**Lezione 2**

Le ontologie risultano particolarmente utili nell’ambito dell’analisi automatica del linguaggio naturale, soprattutto quando si fa riferimento ad approcci cognitivi come DOLCE, utilizzati per la costruzione di ontologie fondazionali. Tali ontologie sono impiegate principalmente da coloro che hanno l’esigenza di ancorare e strutturare ontologie di dominio su basi concettuali molto generiche.

All’interno del livello più alto delle ontologie fondazionali, si individua una suddivisione iniziale fondamentale tra Endurant e Perdurant, a cui si affiancano altri elementi come le Qualità. Questo modo di partizionare l’ontologia a livello top in entità e processi è una caratteristica propria non solo di DOLCE, ma anche di altre ontologie fondazionali di tipo top-level.

I tratti distintivi di DOLCE includono appunto le categorie di Endurant e Perdurant, nonché il ruolo delle Qualità, da intendere anche secondo un approccio moltiplicativo, ovvero un approccio che consente la collocazione spaziotemporale differenziata di oggetti o eventi.

Gli Endurant rappresentano entità che permangono nel tempo: si tratta di oggetti che hanno una relazione di partecipazione e di esistenza indicizzata temporalmente. Questo significa che tali oggetti possono mutare nel tempo — ad esempio, una mela può cambiare colore, marcire — pur restando lo stesso oggetto. Le loro parti possono essere inessenziali, e tutte le parti dell’oggetto sono compresenti per l’intera durata della sua esistenza. Non vi sono, dunque, parti che esistono solo in alcuni momenti della vita dell’oggetto.

I Perdurant, invece, sono eventi o processi. A differenza degli Endurant, essi non si mantengono nel tempo in modo unitario, ma si verificano nel tempo. Non mutano al loro interno durante il tempo, ma si compongono di parti temporali distinte: una chiacchierata, per esempio, è costituita da momenti successivi che non esistono simultaneamente. In un Perdurant, solo alcune delle sue componenti sono presenti in ciascun istante temporale. Tutte le sue parti sono essenziali e contribuiscono alla sua identità.

La relazione fondamentale tra Endurant e Perdurant è quella della partecipazione: gli Endurant partecipano nei Perdurant. Per esempio, una persona che parla è un Endurant che partecipa in una conversazione, che è un Perdurant.

Infine, è importante soffermarsi sul concetto di sottoclasse, che in un'ontologia non va inteso secondo l’accezione usuale della programmazione, ma piuttosto come una relazione logica e semantica che consente di costruire una gerarchia concettuale. Tale gerarchia è l’ossatura stessa dell’ontologia e ne definisce la struttura ad albero.

Nel contesto delle ontologie, è possibile affermare che, se una classe A è sottoclasse di una classe B, allora tutte le istanze di A sono anche istanze di B. In altri termini, A è un sottoinsieme di B, e le istanze di A ereditano i valori dei componenti di B. Ad esempio, se si sostiene che ogni essere umano è un mammifero, ne consegue che ogni essere umano è anche un vertebrato e respira aria, sfruttando la gerarchia delle classi.

Tuttavia, questa relazione non è sempre semplice da analizzare, soprattutto quando si ha a che fare con eventi. Mentre nel caso degli oggetti — come una sedia — risulta relativamente semplice individuare gli elementi costitutivi e costruire una gerarchia concettuale, nel caso degli eventi la questione si complica. Si consideri, ad esempio, l’evento rappresentato da una giornata. Si potrebbe affermare che una giornata si articola in sottosequenze o episodi come il risveglio, il lavoro, una partita a tennis, il rientro a casa e il sonno. Tuttavia, questi elementi non risultano automaticamente in relazione di sottoclasse rispetto all’intero evento "giornata".

A questo proposito, è utile riflettere su come riconoscere se due entità sono effettivamente in relazione di sottoclasse. Un esempio calzante è quello della distinzione tra *time duration* e *time interval*. Una *time duration* potrebbe essere un’ora, mezza giornata, una durata arbitraria; un *time interval*, invece, corrisponde a un segmento temporale ben delimitato da un inizio e una fine specifici. Per valutare se tra queste due entità sussiste una relazione di sottoclasse, si può ricorrere al concetto di *criterio di identità*.

Il criterio di identità consiste nell’insieme delle condizioni necessarie per determinare l’eguaglianza tra entità. Per esempio, due durate temporali sono identiche se hanno la stessa lunghezza, mentre due intervalli temporali sono identici se hanno il medesimo inizio e la stessa fine. Questo porta a concludere che *time duration* e *time interval* non possono essere considerate in una relazione di superclasse e sottoclasse, poiché non condividono lo stesso criterio di identità.

La difficoltà nell’analisi degli eventi si lega dunque alla complessità nel definire chiaramente i criteri di classificazione e di identità. Un'altra relazione fondamentale nell’ambito delle ontologie, particolarmente utile per l’analisi degli eventi e degli oggetti complessi, è la *relazione part-of*, o di meronia.

Chi si occupa di ontologia, in particolare in contesti articolati, considera questa relazione essenziale. La capacità di individuare le componenti o le parti di un'entità è spesso uno dei primi passi nella rappresentazione concettuale di un dominio. Le relazioni *part-of* possono assumere diverse forme. Alcuni esempi includono:

* **Componente**: come la maniglia che è parte della portiera di un’automobile.
* **Ingrediente**: come la farina che è ingrediente del pane.
* **Porzione**: ad esempio, una fetta che è parte del pane.
* **Area**: come una regione geografica all’interno di un territorio più ampio.
* **Membro**: ad esempio, una nave che è membro di una flotta.
* **Partner**: come i componenti di un trio, come evocativamente Kernighan e Ritchie.
* **Pezzo**: inteso come un frammento staccabile, separato dal tutto cui appartiene.

Queste varianti della relazione meronica mostrano la ricchezza e la flessibilità dell’approccio ontologico nella descrizione del mondo, sia esso materiale o concettuale.

DOLCE è una delle ontologie fondazionali più note e utilizzate, sia in ambito accademico che produttivo. Non rappresenta l’unico approccio possibile alla costruzione ontologica, ma è ampiamente adottata per la sua struttura concettuale chiara e per la capacità di adattarsi a domini molto differenti. In particolare, la distinzione tra *Endurant* e *Perdurant* costituisce uno degli assi portanti del suo impianto teorico.

Gli *Endurant* sono entità che persistono nel tempo pur potendo cambiare. Possono modificare le proprie qualità e mantenere comunque la propria identità. Un esempio classico è una mela, che può maturare, marcire o cambiare colore, restando pur sempre la stessa mela. Al contrario, i *Perdurant* sono eventi o processi che si sviluppano nel tempo, e che non mutano nel senso in cui cambiano gli oggetti: un evento come una partita di tennis o una chiacchierata è costituito da una successione di fasi, e solo alcune sue parti sono presenti in ogni singolo momento.

Esiste una relazione fondamentale tra queste due categorie: gli *Endurant* partecipano nei *Perdurant*. Un esempio è una persona (Endurant) che prende parte a una riunione (Perdurant).

Una caratteristica rilevante di DOLCE è anche la teoria delle *Qualità*. Questa teoria assume particolare importanza perché consente di reintrodurre un orizzonte numerico all’interno di un impianto concettuale fondato sulla logica e sugli assiomi. In altre parole, permette di rappresentare formalmente informazioni quantitative, come il colore, la temperatura, o altre caratteristiche misurabili, pur rimanendo nel quadro logico-ontologico.

L’intuizione alla base della teoria delle qualità si fonda sulla distinzione tra la *qualità* e il *valore* assunto da quella qualità. Prendendo come esempio il colore di una rosa, si distingue tra la qualità “colore” (intesa come attributo) e il valore specifico del colore (ad esempio, rosso). In ambiti produttivi, un esempio simile può essere il colore richiesto per un componente meccanico, che deve rispettare determinati standard cromatici.

La rappresentazione di un valore di qualità non avviene direttamente attraverso numeri, ma attraverso il concetto di *quale*, ovvero un'entità che occupa una posizione in uno *spazio di qualità*. Ad esempio, il colore rosso può essere rappresentato come un punto nello spazio del colore. Quando si afferma che due oggetti hanno lo stesso colore, si sta dicendo che le rispettive qualità di colore — distinte perché riferite a oggetti diversi — occupano la stessa posizione nello spazio del colore.

Questa modalità di rappresentazione è simile, concettualmente, alla gestione degli indirizzi in memoria in informatica: si confrontano posizioni piuttosto che oggetti in sé. Sebbene non si parli di memoria centrale, si indicizzano le posizioni all’interno di uno spazio astratto, come lo spazio dei colori.

Questa struttura consente una notevole espressività: si possono rappresentare affermazioni differenti che coinvolgono la stessa qualità. Per esempio: “questa rosa è rossa” implica un aggettivo riferito a un’entità concreta; “il rosso è un colore” invece trasforma il rosso in un sostantivo. Ancora, affermazioni come “il colore della rosa vira verso il marrone” o “il colore sta cambiando” presuppongono un movimento all’interno dello spazio di qualità, mantenendo però una continuità identitaria. È come immaginare un punto che si sposta all’interno di una regione colorimetrica, rappresentando così dinamicamente il cambiamento senza perdere la coerenza logica della rappresentazione.

Nel caso sei, “Red is opposite to green and close to brown”, non si parla più propriamente di colori specifici, bensì di *regioni* di colore. Questo rende il riferimento agli *spazi di qualità* particolarmente immediato: è in questi casi che si coglie con maggiore chiarezza l’utilità del concetto. Un esempio classico è il *color spindle*, uno schema tridimensionale che rappresenta le dimensioni di *brightness*, *intensity* e *hue*. È un modello che consente di collocare punti nello spazio per rappresentare colori, permettendo idealmente di dire: “questo oggetto ha questo colore” e “quest’altro ha lo stesso colore”. Si tratta di entità distinte, con caratterizzazioni autonome, che però occupano lo stesso punto nello spazio cromatico, nel caso condividano la stessa tonalità.

Immagine che contiene diagramma, linea, testo, cerchio

Il contenuto generato dall'IA potrebbe non essere corretto.

Un possibile schema utile per memorizzare questo meccanismo potrebbe essere strutturato come segue: si consideri una rosa, la quale può essere parzialmente collocata all’interno di uno *spazio della rossitudine*. A partire da ciò, si può affermare che esista un certo colore associato a un individuo — ad esempio, un colore specifico della *rosa 1*. Tale associazione si realizza attraverso una qualità che la rosa possiede e che corrisponde a un certo *quale* (valore in uno spazio di qualità). È poi possibile collegare questo colore a diversi elementi o individui, ciascuno dotato della propria qualità specifica. Qualora si desideri, è anche possibile aggiungere un’informazione temporale, estendendo la rappresentazione a una tripla, che includa un indice temporale: in questo modo si può affermare che, in un determinato istante, un oggetto possiede un certo colore.

Immagine che contiene testo, schermata, diagramma, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Naturalmente, questa modalità di rappresentazione non si limita ai colori. Può essere estesa ad altre qualità numeriche e rilevanti per il dominio applicativo, come la temperatura, la pressione, la densità, ecc. La teoria delle qualità, così come proposta in DOLCE, si rivela in questo senso estremamente funzionale: è intuitiva da comprendere dal punto di vista geometrico e al contempo molto efficiente nel dotare le ontologie di una capacità espressiva arricchita, tale da includere informazioni quantitative di grande utilità.

A questo punto si conclude la parte relativa alla teoria delle qualità, che avrebbe dovuto essere affrontata nella lezione precedente, e si introduce un nuovo tema, strettamente legato a quanto già trattato: le *reti semantiche*.

Si parte dall’analisi delle forme più semplici, storicamente più antiche, di rete semantica, per poi confrontare le loro caratteristiche con quelle dei sistemi basati su logica. Si evidenziano similitudini e differenze, con uno sguardo anche ai sistemi a *frame* e alla cosiddetta *semantica procedurale*.

Pianificatori e algoritmi come A\*, IDA\*, ecc., pur essendo strumenti brillanti per risolvere problemi all’interno di spazi degli stati, non fanno uso della conoscenza di dominio. L’algoritmo A\*, ad esempio, funziona allo stesso modo sia che venga applicato per trovare un percorso in un grafo, sia che venga usato per risolvere il gioco del 15. La struttura del problema viene adattata all’algoritmo, non viceversa.

Il gioco del 15, per chi non lo conoscesse, è un classico esempio di problema combinatorio: una griglia quadrata di lato 4 con 15 tessere da disporre in ordine crescente, a partire da una configurazione iniziale casuale. La soluzione consiste in una sequenza di mosse che trasformano la configurazione iniziale in quella target. Tuttavia, si tratta di un problema semplificato, di complessità contenuta, utile solo per testare strategie di risoluzione, ma lontano dalla complessità dei problemi del mondo reale.

Per superare questo limite e affrontare problemi realistici — come la progettazione di infrastrutture in ambienti complessi — è necessario integrare nei sistemi strumenti per rappresentare e sfruttare la conoscenza del dominio. Questo rappresenta un cambio di paradigma significativo: non basta più una strategia algoritmica generale, occorre anche una modellazione del mondo che consenta di inferire, ragionare, prevedere.

Ciò implica la necessità di disporre di un *linguaggio di rappresentazione*, composto da strutture sintattiche che descrivano il frammento di realtà rilevante per l’applicazione, e di un insieme di *regole* o *operazioni* che agiscano su tali strutture. Tali operazioni devono essere ancorate al significato delle strutture, e devono permettere di condurre inferenze sensate. Idealmente, queste regole devono essere formulate come *procedure effettive*, ovvero realizzabili computazionalmente, con buone probabilità di successo.

Attualmente, nel contesto contemporaneo, quando si parla di inferenza si fa spesso riferimento all’inferenza intesa come apprendimento automatico, in particolare come classificazione. Tuttavia, nel contesto di cui si discute qui, il significato di *inferenza* è diverso: essa è più strettamente legata alla sequenza di operazioni che costituiscono una strategia di risoluzione, piuttosto che all’identificazione statistica di pattern.

Storicamente, per ragioni legate all’affinità e alla comune origine disciplinare, i primi strumenti utilizzati per rappresentare il mondo nell’ambito dell’intelligenza artificiale sono stati strumenti di tipo logico. Il mondo veniva descritto attraverso un insieme di enunciati formulati in logica. Se si analizzano le basi di conoscenza utilizzate da grandi algoritmi — ad esempio, i pianificatori classici — si nota che esse sono costituite da insiemi di precondizioni e postcondizioni associate agli operatori.

Questo tipo di rappresentazione presenta però dei limiti. Le formule logiche rappresentano proposizioni indipendenti, e ciò rende difficile il collegamento tra esse. Anche se questa osservazione è stata in parte rivista e studiata, poiché esistono tentativi di strutturare logicamente le formule in regioni o in segmenti coesi, rimane comunque il fatto che, man mano che la base di conoscenza cresce, organizzare queste formule in blocchi significativi può diventare un compito più arduo del problema stesso che si tenta di risolvere.

Si può dunque assumere come punto critico il fatto che, in una rappresentazione puramente logica, non è agevole collegare le varie formule e non è agevole organizzare la conoscenza in blocchi omogenei e navigabili. La logica, in questo senso, è un linguaggio di rappresentazione che richiede meccanismi di calcolo aggiuntivi per produrre inferenze o risultati utili. Questo costituisce una delle grandi limitazioni del formalismo logico, oltre al fatto che un insieme di formule, dal punto di vista dell’interpretazione da parte di un essere umano, non è sempre immediatamente comprensibile.

In contesti nei quali è necessario tracciare il ragionamento — come nella diagnosi medica o nel diritto — è fondamentale poter seguire il filo della computazione e identificare i passaggi intermedi che conducono a una determinata conclusione. La logica non facilita questo tipo di accesso, creando un problema di trasparenza e tracciabilità.

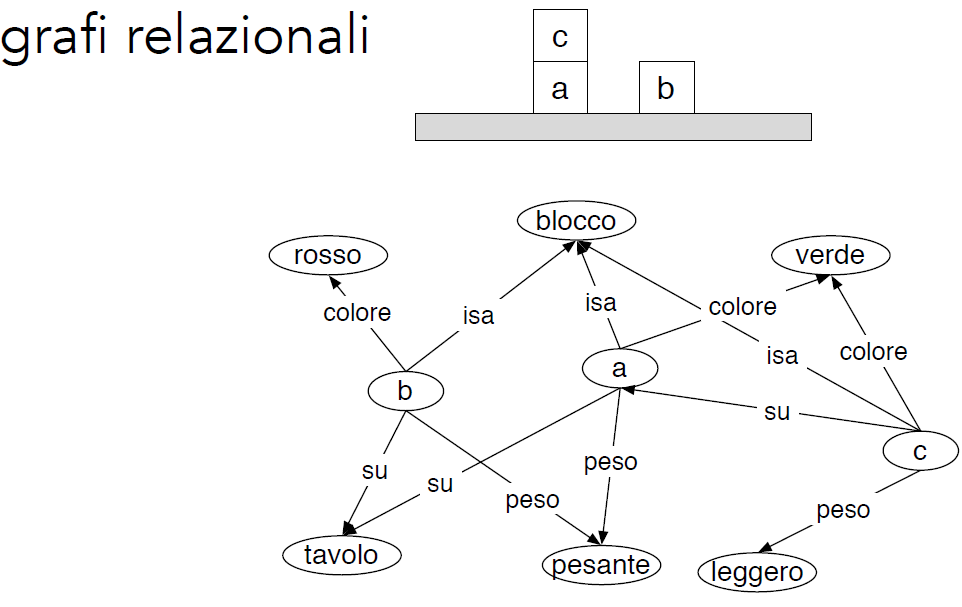
Per rispondere a queste esigenze, nel corso del tempo sono stati sviluppati formalismi alternativi che consentissero di aggregare conoscenze elementari in strutture più articolate e, soprattutto, permettessero un accesso diretto e intuitivo alla conoscenza. I formalismi grafici sono un esempio rilevante: essi consentono di effettuare ragionamenti che, in un sistema logico, richiederebbero l’applicazione di regole di inferenza complesse, con un costo computazionale elevato.

In ambito logico, tali ragionamenti implicherebbero l’utilizzo di dimostratori di teoremi o sistemi deduttivi formali, con tutte le difficoltà legate all’efficienza del calcolo inferenziale. I formalismi grafici, al contrario, semplificano tali ragionamenti integrandoli direttamente nella struttura del sistema: sono meccanismi *cablati*, che permettono di accedere istantaneamente alla conoscenza desiderata, senza dover passare attraverso un percorso deduttivo esplicito.

Le reti semantiche rappresentano una classe generale di strumenti utilizzati per la rappresentazione della conoscenza. Si tratta di un formalismo nato nei primi tentativi di traduzione automatica, oggi considerati quasi teneramente ingenui. Questi formalismi sono detti grafici perché modellano il mondo sotto forma di grafo. In tale struttura, un frammento di realtà rilevante per una determinata applicazione viene rappresentato tramite un grafo in cui i nodi corrispondono ai concetti, mentre gli archi esprimono relazioni tra concetti o proprietà degli stessi.

La forma più semplice di rete semantica è costituita dal grafo relazionale, che consente di descrivere esclusivamente le relazioni tra le entità rappresentate nel grafo. Questo tipo di rappresentazione è stato ampiamente impiegato, in particolare per modellare e rappresentare stati nel contesto del cosiddetto "mondo dei blocchi". Tale ambiente viene spesso introdotto per spiegare i problemi di pianificazione, nei quali si parte da una configurazione iniziale – con un tavolo e un numero variabile di blocchi disposti su di esso, eventualmente impilati – e si cerca di raggiungere una configurazione obiettivo. La soluzione a tali problemi consiste in una sequenza di operazioni che permette il passaggio dallo stato iniziale a quello finale.

Nel caso specifico dei blocchi A, B e C, vengono definite diverse informazioni per ciascuno di essi. Si specifica, ad esempio, la loro appartenenza a una determinata categoria attraverso la relazione "is a" (o "isa"), che indica che un certo elemento è un’istanza di una classe. Per esempio, l’affermazione "a isa blocco" implica che "a" appartiene alla categoria dei blocchi. Il grafo consente anche di annotare ulteriori proprietà, come il colore di un blocco (ad esempio, "b" può essere rosso) o la sua posizione (ad esempio, "b" è sul tavolo, "a" è sul tavolo, "c" è sopra "a").



Questa rappresentazione può essere via via arricchita e resa più complessa, ad esempio introducendo regole che impongono vincoli sulle operazioni ammissibili a seconda delle caratteristiche dei blocchi. Nel mondo dei blocchi, si opera tipicamente su pile (stack), rispettando le limitazioni meccaniche implicite. In questo modo, è possibile condensare una molteplicità di informazioni in una forma sufficientemente leggibile anche per un essere umano, attraverso l’uso di grafi relazionali.

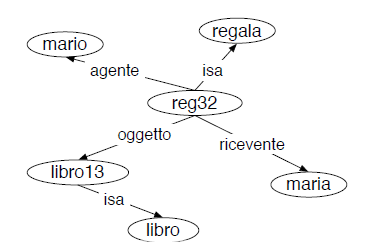
Dal punto di vista formale, i grafi relazionali implementano un sottoinsieme proprio e stretto del calcolo dei predicati del primo ordine. In questa corrispondenza, gli archi rappresentano i predicati mentre i nodi rappresentano i termini. Alcune delle asserzioni esprimibili nel calcolo dei predicati possono essere rappresentate anche mediante grafi relazionali. Tuttavia, questi ultimi offrono un vantaggio notevole in termini di efficacia espressiva grazie alla loro leggibilità visiva.

Tuttavia, vi sono delle limitazioni. Una rappresentazione basata su grafi relazionali può esprimere esclusivamente congiunzioni di elementi, traducibili come catene di congiunzioni. L’espressione di disgiunzioni o implicazioni risulta più complessa e richiede ulteriori passaggi. Un’ulteriore limitazione riguarda l’uso dei quantificatori: nei grafi relazionali tutto è implicato in modo esistenziale, mentre risulta difficile rappresentare quantificazioni universali, come nell’affermazione "tutti i ragazzi amano qualche ragazza". Anche questo tipo di espressione richiede strutture più articolate rispetto a quelle offerte dai semplici grafi relazionali.

Un'altra questione fondamentale riguarda l’arità delle relazioni: mentre i predicati logici possono avere arità arbitraria, le relazioni espresse dagli archi nei grafi relazionali sono intrinsecamente binarie. Di conseguenza, esprimere una situazione come *Regala(Mario, Libro, Maria)* – il cui significato è immediatamente comprensibile – richiede un’estensione della capacità espressiva del grafo, poiché la rappresentazione standard non è sufficiente.

Una soluzione possibile è la reificazione del predicato. In questo approccio, l’evento del regalare non viene più rappresentato come un predicato ternario, ma come un’entità autonoma. Ad esempio, si introduce un nodo *Reg32*, che rappresenta un’istanza del tipo *Regala*. A partire da questa istanza, si specificano le relazioni binarie con gli altri elementi coinvolti: *Mario* diventa l’agente dell’azione, *Maria* il destinatario e *Libro13* l’oggetto. Questo ultimo elemento può essere ulteriormente descritto come appartenente alla classe *Libro*.

Con questa struttura, si acquisisce la possibilità di distinguere tra diverse istanze dell’azione di regalare, come ad esempio *Reg32* e *Reg56*, quest’ultimo riferito a un regalo compiuto da Paolo. Questo consente di aumentare la granularità delle informazioni rappresentate e, di conseguenza, anche la potenza espressiva del grafo, avvicinandolo alla capacità di rappresentazione del calcolo dei predicati.

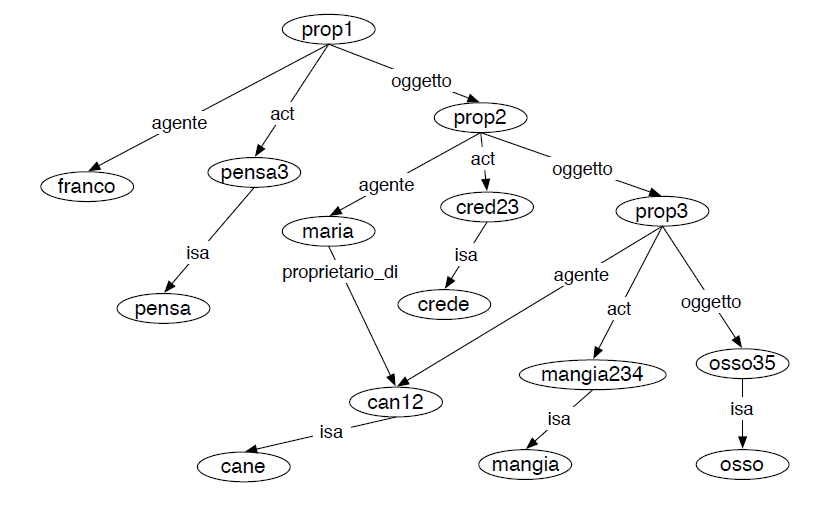


Tuttavia, con l’aumentare della complessità del grafo, la leggibilità per l’essere umano si riduce. Questo comporta un trade-off importante, soprattutto quando queste strutture vengono impiegate in fasi esplorative o in ambiti poco noti, dove la trasparenza e l’intelligibilità della rappresentazione sono essenziali.

Dal punto di vista formale, si può quindi affermare che esiste un isomorfismo tra rappresentazione grafica e logica. Tuttavia, dal punto di vista cognitivo e psicologico, le due modalità mettono in risalto aspetti differenti dell’informazione.

Un ulteriore esempio complesso riguarda la rappresentazione di frasi contenenti contesti indiretti o obliqui, ovvero enunciati che riportano credenze, ipotesi o stati mentali. Un caso emblematico è: *Franco pensa che Maria creda che il suo cane stia mangiando un osso*. Questo tipo di contenuto pone sfide particolari, poiché può essere vero o falso a seconda della correttezza o meno delle convinzioni attribuite.

Per rappresentare strutture simili si ricorre alle cosiddette reti proposizionali, un’estensione dei grafi relazionali che permette di modellare proposizioni annidate, simili all’analisi del periodo grammaticale. Alla radice della struttura si trova la proposizione principale, ossia l’atto del pensare da parte di Franco. Da essa dipendono elementi come l’agente (*Franco*) e un oggetto proposizionale composto: *Maria crede che il suo cane stia mangiando un osso*. Quest’ultima, a sua volta, contiene un altro agente (*Maria*), un atto (*credere*) e un ulteriore oggetto, che è ancora una proposizione.



A questo punto, la struttura grafica smette di essere un albero in senso stretto: compare infatti un elemento con due genitori, violando la definizione canonica di albero, secondo cui ogni nodo può avere al più un genitore. Questo è uno dei problemi noti anche nei parser linguistici, i quali incontrano difficoltà nell’associare correttamente i possessivi, come nel caso di “il suo cane”.

La frase *Franco pensa che Maria creda che il suo cane stia mangiando un osso* risulta ambigua. Se *suo* si riferisce a Maria, il significato è differente rispetto al caso in cui si riferisca a Franco. Tale ambiguità viene risolta esplicitando la specificazione *di lei*, altrimenti si rischia di avere interpretazioni divergenti. In assenza di questa specificazione, le letture potrebbero variare anche significativamente tra diversi individui, rendendo problematica una comprensione univoca del messaggio.

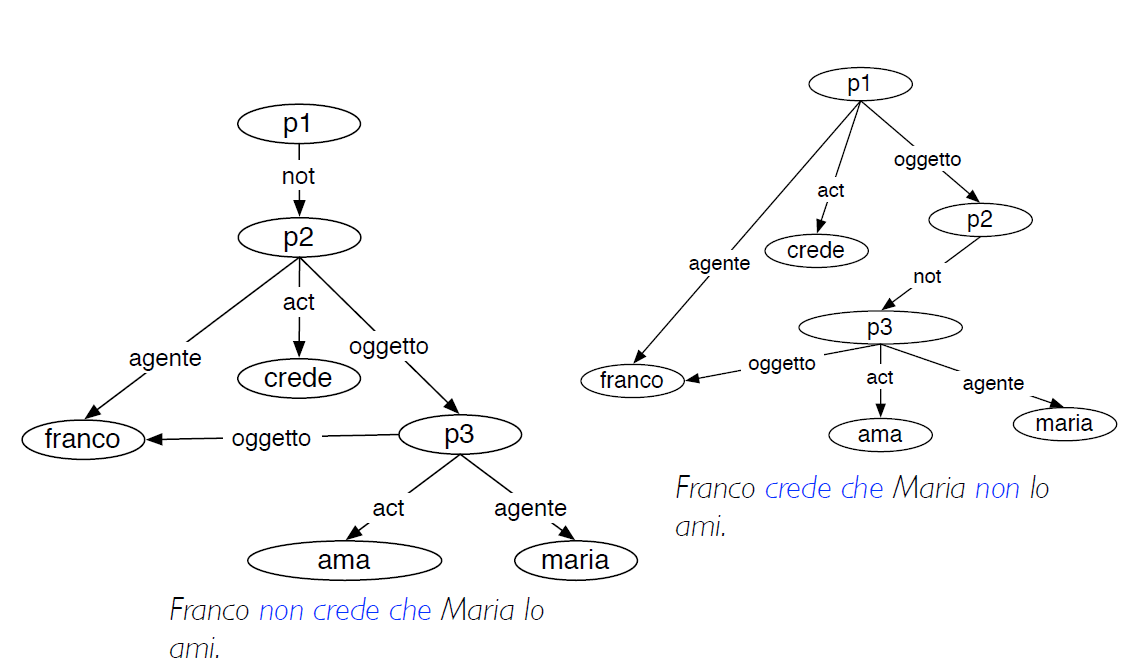
È possibile arricchire ulteriormente la rappresentazione grafica introducendo connettivi logici e contesti all'interno dei quali possano operare dei quantificatori. Tali strumenti consentono di esprimere qualsiasi enunciato formulabile nel linguaggio del calcolo dei predicati del primo ordine.

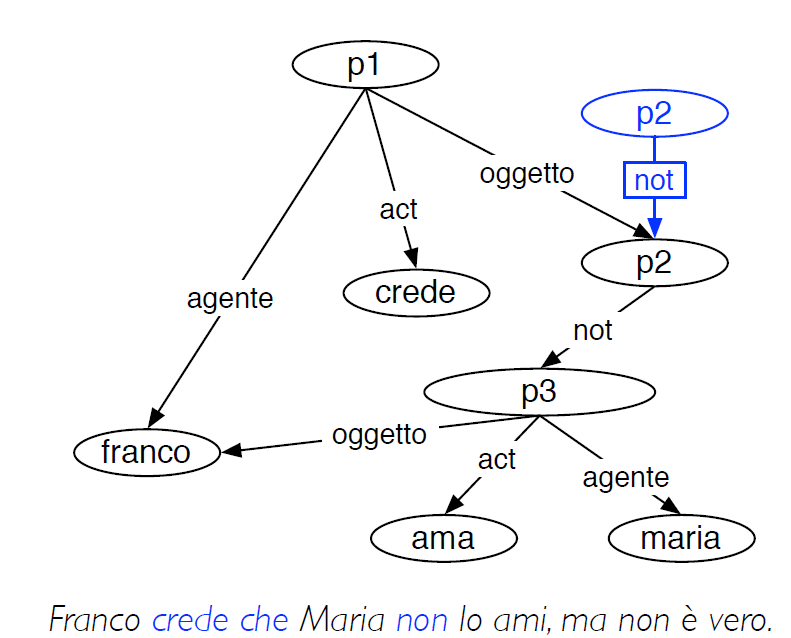
Il primo passo in questa direzione è l’introduzione della negazione. La negazione rappresenta un’evoluzione rispetto alla semplice congiunzione, perché, attraverso le leggi di De Morgan, consente di derivare anche la disgiunzione e, da lì, l’implicazione e altri connettivi logici.

La negazione di una proposizione si rappresenta creando un nodo che si configura come la negazione di un’altra proposizione. Ad esempio, un nodo PDX può essere definito come la negazione della proposizione PY. Questo meccanismo permette di distinguere chiaramente tra due enunciati simili ma semanticamente diversi: *Franco non crede che Maria lo ami* e *Franco crede che Maria non lo ami*.

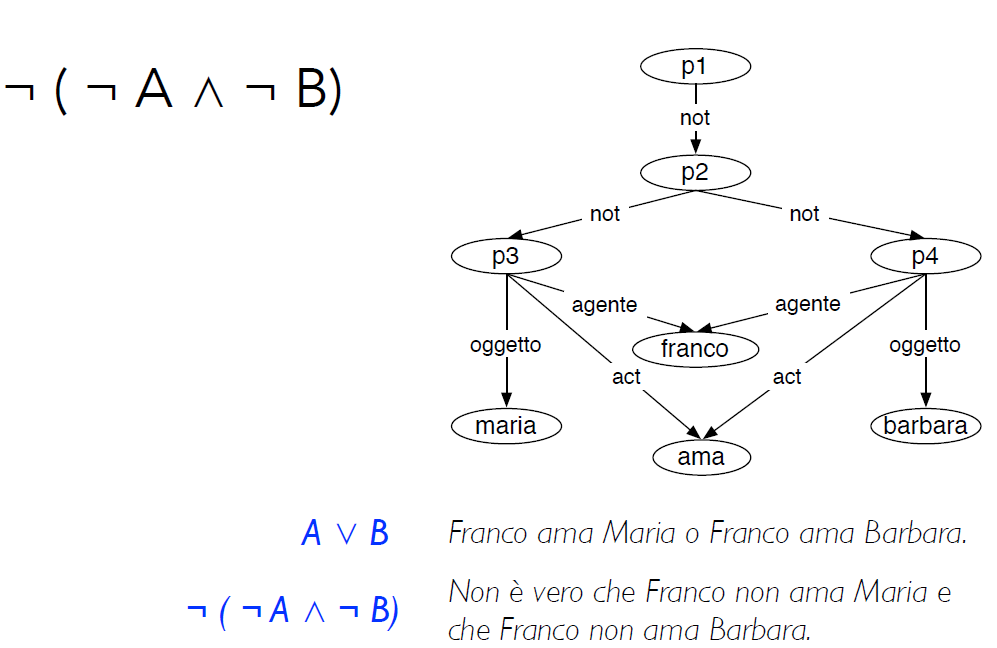
Nel primo caso, la negazione agisce a livello della proposizione principale. Il nodo P1 rappresenta l’intero enunciato “Franco crede che Maria lo ami”, mentre un nodo superiore, P0, costituisce la negazione di P1. Nel secondo caso, invece, è la proposizione interna a essere negata: Franco crede qualcosa, e ciò in cui crede è la negazione del fatto che Maria lo ami. Pertanto, il nodo P2 esprime la credenza di Franco, mentre P3 rappresenta l’enunciato *Maria ama Franco*, e la sua negazione è formalizzata come P4, subordinata a P2.

Questa struttura consente anche di rappresentare situazioni in cui il contenuto creduto non corrisponde alla realtà. Per esempio, *Franco crede che Maria lo ami, ma non è vero*, implica una negazione esterna alla struttura che rappresenta la credenza, attribuita quindi al punto di vista del narratore. La stessa logica si applica anche a frasi più complesse, come *Franco crede che Maria non lo ami, ma non è vero*, dove alla negazione interna si aggiunge una negazione esterna, che determina un doppio livello di valutazione: quello interno alla credenza e quello esterno, attribuito alla realtà.





L’aggiunta della disgiunzione costituisce il passaggio logico successivo. Come noto, secondo De Morgan, *A or B* equivale a *non (non A and non B)*. Di conseguenza, una frase come *Franco ama Maria oppure Barbara* può essere rappresentata componendo più proposizioni: due proposizioni elementari – *Franco ama Maria* e *Franco ama Barbara* – vengono prima negate individualmente, poi congiunte, e infine l’intera congiunzione è negata. Questo porta a una struttura composta da una negazione principale e da due negazioni subordinate, ognuna delle quali si applica a una proposizione elementare.



Combinando in questo modo elementi semplici, si ottengono rappresentazioni grafiche capaci di esprimere enunciati logicamente articolati. È inoltre possibile introdurre quantificatori all’interno di queste reti, raggiungendo così una capacità espressiva equivalente a quella del calcolo dei predicati.

Ne consegue che la scelta tra una rappresentazione grafica e una logico-formale non può essere basata sulla sola espressività, poiché entrambe, se opportunamente estese, raggiungono livelli comparabili. La scelta dipende piuttosto da criteri quali la leggibilità, la flessibilità e l’efficienza della rappresentazione, nonché dalla facilità con cui determinati concetti possono essere modellati.

Uno dei punti di forza dei formalismi grafici risiede nella loro naturale propensione a organizzare le informazioni in forma gerarchica. Esistono numerose evidenze a sostegno del fatto che la conoscenza umana è strutturata gerarchicamente, e ciò appare evidente anche solo osservando la maniera in cui classifichiamo gli oggetti del mondo reale: esistono classi che si organizzano in superclassi, e queste in categorie ancora più generali. Questo principio vale non solo per gli oggetti, ma anche per le azioni, che possono anch’esse essere categorizzate in gerarchie funzionali.

Si prenda come esempio il caso, spesso citato, degli elefanti che sono mammiferi. In un simile contesto, si può rappresentare un individuo, come ad esempio Clyde, mediante un nodo che lo identifica come appartenente alla classe degli elefanti. Poiché gli elefanti sono a loro volta classificati come mammiferi, inserendo Clyde nella gerarchia si può dedurre, semplicemente risalendo gli archi della struttura, che egli è un mammifero. Questo tipo di inferenza è possibile ogniqualvolta ci si trova di fronte a una relazione di sottoclasse, come nel caso di affermazioni del tipo "gli X sono Y". Tale relazione trova una rappresentazione naturale e intuitiva nelle strutture a grafo.

Sorge quindi spontanea la domanda su quali conoscenze debbano essere rappresentate esplicitamente e quali, invece, possano essere lasciate implicite e dedotte solo al bisogno. Ad esempio, la conoscenza che Clyde sia un mammifero può essere inferita nel momento in cui è necessaria, senza doverla scrivere esplicitamente nella base di conoscenza.

Si individuano due strategie opposte. Da un lato, si può optare per una rappresentazione minimale, in cui vengono specificati unicamente i legami essenziali, ovvero quelli che collegano ciascun nodo alla classe immediatamente superiore tramite relazioni *Is-A*. Questo approccio è economico dal punto di vista della memoria e della fase di costruzione della base di conoscenza, ma comporta la necessità di verificare dinamicamente le proprietà di sottoclasse tramite la ricerca di cammini nella gerarchia. Dall’altro lato, si può decidere di esplicitare tutte le relazioni possibili, adottando una strategia detta di *chiusura transitiva*, in cui ogni nodo è connesso direttamente a tutte le classi superiori pertinenti. Questa strategia semplifica e velocizza la verifica delle proprietà, ma comporta un costo elevato in termini di spazio e rende complessa la manutenzione: qualunque modifica strutturale, come il riposizionamento di una classe, impone la revisione di tutti gli archi coinvolti.

Nella pratica, si adotta un compromesso. Alcune relazioni, più frequentemente utilizzate, vengono rappresentate esplicitamente come scorciatoie (*shortcuts*), mentre altre, meno comuni, sono lasciate implicite. Questo approccio è coerente con il principio di economia rappresentativa che guida l’organizzazione della conoscenza negli esseri umani: si conserva e si elabora solo ciò che è più rilevante o frequente.

Tale principio è anche alla base del meccanismo di ereditarietà, sviluppato nei linguaggi di programmazione. In una struttura ad albero, le proprietà vengono dichiarate al livello più alto in cui esse sono valide. Per verificare, ad esempio, se una proprietà PDX è vera per un determinato nodo, è sufficiente risalire la gerarchia. La ricerca in profondità in una struttura ad albero è non solo possibile, ma anche molto efficiente, soprattutto se paragonata a meccanismi basati su regole inferenziali complesse. In questo modo, una proprietà viene asserita una sola volta e in un unico punto della gerarchia, semplificando la gestione e la manutenzione della base di conoscenza.

Questo modello consente anche di gestire eccezioni tramite un approccio procedurale. Si consideri, ad esempio, il caso degli uccelli: la maggior parte di essi vola, ma alcune specie, come lo struzzo o il pinguino, no. Se si sa che un animale è un uccello, ci si aspetta, salvo indicazioni contrarie, che sia in grado di volare. Allo stesso modo, i mammiferi sono generalmente vivipari, ma esistono eccezioni, come l’ornitorinco, che è oviparo.

Questo è lo stesso meccanismo utilizzato in linguaggi come Java, dove la ricerca di metodi o proprietà inizia nelle sottoclassi e prosegue verso l’alto, nelle superclassi. Si tratta di un algoritmo bottom-up: si parte dal nodo che rappresenta l’oggetto (ad esempio, l’ornitorinco) e si verifica se la proprietà cercata è definita specificamente a quel livello; solo in caso negativo si risale nella gerarchia, fino a trovare una definizione appropriata. Nel caso dell’ornitorinco, la proprietà relativa alla modalità di riproduzione sarà sovrascritta a livello specifico (oviparità), mentre nel caso di un elefante, tale proprietà sarà ereditata dalla classe dei mammiferi (viviparità).

Il vantaggio di questa organizzazione gerarchica risiede nella possibilità di semplificare la manutenzione, migliorare l’efficienza delle ricerche, sfruttare l’ereditarietà e gestire eccezioni attraverso meccanismi di default. Tuttavia, il passaggio da una struttura ad albero a una più generale a grafo introduce complessità significative. Nei grafi, infatti, la ricerca non ha più una complessità lineare, ma può crescere esponenzialmente con la dimensione della base di conoscenza. Inoltre, a differenza delle ricerche su spazi di stato, qui non si possono applicare euristiche efficaci per guidare l’esplorazione, e il risultato di una verifica dipende dalla strategia adottata. Una ricerca in profondità restituisce il primo risultato trovato lungo un cammino, mentre una ricerca in ampiezza può portare a un risultato diverso. Le strategie di esplorazione, pertanto, influenzano direttamente il comportamento e le risposte del sistema.

Si consideri il seguente esempio: gli elefanti sono una specie in via d’estinzione, e le specie in via d’estinzione sono oggetto di studio da parte dei naturalisti. A partire da queste premesse, si potrebbe erroneamente concludere che Clyde, un individuo elefante, sia egli stesso una specie in via d’estinzione, e che quindi sia studiato con interesse dai naturalisti. Questa inferenza, tuttavia, si rivela problematica se si osserva con attenzione la struttura semantica della rappresentazione usata.

Il problema nasce dal fatto che, nella costruzione della rappresentazione, si è operata una semplificazione che appiattisce e confonde due concetti distinti: la relazione di **appartenenza**, che intercorre tra un individuo e una classe, e la relazione di **inclusione**, che intercorre tra due classi. Entrambe vengono trattate tramite la medesima relazione *ISA*, che così viene a designare due cose profondamente diverse. Per raffinare l’analisi, si può allora distinguere formalmente tra questi due tipi di relazione: nel caso dell’individuo e della classe, si parla di appartenenza; nel caso di due classi, di inclusione.

Questa confusione mette in luce un limite originario dei formalismi grafici, ovvero la loro **assenza di una semantica formale**. In tali formalismi, infatti, il significato della rappresentazione non è implicito nella struttura stessa della rete, ma è determinato dalle procedure che la manipolano: si parla in questo caso di **semantica procedurale**. Invece, nelle logiche formali, ogni formula ha un significato ben definito indipendentemente dal modo in cui viene elaborata.

L’ambiguità interpretativa delle reti grafiche, come nel caso della doppia valenza della relazione ISA, può dunque portare a inferenze errate, come l’attribuzione a un singolo individuo di proprietà che si applicano in realtà solo alla classe cui egli appartiene. Per prevenire tali distorsioni, è necessario adottare una semantica più rigorosa e distinguere in modo netto tra tipi diversi di relazioni.

Uno dei formalismi grafici che ha cercato di superare alcuni di questi limiti è quello dei **frame**, introdotto da Marvin Minsky negli anni Settanta. Minsky, uno dei pionieri dell’intelligenza artificiale, si occupava in quel periodo di teoria della visione artificiale. La sua intuizione fu che, nel riconoscimento visivo, le persone non analizzano ogni dettaglio dell’ambiente, ma si basano su schemi, su moduli cognitivi predefiniti, che consentono di orientare l’attenzione verso aspetti rilevanti della scena. Questi moduli, denominati appunto *frame*, rappresentano degli scenari concettuali che guidano l’interpretazione.

Tale concetto può essere esteso anche all’ambito linguistico e cognitivo generale. Quando un individuo si trova in una situazione nuova, non parte mai da zero: attinge al proprio bagaglio esperienziale, che gli fornisce delle aspettative strutturate. Per esempio, se entra per la prima volta in un’aula universitaria, pur non conoscendone la disposizione esatta, si aspetta di trovare sedie, banchi, una cattedra, e una persona che tiene una lezione. Questo schema rappresenta il *frame* della lezione.

Il frame, dunque, è uno **scenario semilavorato**, una struttura stereotipata che può essere adattata ai dettagli specifici del contesto. Permette di organizzare la conoscenza in modo che sia facilmente recuperabile e utilizzabile per interpretare la realtà e orientare il comportamento.

Strutturalmente, un frame può essere immaginato come un oggetto composto da **slot**: ogni slot contiene informazioni tipiche di quello scenario. Tali informazioni possono essere valori fissi, ma anche regole, vincoli, o riferimenti ad altri frame. Il frame funge quindi da cornice concettuale che consente di interpretare i nuovi dati alla luce dell’esperienza precedente. il frame consente di ridurre l’ambiguità, orientando l’interpretazione secondo modelli noti. Risolvere ambiguità nell’interpretazione di una scena significa riuscire a determinare il significato più plausibile di ciò che si percepisce, tenendo conto del contesto in cui l’osservazione avviene. Si immagini, ad esempio, di vedere un frammento di oggetto simile all’impugnatura di una pinza. Se ci si trova in un’officina meccanica, dove l’ambiente è caratterizzato da attrezzi, stracci sporchi di grasso e rumori metallici, sarà naturale riconoscere quell’oggetto come una pinza, anche se solo parzialmente visibile. Lo stesso oggetto, però, se osservato sotto un tovagliolo su una tavola elegantemente apparecchiata con lino pregiato, sarà interpretato come uno strumento da tavola, come uno schiaccianoci, poiché l’ambiente suggerisce un contesto completamente diverso.

