasera Inter Milan  a
caccia  della  finale  di  Champions
league  code per i biglietti  tutto
esaurito
```

Parte di testo dopo la pulizia

I file risultanti da questo processo sono stati raggruppati, per anno, in due grandi corpora: il corpus del 1995 contenente tutte le notizie pulite di quell'anno e il corpus del 2003 con il medesimo contenuto relativo alla sua annata. Da questi corpora sono state eliminate le notizie con lunghezza minore di 10 token in quanto prive di contenuto significativo ai fini della nostra analisi.

Il corpus del 1995 comprende 2813 notizie, la cui lunghezza media è di 69 token, la massima di 3241 token e la minima di 10 token. Il corpus del 2003 invece include 1252 notizie con lunghezza media di 77 token, massima di 782 token e minima di 10 token.

4. Metodi di allineamento

In questo capitolo si entrerà nel vivo degli studi effettuati, mostrando i metodi di allineamento applicati ai corpora descritti.

Ciascun metodo illustrato avrà come input il corpus del 2003 ripetuto due volte, al fine di trovare le notizie riguardanti gli stessi argomenti. Solo i metodi di allineamento più interessanti sono stati eseguiti anche sul corpus del 1995. Questa scelta è stata fatta per motivi di efficienza, essendo il corpus del 2003 di dimensioni nettamente minori rispetto a quello del 1995.

4.1 Primo metodo: sovrapposizione token

Il primo metodo realizzato utilizza, come indice statistico, l'*indice di Jaccard*, noto anche come coefficiente di similarità di Jaccard. Tale indice misura la similarità di due campioni tramite il rapporto tra la dimensione dell'intersezione e la dimensione dell'unione degli insiemi campionari:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Se si considera con A la prima frase e con B la seconda frase, l'intersezione tra A e B, nel primo metodo utilizzato, è data dalla sovrapposizione dei token delle due frasi, ossia il numero di token presenti in entrambe.
L'unione tra A e B, invece, può esser vista come la somma dei token di entrambe le frasi, per cui il valore del denominatore è dato dalla dimensione delle due frasi messe insieme.
Il calcolo in termini di *token* può essere così rappresentato:

$$Sim_1(A,B) = \frac{n°\ di\ token\ sovrapposti(A,B)}{n°\ di\ token\ di\ A + n°\ di\ token\ di\ B}$$

Per applicare tale metodo al corpus precedentemente descritto, è stata implementata in *Python* la funzione *Confronto*.

Nella funzione principale, *main*, se le due frasi sono diverse[2], viene chiamata la funzione *Confronto*. Ciò che questa funzione restituisce – ossia il valore di similarità calcolato, il numero di token uguali e la lista di token uguali – è memorizzato nel dizionario *AllineamentoFrasi*, che ha come chiave le due frasi confrontate.

```
def main(file1, file2):
  (…)
  AllineamentoFrasi = {}
  i=0
  while i < len(ListaFrasi1):
      j = i + 1
      while j < len(ListaFrasi2):
                if ListaFrasi1[i] != ListaFrasi2[j]:
                            x, nTokUguali, TokUguali =
Confronto(ListaFrasi1[i], ListaFrasi2[j])
                           AllineamentoFrasi[ListaFrasi1[i],
ListaFrasi2[j]] = [x, nTokUguali, TokUguali]
                j = j + 1
      i = i + 1
 (…)
```

Codice 3. Funzione *main*

Nella funzione *Confronto* vengono tokenizzate le due frasi esaminate e contati i token sovrapposti per effettuare il calcolo di similarità, evidenziato in grassetto.

```
def Confronto(frase1, frase2):
       tokensF1 = nltk.word_tokenize(frase1)
       tokensF2 = nltk.word_tokenize(frase2)
       nTokensF1 = len(tokensF1)
       nTokensF2 = len(tokensF2)
       nTokUguali = 0
       TokUguali = []
       for tok1 in tokensF1:
              for tok2 in tokensF2:
                     if tok1 == tok2:
                            nTokUguali = nTokUguali + 1
                            #tolgo l'elemento contato sennò lo conto 2 volte
                            tokensF2.remove(tok2)
                            TokUguali.append(tok1)
                            break
       calcolo = (nTokUguali*1.0)/(nTokensF1*1.0+nTokensF2*1.0)
       return calcolo, nTokUguali, TokUguali
```

Codice 4. Funzione *Confronto* per il primo metodo

[2] L'indice che scorre la seconda lista di frasi è sempre maggiore di 1 rispetto a quello che scorre la prima lista. In tale modo si evita che venga confrontata una frase con se stessa, dato che le due liste contengono le medesime frasi.

4.2 Secondo metodo: sovrapposizione lemmi

Durante il processo di elaborazione dei metodi, si è aggiunta a questi sempre più competenza linguistica.

Il secondo metodo sviluppato, è una leggera variazione del primo, in quanto considera i *lemmi*, anziché i *token*, in modo da riuscire a sovrapporre anche forme diverse dello stesso lemma. Il calcolo della similarità delle due frasi, è stato quindi così modificato:

$$Sim_2(A, B) = \frac{n° \; di \; lemmi \; sovrapposti(A, B)}{n° \; di \; lemmi \; di \; A + n° \; di \; lemmi \; di \; B}$$

Per implementare in *Python* tale funzione di calcolo di similarità, si è sfruttato un sistema in grado di restituire le frasi lemmatizzate. Lo strumento scelto è stato *LinguA*[3], un software utile per effettuare e visualizzare diverse operazioni di annotazione automatica del testo. Tramite tale sistema è possibile visualizzare le informazioni estratte dall'analisi del testo in una tabella, in cui ogni frase è separata da una riga vuota e a ogni riga corrisponde un token, identificato con un ID.

	ID	Token	Lemma	C-POS	F-POS	Morphosyntactic feats	HEAD	DEP
1	1	LinguA	lingua	S	S	num:s gen:f	2	subj
	2	è	essere	V	V	num:s mod:i per:3 ten:p	0	ROOT
	3	una	uno	R	RI	num:s gen:f	4	det
	4	catena	catena	S	S	num:s gen:f	2	pred
	5	di	di	E	E		4	comp
	6	tool	tool	S	S	num:s gen:m	5	prep
	7	per	per	E	E		4	comp
	8	l'	il	R	RD	num:s gen:n	9	det
	9	analisi	analisi	S	S	num:n gen:f	7	prep
	10	linguistica	linguistico	A	A	num:s gen:f	9	mod
	11	.	.	F	FS		2	punc
2	1	Prova-	provare	V	V	num:s mod:m per:2 ten:p	0	ROOT
	2	la	la	P	PC	num:s gen:f per:3	1	obj
	3	!	!	F	FS		1	punc

Figura 4. Tabella che rappresenta l'annotazione linguistica del testo "LinguA è una catena di tool per l'analisi linguistica. Provala!" svolta da *LinguA*

Per ogni token, vengono riportati anche il lemma, due livelli di *part-of-speech* (una più generale, la C-POS, e una più specifica, la F-POS), i tratti morfologici (numero,

[3] http://linguistic-annotation-tool.italianlp.it/

genere, persona, modo, tempo o superlativo), l'ID della testa sintattica da cui dipende e il tipo di dipendenza da questa, utilizzando il tagset morfo-sintattico e il tagset a dipendenze ISST-TANL[4]. Questo sistema, permette infine di scaricare i risultati dell'analisi in formato CoNLL (Nilsson e altri, 2007), di cui è riportato un esempio (v. fig. 5).

