UNIVERSITÀ DEGLI STUDI DI BARI
"ALDO MORO"

DIPARTIMENTO DI INFORMATICA

CORSO DI LAUREA TRIENNALE IN INFORMATICA

TESI DI LAUREA

In Metodi per il Ritrovamento dell'Informazione

Induzione automatica del numero dei significati di una parola con tecniche di Semantica Distribuzionale

Relatori:
Prof. Pierpaolo Basile
Dott. Pierluigi Cassotti

Laureanda:
Lorena Capotorto

ANNO ACCADEMICO 2021/2022

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Lo studio di sistemi informatici per la comprensione e la produzione di linguaggio naturale da una prospettiva computazionale è chiamato linguistica computazionale [1]. I linguisti computazionali sono interessati nel provvedere modelli computazionali per risolvere ed osservare varie tipologie di fenomeni linguistici.

Il lavoro nella linguistica computazionale in alcuni casi è motivato da una prospettiva scientifica dove si cerca di provvedere una spiegazione computazionale per una particolare linguistica o per un fenomeno psico-linguistico; in altri casi la motivazione potrebbe essere puramente tecnologica: si vuole fornire una componente funzionante di un sistema vocale o di linguaggio naturale. Infatti, il lavoro della linguistica computazionale al giorno d'oggi è incorporato in molti sistemi lavorativi, includendo sistemi di riconoscimento vocale, sintetizzatori vocali, sistemi di risposta automatica, motori di ricerca su web, editori di testo e materiali didattici linguisti per nominarne qualcuno.

Pertanto, nella linguistica computazionale la Word Sense Induction (WSI) appare come un problema ancora aperto per quanto riguarda il campo della NLP[1] (Natural Language Processing) e che influisce su altri ambiti relativi ai computer, come il discorso, il miglioramento della pertinenza dei motori di ricerca, la risoluzione dell'anafora, la coerenza e l'inferenza. La WSI riguarda l'identificazione automatica del numero di sensi di una parola all'interno di un corpus, utilizzando un task di apprendimento non supervisionato. Poiché il risultato di questa induzione è un set di sensi per la parola selezionata o target, anche chiamato "sense inventory", questo task è strettamente relazionato a quello che riguarda la Word Sense Disambiguation (WSD).

La Word Sense Disambiguation è identificabile come il processo di identificazione di quali sensi sono propri di una parola in una frase o in un segmento di contesto. Nell'elaborazione e nella cognizione del linguaggio umano, questo

[1]Algoritmi di intelligenza artificiale in grado di analizzare, rappresentare e quindi comprendere il linguaggio naturale.

processo è di solito automatico o legato al subconscio, ma può spesso venire all'attenzione in maniera cosciente quando l'ambiguità della parola compromette la chiarezza della comunicazione, data la dilagante polisemia nel linguaggio naturale. Poichè il linguaggio naturale richiede il riflesso della realtà neurologica, modellata grazie alle capacità fornite dalle reti neurali del cervello, l'informatica ha avuto a che fare con una sfida a lungo termine nello sviluppo della capacità nei computer di poter eseguire l'elaborazione del linguaggio naturale e dell'apprendimento automatico.

Sono state studiate numerose tecniche, inclusi metodi basati su dizionari che utilizzano la conoscenza codificata in risorse lessicali, metodi di apprendimento automatico supervisionato in cui un classificatore viene addestrato per ogni parola in maniera distinta su un corpus di esempi con annotazioni e infine metodi completamente non supervisionati che raggruppano occorrenze di parole, inducendo così i sensi di queste.

Il presente progetto di tesi sarà incentrato su questi ultimi modelli, per poter effettuare una induzione sul numero dei significati di una serie di parole target polisemiche, ovvero che presentano una moltitudine di sensi che possono variare o meno in base al contesto in cui vengono ritrovate. In particolare, è obiettivo della tesi effettuare un'induzione tramite l'uso di tecniche di semantica distribuzionale, secondo la quale due parole sono tanto più simili semanticamente, quanto più tendono a comparire nello stesso contesto linguistico.

Il lavoro di tesi è così strutturato: nel secondo capitolo verrà presentato uno studio incentrato sulla WSI e sulla sua controparte WSD e su come queste due siano state affrontate in altri progetti ed articoli. Nel terzo capitolo verrà presentata la struttura e il funzionamento del progetto di tesi, e le scelte progettuali compiute per ottenere le parole target polisemiche ed effettuare induzione dei sensi su di esse. Il quarto capitolo illustra gli strumenti e script utilizzati per la realizzazione degli algoritmi per il soddisfacimento degli obiettivi prefissati. Infine nel quinto capitolo si affronterà la valutazione e sperimentazione dei dati raccolti facendo uso delle tecniche e delle metriche selezionate, confrontando gli ambienti di lavoro ottenuti per stabilire il raggiungimento o meno dell'obiettivo.

Capitolo 2

Stato dell'arte

In questo capitolo verrà affrontato il problema ancora aperto della Word Sense Induction, facendo riferimento alla sua controparte supervisionata, ovvero la Word Sense Deduction, esplorando come questa venga affrontata nel tema ricorrente di Word In Context.

Si osserverà come questo task venga approcciato nei confronti di altri lavori e progetti e di come possa essere uno spunto per questo progetto di tesi.

2.1 SemEval-2010

I significati si presentano molto più utili delle semplici forme delle parole per una varietà di attività, tra cui il recupero delle informazioni e la traduzione automatica. Tuttavia, i sensi sono generalmente rappresentati come un elenco fisso di definizioni che fanno parte di un database lessicale costruito manualmente. Questa stessa rappresentazione causa diversi svantaggi, ad esempio i database lessicali mancano dei principali sensi specifici del dominio, spesso contengono definizioni generali e soffrono della mancanza di collegamenti semantici o contestuali espliciti tra i concetti. Ancora più importante, le definizioni dei database lessicali realizzati a mano spesso non riflettono il significato esatto di una parola target in un dato contesto.

La Unsupervised Word Sense Induction (WSI)[1] mira a superare questi limiti apprendendo i sensi di una parola target direttamente dal testo senza fare affidamento su risorse artigianali.

Lo scopo principale del task di **SemEval-2010**[2] è quello di consentire il confronto di sistemi di induzione del senso delle parole e di disambiguazione, senza supervisione.

Il set di dati delle parole target su cui si basa il progetto è composto da 100 parole, 50 nomi e 50 verbi. Per ogni parola obiettivo, ai partecipanti è stato

[1]Problema aperto di elaborazione del linguaggio naturale, che riguarda l'identificazione automatica dei sensi di una parola.

fornito un set di formazione per apprendere i significati di una parola. Nella fase successiva, ai sistemi partecipanti è stato chiesto di disambiguare istanze ancora mai viste delle stesse parole usando i loro sensi appresi. Le risposte dei sistemi sono state poi inviate agli organizzatori per la valutazione.

2.1.1 Descrizione del Task

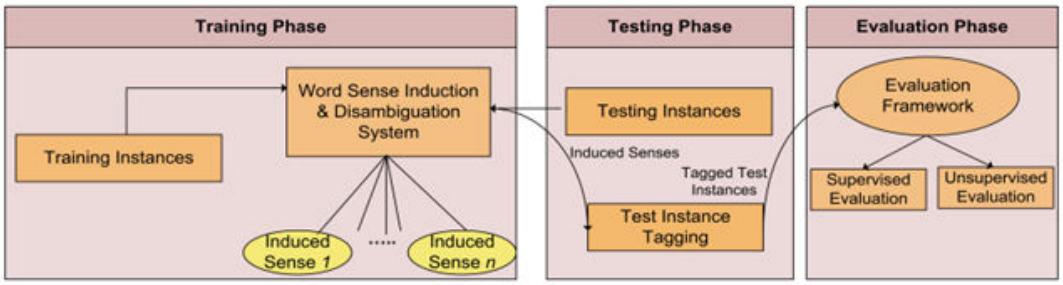


FIGURA 2.1: Fasi di addestramento, test e valutazione

La figura2.1 fornisce una panoramica dell'attività. Come si può osservare, il task consisteva di tre fasi separate. Nella prima, quella di addestramento, ai sistemi partecipanti è stato fornito un set di dati che consisteva in un insieme di istanze (frasi) di parole target. Ai partecipanti è stato quindi chiesto di utilizzare questo set di dati di addestramento per indurre i sensi della parola target. Non è stato consentito l'utilizzo di altre risorse ad eccezione dei componenti NLP[2] per la morfologia e la sintassi.

Nella seconda fase, quella di test, ai sistemi partecipanti è stato fornito sempre un set di dati di test che consisteva in un insieme di istanze di parole target, ma è stato chiesto di disambiguare con delle etichette ogni istanza con i sensi indotti durante la fase di addestramento.

Nella terza e ultima fase, le istanze di test etichettate sono state utilizzate per valutare le risposte dei sistemi in un framework supervisionato e non supervisionato.

	Training set	**Testing set**	**Senses (#)**
All	879807	8915	3.79
Nouns	716945	5285	4.46
Verbs	162862	3630	3.12

TABELLA 2.1: Dettagli dei set di addestramento e test.

[2]Algoritmi di intelligenza artificiale in grado di analizzare, rappresentare e quindi comprendere il linguaggio naturale.

La tabella2.1 mostra il numero totale di istanze di parole target nel set di addestramento e test, nonché il numero medio di sensi nel gold standard[3]. Trattare i dati di test come nuove istanze mai viste prima garantisce una valutazione realistica che consente di valutare i modelli di clustering di ciascun sistema partecipante.

2.1.2 Valutazione non supervisionata

Per quanto riguarda la valutazione non supervisionata sono state utilizzate due misurazioni: **V-Measure** e **paired F-Score**.

Valutazione con V-Measure

Sia w una parola target con N istanze nel set di dati di test. Sia $K = \{C_j | j = 1...n\}$ sia un insieme di cluster generati automaticamente che raggruppano queste istanze e $S = \{G_i | i = 1...m\}$l'insieme delle classi gold standard che contengono i raggruppamenti desiderabili di w istanze. La V-measure valuta la qualità di una soluzione di clustering misurandone esplicitamente l'omogeneità e la completezza. L'omogeneità si riferisce al grado con cui ogni cluster è costituito da punti di dati principalmente appartenenti a una singola classe, mentre la completezza si riferisce al grado con cui ogni classe è costituita di punti di dati assegnati principalmente a un singolo cluster.

Considerando h l'omogeneità e c la completezza, la V-Measure è la media armonica di h e c, cioè $VM = \frac{2 \cdot \mathrm{h} \cdot \mathrm{c}}{\mathrm{h+c}}$.

Valutazione con paired F-Score

Poiché la prima misurazione (V-measure) tendeva a favorire sistemi che producessero un numero più alto di cluster rispetto all'effettivo numero di sensi, è stata introdotta una seconda misurazione che penalizzasse i sistemi prodotti: un alto numero di cluster avrebbe significato un richiamo basso oppure un basso numero di cluster avrebbe significato una bassa precisione, rispetto al numero di sensi.

Per questo tipo di metrica, il problema da essere di clustering è diventato di classificazione. Per ogni cluster C_i viene generata una coppia di istanze, dove $|C_i|$ è il numero totale delle istanze che appartengono al cluster C_i, lo stesso avviene per la classe G_i. $F(K)$ sarà il set di coppie di istanze che si trovano

[3]Diagnostica, test o benchmark che è il migliore disponibile a condizioni ragionevoli, non per forza in termini assoluti.

nei cluster indotti automaticamente e $F(S)$ sarà il set di coppie di istanze che si trovano nei gold standard.

La precisione, quindi, può essere definita come il numero di coppie di istanze comuni tra i due set, moltiplicato al numero di coppie totali nei cluster di soluzione. Mentre il richiamo può essere definito come il numero di coppie di istanze comuni tra i due set moltiplicato per il numero totale di coppie nel gold standard. Infine precisione e richiamo vengono combinati per ottenere la media armonica ($FS = \frac{2 \cdot P \cdot R}{P + R}$).

Si possono applicare queste misurazioni (V-Measure e paired F-Score) a due punti di riferimento: *Most Frequent Sense* (MFS), che raggruppa tutte le istanze di test di una parola target in un cluster e il secondo, *Random*, che assegna casualmente una istanza ad uno dei 4 cluster. Il numero di cluster di *Random* è stato scelto per essere uguale alla media del numero dei sensi.

System	FS (%) (All)	FS (%) (Nouns)	FS (%) (Verbs)	#Cl
MFS	63.5	57.0	72.7	1
Duluth-WSI-SVD-Gap	63.3	57.0	72.4	1.02
KCDC-PT	61.8	56.4	69.7	1.5
KCDC-GD	59.2	51.6	70.0	2.78
Duluth-Mix-Gap	59.1	54.5	65.8	1.61
Duluth-Mix-Uni-Gap	58.7	57.0	61.2	1.39
KCDC-GD-2	58.2	50.4	69.3	2.82
KCDC-GDC	57.3	48.5	70.0	2.83
Duluth-Mix-Uni-PK2	56.6	57.1	55.9	2.04
KCDC-PC	55.5	50.4	62.9	2.92
KCDC-PC-2	54.7	49.7	61.7	2.93
Duluth-WSI-Gap	53.7	53.4	53.9	1.4
KCDC-PCGD	53.3	44.8	65.6	2.9
Duluth-WSI-Co-Gap	52.6	53.3	51.5	1.6
Duluth-MIX-PK2	50.4	51.7	48.3	2.66
UoY	49.8	38.2	66.6	11.54
Duluth-Mix-Narrow-Gap	49.7	47.4	51.3	2.42
Duluth-WSI-Co	49.5	50.2	48.2	2.49
Duluth-Mix-Narrow-PK2	47.8	37.1	48.2	2.68
Duluth-R-12	47.8	44.3	52.6	2
Duluth-WSI-SVD	41.1	37.1	46.7	4.15
Duluth-WSI	41.1	37.1	46.7	4.15
Duluth-R-13	38.4	36.2	41.5	3
KSU KDD	36.9	24.6	54.7	17.5
Random	31.9	30.4	34.1	4
Duluth-R-15	27.6	26.7	28.9	4.97
Hermit	26.7	24.4	30.1	10.78
Duluth-R-110	16.1	15.8	16.4	9.71

FIGURA 2.2: Valutazione non supervisionata con paired F-Score

Dalla figura2.2 si può vedere come la maggioranza dei sistemi lavorino meglio rispetto a *Random*. Nonostante questo, nessuno supera le performance del *MFS*. Sembra come se i sistemi che generino piccoli numeri di cluster, rispetto al numero di sensi, siano prevenuti verso il *MFS*; infatti, non sono capaci di produrre risultati migliori. D'altro canto, sistemi che generano un alto numero di cluster sono penalizzati da queste misurazioni non supervisionate.

2.1.3 Valutazione supervisionata

In questa valutazione, il set di dati di test è suddiviso in un corpus di mappatura e uno di valutazione. Il primo viene utilizzato per mappare i cluster indotti automaticamente ai sensi, mentre il secondo è usato per valutare metodi in un ambiente WSD[4]. I risultati finali equivarranno ad una media di 5 suddivisioni casuali del set di dati, per evitare che diverse suddivisioni possano restituire graduatorie differenti.

In una prima valutazione, è stata utilizzata una suddivisione del set di dati di *80-20*, rispettivamente per mappatura e valutazione. La valutazione supervisionata cambia la distribuzione dei cluster mappandone ciascuno in base a un vettore pesato di sensi. Questo può potenzialmente favorire il sistema di valutazione generando un alto numero di cluster omogenei.

In seguito, come seconda valutazione è stata scelta una suddivisione del set di dati di *60-40*. Riducendo la dimensione del corpus per la mappatura si è osservato come sistemi con un numero alto di cluster possano soffrire di mappature non affidabili.

System	SR (%) (All)	SR (%) (Nouns)	SR (%) (Verbs)	#S
UoY	62.0	58.6	66.8	1.66
Duluth-WSI-Co	60.1	54.6	68.1	1.56
Duluth-WSI-Co-Gap	59.5	53.5	68.3	1.2
Duluth-WSI-SVD	59.5	53.5	68.3	1.73
Duluth-WSI	59.5	53.5	68.3	1.73
Duluth-WSI-Gap	59.3	53.2	68.2	1.11
KCDC-PCGD	59.1	52.6	68.6	1.54
KCDC-PC-2	58.9	53.4	67.0	1.25
KCDC-PC	58.9	53.6	66.6	1.44
KCDC-GDC	58.3	52.1	67.3	1.41
KCDC-GD	58.3	51.9	67.6	1.42
MFS	58.3	52.5	66.7	1
KCDC-PT	58.3	52.2	67.1	1.11
Duluth-WSI-SVD-Gap	58.2	52.5	66.7	1.01
KCDC-GD-2	57.9	51.7	67.0	1.44
Duluth-R-12	57.7	51.7	66.4	1.27
Duluth-R-13	57.6	51.1	67.0	1.48
Hermit	57.3	52.5	64.2	2.27
Duluth-R-15	56.5	50.0	66.1	1.76
Random	56.5	50.2	65.7	1.65
Duluth-Mix-Narrow-Gap	56.2	47.7	68.6	1.51
Duluth-Mix-Narrow-PK2	55.7	46.9	68.5	1.51
Duluth-R-110	53.6	46.7	63.6	2.18
Duluth-MIX-PK2	50.5	39.7	66.1	1.31
KSU KDD	50.4	44.3	59.4	1.92
Duluth-Mix-Gap	49.8	38.9	65.6	1.04
Duluth-Mix-Uni-PK2	19.1	1.8	44.4	0.63
Duluth-Mix-Uni-Gap	18.9	1.5	44.2	0.56

FIGURA 2.3: Richiamo supervisionato con suddivisione (test di set 60-40

[4]Processo di identificazione di quali sensi sono propri di una parola in una frase o in un segmento di contesto.

Da quello che si può vedere dalla figura 2.3 la riduzione del corpus di mappatura porta con sé un impatto differente su sistemi che generano grandi numeri di cluster rispetto al numero di sensi.

Si può dire quindi che la performance di questa valutazione dipenda sia dalla riduzione del corpus di mappatura che dalla distribuzione di istanze nei cluster. Sistemi che generano delle distribuzioni distorte, dove un piccolo numero di cluster omogenei comprende la maggioranza delle istanze e un numero grande di cluster ne comprende solo poche, sono più predisposti ad avere delle performance migliori rispetto a sistemi che producono distribuzioni uniformi.

2.2 Novel sense WSI

La Word Sense Induction (WSI) è il compito di indurre automaticamente i diversi significati di una parola, generalmente sotto forma di un task di apprendimento non supervisionato con i sensi rappresentati come gruppi di istanze tokenizzate. Va in contrasto con la disambiguazione del senso di una parola (WSD), in cui si presume che esista un inventario di sensi fisso e che le istanze tokenizzate di una data parola siano disambiguate in base all'inventario stesso.

Nonostante la WSI si presenti come intuitivamente interessante come attività, non ci sono stati esempi reali di implementazioni di successo in applicazioni per gli utenti finali nel contesto di recupero delle informazioni. Lo scopo del **Novel Sense WSI**[3] è l'applicazione riuscita di WSI al task lessicografico di rilevamento di nuovi sensi, ovvero l'identificazione di parole che hanno assunto nuovi sensi nel corso del tempo. Una delle sfide chiave in WSI è l'apprendimento della granularità dei sensi appropriata per una data parola, ovvero il numero di sensi che cattura meglio le occorrenze simboliche di quella parola. Il WSI viene affrontato tramite modellazione della tematica, utilizzando Latent Dirichlet Allocation[4] (**LDA**) e il modello stesso per determinare la granularità sensoriale appropriata.

La modellazione tematica è un approccio non supervisionato per apprendere contemporaneamente argomenti, sotto forma di distribuzioni multinomiali di probabilità sulle parole, e assegnazioni di argomenti per documento, sotto forma di distribuzioni multinomiali di probabilità sugli argomenti. LDA è allettante per la WSI poiché entrambi assegnano sensi alle parole e inoltre produce una rappresentazione di ogni senso come un elenco ponderato di parole. LDA offre una soluzione alla questione della determinazione della granularità dei sensi tramite formulazioni non parametriche, così come nel **Hierarchical Dirichlet Process** [5][6](**HDP**).

2.2.1 Metodologia

Nella modellazione tematica, si presume che i documenti espongano più argomenti, con ogni documento che ha la propria distribuzione su di essi. Le parole vengono generate in ciascun documento campionando prima un argomento dalla distribuzione dei suoi argomenti e poi campionando una parola da quell'argomento. In questo articolo viene usata l'assegnazione probabilistica degli argomenti alle parole.

Rappresentazione dei dati

Nel contesto della WSI, gli argomenti formano la rappresentazione sensoriale e le parole in una frase sono generate in base ad un senso particolare della parola target. Il "documento" è una singola frase o un breve frammento di documento contenente la parola target, poiché non ci si aspetterebbe di poter generare un documento completo dal senso di una singola parola.

