



UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria

Corso di laurea in: Ingegneria Biomedica

**Valutazione di ChatGPT e Gemini Large Language
Models per la generazione di modelli farmacocinetici
con SimBiology**

**Evaluation of ChatGPT and Gemini Large Language
Models for the generation of pharmacokinetics models
with SimBiology**

Relatore:

Dott. Micaela Morettini

Candidato:

Alessandro Morelli

Correlatori:

Dott. Andrea Tura

Dott. Agnese Piersanti

A.A 2023-2024

Abstract

I Large Language Models (LLMs) sono algoritmi di deep learning capaci di riconoscere, generare, riassumere, tradurre e prevedere contenuti utilizzando grandi set di dati. Tali algoritmi stanno apportando una forte spinta all'innovazione tecnologica in tutti i campi incluso quello dell'ingegneria biomedica. In questo campo, i modelli matematici rappresentano un potente strumento per descrivere il comportamento di sistemi biologici, tra cui quello metabolico. Nella modellazione del sistema metabolico, vengono spesso utilizzati i modelli farmacocinetici compartimentali basati su equazioni differenziali regolate dalla legge del bilancio di massa. L'obiettivo di questo lavoro di tesi è la valutazione di come gli strumenti di Intelligenza Artificiale (IA) Generativa basati su LLMs, quali ChatGPT e Gemini, possano supportare il processo di modellazione farmacocinetica. In particolare, si è valutato come tali strumenti siano in grado di generare istruzioni nell'ambito di SimBiology, strumento di MATLAB dedicato alla modellazione, simulazione e analisi di sistemi biologici.

I risultati ottenuti mostrano che ChatGPT ha offerto una maggiore accuratezza e flessibilità nella generazione di codici MATLAB rispetto a quella prodotta da Gemini, sebbene entrambi abbiano presentato errori, sia strutturali che sintattici. In conclusione, l'intelligenza artificiale generativa rappresenta una promettente tecnologia per accelerare lo sviluppo di modelli; tuttavia, la completa automazione non risulta ancora possibile e richiede la supervisione umana per correggere gli errori e migliorare l'affidabilità.

INDICE

INTRODUZIONE	1
CAPITOLO 1: Modelli compartimentali farmacocinetici per lo studio del metabolismo.....	4
1.1 Modelli compartimentali farmacocinetici.....	7
CAPITOLO 2: Tecniche di Deep Learning e Intelligenza Artificiale Generativa.....	10
2.1 Deep Learning.....	10
2.2 Intelligenza Artificiale Generativa e Large Language Models.....	12
2.2.1 Metodi di allenamento e valutazione.....	14
2.2.2 Sicurezza e regolamentazione.....	17
2.3 Applicazioni.....	19
CAPITOLO 3: Sviluppo di metodi basati sull'intelligenza artificiale generativa per la creazione di modelli farmacocinetici.....	23
DISCUSSIONE E CONCLUSIONE	36
BIBLIOGRAFIA	37

RINGRAZIAMENTI

INTRODUZIONE

Gli strumenti di Intelligenza Artificiale (IA) Generativa basati su Large Language Models (LLM) stanno apportando una forte spinta all'innovazione tecnologica nel campo dell'ingegneria biomedica.

I LLM sono algoritmi di deep learning capaci di riconoscere, generare, riassumere, tradurre e prevedere contenuti utilizzando grandi set di dati. La loro rilevanza attuale deriva dalle loro avanzate capacità che includono anche la scrittura di codice.

Nel campo dell'ingegneria biomedica, un aspetto tradizionalmente importante è quello della creazione di modelli matematici (in-silico) che descrivono il comportamento di sistemi biologici complessi. Nella modellazione del sistema metabolico, vengono spesso utilizzati i modelli farmacocinetici compartimentali basati su equazioni differenziali regolate dalla legge del bilancio di massa. La comprensione di tale sistema, in particolare quello inerente al metabolismo del glucosio (che fornisce la fonte di energia principale per l'organismo) rappresenta una tematica cruciale poiché disordini dello stesso sono legati a patologie estremamente comuni, quali il diabete.

Nel recente studio di Shin et al., si è valutato come strumenti di IA generativa, basati su LLMs, quali ChatGPT e Gemini, siano in grado di generare istruzioni per il processo di modellazione farmacocinetica. In tale studio, si era preso in considerazione NONMEM come software dedicato a studi farmacocinetici.

L'obiettivo di questo lavoro di tesi è la valutazione degli stessi strumenti di IA generativa, nell'ambito di SimBiology, software di MATLAB dedicato alla modellazione, simulazione e analisi di sistemi biologici.

CAPITOLO 1: Modelli compartimentali farmacocinetici per lo studio del metabolismo

La nutrizione di un organismo vivente fa riferimento a tutti i fenomeni che riguardano il metabolismo energetico e l'utilizzo dei nutrienti per il corretto funzionamento dell'organismo. In particolar modo il metabolismo è il complesso di tutte le reazioni organiche di trasformazione tra energia e materia e si divide in anabolismo e catabolismo. L'anabolismo è la sintesi di molecole complesse con consumo di energia, mentre il catabolismo è la degradazione di molecole complesse in molecole più semplici senza dispendio energetico [1]. L'intero processo di metabolismo può essere diviso in due fasi:

- Fase di assorbimento: in cui i nutrienti ingeriti entrano nel sangue attraverso il tratto gastrointestinale.
- Fase post-assorbimento, durante la quale il tratto gastrointestinale non contiene sostanze nutritive ma attinge energia dalle riserve del proprio organismo.

Tra queste due fasi esiste un legame dovuto al fatto che una parte dei nutrienti andrà a fornire energia nell'immediato, mentre la parte rimanente verrà "immagazzinata" come energia di riserva e poi utilizzata nella fase di post-assorbimento, garantendo all'organismo una tolleranza ad un digiuno prolungato, a patto che sia disponibile l'acqua.

Entrando nel dettaglio, nella fase di assorbimento viene fornita energia attraverso i nutrienti ingeriti, in particolare dai macronutrienti (proteine, lipidi, carboidrati). Alcuni dei principali carboidrati che vengono assorbiti sono il galattosio e il fruttosio, successivamente convertiti in glucosio dal fegato.

Il glucosio è la fonte principale di energia di un organismo e gran parte di esso nelle cellule durante la fase di assorbimento, per poi essere catabolizzato in anidride carbonica e acqua (reazione di ossidazione), liberando energia, che viene successivamente utilizzata per la formazione di molecole di adenosina trifosfato (ATP). Il destinatario principale del glucosio è il muscolo scheletrico in quanto costituisce circa l'80% della massa totale di un individuo, sia quando è in attività che quando è in condizione di riposo. Oltre a catabolizzare il glucosio nella fase di assorbimento, l'apparato muscolo-scheletrico immagazzina nelle cellule muscolari una parte di glucosio convertita in polisaccaride glicogeno [2].

Il glucosio viene assorbito anche dal tessuto adiposo (tessuto formato da cellule dette adipociti) che viene trasformato in trigliceridi durante la fase di assorbimento. Questi ultimi vengono immagazzinati come energia di riserva nella cellula. Un'altra grande parte di glucosio assorbito va negli epatociti

(fegato), dove si verifica un netto assorbimento di glucosio, immagazzinato poi sotto forma di glicogeno oppure trasformato in glicerolo-3-fosfato e acidi grassi. Si può affermare quindi che durante la fase di assorbimento, il glucosio è principalmente utilizzato per fornire energia.

Terminata la fase di assorbimento, la sintesi di glicogeno, trigliceridi e proteine cessa, dando inizio al catabolismo. Durante questa fase di produzione di energia, il glucosio contenuto nel glicogeno viene utilizzato quando cala la concentrazione ematica di glucosio, la quale, deve essere mantenuta in maniera omeostatica per evitare alterazioni dell'attività nervosa. Il glicogeno viene idrolizzato in glucosio attraverso il processo di glicogenolisi: attraverso cui viene fornito il glucosio per il mantenimento omeostatico della sua concentrazione plasmatica

La prima difesa dedicata al mantenimento della concentrazione ematica di glucosio è il processo di glicolisi. Oltre a questa, la concentrazione ematica del glucosio può essere mantenuta attraverso il catabolismo dei trigliceridi, processo definito di "lipolisi", dove il glicerolo e gli acidi grassi entrano nel sangue per diffusione e il glicerolo diretto al fegato viene utilizzato per la sintesi di glucosio.

Questo processo viene definito "gluconeogenesi", ovvero il processo metabolico endoergonico di sintesi biologica del glucosio a partire da sostanze diverse dai carboidrati e, la sua funzione si esplica principalmente nel fegato e in misura minore nel rene. I principali precursori sono l'acido lattico, gli aminoacidi e il glicerolo, derivati dal catabolismo del glucosio, delle proteine e dei lipidi. Lo scopo del processo di gluconeogenesi è quello di contribuire a mantenere costante la glicemia ed è fondamentale per garantire un adeguato apporto di glucosio ai tessuti insulinodipendenti (cervello, globuli rossi e muscoli) [1].

La gluconeogenesi produce circa 180g di glucosio al giorno, dal fegato e dai reni, fornendo un'energia di circa 720kcal su un fabbisogno giornaliero di un individuo adulto che è in media di 1500-3000kcal; pertanto, questa reazione non può dare all'organismo tutta l'energia necessaria per la durata della giornata. Per questo motivo durante la transizione di fase c'è un processo di adattamento che riduce il catabolismo del glucosio e aumenta l'uso dei lipidi facendoli diventare la prima fonte di energia (processo di lipolisi, fornitura glicerolo al fegato vista in precedenza) [2]. Questa diversificazione delle fonti di "carburante" permette un risparmio di glucosio nella fornitura giornaliera di energia, andandolo a sostituire con gli acidi grassi prodotti dalla lipolisi e i corpi chetonici provenienti dal fegato (rilasciati nel sangue durante un periodo di digiuno prolungato), per essere poi utilizzati dal cervello e dal sistema nervoso. Nei casi di una massiccia attivazione della gluconeogenesi per situazioni di eccessiva richiesta e consumo da parte dell'organismo, sia che queste siano dovute ad attività fisica prolungata che a denutrizione o digiuno, l'acido lattico e il glicerolo non sono sufficienti ad esaudire la richiesta di glucosio dell'organismo, per cui verranno attivati altri meccanismi di

“riserva” e, in particolare, aumenterà il catabolismo proteico per fornire gli amminoacidi glugenici richiesti. In altre parole, dopo aver esaurito le fonti e le riserve classiche di glucosio, l’organismo inizierà prima ad interrompere le funzioni fisiologiche non essenziali al mantenimento in vita, poi inizierà letteralmente a “mangiare sé stesso”.

1.3 Modelli compartimentali farmacocinetici

I modelli compartimentali sono utilizzati per comprendere fenomeni farmacologici che si verificano all'interno del sistema biologico. Il loro funzionamento si basa sulla suddivisione dei sistemi di interesse scientifico in unità concettuali più piccole fino a che i meccanismi sottostanti non diventano evidenti. Ad esempio, l'organismo a cui viene somministrato un farmaco è pensato come un sistema interconnesso di compartimenti con diverse proprietà e flussi di scambio di massa. In base alla loro costruzione, i modelli compartimentali si possono suddividere in due categorie: (1) modelli meccanicistici e (2) modelli fisiologici. Nel primo caso, i compartimenti non riflettono necessariamente oggetti funzionali dell'organismo, mentre nel secondo caso, ogni compartimento si riferisce ad una struttura ben definita ed interconnessa dell'organismo. L'obiettivo di un modello compartimentale è quello di riuscire a risolvere l'equazione (1), cioè, trovare una soluzione della matrice $A(t)$ in grado di risolvere l'equazione:

$$\mathbf{c} = \mathbf{A}(t)\mathbf{c} \quad (1)$$

- \mathbf{c} =vettore della quantità di massa.
- \mathbf{A} = matrice invertibile.