Questa capacità di attribuire un significato diverso allo stesso stimolo percettivo a seconda del contesto è resa possibile dall’attivazione di un *frame*, ovvero di uno scenario cognitivo che guida il riconoscimento e l’interpretazione. Il frame regola la relazione dell’individuo con l’ambiente circostante, consentendogli di “acclimatarsi” e di contestualizzare rapidamente ciò che percepisce, sulla base dell’esperienza e delle informazioni sensorie disponibili. Questo processo non si applica solo alla percezione visiva, ma può essere esteso anche all’interpretazione linguistica.

Il frame consente di organizzare la conoscenza in un dominio specifico e di semplificare l’interpretazione, poiché l’individuo non procede da zero, ma attiva pattern già consolidati nella memoria. Oltre a essere una struttura che raccoglie conoscenze, il frame include anche le **procedure** operative necessarie per accedere e manipolare tali conoscenze. Tali procedure permettono di condurre inferenze a partire dai dati contenuti nel frame stesso.

Prendendo in considerazione due brevi sequenze linguistiche, si osserva questa dinamica:

**Sequenza A**: *Giacomo andò al ristorante, chiese al cameriere una bistecca con patatine, pagò il conto e uscì.*

**Sequenza B**: *Giacomo andò al parco, chiese al nano un topolino, prese la scatola e se ne andò.*

Dal punto di vista sintattico e semantico superficiale, le due frasi sono analoghe: entrambe hanno una struttura regolare e i loro elementi sono di per sé comprensibili. Tuttavia, la seconda sequenza risulta poco interpretabile, perché manca un frame coerente in grado di dare senso globale all’interazione descritta. La prima frase, invece, attiva il frame del ristorante, che integra e rende coerenti tutte le azioni narrative. L’assenza di un frame nella seconda sequenza impedisce l’unificazione interpretativa.

Questa osservazione rafforza l’idea che i frame agiscano come **principi cognitivi di alto livello**, capaci di integrare e organizzare i significati delle frasi, fornendo un contesto globale di comprensione. Questo si ricollega a quanto detto sui formalismi grafici: questi ultimi permettono di rappresentare in modo dichiarativo conoscenze sul mondo, ma sono privi di una semantica formale autonoma. Il frame, inteso come oggetto contenente blocchi di conoscenza, assume significato solo se accompagnato da procedure che ne guidano l’uso. Senza queste, il frame resta una struttura vuota, non operativa.

Nello stesso periodo in cui Minsky elaborava la teoria dei frame, un’altra figura di spicco nel panorama dell’intelligenza artificiale e della psicologia cognitiva, Eleanor Rosch, introduceva la teoria dei **prototipi**. È importante distinguere i frame dai prototipi, sebbene entrambi siano modelli cognitivi che organizzano la conoscenza.

All’interno di queste categorie, la teoria del prototipo propone che l’appartenenza a una categoria non dipenda più da un insieme rigido di attributi necessari e sufficienti, come avviene nella rappresentazione logica tradizionale (ad esempio, nella definizione di triangolo), ma piuttosto da una **somiglianza con un esemplare tipico**, ovvero il prototipo. In questo modo, la categorizzazione diventa più flessibile, e riflette più fedelmente il funzionamento effettivo della cognizione umana. Un *passero* è miglior esemplare della categoria *uccello* rispetto ad un *airone*, che è miglior esemplare rispetto allo *struzzo*.

Queste idee della Rosch sono state recepite nei linguaggi di rappresentazione dei *frame*: tutti i sistemi a *frame* permettono di ragionare intorno a classi di oggetti usando delle rappresentazioni prototipiche che, valide in linea di massima, hanno bisogno di venire adattate e modificate per tener conto della complessità del mondo.

I sistemi a *frame* organizzano le conoscenze in strutture gerarchiche i cui elementi sono collegati fra loro da espressioni di tipo *isA* o *ako* (*a kind of*) che consentono la trasmissione ereditaria delle proprietà. Le proprietà dei *frame* ad alto livello restano fisse (rappresentando fatti veri per la classe), mentre i livelli più bassi, sottoclassi o istanze individuali, possono essere contraddistinti da proprietà specifiche, anche in contrasto con quelle delle superclassi.

In informatica, le ontologie sono una concettualizzazione formale del mondo. Di solito definiscono relazioni gerarchiche e laterali tra i concetti o le classi rappresentate. Un'ontologia formale specifica un insieme di vincoli che dichiarano ciò che deve necessariamente valere in un modello.

**Lezione 3**

La semantica lessicale è la disciplina che si occupa di stabilire cosa significhino le parole. Analizza il contributo delle parole al significato della frase, e di come il contesto delle parole influenzi il loro significato.

Ci sono due principali problemi:

- Polisemia

• Ho acquistato un *sacco* (uno *zaino*)

• Si è infilato in un *sacco di guai* (*molti* guai)

- Semantica frasale

• Il ladro ha *vuotato il sacco*

• Anche con espressioni meno idiomatiche: '*giacca a vento'*, *'mulino a vento'*

Il contesto è l'insieme degli elementi adiacenti a una parola. Può essere di natura semantica o sintattica, ma anche situazionale o pragmatica.  
Il contesto sintattico è costituito dagli elementi che circondano una parola, considerati dal punto di vista delle loro proprietà sintattiche. Può trattarsi, ad esempio, di un contesto nominale, verbale, aggettivale, ecc.  
Il contesto semantico riguarda invece gli elementi circostanti dal punto di vista del loro significato: si pensi alla differenza tra “saltare un fosso” e “saltare un pasto”.

Il contesto può essere:

* **linguistico**, cioè sintattico o semantico;
* **situazionale**, detto anche pragmatico o extralinguistico.

Nel caso dell'enunciato “ho mangiato un sacco”, il significato del termine “sacco” risulta chiaro grazie agli elementi linguistici che lo accompagnano. Al contrario, in “il tuo amico è forte”, il termine “forte” può riferirsi a caratteristiche fisiche o psicologiche, e il suo significato si chiarisce solo nel contesto specifico in cui viene usato.

L’ambiguità è la proprietà di una forma lessicale di possedere più di un significato.  
Si distingue tra:

* **ambiguità contrastiva** (o omonimia): si verifica quando i diversi significati sono tra loro contraddittori e incompatibili nello stesso contesto, come in “miglio” inteso come pianta e “miglio” inteso come unità di misura;
* **ambiguità complementare** (o polisemia): si verifica quando i significati, pur diversi, derivano da un’unica origine e possono emergere in contesti diversi, ad esempio “massaggiarsi il collo” e “maglione a collo alto”.

La polisemia risponde al principio di economicità linguistica: si utilizzano le stesse parole per esprimere più significati, riducendo le dimensioni del lessico. Tuttavia, ciò comporta un costo cognitivo maggiore nell’apprendimento e nell’uso.  
I verbi tendono a essere più polisemici dei sostantivi, probabilmente perché il loro significato viene completato dagli argomenti con cui si combinano. Un esempio è il verbo “aprire”, che può riferirsi ad azioni molto diverse: aprire una finestra, una bottiglia, un conto in banca, il gas, un’attività commerciale, un dibattito.

Teoria referenziale del significato

Secondo questa teoria, le parole servono a fare riferimento a elementi della realtà (Frege, 1862). Ad esempio, “sedia” fa riferimento a un oggetto, mentre “acquistare” a un evento.

Il significato consiste nella capacità della parola di stabilire una relazione con un elemento esterno al linguaggio. Tuttavia, espressioni diverse possono avere lo stesso referente ma significati differenti, come “la donna con cui vivo” e “mia moglie”.

Si identificano le seguenti teorie sulla natura del significato:

1. **Denotazione e designazione**

Il riferimento può essere:

* denotativo, come in “questa sera si mangia pesce”, dove si intende la classe degli oggetti tipo “pesce”;
* designativo, come in “il pesce della cena di ieri sera era troppo salato”, dove si fa riferimento a un elemento specifico di quella classe.

In un’interpretazione ampia, ogni elemento linguistico instaura un riferimento con la realtà.

In un’interpretazione più restrittiva (Searle, 1969), si distinguono due atti: l’atto di riferimento (identificare entità) e quello di predicazione (attribuire proprietà). Ad esempio, in “Luca è ubriaco”, “Luca” rappresenta il riferimento, mentre “è ubriaco” è la predicazione. Le espressioni predicative non hanno un referente proprio.

1. **Teoria mentalista o concettuale**  
   Questa teoria arricchisce la teoria referenziale sostenendo che il riferimento tra parole e realtà è mediato da rappresentazioni mentali, chiamate concetti.  
   Quando si parla di “sedia”, non si fa riferimento diretto all’oggetto, ma alla rappresentazione mentale che si ha di esso. Il significato deriva dalla concettualizzazione della realtà, non dalla realtà in sé.

Questa teoria permette di spiegare come si possa parlare di:

* entità astratte, immaginarie o ipotetiche;
* lo stesso evento in modi diversi (es. “Luca ha rotto il vetro” vs “Il vetro si è rotto”).

La semantica cognitiva, basata su questa teoria, pone l’accento sulla categorizzazione dell’esperienza e sull’organizzazione dei concetti, concepiti come entità psicologiche. I concetti sono legati all’esperienza sensoriale e percettiva.  
I concetti cognitivi sono instabili e variano da individuo a individuo o tra culture. Appartengono alla struttura mentale.  
I concetti lessicalizzati, invece, sono condivisi socialmente e appartengono alla struttura linguistica. Essendo legati alla lingua, possono variare da una lingua all’altra.

1. **Teoria strutturale**  
   In questa visione, il significato non dipende da un riferimento esterno o da un’immagine mentale, ma dal valore che una parola assume rispetto alle altre parole dello stesso campo semantico.  
   Il termine “sedia”, ad esempio, acquisisce il proprio significato in relazione a parole come “sgabello”, “sedile”, “trono”, “poltrona”, “panca”.  
   Il significato è quindi il contenuto informativo che emerge per esclusione all’interno del campo semantico.
2. **Teoria dei prototipi**

Evoluzione della teoria mentalista, considera il prototipo come l’esemplare più rappresentativo di una categoria.  
Quando si categorizza un oggetto come “tazza” piuttosto che “bicchiere”, si fa riferimento a un prototipo.  
Gli oggetti appartenenti a una categoria condividono somiglianze salienti, ma non necessariamente tutti i tratti.  
A differenza della teoria classica, che prevede confini netti e condizioni necessarie e sufficienti, la teoria dei prototipi ammette una struttura interna alla categoria, con membri centrali e periferici.

1. **Teoria distribuzionale**  
   Fondata sull’ipotesi di Zellig Harris (1956), sostiene che il significato delle parole è determinato dal contesto di co-occorrenza.  
   Due parole sono tanto più simili quanto più simili sono i contesti in cui compaiono.

L’avvento dei grandi corpora e degli strumenti computazionali ha rilanciato questa teoria, introducendo la metafora geometrica del significato: i significati corrispondono a punti in uno spazio multidimensionale. Parole vicine nello spazio hanno significati simili. Questo ha portato allo sviluppo delle **rappresentazioni vettoriali**.

La semantica cerca di spiegare il significato delle parole (semantica lessicale) e come si compone il significato delle frasi (semantica frasale). In quest’ambito, due problemi fondamentali sono:

* la **contestualità** del significato: le parole assumono significati diversi in contesti diversi;
* la **polisemia**

Ci sono due approcci al significato:

1. **Enumerazione dei sensi**: ogni parola è associata a più significati, ciascuno con le proprie restrizioni contestuali.
2. **Concezione dinamica del significato**: il significato è fluido e interagisce con quello delle parole vicine. Le parole sono entità permeabili.

**Principi di interazione semantica (Pustejovsky, 1995)**

1. **Co-composizione**  
   Il significato di un verbo è completato dai suoi argomenti:

* “Luca ha tagliato il pane” (affettare),
* “Luca ha tagliato l’erba” (falciare),
* “Luca si è tagliato un dito” (ferirsi),
* “Luca ha tagliato i capelli” (accorciare).

Il verbo ha un significato base, che viene specificato dal complemento.

1. **Forzatura o conversione di tipo (coercion)**  
   Verbi come “iniziare”, “smettere”, “durare” richiedono eventi come oggetti:

* “iniziare un nuovo articolo” implica iniziare a scrivere o a leggere.
* Il nome, pur indicando un oggetto, assume un’interpretazione eventiva.

1. **Legamento selettivo**  
   Avviene nei gruppi nome-aggettivo:

* “buon coltello”, “buon medico”, “buon libro” → l’aggettivo seleziona lo scopo o la funzione del nome.
* In altri casi, modifica proprietà fisiche: “vestito troppo lungo”.

WordNet è un sistema di riferimento lessicale online, il cui design si ispira a teorie psicologiche sul funzionamento della memoria lessicale umana. Al suo interno, i sostantivi, i verbi e gli aggettivi della lingua inglese sono organizzati in insiemi di sinonimi, ognuno dei quali rappresenta un concetto lessicale di base. Questi insiemi di sinonimi sono collegati tra loro da diverse relazioni semantiche.

Il problema che WordNet intende affrontare nasce dal fatto che i procedimenti alfabetici standard, utilizzati nei dizionari tradizionali, organizzano l'informazione lessicale accorpando parole che si scrivono allo stesso modo, ma disperdono in modo casuale parole dal significato simile o correlato all'interno dell’elenco. Non esiste un'alternativa ovvia o una modalità semplice e intuitiva che consenta agli utenti di trovare facilmente il termine desiderato, se non quella di scorrere alfabeticamente l’intero dizionario.

Sebbene i dizionari digitali permettano di cercare molto più rapidamente rispetto alla consultazione manuale, rivelano presto un limite sostanziale: l’uso dei computer come semplici “sfogliatori rapidi” si rivela inefficiente, non sfruttando appieno le loro potenzialità computazionali.

Nel 1985, un gruppo di psicologi e linguisti dell’Università di Princeton intraprese lo sviluppo di un database lessicale. L’idea iniziale era quella di creare uno strumento in grado di aiutare la consultazione dei dizionari non solo in modo alfabetico, ma anche concettuale. Il progetto era pensato per essere utilizzato in stretta connessione con un dizionario online di tipo convenzionale. Tuttavia, con l’avanzare dei lavori, emerse la necessità di formulare principi e obiettivi più ambiziosi, dando al progetto una struttura autonoma.

Una delle differenze più evidenti tra WordNet e un dizionario tradizionale è che WordNet suddivide il lessico in quattro categorie: sostantivi, verbi, aggettivi e avverbi. Lo scopo del progetto è quello di organizzare l’informazione lessicale in base ai significati delle parole. I sostantivi, ad esempio, sono organizzati come gerarchie nella memoria lessicale; i verbi, invece, sono strutturati attraverso una varietà di relazioni di implicazione logica; aggettivi e avverbi vengono sistemati all'interno di spazi multidimensionali.

Ciascuna di queste strutture lessicali riflette una modalità diversa di categorizzare l’esperienza umana. Cercare di imporre un unico principio organizzativo a tutte le categorie sintattiche significherebbe rappresentare in modo scorretto la complessità psicologica della conoscenza lessicale.

La semantica lessicale parte dal riconoscimento che una parola è un’associazione convenzionale tra un concetto lessicalizzato e un’unità fonica o grafica che svolge un ruolo sintattico. Poiché il termine “parola” viene comunemente utilizzato sia per indicare l’enunciato fisico sia per riferirsi al concetto associato, le discussioni su questa associazione lessicale rischiano spesso di generare ambiguità terminologiche. Per ridurre tale ambiguità, in questo contesto si preferisce usare “forma della parola” per indicare l’enunciato o l’iscrizione fisica, e “significato della parola” per riferirsi al concetto lessicalizzato che una forma può esprimere.

Si immagina dunque una matrice lessicale, in cui le forme delle parole costituiscono le intestazioni delle colonne e i significati delle parole quelle delle righe. Un’entrata in una cella della matrice indica che, in un contesto appropriato, la forma indicata nella colonna può essere utilizzata per esprimere il significato indicato nella riga corrispondente.

Immagine che contiene testo, schermata, numero, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Ad esempio, l’entrata E1,1 segnala che la forma F1 può essere usata per esprimere il significato M1. Se ci sono due entrate nella stessa colonna, significa che quella forma è polisemica, cioè può esprimere più significati. Se, invece, ci sono due entrate nella stessa riga, vuol dire che le due forme corrispondenti sono sinonimi, in un determinato contesto.

Le corrispondenze tra forme e significati sono molte-a-molte: alcune forme hanno più significati e alcuni significati possono essere espressi da più forme diverse. I problemi complessi della lessicografia, come la polisemia e la sinonimia, possono essere considerati aspetti complementari di questa mappatura. La polisemia e la sinonimia emergono, infatti, nel momento in cui si accede all’informazione contenuta nel lessico mentale: chi legge e riconosce una forma deve affrontarne la polisemia, mentre chi parla o scrive, dovendo esprimere un significato, deve scegliere tra più sinonimi. Il significato della parola M1 può essere rappresentato semplicemente elencando le forme lessicali che possono essere utilizzate per esprimerlo: {F1, F2, …}. Ad esempio, una persona che sa che la parola *board* può indicare sia un pezzo di legno sia un gruppo di persone riunite per uno scopo specifico sarà in grado di distinguere il significato inteso anche solo grazie ad altre parole come *plank* o *committee*.

I gruppi di sinonimi {board, plank} e {board, committee} possono quindi fungere da designatori univoci per rappresentare in modo chiaro e distinto i due diversi significati della parola *board*.

Proseguendo lungo la linea già tracciata, si può fare un ulteriore passo verso la nozione di *Synset*. L’intuizione che ha dato origine al concetto di Synset è tanto semplice quanto efficace: di fronte all’impossibilità di cogliere in modo esaustivo e oggettivo il significato dei termini, tenendo conto anche della variabilità soggettiva che ciascuno associa ai diversi sensi, si è deciso di raggruppare insieme termini sinonimi. Questo procedimento permette, ad esempio, di disambiguare un termine come *board*, fortemente polisemico, semplicemente accostandolo a un altro termine che ne condivide il significato. Se si dice *board* e *plank*, il riferimento è chiaramente all’ambito delle costruzioni; se invece si dice *board* e *committee*, diventa evidente che si intende il comitato. Questa è l’essenza del Synset: gruppi di termini sinonimi che, insieme, costituiscono una sorta di presidio lessicale. Non si tenta nemmeno di rappresentare esplicitamente il significato vero e proprio del termine, perché si riconosce la complessità intrinseca del concetto. Piuttosto, si raggruppano termini in insiemi di sinonimi, costruendo così una rete dove i nodi sono appunto i Synset e i termini vengono organizzati in base al significato sottostante.

Questa operazione, per quanto possa sembrare semplice, ha avuto conseguenze importanti, dando impulso a discipline come la semantica lessicale. Dato che il progetto WordNet è nato in ambito anglofono e per un pubblico anglofono, i suoi sviluppatori non si sono preoccupati di fornire definizioni complete dei concetti, assumendo che gli utenti avessero già una comprensione pregressa dei significati. Questo può rendere l’utilizzo della risorsa meno immediato per chi apprende l’inglese come lingua seconda, ma l’idea di base è che chi consulta WordNet sia in grado di cogliere il significato dal contesto, soprattutto quando i termini sono inseriti in combinazioni come *board–plank* o *board–committee*.

All’interno di WordNet sono presenti brevi descrizioni – le cosiddette *glosse* – che non hanno lo scopo di spiegare il significato del concetto, ma solo di differenziarlo da altri. Gli autori parlano infatti di *differential purposes*, intendendo che queste glosse servono a distinguere tra i sensi dei vari Synset. Oltre a queste descrizioni, sono spesso forniti anche degli esempi d’uso, che oggi sono ampiamente sfruttati in ambito educativo e linguistico, proprio perché, come nei dizionari tradizionali, offrono un contesto utile a comprendere meglio il significato dei termini. Circa il 30% dei Synset è accompagnato da esempi d’uso, mentre tutti includono almeno una glossa.

Questi Synset costituiscono i nodi di una rete che, idealmente, abbraccia l’intero lessico di una lingua. Ma se ci sono nodi, ci devono essere anche archi: WordNet è strutturato come un grafo, e quindi comprende anche relazioni – tipicamente binarie – che collegano i vari Synset. Le principali di queste relazioni sono semantiche e, per la maggior parte, simmetriche.

La prima relazione, e forse la più importante, è quella di *sinonimia*: tutti i termini che risultano sinonimi vengono inclusi nello stesso Synset. Va però sottolineato che la sinonimia è un concetto tutt’altro che semplice da definire con precisione. In senso stretto, due termini sono sinonimi solo se possono essere scambiati in qualunque contesto senza alterare il valore di verità dell’enunciato. Si tratta, evidentemente, di una definizione molto rigorosa. Per questo motivo, i creatori di WordNet hanno introdotto una versione più "rilassata" del concetto di sinonimia, basata su un contesto linguistico specifico, indicato con *C*. Due espressioni sono quindi considerate sinonime in un determinato contesto *C* se la loro sostituzione reciproca in quel contesto non altera il valore di verità dell’enunciato. Ad esempio, all’interno di un discorso sull’edilizia, parlare di *plank* o di *board* è pressoché equivalente: si tratta di sinonimi accettabili nello stesso contesto.

Un’altra relazione importante all’interno della rete è quella di *antonimia*, ossia l’opposizione di significato: termini come *ricco–povero*, *alto–basso* e così via. Anche questa relazione presenta delle criticità, perché termini apparentemente opposti non sempre sono autentici antonimi. Non essere ricco, ad esempio, non implica automaticamente essere povero, e lo stesso vale per l’opposizione *alto–basso*. Queste sfumature rendono l’implementazione di relazioni antonimiche una sfida, ma anche una componente fondamentale per costruire una risorsa linguistica complessa e ricca come WordNet. Le relazioni tra i termini, come si è visto, possono essere complesse e talvolta difficili da definire in modo netto. Esistono dei test che si possono condurre per verificare se certe coppie di parole possono essere considerate autentici antonimi. In pratica, si intervistano dei parlanti nativi della lingua per vedere se accettano determinate espressioni come linguisticamente corrette e naturali.

Ad esempio, si è osservato che coppie come *rise and fall* e *ascend and descend*, considerate entrambe sinonime all’interno delle rispettive coppie, non risultano più accettabili se vengono “mescolate”: espressioni come *rise and descend* oppure *ascend and fall* non suonano corrette. Questo suggerisce che, nel valutare l’antonimia, non ci si basa soltanto sul significato, ma entrano in gioco anche altri elementi, come l’origine etimologica o la struttura morfologica delle parole.

Nel caso di *ascend* e *descend*, ad esempio, si tratta di termini di origine latina che portano in sé un prefisso riconoscibile (*a-* e *de-*) che contribuisce alla percezione dell’opposizione. Per un parlante nativo, questa opposizione è immediatamente comprensibile. Al contrario, quando si combina un termine come *rise* – di origine germanica e senza un prefisso corrispondente – con *descend*, l’effetto di antonimia si perde. Ciò fa pensare che la relazione di antonimia sia meno una questione strettamente semantica e più un fatto di natura lessicale, legato alla forma e alla collocazione abituale dei termini nel linguaggio.

Un altro tipo di relazione fondamentale in WordNet è l’iponimia, ovvero la relazione tra un termine più specifico (iponimo) e uno più generico (iperonimo). Anche qui, il metodo utilizzato è stato quello di sottoporre frasi a dei parlanti nativi – in questo caso, studenti universitari – chiedendo loro se un’espressione del tipo “x è un tipo di y” suonasse naturale. Se la frase veniva accettata come plausibile, allora la relazione veniva registrata nella struttura di WordNet; in caso contrario, veniva scartata. Questa intuizione, che oggi può sembrare ovvia, non era affatto scontata negli anni Ottanta, almeno non in ambito linguistico. Secondo questo modello, un iponimo eredita tutte le caratteristiche (le *feature*) del concetto più generale da cui dipende, ma aggiunge o modifica almeno una caratteristica specifica che lo distingue all’interno della classe. Il sistema funziona, quindi, come una sorta di gerarchia: il puntatore va “verso l’alto” nella rete, verso il termine più generico, e ogni livello aggiunge un grado di specificità.

Accanto alla sinonimia, antonimia e iponimia, esiste anche la *meronimia*, un’altra relazione semantica importante. La meronimia descrive i rapporti tra un tutto e le sue parti. È una relazione che può essere formulata con espressioni come “un *y* ha un *x* come sua parte” oppure “*x* è una parte di *y*”. Come nel caso delle altre relazioni, anche qui si è fatto ricorso al giudizio dei parlanti per determinare quali combinazioni risultassero accettabili. La meronimia è una relazione che può essere considerata transitiva e simmetrica, ma presenta comunque delle complessità nella sua applicazione. Per esempio, non tutte le parti di un oggetto si considerano meronimicamente rilevanti, e ci sono ambiguità nella categorizzazione. Questo rende più difficile rappresentare tali relazioni in modo sistematico.

Nel complesso, ciò che ne risulta è una rete lessicale molto articolata, che può essere vista come un insieme di reti distinte, a seconda della relazione che si considera. A seconda che si scelga di esaminare i Synset in base alla sinonimia, all’iponimia, all’antonimia o alla meronimia, si ottengono strutture diverse. E proprio questa pluralità di reti è uno degli aspetti più interessanti del modello: ogni tipo di relazione apre una diversa prospettiva sul lessico e ne rivela una struttura specifica. Uno degli aspetti affascinanti, ad esempio, è la dimensione *regionale*, cioè come certi termini o concetti possano essere organizzati in insiemi coerenti sulla base di determinati criteri semantici o contestuali.

Conoscere la posizione di un determinato synset all’interno della rete semantica di WordNet rappresenta una componente significativa della conoscenza che si ha su quell’elemento. Si è visto che le definizioni, nella struttura di WordNet, fanno riferimento all’elemento immediatamente superordinato e poi introducono le cosiddette *distinguishing features*, ovvero le caratteristiche distintive che precisano il significato del termine in questione. In questo sistema, i concetti sono caratterizzati da tre elementi principali: gli *attributi*, che rappresentano le proprietà utilizzate per esprimere diverse forme di modificazione (ad esempio aggettivi o altri modificatori); le *parti*, che corrispondono ai meronimi, ovvero gli elementi che compongono un intero; e infine le *funzioni*, che possono essere intese come le forme di predicazione, ossia le azioni o i ruoli associati a un dato concetto.

Un esempio emblematico è dato dal termine *albero*. Se si considera la definizione fornita da un dizionario tradizionale italiano, come potrebbe essere il Sabatini-Coletti o quello del Corriere della Sera, si nota che *albero* viene definito inizialmente come una “pianta perenne con fusto legnoso ramificato”. Da lì si procede con altre accezioni, più generiche o simboliche – come *albero di Natale* o *albero della cuccagna* – fino a quelle più tecniche o settoriali, come l’uso del termine in meccanica (*albero di trasmissione*) o nelle telecomunicazioni (*albero come antenna*).

Se si passa a un dizionario inglese, si può osservare che la struttura delle definizioni è simile: la voce *tree* viene descritta come una “pianta perenne, tipicamente dotata di un unico tronco, che raggiunge altezze considerevoli”. Anche qui, si parte dall’elemento superordinato – la *pianta* – e si aggiungono tratti distintivi. Tuttavia, in questa descrizione manca la menzione del significato tecnico di *tree* in ambito teorico o informatico, come nella teoria dei grafi, dove il termine assume un significato completamente diverso. Questo dimostra come anche nei dizionari più completi vi siano limiti nella rappresentazione della polisemìa.

Un ulteriore problema è che, quando si fa riferimento a un concetto superordinato come *pianta*, il termine stesso spesso non viene definito in modo esaustivo. Per comprendere pienamente cosa sia un albero, si è costretti ad andare a cercare la definizione di *pianta*, ma anche in quel caso ci si imbatte in un’altra ambiguità: il termine *plant*, in inglese, è notoriamente polisemico e può riferirsi tanto a un organismo vegetale quanto a un impianto industriale. Questo comporta che, nel tentativo di comprendere un termine tramite i suoi superordinati, ci si trovi continuamente di fronte a scelte interpretative ambigue. Ogni passaggio nella rete semantica implica la possibilità di imboccare direzioni sbagliate, dovute alla molteplicità di sensi associati a un singolo termine.

In sostanza, anche se le definizioni in WordNet (così come nei dizionari tradizionali) si fondano su una struttura apparentemente rigorosa – basata su relazioni di tipo iperonimico e su tratti distintivi – rimangono comunque esposte ai limiti della polisemìa e dell’incompletezza definitoria. Questo rende necessaria una costante attenzione interpretativa da parte dell’utente, che deve saper navigare tra i molteplici sensi possibili e riconoscere quello rilevante nel contesto di interesse.

Un aspetto significativo che manca nelle tradizionali rappresentazioni lessicali, come i dizionari, è l'indicazione esplicita dei termini coordinati, ovvero dei “fratelli” di un certo concetto all’interno della stessa gerarchia. Se si immagina la struttura concettuale come un albero genealogico, per così dire, conoscere quali sono i termini che condividono lo stesso “genitore” semantico del termine *albero* – ad esempio altri tipi di *plant* – sarebbe fondamentale per comprendere appieno la posizione dell’elemento nella rete del significato.

Nel caso in cui si identifichi correttamente *albero* come un tipo di *pianta* nel senso naturale, sorge spontanea la domanda: quali sono gli altri elementi che, come *albero*, discendono da *pianta*? In un dizionario tradizionale, questo tipo di informazione non è accessibile in modo diretto, e il ricercatore deve affidarsi alla propria pazienza e dedizione, passando in rassegna le voci per rintracciare altri concetti con lo stesso superordinato.

A ciò si aggiunge l’assenza di riferimenti sistematici agli iponimi, ovvero agli elementi subordinati. Una volta identificato il termine superordinato – come *plant* – e magari anche i suoi fratelli (coordinati), resta ancora da esplorare la rete dei suoi discendenti: cioè tutti i sottotipi che ricadono sotto la stessa categoria. Anche questa è un’operazione che, nei dizionari convenzionali, richiede una lettura estensiva e non è facilitata da una struttura esplicitamente navigabile. Tutto ciò rende evidente una carenza fondamentale: la rappresentazione tradizionale del significato lessicale non supporta in modo efficiente un’esplorazione sistematica e gerarchica dei concetti, complicando il lavoro di chi cerca di orientarsi nella rete semantica della lingua.

In più, va considerata l’esistenza di cicli definitori: trattandosi di manufatti realizzati da esseri umani, i dizionari tradizionali non sempre riescono a evitare che un termine venga definito usando un altro termine che, a sua volta, dipende dal primo, creando così dei circuiti che mettono in crisi la comprensione e l’accesso coerente al significato.

Di fronte a queste limitazioni, WordNet offre un approccio strutturato e navigabile alla rappresentazione del lessico. Sebbene la sua interfaccia possa sembrare datata – quasi anni ’70 – permette una vera esplorazione della rete semantica. Le definizioni non sono reinventate, ma organizzate in modo tale da rendere accessibile la loro collocazione gerarchica. Ogni senso di un termine è infatti situato in modo specifico all’interno della struttura e consente di visualizzare sia gli iponimi (cioè i concetti più specifici) sia gli iperonimi (quelli più generali).

Ad esempio, il primo senso del termine *tree*, inteso come pianta perenne con tronco e rami, mostra come iponimi termini come *yellowwood tree*, *lancewood*, *anise tree*, *guinea pepper* e così via. WordNet permette di visualizzare tutti questi sensi e di esplorare i numerosi tipi di alberi censiti nel suo database. Va detto che, a volte, l’interfaccia reagisce lentamente quando si tenta di esplorare porzioni molto estese della gerarchia: in questi casi, può essere necessario accedere direttamente al database per eseguire query più efficienti.

Ogni voce è strutturata in blocchi informativi standard, accessibili anche tramite le API più diffuse, come NLTK. Questi blocchi includono informazioni utili come la frequenza d’uso di ciascun senso all’interno del corpus, un identificativo numerico, e soprattutto i *super sense tags*. Si tratta delle primitive semantiche che fungono da radici dei vari alberi della multigerarchia di WordNet. Tra questi super-sensi troviamo ad esempio *noun.plant*, *noun.cognition*, *noun.artifact* e altri, fino a un totale di 25 categorie principali.

Ogni senso ha anche un *offset*, un identificatore univoco utilizzato nelle interfacce programmatiche, e viene distinta in modo chiaro la categoria grammaticale a cui appartiene il termine: ad esempio, *tree* come sostantivo è trattato separatamente rispetto a *tree* come verbo, rendendo esplicita la distinzione tra i diversi usi polisemici.

La struttura su cui si basa WordNet deriva da una solida base di evidenze sperimentali e da studi psicolinguistici condotti dal gruppo di sviluppo, composto da una combinazione di psicologi, linguisti e lessicografi. Non si tratta, dunque, di un modello puramente teorico, ma di una risorsa costruita a partire da dati concreti sul funzionamento cognitivo e linguistico dei parlanti.

Tra le metodologie utilizzate per raccogliere tali dati, un ruolo importante è stato svolto da indagini empiriche in cui si sottoponevano a parlanti di lingua inglese una serie di affermazioni, richiedendo loro di rispondere in modo binario (vero/falso) e misurando, al contempo, il tempo impiegato per dare la risposta. Le frasi utilizzate spaziavano da proposizioni ovvie e familiari, come “un canarino può cantare”, a frasi più eccentriche o meno pertinenti, come “un canarino ha la pelle”.

L'analisi dei tempi di risposta ha permesso di osservare che le risposte diventano più lente e meno sicure man mano che la proprietà associata si allontana dalla rappresentazione mentale che i parlanti hanno del concetto in questione. Questo fenomeno suggerisce l’esistenza di una struttura gerarchica dell’organizzazione semantica nella mente dei parlanti, dove alcune feature sono direttamente associate a un concetto, mentre altre richiedono un “viaggio” mentale più lungo lungo i nodi della gerarchia per essere recuperate.

L’ipotesi di fondo è che le informazioni siano memorizzate nei nodi della rete semantica, e che il tempo di accesso sia proporzionale alla distanza tra il nodo attivato e la feature ricercata. Tuttavia, studi successivi hanno mostrato che non tutto dipende da questo meccanismo gerarchico: la frequenza con cui un’informazione viene recuperata gioca un ruolo altrettanto determinante. In altre parole, concetti di uso quotidiano e parole frequentemente utilizzate sono accessibili più rapidamente, come se disponessero di autostrade cognitive, mentre concetti rari o marginali risultano più difficili da attivare, quasi fossero collocati in una periferia del lessico mentale.

Passando alla struttura interna di WordNet, essa è organizzata secondo una cosiddetta *multi-hierarchy*. Alla base di questa organizzazione si trovano 25 *supersense tags*, categorie semantiche fondamentali che fungono da radici per le diverse ramificazioni gerarchiche. Tra questi supersensi troviamo, ad esempio, *noun.cognition*, *noun.plant*, *noun.feeling*, *noun.act*, *noun.artifact* e così via. Si tratta di categorie molto eterogenee: alcune sono altamente astratte, come *tempo* o *emozione*, altre sono decisamente più concrete, come *pianta* o *oggetto*.

Questa disomogeneità riflette l’intento generale di WordNet, che non è quello di costruire un’ontologia formale (come farebbe, ad esempio, una risorsa ispirata al modello di *DOLCE*), ma di fornire una rappresentazione linguistica del lessico, utile per descrivere e classificare i nomi secondo la loro distribuzione semantica osservata nei dati. I creatori di WordNet non miravano quindi a distinguere con rigore tra eventi, processi ed entità, bensì a coprire in maniera ampia ed empiricamente fondata i domini concettuali che emergono nella lingua ordinaria.

In quest’ottica, alcune categorie più generali come *thing*, *entity*, *living thing*, *organism* e *object* avrebbero potuto costituire un livello superiore, una sorta di “top level” gerarchico, ma non è stato realizzato un simile ordinamento formale, proprio perché gli obiettivi del progetto erano linguistici e non ontologici.

Infine, tornando sul tema delle *feature*, queste possono essere articolate a diversi livelli, come si accennava richiamando le ricerche di Eleanor Rosch. Le conoscenze che attiviamo e utilizziamo per interpretare il lessico non sono distribuite uniformemente, ma organizzate su piani distinti e secondo diversi gradi di specificità, e WordNet cerca di catturare almeno parte di questa articolazione nella sua rete di relazioni semantiche.

Esiste un livello intermedio nella categorizzazione concettuale che sembra giocare un ruolo centrale nei processi cognitivi e linguistici. Questo livello, spesso associato ai cosiddetti prototipi, è ritenuto il più immediato nell’acquisizione linguistica: è il primo a essere appreso durante le fasi iniziali dello sviluppo linguistico, per poi fungere da base sia per la specializzazione verso concetti più specifici, sia per l’astrazione verso categorie più generali. Le ricerche suggeriscono che la maggior parte delle *feature*, ovvero le caratteristiche distintive associate ai concetti, si collochi proprio a questo livello intermedio. Quando un concetto si trova in questa fascia, né troppo astratto né eccessivamente specifico, i parlanti sono in grado di elencarne un numero maggiore di tratti distintivi. Al contrario, man mano che ci si allontana da questo livello, le descrizioni si fanno più scarne e generiche.

Per quanto riguarda la struttura interna di WordNet, ogni parte del discorso — sostantivi, verbi, aggettivi e avverbi — è organizzata in una struttura separata. Questo riflette il fatto che ciascuna parte del discorso codifica un tipo di informazione diverso, ed è soggetta a modalità differenti di accesso e utilizzo. In particolare, si può osservare come gli attributi siano tipicamente associati agli aggettivi, mentre le funzioni siano legate ai verbi, e le entità ai sostantivi.

Gli attributi all’interno di WordNet si presentano come proprietà che possono essere assegnate a un’entità: ad esempio, si può dire che un canarino ha una *size* il cui valore è “piccolo”. L’assegnazione degli attributi, tuttavia, è soggetta a limiti semantici. Alcuni aggettivi risultano perfettamente accettabili in relazione a certi sostantivi — dire che un canarino è affamato o sazio è naturale — mentre altri accostamenti generano dissonanze percettibili, come dire che un canarino è *stingy* (tirchio), che appare chiaramente fuori luogo. Questo perché non tutti gli attributi sono semanticamente compatibili con tutti i nomi.

Attualmente non esistono, all’interno di WordNet, meccanismi espliciti che impediscano questi accostamenti anomali. La risorsa, infatti, non è progettata per regolamentare la coerenza semantica tra gli elementi, ma per fornire una mappa relazionale tra i significati. È costruita come un insieme di atomi di significato — i *synset* — che rappresentano sensi distinti dei termini e che sono collegati tra loro da relazioni semantiche. WordNet non stabilisce quali combinazioni siano appropriate, ma consente di esplorare le connessioni tra i sensi.

La forza di WordNet risiede nella sua struttura relazionale: permette di navigare tra i sensi e identificare quelli sinonimici o affini. Ogni *synset* corrisponde a un significato distinto, definito con precisione, e il sistema offre una copertura ampia della lingua inglese. Tuttavia, proprio per la sua natura atomica e relazionale, non è attrezzata per esprimere la compatibilità o l’incompatibilità semantica tra certi accostamenti linguistici — un compito che oggi potrebbe essere affrontato in modo complementare attraverso strumenti come gli *embedding* contestuali, capaci di cogliere simili vincoli sulla base dell’uso reale nei corpora linguistici.

WordNet, quindi, rimane una risorsa fondamentale per lo studio del significato, ma deve essere integrata con approcci più dinamici per catturare la complessità dell’uso linguistico quotidiano.

**Lezione 4**

WordNet è stato introdotto come il primo, e di gran lunga il più influente, sistema per la rappresentazione e gestione dei meccanismi di senso nel campo della linguistica computazionale. La sua costruzione poggia su solide basi teoriche e si configura come una rete semantica articolata, nella quale i significati sono connessi tra loro attraverso una pluralità di relazioni semantiche. L’elemento costitutivo fondamentale di questa rete è il *synset*, un costrutto che, pur arbitrario in una certa misura, raggruppa termini sinonimici e funge da unità di base per la descrizione dei sensi. L’attenzione si è concentrata sulla porzione che riguarda i sostantivi, una delle quattro categorie grammaticali principali trattate da WordNet insieme ai verbi, agli aggettivi e agli avverbi.

Nel corso della trattazione si è fatto riferimento a relazioni semantiche come la meronimia — la relazione parte-tutto — che ha come inverso l’olonimia. Questa relazione è stata discussa anche in relazione all’ontologia, evidenziando come possa fungere da veicolo per il trasferimento di proprietà: se, ad esempio, il becco e l’ala sono meronimi di bird (uccello), e canary (canarino) è un iponimo di bird, allora si può inferire che anche il canarino possiede becco e ali. Questo meccanismo, apparentemente complesso quando analizzato esplicitamente, è in realtà gestito con estrema rapidità e naturalezza nella comunicazione quotidiana, sia nella produzione che nella comprensione linguistica.

La meronimia è una relazione asimmetrica e, in alcuni casi, transitiva. Tuttavia, tale transitività presenta dei limiti. Prendendo ad esempio una maniglia, che è parte di una porta, e considerando la porta come parte di una casa, risulta poco naturale dire che la maniglia è parte della casa, sebbene la relazione sia tecnicamente transitiva. È lo stesso con esempi come il ramo, parte dell’albero, che a sua volta è parte della foresta; ma non si afferma comunemente che la foresta abbia rami. Questi esempi dimostrano che la transitività della relazione parte-tutto funziona solo entro certi confini semantici e pragmatici.

Queste relazioni, che rientrano tutte sotto il vasto ombrello della meronimia, sono complesse e ad oggi non esiste una modellazione generale che sia valida per ogni dominio. Le implementazioni più efficaci si concentrano su casi specifici, come *componente-oggetto* (es. ramo-albero), *membro-collezione*, e *staff-oggetto*. Sono relazioni relativamente più semplici da modellare e si rivelano spesso le più utili nei contesti applicativi.

Quando si passa a considerare entità più dinamiche, come processi o fasi, la situazione si complica ulteriormente. I processi biologici e naturali, così come quelli di lavorazione, presentano maggiori difficoltà di rappresentazione e mancano ancora di implementazioni soddisfacenti. Tuttavia, in alcuni domini specifici, è possibile ricorrere alla rappresentazione delle funzioni come criterio per associare i sensi: per esempio, si può dire che la funzione della matita è quella di scrivere, o che il coltello serve a tagliare. Nonostante ciò, non tutto può essere ricondotto a una funzione. Ci sono elementi del mondo, come il canarino spesso utilizzato come esempio, che non hanno una funzione esplicita o univoca. Non si può affermare con certezza che la funzione del canarino sia volare o cinguettare, perché queste attività non costituiscono un fine strumentale nel senso classico. Per integrare in modo più accurato i meccanismi funzionali nella rappresentazione semantica, è necessario ricorrere a strutture concettuali più elevate, come quelle offerte dall’analisi dei ruoli semantici. Questo passaggio consente di ancorare le relazioni semantiche non solo agli oggetti e alle loro proprietà, ma anche ai ruoli che tali oggetti assumono all’interno di eventi o azioni. Nel corso della trattazione, pur rimanendo formalmente nell’ambito dei sostantivi e dei concetti nominali, si osserva un progressivo spostamento del focus verso l’universo dei verbi. Quando si inizia a parlare di *funzioni* e di *concetti nominali*, si entra infatti in un’area semantica che si avvicina sempre di più a quella verbale, perché i concetti di funzione e uso sono fortemente legati alle azioni e quindi ai verbi.

È in questo contesto che si introduce il concetto di *semantic role* — ruoli semantici — che costituiscono una sorta di corredo necessario e complementare al verbo. Si tratta di quegli elementi ricorrenti e tipici che accompagnano un verbo e che definiscono i partecipanti all’azione. Per esempio, in una relazione come quella tra *coltello* e *tagliare*, si può identificare il coltello come lo *strumento* dell’azione. E lo stesso vale per coppie come *scavare* e *buco*, *dipingere* e *quadro*, *contenere* e *scatola*: in tutti questi casi, si nota come i secondi termini siano in realtà delle *nominalizzazioni*, ovvero sostantivi derivati da verbi.

Questa riflessione sulle funzioni conduce inevitabilmente in un’area semantica particolarmente densa e complessa. Si tratta di un territorio ancora in gran parte inesplorato — non tanto perché ignoto, ma perché estremamente ricco di sfumature e variabilità. Il linguaggio comune produce significati in modo tanto vario e stratificato che risulta difficile, persino per i linguisti, racchiuderli all’interno di una teoria generale, e ancor più per l’informatica linguistica riuscire a implementarli operativamente in modo efficace.

Un esempio particolarmente indicativo di questa complessità si trova negli aggettivi, come *buono*. Cosa significa *buona matita*? Vuol dire che scrive bene. *Buon coltello*? Che taglia con precisione. Ma *buon amico*? Qui la nozione di bontà si sposta su un piano relazionale ed emotivo. Il significato dell’aggettivo cambia a seconda del sostantivo che regge — il cosiddetto *head noun*. L’aggettivo *buono* proietta quindi tratti distintivi diversi a seconda del contesto in cui si trova.

Fino a qualche decennio fa, i parser linguistici includevano componenti lessicali esplicite, vere e proprie basi di conoscenza che associavano informazioni semantiche a ogni verbo, compilate manualmente e aggiornate all’occorrenza. Era un processo potenzialmente infinito: ogni nuovo uso richiedeva un’estensione della base di dati, rendendo questi sistemi difficili da mantenere ed estremamente laboriosi da sviluppare.

