questo libro, ma anche per la sua stessa natura, e dunque non è possibile affrontarlo in modo definitivo.
La storia della relazione tra cervello e linguaggio dunque non finisce certamente qui, semmai inizia: quel che è certo è che né la neuropsicologia può fare a meno della linguistica né la linguistica della neuropsicologia. Per fortuna, il linguaggio, come l'universo, è patrimonio comune di chiunque sia interessato ad indagarlo.

NOTA BIBLIOGRAFICA

Forniamo qui di seguito una serie di indicazioni bibliografiche utili per tutti i lettori che vogliano approfondire i temi trattati nell'appendice.
Abutalebi *et al.* [2007], Abutalebi *et al.* [2009], Anderson [2008], Bates *et al.* [2003], Capitani *et al.* [2003], Cappa, Cavallotti e Vignolo [1981], Cappa [2008], Cappa [2012], Caramazza e Shelton [1998], Chesi e Moro [2012], Chomsky [1956; 2012], Demonet *et al.* [1992], Dronkers *et al.* [2007], Embick *et al.* [2000], Friederici *et al.* [2006], Friston [1997; 2002], Gainotti [2000], Goodglass e Kaplan [1983], Gorno-Tempini *et al.* [2004], Graffi [2001; 2010], Indefrey e Levelt [2004], Lazzari *et al.* [2010], Lenneberg [1967], Magrassi *et al.* [2010], Mahon e Caramazza [2008], Marcus e Fisher [2003], Mazzocchi e Vignolo [1979], McCarthy e Warrington [1988], Minagar, Ragheb e Kelley [2003], Monti, Parsons e Osherson [2009], Moro [2018], Moro *et al.* [2001], Musso *et al.* [2003], Odifreddi [1989], Pallier, Devauchelle e Dehaene [2011], Patterson, Nestor e Rogers [2007], Petersen *et al.* [1988], Pinel *et al.* [2012], Rizzi [2009], Rorden, Karnath e Bonilha [2007], Sahin *et al.* [2009], Seghier *et al.* [2008], Shallice [1988], Terrace *et al.* [1979], Tettamanti *et al.* [2002], Tettamanti *et al.* [2009], Vernes *et al.* [2008], Visser *et al.* [2010], Vitali *et al.* [2007].

2. Linguaggio e Intelligenza Artificiale

1. I MODELLI LINGUISTICI STATISTICI

Il 20 novembre 2022 verrà ricordato nella storia dell'**Intelligenza Artificiale** (o *Artificial Intelligence, AI*) come il giorno del lancio dell'assistente virtuale **ChatGPT**. OpenAI, l'azienda che lo ha creato, si era specializzata da qualche anno nello sviluppo di soluzioni legate all'AI ed era da poco salita agli onori di cronaca per un altro prodotto, DALL-E, capace di generare immagini originali partendo da istruzioni testuali (ad esempio, «vorrei un'immagine in stile pop art della torre di Pisa sulla luna»). Come molti predecessori, ChatGPT accettava domande formulate in lingua naturale e forniva risposte utili a spiegare un argomento complesso, riformulando a comando la descrizione fornita in modo da renderla più semplice o più ricca di dettagli. Poteva inoltre tradurre un testo da una lingua all'altra o riassumerne il contenuto in poche frasi. Riusciva poi ad inventare filastrocche e scrivere l'incipit di un romanzo (o di una tesi di laurea). Vista la qualità delle produzioni, il successo del servizio è diventato presto virale: in cinque giorni oltre un milione di utenti singoli si sono iscritti al servizio chiedendo all'agente conversazionale (un *chatbot*, in gergo tecnico) di risolvere problemi logici, inventare nuove ricette, scrivere e correggere software in vari linguaggi di programmazione. Mai nella storia dei social network era stato registrato un successo tanto evidente.
Questo software, a cui nel corso dei mesi sono seguite nuove e più performanti versioni, ha segnato un'ondata di prodotti simili (Google Gemini, LLaMA di Meta, Claude di Antropic, Mistral AI, giusto per citarne alcuni) che hanno reso evidenti i successi ottenuti negli ultimi anni dalle tecniche di **apprendimento automatico** (*machine learning, ML*). Tutti questi sistemi sono basati su **modelli linguistici statistici di grandi dimensioni** (o *very Large Language Model, vLLM*). In queste pagine cercheremo di affrontare alcune questioni

fondamentali legate a come lo studio della linguistica sia utile a comprendere come funzionano questi modelli e in cosa siano simili e dissimili dalla facoltà espressa dalla nostra competenza linguistica.

1.1. Parlare con le macchine

In una introduzione al tema dell'Intelligenza Artificiale, Jerry Kaplan definisce questo termine, coniato da John McCarthy durante un convegno nell'estate del 1964, come un'eccellente trovata pubblicitaria che ha garantito a questa disciplina un duraturo interesse sia in ambito accademico che fuori. Il paragone con il nome scelto per la disciplina che studia il «volo artificiale», cioè l'aeronautica, è calzante: se gli aerei fossero stati chiamati «uccelli artificiali» ci saremmo forse preoccupati (come sta succedendo adesso rispetto ai rischi legati all'**Intelligenza Artificiale Generale**, AGI) dei pericoli che avremmo corso quando queste macchine avrebbero iniziato a costruirsi nidi artificiali e a riprodursi? Ci sembra questo un buon argomento, utile per affrontare con la giusta disillusione l'approccio informatico alla manipolazione del linguaggio umano.

Alan Turing è stato tra i primi a porsi il problema della simulazione dell'intelligenza umana attraverso dispositivi meccanici che oggi chiamiamo computer: il **test di Turing** (*imitation game*, Turing, 1950) definisce una procedura abbastanza semplice attraverso cui un programma può essere valutato da parlanti nativi. Se dopo un ragionevole lasso di tempo, un giudice umano che interagisce con due entità distinte utilizzando un'interfaccia testuale non si accorge di quale delle due entità sia umana e quale artificiale, il software che gestisce la conversazione artificiale si ritiene abbia superato il test. Nel corso degli anni, varie versioni del test di Turing sono state proposte e un notevole numero di conversatori artificiali sono stati valutati. Recentemente, anche **ChatGPT**, nella sua quarta e più performante versione (marzo 2024), è stato testato. La sua interazione è stata giudicata indistinguibile da quella umana nel 54% dei casi [Jones e Bergen 2024]. Interessante notare come i sospettosi giudici umani abbiano attribuito lo status di «esseri umani» solo il 67% delle volte in cui, in effetti, interagivano con persone reali. Sebbene la questione della valutazione sia piuttosto complessa, in questa appendice vale la pena di formulare una domanda più squisitamente linguistica: un software che simula così bene la competenza linguistica deve possedere una conoscenza (almeno implicita) di tutto ciò che è stato descritto in questo libro in quanto parte della competenza linguistica umana? La risposta breve è no, non c'è bisogno di avere una competenza linguistica umana per far credere di possederla. Per essere formulata correttamente, questa posizione ha però bisogno di argomentazioni precise che proveremo a sviluppare nelle pagine che seguono.

Trucchi pseudolinguistici

La prima cosa da chiarire è che ci sono, in effetti, abili trucchi a cui i sistemi di Intelligenza Artificiale ricorrono per manipolare i simboli linguistici che noi umani utilizziamo per comunicare. Un primo esempio consiste nell'utilizzo delle **espressioni regolari** (*regular expressions, RE*): le espressioni regolari sono notazioni algebriche che permettono di identificare un insieme preciso di stringhe in base ad uno schema di identificazione. Ad esempio, l'espressione regolare «..? capitol[oi]» identifica sia le parole *il capitolo* che *i capitoli*, ma anche *a capitolo* e numerose altre espressioni non necessariamente presenti in nessun corpus della lingua italiana, tipo *li capitolo*. Nella notazione algebrica che definisce lo schema di riconoscimento di un'espressione regolare, il simbolo «.» significa «qualsiasi carattere», il simbolo «?» segue un carattere che può essere o non essere presente, mentre le parentesi quadre racchiudono i caratteri che vengono accettati come alternative, «[oi]» significa quindi «o» oppure «i»; infine, le parentesi tonde identificano un blocco di riconoscimento che può essere utilizzato nell'operazione di sostituzione come mostreremo di seguito. L'ordine dei caratteri nello schema di riconoscimento è fondamentale e l'espressione regolare «abc» sarà chiaramente diversa da «bca» (per fare un po' di esercizio, conviene notare che il pattern «[abc]» non riconosce nessuna delle due stringhe precedenti, ma solo i caratteri *a*, oppure *b*, oppure *c* in alternativa; per un approfondimento, v. Lenci *et al.* [2022]).