In cui il vettore \mathbf{c} rappresenta la quantità di massa nei vari compartimenti sotto la condizione che nessun elemento possa essere negativo e che la matrice $\mathbf{A}(t)$ sia invertibile. Se $\mathbf{A}(t)$ è indipendente dal tempo il modello è definito come un modello lineare. Tuttavia, in farmacologia i modelli più importanti sono non-lineari e con alcuni elementi della matrice che decrescono nel tempo.

Il concetto più importante che deve caratterizzare un modello compartimentale applicato alla farmacocinetica è quello della conoscenza del flusso sanguigno nell'organismo, spesso definito come compartimento 'centrale' e che può essere sottoposto a misure continue. I farmaci somministrati entrano nel compartimento centrale in base alla diversa modalità di azione del farmaco, ad esempio, una somministrazione costante, un flusso di prim'ordine esponenzialmente decrescente o un flusso più complicato attraverso come il ricircolo epatico.

Il modello farmacocinetico più semplice è composto da un singolo compartimento, generalmente il sistema circolatorio, in cui esiste una singola via di assorbimento ed eliminazione del farmaco. concentrazione del farmaco nei vari compartimenti in funzione del tempo, provvedendo alla quantificazione dei flussi di assorbimento ed eliminazione e delle costanti che li regolano [3].

Nel corso degli anni numerosi lavori di ricerca sono stati svolti sull'uso di questi modelli, per la loro applicazioni in ambito medico e biologico, in particolare in alcuni settori, come la

farmacocinetica, vengono usati per prevedere la distribuzione, l'assorbimento e l'eliminazione dei farmaci attraverso una ricostruzione modellistica dell'organismo. I modelli compartimentali non sono altro che dei modelli matematici basati su equazioni differenziali ordinarie (ODE) usate per descrivere le modalità in cui i materiali vengono scambiati tra i vari compartimenti e, in alcune applicazioni, si possono trovare equazioni differenziali frazionarie (FDE) che rappresentano un'estensione matematica di modelli ben definiti che sono basati su ODE. Un esempio di un modello compartimentale, composto da due compartimenti è mostrato in figura 1. Il modello si divide in due compartimenti: un compartimento centrale (A_1) in cui vengono rappresentati gli organi irrorati maggiormente dal sangue, come fegato e reni, e un compartimento secondario, (A_2), che va a rappresentare la parte periferica del corpo, non supportata in modo elevato dal sangue. Entrambi i compartimenti sono connessi da frecce orientate in entrambe le direzioni. Lo scambio di flussi tra i compartimenti viene regolato da questo tipo di equazione:

$$\frac{dA_1(t)}{dt} = -k_{10}A_1(t) - k_{12}A_1(t) + k_{21}A_2(t) \quad (2)$$

$$\frac{dA_2(t)}{dt} = k_{12}A_1(t) - k_{21}A_2(t) \quad (3)$$

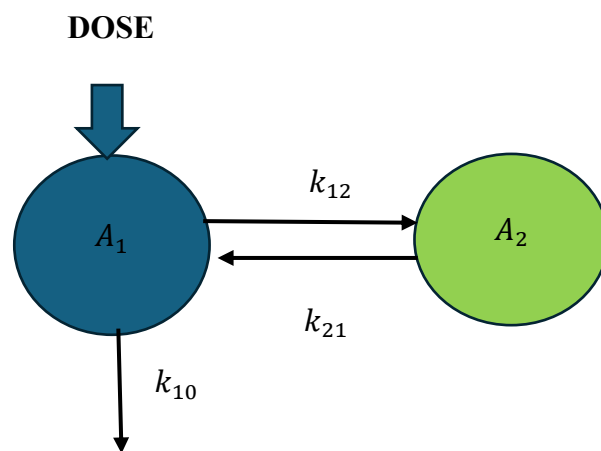


Figura 1: Esempio di modello compartimentale farmacocinetico a due compartimenti.

in cui A_1 e A_2 rappresentano la quantità del farmaco nei rispettivi compartimenti, mentre le costanti (k_{10} , k_{12} , k_{21}) regolano lo scambio di massa tra i compartimenti e il flusso di eliminazione dal primo compartimento.[4]

Si può anche andare ad integrare i modelli compartimentali con delle tecnologiche più avanzate, come il deep learning ed in particolare il Deep Compartment Model (DCM), in cui una rete neurale viene usata per prevedere i parametri latenti di un modello rappresentato da ODE. Questo tipo di

implementazione permette di automatizzare la modellizzazione farmacocinetica più complessa, combinando i punti di forza del deep learning con la farmacocinetica tradizionale, andando a migliorarne le previsioni e l'applicabilità in scenari reali [5].

CAPITOLO 2: Tecniche di Deep Learning e Intelligenza Artificiale Generativa

L'impatto dell'Intelligenza Artificiale (IA) nell'assistenza sanitaria ha migliorato l'ambito clinico e operativo. Una sfida per i sistemi IA è la loro integrazione con i flussi di lavoro del settore sanitario che includono I primi modelli medici digitali erano volti alla digitalizzazione delle cartelle dei pazienti (EHR), dopodiché le informazioni sulla salute online hanno iniziato ad essere scambiate grazie all'espansione di internet e, come risultato, l'applicazione di esse ha consentito alle persone di monitorare il proprio stato di salute, l'esercizio fisico e monitoraggio di malattie croniche. In futuro, tecnologie come l'intelligenza artificiale generativa basata su 'Large Language Models '(LLMs) potrebbe avere un ulteriore impatto nel settore sanitario [6]

2.1 Deep learning

Il deep learning (DL) permette ad un sistema di apprendere delle rappresentazioni complesse in modo diretto, ovvero dai dati grezzi ('representation learning'), grazie all'uso di modelli non lineari a più livelli che elaborano i dati in modo sempre più astratto. L'aspetto chiave del DL è che le caratteristiche dei dati (features), vengono autonomamente imparati tramite una procedura di apprendimento che rende il DL particolarmente adatto ed efficace per analisi di dati ad alta dimensione, trovando applicazioni nei più svariati campi e mostrando una maggiore accuratezza rispetto ai metodi di machine learning tradizionali. [7] L'architettura più diffusa è quella delle deep neural networks (DNNs), che si basa su dei livelli gerarchici di rappresentazione, riuscendo a cogliere relazioni tra i dati non strutturati e, rilevando dei pattern nascosti tra essi, migliorando la precisione delle previsioni attraverso la retro-propagazione dell'errore per ottimizzare i parametri. Il loro processo di addestramento avviene utilizzando delle tecniche come quella del Gradiente Discendente Stocastico (SGD), che va a minimizzare l'errore su un insieme di dati di allenamento attraverso un processo regressivo.

Tecniche di DL vengono applicate con successo anche nel campo della medicina di precisione, grazie alla enorme quantità di dati (big data) esistenti [8].

Le principali architetture di DL includono:

- *Convolutional Neural Network (CNN)*: ideali per analisi di immagini, queste reti utilizzano convolutional layers, pooling layers e fully connected layers, per estrarre feature significative e ridurre la dimensionalità delle immagini.
- *Recurrent Neural Network (RNN)*: utilizzate per riconoscere dei pattern di flusso di dati come testi o segni vocali, memorizzando informazioni sulle sequenze passate per fare previsioni sui dati futuri. Le varianti Long Short Term Memory (LSTM) e Gated Recurrent Units (GRU) risolvono il problema del gradiente scomparso durante il processo di training, migliorando le prestazioni per sequenze lunghe.
- *Deep Belief Networks (DBNs)*: usano combinazioni di reti neurali e macchine di Boltzmann per apprendere vaste varietà di dati ad alta dimensione
- *Auto-encoder (AE)*: sono delle reti neurali non supervisionate che cercano di comprimere i dati in rappresentazioni semplificate e riducono il rumore [9].

I recenti sviluppi nel campo del DL hanno permesso a questi modelli di acquisire un livello importante di accuratezza, permettendo così l'applicazione nel settore sanitario per l'identificazione di farmaci, analisi dati genomici e medici, prevedere effetti collaterali e ottimizzare i trattamenti specifici per paziente. Nel campo della chirurgia l'uso del DL e del reinforcement learning possiede il potenziale per supportare robot chirurgici in operazioni automatizzate, anche se al momento il completo affidamento ai robot per interventi chirurgici complessi rimane una sfida a causa delle numerose variabili umane e mediche coinvolte nel processo. Un esempio attuale di un modello DL è ChatGPT-4, il quale evidenzia la capacità di interazione nei linguaggi naturali e nell' image processing, aprendo la strada a nuove applicazioni nella medicina personalizzata. Inoltre, data la vasta quantità di dati forniti nell'addestramento di questo modello, le sue applicazioni risultano diverse e variano dal campo medico, come appena citato, a diversi campi di applicazione come quello informatico per lo sviluppo di software, nell'ambito finanziario per l'analisi e revisione dati, nel settore della formazione fornendo supporto a studenti e molte altre applicazioni. In generale, ChatGPT, diventerà nel prossimo parte integrante nella quotidianità lavorativa e non data la sua estrema versatilità e il suo continuo potenziamento e sviluppo [10].

2.2 Intelligenza Artificiale Generativa e Large Language Models (LLM)

Tra le tecnologie IA in rapida evoluzione, i modelli di IA Generativa come il Generative Pre-Trained Trasformer (GPT) sviluppati da OpenAI risultano essere degli strumenti di grande potenziale. Il Natural Language Process (NLP).ad esempio è un sottocampo fondamentale dell'IA che si focalizza sull'interazione tra il linguaggio umano e i computers. L'obiettivo primario è quello di consentire alle macchine di comprendere, interpretare e generare linguaggio umano in modo significativo e contestualmente rilevante; comprende un ampio campo di applicazioni, tra cui la transizione linguistica, riconoscimento vocale, riepilogo del testo, risposta alle domande e altro ancora, ciascuno progettato per colmare il divario tra la comunicazione umana e la comprensione computazionale. Nel campo del NLP, i modelli GPT sono emersi come degli sviluppi degni di nota e costituiscono una famiglia di modelli linguistici sofisticati progettati per sfruttare il potenziale delle reti neurali del deep learning. Questi modelli sono in grado di predire parole o frasi successive ad un contesto precedentemente fornito e possono generare anche nuovi contenuti (foto, video, musica...). Alla base del funzionamento dei GPT vi è un trasformatore la cui architettura eccelle nel processare dati sequenziali, specialmente dati di testo, usando la "self-attention", che consente al modello di comprendere l'importanza e la relazione tra parole individuali inserite in una frase durante l'elaborazione, consentendogli di fornire maggiore attenzione alle parole correlate semanticamente e trasportare un numero maggiore di informazioni inerenti al contesto. L'aspetto rivoluzionario dei trasformatori risiede nella loro capacità di gestire in modo efficiente le dipendenze a lungo raggio all'interno delle sequenze linguistiche migliorando la loro competenza nella comprensione linguistica e semantica. L'evoluzione e sviluppo degli GPT ha subito un rapido progresso in un periodo conciso di tempo, il primo modello è stato introdotto nel 2018, GPT-1, dimostrando il potenziale del pre-training su larga scala; il secondo è stato rilasciato da OpenAI nel 2019, GPT-2, presentando un significativo passo in avanti con la cifra di 1.5 miliardi di parametri, ma la sua introduzione nel mercato ha destato preoccupazione circa il suo possibile uso improprio e la sua abilità nel generare testi simili a quelli umani; nel 2020, è stato rilasciato il modello GPT-3, un importante avanzamento consistente di circa 175 miliardi di parametri, riconoscendolo come il più grande Language model creato, ha dimostrato di avere capacità di generare linguaggio senza precedenti, incluse le abilità di traslazione, riassunto, domanda-risposta e scrittura creativa; l'evoluzione del GPT-3 è il modello GPT-4 allenato su un trilione di parametri e un modello

intermedio, GPT-3.5, cioè uno specifico modello allenato per predire la parola successiva in una sequenza usando un enorme dataset di testo, capace di ingaggiare conversazioni naturali e dinamiche.