Nel caso dei set di dati SemEval[5], vengono usati i contesti di parole forniti nel set di dati, mentre nei nuovi esperimenti, viene usata una finestra di contesto di tre frasi, una frase su entrambi i lati dell'occorrenza simbolica della parola target. Come rappresentazione di base, si utilizza un insieme di parole, in cui viene mantenuta la loro frequenza ma non l'ordine. Tutte le parole sono poi lemmatizzate e le stopword e i termini a bassa frequenza vengono rimossi. Si sperimenta anche l'aggiunta di informazioni sulle parole di contesto posizionali, come comunemente vengono usate in WSI. Ovvero, viene introdotta una funzione di parola aggiuntiva per ciascuna delle tre parole a sinistra e a destra della parola target. **Pado e Lapata**[7] hanno dimostrato l'importanza delle relazioni di dipendenza sintattica nella costruzione di modelli di spazio semantico, come per il WSD. Sulla base di questi risultati, vengono incluse le relazioni di dipendenza come caratteristiche aggiuntive nei modelli tematici di questo articolo, ma solo per quelle che coinvolgono la parola target.

Modellazione tematica

Viene usata la modellazione tematica LDA, che richiede di impostare T, il numero di argomenti che devono essere appresi dal modello. Il processo generativo dell'LDA è: disegnare un argomento latente z da una distribuzione di argomenti specifica del documento $P(t = z|d)$ ed estrarre una parola w dall'argomento scelto $P(w|t = z)$. Pertanto, la probabilità di produrre una singola copia della parola w dato un documento d è data da:

[5] `https://semeval.github.io/`

$$P(w|d) = \sum_{z=1}^{T} P(w|t = z)P(t = z|d)$$

Nell' LDA standard, l'utente deve specificare il numero di argomenti T. Nelle varianti non parametriche di LDA, il modello apprende dinamicamente il numero di argomenti come parte della loro modellazione. Viene applicata una particolare implementazione del modello di argomento non parametrico ovvero la Hierarchical Dirichlet Process[5] (**HDP**), dove, per ogni documento, viene campionata una distribuzione delle combinazioni dei componenti $P(t|d)$ da una distribuzione di base $G0$ in questo modo: si sceglie una distribuzione di base $G0 \sim DP(\gamma, H)$, per ogni documento d, si genera la distribuzione $P(t|d) \sim DP(\alpha 0, G0)$, viene ricavato un argomento latente z dalla distribuzione delle combinazioni delle componenti del documento $P(t|d)$, allo stesso modo dell'LDA, e infine viene estratta una parola w dall'argomento scelto $P(w|t = z)$. Sia per LDA che per HDP, viene modellato individualmente l'argomento di ciascuna parola target e viene determinata l'assegnazione del senso z per una data istanza, aggregando le assegnazioni dell'argomento per ciascuna parola nell'istanza e selezionando il senso con la probabilità aggregata più alta, $P(t = z|d)$.

2.2.2 Sperimentazione

Per facilitare il confronto del metodo proposto per WSI con approcci precedenti, viene utilizzato il set di dati del task di induzione del senso di una parola SemEval 2010[2], il quale contiene 100 parole target: 50 nomi e 50 verbi. Verranno proposti risultati per i seguenti modelli:

1. **LDA+Variable T**: LDA con la variabile T per ogni parola target in base al numero di sensi gold standard.
2. **LDA+Fixed T**: LDA con T fissa per ciascuno dei nomi e dei verbi.
3. **LDA+Fixed T+Position**: LDA con T fissa e features di parola posizionale extra.
4. **HDP+Position**: HDP (che apprende automaticamente la T) con ulteriori features delle parole posizionali.
5. **HDP+Position+Dependency**: HDP con sia features di dipendenza che di parola posizionale.

I modelli sono confrontati con due punti di riferimento dell'attività SemEval-2010: *Random*, che assegna in modo casuale ogni istanza di test a uno dei quattro significati, *MFS*, punto di riferimento del senso più frequente, che assegna tutte le istanze del test a un senso e infine anche un sistema di riferimento (UoY), sotto forma del sistema dell'Università di York[8], che ha ottenuto i migliori risultati complessivi per WSD nel task originale SemEval-2010.

System	WSD (80%/20%)		
	All	Verbs	Nouns
Baselines			
Baseline Random	0.57	0.66	0.51
Baseline MFS	0.59	0.67	0.53
LDA			
Variable T	0.64	0.69	0.60
Fixed T	0.63	0.68	0.59
Fixed T +Position	0.63	0.68	0.60
HDP			
+Position	0.68	0.72	0.65
+Position+Dependency	0.68	0.72	0.65
Benchmark			
UoY	0.62	0.67	0.59

FIGURA 2.4: F-score con approccio WSD sul dataset SemEval-2010.

Dalla figura 2.4 si può vedere che il primo approccio LDA (variabile T) è molto competitivo, superando così il sistema di benchmark. In questo approccio, tuttavia, si assume una perfetta conoscenza del numero di sensi gold di ciascuna parola target, il che significa che il metodo non è veramente non supervisionato. Quando è stata fissata T per ciascuno dei nomi e dei verbi, si è visto un piccolo calo nel F-score, ma è incoraggiante che il metodo funzioni ancora al di sopra del benchmark. L'aggiunta di features delle parole posizionali migliora leggermente i risultati per i nomi.

Passando a HDP, si osserva un netto miglioramento del F-score rispetto a LDA. Questo è molto incoraggiante e in qualche modo sorprendente, poiché nascondendo le informazioni sulla granularità dei sensi dal modello, i risultati sono effettivamente migliorati.

Per la feature finale, si aggiungono features di dipendenza al modello HDP (oltre a mantenere le features delle parole posizionali), ma non si vede alcun cambiamento nei risultati. Sebbene le features di dipendenza non abbiano ridotto il F-score, la loro utilità è discutibile in quanto la generazione delle funzionalità dal parser di Stanford[6] è computazionalmente costosa.

[6]Un programma che elabora la struttura grammaticale delle frasi utilizzato per generare analisi di dipendenza di frasi per una varietà di lingue.

Confrontando i risultati per HDP con quelli per LDA, HDP tende ad apprendere quasi il doppio del numero di sensi per parola target rispetto ai gold standard (e quindi sono utilizzati per la versione "Variable T" di LDA). Tuttavia, gli argomenti in più sono dominati da argomenti spazzatura e aumentano il F-score WSD per gli argomenti "genuini". Sulla base di questa intuizione, è stato eseguito LDA ancora una volta con la variabile T (e caratteristiche posizionali e di dipendenza), ma questa volta impostando T sul valore appreso da HDP, per dare a LDA la possibilità di utilizzare gli argomenti spazzatura. Ciò ha portato a un F-score di *0,66* in tutte le classi di parole (verbi = *0,71*, sostantivi = *0,62*), dimostrando che, sorprendentemente, anche per la stessa impostazione T, HDP raggiunge risultati superiori a LDA. Cioè, non solo HDP apprende T automaticamente, ma il modello di argomento appreso per un dato T è superiore a quello per LDA.

2.2.3 Identificazione di nuovi sensi

Dopo aver stabilito l'efficacia dell'approccio al WSI, si passa poi ad una applicazione dello stesso, per identificare le parole che hanno assunto nuovi sensi nel tempo, sulla base dell'analisi dei dati diacronici. L'approccio di modellazione dell'argomento è particolarmente interessante per questo task in quanto, non solo esegue simultaneamente WSI a livello di tipo e WSD a livello di token in base ai sensi indotti, ma è possibile riassumere i sensi indotti attraverso i contenuti dell'argomento (tipicamente utilizzando le parole dell'argomento con la più alta probabilità marginale).

I significati delle parole possono cambiare nel tempo; in particolare, le parole possono assumere nuovi sensi. Alcuni sensi delle parole non sono inclusi in molti dizionari o lessici computazionali, anche se sembrano essere di uso regolare, magari nel caso di testi relativi alla cultura pop o nei media online. L'identificazione manuale di tali nuovi sensi delle parole è una sfida nella lessicografia oltre al task di identificazione di parole nuove ed è essenziale per mantenere aggiornati i dizionari. Inoltre, i lessici che riflettono meglio l'uso contemporaneo potrebbero avvantaggiare le applicazioni di NLP che utilizzano inventari sensoriali. La sfida di identificare i cambiamenti nel senso delle parole è stata presa in considerazione solo di recente nella linguistica computazionale.

Metodo

Dati due corpus, uno di riferimento preso per rappresentare l'uso standard, e un secondo di testi più recenti, si identificano i sensi che sono nuovi per il secondo

corpus rispetto al corpus di riferimento. Per una data parola w, vengono riuniti tutti gli usi di w nel corpus di riferimento e nel secondo corpus, e si esegue il metodo HDP WSI su questo super-corpus per indurre i sensi di w. Quindi vengono contrassegnati tutti gli usi di w in entrambi i corpus con il loro unico senso che probabilmente è stato indotto automaticamente. Intuitivamente, se una parola w è usata in qualche senso s nel secondo corpus, e w non è mai usata in quel senso nel corpus di riferimento, allora w ha acquisito un nuovo senso, cioè s. Si cattura questa intuizione in un punteggio di novità ("Nov") che indica se una data parola w ha un nuovo senso nel secondo corpus, s, rispetto al corpus di riferimento, r:

$$Nov(w) = \max\left(\left\{\frac{p_s(t_i) - p_r(t_i)}{p_r(t_i)} : t_i \in T\right\}\right)$$

Dove $p_s(t_i)$ e $p_r(t_i)$ sono rispettivamente la probabilità del senso t_i nel secondo corpus e nel corpus di riferimento, calcolate usando stime di massima verosimiglianza livellate, e T è l'insieme dei sensi indotti per w. La novità è alta se c'è un senso t che ha una frequenza relativa molto più alta in s che in r e che è anche relativamente poco frequente in r.

Dati

Poiché vi è l'interesse all'identificazione di nuovi sensi delle parole, ci si concentra sui sensi delle parole coniati relativamente di recente. In particolare, viene preso come corpus di riferimento la parte scritta del British National Corpus[7] (BNC), costituita principalmente da testi in inglese britannico della fine del XX secolo, e un campione casuale di dimensioni simili di documenti provenienti dall'**ukWaC**[9][8] come secondo corpus, utilizzando TreeTagger[10] per tokenizzare e lemmatizzare entrambi.

C'è bisogno di un insieme di parole che abbiano notoriamente acquisito un nuovo senso tra la fine del XX e l'inizio del XXI secolo. Per garantire che le parole identificate dai dizionari abbiano effettivamente un nuovo senso nel campione ukWaC rispetto al BNC, viene esaminato l'uso di queste parole nei corpus. Si estrae così un campione casuale di 100 usi di ogni lemma (597) dal campione BNC e ukWaC e li si annotano per stabilire se corrispondano o meno al nuovo senso.

Al termine di questo processo si sono individuati 5 lemmi che hanno i sensi nuovi indicati nell'ukWaC rispetto al BNC. Ma si richiedono anche elementi

[7] `http://www.natcorp.ox.ac.uk/`

[8] Un corpus Web creato dal dominio .uk nel 2007 che include un'ampia gamma di tipi di testo.

che non abbiano acquisito un nuovo senso in ukWaC. Quindi per ciascuno dei 5 lemmi precedenti è stato identificato un lemma distrattore della stessa parte del discorso che ha una frequenza simile nel BNC.

Risultati

Viene calcolata la novità (Novelty) per tutti i 10 elementi nel set di dati, in base all'output della modellazione dell'argomento.

Lemma	Novelty	Freq. ratio	Novel sense freq.
domain (n)	116.2	2.60	41
worm (n)	68.4	1.04	30
mirror (n)	38.4	0.53	10
guess (v)	16.5	0.93	–
export (v)	13.8	0.88	28
founder (n)	11.0	1.20	–
cinema (n)	9.7	1.30	–
poster (n)	7.9	1.83	4
racism (n)	2.4	0.98	–
symptom (n)	2.1	1.16	–

FIGURA 2.5: Risultati della novità, del rapporto delle frequenze e della frequenza dei nuovi sensi.

I lemmi con un senso nuovo hanno punteggi di novità più alti rispetto ai distrattori secondo un test della somma dei ranghi di Wilcoxon[9] unilaterale ($p < .05$). Quando un lemma assume un nuovo senso, potrebbe anche aumentare di frequenza.

Si considera quindi anche un punto di riferimento in cui i lemmi sono classificati in base al rapporto tra la loro frequenza nel secondo corpus e in quello di riferimento. Questi risultati sono mostrati nella colonna "*Freq. ratio*" nella figura 2.5. La differenza tra i rapporti di frequenza per i lemmi con un senso nuovo e i distrattori non è significativa ($p > .05$). Essendo stato dimostrato che il metodo per identificare nuovi sensi può distinguere i lemmi che hanno un nuovo senso in un corpus rispetto a un altro da quelli che non lo hanno, ora viene considerato se può anche identificare automaticamente gli usi del nuovo senso indotto.

Per ogni lemma con un nuovo senso del gold standard, viene definito il nuovo senso indotto automaticamente come il singolo senso corrispondente al massimo nella formula della novità. Così viene calcolata la precisione, il richiamo e il F-score di questo nuovo senso rispetto al nuovo senso dei gold standard, basato

[9]Test dei segni per ranghi non parametrico che ipotizza che la variabile dipendente derivi da una variabile casuale continua misurabile almeno su intervalli.

sui 100 token annotati per ciascuno dei 5 lemmi con un nuovo senso. I risultati sono mostrati nelle prime tre colonne numeriche della figura 2.6.

Lemma	Topic Selection Methodology								
	Nov			Oracle (single topic)			Oracle (multiple topics)		
	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
domain (n)	1.00	0.29	0.45	1.00	0.56	0.72	0.97	0.88	0.92
export (v)	0.93	0.96	0.95	0.93	0.96	0.95	0.90	1.00	0.95
mirror (n)	0.67	1.00	0.80	0.67	1.00	0.80	0.67	1.00	0.80
poster (n)	0.00	0.00	0.00	0.44	1.00	0.62	0.44	1.00	0.62
worm (n)	0.93	0.90	0.92	0.93	0.90	0.92	0.86	1.00	0.92

FIGURA 2.6: Risultati per l'identificazione dei nuovi sensi nel gold-standard.

Nel caso di *export* e *worm* i risultati sono decisamente buoni, con precisione e richiamo entrambi superiori a *0.90*. Per *domain*, il basso richiamo è il risultato della maggior parte degli usi del nuovo senso dei gold standard ("dominio di Internet") che viene suddiviso tra due sensi indotti.

Si possono osservare grandi miglioramenti nel F-score per *domain* e *poster*. Questo risultato incoraggiante suggerisce che il perfezionamento dell'euristica di selezione dei sensi potrebbe teoricamente migliorare il metodo per l'identificazione di nuovi sensi e che l'approccio di modellizzazione dell'argomento proposto in questo articolo ha notevoli promesse per il rilevamento automatico di nuovi sensi.

Infine, si considera la terza colonna vedendo di nuovo un aumento del F-score a *0,92* per *domain*. Per questo lemma gli usi del nuovo senso dei gold standard sono stati suddivisi su più argomenti indotti, e quindi non si è sorpresi nel scoprire che un metodo in grado di selezionare più argomenti come nuovo senso, funzioni bene.

2.3 Bayesian WSI

La Word Sense Induction (WSI) è il compito della scoperta automatica di tutti i possibili sensi di una parola ambigua (polisemica). Esso è correlato, ma in maniera distinta, alla Word Sense Disambiguation (WSD) in cui si presume che i sensi siano conosciuti e lo scopo è quello di identificare il previsto significato della parola ambigua nel contesto. Sebbene di solito la maggior parte dei progetti sia dedicata al problema della disambiguazione, ci sono buone ragioni per credere che l'induzione dei sensi possa essere in grado di superare alcuni dei problemi associati alla WSD.

Poiché la maggior parte dei metodi di disambiguazione assegna i sensi in base a, e con l'aiuto di, dizionari o altre risorse lessicali, è difficile adattarli a nuovi

domini o a lingue in cui tali risorse sono scarse. Un problema correlato riguarda la granularità delle distinzioni di senso che è fissa e potrebbe non essere del tutto adatta per applicazioni diversificate. Al contrario, quando le distinzioni di senso sono dedotte direttamente dai dati, è più probabile che rappresentino il task e il dominio a portata di mano. C'è un piccolo rischio però che un senso importante venga tralasciato o che sensi irrilevanti influenzino i risultati.

L'induzione sensoriale viene generalmente trattata come un problema di clustering non supervisionato. L'input per l'algoritmo di clustering sono istanze della parola ambigua con i relativi contesti di accompagnamento (rappresentati da vettori di co-occorrenza) e l'output è un raggruppamento di queste istanze in classi corrispondenti ai sensi indotti. In altre parole, i contesti raggruppati nella stessa classe rappresentano uno specifico senso della parola.

In questo articolo, Bayesian WSI[11], si adotta un nuovo approccio bayesiano e si formalizza il problema dell'induzione in un modello generativo. Per ogni parola ambigua viene prima tracciata una distribuzione sui sensi, poi vengono generate parole di contesto in base a questa distribuzione. Si presume quindi che sensi diversi corrispondano a distinte distribuzioni lessicali. In questo framework, le distinzioni di senso sorgono naturalmente attraverso il processo generativo: il modello postula che i dati osservati (contesti di parole) siano esplicitamente destinati a comunicare una struttura latente (il loro significato).

2.3.1 LDA

Il lavoro è legato alla Latent Dirichlet Allocation[4] (**LDA**), un modello probabilistico di generazione del testo. LDA modella ogni documento utilizzando una combinazione di argomenti K, che sono a loro volta caratterizzati come distribuzioni su parole.

Le parole nel documento vengono generate campionando ripetutamente un argomento in base alla sua distribuzione e selezionando una parola dato l'argomento scelto. Mentre LDA genera parole da argomenti globali corrispondenti all'intero documento, il modello bayesiano invece genera parole da argomenti locali scelti in base a una finestra di contesto che gira attorno alla parola ambigua. Gli argomenti a livello di documento assomigliano a etichette di dominio generali (ad esempio, finanza, istruzione) e non possono modellare fedelmente distinzioni di significato più dettagliate. Quindi viene creato un modello individuale per ogni parola "ambigua" piuttosto che un modello globale per un'intera raccolta di documenti. Viene mostrato anche come più fonti di informazioni possono essere integrate direttamente senza modificare il modello probabilistico

sottostante. Ciò è in netto contrasto con i precedenti modelli basati su LDA che perlopiù prendono in considerazione solo le informazioni basate su parole.

2.3.2 Costruzione del modello induttivo

L'idea alla base dell'induzione sensoriale è che le informazioni contestuali forniscano importanti spunti sul significato di una parola. Ci si dovrebbe aspettare che sensi diversi vengano segnalati da diverse distribuzioni lessicali. Si può collocare l'induzione sensoriale in un contesto probabilistico, modellando le parole di un contesto attorno alla parola target ambigua come campioni di una distribuzione sensoriale multinomiale. Più formalmente, si scriverà $P(s)$ per la distribuzione sui sensi s di una parola target ambigua in una specifica finestra di contesto e $P(w|s)$ per la distribuzione di probabilità sulle parole di contesto w dato il senso s. Ogni parola w_i nella finestra di contesto viene generata campionando prima un senso dalla distribuzione dei sensi, poi scegliendo una parola dalla distribuzione senso-contesto. $P(s_i = j)$ denota la probabilità che il j-esimo senso sia stato campionato per l'i-esimo token di parola e $P(w_i|s_i = j)$ la probabilità che la parola contestuale abbia un senso j. Il modello specifica quindi una distribuzione sulle parole all'interno di una finestra di contesto:

$$P(w_i) = \sum_{j=1}^{S} P(w_i|s_i = j)P(s_i = j)$$

dove S è il numero di sensi. Si suppone che ogni parola target abbia C contesti e ogni contesto sia costituito da Nc token di parole. Si scriverà $\phi_{(j)}$ come abbreviazione per $P(w_i|s_i = j)$, ovvero la distribuzione multinomiale sulle parole per il senso j, e $\theta_{(c)}$ come abbreviazione per la distribuzione dei sensi nel contesto c. Seguendo il LDA si assumerà che la proporzione di miscelazione sui sensi θ verrà ricavata da Dirichlet con parametri α.Il ruolo dell'iperparametro α è quello di creare una distribuzione di senso uniforme. Viene posto anche un Dirichlet simmetrico β su ϕ. L'iperparametro β può essere interpretato come il conteggio delle ϕ sul numero di volte in cui le parole del contesto vengono campionate da un senso prima che venga osservata qualsiasi altra parola del corpus.

Il modello abbozzato nella figura 2.7 prende in considerazione solo la parola in formazione. I metodi sviluppati per la WSD supervisionata spesso utilizzano una varietà di fonti di informazioni basate non solo su parole ma anche su lemmi, parti del discorso, collocazioni e relazioni sintattiche[12]. Una prima idea di modifica è quella di utilizzare lo stesso modello trattando le varie

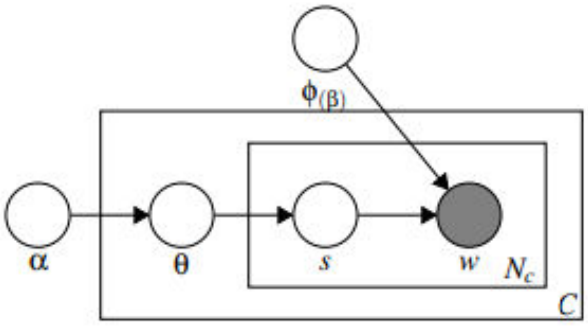


FIGURA 2.7: Modello di induzione dei sensi bayesiano.

caratteristiche come elementi simili a parole. In altre parole, si potrebbe semplicemente supporre che i contesti che si desidera modellare siano l'unione di tutte le caratteristiche. Anche se semplice, questa soluzione è indesiderabile: unisce le distribuzioni di categorie di features distinte in una singola ed è quindi concettualmente errato e può influire sulle prestazioni del modello. La soluzione migliore è trattare ogni fonte di informazioni (o tipo di caratteristica) individualmente e dopo combinarle tutte insieme in un modello unificato. Questo perché la finestra di contesto intorno alla parola obiettivo potrebbe avere rappresentazioni multiple, che condividono tutte la stessa distribuzione di senso.