Con l’avvento dei modelli distribuzionali e degli embeddings, molte di queste lacune sono state colmate in maniera implicita. Questi modelli riescono, in parte, a sopperire alla mancanza di una codifica esplicita delle conoscenze, ma non offrono una reale comprensione del modo in cui i significati si strutturano e si combinano nel linguaggio. Mancano, cioè, di *conoscenza esplicita*. Una conseguenza interessante di questa situazione è che un oggetto, pur non appartenendo formalmente a una categoria, può essere descritto come un *buon esempio* di quella categoria, in funzione del contesto d’uso. È celebre, in questo senso, l’esempio proposto da George A. Miller nell’articolo fondativo di WordNet: si può dire che una scatola è una *buona sedia* se la si utilizza per sedersi, anche se non è una sedia in senso stretto. In quel contesto specifico, la scatola assume la funzione di sedia, e ciò basta per attribuirle quel significato, almeno temporaneamente. Questo esempio mostra quanto sia dinamico e flessibile il significato linguistico, e quanto la funzione d’uso possa prevalere sulla classificazione formale.

Il discorso sui ruoli semantici, per quanto importante, viene solo accennato in questa fase, poiché verrà ripreso in modo più approfondito nel momento in cui si affronterà FrameNet. In generale, si può osservare che i ruoli semantici di tipo strumentale sono i più studiati, in quanto più semplici da descrivere e da utilizzare nei sistemi linguistici computazionali. Tuttavia, non tutto può essere rappresentato attraverso ruoli semantici: molte espressioni del linguaggio naturale, infatti, sfuggono a questa classificazione e creano delle difficoltà sia nella teoria sia nell’applicazione. Questo stato di avanzamento rende possibile assumere che, partendo da un’analisi sintattica ben strutturata, si possano ottenere informazioni fondamentali per l’interpretazione semantica. Una delle nozioni chiave in questo contesto è quella di *frame di sottocategorizzazione*, che definisce le strutture sintattiche ammissibili per ciascun verbo. All’interno di questi frame, è possibile individuare informazioni semantiche aggiuntive, come le cosiddette *selectional restrictions* — vincoli semantici che indicano le proprietà attese degli argomenti del verbo.

Analizzare i verbi dal punto di vista semantico è tuttavia complesso. Anche se, quantitativamente, nel lessico esistono meno verbi rispetto ai sostantivi — in un dizionario aggiornato come il *Collins*, si trovano circa 43.000 sostantivi contro 14.000 verbi — la polisemia dei verbi è significativamente più alta. I dati mostrano una media di 1,74 sensi per sostantivo, contro 2,11 per verbo. Questo rende il compito di disambiguazione verbale particolarmente difficile.

Inoltre, se si sbaglia a disambiguare un verbo, si rischia di attivare un intero set di interpretazioni errate: il verbo porta con sé un frame semantico che include le sue aspettative sintattico-semantiche, e scegliere il senso sbagliato può compromettere l’intera analisi della frase.

Un’altra considerazione fondamentale riguarda la flessibilità semantica dei verbi, che è generalmente superiore a quella dei sostantivi. Il significato di un verbo può cambiare a seconda degli argomenti nominali con cui si combina, mentre i sostantivi tendono a mantenere una maggiore stabilità semantica. In un certo senso, è proprio il sostantivo che influisce sul significato del verbo, piuttosto che il contrario.

Infine, un aspetto rilevante è che i verbi più frequenti nell’uso quotidiano sono anche quelli con il maggior numero di significati. Questo è coerente con la distribuzione Zipfiana dei token linguistici: i verbi occupano la parte sinistra della curva, quella della massima frequenza d’uso, e quindi sono anche i più soggetti a fenomeni di polisemia.

I verbi, pur essendo in numero limitato rispetto ai sostantivi, tendono a essere molto più utilizzati e, di conseguenza, più soggetti a fenomeni di polisemia. Più un verbo è impiegato nel linguaggio quotidiano, più si carica di significati diversi. Questo fenomeno risponde a una logica di economia comunicativa: se è possibile esprimersi con un numero ridotto di parole, sfruttando la capacità di ciascuna di adattarsi a contesti differenti, il linguaggio ne esce semplificato dal punto di vista del repertorio, ma diventa più complesso da interpretare. È il concetto di *overloading* semantico: finché l’interlocutore riesce a comprendere il significato implicito nel contesto, il parlante può continuare a "sovraccaricare" quel verbo, attribuendogli nuovi sensi.

Dal punto di vista computazionale, però, questa flessibilità costituisce una vera e propria sfida. I verbi più comuni, come ad esempio *avere*, possono assumere significati completamente differenti a seconda del contesto — *avere una Mercedes* non ha nulla a che vedere con *avere mal di testa*. Questa versatilità semantica non è casuale: si pensa che uno dei fattori alla base della polisemia sia la natura concreta o astratta degli argomenti con cui il verbo si combina.

Un altro elemento che rende l’analisi più complessa è la presenza di usi figurati. Quando si parla di *mangiare la strada*, per esempio, non si intende più un'azione letterale, ma una metafora. In questi casi, le regole di *selectional restriction* — che definiscono quali tipi di argomenti sono compatibili con un certo verbo — vengono apparentemente violate, ma non necessariamente rendono la frase scorretta. Al contrario, sono strategie retoriche che il linguaggio adotta spontaneamente. Tuttavia, per chi si occupa di linguistica computazionale, riconoscere e trattare correttamente questi usi richiede un livello di sofisticazione elevato.

Il verbo *beat* rappresenta un caso esemplare della difficoltà legata alla polisemia verbale. Può significare *colpire* fisicamente, indicare una pulsazione, esprimere la sconfitta in un contesto competitivo, o ancora fare riferimento al ritmo musicale. Questi sensi sono tra loro in parte correlati, ma la distinzione tra loro non è affatto banale. Anche se possono essere raggruppati a un livello concettuale più astratto, nella pratica ognuno porta con sé sfumature e contesti d’uso specifici che ne complicano l’identificazione automatica.

La polisemia può assumere forme più o meno marcate. Alcuni casi, come quello di *board* — che può indicare una tavola da cantiere o un consiglio di amministrazione — sono abbastanza semplici da distinguere in base al contesto. Altri, invece, come quelli legati a *beat*, presentano distinzioni minime tra i significati, rendendo la disambiguazione particolarmente complessa. In questi casi, i sensi non sono separati da distanze semantiche evidenti, ma da sfumature quasi impercettibili, e sono proprio questi gli ambiti più critici per i sistemi di analisi automatica.

WordNet, pur essendo una risorsa preziosa, non sempre agevola il compito. L’organizzazione in *synsets* molto specifici può portare a una frammentazione eccessiva del significato. In totale, WordNet raccoglie circa 150.000 termini distribuiti in 117.000 synsets. La gerarchia dei verbi, in particolare, è articolata in 15 primitive semantiche — un numero più contenuto rispetto a quello dei sostantivi, che ne prevedono 25 — e include categorie come *bodily care and function*, *change*, *cognition*, e così via. Esiste, inoltre, una categoria aggiuntiva non facilmente riconducibile alle altre, legata agli *stati*, che presenta caratteristiche semantiche più sfumate e difficili da sistematizzare.

Gli sviluppatori di WordNet si sono resi conto precocemente che, mentre i sostantivi in qualche modo rispondevano a qualcosa che poteva essere gerarchizzato per i verbi questo non funzionava particolarmente. Anche nel contesto della linguistica computazionale applicata all'inglese, ci si è resi conto che non tutte le relazioni tra verbi sono immediatamente riconosciute dai parlanti. In alcuni test, per esempio, veniva chiesto se *to stroll* (passeggiare) fosse un tipo di *to walk* (camminare), e sorprendentemente molti parlanti non riconoscevano questa come una relazione “naturale” o intuitiva. Questo ha spinto i ricercatori a ipotizzare l’esistenza di un meccanismo semantico chiamato *conflation* — letteralmente una “fusione a caldo” — per spiegare come alcuni verbi inglobino più componenti concettuali all’interno di un’unica espressione verbale.

La *conflation* è diventata una strategia utile per spiegare in che modo i verbi combinano elementi semantici diversi. I casi più frequenti riguardano la modalità (*manner*), la causa (*cause*), ma anche l’intensità. Per esempio, molti verbi di movimento non esprimono soltanto l’idea del muoversi, ma la combinano con la maniera in cui ci si muove: correre, camminare, saltellare sono tutte conflazioni tra l’atto del muoversi e la sua modalità. Allo stesso modo, verbi legati al sonno possono essere declinati secondo l’intensità dell’azione: *doze*, *nap*, *sleep*, *slumber*, *drowse*... in inglese esiste una ricca varietà che rende queste sfumature semantiche più marcate e chiaramente distinguibili.

Da qui nasce l’esigenza di raggruppare i verbi in insiemi più gestibili: quelli che condividono caratteristiche semantiche affini vengono riuniti in blocchi coerenti, detti *synsets*. Questo approccio è alla base della costruzione della rete verbale di WordNet. Tuttavia, mentre per i sostantivi l’organizzazione gerarchica si sviluppa in maniera piuttosto regolare, in una struttura ad albero profonda e ben ramificata, per i verbi il discorso è più complicato.

Una delle difficoltà principali sta nel fatto che non tutti i verbi si lasciano classificare sotto un’unica radice semantica. Alcuni verbi appartengono a più categorie contemporaneamente, il che rende difficile costruire una gerarchia lineare. Per esempio, verbi di movimento possono avere due “genitori” concettuali diversi, riflettendo sia il concetto di movimento in sé sia, ad esempio, quello di cambiamento di stato o di direzione.

Altri verbi, come quelli legati al possesso, presentano una struttura troponimica interessante: i loro derivati (i troponimi) spiegano e articolano in modo più fine aspetti come il trasferimento o la modalità del possesso. È il caso, ad esempio, di *give*, *lend*, *donate*, che tutti implicano il possesso ma lo declinano in modi differenti.

La gerarchia verbale di WordNet ha quindi una morfologia diversa rispetto a quella dei sostantivi. Mentre quest’ultima può raggiungere profondità di circa venti livelli, quella dei verbi risulta molto più compressa. George Miller, uno dei fondatori di WordNet, parlava infatti di una struttura “a cespuglio”: piuttosto che alberi profondi e ramificati, ci si trova di fronte a strutture piatte, con molti verbi collocati sullo stesso livello e poche ramificazioni verticali. Raramente, infatti, si superano i tre o quattro livelli di profondità nella gerarchia verbale, segno che la categorizzazione semantica dei verbi segue una logica diversa, più orizzontale che verticale.

WordNet può essere impiegato per svolgere due compiti fondamentali ma distinti, seppur strettamente collegati: la Sense Similarity e la Word Sense Disambiguation.

Il primo di questi compiti, ovvero la *Sense Similarity*, riguarda la misurazione della somiglianza concettuale tra due termini. In pratica, il sistema riceve in input una coppia di parole e deve restituire un punteggio numerico che rappresenti la vicinanza semantica tra i due termini. Tuttavia, qui si presenta una prima difficoltà concettuale: l’input è costituito da termini, ma il risultato finale – il punteggio – si riferisce in realtà non ai termini in quanto tali, bensì ai *sensi* di quei termini. Questo implica che, per poter calcolare correttamente una distanza semantica, bisogna passare da una rappresentazione superficiale (la parola) a una più profonda e articolata (il senso specifico della parola).

Per esempio, se si considerano i sensi di “car” e “bus”, si potrebbe ottenere un punteggio di similarità pari a 0.8 su una scala che va da 0 (completamente dissimili) a 1 (identici). Questo valore suggerirebbe una buona somiglianza concettuale, coerente con il fatto che entrambi rientrano nella categoria dei mezzi di trasporto.

Gli approcci presentati nella lezione per affrontare il problema della *conceptual similarity* sono ancorati alla struttura ad albero offerta da WordNet. Quest’ultima permette di mappare i sensi all’interno di una rete gerarchica in cui è possibile calcolare distanze e prossimità attraverso la posizione dei *synset* e la profondità delle relazioni.

Per testare l’efficacia degli algoritmi, viene utilizzato un dataset chiamato *WordSim-353*, che è una risorsa ampiamente adottata a livello internazionale. Questo dataset consiste in 353 coppie di termini per ciascuna delle quali è stato assegnato un punteggio di similarità compreso tra 0 e 10. Il punteggio è il risultato della media delle valutazioni fornite da un gruppo di esseri umani, secondo una metodologia sperimentale rigorosa.

In fase di costruzione del *gold standard* – così si chiama una risorsa come questa, fondata sul giudizio umano e considerata punto di riferimento per la valutazione delle prestazioni di un sistema automatico – si selezionano persone che rispondono a stimoli linguistici sottoposti in ordine casuale. I giudizi estremi o incoerenti (gli *outlier*) vengono eliminati, e si calcola la media delle risposte rimanenti. Se, invece, la deviazione standard dei giudizi è troppo elevata, ciò suggerisce che il task non è ben definito o è troppo ambiguo.

In sintesi, il sistema automatico deve cercare di approssimare il più fedelmente possibile ciò che è stato valutato da esseri umani: l’obiettivo è ridurre la distanza tra la valutazione automatica e quella umana, raggiungendo un buon livello di concordanza.

Si propone l’analisi di tre metriche per la misura della similarità concettuale tra sensi utilizzando la struttura ad albero di WordNet. Le metriche si basano su una semplice intuizione: la distanza semantica tra due sensi può essere calcolata in funzione della lunghezza del cammino che li separa all’interno dell’albero gerarchico di WordNet. In sintesi, più due sensi sono lontani, meno sono simili; più sono vicini, maggiore è la loro similarità. Questa intuizione, per quanto rudimentale, costituisce una base utile soprattutto quando non si hanno a disposizione rappresentazioni distribuzionali avanzate come gli *embedding*. Anche qualora si disponga di questi ultimi, il problema non scompare: gli embedding, infatti, sono spesso calcolati a livello di parola, non di senso, e quindi non aiutano direttamente a distinguere le accezioni diverse di uno stesso termine.

Per valutare l’efficacia dei risultati ottenuti con queste funzioni, si farà ricorso agli indici di correlazione di Pearson e Spearman. Entrambi sono strumenti statistici consolidati, basati sul rapporto tra covarianza e deviazione standard. Pearson misura la correlazione lineare tra due insiemi di valori, mentre Spearman si basa sul rango, cioè sull’ordine relativo degli elementi nella serie, offrendo una misura non parametrica. L’implementazione di tali indici è disponibile in molte librerie di Python, come SciPy.

Immagine che contiene testo, Carattere, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, Carattere, schermata, algebra

Il contenuto generato dall'IA potrebbe non essere corretto.

La prima delle metriche è quella proposta da Wu e Palmer, due noti studiosi del settore. In questo modello, la similarità concettuale tra due sensi viene calcolata nel seguente modo: si prende il doppio della profondità del *lowest common subsumer* (LCS), cioè del nodo comune più profondo tra i due sensi nell’albero di WordNet, e lo si divide per la somma delle profondità dei due sensi considerati. La profondità, in questo contesto, corrisponde alla distanza del nodo dalla radice lungo la catena di iponimia.

Immagine che contiene testo, Carattere, linea, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

È importante osservare che, sebbene WordNet sia spesso rappresentato come un albero, in realtà è una struttura più complessa, tecnicamente un grafo orientato aciclico. Per semplificare l’implementazione, si può ridurre il problema a una struttura ad albero scegliendo, ad esempio, il primo iponimo disponibile – un’operazione che consente di mantenere la complessità computazionale sotto controllo.

Durante il calcolo, per ottenere il *lowest common subsumer*, si può procedere così: mentre si risale la gerarchia per calcolare la profondità di ciascun nodo, si memorizzano tutti gli antenati di ciascun senso; successivamente, si identifica il primo elemento comune tra le due liste di antenati. La profondità di questo nodo condiviso serve poi come base per completare il calcolo della similarità secondo la formula di Wu e Palmer.

Infine, viene fornito un elemento di confronto per valutare le prestazioni attese. L’output del sistema – cioè i punteggi di similarità ottenuti – dovrà essere confrontato con i giudizi umani presenti in dataset come WordSim-353. Tuttavia, la correlazione attesa tra le valutazioni automatiche e quelle umane non è elevata: si assesta generalmente intorno al 30-35%. Risultati significativamente superiori a questa soglia, paradossalmente, potrebbero indicare un errore nell’implementazione del sistema, piuttosto che una sua eccezionale efficacia.

La seconda delle tre metriche è la *shortest path similarity*, una metrica che si basa sulla distanza più breve tra due sensi (synset) nella gerarchia di WordNet. L’intuizione di fondo è semplice e immediata: quanto più breve è il percorso che connette due sensi all’interno dell’albero, tanto maggiore sarà la loro similarità concettuale.

Immagine che contiene testo, Carattere, bianco, calligrafia

Il contenuto generato dall'IA potrebbe non essere corretto.

Il valore della profondità massima (indicata come *Depthmax*) è una costante che può essere trovata nella documentazione ufficiale di WordNet. Per la versione 3.0, spesso utilizzata nelle interfacce di librerie Python come NLTK, questo valore è pari a 19, ma è sempre bene verificarlo in base al contesto applicativo. Il valore risultante dalla formula può variare da 0 fino a **2 × Depthmax**, e si può naturalmente riscalare in un intervallo arbitrario (come da 0 a 1) per esigenze di confronto o normalizzazione tra modelli diversi.

La terza metrica presentata è quella proposta da **Leacock e Chodorow**, due autori noti per i loro studi sulla similarità semantica. Questa metrica introduce una componente logaritmica.

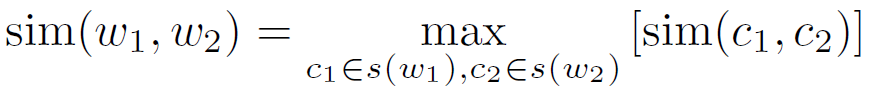
Immagine che contiene testo, Carattere, bianco, calligrafia

Il contenuto generato dall'IA potrebbe non essere corretto.

Anche in questo caso, la distanza tra i due sensi viene calcolata come lunghezza del cammino nel grafo gerarchico di WordNet, ma anziché essere sottratta da un valore massimo, viene convertita in un punteggio tramite logaritmo negativo. L’intervallo entro cui può variare questa metrica va da poco sopra 0 (escludendo 0 stesso per via del logaritmo) fino a **log(2 × Depthmax + 1)**. Questa formulazione introduce una maggiore sensibilità per le distanze più brevi, penalizzando le distanze crescenti in maniera più marcata rispetto alla metrica precedente.

Tutte queste metriche sono già state implementate in modo esteso in diverse librerie Python, quindi non è necessario reinventarle. Tuttavia, l’esercitazione pratica prevede di confrontare le proprie implementazioni con quelle di libreria, senza però limitarsi a richiamare funzioni esistenti.

Un punto di attenzione è che l'input è costituito da coppie di *termini*, mentre le formule utilizzano i *sensi*. Per calcolare la similarity fra 2 termini si considera la massima similarity fra tutti i sensi del primo termine e tutti i sensi del secondo termine. L'ipotesi è cioè che i due termini funzionino come contesto di disambiguazione l'uno per l'altro. Nella formula *c* sono i concetti che appartengono ai synset associati ai termini *w1* e *w2*.



Si considera il dataset utilizzato per l'esperimento: un semplice file composto da coppie di termini e da un punteggio numerico associato a ciascuna coppia, che rappresenta la similarità percepita da giudizi umani. Il file ha una struttura molto lineare, con tre colonne: *word1*, *word2* e *human*, dove *human* è il punteggio medio attribuito da valutatori umani. Ad esempio, una coppia come *tiger* e *cat* riceverà un punteggio alto, mentre *tiger* e *tiger* avranno similarità massima (pari a 1, o a 10 nella scala originale).

Il passaggio cruciale è che, mentre i dati in input sono parole, le funzioni viste operano su sensi. Pertanto, è necessario implementare una strategia per mappare le parole ai loro sensi. Il metodo suggerito consiste nel calcolare tutte le possibili coppie di sensi derivanti dai due termini dati, applicare la funzione di similarità a ciascuna coppia, e infine selezionare il valore massimo ottenuto. In questo modo, si cerca di simulare il comportamento umano che tende a selezionare il significato più rilevante o coerente nel contesto del confronto.

In questo contesto, ogni senso lessicale può essere inteso come una sorta di contesto minimo ma sufficiente per disambiguare l’altro termine nella coppia. L’idea è che, quando si mette in relazione una parola con un’altra, il senso di una delle due possa funzionare come filtro o vincolo interpretativo per l’altra, restringendone il campo semantico.

Si consideri, ad esempio, la coppia *calcio* e *potassio*. In questo caso, è piuttosto evidente che entrambi i termini fanno riferimento a elementi chimici. Il senso “elemento chimico” di *calcio* agisce quindi da cornice interpretativa che rende chiaro e inequivocabile anche il significato di *potassio*, che del resto non presenta ambiguità rilevanti.

Diverso è il caso di una coppia come *calcio* e *Juventus*. Qui il significato di *calcio* come elemento chimico non è più plausibile. Il contesto fornito da *Juventus* orienta invece verso il significato sportivo della parola *calcio*. Allo stesso modo, in una coppia come *calcio* e *pistola*, il contesto suggerisce un’ulteriore accezione, quella del *calcio* come parte dell’arma da fuoco. In tutti questi casi, la vicinanza semantica fra i sensi è ciò che consente l’individuazione del significato corretto.

L’algoritmo che si propone utilizza quindi una strategia di massimizzazione: prende in considerazione tutte le possibili coppie di sensi tra due termini e seleziona quella con il punteggio di similarità più elevato. L’ipotesi di fondo è che questa operazione porti ad individuare la coppia di significati più affini tra loro, ovvero quelli che con maggiore probabilità vengono attivati contemporaneamente in un contesto naturale.

Naturalmente, si potrebbe anche scegliere di non restituire solo il punteggio massimo, ma anche gli argomenti (cioè i sensi specifici) che lo producono. In questo modo, oltre al valore numerico della similarità, si otterrebbe anche l’identificazione esplicita dei due sensi che maggiormente correlano: ad esempio, il senso sportivo di *calcio* in relazione a *Juventus*, oppure il senso legato alle armi nel confronto tra *calcio* e *pistola*. Questa informazione può rivelarsi preziosa per fini di disambiguazione più approfondita o per l’analisi semantica fine.

La Word Sense Disambiguation (WSD) è considerata a tutti gli effetti un problema aperto nel campo dell’elaborazione del linguaggio naturale. Anche i sistemi più avanzati, che affrontano il task all’interno di cornici ben definite e scientificamente strutturate, riescono difficilmente a superare soglie di accuratezza dell’82-83%. Si tratta di punteggi che, pur essendo notevoli dal punto di vista tecnico, non sempre offrono garanzie sufficienti per applicazioni in cui è richiesta una precisione elevata, ad esempio in ambito medico, giuridico o tecnico. È quindi evidente che non siamo di fronte a un problema risolto, bensì a una sfida ancora attuale.

Il compito, in sé, è semplice nella formulazione: dato un termine polisemico e un contesto in cui esso appare, si tratta di individuare in quale dei possibili sensi sia stato utilizzato. Questa operazione, per quanto possa sembrare lineare, rappresenta un passaggio fondamentale per molte attività complesse come la traduzione automatica, il question answering, l’analisi del sentiment, la generazione del linguaggio e molte altre.

Nella fase introduttiva di approccio al problema, si lavora utilizzando WordNet, che è lo strumento lessicografico attualmente a disposizione. Il funzionamento di base è molto semplice: si interroga il sistema fornendo un termine, e in risposta si ottiene una lista di *Synset*, ovvero insiemi di sinonimi che rappresentano ciascun senso distintivo della parola. Ogni Synset è dotato di definizione e, spesso, di esempi d’uso, oltre a collegamenti semantici con altri Synset, come iponimi e iperonimi.

Un esempio classico di parola polisemica è *bank*, che può riferirsi all’istituzione finanziaria, ma anche alla riva di un fiume, a un pontile o ad altri concetti ancora. WordNet consente di esplorare tutti questi sensi in modo strutturato, fornendo quindi una base per provare a disambiguare il termine una volta inserito in un contesto più ampio.

All’interno del materiale del corso, viene messo a disposizione un notebook che illustra come interrogare WordNet, ad esempio con il termine *bank*, e ottenere i sensi corrispondenti. Sebbene venga proposto con l’utilizzo della libreria NLTK di Python, non vi è alcuna restrizione nell’uso di altri linguaggi o strumenti, purché si acceda in modo analogo alle informazioni strutturate offerte da WordNet. Questo approccio costituisce il primo passo per affrontare il problema della disambiguazione del significato delle parole in contesto, una sfida tanto elementare nella forma quanto complessa nella sostanza.

Tramite codice, è possibile interrogare facilmente WordNet (vedi file fornito). È possibile accedere a diverse informazioni rilevanti legate ai singoli sensi delle parole. Alcune di queste sono forse meno interessanti dal punto di vista pratico, come ad esempio l’**offset**, un costrutto utile per risalire a un identificatore univoco del *Synset*, ma che raramente è necessario nelle attività di base. Più interessanti, invece, sono due risorse fondamentali: le **definizioni** (o glosse) e gli **esempi d’uso**. Le glosse descrivono il significato specifico di un senso e servono a differenziarlo dagli altri sensi della stessa parola, mentre gli esempi mostrano in che modo il termine viene impiegato in quel preciso significato.

Prendendo ad esempio la parola *bank*, si ottiene una lista di Synset ordinati. Ognuno di essi è identificato da una combinazione che include anche l’informazione grammaticale (part of speech), dato che *bank* può essere sia un sostantivo che un verbo. Se si accede al primo Synset (ad esempio, *bank.n.01*), è possibile ottenere l’offset, la definizione ("sloping land, especially a slope beside a body of water") e anche una serie di esempi che aiutano a chiarire l’uso contestuale di quel senso.

Sono disponibili poi diverse funzionalità che permettono di accedere, oltre che a definizioni ed esempi, anche alla **gerarchia semantica**. È possibile, ad esempio, navigare tra gli **iponimi** (i concetti più specifici) e gli **iperonimi** (quelli più generici) di un determinato Synset. Se si prende *pasta*, ad esempio, si possono ottenere gli iponimi come *cannelloni*, *lasagne* e *spaghetti*. Allo stesso modo, si può risalire agli iperonimi che classificano il concetto in categorie più ampie.

Le relazioni semantiche navigabili non si fermano qui. Un’altra categoria interessante è quella dei **meronimi**, ovvero le parti che compongono un oggetto. Per esempio, associato a *car.n.01*, è possibile elencare elementi come *acceleratore*, *airbag*, *motore*, *clacson*, e così via.

Sono inoltre disponibili strumenti per calcolare la **similarità tra sensi**, come già visto in precedenza, e queste funzionalità sono tutte accessibili programmaticamente, anche se è sempre raccomandato esercitarsi autonomamente piuttosto che affidarsi semplicemente alle implementazioni già pronte.

WordNet contiene anche alcune **funzionalità avanzate**, come ad esempio l’*entailment* (implicazione semantica), ma queste sono da considerarsi ancora sperimentali e non sempre affidabili, almeno nella forma in cui sono rese disponibili attraverso l’interfaccia standard. Si tratta di aspetti che potranno essere approfonditi più avanti, magari nel contesto di strumenti più sofisticati come i *transformer*.

Infine, un aspetto importante della risorsa è la gestione delle **espressioni composte** o *multi-word expressions*, come *pickup*. WordNet supporta nativamente un’ampia gamma di termini composti, offrendo così l’opportunità di esplorare anche i meccanismi lessicali legati a questo tipo di elementi, che spesso pongono sfide aggiuntive nella disambiguazione del significato.

Una delle prime questioni che ci si pone nel trattare la disambiguazione dei termini polisemici è la quantità di contesto necessaria per risolvere efficacemente l’ambiguità. In realtà, nella maggior parte dei casi, il contesto richiesto può essere sorprendentemente ridotto. Lo si intuisce facilmente affrontando i primi esercizi pratici: bastano infatti combinazioni semplici come *calcio e stagno* o *calcio e potassio* per identificare con chiarezza il significato desiderato. Questo dimostra che, in molti casi, un contesto minimo è già sufficiente.

Idealmente si può immaginare di utilizzare una sorta di finestra scorrevole sul testo, centrata attorno al termine polisemico, ma anche con margini asimmetrici se necessario. Tuttavia, l’idea che un’estensione eccessiva della finestra – e quindi del contesto – porti automaticamente a una migliore disambiguazione non è supportata da evidenze sperimentali. Anzi, oltre una certa soglia, il rischio è che si introduca rumore informativo: si inizia a parlare d’altro, si aprono nuove unità di senso e il termine target si perde all’interno di un contesto troppo ampio e poco coerente. Per questo, una finestra più ristretta ma ben focalizzata tende a essere più efficace.

Nel rappresentare il contesto di occorrenza del termine ambiguo, si possono adottare diversi approcci. Un primo passo consiste nell’estrarre delle **feature** dal contesto. Un esempio tipico è l’utilizzo delle informazioni grammaticali come i **part-of-speech tag** (POS tag), cioè le categorie grammaticali dei termini. Il task della disambiguazione semantica è infatti strettamente legato, anche nella letteratura scientifica, al POS tagging. A volte, si procede alla **lemmatizzazione**, ossia alla riduzione dei termini alla loro forma base. Più raramente, si utilizzano gli **stemm** (radici lessicali), che risultano troppo approssimativi per i fini della disambiguazione, in quanto trascurano molte sfumature semantiche.

Tutte queste informazioni possono essere organizzate in **vettori**, che rappresentano in modo numerico le caratteristiche salienti del contesto. In generale, si distinguono due tipi principali di vettori:

1. **Collocational features**: si tratta di vettori che preservano la **struttura sequenziale** del testo. Le frasi vengono considerate come sequenze ordinate, e si mantiene memoria dell’ordine e della posizione relativa dei termini. Questo approccio permette, ad esempio, di costruire finestre scorrevoli che raccolgano due o tre parole a sinistra e a destra del termine ambiguo, includendo eventualmente anche il POS tag di ciascun elemento.
2. **Bag of words**: è un approccio più semplice e computazionalmente efficiente, ma anche più rudimentale. In questo caso, la struttura della frase viene completamente ignorata. Le parole sono raccolte in un insieme non ordinato e la loro posizione relativa non viene conservata. Questo può portare a problemi: frasi con significati opposti o molto diversi possono essere rappresentate con la stessa "bag", rendendo impossibile cogliere la differenza semantica. Ad esempio, frasi come *“il corso non è interessante ma l’insegnante è simpatica”* e *“il corso è interessante ma l’insegnante non è simpatica”* finiscono per avere una rappresentazione identica, pur trasmettendo contenuti molto diversi.

L'approccio collocazionale, pur essendo più complesso da gestire, consente quindi una rappresentazione del contesto più ricca e fedele alla realtà linguistica, offrendo una base più solida per la disambiguazione.

Si prenda in considerazione una frase qualsiasi, come ad esempio *electric guitar and bass player stand off to one side, not really part of the scene*. In questo caso si intende analizzare il termine polisemico *bass*, utilizzando una rappresentazione del contesto basata sulle **collocational features**, ovvero attraverso una finestra scorrevole che preleva parole e informazioni grammaticali dai dintorni del termine.

Supponendo che *bass* sia la parola in posizione i all’interno della frase, si prelevano parole in posizioni relative come i−2, i−1, e anche le rispettive etichette grammaticali (POS tag). Nel caso specifico, dalla finestra estratta si ottengono termini come *guitar* (sostantivo), *and* (congiunzione), *player* (sostantivo), e *stand* (verbo). Non tutte queste parole sono ugualmente informative per la disambiguazione, ma almeno due — *guitar* e *player* — forniscono segnali importanti a favore dell’interpretazione musicale di *bass* (cioè basso elettrico) rispetto all’alternativa zoologica (il pesce).

Questa tecnica permette di preservare una certa struttura e sequenzialità nella rappresentazione del contesto, rendendola utile per distinguere tra i possibili sensi di un termine.

Dall’altra parte, esiste un approccio più semplice e computazionalmente efficiente: quello basato sulla **bag of words**. In questo modello, le frasi vengono trattate come insiemi non ordinati di termini, e l’informazione raccolta consiste, nella forma più elementare, nella sola presenza o assenza di un termine. In pratica, si costruisce un vettore binario in cui ogni posizione rappresenta un termine del vocabolario generale e contiene un “1” se la parola è presente nel contesto, “0” altrimenti.

Questo tipo di vettore può rapidamente diventare molto lungo, perché la sua dimensione corrisponde al numero totale di termini distinti (il dizionario) presenti nella collezione di testi analizzati. Quindi anche per frasi brevi, il vettore risultante può avere centinaia o migliaia di posizioni, la maggior parte delle quali vuote. Nel caso della frase citata, solo due posizioni — quelle corrispondenti a *guitar* e *player* — saranno attivate (impostate a 1), mentre tutte le altre resteranno a zero.

Questo approccio, pur essendo meno preciso rispetto a quello collocazionale, ha il vantaggio della semplicità e della velocità computazionale. Inoltre, può essere arricchito in vari modi. Ad esempio, si possono contare le occorrenze invece di registrare solo la presenza, oppure si può applicare una conoscenza di dominio per costruire vettori in cui le prime posizioni codificano termini legati a un certo ambito semantico — come pesci, strumenti musicali, e così via.

Un ulteriore approfondimento: l’algoritmo di LESK, che si basa su un’analisi semplificata delle occorrenze. In questo contesto, può essere sufficiente annotare semplicemente se una parola si presenta o meno, senza la necessità di contare quante volte compare. Anche questo è un compromesso tra efficienza e precisione: riduce il carico computazionale, pur sacrificando parte dell’informazione.

La differenza tra le due rappresentazioni — quella basata sulle collocazioni e quella basata sulla bag of words — è quindi non solo teorica, ma ha implicazioni dirette sulla qualità dei risultati, sulla scalabilità del sistema e sulle risorse computazionali necessarie. Chi ha già affrontato temi come l’information retrieval probabilmente ha già incontrato queste problematiche, poiché la cardinalità dei vettori in una bag of words corrisponde, in quel contesto, al numero di termini presenti in tutti i documenti della collezione in esame.

L'algoritmo di Lesk si basa su un approccio piuttosto semplice e diretto per la disambiguazione dei termini polisemici, attraverso un processo di confronto tra il contesto di occorrenza e le definizioni dei vari sensi di un termine. L'algoritmo è costituito da 14 istruzioni, molte delle quali non particolarmente sofisticate, ma che, nel complesso, offrono un metodo operativo funzionante.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Il primo passo dell'algoritmo prevede l'inizializzazione del *best sense*, che viene impostato al senso con il numero più alto di occorrenze. Questo numero è quello che si trova nella parte superiore sinistra in WordNet, dove i sensi sono ordinati per frequenza. Ad esempio, quando si interroga un termine come *bank*, il primo senso che appare è generalmente quello con il numero maggiore di occorrenze, che corrisponde al senso più comune del termine nel corpus.

Successivamente, il processo si sposta sull'analisi del contesto di occorrenza del termine polisemico.

L'algoritmo azzera la variabile *max overlap*, che rappresenta il massimo overlap tra la *signature* di un senso e il contesto, e quindi imposta il contesto stesso con i termini presenti nella frase in cui il termine polisemico appare. Questo contesto è l'insieme di parole che circondano il termine e che può includere sia nomi che verbi, a seconda del caso.

Il cuore dell'algoritmo risiede in un ciclo che esamina ogni possibile senso del termine, da 6 a 13, a seconda del numero di sensi disponibili nel dizionario. Per ogni senso, viene creata una *signature*, che è un insieme di termini contenuti nella glossa e negli esempi d'uso associati a quel senso. Questa *signature* è specifica per ogni senso del termine e rappresenta un "riassunto" del significato di quel senso.

Dopo aver creato la *signature*, l'algoritmo calcola l'*overlap*, ovvero l'intersezione tra i termini presenti nella *signature* e quelli nel contesto di occorrenza. In pratica, si conta quante parole comuni ci sono tra la descrizione di un senso e il contesto in cui il termine appare. Se l'*overlap* è maggiore di zero, significa che il senso in questione ha una connessione con il contesto, e quindi si aggiorna il *best sense* con quel senso e si imposta il nuovo valore di *max overlap*. In altre parole, il senso che ha la maggiore sovrapposizione con il contesto viene selezionato come il più probabile.

Il ciclo continua iterando su tutti i sensi, e alla fine, al termine del ciclo, l'algoritmo restituisce il *best sense*, ovvero il senso che ha avuto la maggiore sovrapposizione con il contesto, anche se non è detto che sia sempre quello più appropriato. In ogni caso, l'algoritmo restituirà sempre un risultato, anche se talvolta non sarà il senso migliore, ma comunque sarà uno dei sensi possibili, selezionato sulla base della sovrapposizione con il contesto.

Questo metodo, pur essendo semplice e basato su una logica euristica, è utile per la disambiguazione e funziona in modo abbastanza affidabile, seppur non garantendo sempre il risultato ottimale.

L'algoritmo di Leskia funziona prendendo in input un termine polisemico e il contesto in cui appare, per poi restituire il "best sense", ovvero il senso più appropriato del termine. Una parte centrale del processo è l'uso della gloss, che rappresenta la definizione di un termine, e degli esempi d'uso. In pratica, la gloss viene analizzata per estrarre i "content words" (cioè sostantivi e verbi significativi) e non le "stop words" come congiunzioni o articoli, che potrebbero interferire con il processo di disambiguazione.

Per ogni senso di un termine, l'algoritmo costruisce una *signature* che consiste nell'insieme dei "content words" estratti dalla gloss e dagli esempi d'uso. Questo è un approccio di tipo *bag of words*, che destruttura completamente la definizione, raccogliendo semplicemente un elenco di parole da confrontare con il contesto di occorrenza del termine polisemico. In pratica, l'algoritmo verifica quante di queste parole siano comuni con il contesto in cui il termine appare nella frase di input, e calcola l'overlap, ovvero l'intersezione tra la *signature* e il contesto. L'overlap è un numero che rappresenta quante parole della *signature* compaiono anche nel contesto.

Nel caso dell’esempio di "bank" in una frase come *“bank can guarantee deposits will eventually cover future tuition costs because it invests in adjustable rate mortgage securities”*, il contesto sembra chiaramente riferirsi a un'istituzione finanziaria, dato l'uso di termini come "deposits", "mortgage", "securities". Per ogni possibile senso del termine, l'algoritmo esamina la gloss e gli esempi associati, costruisce la *signature* e calcola l'overlap con il contesto. Se per il primo senso l'overlap è 2, con parole come "deposits" e "mortgage", mentre per il secondo senso l'overlap è 0, l'algoritmo sceglierà il primo senso come quello più probabile, in quanto ha un maggiore sovrapporsi con il contesto.

Questa operazione di disambiguazione viene ripetuta per ogni senso del termine, e alla fine viene restituito il *best sense* in base al massimo overlap riscontrato. Anche se l'algoritmo non è perfetto, e non sempre restituisce il senso giusto, può comunque portare a un miglioramento rispetto all'assegnazione del senso più frequente senza alcun altro tipo di analisi. L'accuratezza di questo algoritmo varia tra il 50 e il 60%, a seconda della difficoltà del testo e della chiarezza dei sensi. Tuttavia, l'algoritmo basato sulla scelta del senso più frequente è spesso difficile da battere, e offre comunque un discernimento significativo rispetto a una scelta completamente casuale.

Un'altra limitazione dell'algoritmo, però, emerge quando il contesto o la gloss sono troppo brevi. Se la gloss non contiene abbastanza termini significativi, la *signature* sarà povera, e l'overlap sarà difficile da calcolare. In questi casi, l'algoritmo si trova a operare con una quantità minima di informazioni, il che può ridurre notevolmente l'efficacia della disambiguazione. Inoltre, quando non ci sono esempi d'uso a disposizione, la disambiguazione si basa esclusivamente sulla gloss, che può essere insufficiente, limitando ulteriormente le capacità dell'algoritmo.

Il *corpus Lesk* è una delle varianti di questo approccio, che cerca di affrontare alcuni di questi problemi, ma la difficoltà rimane legata alla quantità di informazioni a disposizione per costruire le *signature*. Se i termini della gloss sono troppo pochi, l'algoritmo potrebbe non essere in grado di fare una distinzione efficace tra i vari sensi del termine polisemico.

L'algoritmo Corpus Lesk, invece di limitarsi a contare semplicemente le parole che si sovrappongono tra il contesto e la gloss, applica anche un peso a ciascuna parola che si sovrappone. Questo peso è dato dalla frequenza inversa del documento, o IDF (Inverse Document Frequency), che è una misura standard nell'ambito della recupero dell'informazione. L'IDF misura in quante "collezioni" o "documenti" (in questo caso, le gloss e gli esempi) compare una determinata parola.

La logica alla base dell'IDF è che le parole che compaiono frequentemente in molti documenti, come le funzioni grammaticali (ad esempio "il", "di", "e"), hanno un IDF basso, mentre le parole di contenuto, cioè quelle che portano maggiore significato (come sostantivi e verbi specifici), hanno un IDF più alto. Questo permette di ridurre l'importanza delle parole di funzione, che tendono a non essere indicative del significato, e di enfatizzare invece le parole di contenuto che sono maggiormente rappresentative del senso del termine polisemico.

Per calcolare l'IDF di una parola, si può utilizzare la formula:

Immagine che contiene Carattere, testo, linea, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

**Lezione 5**

Frame.net rappresenta un passo importante nell’evoluzione delle risorse linguistiche, in particolare nell’ambito della semantica computazionale. Dopo aver esplorato come funziona l'analisi semantica lessicale, ad esempio con WordNet, Frame.net introduce un concetto che ha avuto una grande influenza anche nei lavori più recenti nel campo.

Per introdurre il concetto di Frame.net, bisogna fare riferimento a quanto proposto da Minsky nella sua teoria della visione artificiale. Minsky suggeriva che per semplificare l'interpretazione di un grande numero di informazioni, un sistema potesse fare affidamento su schemi o "frame" che permettessero di processare gli input in maniera più efficiente. Un esempio che avevamo visto riguarda la percezione di oggetti come le tenaglie: in un contesto, come quello di un'officina, un oggetto parzialmente visibile può essere immediatamente identificato come tenaglie, ma lo stesso oggetto, se riportato in un altro contesto, potrebbe essere interpretato in modo diverso, sulla base del nuovo frame in cui è collocato.

L'analogia con il linguaggio è semplice: proprio come nella visione, anche nel linguaggio l'interpretazione dipende dal contesto, e i "frame" possono essere utilizzati per semplificare il processo di disambiguazione e interpretazione semantica. I frame, in questa visione, sono un meccanismo per mappare le esperienze precedenti e applicarle al contesto attuale, proprio come un sistema che attinge a una base di conoscenza per risolvere un problema o rispondere a una domanda.

rame.net, che è stato lanciato all'inizio degli anni 2000 a Berkeley, ha l'obiettivo di unire due componenti cruciali per l'analisi semantica del linguaggio: da un lato, le parole e i termini linguistici, dall'altro i significati. La risorsa è focalizzata sull'inglese e cerca di catturare i modi in cui le parole e le espressioni ricorrono nel linguaggio e vengono interpretate nei vari contesti. Questi "frame" descrivono in modo astratto gli scenari di attività umana e il loro utilizzo linguistico. Frame.net è una risorsa bipartita, in quanto da un lato offre oltre 1100 frame che descrivono situazioni tipiche, dall'altro fornisce anche una descrizione dei significati associati a questi frame.

I frame non sono mai statici, ma sono pensati per essere estesi e adattati. In pratica, chi lavora con i frame sviluppa e personalizza questi modelli per le esigenze specifiche del contesto in cui opera. Ciò significa che non si parte mai da zero, ma si utilizzano frame generali, che poi vengono estesi attraverso l’ereditarietà per adattarsi alle necessità specifiche. Le operazioni principali in questo processo includono la caratterizzazione dei frame, cioè l’identificazione degli elementi rilevanti all’interno di un dominio, che è un’attività che, come per le ontologie, implica un certo livello di decisioni soggettive su cosa è rilevante e cosa no.

Una volta identificati i frame, il passo successivo è trovare i termini che si adattano a ciascun frame. Ogni frame ha un lessico associato, ovvero una serie di parole che descrivono gli elementi che lo compongono. In questo contesto, i linguisti hanno un ruolo fondamentale, ma gli informatici, soprattutto quando si trovano a lavorare in ambiti specialistici come quello giuridico o medico, devono spesso adattarsi e comprendere la terminologia di quel dominio per fare un buon lavoro. Oggi, grazie a tecniche come gli embedding, è possibile affrontare questo passaggio in maniera più automatica, trovando e associando termini con una certa coerenza semantica, anche se la terminologia specifica non è ancora completamente definita.

Un'altra parte importante del processo consiste nell'estrazione di frasi di esempio che illustrano come un determinato frame si realizzi nel linguaggio. Queste frasi, tratte da un corpus linguistico, forniscono esempi pratici di come i concetti descritti dai frame vengono effettivamente espressi nel linguaggio. L'annotazione di queste frasi è una parte costosa e complessa, ma è essenziale per costruire un sistema di apprendimento linguistico che possa essere utilizzato in contesti pratici.

In alcuni casi, per velocizzare il processo, si utilizzano approcci supervisionati, dove un piccolo insieme di frasi annotate manualmente viene usato per imparare i criteri di annotazione. Questi dati "silver", pur non essendo perfetti, possono comunque essere utili per l’apprendimento automatico e per generare modelli che, sebbene meno precisi di quelli completamente annotati, possono comunque risultare applicabili in molte situazioni. Infine, si derivano varianti delle modalità con cui un frame può essere espresso, cercando di coprire tutte le possibili sfumature e configurazioni del concetto che quel frame rappresenta.

Tutto questo processo, benché complesso e dispendioso in termini di tempo, ha permesso di creare una risorsa semantica robusta che può essere utilizzata per risolvere vari problemi linguistici, come la disambiguazione del significato delle parole, la traduzione automatica e l’analisi del linguaggio naturale in generale.

Recentemente, è capitato di dover lavorare con dei medici, una categoria che, come molte altre, è spesso diffidente nei confronti dell’informatica. In particolare, quando si tratta di risposte numeriche, c'è una certa riluttanza nell'accettare un risultato fornito da un sistema automatico. Un esempio di questo tipo di situazione si è verificato in un’applicazione in cui dovevamo analizzare documenti di pronto soccorso per rilevare lesioni che potessero derivare da eventi di violenza.