Questi modelli di IA Generativa sono dei sottoinsiemi dei Large Language Models (LLMs), modelli che sono pre-allenati da un enorme quantità di dati e spesso sono riferiti come “foundation model” o “base model” di IA generativa essendo il punto iniziale per lo sviluppo di più complessi e avanzati sistemi. La natura “generativa” intrinseca di questi modelli garantisce loro la straordinaria capacità di produrre autonomamente testi simili a quelli di mano umana, implicando un applicazioni in una vastata gamma di settori e lo step del pre-training è fondamentale per creare i modelli GPT. Durante questa prima fase, il modello viene esposto ad un enorme quantità di diversi testi e dataset consentendogli di acquisire una robusta comprensione dei pattern e strutture del linguaggio, la fusione poi tra il preallenamento e l’architettura del trasformatore equipaggia i modelli di GPT di abilità eccelse in vari compiti di NLP facilitando la generazione di testo con elevata fluidità e coerenza contestuale, il più possibile simile a quella umana. Dopo la fase di acquisizione delle conoscenze, il prossimo step è la messa a punto del suo comprendimento (fine-tuning), cioè si utilizzano degli specifici dati etichettati in base alla loro attività, come i dati nel settore sanitario, adattando il modello all’esecuzione di specifiche attività del NLP rilevanti l’assistenza medica, come la risposta a domande di carattere medico, classificazione testi clinici e la traduzione di cartelle mediche. La confluenza delle capacità di comprensione del linguaggio dei GPT, unita al pre-training e fine-tuning, consente di eccellere in un’ampia gamma di attività di NLP sanitarie, data la conoscenza elevata del linguaggio medico acquisito dal modello, portando a un miglioramento delle prestazioni del modello e riducendo tempo e risorse per l’addestramento completo del modello. Grazie alla generazione di testo simile a quello umano e la conoscenza specifica in compiti specifici sanitari, i LLMs hanno dimostrato un enorme potenziale nello sviluppare soluzioni sanitarie basate sull’IA e l’integrazione di quest’ultima è promettente per migliorare il supporto alle decisioni cliniche, miglior comunicazione con i pazienti, cura del paziente, diagnosi delle malattie più veloce e assistenza ai medici durante il processo decisionale [11].

2.2.1 Metodi di allenamento e valutazione

I modelli di IA Generativa e LLMs sono alimentati da tecnologie di deep learning. Una serie di modelli di DL utilizza l'architettura a trasformatore ricorrendo a meccanismi di self-attention per processare il vasto dataset di linguaggio umano. Esistono tre fattori chiave per la scelta di un LLMs allo scopo di risolvere problemi pratici ovvero modelli, data e downstream task.

In base alla strategia di training, architettura e casi usati, i LLMs sono classificati approssimativamente in due tipi: il primo è l'encoder-decoder", il secondo è il decoder. I primi sono rappresentati dai modelli BERT (bidirectional encoder representation from transformers), i più promettenti perché la maggior parte è open-source, garantendo opportunità di continuo sviluppo ed esplorazione; i secondi, sono rappresentati dai GPT, come Pathways Language Models (PaLM), introdotta da Google e LLaMA da Meta, rappresentano la base dei modelli generativi di IA.

Inoltre, in base al dataset scelto per il training, i LLMs sono classificati in foundations LLM e instruction fine-tuned LLMs, con i primi che sono allenati a predire la parola successiva più probabile basandosi su un set di dati di allenamento, però la direzione dell'output può essere imprevedibile; i secondi invece non sono altro che modelli base LLMs messi a punto attraverso varie tecniche, incluso il Reinforcement Learning from Human Feedback (RLHF). La qualità, la quantità e la diversità dei dati influenzano in modo significativo le prestazioni dei LLMs.

I LLMs possono essere applicati a quattro tipi di compiti: comprensione del linguaggio naturale, compiti ad alta intensità di conoscenza, ragionamento e generazione di linguaggio. I compiti tradizionali di comprensione del linguaggio naturale includono la classificazione del testo, estrazione di concetti o named entity recognition (NER), estrazione di relazioni e previsione delle implicazioni, di cui molte di questi compiti sono fasi intermedie in estesi sistemi AI come il NER per la costruzione di conoscenza grafica. La generazione di linguaggio naturale comprende due tipi principali di compiti: (1) conversione di testi input in una nuova sequenza di simboli (sintesi e traduzione automatica) e (2) generazione "open-ended" che mira a generare nuovo testo o simboli in risposta alla richiesta di input (risposta-domanda, creazione e-mail...). I task NLP ad alta intensità di conoscenza si riferiscono a compiti che richiedono una quantità sostanziale di conoscenze di base, sia che siano dominio-specifiche che generali o acquisite nel tempo; questo tipo di task richiedono non solo un riconoscimento di pattern o analisi sintattica, ma dipendono anche dalla memorizzazione e dall'utilizzo corretto di conoscenze specifiche dopo l'esposizione a un miliardo di token, i LLMs sono eccellenti per soddisfare questo tipo di attività ma se non hanno tutte le conoscenze necessarie potrebbero produrre "allucinazioni".

Per quanto riguarda l'addestramento di questi modelli possono essere applicate diverse strategie, come per il caso dei modelli "fine-tuned" in cui si può agire sul numero di parametri, la grandezza dei dataset di training o la quantità di computazione usata per il training. I modelli messi appunto ingrandiranno i LLMs di base e miglioreranno le performance dei modelli pre-allenati nel ragionamento per sbloccare abilità riferite a competenze specifiche che non sono presenti nel modello di base ma che diventano salienti con la scalabilità del modello, queste includono la comprensione di concetti sfumati, manipolazione di parole complesse, ragionamento logico avanzato e compiti di codifica complessi (es. modello PaLM scalato da 8 miliardi di parametri a 540 miliardi, con accuratezza del 67.2% nelle risposte del USMLE). Questo processo di scalabilità ha portato a prestazioni che si avvicinano a quelle umane, sia per il ragionamento aritmetico che quello linguistico, di fondamentale importanza nell'ambito medico e per l'automazione di processi sanitari essenziali.

Un altro metodo può essere quello del Reinforcement Learning from Human Feedback (RLHF, precedentemente citata), una tecnica semplice ed efficiente dal punto di vista dei dati e parametri per migliorare le capacità generali e allineare i LLMs al settore sanitario e medico critico per la sicurezza. Sono dei metodi che combinano tre processi di training del modello: raccolta feedback, modellazione del reward e ottimizzazione delle politiche; questo metodo non migliora solo l'accuratezza ma riduce i pregiudizi e migliora la coerenza e sicurezza delle risposte in ambito medico, simili a quelle umane.

Infine, l'ingegneria degli stimoli (prompt engineering) consente di raffinare gli stimoli per generare testi o immagini utilizzando cinque diverse istruzioni: zero-shot, few-shot, chain-of-thought, self-consistency ed ensemble refinement learning. La prima consente nel training dei LLMs per attività specifiche di NLP attraverso singole istruzioni eliminando la necessità di dati annotati; la seconda addestra i LLMs su compiti NLP specifici con un numero limitato di esempi; la terza imita il processo umano di ragionamento nella risoluzione di problemi attraverso pochi esempi presenti nel prompt con una serie di frasi di ragionamento articolate che portano alla risposta finale; la quarta campiona un insieme diversificato di percorsi di ragionamento invece di prendere quello più "avido", selezionando la risposta più coerente; la quinta prevede due fasi, nella prima sono fornite un insieme di domande e risposte, mentre nella seconda il modello viene raffinato sulla base dell'istruzione originale e alle risposte aggregate dalla prima fase (combinazione di chain-of-thought e self-refining).

Dopo queste fasi di sviluppo e miglioramento si passa alla valutazione dei modelli, in particolare si discute di tre parametri che ostacolano l'uso dei LLMs in compiti reali, (1) rumore di dati in input non strutturati, (2) compiti definiti male e difficile da classificare nelle categorie di compiti dei NLP

predefiniti, (3) istruzioni ambigue che possono contenere molteplici intenti. Nonostante la migliori performance nei modelli ottimizzati nel trattare questo tipo di problematiche, la gestione in input di dati reali dei modelli di base deve ancora essere valutata [12].

2.2.2 Sicurezza e Regolamentazione

L'applicazione e l'uso dei LLMs in vasta gamma non è esente da problematiche legate alla loro sicurezza, dato che vengono sottoposti durante il periodo di addestramento ad una vasta quantità di dati online sono suscettibili a manomissione esterne capaci di creare rischi per la privacy e sicurezza dei modelli. Ricercatori hanno dimostrato la facilità con cui gli "aggressori" riescono ad iniettare informazioni false e/o fuorvianti nei dataset del training, cosa che può verificarsi anche quando viene alterata solo una piccola frazione (circa 0.1%) delle istanze. Per aumentare la difesa dei modelli si possono prendere in considerazione diverse strategie come, ad esempio, la convalida e preelaborazione dei dataset per il training, usare algoritmi di rilevamento degli outlier per eliminare le informazioni sospette e l'impiego di programmi di training che introducono esempi spuri per allenare i modelli a distinguere tra gli input autentici e quelli fuorvianti. Un secondo problema riguardante la sicurezza è la privacy dei dati, specialmente per i dati sulle informazioni mediche personali e anche in assenza di intenti dannosi la privacy dell'utente può essere compromessa, per ridurre tali rischi i dati di training devono essere opportunamente curati, provenire da fonti affidabili e de duplicati per limitarne la ripetuta inclusione di informazioni personali [13].

I modelli LLMs sono stati rilasciati a livello globale e non specifici per paese, cioè questo richiede una regolamentazione globale e, inoltre, dato l'enorme polivalenza di questi modelli non è chiaro in quale categoria tecnica rientreranno da un punto di vista normativo; tuttavia, in base alle differenze tra i vari LLMs e precedenti metodi di DL, potrebbe essere necessaria una nuova categoria per affrontare le sfide e i rischi specifici dei vari modelli. Un organismo di regolamentazione deve solo elaborare regolamenti per gli LLMs se gli sviluppatori lo dichiarano utilizzabile per scopi medici; o se i LLMs sono sviluppati, adattati, modificati e riferiti a scopi specificatamente medici. Anche se gli attuali LLMs non rientrano in nessuna delle due categorie, le alternative dei modelli allenati specificatamente su dati medici probabilmente entreranno in queste due. Un esempio è quello di Med-PaLM in cui è stato proposto un quadro per la valutazione umana delle risposte del modello lungo molteplici assi che includono la fattualità, precisione, pericolosità e pregiudizio; inoltre utilizzando una combinazione di strategie di suggerimento si è riuscito a migliorare l'accuratezza del modello osservando che la valutazione umana rileva lacune fondamentali nelle risposte date. Visto però il continuo sviluppo e avanzamento, come il modello GPT-4, l'idea di regolamentare modelli attuali non è sufficiente visto che non possiamo considerarli come nuove interazioni rispetto a quelle avanzate con funzionalità implementate; quindi, una regolazione di questi modelli potrebbe perdere aggiornamenti importanti nel momento in cui essi diventano ampiamente accessibili. Alcune delle sfide per regolare l'uso dei LLMs potrebbero essere:

- *Privacy dati paziente*: assicurare che i dati usati per addestrare i LLMs sono completamente anonimi e protetti da potenziali falle, visto che ogni violazione può condurre a importanti conseguenze rispetto le leggi sulla privacy.
- *Proprietà intellettuale*: se un modello genera contenuti simili a quelli dei proprietari di una ricerca medica può creare problemi di diritto intellettuale.
- *Negligenza medica*: determinare chi è responsabile quando una raccomandazione data da un'IA mette a rischio il paziente.
- *Controllo qualità e standardizzazione*: sono richieste delle regolazioni per assicurare la replicabilità e consistenza dei consigli medici dati da GenIA.
- *Consenso informativo*: i pazienti necessitano di essere informati e dare consenso per usare strumenti IA.
- *Interpretabilità e trasparenza*: garantire la trasparenza su come le decisioni vengono prese dall'IA, può risultare difficile a causa delle “black box” dei loro algoritmi complessi.
- *Correttezza e pregiudizio*: prevenire eventuali pregiudizi nei modelli che possono essere introdotti durante il periodo di addestramento.
- *Monitoraggio continuo e Validazione*: assicurare la continuità della performance, accuratezza e validità di strumenti IA lungo un periodo di tempo e su diverse popolazioni [13].