```
1       LinguA  lingua  S       S       num=s|gen=f     2       subj
2       è       essere  V       V       num=s|per=3|mod=i|ten=p 0       ROOT
3       una     uno     R       RI      num=s|gen=f     4       det
4       catena  catena  S       S       num=s|gen=f     2       pred
5       di      di      E       E       _       4       comp
6       tool    tool    S       S       num=s|gen=m     5       prep
7       per     per     E       E       _       4       comp
8       l'      il      R       RD      num=s|gen=n     9       det
9       analisi analisi S       S       num=n|gen=f     7       prep
10      linguistica     linguistico     A       A       num=s|gen=f     9       mod
11      .       .       F       FS      _       2       punc

1       Prova-  provare V       V       num=s|per=2|mod=m|ten=p 0       ROOT
2       la      la      P       PC      num=s|per=3|gen=f       1       obj
3       !       !       F       FS      _       1       punc
```

Figura 5. Esempio di download in formato CoNLL generato da *LinguA*

Analizzando con *LinguA* il corpus oggetto dell'allineamento, si è potuto scaricare il relativo file in formato CoNLL. Il programma *.py*, in questo caso, ha preso come file in input l'analisi del corpus in formato CoNLL, per poter sfruttare la lemmatizzazione ai fini del calcolo della similarità.

La funzione relativa al confronto tra le due frasi lemmatizzate si è quindi presentata in questo modo:

```
def Confronto(LF1, LF2):
    #mi "copio" i lemmi di ogni frase in una nuova lista modificabile.
In modo da non intaccare la lista originale
    lemmiF1 = list(LF1)
    lemmiF2 = list(LF2)
    nLemmiF1 = len(lemmiF1)
    nLemmiF2 = len(lemmiF2)
    nLemUguali = 0
    LemUguali = []
    for lem1 in lemmiF1:
        for lem2 in lemmiF2:
            if lem1 == lem2:
                nLemUguali = nLemUguali + 1
                #tolgo l'elemento contato sennò lo conto 2 volte
                lemmiF2.remove(lem2)
                LemUguali.append(lem1)
                break
    calcolo = (nLemUguali*1.0)/(nLemmiF1*1.0+nLemmiF2*1.0)
```

[4] L'inventario delle categorie morfo-sintattiche e delle dipendenze sintattiche utilizzate, con relativa descrizione, è consultabile alle pagine http://www.italianlp.it/docs/ISST-TANL-POStagset.pdf e http://www.italianlp.it/docs/ISST-TANL-DEPtagset.pdf.

```
return calcolo, nLemUguali, LemUguali
```

Codice 5. Funzione *Confronto* per il secondo metodo

4.3 Terzo metodo: sovrapposizione lemmi POS rilevanti

In terza istanza, si è voluto sviluppare un metodo che considerasse soltanto i lemmi relativi alle *part-of-speech* rilevanti.
Il calcolo di similarità tra le due frasi è stato quindi impostato in questo modo:

$$Sim_3(A,B) = \frac{n°\ di\ lemmi\ sovrapposti\ POS\ rilevanti(A,B)}{n°\ di\ lemmi\ POS\ rilevanti\ di\ A + n°\ di\ lemmi\ POS\ rilevanti\ di\ B}$$

Anche per quanto riguarda questo metodo, il programma *.py* ha preso come file di input l'analisi del corpus di partenza in formato CoNLL. Questo perché, come si è detto in precedenza, tale formato specifica per ogni token la sua *part-of-speech*[5]. Tra le parti del discorso, nel calcolo, sono stati considerati come rilevanti i sostantivi, i verbi (ad eccezione degli ausiliari), gli aggettivi e i numerali.
La funzione di confronto, in questo caso, non differisce dalla precedente, l'unica cosa che varia è la lista di lemmi presi in esame.

4.4 Quarto metodo: variante logaritmica del terzo metodo

Questo quarto metodo è stato sviluppato per evitare alcuni casi che si potrebbero verificare con il sistema precedente. I lemmi dalle *part-of-speech* rilevanti, nelle frasi brevi, potrebbero essere pochi, per cui, con il calcolo precedente, avendo un denominatore basso e una sola sovrapposizione, si otterrebbe un valore di similarità che non rispecchierebbe la realtà.
Se si considerano ad esempio le frasi:

1- "Quanto costa a una famiglia un investimento così?"

[5] Il *Part-of-speech tagger* ingrato da *Lingua* registra un'accuratezza (calcolata come il rapporto tra il numero di token classificati correttamente e il numero totale di token analizzati) del 96,34% nell'identificazione simultanea della categoria grammaticale e dei tratti morfologici associati.

2- “La famiglia prima di tutto!”

Nella prima frase i lemmi rilevanti sono “costare”, “famiglia”, “investimento”, mentre nella seconda, solamente, “famiglia” – unico lemma comune tra le due.

Il calcolo di similarità, utilizzando il terzo metodo, avrebbe come risultato 0.25, che su una scala che ha come massimo 0.5, è un valore decisamente troppo alto per due frasi così diverse.

Per normalizzare casi di questo tipo, si è introdotto il logaritmo nella funzione di calcolo:

$$Sim_4(A,B) = \frac{n°\ lemmi\ sovrapposti\ POS\ rilevanti(A,B)}{n°\ lemmi\ POS\ rilevanti\ A + n°\ lemmi\ POS\ rilevanti\ B} \cdot \log(n°\ lemmi\ sovrapposti\ POS\ rilevanti(A,B))$$

In questo modo, tutte le frasi con un unico lemma sovrapposto avranno come valore di similarità 0.

Si riporta, per completezza la funzione di *Confronto* risultante da questa piccola aggiunta:

```
def Confronto(LF1, LF2):
  (…)
 calcolo = ((nLemUguali *1.0)/(
 nLemmiF1*1.0+nLemmiF2*1.0))*math.log(nLemUguali *1.0)
 (…)
```

Codice 6. Funzione *Confronto* per il quarto metodo

4.5 Versioni “tipo”

Per ogni metodo descritto fino a questo momento, è stata elaborata anche una versione *tipo* - ossia una versione che considera solamente gli elementi linguistici distinti -.

Ciò è stato realizzato per capire se vi sono differenze sostanziali tra i due diversi approcci nel *task* dell’allineamento.

Nel caso del primo metodo, si sono sostituiti i *token* con le *parole tipo*:[6]

$$Sim_1{}^1(A,B) = \frac{n°\ di\ parole\ tipo\ sovrapposte(A,B)}{n°\ di\ parole\ tipo\ di\ A + n°\ di\ parole\ tipo\ di\ B}$$

[6] Ogni equazione che utilizza le parole tipo o i lemmi tipo è indicata con l’apice “1”.

Nel secondo, terzo e quarto metodo, invece, si sono usati i *lemmi tipo*:

$$Sim_2{}^1(A,B) = \frac{n°\ di\ lemmi\ tipo\ sovrapposti(A,B)}{n°\ di\ lemmi\ tipo\ di\ A + n°\ di\ lemmi\ tipo\ di\ B}$$

$$Sim_3{}^1(A,B) = \frac{n°\ di\ lemmi\ tipo\ sovrapposti\ POS\ rilevanti(A,B)}{n°\ di\ lemmi\ tipo\ POS\ rilevanti\ di\ A + n°\ di\ lemmi\ tipo\ POS\ rilevanti\ di\ B}$$

$$Sim_4{}^1(A,B) = \frac{n°\ lemmi\ tipo\ sovrapposti\ POS\ rilevanti(A,B)}{n°\ \ lemmi\ tipo\ POS\ rilevanti\ \ A + n°\ \ lemmi\ \ tipo\ POS\ rilevanti\ B} \cdot \log(n°\ lemmi\ tipo\ sovrapposti\ POS\ rilevanti(A,B))$$

4.6 Allineamento con word embedding

L'ultimo metodo sviluppato non si basa, come gli altri, sulla sovrapposizione lessicale, ma sul concetto di *similarità semantica*. Il corpus è considerato come spazio vettoriale, entro cui i vettori delle parole sono più vicini quando queste si presentano negli stessi contesti linguistici, cioè quando sono riconosciute come semanticamente più simili (secondo l'ipotesi della semantica distribuzionale). I *word embedding* si possono definire come rappresentazioni compresse del contesto.