Si tratta di una questione particolarmente delicata, come facilmente si può immaginare. Nei reparti di pronto soccorso, infatti, chi è incaricato della refertazione iniziale lavora sotto una pressione enorme, con urgenze continue che non permettono di fermarsi troppo a riflettere o di navigare attraverso complesse interfacce. Il medico, in quel contesto, ha pochissimo tempo e, spesso, si trova a dover selezionare velocemente una voce tra le 250 opzioni di un menu a tendina, senza poter dedicare molto tempo alla scelta. Nel frattempo, la situazione è caotica, i pazienti diventano aggressivi, e il personale è sotto stress.

Questa condizione porta spesso al fatto che informazioni cruciali, come l'origine delle lesioni (specialmente quelle legate alla violenza), vengano trascurate, soprattutto in ospedali più piccoli o con carenze di personale. In questi casi, i sistemi informatici possono intervenire per raccogliere e affinare i dati che vengono già inseriti dai medici durante il triage, per cercare di completare l'informazione che altrimenti verrebbe omessa.

Nel nostro caso, abbiamo deciso di implementare un sistema che, dato un referto, classificasse se le lesioni avevano un’origine violenta o meno, utilizzando un classificatore binario. Nonostante l’accuratezza impressionante del nostro sistema, che si basava su dati raccolti e annotati dall’Istituto Superiore di Sanità, i medici erano ancora riluttanti ad accettare l’output del sistema. I risultati erano così precisi che, in alcuni casi, avevano anche permesso di correggere annotazioni umane che risultavano incomplete o addirittura errate. Tuttavia, il fatto che l'output provenisse da una rete neurale rendeva difficile per loro fidarsi di quel dato numerico.

Per superare questa diffidenza, abbiamo dovuto progettare un modello esplicativo che non solo fornisse il risultato della classificazione, ma che spiegasse anche il motivo per cui il sistema aveva dato quella risposta. L’idea era quella di mostrare che non si trattava solo di un numero, ma che quel numero si basava su informazioni rilevanti provenienti dal testo del referto. La spiegazione doveva permettere ai medici di comprendere e giustificare l’output del sistema.

Abbiamo così pensato di usare il concetto di "frame". In particolare, per il caso delle lesioni violente, abbiamo rappresentato l’evento di violenza come un frame, caratterizzato da vari elementi chiave: il perpetratore della violenza, la zona del corpo colpita (ad esempio, la faccia, che è una parte frequentemente colpita nei casi di violenza di genere), gli strumenti utilizzati, il luogo dell'incidente e altri dettagli che potessero contribuire a dare un contesto al referto. Questi elementi sono quelli che costituiscono il frame della violenza.

Sebbene non esistesse un frame predefinito per descrivere le lesioni derivanti da eventi violenti, siamo riusciti a progettare un nuovo frame, adattato alle esigenze specifiche del nostro caso. In pochi passaggi e con un apprendimento rapido, siamo riusciti a integrare il sistema di classificazione con il meccanismo esplicativo basato sui frame, ottenendo un risultato soddisfacente in tempi brevi.

I frame, quindi, si rivelano uno strumento molto potente. Essi codificano e unificano informazioni relative a situazioni o eventi specifici, raggruppando concetti ed elementi rilevanti che caratterizzano quella situazione. Ogni frame include una serie di concetti che sono legati ad esso, e la loro applicazione può essere estremamente utile per semplificare e rendere comprensibili i dati elaborati dai sistemi automatizzati, facilitando il loro utilizzo in contesti complessi come quello sanitario.

mmaginando il concetto di frame, si può pensare a un oggetto costituito da blocchi informativi, alcuni dei quali sono essenziali, mentre altri possono essere presenti solo occasionalmente. Per esempio, nel contesto della violenza, alcuni elementi come il perpetratore, lo strumento usato, il tipo di violenza e altre caratteristiche sono componenti costitutive del frame. Tuttavia, in scenari come quello dei referti di pronto soccorso, i dati relativi alla violenza sono relativamente rari, rappresentando solo circa il 5% degli accessi, il che li rende difficili da gestire e analizzare.

Questa situazione crea dei problemi nei sistemi di apprendimento automatico, poiché tali sistemi tendono a comportarsi meglio quando predicono "no" (cioè che non ci sia violenza), ma ciò può portare a una precisione alta con una recall molto bassa. Di conseguenza, la modellazione e l’approccio da utilizzare dipendono fortemente dal contesto applicativo in cui si lavora. Quando si usano i frame, l’obiettivo non è semplicemente cercare delle parole chiave, ma si tratta di un processo più articolato. Infatti, i frame non vengono individuati attraverso una ricerca per parole, ma piuttosto cercando i termini nel linguaggio che evocano il frame stesso.

In un contesto di lavoro con migliaia di frame, come ad esempio in FrameNet, diventa cruciale sapere come identificare e selezionare quello giusto. Ogni frame, infatti, è associato a una terminologia specifica che aiuta a individuare il frame pertinente in modo efficiente. La selezione del frame corretto dipende molto dall’ambito di applicazione, ma uno degli ostacoli principali in questa fase è la difficoltà di disambiguare il significato dei termini. Qui entra in gioco uno strumento chiamato "lexical unit". Una "lexical unit" è fondamentalmente un termine legato in modo univoco a un frame. Il fatto che un termine faccia parte di un determinato frame ne determina il significato in quel contesto specifico, risolvendo così la questione della disambiguazione. Ad esempio, lo stesso termine può appartenere a più di un frame, ma in ciascun contesto, il termine assume un significato specifico proprio grazie al frame di appartenenza. In altre parole, l’appartenenza a un frame implica già una disambiguazione, senza la necessità di ulteriori chiarimenti sul significato del termine in quel contesto.

Questa situazione è simile a quanto accade in WordNet, dove esistono problemi di disambiguazione che possono variare considerevolmente. Quando si deve scegliere tra sensi molto distanti, la disambiguazione è relativamente semplice, poiché le differenze tra i sensi sono evidenti. Tuttavia, quando i sensi sono simili o solo parzialmente sovrapposti, la disambiguazione diventa molto più complicata, sia per i sistemi automatici che per gli esseri umani.

In questo contesto, FrameNet si inserisce in un continuum di risorse che vanno dalla granularità elevata (e quindi più precisa) a una più generale. La risorsa è talmente dettagliata che, in alcuni casi, l’eccesso di distinzioni tra i sensi può risultare problematico. Infatti, la presenza di troppe distinzioni micro-distanziate complica ulteriormente la disambiguazione, creando difficoltà sia per i sistemi automatici che per gli esseri umani. Non è raro che si possa pensare che, in alcuni casi, non ci sia una sola etichetta di disambiguazione, ma che possano coesistere più etichette per un determinato elemento, complicando ulteriormente la gestione del linguaggio naturale.

Fondamentalmente, è l'essere umano che decide come disambiguare un termine. Se diverse persone suggeriscono più di un significato per lo stesso termine e se alcune di queste interpretazioni sono comuni, ciò evidenzia una difficoltà nella disambiguazione. Questo problema non riguarda solo le persone diverse, ma anche lo stesso annotatore. Anche se gli annotatori sono generalmente studenti universitari altamente qualificati, il riannotare un testo a distanza di tempo può portare a risposte non sempre coerenti. E questo è un tema ricorrente nelle esperimentazioni recenti, dove si è riscontrato che le interpretazioni a distanza di tempo possono differire, anche se i soggetti sono gli stessi.

In questo contesto, risorse come FrameNet e WordNet, sviluppate da linguisti esperti, sono strumenti che ci permettono di fare distinzioni molto precise, ma che spesso possono complicare l'interpretazione semantica. Se da un lato sono strumenti che consentono di cogliere sottili differenze di significato, come nel caso delle variazioni di "guadagnare", dall'altro presentano un costo significativo in termini di tempo e di attività manuale. Ad esempio, in un'analisi semantica, mentre una risorsa come FrameNet permette di rappresentare con precisione le sfumature di significato, altre tecniche più automatiche, come l’utilizzo di vettori di embedding (come per il termine "earned"), forniscono una rappresentazione più "grossolana", ma che può essere più facilmente applicata in vari contesti.

Nel caso del termine "replace", che può significare "rimettere a posto" o "sostituire", la difficoltà di disambiguazione è ancora più evidente. In scenari come questi, dove i significati sono sottilmente distinti e vicini, è difficile decidere quale sia il significato corretto senza un sistema come FrameNet che aiuti a fare chiarezza. Qui si notano le differenze tra l'approccio più diretto e semplice di WordNet, che cerca di disambiguare i significati in modo rapido, e il metodo più complesso e preciso di FrameNet, che si concentra su un'interpretazione semantica più completa e articolata.

FrameNet, infatti, non si limita a disambiguare un singolo termine all'interno di una frase, ma cerca di cogliere il significato complessivo di una frase intera, assegnando i ruoli semantici alle diverse parti della frase. Questo processo assomiglia a una sorta di "riempimento di slot", in cui gli elementi che costituiscono il frame vengono associati ai ruoli semantici previsti. Tuttavia, anche in questo caso, ci sono complicazioni: gli elementi della frase potrebbero non essere esplicitamente specificati, o potrebbero essere descritti in modo obliquo. La struttura invariabile di un frame consente infatti di mantenere il riconoscimento semantico anche in presenza di variazioni linguistiche. Quindi, mentre da una parte l’interpretazione semantica richiede una precisione elevata, dall’altra è necessaria una certa flessibilità per poter riconoscere ruoli e significati anche quando i termini non sono esplicitamente chiariti o sono espressi in forme linguistiche meno dirette.

Uno degli ostacoli principali che si incontrano frequentemente nell'analisi semantica riguarda i verbi obliqui, ossia quei verbi che riportano, asseriscono, ipotizzano o comunicano pensieri ed eventi che sono distinti dalla proposizione principale della frase che si sta analizzando. Questi verbi e i concetti ad essi collegati, come le credenze o i pensieri, richiedono una corretta interpretazione, in quanto la loro semantica non è sempre immediatamente chiara dal solo contesto della frase principale.

Per esempio, se si considera la frase "i forti venti spiegano la presenza e la diffusione dei mulini", l'uso del verbo "spiegano" ha un senso chiaro, in quanto i venti sono realmente la causa della presenza e diffusione dei mulini. Ma in altre situazioni, come nella frase "il vento non spiega", il significato del verbo "spiegare" diventa meno evidente. Allo stesso modo, nella frase "la sua assenza agli incontri, alle riunioni suggerisce che è scontento del proprio lavoro", l'assenza di qualcuno non è un evento che spiega qualcosa in modo diretto, ma il verbo "suggerire" implica una relazione più sottile tra l'assenza e il sentimento di insoddisfazione.

In queste situazioni, la comprensione della semantica diventa più complessa, ma se si fa uso di un frame, è possibile ricostruire il contesto in cui determinati significati possono essere assegnati anche a frasi che, sebbene possano sembrare semplici, nascondono una struttura semantica articolata e difficile da interpretare senza un modello appropriato.

L'idea della stratificazione dei livelli di analisi semantica implica che per affrontare correttamente analisi a livelli più alti, sia necessario comprendere e risolvere prima quelli più bassi. I frame, infatti, consentono di astrarre e di operare su contesti che possono contenere strutture complementari, come quelle che appaiono in alcune tradizioni della linguistica computazionale, che si riferiscono a frame di sottocategorizzazione differenziati. Un esempio di questa complessità è il verbo "lamentarsi". Esistono diversi sensi di questo verbo, come il caso in cui un paziente si lamenta di un sintomo, come il mal di schiena, rivolgendosi al medico, e il caso in cui qualcuno si lamenta con un'altra persona, come nel contesto lavorativo con un superiore. La differenza tra questi due usi del verbo è semantica, ma la struttura del verbo è altrettanto importante: non si può semplicemente "lamentarsi di qualcosa", ma si deve considerare anche il ruolo di chi si lamenta "con qualcuno".

Nel caso della lingua inglese, la stessa struttura semantica è mantenuta, come nel caso della frase "she complained", dove l’oggetto delle lamentele può essere seguito da una proposizione che esplicita il contenuto della lamentela. Questo tipo di strutture complesse, che ricordano le reti proposizionali, vengono interpretate con l'ausilio dei frame, i quali evitano la necessità di costruire esplicitamente grafi complessi o calcolare strutture intricate. Invece di creare una rappresentazione grafica, l'approccio si concentra sull’identificazione del frame giusto e sull'inserimento delle varie componenti semantiche nei rispettivi "slot" del frame.

Inoltre, un aspetto importante della semantica che i frame riescono a catturare è la gestione delle omissioni, ossia quelle forme di ellissi che si verificano quando un argomento non è esplicitamente menzionato nella frase. Un parser che si occupa di analizzare una frase con elementi mancanti può avere delle difficoltà, ma il frame può compensare queste difficoltà, permettendo comunque di riconoscere il ruolo semantico degli argomenti. Per esempio, se qualcuno dice "ho già dato il mio contributo", senza specificare cosa sia stato dato, il contesto aiuta a interpretare correttamente il significato della frase. Se invece si dice "prendi un regalo per tua figlia", anche senza specificare il regalo, il frame relativo al compleanno o allo scambio di regali permette comunque di attribuire correttamente il significato.

Altri fenomeni linguistici che sfuggono a una comprensione immediata sono legati ai derivati nominali di verbi, ovvero quei sostantivi che derivano da verbi e che mantengono il senso dell'azione. Come notato in precedenza, questi sostantivi, che sono spesso considerati nominalizzazioni, si riferiscono a concetti che derivano da verbi, ma la loro interpretazione richiede la capacità di risalire all’azione originale e comprenderne il significato semantico nel contesto.

Tutti questi fenomeni costituiscono esempi concreti delle difficoltà che si incontrano nell’analisi semantica.

Si pensi, ad esempio, al verbo da cui derivano "adesione" e "aderenza". Entrambe le forme provengono dallo stesso verbo, ma nel passaggio alla forma nominale assumono significati profondamente diversi: "adesione" si riferisce a un concetto astratto, come l'adesione a una credenza o a un ideale, mentre "aderenza" descrive un fenomeno concreto, come l'attaccarsi di un cerotto alla pelle. Un fenomeno analogo si riscontra nel contrasto tra "osservanza" e "osservazioni", dove i significati divergono nonostante la comune origine lessicale.

Un altro esempio interessante è rappresentato dal termine inglese *commitment*, che non ha una traduzione diretta in italiano. Esso può riferirsi all’impegno ideale verso una causa, come in *committing to a cause*, o all’atto concreto di commettere un crimine, come in *committing a crime*. Anche qui si osserva uno slittamento di significato, tra ambiti astratti e concreti, che complica ulteriormente l'analisi automatica. Il passaggio dall'astratto al concreto è, infatti, un tema che meriterebbe di essere approfondito a parte.

Strumenti come FrameNet risultano particolarmente utili proprio per gestire queste differenze di significato e i meccanismi di variazione semantica. A titolo di esempio, si può considerare il frame della vendetta (*revenge*), uno dei primi sviluppati nella tradizione di Fillmore e Baker, linguisti della scuola di Berkeley che hanno introdotto il formalismo dei frame e creato la relativa risorsa.

Il frame della vendetta si struttura attorno ad alcuni elementi fondamentali: vi è un individuo A che ha arrecato un danno a un individuo B; successivamente, B si vendica su A. La vendetta implica una reazione al torto subito, ma soprattutto si caratterizza per avvenire al di fuori di qualsiasi canale legale o istituzionale. Non si tratta di una causa giudiziaria, ma di un atto diretto, spesso violento, consumato senza mediazioni ufficiali.

La prima operazione utile nell’analizzare questo frame consiste nell'individuare, in maniera articolata e per categorie grammaticali, i vari elementi che evocano e costituiscono il frame stesso. Si può cominciare ragionando sui possibili verbi associati al concetto di vendetta. Storicamente, questa attività è stata svolta da linguisti, ma nulla vieta di pensare a procedure automatiche per apprendere questo vocabolario di base.

Una possibile metodologia consiste nel selezionare un termine seme, ad esempio un verbo tipico della vendetta, e analizzare i contesti in cui esso ricorre. Da questa analisi si possono estrarre verbi, nomi e aggettivi frequentemente cocorrenti, studiando non solo la presenza di certe parole, ma anche la loro struttura sintattica, osservando per esempio le dipendenze grammaticali nei parse tree, dove il verbo selezionato agisce come radice.

Lo stesso principio vale anche per gli aggettivi e può essere esteso alle espressioni complesse, come sintagmi verbali o nominali. L'uso di espressioni complesse, anziché limitarsi a singoli token, ha un duplice vantaggio: da una parte, si avvicina maggiormente all’uso reale della lingua, dall'altra, consente di riconoscere blocchi significativi che rappresentano in modo tipico un certo frame. È proprio grazie a questo tipo di approccio che diventa possibile costruire dizionari evocativi dei frame, più ricchi e più fedeli al linguaggio naturale.

Il primo passaggio nello sviluppo di un frame consiste nella costruzione di un vocabolario descrittivo per i suoi componenti principali, che vengono chiamati **frame elements**. I frame elements rappresentano i ruoli semantici o, come si è detto in precedenza, gli slot all'interno della struttura del frame. Ogni frame è dunque articolato in diversi slot, ciascuno dei quali corrisponde a un frame element specifico.

I nomi dei frame elements vengono poi utilizzati per etichettare i costituenti delle frasi che sono associate a quel frame. L’obiettivo finale è creare una struttura ben definita in cui vi sia una descrizione testuale chiara e sintetica del frame — costituita da poche frasi che delineano il contesto — seguita da un elenco organizzato dei frame elements.

Nella risorsa FrameNet, i frame elements sono suddivisi in **core** e **non core**. I core elements sono quelli fondamentali per la definizione del frame stesso. Prendendo come esempio il frame della vendetta, è evidente come elementi quali il "vendicatore" (avenger) e l'"offender" — colui che ha causato il danno iniziale — siano essenziali: senza la presenza di questi ruoli, difficilmente si potrebbe parlare di una situazione di vendetta.

Dal punto di vista dell'elaborazione del linguaggio naturale (NLP), i frame elements sono ruoli semantici molto precisi. A differenza di schemi semantici più generali, in cui si ricorre a ruoli come "agente" o "paziente", in FrameNet si utilizza una terminologia specifica che aumenta notevolmente il livello di informazione disponibile. La scelta di etichettare i ruoli in modo così dettagliato testimonia l'intenzione degli autori di fornire una risorsa dotata di grande precisione semantica.

Una volta definita la lista dei ruoli semantici, il passo successivo consiste nell’estrarre esempi reali da un corpus. Si selezionano frasi autentiche nelle quali si identificano e si annotano non solo i termini che evocano il frame, ma anche il ruolo semantico che ciascun termine svolge. Questo tipo di annotazione è particolarmente importante perché stabilisce un collegamento diretto tra la funzione sintattica di un costituente e il suo ruolo semantico. Si parte dal presupposto che effettuare il parsing sintattico di una frase sia generalmente più semplice rispetto alla sua interpretazione semantica e che quest'ultima possa essere successivamente inferita proprio a partire dalle informazioni ottenute tramite il parsing. L'annotazione, dunque, rappresenta un ponte che consente sia agli esseri umani sia ai sistemi automatici di apprendere e ispezionare le corrispondenze tra sintassi e semantica.

si è osservato come la struttura della risorsa FrameNet sia composta da due elementi principali: da un lato vi è la definizione del frame con i relativi frame elements, dall’altro un insieme di esempi annotati. Guardando al frame della vendetta, si trova una descrizione sintetica seguita da un elenco dei core elements — come l'avenger e l'offender — e da numerosi non core elements, come il grado dell'evento, la durata, lo strumento, la maniera e altri.

Un aspetto fondamentale di FrameNet è che, negli esempi annotati, ciascun costituente della frase è etichettato con il ruolo semantico specifico, ossia con il nome preciso del frame element a cui appartiene. Questo tipo di annotazione è di valore inestimabile, poiché consente di osservare come la semantica si sovrapponga e si organizzi sopra la struttura sintattica di una frase.

Quanto al corpus utilizzato per costruire FrameNet, si è fatto affidamento principalmente sul **British National Corpus** (BNC), una raccolta di circa 100 milioni di parole. A questa base sono stati aggiunti altri documenti provenienti da fonti diverse, portando il totale a circa 200 milioni di parole. Il BNC, rilasciato dal **Linguistic Data Consortium**, è una risorsa di altissimo valore, ma anche molto costosa, poiché le iniziative di questo tipo richiedono investimenti significativi per la loro realizzazione e manutenzione.

Il corpus di FrameNet, pur non essendo enorme se confrontato con risorse più recenti come l'intero dump di Wikipedia o i dataset utilizzati per addestrare modelli transformer, rappresenta comunque il frutto di uno sforzo straordinario, sia in termini di energie umane sia di risorse economiche. La sua peculiarità risiede nel fatto che l'annotazione è stata effettuata manualmente, il che conferisce al corpus un valore qualitativo altissimo.

L’annotazione degli esempi segue un procedimento piuttosto preciso. Si seleziona un verbo, individuandolo insieme ai vari possibili frame elements associati. L'interfaccia di annotazione mostra l'elenco dei frame elements e, accanto, le **lexical units**, cioè le parole o espressioni che veicolano i diversi ruoli semantici all'interno del frame.

Prendendo ad esempio il verbo *avenge*, che è centrale nel frame della vendetta, l'annotazione avviene evidenziando le espressioni della frase che corrispondono ai diversi ruoli semantici. Nell’interfaccia, i ruoli semantici sono rappresentati visivamente con colori differenti, che permettono di distinguere rapidamente le varie funzioni pur nella grafica un po' datata dello strumento.

Considerando la frase:  
*"With this El Cid at once avenged the death of his son and once again showed that any attempt to reconquer Valencia was fruitless while he still lived"*,  
si osserva che:

* **With this** viene annotato come *Punishment*, ossia l'atto di punizione inflitto. Qui è importante notare che "with this" è un complemento che un parser sintattico sarebbe in grado di riconoscere, e il fatto che venga associato al ruolo semantico di strumento indica una connessione tra struttura sintattica e semantica.
* **El Cid** è marcato come *Avenger*, il vendicatore. Sintatticamente, è il soggetto della frase, e sapere che il soggetto spesso coincide con l’avenger aiuta a costruire modelli di interpretazione automatica.
* **The death of his son** viene identificato come *Injury*, ovvero l'offesa o il danno subito, e sintatticamente si presenta come l'oggetto del verbo "avenged".

La disponibilità di queste annotazioni consente analisi retrospettive estremamente preziose. È possibile apprendere dai dati marcati quali realizzazioni sintattiche sono associate a determinati ruoli semantici, migliorando così la capacità di un sistema di NLP di interpretare nuovi testi.

In questa prospettiva, i frame si configurano come strutture astratte che descrivono eventi o situazioni, mentre il corpus annotato offre esempi concreti che popolano queste strutture. Tale rappresentazione permette di osservare e studiare fenomeni come la **variazione di Valenza**: vale a dire, le diverse modalità con cui un verbo può combinarsi con i propri argomenti a livello sintattico.

La Valenza, come concetto, viene mutuata dal linguaggio della chimica, dove indica la capacità di un elemento di legarsi con altri elementi. Analogamente, nella linguistica, la Valenza descrive il numero e il tipo di argomenti che un predicato può reggere. Si possono avere predicati con Valenza nulla (verbi impersonali), predicati intransitivi, transitivi o persino tritransitivi, come nell'esempio inglese *"He gave him a book"*, dove il verbo "gave" governa due oggetti: uno che corrisponde a un complemento di termine ("him") e uno che è l'oggetto diretto ("a book").

In sintesi, FrameNet fornisce uno strumento unico che non solo rappresenta semanticamente il linguaggio naturale in maniera estremamente dettagliata, ma consente anche di esplorare le interazioni profonde tra sintassi e semantica.

Nella seconda parte della lezione si propone un esempio semplice di come i frame di FrameNet possano essere utilizzati in un'applicazione per la comprensione testuale. Sebbene esistano oggi numerosi prodotti commerciali maturi per la comprensione del testo, l'esempio scelto riguarda un estrattore di notizie: dato un breve testo narrativo, si individuano gli elementi salienti, si riconducono a un frame specifico e si analizza come le necessità semantiche espresse da quel frame si combinano con le frasi del testo.

Per semplicità, si è scelto un testo tratto da un articolo di cronaca evenemenziale, che descrive eventi concreti e sequenziali, come un arresto, un processo, un'aggressione. L'analisi di testi di questo tipo risulta più agevole rispetto a quella di testi di economia o politica monetaria, che richiederebbero una trattazione semantica più complessa.

In questo esempio si immagina di disporre già di strumenti complementari pienamente funzionanti: parser sintattici affidabili, risolutori di anafore, moduli di disambiguazione semantica e di composizione semantica, oltre a sistemi per la scelta tra diverse analisi sintattiche possibili. Inoltre, si dà per acquisita la possibilità di attivare un lessico strettamente connesso ai temi espressi nei frame.

Il testo analizzato riguarda Robert Pickett, sospettato di aver compiuto un attentato nei pressi della Casa Bianca. I paragrafi riportano le seguenti informazioni:

* Pickett è stato arrestato mercoledì e trattenuto senza cauzione.
* Un magistrato federale lo ha informato delle accuse, che comprendono l'assalto a un ufficiale federale con un'arma mortale, reato punibile fino a dieci anni di carcere.
* È stata fissata un'udienza per il martedì successivo e la cauzione è stata nuovamente negata.
* Pickett, ferito al ginocchio dai servizi segreti dopo aver presumibilmente sparato due colpi nei pressi della Casa Bianca, è comparso in aula su delle stampelle.
* Non ha rilasciato dichiarazioni.

Il primo passo nell'analisi è l'elaborazione dell'albero sintattico a dipendenze, che permette di individuare le relazioni grammaticali tra i vari elementi della frase. Questa struttura può essere rappresentata sia in forma tabellare sia grafica, dove diventa più intuitivo riconoscere la struttura della frase. Nell'esempio, l'albero a dipendenze mostra chiaramente la sequenza degli eventi: Robert Pickett, identificato come presunto attentatore, viene arrestato, tradotto in tribunale e trattenuto.

Immagine che contiene testo, diagramma, linea, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Dal punto di vista della costruzione narrativa, il testo segue un modello standard nella cronaca giudiziaria anglosassone:

* La prima frase annuncia l'arresto.
* La seconda informa delle accuse.
* La terza descrive l'organizzazione dell'udienza.
* La quarta e quinta forniscono dettagli sulla condizione dell'accusato e il suo comportamento in aula.

Attraverso l'analisi sintattica e semantica, è possibile ricondurre ciascun evento descritto ai relativi frame e frame elements previsti da FrameNet, dimostrando così come questa risorsa possa essere utilizzata concretamente per la comprensione e l'estrazione di informazioni dai testi.

La prima operazione effettiva consiste nell’individuazione degli elementi lessicali rilevanti. Nella pratica, infatti, l'output di un comune word tokenizer, come quelli disponibili in librerie NLP ampiamente utilizzate, restituisce spesso frammenti di testo poco significativi.

La selezione dei termini più evocativi e rilevanti è dunque essenziale. Oggi, grazie a tecniche come gli embeddings – ad esempio GloVe, Word2Vec, FastText – è possibile svolgere questa selezione in modo piuttosto efficiente, soprattutto se si dispone di modelli addestrati su un dominio specifico, come quello della cronaca giudiziaria. Estendere il procedimento a domini più generali risulta invece più complicato, ma comunque affrontabile con strumenti adeguati.

Il motivo per cui si insiste sull'importanza dei termini evocativi è che ogni termine ha il potenziale di attivare un frame diverso. È quindi fondamentale individuare i termini che portano il massimo peso semantico, quelli che meglio consentono di risalire al frame o ai frame implicati nell’interpretazione della frase o del testo. Non esiste un metodo standard imposto da FrameNet per questo passaggio: ciascuno può implementare la propria strategia di individuazione dei frame. Tuttavia, qualsiasi algoritmo di accoppiamento deve seguire una logica di base: da un lato analizzare le necessità semantiche previste da ciascun frame, dall'altro cercare di verificare in che misura l'input testuale soddisfa tali necessità. A seconda della strategia adottata, possono essere selezionati uno o più frame, privilegiando quelli i cui core elements risultano maggiormente riempiti dall’input. In alcuni casi si può assegnare un peso maggiore a certi frame elements ritenuti più cruciali.

Al termine dell'operazione di mapping, sulla base di una metrica scelta, viene selezionato il frame che meglio si adatta alla frase corrente. L’algoritmo può anche optare per mantenere più alternative in una piccola cache di frame candidati, rimandando la decisione definitiva a una fase successiva. Se si dispone di embeddings, la scelta può essere guidata anche dalla distanza semantica tra i frame e l’input.

Nel caso della seconda frase analizzata, il verbo dominante è "informed", che attiva un frame in cui è prevista una situazione di comunicazione tra due persone: un comunicatore e un destinatario. Dal punto di vista sintattico, come si evince anche dall'analisi dei corpora annotati, il comunicatore – che si potrebbe considerare, seppur impropriamente, l'agente del frame – è normalmente espresso come soggetto della frase. Il destinatario segue subito dopo, mentre il contenuto del messaggio viene spesso introdotto tramite una prepositional phrase, come nel caso di "of the charges against him". Questo schema ricorrente consente di procedere con un certo grado di sicurezza nell’individuazione dei ruoli semantici all'interno della frase.

Tuttavia, la realtà è sempre più articolata di quanto si possa pensare inizialmente. Spesso si verifica una compresenza o una stratificazione di più frame all'interno della stessa porzione testuale, rendendo non immediato stabilire quale elemento appartenga a quale frame. Nel caso in esame, oltre al frame della comunicazione, emerge anche quello legato alla procedura giudiziaria, come la formalizzazione delle accuse. La funzione della corte, infatti, prevede anche di informare ufficialmente l'imputato delle accuse a suo carico.

Pertanto, elementi come "un magistrato federale ha informato Pickett delle accuse" devono essere riconosciuti, estratti e assegnati ai rispettivi slot per completare l'interpretazione semantica di quella sezione di testo. A ciò si aggiunge che, per esempio, la figura del magistrato federale può essere disambiguata automaticamente grazie al frame di appartenenza: il termine "magistrato", in quel contesto, non si presta a equivoci sul suo ruolo.

Analogamente, termini fortemente polisemici come "charge" – che può indicare un’accusa, una carica ufficiale o addirittura un importo economico – vengono interpretati correttamente solo all’interno di un frame che garantisca coerenza semantica. Il criterio di coerenza è fondamentale: permette di unificare logicamente i diversi pezzi del discorso. Era un concetto già emerso in una lezione precedente, in cui si era osservato come una narrazione coerente (andare al ristorante, ordinare, pagare) si distingua radicalmente da un racconto senza filo logico (incontrare un nano, prendere un topolino) pur avendo complessità sintattica simile.

Un’altra questione rilevante riguarda l’interpretazione di unità linguistiche complesse come "White House", nomi propri di persona, o entità come "Secret Service". Questi termini non dovrebbero essere considerati isolatamente, ma trattati come blocchi unitari. Allo stesso modo, anche sintagmi come "held without bond", "assaulting federal officer" o "preliminary hearing" devono essere riconosciuti come unità concettuali. Qui FrameNet può offrire un supporto significativo, ma laddove un frame specifico manchi, è necessario ricostruirne uno che garantisca la coerenza semantica.

Molti parser moderni offrono già strumenti utili, come la segmentazione in "stanze" o l'identificazione automatica delle collocazioni, che facilitano l’individuazione di questi blocchi di significato.

Infine, si inserisce un ulteriore livello di complessità: il frame dell’evidentiality. In questo testo è riconoscibile grazie alla ripetizione di espressioni come "presunto" o "a quanto si dice", tipiche del linguaggio giornalistico. Questo meccanismo serve a insinuare un dubbio o a segnalare l’incertezza dei fatti riportati, in quanto, fino a una condanna definitiva, l'accusato deve essere considerato innocente. Questo frame non riguarda direttamente l’evento, ma il modo in cui l’evento viene narrato, aggiungendo così un livello di meta-narrazione.

Il testo, suddiviso in quattro paragrafi più una frase finale, attiva dunque numerosi frame: il frame della corte, che include elementi come giudice, sala del tribunale, processo penale; il frame del confinement, relativo alla detenzione preventiva; il frame della comunicazione; quello della formalizzazione delle accuse; e infine il frame dell’evidentiality.

Tutto questo mostra anche come i frame siano organizzati in una gerarchia: il processo criminale, ad esempio, comprende sotto-frame come la comparizione in tribunale (court appearance), l’arresto, e la formulazione delle accuse (arraignment), a dimostrazione che FrameNet non è una raccolta disordinata di concetti, ma un vero e proprio sistema strutturato.

**Lezione 6**

Dopo le necessarie interruzioni, è stato ripreso il filo del discorso affrontato nelle lezioni precedenti, che aveva portato all’introduzione dei metodi per rappresentare i sensi delle parole e combinarli. In particolare, era stata presentata WordNet come risorsa semantica di riferimento, ed erano stati avviati alcuni esercizi sperimentali per esplorarne il funzionamento.

Una delle prime attività sperimentali ha avuto come obiettivo l’analisi della *prossimità semantica* tra sensi. Nella prima parte dell’esercitazione, si è utilizzata una modalità semplificata – definita *crude* – in cui la prossimità tra due sensi veniva calcolata come la lunghezza del cammino che li separa all’interno della struttura gerarchica di WordNet. Si trattava, in sostanza, di valutare quanto certi accoppiamenti di sensi potessero risultare problematici da distinguere, partendo dalla loro collocazione nella rete semantica.

La seconda parte dell’esercitazione si è concentrata sull’uso di un dataset specifico, costituito da frasi tratte dal *British National Corpus*, noto come *TenCore*, in cui ogni termine era già stato disambiguato. Questo corpus era stato arricchito con etichette di senso tratte da WordNet (denominate *SenseTag*), consentendo così di lavorare su dati già marcati. L'obiettivo era disambiguare un termine dato all’interno di una frase, analizzandone il contesto linguistico.

Il metodo seguito prevedeva di campionare un certo numero di frasi, individuare in ciascuna un termine target e, grazie all’analisi del contesto, assegnare il senso corretto tra quelli possibili. Il compito di disambiguazione era stato svolto con un algoritmo denominato *Simplify Place*, un sistema piuttosto essenziale ma efficace, basato sull’intersezione dei contesti: due sensi risultano tanto più vicini quanto maggiore è il numero di parole che i loro contesti condividono, sia nei glossari di WordNet sia nella frase in cui compaiono.

Successivamente, l’attenzione si è spostata su *FrameNet*, una risorsa semantica che si fonda su un paradigma completamente diverso rispetto a WordNet. Mentre quest’ultima struttura i sensi in una gerarchia di concetti atomici – in particolare, per i sostantivi, una multigerarchia che parte da 25 *supersensi* – FrameNet adotta un approccio di tipo frame-based. In WordNet, il compito semantico è ricondotto all’individuazione del punto esatto in cui un senso si colloca all’interno dell’albero, mentre in FrameNet si tratta di completare un frame, ossia una struttura semantico-concettuale, riempiendo i suoi *slot* con i costituenti appropriati.

I frame sono costituiti da due componenti principali: da un lato ci sono i *frame elements*, che rappresentano i ruoli semantici coinvolti nella situazione descritta; dall’altro, c’è un insieme di *lexical units*, ovvero i termini che, appartenendo a quel frame, risultano automaticamente disambiguati nel contesto del frame stesso. La risorsa FrameNet è organizzata in modo bipartito: non comprende soltanto un inventario di frame predefiniti, ma include anche un corpus annotato in cui le occorrenze linguistiche sono marcate sia sintatticamente sia semanticamente. Questa annotazione a più livelli consente di osservare le corrispondenze tra le realizzazioni sintattiche (come il soggetto, l’oggetto, ecc.) e i ruoli semantici previsti dai frame.

Si vuole approfondire il funzionamento di *FrameNet* e, in particolare, di esplorare in che modo questa risorsa possa essere fatta interagire con *WordNet.* Questa integrazione viene proposta attraverso la possibilità di mappare gli elementi costitutivi dei frame — ossia i ruoli semantici — su identificatori univoci (synset) già presenti all’interno di WordNet. A tal fine, si è previsto l’impiego di due strategie: la prima si fonda su un criterio di intersezione di contesti, in maniera simile all’approccio utilizzato dall’algoritmo di Lesk, il quale si basa sull’analisi della sovrapposizione tra i contesti lessicali associati ai sensi delle parole. La seconda strategia, invece, è di tipo strutturale e grafico: si costruiscono dei sottoalberi della gerarchia semantica di WordNet, e si analizzano le distanze tra i nodi, con l’intento di interpretare tali distanze come indicatori probabilistici di affinità semantica. Sebbene non ci sia un vincolo rigido rispetto al linguaggio di programmazione da utilizzare, dal punto di vista operativo si è fatto riferimento a un insieme di strumenti che compongono un’interfaccia funzionale, tra cui il modulo *Frame*, reso disponibile all’interno della libreria *NLTK*. In termini computazionali, si potrebbe idealmente ipotizzare che una tale operazione abbia un costo costante, una volta completati i necessari passaggi preliminari — come il caricamento della frase e la predisposizione delle risorse. Se poi si intende procedere alla disambiguazione di un secondo termine (T2), si hanno almeno due opzioni: trattare T2 in maniera indipendente da T1, oppure fare in modo che la disambiguazione di T2 tenga conto del senso già assegnato a T1. Quest’ultima opzione apre la strada a un tipo di elaborazione più sofisticata, che prende in considerazione l’insieme di scelte precedenti come parte del contesto utile a interpretare i termini successivi. Un tale approccio introduce inevitabilmente una maggiore complessità: il costo computazionale non è più costante o lineare rispetto al numero di termini da disambiguare, ma tende a crescere secondo un andamento quadratico, legato alla profondità del grafo semantico — definito qui come *triangle* — che si viene costruendo man mano che l’analisi avanza. Questa riflessione è stata collegata a un’intuizione più ampia e realistica: l’interpretazione di un testo, sia da parte di un sistema automatico sia di un lettore umano, raramente procede in maniera perfettamente lineare. Piuttosto, si tratta di un processo dinamico e ricorsivo, nel quale si ritorna su elementi già incontrati, rivedendo le scelte interpretative alla luce di nuove informazioni. È il caso, ad esempio, di quando un termine illuminante ci costringe a riconsiderare il significato globale della frase o del paragrafo: si torna indietro, si rivalutano i passaggi e si ricostruisce un’interpretazione coerente. Questo meccanismo, che avvicina il funzionamento della disambiguazione al *brain tracking* e alla revisione del contesto, risulta affascinante ma anche più oneroso da implementare.

La libreria *NLTK*, che costituisce oggi uno standard di riferimento per la sperimentazione e l’insegnamento nell’ambito dell’NLP.

Per poter utilizzare operativamente *FrameNet* all’interno di un ambiente di lavoro, è sufficiente scaricarlo attraverso il modulo fornito nella libreria *NLTK*. Non è necessario compiere operazioni complesse: l’interfaccia messa a disposizione è estremamente semplice e consente di svolgere numerose attività con pochissimi comandi.

Uno degli aspetti interessanti è la possibilità di cercare i *frame* utilizzando espressioni regolari oppure nomi precisi. Nel caso si conosca già il nome esatto del frame desiderato, la ricerca è immediata. Tuttavia, nelle applicazioni reali non è comune avere a disposizione l’elenco completo dei frame possibili, e dunque risulta più utile eseguire ricerche flessibili, capaci di individuare un insieme di possibili candidati tra cui selezionare quelli più rilevanti. Questo approccio si rivela particolarmente vantaggioso nella fase di identificazione del frame o dei frame più adatti per analizzare una data frase. È infatti fondamentale, per l’elaborazione semantica, essere in grado di associare correttamente le frasi a frame pertinenti. La ricerca può essere estesa anche ai *frame element* e alle *lexical unit*, sempre con la medesima logica, sfruttando espressioni regolari che rendono il processo più flessibile e adattabile alle varie esigenze linguistiche. Accanto a queste funzionalità principali, *FrameNet* fornisce anche accesso a ulteriori elementi come i *semantic types* e le *relazioni tra frame*. Sebbene questi ultimi non vengano utilizzati in modo diretto nelle esercitazioni iniziali, sono comunque significativi poiché evidenziano come i frame siano inseriti in strutture gerarchiche. Nelle lezioni precedenti è stato infatti illustrato il meccanismo di eredità tra frame, dove alcuni elementi vengono trasmessi dai frame più generali a quelli più specifici. Questo tipo di struttura gerarchica è importante per comprendere meglio le relazioni semantiche tra concetti affini. È stato quindi mostrato come i frame possano essere ricercati attraverso semplici stringhe: per esempio, un termine come “rumore” può fungere da chiave per estrarre tutti i frame che lo contengono. Questo consente di focalizzare l’analisi su elementi rilevanti della frase in esame. In alternativa, si può utilizzare una ricerca più avanzata, sfruttando espressioni regolari case-insensitive, che permettono di aggirare varianti morfologiche e flessioni.

In un esempio concreto, è stato mostrato come la ricerca del termine “medico” restituisca frame come *Medical Conditions* o *Medical Instruments*, a dimostrazione della varietà di contesti in cui un termine può apparire. Se poi si conosce già il nome specifico del frame o addirittura il suo identificatore numerico, è possibile accedere direttamente ai relativi contenuti. Una volta ottenuto il riferimento al frame, si ha accesso immediato a tutti i suoi campi, inclusa naturalmente la definizione, che costituisce un punto di partenza fondamentale per qualunque operazione di analisi semantica.

Questo aspetto risulta interessante e utile dal punto di vista dell’analisi semantica, perché consente di accedere immediatamente a un contesto d’uso semanticamente vicino a quello della frase in esame. Si tratta, quindi, di uno strumento fondamentale per costruire un *bag of words* da utilizzare successivamente in operazioni di intersezione, come ad esempio nella disambiguazione dei significati lessicali.

Nel caso specifico trattato a lezione, l’esempio ruota attorno al frame *Medical Specialties*, ma la metodologia è generalizzabile. Ogni frame, infatti, mette a disposizione una serie di informazioni rilevanti: l’URL di riferimento, la definizione, gli eventuali *semantic types* e le relazioni con altri frame. Queste relazioni permettono di esplorare la rete di frame attraverso i legami semantici che li connettono. Ad esempio, è stato osservato che *Medical Specialties* è utilizzato all’interno del frame *Medical Interaction Scenario*. Questo tipo di connessione può sembrare di utilità limitata in un contesto generale, ma diventa rilevante se si lavora all’interno di un’ontologia o di un sistema descrittivo formalizzato del dominio applicativo. In quel caso, diventa importante estendere e posizionare correttamente i nuovi frame nella gerarchia, sfruttando appieno queste relazioni semantiche.

Un altro aspetto evidenziato riguarda le *lexical units* associate a un frame: si tratta di tutte le espressioni linguistiche che attivano o evocano quel determinato frame. Nell’esempio del frame *Medical Specialties*, sono elencate numerose discipline mediche — dalla cardiologia alla dermatologia — a dimostrazione della ricchezza terminologica che può essere associata a un solo frame.

Sono inoltre accessibili i *frame element*, distinti tra quelli *core*, ossia indispensabili per l’identificazione del frame, e quelli opzionali. Ad esempio, nel caso in questione, *specialty* è un elemento core, mentre altri possono comparire o meno senza compromettere la riconoscibilità del frame. Sono inoltre accessibili i *frame element*, distinti tra quelli *core*, ossia indispensabili per l’identificazione del frame, e quelli opzionali. Ad esempio, nel caso in questione, *specialty* è un elemento core, mentre altri possono comparire o meno senza compromettere la riconoscibilità del frame.

Dopo averne discusso la struttura teorica, è utile analizzare nel dettaglio come sia effettivamente composto un frame secondo l'interfaccia offerta da FrameNet. L'organizzazione delle informazioni è piuttosto chiara e accessibile, grazie all’utilizzo di strutture dati semplici, come i dizionari, che permettono di accedere agevolmente ai vari componenti.

Ogni frame è identificato da alcuni attributi fondamentali: il *name*, cioè il nome del frame, la sua *definizione* descrittiva e l’*ID*, un codice identificativo univoco. A questi si affiancano i *semantic types*, anch’essi dotati di un nome e di un proprio identificatore, che contribuiscono a caratterizzare ulteriormente il significato del frame. Un altro elemento centrale sono le *lexical units*, ovvero le espressioni linguistiche che possono evocare quel frame in un testo, e i *frame elements*, che costituiscono le componenti semantiche da riempire per attivare il frame in modo completo.

Dal punto di vista strutturale, quindi, un frame ha una forma piuttosto semplice a livello superficiale, ma la complessità si manifesta nei suoi componenti interni. I *frame elements*, ad esempio, sono essi stessi oggetti articolati: ognuno di essi possiede un nome, una definizione, un ID, e può presentare abbreviazioni, una classificazione come *core* o meno (a seconda che sia elemento costitutivo del frame o opzionale), e anche dei tipi semantici, anch'essi corredati di nome e ID. Alcuni *frame elements* possono inoltre includere meccanismi di controllo o restrizione semantica, come l’obbligatorietà o l’esclusione di altri *frame elements*, oppure relazioni gerarchiche che funzionano in modo simile alla sottocategorizzazione sintattica, ma in chiave semantica.

Sono disponibili anche le relazioni tra frame, che consentono di esplorare la rete semantica nella quale ogni frame è inserito. Per esempio, è stato mostrato come il frame *Medical Interaction Scenario* faccia uso del frame *Medical Specialties*, evidenziando una relazione di dipendenza semantica.

n’ulteriore funzionalità utile dell’interfaccia è la possibilità di cercare i frame sulla base delle *lexical units*, ovvero delle parole o espressioni concrete che compaiono nei testi. Questo è particolarmente rilevante, poiché normalmente non si conosce a priori quale frame sia evocato da una certa frase o da un termine specifico. Disporre di uno strumento che consente la ricerca tramite parole chiave — anche parziali — permette di identificare con facilità un insieme di frame candidati pertinenti.