2.3 Applicazioni

Uno studio condotto sul dataset “Pima Indians Diabetes” ha esplorato l’uso della GenAI per migliorare l’accuratezza e l’interpretabilità dei modelli diagnostici basati su regole per pazienti con diabete di tipo 2, rispetto ad altri modelli di ML già presenti che spesso mancano di interpretabilità. Analizzando il dataset citato, è stato ampliato il framework Numeric Association Rule Mining (NiaARM) per il mining delle regole associative, mentre il LightGBM usato per comparare le diverse performance decisionali dei GPT; per l’interpretabilità locale dei modelli è stato utilizzato lo Shapley Additive Explanations (SHAP) per capire come vengono prese le decisioni enfatizzando le decisioni su livello individuale, inoltre è stato anche impiegato GPT-3.5 per la previsione e interpretazione dei risultati. (codice sorgente: https://github.com/lkopitar/GenAI_NiaARM/). I risultati prodotti dallo studio hanno generato circa 350 regole dal framework NiaARM per prevedere il diabete e LightGBM ha dimostrato di avere una miglior performance rispetto al GPT association rules- based; il confronto tra le regole generate dal GPT e il NiaARM ha rilevato una sostanziale variazione fra queste e l’analisi della loro comparazione ha comunque portato ad un punteggio di similarità fra le due; infine, anche se il modello GPT ha prodotto una quantità superiore di regole per la determinazione del diabete, la decisione sulla miglior performance del LightGBM è stata presa principalmente su feature quali glucosio, età e BMI , cioè le caratteristiche più importanti e influenti per lo sviluppo della malattia. In conclusione, la combinazione del mining delle regole associative dei modelli basati sul GPT per la classificazione dei pazienti con diabete di tipo 2 ha mostrato un’efficacia limitata mostrando il bisogno di miglioramenti come la preelaborazione e ottimizzazione dei parametri. Le sfide di interpretabilità e dipendenza del GPT dalle regole forniscono la necessità di migliori tecniche nell’ingegneria dei prompt e metodi accurati per un punteggio di similarità, inoltre le differenze in classifica di importanza delle feature sottolineano la complessità della malattia [14].

L’uso della GenAI nel contesto gestionale del diabete può essere dedicato alla creazione di dati sintetici dei pazienti, cioè dei dati cruciali per la ricerca dato che permettono di produrre delle simulazioni su vasta gamma di scenari clinici senza andare a minacciare la privacy del paziente eseguendo dei test in ambienti controllati. Gli strumenti di GenAI sono in grado di analizzare grandi volumi di dati medici per andare ad assistere il processo decisionale degli operatori medici ed individuare i trattamenti più efficaci per ogni paziente basandosi su dati pregressi e attuali. Recentemente è stato esaminato l’uso di modelli GPT per ricreare un sistema di messaggi automatici personalizzato (PAMS) il quale è stato sviluppato come un’iniziativa di ricerca digitale a supporto di pazienti connessi con un programma di prevenzione digitale commerciale del diabete (dDPP); le componenti principali dell’intervento del PAMS riguardano una libreria di messagistica per il

cambiamento del comportamento basato sulla teoria, una piattaforma di messaggi automatici personalizzati inviati e dati integrati nelle visualizzazioni dell'EHR. La libreria del PAMS usa un framework integrato che combina modelli teorici convalidati per il cambio di comportamento con delle strategie di progettazione incentrate sull'essere umano per massimizzare le condizioni basate sull'evidenza del cambiamento di comportamento, l'accettazione e utilizzo dell'utente di un prodotto sanitario digitale. L'esperimento svolto si basa sulla ricreazione del PAMS utilizzando un sistema di GenAI(ChatGPT), valutarne l'efficienza del suo utilizzo e confrontare la documentazione generata rispettivamente dagli esseri umani rispetto a quella derivata dal GPT; tramite una serie di prompts forniti a ChatGPT-4 per assistere con la generazione della teoria, contenuti, storico utenti, documenti richiesti, diagrammi di progettazione e un subset di requisiti; gli output del modello sono stati revisionati e confrontati con la documentazione prodotta da 11 membri del team di valutazione (medici, programmatori, ricercatori nella salute digitale). Questi ultimi hanno valutato in modo indipendente la qualità di vari aspetti delle informazioni fornite da ChatGPT su una scala Likert dove i punteggi più alti indicano maggiore qualità e la valutazione ha incluso diversi aspetti: comprensibilità, novità, usabilità, creazione e potenziale pregiudizio. I risultati hanno mostrato che l'uso della GenAI ha migliorato la comunicazione e la comprensione all'interno del gruppo multidisciplinare fornendo descrizioni dettagliate delle funzionalità e supportando il lavoro degli ingegneri del software. Gli output generati da ChatGPT sono stati complessivamente accurati nel catturare i requisiti previsti anche se con alcune ambiguità che hanno richiesto chiarimenti o modifiche da parte del team ed oltre a questo il GPT ha facilitato il processo di sviluppo efficientemente producendo in 5 giorni quello che usualmente richiede più di 200 ore di lavoro umano, quindi il suo uso combinato con l'esperienza umana può ridurre significativamente il tempo impiegato nelle fasi iniziali di modellazione e concettualizzazione del sistema. Oltre a questi vantaggi sono state evidenziate alcune limitazioni significative degli output di ChatGPT che, sebbene fosse in grado di fornire orientamenti generali sulla progettazione degli strumenti non è stato in grado di fornire prove a sostegno delle scelte da esso effettuate, cioè mancanza di interpretabilità molto importante per interventi tecnologici sanitari basati su prove. Per quanto riguarda la generazione del codice all'inizio ha mostrato difficoltà nel produrre una funzione specifica per calcolare la tendenza di coinvolgimento dei pazienti su un periodo di tre settimane, ma dopo diverse iterazioni è stato possibile ottenere il codice desiderato anche se con costrutti non idiomatici e bug. In conclusione, il modello di GenAI ha delle ottime potenzialità ma allo stesso tempo delle limitazioni che richiedono il continuo supporto umano per interpretare i risultati generati e migliorare la qualità dei prompt forniti che devono essere ben strutturati (ruolo + descrizione problema +richiesta) [15].

Un' ulteriore esempio di applicazione è uno studio svolto sull'uso di modelli generativi per la generazione di dati sintetici applicati alla farmacocinetica /farmacodinamica (PK/PD) con l'obiettivo di esplorare soluzioni avanzate per affrontare sfide legate alla condivisione e all'accesso dei dati clinici, difficoltà dovute alle restrizioni di accesso ad esse dovute alla privacy e problemi etici. Inoltre, le caratteristiche di alcuni gruppi di pazienti sono raramente sufficienti per una modellazione accurata e per questo la generazione di dati sintetici può fungere da soluzione a questi problemi consentendo la simulazione di dati che rispecchiano caratteristiche di dati reali. Il main focus della ricerca si concentra sulla valutazione di tecniche DL per generare dati sintetici, considerando sia covariate che endpoint clinici per la PK/PD, confrontando vari modelli generativi prestando maggior attenzione al Multi-layer Perceptron Conditioning Generative Adversarial Neural Network (MLP cGAN), il Time-series Generative Adversarial Network (TimeGAN) e approcci più tradizionali come la regressione probabilistica. Questi modelli sono testati su dataset clinici contenenti covariate e variabili legati a PK/PD per determinare la loro capacità di produrre dati sintetici il più possibile vicini a quelli reali per poi essere utilizzati nelle analisi farmaco metriche. I metodi valutati nello studio includono: (1) MLP cGAN, ovvero un modello avanzato di rete neurale generativa in cui viene valutata la qualità dei dati sintetici generati e ne migliora gradualmente la precisione ed è condizionato da covariate cliniche per garantire che i dati sintetici prodotti siano realistici; (2) TimeGAN, modello progettato per generare dati temporali, particolarmente utile per dataset PK/PD che coinvolgono la misurazione di variabili in differenti momenti, combinando la capacità predittiva dei GAN con modelli sequenziali per generare dati sintetici in forma di serie temporale; Modelli di Regressione Probabilistica, usati come confronto fornendo un approccio più convenzionale alla generazione di dati, ma risultano limitati nella capacità di comprendere complessità e dipendenze non lineari presenti nei dati clinici reali. L'analisi ha mostrato che modelli come MLPcGAN e TimeGAN hanno di gran lunga superato i metodi tradizionali nella generazione di dati, ed in particolare, i modelli condizionali GAN si sono distinti per la capacità di generare dati che replicano in modo accurato le relazioni tra le covariate e gli endpoint PK/PD, permettendo una migliore affidabilità dei dati prodotti e di conseguenza delle simulazioni più precise delle risposte. Tuttavia, i modelli GAN presentano sfide, come l'allenamento, che richiede un'enorme quantità di dati per essere efficace e il processo di ottimizzazione può essere complicato con rischi di instabilità nella generazione dati ed inoltre non sostituiscono completamente la necessità di validare i risultati con dati clinici originali. Infine, la loro integrazione per la generazione dei dati ha grandi implicazioni per PK/PD, cioè, possono essere utilizzati per previsioni PK/PD, ottimizzare degli specifici trattamenti e espandere i dataset per la ricerca.[16]

Un altro studio sull'utilizzo di tecnologie generative è stato condotto per valutare le prestazioni e l'efficacia di modelli GPT, quali ChatGPT4.0 e Gemini Ultra 1.0, nella generazione di codice

utilizzando il linguaggio NONMEM per applicazioni in farmacologia (revisione effettuata da due esperti in programmazione).NONMEM è uno strumento fondamentale per la modellazione farmacocinetica e farmacodinamica (PK e PD), ma la scrittura del codice in questo linguaggio può essere soggetta a errori, soprattutto su utenti non esperti. Lo studio ha come focus principale quello di determinare se questi due strumenti generativi siano in grado o meno di assistere i ricercatori e programmatori nella creazione di codice NONMEM corretto e funzionante, così da andare a diminuire il tempo necessario per lo sviluppo e di conseguenza anche gli errori umani. Nella parte pratica sono stati fornite ai due GPT degli specifici prompt per generare dei codici NONMEM in diversi scenari, per poi analizzare il codice in output cercando eventuali errori, sia sintattici che strutturali; infine viene testata la capacità effettiva di generazione di output coerenti alla richiesta in input. I ricercatori hanno suddiviso lo studio in quattro casi, inserendo rispettivamente gli stessi prompt nei due sistemi IA caso per caso, partendo da un'introduzione a NONMEM per poi arrivare ad una vera e propria programmazione utilizzando questo linguaggio. Come risultati principali hanno ottenuto che entrambi i modelli sono stati in grado di produrre codice NONMEM di base mostrando ottime capacità di comprensione della richiesta, ma allo stesso tempo hanno presentato degli errori suddivisibili in due categorie. I primi, gli errori strutturali, legati alla mancanza di vere e proprie sezioni essenziali di codice oppure disposte in ordine errato, mentre per il secondo gruppo di errori, cioè quello degli errori sintattici, sono stati riscontrati dalla sintassi specifica di NONMEM, come l'uso incorretto di comandi o mancanza di dichiarazioni necessarie. In conclusione, i due modelli hanno dimostrato di essere utili nella generazione di codici pur presentando limiti come insufficiente affidabilità dato l'elevato numero di errori e la necessità di competenza umana per intervenire sul codice ed apportare le dovute correzioni, inoltre la diversa variabilità di risposta può creare incertezza. Lo studio infine riporta che l'integrazione di tecnologie avanzate come ChatGPT e Gemini nel campo della farmacocinetica e farmacodinamica, rappresenta un enorme passo in avanti nell'automazione dei processi scientifici, pur se l'affidabilità dei codici attualmente prodotti non è sufficiente per un'adozione completa senza supervisione, necessitando di una stretta collaborazione tra esperti del settore farmacologico e sviluppatori dei modelli [17]