Una versione delle espressioni regolari è utilizzata nel primo agente conversazionale ELIZA [Weizenbaum 1966] che forse, in un'epoca pre-social, ha ricevuto attenzioni paragonabili a quelle suscitate dai primi modelli **ChatGPT**. Questo è un esempio di conversazione elicitata (in inglese e tradotta in italiano) utilizzando il simulatore di ELIZA ancora disponibile sul web:

(1) ELIZA: Ciao, come va? Parlami del tuo problema.
UTENTE: Bene, ma devo scrivere un libro e non so da dove iniziare.
ELIZA: Hai detto che devi scrivere un libro.

TAB. 1. Alcuni esempi d'uso delle espressioni regolari per il riconoscimento di stringhe di testo

SCHEMA DI RICONOSCIMENTO	SPIEGAZIONE	TESTO IDENTIFICATO
L[ae] cas[ae]	Le parentesi quadre includono i caratteri in alternativa	*La casa* oppure *Le case*, ma anche *La case*
L(a|e) cas(a|e|et-ta|ette)	Le parentesi tonde esprimono blocchi all'interno dei quali le alternative sono separate dal segno «|»	*La casa* oppure *La casetta*, ma anche *La casette*
Sal.?ti	Il segno «.» esprime qualsiasi carattere, «?» la presenza opzionale del simbolo precedente	*Salti* oppure *saluti*, *salati*, ma anche *saltti*
Sal.*ti	Il simbolo «*» indica la ripetizione del simbolo (o del blocco) precedente	Stessi match precedenti, ma anche *salutati* e altre infinite espressioni che iniziano per *Sal* e terminano con *ti*

UTENTE: Esattamente. Tutti dicono che dovrei farlo.
ELIZA: Chi, ad esempio?

L'obiettivo del programma è quello di simulare il comportamento di uno psicologo e far parlare l'utente dei propri problemi, senza un reale intento comunicativo. La conversazione può procedere all'infinito e le risposte di ELIZA vengono semplicemente gestite da una cascata di regole di tipo «*se* l'input contiene questo pattern, *allora* rispondi questo»; ecco le due semplici regole che sono state applicate per gestire la conversazione precedente (il codice sorgente di ELIZA è pubblicamente disponibile sul web, «x» corrisponde a un'espressione regolare tipo «.*»):

(2) a. *se* «devo (x)» *allora* «hai detto che devi (x)»
b. *se* «tutti» *allora* «chi, ad esempio?»

Alcune parole chiave (ad esempio *tutti*) vengono ordinate per rilevanza e permettono di gestire la priorità delle regole che scattano al riconoscimento di una sequenza di caratteri. È chiaro come questo approccio non abbia niente a che fare con la nostra competenza linguistica, ma sia sufficiente a illudere un giudice ingenuo (questo avviene nel 22% dei casi, sempre secondo lo studio di Jones e Bergen [2024]). Viene ovviamente da chiedersi se i moderni agenti conversazionali si basino su approcci paragonabili. Secondo alcuni questa affermazione è potenzialmente corretta tanto che si è parlato di **pappagalli statistici** (*stochastic parrots*, Bender *et al.* [2021]) in riferimento al fatto che questi modelli manipolano simboli senza in realtà comprendere alcunché del significato delle frasi che ricevono in input o generano come output.

1.2. Dalla traduzione automatica all'apprendimento per esempi

Come appena mostrato con Eliza, i primi approcci alla manipolazione del linguaggio erano metalinguisticamente espliciti: il programmatore doveva prima rappresentare i problemi linguistici in modo computazionale (definizione di un lessico e di una grammatica strutturati in un certo modo, o, semplicemente, di particolari regole, specificazione dell'input, dell'output e delle possibili rappresentazioni intermedie), quindi trovare una procedura algoritmica per trasformare un dato input in un determinato output impiegando al meglio le risorse computazionali disponibili, completando inoltre la computazione, per ogni possibile input ricevuto, in un tempo umanamente accettabile. La soluzione adottata da Weizenbaum, in effetti, ha però ben poco a che vedere con ciò che abbiamo imparato sul linguaggio umano in questo libro.

Per riconciliarci in modo più diretto con questioni più propriamente grammaticali, il dominio linguistico in cui si può apprezzare al meglio questo approccio è quello della **Traduzione Automatica** (o *Machine Translation, MT*). È in una famosa lettera di Weaver a Wiener del marzo 1947, dove si sottolinea per la prima volta il possibile parallelismo tra il problema della decifrazione di codici e la traduzione automatica, aprendo alla possibilità di applicare anche in questo ambito la **Teoria Matematica della Comunicazione** (un approccio probabilistico basato sulle nozioni di informazione ed entropia [Shannon 1948]): «quando vedo un articolo in russo, dico "questo è in realtà scritto in inglese, ma è stato codificato utilizzando strani simboli. Procediamo quindi con la decodifica"». Bisognerà attendere il 1954 per assistere alla prima dimostrazione pubblica di un sistema di traduzione automatica sviluppato dalla IBM in collaborazione con la Georgetown University. La dimostrazione del sistema (contenente un dizionario bilingue di appena 250 parole e sole 6 regole di traduzione) ebbe un notevole successo mediatico e segnò a tutti gli effetti l'inizio dell'era della linguistica computazionale o, più precisamente, del **processamento automatico del linguaggio** (*Natural Language Processing, NLP*). Nonostante la poco lusinghiera valutazione dell'Automatic Language Processing Advisory Committee del 1964 (i sistemi di traduzione automatica, si leggeva nel report della commissione, producono testi meno intelligibili e, nel complesso, costano di più dei traduttori umani), le ricerche in questo campo non si sono mai arrestate. A cavallo tra gli anni Settanta e Ottanta sono stati tentati diversi tipi di approcci simbolici al problema linguistico. Lo schema piramidale della figura B.1, in cui alla base sta la **traduzione diretta** (parola per parola) e al vertice una traduzione basata su una forma di **interlingua**, cerca di rappresentarli comparativamente.