Ad esempio, effettuando una ricerca per la radice “epidemiol”, il sistema è in grado di ricondurre questo lemma al frame *Medical Specialties*. All’interno di tale frame, infatti, esiste una lexical unit associata all’epidemiologia, disciplina medica che si occupa dello studio delle epidemie. È proprio questa connessione semantica che giustifica la restituzione di quel particolare frame come risultato della ricerca.

Le *lexical units* meritano un approfondimento: esse rappresentano l'accoppiamento tra un’espressione linguistica e un significato. Per esempio, il frame *Apply\_heat* descrive una situazione tipica in cui viene applicato calore a del cibo tramite uno strumento. Questo frame è evocato da verbi come *bake*, *blanch*, *boil*, *broil*, e così via. Tali verbi sono appunto le lexical units che attivano il frame. In caso di termini polisemici, ogni significato distinto corrisponde a una lexical unit separata. È importante notare, tuttavia, che all’interno di un singolo frame, una lexical unit non è mai ambigua: il suo significato è univoco. Questa caratteristica rende del tutto naturale e intuitivo l’accostamento di ciascuna lexical unit a un identificatore di senso, come un synset di WordNet. Anche se un termine come *bake* può avere più accezioni in generale, ogni sua occorrenza in un frame specifico corrisponde sempre a un significato unico. Questo è il fondamento dell’esercitazione proposta: attribuire un identificatore semantico preciso agli elementi di FrameNet. L’assenza di questa associazione nella versione originale dello strumento è riconducibile al fatto che, per i suoi sviluppatori, sarebbe stato un compito aggiuntivo molto oneroso, non ritenuto prioritario rispetto agli obiettivi iniziali del progetto. Uno degli svantaggi dell’algoritmo di Lesk, come già osservato, è rappresentato dalla difficoltà di raccogliere a mano un contesto sufficientemente ricco di termini distintivi. L’approccio qui adottato consente invece di ottenere in modo semplificato molte informazioni di natura lessicale che permettono di identificare e caratterizzare sia i frame sia i loro elementi. Per ciascun *frame element* si ha accesso alla definizione, agli eventuali tipi semantici associati, e ad altri metadati come abbreviazioni o codifiche interne. Questo consente di costruire una base lessicale molto più articolata, su cui poter poi impostare attività di elaborazione semantica.

Nel processo di esplorazione dei frame, gli elementi vengono trattati separatamente: da una parte i *frame elements*, dall’altra le *lexical units*. Questo avviene attraverso l’accesso diretto alle chiavi specifiche presenti nei dizionari ritornati dall’interfaccia. Prendendo ad esempio il frame *Medical Specialties*, si nota come sia facile raccogliere molte informazioni utili, sia per quanto riguarda i frame element che le lexical unit. Questo approccio permette un accesso selettivo e mirato, utile per affrontare determinati task linguistici o semantici.

Anche le *lexical units* meritano particolare attenzione. Esse rappresentano l’associazione tra un’espressione linguistica e un significato ben definito. Si pensi, ad esempio, al frame *Apply\_heat*, che descrive una situazione tipica in cui si applica calore a del cibo usando uno strumento adatto. Questo frame è evocato da verbi come *bake*, *blanch*, *boil*, *broil*, che ne costituiscono le lexical units. Ogni senso diverso di un termine polisemico corrisponde a una lexical unit distinta. È importante sottolineare che all’interno di un singolo frame, la lexical unit non è mai ambigua: una volta identificata, corrisponde sempre a un significato specifico e univoco.

Questa caratteristica è proprio ciò che rende naturale e sensata l’operazione proposta nell’esercitazione, ovvero l’attribuzione di un identificatore semantico (come un synset di WordNet) a ciascuna lexical unit. Infatti, sebbene termini come *bake* possano avere più significati in generale, all’interno di ciascuno dei frame in cui compaiono, il significato è sempre univoco. È per questo che l’associazione tra lexical unit e synset si presenta come un passaggio logico, intuitivo e utile.

anche le *lexical units* possono essere ricercate utilizzando lo stesso strumentario adottato per i frame. È quindi possibile individuarle a partire da una stringa o da un’espressione regolare. Ad esempio, cercando l’espressione “a little”, si ottengono tutte le *lexical units* contenenti quella sequenza di caratteri, che possono includere anche altre espressioni linguistiche sostanzialmente corrispondenti. Oppure, inserendo un termine più specifico, come il verbo “foresee”, si può accedere all’insieme delle *lexical units* che includono quella forma, eventualmente insieme ad altre varianti lessicali.

Questa modalità permette anche di partire da un lemma per risalire a tutti i frame che contengono quel termine in una delle sue possibili accezioni. Si tratta, quindi, di una chiamata potenzialmente molto utile ai fini dell’esplorazione semantica e della disambiguazione lessicale. All’interno di FrameNet sono disponibili circa 13.500 *lexical units*, una copertura che, pur non essendo vastissima rispetto all’intero lessico inglese, è comunque significativa per molti ambiti di analisi.

Quando si interroga una specifica *lexical unit*, si ha accesso a informazioni dettagliate: nome, definizione, ed eventuali ulteriori dati lessicali relativi alla parola. Analogamente, è possibile visualizzare la struttura completa di un frame, come nel caso del frame *Revenge*, già analizzato precedentemente. Questo frame comprende tutto ciò che è stato visto nella definizione generale, a partire dall’indirizzo nella risorsa, passando per le definizioni, i tipi semantici coinvolti, le relazioni con altri frame e l’elenco completo delle *lexical units* che lo compongono.

All’interno del frame selezionato, è possibile stampare tutti i *frame elements*, e per ciascuno di essi si può accedere a un ricco insieme di informazioni. Questo è fondamentale nel momento in cui si rende necessario procedere con la disambiguazione. Per farlo, si può costruire una lista di termini associati, naturalmente dopo aver effettuato un filtraggio, eliminando elementi grammaticali non informativi come preposizioni, articoli, ausiliari e in generale le *stop words*. L’obiettivo è identificare intersezioni semantiche utili per prendere decisioni sull’attribuzione dei significati, quindi sull’identificazione dei sensi corretti.

La disambiguazione dei *frame elements* rappresenta uno dei task previsti. Anche in questo caso, l’interfaccia di NLTK offre un accesso selettivo alle definizioni, così come avviene per le *lexical units*, senza bisogno di interrogare un database separato. Il sistema consente infatti di accedere direttamente ai campi informativi presenti negli oggetti, frammentandoli e rendendo estremamente agevole la navigazione e la consultazione.

Un altro elemento di cui si è accennato, pur non essendo centrale per l’esercitazione, riguarda l’insieme delle relazioni in cui un frame può essere coinvolto. FrameNet, infatti, prevede diverse tipologie di relazioni tra frame. Alcune di queste sono particolarmente rilevanti, come *CausativeOf*, *Uses*, *Precedes*, *Subframe* o *Inheritance*. Altre, invece, risultano poco informative, come *SeeAlso*, che crea un legame generico e non sempre significativo tra entità diverse, senza indicare con precisione il tipo di nesso. Le relazioni più forti e ben definite sono invece potenzialmente molto utili nell’analisi semantica approfondita. Ad esempio, si può esplorare cosa venga causato dal frame *MakeNoise*, nel caso esista una relazione causativa con un altro frame, anche se in alcuni casi le relazioni possono risultare ridondanti o tautologiche.

In ogni caso, l’insieme delle relazioni offre una prospettiva ulteriore per ampliare l’esplorazione semantica della risorsa.

Si è sottolineato come alcune operazioni che per un essere umano risultano banali possano invece rappresentare una sfida per un sistema computazionale, soprattutto in termini di comprensione semantica. Una delle possibilità offerte dall’interfaccia di *FrameNet* è l’esplorazione delle relazioni che un frame intrattiene con altri. Ad esempio, il frame *MakeNoise* risulta collegato ad altri frame quali *AbsorbHeat*, *FluidicMotion* e *Friction*, attraverso la relazione denominata *ReframingMapping*. Per comprenderne appieno il significato, sarebbe opportuno consultare la documentazione ufficiale di FrameNet.

Un ulteriore aspetto affrontato riguarda l’accesso alle annotazioni disponibili nel corpus associato alla risorsa, nello specifico una porzione del *British National Corpus* che è stata annotata dagli sviluppatori di FrameNet. Le annotazioni, sebbene presentate in forma piuttosto semplificata, offrono informazioni utili sui dipendenti sintattici associati alle *lexical units* verbali. È proprio da tali annotazioni che si può trarre beneficio per studiare le dipendenze semantiche e i ruoli assunti dai vari elementi nelle frasi campione.

Un'ultima funzione, scritta appositamente per l’esercitazione, ha uno scopo puramente ausiliario. Supponiamo che un gruppo sia composto da due persone, ad esempio Mario Rossi e Marta Verdi. Questa funzione consente di determinare l’insieme dei frame che i componenti del gruppo devono annotare. L’annotazione, in questo contesto, non si limita all’identificazione generica del frame, ma richiede un'analisi accurata dei suoi elementi.

Annotare un frame significa comprendere nel dettaglio di quale fenomeno semantico si tratti. Spesso questo non è immediato, sia per una possibile distanza tra il concetto italiano e la sua rappresentazione inglese, sia per difficoltà linguistiche vere e proprie. Una volta compreso il frame, occorre procedere con l’annotazione dei suoi elementi costitutivi, ovvero tutti i *frame elements* e tutte le *lexical units* associate, attribuendo a ciascuno di essi un *synset* di WordNet.

Nel caso in cui un termine presenti più di un senso plausibile, è possibile assegnare più *synset*, ma si consiglia di limitarsi, laddove possibile, a un solo significato per garantire maggiore coerenza. L’annotazione deve coprire tutti gli elementi previsti dal frame.

Per quanto riguarda la scelta dei frame, la funzione sopra citata è in grado di generare automaticamente l’elenco in base ai nomi e cognomi dei componenti del gruppo. A differenza di esercitazioni in cui si valuta l’accordo tra annotatori, qui l’obiettivo è estendere il più possibile la gamma di frame, *frame elements* e *lexical units* su cui lavorare.

In presenza di *multi-word expressions*, come ad esempio *religious belief*, si richiede la disambiguazione del termine principale, identificabile come il reggente dell’espressione. In genere, se si tratta di nome + aggettivo, sarà il nome a fungere da reggente; se invece è verbo + sostantivo, sarà il verbo. Questa identificazione può essere effettuata manualmente oppure con il supporto di un parser sintattico.

Lo stesso criterio si applica anche ai *frame elements* e alle *lexical units*, i quali possono anch’essi presentarsi come espressioni complesse. Sebbene l’operazione possa risultare lunga e laboriosa, si prevede comunque un certo margine di flessibilità. Ad esempio, se un gruppo riceve un frame particolarmente ricco di elementi, si potrà valutare l’opportunità di limitarne l’estensione annotativa.

Infine, per procedere con la disambiguazione, si può far uso delle definizioni testuali dei frame e dei *frame elements*, così da costruire un contesto lessicale utile — indicato nella notazione introdotta come *context words* — a guidare le scelte semantiche durante l’annotazione.

Nel contesto del lavoro di disambiguazione semantica basato su FrameNet e WordNet, è possibile utilizzare le definizioni (glosse) associate ai sensi presenti in WordNet, nonché i relativi iponimi e iperonimi, al fine di ottenere un contesto di disambiguazione più ricco. Questa operazione consente di estendere le informazioni semantiche disponibili e costruire così un *context* del senso, indicato con *context di S*, distinguendolo dal *context di W*, che riguarda invece il termine lessicale in sé. Il primo è quindi di natura semantica, mentre il secondo è di natura puramente lessicale.

Dato il contesto del termine ambiguo, il *context di W* e dati i contesti dei vari sensi potenzialmente associabili a quel termine, il *context di S*, l’obiettivo generale è trovare un mapping, ossia una funzione che permetta di associare correttamente un termine al suo senso più appropriato.

Questo mapping può essere realizzato attraverso due diversi approcci. Il primo è l’approccio *Bag of Words*, lo stesso adottato anche dal sistema LESC. Esso prevede la selezione del senso che massimizza l’intersezione tra gli elementi del contesto del termine e quelli del contesto del senso. In termini pratici, si calcola uno score per ogni senso candidato, definito come la cardinalità dell’intersezione tra il *context di S* e il *context di W*. Il contesto del termine viene costruito “a strascico”, raccogliendo tutti i sostantivi, verbi e altri elementi significativi presenti nelle descrizioni del frame, dei frame elements e delle lexical units. Il senso con lo score più alto viene selezionato come quello più probabile. Pur essendo un metodo semplice, questo approccio si è dimostrato generalmente efficace.

Il secondo approccio è di tipo grafico. Esso prevede la costruzione di un grafo che contiene tutti i synset associati ai termini presenti nei frame di FrameNet, includendo non solo i termini stessi, ma anche le descrizioni associate a frame elements e lexical units. L’idea è quella di costruire un contesto il più ricco possibile, senza tralasciare alcun dettaglio.

Per ciascun termine www presente in un frame element o in una lexical unit, si cerca il senso sss di WordNet che massimizza la probabilità condizionata P(s∣w)P(s|w)P(s∣w), ovvero la probabilità di ottenere il senso sss dato il termine www. Questo calcolo si realizza individuando il senso sss che massimizza tale probabilità tra tutti i possibili sensi di www. Formalmente, si seleziona l’argomento sss che massimizza una frazione in cui al numeratore si trova lo score tra sss e www, e al denominatore la somma degli score tra tutti i sensi s′s's′ e tutti i termini w′w'w′ appartenenti al contesto di www. In questo modo, il problema della disambiguazione viene ricondotto al calcolo di questi score.

La costruzione del *context di W*, ovvero il contesto lessicale di un termine, si basa sull’estrazione dei termini lemmatizzati dalle descrizioni dei frame e dei frame elements, dopo aver applicato un filtraggio delle *stopwords*. Questo passaggio è essenziale per evitare che l’intersezione tra i contesti sia influenzata da termini grammaticalmente funzionali ma semanticamente irrilevanti, come articoli, congiunzioni o preposizioni. L’eliminazione delle stopwords si fonda sulla distinzione tra *content words*, che veicolano il significato della frase, e *function words*, che sono invece meno rilevanti dal punto di vista semantico. Sebbene esista l’intuizione che termini più frequenti possano essere più significativi, vi è anche il principio opposto: alcuni termini sono talmente diffusi nel linguaggio da avere un peso semantico trascurabile. Questo doppio criterio, basato su frequenza e funzione, guida la selezione dei termini utili alla costruzione dei contesti.

Se si tenesse conto soltanto della frequenza di un termine all’interno di un documento, senza metterla in relazione con la frequenza generale in tutta la collezione, si perderebbe questa informazione fondamentale. Per raffinare la capacità di distinguere termini veramente significativi da quelli generici, è necessario un ulteriore strumento di analisi: l’*inverse document frequency*, ovvero l’inverso della frequenza dei documenti in cui un termine compare. Anche se in questo contesto non si approfondisce l’aspetto quantitativo della questione, è importante comprendere l’intuizione che sta alla base: ci sono parole che, proprio perché troppo diffuse, non portano alcun valore discriminante nel contesto dell’analisi.

Quando si lavora con la disambiguazione semantica o con il mapping tra termini e sensi in WordNet, è fondamentale che i termini usati per il confronto — ad esempio quelli su cui si calcola un’intersezione — siano *pienamente significativi*. Per questo, si adotta un filtro che rimuove le cosiddette *stop word*: parole ausiliarie, congiunzioni, preposizioni e altri elementi grammaticali che non contribuiscono direttamente al contenuto semantico del testo.

n un approccio iniziale o semplificato, ci si accontenta di utilizzare liste di stop word generiche, che possono essere facilmente reperite online.

Tra queste liste, una delle più note è quella prodotta da Ricardo Baeza-Yates, noto per essere stato direttore scientifico di Google. Oggi, strumenti come la libreria NLTK (Natural Language Toolkit) mettono a disposizione liste già pronte di stop word, facilmente scaricabili e utilizzabili. È questo il tipo di soluzione operativa che viene adottata in prima battuta, con l’intenzione di raffinare l’approccio in un secondo momento.

Pertanto, nel processo di costruzione dei contesti per la disambiguazione o per il mapping dei termini su WordNet, un passaggio essenziale consiste nell’applicare il filtro delle stop word, così da conservare solo i termini realmente significativi ai fini dell’analisi.

La funzione di *mapping* che permette di associare un termine ambiguo a uno dei suoi sensi possibili si basa, come già anticipato, sulla costruzione del contesto di disambiguazione. Questo contesto, relativo ai termini www provenienti da FrameNet, è ottenuto a partire dai testi descrittivi dei frame e dei *frame elements*, dopo averli lemmatizzati e privati delle *stop word*, cioè dei termini non significativi per il contenuto semantico.

A partire da questo contesto filtrato, si calcola lo *score* ​, un valore che esprime quanto un certo senso S, preso da WordNet, sia rilevante per il termine w in esame. Il calcolo di questo punteggio implica la costruzione — o meglio, la ricostruzione — di un sottografo di WordNet, in cui vengono presi in considerazione tutti i *scene set* (insiemi di sensi semanticamente affini) che si trovano lungo percorsi di lunghezza minore o uguale a un certo valore L. Tale parametro può essere variato per sperimentazione, partendo da valori contenuti come L=3 per motivi di efficienza computazionale, ed eventualmente estendendolo in presenza di risorse hardware più potenti. L’idea di fondo è semplice: si parte dalla volontà di massimizzare la probabilità condizionata di un senso S dato un termine www, cioè P(S∣W). Questa probabilità viene riformulata come probabilità congiunta, e il suo calcolo si riduce a un rapporto tra lo *score* relativo alla coppia (S,W) e la somma degli *score* di tutte le possibili coppie (S′,W′), con S′ e W’ presi rispettivamente tra i sensi e i termini del contesto.

La funzione di punteggio per il senso S di un termine W si basa su un meccanismo di esplorazione del grafo semantico. Per ogni parola del contesto di disambiguazione (cioè un termine che appartiene a FrameNet e che si trova, ad esempio, nella descrizione del frame o dei frame elements), si risale ai suoi possibili sensi S′ in WordNet. Per ciascuno di questi sensi, si costruiscono tutti i percorsi (path) che collegano il senso di partenza S al senso S′, limitandosi a quelli con lunghezza massima L. A ogni percorso viene attribuito un punteggio decrescente in base alla sua lunghezza: più il cammino è corto, più i due sensi sono considerati semanticamente vicini.

Questa è l’intuizione centrale: la vicinanza semantica si riflette nella distanza nel grafo. Quanto più breve è il cammino tra due sensi in WordNet, tanto più è probabile che essi siano concettualmente affini. Questo principio, già incontrato nella prima esercitazione attraverso meccanismi basati su path semantici, viene qui esteso e formalizzato nel contesto della disambiguazione lessicale, con una tecnica che si ispira all’algoritmo utilizzato in una risorsa importante e complessa come BabelNet (a cui si fa riferimento con il nome di "Babel-Nebla").

Dal punto di vista operativo, il procedimento è chiaro: si parte da un termine da disambiguare, si raccolgono i suoi possibili sensi da WordNet, si costruisce il contesto a partire da FrameNet, si identificano i sensi delle parole del contesto, e infine si valutano i percorsi semantici che collegano il termine iniziale agli altri. Il senso scelto sarà quello che risulta più connesso alla rete semantica formata dai sensi degli altri termini del contesto, in base ai criteri di prossimità definiti dal grafo di WordNet.

(il resto della trascrizione è una ricapitolazione dell’esercitazione richiesta)

**Lezione 7**

Si introducono un po' di risorse e strumenti che permettono di rappresentare i sensi linguistici e di esplorare come questi sensi possano essere trattati in formato sperimentale. L’obiettivo principale è esaminare come un approccio probabilistico possa funzionare efficacemente, come complemento al lavoro precedente, e come si possano combinare i due approcci, tradizionalmente trattati, che sono nati come modelli distinti e si sono sviluppati nel tempo.

I language models sono strumenti teoricamente semplici ma che, in molti casi, risultano molto efficaci. Questi modelli sono stati tra i primi a essere sviluppati e, nonostante la loro semplicità, sono in grado di funzionare molto bene in numerosi contesti. Un aspetto positivo dei language models è che risultano essere più rapidi da calcolare e più agili da utilizzare rispetto a modelli più complessi, come quelli attualmente in uso, come BERT o GPT-2 e GPT-3. Inoltre, i modelli più semplici, come quelli basati su n-grammi, sono anche più trasparenti, a differenza di modelli più complessi che tendono ad essere opachi.

A differenza di approcci precedenti, in cui l'analisi linguistica e la creazione di sistemi cognitivamente plausibili avevano un ruolo centrale, la modellazione probabilistica si basa esclusivamente su stime di probabilità. In pratica, si tratta di stimare la probabilità di un certo evento linguistico contando le occorrenze di determinati fenomeni linguistici in un corpus.

Per farlo, si utilizzano set di dati organizzati in un ciclo di addestramento, sviluppo e test, come accade nei tradizionali processi di training dei modelli. I modelli di linguaggio probabilistico partono inizialmente dagli unigrammi, espandendosi successivamente verso bigrammi e, eventualmente, trigrammi, per arrivare ad una generalizzazione più complessa. I conteggi di frequenze di un dato evento vengono quindi convertiti in probabilità, che possono essere utilizzate come base per il calcolo probabilistico, permettendo di fare previsioni su sequenze linguistiche future.

Un aspetto fondamentale di questa metodologia è il Maximum Likelihood Estimation (MLE), un concetto che alcuni potrebbero già conoscere. Questo metodo di stima si utilizza per calcolare la probabilità di occorrenza di una sequenza di parole in base ai dati a disposizione, in modo tale che il modello trovi i parametri che massimizzano la verosimiglianza dei dati osservati.

Oltre alla costruzione del modello, un passo fondamentale è la valutazione delle prestazioni del modello stesso. Una volta che un modello di linguaggio è stato addestrato su un corpus, occorre capire quanto bene il modello funzioni e misurarne l’efficacia. In questa fase, si introducono delle metriche specifiche per validare il modello e misurare la sua capacità predittiva.

Infine, uno degli aspetti critici dei language models riguarda il fenomeno noto come "data sparsity", ovvero il problema che si presenta quando un modello incontra sequenze di parole mai viste in precedenza. I modelli di linguaggio soffrono infatti quando si trovano di fronte a sequenze nuove, mai osservate nel corpus di addestramento. Questo tipo di problema è particolarmente rilevante per modelli basati su n-grammi, che utilizzano informazioni limitate sul contesto e possono non essere in grado di gestire correttamente sequenze sconosciute. Per affrontare questa difficoltà, si utilizzano tecniche di smoothing, che permettono di “appiattire” la distribuzione delle probabilità in modo da ridurre l’impatto di eventi mai osservati. Queste tecniche permettono di rendere i modelli più robusti e in grado di gestire meglio casi di data sparsity. Nel contesto della modellazione linguistica, uno degli aspetti cruciali riguarda il problema della "out-of-vocabulary" (OOV), ossia dei termini che non fanno parte del vocabolario utilizzato per addestrare il modello. Questo fenomeno rappresenta una delle principali difficoltà nella stima probabilistica del linguaggio, poiché i modelli di linguaggio devono affrontare la sfida di trattare parole o frasi che non sono mai state osservate durante la fase di addestramento. La probabilità di una sequenza linguistica è un concetto centrale in vari ambiti applicativi, come la traduzione automatica, il controllo ortografico (spell checking) e il riconoscimento del parlato. Nella traduzione automatica, per esempio, è possibile stimare la probabilità di una "emissione" linguistica, ossia di una sequenza di parole, per determinare quanto sia probabile una data traduzione. In questo caso, si può pensare a due sequenze linguistiche come a due possibili traduzioni di una medesima frase, e calcolare la probabilità di ciascuna sequenza. Ad esempio, la probabilità della sequenza "please complete this task" potrebbe essere maggiore rispetto a quella di "please complete this abstract", a causa della maggiore frequenza di uso della prima in un dato contesto linguistico.

Un altro campo in cui i modelli probabilistici sono impiegati è il controllo ortografico. In un caso come "please turn your homework in", se "turn" fosse stato scritto erroneamente come "tun", il modello potrebbe stimare che "turn" è molto più probabile rispetto alla sequenza con l'errore. Questi modelli sono alla base di molti strumenti di correzione ortografica e sono essenziali per identificare errori e suggerire correzioni.

In generale, stimare la probabilità di una sequenza di termini è utile in molte applicazioni, incluse le attività di riconoscimento del parlato, la generazione automatica di testo e altre attività chiamate "downstream tasks", che fanno parte dell'elaborazione del linguaggio naturale (NLP). I language models sono esattamente gli strumenti che attribuiscono probabilità a sequenze di termini, e la loro funzione principale è stimare la distribuzione probabilistica di tali sequenze.

Esistono modelli di linguaggio relativamente semplici, come gli unigrammi, che si concentrano sulla probabilità di singoli termini all’interno di una sequenza. Tuttavia, man mano che si passa a modelli più complessi, come i bigrammi e i trigrammi, che considerano coppie o terzetti di parole, la predizione diventa più accurata, anche se richiede risorse computazionali maggiori. I modelli di trigrammi, ad esempio, considerano sequenze di tre termini consecutivi e permettono una predizione più precisa rispetto ai bigrammi. Tuttavia, l'uso di sequenze più lunghe comporta anche un aumento della sparsità dei dati, rendendo necessario l’utilizzo di set di dati più ampi per evitare che il modello incappi in situazioni in cui non ci sono occorrenze sufficienti per alcune combinazioni di parole.

Per costruire un language model, il primo passo consiste nell’acquisire un corpus ampio, come quello di Wikipedia, e nel contare le occorrenze di termini o sequenze di termini all'interno di quel corpus. Una volta raccolti i dati, è possibile calcolare la probabilità di un termine W dato una storia H, ossia di predire la probabilità di una parola in base al contesto che la precede. Questo approccio si basa su un concetto semplice: si conta quante volte la storia H è seguita dal termine W e si divide il risultato per il numero totale di occorrenze di H nel corpus.

Tuttavia, questo approccio può risultare inefficace, poiché non tutti i possibili termini o sequenze sono presenti in ogni corpus. La lingua, infatti, è un sistema produttivo e potenzialmente infinito, e non possiamo garantire che tutte le sequenze possibili siano osservabili nei dati disponibili. Questo significa che, nonostante l'ampiezza del corpus, potrebbe esserci sempre una certa quantità di sequenze che non sono state mai viste e che quindi non possono essere predette in modo diretto.

Inoltre, la semplice contabilità delle occorrenze non è sufficiente per gestire adeguatamente la varietà e la complessità del linguaggio naturale. A questo proposito, è necessario adottare tecniche di smoothing, che permettano di stimare la probabilità di sequenze mai osservate, distribuendo "un po' di probabilità" su tutte le possibili sequenze, anche quelle non presenti nel corpus. Questo approccio aiuta a gestire la sparsità e migliora le previsioni anche per sequenze rare o completamente sconosciute.

La probabilità di una stringa, che rappresenta un’emissione linguistica, è spesso scritta in modo semplificato. In pratica, si usa una notazione che consente di indicare la probabilità di una sequenza, per esempio, la probabilità che una stringa x sia uguale a una determinata sequenza. Questa semplificazione notazionale consente di trattare sequenze linguistiche in modo più agevole, facilitando la costruzione di modelli di linguaggio probabilistici che possano essere applicati in vari contesti.

Nel contesto della modellazione probabilistica del linguaggio, si adotta spesso una notazione semplificata per rappresentare la probabilità di una variabile casuale che assume un determinato valore. Ad esempio, invece di scrivere formalmente la probabilità che una variabile casuale ​ assuma il valore "the", ossia P(="the"), si usa semplicemente la notazione abbreviata P(the). Questa semplificazione permette di rendere più leggibile ed efficiente la trattazione.

Quando si lavora con sequenze di parole, queste possono essere rappresentate in modi equivalenti. Ad esempio, una sequenza composta da , , ​ può essere indicata con ​ oppure Entrambe le notazioni si riferiscono alla stessa stringa di parole, dalla prima alla n-esima.

Infine, la probabilità congiunta dell’intera sequenza – cioè la probabilità che ciascuna parola della sequenza assuma un determinato valore – viene indicata come P(, ,). Questa esprime la probabilità che tutte le parole della sequenza compaiano esattamente nell’ordine specificato.

Alla base della modellazione probabilistica del linguaggio si trova un principio fondamentale: la **regola della catena delle probabilità** (*Chain Rule of Probability*), che permette di calcolare la probabilità congiunta di una sequenza di eventi — in questo caso, parole. Applicata al linguaggio, questa regola consente di decomporre la probabilità di un'intera frase nella moltiplicazione di probabilità condizionate. Più precisamente, la probabilità congiunta di una sequenza di parole , , ​ può essere espressa come:

Immagine che contiene testo, Carattere, schermata, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa scomposizione riflette l’idea che la probabilità di ciascun token in una frase possa essere stimata in funzione dei token che lo precedono. Nel contesto dell’elaborazione del linguaggio naturale, tale meccanismo serve a due scopi principali: **l’analisi** e la **generazione**.

Da un lato, l’aspetto analitico permette di valutare quanto una certa sequenza sia “linguisticamente plausibile”, ossia quanto rispecchi le strutture apprese dal modello a partire dai dati osservati. Si tratta di una misura della “naturalità” della frase rispetto alla distribuzione statistica delle sequenze linguistiche.

Dall’altro lato, il language model può essere utilizzato per **generare testo**. Acquisita una distribuzione di probabilità sulle sequenze, è possibile iniziare da un token qualsiasi e generare, attraverso un processo di campionamento, la continuazione più probabile, parola dopo parola.

Tuttavia, stimare la probabilità di lunghe sequenze basandosi sulla co-occorrenza completa di tutti i token precedenti risulta rapidamente impraticabile. Il linguaggio, per sua natura, è estremamente **produttivo e creativo**: esistono molte combinazioni plausibili che, però, non si sono mai manifestate nei dati di addestramento. Ne consegue che il semplice conteggio delle occorrenze diventa inadeguato.

A questo problema si risponde con un’**assunzione di semplificazione** molto potente, nota anche in altri ambiti come quello dei **modelli di Markov**. Si ipotizza, cioè, che la probabilità del prossimo token dipenda soltanto da un numero limitato di elementi precedenti, e non dall’intera sequenza. L’approssimazione più semplice è quella del **modello bigramma**, in cui si assume:

Immagine che contiene Carattere, testo, tipografia, calligrafia

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa ipotesi riduce significativamente la complessità computazionale, mantenendo una buona capacità predittiva in molti contesti. È un compromesso tra precisione e gestibilità, che consente di costruire modelli linguistici efficienti anche su grandi corpus.

In sostanza, si tratta di rinunciare a una parte dell'informazione storica per guadagnare in semplicità, velocità di calcolo e robustezza nelle stime, accettando che la previsione del prossimo token si basi solo sul contesto immediatamente precedente. Questa è l’essenza della modellazione bigrammatica e, più in generale, della modellazione n-grammatica.

Nel passaggio dai modelli bigramma a quelli trigramma, si amplia l’orizzonte del contesto considerato per stimare la probabilità di un’emissione linguistica. Nel caso del bigramma, si assume che la probabilità di un termine dipenda esclusivamente dal termine che lo precede immediatamente. Il trigramma, invece, generalizza questa approssimazione, includendo anche il penultimo termine nella storia. In termini formali, si passa da:

Immagine che contiene testo, Carattere, schermata, tipografia

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa transizione riflette una maggiore fedeltà al comportamento naturale del linguaggio, nel quale il contesto più ampio può influenzare in modo significativo la scelta del prossimo token. Tuttavia, anche se si migliora la capacità predittiva, si introduce una maggiore complessità, sia computazionale sia legata alla **sparsità dei dati**, perché più si allunga la sequenza di riferimento, più è probabile che alcune combinazioni non siano mai state osservate.

Il meccanismo, comunque, rimane fondato su un principio comune: approssimare la probabilità di una sequenza basandosi su frequenze osservate, e dunque su conteggi.

Per illustrarne il funzionamento, si consideri un linguaggio LLL costituito da tre simboli: a, b, e c. Se si assume che questi simboli abbiano uguale probabilità, ciascuno avrà probabilità 1/3. Le coppie possibili (ovvero le sequenze di lunghezza 2) saranno nove, ciascuna con probabilità pari a 1/9​, in virtù dell'indipendenza tra le scelte.

Immagine che contiene testo, numero, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Se però si assegna una distribuzione non uniforme ai simboli, per esempio, P(a)=0,5, P(b)=0,25, P(c)=0,25 allora la probabilità delle sequenze varia, riflettendo queste nuove pesature. Le combinazioni più probabili saranno quelle che coinvolgono i simboli più frequenti.

Immagine che contiene testo, schermata, numero, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Le combinazioni più probabili saranno quelle che coinvolgono i simboli più frequenti. In questo senso, le sequenze che presentano “a” saranno statisticamente più rappresentative del linguaggio LLL, mentre le altre avranno probabilità più bassa.

A questo punto entra in gioco una nozione fondamentale, utile sia per la generazione sia per la valutazione delle sequenze: la **probabilità condizionata**. Questa si ottiene dividendo la frequenza di una sequenza completa per la frequenza della sua sottosequenza iniziale. Per esempio, se si osserva la stringa “ab” nel corpus, la probabilità condizionata si calcola come:

Immagine che contiene Carattere, testo, bianco, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa normalizzazione consente di definire quanto è probabile un certo elemento in funzione del suo contesto immediato. Più il contesto è frequente, più stabile è la stima.

Un esempio didattico può aiutare a chiarire. Si consideri un corpus di 16 parole, tra cui la parola “Peter” compare due volte: all’inizio della prima frase e al centro della seconda. Se si desidera stimare la probabilità della sequenza “Peter Piper”, si contano le occorrenze della coppia e la si normalizza rispetto alla frequenza del primo termine.

Immagine che contiene Carattere, bianco, linea, testo

Il contenuto generato dall'IA potrebbe non essere corretto.

Questo implica che, in questo corpus, ogni volta che compare “Peter”, è sempre seguito da “Piper”. Al contrario, se si cerca la probabilità della sequenza “Peter Pan” e questa non compare mai, ma “Peter” compare due volte, allora:

Immagine che contiene Carattere, testo, bianco, tipografia

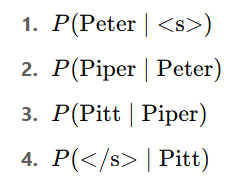
Il contenuto generato dall'IA potrebbe non essere corretto.

ale risultato segnala che, nel dominio del linguaggio osservato, “Peter” non è mai stato seguito da “Pan”, e dunque questa sequenza ha probabilità nulla secondo il modello.

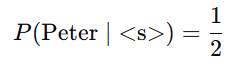
Operativamente, l’estensione da bigrammi a trigrammi e oltre non richiede un cambiamento sostanziale nella procedura di calcolo, ma comporta una maggiore attenzione alla **gestione della sparsità**. Per questa ragione, nei modelli linguistici classici e nelle implementazioni pratiche, si utilizzano strategie di **smoothing** e **backoff**, che permettono di trattare i casi in cui una certa sequenza non sia stata osservata, evitando che la sua probabilità venga forzatamente posta a zero. Conoscere questa dinamica è essenziale per comprendere il comportamento e le limitazioni dei modelli probabilistici n-gramma.

Per costruire un modello di linguaggio probabilistico è prassi comune introdurre due simboli speciali: uno per indicare l’inizio della frase (spesso denotato come <s>, ovvero *start*) e uno per segnare la sua conclusione (indicato come </s>, ovvero *stop* o *terminatore*). Questo accorgimento consente di gestire in modo coerente i margini della sequenza, permettendo al modello di apprendere anche le transizioni che avvengono nei punti di apertura e chiusura di una frase.

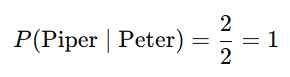
Includendo questi simboli nelle sequenze di addestramento, la stima della probabilità estesa di un’emissione linguistica tiene conto anche delle transizioni iniziali e finali. Si consideri, ad esempio, la sequenza “Peter Piper Pitt”. La probabilità complessiva dell’intera sequenza sarà calcolata come prodotto delle seguenti probabilità condizionate:



Supponendo di avere due frasi nel corpus: una che inizia con “Peter” e una che inizia con “Where”, il conteggio del simbolo di inizio frase <s> è pari a 2. Di queste due, solo una è seguita da “Peter”, dunque:



Analogamente, se “Peter Piper” compare due volte e “Peter” compare due volte, allora:



Se “Piper Pitt” compare due volte e “Piper” compare due volte:

Immagine che contiene Carattere, bianco, testo, Elementi grafici

Il contenuto generato dall'IA potrebbe non essere corretto.

Infine, se “Pitt” è seguito una volta da </s> e una volta da un altro termine, allora:

Immagine che contiene Carattere, testo, tipografia, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Il prodotto delle probabilità è dunque:

Immagine che contiene Carattere, tipografia, linea, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa probabilità descrive quanto è compatibile la sequenza “Peter Piper Pitt” con il modello linguistico appreso dal corpus di riferimento. Una volta acquisito il modello, diventa possibile generare nuove sequenze linguistiche, costruendole token dopo token sulla base delle probabilità condizionate. Sebbene il numero di sequenze possibili sia teoricamente illimitato, nella pratica il modello genererà solo combinazioni che hanno probabilità non nulla, evitando le sequenze mai osservate o prive di supporto nel corpus di addestramento. Nel momento in cui una transizione prevista ha probabilità nulla — ovvero non è mai stata osservata — la generazione linguistica si arresta: si è entrati in uno stato assorbente. Tuttavia, fintanto che ogni nuovo token è scelto tra quelli compatibili con il modello (cioè con probabilità positiva), è possibile continuare indefinitamente. Si noti, però, che l’assenza di garanzie sintattiche o semantiche rende le sequenze prodotte potenzialmente scorrette o prive di senso: ciò che si genera è linguisticamente plausibile solo in quanto coerente con le statistiche del training set.

Per stimare le probabilità necessarie a costruire il modello si utilizza il principio della **massima verosimiglianza** (*Maximum Likelihood Estimation*, MLE). Questo metodo si basa sulla frequenza osservata degli eventi nel corpus. Il procedimento è semplice: si contano le occorrenze dei token o delle sequenze (bigrammi, trigrammi, ecc.) e si normalizzano tali conteggi rispetto al totale delle occorrenze rilevanti. In particolare, per stimare la probabilità di un bigramma (x,y), si utilizza la formula:

Immagine che contiene testo, Carattere, linea, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Il denominatore rappresenta il numero totale di volte in cui il termine xxx è stato osservato come primo elemento di un bigramma. Questo metodo garantisce che la somma di tutte le probabilità condizionate a xxx sia pari a 1, come richiesto da ogni distribuzione probabilistica ben definita. Così operando, si ottiene un modello linguistico coerente con il corpus di addestramento, pronto sia per l’analisi statistica di nuove frasi sia per la generazione automatica di sequenze.

Si consideri un training set composto da tre frasi, ciascuna delle quali viene trattata come una sequenza *padded*, ovvero preceduta e seguita da simboli speciali che marcano l'inizio e la fine della frase. Le frasi in questione sono:

1. *Good Morning Everybody*
2. *Morning Good Everybody*
3. *Every Morning a Gazelle*

L’introduzione dei simboli di *start* e *stop* consente di costruire un modello di linguaggio coerente che includa anche le transizioni iniziali e finali. Si procede quindi all’analisi statistica delle sequenze di parole (bigrammi), con l’obiettivo di stimare le probabilità condizionate.

Ad esempio, per calcolare la probabilità di osservare *Morning* dato *Good*, si esaminano i contesti in cui *Good* è seguito da *Morning*. Nel nostro set, *Good Morning* compare una volta, mentre *Good* compare complessivamente due volte come parola iniziale in un bigramma. Pertanto:

Immagine che contiene Carattere, testo, bianco, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

Va tuttavia osservato che *Morning* appare anche in altri contesti: è preceduto una volta da *Good* e un’altra da *Every*. Per stimare la probabilità di *Everybody* dato *Morning*, si considerano tutte le occorrenze di *Morning*. Nel corpus, *Morning* compare tre volte e una sola volta è seguito da *Everybody*, quindi:

Immagine che contiene Carattere, testo, tipografia, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

Analogamente, la probabilità di *Good* dato *<s>* (cioè in apertura di frase) può essere calcolata tenendo conto del numero di volte in cui *Good* è la prima parola effettiva: ciò accade una sola volta su tre frasi. Di conseguenza:

Immagine che contiene Carattere, testo, bianco, simbolo

Il contenuto generato dall'IA potrebbe non essere corretto.

Tali stime si basano sul principio della frequenza relativa, che rappresenta il cuore dell'approccio di **Maximum Likelihood Estimation**. Se si osserva un termine 400 volte in un corpus di un milione di parole, la sua probabilità stimata sarà circa 4×10−44 \times 10^{-4}4×10−4. Questo dà un’idea dell’ordine di grandezza delle frequenze e delle probabilità nel caso di corpora estesi, ben lontani dai semplici esempi illustrativi.

L’importanza del corpus utilizzato per l’apprendimento del modello non può essere sottovalutata. Modelli come quelli della famiglia GPT sono addestrati su collezioni di testi vastissime — decine di miliardi di parole — con un’enorme varietà di stili, registri e generi. Un modello allenato su romanzi francesi dell’Ottocento produrrà un linguaggio ben diverso da uno addestrato su trascrizioni telefoniche o articoli sportivi.

L’esempio tratto dai materiali di Jurafsky illustra proprio questo punto: partendo da un corpus relativamente piccolo (meno di 10.000 frasi), composto da recensioni legate all’ambiente universitario di Stanford, si può già costruire un modello linguistico significativo. In tale corpus, si osserva che l’espressione “I want” compare 827 volte, mentre “want to” è presente 608 volte. Anche sequenze meno frequenti come “to spend” (211 occorrenze) contribuiscono alla struttura del modello. Da questi conteggi è possibile costruire matrici di frequenza per bigrammi, dove la maggioranza dei valori sarà comunque zero — come è tipico nei modelli di linguaggio sparsi. In parallelo, si raccolgono i conteggi degli *unigrammi*, che servono da denominatore nella normalizzazione delle probabilità condizionate. Una volta completata la raccolta dei dati, si applicano le formule viste in precedenza per calcolare le probabilità vere e proprie, convertendo le frequenze in valori numerici normalizzati compresi nell’intervallo [0,1].

Questo processo è fondamentale per l’apprendimento statistico di un modello linguistico e costituisce la base anche per i sistemi molto più avanzati, sebbene su scala immensamente maggiore.

Tutti questi calcoli rientrano nella stima delle probabilità condizionate secondo il criterio della massima verosimiglianza. La distinzione tra probabilità “a priori” e “a posteriori” in questo contesto non è sempre pertinente, poiché il modello di bigrammi classico si limita a osservare frequenze empiriche, senza coinvolgere inferenze bayesiane. È tuttavia cruciale ricordare che l’affidabilità delle stime è fortemente ancorata alla quantità e alla qualità del corpus utilizzato. In set di dati limitati, l’assenza di una sequenza comporta una probabilità nulla, da cui la necessità — nei casi reali — di applicare tecniche di smoothing per evitare l’assegnazione di zeri assoluti a sequenze plausibili ma non osservate.

Nel contesto dei modelli di linguaggio basati su n-grammi, l’obiettivo generale è stimare la probabilità che una parola specifica venga emessa in una certa posizione della frase, condizionata sulle parole precedenti. In particolare, nei bigrammi si lavora sulla base di una dipendenza di ordine 1, ovvero si stima la probabilità della parola ​ dato . In questo senso, l’operazione può essere letta come una predizione: dato un contesto immediatamente precedente, quale parola è più probabile che venga prodotta.

L’approccio della *Maximum Likelihood Estimation* (MLE) si estende alla valutazione della probabilità complessiva di una sequenza di parole, come nel caso dell’emissione dell’intera frase "I want to eat Chinese food". In un contesto semplificato, si parte da una stima della probabilità della parola iniziale — ad esempio, la probabilità di *I* dato un simbolo di inizio frase (come *<s>*) può essere fissata a 0,25. A partire da qui, si moltiplicano le probabilità condizionate dei successivi bigrammi:



Questa sequenza di moltiplicazioni restituisce la probabilità congiunta dell’intera frase secondo il modello addestrato.

Queste probabilità empiriche, stimate a partire da frequenze osservate nel corpus, tendono a riflettere alcune regolarità di tipo sintattico. Ad esempio, la co-occorrenza frequente di *to* seguito da un verbo, o la presenza di un nome dopo *eat*, suggerisce che le strutture grammaticali siano almeno parzialmente catturate da questi modelli. La sintassi implicita del linguaggio naturale viene quindi rappresentata in forma probabilistica sulla base dei dati disponibili. Allo stesso tempo, anche regolarità di tipo semantico o legate alla specificità del dominio sono potenzialmente apprese: se nel corpus *Chinese food* appare più frequentemente di *Italian food*, questa relazione verrà riflessa nei valori di probabilità.

Il passaggio da bigrammi a trigrammi (o più in generale a n-grammi) comporta l’estensione del contesto condizionante: in un modello a trigrammi, la probabilità della parola corrente è condizionata sulle due parole precedenti, in un 4-gramma sulle tre precedenti, e così via. Questa generalizzazione consente una rappresentazione più ricca del contesto linguistico, ma comporta anche una crescita esponenziale dello spazio delle sequenze possibili e una progressiva rarefazione dei dati. Di conseguenza, è frequente l’adozione di simboli di padding iniziali — come *<s><s>* — per garantire che anche le prime parole della frase dispongano di un contesto sufficiente alla predizione.

Un effetto collaterale dell’aumento della profondità del contesto è la drastica diminuzione delle probabilità congiunte, che possono raggiungere valori estremamente piccoli al punto da risultare problematici per la rappresentazione numerica, anche con sistemi a 64 bit. Per affrontare questa criticità e rendere computazionalmente più stabile il calcolo delle probabilità su lunghe sequenze, si ricorre comunemente ai logaritmi. In questo modo, la moltiplicazione delle probabilità viene trasformata in una somma dei logaritmi delle probabilità, facilitando sia la gestione numerica sia il confronto tra probabilità di sequenze alternative.



Questa trasformazione è particolarmente utile in ambiti come il decoding o la valutazione comparativa di output in modelli di generazione.