CAPITOLO 3: Sviluppo di metodi basati sull'intelligenza artificiale generativa per la creazione di modelli farmacocinetici

Il tool SimBiology presente su MATLAB è in grado di fornire applicazioni e strumenti programmatici per modellare, simulare e analizzare sistemi dinamici. Tale strumento è in grado di creare modelli interattivi utilizzando un editor a blocchi, o programmando con il linguaggio MATLAB oppure utilizzando quelli già presenti come esempi. Il pacchetto offre una varietà di tecniche per analizzare i modelli basati su ODE con diversi gradi di complessità e dimensioni: si possono eseguire simulazione per valutare le fattibilità del target, prevedere l'efficacia e la sicurezza di un farmaco e individuare programmi di dosaggio ottimali; è possibile individuare dei percorsi chiave servendosi di analisi di sensibilità globale e locale valutando la variabilità biologica attraverso lo scambio dei parametri ed inoltre, la loro stima, può essere effettuata con il fitting dei dati usando la regressione non lineare e tecniche a effetti misti non lineari, oltre a analisi non compartimentali.

L'obiettivo dello studio è quello di sviluppare dei codici in linguaggio MATLAB inerenti a SimBiology tramite LLMs presenti conosciuti e che hanno già dimostrato le loro capacità nella generazione di codici in diversi linguaggi di programmazione. In particolare sono stati scelti LLMs gratuiti accessibili da Google, come ChatGPT V4.0 (<https://chat.openai.com/>) e Gemini Google (<https://gemini.google.com/>). La metodologia utilizzata è suddivisa in diverse fasi in cui viene dato un comando per entrambi i modelli. Successivamente sono stati confrontati e valutati gli output generati.

Per l'impostazione e le metodologie utilizzate sono state adottate quelle utilizzate da Shin, Euibeom, et al. [17] apportando delle modifiche, in particolare sul tipo di strumento utilizzato per la simulazione di modelli farmacocinetici per adattarlo alle specifiche esigenze di ricerca.

Di seguito, sono riportati i comandi dati in input e i relativi risultati.

- **CASO 1:** Come primo prompt in input inerente all'insegnamento dei principi di SimBiology inseriamo in input il seguente testo in entrambi i modelli.

“If you were a teacher, what topics related to SimBiology would you teach students?”

Le due diverse risposte sono raffigurate in Figura 2.

◆ Potential Topics for a Symbiology Course

Symbiology is a powerful tool for modeling and simulating biological systems. As a teacher, I would aim to provide students with a comprehensive understanding of its applications and techniques. Here are some potential topics I would cover:

Fundamental Concepts

- **Introduction to Symbiology:** Overview of the software, its capabilities, and its role in biological research.
- **Basic Modeling Principles:** Concepts of systems biology, mathematical models, and their representation in Symbiology.
- **Symbiology Components:** Understanding the various elements used in Symbiology models, such as compartments, species, reactions, and parameters.

Model Building and Analysis

- **Model Construction:** Step-by-step guidance on building biological models in Symbiology, including defining components, setting parameters, and specifying reactions.
- **Simulation Techniques:** Exploring different simulation methods (e.g., deterministic, stochastic) and their appropriate use cases.
- **Data Analysis:** Techniques for analyzing simulation results, including visualization, statistical analysis, and sensitivity analysis.

Applications in Biology

- **Metabolic Modeling:** Using Symbiology to model metabolic pathways and analyze cellular processes.
- **Cell Signaling:** Simulating signaling networks to understand cellular communication and responses.
- **Genetic Regulatory Networks:** Modeling gene expression and regulation to study developmental processes and disease mechanisms.
- **Population Dynamics:** Simulating population growth, competition, and predator-prey interactions.

Advanced Topics

- **Parameter Estimation:** Techniques for estimating model parameters from experimental data.
- **Model Validation:** Methods for assessing the accuracy and reliability of Symbiology models.
- **Model Integration:** Combining Symbiology models with other computational tools (e.g., MATLAB, Python) for more complex simulations.
- **Uncertainty Analysis:** Quantifying uncertainty in model predictions and identifying sensitive parameters.

By covering these topics, I would equip students with the knowledge and skills to effectively use Symbiology for their biological research and problem-solving.

(a)

As a teacher focusing on the subject of symbiology, I would cover a range of topics that encompass the study of symbols, their meanings, and their roles in various cultural, social, and scientific contexts. Here is a structured list of topics I would teach:

1. Introduction to Symbiology

- **Definition and Scope:** What is symbiology? Overview of the field.
- **History of Symbiology:** Key figures, historical development, and milestones.

2. Basic Concepts

- **Symbols and Signs:** Differences between symbols, signs, and icons.
- **Semiotics:** Study of signs and symbols as elements of communicative behavior.

3. Types of Symbols

- **Cultural Symbols:** Examination of symbols within different cultures and their meanings.
- **Religious Symbols:** Symbols used in various religions and their significance.
- **Scientific Symbols:** Symbols used in science, mathematics, and engineering.

4. Symbolism in Art and Literature

- **Art:** Analysis of symbolic elements in visual arts.
- **Literature:** Use of symbolism in literary works and their interpretations.

5. Psychological Aspects of Symbols

- **Jungian Archetypes:** Carl Jung's theories on archetypes and the collective unconscious.
- **Symbolism in Dreams:** Interpretation of symbols in dreams and their meanings.

6. Symbolism in Communication

- **Linguistic Symbols:** Role of symbols in language and communication.
- **Non-Verbal Communication:** Symbols in body language, gestures, and expressions.

7. Symbolism in Rituals and Traditions

- **Rituals:** Symbolic acts and their meanings in various cultural rituals.
- **Traditions:** Examination of traditional symbols and their roles in society.

8. Symbolism in Modern Media

- **Advertising:** Use of symbols in branding and marketing.
- **Film and Television:** Symbolic elements in visual storytelling.

9. Technological Symbols

- **Digital Symbols:** Emojis, icons, and their impact on digital communication.
- **Cyber Symbiology:** Study of symbols in virtual environments and online communities.

10. Symbolism in Nature

- **Natural Symbols:** Symbols derived from natural phenomena and their cultural significance.
- **Ecological Symbiology:** Role of symbols in environmental communication and conservation efforts.

11. Cross-Cultural Symbolism

- **Global Symbols:** Symbols that have universal meanings across cultures.
- **Cultural Variations:** How the same symbol can have different meanings in different cultures.

12. Practical Applications of Symbiology

- **Design and Architecture:** Use of symbols in design and architecture.
- **Education:** Teaching with symbols to enhance learning and understanding.
- **Therapy and Counseling:** Use of symbolic methods in psychological and therapeutic practices.

13. Research and Methodology in Symbiology

- **Research Methods:** Techniques for studying and interpreting symbols.
- **Case Studies:** In-depth analysis of specific symbols and their impact.

14. Future of Symbiology

- **Emerging Trends:** New developments in the study of symbols.
- **Interdisciplinary Approaches:** Connections between symbiology and other fields such as neuroscience, anthropology, and digital humanities.

15. Student Projects and Presentations

- **Symbol Analysis:** Students analyze a symbol of their choice and present their findings.
- **Creative Projects:** Creating new symbols or reinterpreting existing ones.

(b)

FIGURA 2: Risposta dei due modelli al prompt del caso 1: la figura (a) rappresenta la risposta data da Gemini, mentre la figura (b) riporta quella di ChatGpt.

- **ChatGPT:** fornisce una dettagliata scaletta di informazioni che andrebbe a fornire se fosse impiegato come un insegnante competente su questo argomento, partendo da un'introduzione ai concetti di sistemi di biologici e farmacocinetica/farmacodinamica (PK/PD) per poi passare a fornire istruzione per l'installazione di SimBiology, la creazione e simulazione di modelli PK. Dopo questi concetti base passa alla spiegazione di modelli PD legati ai PK, la stima e sensibilità dei parametri e via dicendo fino ad un insegnamento di concetti avanzati per l'uso di SimBiology in applicazioni su casi pratici. Conclude la sua risposta andando a fornire dell'informazione sull'integrazione di tecniche di Machine Learning e alcune considerazioni etiche per la modellazione e regolamentazione. Il modello è quindi in grado di dare una "scaletta" di concetti da insegnare partendo da quelli base fino ad arrivare a concetti complessi, tenendo conto dei nuovi strumenti esistenti da combinare con la modellazione PK/PD.

- **Gemini:** la risposta prodotta segue gli stessi principi di quella data da ChatGPT, cioè parte da un'introduzione di concetti base sulla biologia e il sistema SimBiology per poi passare alla costruzione dei modelli e le loro applicazioni in biologia ed infine fornisce una spiegazione di concetti avanzati legati a SimBiology. A differenza di ChatGPT che offre una spiegazione tecnica più dettagliata, Gemini espone in modo più ampio e specifico le applicazioni pratiche in biologia dei modelli ("Metabolic Modelling", "Cell Signaling" ...); al contrario di ChatGPT non fornisce considerazioni riguardo l'etica e la regolamentazione per l'uso di tali modelli.

- **CASO 2:** come secondo prompt in input ai LLMs nel campo dell'istruzione, si chiede alla macchina come è strutturato un codice per un modello di PK/PD costruito con SimBiology ipotizzando che l'IA sia un esperto di questo tool; inseriamo il prompt successivo in entrambi i due sistemi generativi

"You are an expert SimBiology code writer for population PKPD in the pharmaceutical industry. You are always accurate and precise when formulating SimBiology code. Please describe the structure of a SimBiology code using a basic example."

Il risultato ottenuto è in Figura 3.

Sure! In the context of pharmacokinetics (PK) and pharmacodynamics (PD) within the pharmaceutical industry, SimBiology in MATLAB provides a platform for modeling, simulating, and analyzing biological systems. A SimBiology model can include various components like compartments, species, parameters, reactions, and rules.

Here, I'll describe the structure of SimBiology code with a basic example of a single-compartment PK model with first-order absorption and elimination.

Steps to Create a Basic SimBiology Model

1. **Initialize the SimBiology Model:** Start by creating a SimBiology model object.
2. **Add a Compartment:** Add a compartment to represent the volume in which the drug is distributed.
3. **Define Species:** Define species within the compartment to represent the concentration of the drug.
4. **Add Parameters:** Add parameters for the rate constants for absorption and elimination.
5. **Add Reactions:** Define reactions for the absorption of the drug into the bloodstream and its elimination.
6. **Simulate the Model:** Simulate the model and visualize the results.