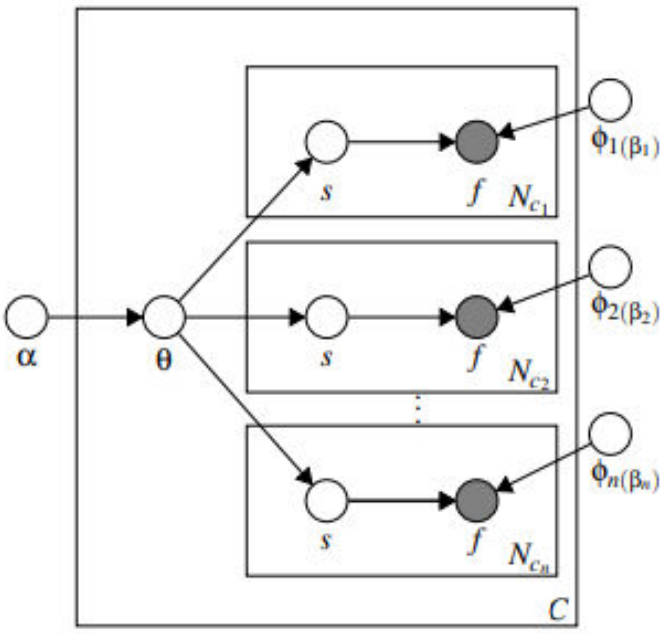


FIGURA 2.8: Modello di induzione dei sensi bayesiano esteso.

Nella figura 2.8 quindi possiamo vedere il modello esteso, dove ogni rettangolo interno (strato) corrisponde a un tipo di caratteristica distinto. Si assumerà l'indipendenza tra più strati con l'idea di modellare ogni livello nel modo più fedele possibile ai dati empirici, combinando allo stesso tempo le informazioni di tutti i livelli per stimare la distribuzione dei sensi di ciascuna istanza target.

Inferenza

La procedura di inferenza si basa sul campionamento di **Gibbs**[13]. La procedura comincia con l'inizializzazione casuale di tutte le variabili casuali non osservate. Ad ogni iterazione, ciascuna variabile s_i viene campionata dalla distribuzione condizionale $P(s_i|s_{-i})$ dove s_{-i} si riferisce a tutte le variabili diverse da s_i . Alla fine, la distribuzione sui campioni ricavata da questo processo convergerà alla distribuzione congiunta incondizionata $P(s)$ delle variabili non osservate (a condizione che siano soddisfatti determinati criteri).

Da queste assegnazioni, si determina la distribuzione di senso dell'istanza nel suo insieme. Questo è lo scopo della procedura di campionamento di Gibbs.

$$p(s_i|\overline{s}_{-i},\overline{f}) \propto \frac{\#(f_i,s_i)+\beta}{\#(s_i)+V\cdot\beta}\cdot\frac{\#m(s_i)+\alpha}{\#m+S\cdot\alpha}$$

Si vede come sia identico all'equazione di aggiornamento nel modello LDA originale basato sulle parole. L'algoritmo di campionamento fornisce stime dirette di s per ogni elemento del contesto. Tuttavia si è più interessati a stimare θ, la distribuzione senso-contesto, tenendo conto di tutte le assegnazioni di senso, senza rimuovere l'assegnazione i. Il sistema, quindi, etichetta ogni istanza con il singolo senso più probabile.

2.3.3 Valutazione

Ci si concentrerà esclusivamente sull'induzione di sensi per i sostantivi, dal momento che costituiscono la maggior parte delle parole di contenuto e per molti compiti e applicazioni sono la parte più frequente e più importante del discorso.

Dataset

Per la valutazione, è stato utilizzato il set di dati di riferimento **Semeval-2007**[14] rilasciato come parte del task di induzione e discriminazione dei sensi. Il set di dati contiene testi tratti dal corpus *Penn Treebank II*, una raccolta di

articoli della prima metà del Wall Street Journal (WSJ) del 1989. È annotato a mano con i sensi **OntoNotes**[15] e ha 35 sostantivi. Si sono utilizzati due corpus per l'addestramento poiché si volevano valutare le prestazioni del modello in diversi domini. Il British National Corpus (**BNC**)[10] è servito come corpus fuori dominio contenente circa 730.000 istanze dei 35 nomi target nel campione lessicale di Semeval. Il secondo corpus, interno al dominio, è stato costruito da porzioni selezionate del *Wall Street Journal*. Sono stati utilizzati tutti gli articoli degli anni 1987-89 e 1994 per creare un corpus di dimensioni simili al BNC, contenente circa 740 mila istanze delle parole target.

Metodo di valutazione

Sono stati usati due schemi di valutazione per analizzare i metodi di induzione sensoriale. Nel primo schema, l'output del sistema viene confrontato con il gold standard utilizzando metriche standard di valutazione del clustering. Però non vi è alcun tentativo di confrontare i sensi indotti con le etichette del gold standard. Nel secondo schema, il gold standard è suddiviso in un corpus di test e di addestramento. Quest'ultimo viene utilizzato per derivare una mappatura dei sensi indotti alle etichette gold standard. La mappatura viene quindi utilizzata per calcolare l' F-score del sistema sul corpus di test.

Purtroppo, il primo schema non è riuscito a discriminare tra i sistemi partecipanti. Il punto di riferimento di un cluster per parola ha superato in performance tutti i sistemi, tranne uno, che era solo marginalmente migliore. Lo schema ignora l'effettiva etichettatura e a causa della predominanza del primo senso nei dati, incoraggia un approccio a senso unico che è ulteriormente amplificato dall'uso di un inventario dei sensi a grana grossa. Ci si è concentrati quindi sul secondo schema. In questo, la maggior parte dei sistemi partecipanti ha superato in performance il punto di riferimento del senso più frequente e il resto ha ottenuto punteggi solo leggermente inferiori.

2.3.4 Sperimentazione

Il modello si presenta condizionato dagli iperparametri di Dirichlet, α e β, e dal numero di sensi S. Ulteriori parametri includono il numero di iterazioni per il campionatore di Gibbs e se agli strati vengono assegnati o meno pesi diversi. La strategia applicata è quella di fissare α e β ed esplorare le conseguenze della variazione di S. Il valore per l'iperparametro α è stato impostato su *0,02*,

[10] Una raccolta di 100 milioni di parole di campioni di lingua scritta e parlata da una vasta gamma di fonti.

ma sono stati sperimentati valori α compresi tra *0,005* e *1*. Il parametro β è stato impostato su *0,1* (in tutti i livelli). Gli esperimenti hanno utilizzato lo stesso numero di sensi per tutte le parole, poiché accordare questo numero individualmente per ogni parola sarebbe stato proibitivo. Si sono sperimentati valori che vanno da tre a nove sensi.

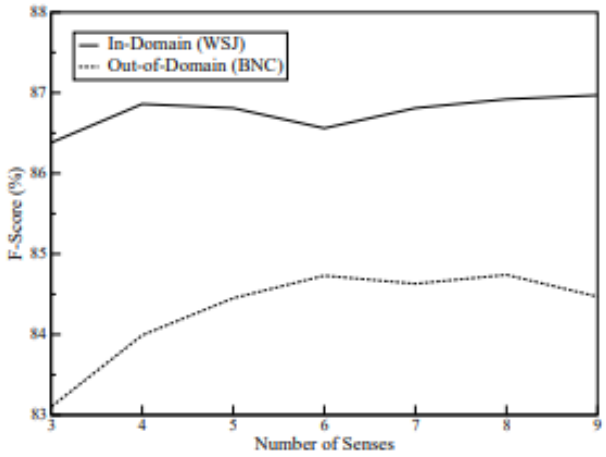


FIGURA 2.9: Performance del modello con un numero variabile di sensi sui corpus WSJ e BCN.

La Figura 2.9 mostra i risultati ottenuti per diversi numeri di sensi quando il modello viene addestrato rispettivamente sui corpus WSJ e BNC.

Per il modello addestrato su WSJ, le prestazioni raggiungono il picco nei quattro sensi, il che è simile all'ambiguità media nei dati di test. Per il modello addestrato sul BNC, invece, i risultati migliori si ottengono utilizzando il doppio dei sensi. L'utilizzo di un minor numero di sensi con il sistema addestrato BNC può comportare un calo della precisione di quasi il *2%*; ciò è dovuto al cambio di dominio. Poiché le divisioni sensoriali del dominio di apprendimento non corrispondono a quelle del dominio di destinazione, è necessaria una granularità più fine per comprendere tutte le distinzioni rilevanti.

Successivamente vengono esaminate quali categorie di caratteristiche individuali sono più istruttive nel compito di induzione sensoriale. Viene indagato anche se la loro combinazione, attraverso il modello a strati, produca miglioramenti delle prestazioni. Sono stati utilizzati 4 sensi per il sistema addestrato su WSJ e 8 per il sistema addestrato sul BNC.

Nella figura 2.10 (lato sinistro) vengono mostrate le prestazioni del modello nel caso dell'utilizzo di un solo livello. Lo strato composto da parole che ricorrono all'interno di una finestra di ± 10 parole (*10w*) e che rappresentano informazioni più ampie e topiche, dà di per sè i punteggi più alti. È seguito

1-Layer		5-Layers		Combination	
10w	86.9	-10w	83.1	**10w+5w**	**87.3%**
5w	86.8	-5w	83.0	5w+pg	83.9%
1w	84.6	-1w	83.0	1w+ng	83.2%
ng	83.6	-ng	83.0	10w+pg	83.3%
pg	82.5	-pg	82.7	1w+pg	84.5%
dp	82.2	-dp	84.7	10w+pg+dep	82.2%
MFS	80.9	all	83.3	MFS	80.9%

FIGURA 2.10: Modello dei F-score, sul corpus WSJ con uno strato (sinistra), 5 strati (centro) e una combinazione di strati (destra).

dalle finestre di parole ±5 (*5w*) e ±1 (*1w*), che rappresentano un contesto locale più immediato. Le parti del discorso p-grammi (*pg*) e le parole n-grammi (*ng*), da sole, ottengono punteggi inferiori, in gran parte a causa dell'eccessiva generalizzazione e della scarsità di dati. L'unico livello con il punteggio più basso è quello di dipendenza (*dp*); le informazioni sulle dipendenze sono molto informative quando presenti, ma estremamente scarse.

Al centro invece vengono mostrati i risultati ottenuti durante l'esecuzione del modello a strati con tutti i livelli tranne uno come input. Poiché si ha a che fare con livelli multipli, è coinvolto un elemento di sovrapposizione. Pertanto, ciascuno degli strati finestra-parola, nonostante abbia di per sé un livello di informazione relativamente alto, non causa così tanti danni quando manca, poiché gli altri strati compensano l'informazione topica e locale. L'assenza dello strato delle parole n-grammi, che fornisce informazioni locali specifiche, non ha un grande impatto quando sono presenti gli strati *1w* e *pg*.

Infine, nel lato destro ci sono le combinazioni più informative a due e tre strati. Ancora una volta, le dipendenze tendono a diminuire le prestazioni. D'altra parte, la combinazione di funzionalità che hanno prestazioni simili da sole è vantaggiosa. Si ottengono le migliori prestazioni complessive con un modello a due livelli che combina contesti topici (*+10w*) e locali (*+5w*).

Lo stesso viene replicato sul corpus BNC. Tuttavia, ci sono alcune differenze interessanti evidenti. Gli strati più radi, in particolare gli n-grammi di parole e le dipendenze, se la passano relativamente peggio. Ciò è previsto, poiché è probabile che le informazioni più precise, locali, varino fortemente tra i domini. Anche quando entrambi i domini si riferiscono allo stesso senso di una parola, è probabile che questa venga usata in un contesto immediato diverso e che le informazioni contestuali locali apprese in un dominio siano meno efficaci nell'altro. Si potrebbe osservare però che il modello combinato senza il livello di

dipendenza funzioni leggermente meglio di ciascuno dei singoli livelli.

2.4 WiC

Una delle principali limitazioni degli incorporamenti di parole (word embeddings) più comuni risiede nella loro natura statica: una parola viene associata allo stesso incorporamento indipendentemente dal contesto in cui appare. Quindi si dimostrano incapaci di riflettere la natura dinamica delle parole ambigue, ovvero parole che possono presentare diversi significati in base al loro uso in un preciso contesto.

Per ovviare a questa limitazione, ci sono state dozzine di proposte, principalmente suddivise in due categorie: incorporamenti multi-prototipo, che sfruttano clustering di contesto per imparare rappresentazioni distinte per significati individuali delle parole, e incorporamenti di parole contestualizzati, i quali invece creano un singolo dinamico incorporamento per una parola data che può adattarsi a diversi contesti per la parola. Tuttavia, nonostante la popolarità della ricerca su questi incorporamenti specializzati, esistono pochissimi benchmark per la loro valutazione.

Così viene proposto **WiC**[16], una nuova tipologia di dataset, che propone benchmark di alta qualità per le valutazioni di incorporamenti di parole sensibili al contesto. Si presenta provvedendo una serie di caratteristiche: adatto per la valutazione di un ampio range di tecniche includendo parole contestualizzate, la rappresentazione dei loro sensi e la loro disambiguazione; si mostra sotto forma di un dataset per la classificazione binaria dove parole identiche vengono accoppiate (in diversi contesti) ed infine è stato costruito utilizzando annotazioni di alta qualità curate da esperti.

Il task identificato è quindi quello di una classificazione binaria: ogni istanza possiede una parola target, *w*, che può essere un verbo o un nome, per la quale sono provvisti due contesti, *c1* e *c2*. Ogni contesto associa uno specifico significato di *w*. Il lavoro si basa sull'identificare se le occorrenze di *w* in *c1* e *c2* corrispondano allo stesso significato o meno. Quindi una istanza può diventare positiva o negativa in base alla corrispondenza dei contesti nei confronti del senso di *w*.

2.4.1 Procedura di costruzione del dataset

L'estrazione delle frasi contestuali è stata effettuata utilizzando degli esempi d'uso per parole presi da 3 risorse lessicali: **WordNet**[11], **VerbNet**[12] e **Wiktionary**[13]. WordNet è stato utilizzato come risorsa base, sfruttando i mappings di BabelNet come ponte tra Wiktionary e Verbnet per poter arrivare a WordNet. Gli esempi lessicografici costituiscono una base forte e sicura per la costruzione del dataset poiché sono stati curati in modo da essere chiaramente distinguibili tra i diversi sensi di una parola.

Per la compilazione del dataset, come detto prima, si è cercato di ottenere tutte le possibili istanze, positive e negative, da tutte le risorse, con la condizione che la forma superficiale della parola compaia in entrambi i contesti. Il numero totale degli esempi d'inizio estratti da tutte le risorse è di *23.949* per WordNet, *10.564* per Wiktionary e *636* per VerbNet.

La fase di test e sviluppo dei set si presenta con due limitazioni: non bisogna avere più di 3 istanze per la stessa parola target e frasi contestuali ripetute tra istanze. Queste limitazioni sono necessarie per l'ottenimento di un set bilanciato e vario che possa coprire più parole uniche possibili. Con questi limiti preimpostati, sono state suddivise *1600* e *800* istanze per il set di test e sviluppo rispettivamente. Le istanze rimanenti, i quali esempi non erano in sincronia con i test e lo sviluppo, hanno formato il dataset iniziale di training.

Potatura

Nonostante ce ne fossero pochissimi, tutte le risorse contenevano degli errori, come ad esempio tags incorretti o istanze deformate; perlopiù la stessa estrazione delle istanze e mappatura tra risorse non sempre era accurata. Quindi per ottenere delle risorse con un minor numero possibile di errori, tutti i set di addestramento, sviluppo e test sono stati semi-automaticamente processati in seguito, con piccole migliorie dove possibile o applicando la rimozione delle istanze problematiche. Poiché WordNet stesso si presenta come una risorsa finemente dettagliata, spesso diventa difficile distinguere i diversi sensi che una stessa parola può presentare. Per evitare situazioni con alti livelli di granularità delle risorse, è stata effettuata una potatura automatica che rimuovesse le istanze con distinzioni tra sensi molto sottili.

Per poter eseguire questa distinzione, pur essendo disponibili diverse strategie, è stato scelto l'adattamento di una piuttosto semplice e di rimuovere tutte

[11] Risorsa standard lessicografica Inglese.
[12] La più grande risorsa avente il più grande dominio basato sui verbi indipendente.
[13] Dizionario online costruito collaborativamente.

le coppie i quali sensi si presentavano con una connessione di primo livello nel grafo semantico di WordNet, includendo sensi familiari e quelli che appartenevano allo stesso supersenso. Ci sono in totale 44 supersensi in WordNet, che comprendono categorie semantiche come "*forma*", "*sostanza*" o "*evento*".

2.4.2 Controllo della qualità

Per verificare la qualità e la difficoltà del dataset creato e per valutare il limite superiore delle prestazioni a livello umano, sono stati campionati randomicamente 4 set da 100 istanze presi dal set di test, con una sovrapposizione di 50 istanze tra due degli annotatori. Ogni set è stato assegnato ad un annotatore al quale è stato chiesto di classificare ogni istanza nel caso in cui due occorrenze della stessa parola si riferissero allo stesso significato o meno. Gli annotatori non erano stati provvisti di alcun tipo di conoscenza da alcuna risorsa lessicale (WordNet). Più precisamente il numero dei sensi e delle distinzioni di senso della parola erano sconosciuti.

E' stato calcolato che l'accuratezza media umana sul dataset fosse dell' *80.0%* (risultati individuali di *79%, 79%, 80% e 82%*). Ciò viene considerato come una estimazione del limite superiore delle prestazioni del dataset a livello umano. Per la sezione sovrapposta, è stato calcolato che l'accordo tra gli annotatori fosse dell' *80%*. Si ricorda che gli annotatori non erano stati provvisti con distinzioni dei sensi per riconoscere scenari più complessi nel caso di modelli non-supervisionati (i quali non beneficiano da risorse di conoscenza basate sui sensi). Avendo avuto accesso ad esse, questo avrebbe aumentato sostanzialmente la barra della performance.

Per controllare l'efficacia della strategia di potatura applicata, inoltre, è stato campionato un set di 100 istanze dall'insieme di istanze che erano state potate dal dataset. Similarmente, agli annotatori è stato chiesto di classificare indipendentemente le istanze nel set. E' stato calcolato che la media dell'accuratezza in questo set fosse del *57% (56% e 58%)*, la quale è sostanzialmente più basso rispetto al set finale con potatura (*80%*). Questo indica il successo dell'utilizzo della strategia di potatura nell'incrementare la chiarezza semantica del dataset.

Statistiche

La tabella 2.2 mostra le statistiche delle diverse divisioni del WiC. Il set Test contiene un gran numero di parole target uniche, riflettendo quindi la varietà del dataset. La grande divisione per l'addestramento di 5,428 istanze fa sì che il

Split	Instances	Nouns	Verbs	Unique words
Training	5,428	49%	51%	1,256
Dev	638	62%	38%	599
Test	1,400	59%	41%	1,184

Tabella 2.2: Statistiche di diverse suddivisioni di WiC.

dataset sia adatto per vari algortimi supervisionati, includendo i modelli di deep learning. Soltanto il *36%* delle parole target nella divisione per il test si sovrappone con quelle presenti nel set di addestramento, con nessuna sovrapposizione delle frasi contestuali tra le divisioni.

2.4.3 Sperimentazione

Sono state effettuate sperimentazioni con una serie di tecniche riguardanti l'incorporamento di parole multi-prototipo e parole contestualizzate.

Incorporamento di parole contestualizzate

Uno dei modelli per l'incorporamento di parole contestualizzate più intraprendente è **Context2Vec**[17], il quale utilizza un percettrone[14] costruito sulla base di un modello linguistico bidirezionale **LSTM**[18]. Inizialmente è stato utilizzato **ELMo**[19], un modello basato sui caratteri che impara incorporamenti di parole dinamiche che possono cambiare in base al contesto. Veniva utilizzato per gli stati interni del modello linguistico basato sull'LSTM, pre-addestrato su un grande corpus testuale. Successivamente però è stato scoperto un modello di contestualizzazione più recente **BERT**[20]. La tecnica è costruita su precedenti rappresentazioni contestuali, tra cui ELMo, ma differisce dal fatto che, a differenza di quei modelli prevalentemente unidirezionali, BERT è bidirezionale, ovvero considera i contesti su entrambe le parti della parola target durante la rappresentazione. Quindi la sperimentazione ha proseguito con l'utilizzo di 2 modelli pre-addestrati di tipo BERT, sia nella forma base, che large. Circa il *22%* delle coppie nel set di test avevano almeno una delle loro parole target non coperte da questi modelli. Per tali casi, è stato utilizzato il tokenizer predefinito di BERT per dividere le parole sconosciute in sottoparole ed è stato prodotto l'incorporamento come il centroide degli incorporamenti delle sottoparole corrispondenti.