In questo studio, per produrre word embedding è stato utilizzato il tool *Word2vec,* che fu originariamente creato da Tomas Mikolov (Mikolov, 2013). Esso prevede un algoritmo che richiede in ingresso un corpus e restituisce un insieme di vettori che rappresentano la distribuzione semantica delle parole nel testo. Per ogni parola contenuta nel corpus, viene costruito un vettore in modo da rappresentarla come un punto nello spazio multidimensionale creato. In questo spazio le parole saranno più vicine se riconosciute come semanticamente più simili.

Il corpus, utilizzato come input, è *itWaC* (Italian web corpus), ovvero il corpus di maggiori dimensioni attualmente disponibile per la lingua italiana (Baroni e altri, 2009). Si tratta di un corpus di testi scaricati con metodi automatici dal web che contiene più di un miliardo e mezzo di parole.

L’output generato si presenta nella forma di *dizionario* e associa ad ogni parola utilizzata come *chiave* il relativo word embedding, composto da 128 componenti, come *valore*.

Al fine di calcolare la similarità delle coppie di frasi prese in esame, per ciascuna frase, sono state estratte le parole relative alle *part-of-speech* rilevanti (ossia i sostantivi, i verbi e gli aggettivi) e si è ricavato il relativo vettore sfruttando l’output prodotto da Word2vec.
Si è poi creato un unico vettore per ogni *part-of-speech* facendo la media dei *word embedding* relativi alle stesse categorie grammaticali. Concatenando i vettori risultanti (uno per i sostantivi, uno per i verbi e uno per gli aggettivi), si è quindi ottenuto il vettore della frase. La similarità è stata quindi calcolata facendo la distanza cosenica tra i vettori delle frasi.

5. Valutazione e analisi dei risultati

In questo capitolo, verrà tracciata una valutazione dei risultati prodotti dall'applicazione dei metodi appena descritti al corpus del 2003.

5.1 Valutazione delle prime 100 coppie di frasi per ogni metodo

Inizialmente, una prima valutazione è stata condotta sulle prime 100 coppie di frasi con valore di similarità più alto per ogni metodo. Leggendo tali coppie, si è potuto stimare quante di queste fossero correttamente allineate.

Di seguito si riporta un grafico che sintetizza i risultati di questa prima analisi:

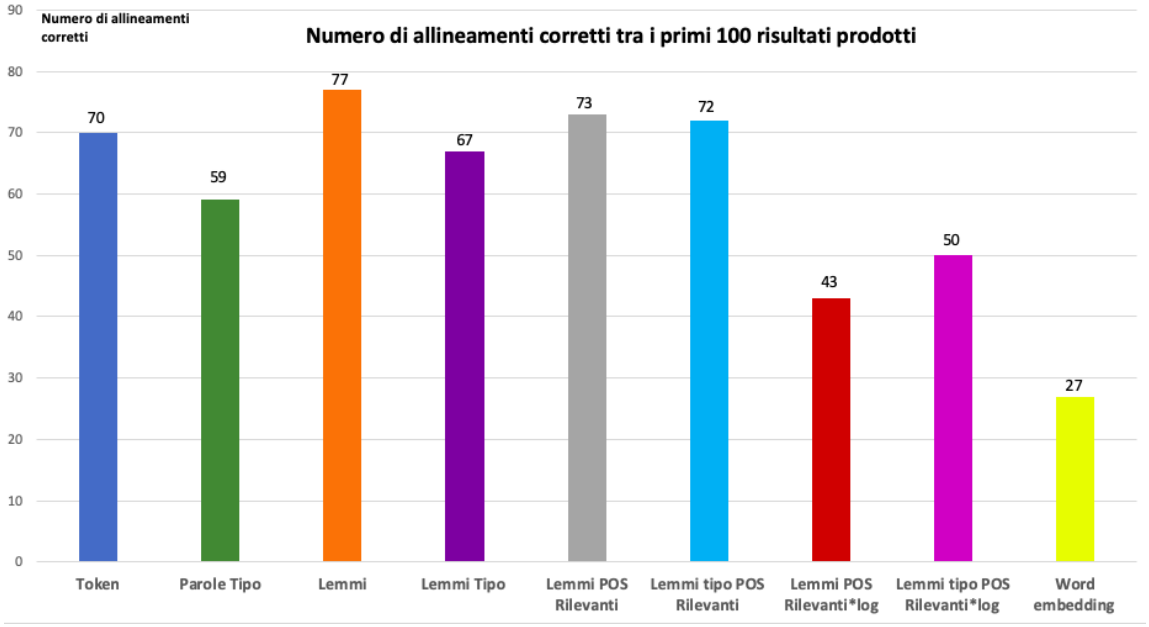

Figura 6. Numero di allineamenti valutati come corretti sulle prime 100 coppie di frasi

Da questo primo studio è emerso che:

1. Utilizzare il numero di occorrenze totali, generalmente, assicura prestazioni migliori. Ciò si può notare sia nel caso delle forme (con uno scarto di 11 coppie di frasi tra il metodo che utilizza i *token* e quello che utilizza le *parole tipo*), sia in quello dei lemmi (con uno scarto di 10 coppie di frasi tra il metodo che utilizza i *lemmi* e quello che utilizza i *lemmi tipo*) . Diverso è il caso del logaritmo, che però, in tutte e due le sue versioni, ha *performance* basse.

2. Aggiungere competenza linguistica dà un netto miglioramento al sistema. Gli esperimenti relativi ai *lemmi* e alle *part-of-speech* rilevanti hanno risultati superiori rispetto a quelli condotti semplicemente sulle forme. Inoltre, il crollo delle performance tra il metodo che utilizza i *token* e quello che si basa sulle *parole tipo* è maggiore rispetto a quanto accade per i *lemmi* e i *lemmi tipo*.
3. Il metodo relativo ai *word embedding* ha prestazioni molto più scarse. Questo può esser dato dal fatto che la sovrapposizione, che si cerca in questo caso, è semantica e non lessicale, quindi il task è molto più complicato.

   Negli altri metodi, è facilmente intuibile che al diminuire del valore di similarità, si avrà una sempre più bassa accuratezza nell'allineamento delle frasi – in quanto la sovrapposizione degli elementi linguistici sarà sempre minore -. Nel caso di questo metodo, invece, dato che non si ha a che fare con un overlap "secco" di parole, non si può supporre a priori l'andamento nelle fasce di similarità più basse. Al fine di conoscerlo è stata condotta un'ulteriore analisi circa l'accuratezza del sistema a diversi livelli di score di similarità.