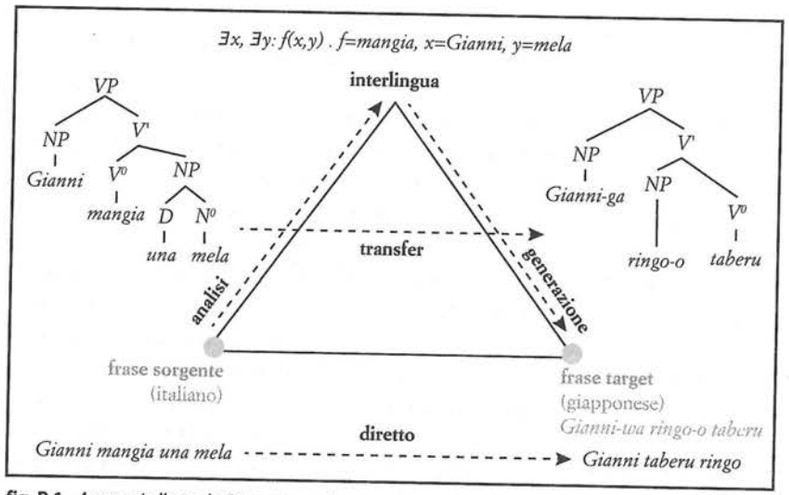

fig. B.1. *Approcci alla traduzione automatica.*

Sul lato dell'analisi (versante sinistro della piramide), i vari sistemi di traduzione automatica si collocano più o meno in alto a seconda della ricchezza dell'elaborazione linguistica intrapresa. Ad esempio, se a un'analisi morfologica che distingue le radici dalle flessioni si aggiunge un'analisi sintattica che produce un albero a costituenti (v. cap. VII), a questa rappresentazione potranno applicarsi delle regole di **transfer** che potrebbero lavorare sui **parametri** di variazione linguistica (come introdotto nel cap. III). Secondo la tradizione generativista, quello rilevante in questo caso potrebbe essere il parametro legato alla linearizzazione **testa-complemento** (o *head-directionality parameter*, Chomsky [1981]): il predicato (la testa del sintagma verbale) può precedere (lingua a testa iniziale) o seguire (lingua a testa finale) il complemento oggetto diretto (argomento interno). La linea orizzontale connette l'analisi alla sintesi (o generazione) sul versante destro della piramide che alla fine del processo produrrà la traduzione della frase di input nella lingua target. Nella figura B.1, ad esempio, poiché l'**italiano** e il **giapponese** si caratterizzano rispettivamente come lingua a testa iniziale la prima, e a testa finale la seconda, si potrà prevedere una semplice regola di transfer che inverta l'ordine di <predicato, oggetto diretto> in <oggetto diretto, predicato> (per una significativa evoluzione dell'approccio parametrico, si può fare riferimento a Roberts [2019]). Inoltre, un'adeguata rappresentazione della struttura dei sintagmi nominali (per un approfondimento si vedano i lavori di Abney [1987] e Giorgi *et al.* [1991], tra gli altri) permetterà di ricondurre l'assenza del determinante in giapponese e la presenza del marcatore di caso «enclitico» (sintagma «K» in (3)) alla salita della testa nominale verso la periferia del sintagma nominale [Longobardi 1994] come mostrato schematicamente utilizzando le parole dell'italiano di seguito:

(3)

Italiano	→	*Giapponese*
$[_K \varnothing [_D$ una $[_N$ mela]]]		$[_K$ mela$_i$-marca_di_caso $[_D \varnothing [_{N-i}]]]$

Proseguendo nell'analisi, idealmente, si dovrebbe ottenere una rappresentazione semantica universale utile, ad esempio, ad inferire che il soggetto, nella frase precedente, sarebbe più appropriatamente marcato come «topic» (*-wa*) anziché come «nominativo» (*-ga*) vista la sua prominenza.
Come si è intuito nel capitolo VIII, il problema della rappresentazione semantica non è di facile soluzione e, ad oggi, non esiste un formato di rappresentazione universalmente riconosciuto, nonostante i chiari vantaggi che questa rappresentazione potrebbe avere: in effetti, contrariamente ad un approccio basato sul transfer che richiede regole specifiche per ogni coppia di lingue gestita dal sistema, l'approccio basato su un'**interlingua** permetterebbe semplicemente di concentrarsi sull'analisi e la generazione nella nuova lingua che si vuol integrare nel sistema di traduzione, senza preoccuparsi delle altre già presenti. Pochi sistemi hanno pienamente adottato questo approccio, peraltro con scarsi risultati (si veda il sistema Rosetta, descritto in Hutchins e Somers [1997]).

Storicamente, ogni tipo di approccio simbolico si è rivelato molto difficile da implementare computazionalmente. Se il generale fallimento di queste prospettive basate su teorie linguistiche esplicite sia da imputarsi alla povertà dei modelli linguistici adottati, alla loro intrinseca complessità o all'impossibilità teorica di gestire il problema computazionale in ottica algoritmica efficiente, non è facile da stabilire e in queste pagine non potremo approfondire la questione.
La reazione a queste difficoltà che ha condotto ai modelli oggi popolari è invece interessante e vale la pena di provare ad esplorarla: il modo migliore per gestire aspetti linguistici è sembrato non quello di imporre una conoscenza (grammaticale) dall'alto, ma cercare di astrarre proprietà utili da esempi. L'approccio generalmente chiamato **apprendimento per esempi** (o *example-based learning*, o semplicemente **machine learning**) è stato la chiave del successo di diversi sistemi di traduzione automatica a partire dagli anni Novanta. Il sistema **Candide** [Berger *et al.* 1994], ad esempio, è stato tra i primi a sfruttare un corpus parallelo francese-inglese. L'approccio adottato è stato da subito quello probabilistico basato sull'aggiustamento di specifici parametri all'interno di una formula che cercava di mappare una rappresentazione vettoriale (cioè numerica e a molte dimensioni) delle frasi di input e di quelle in output (le loro traduzioni nella lingua target). Le previsioni di traduzione cercavano di sfruttare al meglio le probabilità di allineamento (traduzione, appunto) tra pezzi di frasi nella lingua sorgente e in quella target. Sebbene la teoria sui parametri utili a massimizzare la qualità di questo allineamento si sia molto evoluta, il concetto di base è piuttosto semplice ed è utile enunciarlo qui in modo da poter apprezzare la logica che ha guidato la ricerca fino ai vLLM.
Data una frase *f* in francese, il problema della sua traduzione è formulabile in termini di individuazione della frase *e*, in inglese, che «massimizza la probabilità» di allineamento tra *f* e le traduzioni alternative a *e*. Da un lato, il calcolo della probabilità condizionale di una traduzione *e* della frase *f* (cioè *P(e|f)* che si legge come «la probabilità di *e* data *f*») potrebbe ridursi ad un calcolo piuttosto semplice, cioè contare quante volte la frase *f* è stata tradotta esattamente come *e*, moltiplicando questo numero per la frequenza di *e* nel corpus di traduzioni, dividendo quanto ottenuto per la frequenza di *f*. Questo è il **Teorema di Bayes** che ci dice appunto che $P(e|f) = (P(f|e) P(e))/P(f)$. D'altro lato, è però inutile sperare che tutte le frasi che vogliamo tradurre siano in effetti presenti nel corpus parallelo a cui facciamo affidamento: questo è impossibile, visto che l'insieme delle frasi grammaticali in ogni lingua è infinito. Non resta quindi che cercare di calcolare questa probabilità di allineamento tra la frase *f*, che ci è stata fornita, e la sua tradizione *e*, che forse non abbiamo mai visto prima, partendo dalla scomposizione di queste frasi. L'intuizione fondamentale è che una frase sia il prodotto delle concatenazioni di probabilità delle parole che la compongono (l'argomento