La valutazione di un modello linguistico può essere affrontata secondo due modalità principali: una valutazione **estrinseca** e una **intrinseca**. La prima riguarda l'efficacia del modello quando impiegato come componente di un sistema più ampio, destinato a risolvere un compito specifico. Si pensi, ad esempio, all’uso del modello linguistico all’interno di un sistema di *spell checking* o *speech recognition*. In questi casi, la qualità del modello viene valutata indirettamente: si analizza quanto il sistema complessivo riesca ad assolvere il proprio compito, ad esempio identificando correttamente un certo numero di errori ortografici presenti in un dataset annotato. Non si misura quindi il modello in sé, ma la sua capacità di contribuire al funzionamento di un'applicazione.

La valutazione **intrinseca**, invece, mira a esaminare direttamente le caratteristiche del modello, senza dipendere da un'applicazione esterna. Questa modalità di analisi è più raffinata, e consente di confrontare modelli diversi su una base comune, indipendentemente dal contesto applicativo. Un modello può mostrare buone prestazioni in un dominio e risultare meno efficace in un altro; la valutazione intrinseca cerca di fornire un quadro più stabile, misurando quanto un modello sia capace di approssimare il linguaggio in senso generale.

Per realizzare una valutazione intrinseca, si parte da un corpus linguistico, che viene suddiviso in due sottoinsiemi: uno destinato all’addestramento (training set), e uno riservato alla valutazione (test set). Il modello viene addestrato sui dati del training set, e la sua qualità viene stimata sulla base delle prestazioni ottenute sul test set, ovvero su dati che non sono stati visti in fase di apprendimento.

Una delle metriche fondamentali in questo contesto è la **perplexity**, una funzione che permette di valutare quanto un determinato modello linguistico sia capace di assegnare probabilità elevate a una sequenza di parole. La perplexity può essere interpretata come una misura di incertezza: un valore basso indica che il modello è “poco perplesso” e quindi molto sicuro nel predire quella sequenza; un valore alto, al contrario, suggerisce una maggiore incertezza o incapacità di spiegare l’input. Formalmente, la perplexity si calcola a partire dalla probabilità della sequenza, stimata dal modello, mediando tale valore su tutta la lunghezza della frase. Il calcolo coinvolge due elementi: il modello linguistico (ad esempio un modello a n-grammi) e una sequenza linguistica da valutare. Più la sequenza riceve un’alta probabilità dal modello, più la perplexity sarà bassa, e quindi più il modello sarà considerato adatto a quel tipo di linguaggio. Per motivi computazionali e di stabilità numerica, si lavora spesso con i logaritmi delle probabilità, trasformando il prodotto delle probabilità in una somma dei logaritmi. Si consideri una sequenza di parole composta da elementi w1,w2,…,wk indicata come W. Poiché l'obiettivo è valutare il modello su dati non ancora visti, è importante che la sequenza di test sia nuova, ovvero che non faccia parte del set di addestramento.

Dato un modello linguistico, indicato come **LM**, è possibile calcolare la **probabilità** che il modello assegna all'intera frase, ovvero LM(W), cioè la probabilità della sequenza W secondo il modello.

In generale, più alta è la probabilità assegnata alla sequenza dal modello, migliore è considerato il modello stesso.

Per valutare questa probabilità in maniera più gestibile, si utilizza la **log-probabilità media**. Questa si ottiene calcolando la logaritmo della probabilità della sequenza completa log⁡P(W) e poi dividendola per il numero di parole nella sequenza, cioè k. In formula:

Immagine che contiene Carattere, testo, bianco, tipografia

Il contenuto generato dall'IA potrebbe non essere corretto.

Da questa quantità si può derivare un'altra misura molto utile, chiamata **perplessità** (*perplexity*). La perplessità di una sequenza rispetto a un modello linguistico è definita come:

Immagine che contiene Carattere, testo, linea, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Una bassa perplessità corrisponde a una probabilità elevata, il che indica che la sequenza di parole si adatta bene al modello. In altre parole, il modello è stato in grado di **prevedere** efficacemente quella sequenza.

Un'applicazione interessante e recente di questa metrica ha riguardato l'analisi del linguaggio di persone affette da Alzheimer. In questo esperimento, sono stati costruiti due modelli distinti abbastanza semplici a 2 o 5-grammi: uno addestrato su trascrizioni di parlato di pazienti con diagnosi di Alzheimer, e l'altro su parlato di soggetti sani. La perplexity è stata utilizzata per valutare quanto il linguaggio prodotto da un nuovo parlante fosse più vicino a uno o all'altro gruppo. Questo approccio ha dimostrato una sensibilità notevole, riuscendo a discriminare con grande precisione tra i due insiemi. In alcune fasi del lavoro, si sono ottenuti punteggi di accuratezza e F1 score pari a 1, risultato eccezionale che ha richiesto verifiche approfondite per escludere errori metodologici.

In quel caso, il modello linguistico non veniva più valutato, ma diventava uno strumento diagnostico per stimare l’affinità di un campione di linguaggio con un certo profilo clinico. L’utilizzo della perplexity in questo modo si avvicina a una forma di rilevazione in tempo reale, in cui il valore informativo della misura diventa un indicatore della natura del parlato stesso.

Uno dei problemi fondamentali nella stima delle probabilità di sequenze linguistiche tramite modelli n-grammi riguarda la presenza di zeri. Nel momento in cui si calcola la probabilità di un'intera frase, basta che anche solo un singolo n-gramma all'interno della catena non sia stato osservato durante la fase di addestramento perché l'intera probabilità venga annullata: una sola transizione a probabilità zero è sufficiente a rendere nulla la stima complessiva. Questa criticità impone di adottare strategie che permettano di evitare, o almeno mitigare, tale eventualità. Una situazione tipica è quella in cui le parole che compongono una certa sequenza sono state osservate individualmente nel dataset di addestramento, ma non nella combinazione specifica in cui si presentano nel test set. Ad esempio, può accadere che il training set contenga espressioni come *deny the claims* oppure *deny the reports*, mentre il test set contenga una sequenza come *deny the press*, che, pur essendo formata da termini noti, non è mai apparsa in quella configurazione. In tal caso, il modello assegna una probabilità nulla alla sequenza, interrompendo di fatto la possibilità di valutarla. Altra evenienza comune è la comparsa di termini che non sono mai apparsi nel corpus di addestramento. In presenza di un vocabolario costruito esclusivamente a partire dal training set, è facile che termini rari o specifici — ad esempio tratti da un linguaggio tecnico, da nomi propri o da testi letterari — risultino sconosciuti. Per affrontare questa circostanza, si adotta frequentemente l’ipotesi di vocabolario chiuso (*closed vocabulary*), secondo la quale il test set è vincolato a contenere unicamente termini già presenti nel vocabolario estratto dal training set. Si tratta di un’ipotesi che, pur potendo sembrare limitativa, trova applicazione consolidata in contesti pratici, come il riconoscimento vocale (*speech recognition*), dove il dizionario fonetico degli elementi pronunciabili è noto e prestabilito.

Nell’ipotesi di vocabolario aperto (*open vocabulary*), il problema delle parole fuori vocabolario (*out-of-vocabulary words*, o *OOV*) viene affrontato, nei modelli tradizionali, con l’introduzione di un simbolo speciale, comunemente etichettato come <UNK> (*unknown*), al quale vengono assegnate tutte le occorrenze di parole non presenti nel vocabolario. Esistono due strategie principali per gestire queste parole sconosciute. Una prima strategia consiste nel trasformare l’intero dataset in un sistema chiuso: durante la costruzione del vocabolario, tutti i termini non inclusi (per esempio quelli al di sotto di una soglia di frequenza) vengono sostituiti con <UNK>. Questa operazione comporta un’alterazione della distribuzione delle probabilità: i termini poco frequenti, ma distinti, vengono accorpati, concentrando massa probabilistica su un singolo simbolo, con effetti potenzialmente distorsivi.

Tale ristrutturazione risponde a un’esigenza statistica precisa: nel linguaggio naturale, la distribuzione delle parole segue una legge fortemente sbilanciata (legge di Zipf), per cui pochissimi termini sono estremamente frequenti, mentre la stragrande maggioranza appare raramente. Il raggruppamento forzato dei termini rari in <UNK> può generare un picco anomalo nella distribuzione, riducendo la granularità del modello. Per attenuare questo effetto, si può costruire il vocabolario imponendo soglie di frequenza: si conteggiano le occorrenze di ciascun termine e si sostituiscono con <UNK> tutte le parole che appaiono meno di un numero prefissato di volte.

Oltre alla gestione delle parole sconosciute, è necessario affrontare anche i casi in cui le parole sono note, ma la loro combinazione in un dato contesto non è stata mai osservata. Si pensi a una sequenza come *rejected the loans*, dove ciascuna parola è presente nel vocabolario, ma l'intera triade non è mai comparsa nel corpus di addestramento. In tali situazioni si ricorre a tecniche di *smoothing*, che permettono di assegnare una probabilità non nulla anche agli eventi non osservati.

Il principio alla base dello smoothing consiste nel sottrarre una parte della massa di probabilità dagli eventi frequenti, ossia quelli già visti nel training set, per redistribuirla agli eventi non osservati. In questo modo, si garantisce che nessuna sequenza venga mai valutata con probabilità zero. Le tecniche di smoothing appartengono a una famiglia ampia e variegata, che include metodi semplici come lo *add-one smoothing* (Laplace), e metodi più sofisticati come il *Good-Turing smoothing* o l’interpolazione con modelli a n-gramma di ordine inferiore.

L’adozione di strategie di smoothing e la gestione delle parole sconosciute sono elementi essenziali per rendere i modelli n-gramma più robusti e adattabili, specialmente in contesti dove la copertura del linguaggio non può mai essere completa.

Una delle strategie più semplici, ma al contempo fondamentali, per affrontare il problema degli zeri nei modelli di linguaggio basati su n-grammi è rappresentata dalla cosiddetta *correzione della classe*, anche nota in letteratura come *add-one smoothing* o *Laplace smoothing*. Sebbene esistano numerose varianti e generalizzazioni di questo approccio, ci si concentra qui sulla sua formulazione più basilare, che si presta bene come punto di partenza per comprendere l’intuizione sottostante.

Il principio su cui si fonda questa tecnica è semplice: ogni possibile evento linguistico — ovvero ogni possibile n-gramma — viene trattato come se fosse stato osservato almeno una volta. In termini operativi, ciò equivale ad aggiungere un’unità al conteggio di frequenza di ogni possibile sequenza, anche se essa non è mai apparsa nel corpus di addestramento. In questo modo, si evita che il calcolo delle probabilità produca valori nulli, rendendo il modello più stabile e robusto. La formulazione concreta prevede che, dato un n-gramma di interesse, la sua probabilità venga calcolata con un numeratore costituito dal conteggio osservato della sequenza incrementato di uno. Tuttavia, tale modifica impone una correzione coerente anche al denominatore, che rappresenta la somma totale dei conteggi per tutte le sequenze condizionate sullo stesso contesto. Poiché è stata ipoteticamente aggiunta un’occorrenza fittizia per ogni possibile continuazione, il denominatore deve essere aumentato della cardinalità del vocabolario, ovvero del numero totale di termini distinti che potrebbero occupare la posizione successiva nella sequenza.

Si consideri il **Laplace smoothing** per i modelli di unigrammi.

La **stima di massima verosimiglianza (MLE)** della probabilità di un unigramma, ovvero di una parola ​, è data dal suo conteggio normalizzato rispetto al numero totale di token (parole) nel corpus, cioè:

Immagine che contiene Carattere, bianco, Elementi grafici, simbolo

Il contenuto generato dall'IA potrebbe non essere corretto.

dove:

* è il numero di occorrenze della parola ​
* N è il numero totale di token nel corpus.

Il **Laplace smoothing** modifica questa stima aggiungendo **uno a ciascun conteggio**. Poiché si somma 1 a ogni parola del vocabolario (di dimensione V), anche il denominatore viene aggiornato per riflettere l’aggiunta di V nuove osservazioni:

Immagine che contiene Carattere, testo, bianco, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Invece di pensare alla Laplace smoothing come a un cambiamento sia del numeratore sia del denominatore, può essere descritto in termini di **conteggio aggiustato** (adjusted count). In questo caso, si definisce un nuovo conteggio:

Tuttavia, per trasformare questo conteggio aggiustato in una probabilità, è necessario **normalizzarlo** moltiplicandolo per un **fattore di normalizzazione**, che tiene conto della nuova somma totale:

Quindi, la probabilità finale può essere espressa come:

Immagine che contiene Carattere, linea, diagramma, simbolo

Il contenuto generato dall'IA potrebbe non essere corretto.

oppure direttamente:

Immagine che contiene Carattere, numero, linea, testo

Il contenuto generato dall'IA potrebbe non essere corretto.

In questa prospettiva, la **smoothing** può anche essere vista come un processo di **sconto**: cioè si riduce leggermente il valore dei conteggi non nulli per poter **redistribuire** quella massa di probabilità sui conteggi nulli (ovvero sulle parole mai viste).

Il concetto di **sconto relativo** (relative discount) è espresso come il rapporto tra il conteggio scontato e il conteggio originale:

Considerando, a questo punto, i bigrammi possono essere fatte le seguenti considerazioni. Formalmente, se si indica con c(il numero di occorrenze osservate della parola dato il contesto ​, la formula corretta sarà:

Immagine che contiene testo, Carattere, linea, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

dove ∣V∣ rappresenta la cardinalità del vocabolario.

Questa operazione, pur garantendo la completa eliminazione dei valori nulli, comporta un’alterazione sistematica delle probabilità, poiché introduce una redistribuzione forzata della massa probabilistica anche verso eventi mai osservati. Inoltre, risulta computazionalmente onerosa, soprattutto su vocabolari di grandi dimensioni, poiché costringe ad aggiornare tutti i conteggi ogni volta. Per ovviare a tale inefficienza, si può riscrivere il calcolo in modo più compatto ed efficiente, sfruttando una rappresentazione in cui i conteggi sono pre-modificati e normalizzati attraverso una costante. In questo modo si riduce la complessità computazionale, rendendo la tecnica più scalabile.

Esistono anche metodi che, pur partendo dalla stessa intuizione — ovvero quella di distribuire parte della massa probabilistica agli eventi non osservati — introducono raffinamenti più sofisticati. Tra questi, uno dei più noti e utilizzati è il *Additive (o Add-k) smoothing*, in cui invece di aggiungere uno si può aggiungere una costante frazionaria, e il metodo di *Good-Turing*, che si basa sulla stima della probabilità degli eventi mai osservati attraverso la frequenza degli eventi rari.

Per illustrare un esempio concreto, si può fare riferimento a un dataset linguistico con un vocabolario di dimensione 1447. Si supponga che, nel caso originario senza smoothing, si fosse calcolata una probabilità sulla base di 608 occorrenze per una certa sequenza e un totale di 917 sequenze condizionate. Applicando la correzione della classe, il conteggio viene incrementato a 609, mentre il denominatore diventa 917 + 1447, incorporando il contributo teorico di tutte le possibili continuazioni mai osservate. Questa modifica garantisce che nessuna sequenza venga trattata come impossibile, ma al prezzo di un appiattimento delle differenze tra le probabilità stimate.

Va segnalato che la correzione della classe, nella sua forma più semplice, non consente l’utilizzo diretto di strategie di *back-off*, che risultano invece particolarmente efficaci quando si lavora con contesti più lunghi e si vuole fare affidamento su contesti di lunghezza inferiore qualora quelli più specifici non siano disponibili. Il back-off si integra bene in sistemi più flessibili, come i modelli interpolati, ed è spesso impiegato per gestire emissioni in modelli di linguaggio probabilistici come quelli basati su Hidden Markov Models o modelli neurali a struttura gerarchica.

L’intuizione alla base di queste tecniche è che **usare meno contesto** può essere utile per generalizzare in quei casi in cui il contesto è **sottorappresentato** nel corpus di addestramento.

Per esempio, se vogliamo calcolare la probabilità condizionata di una parola ​ dato un trigramma ,​, ma **non abbiamo esempi** di quel particolare trigramma, possiamo stimare la probabilità usando la **probabilità bigramma**:



Se nemmeno il bigramma è presente nei dati, allora possiamo ricorrere alla **probabilità unigramma**, cioè:



Un approccio differente è quello dell’**interpolazione**. In questo caso, invece di scegliere un singolo n-gramma su cui basarsi (come nel backoff), si **combinano** le stime di probabilità ottenute da tutti i modelli n-gramma — unigrammi, bigrammi e trigrammi — attraverso una media pesata.

La probabilità di una parola dato il contesto trigramma , ​ viene stimata così:



dove i coefficienti lambda​ sono **pesi** che regolano l’importanza relativa di ciascun modello, e devono **sommare a 1**.

Questo garantisce che la stima finale sia una vera probabilità. L’interpolazione, a differenza del backoff, usa sempre tutte le fonti di informazione disponibili, bilanciandole in base a quanto si ritiene affidabile ciascun livello del modello.

**Lezione 8**

Il *Vector Space Model* rappresenta uno degli approcci più consolidati e tuttora utilizzati nell’ambito del recupero dell’informazione. La sua popolarità è dovuta non solo alla semplicità concettuale e operativa, ma anche alla facilità con cui può essere illustrato e comunicato anche a interlocutori non tecnici, come clienti o decisori aziendali, rendendolo uno strumento persuasivo e accessibile.

La struttura del modello si fonda su una terminologia ben definita. Un *documento* è un’entità testuale che viene indicizzata all’interno di un sistema informativo e resa disponibile per essere recuperata su richiesta. La *collezione* è l’insieme di tutti i documenti che compongono il corpus su cui si effettuano le ricerche. Un *termine* è un’unità lessicale significativa, che può corrispondere a una parola singola, a una locuzione o a qualsiasi sequenza testuale ritenuta rilevante per l’indicizzazione. L’*interrogazione*, infine, è la richiesta posta al sistema, costituita da uno o più termini, attraverso cui l’utente esprime le proprie esigenze informative.

Il cuore del Vector Space Model è la rappresentazione numerica dei documenti, che vengono descritti mediante vettori di caratteristiche. Ogni documento è trattato come un punto in uno spazio multidimensionale, in cui ogni dimensione corrisponde a un termine del vocabolario complessivo. L’informazione che popola ciascuna componente del vettore è tipicamente una misura di frequenza, che può essere grezza (conteggio assoluto delle occorrenze del termine nel documento) o pesata in modo più raffinato (ad esempio mediante il termine *tf-idf*).

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Ogni riga rappresenta una parola presente nel vocabolario, mentre le colonne rappresentano un documento appartenente a una raccolta di documenti.  
Si consideri, ad esempio, una matrice che descrive quattro opere teatrali di Shakespeare e quattro termini scelti come campione.  
Ogni cella della matrice indica quante volte un determinato termine compare in una specifica opera. "As You Like It" può essere rappresentata come il vettore [1, 114, 36, 20]. Uno spazio vettoriale è semplicemente un insieme di vettori ed è caratterizzato dalle loro dimensioni.  
Nel nostro esempio, i vettori che rappresentano i documenti hanno una dimensione pari a 4, ma nelle matrici termine-documento reali, i vettori che rappresentano ciascun documento avrebbero una dimensionalità pari a |𝑉|, ovvero la dimensione del vocabolario (che corrisponde al numero di righe della matrice, solitamente nell’ordine di decine di migliaia di elementi).  
Anche l’altra dimensione, cioè il numero di documenti presenti nella raccolta, può essere molto grande: si pensi, ad esempio, a tutte le pagine web esistenti.  
La prima dimensione di tutti i vettori corrisponde al numero di volte in cui compare la parola "battle", e ciascuna dimensione può essere confrontata tra i documenti.  
Il sistema che ne deriva è uno spazio vettoriale multidimensionale, in cui ogni documento è un punto e l’interrogazione è anch’essa un punto o un vettore di direzione. La similarità tra l’interrogazione e ciascun documento può essere calcolata con criteri geometrici, come il coseno dell’angolo tra i due vettori. I documenti più prossimi al vettore della query sono quelli che, con maggiore probabilità, soddisfano le esigenze informative dell’utente. Per esempio, riducendo la tabella ad uno spazio bi-dimensionale si ha che:

Immagine che contiene testo, schermata, linea, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Questo paradigma presenta numerosi vantaggi. Innanzitutto, consente di trattare in modo naturale la *partial matching*, ovvero il recupero di documenti che contengono solo parte dei termini richiesti. Inoltre, la rappresentazione numerica permette un’agevole integrazione con tecniche statistiche e machine learning, rendendo il modello adattabile a molteplici scenari applicativi.

In un contesto in evoluzione, in cui i modelli di linguaggio generativi stanno modificando il panorama del recupero dell’informazione, il Vector Space Model conserva comunque un ruolo centrale, sia per la sua trasparenza, sia per la sua efficienza, sia come componente di base in sistemi ibridi più sofisticati.

si introduce una nozione che, pur non essendo immediatamente intuitiva, riveste un'importanza fondamentale per comprendere la logica sottostante al *Vector Space Model*. Si è già visto che i documenti e le interrogazioni vengono rappresentati come vettori in uno spazio multidimensionale. Ogni componente del vettore corrisponde a un termine del vocabolario estratto dalla collezione, quindi il numero totale di dimensioni di questo spazio coincide con la cardinalità del dizionario, ovvero l’insieme di tutti i termini distinti che compaiono nei documenti.

Poiché non è possibile prevedere in anticipo quali saranno i termini utilizzati nelle interrogazioni, si tende a considerare come spazio di riferimento l'intero vocabolario. Ne consegue che i vettori che rappresentano i documenti saranno, nella stragrande maggioranza dei casi, *sparsi* (sparse), ovvero costituiti da un numero elevato di componenti nulle. Questo aspetto, lungi dall’essere un difetto, può anzi essere sfruttato a vantaggio dell’efficienza computazionale.

Si consideri ora un nuovo documento che presenta una frequenza maggiore di un termine già noto. Per esempio, un termine che in un primo documento era presente una sola volta, in questo nuovo documento compare sei volte. Questo esempio permette di introdurre una generalizzazione utile: la rappresentazione vettoriale può essere costruita sia su base binaria — con one-hot vectors, in cui si registra la sola presenza o assenza di un termine — sia su base numerica, in cui si considera il conteggio effettivo delle occorrenze.

Tale modellazione consente di distinguere documenti che contengono gli stessi termini ma in quantità diverse. Tuttavia, due vettori possono anche coincidere esattamente: in tal caso, rappresentano documenti perfettamente sovrapponibili dal punto di vista delle feature selezionate. Questo fatto introduce un aspetto rilevante: tutti i documenti e le interrogazioni sono proiettati nello stesso spazio vettoriale, e questo consente di confrontarli direttamente attraverso misure di distanza o similarità.

Per chiarire questo concetto, si immagini una situazione semplificata in cui si considerano solo due dimensioni: la query è rappresentata dal vettore (1, 1), il documento J dal vettore (8, 2) e il documento K dal vettore (6, 1). L’obiettivo è determinare quale dei due documenti sia più vicino alla query.

Una delle strategie più diffuse per valutare la *vicinanza* tra vettori è quella basata sull’angolo tra essi. Più precisamente, si utilizza il *coseno dell’angolo* tra due vettori, che fornisce una misura della loro similarità direzionale. In questo contesto, due vettori ortogonali hanno coseno pari a zero e sono quindi considerati completamente dissimili, mentre due vettori perfettamente allineati (cioè coincidenti nella direzione) hanno coseno pari a uno e sono considerati massimamente simili.

L’uso del coseno, anziché di una semplice misura euclidea, è giustificato dal fatto che i documenti possono avere lunghezze molto diverse. Un documento lungo tenderà ad avere valori numerici più elevati nei suoi vettori, ma ciò non implica necessariamente una maggiore rilevanza rispetto a una query. Pertanto, si normalizza il prodotto scalare tra i vettori dividendolo per il prodotto delle loro norme (moduli). Questo accorgimento rende la misura indipendente dalla lunghezza assoluta dei vettori e focalizzata unicamente sulla loro direzione.

La formula utilizzata per il *coseno di similarità* tra due vettori A⃗ e B⃗ è la seguente:

Immagine che contiene Carattere, testo, numero, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

dove A⃗⋅B⃗ indica il prodotto scalare tra i due vettori e ∥A⃗∥ e ∥B⃗∥ rappresentano le loro rispettive norme euclidee. In questo modo, si ottiene un valore compreso tra 0 e 1, che rappresenta il grado di similarità tra il documento e la query, indipendentemente dalla loro lunghezza assoluta.

Immagine che contiene testo, Carattere, linea, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Nella rappresentazione vettoriale dei documenti, un aspetto centrale riguarda la pesatura dei termini, ossia la determinazione del contributo che ciascun termine fornisce alla descrizione numerica di un documento. In prima approssimazione, si potrebbe pensare di inserire nel vettore la frequenza con cui un termine compare all'interno del documento. Questo approccio, per quanto semplice e intuitivo, non è sufficiente a distinguere efficacemente i contenuti informativi, in quanto tende a dare troppo peso a parole molto frequenti ma poco significative.

Un'alternativa ancora più rudimentale prevede l’uso di vettori binari: in tal caso, ciascun termine è rappresentato con un valore pari a 1 se presente nel documento e 0 altrimenti. Tuttavia, anche questa modalità perde informazioni importanti legate alla quantità e alla rilevanza dei termini nel contesto specifico.

Per ottenere una rappresentazione più accurata, si adotta una strategia di pesatura più articolata, che si basa su due elementi fondamentali: la *frequenza del termine nel documento* (Term Frequency, TF) e la *rarità del termine nella collezione* (Inverse Document Frequency, IDF). L’idea alla base di questa combinazione è la seguente: un termine ricorrente in un documento specifico è rilevante, ma se è presente anche in gran parte della collezione, allora è poco discriminante. La *Term Frequency* misura quante volte un termine compare in un documento. Può essere rappresentata semplicemente dal numero di occorrenze del termine, oppure può essere normalizzata rispetto alla lunghezza del documento, per evitare che i documenti più lunghi risultino automaticamente più “pesanti”. A questa misura si affianca l'*Inverse Document Frequency*, che serve a penalizzare i termini troppo comuni e a valorizzare quelli rari. Questo valore è calcolato come il logaritmo del rapporto tra il numero totale dei documenti della collezione (solitamente indicato con N) e il numero di documenti in cui il termine compare almeno una volta (indicato con ​). Formalmente:

Immagine che contiene Carattere, bianco, simbolo, testo

Il contenuto generato dall'IA potrebbe non essere corretto.

Tanto più un termine è raro nella collezione, tanto maggiore sarà il valore della sua IDF. Il prodotto tra TF e IDF dà origine alla formula del *TF-IDF*, una delle modalità più efficaci e largamente adottate per la pesatura dei termini:

Immagine che contiene Carattere, tipografia, calligrafia, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa formulazione consente di assegnare un peso maggiore ai termini che sono frequenti in un documento ma rari nella collezione, rendendoli così ottimi indicatori di contenuto specifico.

Un’ulteriore considerazione riguarda il *preprocessing* del testo, una fase preliminare indispensabile per la costruzione di vettori rappresentativi. In questa fase si procede all’eliminazione delle cosiddette *stopword*, ovvero parole altamente frequenti ma scarsamente informative, come preposizioni, articoli, ausiliari e congiunzioni. L’eliminazione delle stopword può essere effettuata tramite liste predefinite oppure in modo dinamico, ad esempio scartando tutti i termini che superano una certa soglia di frequenza nella collezione.

Questa fase di filtraggio, che può essere vista come una generalizzazione dei meccanismi di normalizzazione del linguaggio naturale, permette di ridurre il rumore nei dati e focalizzare l’attenzione del sistema di indicizzazione su elementi realmente distintivi.

Immagine che contiene testo, Carattere, bianco, calligrafia

Il contenuto generato dall'IA potrebbe non essere corretto.

Il metodo di Rocchio è uno strumento sviluppato per affrontare compiti di classificazione testuale, particolarmente utile nei contesti in cui si dispone di esempi rappresentativi per ciascuna classe e si intende costruire un profilo medio che sintetizzi le caratteristiche distintive di ogni categoria. L’idea di fondo si basa sulla possibilità di rappresentare una classe mediante un vettore particolare, detto *profilo di classe* o *centroide*, che rappresenta il punto medio nello spazio vettoriale dei documenti appartenenti a quella classe. Questo profilo è costruito a partire dai vettori dei documenti che compongono una determinata categoria. Ad esempio, nel caso della classe *sport*, si possono selezionare dieci documenti che trattano esclusivamente argomenti sportivi. Ogni documento è rappresentato come un vettore nel consueto spazio delle feature, in cui ogni dimensione corrisponde a un termine del dizionario complessivo della collezione.

Il profilo della classe *sport* si ottiene calcolando la media aritmetica dei vettori dei documenti che appartengono a quella classe. In termini operativi, per ciascuna feature si sommano i valori corrispondenti nei vettori dei documenti positivi (cioè appartenenti alla classe) e si divide per il numero totale di documenti considerati. Il risultato è un nuovo vettore, il centroide, che non corrisponde a un documento reale, ma rappresenta una sorta di *documento ideale* o *prototipo* della classe.

Sia 𝐷 l’insieme dei documenti e 𝑇 il dizionario, ovvero l’insieme di tutti i termini diversi che compaiono nei documenti di 𝐷.  
La frequenza assoluta di un termine 𝑡 ∈ 𝑇 in un documento 𝑑 ∈ 𝐷 è indicata con tf(d, t).  
I vettori dei termini vengono rappresentati attraverso questa notazione.

Il centroide di un insieme di X vettori di termini è definito come la media di questi vettori, e rappresenta un punto centrale nello spazio vettoriale che sintetizza le caratteristiche comuni dei documenti considerati.

Immagine che contiene Carattere, bianco, diagramma, tipografia

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa operazione è concettualmente analoga a quella della prototipizzazione in altri domini cognitivi, dove si costruisce un esemplare medio che riassume le caratteristiche salienti di una categoria. Così come si può parlare di un cane “prototipico” o di una teglia da forno “prototipica”, anche nel contesto del testo si può individuare un vettore che sintetizza le caratteristiche più comuni dei documenti appartenenti a una data classe.

Il metodo di Rocchio non si limita alla costruzione del profilo positivo. Al contrario, integra anche l’informazione proveniente dai documenti *negativi*, cioè quelli appartenenti ad altre classi. L’idea è che un buon classificatore non solo deve premiare la vicinanza di un documento al profilo della classe positiva, ma deve anche penalizzare la vicinanza a classi alternative. In tal senso, si costruisce anche un *profilo negativo*, che può essere calcolato come la media dei vettori dei documenti che non appartengono alla classe in esame.

Formalmente, il nuovo vettore che rappresenta il criterio di classificazione può essere costruito come combinazione lineare del profilo positivo e del profilo negativo, secondo una formula del tipo:

Immagine che contiene testo, Carattere, linea, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

dove ​ è il centroide dei documenti appartenenti alla classe di interesse, ​ è il centroide dei documenti appartenenti alle classi alternative, e α e β sono parametri di bilanciamento che modulano l’importanza relativa dei due termini.

La rappresentazione dei pesi associati alle feature all’interno del metodo di Rocchio avviene in modo piuttosto lineare e intuitivo. Si parte dai vettori che costituiscono il profilo di una determinata classe, che sono ottenuti aggregando l’informazione dai documenti positivi, cioè quelli che appartengono alla classe che si intende modellare. Per ciascuna feature (termine) si calcola quanto essa sia rappresentativa dei documenti positivi, ma si tiene conto anche di quanto la stessa feature sia presente nei documenti negativi, ovvero quelli che appartengono ad altre classi. Operativamente, per una feature data — ad esempio la parola "calcio" nella classe *sport* — si considera la frequenza con cui compare nei documenti sportivi, e la si moltiplica per un fattore di normalizzazione, spesso la cardinalità (cioè il numero) dei documenti positivi. Questa informazione viene poi bilanciata sottraendo un contributo analogo derivante dai documenti negativi. Se "calcio" è molto presente nei documenti sportivi e poco presente altrove, il suo peso all’interno del profilo sarà alto. Al contrario, se la stessa parola è diffusa anche in altri contesti — come ad esempio nei documenti della classe *chimica*, dove "calcio" può essere usato per indicare l’elemento chimico — allora il suo peso complessivo verrà ridotto, riflettendo la minore capacità discriminativa del termine. Il calcolo dei pesi è dunque influenzato sia dalla frequenza relativa nei documenti positivi sia dalla frequenza nei documenti negativi. Questo meccanismo è regolato da due parametri, spesso indicati come α e β, che permettono di pesare diversamente il contributo positivo e quello negativo. In alcuni casi, si usa un solo parametro (ad esempio α), mentre il complemento (1 - α) bilancia automaticamente l’altra componente. La scelta di questi parametri ha un impatto significativo sulla sensibilità del modello e può essere oggetto di ottimizzazione empirica.

Per ciascuna classe 𝑐ᵢ (con 𝑖 che varia tra le classi), si calcola il peso della k-esima caratteristica 𝑓 mediante la seguente formula:

Immagine che contiene testo, Carattere, bianco, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

dove:

* ​ è l’insieme degli esempi positivi per la classe 𝑐ᵢ,
* ​ è l’insieme degli esempi negativi,
* rappresenta il vettore dei termini del documento j.

Va inoltre sottolineato che questa operazione si svolge in uno spazio vettoriale di dimensioni potenzialmente molto elevate, poiché ogni dimensione corrisponde a un termine del dizionario costruito sull’intera collezione documentale. Di conseguenza, il profilo di una classe è un vettore ad alta dimensionalità che codifica la rilevanza relativa di ciascun termine all’interno di quella classe. Infine, è importante notare che il metodo di Rocchio può essere combinato con tecniche di feature selection o riduzione dimensionale, per rendere il calcolo più efficiente e il modello più interpretabile. In pratica, si possono applicare filtri che eliminano le feature poco informative, come ad esempio le stopword, oppure si può fare uso di soglie sulla frequenza dei termini per limitare l’insieme delle dimensioni considerate.

Un elemento critico di questo processo è la definizione dei cosiddetti “documenti negativi”. In teoria, tutti i documenti che non appartengono alla classe target potrebbero essere considerati negativi. Tuttavia, in pratica, questa definizione può essere raffinata per migliorare la capacità discriminativa del classificatore. Ad esempio, si può introdurre il concetto di *neonegativi* (*neonegatives* o *neopositives* in alcune formulazioni), ovvero classi con cui la classe target è frequentemente confusa. Supponendo che la classe *sport* venga spesso confusa con *spettacolo*, sarà più utile penalizzare i termini comuni tra queste due classi piuttosto che tra *sport* e *politica*, se quest’ultima è più chiaramente distinta.

Una tecnica utile in questo contesto è l’analisi delle matrici di confusione. Esse mostrano, per ciascuna coppia di classi, quante volte un classificatore assegna erroneamente documenti da una classe all’altra. Analizzando queste matrici, è possibile individuare le coppie di classi più ambigue e utilizzare tali informazioni per costruire dei profili negativi mirati, anziché generici.

nel contesto della rappresentazione vettoriale dei documenti, la struttura sottostante è tipicamente una matrice in cui ogni riga corrisponde a un documento e ogni colonna a una feature (termine). Il contenuto di ciascuna cella rappresenta il peso attribuito a quel termine nel documento, secondo lo schema prescelto (frequenza assoluta, binaria, TF-IDF, o altra trasformazione). L’analisi di tali matrici consente di calcolare distanze tra documenti e di individuare similarità semantiche, con l’obiettivo finale di classificare correttamente i documenti e comprendere i contenuti trattati.

Nel contesto dell’analisi del linguaggio naturale, l’attenzione può spostarsi dalle righe alle colonne della matrice termine-documento, cambiando così la prospettiva da una rappresentazione centrata sui documenti a una centrata sui termini. In questa visione, ciascuna colonna della matrice può essere considerata come un vettore che descrive il comportamento di un termine all’interno del corpus. I termini che compaiono in contesti simili — ossia in documenti simili — tendono ad avere rappresentazioni vettoriali simili. Questa osservazione apre la strada allo studio di descrittori vettoriali dei termini stessi, piuttosto che dei documenti, fondando la base teorica di molti metodi distribuzionali del significato lessicale, tra cui i *word embeddings*.

Alla base di questa idea c’è un’intuizione che si è affermata già a partire dagli anni ’50, con i lavori fondamentali di linguisti come Zellig Harris e John Firth. Harris introduce il principio secondo cui “i termini che appaiono in contesti simili tendono ad avere significati simili”, mentre Firth sintetizza la stessa intuizione nel celebre enunciato: “You shall know a word by the company it keeps”. Questo significa che il significato di una parola può essere descritto e compreso analizzando i termini che le stanno attorno, ovvero il suo contesto d’uso.

Questa idea si concretizza nella costruzione di rappresentazioni vettoriali dei termini, in cui ciascun termine viene rappresentato come un vettore le cui componenti riflettono la presenza (o frequenza) di altri termini in prossimità. Il contesto viene solitamente definito come una finestra simmetrica di dimensione 2c2c2c, che include ccc termini a sinistra e ccc termini a destra di ogni occorrenza del termine target. Tale finestratura può essere fissa oppure adattiva e non necessariamente simmetrica.

Per costruire queste rappresentazioni, si parte da un insieme di frasi contenenti un termine specifico (ad esempio, “bagel”) e si registrano i termini che compaiono nel contesto definito. Dalle frequenze delle co-occorrenze tra il termine target e quelli del suo contesto si può costruire una tabella, che rappresenta la base della matrice delle co-occorrenze. Alcuni termini possono comparire frequentemente nel contesto del termine target (ad esempio “cheese” e “cream” con “bagel”), e questa ripetizione segnala una vicinanza semantica più marcata.

Si supponga di avere a disposizione tre frasi contenenti la parola *bagel* e di fissare 𝑐 = 3 (dove la dimensione di 𝑐 solitamente varia tra 10 e 20), come ad esempio:

* "A *bagel* and cream cheese"

• "also known as *bagel* with cream cheese"

• "American cuisine The *bagel* is typically sliced "

I conteggi aggregati delle parole per le frasi considerate sono riportati nella tabella.

Immagine che contiene testo, numero, Carattere, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Il vettore di co-occorrenza cattura effettivamente informazioni contestuali significative: la parola *bagel* risulta fortemente associata a *cream* e *cheese*, ma anche ad altri termini rilevanti nel contesto, come *cuisine* e *sliced*.

Questo tipo di rappresentazione vettoriale consente di modellare il significato delle parole sulla base del contesto in cui compaiono, e può essere espresso attraverso un vettore di co-occorrenza:



La presenza ripetuta di certi termini attorno a un dato target suggerisce una relazione contestuale stabile e significativa. Non tutte le co-occorrenze, però, hanno lo stesso valore informativo: alcune potrebbero riflettere semplicemente alta frequenza globale (come le *stopword*), mentre altre potrebbero segnalare una connessione semantica più pregnante. Per questo motivo è prassi comune rimuovere le *stopword* e applicare una fase di *lemmatizzazione*, che riconduce le forme flesse di un termine alla sua forma base o *lemma*, riducendo così la sparsità della rappresentazione.

L’elaborazione dei vettori di co-occorrenza si basa spesso su grandi collezioni di testi (corpora) di dimensioni anche molto elevate, indicati come aventi dimensione mmm, dove mmm rappresenta il numero di termini unici nel vocabolario. I vettori risultanti avranno quindi mmm dimensioni, anche se in seguito possono essere sottoposti a tecniche di riduzione dimensionale (come SVD) per rendere la rappresentazione più compatta e gestibile.

Questa impostazione costituisce il fondamento di molti metodi distribuzionali moderni, e permette di costruire modelli linguistici in grado di cogliere sfumature semantiche tra parole diverse ma affini, come “great” e “fantastic”. La somiglianza tra i loro vettori, derivata dalla similarità dei contesti in cui appaiono, è interpretata come evidenza della loro prossimità semantica. Tali modelli sono oggi alla base di sistemi di raccomandazione, analisi del sentiment, motori di ricerca semantici e altre applicazioni che richiedono comprensione automatica del testo.

I vettori di co-occorrenza rappresentano uno dei punti di partenza più comuni nell’elaborazione distribuzionale del significato linguistico. Per calcolarli si parte da grandi collezioni di documenti testuali, spesso composte da milioni di frasi, e si costruisce un vocabolario di dimensione mmm, dove ogni termine è descritto sulla base della frequenza con cui co-occorre con altri termini entro una finestra contestuale definita. Il risultato è un vettore ad alta dimensionalità — tipicamente con migliaia di feature — che rende difficile sia l’interpretazione sia la manipolazione diretta nello spazio vettoriale.

La gestione di vettori con 10.000 o 15.000 dimensioni risulta infatti computazionalmente onerosa e spesso inefficiente. Per superare questa complessità si ricorre alla riduzione dimensionale o alla costruzione diretta di rappresentazioni dense in spazi vettoriali di dimensione molto più contenuta. I vettori più diffusi nella pratica contemporanea sono quelli composti da circa 300 componenti. Non esiste una motivazione teorica forte per la scelta di questo valore specifico: si tratta di un numero determinato empiricamente, che ha mostrato buone prestazioni su una vasta gamma di compiti. Rappresentare i termini in uno spazio di 300 dimensioni consente un equilibrio tra espressività e generalizzazione, riducendo il rischio di *overfitting* e semplificando l’elaborazione successiva.

Il principio di fondo è sempre lo stesso: un termine è definito dal contesto in cui appare. Questo è il presupposto su cui si basa il modello *Word2Vec*, uno degli strumenti più influenti nello sviluppo degli *embeddings* lessicali. L’algoritmo mira a imparare rappresentazioni numeriche dei termini che riflettano la distribuzione dei loro contesti d’uso. In particolare, la variante *Skip-Gram* si concentra sul prevedere i termini contestuali dati un termine centrale.

Il funzionamento di *Skip-Gram* si fonda su un meccanismo di predizione binaria: la rete neurale addestrata riceve in input una coppia termine-contesto e deve stimare la probabilità che quella coppia sia effettivamente osservabile in un corpus reale. In pratica, invece di contare direttamente le co-occorrenze, si addestra un classificatore a riconoscere se due termini si trovano nello stesso contesto. A questa previsione si aggiunge la tecnica della *negative sampling*, che consente di semplificare e accelerare l’addestramento sostituendo il problema multiclasse con una serie di problemi binari.

Durante l’addestramento, la rete neurale ottimizza una rappresentazione vettoriale per ciascun termine del vocabolario. Quando il modello raggiunge un livello soddisfacente di apprendimento, è possibile “fotografare” lo stato degli *embeddings* ottenuti e utilizzarli come descrittori statici dei termini. Questi vettori catturano in modo compatto e significativo le regolarità semantiche e sintattiche apprese dal modello. Non è raro, ad esempio, osservare che termini simili vengano proiettati in prossimità nello spazio vettoriale.

Un vantaggio rilevante di questo approccio è che si tratta di una forma di apprendimento auto-supervisionato (*self-supervised*), in cui il modello apprende esclusivamente dai dati grezzi, senza la necessità di etichette esterne fornite manualmente. Ciò lo rende particolarmente scalabile e adatto a corpus molto grandi. A differenza dell’apprendimento supervisionato classico, in cui un annotatore umano assegna etichette corrette a ciascun esempio, il modello *Word2Vec* costruisce da solo gli esempi positivi e negativi attraverso la struttura stessa del testo. Questo tipo di apprendimento si è dimostrato estremamente efficace ed è stato alla base di molti sviluppi successivi nel campo della rappresentazione del linguaggio.

Nel contesto degli algoritmi di apprendimento auto-supervisionato (*self-supervised*), una delle innovazioni più significative consiste nella possibilità di utilizzare direttamente grandi quantità di testo non annotato come fonte di esempi per l’addestramento dei modelli. In particolare, nel caso della generazione di *word embeddings*, questi testi diventano una miniera di esempi positivi e negativi per la costruzione di rappresentazioni distribuzionali dei termini, senza che vi sia alcun bisogno di costosi processi di etichettatura manuale. Il meccanismo adottato dall’algoritmo *Skip-Gram* con *negative sampling* prevede un’operazione di classificazione binaria: si tratta, in sostanza, di addestrare un modello che impari a distinguere tra coppie di parole che appaiono effettivamente insieme nei testi (esempi positivi) e coppie artificiali generate sostituendo parole del contesto reale con altre prese a caso dal vocabolario (esempi negativi). L’addestramento avviene tramite regressione logistica, evitando l’uso di architetture neurali profonde e multilivello, con conseguente riduzione della complessità computazionale. L’intero processo può essere schematizzato in pochi passi essenziali. In primo luogo, si seleziona un termine centrale (*target word*) e i termini che lo circondano all’interno di una finestra contestuale prefissata — ad esempio, due parole a sinistra e due parole a destra — generando così gli esempi positivi. Successivamente, si effettua un campionamento automatico di termini che non appaiono nel contesto reale del termine centrale, i quali vengono utilizzati per creare gli esempi negativi. Per ciascuna coppia (termine centrale, contesto), il modello impara a stimare la probabilità che tale coppia sia effettivamente compatibile, aggiornando i pesi associati alle rappresentazioni vettoriali dei termini. Nel momento in cui l’addestramento è sufficientemente avanzato, è possibile estrarre i vettori che il modello ha appreso per ciascun termine. Questi vettori — detti *embeddings* — costituiscono rappresentazioni numeriche dense e significative, che possono essere utilizzate in numerosi compiti linguistici, dalla classificazione alla traduzione automatica. Per chiarire il funzionamento, si può fare riferimento a un esempio pratico. Si immagini di analizzare una frase qualsiasi, come: "Thou shalt not make a machine in the likeness of a human mind". Si supponga di considerare un contesto formato da due parole prima e due parole dopo la parola target.  
La frase viene analizzata tramite una finestra mobile (*sliding window*), in modo tale che ciascuna parola, a turno, venga trattata come parola target.

Questo approccio consente di esaminare sistematicamente tutte le combinazioni di parole nel contesto locale della target, permettendo di costruire vettori di co-occorrenza basati sui termini che compaiono vicino a ciascuna parola nella frase.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, numero, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Il modello non ancora addestrato viene utilizzato per prevedere le parole vicine a quella target.  
L’output è un vettore di previsione, in cui a ciascuna parola del vocabolario viene assegnata una probabilità. Questi vettori non solo permettono di confrontare i termini in base alla loro similarità distribuzionale, ma fungono anche da input per una grande varietà di modelli più complessi nel campo del Natural Language Processing.