5.2 Accuratezza nelle diverse fasce di similarità

Per ogni sistema basato sull'overlapping lessicale, sulle prime 100 coppie di frasi prodotte, si è calcolata l'accuratezza dell'allineamento in ciascuna fascia di valore di similarità. Per i primi tre metodi e le relative versioni "tipo", sono state considerate le fasce che vanno da 0.2 a 0.3, da 0.3 a 0.4 e da 0.4 a 0.5 (il massimo valore restituibile da tali sistemi). Poiché le prime 100 coppie di frasi generate dal metodo relativo ai *word embedding* rientravano solamente nella prima fascia di quel sistema (che va da 0.9 a 1), è stato necessario riapplicarlo per generare nuovi risultati appartenenti a diverse classi di valori. In particolare, sono state estratte frasi appartenenti alle fasce da 0.3 a 0.4, da 0.4 a 0.5, da 0.5 a 0.6, da 0.6 a 0.7, da 0.7 a 0.8, da 0.8 a 0.9 e infine, da 0.9 a 1, valore assegnato alle frasi con massimo grado di similarità semantica. Per motivi di efficienza, questa nuova applicazione del metodo è stata condotta su una porzione del corpus del 2003, composta da 125 frasi. Dei 7750 allineamenti prodotti, per ogni fascia considerata, è stata valutata l'accuratezza di 50 coppie di frasi estratte in maniera casuale. Per la prima classe di valori (0.9-1), è stato possibile calcolare una stima solo su 20 coppie, le uniche presenti.

La percentuale di accuratezza, per le diverse fasce di valori, è stata ottenuta applicando la seguente formula:

$$Accuracy_{fascia\ x} = \frac{n°\ di\ coppie\ di\ frasi\ valutate\ come\ correttamente\ allineate}{n°\ coppie\ di\ frasi\ allineate} \cdot 100$$

In appendice sono riportate, per ogni metodo, le tabelle con i risultati di tale calcolo, insieme al numero dei corretti allineamenti. Di seguito vengono illustrati i risultati delle analisi condotte sui metodi più interessanti.

Per primi, si descrivono i metodi basati sulla sovrapposizione lessicale. In particolare, verranno presi in esame sia quelli che utilizzano il numero di occorrenze totali, sia quelli che considerano elementi linguistici distinti.

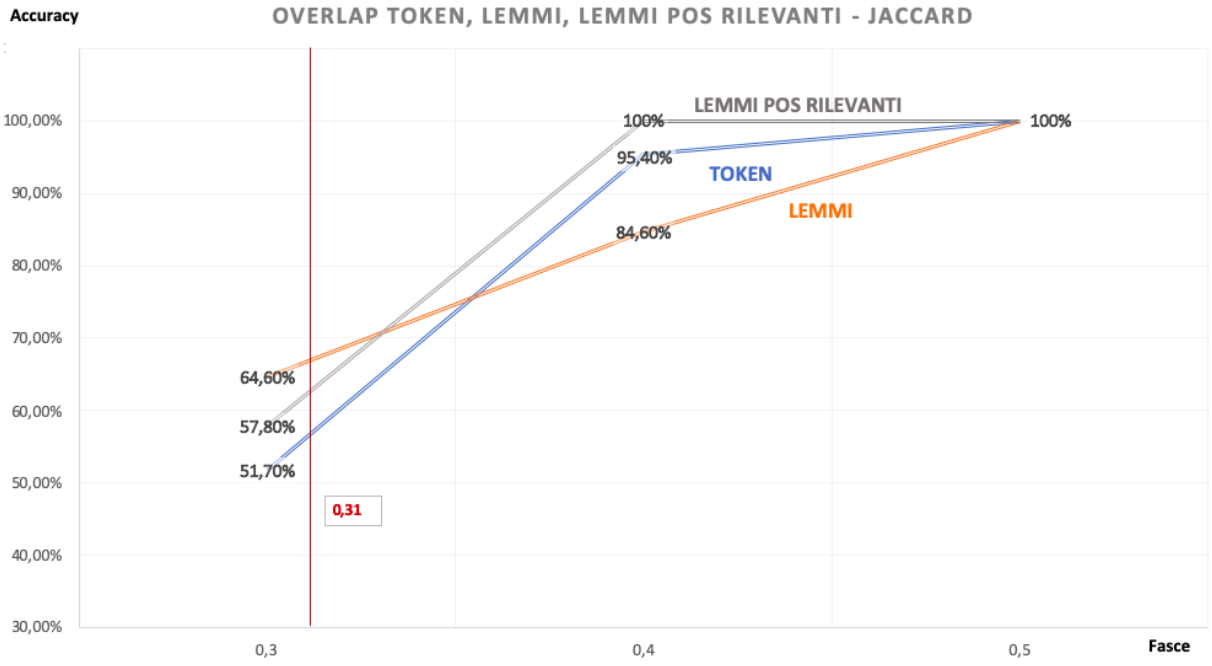

Figura 7. Calcolo dell'accuratezza dei metodi relativi alla sovrapposizione dei *token*, dei *lemmi* e dei *lemmi con part-of-speech rilevanti*

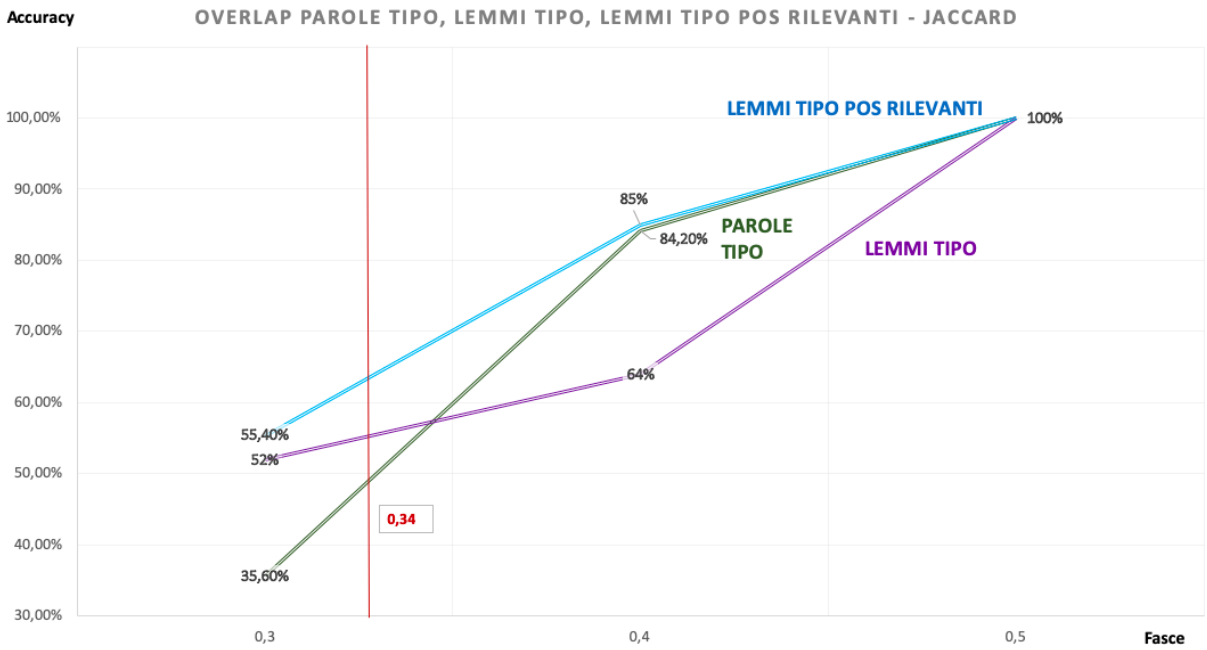

Figura 8. Calcolo dell'accuratezza dei metodi relativi alla sovrapposizione delle *parole tipo*, dei *lemmi tipo* e dei *lemmi tipo con part-of-speech rilevanti*

Nelle figure 7 e 8, vengono illustrati i valori di accuratezza per ciascun metodo: in particolare, il valore indicato come percentuale di accuratezza si riferisce alla fascia indicata con il suo limite superiore nell'asse delle *x*. Così, ad esempio, l'accuratezza del 35.6% per il metodo relativo alle *parole tipo* si riferisce alla fascia compresa tra 0.2 e 0.3.