[14]Modello di rete neurale artificiale.

Incorporamento di parole multi-prototipo

Per questa tipologia di sperimentazione sono state utilizzate 3 tecniche che producono incorporamenti di parole pre-addestrati multi-prototipo:

- **JBT**[21], induce sensi diversi utilizzando il clustering di grafi costruiti usando incorporamenti di parole e ne crea per ogni cluster di sensi.
- **DeConf**[22], sfrutta la conoscenza codificata presente in WordNet. Per ogni senso estrae dalla risorsa lessicale il set delle parole semanticamente correlate (sense biasing), le quali a loro volta vengono utilizzate per creare gli incorporamenti di sensi.
- **SW2V**[23], una estensione del Word2Vec[24] per l'apprendimento sia di incorporamenti di parole che di sensi, producendo uno spazio vettorale condiviso di parole e sensi come risultato.

Per questi 3 metodi è stata seguita la strategia di disambiguazione dove per ogni esempio ritrovava l'incorporamento di sensi più vicino al vettore di contesti.

Punti di riferimento a livello di frase

Sono state eseguite sperimentazioni anche su due modelli per punti di riferimento che sono visti come task di similarità di contesti (frasi). Ritroviamo il sistema *BoW*, il quale vede una frase come un insieme di parole e costruisce un semplice incorporamento come media delle sue parole. Il secondo punto di riferimento è il *Sentence LSTM*, il quale a differenza di altri modelli, non ottiene esplicitamente delle rappresentazioni codificate delle parole target o delle frasi. Sono stati utilizzati due semplici classificatori binari nelle sperimentazioni:

- **MLP**, una semplice rete neurale con 100 neuroni nascosti e uno di output, settata sul set di sviluppo.
- **Threshold**, un semplice classificatore basato sulla distanza del coseno dei due vettori di input, settato sul set di sviluppo.

Infine dalla figura 2.11 possiamo vedere i risultati finali su WiC. In generale il dataset si dimostra essere molto complesso per tutte le tecniche usate, anche con il miglior modello (BERT), provvedendo intorno al *15.5%* di assoluto miglioramento sul punto di riferimento *Random*. Tra i due classificatori, il threshold si dimostra essere più efficiente dell' MLP il quale non era adatto con dati di addestramento minori. Tra i modelli per la contestualizzazione basati

	MLP	Threshold
Contextualized word-based models		
Context2vec	57.9 ± 0.9	59.3
$ELMo_1$	56.4 ± 0.6	57.7
$ELMo_3$	57.2 ± 0.8	56.5
$BERT_{base}$	**60.2** ± 0.4	65.4
$BERT_{large}$	57.4 ± 1.0	**65.5**
Multi-prototype models		
DeConf*	52.4 ± 0.8	58.7
SW2V*	54.1 ± 0.5	58.1
JBT	54.1 ± 0.6	53.6
Sentence-level baselines		
BoW	54.2 ± 1.3	58.7
Sentence LSTM	53.1 ± 0.9	

FIGURA 2.11: Percentuale dell'accuratezza delle prestazioni di diversi modelli sul dataset WiC.

su LSTM, Context2Vce, che non include gli incorporamenti delle parole nelle sue rappresentazioni, si dimostra molto più competitivo rispetto ad ELMo. Nonostante ciò, nessuno dei due riesce a superare BoW. Inoltre tra le tecniche di multi-prototipazione DeConf è il migliore: indirettamente prende beneficio dalle informazioni a livello dei sensi dai dataset codificati di WordNet, nei suoi incorporamenti. Lo stesso si può applicare per SW2V, il quale fa leva su conoscenza presa da una risorsa lessicale molto più grande, ovvero BabelNet.

2.5 XL-WiC

Una delle proprietà desiderabili dei modelli di contestualizzazione, come **BERT**[20] e i suoi derivati, risiede nella loro capacità di associare rappresentazioni dinamiche a delle parole, ovvero degli incorporamenti (embeddings) che possono cambiare a seconda del contesto. Questo fornisce la base del modello per distinguere diversi significati (sensi) di parole senza la necessità di ricorrere a una fase di disambiguazione di senso esplicita. Il quadro di valutazione convenzionale per questa proprietà è stato il Word Sense Disambiguation[25] (WSD). Tuttavia, i benchmark di valutazione per WSD sono solitamente legati a inventari sensoriali esterni (spesso WordNet[26]), rendendo estremamente difficile valutare sistemi che non modellano esplicitamente distinzioni tra sensi nei loro inventari, restringendo di fatto il benchmark a tecniche di rappresentazione di sensi basate sugli inventari e a sistemi di WSD.

Il progetto **Word-in-Context**[16] (WiC), sotto forma di dataset, risolve la dipendenza da inventari di senso (con WSD) riformulando il compito di disambiguazione standard come un semplice problema di classificazione binaria: data una parola target *w* in due diversi contesti, *c1* e *c2*, il compito è quello di identificare se lo stesso significato di *w* è inteso in entrambi i contesti o meno. Nonostante abbia permesso ad una gamma significativamente più ampia di modelli di accedere a valutazioni dirette di WSD, WiC è limitato alla lingua inglese, impedendo così la valutazione dei modelli in altri lingue e in contesti interlinguistici.

Si propone così un nuovo benchmark di valutazione chiamato **XL-WiC**[27], che estende il dataset WiC a ben 12 nuove lingue di varie famiglie linguistiche e con diversi gradi di disponibilità delle risorse: Bulgaro (BG), Cinese (ZH), Croato (HR), Danese (DA), Olandese (NL), Estone (ET), Persiano (FA), Francese (FR), Tedesco (DE), Italiano (IT), Giapponese (JA) e Coreano (KO). Con oltre 80 mila istanze, questo benchmark può servire come un framework affidabile per le valutazioni per modelli contestualizzati in un ampio range di linguaggi eterogenei.

Inoltre sono stati testati su di esso diversi modelli multilinguisti preaddestrati, dimostrando come siano generalmente efficaci nel trasferire la conoscenza della distinzione dei sensi dall'inglese ad altre lingue con una impostazione zero-shot[15]. Tuttavia, con l'aumento di dati di formazione disponibili per i linguaggi target, gli approcci monolingue guadagnano terreno, superando di gran lunga le loro controparti multilinguistiche.

2.5.1 Word Sense Disambiguation

La capacità di identificare il senso inteso di una parola polisemica in un dato contesto è uno dei problemi fondamentali della semantica lessicale. Di solito viene affrontato con due diversi tipi di approccio che si basano su corpus di annotazioni di sensi oppure basi di conoscenza.

Queste attività sono inquadrate come problemi di classificazione, dove la disambiguazione di una parola è definita come la selezione di uno dei sensi predefiniti della parola, elencati da un inventario dei sensi. Questo però comporta diverse limitazioni come la restrizione dei sensi soltanto a quelli definiti dall'inventario, oppure la costrizione del sistema WSD a modellare esplicitamente le distinzioni dei sensi al livello di granularità definito dall' inventario stesso.

[15]Configurazione in cui un modello può imparare a riconoscere cose che non ha visto esplicitamente prima durante l'addestramento.

2.5.2 Procedura di costruzione del dataset

Il framework si basa sull'originale dataset WiC, estendendolo per molteplici linguaggi. Ogni istanza del dataset originale è composta da una parola target e da due frasi dove le parole target occorrono. Il task è di classificazione binaria: decide quando lo stesso senso della parola target è inteso nei due contesti o meno; è stato costruito utilizzando frasi d'esempio da risorse come **Wiktionary**, **WordNet** e **VerbNet**.

XL-WiC è stato costruito sugli usi d'esempio delle parole negli inventari dei sensi: questi sono curati in modo da essere autonomi e chiaramente distinguibili attraverso i diversi sensi di una parola; pertanto, forniscono una base affidabile per la classificazione binaria. Specificatamente per una parola w e per tutti i suoi sensi $\{s_1^w, ..., s_n^w\}$, vengono estratti dall'inventario tutti i suoi esempi d'uso. Successivamente si accoppiano gli esempi che corrispondono allo stesso senso w_i per formare una istanza positiva (true label) mentre esempi di sensi diversi (s_i^w e s_j^w dove i $\neq$ j) sono accoppiati come istanze negative (false label). Per questa estensione del dataset originale sono stati sfruttati due principali inventari di senso: WordNet multilinguistico e Wiktionary.

WordNet multilinguistico

La risorsa fu originariamente costruita come un database lessicale inglese nel 1995, ma sin da allora ci sono stati molti sforzi nell'estenderlo ad altri linguaggi. Si è preso vantaggio di queste estensioni per costruire XL-WiC e in particolare sono state processate le versioni di WordNet per la lingua Bulgara, Cinese, Croata, Danese, Olandese, Estone, Coreana e Persiana.

Per l'utilizzo del **FarsNet**[28], poiché questo comprende 30 mila synsets (insiemi di sinonimi) con oltre 100 mila parole, molti di questi furono mappati nel database inglese ma ognuno di essi provvedeva solo un esempio d'uso per una parola target. Questo impedisce l'applicazione dell'estrazione automatica degli esempi positivi. Pertanto per la costruzione del set Persiano è stata utilizzata una procedura semi-automatica: per ogni parola, si estraevano tutti gli esempi d'uso dal dataset originale e si chiedeva agli annotatori di raggrupparli in coppie negative e positive. In questo modo si poteva mirare a creare un dataset impegnativo con una distinzione tra sensi che fosse facilmente interpretabile anche per l'uomo.

WordNet è spesso considerato essere una risorsa molto dettagliata, specialmente per i verbi, tanto che in alcuni casi l'esatto significato di una parola può

essere difficile da valutare, persino per gli umani. Per far fronte a questa problematica si è seguito il procedimento usato in WiC e filtrato tutte le coppie i cui sensi target erano collegati da un margine nella rete semantica di WordNet o se appartenevano allo stesso supersenso; ad esempio "*Un organizzazione creata per scopi di business*" e "*Una istituzione creata per gestire business*", sono raggruppati sotto lo stesso supersenso in WordNet, "*Gruppo*". Infine tutti i dataset sono divisi in set di sviluppo e test. Dopo questa fase, si avrà lo stesso numero di istanze positive e negative.

Lang.	Target Word	Sentence 1	Sentence 2	Label
EN	Beat	We beat the competition.	Agassi beat Becker in the tennis championship.	True
DA	Tro	Jeg tror på det, min mor fortalte.	Maria troede ikke sine egne øjne.	True
ET	Ruum	Ühel hetkel olin väljaspool aega ja ruumi.	Ümberringi oli lõputu tühi ruum.	True
FR	Causticité	Sa causticité lui a fait bien des ennemis.	La causticité des acides.	False
KO	뜸팀	뜸팀이 있는지 없는지 세어 보시오.	그 아이 하는 짓에 뜸팀이 있다면 모두 이 어미 괴이지요.	False
ZH	發	建築師希望發大火燒掉城市的三分之一。	如果南美洲氧壓偏低,則印度可能發乾旱	True
FA	صرف	صرف غذا نیم ساعت طول کشید.	معلم صرف افعال ماضی عربی را آموزش داد.	False

FIGURA 2.12: Istanze d'esempio da XL-WiC per diversi linguaggi.

Wiktionary

Wiktionary è uno dei più ricchi e collaborativi database lessicali gratuiti, disponibile per decine di lingue. In questa risorsa disponibile online, ogni parola è provvista di definizioni per i suoi vari significati potenziali, alcuni dei quali sono abbinati ad esempi d'uso. Tuttavia, ogni linguaggio ha un formato specifico, quindi la compilazione di questi esempi richiede un'attenta analisi specifica del linguaggio.

Sono stati estratti esempi per 3 linguaggi europei per i quali non si aveva a disposizione materiale basato su WordNet: Francese, Tedesco e Italiano. Una volta compilati questi esempi, il processo di costruzione del dataset finale è stato analogo a quello per i dataset basati su WordNet, ad eccezione della fase di filtraggio, non realizzabile in quanto le voci di Wiktionary non sono collegate attraverso relazioni paradigmatiche come su WordNet. Quindi poiché nel caso di Wiktionary, il numero di esempi era considerevolmente più alto, è stato compilato un set di training specifico per i linguaggi, che ha permesso un confronto tra modelli interlinguistici e monolinguistici. Così tutti i dataset di Wiktionary sono suddivisi in suddivisioni bilanciate di formazione, sviluppo e test, dove in ognuna vi è un ugual numero di istanze positive e negative.

Split	Stat	WiC	Multilingual WordNet									Wiktionary		
		EN	BG	DA	ET	FA	HR	JA	KO	NL	ZH	DE	FR	IT
Train	Instances	5428	–	–	–	–	–	–	–	–	–	48 042	39 428	1144
	Unique Words	1265	–	–	–	–	–	–	–	–	–	23 213	20 221	721
	Avg. Context Len	16.8	–	–	–	–	–	–	–	–	–	32.7	32.3	23.2
Dev	Instances	638	998	852	98	200	104	208	404	250	3046	8870	8588	198
	Unique Words	599	354	542	63	174	82	137	183	150	867	4383	3517	136
	Avg. Context Len	17.1	8.4	32.6	19.8	24.8	17.9	20.8	5.7	20.1	45.7	32.5	34.3	23.2
Test	Instances	1400	1220	3406	390	800	408	824	1014	1004	5538	24 268	22 232	592
	Unique Words	1184	567	2088	276	533	305	476	475	600	1888	11 734	3517	394
	Avg. Context Len	17.2	8.5	32.6	19.4	23.5	18.1	20.6	6.0	19.9	46.0	32.9	36.4	23.4

FIGURA 2.13: Statistiche per i dataset di WordNet e Wiktionary per diversi linguaggi

2.5.3 Statistiche

La figura 2.13 mostra le statistiche di tutti i dataset, includendo il numero totale delle istanze, parole uniche e la media della lunghezza del contesto. I dataset basati su Wiktionary sono sostanzialmente più grandi di quelli basati su WordNet, inoltre provvedono set di training. I dataset cinesi presentano contesti più lunghi in media e contengono il maggior numero di istanze di sviluppo e test tra i dataset basati su WordNet. Il coreano, d'altro canto, è quello con i contesti più corti, il che è previsto data la sua natura agglutinante. Per i corpus di training invece, i dataset tedeschi e francesi contengono almeno 10 volte il numero di istanze rispetto al set di training inglese. Questo permette di elaborare una comparazione su larga scala tra ambienti interlinguistici e monolinguistici così come un'analisi few-shot[16].

Per verificare l'attendibilità dei dataset, è stata effettuata una valutazione manuale per quei linguaggi a cui gli annotatori avevano avuto accesso. Per questo motivo, è stato fornito un set di 100 istanze, scelte casualmente da ogni dataset, al corrispondente annotatore nella lingua target. Gli annotatori erano tutti madrelingua della lingua target presentando un'educazione di alto livello. Sono stati provvisti di linee guida minimali: una breve spiegazione del compito e pochi esempi correlati. Non sono state provviste alcun tipo di risorse (o altre istruzioni dettagliate) agli annotatori proprio con l'obiettivo di creare un dataset impegnativo con distinzioni tra sensi che potessero essere facilmente interpretabili per un inesperto. Data una istanza, ovvero una coppia di frasi contenenti una stessa parola target, il loro compito consisteva nel segnare con etichetta *Vero* o *Falso*, a seconda del significato inteso della parola nei due contesti.

[16]L'apprendimento few-shot consiste nel fare predizioni su un numero limitato di esempi.

WiC	WordNet						Wiktionary	
EN	DA	FA	IT	JA	KO	ZH	DE	IT
80.0*	87.0	97.0	82.0	75.0	76.0	85.0	74.0	78.0

FIGURA 2.14: Prestazione umana (in termini di accuratezza) per diversi linguaggi in XL-WiC.

La figura 2.14 riporta un campione di prestazione umana per 8 dataset in XL-WiC. Tutte le figure di accuratezza sono attorno l'*80%*. L'unica eccezione è data dal Persiano, per il quale gli annotatori si sono accordati su un'etichetta gold standard nel *97%* delle istanze (secondo la media). Questo conferma l'enfasi sulla procedura di annotazione, per questo dataset creato manualmente, di avere distinzioni tra sensi che sono facilmente interpretabili dagli umani.

Così come per Wiktionary, gli accordi umani sono più bassi rispetto quelli per la controparte di WordNet. Ciò era in parte previsto poiché la fase di filtraggio basato sul network non era fattibile nei casi dei dataset di Wiktionary a causa della natura della risorsa sottostante.

2.5.4 Sperimentazione

Per la parte di sperimentazione, è stato implementato un semplice, ma efficace, punto di riferimento basato su un codificatore di testo regolato su Transformer[29] e un classificatore di regressione logistico. Il modello prende come input i due contesti e come prima azione li tokenizza, suddividendo le parole di input in token secondari. Le rappresentazioni codificate delle parole target vengono poi concatenate e inviate al classificatore logistico. Per quei casi in cui la parola target era stata suddivisa dal tokenizer in più sub-token, si è seguito BERT[20] ed è stata considerata la rappresentazione del suo primo sub-token. Per quanto riguarda il codificatore di testo, si sono effettuati gli esperimenti utilizzando tre diversi modelli multilingue, vale a dire la versione multilingue di BERT (mBERT) e la versione base e large di **XLM-RoBERTa** [30] (XLMR-base e XLMR-large, rispettivamente).

Impostazioni di Valutazione

Sono state considerate più configurazioni, in dipendenza dai dati utilizzati per la fase di addestramento e di messa a punto (fine-tuning).

- **Cross-Lingual Zero Shot**, mira ad accedere alle capacità dei modelli multilinguistici nel trasferire conoscenza dall'inglese alle altre lingue. Per l'addestramento è stato usato il set di WiC e per la messa a punto sempre il set di svluppo di WiC oppure il set specifico per la lingua di XL-WiC.
- **Multilingual Fine-Tuning**, i modelli sono addestrati inizialmente sul set di addestramento inglese di WiC e successivamente messi a punto sui set di sviluppo delle lingue target di XL-WiC.
- **Monolingual**, ogni modello viene addestrato soltanto sul corrispondente set di addestramento della lingua target. Nel caso dei dataset di WordNet, dove non sono disponibili set di addestramento, sono stati utilizzati i set di sviluppo al loro posto con una suddivisione di 9:1. Lo stesso vale per i dataset di Wiktionary.
- **Translation**, vengono utilizzati modelli di traduzione basati su reti neurali per tradurre il set di addestramento o di test, essenzialmente per ridurre il problema interlinguistico ad uno monolinguistico.

Risultati

Gli esperimenti sono stati suddivisi principalmente in due parti, sulla base dei dataset di test, ovvero WordNet e Wiktionary.

Per WordNet, XLMR-large ottiene i migliori risultati, mentre BERT e XLMR-base ottengono dei risultati pressoché simili. Infatti la fase preponderante di pre-addestramento e il numero di parametri di XLMR-large, hanno avuto un ruolo importante nell'ottenimento di buoni risultati. Per quanto riguarda le impostazioni di traduzione, le prestazioni generalmente rimangono leggermente indietro rispetto alla controparte interlinguistica zero-shot. Ciò dimostra che l'utilizzo di dati in inglese di buona qualità e di modelli multilinguistici rappresenta una notevole differenza nell'addestramento rispetto all'uso di traduzioni automatiche dei dati.

Per Wiktionary invece, a differenza dei risultati proposti da WordNet, i modelli sono meno efficaci nelle configurazioni zero-shot, di almeno la metà. Questo può essere attribuito alla dimensione dei dati disponibili di addestramento. Infatti i set di Wiktionary sono molto più grandi, fornendo quindi dati sufficienti ai modelli per generalizzare meglio. Ancora una volta XLMR-large si dimostra essere il miglior modello in una configurazione zero-shot e un'alternativa competitiva ai modelli specifici per i linguaggi, come BERT, in un ambiente monolinguistico.

Model	**EN**	**BG**	**DA**	**ET**	**FA**	**HR**	**JA**	**KO**	**NL**	**ZH**
										Train: *EN* – ***Dev***: *Target Language*
mBERT	–	59.18	64.59	63.08	70.38	64.95	59.95	63.31	64.04	70.48
XLMR-base	–	60.74	64.80	60.77	67.75	62.50	57.65	66.96	61.85	65.78
XLMR-large	–	**66.48**	**71.10**	**68.72**	**73.63**	**72.30**	**60.92**	**69.63**	**69.62**	**73.15**
										Train: *EN+Target Language* – ***Dev***: *EN*
mBERT	–	71.72	62.62	63.08	69.38	**72.30**	60.92	70.91	62.95	76.72
XLMR-base	–	64.51	64.45	60.00	65.38	71.57	58.37	65.68	64.54	73.46
XLMR-large	–	**75.41**	**70.52**	**68.97**	**73.75**	69.61	**63.11**	**73.47**	**74.50**	**77.52**
										Train: *EN+All Languages* – ***Dev***: *EN*
mBERT	–	73.03	65.09	62.31	73.63	72.30	65.53	71.01	67.73	76.53
XLMR-base	–	67.30	67.62	59.49	64.50	66.18	57.77	67.06	66.33	71.02
XLMR-large	–	**78.44**	**71.49**	**72.05**	**78.25**	**76.96**	**66.38**	**76.53**	**77.49**	**78.95**
										Train: *Target Language* – ***Dev***: *Target Language*
mBERT	66.71	**82.30**	62.13	58.21	63.75	77.45	**61.04**	70.71	64.84	76.09
XLMR-base	64.36	79.75	64.00	**64.36**	66.25	**79.17**	58.86	70.61	66.33	78.11
XLMR-large	70.14	82.05	**66.53**	59.23	68.00	76.72	55.22	**73.08**	**69.42**	**81.83**
L-BERT	69.60	81.23	62.60	58.46	**76.63**	76.47	56.07	58.68	68.73	77.36

FIGURA 2.15: Risultati dei set di test di WordNet usando dati specifici per i linguaggi, per addestramento o messa a punto (fine-tuning).