In altre parole, il modello cerca di indovinare quali parole potrebbero trovarsi intorno a quella data, distribuendo le probabilità tra tutte le parole conosciute.

Immagine che contiene testo, schermata, Carattere, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

Nel processo di addestramento di un modello non supervisionato come *Skip-Gram with Negative Sampling*, inizialmente il modello non possiede alcuna conoscenza, dunque non è in grado di restituire una distribuzione significativa. Tuttavia, il meccanismo di apprendimento permette una correzione progressiva attraverso una procedura iterativa piuttosto lineare: da un lato si genera una predizione, dall’altro si calcola l’errore come scostamento tra il valore atteso (il *target*) e l’output effettivo del modello. Nel dettaglio, il valore atteso è rappresentato da un vettore binario in cui il termine corretto (cioè il *target* reale) assume valore 1, mentre tutti gli altri valori della distribuzione sono posti a zero. Questa rappresentazione consente di identificare in maniera netta la direzione dell’errore, il quale viene poi propagato all’indietro nel modello per aggiornare i pesi associati ai vettori. Attraverso successive iterazioni, il modello affina progressivamente le sue rappresentazioni, migliorando la capacità di predizione.

Immagine che contiene testo, schermata, diagramma, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, diagramma, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

L’addestramento, tuttavia, comporta un costo computazionale elevato. Questo dipende principalmente dalla necessità di considerare ogni posizione del testo e di applicare la finestra contestuale su ogni termine all’interno del corpus. Per esempio, con una finestra di dimensione cinque, ogni termine richiede l’elaborazione di cinque parole adiacenti, il che, su un corpus di grandi dimensioni, implica un numero significativo di operazioni. Per rendere il processo più efficiente e computazionalmente sostenibile, si adotta una semplificazione del task. Invece di prevedere direttamente la parola successiva a partire da una parola in ingresso, il modello si limita a stabilire se una coppia di parole (input word, output word) forma o meno un contesto plausibile. Il problema diventa, quindi, una valutazione binaria della coerenza semantica tra due termini: si restituisce un valore numerico che indica la probabilità che i due termini co-occorrono in un contesto reale. Si passa da una rete neurale a un modello di regressione logistica, che è più semplice e veloce da calcolare. In questo schema, ogni esempio positivo è costituito da una coppia effettivamente osservata nella finestra contestuale, mentre gli esempi negativi sono generati automaticamente selezionando parole che non fanno parte del contesto reale del termine centrale. La distinzione è netta: gli elementi che compongono effettivamente il contesto ricevono etichetta 1, mentre quelli inseriti artificialmente come distrattori ricevono etichetta 0.

Questa strategia è essenziale: in assenza di esempi negativi, il modello imparerebbe rapidamente a classificare tutte le coppie come plausibili, compromettendo completamente la sua capacità discriminante. Il bilanciamento tra esempi positivi e negativi è dunque fondamentale per permettere al modello di apprendere rappresentazioni vettoriali affidabili.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

In definitiva, l’algoritmo *Skip-Gram with Negative Sampling* si fonda su un principio di semplicità ed efficienza: da un’operazione complessa di predizione distribuzionale si passa a una classificazione binaria per coppie di termini, con l’obiettivo di apprendere, attraverso esempi positivi e negativi, rappresentazioni dense e semanticamente significative per ogni parola del vocabolario.

Nel processo di addestramento di un modello basato su *Skip-Gram with Negative Sampling*, si impiegano due matrici principali. Queste matrici rappresentano, rispettivamente, gli *embedding* delle parole e i vettori associati ai termini del contesto. Le dimensioni delle due matrici sono significative: una ha dimensione pari al prodotto tra la *vocabulary size* (cioè il numero totale di parole nel vocabolario) e la *embedding size* (la lunghezza del vettore di ciascuna parola), mentre l’altra ha struttura analoga, ma svolge la funzione di rappresentare i vettori associati ai contesti, piuttosto che alle parole centrali.

Immagine che contiene testo, schermata, linea, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

In generale, un vettore di embedding più lungo consente una maggiore espressività semantica: più dimensioni implicano una rappresentazione potenzialmente più precisa. Tuttavia, questo vantaggio comporta un costo computazionale maggiore, poiché l’aumento della dimensionalità implica più operazioni e una maggiore complessità nel calcolo dei gradienti.

Durante l’addestramento, entrambe le matrici vengono inizializzate con valori casuali. Per ciascun passo, si seleziona un termine centrale (ad esempio, “not”) e i relativi contesti osservati nella finestra, che costituiscono gli esempi positivi. A questi si affiancano alcuni esempi negativi, cioè parole selezionate casualmente dal vocabolario che non appartengono al contesto effettivo.

Immagine che contiene testo, schermata, diagramma, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Da queste coppie input-output si estraggono i rispettivi vettori: il vettore della parola centrale dalla matrice degli *embedding* e i vettori dei termini del contesto dalla matrice dei contesti. Il passo successivo consiste nel calcolare il prodotto interno tra il vettore della parola centrale e ciascun vettore del contesto, sia positivo che negativo. Il prodotto interno costituisce una misura di affinità semantica tra i vettori: maggiore è il valore, maggiore è la similarità stimata tra i due termini.

Immagine che contiene testo, schermata, diagramma, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Tutti i risultati dei prodotti scalari vengono poi passati attraverso una funzione logistica (sigmoide), che restituisce un valore compreso tra 0 e 1, interpretabile come probabilità.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

A questo punto, si confronta l’output del modello con il target effettivo: 1 per le coppie positive, 0 per quelle negative. L’errore, calcolato come differenza tra valore atteso e valore ottenuto, viene utilizzato per aggiornare i pesi delle matrici. L’aggiornamento avviene tramite retro propagazione del gradiente, e il processo viene reiterato su tutto il dataset per più epoche, in modo da convergere a una configurazione ottimale dei vettori.

Immagine che contiene testo, schermata, numero, diagramma

Il contenuto generato dall'IA potrebbe non essere corretto.

La fase di apprendimento può essere formalizzata considerando ogni parola centrale www e i suoi termini di contesto ccc all’interno di una finestra simmetrica (ad esempio, due termini a sinistra e due a destra). L’obiettivo è massimizzare la probabilità che il termine ccc appartenga al contesto della parola www, e allo stesso tempo minimizzare la probabilità che termini negativi, selezionati casualmente, siano erroneamente classificati come appartenenti al contesto. In sostanza, il modello cerca di ottimizzare una funzione obiettivo che spinge i vettori di parole che compaiono insieme ad avvicinarsi nello spazio semantico, e quelli che non compaiono insieme ad allontanarsi. Questa approssimazione si basa sul calcolo del prodotto interno tra i vettori, seguito dalla trasformazione logistica, e costituisce il cuore operativo del modello.

**Lezione 9**

Quando si lavora con grandi corpus testuali contenenti miliardi di parole, l’efficacia del modello *Skip-Gram with Negative Sampling* si manifesta con chiarezza. L’idea fondamentale alla base del modello, e si traduce operativamente, dal punto di vista implementativo, nell’uso di due matrici: la matrice degli *embedding* WWW, che rappresenta i vettori delle parole centrali (quelle che vogliamo rappresentare), e la matrice del contesto CCC, che rappresenta i vettori delle parole che compaiono nei dintorni delle parole centrali. Al termine della fase di apprendimento, possono essere applicate diverse forme di post-elaborazione. Ad esempio, i vettori corrispondenti a uno stesso termine presenti in entrambe le matrici possono essere sommati per ottenere una rappresentazione più robusta; oppure, si può mantenere una sola delle due matrici come set finale di descrittori numerici per l’intero vocabolario. In entrambi i casi, il risultato è una rappresentazione vettoriale compatta e computazionalmente efficiente delle parole.

Ciò che risulta particolarmente rilevante di questo approccio è che, pur operando con strumenti matematici apparentemente distanti dalla semantica tradizionale — come quella di tipo formale, ad esempio la semantica di Montague — i vettori ottenuti riescono comunque a catturare elementi significativi della semantica lessicale. Si tratta di una semantica emergente, che si sviluppa attraverso l’analisi statistica dei co-occurrences delle parole nei testi. I vettori ottenuti descrivono gli “atomi” della lingua, intesi come unità minime di significato lessicale osservate nel loro contesto.

Questa capacità di cogliere informazioni semantiche rende possibile una serie di operazioni che preludono all’accesso semantico vero e proprio. Tra queste, le più comuni sono il calcolo della similarità semantica tra parole e la risoluzione di analogie. La similarità viene spesso valutata mediante la distanza coseno tra due vettori, mentre le analogie sfruttano il principio secondo cui certe relazioni semantiche si mantengono costanti nello spazio vettoriale. Un esempio celebre è la relazione tra un paese e la sua capitale: la differenza vettoriale tra “Cina” e “Pechino” rispecchia, in modo sorprendente, quella tra “Portogallo” e “Lisbona”. Questo consente di formulare domande del tipo: *“Pechino sta alla Cina come x sta al Portogallo?”*, e ottenere la risposta corretta sulla base di un semplice calcolo vettoriale.

Tuttavia, uno degli aspetti più interessanti e problematici di questi vettori è la loro capacità di riflettere — e talvolta amplificare — i *bias* presenti nei dati di addestramento. È stato osservato, ad esempio, che le relazioni stereotipate e discriminatorie presenti nei testi si riflettono nei vettori: così come “uomo” sta a “medico”, “donna” sta a “infermiera”, una corrispondenza che evidenzia uno squilibrio di genere strutturale. Questi bias non sono inseriti deliberatamente nel modello, ma emergono dai dati. Di conseguenza, i vettori possono incorporare pregiudizi culturali, sociali o razziali, rivelando in maniera implicita aspetti profondi della società che ha prodotto quei testi.

Questa osservazione ha avuto un impatto notevole in ambito etico e legale, portando a interrogativi circa l’uso di tali modelli in contesti sensibili come la selezione del personale o l’analisi automatica dei curriculum. In certi casi, è stato ipotizzato che vettori contenenti bias impliciti potessero influenzare negativamente la valutazione di candidati in base a genere, etnia, orientamento sessuale o convinzioni religiose.

Da qui, è emersa un’intera area di ricerca dedicata al *debiasing*, ovvero alla rimozione sistematica dei bias dai vettori. Al tempo stesso, i bias stessi sono diventati oggetto di studio: osservare le distorsioni presenti nei vettori consente di riflettere, in modo quantitativo, sulle forme di discriminazione latenti nei testi e nei gruppi sociali che li hanno generati. È possibile, ad esempio, studiare le rappresentazioni testuali prodotte da una specifica comunità (come, in casi estremi, gruppi con ideologie razziste) per ottenere una mappa dei loro pregiudizi impliciti. Questo tipo di analisi può essere condotto con rigore metodologico, offrendo strumenti inediti per comprendere meglio la struttura ideologica e culturale di una società.

Oltre a Word2Vec, esistono numerosi altri modelli di *word embeddings* che condividono la stessa filosofia di rappresentazione distribuita delle parole, ma si distinguono per le tecniche di apprendimento adottate. Word2Vec, storicamente il primo ad aver avuto un impatto significativo, resta un punto di riferimento essenziale per comprendere i modelli successivi. La comprensione del suo meccanismo permette infatti di interpretare con buona approssimazione anche il funzionamento di altri algoritmi, come GloVe e FastText, che rappresentano evoluzioni concettuali e tecniche della stessa idea di base.

Tutti questi modelli producono rappresentazioni vettoriali distribuite, ossia rappresentazioni numeriche delle parole codificate come liste di valori. Al di là del formato di salvataggio — che può variare nei dettagli, ad esempio nell’uso di tabulazioni o separatori — ciascun file di embeddings segue una struttura comune: ogni riga presenta una parola seguita da una sequenza di numeri a virgola mobile che ne costituiscono la descrizione.

FastText, introdotto da Facebook AI Research nel periodo successivo al 2016, è particolarmente interessante per l’innovazione che introduce rispetto alla gestione delle parole rare o sconosciute. Il modello affronta direttamente il problema delle *out-of-vocabulary words* (parole non viste durante il training), che nei modelli precedenti costituiva una limitazione significativa. Per risolvere tale problema, FastText utilizza un meccanismo di rappresentazione basato su unità sub-lessicali, ovvero segmenti più piccoli delle parole, come prefissi, suffissi o sottostringhe. Questo approccio è noto come *subword modeling*.

Nel caso di FastText, ogni parola viene rappresentata non solo dal proprio vettore, ma anche dai vettori dei suoi n-grammi di caratteri. Ad esempio, per la parola “where”, il modello include anche rappresentazioni di segmenti come “whe”, “her”, “ere”, e così via. In questo modo, anche termini mai osservati durante il training possono comunque ricevere una rappresentazione numerica coerente, derivata dalla combinazione dei segmenti che li compongono. Questo consente al modello di gestire efficacemente fenomeni come la derivazione morfologica, le inflessioni o le varianti ortografiche, risolvendo in maniera elegante i problemi di *sparsità* e *coverage* linguistico.

Un ulteriore vantaggio di FastText è la sua disponibilità in più di 150 lingue, rendendolo uno strumento prezioso per lo sviluppo di applicazioni multilingue, soprattutto in contesti dove le risorse linguistiche sono scarse o frammentarie. Sebbene le prestazioni possano variare tra le diverse lingue in funzione della qualità e della quantità dei dati di addestramento, la disponibilità di modelli preaddestrati ne facilita l’utilizzo in numerosi ambiti applicativi.

GloVe (Global Vectors for Word Representation), sviluppato da Stanford, adotta un approccio differente rispetto a Word2Vec e FastText. Si basa sull’analisi delle co-occorrenze globali delle parole in un corpus, sfruttando una matrice di co-occorrenza che viene fattorizzata per ottenere rappresentazioni dense delle parole. A differenza di Word2Vec, che costruisce i vettori in modo predittivo attraverso una finestra scorrevole di contesto, GloVe si fonda su un'ottimizzazione esplicita della relazione tra parole, cercando di modellare matematicamente la probabilità che una parola appaia nel contesto di un’altra.

Anche nel caso di GloVe, i vettori risultanti sono solitamente di 100, 200 o 300 dimensioni, e la dimensione della finestra contestuale è un parametro regolabile che influisce sulla qualità e sul tipo di relazioni catturate dal modello. La scelta della dimensionalità e della finestra dipende dagli obiettivi specifici dell'applicazione e dal bilanciamento tra precisione e costi computazionali.

L’utilizzo di questi vettori preaddestrati è reso particolarmente agevole dalla libreria **Gensim**, una delle più diffuse nell’ambito del *natural language processing* in Python. Gensim fornisce implementazioni ottimizzate per il caricamento, la manipolazione e la ricerca nei modelli di embeddings, e supporta in modo nativo i formati di Word2Vec, GloVe e FastText. La libreria consente di gestire in memoria anche centinaia di migliaia di vettori ad alta dimensionalità (tipicamente 300 dimensioni), facilitando l’integrazione di questi strumenti in sistemi di ricerca semantica, classificazione testuale o generazione automatica del linguaggio.

Una volta reso disponibile il file contenente i vettori preaddestrati, l’interazione con tali risorse può avvenire in maniera estremamente semplice. È sufficiente indicare il percorso del file per caricare i dati tramite classi apposite, come ad esempio KeyedVectors nella libreria Gensim. Questo oggetto consente di accedere in modo immediato e trasparente all’intero spazio vettoriale e di svolgere operazioni semantiche complesse con una sola chiamata di funzione.

Nel caso in esame, si è utilizzato un set di vettori preaddestrati composto da 400.000 elementi, ciascuno rappresentato con una dimensionalità di 100 (o 300, a seconda del file scelto, come indicato dal nome stesso del file). Una delle operazioni principali che si possono eseguire su questi oggetti è la ricerca dei termini semanticamente più simili a una parola data. Si può specificare anche il numero di risultati da restituire, attraverso un parametro opzionale. In assenza di indicazioni esplicite, il valore di default è impostato a 10, per cui, ad esempio, la richiesta di termini simili a *cactus* produrrà un array di 10 elementi ordinati per similarità.

Va notato che l’output può includere varianti morfologiche del termine interrogato, oppure forme vicine semanticamente, a seconda di come è stato costruito e addestrato il modello di partenza. Se, ad esempio, la risorsa utilizzata è stata semplificata o normalizzata durante la fase di preprocessing, le corrispondenze ritornate possono riflettere scelte specifiche di rappresentazione.

Un’altra operazione particolarmente rilevante resa possibile dall’uso degli *embeddings* è l’analogia semantica. Grazie alla struttura dello spazio vettoriale, è possibile formulare analogie del tipo: “king sta a man come queen sta a woman”. In questo caso, la distanza vettoriale che separa *king* da *man* è approssimativamente la stessa che separa *queen* da *woman*. Tali relazioni possono essere espresse tramite operazioni aritmetiche sui vettori, specificando elementi positivi (da sommare) e negativi (da sottrarre) rispetto ai quali calcolare la similarità.

Ad esempio, la funzione può essere invocata indicando positive=["queen", "man"], negative=["king"], ottenendo così come risultato *woman*, che riflette la relazione cercata. Questa tipologia di interrogazione risulta estremamente potente e viene sfruttata anche per analogie di tipo geografico, come nel caso della relazione tra uno stato e la sua capitale.

La logica sottostante a tali operazioni può essere illustrata anche graficamente, utilizzando riduzioni dimensionali e tecniche di visualizzazione in cui si assegnano colori simili a vettori con pesi simili. Tale rappresentazione rende visibile la prossimità semantica tra termini, pur rimanendo spesso opaca e non interpretabile a livello umano. Le componenti latenti dei vettori, infatti, non hanno un significato immediatamente riconoscibile, ma riflettono distribuzioni statistico-semantiche apprese dai dati.

Nonostante questa opacità interpretativa, lo spazio semantico così costruito permette di cogliere affinità tra termini anche in assenza di supervisione o di conoscenza esplicita delle relazioni. Le analogie tra termini come *regina* e *donna*, o *ragazzo* e *ragazza*, emergono come espressione spontanea delle regolarità linguistiche presenti nel corpus di addestramento.

Infine, è utile osservare che tutti questi strumenti offrono interfacce implementative estremamente semplici e pronte all’uso, adatte a essere integrate direttamente in flussi di lavoro più complessi o applicazioni finali.

I **Sense Embeddings** e le **metriche di similarità semantica** sono due concetti profondamente connessi ma distinti, ciascuno con finalità e modalità operative specifiche. Il lavoro su queste due dimensioni nasce dall’esigenza di superare i limiti dei modelli che si basano esclusivamente sulla rappresentazione vettoriale dei termini, per approdare invece a rappresentazioni più raffinate, orientate al significato. L’adozione di risorse lessicali e vettoriali orientate al senso è motivata dalla necessità di distinguere tra le diverse accezioni che una parola può assumere in contesti differenti. Per esempio, un termine come “banca” può indicare un istituto finanziario oppure una riva fluviale. Le rappresentazioni tradizionali, come i Word Embeddings, non sono progettate per cogliere questa differenziazione, poiché assegnano un unico vettore a ciascun lemma. Da qui nasce l’interesse per i Sense Embeddings, che associano un vettore a ogni senso distinto di un termine, offrendo una risoluzione semantica più fine.

Il primo elemento fondamentale da trattare riguarda le risorse lessicali su cui questi modelli si basano. Le risorse di riferimento sono strutture grafiche organizzate semanticamente, note anche come reti semantiche. Tali risorse modellano la conoscenza lessicale attraverso nodi (che rappresentano concetti o sensi) e archi (che rappresentano relazioni semantiche tra i nodi), come per esempio *IS-A*, *PART-OF*, o *USED-FOR*. Una delle reti più conosciute è **WordNet**, che si presenta come una gerarchia quando si considerano le relazioni di iperonimia e iponimia (cioè superclasse e sottoclasse), ma diventa un grafo a tutti gli effetti nel momento in cui si considerano relazioni semantiche più articolate. Il concetto di **synset** (insieme di sinonimi che condividono lo stesso significato) costituisce la base per la strutturazione semantica in WordNet. Ogni synset rappresenta un senso distinto e viene utilizzato per distinguere tra significati diversi associati a un medesimo lemma. A livello linguistico, questo approccio affronta la polisemia, ma in realtà essa è un contenitore complesso che include fenomeni molto diversi: significati figurati, metonimie, e forme complesse di referenza. Un esempio emblematico è quello della "Casa Bianca", che può essere intesa sia come edificio fisico sia come entità istituzionale e politica. In questi casi, le parole assumono significati multipli a seconda del contesto, e la rete semantica aiuta a distinguere e rappresentare ciascuna accezione. Il passaggio a una modellazione vettoriale dei sensi implica anche una riflessione sulle metriche da adottare per confrontare tali rappresentazioni. La **cosine similarity** è lo standard de facto per misurare la vicinanza tra vettori nello spazio semantico. Tuttavia, nel contesto dei Sense Embeddings, si è osservato che, in alcuni casi, metriche alternative possono risultare più efficaci, pur essendo altrettanto semplici da calcolare. Queste metriche alternative possono essere progettate per tenere conto di aspetti specifici delle rappresentazioni sensibili al contesto, superando in alcuni ambiti le performance della cosine similarity.

Sono stati introdotti strumenti particolarmente potenti, in grado di supportare numerosi task linguistici grazie alla disponibilità di descrizioni vettoriali numeriche. Queste rappresentazioni, oltre ad essere efficienti, sono relativamente semplici da utilizzare. L'aspetto interessante è che, a differenza delle risorse lessicali tradizionali che richiedono anni di lavoro manuale – come nel caso di WordNet, che copre esclusivamente l’inglese – i modelli di tipo distribuzionale, come quelli sviluppati da FastText, offrono copertura per ben 157 lingue. Questo elemento rende evidente l’importanza della dimensione multilinguistica e dell’automatizzazione nella costruzione di risorse linguistiche moderne.

Le rappresentazioni distribuzionali si fondano su un assunto semplice ma efficace: termini che compaiono in contesti simili tendono ad avere significati simili. Per esempio, in frasi come *"ho mangiato il pranzo al tavolo"* oppure *"ho mangiato il pranzo al banco"*, le parole *tavolo* e *banco* condividono una similarità semantica data dal contesto d’uso. Questa ipotesi costituisce il fondamento delle tecniche distribuzionali, permettendo la costruzione di vettori semantici in grado di catturare simili significati d’uso.

Nel confronto tra rappresentazioni basate su reti semantiche e rappresentazioni distribuzionali emergono vantaggi e svantaggi specifici. Le reti semantiche, come WordNet, si distinguono per la loro altissima specificità: offrono una descrizione dettagliata e sistematica dei sensi di una lingua, con una precisione definitoria che difficilmente trova eguali. Nonostante possa accadere che manchi qualche senso specifico, WordNet resta una risorsa uniforme e ben strutturata. In aggiunta, la natura profondamente linguistica di queste risorse consente di associare a uno stesso senso termini in lingue diverse, come dimostrato da progetti come *BabelNet*, sviluppato in Italia presso l’Università La Sapienza di Roma. Questo tipo di risorsa preserva la struttura concettuale (i nodi e le relazioni) e vi sovrappone le lessicalizzazioni in lingue differenti, creando un ponte tra rappresentazioni concettuali e multilinguismo. Uno dei principali limiti delle reti semantiche riguarda però la difficoltà nel confrontare i sensi. Anche adottando metriche più sofisticate, come il confronto tra sottoalberi anziché tra singoli nodi, resta il fatto che la vicinanza semantica viene misurata in funzione della distanza tra i nodi nel grafo. Questo approccio presenta delle criticità, poiché la densità del grafo può variare notevolmente a seconda della zona considerata. Alcune aree della rete – per esempio, quelle legate al mondo naturale – sono meno dettagliate rispetto ad altre, come quelle relative agli artefatti o agli strumenti tecnologici. Tale disomogeneità implica che una distanza fissa (ad esempio, tre passi) può rappresentare significati semanticamente molto più distanti in un’area della rete rispetto a un’altra. Questo rende le metriche strutturali potenzialmente inaffidabili.

Dal punto di vista operativo, le reti semantiche sono meno adatte ad applicazioni downstream, ovvero a task come classificazione, traduzione automatica, riassunto e simili. La loro struttura teoricamente solida risulta però più difficile da integrare in pipeline di elaborazione automatica rispetto ai modelli vettoriali.

Le rappresentazioni distribuzionali, invece, presentano una caratteristica intrinseca che costituisce al contempo un vantaggio e un limite: comprimono tutti i sensi di una parola in un unico vettore. Se da una parte ciò semplifica l’elaborazione, dall’altra comporta la perdita di informazioni legate alla polisemia. Inoltre, la qualità degli embeddings dipende fortemente dalla disponibilità di corpora linguistici estesi. In assenza di grandi set di dati (come nel caso del *Gigaword* per l’inglese), l’addestramento può risultare incompleto o distorto.

Tuttavia, una volta ottenuti i vettori, il confronto tra significati diventa semplice ed efficiente. Questa semplicità operativa ha favorito l’adozione degli embeddings nelle applicazioni di NLP, poiché garantiscono prestazioni superiori nei task downstream rispetto alle risorse strutturate. È questo il motivo per cui tali rappresentazioni sono oggi largamente utilizzate nei modelli più avanzati, come i transformer.

Infine, si è accennato a *BabelNet*, che può essere considerata una versione estesa e multilingue di WordNet. Essa offre una copertura per circa 300 lingue e permette di recuperare i sensi associati a un termine indipendentemente dalla lingua in cui è espresso. Il principio alla base è lo stesso: a un input lessicale viene associato un insieme di significati strutturati, ma con la potenza aggiuntiva di una copertura linguistica ampia e flessibile. BabelNet rappresenta così un esempio emblematico di come sia possibile integrare risorse lessicali tradizionali con grandi corpus multilingue, sfruttando la complementarità tra approccio lessicale e rappresentazione automatica.

Nel panorama delle rappresentazioni linguistiche, si delineano due approcci distinti: da un lato le rappresentazioni orientate al senso, che offrono una descrizione semantica fine ma non si appoggiano su strutture vettoriali continue; dall’altro le rappresentazioni orientate ai termini, che forniscono vettori numerici utili all’elaborazione automatica, ma che tendono a mescolare i molteplici sensi associati a uno stesso lemma. Questo limite ha motivato la progettazione di una risorsa capace di produrre rappresentazioni vettoriali non più riferite semplicemente ai termini, ma ai sensi. In questo contesto, le due risorse di riferimento impiegate sono *BabelNet* e *Numberbatch*. Il processo si basa su un’operazione concettualmente semplice: a partire da un senso, si mira a costruire un vettore che lo rappresenti nel medesimo spazio in cui si trovano i vettori terminologici. Per fare questo, si accede a *BabelNet* per individuare un determinato senso, che tipicamente è rappresentato da un insieme di lemmi – ossia di termini che condividono lo stesso significato in diverse lingue o varianti. Ogni termine che lexicalizza il senso viene poi associato al proprio vettore tramite *Numberbatch*. L’idea centrale è che, aggregando i vettori terminologici dei lemmi associati a un certo senso, si possa costruire un vettore rappresentativo del senso stesso. Questo vettore di senso non è altro che il risultato della combinazione – spesso una media – dei vettori dei termini corrispondenti.

Una volta costruiti, questi vettori di senso possono essere visualizzati e analizzati tramite tecniche di riduzione dimensionale, come l’analisi delle componenti principali (PCA). La proiezione risultante mostra che i sensi tendono a raggrupparsi naturalmente nello spazio vettoriale: i termini associati a uno stesso senso si distribuiscono con una prossimità coerente, confermando che la rappresentazione vettoriale così costruita mantiene una certa coesione semantica.

Un primo aspetto rilevante di questo approccio è che i vettori di senso risultano posizionati nello stesso spazio dei vettori terminologici. I sensi, cioè, vengono mappati in uno spazio semantico già occupato dai termini, facilitando quindi l’integrazione di queste rappresentazioni nei modelli di linguaggio esistenti e nei task downstream.

L’approccio mostra la sua efficacia anche in contesti multilinguistici. Considerando un esempio concreto, si può osservare il caso del termine *gate*. In italiano, il termine viene usato quasi esclusivamente con il significato di accesso aeroportuale. In inglese, invece, *gate* può riferirsi sia a un cancello fisico (di una casa, per esempio) sia a una porta logica in ambito informatico. Quando si interroga *BabelNet* per ottenere i sensi del termine, si ottengono gruppi distinti di termini associati a ciascuna accezione. Ogni gruppo viene quindi associato ai propri vettori tramite *Numberbatch*, e da questi si genera un vettore di senso specifico.

Nel caso del *gate* aeroportuale, il vettore risultante sarà orientato nella direzione semantica di termini come *airport*, *terminal* e altri correlati. Viceversa, per il *gate* inteso come cancello, la rappresentazione verrà costruita a partire da termini come *barrier*, *fence*, *gateway* e simili. Anche se il vettore originario di *gate*, ricavato direttamente da *Numberbatch*, è unico e ingloba tutti i possibili sensi, il processo di disambiguazione vettoriale consente di derivare versioni specifiche del vettore che corrispondono a ciascun significato distinto.

In questo modo, si ottengono vettori disambiguati e orientati al senso, capaci di cogliere la differenza semantica tra accezioni che altrimenti verrebbero fuse in una sola rappresentazione. Questa metodologia offre un ponte tra la precisione semantica delle risorse strutturate e la flessibilità computazionale delle rappresentazioni distribuzionali.

Il meccanismo descritto si fonda su un utilizzo consapevole dei legami cross-linguistici, propri della struttura stessa di *BabelNet*. I termini raccolti all'interno di uno stesso synset – anche se appartenenti a lingue diverse – vengono inclusi nella costruzione del vettore di senso. Questa scelta permette di sfruttare appieno la ricchezza della rappresentazione multilinguistica. Tuttavia, è importante osservare che, come già discusso in fase preliminare, possono emergere disallineamenti semantici tra le lingue: un termine in una lingua può avere un'estensione o una sfumatura concettuale differente rispetto alla sua controparte in un'altra lingua. Nonostante ciò, l'inclusione di tutte le lexicalizzazioni disponibili in un synset consente una copertura ampia e una maggiore robustezza del vettore risultante.

La medesima procedura viene applicata anche nel caso di concetti astratti o altamente specializzati, come nel caso delle “porte logiche” in ambito computazionale. Alla fine di questo processo, si ottengono vettori numerici che rappresentano non più i termini, ma i *bubble synset*, ossia delle unità concettuali definite da BabelNet. Questi vettori di senso non solo condividono lo spazio semantico dei tradizionali vettori terminologici (ad esempio quelli derivati da Word2Vec, GloVe o ConceptNet Numberbatch), ma possono essere trattati con le stesse operazioni computazionali: misurazioni di similarità, clustering, inferenza semantica.

Da un punto di vista applicativo, la transizione da vettori terminologici a vettori di senso è resa estremamente agevole: i vettori di senso sono compatibili per formato e dimensionalità con quelli preesistenti e possono essere inseriti nei sistemi in modo trasparente. Sono stati valutati anche altri metodi alternativi per la composizione dei vettori, ma si è concluso che la metodologia proposta, basata sulla media ponderata dei vettori dei termini lexicalizzanti un senso, rappresenta un compromesso ottimale tra semplicità ed efficacia.

Da un punto di vista applicativo, la transizione da vettori terminologici a vettori di senso è resa estremamente agevole: i vettori di senso sono compatibili per formato e dimensionalità con quelli preesistenti e possono essere inseriti nei sistemi in modo trasparente. Sono stati valutati anche altri metodi alternativi per la composizione dei vettori, ma si è concluso che la metodologia proposta, basata sulla media ponderata dei vettori dei termini lexicalizzanti un senso, rappresenta un compromesso ottimale tra semplicità ed efficacia.

La risorsa risultante è stata descritta anche attraverso alcune statistiche aggregate, che ne illustrano la copertura rispetto alle diverse categorie grammaticali. Per la valutazione empirica sono stati condotti due tipi di test: uno intrinseco e uno estrinseco. Il primo è volto a valutare direttamente la qualità della rappresentazione in sé, mentre il secondo misura l'impatto dei nuovi vettori all'interno di task linguistici concreti, confrontandoli con quelli ottenuti da altre risorse distribuzionali.

Il primo compito di riferimento è stato quello della *semantic similarity*. Si è utilizzato un dataset contenente coppie di parole annotate manualmente in base al grado di similarità semantica. Le stesse coppie sono state processate utilizzando i vettori di senso, calcolando per ciascuna la similarità tramite il coseno. L'output è stato confrontato con i punteggi umani attraverso indici di correlazione standard, come Pearson e Spearman. Tuttavia, questo approccio presenta un limite strutturale: in risorse come BabelNet, che include milioni di item acquisiti automaticamente da fonti come Wikipedia, le entità tendono a dominare il calcolo della similarità. Questo accade perché, data la densità e la varietà di entità presenti (persone, luoghi, istituzioni, ecc.), la similarità massima tende a emergere tra entità affini ma concettualmente lontane dal dominio lessicale astratto.

Un esempio emblematico riguarda la coppia *teacher* e *student*: mentre per l’essere umano i due termini appaiono naturalmente correlati, il sistema basato su similarità del coseno può restituire un valore elevato per due entità specifiche omonime, ma semanticamente scollegate dal concetto generale. Questo fenomeno suggerisce la necessità di una metrica alternativa.

Per rispondere a questa esigenza, è stata sviluppata una nuova misura di similarità – denominata *anchored similarity* – che si fonda su due componenti: una che valuta la posizione relativa dei vettori nella distribuzione delle distanze (rango), e una che misura la presenza effettiva dei termini associati ai sensi. Un parametro consente di bilanciare il peso delle due componenti. Questa metrica è calcolabile solo in uno spazio vettoriale misto, dove convivono rappresentazioni terminologiche e rappresentazioni di senso. In questo schema, i vettori dei termini fungono da riferimento e supporto nel calcolo delle similarità tra vettori di senso.

Tale metrica è stata testata su diversi dataset di riferimento e confrontata con sistemi e risorse allo stato dell’arte. I risultati ottenuti sono stati ottimali in molteplici configurazioni. Inoltre, si è proceduto con una valutazione estrinseca utilizzando task di *text similarity*. In particolare, sono state considerate coppie di testi annotate da giudici umani rispetto al grado di similarità semantica. È stata utilizzata una rete neurale precedentemente vincitrice in un concorso specifico, modificandola solo nella componente dei vettori: quelli di GloVe sono stati sostituiti con i nuovi vettori di senso. Il sistema così configurato è stato sottoposto a test su diverse coppie linguistiche: inglese-inglese, spagnolo-spagnolo, arabo-inglese.

La sperimentazione ha mostrato che, senza necessità di riaddestramento dei vettori, la risorsa sviluppata è in grado di mantenere prestazioni competitive anche in contesti cross-linguistici. Questo risultato è reso possibile dalla natura intrinsecamente multilingue di BabelNet, che garantisce l’allineamento semantico tra le lingue e consente una gestione uniforme dei sensi attraverso le diverse varianti linguistiche.

**Lezione 10**

Per completare la trattazione sui Sense Embedding, si può partire dalla descrizione di un tipico task di similarità semantica. In questo contesto, due termini vengono messi a confronto — per esempio Greeting e Wave — e il sistema è incaricato di produrre un valore numerico che esprima quanto i due termini risultano semanticamente affini. La similarità viene spesso espressa come un valore compreso tra 0 e 1, in cui il numero più alto indica una maggiore prossimità semantica.

Tipicamente, questi task si basano su dataset annotati da esseri umani, i quali hanno fornito giudizi di similarità tra coppie di termini. I sistemi computazionali sono quindi valutati sulla base della loro capacità di approssimare tali giudizi. Un esempio classico è quello in cui la coppia Greeting–Wave viene valutata tenendo conto del fatto che Wave può avere più significati: può indicare il gesto del saluto, ma anche un fenomeno atmosferico come un’ondata di freddo. Il sistema, pertanto, deve riuscire a comprendere il contesto e a selezionare il significato più plausibile tra quelli disponibili, disambiguando i termini.

L’annotatore umano, nel momento in cui assegna un valore di similarità a una coppia, esegue implicitamente una serie di operazioni: seleziona i sensi appropriati per ciascun termine, li confronta, e infine esprime un giudizio. Tuttavia, per lungo tempo, questa componente fondamentale è rimasta implicita e inesplorata nei processi di valutazione: nessuno si è realmente posto il problema di individuare in modo sistematico i sensi specifici che stanno alla base dei giudizi umani di similarità.

A partire da questa constatazione, è stato sviluppato un nuovo approccio che cerca di esplicitare i sensi coinvolti in ciascuna annotazione. A tal fine, è stato utilizzato un dataset standard, tipicamente impiegato in challenge promosse dalla CIEL, un’organizzazione internazionale che organizza task di valutazione su varie problematiche linguistiche. L’idea è stata quella di estendere tali dataset, aggiungendo un’annotazione manuale dei sensi coinvolti in ciascuna coppia di termini, per comprendere meglio la relazione semantica giudicata dagli annotatori.

Questa attività, seppur metodologicamente semplice, si è rivelata onerosa sia in termini di tempo che di risorse, anche perché richiede l’intervento di più annotatori e la successiva valutazione del livello di accordo tra di essi. Inoltre, è emersa una questione rilevante legata alla difficoltà intrinseca di certe annotazioni: quando i sensi dei termini sono troppo ambigui o quando i significati appartengono a domini molto diversi (come nel caso delle entità), l’operazione diventa particolarmente complessa.

Nel caso specifico, si è visto come termini come Wave possano assumere una molteplicità di sensi, alcuni dei quali riconducibili a entità concrete, il che introduce un ulteriore livello di complessità. L’integrazione di entità in spazi semantici originariamente concepiti per rappresentare solo sensi lessicali non è sempre immediata.

Per ovviare ai limiti dei modelli tradizionali, è stato introdotto un meccanismo denominato massimilarity, il cui obiettivo è quello di non limitarsi alla similarità massima tra i possibili accoppiamenti di sensi, ma piuttosto di individuare la coppia più plausibile, ossia quella che si sospetta stia alla base del giudizio umano.

Per la rappresentazione visiva dei vettori associati ai diversi sensi, si è fatto uso di proiezioni su due dimensioni, che permettono di visualizzare con maggiore chiarezza la distribuzione e la distanza tra i significati. Questa modalità facilita la comprensione della struttura dello spazio semantico e del comportamento dei modelli nei task di similarità.

Nella rappresentazione visiva dei sensi associati a un termine come *wave*, vengono adottate convenzioni cromatiche per distinguerne le categorie: i sensi evidenziati in rosso si riferiscono ai significati possibili del termine *wave*, mentre quelli in verde rappresentano i sensi associati al termine *weather*. All'interno di questo spazio semantico, si osserva come l'applicazione del criterio classico di massimizzazione della similarità porti talvolta alla selezione di significati inattesi o non rilevanti — come, ad esempio, album musicali o entità nominali — che però risultano artificialmente vicini nel modello vettoriale.

Questa osservazione ha reso evidente la necessità di una metodologia che non si limiti alla selezione cieca del massimo valore di similarità, ma che tenga conto anche della coerenza semantica tra i sensi selezionati e i concetti centrali in gioco. In questo senso, è stato introdotto il concetto di *ricchezza similare* (*similarity richness*), una metrica che integra il contributo della massima similarità tra i sensi con la loro prossimità a un centro semantico condiviso. Tale centro può essere pensato come una sorta di punto di equilibrio attorno al quale gravitano i significati più rappresentativi del termine. Questo approccio ha dimostrato di offrire vantaggi concreti rispetto alla semplice massimizzazione, specialmente in quei contesti in cui le ambiguità lessicali generano una molteplicità di sensi distribuiti con ampia varianza.

Una delle scoperte più rilevanti è stata che certi modelli, come Hornet e Labernet, tendono a produrre spazi semantici molto sparsi, in cui i sensi si distribuiscono in modo tale da non permettere una chiara identificazione di cluster concettuali. Tuttavia, nel caso di set di dati in cui si osserva una forte vicinanza tra determinati sensi, è possibile sfruttare questa proprietà per individuare delle *unità di senso*, ovvero insiemi coesi di significati che condividono una medesima radice semantica. Questo raggruppamento consente una semplificazione utile del problema, riducendo la complessità interpretativa e migliorando l'affidabilità del sistema.

L’analisi di una tabella delle distanze tra i sensi associati a *weather* e quelli di *wave* mostra chiaramente come sia possibile partizionare i significati in sottogruppi coerenti. Alcuni sensi di *wave* fanno riferimento a entità fisiche — come il moto ondoso, il fenomeno ondulatorio in fisica, o altri elementi di natura concreta — mentre altri si riferiscono a entità musicali, probabilmente sovrarappresentate nei dati a causa della frequenza con cui appaiono nei corpora da cui sono stati derivati. Un ulteriore sottogruppo si concentra sul senso gestuale di *wave*, ossia il saluto con la mano, distinguibile dagli altri sia per posizione nello spazio semantico sia per il significato associato.

La *Sprengs Similarity*, o rete di similarità estesa, si rivela particolarmente utile per identificare questi raggruppamenti. Essa consente di “domare” la complessità che deriva dalla molteplicità di sensi, specialmente quando questi risultano altamente dispersi. Da questa intuizione si può derivare la costruzione di strumenti computazionali più rigorosi, capaci di implementare meccanismi di raggruppamento semantico che vadano oltre la semplice vicinanza numerica dei vettori.

In contesti in cui i sensi sono numerosi e distribuiti in modo relativamente uniforme, si pone un problema noto come *sparsità semantica*: la presenza di molti sensi poco differenziati rende difficile per il sistema selezionare quello più pertinente. In questi casi, l’identificazione di unità di senso coese rappresenta un’opportunità per ridurre l’ambiguità e rafforzare la coerenza del giudizio automatico, limitando le probabilità di errore del sistema.

Le metriche discusse presentano un ulteriore aspetto di interesse: possono essere utilizzate come istruzioni operative per effettuare reclutamenti semantici *online*, senza la necessità di riscrivere o rigenerare le risorse sottostanti. In altre parole, l’embedding di riferimento, come nel caso di un *circuit embedding*, rimane fisso e invariato, mentre le metriche permettono di selezionare dinamicamente i sensi pertinenti in fase di utilizzo. Va però sottolineato che questi circuit embedding, pur essendo stabili, non sempre sono verificati, e il loro impiego per operazioni come caricamento in memoria, notifiche, scaricamento o taglio può risultare oneroso, richiedendo risorse computazionali e spazio anche dell’ordine di gigabyte.

Il vantaggio dell’impiego online di tali strutture è evidente nella misura in cui consentono una manipolazione dinamica e più efficiente delle rappresentazioni semantiche, senza dover ricorrere ogni volta alla rigenerazione dell’embedding. L’intuizione di fondo consiste nel determinare un raggio semantico entro cui individuare elementi potenzialmente collaboranti, ovvero sensi semanticamente compatibili o attivi rispetto a un centro. Questo raggio può essere definito in maniera più o meno estensiva, influenzando la finezza o la granularità della selezione: un raggio stretto limita il numero di alternative, concentrandosi su sensi più vicini ma potenzialmente più grezzi; un raggio più ampio include invece un numero maggiore di alternative, con un aumento della complessità.

L’intera operazione parte dall’individuazione di un senso pivot, o di una serie di sensi centrali. A partire da questi, è possibile operare una trasformazione dell’approccio classico alla similarità semantica: si abbandona la mera massimizzazione e si adotta un criterio basato sulla massimizzazione della rete similare, cioè si cerca non il massimo assoluto, ma l’argomento che massimizza la *rete* di similarità. Questo approccio consente di individuare nodi centrali, attorno ai quali si organizza la cosiddetta *nebulosimilarity*.

La nebulosimilarity è una metrica che consente di raccogliere, in un intorno geometrico centrato su un senso di riferimento, tutti gli elementi semantici che lo circondano e che sono sufficientemente compatibili. Questo centro può essere inteso anche in senso geometrico, come si osserva in alcune visualizzazioni in cui i sensi si distribuiscono in forma circolare o clusterizzata. Un’applicazione concreta di questa ipotesi si può osservare nel caso del termine *wave* inteso come fenomeno fisico: i sensi pertinenti si trovano raccolti all’interno di una regione compatta, visualizzabile graficamente, che corrisponde a una parte specifica dello spazio semantico.

La sperimentazione condotta si è basata su un campione di 500 termini selezionati da un dataset utilizzato per competizioni internazionali. Ogni termine è stato annotato manualmente e testato in combinazione con una batteria di risorse di embedding, tra cui quelle che rappresentano i termini e quelle che rappresentano i sensi. Come prevedibile, gli embedding dei sensi si sono rivelati più precisi, in quanto derivano da rappresentazioni che tengono conto della disambiguazione, ma soffrono maggiormente del problema della frammentazione semantica.

Le risorse testate includono embedding allo stato dell’arte, e per ciascuna è stato rilevato il tasso di copertura del dataset e la prestazione secondo tre metriche principali: correlazione di Spearman, correlazione di Pearson, e la loro media armonica. In particolare, si è osservato che le metriche *Ranked Similarity* e *Nebulous Similarity* forniscono punteggi costantemente superiori o almeno equivalenti rispetto alla semplice massimizzazione della *Ranked Similarity*. Questo risultato è stato ancora più evidente nel contesto del task di *Semantic Identification*, che si concentra non solo sulla similarità tra coppie di termini, ma sull’identificazione esatta dei sensi coinvolti.

L’interesse specifico non era tanto nel promuovere una particolare metrica, quanto nello studio comparativo delle performance su vari set di embedding. Si è constatato che la sostituzione delle metriche tradizionali con approcci più raffinati, come appunto la *Nebulous Similarity* o una *Ranked Similarity* adattata, può essere implementata in modo *live* nei sistemi, migliorandone la precisione e l’efficienza. La possibilità di modulare dinamicamente la metrica rappresenta un elemento di grande valore applicativo, specialmente in contesti di alto carico semantico e frammentazione dei sensi.