Inoltre, la linea colorata in rosso indica il valore medio di similarità a cui corrisponde il primo errore nell'allineamento delle frasi per i tre metodi presi in esame in ciascuna figura. In particolare, nella figura 7, 0.31 è il valore medio tra 0.321428571429 per il metodo dei *token*, 0.321428571429 per il metodo dei *lemmi* e 0.288461538462 per il metodo dei *lemmi con part-of-speech rilevanti*. Il numero di coppie corrette alla destra della linea rossa varia a seconda del metodo considerato, ma si aggira sempre intorno a 38, con un massimo di 40 per il metodo dei *lemmi* e un minimo di 35 per quello dei *token*. Sebbene tali valori siano molto vicini tra loro, il metodo dei *lemmi* ha prestazioni migliori degli altri in quanto riesce ad allineare un maggior numero di frasi prima di incorrere nel primo errore.

Il valore medio del primo errore, per i metodi considerati nella figura 8, è più elevato rispetto a quello indicato nella figura 7. Ciò significa che tali sistemi tendono a sbagliare prima. Inoltre, ogni singolo metodo, presenta il primo errore a valori

maggiori rispetto a quelli di ciascun metodo considerato nella figura 7. In particolare il valore del primo errore del metodo delle *parole tipo* è di 0.34456928839, quello del metodo dei *lemmi tipo* è di 0.324786324786 e quello dei *lemmi tipo con part-of-speech rilevanti* è di 0.339622641509. In questo caso il sistema che riconosce più coppie di frasi corrette prima della soglia di errore è quello con le *part-of-speech* con un totale di 37 coppie. Gli altri metodi, invece, riescono a individuare correttamente solo 33 coppie.

In generale, tutte le funzioni che si basano sulla mera sovrapposizione lessicale, hanno un'accuratezza che decresce al diminuire del valore di similarità. Infatti, in tutti i metodi, a *score* più alti si trovano percentuali più elevate di coppie di frasi correttamente allineate. Più precisamente, nelle fascia da 0.4 a 0.5, ciascun sistema presenta il 100% di accuratezza, dimostrandosi in grado di allineare correttamente una media di 24 coppie di frasi sia per i metodi in figura 7, sia per quelli in figura 8.
Man mano che si scende nelle fasce di valori di similarità più basse, le coppie di frasi correttamente allineate sono sempre più rare e aumentano gli errori. Più specificamente, nella classe di valori compresa tra 0.2 e 0.3, i match corretti per il metodo dei *token* sono solamente 31 su 60, per quello dei *lemmi* 31 su 48, per quello dei *lemmi con part-of-speech rilevanti* 37 su 64, per quello delle *parole tipo* 21 su 59, per quello dei *lemmi tipo* 26 su 50 e per quello dei *lemmi tipo con part-of-speech rilevanti* 31 su 56.

Confrontando l'andamento del sistema relativo ai *token* con quello relativo ai *lemmi* si può notare che, nelle fasce più alte di valori (0.4-0,5 e 0.3-0.4), il primo è più accurato del secondo. La situazione si rovescia, invece, nell'ultima fascia (0.2-0.3). Ciò può essere spiegato dal fatto che la lemmatizzazione aggiunge un livello di astrazione superiore e quindi permette di trovare corretti allineamenti anche a valori di similarità più bassi. Si può osservare lo stesso andamento tra *parole tipo* e *lemmi tipo*, che però, rispetto ai *token* e ai *lemmi*, registrano percentuali di accuratezza inferiori sia nella fascia tra 0.3-0.4, sia in quella tra 0.2-0.3: in particolare le *parole tipo* riportano rispettivamente l'84.2% e il 35.6% (contro al 95.4% e il 51.7% dei *token*), mentre i *lemmi tipo* il 64% e il 52% (contro al 84.6% e il 64.6% dei *lemmi*). Ciò confermerebbe quanto è emerso dalla valutazione delle prime 100 coppie di frasi riportata nel capitolo

5.1: usare il numero totale di occorrenze, anziché i singoli elementi linguistici, assicura prestazioni migliori.

Anche nel caso dei *lemmi con part-of-speech rilevanti*, si rileva, in tutte le fasce, una maggiore accuratezza nella versione che utilizza il numero completo di frequenze. Inoltre è interessante constatare che entrambe le versioni del metodo risultano le più accurate nelle due classi di valori superiori (0.3-0.4, 0.4-0.5). Infatti, come già osservato, aggiungere competenza linguistica garantisce maggiori performance.

Di seguito, si inseriscono alcuni esempi di allineamento significativi per i metodi analizzati in figura 7, in quanto migliori a quelli in figura 8 in termini di accuratezza.

```
PRIMA FRASE: quello che dice il
giudice lo rispetto lo accetto
ma non mi convince come
motivazione questo fatto potrebbe
essere un argomento in più a
favore di chi dice che in realtà
i manifestanti non hanno fatto
assolutamente nulla e quello che
han fatto han fatto tutto i
poliziotti
```

```
SECONDA FRASE: quello che dice il
giudice lo rispetto lo accetto
ma non mi convince come
motivazione questo fatto potrebbe
essere un argomento in più a
favore di chi dice che in realtà
i manifestanti non hanno fatto
assolutamente nulla e quello che
hanno fatto han fatto tutto i
poliziotti
Valore similarità: 0.49
```

```
PRIMA FRASE: sicuramente noi
intanto chiediamo che si prosegua
l'indagine sulle forze di polizia
che fecero quello che fecero
quella notte credo che insomma la
prima volontà da parte di tutti
sia quella di veder accertata la
verità all'interno di un'aula di
tribunale
```

```
SECONDA FRASE: sicuramente noi
intanto chiediamo che si prosegua
l'indagine sulle forze di polizia
che fecero quello che fecero
quella notte credo che insomma la
prima volontà da parte di tutti
sia quella di vedere accertata la
verità all'interno di un'aula di
tribunale
Valore similarità: 0.488095238095
```

Esempi 1-2. Esempi relativi al metodo dei *token*

Come si può notare le frasi negli esempi 1 e 2 variano solamente per una parola e hanno un contenuto pressoché uguale. In giallo è evidenziata l'unica parola diversa nelle le due coppie. Le stesse coppie di frasi, con il metodo relativo ai *lemmi*, ricevono un punteggio di similarità massimo (0.5).

PRIMA FRASE: polmonite atipica diminuisce in Cina il numero delle persone in quarantena al momento sono circa diecimila è sempre alto l'allarme a Taiwan dove l'esercito è da ieri impegnato a fronteggiare il contagio proprio da Taiwan era giunto l' uomo d'affari morto nelle ultime ore in Nigeria è la prima vittima del Corona virus in Africa dove le autorità sanitarie temono che possa originare un' epidemia difficile da controllare

SECONDA FRASE: polmonite atipica altre dieci persone sono morte in Cina dove tuttavia diminuiscono i nuovi ammalati e le persone in quarantena è sempre alto l'allarme a Taiwan dove l'esercito è da ieri impegnato a fronteggiare il contagio preoccupazione delle autorità sanitarie mondiali per l'Africa dove si teme l'inizio di un'epidemia difficile da controllare in Nigeria è morto un uomo d'affari di Taiwan che potrebbe aver infettato alcuni conoscenti
Valore similarità: 0.368055555556

Esempio 3. Esempio relativo al metodo dei *lemmi*

Queste frasi, analizzate con il metodo dei *token*, ottengono un valore di similarità più basso (0.300751879699) in quanto non vengono considerate alcune sovrapposizioni, come quella tra *diminuisce* e *diminuiscono* evidenziata nell'esempio 3.