è qui necessariamente semplificato; per un approfondimento v. Ježek e Sprugnoli [2023]). Dovremo quindi aggiungere altri parametri alla nostra formula. Ad esempio, potremmo considerare non la frase completa, ma la sua scomposizione in trigrammi, cioè in sequenze di tre parole. Nell'esempio in figura B.1, la frase «Gianni mangia una mela», considerando il simbolo «#» come inizio e fine di un'espressione da tradurre, è composta da quattro trigrammi, cioè «# Gianni mangia», «Gianni mangia una», «mangia una mela» e «una mela #». Se è possibile che l'intera frase non sia presente nel corpus di riferimento, è però probabile che almeno qualche trigramma in cui viene scomposta la frase lo sia. Si considerano poi anche i bigrammi (sequenze di due parole), o addirittura i singoli unigrammi (la frequenza delle singole parole). Tutte queste probabilità di allineamento tra frammenti di frase nella nostra lingua sorgente e frammenti di frase nella lingua target finiscono per essere inseriti in una formula di allineamento più complessa. Ogni singola componente della formula è quindi pesata da specifici **parametri** in modo che il contributo di ogni componente possa essere modificato in fase di apprendimento. Ad esempio, è ragionevole che il modello concluda che le probabilità di allineamento per trigrammi siano molto più affidabili di quelle dei bigrammi. In questo caso, i parametri associati ai trigrammi avranno un valore più alto, premiando queste probabilità (quando disponibili) rispetto ad altre (quando disponibili). Questi e altri parametri furono utilizzati dal software Candide. La vera rivoluzione fu però nell'utilizzo di **parametri** associati a stati/processi **«nascosti»** (*hidden*): uno stato «nascosto» è ad esempio la categoria morfosintattica di una parola. «Gianni», in qualità di nome proprio, o in qualità di soggetto grammaticale, potrebbe ricevere proprietà speciali di cui tener conto in fase di traduzione (ad esempio, in giapponese, deve ricevere il caso nominativo o il marcatore di topic). Gli stati nascosti non sono direttamente osservabili (cioè contabili, come invece si può fare con le parole), ma devono essere postulati *a priori* nel modello. Dal punto di vista probabilistico la situazione non cambia di molto: se prima cercavamo di calcolare la probabilità di *e* data *f*, adesso cerchiamo di fare la stessa cosa, considerando anche *h*, cioè gli stati nascosti (*P(e|f, h)*). Da un lato, abbiamo il contesto (calcolato stimando la probabilità di traduzione tra pezzi di frasi e non frasi intere) che trasforma «*f*» ed «*e*» in concatenazioni di *n*-grammi, dall'altro, consideriamo l'ipotesi che fattori morfosintattici impliciti debbano entrare nel computo della probabilità di una determinata traduzione. L'utilizzo dei parametri associati a stati/processi nascosti permette di teorizzare modelli più complessi noti come **Hidden Markov Models** (*HMM*). Due problemi sostanziali rendono però questi modelli difficili da applicare al contesto linguistico della traduzione automatica: il primo riguarda la necessità di determinare la giusta traduzione concatenando pezzi di frasi non necessariamente contigui, il secondo riguarda la difficoltà di stimare *a priori* le probabilità di ogni stato/processo nascosto. Gli attuali vLLM forniscono buone soluzioni per questi problemi preliminari.

1.3. Architetture di rete adatte all'apprendimento di aspetti linguistici

Un'ulteriore metafora ha segnato il successo dell'approccio basato sull'apprendimento automatico: quella delle **Reti Neurali Artificiali** (o *Artificial Neural Network, ANN*). L'intuizione di base si fondava sull'implementazione di semplici unità computazionali ispirate al comportamento dei neuroni (si veda appendice precedente) connesse attraverso connessioni pesate [McCulloch e Pitts 1943] che mettevano in relazione un input con un output discreto. Aggiungendo inoltre uno o più unità nascoste (*hidden layers*) per risolvere problemi logicamente impossibili da risolvere con un'architettura a un solo livello [Minsky e Papert 1988] si è aperta la strada a nuovi metodi di apprendimento sempre però volti a risolvere il problema dell'individuazione dei migliori valori di specifici **parametri** all'interno di una formula complessa (in questo caso, i valori delle connessioni pesate).

Per chiarire il concetto, prendiamo una rete associativa a tre livelli, uno di input, uno di output e l'altro «nascosto» (fig. B.2).

L'obiettivo della rete associativa è proporre sinonimi di diversi nomi visti precedentemente in fase di addestramento. Le unità di input (i_1-i_n) e di output (o_1-o_n) sono entrambe di *n* nodi, con *n* uguale al vocabolario utilizzato. La codifica dell'input e dell'output segue infatti la tecnica dell'**one-hot encoding**, ovvero ogni nodo rappresenterà esclusivamente una sola parola del dizionario. Input e output saranno quindi delle sequenze di 0 e un 1 soltanto. Ad esempio, *casa* = <1, 0, 0, 0, …, 0>, *albergo* = <0, 1, 0, 0, …, 0>, *abitazione* <0, 0, 1, 0, …, 0>. Questa codifica garantisce l'assenza di bias nella rappresentazione delle parole. Per apprendere che *casa* e *abitazione* sono sinonimi, la rete dovrà essere addestrata, cioè, dovremo mostrarle l'abbinamento che ci aspettiamo produca. La rete apprenderà, semplicemente modificando i pesi u_1-u_n e v_1-v_n cercando di minimizzare l'errore prodotto al primo tentativo. Matematicamente parlando, la formula da applicare per determinare il valore di attivazione che riceverà il nodo o_1 è la seguente:

(4) $\Sigma_{i=1}^{3} v_i h_i$

In parole semplici, poiché ci sono solo 3 nodi nell'hidden layer, solo tre pesi dovranno essere aggiustati per produrre un valore più vicino all'1 o allo 0, a seconda se vogliamo che la parola prodotta sia <1, …> o <0, …>. Se in input riceviamo la parola *abitazione*, cioè <0, 0, 0, 1, …, 0>, allora potrebbe essere sensato avere come output <1, 0, 0, 0, …, 0>, cioè *casa*. La somma dell'attivazione di h_1 per v_1, più h_2 per v_2, più h_3 per v_3 determinerà il valore di attivazione di o_1 attraverso una funzione di attivazione (ad esempio una funzione

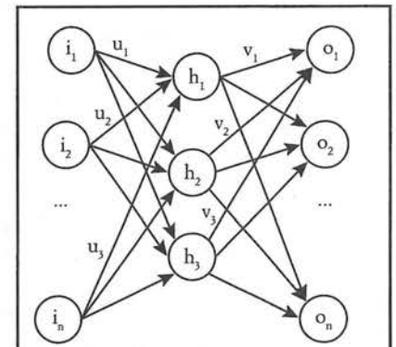

fig. B.2. *Rete associativa completamente connessa, con «n» nodi di input (i), «n» nodi di output (o) e tre nodi nell'hidden layer (h).*

a soglia che restituirà 1 se la sommatoria supererà un certo numero, o 0 altrimenti). Considerando la situazione semplicistica in cui i valori v oscillano da 0 a 1, è facile intuire che più vicino a 1 sarà il valore del peso, più influente sarà il contributo di h nell'attivazione di o. Alternativamente, più vicino a 0 il peso, più piccolo il contributo della connessione all'attivazione prodotta. Lo stesso calcolo andrà fatto per tutti i nodi di output e le connessioni che li raggiungono. Inoltre, la stessa riflessione va fatta a ritroso, con il fine di registrare le connessioni tra i nodi di input e quelli dell'hidden layer. I pesi delle connessioni varieranno in base all'errore prodotto al primo tentativo: ad esempio se la rete avrebbe dovuto produrre <1, ...> e invece ha prodotto <0, ...> allora quelle connessioni che hanno contribuito erroneamente a produrre 1 verranno «punite» (il loro valore ridotto), mentre quelle che hanno prodotto 0 anziché 1 dovranno essere leggermente alzate. La variazione sarà determinata da un parametro definito come **tasso di apprendimento** (*learning rate*). Questa variazione verrà ridistribuita proporzionalmente tra tutti i nodi che hanno prodotto l'errore. La procedura di propagazione all'indietro dell'errore è chiamata **back-propagation** ed è proporzionalmente più complessa e difficile da attuare a mano a mano che la dimensione della rete aumenta e la struttura si fa più articolata. In pratica, per aiutare la rete ad apprendere, ci vorranno molti più esempi e molte ripetizioni di ogni esempio (epoche di apprendimento).

Nel caso precedente, ogni volta che mostriamo un esempio alla rete (ad esempio la coppia *casa-abitazione*), la rete proverà a ridurre il proprio errore modificando (in modo casuale, all'interno dei valori ammissibili a seconda del learning rate scelto) il valore dei parametri della nostra formula (che, grazie alla metafora che abbiamo adottato sono diventati i «pesi» delle connessioni).