Gli embedding tradizionali presentano una limitazione strutturale significativa: tendono a unificare, fondere e appiattire in un’unica rappresentazione tutti i possibili sensi associabili a un termine. Di conseguenza, l’embedding per una parola come *bank* risulta identico sia che questa si riferisca a un’istituzione finanziaria, sia che indichi la riva di un fiume. Questo approccio non consente una distinzione contestuale dei significati e riduce la sensibilità semantica delle rappresentazioni vettoriali.

L’introduzione degli embedding contestuali rappresenta un cambiamento radicale rispetto a questa impostazione. A differenza dei modelli statici, questi embedding consentono di costruire rappresentazioni del significato che variano a seconda del contesto in cui il termine compare. L’embedding di *bank* sarà quindi differente a seconda che si trovi nella frase *bank of America* o *bank of the river*, grazie a un processo che adatta dinamicamente la rappresentazione del termine al suo contesto linguistico immediato.

Tuttavia, è importante chiarire che questa variabilità non implica che le rappresentazioni ottenute siano direttamente orientate alla disambiguazione del senso lessicale. L’operazione riguarda piuttosto una trasformazione contestuale: l’embedding viene riconfigurato sulla base degli altri elementi presenti nella frase, senza assumere necessariamente una categorizzazione semantica esplicita.

Si consideri ad esempio la frase *bank of the river*. In questo caso, si immagina un embedding contestuale capace di attribuire peso e importanza differenziati ai vari termini della frase, selezionando automaticamente quali parole influenzano maggiormente la rappresentazione finale di *bank*. A livello computazionale più basso, le architetture che permettono tale adattamento operano partendo da rappresentazioni che possono essere assimilabili a vettori standard — per esempio, gli embedding tradizionali di parole isolate — ma vengono successivamente ricalibrati in funzione del contesto sintattico e semantico.

Il termine *river* fornisce, in questo esempio, un’informazione disambiguante: guida l’orientamento dell’embedding di *bank* verso il senso naturale. Tale riorientamento non riguarda un singolo termine, ma si applica a tutti gli elementi della frase, ciascuno dei quali viene ridefinito in relazione agli altri presenti nella sequenza.

L’architettura alla base di questi modelli è, in senso generale, quella dei *transformer*. Quando si fa riferimento a un transformer, ci si riferisce a un’organizzazione modulare a livelli sovrapposti, in cui ciascun livello implementa una trasformazione basata su una media pesata degli input ricevuti. L’output di un livello costituisce l’input per il successivo. Il numero di strati può variare da un minimo di due a oltre venti, a seconda della complessità del modello.

La struttura di base proposta nella pubblicazione originaria — un articolo complesso ma rivoluzionario — prevede due sezioni principali: l’encoder e il decoder. Entrambi sono composti da un certo numero di strati identici (nella versione canonica, sei per ciascuno). Un aspetto chiave di questa architettura è l’assenza di meccanismi ricorrenti: la ricorrenza è stata sostituita dal meccanismo dell’attenzione, che presenta vantaggi sia in termini di accuratezza, sia per quanto riguarda l’efficienza computazionale.

Il meccanismo dell’attenzione consente di stabilire relazioni tra i termini di una sequenza in modo indipendente dalla loro distanza. A differenza delle reti ricorrenti, che richiedono un numero crescente di operazioni man mano che la distanza tra i termini aumenta, l’attenzione permette di mettere in relazione direttamente ogni parola con tutte le altre, pesandone l’influenza sulla base del contesto. Ciò si traduce in una maggiore efficacia nella cattura delle dipendenze a lungo raggio e nella risoluzione di fenomeni linguistici complessi, come la co-referenza.

All’interno di ciascun livello dell’encoder e del decoder, il calcolo dell’attenzione è una delle due principali funzioni implementate. Questo meccanismo rappresenta un’innovazione fondamentale nel modo in cui si costruiscono rappresentazioni linguistiche sensibili al contesto e costituisce il cuore delle architetture moderne che hanno trasformato l’elaborazione del linguaggio naturale.

Immagine che contiene testo, diagramma, schermata, Piano

Il contenuto generato dall'IA potrebbe non essere corretto.

All'interno dell'architettura transformer, uno degli elementi fondamentali è il meccanismo di attenzione multi-head. Questo meccanismo implementa il principio di tesatura, ossia la capacità di individuare e rappresentare le relazioni interne tra i termini di un’espressione. Ogni *head* di attenzione lavora in parallelo alle altre, concentrandosi su differenti aspetti o sottoinsiemi della frase, permettendo così una ricostruzione più ricca e articolata delle dipendenze tra le parole.

Accanto al meccanismo di attenzione vi è una rete neurale feed-forward che prende in input le rappresentazioni calcolate e ne produce di nuove, mantenendo però inalterata la dimensione dei vettori. La rappresentazione interna di ogni elemento viene elaborata e convertita in un vettore a lunghezza fissa, che resta costante lungo tutto il percorso del modello. Questa fissità dimensionale — che può essere, ad esempio, di 512 unità — è determinante dal punto di vista computazionale: permette infatti una gestione ottimizzata dei dati e una maggiore efficienza nelle operazioni, poiché i vettori hanno tutti la stessa forma.

L’architettura del decoder, posizionata nella parte destra dello schema canonico del transformer, è strutturalmente simile all’encoder. Tuttavia, al suo interno è presente un ulteriore strato di attenzione che, a differenza di quelli precedenti, riceve come input non solo i dati provenienti dai livelli inferiori, ma anche l’output finale dell’encoder. Questo passaggio è ciò che consente al decoder di integrare l’informazione codificata nel segmento sorgente (per esempio, nella lingua di partenza) nel processo di generazione dell’output (nella lingua di arrivo).

Procedendo per macro-operazioni, si può osservare che la prima fase di elaborazione consiste nell’inserimento dell’informazione posizionale. Questa componente è necessaria perché l’architettura transformer, a differenza delle reti ricorrenti, non ha alcuna nozione intrinseca di ordine sequenziale. L’informazione posizionale comunica quindi al modello la distanza e la disposizione relativa degli elementi all’interno della sequenza. Essa gioca un ruolo centrale nel determinare l’intensità e la qualità delle connessioni tra i termini, connessioni che possono essere interpretate come relazioni semantiche o sintattiche.

Successivamente, ciascun termine viene rielaborato tramite il meccanismo di self-attention. Questo consente di costruire una rappresentazione contestualizzata per ogni parola, in cui il significato dipende dagli altri termini presenti nella stessa sequenza. Dopo l’attenzione, si applicano operazioni ausiliarie come *add and normalize*. Queste funzioni, pur non essendo centrali nella determinazione del significato, risultano essenziali per la stabilità e l’efficacia del modello. In particolare, la normalizzazione evita che valori numerici eccessivi o incoerenti compromettano l’addestramento e la propagazione dell’informazione.

Tutti i blocchi all’interno dell’encoder e del decoder sono organizzati in stack, cioè in sequenze di livelli identici che si ripetono. Ogni livello riceve in input una lista di vettori — derivanti dagli embedding statici iniziali o dalle elaborazioni dei livelli precedenti — e restituisce a sua volta un insieme di nuovi vettori della stessa dimensione. L’output finale dell’encoder rappresenta la sintesi informativa dell’intera sequenza sorgente ed è utilizzato come guida nei successivi passaggi di generazione dell’output da parte del decoder.

Nel caso di un task come la traduzione automatica, il decoder elabora la sequenza di output tenendo conto simultaneamente dei token già generati e delle informazioni provenienti dall’encoder. Le ultime operazioni, spesso realizzate tramite funzioni di tipo *softmax*, producono una distribuzione di probabilità che consente di selezionare il termine più probabile da generare in quella posizione.

Si consideri, infine, una frase semplice come *The animal didn't cross the street because it was too tired*. In una sequenza come questa, la comprensione di chi o cosa rappresenti *it* richiede un’analisi attenta del contesto precedente. Il meccanismo di attenzione è in grado di identificare che *it* fa riferimento a *the animal* e non, ad esempio, a *the street*. Questo è possibile perché l’attenzione mette in relazione ogni parola con tutte le altre, indipendentemente dalla distanza, permettendo una comprensione contestuale sofisticata ed efficace. Nel processo di risoluzione dei riferimenti anaforici, come nel caso della frase *“The animal didn’t cross the street because it was too tired”*, la difficoltà non sta tanto nella comprensione umana — dove la stanchezza è un attributo normalmente associato a un essere vivente — quanto nella capacità di un sistema computazionale di stabilire correttamente a quale termine faccia riferimento il pronome *it*. In questo tipo di situazione, l’animale è menzionato più lontano nella frase rispetto ad altri candidati, come ad esempio *the street*. Il sistema non opera una comprensione semantica nel senso umano del termine, ma si affida a un meccanismo strutturale: il meccanismo di attenzione.

L’attenzione è, di fatto, il dispositivo incaricato di valutare i pesi delle relazioni tra i termini e di rafforzare la connessione tra quelli che risultano, in base all’apprendimento, più coerenti nel contesto. Non si tratta di un’analisi linguistica profonda basata su parsing sintattico o regole semantiche esplicite. Al contrario, l’intero processo si fonda sulla manipolazione di pesi appresi, che sono contenuti in matrici. Questi pesi determinano l’intensità della connessione tra le parole in base al contesto in cui compaiono.

A partire dagli embedding iniziali, indicati con X, si definiscono tre matrici fondamentali: Q (Query), K (Key) e V (Value). Queste matrici non sono fissate a priori, ma vengono apprese durante l’addestramento del modello. Ciascun vettore di input viene trasformato attraverso queste matrici, ottenendo così tre rappresentazioni distinte per ogni termine della sequenza: la query che rappresenta ciò che sta cercando, la key che rappresenta ciò che offre, e il value che rappresenta il contenuto effettivo.

Il primo passaggio computazionale consiste nella moltiplicazione della matrice Q per la trasposta di K, che misura la compatibilità tra i termini. Il risultato di questa operazione è una matrice di punteggi grezzi di attenzione, che viene poi normalizzata rispetto alla radice quadrata della dimensione del vettore (ad esempio √64 se la dimensione dei vettori è 64) per evitare valori numericamente troppo elevati. Successivamente, viene applicata una funzione *softmax* per convertire questi punteggi in una distribuzione di probabilità, che esprime quanto ciascun termine della sequenza debba “prestare attenzione” agli altri.

Il passo successivo consiste nel moltiplicare questa matrice di attenzione per V, ottenendo una rappresentazione pesata dei valori, che tiene conto delle relazioni contestuali tra i termini. Il risultato è una nuova matrice, indicata spesso come Z, che rappresenta l’output del meccanismo di attenzione per un singolo *head*.

Nel caso del meccanismo multi-head, l’intero processo descritto viene eseguito in parallelo su differenti sotto-spazi della rappresentazione vettoriale. Ad esempio, se la dimensione totale è 512, essa può essere suddivisa in otto teste da 64 dimensioni ciascuna. Ogni testa produce un proprio output Z. Questi output vengono poi concatenati e trasformati mediante una matrice appresa, per ottenere la rappresentazione finale che sarà trasmessa agli strati successivi della rete.

Tutto il processo ha inizio con l’aggiunta dell’informazione posizionale, un elemento essenziale in quanto l’architettura transformer, per sua natura, non dispone di un meccanismo implicito per rappresentare l’ordine dei termini. L’informazione posizionale, che viene codificata tramite segnali sinusoidali o mediante embedding appresi, viene sommata agli embedding dei token in ingresso. Questo produce una rappresentazione che incorpora non solo il significato del termine, ma anche la sua posizione relativa all’interno della sequenza.

Sebbene si parli talvolta di *time signal*, è importante chiarire che non si tratta di un segnale temporale in senso stretto. Esso rappresenta l’indice del token nella sequenza ed è un modo per comunicare al modello l’ordine dei termini. La combinazione tra embedding e informazione posizionale genera un vettore di input completo per ciascun token.

Nel decoder, la situazione si complica leggermente. L’output dell’encoder viene convertito in matrici K e V che saranno utilizzate in un particolare livello del decoder chiamato *encoder-decoder attention*. Questo livello è una variante del meccanismo di attenzione già descritto, ma in cui le query provengono dagli stati interni del decoder, mentre keys e values provengono dall’encoder. Questo consente al decoder di focalizzarsi sugli elementi rilevanti della sequenza di input durante la generazione della sequenza di output.

Il decoder include due meccanismi di attenzione distinti: il primo è un self-attention simile a quello dell’encoder, ma mascherato per evitare che il modello acceda a token futuri nella generazione sequenziale; il secondo è l’encoder-decoder attention, che introduce informazioni contestuali esterne provenienti dalla sequenza sorgente. I risultati di questi meccanismi sono poi passati attraverso ulteriori strati feed-forward e di normalizzazione, fino alla generazione del risultato finale.

Nel funzionamento di un modello di tipo Transformer, in particolare nella fase di generazione dell’output, il decoder prende in input non solo l’output dell’encoder finale (quello posto in cima allo stack), ma anche l’output dei livelli inferiori, all'interno dello stesso encoder. Questo flusso informativo multilivello consente al decoder di elaborare in modo più raffinato il contesto e di focalizzarsi sugli elementi salienti della sequenza di input.

Una volta completato il percorso del decoder, l’output viene proiettato nello spazio del vocabolario tramite un livello *linear*. Questo livello è una rete completamente connessa (fully connected) che ha il compito di trasformare i vettori di rappresentazione interni in vettori della stessa dimensione del vocabolario. Poiché il vocabolario può comprendere decine di migliaia di termini, il livello linear produce un vettore di dimensioni molto elevate, uno per ogni token del vocabolario.

Tuttavia, i valori in uscita da questo livello non sono ancora interpretabili come probabilità. Per ottenere una distribuzione di probabilità vera e propria, viene applicata la funzione *softmax*. Questa funzione garantisce due proprietà fondamentali: tutti i valori risultanti sono positivi, e la loro somma è pari a uno. In questo modo, ogni valore può essere interpretato come la probabilità che il token corrispondente sia quello corretto da emettere in output.

Il livello softmax trasforma quindi i *logits* (i valori grezzi forniti dal livello linear) in una distribuzione probabilistica sul vocabolario. Questo consente di confrontare l’output del modello con l’etichetta corretta tramite una funzione di *loss*, la quale calcola la distanza tra la previsione del modello e l’output atteso.

Nel processo di apprendimento, il modello parte da una condizione in cui i pesi delle reti neurali sono inizializzati casualmente o con strategie non informate. A partire da questa condizione iniziale, il modello genera output che vengono confrontati con le etichette corrette, rappresentate tipicamente in forma *one-hot*. Una codifica one-hot assegna un valore 1 all’indice corrispondente al termine corretto e 0 a tutti gli altri. Ad esempio, in un vocabolario semplificato di cinque termini, se l’output corretto è il termine all’indice 1, la rappresentazione sarà un vettore con 1 in posizione 1 e 0 altrove.

Attraverso la funzione di perdita (loss function), viene calcolato l’errore tra l’output predetto e quello corretto. Questo errore guida il processo di aggiornamento dei pesi del modello, in modo che la probabilità assegnata al termine corretto aumenti progressivamente nel tempo. In uno scenario ideale, il modello arriva a generare distribuzioni in cui la probabilità attribuita al termine corretto si avvicina molto a 1, segno che il modello ha appreso efficacemente la relazione tra input e output.

Tutto ciò presuppone un processo preliminare fondamentale: la *tokenizzazione*. Uno dei problemi principali in questa fase è la gestione di parole che non sono mai state viste durante l’addestramento, o che compaiono raramente. Per affrontare questo problema, vengono utilizzate tecniche di tokenizzazione a sottoparole. Una delle più diffuse è il *Byte Pair Encoding* (BPE), un algoritmo che consente di costruire un vocabolario flessibile e compatto partendo da unità minime come i caratteri.

Inizialmente, ogni carattere è considerato un simbolo indipendente, compresi i caratteri speciali come i delimitatori di parola. Poi, in maniera iterativa, le coppie di simboli più frequenti vengono unite per formare nuovi simboli composti. Ad esempio, in una determinata lingua o corpus, la sequenza “th” può essere estremamente frequente, così come “the” o “tion”. Queste sequenze vengono promosse a simboli nel vocabolario, riducendo la lunghezza delle sequenze da elaborare e migliorando la copertura anche per termini non presenti esattamente nel training set.

Questo approccio presenta due vantaggi principali: il primo è l’economia di spazio, perché non è necessario rappresentare esplicitamente ogni parola del linguaggio, ma solo un sottoinsieme ben scelto di sottosequenze; il secondo è la maggiore capacità di generalizzazione, perché anche parole sconosciute possono essere suddivise in componenti note e quindi elaborate dal modello.

Inoltre, l’uso di tokenizzazione a livello di sottoparola permette di fissare un limite superiore alla dimensione del vocabolario, tipicamente intorno ai 50.000 elementi. Senza questo vincolo, la crescita del vocabolario sarebbe teoricamente illimitata, rendendo proibitivo il calcolo delle proiezioni nello spazio del vocabolario a causa della moltiplicazione tra le dimensioni degli embedding e il numero totale di token.

In termini computazionali, questa riduzione è significativa. Se si utilizzasse un vocabolario di un miliardo di termini, la matrice che proietta i vettori nello spazio delle parole richiederebbe risorse incompatibili con l’efficienza operativa. Al contrario, limitando il vocabolario a 50.000 unità composte da sottoparole, è possibile mantenere sotto controllo la complessità computazionale e allo stesso tempo garantire una copertura linguistica robusta. Operativamente, ciò significa che ogni token in input o in output è rappresentato da un embedding, che incorpora sia il significato lessicale che la posizione all’interno della frase, e che può essere appreso e raffinato nel corso dell’addestramento del modello.

Ogni set di embedding possiede un proprio tokenizer, strettamente legato alla struttura interna del modello stesso. Ad esempio, il modello BERT utilizza un tokenizer specificamente progettato per BERT.

Questa coerenza è necessaria poiché il modo in cui il testo viene suddiviso in token deve corrispondere esattamente a quello previsto durante l'addestramento del modello. A lezione, è stato preparato un esempio pratico che mostra come caricare modelli differenti e come ciascuno di essi adotti una strategia di tokenizzazione distinta. Tali esempi consentono di testare direttamente il funzionamento della tokenizzazione.

Un altro aspetto fondamentale è rappresentato dai **positional embeddings**, i quali servono a introdurre l’informazione di posizione dei token nella sequenza. Senza questo tipo di encoding, un modello basato su Transformer non avrebbe alcun modo per distinguere l’ordine dei token in una frase. Consideriamo, per esempio, la parola "bank": a seconda della posizione in cui compare nella frase, può assumere significati differenti. Se è prossima al termine "finance", è probabile che si riferisca a un istituto finanziario; se invece è vicina a "river", il significato cambia completamente. Il posizionamento relativo è, quindi, informazione cruciale.

Prima dell’avvento dei Transformer, l’informazione posizionale veniva rappresentata in modo esplicito tramite strutture separate. I Transformer, invece, sommano agli embedding delle parole i vettori addizionali che rappresentano le posizioni. In questo contesto, si definiscono due vettori: uno rappresenta il significato del token (embedding lessicale), l’altro rappresenta la posizione (positional encoding). Questi due vettori vengono sommati per ottenere la rappresentazione finale del token.

Il positional encoding viene calcolato tramite funzioni trigonometriche: si usa il seno per le componenti di indice pari e il coseno per quelle dispari. Questa tecnica permette di codificare l’informazione posizionale in maniera continua e differenziabile, mantenendo la compatibilità con l’architettura neurale e garantendo che i modelli possano apprendere correlazioni relative tra le posizioni.

Passando ora al meccanismo di attenzione, esso si basa su tre componenti: **query**, **key** e **value**. Per comprendere intuitivamente questa struttura, si può fare ricorso a una semplice analogia: immaginare di trovarsi in un negozio alla ricerca di un prodotto. La **query** rappresenta ciò che si sta cercando, le **key** sono le etichette dei prodotti sugli scaffali e i **value** rappresentano le informazioni utili associate a ciascun prodotto. Il modello confronta la query con tutte le key, assegnando un livello di attenzione a ciascun elemento. I prodotti che risultano più simili alla query ricevono maggiore attenzione, e l’informazione viene recuperata attraverso i relativi value.

Formalmente, il calcolo dell’attenzione avviene attraverso il prodotto scalare tra la query e ogni key, normalizzato per la radice quadrata della dimensione del vettore. Tale normalizzazione serve a stabilizzare i valori numerici. Successivamente, si applica una funzione softmax per convertire i punteggi in probabilità: in questo modo, l’attenzione si distribuisce sui diversi token. I pesi ottenuti vengono infine utilizzati per calcolare una media pesata dei value, ottenendo così il vettore di uscita dell’attenzione.

Questo processo viene ripetuto per ogni token nella sequenza, confrontando ogni query con tutte le key, inclusa la propria. Il risultato è una matrice che rappresenta il livello di attenzione che ogni token ha rispetto a tutti gli altri. Questo è il cuore del meccanismo di self-attention che caratterizza l’architettura Transformer: la capacità di ogni token di modulare dinamicamente la propria rappresentazione in funzione del contesto in cui si trova.

Nel caso specifico, se si considera un esempio con 4 token e si calcolano i prodotti scalari tra una query e le key corrispondenti, si otterranno dei valori (ad esempio 40, 16, 8, 24) che, una volta normalizzati e passati attraverso la softmax, produrranno delle probabilità. Queste probabilità saranno poi utilizzate per pesare i corrispondenti value e ottenere il vettore finale per quel token. Questo processo viene svolto in parallelo per ogni token nella frase, permettendo un’elaborazione altamente efficiente e contestualmente ricca.

**Il problema della fusione del significato (meaning conflation problem)**

Gli algoritmi basati sulla similarità distribuzionale tendono a fondere tutti i significati di una parola in un unico embedding. Ad esempio, la parola *bank* riceve una sola rappresentazione numerica, anche se può assumere significati molto diversi: in ambito finanziario (come in “la banca concede prestiti”) oppure in ambito naturale (come in “la riva del fiume”).

Le reti neurali basate su architetture Transformer sono state progettate per generare *contextualized embeddings*, ovvero rappresentazioni numeriche che variano a seconda del contesto in cui il termine compare. In questo modo, *bank* riceverà rappresentazioni differenti a seconda che si parli di una banca o di una riva fluviale.

**Embedding contestualizzati**

Ogni parola riceve un embedding contestualizzato che consiste in una media pesata degli embedding input indipendenti dal contesto, analogamente a quanto avviene con word2vec.

Per esempio, se in una frase compare la parola *river*, l’embedding risultante di *bank* tenderà a riflettere un’interpretazione legata all’ambiente naturale. Lo stesso vale per tutti i termini presenti nella frase: ciascuno riceve una rappresentazione numerica distinta, derivata dal contesto in cui appare.

Questo comportamento è illustrato dalla Figura 12.1, in cui ogni embedding di output rappresenta una media pesata degli embedding input. I pesi utilizzati sono specifici per ciascuna parola, permettendo così alla rete di produrre rappresentazioni differenti per ciascun termine, a seconda del contesto.

Immagine che contiene diagramma, linea, origami, modello

Il contenuto generato dall'IA potrebbe non essere corretto.

**Stack di livelli nei Transformer**

In pratica, le reti Transformer sono composte da più livelli. Ciascun livello implementa una media pesata degli embedding di input, come descritto nella pagina precedente.

L’output prodotto da un livello *i* viene utilizzato come input per il livello *i+1*. Il numero totale di livelli varia tipicamente tra 2 e 24. L’impiego di più livelli consente alla rete di apprendere funzioni più complesse nella costruzione degli embedding di output, migliorando la qualità delle rappresentazioni.

**Architettura generale dei Transformer**

Immagine che contiene testo, diagramma, schermata, Piano

Il contenuto generato dall'IA potrebbe non essere corretto.

L’input viene elaborato nel lato sinistro del modello, l’**encoder**, attraverso due sotto-componenti: un meccanismo di attenzione e una rete feed-forward. A destra, gli **output attesi** vengono gestiti dal **decoder**, che include due livelli di attenzione e una rete feed-forward.

A differenza di altre architetture (come RNN o LSTM), il Transformer non fa uso di ricorrenze: la ricorrenza è stata completamente sostituita dal meccanismo di attenzione. Questo porta benefici in termini di efficienza computazionale, poiché l’attenzione non dipende dalla distanza tra le parole nella sequenza. In ciascun livello di attenzione, il modello Transformer originale esegue otto meccanismi di attenzione in parallelo (multi-head attention), accelerando notevolmente i calcoli.

**Architettura a sei livelli**

Il modello Transformer originario è costituito da una pila di sei livelli. L’output di un livello *l* diventa l’input del livello *l+1*, e così via, fino alla generazione finale dell’output.

Sia l’encoder (a sinistra) che il decoder (a destra) sono formati da sei livelli identici ciascuno. Ogni livello comprende due sotto-componenti principali, come descritto sopra.

**Pagina 12**

Il **decoder** del Transformer è composto, come l’encoder, da uno stack di sei strati identici. Tuttavia, presenta una differenza strutturale: oltre ai due sottolivelli comuni all’encoder (self-attention multi-testa e feedforward), il decoder introduce un terzo sottolivello che realizza un meccanismo di attenzione multi-testa sugli output prodotti dallo stack dell’encoder. Anche qui si applicano connessioni residue attorno a ciascun sottolivello, seguite da una normalizzazione per strato. Una modifica importante riguarda la self-attention del decoder, che è progettata per **evitare che una posizione si riferisca a posizioni successive**: ciò viene realizzato tramite un meccanismo di mascheramento. Questo, assieme al fatto che gli embedding di output sono spostati di una posizione, fa sì che la predizione relativa alla posizione *i* dipenda solo dagli output noti delle posizioni precedenti.

**Pagina 13**

Passando all’**attenzione**, essa può essere definita come una funzione che, dati una **query** e una serie di coppie **chiave-valore**, restituisce un output, dove tutti questi elementi sono vettori. L’output viene calcolato come una somma pesata dei valori, con i pesi determinati da una funzione di compatibilità tra la query e ciascuna chiave.

**Pagina 14**

Nella variante usata nel Transformer, questa attenzione è chiamata **attenzione “scaled dot-product”**. L’input consiste in query e chiavi di dimensione dkd\_kdk​ e valori di dimensione dvd\_vdv​. Si calcolano i prodotti scalari tra le query e le chiavi, che rappresentano la similarità tra questi vettori. L’output è una combinazione pesata dei valori, dove i pesi sono ricavati applicando una funzione softmax ai prodotti scalari, normalizzati rispetto alla radice quadrata di dkd\_kdk​. Questo metodo è efficace sia in termini computazionali che in termini di prestazioni.

**Pagina 15**

L’**architettura** di ogni livello del Transformer implementa una sequenza di cinque operazioni. La prima consiste nell’aggiunta dell’**informazione posizionale** all’embedding di ciascuna parola: l’embedding cambia a seconda della posizione della parola nel testo. La seconda operazione è quella della **self-attention**, nella quale ciascun embedding viene aggiornato come media pesata di tutti gli altri embedding presenti nella sequenza: le parti del testo considerate più rilevanti ricevono un peso maggiore. Questo consente di modellare in modo flessibile le dipendenze tra parole, a prescindere dalla distanza.

**Pagina 16**

Le operazioni successive sono meno importanti concettualmente, ma hanno un impatto significativo dal punto di vista empirico. In particolare, la componente **“add and normalize”** somma gli embedding in input e in output da ciascun modulo precedente (per esempio, gli input alla self-attention e gli output derivanti dalla media pesata) e **normalizza il risultato**, evitando che valori troppo grandi compromettano l’apprendimento durante la discesa del gradiente. Questa fase stabilizza i valori che fluiscono attraverso i vari livelli del modello e migliora la convergenza.

**Pagina 17 – Tipologie di modelli Transformer (1)**

Esistono diversi tipi di architetture Transformer, ciascuna progettata per scopi differenti. I modelli **solo encoder** trasformano una sequenza di testo in una rappresentazione numerica, dove ogni token è rappresentato tenendo conto sia del contesto a sinistra sia di quello a destra. Questo meccanismo prende il nome di **attenzione bidirezionale**. Questi modelli risultano particolarmente adatti per compiti come la classificazione del testo o il riconoscimento di entità nominate. Un esempio noto di questa categoria è **BERT** e le sue varianti.

**Pagina 18 – Tipologie di modelli Transformer (2)**

I modelli **solo decoder**, invece, hanno come obiettivo la generazione del testo, completando una sequenza fornita come input tramite la previsione iterativa del termine successivo più probabile. Questo approccio è chiamato anche **attenzione causale** o **autoregressiva**. I modelli della famiglia **GPT** rientrano in questa tipologia. In tali architetture, la rappresentazione di ciascun token si basa esclusivamente sul contesto a sinistra.

Infine, i modelli **encoder-decoder** sono progettati per compiti più complessi che richiedono la trasformazione di una sequenza testuale in un'altra. Sono ideali per applicazioni come la **traduzione automatica** o il **riassunto testuale**. I modelli **BART** e **T5** appartengono a questa categoria.

**Pagina 19 – Transformer visuale**

Questa sezione rimanda a una risorsa grafica esplicativa ben nota nel settore: il sito di Jay Alammar, che fornisce un'introduzione illustrata al funzionamento dell'architettura Transformer. Tale fonte è spesso utilizzata per spiegare visivamente i meccanismi interni del modello.

<https://jalammar.github.io/illustrated-transformer/>

**Pagina 20 – Encoder**

L'encoder di un Transformer è costituito da una pila di **sei strati identici**. Ogni livello è formato da due sottolivelli principali: un meccanismo di attenzione multi-testa (multi-head attention) e una rete neurale completamente connessa (feed-forward) applicata separatamente a ciascuna posizione.

Intorno a ciascun sottolivello viene inserita una **connessione residua**, seguita da una **normalizzazione del livello** (layer normalization). Tutti i sottolivelli e gli strati di embedding producono uscite della stessa dimensione, tipicamente fissata a **512 elementi**.

**Pagina 21 – Decoder**

Anche il decoder è composto da **sei strati identici**. Tuttavia, rispetto all’encoder, include un **terzo sottolivello**: un meccanismo di attenzione multi-testa che si applica sull’output della pila dell’encoder. Le connessioni residue e la normalizzazione del livello sono mantenute anche qui.

Il meccanismo di attenzione self-attention nel decoder è modificato in modo da impedire a una posizione di “guardare” le posizioni successive. Questo mascheramento, unito al fatto che le uscite sono traslate di una posizione, garantisce che la previsione per la posizione *i*

Nel contesto dei Transformer, la rappresentazione delle parole in input avviene attraverso un meccanismo di embedding: ogni parola viene trasformata in un vettore. Questi vettori hanno dimensione fissa (tipicamente 512) e vengono organizzati in una lista la cui lunghezza ideale riflette la massima lunghezza delle frasi osservate nel dataset di addestramento.

Una volta ottenuta questa lista vettoriale, l'encoder inizia a elaborarla. Ogni vettore viene prima passato attraverso un modulo di *self-attention*, dopodiché attraversa una rete neurale feed-forward. L'output di ciascun passo viene trasmesso allo strato successivo dell'encoder, in un'architettura a più livelli sovrapposti.

Il sottolivello feed-forward svolge un ruolo cruciale, introducendo non linearità nel modello e trasformazioni delle caratteristiche apprese durante il passaggio attraverso il modulo di attenzione. Questo permette alla rete di apprendere rappresentazioni più astratte e significative, che possono includere ruoli sintattici, classi semantiche, o categorie latenti come i tipi di verbi o le entità nominate.

Tecnicamente, questo sottolivello è composto da due strati lineari separati da una funzione di attivazione non lineare (come ReLU o GELU). Il primo strato aumenta la dimensionalità del vettore (ad esempio da 768 a 3072), e il secondo la riporta alla dimensione iniziale.

Durante l'addestramento, questi sottolivelli non ricevono tipicamente una supervisione diretta a livello di token. Piuttosto, apprendono per retropropagazione sulla base dell'obiettivo globale del modello. Un esempio classico di tale obiettivo è il *Masked Language Modeling* (come in BERT), in cui una parola viene oscurata (es. “[MASK]”) e il modello deve predirla correttamente, come in “The [MASK] sat on the mat.” con target “cat”.

Anche se la perdita (loss) viene calcolata solo sul token mascherato, il gradiente viene propagato attraverso tutta la rete, inclusi i sottolivelli feed-forward, per ogni token nella sequenza.

Il compito principale del livello feed-forward è dunque quello di apprendere trasformazioni intermedie delle rappresentazioni dei token, utili per il raggiungimento dell'obiettivo finale. Queste trasformazioni possono includere la codifica di ruoli sintattici (soggetto, oggetto, modificatore), l’evidenziazione di ruoli semantici (agente, paziente, luogo), l’enfasi su tratti salienti (animazione, polarità, tempo verbale) e la creazione di cluster latenti di funzioni simili.

Infine, per comprendere il concetto di self-attention si può considerare un esempio: nella frase “The animal didn't cross the street because it was too tired”, il pronome “it” si riferisce ad “animal” e non a “street”. Il meccanismo di self-attention è ciò che consente al modello di stabilire questa relazione interna tra i termini, identificando e pesando adeguatamente i legami semantici presenti nel contesto. In questo senso, si può pensare alla self-attention come al meccanismo che stabilisce “chi parla con chi” all’interno di una sequenza linguistica.

**Pagina 27 – Self-Attention**  
Consideriamo la frase: *"The animal didn't cross the street because it was too tired."* Il pronome “it” può riferirsi a “animal” o a “street”. Per un essere umano, nella maggior parte dei casi è semplice dedurre che “it” fa riferimento all’animale. Tuttavia, per un sistema automatico, questa risoluzione non è affatto banale. È qui che interviene il meccanismo di *self-attention*, progettato proprio per gestire associazioni semantiche interne a una sequenza. Questo meccanismo consente al modello di valutare “chi parla con chi” nel testo, ovvero quali termini sono rilevanti rispetto ad altri e quanto.

**Pagina 28 – Positional Encoding**  
Il *positional encoding* è il meccanismo attraverso cui il Transformer tiene traccia dell’ordine delle parole in una sequenza, dal momento che, a differenza delle reti ricorrenti, non è dotato di una nozione intrinseca di sequenzialità. Viene quindi aggiunto un vettore a ciascun embedding in input. Questo vettore segue uno schema fisso, che la rete apprende ad associare alle posizioni. Così, anche in assenza di una struttura temporale esplicita, il modello riesce a dedurre la posizione relativa delle parole, permettendogli di cogliere la distanza e la relazione tra i termini nella sequenza.

**Pagina 29 – (vuota, solo intestazione)**

**Pagina 30 – Residual Connections e Normalizzazione**  
All’interno di ciascun encoder, ogni sottolivello (sia di tipo *self-attention* che *feedforward*) è avvolto da una connessione residua, seguita da un’operazione di normalizzazione. Le connessioni residue servono a evitare il problema della degradazione del gradiente nelle reti profonde, facilitando la propagazione dell’informazione. La normalizzazione contribuisce a stabilizzare i valori numerici durante l’addestramento, migliorando la convergenza e il comportamento della rete.

**Pagina 31 – Decoder e Encoder-Decoder Attention**  
L’output dell’ultimo encoder viene convertito in due insiemi di vettori: *Keys* e *Values*. Questi vettori sono poi utilizzati dai moduli di *attention* presenti all’interno del decoder, in particolare nel livello chiamato *encoder-decoder attention*. Questo modulo funziona analogamente alla *multi-head attention*, ma con una differenza: le *query* vengono calcolate usando l’output del decoder (quindi dalla parte “sotto”), mentre le *key* e *value* provengono direttamente dall’output dello stack di encoder. In questo modo, il decoder può concentrarsi sui punti rilevanti della sequenza di input quando genera l’output.

**Pagina 32**

Nel decoder, lo stack finale di trasformazioni produce un vettore di numeri reali (float). Questo vettore viene poi trasformato in una parola attraverso un livello finale lineare (Linear Layer), seguito da un livello Softmax. Il livello lineare è una rete neurale completamente connessa che proietta il vettore prodotto dallo stack di decoder in un vettore molto più grande chiamato *logits vector*.

Ad esempio, se il modello ha appreso 10.000 parole inglesi uniche durante l’addestramento (questo è il vocabolario di output), il vettore dei logits sarà largo 10.000 celle, ciascuna associata a una parola specifica.

Il livello softmax trasforma i punteggi (logits) in probabilità vere e proprie: tutti i valori sono positivi e sommano a 1. In ogni passaggio temporale, la parola associata alla cella con la probabilità più alta viene scelta come output.

**Pagina 33**

La fase di *training* (addestramento) segue lo stesso schema visto nel *forward pass* attraverso un Transformer già addestrato. Tuttavia, in fase di addestramento, il modello non è ancora stato istruito: i parametri iniziali sono casuali.

Dato che stiamo usando un dataset etichettato, possiamo confrontare l’output generato dal modello con quello corretto. Questo confronto consente di misurare l’errore e aggiornare i parametri del modello per migliorare le prestazioni.

**Pagina 34**

Supponiamo che il vocabolario di output sia composto da solo sei parole: "a", "am", "i", "thanks", "student" e "<eos>" (che sta per "end of sentence"). Ciascuna di queste parole può essere rappresentata da un vettore con lo stesso numero di dimensioni, dove un singolo 1 indica la posizione della parola corretta (e tutti gli altri valori sono 0). Questa rappresentazione si chiama *one-hot encoding*.

**Pagina 35**

Il nostro obiettivo è che l’output del modello sia una distribuzione di probabilità che assegni il valore più alto alla parola corretta, ad esempio "thanks". Tuttavia, dal momento che il modello non è ancora stato addestrato, è improbabile che riesca a farlo già al primo tentativo.

**Pagina 36**

Ma come confrontare due distribuzioni di probabilità? La tecnica più semplice è sottrarre una distribuzione dall’altra.

Questa operazione permette di quantificare quanto l’output del modello si discosti dall’output corretto, e questa differenza viene utilizzata per calcolare la *loss* (funzione di perdita), che a sua volta guida l’aggiornamento dei pesi del modello durante l’addestramento.

**Pagina 37 — Funzione di Loss**

Durante l’apprendimento, l’obiettivo è che l’output del modello rappresenti una distribuzione di probabilità che evidenzi chiaramente la parola corretta, ad esempio “thanks”. Tuttavia, dato che all’inizio della fase di addestramento il modello è completamente non addestrato, è improbabile che tale previsione venga correttamente generata al primo tentativo.

**Pagina 38 — Come confrontare due distribuzioni**

Il confronto tra due distribuzioni di probabilità può essere effettuato in modo semplice: sottraendo una distribuzione dall’altra. Questo tipo di confronto fornisce le basi per il calcolo della loss, ossia della funzione obiettivo che quantifica quanto l’output del modello si discosti dalla risposta desiderata.

**Pagina 39 — Funzione di Loss (continuazione)**

Viene rafforzato il concetto che la funzione di loss rappresenta un meccanismo per misurare la distanza tra la distribuzione di probabilità prodotta dal modello e la distribuzione corretta (spesso espressa in formato one-hot). Questa differenza è ciò che guida l’ottimizzazione e l’aggiornamento dei pesi della rete.

**Pagina 40 — Embedding Posizionali**

Si introduce un concetto chiave: in una frase come *"Bank of America financed a repair of the river bank"*, è fondamentale che la rete riconosca che la prima occorrenza di “bank” si riferisca a un’istituzione finanziaria e non alla riva di un fiume. Questo è reso possibile grazie all’informazione posizionale, cioè alla capacità del modello di tracciare dove ciascuna parola si trovi nel testo.

**Pagina 41 — Embedding Posizionali (continuazione)**

Prima dell’avvento dei transformer, i modelli neurali rappresentavano la posizione delle parole tramite una codifica numerica separata. L’embedding in input era una concatenazione tra il vettore semantico della parola (es. word2vec) e un vettore che codificava la sua posizione nel testo. Durante l’addestramento, la rete imparava sia i significati lessicali delle parole sia le rappresentazioni numeriche associate alle posizioni.

**Pagina 42**

Prima dell’introduzione dei transformer, le reti neurali gestivano le informazioni posizionali attraverso una rappresentazione numerica separata. In pratica, ogni parola era rappresentata da un vettore di embedding composto da due parti: da un lato l’embedding classico del termine (ad esempio, come nei modelli word2vec), dall’altro un vettore che descriveva numericamente la posizione del termine nella sequenza. Durante l’addestramento, anche queste rappresentazioni posizionali venivano apprese. L’embedding d’ingresso risultava quindi dalla concatenazione tra la rappresentazione della parola e quella della posizione.

**Pagina 43**

Nei transformer, invece, per ogni posizione nel testo in ingresso viene generato un vettore unico che serve a identificare quella posizione specifica. In questo schema, l’embedding d’ingresso di una parola non è più una concatenazione, ma la somma tra il suo embedding e il vettore posizionale corrispondente. È importante notare che questa funzione posizionale è predefinita (hard-coded) e non appresa dal modello.

**Pagina 44**

L’idea alla base dell’approccio dei transformer è quella di aggiungere direttamente all’embedding d’ingresso un valore che incorpora l’informazione posizionale, anziché gestire un vettore separato. Siccome ciascun embedding ha una certa dimensionalità (512 dimensioni nel modello base), è necessario un metodo per inserire in ciascuna dimensione informazioni sulla posizione della parola. L’obiettivo è fare in modo che ogni posizione nella sequenza venga rappresentata in modo unico e distinguibile lungo le varie dimensioni.

**Pagina 45**

Per ottenere l’informazione posizionale, si usano funzioni sinusoidali: il seno viene applicato alle dimensioni pari e il coseno a quelle dispari. La formula adottata fa uso della posizione nella sequenza (pos) e della dimensione del modello (dmodel). Questo metodo è vantaggioso perché consente al modello di ricostruire la posizione del termine anche a posteriori, basandosi solo sui valori nei vettori. Inoltre, il pattern creato da seno e coseno è facilmente distinguibile e non dipende da una quantità arbitraria di dati appresi.

**Pagina 46**

Un esempio pratico di utilizzo dei positional encoding è dato da una frase complessa, nella quale i termini "arraignment" e "court" si trovano rispettivamente agli indici 11 e 27. Si parte dagli embedding dei termini, si somma il vettore posizionale corrispondente e si ottiene un nuovo vettore risultante che incorpora anche la posizione del termine. Tuttavia, questo tipo di somma può modificare in maniera significativa l’embedding originale. Per ridurre il rischio che l’informazione originale venga diluita o mascherata, si può amplificare il valore del positional encoding attraverso un fattore di normalizzazione, tipicamente la radice quadrata della dimensionalità del modello.

**Pagina 47 – Positional Encoding**

Nel contesto delle reti Transformer, è possibile sommare direttamente il valore di codifica posizionale al vettore di embedding della parola. Per esempio, se prendiamo il vettore di embedding della parola *court* (che nel nostro esempio ha indice 27), possiamo aggiungervi il corrispondente vettore posizionale. Tuttavia, questo metodo presenta un problema: si rischia di alterare o addirittura perdere l'informazione contenuta nel vettore di embedding originale. Per evitare che la codifica posizionale sovrascriva o mascheri troppo il significato originale della parola, è possibile amplificare i valori posizionali (ad esempio moltiplicandoli per la radice quadrata della dimensione del modello) così da bilanciare meglio le due componenti.

**Pagina 48 – Sub-word Tokenization**

Le reti Transformer non operano direttamente su parole intere, ma su unità più piccole chiamate *subword units*, ovvero frammenti di parola. Questi token sono ottenuti automaticamente utilizzando un algoritmo chiamato **Byte Pair Encoding (BPE)**. Questo metodo costruisce un vocabolario di simboli partendo dai singoli caratteri (incluso un simbolo speciale </w> che indica la fine di una parola). Successivamente, il sistema conta le coppie di simboli più frequenti in un ampio corpus testuale e le unisce per formare nuovi simboli, che vengono aggiunti al dizionario.

**Pagina 49 – Sub-word Tokenization (esempio)**

Prendiamo ad esempio una frase come *“bank of the river”* suddivisa inizialmente in:  
b a n k </w> o f </w> t h e </w> r i v e r </w>  
Se la sequenza di caratteri più frequente fosse *“t h”*, allora questi due caratteri verrebbero uniti in un singolo simbolo, producendo qualcosa come:  
b a n k </w> o f </w> th e </w> r i v e r </w>  
Successivamente, se anche *“th e”* risultasse una sequenza frequente, verrebbe anch'essa unita:  
b a n k </w> o f </w> the </w> r i v e r </w>  
E così via, in modo iterativo. Alla fine, il vocabolario conterrà tra i **30.000 e i 50.000 simboli**: un numero molto inferiore rispetto al vocabolario totale di una lingua, motivo per cui le parole meno comuni verranno suddivise in più unità sublessicali.

**Pagina 50 – Vantaggi della Sub-word Tokenization**

La suddivisione in sottoparole rende i Transformer molto più robusti nei confronti di parole sconosciute. Ad esempio, se la parola *“transformers”* appare per la prima volta dopo la fase di training, un algoritmo di embedding tradizionale potrebbe non sapere come gestirla. Ma una rete Transformer può comunque segmentarla in sottoparti già note, come *“transform”* e *“ers”*, per le quali possiede già vettori addestrati.  
Un altro vantaggio è di tipo computazionale: i vocabolari delle lingue crescono indefinitamente. Se un vocabolario contenesse un miliardo di parole, con vettori di 512 dimensioni a 4 byte ciascuno, occorrerebbero oltre **2 GB** di spazio. Un vocabolario di 50.000 simboli, invece, richiede **meno di 100 MB**, offrendo una riduzione significativa dello spazio necessario per memorizzare gli embedding.

**Pagina 51 – Input Embedding**

Il livello degli *input embeddings* è responsabile della conversione dei token in vettori numerici di dimensione fissata, tipicamente 512. Il tokenizer entra in gioco trasformando le frasi di input in sequenze di token. Ogni tipo di tokenizer (ad esempio quello di BERT o di RoBERTa) ha il proprio metodo specifico di segmentazione, ma il risultato finale è sempre un insieme di token compatibile con il sistema di embedding utilizzato.

Inizio modulo

Fine modulo