PRIMA FRASE: e un altro devastante attentato suicida ha colpito ieri la Cecenia il bilancio è di quarantatré morti e trecento feriti di cui una sessantina in gravi condizioni un kamikaze alla guida di un camion imbottito di tritolo forse una tonnellata di esplosivo si è scagliato contro la sede dell'amministrazione distrettuale filorussa a Znamjenskoje nel nord della repubblica separatista ribelle a Mosca ad una ottantina di chilometri dalla capitale Grozny il presidente russo Putin ha accusato APERTAMENTE i separatisti islamici ma la guerriglia cecena ha NEGATO ogni responsabilità per l'attacco tra i morti sette bambini quattordici donne e dieci ufficiali russi dei servizi segreti l'ex KGB e ora in Iraq si è consegnata agli americani la direttrice dei programmi batteriologici iracheni la cosiddetta dottoressa germe definita dalla CIA la donna più pericolosa del mondo arrestato anche l'ultimo capo di stato maggiore di Saddam tutto questo il giorno in cui si è insediato il nuovo capo dell' amministrazione civile l' ex diplomatico americano Paul Bremer

SECONDA FRASE: gli aggiornamenti sull'attentato in Cecenia è di quarantatré morti e trecento feriti dei quali una sessantina in gravi condizioni l'ultimo bilancio ufficiale tra i morti dieci ufficiali russi dei servizi segreti l'ex KGB autore della strage un kamikaze che alla guida di un camion carico di tritolo si è scagliato contro la sede dell'amministrazione distrettuale filorussa in una località a ottanta chilometri dalla capitale Grozny il presidente Putin ha accusato i separatisti islamici ma i guerriglieri ceceni hanno negato ogni responsabilità

Valore similarità: 0.272058823529

Esempio 4. Esempio relativo al metodo dei *lemmi con part-of-speech rilevanti*

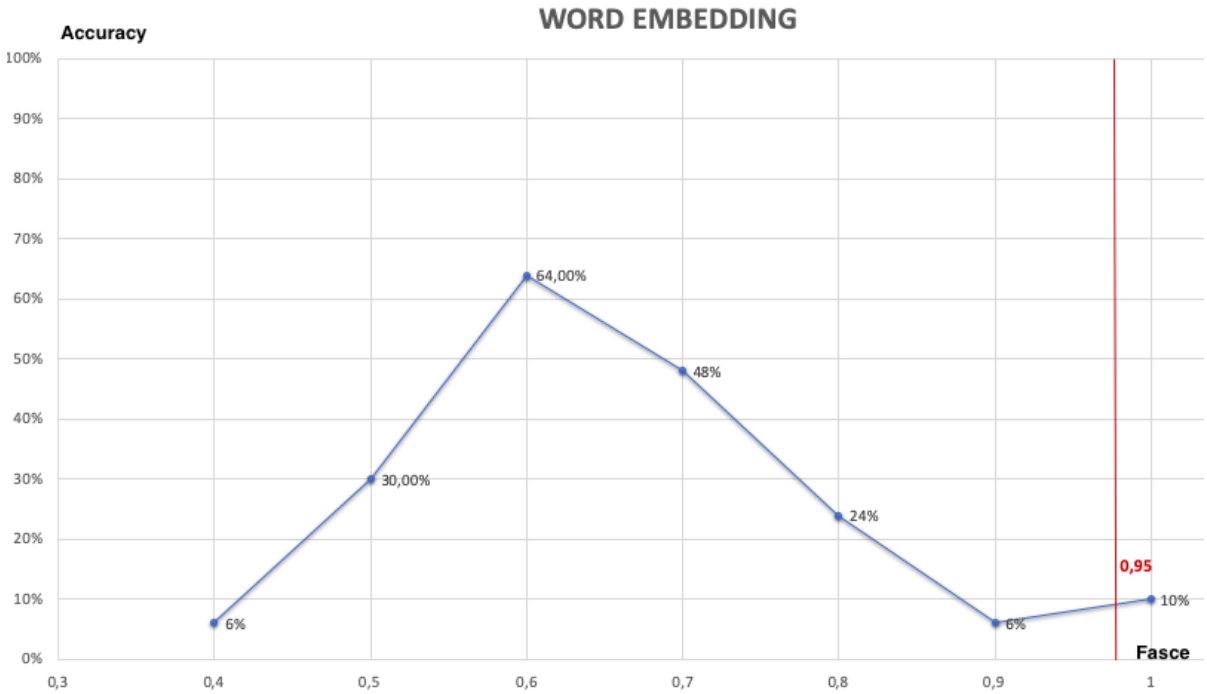

Figura 9. Grafico relativo all'*accuracy* del metodo relativo al *word embedding*

Il grafico 9 riporta i risultati relativi al sistema basato sul *word embedding*. Diversamente da quanto accade con gli altri metodi, nella fascia con valore di similarità più alto, si trova subito il primo errore nell'allineamento delle coppie di frasi. In particolare tale errore si colloca a uno *score* di similarità di 0.95. Ciò è dato dal fatto che il sistema calcola la similarità non più in termini di overlapping lessicale, ma semantico. Questo tipo di task è molto più complesso, soprattutto se si considera che la strategia usata si basa su una media di embedding delle stesse *part-of-speech* e le frasi che vengono analizzate sono molto lunghe.

A differenza di tutti gli altri metodi, scendendo nelle classi di valori di similarità sempre più basse, il sistema inizia subito a crescere in accuratezza, arrivando a un massimo di 64% nella fascia compresa tra 0.5 e 0.6. Raggiunto tale apice, si assiste poi a un crollo del 58% fino alla fascia compresa tra 0.3 e 0.4, in cui si registra una percentuale di accuratezza del 6%.

Nonostante questo metodo risulti meno accurato rispetto agli altri, a ogni fascia è comunque possibile trovare qualcosa di rilevante. Negli altri metodi, i risultati prodotti, seppur più accurati, risultano meno interessanti dal punto di vista qualitativo, in quanto restituiscono frasi pressoché uguali.

PRIMA FRASE: se l’inquilino non paga l’affitto i proprietari dell’appartamento potranno pagare meno tasse lo ha stabilito una sentenza della Cassazione sentiamo Laura Guida

SECONDA FRASE: qualche volta tocca anche al contribuente avere ragione per il fisco il proprietario di un appartamento doveva pagare le tasse in base ai contratti di affitto e non in base ai canoni mai versati mensilmente dell’affittuario non pagante ma la Cassazione smentendo il fisco e dando ragione al contribuente ha detto che le tasse vanno commisurate alle effettive ricchezze possedute e non ai contratti disattesi quindi non può essere tassato un proprietario che prima non ha incassato Raffaello Lupi è docente di diritto tributario all’Università di Tor Vergata

Valore similarità: 0.844981164989

Esempio 5. Primo esempio relativo all’allineamento con *word embedding*

Analizzando le medesime frasi con il sistema basato sulla sovrapposizione dei lemmi, si ottiene uno score di similarità molto basso, ossia 0.130434782609 su un massimo di 0.5.