Una questione di tempo

Una volta che la rete è stata addestrata, se le si chiede di produrre un sinonimo per *casa*, la sua risposta non sarà influenzata da quella che è stata data alla richiesta precedente (tipo produrre un sinonimo per *sedia*). Se questo è accettabile per il task appena descritto, le cose si fanno molto meno plausibili per quanto riguarda la traduzione automatica. Prendendo spunto dalla nostra semplicistica previsione per n-grammi, è logicamente sbagliato assumere che la scelta fatta per tradurre *mangia una mela* sia indipendente da quella fatta per tradurre il trigramma precedente # *Gianni mangia*. È inoltre cognitivamente poco plausibile pensare che l'unico modo per insegnare qualcosa a questi modelli sia quello di mostrare loro esattamente quello che devono fare (**apprendimento supervisionato**, *supervised learning*). Come si è visto nel capitolo XII, il bambino apprende il linguaggio ignorando beatamente le correzioni proposte dall'adulto e partendo da un input probabilmente insufficiente a inferire le complesse proprietà grammaticali che siamo in grado di esprimere come parlanti adulti. Possiamo quindi ragionevolmente assumere che l'**apprendimento** linguistico umano sia fondamentalmente **non supervisionato** (*unsupervised learning*).

Entrambi questi problemi sono stati brillantemente affrontati da Jeffry Elman all'inizio degli anni Novanta. Elman ha introdotto una sostanziale modifica alla struttura delle reti come quelle illustrate in figura B.2, aggiungendo un **layer contestuale** (*context layer*) [Elman 1990] con l'obiettivo di conservare lo stato di attivazione dei nodi dell'hidden layer e riproporlo in aggiunta all'input nella previsione successiva (fig. B.3a).

In pratica, nella sequenza di previsioni, la rete dovrà tener conto non solo nel nuovo input ricevuto al tempo t_i, ma anche di quella che è stata l'attivazione del suo hidden layer al tempo t_{i-1} (fig. B.3b). Le connessioni tra hidden e context layer non sono quindi parametri da modificare, ma connessioni 1-a-1, in cui tutta l'attivazione al tempo t_i viene riprodotta sullo stesso layer al tempo t_{i+1}.

Possiamo considerare questa intuizione come una vera e propria rivoluzione per due motivi: prima di tutto, grazie a questo escamotage possiamo scegliere di non rinunciare a dare in pasto alla rete sequenze infinite di simboli. Questo era in effetti impossibile prima, visto che il numero di nodi di input determinava la lunghezza massima della sequenza processabile. Con l'idea del context layer, possiamo fornire alla rete una parola dopo l'altra, sperando che il processamento dell'ennesima parola possa essere influenzato dalle parole processate precedentemente.

Particolarmente interessante è inoltre il compito di addestramento utilizzato da Elman che potremmo definire come **autosupervisionato** (*self-supervised*): alla rete viene chiesto di prevedere semplicemente la parola successiva, quindi la rete stessa potrà utilizzare l'input immediatamente successivo per correggere la propria previsione. Concretamente, se volessimo dare in pasto alla rete la frase «Gianni mangia una mela», forniremmo al tempo t_0 la parola «Gianni» e ci aspetteremmo che la rete produca «mangia», al tempo t_1 daremo alla rete in input «mangia» e ci aspetteremmo che preveda la parola «una», tenendo conto non solo dell'input al tempo t_1, ma anche dello stato di attivazione dell'hidden layer al tempo t_0. La rete sarà quindi costretta ad astrarre delle regolarità per riuscire a prevedere che dopo un nome può seguire un verbo che si accorda alla terza persona singolare («Gianni mangia» e non «Gianni mangiano»). Infine, una volta che la rete avrà appreso, si potranno comparare

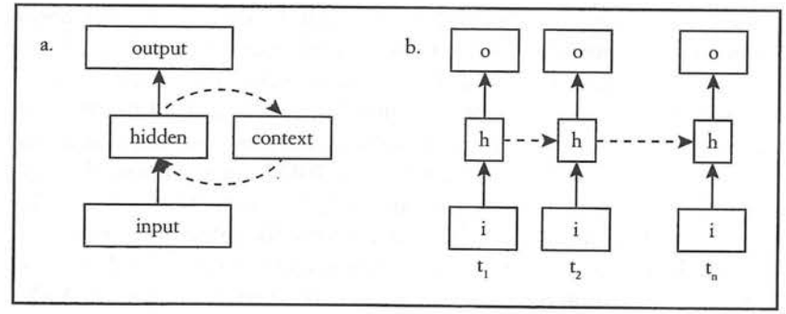

fig. B.3. *Rete ricorrente (a). Espansione della sequenza di processamento della stessa rete ricorrente (b).*

gli stati di attivazione dell'hidden layer per parole diverse: poiché ognuna di queste attivazioni corrisponde ad un vettore (una sequenza di valori, sempre della stessa lunghezza, visto che il numero di nodi dell'hidden layer è definito una volta per tutte), si potranno calcolare le distanze tra questi vettori (matematicamente il calcolo viene fatto computando il coseno tra i due vettori). Più simili saranno le attivazioni dell'hidden layer (cioè più vicino a 1 sarà il coseno tra i due vettori), più simile sarà la rappresentazione che la rete si è fatta di queste parole che ha processato in un determinato contesto. Alla fine, facendo un'**analisi gerarchica dei raggruppamenti** (o *hierarchical clustering analysis* [Elman 1990]), si può facilmente osservare che una rete che ha appreso riesce abbastanza bene a inferire proprietà morfosintattiche e semantiche delle parole, raggruppando nomi e verbi, e riuscendo inoltre a distinguere tra nomi animati, inanimati, oggetti «che si rompono» e oggetti «che si mangiano». Questo tipo di risultati è alla base di varie rappresentazioni semantiche distribuzionali [Baroni *et al.* 2014], evoluzioni di quello che in gergo si definisce il **word embedding**, cioè una rappresentazione semantica su scala dimensionalmente ridotta del vocabolario di una lingua, basata sulle proprietà distribuzionali delle parole.

Bisogna sottolineare che queste proprietà venivano inferite semplicemente sfruttando la distribuzione delle parole mediante una procedura che le elaborava una dopo l'altra, sequenzialmente. Questa procedura di apprendimento, basata sulla previsione della parola successiva, è fondamentalmente ancora oggi quella utilizzata per allenare i moderni vLLM.

Elman [1993] ha inoltre mostrato come queste reti possano essere in grado di catturare non solo dipendenze locali (ad esempio, l'accordo tra determinante e nome), ma anche alcune dipendenze non-locali, come l'accordo tra soggetto e verbo quando il soggetto è modificato da una frase relativa che allunga la distanza tra soggetto grammaticale e predicato («il cane/i cani che il padrone ha liberato ha/hanno abbaiato tutta la notte»). Ciò potrebbe suggerire che queste architetture siano in grado di astrarre proprietà gerarchiche da mere sequenze di parole. La questione si è rivelata in effetti un po' più complessa di così, ma sta di fatto che sofisticazioni di queste reti hanno permesso di mitigare vari problemi, quali il **problema dell'azzeramento del gradiente dell'errore**, ovvero il concreto rischio di non riuscire a ridistribuire un errore sufficientemente apprezzabile indietro nel tempo. Reti come le **Long-Short Term Memory networks** [Hochreiter e Schmidhuber 1997] raffinano l'idea del context layer aggiungendo delle unità che cancellano (*forget gates*) o altre che preservano (*output gates*) informazioni linguistiche utili ad arbitraria distanza temporale. Sebbene a loro volta queste reti siano state soppiantate da più performanti *transformers* [Vaswani *et al.* 2017] su cui si basano i vLLM come **ChatGPT**, a un'attenta disamina della loro performance linguistica pare che queste architetture risultino comunque sufficientemente performanti rispetto alla gestione di complesse configurazioni sintattiche [Wilcox *et al.* 2023]. Per un approfondimento sul tema del NLP attraverso l'uso di ANN, v. Tamburini [2022].