Example Code

Here's how you can structure and write SimBiology code for a basic single-compartment PK model:

```

matlab
% Initialize the SimBiology model
model = sbiomodel('PKModel');

% Add a compartment
compartment = addcompartment(model, 'Central', 'CapacityUnit', 'liter', 'InitialAmount', 1);

% Define species
drug = addspecies(compartment, 'Drug', 'InitialAmount', 0, 'InitialAmountUnit', 'milligram');

% Add parameters
ka = addparameter(model, 'ka', 1.0, 'ValueUnits', '1/hour'); % Absorption rate constant
ke = addparameter(model, 'ke', 0.5, 'ValueUnits', '1/hour'); % Elimination rate constant

% Add dosing
dose = adddose(model, 'Dose', 'Schedule');
dose.TargetName = 'Drug';
dose.Amount = 100; % Dose amount in mg
dose.TimeUnit = 'hour';
dose.AmountUnit = 'milligram';
dose.Rate = 50; % Dose rate in mg/hour
dose.StartTime = 0;

% Define reactions
% Absorption: dose -> drug in compartment
reaction1 = addreaction(model, 'null -> Drug');
kineticLaw1 = addkineticLaw(reaction1, 'MassAction');
set(kineticLaw1, 'ParameterVariableNames', 'ka');
addparameter(kineticLaw1, 'ka', 'Value', ka.Value);

% Elimination: drug in compartment -> null
reaction2 = addreaction(model, 'Drug -> null');
kineticLaw2 = addkineticLaw(reaction2, 'MassAction');
set(kineticLaw2, 'ParameterVariableNames', 'ke');
addparameter(kineticLaw2, 'ke', 'Value', ke.Value);

% Simulate the model
cs = getConfigset(model);
set(cs, 'StopTime', 24); % Simulation time in hours
simData = sbiosimulate(model);

% Plot the results
sbioplot(simData);
xlabel('Time (hours)');
ylabel('Concentration (mg/L)');
title('Drug Concentration Over Time');

```

(a)

Structure of Simbiology Code: A Basic Example

Simbiology is a powerful tool for modeling and simulating biological systems, including population PKPD (Pharmacokinetic/Pharmacodynamic) models. While it doesn't directly use NONMEM syntax, it provides a similar structure for defining and simulating models.

Basic Structure

A Simbiology model typically consists of the following components:

1. **Compartments:** These represent physical or conceptual spaces within the body. For example, a PKPD model might have compartments for the central compartment (blood), peripheral compartments (issues), and elimination compartments.
2. **Species:** These are the entities that move between compartments. In PKPD, species often represent drugs or metabolites.
3. **Reactions:** These define the interactions between species and their movement between compartments. For example, a reaction might represent drug absorption from the gut to the central compartment or drug elimination from the central compartment to the environment.
4. **Parameters:** These are numerical values that define the characteristics of the model, such as drug clearance, volume of distribution, and rate constants.

Example: One-Compartment PK Model

Let's consider a simple one-compartment PK model for a drug administered intravenously. The drug is absorbed directly into the central compartment and eliminated from the body.

```

Matlab
% Create a new Simbiology model
model = sbiomodel('OneCompartmentPK');

% Define the compartment
comp = sbiocompartment(model, 'Central');

% Define the species
drug = sbiospecies(model, 'Drug', comp);

% Define the reaction (elimination)
rxn = sbioreaction(model, 'Elimination', [drug, '-kDrug']);
rxn.kineticLaw = 'MassAction';

% Set the parameter
k = sbioparameter(model, 'EliminationRateConstant', 0.1);

% Simulate the model
[t, x] = sbiosimulate(model, 0:1:24);

% Plot the drug concentration
plot(t, x(:, 1));
xlabel('Time (hours)');
ylabel('Drug Concentration (mg/L)');

```

(b)

FIGURA 3: risposte date al prompt del caso 2 dai due strumenti generativi, con la figura (a) riferita a ChatGPT, mentre la (b) a Gemini.

➤ **ChatGPT:** la risposta fornita da ChatGPT dimostra che è in grado di generare con facilità un codice per un modello riferito a farmacocinetica e farmacodinamica nel contesto dell'industria farmaceutica sfruttando le funzionalità di MATLAB e SimBiology per creare modelli, simulazioni e analisi dei sistemi biologici. L'output generato ci dà una sequenza di step necessari per creare un modello simulativo su SimBiology tramite la scrittura di un codice MATLAB: si parte dall'iniziazione del modello di SimBiology fatta creando un oggetto

modello (% Initialize the SimBiology model), nel quale viene usata la funzione `'sbiomodel'`, successivamente si vanno ad aggiungere dei compartimenti al modello usando `'addcompartment'` per rappresentare il volume nel quale il farmaco è distribuito (%Add a compartment) e poi si definiscono le specie all'interno del compartimento andando a rappresentare la concentrazione del farmaco (%Define Species) con la funzione `'addspecies'`. Negli step successivi ChatGPT va ad aggiungere i parametri, le dosi e le reazioni, con i primi per le costanti di velocità per assorbimento ed eliminazione (%Add Parameters) con le funzioni `'addparameters'`, le dosi vengono classificate in base alla quantità, target, tempo di assunzione unità di misura, ratio e inizio assunzione (%Add Dosing), ed in questo caso la funzione utilizzata è `'adddose'`; mentre per la definizione delle reazioni è necessario conoscere l'assorbimento del farmaco nel flusso sanguigno e conseguente eliminazione (%Define Reactions), attraverso le due funzioni riferite rispettivamente alle reazioni `'addreaction'` e alle leggi cinetiche che ne regolano il flusso con `'addkineticlaw'`. Infine, come ultimi step si passa alla simulazione del modello con una configurazione data da `'getconfigset'` e `'simbiosimulat'`, ed infine conseguente visualizzazione del risultato con la `'sbioplot'` (%Simulate the results, %Plot the results). Una volta generato il codice da ChatGPT, lo esportiamo e incolliamo su MATLAB per andare a verificare il suo funzionamento utilizzando la funzione Run e come risultato otteniamo un modello vuoto di dimensione 1x1 funzionante su SimBiology, modello che è personalizzabile ed estendibile con l'aggiunta di dataset, compartimenti, specie, parametri e reazioni per specifici studi. Il codice però, pur essendo funzionante, presenta degli errori captati da MATLAB durante il suo run: "InitialAmount" non è una proprietà accessibile per un'istanza della classe "SimBiology.Compartment" ed un altro errore è presente sulla linea 5 del codice quando andiamo a definire i "compartment", connesso alla variabile 'InitialAmount.

- **Gemini:** allo stesso modo di ChatGPT, Gemini fornisce in risposta un esempio base di codice SimBiology indicando che, sebbene non usi direttamente una sintassi NONMEM, fornisce una struttura simile per definire e simulare i modelli. All'inizio viene fatta un'introduzione sulle componenti tipiche di un modello SimBiology: 1) Compartements, che rappresentano lo spazio fisico o concettuale nel corpo; 2) Species, l'entità che si spostano tra i compartimenti e nella PKPD rappresentano i farmaci o i metaboliti; 3) Reactions, definiscono le reazioni tra le specie e i loro movimenti nei compartimenti; 4) Parameters, valori numerici che definiscono le caratteristiche del modello come la farmaco clearance, volume di distribuzione e costanti di velocità. Successivamente è fornito un esempio di un modello PK ad un compartimento per

un farmaco somministrato per via intravenosa e assorbito direttamente nel compartimento centrale ed eliminata dal corpo; Gemini poi da un in output un codice MATLAB per la creazione di questo modello e dopo la scrittura di questo riassume una serie di punti chiave nella scrittura del codice come, “Model Definitions”, modello creato usando *sbiomodel*, in cui i compartimenti, specie e reazioni sono aggiunte usando le seguenti funzioni in sequenza: *'sbio compartment'*, *'sbiospecies'* e *'sbioreaction'* ; nella parte “Kinetic Laws”, le reazioni sono associate alle leggi cinetiche che definiscono le costanti di reazioni con la funzione come *'MassAction'*, mentre nella sezione “Parameters”, definiti usando *'sbioparameter'* e assegnati ai valori iniziali; “Simulation”, modello simulato usando *'sbiosimulate'* che dà in ritorno le concentrazioni delle specie e i punti temporali; “Visualization”, vengono visualizzati i risultati usando le funzioni standard plot presenti su MATLAB, le stesse modalità per valutarne il corretto funzionamento vengono applicate al codice generato da Gemini, quindi esportiamo il codice in MATLAB e lo facciamo eseguire ottenendo come risultato un modello vuoto di dimensione 1x1, il quale poi aprendo il tools SimBiology viene importato ed osserviamo che effettivamente il modello è stato creato correttamente per poi poter essere “riempito” nelle varie sezioni. Il codice generato però presenta degli errori nella scrittura, ovvero non è riconosciuta la funzione *'sbio compartment'* ed un errore nella linea 5 di codice, conseguente del precedente, cioè nella definizione dei compartment (“comp=sbio compartment”).

- **CASO 3:** in questo caso, a differenza dei due precedenti, sarà testata l'effettiva capacità dei due modelli generativi nella creazione di codici SimBiology, in particolare valuteremo se sono in grado di fornire un modello lineare PK con somministrazione orale di un farmaco. In input andremo ad inserire il seguente prompt in entrambi i modelli:

“Can you provide the SimBiology code for a linear PK model with oral administration?”.

Nella figura successiva sono mostrati i risultati ottenuti dai modelli. (Figura 4)

```

% Initialize the SimBiology model
model = sbiomodel('LinearPKModel');

% Add compartments
central = addcompartment(model, 'Central', 'CapacityUnit', 'liter', 'InitialAmount', 1);
gut = addcompartment(model, 'Gut', 'CapacityUnit', 'liter', 'InitialAmount', 1);

% Define species
drug_gut = addspecies(gut, 'Drug_Gut', 'InitialAmount', 0, 'InitialAmountUnit', 'milligram');
drug_central = addspecies(central, 'Drug_Central', 'InitialAmount', 0, 'InitialAmountUnit', 'milligram');

% Add parameters
ka = addparameter(model, 'ka', 1.0, 'ValueUnits', '1/hour'); % Absorption rate constant
ke = addparameter(model, 'ke', 0.5, 'ValueUnits', '1/hour'); % Elimination rate constant
V = addparameter(model, 'V', 10, 'ValueUnits', 'liter'); % Volume of distribution

% Add reactions
% Absorption: Drug in gut -> Drug in central compartment
reaction1 = addreaction(model, 'Drug_Gut -> Drug_Central');
kineticLaw1 = addkineticLaw(reaction1, 'MassAction');
set(kineticLaw1, 'ParameterVariableNames', 'ka');
addparameter(kineticLaw1, 'ka', 'Value', ka.Value);

% Elimination: Drug in central compartment -> null
reaction2 = addreaction(model, 'Drug_Central -> null');
kineticLaw2 = addkineticLaw(reaction2, 'MassAction');
set(kineticLaw2, 'ParameterVariableNames', 'ke');
addparameter(kineticLaw2, 'ke', 'Value', ke.Value);

```

```

Matlab
% Create a new Simbiology model
model = sbiomodel('OralPKModel');

% Define compartments
central = sbiocompartment(model, 'Central');
gut = sbiocompartment(model, 'Gut');

% Define species
drug = sbiospecies(model, 'Drug', central);

% Define reactions
absorption = sbioreaction(model, 'Absorption', [drug], ka*drug);
absorption.kineticLaw = 'MassAction';
elimination = sbioreaction(model, 'Elimination', [drug], k*drug);
elimination.kineticLaw = 'MassAction';

% Set parameters
ka = sbioparameter(model, 'AbsorptionRateConstant', 1);
k = sbioparameter(model, 'EliminationRateConstant', 0.2);

% Set initial condition
drug.InitialAmount = 100;

% Simulate the model
[t, x] = sbiosimulate(model, 0:1:24);

% Plot the drug concentration
plot(t, x(:, 1));
xlabel('Time (hours)');
ylabel('Drug Concentration (mg/L)');

```

```

% Add dosing
dose = adddose(model, 'OralDose', 'Schedule');
dose.TargetName = 'Drug_Gut';
dose.Amount = 100; % Dose amount in mg
dose.TimeUnit = 'hour';
dose.AmountUnit = 'milligram';
dose.Rate = 0; % Instantaneous dose
dose.StartTime = 0;

% Simulate the model
cs = getconfigset(model);
set(cs, 'StopTime', 24); % Simulation time in hours
simData = sbiosimulate(model);

% Plot the results
sbioplot(simData);
xlabel('Time (hours)');
ylabel('Concentration (mg/L)');
title('Drug Concentration Over Time');

```

(a)

(b)

FIGURA 4: risultati ottenuti dal terzo prompt inserito nel caso 3, figura (a) che rappresenta il codice fornito da ChatGPT e la figura (b) quello dato da Gemini.