	Model	**DE**	**FR**	**IT**
Z-Shot	mBERT	58.27	56.00	58.61
	XLMR-base	58.30	56.13	55.91
	XLMR-large	**65.83**	**62.50**	**64.86**
Mono	mBERT	81.58	73.67	71.96
	XLMR-base	80.84	73.06	68.58
	XLMR-large	**84.03**	76.16	72.30
	L-BERT	82.90	**78.14**	**72.64**

FIGURA 2.16: Risultati dei set di test di Wiktionary in diverse impostazioni di addestramento.

Capitolo 3

Metodologia

Lo scopo del progetto è quello di eseguire una induzione sul numero dei significati di un insieme di parole target utilizzando tecniche di semantica distribuzionale. Le parole verranno selezionate solo se polisemiche, esplorando due tipologie di *dizionari online*, ovvero MultiWordNet e Wiktionary.

Questo capitolo ha il compito di fornire una panoramica completa dell'architettura del sistema, mediante gli algoritmi e le scelte architetturali utilizzate per raggiungere l'obiettivo prefissato.

3.1 MultiWordNet

WordNet[26] è un ampio database lessicale dell'inglese, anche detto *Priceton WordNet*[1]. Sostantivi, verbi, aggettivi e avverbi sono raggruppati in insiemi di sinonimi cognitivi (synsets), ognuno dei quali esprime un concetto distinto. I synset sono interconnessi per mezzo di relazioni concettuali-semantiche e lessicali.

Nel nostro caso, è stato esplorato il MultiWordNet[2], ovvero un database lessicale multilinguistico dove il WordNet italiano è strettamente allineato a quello originale inglese. I synset italiani vengono creati in corrispondenza dei synset del Princeton WordNet, quando possibile, e le relazioni semantiche vengono importate dai corrispondenti synset inglesi.

Si può dire che WordNet assomigli superficialmente a un thesaurus[3], in quanto raggruppa le parole in base al loro significato. Tuttavia, ci sono alcune importanti distinzioni:

[1] Elaborato dal linguista George Armitage Miller presso l'Università di Princeton.

[2] `https://multiwordnet.fbk.eu/english/home.php`

[3] Lessico dei termini relativi a un ambito generale o specifico di conoscenze, collegati tra loro in una rete gerarchica e relazionale.

- **WordNet** interconnette non solo le forme delle parole (stringhe di lettere) ma anche i loro sensi specifici, quindi le parole che si trovano molto vicine l'una all'altra nella rete sono semanticamente disambiguate;
- **WordNet** etichetta le relazioni semantiche tra le parole, mentre i raggruppamenti di parole in un thesaurus non seguono alcun modello esplicito diverso dalla somiglianza di significato.

3.1.1 Estrazione di parole polisemiche

La relazione principale tra le parole in WordNet è la *sinonimia*. I sinonimi, parole che denotano lo stesso concetto e sono intercambiabili in molti contesti, sono raggruppati in insiemi non ordinati (synsets). Ciascuno dei 117.000 synset di WordNet è collegato ad altri synset per mezzo di un piccolo numero di relazioni concettuali. In WordNet le forme di parole con diversi significati distinti sono rappresentate in altrettanti synset distinti. Pertanto, ogni coppia forma-significato in WordNet è unica.

Ma vi è un'altra importante relazione, la *polisemia*, ovvero la proprietà che ha una parola di esprimere più significati. Nella ricerca delle parole target, ci siamo basati proprio su questa relazione: all'interno di WordNet non è definita direttamente come la relazione di sinonimia, quindi si è creato un algoritmo che potesse ritrovare parole polisemiche, esplorando il database, sulla base del conteggio di quanti synset una parola può contenere. Se questa avesse contenuto almeno più di un synset, allora sarebbe stata considerata polisemica, altrimenti sarebbe stata scartata.

Ovviamente di queste parole polisemiche, viene estratto il lemma. In lessicografia, il lemma, è la forma di citazione di una parola in un dizionario; rappresenta la forma di base da cui deriva un intero sistema flessionale nominale, aggettivale e verbale; ad esempio, nel caso della nostra lingua, si troverà nel vocabolario il nome al singolare, l'aggettivo al maschile. Quindi si dimostra importante l'estrazione del lemma, soprattutto nei confronti della classe semantica dei verbi, per non incorrere in tutte le molteplici flessioni.

Per quanto riguarda ciò, sappiamo che la maggior parte delle relazioni di WordNet collega parole dalla stessa parte del discorso (**POS**). Pertanto, consiste in realtà di quattro sottoreti, una per nomi, verbi, aggettivi e avverbi, con pochi puntatori cross-POS. È proprio tramite l'utilizzo di questi collegamenti che abbiamo basato l'estrazione della parole, per poi suddividerle nelle 4 classi semantiche corrispettive.

Le parole chiave utilizzate erano:

- **N**, per identificare la classe dei nomi;
- **V**, per identificare la classe dei verbi;
- **R**, per identificare la classe dei avverbi;
- **A**, per identificare la classe dei aggettivi.

3.2 Wiktionary

Wiktionary[31] è un progetto multilingue basato sul Web per creare un dizionario di termini a contenuto gratuito in tutte le lingue naturali. È modificato in modo collaborativo tramite un wiki, infatti come il suo progetto gemello Wikipedia[4], Wiktionary è gestito dalla Wikimedia Foundation ed è scritto in collaborazione da volontari.

È disponibile in 186 lingue e in inglese semplice; le voci corrispondenti sono collegate tra diverse edizioni linguistiche. Ogni edizione linguistica comprende un dizionario multilingue con una notevole quantità di voci. La maggior parte delle edizioni linguistiche di Wiktionary fornisce definizioni e traduzioni di termini da molte lingue e alcune di esse offrono informazioni aggiuntive che si trovano tipicamente nei thesauri.

3.2.1 Estrazione di parole polisemiche

Per poter accedere ai dati del Wiktionary italiano, sono stati utilizzati 4 file JSON[5], ciascuno per ogni classe semantica (nomi, aggettivi, verbi, avverbi), i quali sono stati creati da un processo di estrazione effettuato all'interno di un progetto chiamato **Wiktextract**.

Wiktextract[32] è il primo estrattore in grado di espandere i modelli di Wiktionary ed eseguire moduli Lua[6]; molte parti delle pagine di Wiktionary sono generate dai cosiddetti moduli. Mentre alcuni estrattori utilizzano le pagine HTML pre-generate come fonte, ci sono anche informazioni utili che non sono facilmente accessibili in questo maniera; quindi il formato principale di distribuzione dei dati scelto è JSON, utilizzando un oggetto JSON per riga del file, con solitamente una parte del discorso per parola in ogni oggetto. Questo formato è facile da analizzare nella maggior parte dei linguaggi di programmazione.

L'oggetto JSON su ciascuna riga descrive un reindirizzamento o una parte del discorso per una parola. Ogni parola è rappresentata da un oggetto JSON

[4] `https://www.wikipedia.org/`
[5] `https://kaikki.org/dictionary/Italian/`
[6] `https://www.lua.org`

(una mappatura del valore chiave, corrispondente a un dizionario in Python). Ogni voce ha i seguenti campi: *word* è la forma della parola, *lang* è il nome della lingua, *pos* è la parte del discorso (ad es. nome, verbo, aggettivo, avverbo) e *senses* è un elenco di sensi della parola (sotto forma di un array JSON). Ci sono anche molti altri campi possibili, come traduzioni, forme, iperonimi, sinonimi, ecc. Ogni significato di una parola è rappresentato da un oggetto JSON che in genere ha i seguenti campi: *glosses* e *raw_glosses* che contengono i vari sensi o significati che una parola può possedere, *form_of* che contiene il lemma per forme flesse (per i verbi ad esempio), e una serie di altri campi.

Pertanto, è stato applicato lo stesso algoritmo utilizzato per MultiWordNet (3.1): una volta selezionata la parola da verificare, si effettua un conteggio del numero di sensi che contiene; in questo caso sarà basato sul numero di *raw_glosses* che appariranno per ogni parola. Se il numero restituito è maggiore almeno di 1, allora la parola sarà polisemica, altrimenti verrà scartata. Grazie alla presenza del campo *form_of* si potrà sempre ricavare il lemma della parola, nel caso dei verbi, altrimenti si utilizzerà normalmente il campo *word*.

3.3 itWaC

Il web-corpus itWaC è un corpus italiano composto da testi raccolti da Internet. È articolato da 1,5 miliardi di parole ed è stato preparato da Marco Baroni. I testi sono contrassegnati e lemmatizzati con lo strumento *TreeTagger*[7]: viene utilizzato per annotare il testo con informazioni su parti del discorso e lemmi. Inoltre, gli utenti possono esplorare il comportamento grammaticale e collocazionale delle parole italiane come risultato di un word sketch grammar[8].

Pertanto dopo aver effettuato la fase di estrazione di parole polisemiche, ci si è occupati della fase di confronto e ritrovamento di esse all'interno del corpus italiano: se le parole fossero state ritrovate allora avrebbero potuto esser restituite assieme alla loro frequenza assoluta.

3.3.1 Frequenza assoluta

Consideriamo una variabile statistica X rilevata su un campione di n unità statistiche e che possa manifestarsi con k possibili modalità $v_1, v_2, ..., v_k$. La frequenza assoluta della modalità v_i con $1 \leq i \leq k$, si indica con $f(v_i)$ ed è un

[7] `https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`

[8] Un insieme di regole che definiscono le relazioni grammaticali in un Word Sketch.

numero naturale compreso tra 0 e il numero totale di unità statistiche

$$0 \leq f(v_i) \leq n$$

La somma delle frequenze assolute delle k modalità $v_1, v_2, ..., v_k$ deve restituire il numero totale di unità statistiche ossia:

$$\sum_{i=1}^{k} f(v_i) = n$$

Nel nostro caso, la frequenza assoluta veniva calcolata in base al numero di occorrenze di una parola all'interno del corpus esplorato. Una volta ritrovate le parole polisemiche e ottenuta la frequenza assoluta, queste sono state raccolte per poter eseguire un confronto, per ogni parola, tra: numero di sensi presenti in WordNet, numero di sensi presenti in Wiktionary, frequenza assoluta e frequenza relativa.

3.3.2 Frequenza relativa

Siano n il numero totale di unità statistiche e X una variabile statistica che può assumere k possibili valori o modalità $v_1, v_2, ..., v_k$. La frequenza relativa della modalità v_i, con $i = 1, 2, ..., k$, si indica con $f_r(v_i)$ e si calcola con la formula:

$$f_r(v_i) = \frac{f(v_i)}{n}$$

Dove $f(v_i)$ è la frequenza assoluta della modalità v_i, ossia il numero di volte in cui il valore v_i si presenta nell'indagine. Poichè la frequenza assoluta $f(v_i)$ è un numero naturale compreso tra 0 e n, la frequenza relativa $f_r(v_i)$ è un numero decimale compreso tra 0 e 1

$$0 \leq f_r(v_i) \leq 1$$

Osserviamo inoltre che la somma delle frequenze relative dei k valori $v_1, v_2, ..., v_k$ è uguale a 1, in quanto la somma delle frequenze assolute deve essere uguale al numero di unità statistiche n

$$\sum_{i=1}^{k} f(v_i) = n \Longrightarrow \sum_{i=1}^{k} f_r(v_i) = 1$$

Infine nel nostro caso, la frequenza relativa viene calcolata eseguendo un rapporto tra la frequenza assoluta di una parola e il numero totale di frasi che

compongono il corpus.

3.4 Apprendimento non supervisionato

L'apprendimento non supervisionato è un tipo di algoritmo che apprende schemi da dati senza etichette o label. L'obiettivo è che attraverso il mimetismo, che è un'importante modalità di apprendimento nelle persone, la macchina sia costretta a costruire una rappresentazione concisa del suo mondo e quindi a generare contenuti immaginari o predittivi da essa.

Contrariamente all'apprendimento supervisionato in cui i dati vengono etichettati da un esperto, i metodi non supervisionati mostrano un'auto-organizzazione che cattura schemi come densità di probabilità o una combinazione di preferenze di caratteristiche neurali codificate nei pesi e nelle attivazioni della macchina.

L'induzione sensoriale viene generalmente trattata come un problema di clustering non supervisionato. L'input per l'algoritmo di clustering sono istanze della parola ambigua con i relativi contesti di accompagnamento (rappresentati da vettori di co-occorrenza) e l'output è un raggruppamento di queste istanze in classi corrispondenti ai sensi indotti. In altre parole, i contesti raggruppati nella stessa classe rappresentano uno specifico senso della parola.

Nel nostro caso invece ci affideremo all'uso di metodi di apprendimento non supervisionato che utilizzeranno principalmente delle semplici reti neurali, Transformers, inserite in un contesto di NLP.

3.4.1 Transformers

Un Transformer[29] è un modello di deep learning[9] che adotta il meccanismo dell'auto-attenzione, ponderando in modo differenziale il significato di ciascuna parte dei dati di input. Viene utilizzato principalmente nel campo dell'elaborazione del linguaggio naturale (NLP).

Così come le reti neurali ricorrenti (RNN)[10], i transformers sono progettati per elaborare dei dati di input sequenziali, come nel caso del linguaggio naturale. Tuttavia, a differenza degli RNN, i trasformatori elaborano l'intero input tutto in una volta.

[9] Tipologia di metodi di apprendimento automatico supervisionato e non, basato su reti neurali artificiali con apprendimento della rappresentazione.

[10] Classe di reti neurali artificiali in cui le connessioni tra i nodi possono creare un ciclo, consentendo all'output di alcuni nodi di influenzare il successivo input agli stessi nodi.

Quando vengono aggiunti agli RNN i meccanismi di attenzione[11] aumentano le prestazioni del modello. Lo sviluppo dell'architettura Transformer ha rivelato che i meccanismi di attenzione erano di per sé potenti e che l'elaborazione sequenziale ricorrente dei dati non era necessaria per ottenere i guadagni di qualità degli RNN con applicazione di attenzione.

I trasformatori utilizzano un meccanismo di attenzione senza un RNN, elaborando tutti i token contemporaneamente e calcolando i pesi di attenzione tra di loro in strati successivi. L'auto-attenzione consente di guardare altre posizioni nella sequenza di input alla ricerca di indizi che possano aiutare a portare ad una migliore codifica per una parola. Poiché il meccanismo di attenzione utilizza solo informazioni su altri token provenienti da livelli inferiori, può essere calcolato per tutti i token in parallelo, il che porta ad una migliore velocità di addestramento.

Architettura Encoder-Decoder

Il modello Transformer utilizza un'architettura encoder-decoder (codificatore-decodificatore).

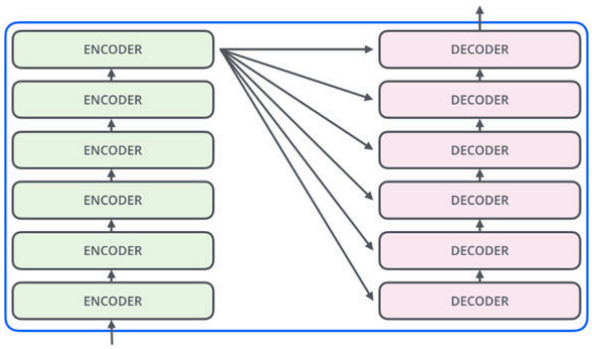


FIGURA 3.1: Struttura a strati Encoder-Decoder.

Il codificatore è composto da una pila di $N = 6$ strati identici. Ogni strato ha due sotto-strati. Il primo è un meccanismo di auto-attenzione a più teste[12] e il secondo è una semplice rete feed-forward[13] completamente connessa in base alla posizione.

Gli input del codificatore fluiscono prima attraverso uno strato di auto-attenzione, che aiuta il codificatore a osservare altre parole nella frase di input

[11] Una tecnica con lo scopo di imitare l'attenzione cognitiva migliorando alcune parti dei dati di input e diminuendone altre.

[12] Meccanismo che consente al modello di prestare attenzione congiuntamente alle informazioni provenienti da diverse rappresentazioni di sottospazi in diverse posizioni.

[13] Una rete neurale artificiale in cui le connessioni tra i nodi non formano un ciclo, diversamente dalla sua discendente RNN.

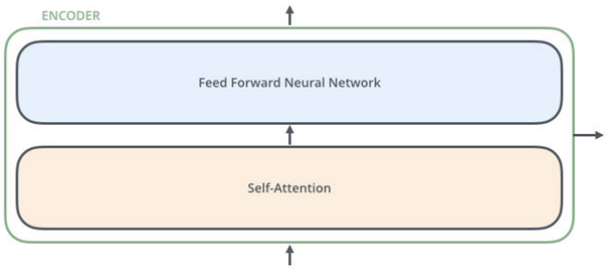


FIGURA 3.2: Struttura degli strati dell' Encoder.

mentre codifica una parola specifica. Gli output del livello di auto-attenzione vengono inviati ad una rete neurale feed-forward. La stessa identica rete feed-forward viene applicata indipendentemente a ciascuna posizione.

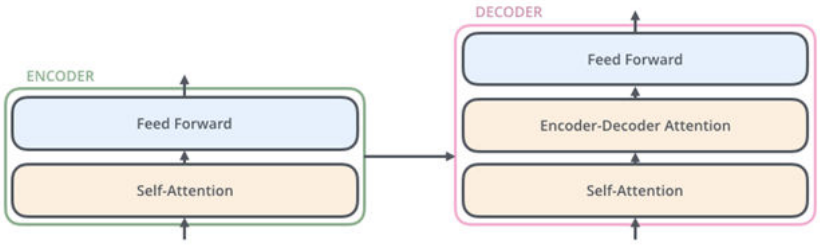


FIGURA 3.3: Struttura degli strati del Decoder.

Il decodificatore possiede entrambi questi livelli, prendendo come input gli output prodotti dal codificatore, ma tra di loro c'è un ulteriore meccanismo di attenzione che invece prende informazioni rilevanti dalle codifiche generate dai codificatori e aiuta il decodificatore a concentrarsi su parti significative della frase di input. La funzione di ciascun livello del codificatore è quella di generare codifiche che contengono informazioni su quali parti degli input sono rilevanti l'una per l'altra. Quindi passa le sue codifiche al successivo livello di codifica come input. Ogni livello di decodifica fa l'opposto, prendendo tutte le codifiche e usando le loro informazioni contestuali incorporate per generare una sequenza di output.

3.4.2 BERT

I trasformatori sono stati introdotti nel 2017 da un team di *Google Brain*[14] e sono sempre di più il modello preferito per il risolvimento di problemi NLP, sostituendo i modelli RNN. La parallelizzazione aggiuntiva dell'addestramento

[14]Team di ricerca sull'intelligenza artificiale di deep learning al di sotto di Google AI, una divisione di ricerca di Google dedicata all'intelligenza artificiale.

lo consente su set di dati più grandi. Ciò ha portato allo sviluppo di sistemi pre-addestrati come **BERT** (Bidirectional Encoder Representations from Transformers) e **GPT** (Generative Pre-trained Transformer), che sono stati addestrati con set di dati linguistici di grandi dimensioni e possono essere ottimizzati per attività specifiche.

Bidirectional Encoder Representations from Transformers (BERT)[20] è una tecnica di machine learning basata su Transformer con pre-addestramento sull'elaborazione del linguaggio naturale (NLP) sviluppata da Google. Rispetto a molti altri modelli, così come dice il nome, BERT è bidirezionale, ovvero considera i contesti su entrambe le parti della parola target durante la rappresentazione. È stato creato e pubblicato nel 2018 da Jacob Devlin e dai suoi colleghi di Google.

BERT è pre-addestrato su due task, Masked Language Model (modello linguistico mascherato) e Next Sentence Prediction (predizione della frase successiva).

In questo progetto è stato fatto uso della prima tipologia di task.

Masked Language Model

MLM consiste nel fornire a BERT una frase e nell'ottimizzare i pesi all'interno di essa per emettere la stessa frase dall'altra parte, mascherando una percentuale del *15%* di tutti i token di input, con il token *[MASK]*, in ogni sequenza in modo casuale e quindi far predire quelli mascherati. Ciò consente al modello di concentrarsi su entrambi i contesti destro (token sul lato destro della maschera) e sinistro (token a sinistra della maschera).