Modelli più grandi, modelli più belli? Non sempre

Gli esperimenti fin qui descritti sono tutti partiti da un corpus di allenamento ragionevolmente piccolo (dell'ordine cioè di pochi milioni di parole). Per arrivare però alle performance a cui ci siamo abituati con i modelli GPT di ultima generazione le cose cambiano sostanzialmente. Il vecchio modello GPT3 era stato allenato su una quantità di dati dell'ordine delle centinaia di miliardi di parole. Per avere un'idea, l'intera Wikipedia rappresentava circa il 3% del training set utilizzato [Brown *et al.* 2020]. Un ulteriore problema riguarda la dimensione di questi modelli. La misura adottata per misurarli è il numero di **parametri**, che grossolanamente corrisponde ai pesi modificabili delle connessioni all'interno del modello. Anche in questo caso, i dati pubblicamente disponibili ci dicono che il modello GPT3 aveva ben 175 miliardi di parametri, mentre indiscrezioni parlano di circa 1,8 trilioni di parametri per il modello GPT4. Un'ipotesi abbastanza in voga sulla predizione delle performance di questi modelli (la **scaling hypothesis** [Kaplan *et al.* 2020]) ipotizza che un significativo miglioramento sia apprezzabile solo se si raddoppia la dimensione del modello stesso (cioè il numero di parametri), il training set e, conseguentemente, i tempi e le risorse necessarie per l'allenamento. Sempre per avere un'idea pratica, è stato stimato [Strubell *et al.* 2019] che per allenare il modello GPT3 l'impronta CO_2 equivalente sia di circa 552 tonnellate, circa quanto occorre per 297 voli andata e ritorno tra Roma e New York.

Al di là dell'insostenibilità ambientale, in questa appendice la questione interessante è il ritorno in termini linguistico-teorici di questi modelli. Va ricordato, in effetti, che il principale scopo di questi prodotti è quello di risolvere problemi linguistici che vanno dalla generazione creativa di testo, alla risposta a domande, alla traduzione automatica. In pratica, quindi, oltre a gestire una significativa competenza grammaticale, questi modelli devono immagazzinare un'enorme mole di conoscenza in forma di ontologia. Si stima che gran parte dei parametri sia in effetti dedicata a rappresentare conoscenze fattuali (conoscenza del mondo), piuttosto che questioni linguistiche (per apprendere queste ultime, basterebbe un allenamento su appena 10-100 milioni di parole, stando alle stime di Zhang *et al.* [2020]). Ma come facciamo a valutare questi modelli in termini linguistici precisi? La questione è meno semplice del previsto, visto che, essenzialmente, ancora non abbiamo capito come facciano esattamente queste architetture a performare linguisticamente così bene. La questione può sembrare abbastanza sconcertante di primo acchito, ma considerando da dove siamo partiti (la difficoltà di concepire sistemi di traduzione automatica metalinguisticamente espliciti), il risultato ottenuto è in linea con quanto auspicato: risolvere un problema complesso mettendo insieme un'architettura computazionale basata su unità relativamente semplici [Brooks 1990]. Vale quindi la pena di fare un passo indietro e provare a riflettere sulle prospettive linguistiche che questi modelli riescono ad esprimere.

2. L'IMPORTANZA DEI MODELLI TEORICI

L'acceso dibattito tra una prospettiva dell'Intelligenza Artificiale «forte», cioè simbolica/algoritmica ed esplicita (cioè intelligibile in ogni suo passaggio) e una «debole», cioè subsimbolica, basata sull'approssimazione probabilistica o su una visione emergente della complessità a partire da combinazioni di operazioni estremamente semplici, quali quelle implementate dai neuroni artificiali che abbiamo discusso in § 1.3 [Rumelhart *et al.* 1999] si è molto sopito al giorno d'oggi, visto l'evidente vantaggio che ha caratterizzato questa seconda prospettiva.

Resta però un'importante questione di valutazione della **competenza** linguistica espressa da questi modelli di AI, che richiede di riflettere precisamente sul livello di **adeguatezza** che una certa teoria riesce ad esprimere [Chomsky 1965)]. Secondo Chomsky, tre sono i livelli di adeguatezza per una teoria linguistica. Il primo, e il più semplice da ottenere, è quello dell'**adeguatezza osservativa**: la teoria è in grado di distinguere le frasi ben formate da quelle malformate (agrammaticali) coerentemente rispetto a quanto farebbe un parlante nativo. Il secondo livello è quello dell'**adeguatezza descrittiva**: il modello grammaticale riesce non solo a distinguere le frasi grammaticali da quelle agrammaticali, ma fornisce anche descrizioni delle frasi grammaticali coerenti con le intuizioni del parlante nativo. Infine, c'è il livello di **adeguatezza esplicativa**: una teoria raggiunge questo livello se riesce a fornire un modello descrittivamente adeguato solo basandosi sui dati empirici primari disponibili ad un bambino che apprende la lingua.

I livelli di adeguatezza più interessanti per un linguista sono ovviamente quello descrittivo e quello esplicativo; quelli cioè che permettono, attraverso generalizzazioni linguisticamente interessanti, di avere una teoria compatta, spiegando inoltre non solo come produciamo e comprendiamo le frasi della nostra lingua, ma anche com'è possibile che un bambino riesca ad apprendere questa lingua dato un certo input, che, per molti versi, risulta essere estremamente povero. In effetti, come si è visto in XII.1., il problema della **povertà dello stimolo** [Chomsky 1981] suggerisce che nell'input linguistico primario a cui hanno accesso i bambini che apprendono una prima lingua non siano presenti dati sufficienti per inferire i principi che regolano le produzioni linguistiche umane. Un classico esempio a supporto di questo argomento riguarda l'inversione del soggetto e dell'ausiliare in inglese [Crain e Nakayama 1987]. È facile tentare un semplice esperimento con un vLLM aggiornato per forzarlo a produrre una domanda di tipo Sì/No in una lingua come l'inglese:

(5) PROMPT: Ask someone if the man who is tall was happy
Chiedi se l'uomo che è alto era felice
ANSWER: Was the man who is tall happy?
Era l'uomo che è alto felice?

Come mostrato da Crain e Nakayama, i bambini molto piccoli (intorno ai 3-4 anni) fanno spesso errori (ad esempio aggiungendo un ausiliare intrusivo ad inizio frase), ma non commettono mai l'errore di invertire l'ausiliare sbagliato («*Is the man who tall was happy?» (l'esempio è tratto dal titolo del bel documentario del 2013 su Noam Chomsky di Michel Gondry). Nel tranello non cade neppure GPT-4o con cui è stata intrattenuta la conversazione in (5). Come sottolineato precedentemente, ci sono almeno due aspetti che rendono questo risultato meno interessante del previsto: da un lato, questi modelli sono allineati su moli di dati mastodontiche e quindi irrealistiche come input primario per il bambino; inoltre, sono sottoposti a una **messa a punto** (o *fine-tuning*) basata su un esplicito apprendimento supervisionato per esempi [Brown *et al.* 2020]. Visto che i dati utilizzati per training e fine-tuning non sono noti, è plausibile pensare che specifici esempi come quello riprodotto dalla conversazione in (5) possano essere stati utilizzati in fase di training o, addirittura, per affinare il modello. La cosa è meno improbabile di quanto si possa pensare, visto che l'enorme corpus di dati utilizzato per l'apprendimento ha sicuramente incluso pubblicazioni di carattere scientifico, in cui questi contrasti sono puntualmente descritti. Questo tipo di informazione metalinguistica è chiaramente irrilevante per il bambino ma forse necessaria a questi modelli.

Resta quindi estremamente sensato cercare di simulare nel modo più accurato possibile l'input ricevuto dal bambino per vedere cosa in effetti queste architetture basate su reti neurali di ultima generazione riescano ad apprendere.