- **ChatGPT:** in questo caso abbiamo chiesto all'intelligenza artificiale di generare effettivamente un codice in SimBiology per un modello PK lineare con somministrazione orale, dicendo al modello di comportarsi come un codificatore esperto in SimBiology e in output abbiamo ricevuto un codice MATLAB che genera il modello da noi desiderato. Come nel caso due, ChatGPT ci dà un processo composto

da step e indica che quest'esempio di codice include un modello a un compartimento con un assorbimento di primo ordine ed altrettanto vale per l'eliminazione. In questo caso il codice segue gli stessi step di quello precedente con la differenza che rispetto al caso due il codice generato è più articolato, infatti si può vedere che nella sezione di definizione dei compartimenti si dividono in 'central' per il flusso sanguigno e 'gut' per il luogo di assorbimento usando le stesse funzioni 'addcompartment', conseguentemente anche nella sezione di "Define Species" c'è una suddivisione nella concentrazione del farmaco nel sangue e nel sito di assorbimento, cioè tra 'gut' e 'central', usando per entrambi la funzione 'addspecies'; nella parte di aggiunta dei parametri viene aggiunto alle due costanti di assorbimento e eliminazioni presenti anche nel caso due, 'ka' (costante assorbimento) e 'ke' (costante eliminazione), il parametro 'V' che indica il volume della distribuzione, e tutte le costanti vengono definite utilizzando la funzione 'addparameter'; successivamente viene prima scritta la parte riguardante le reazioni(%add reaction) rispetto al caso due e dopo la parte riguardo le concentrazioni (%Add dosing), per la prima la scrittura e le funzioni ('addreaction' e 'addkineticlaw') rimangono le stesse del caso due sempre considerando le vie di assorbimento ed eliminazione, mentre per la sezione riguardante il dosaggio cambia il tasso di concentrazione rispetto al caso trattato precedentemente, ma la funzione 'adddose' rimane uguale; infine la simulazione e visualizzazione del risultato rimane con la stessa codifica. A questo punto, copiato il codice lo inseriamo in MATLAB e vediamo che riesce a generare come risultato un modello vuoto di dimensione 1x1 il quale poi, importato in SimBiology è pronto per essere utilizzato. Il codice presenta gli stessi errori di quello precedente, cioè l'"InitialAmount" non è accessibile e conseguenzialmente c'è un errore nella parte di definizione dei compartimenti, 'central'.

- **Gemini:** una volta inserito il prompt, Gemini da subito come output il codice in MATLAB che secondo l'IA è più corretto e dopo fornisce una spiegazione dei vari passaggi che l'hanno portato a scriverlo in un certo modo. Il codice generato nelle prime due sezioni usa lo stesso ordine di definizione simile a quello dato da ChatGPT, ma cambia radicalmente nell'uso delle funzioni per definire i compartimenti e le species dei farmaci, utilizza infatti rispettivamente le funzioni 'sbiocompartment' e 'sbiospecies', inoltre nella parte di %Define Species considera solo il farmaco nel reparto centrale di assorbimento; lo step successivo è quello di andare a definire prima

le reazioni di assorbimento ed eliminazione associati alle leggi cinetiche con la funzione *'sbioreaction'* e poi si vanno a impostare i parametri, cioè le costanti di assorbimento ed eliminazione utilizzando *'sbioparameter'*. In questo caso viene solo impostata la condizione iniziale della quantità nel farmaco, uguale a 100, ma non viene specificata l'unità di misura del peso e del tempo, il tasso di dose istantanea e il target del farmaco rispetto a ChatGPT. Per la parte di simulazione usa la funzione *'sbiosimulate'*, nella visualizzazione viene settata manualmente. Gemini esplicita i punti chiave del codice generato, ovvero: la creazione del modello, la definizione dei due compartimenti 'central' e 'gut', le specie definite nel compartimento 'central', le due reazioni di assorbimento ed eliminazione tra cui la prima rappresenta il farmaco che va dal 'gut' al 'central' con la costante di velocità ka e l'altra di eliminazione dal 'central' con la k ; infine si definiscono i parametri (ka , k) e le condizioni iniziali, per poi passare alla simulazione del modello per 24 ore e il grafico della concentrazione del farmaco nel 'central' nel tempo. Gli stessi errori presenti nel caso due del codice persistono anche in questa situazione.

- **CASO 4:** nel quarto ed ultimo caso, andiamo a chiedere ai due modelli generativi di fornire un ulteriore codice Simiology, ma questa volta dovrà essere un modello con un compartimento con due vie parallele per l'assorbimento e una eliminazione lineare, quindi come prompt in input inseriamo:

“Provide Simbiology code for a one-compartment model with two parallel absorptions and linear elimination.”

(Figura 5)

```

% Initialize the SimBiology model
model = sbiomodel('DiabetesPKModel');

% Add compartments
central = addcompartment(model, 'Central', 'CapacityUnit', 'liter', 'InitialAmount', 1);
gut = addcompartment(model, 'Gut', 'CapacityUnit', 'liter', 'InitialAmount', 1);

% Define species
drug_gut = addspecies(gut, 'Drug_Gut', 'InitialAmount', 0, 'InitialAmountUnit', 'milligram');
drug_central = addspecies(central, 'Drug_Central', 'InitialAmount', 0, 'InitialAmountUnit', 'milligram');

% Add parameters
% Adjust ka and ke to reflect typical values for people with diabetes if known
ka = addparameter(model, 'ka', 0.8, 'ValueUnits', '1/hour'); % Absorption rate constant
ke = addparameter(model, 'ke', 0.4, 'ValueUnits', '1/hour'); % Elimination rate constant
V = addparameter(model, 'V', 10, 'ValueUnits', 'liter'); % Volume of distribution

% Add reactions
% Absorption: Drug in gut -> Drug in central compartment
reaction1 = addreaction(model, 'Drug_Gut -> Drug_Central');
kineticLaw1 = addkineticLaw(reaction1, 'MassAction');
set(kineticLaw1, 'ParameterVariableNames', 'ka');
addparameter(kineticLaw1, 'ka', 'Value', ka.Value);

% Elimination: Drug in central compartment -> null
reaction2 = addreaction(model, 'Drug_Central -> null');
kineticLaw2 = addkineticLaw(reaction2, 'MassAction');
set(kineticLaw2, 'ParameterVariableNames', 'ke');
addparameter(kineticLaw2, 'ke', 'Value', ke.Value);

```

```

% Add dosing
dose = adddose(model, 'OralDose', 'Schedule');
dose.TargetName = 'Drug_Gut';
dose.Amount = 100; % Dose amount in mg
dose.TimeUnit = 'hour';
dose.AmountUnit = 'milligram';
dose.Rate = 0; % Instantaneous dose
dose.StartTime = 0;

% Simulate the model
cs = getconfigset(model);
set(cs, 'StopTime', 24); % Simulation time in hours
simData = sbiosimulate(model);

% Plot the results
sbioplot(simData);
xlabel('Time (hours)');
ylabel('Concentration (mg/L)');
title('Drug Concentration Over Time in Diabetic Patients');

```

(a)

```

% Create a new Simbiology model
model = sbiomodel('OneCompartmentModelWithTwoParallelAbsorptions');

% Define compartments
central = sbiocompartment(model, 'Central');
gut1 = sbiocompartment(model, 'Gut1');
gut2 = sbiocompartment(model, 'Gut2');

% Define species
drug = sbiospecies(model, 'Drug', central);

% Define reactions
absorption1 = sbioreaction(model, 'Absorption1', [drug], ka1*drug);
absorption1.kineticLaw = 'MassAction';
absorption2 = sbioreaction(model, 'Absorption2', [drug], ka2*drug);
absorption2.kineticLaw = 'MassAction';
elimination = sbioreaction(model, 'Elimination', [drug], k*drug);
elimination.kineticLaw = 'MassAction';

% Set parameters
ka1 = sbioparameter(model, 'AbsorptionRateConstant1', 0.5);
ka2 = sbioparameter(model, 'AbsorptionRateConstant2', 0.3);
k = sbioparameter(model, 'EliminationRateConstant', 0.2);

% Set initial condition
drug.InitialAmount = 100;

% Simulate the model
[t, x] = sbiosimulate(model, 0:1:24);

% Plot the drug concentration
plot(t, x(:, 1));
xlabel('Time (hours)');
ylabel('Drug Concentration (mg/L)');

```

(b)

Figura 5: risultati ottenuti inserendo l'ultimo prompt del caso 4. A sinistra (a) l'output di ChatGPT, a destra (b) la risposta di Gemini.

- **ChatGPT:** l'IA genera come uscita un codice completo rappresentante il modello a due vie parallele di assorbimento e una lineare per l'eliminazione, assumendo che il farmaco viene assorbito attraverso due strade diverse, ognuna delle quali ha la propria velocità costante di assorbimento per poi essere eliminato dal compartimento centrale con una ratio costante. Il codice generato segue gli stessi step dei due casi precedenti utilizzando le stesse funzioni, ma questa volta data la nostra richiesta di un modello a due vie di assorbimento cambiano le varie sezioni di codifica, in particolar modo si può osservare che nelle parti di definizione delle species, dei parametri, delle reazioni e del dosing il codice deve considerare separatamente le due vie, mentre per quella di eliminazione rimane una sola via e la codifica è uguale a quella del caso 2 e caso 3. ChatGPT quindi, ha generato esattamente quello che avevamo chiesto in input separando le due linee di assorbimento definendole separatamente (`%Define species`), considerando le due diverse costanti di assorbimento ($ka1, ka2$) e le due diverse reazioni che portano all'assorbimento del farmaco con le rispettive leggi cinetiche che lo regolano, infine come per le reazioni il codice considera separatamente le dosi per l'assorbimento delle due linee separate; alla fine, stesso dei casi precedenti, usa la stessa codifica per la simulazione e visualizzazione del modello. Il codice esportato in MATLAB funziona e riesce a creare un modello in SimBiology, portandosi dietro sempre gli stessi errori di codifica che potremmo definire "cronici", cioè "InitialAmount" non accessibile e il conseguente errore nella definizione dei *'compartment'*. Questo modello può ovviamente essere modificato in base alle esigenze diverse per riflettere diversi scenari fisiologici o dei setup sperimentali.
- **Gemini:** dopo avere inserito il nostro prompt, Gemini genera un codice in output che ritiene migliore, scegliendo tra 3 bozze diverse, il quale segue lo stesso path di codifica delle richieste precedenti. In questo caso però, l'IA ha tenuto conto della nostra direttiva di considerare due vie parallele e diverse per l'assorbimento, quindi dopo aver creato il modello come di consuetudine, il codice va a definire i compartimenti separati come *'central'* *'gut1'* *'gut2'*, con gli ultimi due che sono riferiti alle vie d'assorbimento; per definire le specie del farmaco usa unicamente la funzione *'sbiospecies'* e successivamente va a codificare le due differenti reazioni di assorbimento tenendo conto separatamente delle leggi cinetiche e delle costanti diverse di assorbimento ($ka1, ka2$), utilizzando sempre la funzione *'sbioreaction'* e la proprietà *'MassAction'*. Per quanto riguarda il setup delle condizioni iniziali, Gemini

assume che la quantità iniziale sia di 100 considerando di default l'unità di misura in 'mg', senza però specificare l'inizio di tempo; nell'ultima parte usa le stesse linee di codice mostrate precedentemente per la visualizzazione e simulazione del modello. Esportato il codice in MATLAB osserviamo la ripetizione degli stessi errori precedenti e la creazione del modello importabile in SimBiology.