Si riporta un ulteriore esempio a dimostrazione del fatto che sia possibile trovare allineamenti significativi a diverse fasce di valori.

PRIMA FRASE: lo sport calcio notte di
stelle a San Siro per l'euroderby che vale una stagione stasera Inter Milan a caccia della finale di Champions league code per i biglietti tutto esaurito

SECONDA FRASE: una notte in coda anche per nove ore per acquistare uno degli ultimi tremila biglietti e poter dire quella notte io c'ero a San Siro venti e quarantacinque l'euroderby che vale una stagione la statistica dice che sarà la duecentocinquantacinquesima stracittadina ma mai come stasera avrà un peso particolare Inter e Milan infatti si ritroveranno di fronte ritorno semifinale di Champions league per decidere chi potrà andare a giocarsi la finale a Manchester il ventotto maggio contro la vincente di domani sera tra Juventus e Real Madrid lo zero a zero di mercoledì scorso fa sì che il Milan tra virgolette in trasferta si qualificherebbe anche con un pareggio ma con gol

Valore similarità: 0.51

Esempio 6. Secondo esempio relativo all’allineamento con *word embedding*

Al fine di comprendere meglio i tipi di errori commessi da tale metodo, si riporta un esempio significativo di frasi non correttamente allineate nella fascia che va da 0.9 a 1.

PRIMA FRASE: era stato ricoverato nell'ospedale di Palombara Sabina per due settimane per un problema al fegato colpito dal morbo della legionella era stato quindi trasferito all'Aurelia hospital di Roma qui è rimasto alcuni giorni poi il decesso il dirigente sanitario della ASL RMG ha disposto in collaborazione con il dipartimento di prevenzione e di emergenza la bonifica delle cisterne idriche e la sostituzione delle rubinetterie nei reparti di medicina accettazione e day-hospital oncologico sono state inviate all'Arpa Lazio campionature di acqua dei tre reparti e anche delle centrali idriche e termiche in provincia di Rieti il mese scorso un caso sospetto di legionella la morte del sindaco di Morro Reatino

Valore similarità: 0.951883264947

SECONDA FRASE: il bambino di tre anni ricoverato all'ospedale Santo Bono di Napoli non rischia più la vita le sue condizioni sono in netto miglioramento assicurano quasi all'unisono i medici napoletani e i colleghi di Pontecorvo il piccolo era arrivato a Pontecorvo in provincia di Frosinone sabato mattina dal capoluogo campano in compagnia dei genitori per una breve visita ai parenti in località Tordone nel pomeriggio l'incidente il bambino è rimasto colpito da un pesante cancello scorrevole sorretto da una malferma colonna a vista attivato manualmente si è sganciato improvvisamente davanti all'abitazione una modesta casetta di campagna con vistose crepe i familiari lo hanno portato subito all'ospedale di Pontecorvo da qui il trasferimento al Santo Bono dove il piccolo è stato ricoverato per un trauma cranico e sospette fratture ad un braccio e ad una gamba il sopralluogo dei carabinieri della compagnia di Pontecorvo per accertare eventuali responsabilità

Esempio 7. Terzo esempio relativo all'allineamento con *word embedding*

Il non corretto allineamento, nell'esempio sopra riportato, risulta evidente poiché i pazienti di cui è riferita la storia clinica sono affetti da patologie completamente diverse. Nel primo caso si tratta di una patologia infettiva, che ha condotto al decesso del paziente. Nel secondo, invece, si tratta di una patologia traumatica che per fortuna si conclude con un esito favorevole. Tuttavia in entrambi i casi, il tema trattato è di tipo sanitario.

PRIMA FRASE: ora la cronaca con l'emergenza rifiuti in Campania riaprono le scuole e si torna lentamente alla normalità ma anche ieri ci sono state proteste

SECONDA FRASE: Belcolle l'unico ospedale del Lazio ad avere una scuola di ecografia per medici di base a Roma esistono infatti solo due strutture simili ma nelle Università della Sapienza e Policlinico Gemelli a Viterbo dall'inizio dell'anno il dipartimento diagnostica per immagini ha formato decine di medici e i responsabili hanno tracciato un primo bilancio CENTO i pazienti affetti da tumore del fegato trattati con ottimi risultati senza intervento chirurgico ma introducendo un ago attraverso la cute in grado di emettere radiofrequenze vicino alla parte malata l'operazione è seguita con l'ecografia la scuola è stata istituita dalla Società italiana di Ultrasonica
Valore similarità: 0.300260863905

Esempio 8. Quarto esempio relativo all'allineamento con *word embedding*

Nell'esempio appena citato è evidente il non corretto allineamento. Infatti, il contenuto della prima frase si riferisce all'emergenza rifiuti in Campania, mentre nella seconda si tratta di un tema di organizzazione sanitaria, ovvero della presenza di una scuola di ecografia per medici di base al di fuori delle strutture universitarie. I temi trattati sono completamente diversi e non hanno niente in comune a differenza dell'esempio precedente.

6. Conclusioni

Il presente lavoro si è concentrato sull'analisi di diversi metodi di allineamento, al fine di capire quale di questi fosse il più adatto per la creazione di un corpus di parafrasi del parlato.

Dallo studio è emerso che i sistemi che hanno prestazioni migliori, in termini quantitativi, sono quelli che si basano sulle occorrenze e fanno uso di competenza linguistica. Più in particolare, dalla valutazione delle prime 100 coppie di frasi con valore di similarità più alto per ogni metodo, risulta che il sistema che genera più coppie di frasi correttamente allineate è quello che si basa sulla sovrapposizione dei lemmi.

Andando a valutare la qualità dei risultati dei singoli metodi, si è notato che a ognuno di essi corrisponde un diverso tipo di allineamento comunque sempre valido. Per questo motivo la scelta di utilizzarne uno piuttosto che un altro discende dalle esigenze che uno ha.

In questo studio, per la creazione del corpus di parafrasi del parlato, si è deciso di utilizzare il metodo con risultati migliori dal punto di vista quantitativo, ossia quello relativo ai *lemmi*, insieme a quello più efficace dal punto di vista qualitativo, ovvero il sistema relativo ai *word embedding*, che si basa sulla similarità semantica e non sulla sovrapposizione lessicale. Inoltre, sebbene il task dell'allineamento con i *word embedding* abbia prestazioni peggiori rispetto agli altri metodi, in quanto risulta più difficile associare correttamente contenuti simili dal punto di vista semantico, si dimostra in grado individuare coppie di frasi più interessanti.

Dopo avere applicato i suddetti metodi per operare un allineamento tra *parlato* e *parlato*, utilizzando le trascrizioni radiofoniche di emittenti diverse, si prevede una ulteriore loro applicazione per allineare lingua *scritta* e *parlata*. Ciò consentirà di studiare la variazione diamesica della lingua. Questo nuovo task è attualmente in corso, in quanto si sta procedendo con l'estrazione dal Web di articoli del giornale *La Repubblica* relativi alle stesse giornate delle trascrizioni adoperate. Questo permetterà di applicare i metodi trattati a un corpus più ampio, probabilmente con risultati più soddisfacenti. Si stanno comunque studiando anche nuovi metodi volti a migliorare l'accuratezza dell'allineamento. Infine, poiché un problema di tale task, soprattutto se

viene utilizzato *word embedding*, è quello di soffrire nelle prestazioni quando vengono confrontate frasi molto lunghe e complesse, una possibile soluzione potrebbe essere quella della segmentazione del testo per poter paragonare sue sotto-parti. A tale scopo sono in studio metodi per suddividere il testo in maniera appropriata.