3.4.3 Estrazione dei contesti e Masking

Proprio per questo, ci si è occupati inizialmente dell'estrazione dei contesti in base a delle parole target preselezionate. Per ogni parola, si esplorava un corpus ridotto e modificato di frasi scelte e si ricercavano le frasi all'interno del quale la parola target comparisse. Una volta individuata la frase, la si scomponeva in contesto destro, contesto sinistro e occorrenza della parola target. In seguito, una volta raccolte tutte le possibili frasi scomposte, se ne sceglievano *100* in maniera casuale e non in ordine di comparizione.

A questo punto si è proceduto con la fase di masking: la maschera veniva applicata al posto dell'occorrenza della parola target e venivano fatte prevedere le top *20* possibili parole da poter sostituire, salvandole assieme agli score di

"somiglianza" in dei file di testo. Questi, con anche dei file di vettori *cls* e *Token* (4.5.3) saranno utilizzati nella fase di sperimentazione (5.2).

XLM-RoBERTa

Quest'ultimo task è stato applicato anche nei confronti della lingua inglese, precisamente su un insieme di coppie di parole target Ita-Eng, con l'utilizzo del modello **"xlm-roberta-base"**.

XLM-RoBERTa è una versione multilingue di **RoBERTa**. È preaddestrato su *2,5 TB* di dati CommonCrawl filtrati contenenti *100* lingue.

RoBERTa (Robustly Optimized BERT pre-training Approach) è un modello di trasformatori preaddestrato su un grande corpus in modo auto-supervisionato. Ciò significa che è stato preaddestrato solo sui testi grezzi, senza che gli esseri umani li etichettassero in alcun modo con un processo automatico per generare input ed etichette da quei testi. Rispetto RoBERTa, XLM-RoBERTa è architetturalmente identico ma nella fase di addestramento gli sono stati forniti molti più dati e in più risulta adatto per task multilingua.

Entrambi si presentano come varianti di BERT, ma principalmente RoBERTa è vista come una versione migliore di esso: in BERT erano state ritrovate delle criticità legate ai parametri del modello stesso mentre RoBERTa invece si presentava più robusta ed efficiente. Ad esempio, ritroviamo in RoBERTa un masking dinamico, a confronto di quello statico di BERT oppure la rimozione del task di NSP (Next Sentence Prediction) poiché considerato inutile per il pre-addestramento.

Capitolo 4

Progettazione

Per l'implementazione degli algoritmi e delle strutture dati utilizzati all'interno del progetto è stato utilizzato l'editor di codice sorgente VSCode e il linguaggio di programmazione Python, il tutto è stato gestito con gli altri componenti del team tramite il servizio di hosting GitHub, per la condivisione e revisione del codice. Sono evidenziate e documentate di seguito le loro principali caratteristiche e funzionalità sviluppate.

4.1 Strumenti utilizzati

Esploriamo brevemente i tre strumenti principali che sono stati utilizzati per l'implementazione e condivisione degli algoritmi, ovvero **Vscode**, **Python** e **Github**.

4.1.1 Visual Studio Code

FIGURA 4.1: Logo Visual Studio Code

Visual Studio Code[1] combina la semplicità di un editor di codice con ciò di cui gli sviluppatori hanno bisogno per la creazione e il debug di moderne applicazioni web e cloud. Fornisce un supporto completo per la modifica, la navigazione e la comprensione del codice insieme a un debugging leggero, un

[1] `https://code.visualstudio.com/`

ricco modello di estensibilità e un'integrazione leggera con gli strumenti esistenti. Viene inoltre aggiornato mensilmente con nuove funzionalità e correzioni di bug.

4.1.2 Python

FIGURA 4.2: Logo Python

Python[2] è un linguaggio di programmazione di *alto livello*[3], orientato a oggetti, adatto, tra gli altri usi, a sviluppare applicazioni distribuite, scripting[4], computazione numerica e system testing. Ideato da Guido van Rossum all'inizio degli anni Novanta, è spesso paragonato a Perl, JavaScript, o Visual Basic.

L'intero progetto è stato sviluppato in Python.

4.1.3 Github

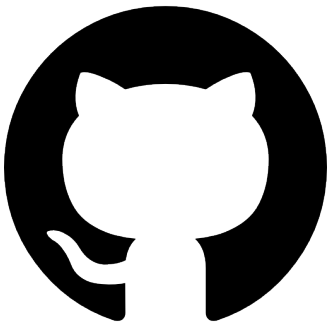


FIGURA 4.3: Logo Github

GitHub[5] si presenta come un servizio di hosting per progetti software, di proprietà della società GitHub Inc., con sede legale a San Francisco in California.

[2] `https://www.python.org/`

[3] Ci si riferisce al livello di astrazione che definisce quanto sia di "alto livello" un linguaggio di programmazione.

[4] Linguaggio di programmazione interpretato destinato in genere a compiti di automazione del sistema operativo o delle applicazioni o a essere usato nella programmazione web.

[5] `https://github.com/`

Il nome deriva dal fatto che "GitHub" sia una implementazione dello strumento di controllo di versione distribuito **Git**.

Il sito viene principalmente utilizzato dagli sviluppatori, che caricano il codice sorgente dei loro programmi e lo rendono scaricabile dagli utenti. Questi ultimi possono interagire con lo sviluppatore tramite un sistema di ricerca di errori, pull request e commenti che permette di migliorare il codice del repository risolvendo bug o aggiungendo funzionalità.

Il progetto è stato creato utilizzando questa piattaforma in collaborazione con gli altri componenti del team, caricando e revisionando possibili errori o modifiche da apportare.

4.2 Estrazione parole polisemiche da WordNet

Per quanto riguarda l'estrazione di parole polisemiche dalla risorsa online WordNet, ci si è affidati all'utilizzo di **NLTK**.

4.2.1 NLTK

NLTK[6] (Natural Language ToolKit), è una suite di librerie e programmi per l'analisi simbolica e statistica nel campo dell'elaborazione del linguaggio naturale principalmente in lingua inglese e scritta in linguaggio Python.

Dopo aver installato il package di NLTK, si sono cercati e installati anche i dataset e modelli necessari per proseguire, ovvero nel nostro caso WordNet.

```
from nltk.corpus import wordnet as wn
```

FIGURA 4.4: Import del corpus dalla libreria nltk

Così si è costruita una funzione che potesse interrogare il dataset installato e ricercare parole polisemiche italiane in base alla classe semantica.

```
def searchingPolysemyWordnet(pos):

    cont = 0

    wordnet_dict = {}
    lemma_list = []
    synset_list = list(wn.all_synsets(pos))
```

FIGURA 4.5: Funzione searchingPolysemyWordNet

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

Come possiamo vedere dalla figura 4.5, inizialmente vengono raccolte tutte le parole di una precisa classe semantica che verrà assegnata al momento in base a delle parole chiave illustrate nella sezione (3.1.1).

In seguito si pone un filtro alle parole già estratte, selezionando solo i synset appartenenti alla lingua italiana (lang = *"ita"*). Da qui si procede con l'applicazione dell'algoritmo per il controllo della polisemia: si contano i synset associati ad ogni lemma, se il numero ottenuto è maggiore di *1*, si mantiene la parola trovata, altrimenti si continua la ricerca.

```
for synset in synset_list:
    lemma_list.extend(synset.lemma_names(lang="ita"))
    for i in range(cont, len(lemma_list)):
        lemma = lemma_list[cont]
        if(len(wn.synsets(lemma, lang="ita")) > 1):
            sense_number = len(wn.synsets(lemma, lang="ita"))
            lemma = lemma.replace("_", " ").lower()
            wordnet_dict[lemma] = sense_number
            cont += 1
        else:
            lemma_list.remove(lemma)

if 'gap!' in wordnet_dict:
    del wordnet_dict['gap!']
if 'pseudogap!' in wordnet_dict:
    del wordnet_dict['pseudogap!']

wordnet_dict = dict(sorted(wordnet_dict.items(), key=lambda x: x[1], reverse=True))
```

FIGURA 4.6: Algoritmo di ricerca di parole polisemiche per WordNet

Infine, le parole polisemiche ritrovate vengono raccolte all'interno di un dizionario dove la chiave sarà la parola stessa e il valore della chiave il numero di sensi che la parola contiene. Questo dizionario verrà ordinato in maniera decrescente e sarà privo di duplicati, proprio perché avendo scelto di mettere la parola come chiave, non si potranno avere ripetizioni di chiavi nello stesso dizionario.

4.3 Estrazione parole polisemiche da Wiktionary

Nei confronti invece della risorsa online Wiktionary, come detto nella sezione (3.2.1), si sono utilizzati 4 file JSON, uno per ogni classe semantica.


```
{"pos": "verb", "head_templates": [{"name": "head", "args": {"1": "it", "2": "past participle form", "g": "f-p"}, "expansion": "aberrate f pl"}], "etymology_text": "", "etymology_templates": [], "word": "aberrate", "lang": "Italian", "lang_code": "it", "categories": [], "senses": [{"raw_glosses": ["feminine plural of aberrato"], "tags": ["feminine", "form-of", "participle", "plural"], "glosses": ["feminine plural of aberrato"], "form_of": [{"word": "aberrato"}], "id": "aberrate-it-verb-oHy7g6C9"}]}
```


FIGURA 4.7: Esempio oggetto JSON

Come possiamo vedere dalla figura 4.7, per ogni riga di un file JSON vi è un oggetto json che rappresenta una parola, con i campi associati ad essa.

Il campo *word*, ci fornisce la parola, mentre nel caso della classe semantica dei verbi, sarà il campo *form_of* a fornirci il lemma, così da non imbattersi nel problema di dover scegliere quale flessione del verbo selezionare. Mentre invece i sensi associati ad una parola li ritroveremo nel campo *senses*, che appare sotto forma di un array json. Al suo interno troviamo un sottocampo grazie al quale poter effettuare il conteggio dei sensi, ovvero *raw_glosses*.

```
def searchingPolysemyWiktionary(file):

    i = 0

    wiktionary_dict = {}
    sense_word_list = []

    synset_list = [json.loads(line) for line in open(file,'r')]
```

FIGURA 4.8: Funzione searchingPolysemyWiktionary

Pertanto, è stata costruita una funzione che potesse estrarre i dati da ogni file json, caricandoli inizialmente, riga per riga, in una lista di supporto.

```
for synset in synset_list:
    limite = len(synset['senses'])
    try:
        while i < limite:
            try:
                lemma = synset['senses'][i]['form_of'][0]['word']
                lemma = lemma.lower()
                sense_word_list.append(synset['senses'][i]['raw_glosses'])
                i += 1
                sense_number = len(sense_word_list)
                if(sense_number > 1):
                    wiktionary_dict[lemma] = sense_number
            except KeyError: #viene gestita l'eccezione nel caso in cui la chiave form_of
                lemma = synset['word']
                lemma = lemma.lower()
                sense_word_list.append(synset['senses'][i]['raw_glosses'])
                i += 1
                sense_number = len(sense_word_list)
                if(sense_number > 1):
                    wiktionary_dict[lemma] = sense_number
    except KeyError: #viene gestita l'eccezione nel caso in cui la chiave raw_glosses non
        i = 0
        sense_word_list.clear()
    i = 0
    sense_word_list.clear()

wiktionary_dict = dict(sorted(wiktionary_dict.items(), key=lambda x: x[1], reverse=True))
return wiktionary_dict
```

FIGURA 4.9: Algoritmo di ricerca di parole polisemiche per Wiktionary

Successivamente viene applicato l'algoritmo di ricerca di parole polisemiche: a differenza dell'approccio utilizzato per WordNet in questo caso il conteggio verrà fatto su quanti *raw_glosses* una parola può contenere all'interno del campo *senses*, ma applicando la condizione sul numero di essi, ovvero che se sarà

almeno maggiore di *1*, allora la parola potrà essere considerata polisemica. Infine, le parole ritrovate verranno salvate in un dizionario ordinato in maniera decrescente e privo di duplicati.

4.4 Confronto e calcolo delle frequenze

Come detto nella sezione (3.3), con l'utilizzo del corpus italiano, si procede verso una fase di confronto e di calcolo della frequenza assoluta e relativa.

FIGURA 4.10: Calcolo delle frequenze e costruzione del dizionario

Nella funzione presente nella figura 4.10, possiamo vedere come vengano esplorati 3 dizionari: il dizionario delle parole ritrovate nel corpus, il dizionario delle parole ritrovate su WordNet e il dizionario delle parole estratte da Wiktionary. Queste 3 strutture verranno utilizzate per creare un dizionario finale che possa contenere 5 campi messi a confronto: parola, numero di sensi di WordNet, numero di sensi di Wiktionary, frequenza assoluta e frequenza relativa. La frequenza assoluta viene estratta direttamente dal dizionario passato via parametro, mentre la frequenza relativa viene calcolata ogni volta per ciascuna parola in base alla frequenza assoluta. Nel caso in cui una parola non fosse presente all'interno dei dizionari di Wordnet o Wiktionary, come numero di sensi verrà posto un *"-1"* ad indicare la mancata presenza.

4.5 Apprendimento non supervisionato

Per quanto riguarda la fase di apprendimento non supervisionato, come detto nella sezione (3.4), è stato utilizzato il Transformer **BERT**, grazie all'utilizzo dell'omonima libreria *Transformers* e si è implementato il *deep learning* grazie all'utilizzo della libreria *Torch* che si basa su **Pytorch**.

```
from transformers import AutoTokenizer
from transformers import AutoModel
from transformers import pipeline
import torch
```

FIGURA 4.11: Import delle librerie transformers e torch.

4.5.1 Pytorch

PyTorch è un framework di apprendimento automatico basato sulla libreria *Torch*, utilizzato per applicazioni come la visione artificiale e l'elaborazione del linguaggio naturale. Un certo numero di software di deep learning è costruito su PyTorch tra cui proprio i Transformers di *Hugging Face*[7]. PyTorch offre due funzionalità di alto livello:

- Tensor computing (come *NumPy*) con forte accelerazione tramite unità di elaborazione grafica (GPU);
- Reti neurali profonde costruite su un sistema di differenziazione automatica basato su nastro.

PyTorch inoltre definisce una classe chiamata *Tensor* (torch.Tensor) per memorizzare e operare su matrici rettangolari multidimensionali omogenee di numeri. I tensori PyTorch sono simili agli array NumPy, ma possono essere utilizzati anche su una GPU NVIDIA compatibile con CUDA.

4.5.2 Estrazione dei contesti

Inizialmente ci si è occupati dell'estrazione delle parole target da file forniti contenenti 1000 parole target preselezionate delle 4 classi semantiche (verbi, avverbi, aggettivi, nomi).

In base alla classe semantica che poteva apparire nei file sotto forma di *'VERB'*, per i verbi, *'NOUN'*, per i nomi, *'ADJ'*, per gli aggettivi e *'ADV'*, per gli avverbi, le parole target venivano estratte e salvate all'interno di una lista chiamata *targets*.

Successivamente, una volta ottenute le parole target, venivano utilizzati altri file, contenenti delle frasi estratte dal corpus ITWAC. L'obiettivo era di ritrovare quelle frasi dove le parole target comparissero all'interno di esse e una volta individuate, procedere con l'estrazione dei contesti:

[7]Azienda americana che sviluppa strumenti per la creazione di applicazioni utilizzando l'apprendimento automatico.

```
def target_extraction(filename, semantic_class):

    targets = []

    with open(filename, 'r', encoding = "utf-8") as f:
      f.readline()
      for line in f:
          line = line.strip().split("\t")[1].split("_")
          word = line[0]
          sem_class = line[1]
          if sem_class == semantic_class:
              targets.append(word)
    return targets
```

FIGURA 4.12: Funzione che estrae le parole target.

- contesto sinistro, tutto ciò che si trova prima dell'occorrenza della parola target nella frase;
- contesto destro, tutto ciò che si trova dopo l'occorrenza della parola target.

Ovviamente anche l'occorrenza della parola target veniva salvata. Questo perché la frase individuata, veniva scomposta nei contesti, ma mantenuta come frase appartenente alla parola target.

```
#ricavo i componenti necessari
first_word = file_content[0] #parola ad inizio riga
start = int(file_content[1]) #inizio della occorrenza target
end = int(file_content[2]) #fine dell'occorrenza target
file_content = file_content[3] #contesto

#controllo se la parola ad inizio riga è uguale alla parola
if first_word == target_word:

    #estraggo i contesti
    target_occ = file_content[start:end]
    left_context = file_content[:start]
    right_context = file_content[end:]

    sentence = [left_context,target_occ,right_context]#costr
    temp_list.append(sentence)
```

FIGURA 4.13: Estrazione dei contesti per ogni frase.

Come vediamo dalla figura 4.13, i due contesti e l'occorrenza della parola target vengono salvati in una *frase* momentanea. Per il ritrovamento dei contesti si è fatto uso di 2 indici che comparivano per ogni frase, la quale appariva in questo modo:

$$[parola][tab][start\_index][tab][end\_index][frase]$$

Start_index e *end_index* sono gli indici di inizio e fine di occorrenza della parola target. In questo modo è stato possibile estrarre in maniera semplice tutti i dati necessari.

Infine poiché per alcune parole poteva essere ritrovato un numero elevato di frasi, si è posto un limite di 100 per ogni parola, selezionate in maniera casuale invece che per ordine di apparizione. Utilizzando la libreria *random* si sono selezionati 100 numeri casuali, i quali avrebbero rappresentato gli indici da usare per prelevare le frasi salvate per ogni parola.

```
lun = len(temp_list)
number_range=range(0,lun)

if len(temp_list) > 100:
    number_list = random.sample(number_range,100)
    l = 0
    while l < 100:
        sentence = temp_list[number_list[l]]
        sentences[target_word] += [sentence] #aggi
        l += 1
else:
    for el in temp_list:
        sentence = el
        sentences[target_word] += [sentence]
```

FIGURA 4.14: Algoritmo di selezione di 100 frasi in maniera casuale.

Quindi le parole target e le loro frasi scomposte sono state salvate in un dizionario apposito, dove ogni chiave è rappresentata da una parola mentre i valori sono costituiti da vettori di frasi scomposte.

4.5.3 Creazione dei token

Dopo aver ottenuto il dizionario di parole target e frasi, ci si è occupati della fase riguardante l'uso del transformer. Il modello utilizzato è stato BERT, precisamente grazie l'utilizzo della libreria Transformers, si è creato il modello pre-addestrato basandolo su ***'dbmdz/bert-base-italian-xxl-uncased'*** , adatto per parole italiane. Il modello così costruito è stato basato anche su CUDA: una piattaforma di elaborazione parallela e un'interfaccia di programmazione dell'applicazione (API) che consente al software di utilizzare determinati tipi di unità di elaborazione grafica (GPU) per l'elaborazione generica.

```
model_checkpoint = 'dbmdz/bert-base-italian-xxl-uncased'

tokenizer = AutoTokenizer.from_pretrained(model_checkpoint)

model = AutoModel.from_pretrained(model_checkpoint,output_hidden_states=True)

model.to('cuda')

nlp_fill = pipeline('fill-mask', model='dbmdz/bert-base-italian-xxl-uncased')
```

FIGURA 4.15: Costruzione del modello pre-addestrato

Ovviamente prima di poter proceder con l'utilizzo del modello creato, ci si è dovuti occupare della tokenizzazione delle frasi estratte precedentemente.

A tal proposito è stata creata una funzione che si occupa della creazione di word embeddings.

Word Embeddings

I *word embeddings*, ovvero incorporamenti di parole, sono una classe di tecniche in cui le singole parole sono rappresentate come vettori a valori reali in uno spazio vettoriale predefinito. Ogni parola è mappata su un vettore e i valori del vettore vengono appresi in un modo che possa assomigliare ad una rete neurale, quindi è una tecnica spesso utilizzata nel campo del deep learning.

Alla base di questo approccio vi è l'idea di utilizzare una rappresentazione distribuita densa per ogni parola. Ogni parola è rappresentata da un vettore a valori reali.

```
def get_embedding(sentence_l,target,sentence_r,strategy='mean'):
    model.eval()
    with torch.no_grad():
      tokens = []
      tokens_l = tokenizer.tokenize(sentence_l)
      tokens = tokens + tokens_l

      start = len(tokens)

      tokens_target = tokenizer.tokenize(target)
      tokens = tokens + tokens_target

      end = len(tokens)

      tokens_r = tokenizer.tokenize(sentence_r)
      tokens = tokens + tokens_r

      if len(tokens) > 512:
          return None

      input_ids = tokenizer.convert_tokens_to_ids(tokens)

      input_mask = torch.tensor([[1] * len(input_ids)])

      input_ids = torch.tensor([input_ids])

      outputs = model(input_ids.to('cuda'),attention_mask=input_mask.to('cuda'))
      hidden_states = outputs['hidden_states'][-1].cpu().detach().numpy()

      if strategy == 'mean':
        return np.mean(hidden_states[-1][start:end],axis=0)
      else:
        return hidden_states[-1][0]
```

FIGURA 4.16: Funzione get embeddings.

La funzione *get_ embeddings* che vediamo in figura 4.16 , in base al tipo di *"strategia"* passata come parametro, ci restituisce un vettore del primo token della frase, ovvero il token *CLS*, un token speciale utilizzato per indicare l'inizio di una frase; oppure ci restituisce un vettore rappresentante la parola, ottenuto come centroide dei vettori delle sotto-parole in cui la parola è stata tokenizzata.