Oltre a questi quattro casi, ne abbiamo studiato un altro più complesso andando ai due strumenti IA la creazione di un codice SimBiology per un modello che considera il monitoraggio continuo di glucosio e conseguente somministrazione di insulina in base alla necessità, osservando che le due IA sono in grado di capire questo quesito più complesso andando a considerare la dinamica di glucosio e insulina nel sangue, l'allarme che fa scattare la somministrazione di insulina e la dinamica di quest'ultima quando diminuisce il livello di glucosio nel sangue. Si è potuto osservare che sono in grado di generare codici che riguardano anche la farmaco dinamica, comprendendo autonomamente la separazione della cinetica rispetto alla dinamica insulina-glucosio e imponendo una regola per il monitoraggio e somministrazione di insulina.

In definitiva, i due modelli sono stati valutati in diversi casi per delinearne le capacità e limiti di generazione, comportandosi come se fossero effettivamente degli "esperti" in codifica SimBiology, rispondendo correttamente a tutte le nostre richieste fatte. Attraverso questo studio possiamo dire che entrambi gli strumenti possono essere più che utili per la creazione di codici in SimBiology anche da parte di persone non esperte del settore, le quali possono andare ad inserire diversi tipi di prompt scritti in un linguaggio non specifico per poi ottenere dei veri e propri codici che potranno poi essere implementati e modificati in base alle diverse esigenze e casi di studio.

I codici generati in output però, non sono esenti da errori poi rilevati nella simulazione vera e propria in MATLAB, errori che potremmo andare a definire come "errori cronici" perché entrambi i LLMs hanno presentato lo stesso tipo di errori in loop nei casi di codifica vera e propria; errori che però non sono gli stessi per tutti e due i modelli ma differiscono in base allo strumento che ha generato il codice. Per quanto riguarda il caso ChatGPT, l'errore che persiste nei diversi casi è quello della incapacità di MATLAB nell'accedere alla proprietà 'InitialAmount' per un'istanza della classe 'SimBiology.compartment' e come conseguenza ci dà un errore di codifica nella linea 5 (caso2,3), riferita alla parte della definizione dei 'compartment'. Inserendo però l'errore dato da Matlab in ChatGPT, quest'ultimo fornisce in output una correzione, ammettendo che 'InitialAmount' non è una proprietà di un compartment in SimBiology ma piuttosto una proprietà delle specie all'interno del compartimento, quindi va a correggere il codice assicurandosi che le specie vengono aggiunte ai

compartimenti con il loro 'InitialAmount' specificato e non i compartimenti stessi; inoltre ci dà anche una serie di punti chiave adottati per questa correzione, cioè ci dice di settare la capacità del compartimento usando a posto di 'InitialAmount' la proprietà 'Capacity', poi di inserire la proprietà 'InitialAmount' nell'inizializzazione delle specie ed infine cambia la codifica della visualizzazione utilizzando *'sbioplot'* che grafica automaticamente i risultati di simulazione.

Nel caso di Gemini, il tipo di errore è completamente differente e sicuramente di maggior gravità rispetto a quello di ChatGPT che si tratta di un errore nello specificare una proprietà del modello, mentre quello generato da Gemini riguarda un non riconoscimento da parte di Matlab della funzione *'sbiocompartment'*, la quale infatti non è presente in SimBiology. Andando ad approfondire questo tipo di errore notiamo che anche le funzioni 'sbioreaction', 'sbiospecies' non sono presenti mentre per la funzione 'sbioparameter' la quale non è riconosciuta lo stesso, ci trova la funzione 'sbioparameterci'; l'unica funzione presente è 'sbiosimulate'. Inserendo questo tipo di errore in Gemini, ci viene dato in ritorno un processo per verificare la corretta installazione del tool SimBiology in MATLAB, che però risulta non esatta dato che dopo aver effettuato un controllo SimBiology è installato correttamente con tutte le sue funzioni.

Discussione e Conclusione

In questo studio, l'obiettivo principale è stato quello di sviluppare e valutare l'efficacia di strumenti IA generativi, in particolare i LLMs, utilizzandoli per creare dei modelli biologici farmacocinetici attraverso la scrittura di codici MATLAB per il tool SimBiology. A tal fine sono stati utilizzati due strumenti tra i più comuni e accessibili, cioè ChatGPT e Gemini, andando a inserire delle specifiche richieste sulla creazione di codici MATLAB per la modellazione di sistemi biologici applicati alla farmacocinetica e poi valutarne la loro capacità nel farlo. I risultati ottenuti da entrambi i modelli dimostrano come, sia ChatGPT che Gemini, sono effettivamente capaci di comportarsi come dei codificatori esperti in SimBiology, cioè in grado di generare codici di una certa qualità che presentano però delle importanti differenze nelle prestazioni. Valutando l'effettivo funzionamento di questi codici, ChatGPT si è rivelato essere il LLMs più accurato e tecnico per quanto riguarda la scrittura dei codici utilizzando le corrette funzioni di SimBiology e, soprattutto, ha dimostrato di avere un'elevata capacità di autovalutazione e correzione dei propri errori. Infatti, i codici generati da esso presentavano degli errori sulla proprietà "InitialAmount" nei compartimenti, ma ChatGPT è stata in grado di ammettere l'errore e correggerlo in modo appropriato. D'altro canto, Gemini, pur essendo in grado di generare dei codici funzionanti in MATLAB, ha mostrato una tendenza a generare degli output più "user-friendly" e più semplificati, presentando inoltre delle maggiori limitazioni nella correzione dei propri errori e soprattutto nella gestione delle funzioni utilizzabili in SimBiology. Una importante osservazione, che è valida per entrambi i modelli generativi, è la presenza di errori "cronici" in tutti i codici generati, cioè degli errori uguali presenti in ognuna delle nostre quattro richieste, rispettivamente diversi per i due modelli. Questa cronicità va a sottolineare i limiti attuali delle due IA per l'automazione di processi complessi come la simulazione biologica, tuttavia, i vantaggi di questi risiedono nella possibilità di creare modelli base che possono poi essere modificati manualmente per adattarli ai diversi tipi di studio.

In conclusione, l'introduzione e l'applicazione di strumenti come ChatGPT e Gemini nell'ambito della modellistica farmacocinetica, ha dimostrato un notevole potenziale sebbene questi strumenti non siano ancora in grado di sostituire completamente l'essere umano. Gli strumenti generativi risultano essere validi per supportare la creazione di modelli farmacocinetici per utenti meno esperti nel settore, ma richiedono un ulteriore sviluppo per riuscire a superare le attuali limitazioni.

Bibliografia

- [1] Pruneti, C. "Psicopatologia e fisiologia dei disturbi del comportamento alimentare: Dall'analisi al trattamento dei problemi della nutrizione e della percezione del sé corporeo". Società Editrice Esculapio, 2021.
- [2] Widmaier, E. P., Raff, H., Strang, K. T.. "Vander Fisiologia" (Imeri, L., Ricci, V., Rivolta, I., a cura di). C.E.A. Casa Editrice Ambrosiana. (2018)
- [3] Holz, M. ,Fahr,A.."Compartment modelling." *Advanced Drug Delivery Reviews* 48.2-3 (2001): 249-264.
- [4] Chen, B., Abuassba, A.O.."Compartmental models with application to pharmacokinetics". *Procedia Computer Science* 187 (2021), 60-70.
- [5] Janssen, A., Leebeek, F. W., Cnossen, M. H., Mathôt, R .A.. "Deep compartment models: a deep learning approach for the reliable prediction of time-series data in pharmacokinetic modeling." *CPT: Pharmacometrics & Systems Pharmacology* 11.7 (2022): 934-945.
- [6] Park, S. G., Kim, A., Yoon, T., Kamyod, C., Kim, C. G.. "A Study of Generative Large Language Model for Healthcare." 2023 7th International Conference on Information Technology (InCIT). IEEE, 2023.
- [7] LeCun, Y., Bengio, Y., Hinton, G.. "Deep learning." *nature* 521.7553 (2015): 436-444.
<https://hal.science/hal-04206682>
- [8] Zhang, S., Bamakan, S. M. H., Qu, Q., Li, S. et al. "Learning for personalized medicine: a comprehensive review from a deep learning perspective." *IEEE reviews in biomedical engineering* 12 (2018): 194-208.
- [9] Shamshirband, S., Fathi, M., Dehzangi, A., Chronopoulos, A. T., Alinejad-Ronky, H.. "A review on deep learning approaches in healthcare systems: Taxonomies, challenges, and open issues." *Journal of Biomedical Informatics* 113 (2021): 103627.
<https://doi.org/10.1016/j.jbi.2020.103627>
- [10] Huang, T., Ma, L., Zhang, B., Liao, H.. "Advances in deep learning: From diagnosis to treatment." *BioScience Trends* 17.3 (2023): 190-192.
- [11] Zhang, P., Kamel Boulos, M. N.. "Generative AI in medicine and healthcare: promises, opportunities and challenges." *Future Internet* 15.9 (2023): 286.

- [12] Yu, P., Xu, H., HU, X., Deng, C.. "Leveraging generative AI and large Language models: a Comprehensive Roadmap for Healthcare Integration." *Healthcare*. Vol. 11. No. 20. MDPI, 2023. <https://doi.org/10.3390/healthcare11202776>
- [13] Khullar, D., Wang, X., Wang, F.. "Large Language Models in Health Care: Charting a Path Toward Accurate, Explainable, and Secure AI." *Journal of General Internal Medicine* (2024): 1-3. <https://doi.org/10.1007/s11606-024-08657-2>
- [14] Kopitar, L., Fister Jr, I., Stiglic, G.. "Using Generative AI to Improve the Performance and Interpretability of Rule-Based Diagnosis of Type 2 Diabetes Mellitus." *Information* 15.3 (2024) : 162. <https://doi.org/10.3390/info15030162>
- [15] Rodriguez, D. V., Lawrence, K., Gonzalez, J., Brandfield-Harvey, B., Xu, L., Tasneem, S., Mann, D.. "Leveraging generative AI tools to support the development of digital solutions in health care research: case study." *JMIR Human Factors* 11.1 (2024): e52885, DOI: 10.2196/52885 <https://humanfactors.jmir.org/2024/1/e52885>
- [16] Jiang, Y., García-Durán, A., Losada, I. B., Girard, P., Terranova, N.. "Generative models for synthetic data generation: application to pharmacokinetic/pharmacodynamic data." *Journal of Pharmacokinetics and Pharmacodynamics* (2024): 1-9.
- [17] Shin, E., Yu, Y., Bies, R. R., Ramanathan, M.. "Evaluation of ChatGPT and large language models for pharmacometrics with NONMEM." *Journal of Pharmacokinetics and Pharmacodynamics* (2024): 1-11.

Ringraziamenti