Il BabyLM Challenge [Warstadt *et al.* 2023] ha cercato di rispondere a questo quesito preparando un compito condiviso (o *shared task*), cioè una competizione in cui si chiede ai vari modelli linguistici di confrontarsi sullo stesso set di dati di apprendimento e, successivamente, venir valutati su precisi contrasti linguistici. Il corpus di allenamento (o *training set*) è in effetti di molti ordini di grandezza più piccolo rispetto a quello richiesto dai normali vLLM. Tale corpus comprende dati estratti da CHILDES [MacWhinney 2000], sottotitoli di film, canzoni, conversazioni e una piccola parte di Wikipedia. La dimensione del corpus di allenamento più ristretto raggiunge appena i 10 milioni di parole che, secondo le stime [Hart e Risley 1992; Hsu e Chater 2010], corrispondono, all'incirca, all'input ricevuto da un bambino nei primi dieci anni di vita.

I risultati della competizione completata nel 2023 sono stati molto interessanti: il modello maggiormente performante è stato quello effettivamente basato su un'architettura a Transformer, ottimizzata in modo da ignorare quelle connessioni ininfluenti a determinare la natura linguistica dell'input [Charpentier e Samuel 2023]. Ancora più interessante però, dal nostro punto di vista, è stato il risultato psicolinguistico discusso in Steuer *et al.* [2023]. Gli autori mostrano come a un miglioramento della performance dei modelli basati su transformer corrisponda in realtà un allontanamento della loro predittività in termini psicolinguistici (ad esempio i tempi di lettura registrati durante il processamento incrementale della frase). In questo senso, come sottolineato nel loro contributo che si è meritato la menzione d'onore alla competizione,

i modelli GPT sono «cattivi bambini». Per capire a questo punto il (fondamentale) ruolo del linguista teorico nel nuovo mondo fatto di probabilità e approssimazioni è quindi utile capire su che tipo di dati la performance di questi modelli venga valutata.

2.1. La valutazione della competenza linguistica

Capire se e come questi modelli siano in grado di apprendere aspetti linguistici interessanti da un punto di vista teorico e/o cognitivo è diventato un tema centrale. Secondo alcuni [Piantadosi 2023], questi modelli assurgono al ruolo di migliori modelli linguistici teorici visto che riescono laddove le più avanzate teorie linguistiche hanno fallito. Per sostenere questa controversa affermazione, Piantadosi si appella a precise metriche di valutazione: è pratica comune dei linguisti computazionali quella di valutare comparativamente i sistemi di NLP attraverso test standardizzati. Il modello migliore sarà quindi quello che si comporterà meglio in un determinato test. Un esempio di questi test è il BLiMP, il Benchmark for Linguistic Minimal Pairs in English [Warstadt *et al.* 2020]. Tale test comprende un migliaio di coppie di frasi, ciascuna versione delle quali include una minima variazione morfosintattica che rende la frase di partenza agrammaticale come indicato in (6):

(6) a. Who_i would Mark visit $\__i$ [while kissing Marla]?
Chi_i avrebbe Mark visitato $\__i$ mentre stava baciando Marla?
A chi avrebbe fatto visita Mark mentre stava baciando Marla?
b. *Who_i would Mark visit Marla [while kissing $\__i$]?
Chi_i avrebbe Mark visitato Marla mentre stava baciando $\__i$?
Chi avrebbe fatto visita Mark a Marla mentre stava baciando?

La seconda opzione è chiaramente agrammaticale visto che l'elemento interrogativo dovrebbe essere interpretato come parte del sintagma aggiunto «mentre...» violando quello che in gergo si chiama un vincolo d'«isola». I contrasti inclusi nel BLiMP sono solidamente interiorizzati nella competenza linguistica di un parlante adulto inglese, come mostrato dai giudizi raccolti sottoponendo queste coppie ad un nutrito gruppo di parlanti nativi. La performance di vari vLLM è molto buona su questi test, suggerendo che questi modelli riescano ad inferire, con buona approssimazione, interessanti generalizzazioni puramente linguistiche. Come sottolineato in altri lavori [Chesi 2024], in realtà, il dato potrebbe essere meno interessante di quanto sembri, dal momento che, come si è visto in § 1.3, un vLLM (i) viene allenato su dataset enormi e, per rispondere puntualmente a giudizi di accettabilità o scelta forzata, potrebbe aver avuto bisogno di una **messa a punto** particolare: mostrare al modello come avrebbe dovuto rispondere a un determinato contrasto come quello espresso in (6) è chiaramente sleale nei confronti del bambino che, già tra i 6 e i 7 anni, performa decisamente all'altezza di un

vLLM di ultima generazione rispetto all'individuazione di sottili contrasti grammaticali, pur non avendo subito alcuna «messa a punto» esplicita (la scuola ci proverà, inculcando nel bambino nozioni grammaticali normative, cui però l'adulto riuscirà sistematicamente a ribellarsi).
Questa osservazione non si basa su semplici intuizioni, ma adotta lo stesso rigoroso metodo imposto per la valutazione dei vLLM all'interno della competizione BabyLM: in italiano, la scelta delle coppie minime da valutare è ricaduta sul test COnVERSA [Chesi *et al.* 2024], un test volto a valutare l'abilità di discriminazione di coppie minime in bambini dai 6 anni e mezzo di età, con particolare attenzione al caso dei bambini sordi e dei bilingui. La struttura del COnVERSA è identica a quella del BLiMP inglese: ogni opposizione è costruita su un'unica variazione morfosintattica e viene proposta al bambino che dovrà scegliere la versione che preferisce. Le opposizioni strutturali includono, ad esempio, violazioni di accordo (7a), di selezione tematica (7b), di uso di pronomi (7c) o di risposta a domande (7d):

(7) a. Il piatto è pieno. vs. *Il piatto è piena.
b. Il libro cade dal tavolo. vs. *Il libro cade il tavolo.
c. Il ragazzo scivola. vs. *Il ragazzo si scivola.
d. La bambina mangia? Sì. vs. *Una torta.

I risultati della comparazione [Chesi *et al.* 2023] mostrano che il modello **GPT** più performante si comporta globalmente come un bambino di 9 anni, sebbene mostri una performance assolutamente bizzarra: nessun bambino di 9 anni è riuscito a rispondere accuratamente al cento per cento dei contrasti nelle risposte che richiedevano il processamento di frasi relative oggetto (per es.: «Ci sono due bambine. Una corre, l'altra salta ed è chiamata dai cugini. Quale bambina salta? Quella che i cugini chiamano» vs. «*Quella che chiama i cugini»), mentre i sistemi GPT sono perfetti nel gestire queste strutture complesse. Al contrario, mentre i bambini di 9 anni non mostrano problemi con l'uso di pronomi clitici oggetto diretto o indiretto (per es.: «La zia incontra il nipote e gli offre una caramella» vs. «*La zia incontra il nipote e lo offre una caramella») i modelli GPT si comportano in modo casuale con questi contrasti.
Come discusso all'inizio di questo manuale (v. cap. II.3), poiché la competenza linguistica è composita, ognuna delle componenti ritenute rilevanti può essere valutata attraverso un test di valutazione appositamente studiato. In questo senso, il lavoro del linguista teorico risulta fondamentale per individuare i contrasti più promettenti e quelli più complessi da inferire su base meramente statistica. Esempi che vanno in questa direzione sono i test set sviluppati per le varie competizioni di EVALITA (eventi biennali organizzati dall'Associazione Italiana di Linguistica Computazionale, AILC). Due esempi di test che mirano alla valutazione delle competenze morfosintattiche sono il dataset **ItaCoLA** [Trotta *et al.* 2021], che include giudizi di grammaticalità binari sul modello dell'inglese **CoLA** [Warstadt *et al.* 2018] e **AcCompl-**

it (Acceptability & Complexity evaluation task for Italian, Brunato *et al.* [2020]), costituito sia da giudizi di accettabilità (su scale da 1, inaccettabile, a 7, perfetto) e di complessità percepita (sempre su scale da 1, semplice, a 7, complesso).
Anche la componente interpretativa è finita sotto esame: un rilevante esempio di test è quello proposto nella competizione **PreTENS** [Zamparelli 2022] proposta per SemEval 2022. In questo caso si chiede di valutare l'incongruenza di una frase rispetto all'uso di determinati connettivi che mettono in relazione due sintagmi nominali in una particolare relazione di iponimia/iperonimia come nell'esempio:

(8) a. Mi piacciono i fiori *più*... #delle rose vs. delle foglie
b. Mi piacciono i fiori, *ad eccezione*... delle rose vs. #delle foglie

3. PROSPETTIVE: VALUTAZIONE E OTTIMIZZAZIONE

Se da un lato è certo che questi sistemi di AI continueranno a sorprenderci con sempre migliori performance, dall'altro lato è evidente come dal punto di vista descrittivo questi modelli mostrino relativamente poca consapevolezza della competenza linguistica umana e siano ragionevolmente deboli come teorie cognitive. Peggio ancora, per capire cosa sanno, oltre a torturarli con esami linguistici appositamente studiati, un altro approccio molto in voga è quello di perturbare l'input attraverso modifiche puntuali in modo da dedurre modelli approssimati, quindi più semplici da spiegare [Ribeiro *et al.* 2016]. Da una parte, la necessità di rendere i vLLM più performanti li ha resi sempre più grandi e complessi, dall'altra è cresciuta anche la necessità di comprenderli e, parallelamente, di provare a spiegare precisamente la natura della nostra competenza linguistica suggerendo una tendenza inversa di riduzione, ottimizzazione e semplificazione di questi modelli.
Una distinzione fondazionale introdotta nel cap. II. 3, quella tra **competenza** ed **esecuzione**, è stata finora il nostro faro nello studio della linguistica: è da subito risultato chiaro che certi aspetti dovessero essere astratti e purificati dalle concrete limitazioni della nostra memoria di lavoro, attenzione e conoscenza del mondo. Resta però il fatto che gli unici dati empirici a nostra disposizione, quelli ad esempio recuperabili da un qualsiasi corpus di produzioni spontanee, siano in realtà il frutto della nostra esecuzione linguistica. Sin dagli anni Ottanta [Rumelhart *et al.* 1999], i modelli subsimbolici non si sono preoccupati di questa distinzione, suggerendo che un'adeguata struttura di processamento potesse essere in grado di simulare una performance umana basandosi su una rappresentazione approssimata della competenza implicita necessaria ad attuarla. Sebbene molti abbiano contestato a questo approccio connessionista una prospettiva negazionista rispetto all'idea innatista (necessaria per risolvere il **problema della povertà dello stimolo**), prospettive più concilianti non sono rare [Elman 1999] e

suggeriscono fermamente come i progressi fatti siano essenzialmente legati a precise intuizioni architetturali (ad esempio le reti ricorrenti e le loro evoluzioni LSTM).
In effetti, un altro modo per ricollocare la dicotomia competenza vs. esecuzione è quella di inquadrarla all'interno dei livelli descrittivi proposti da Marr [1982]: se la **competenza** corrisponde idealmente al **livello computazionale** (quello cioè che definisce il dominio di riferimento ed esprime quindi un requisito minimo di adeguatezza osservativa, come suggerito da Chomsky), l'**esecuzione** è ragionevolmente legata a un **livello algoritmico** (quello cioè che specifica, passo per passo, la procedura necessaria per trasformare un input in un output). Quanto sia possibile separare il livello di **adeguatezza descrittiva** da questo livello algoritmico rimane una questione puramente empirica: se questa separazione sembrava logicamente possibile negli anni Ottanta, i recenti approcci alla descrizione grammaticale si sono fatti man mano più «algoritmici» [Chomsky 1995]. Sebbene sia stato a più riprese sottolineato come la derivazione ipotizzata sia esclusivamente astratta, con nessuna implicazione per il tempo reale in cui nel nostro cervello le singole procedure vengono eseguite [Chomsky *et al.* 2023], viene da chiedersi se le architetture adottate per allenare i vLLM non abbiano qualcosa di intrinsecamente interessante da dirci su come le operazioni basilari di «costruzione della struttura» possano essere realmente implementabili.
Tornando per un momento alla metafora del «volo artificiale», se da un lato è innegabile che gli aerei non assomigliano molto agli uccelli reali valutandoli in base al fatto che non sanno stare in equilibrio su un filo teso, non fanno i nidi tra i rami degli alberi e non cercano di procurarsi il cibo da soli, risulta però evidente che questi «uccelli artificiali» riescono a raggiungere velocità anche superiori a quella del suono, altezze altrettanto proibitive e affrontano voli transoceanici con carichi notevoli che nessun essere vivente potrebbe mai gestire. Il caso dei very Large Language Models è forse simile in questo senso: nessun essere umano potrebbe mai riuscire a manipolare con l'efficienza di un vLLM tutta l'informazione contenuta su Wikipedia; ma riesce senza troppi problemi a produrre frasi grammaticali e ha intuizioni genuine su cosa sia andato storto in una frase come «*il gatto si scivola».
In conclusione, ci pare ragionevole affermare che, se da un lato lo studio della linguistica resterà fondamentale per comprendere cosa queste «intelligenze artificiali» sapranno e non sapranno (mai) fare, dall'altro la ricerca in ambito computazionale è stata in grado di trovare soluzioni decisamente efficienti (e probabilmente ancora migliorabili) che ci hanno consentito non solo di simulare una comprensione linguistica simile a quella umana ma, soprattutto, di archiviare in modo efficace un'incredibile quantità di informazioni linguistico-enciclopediche, che la nostra limitata mente umana, con il suo limitato *wetware* cerebrale, non potrebbe mai gestire da sola in modo così efficiente.

NOTA BIBLIOGRAFICA

Forniamo qui di seguito una serie di indicazioni bibliografiche utili per tutti i lettori che vogliano approfondire i temi trattati in questa appendice.

Abney [1987]; Baroni, Bernardi e Zamparelli [2014]; Bender *et al.* [2021]; Berger *et al.* [1994]; Brooks [1990]; Brown *et al.* [2020]; Brunato *et al.* [2023]; Charpentier e Samuel [2023]; Chesi [2024]; Chesi, Vespignani e Zamparelli [2023]; Chesi *et al.* [2024]; Chomsky [1965; 1981; 1995]; Chomsky *et al.* [2023]; Crain e Nakayama [1987]; Elman [1990]; Elman [1993]; Elman [1999]; Giorgi, Longobardi *et al.* [1991]; Hart e Risley [1992]; Hochreiter e Schmidhuber [1997]; Hsu e Chater [2010]; Hutchins [2004]; Hutchins e Somers [1997]; Ježek e Sprugnoli [2023]; Jones e Bergen [2024]; Kaplan *et al.* [2020]; Lenci, Montemagni e Pirrelli [2022]; Longobardi [1994]; MacWhinney [2000]; Marr [1982]; McCulloch e Pitts [1943]; Mei *et al.* [2024]; Minsky e Papert [1988]; Piantadosi [2023]; Ribeiro, Singh e Guestrin [2016]; Roberts [2019]; Ross [1967]; Rumelhart, McClelland & PDP Research Group [1999]; Shannon [1948]; Steuer, Mosbach e Klakow [2023]; Strubell, Ganesh e McCallum [2019]; Tamburini [2022]; Trotta *et al.* [2021]; Turing [1950]; Vaswani *et al.* [2017]; Warstadt *et al.* [2020]; Warstadt *et al.* [2023]; Warstadt, Singh e Bowman [2018]; Weizenbaum [1966]; Wilcox, Futrell e Levy [2023]; Zamparelli *et al.* [2022]; Zhang *et al.* [2020].

Riferimenti bibliografici