Perlopiù viene aggiunto un controllo sul limite dei token per ogni scomposizione, se si arriva al di sopra di 512, la frase viene scartata e ignorata. Nella funzione viene fatto uso dei tensori e delle funzioni di gestione di essi forniti dalla libreria torch (torch.tensor e torch.no_grad).

4.5.4 Masking

Infine ci si è occupati della fase di masking e come detto nella sezione (3.4.3) il MLM è uno dei principali task proposti da BERT.

```
for i,target in enumerate(targets):
  vecs_token = []
  vecs_cls = []
  mapp = {}

  with open('File_elaborati/Token_Nomi/Token_Score/%s_TokenScore.txt'%target, 'w+', encoding = 'utf-8') as fp:

    for j, sent in enumerate(sentences[target]):
      left_context, target_occ, right_context = sent

      vec_token = get_embedding(left_context,target_occ,right_context,strategy='mean')
      if vec_token is not None:
        vecs_token.append(vec_token)
        vec_cls = get_embedding(left_context,target_occ,right_context,strategy='cls')
        vecs_cls.append(vec_cls)
        try:
          fills = nlp_fill(left_context + '[MASK]' + right_context, top_k=20)
        except RuntimeError:
          del vec_token[-1]
          del vec_cls[-1]
          continue
        mask_line = ""
        for f in fills:
          token = f['token_str']
          score = f['score']
          if len(mask_line) == 0:
              mask_line = token + ' ' + str(score)
          else:
              mask_line = mask_line + '\t' + token + ' ' + str(score)
        fp.write(f'{mask_line}\n')

        mapp[i] = j

  with open('File_elaborati/Token_Nomi/mapping_dict.txt', 'wb+') as f:
        pickle.dump(mapp,f)

  vecs_token = np.array(vecs_token)
  np.save('File_elaborati/Token_Nomi/Vecs_Token/vecs_token_%s.npy'%target,vecs_token)

  vecs_cls = np.array(vecs_cls)
  np.save('File_elaborati/Token_Nomi/Vecs_cls/vecs_cls_%s.npy'%target,vecs_cls)
```

FIGURA 4.17: Funzione che si occupa del masking e del salvataggio dei vettori in file.

Nel nostro caso abbiamo effettuato masking su ogni frase di ogni parola target, dove la maschera veniva applicata al posto dell'occorrenza della parola target e di conseguenza venivano salvate in un file chiamato *Token_Score* tutte le parole proposte dal modello con il loro score di "vicinanza" alla parola originale. Nel caso in cui qualche frase fosse stata scartata, per quanto riguarda i limiti imposti dalla funzione di tokenizzazione, si è provveduto con la creazione

di un dizionario che potesse mappare le posizioni di ogni frase estratta, così da poter tenere conto di quelle ignorate.

Successivamente i vettori di token prodotti nella fase precedente, sono stati salvati trasformati in vettori *Numpy* e salvati in dei file *Vecs_ cls* e *Vecs_ Token* rispettivamente, i quali, assieme ai file *Token_ Score*, verranno utilizzati nella sezione riguardante la sperimentazione.

XLM-RoBERTa

Per quanto riguarda invece la gestione delle parole inglesi, il task si è presentato identico tranne per alcune modifiche. Ad esempio per l'utilizzo del modello, invece di una versione di BERT, è stato utilizzato ***"xlm-roberta-base"*** facente parte della famiglia dei modelli RoBERTa e adatto per gestione di multilingue, quindi ottimo nel nostro caso per una gestione di coppie di parole italiane-inglesi.

Infatti i file da cui sono state estratte le parole inglesi, presentavano la loro corrispondente traduzione in italiano. Questo però ha portato all'esclusione della fase di masking: essendo coppie di parole italiano-inglese, nel caso in cui si fosse applicato, i valori prodotti sarebbero rimasti inutilizzati, a differenza della controparte italiana, poiché sarebbe stato impossibile effettuare un confronto tra le coppie di parole in due lingue differenti, con i mezzi a disposizione.

Capitolo 5

Sperimentazione

In questo capitolo valuteremo l'efficacia dei valori ottenuti nella sezione precedente, esplorando le tecniche di semantica distribuzionale, per poter comprendere la relazione che ci possa essere tra gli score di similarità dei dati elaborati e il numero di sensi legato alle parole da cui sono stati estratti.

5.1 Semantica Distribuzionale

La semantica distribuzionale [33] comprende una serie di teorie e metodi di linguistica computazionale per lo studio della distribuzione semantica delle parole nel linguaggio naturale. Questi modelli derivano da una prospettiva empiristica e assumono che una distribuzione statistica dei termini sia dominante nel delinearne il comportamento semantico. Questa teoria propone il paradigma per cui le parole sono distribuite in uno spazio nel quale sono, tra di loro, ad una distanza proporzionale al loro grado di similarità. Quest'ultima segue l'ipotesi fondamentale della semantica distribuzionale (chiamata ipotesi distribuzionale) secondo la quale due parole sono tanto più simili o vicine semanticamente, quanto più sono inclini nel comparire nello stesso contesto linguistico.

5.1.1 Realizzazioni

Per le realizzazioni concrete dell'ipotesi distribuzionale si procede costruendo degli spazi semantici distribuzionali utilizzando rappresentazioni geometriche per rappresentare la semantica del testo. Ogni punto nello spazio è caratterizzato da n dimensioni, cioè dalle coordinate rispetto agli n assi che formano lo spazio vettoriale in osservazione. In questo modo ogni parola diventa un vettore, le cui dimensioni dipendono dai contesti linguistici in cui la parola può ricorrere e la distanza tra i punti è proporzionale alla distanza semantica tra le parole (sempre in base all'ipotesi distribuzionale). Formalmente lo spazio semantico viene definito tramite quattro variabili fondamentali:

- **T**, rappresenta l'insieme delle parole target che vanno a formare lo spazio semantico;
- **B**, la base che definisce le dimensioni dello spazio osservato e contiene i contesti linguistici sui quali viene valutata la similarità;
- **M**, la matrice di co-occorrenza che rappresenta i vettori di T;
- **S**, la metrica che misura la distanza dei punti nello spazio semantico;

Si può riassumere quindi che ogni parola target T corrisponde ad una riga della matrice M e ogni contesto B definisce le colonne della matrice stessa. Le celle contengono, nella situazione più semplice, la frequenza di co-occorrenza della parola T in un contesto B. Le differenze tra i modelli dipendono dal metodo con cui caratterizzano B, cioè da come definiscono il contesto. Comunemente viene fatto in base ad una finestra **W** di parole che si trovano attorno alla parola target T. In questo caso B è un sottoinsieme delle parole-tipo ottenuto escludendo le stopwords (che non hanno rilevanza dal punto di vista semantico) e includendo le parole contenute più frequenti nel vocabolario del testo.

5.1.2 Statistiche

Se bisogna calcolare la distanza semantica tra due parole, è necessario quindi calcolare la distanza tra i due vettori su tutte le dimensioni. Più le dimensioni sono simili tra i due vettori, più i significati delle parole che formano i vettori sono simili (sempre secondo l'ipotesi distribuzionale). La misura più comune della vicinanza spaziale è il coseno dell'angolo formato dagli stessi (se i vettori hanno dimensioni uguali, l'angolo è *0* e il coseno ha come massimo *1*; se i vettori sono indipendenti, l'angolo è di *90°* e il coseno ha come minimo *0*). Un altro metodo è dato dal calcolo della *Distanza euclidea* generalizzando ad uno spazio multidimensionale.

In seguito alla verifica dei risultati e della loro correttezza emerge che l'ipotesi distribuzionale abbia frequenti riscontri con l'interpretazione semantica fornita dalle persone, in particolare gli spazi semantici distribuzionali possono essere usati come modelli per vari compiti legati alla distanza semantica tra le parole, più accuratamente rispetto a modelli lessicali basati su simboli con reti lessicali (come **Wordnet**)[26].

5.2 Valutazione risultati

Come abbiamo visto nella sezione (3.4), i vettori *cls* e *token* sono stati elaborati e raccolti. A questo punto, si applicheranno delle tecniche di valutazione ma prima bisognerà applicare la metrica scelta per calcolare la distanza semantica tra le parole target, ovvero nel nostro caso il coseno di similarità. Infine verrà applicato il coefficiente di Spearman sugli score di similarità raccolti, sui numeri di significati delle parole per WordNet e Wiktionary, sui numeri di frequenze assolute delle parole e sulle medie calcolate dall'intersezione di liste di "maschere" delle parole target. Oltre agli score prodotti dalla correlazione di Spearman, verranno forniti anche i valori p (p-value) per ogni punteggio.

5.2.1 Similarità del coseno

Nell'analisi dei dati, la similarità del coseno è una misura della somiglianza tra due sequenze di numeri. Per definire meglio, le sequenze sono viste come dei vettori in uno spazio prodotto interno[1] e la similarità del coseno è definita come il coseno dell'angolo tra di loro, cioè il prodotto scalare dei vettori diviso per il prodotto delle loro lunghezze.

Ne segue che la similarità del coseno non dipende dalle grandezze dei vettori, ma solo dal loro angolo. Appartiene sempre all'intervallo **[-1,1]**. Inoltre un vantaggio è sicuramente la sua bassa complessità, specialmente per i vettori sparsi: devono essere considerate solo le coordinate diverse da zero.

Definizione

Il coseno di due vettori diversi da zero può essere derivato utilizzando la formula del prodotto scalare euclideo:

$$A \cdot B = \| A \| \| B \|$$

Dati due vettori di attributi, **A** e **B**, la somiglianza del coseno, $\cos\theta$, è rappresentata utilizzando un prodotto scalare e una grandezza come

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

[1]Spazio vettoriale reale o uno spazio vettoriale complesso con un'operazione chiamata prodotto interno.

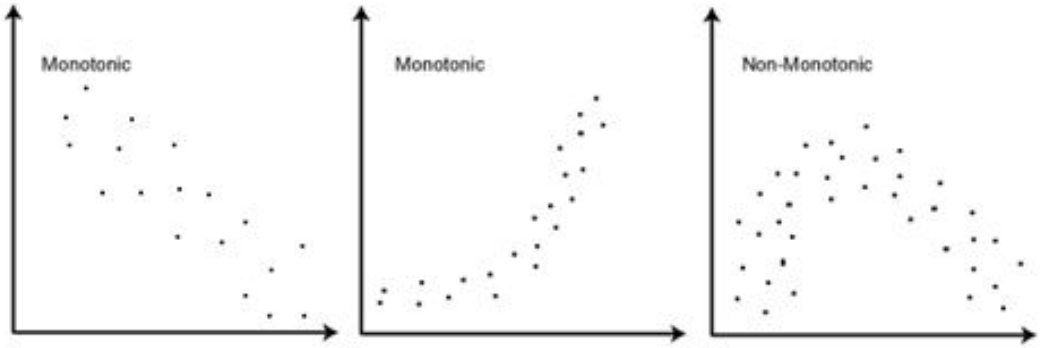
dove A_i e B_i sono i componenti del vettore **A** e **B** rispettivamente.

La somiglianza risultante varia da *-1* che significa esattamente opposto, a *1* che significa esattamente lo stesso, con *0* che indica ortogonalità o decorrelazione, mentre i valori intermedi indicano somiglianza o dissomiglianza intermedia.

5.2.2 Coefficiente di correlazione di Spearman

In statistica, il coefficiente di correlazione di Spearman o ρ di Spearman, dal nome di Charles Spearman, è una misura non parametrica della correlazione di rango[2]. Misura la forza e la direzione dell'associazione tra due variabili classificate e può essere descritta utilizzando una funzione monotona.

Una funzione monotona è una funzione che esegue una delle seguenti operazioni: all'aumentare del valore di una variabile, aumenta anche il valore dell'altra variabile; oppure all'aumentare del valore di una variabile, il valore dell'altra variabile diminuisce.



La correlazione di Spearman tra due variabili è uguale alla correlazione di Pearson tra i valori di rango di queste due variabili; mentre la correlazione di Pearson valuta le relazioni lineari, la correlazione di Spearman valuta le relazioni monotone (lineari o meno). Se non ci sono valori di dati ripetuti, una perfetta correlazione di Spearman di *+1* o *-1* si verifica quando ciascuna delle variabili è una perfetta funzione monotona dell'altra. Il coefficiente di Spearman è appropriato per variabili ordinali sia continue che discrete.

Definizione e Interpretazione

Il coefficiente si mostra come un caso particolare del coefficiente di correlazione di Pearson dove i valori vengono convertiti in ranghi prima di calcolare il

[2]Dipendenza statistica tra le classifiche di due variabili.

coefficiente,

$$\rho_s = \frac{\sum_i (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\sum_i (r_i - \bar{r})^2} \sqrt{\sum_i (s_i - \bar{s})^2}}.$$

ma solitamente si esegue un calcolo più semplice, in quanto si calcola la differenza D tra i ranghi delle due misure di un'osservazione, ottenendo così

$$\rho_s = 1 - \frac{6 \sum_i D_i^2}{N(N^2 - 1)}$$

dove $D_i = r_i - s_i$ è la differenza dei ranghi (essendo r_i e s_i rispettivamente il rango della prima variabile e della seconda variabile della i-esima osservazione) e N il numero complessivo di osservazioni.

Il segno della correlazione di Spearman indica la direzione di associazione tra X (la variabile indipendente) e Y (la variabile dipendente). Se Y tende ad aumentare quando X aumenta, il coefficiente di correlazione di Spearman è positivo. Se Y tende a diminuire quando X aumenta, il coefficiente di correlazione di Spearman è negativo. Una correlazione di Spearman pari a zero indica che non vi è alcuna tendenza per Y ad aumentare o diminuire quando X aumenta. La correlazione di Spearman aumenta di grandezza man mano che X e Y si avvicinano ad essere funzioni perfettamente monotone l'una dell'altra. Quando X e Y sono perfettamente correlati monotonamente, il coefficiente di correlazione di Spearman diventa 1.

Il coefficiente di correlazione di Spearman è spesso descritto come "non legato ai parametri". Questo può avere due significati. In primo luogo, si ottiene una perfetta correlazione di Spearman quando X e Y sono correlati da qualsiasi funzione monotona. Confrontando con la correlazione di Pearson, che fornisce un valore perfetto solo quando X e Y sono correlate da una funzione lineare. L'altro senso in cui la correlazione di Spearman è non parametrica è che la sua esatta distribuzione campionaria può essere ottenuta senza richiedere la conoscenza (cioè conoscere i parametri) della distribuzione di probabilità unita di X e Y.

P-value

Il valore p, o valore di probabilità, è un numero che descrive quanto è probabile che i propri dati si siano verificati per caso, ovvero che l'ipotesi nulla[3] sia vera.

[3]Affermazione probabilistica secondo la quale non ci sia differenza oppure non vi sia relazione tra due fenomeni misurati.

Il livello di significatività statistica è spesso espresso come un valore p compreso tra 0 e 1. Più piccolo è il valore p, più forte è l'evidenza che si dovrebbe rifiutare l'ipotesi nulla. Un valore p inferiore a 0,05 (tipicamente $\leq$ 0,05) è statisticamente significativo. Indica una forte evidenza contro l'ipotesi nulla, poiché c'è meno del 5% di probabilità che il valore nullo sia corretto e che quindi i risultati siano dati dalla casualità. Pertanto, si può rifiutare l'ipotesi nulla e accettare l'ipotesi alternativa proposta. Tuttavia, se il valore p è inferiore alla soglia di significatività (in genere p $<$ 0,05), è possibile comunque rifiutare l'ipotesi nulla. Rimane però che il valore p sia subordinato al fatto che l'ipotesi nulla sia vera, ma non è correlato alla verità o alla falsità dell'ipotesi alternativa.

Un valore p superiore a 0,05 ($>$ 0,05) non è statisticamente significativo e indica una forte evidenza per l'ipotesi nulla. Ciò significa che si mantiene l'ipotesi nulla e che si rifiuta l'ipotesi alternativa. Infine un risultato statisticamente significativo non può dimostrare che un'ipotesi di ricerca è corretta, ma perlomeno si può affermare che i risultati ottenuti "forniscano supporto per" o "forniscano le prove per" le ipotesi di ricerca.

5.2.3 Risultati

Poiché la correlazione di Spearman si basa sull'uso di una funzione monotona, è stata scelta come metrica per poter osservare la relazione che ci possa essere principalmente tra la similarità del coseno e il numero di sensi.

Ovvero, prendendo in considerazione una parola target, nel caso in cui ci si ritroverà con un punteggio di similarità del coseno basso, il numero di sensi di questa parola sarà alto, poiché significherebbe che la parola, all'interno delle frasi in cui compariva, abbia assunto dei significati diversi mostrando così un indice di polisemia. Al contrario invece, più la parola target mantiene un significato simile tra le frasi, più la similarità del coseno sarà alta; quindi il numero di sensi con cui quella parola appare, sarà più basso.

Pertanto, sapendo che la correlazione di Spearman può presentare dei valori estremi di -1, 1 e 0, il risultato convergerà verso -1 quando ci sarà una correlazione inversa, al crescere di una variabile diminuirà l'altra, proprio come nel caso della relazione tra il coseno e il numero di sensi. Ma allo stesso modo il risultato potrà convergere verso l'1, nel caso in cui ci sia una correlazione molto alta e quindi le due variabili aumentino contemporaneamente. Infine il risultato si avvicinerà di più verso lo 0 quando non ci sarà alcuna tendenza di una variabile di aumentare o diminuire quando l'altra aumenta, quindi fondamentalmente non sono correlate.

Inoltre, è stata posta l'attenzione anche su un'altra "metrica", ovvero l'intersezione delle maschere estratte per ogni frase di ogni parola target. In questo caso si vuole ricercare il rapporto che ci possa essere tra il numero di sensi di una parola e il numero di maschere/token indotti per la parola stessa, attraverso la loro presenza nelle frasi estratte.

Mostriamo di seguito delle tabelle che racchiudono i risultati ottenuti sia per quanto riguarda il task con sole parole italiane che il task con coppie di parole italiano-inglese, dove per ciascuno viene mostrata una legenda dei simboli utilizzati.

Italiano

	Sim. (Vecs_cls)	**Sim. (Vecs_Token)**	**Intersezione**	**Fr.ass**
Nomi	-0.14071 | pv: 0.008	**-0.20383** | pv: 0.0001	-0.16215 | pv: 0.002	0.02990 | pv: 0.57
Aggettivi	-0.11324 | pv: 0.29	**-0.24717** | pv: 0.02	-0.16699 | pv: 0.12	0.00168 | pv: 0.98
Verbi	-0.09005 | pv: 0.27	**-0.18264** | pv: 0.02	-0.14446 | pv: 0.08	0.04539 | pv: 0.58
Avverbi	-0.13689 | pv: 0.52	-0.09322 | pv: 0.66	**-0.23649** | pv: 0.26	0.11530 | pv: 0.59

Tabella 5.1: Risultati Wiktionary

	Sim. (Vecs_cls)	**Sim. (Vecs_Token)**	**Intersezione**	**Fr.ass**
Nomi	-0.19616 | pv: 0.0002	**-0.26868** | pv: 2.9e-07	0.05798 | pv: 0.27	0.09886 | pv: 0.06
Aggettivi	-0.16323 | pv: 0.13	**-0.39881** | pv: 0.0001	-0.10069 | pv: 0.35	0.09720 | pv: 0.37
Verbi	**-0.15551** | pv: 0.05	0.05195 | pv: 0.53	0.08504 | pv: 0.3	-0.04720 | pv: 0.57
Avverbi	**-0.19156** | pv: 0.36	-0.45958 | pv: 0.02	-0.03822 | pv: 0.85	0.13895 | pv: 0.51

Tabella 5.2: Risultati WordNet

	Sim. (Vecs_cls)	**Sim. (Vecs_Token)**
Nomi	0.04436 | pv: 0.4	**-0.07487** | pv: 0.16
Aggettivi	**0.07148** | pv: 0.51	0.05265 | pv: 0.6
Verbi	**0.00728** | pv: 0.93	-0.05335 | pv: 0.52
Avverbi	**-0.06347** | pv: 0.76	-0.25478 | pv: 0.22

Tabella 5.3: Correlazione similarità del coseno e frequenze assolute

	WikSenses - WordSenses
Nomi	0.29679 | pv: 1.3e-08
Aggettivi	0.13293 | pv: 0.21
Verbi	0.27498 | pv: 0.0007
Avverbi	-0.20908 | pv: 0.32

Tabella 5.4: Correlazione numero di sensi in Wiktionary e numero di synset in Wordnet

	Sim. (Vecs_cls) - Int.	**Sim. (Vecs_Token) - Int.**
Nomi	0.08281 | pv: 0.12	**0.36915** | pv: 7.7e-13
Aggettivi	0.02064 | pv: 0.84	**0.43190** | pv: 2.9e-05
Verbi	-0.04491 | pv: 0.58	**0.22404** | pv: 0.006
Avverbi	0.31130 | pv: 0.13	**0.38173** | pv: 0.06

TABELLA 5.5: Correlazione intersezioni e similarità del coseno

Legenda:

Sim. = lista di score della similarità del coseno di ogni parola per la classe semantica.

Intersezione/Int. = lista di score dell'intersezione di ogni token di ogni parola per la classe semantica.