7. Bibliografia

Baroni M., A. Lenci. 2010. “Distributional memory: a general framework for corpus-based semantics”. *Computational Linguistics*, 36.4, pp. 673–721.

Baroni M., G. Dinu, G. Kruszewski. 2014. “Don’t count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors”. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 238–247.

Baroni, Marco e altri. 2009. *The WaCky wide web. A collection of very large linguistically processed web-crawled corpora*, «Language resources and evaluation» 43, 3, pp. 209-231.

Brunato D., Cimino A., Dell’Orletta F., Venturi G. 2016. *PaCCSS–IT: A Parallel Corpus of Complex–Simple Sentences for Automatic Text Simplification*. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), 1-5 November, Austin, Texas, USA, pp. 351-361.

Budassi, Marco e Edoardo Maria Ponti (a cura di). 2014. *Compter parler soigner. Tra linguistica e intelligenza artificiale.* Atti, Pavia, Collegio Ghislieri, 15-17 dicembre.

Curran J. R. 2004. “From distributional to semantic similarity”. Tesi di dott. University of Edinburgh.

Ferret O. 2013. “Identifying bad semantic neighbors for improving distributional thesauri”. *51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pp. 561–571.

Grefenstette G. 1994. *Explorations in automatic thesaurus discovery*. Norwell: Kluwer Academic Publishers.

Harris Z. S. 1954. “Distributional structure”. *Word*, 10.2-3, pp. 146–162.

Harris Z. S. 1991. *A theory of language and information: a mathematical approach*. Oxford: Clarendon Press.

Hongyan Jing. 2002. Using hidden markov modeling todecompose human-written summaries. *Comput. Linguist.*, 28:527–543, December.

Kiela D., S. Clark. 2014. “A systematic study of semantic vector space model parameters”. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) at EACL*, pp. 21–30.

Lapesa G., S. Evert. 2014. “A large scale evaluation of distributional seman- tic models: Parameters, interactions and model selection”. *Transactions of the Association for Computational Linguistics*, 2, pp. 531–545.

Lenci, Alessandro. 2014. *Semantica distribuzionale. Un modello computazionale del significato*. In: Budassi, Marco e Edoardo Maria Ponti (a cura di). *Compter parler soigner. Tra linguistica e intelligenza artificiale.* Atti, Pavia, Collegio Ghislieri, 15-17 dicembre. pp. 39-53.

Lund C., K. Burgess. 1997. “Modelling parsing constraints with high-dimensional context space”. *Language and cognitive processes*, 12.2-3, pp. 177–210.

N. Maraschio, S. Stefanelli, S. Buccioni, M. Biffi. 2004. *Dal* corpus LIR: *prove e confronti lessicali*, in F. Albano Leoni, F. Cutugno, M. Pettorino, R. Savy(a c. di), *Il Parlato Italiano*, Atti del Convegno Nazionale “Il Parlato Italiano”, Napoli 13-15 febbraio 2003 (redazione a cura di Manuela Senza Peluso), Napoli, M. D’Auria Editore, CD-ROM, documento PDF, pp. 36.

Mikolov T., K. Chen, G. Corrado, J. Dean. 2013. “Efficient estimation of word representations in vector space”. *Proceedings of Workshop at ICLR 2013*, pp. 1– 12.

Miller, George A., e Walter G. Charles. 1991. *Contextual correlates of semantic similarity, Language and Cognitive Processes*, 6:1, 1-28.

Nilsson, Jens; Sebastian Riedel; Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In “Proceedings of the CoNLL shared task session of EMNLP-CoNLL”. sn. 2007, pp. 915–932.

Padó S., M. Lapata. 2007. “Dependency-based construction of semantic space models”. *Computational Linguistics*, 33.2, pp. 161–199.

Rani Nelken and Stuart M. Shieber. 2006. Towards robust context-sensitive sentence alignment for monolingual corpora. *Proceedings of the 11th Conference ofthe European Chapter of the Association for Computational Linguistics* (EACL-06), 3–7 April.

Regina Barzilay and Noemi Elhadad. 2003. Sentence alignment for monolingual comparable corpora. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Sahlgren M. 2008. "The distributional hypothesis". *Italian Journal of Linguistics*, 20.1, pp. 33–54.

Salton G., A. Wong, C.-S. Yang. 1975. "A vector space model for automatic indexing". *Communications of the ACM*, 18.11, pp. 613–620.

Stefan Bott and Horacio Saggion. 2011. An unsupervised alignment algorithm for text simplification corpus construction. *Proceedings of the Workshop on Monolingual Text-To-Text Generation, co-located with ACL 2011*, Porland, Oregon.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*.

William Coster and David Kauchak. 2011. Simple english wikipedia: a new text simplification task. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

8. Appendice

Si riportano le tabelle relative ai dati utilizzati per il calcolo dell'*accuracy* citato nel quinto capitolo.

Tabella 5. Calcolo dell'accuracy per il primo metodo: la sovrapposizione dei token

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	22	22	100%
0.3-0.4	17	18	94,4%
0.2-0.3	31	60	51,7%

Tabella 6. Calcolo dell'accuracy per la versione *tipo* del primo metodo, ossia la sovrapposizione delle parole tipo

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	22	22	100%
0.3-0.4	16	19	84,2%
0.2-0.3	21	59	35,6%

Tabella 7. Calcolo dell'accuracy per il secondo metodo: la sovrapposizione dei lemmi

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	26	26	100%
0.3-0.4	22	26	84,6%
0.2-0.3	31	48	64,6%

Tabella 8. Calcolo dell'accuracy per la versione *tipo* del secondo metodo, ossia la sovrapposizione dei lemmi tipo

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	25	25	100%
0.3-0.4	16	25	64%
0.2-0.3	26	50	52%

Tabella 9. Calcolo dell'accuracy per il terzo metodo: la sovrapposizione dei lemmi delle POS rilevanti

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	24	24	100%
0.3-0.4	12	12	100%
0.2-0.3	37	64	57,8%

Tabella 10. Calcolo dell'accuracy per la versione *tipo* del terzo metodo, ossia la sovrapposizione dei lemmi tipo delle POS rilevanti

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
0.4-0.5	24	24	100%
0.3-0.4	17	20	85%
0.2-0.3	31	56	55,4%

Tabella 11. Calcolo dell'accuracy per il quarto metodo

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
1.5-2.22	21	21	100%
1-1.5	14	30	46,6%
0.9-1	1	22	4,5%
0.8-0.9	7	27	25,9%

Tabella 12. Calcolo dell'accuracy per la versione *tipo* del quarto metodo

Fascia	Numero coppie correttamente allineate	Numero coppie allineate	Accuracy
1.5-2.13	20	21	95,2%
1-1.5	15	26	57,6%
0.9-1	4	14	28,6%
0.8-0.9	8	26	30,8%
0,7-0,8	3	13	23%

Tabella 13. Calcolo dell'accuracy per l'ultimo metodo, ossia quello relativo ai word embedding

Fascia	**Numero coppie correttamente allineate**	**Numero coppie allineate**	**Accuracy**
0,9-1	2	20	10%
0,8-0,9	3	50	6%
0,7-0,8	12	50	24%
0,6-0,7	24	50	48%
0,5-0,6	32	50	64%
0,4-0,5	15	50	30%
0,3-0,4	3	50	6%