WikSenses = lista di numero di sensi di Wiktionary di ogni parola per la classe semantica.

WordSenses = lista di numero di sensi di WordNet di ogni parola per la classe semantica.

Fr.ass = lista di numero di frequenze assolute di ogni parola per la classe semantica.

Possiamo osservare come nella tabella (5.1), per quanto riguarda i risultati di Wiktionary, questi siano tutti negativi; quindi rappresentano delle correlazioni inverse tranne nel caso dell'ultima colonna dove ritroviamo il confronto tra i sensi di Wiktionary e le frequenze assolute, dove i valori sono positivi ma rimangono molto vicini allo 0. Anche se per la classe semantica degli Avverbi viene mostrato un risultato di 0.11530. Nonostante questo vi è presenza di significatività, soprattutto per i valori della classe semantica dei Nomi per i primi 3 approcci (similarità Vecs_cls, similarità Vecs_Token e intersezione) e per i risultati dell'approccio della similarità dei Vecs_Token per le classi Aggettivi e Avverbi con dei p-value di 0.02 ciasuno. Inoltre il maggior valore per la terza colonna è dato dalla classe degli Avverbi con un valore di correlazione negativa di -0.23649.

I risultati, nei confronti di WordNet, si mantengono prevalentemente simili per la tabella (5.2) ma con una presenza maggiore di risultati positivi e quindi di possibili correlazioni che non siano inverse; precisamente li ritroviamo nelle ultime due colonne, con gli approcci dell'intersezione e delle frequenze assolute. Ma rimangono comunque dei valori molto vicini allo 0 e inoltre non ci sono p-value che li possano rendere significativi. Nella seconda colonna, la similarità dei Vecs_Token, il valore migliore ed anche significativo è dato dalla classe degli Avverbi con un valore di -0.45958 e un p-value di 0.02. Questo ci indica che vi è una correlazione inversa tra la similarità del coseno delle parole rappresentate dai Vecs_Token e il numero di synset di Wordnet, all'aumentare di uno diminuisce l'altro; è un risultato che si avvicina molto alla metà della soglia per poter raggiungere una perfetta correlazione inversa (-1). Complessivamente

ci sono una serie di valori significativi, sia nella prima colonna che nella seconda (Vecs_cls e Vecs_Token), ma la classe semantica dei Nomi mantiene la significatività in entrambe.

Per quanto riguarda la tabella (5.3), la quale si occupa di mostrare la relazione tra la similarità del coseno sia dei Vecs_cls che dei Vecs_Token e le frequenze assolute, si può notare come i risultati siano perlopiù positivi e che solo nella seconda colonna siano quasi tutti negativi, mostrando per l'appunto una correlazione inversa rispetto alle frequenze. Purtoppo non ci sono valori significativi, i p-value presenti sono tutti troppo alti per il valore di confronto ($p < 0.05$).

Nella tabella 5.4 gli score si presentano tutti positivi tranne che nel caso della classe semantica Avverbi con un valore di -0.20908. Nonostante ciò i valori si avvicinano alla soglia dello 0.3, con le classi semantiche dei Nomi e dei Verbi, i quali si mostrano essere anche valori significativi con p-value < 0.05. Questo significa che vi è un iniziale approccio verso una correlazione, quindi ad una "crescita" uguale di entrambe le tipologie di sensi, e i valori significativi danno supporto agli score ottenuti.

Infine nell'ultima tabella (5.5), vengono confrontate le due metriche principali, la similarità e l'intersezione. Si può vedere come a differenza dei Vecs_cls, i Vecs_Token riescano a superare la soglia del 0.3 e addirittura nel caso della classe semantica Aggettivi arrivino fino a 0.43190. Inoltre i Vecs_Token riportano quasi tutti valori significativi tranne che nel caso della classe semantica degli Avverbi. Per i valori Vecs_cls, a differenza dei Vecs_Token i valori rimangono molto vicini allo 0, a differenza dell'unico valore che si avvicina al 0.3, della classe degli Avverbi.

Italiano-Inglese

	Sim. - WordSenses	**Sim. - Fr. ass**
Nomi	**-0.05664** | pv: 0.18	0.05653 | pv: 0.18
Aggettivi	**-0.04441** | pv: 0.59	-0.08996 | pv: 0.28
Verbi	**-0.03989** | pv: 0.49	-0.01103 | pv: 0.85
Avverbi	**-0.16891** | pv: 0.18	0.12567 | pv: 0.33

Tabella 5.6: Vecs_cls Ita-Eng

	Sim. - WordSenses	**Sim. - Fr. ass**
Nomi	**0.06459** | pv: 0.12	0.04698 | pv: 0.26
Aggettivi	**0.05301** | pv: 0.52	0.035610 | pv: 0.67
Verbi	0.00854 | pv: 0.88	**0.01224** | pv: 0.83
Avverbi	**0.09631** | pv: 0.45	0.02716 | pv: 0.83

Tabella 5.7: Vecs_Token Ita-Eng

	WordSenses - Fr.ass
Nomi	0.31631 | pv: 2.07e-14
Aggettivi	0.23409 | pv: 0.004
Verbi	0.29395 | pv: 3.6e-07
Avverbi	0.27402 | pv: 0.03

TABELLA 5.8: Confronto Sensi e Frequenza Ita-Eng

Legenda:

Sim. = lista di score della similarità del coseno di ogni coppia di parole ita-eng per la classe semantica.

WordSenses = lista dell'intersezione del numero di sensi di WordNet di ogni coppia di parole ita-eng per la classe semantica.

Fr.ass = lista dell'intersezione del numero di frequenze assolute di ogni coppia di parole ita-eng per la classe semantica.

Allo stesso modo, per quanto riguarda le coppie di parole italiano-inglese, possiamo vedere come nella tabella (5.6) i risultati siano complessivamente negativi. Quindi si ha la presenza di una correlazione inversa con uno score migliore di -0.16891 nel caso della classe Avverbi, ma nessun risultato risulta effettivamente significativo, poichè tutti i p-value si presentano come maggiori di 0.05.

Nella tabella (5.7) al contrario, i risultati sono tutti positivi, quindi avremo la presenza di una correlazione positiva, con il miglior risultato sempre nei confronti della classe semantica degli Avverbi con un punteggio di 0.09631. Anche in questo caso nessun risultato sarà significativo.

Infine osserviamo come nella tabella (5.8) ci siano solo valori positivi e che il migliore tra di essi sia quello della classe semantica dei Nomi con un punteggio di 0.31631. I dati si presentano come tutti significativi presentando dei p-value < 0.05 per ogni classe semantica.

Discussione finale

Riassumendo, abbiamo potuto vedere come, per i risultati di parole italiane, ci sia una prevalenza di valori negativi, ovvero delle correlazioni inverse, soprattutto nel caso delle due tabelle che mostrano i risultati degli approcci nei confronti di WordNet e Wiktionary. Le relazioni osservate tra i vari risultati, non hanno una crescita e sviluppo costante, ma crescono inversamente: all'aumentare di una variabile l'altra decresce. Quindi effettivamente, come nel caso della similarità del coseno a confronto col numero di sensi di una parola (che siano di Wordnet o Wiktionary), questo indicherà che all'aumentare del valore della similarità, il numero di significati che una parola può assumere diminuiscano, proprio perchè in tutte le frasi in cui la parola target compare, riesce a

mantenere una cerchia stretta di significati diversi permettendole di rimanere "simile" nella varie frasi.

Possiamo notare in maniera generale come tra tutti gli approcci osservati (Vecs_cls, Vecs_Token, intersezione, frequenze assolute) i Vecs_Token siano autori delle migliori performance in quasi tutte le tabelle, specialmente nei confronti di Wiktionary e nel caso della correlazione tra le intersezioni e la similarità del coseno. Infatti a differenza dei valori di Vecs_cls, che siano positivi o negativi, sono sempre maggiori e si avvicinano alla metà della soglia del valore ricercato. Nel confronto con l'intersezione, tutte le classi semantiche mostrano valori più alti rispetto alla controparte di Vecs_cls. Spesso inoltre per i Vecs_Token vengono ritrovati valori significativi. Poichè i Vecs_Token considerano solo i token della parola target, questa differenza di performance ci indica che di fatto Vecs_Token sia una migliore rappresentazione per la parola target, mentre invece per l'appunto Vecs_cls rappresenta soltanto il primo token di una parola e dando performance così scadenti dimostra di non essere la scelta migliore tra le due.

Per quanto riguarda la presenza di valori significativi, guardando tutti i risultati raccolti e suddivisi, spesso la classe semantica Nomi è artefice di risultati che riportano significatività, ovvero p-value che sono $p < 0.05$, come ad esempio si può vedere per tutti i valori di Wiktionary, Wordnet ed anche per il confronto tra similarità e intersezione. In questo modo, grazie alla presenza dei valori significativi, si può ricevere supporto da essi nella validazione degli score ottenuti, quindi contribuire a dimostrare che siano valori effettivi e non ritrovati in maniera casuale.

Invece nei confronti delle coppie di parole italiane-inglesi, complessivamente i valori riportati sono prevalentemente positivi, tranne che nel caso del confronto tra la similarità del coseno e i sensi di WordNet: sono tutte correlazioni negative, molto vicine allo 0, tranne che nel caso della classe degli Avverbi che si distacca di poco avvicinandosi al -0.2. Questo nuovamente ci riporta all'ipotesi che la similarità cresca inversamente rispetto al numero di sensi, ma in questo caso, per i risultati ottenuti, si potrebbe dire che questa correlazione sia debole, visti i valori così bassi.

Anche in questo caso vi è una differenza tra i valori di Vecs_Token e Vecs_cls, ma sarà più legata al tipo di correlazione poichè i Vecs_cls avranno valori negativi mentre i Vecs_Token saranno positivi.

Come ultima osservazione, notiamo come ci sia grande assenza di significatività, i valori sono troppo alti e non rientrano nella soglia ottimale (p

< 0.05), tranne che nel caso del confronto tra il numero di synset di WordNet e le frequenze assolute dove tutti i valori, per ogni classe semantica, sono significativi.

Capitolo 6

Conclusioni

L'obiettivo finale di questo progetto è stato quello di effettuare una induzione automatica sui significati di una serie di parole polisemiche selezionate ed estratte da risorse online lessicali, applicando tecniche di semantica distribuzionale.

Come abbiamo visto nella sezione (3.4), grazie al lavoro svolto nelle precedenti fasi di estrazione delle parole dalle risorse Wordnet e Wiktionary, si sono potuti raccogliere e selezionare i dati desiderati, ovvero le parole polisemiche target, su cui poter applicare un'induzione automatica, utilizzando un modello basato sui Transformers adatto ad un contesto di NLP, sul numero di significati, tramite l'utilizzo di tecniche di semantica distribuzionale.

L'utilizzo di un modello basato sull'apprendimento non supervisionato ha portato interessanti risultati. Grazie al lavoro svolto sulla valutazione degli score ottenuti tramite l'utilizzo delle statistiche scelte, si è potuto riscontrare come la similarità tra le parole target scelte sia effettivamente legata al numero di significati di una parola ma in una correlazione inversa: al crescere di uno diminuisce l'altro. Questo significa che effettivamente una parola rimane legata ai suoi significati nonostante possa ritrovarsi in frasi di contesto diverse. L'utilizzo di tecniche semantiche quindi ha dato la possibilità di calcolare la "distanza" semantica tra parole diverse e quindi ci ha permesso di poter verificare l'ipotesi esposta. Inoltre con l'uso del p-value, si è potuto dare supporto statistico all'ipotesi proposta, avendo a disposizione una metrica che potesse dare un riscontro sui valori ottenuti nella fase di Sperimentazione, con l'utilizzo del coefficiente di Spearman. Inoltre dalla discussione finale si è dedotto come i Vecs_Token, rispetto ai Vecs_cls, siano una migliore rappresentazione delle parole target, grazie ai risultati delle loro performance, valutate sempre nella fase di Sperimentazione.

Ci potrebbero essere degli sviluppi futuri di questo progetto di tesi, sia dal punto di vista del miglioramento tecnico delle funzionalità, che da quello di possibili estensioni che potrebbero essere implementate ed aggiunte al sistema attuale.

Ad esempio potrebbe essere usato uno stesso modello di apprendimento non supervisionato basato sui Transformers, ma ci si potrebbe occupare di una fase di training del modello, piuttosto che fare uso di uno pre-addestrato: questo potrebbe rendere le predizioni più accurate e quindi verrebbero forniti dei risultati migliori da cui partire per la verifica delle ipotesi. In questo progetto di tesi inoltre non è stato approfondito l'aspetto dell'intersezione delle parole mascherate per ogni frase di ogni parola target osservata: questo potrebbe portare a riscontri interessanti, soprattutto per quanto riguarda il rapporto delle parole in contesti diversi. Infine un altro aspetto che non ha avuto modo di essere applicato, ma potrebbe esserlo in futuro, è quello del task di Masking: uno dei compiti principali del modello utilizzato nei confronti delle coppie di parole italiano-inglesi; non vi era modo di effettuare un confronto tra parole italiane e inglesi col materiale a disposizione quindi si è preferito evitare l'applicazione del task stesso.

Bibliografia

[1] A. G. Oettinger. «Computational Linguistics». In: *The American Mathematical Monthly* 72.2 (1965), pp. 147–150.
ISSN: 00029890, 19300972. URL: http://www.jstor.org/stable/2313322 (visitato il 23/11/2022).

[2] Suresh Manandhar et al. «SemEval-2010 Task 14: Word Sense Induction &Disambiguation». In: *Proceedings of the 5th International Workshop on Semantic Evaluation*. Uppsala, Sweden: Association for Computational Linguistics, lug. 2010, pp. 63–68. URL: https://aclanthology.org/S10-1011.

[3] Jey Han Lau et al. «Word sense induction for novel sense detection». In: *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. 2012, pp. 591–601.

[4] David M Blei, Andrew Y Ng e Michael I Jordan. «Latent dirichlet allocation». In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[5] Yee Whye Teh et al. «Hierarchical Dirichlet Processes». In: *Journal of the American Statistical Association* 101.476 (2006), pp. 1566–1581.
DOI: 10.1198/016214506000000302. eprint: https://doi.org/10.1198/016214506000000302. URL: https://doi.org/10.1198/016214506000000302.

[6] Xuchen Yao e Benjamin Van Durme. «Nonparametric Bayesian Word Sense Induction». In: *Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing*. Portland, Oregon: Association for Computational Linguistics, giu. 2011, pp. 10–14. URL: https://aclanthology.org/W11-1102.

[7] Sebastian Padó e Mirella Lapata. «Dependency-Based Construction of Semantic Space Models». In: *Computational Linguistics* 33.2 (2007), pp. 161–199.
DOI: 10.1162/coli.2007.33.2.161. URL: https://aclanthology.org/J07-2002.

[8] Ioannis Korkontzelos e Suresh Manandhar. «UoY: Graphs of Unambiguous Vertices for Word Sense Induction and Disambiguation». In: *Proceedings of the 5th International Workshop on Semantic Evaluation*. Uppsala, Sweden: Association for Computational Linguistics, lug. 2010, pp. 355–358. URL: https://aclanthology.org/S10-1079.

[9] Adriano Ferraresi et al. «Introducing and evaluating ukWaC , a very large web-derived corpus of English». In: 2008.

[10] Helmut Schmidt. «Probabilistic part-of-speech tagging using decision trees». In: 1994.

[11] Samuel Brody e Mirella Lapata. «Bayesian word sense induction». In: *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*. 2009, pp. 103–111.

[12] Yoong Keok Lee e Hwee Tou Ng. «An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation». In: *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. 2002, pp. 41–48.

[13] Stuart Geman e Donald Geman. «Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6 (1984), pp. 721–741.
DOI: 10.1109/TPAMI.1984.4767596.

[14] Eneko Agirre e Aitor Soroa. «Semeval-2007 task 02: Evaluating word sense induction and discrimination systems». In: *Proceedings of the fourth international workshop on semantic evaluations (semeval-2007)*. 2007, pp. 7–12.

[15] Eduard Hovy et al. «OntoNotes: The 90% Solution». In: *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. New York City, USA: Association for Computational Linguistics, giu. 2006, pp. 57–60. URL: https://aclanthology.org/N06-2015.

[16] Mohammad Taher Pilehvar e Jose Camacho-Collados. «WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations». In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, giu. 2019,

pp. 1267–1273.
DOI: 10.18653/v1/N19-1128. URL: https://aclanthology.org/N19-1128.

[17] Oren Melamud, Jacob Goldberger e Ido Dagan. «context2vec: Learning generic context embedding with bidirectional lstm». In: *Proceedings of the 20th SIGNLL conference on computational natural language learning*. 2016, pp. 51–61.

[18] Sepp Hochreiter e Jürgen Schmidhuber. «Long short-term memory». In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[19] Marilyn A. Walker, Heng Ji e Amanda Stent, cur. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018.
ISBN: 978-1-948087-27-8. URL: https://aclanthology.org/volumes/N18-1/.

[20] Jacob Devlin et al. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, giu. 2019, pp. 4171–4186.
DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

[21] Maria Pelevina et al. «Making Sense of Word Embeddings». In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. Berlin, Germany: Association for Computational Linguistics, ago. 2016, pp. 174–183.
DOI: 10.18653/v1/W16-1620. URL: https://aclanthology.org/W16-1620.

[22] Mohammad Taher Pilehvar e Nigel Collier. «De-Conflated Semantic Representations». In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, nov. 2016, pp. 1680–1690.
DOI: 10.18653/v1/D16-1174. URL: https://aclanthology.org/D16-1174.

[23] Massimiliano Mancini et al. «Embedding Words and Senses Together via Joint Knowledge-Enhanced Training». In: *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Vancouver, Canada: Association for Computational Linguistics, ago. 2017, pp. 100–111.
DOI: 10.18653/v1/K17-1012. URL: https://aclanthology.org/K17-1012.

[24] Tomás Mikolov et al. «Efficient Estimation of Word Representations in Vector Space». In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. A cura di Yoshua Bengio e Yann LeCun. 2013. URL: http://arxiv.org/abs/1301.3781.

[25] Roberto Navigli. «Word sense disambiguation: A survey». In: *ACM computing surveys (CSUR)* 41.2 (2009), pp. 1–69.

[26] Christiane Fellbaum. «WordNet». In: *Theory and applications of ontology: computer applications*. Springer, 2010, pp. 231–243.

[27] Alessandro Raganato et al. «XL-WiC: A Multilingual Benchmark for Evaluating Semantic Contextualization». In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, nov. 2020, pp. 7193–7206.
DOI: 10.18653/v1/2020.emnlp-main.584. URL: https://aclanthology.org/2020.emnlp-main.584.

[28] Mehrnoush Shamsfard et al. «Semi automatic development of farsnet; the persian wordnet». In: *Proceedings of 5th global WordNet conference, Mumbai, India*. Vol. 29. 2010.

[29] Ashish Vaswani et al. «Attention is all you need». In: *Advances in neural information processing systems* 30 (2017).

[30] Alexis Conneau et al. «Unsupervised Cross-lingual Representation Learning at Scale». In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, lug. 2020, pp. 8440–8451.
DOI: 10.18653/v1/2020.acl-main.747. URL: https://aclanthology.org/2020.acl-main.747.

[31] Torsten Zesch, Christof Müller e Iryna Gurevych. «Using Wiktionary for Computing Semantic Relatedness.» In: *AAAI*. Vol. 8. 2008, pp. 861–866.

[32] Tatu Ylönen. «Wiktextract: Wiktionary as Machine-Readable Structured Data». In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference, LREC 2022, Marseille, France, 20-25 June 2022*. A cura di Nicoletta Calzolari et al. European Language Resources Association, 2022, pp. 1317–1325. URL: https://aclanthology.org/2022.lrec-1.140.

[33] Alessandro Lenci. «Distributional semantics in linguistic and cognitive research». In: *Italian journal of linguistics* 20.1 (2008), pp. 1–31.

Ringraziamenti

Vorrei ringraziare il relatore Pierpaolo Basile che mi ha permesso di poter partecipare a questo progetto con l'aiuto e l'assistenza degli altri componenti del team. E' stata un'esperienza che mi ha dato modo di accedere ad argomenti a me sconosciuti e poter così esplorare nuovi aspetti della ricerca dell'informazione.

Vorrei ringraziare la mia famiglia che mi ha dato sostegno e supporto durante questi anni, tra bocciature, delusioni e sudate promozioni.

Vorrei ringraziare tutti i miei amici e colleghi universitari, mi sono rimasti accanto e mi hanno accompagnato in questo percorso, chi è rimasto fino alla fine chi meno, ma soprattutto uno speciale ringraziamento al mio amico e collega Roberto: non è stato difficile fare amicizia sin dal primo giorno di lezione e nonostante il covid, la distanza e le possibili incompresioni, siamo rimasti sempre vicini e legati. Abbiamo condiviso bei momenti, situazioni stressanti e tristi, ma ci siamo dati la forza l'un l'altro e abbiamo affrontato anche la nostra carriera universitaria assieme.

Ma soprattutto vorrei ringraziare me stessa: nonostante io spesso sia molto dura ed esigente con i miei obiettivi, sono riuscita ad accompagnarmi verso questa laurea, ho avuto fiducia e sono sempre andata avanti, nonostante tutte le difficoltà incontrate nel cammino.