POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Gestionale
Classe LM31 – Percorso Finance

“Revisione sistematica della letteratura in merito all'utilizzo di tecniche AI per potenziare modelli di system dynamics”

Relatore
Prof. Giovanni Zenezini

Candidato
Carbone
Michele Vittorio
320070

Co-relatore
Prof. Filippo Maria Ottaviani

A.A. 2024/2025

Indice

Introduzione

Con la crescente complessità dei sistemi dinamici, è sempre più difficile modellare e prevedere il comportamento dei fenomeni reali in settori come economia, energia, sanità e finanza.

Agli inizi degli anni ’50 Jay Forrester, ingegnere elettrotecnico e informatico teorizza e sviluppa la disciplina del System Dynamics, la quale iniziale funzione era quella di determinare tramite metodi ingegneristici i fattori che determinano il successo o il fallimento delle aziende.

Questa disciplina, molto utilizzata negli ambiti della simulazione, ha però dei limiti, quali:

- Dipendenza da parametri predefiniti: i modelli richiedono infatti dati precisi, che in un ambiente dinamico sono molto difficili da reperire,

- Difficoltà nell’adattamento a grandi dataset: risulta complesso sfruttare appieno dei grandi dataset a disposizione,

- Limitazioni nella capacità predittiva: questa disciplina è molto utile per capire le relazioni di causalità, ma ha delle difficoltà nel creare previsioni accurate quando i sistemi diventano altamente instabili.

L’Intellingenza Artificiale può essere uno strumento per superare queste criticità, infatti può:

- Identificare tendenze nascoste nei dati, migliorando la comprensione delle dinamiche all’interno del sistema.

- Automatizzare la calibrazione dei parametri, evitando continui interventi manuali,
- Utilizzare tecniche di machine learning, deep learning e ottimizzazione per migliorare lacapacità predittiva

Oltre questi aspetti è possibile utilizzare tecniche AI per creare, migliorare approssimare passaggi o per identificare anomalie all'interno dei modelli.

L'integrazione tra questi due strumenti è un nuovo campo emergente, con un numero crescente di studi e applicazioni. Nella letteratura, manca tuttavia una revisione sistematica che sintetizzi le metodologie più efficaci nel risolvere questi problemi e identifichi le sfide aperte.

Questa tesi mira a colmare questa lacuna, fornendo una panoramica completa delle varie metodologie e delle stradegie usate per integrare i sistemi SD con l'AI e delle loro diverse applicazioni nei diversi settori.

Nel Capitolo 1 analizzerò il background teorico già esistente, dove illustrerò i principi basilari del System Dynamics e le varie metodologie AI ed eventuali vantaggi possibili dalle intersezioni tra le due.

Nel Capitolo 2 spiegherò la metodologia usata per la ricerca della letteratura su database che raccolgono articoli scientifici e come abbia trovato i miei risultati.

Nel Capitolo 3 analizzerò gli articoli selezionati, portando i risultati della ricerca, insieme ad eventuali gap di letteratura da riempire.

Il Capitolo 4 avrà un'analisi critica dei risultati presentati nel Capitolo 3, analizzando benefici, sfide tecniche e opportunità di ricerca futura.

Il Capitolo 5 è dedicato alle conclusioni ricavate dai capitoli precedenti, dove ci sarà una sintesi delle scoperte e le implicazioni per il mondo della ricerca.

1. Background teorico

1.1 System Dynamics (SD)

1.1.1 Storia

System Dynamics è stato creato durante la metà degli anni ’50 da Jay Forrester, ingegnere elettrotecnico ed informatico e professore del Massachusetts Institute of Technology, (MIT).

Dopo aver accettato una cattedra alla MIT School of Management, Forrester, con il suo background ingegneristico, inizia a pensare come si possano comprendere i fattori che determinano il successo o il fallimento delle aziende.

In seguito ad un incontro con i dirigenti della General Electric, (GE), preoccupati per i cicli di occupazione in alcuni dei loro impianti e tramite alcune simulazioni, Forrester è stato in grado di mostrare come questa instabilità nell’occupazione fosse causata dalla struttura interna della società e non da una forza esterna, quale possa essere un ciclo macroeconomico.

Queste simulazioni sanciscono di fatto la nascita del System Dynamics come disciplina. Negli anni successivi Forrester riunisce un team di dottorandi e crea una modellazione formale con le prime simulazioni su un compilatore. Successivamente Richard Bennett crea il primo linguaggio di modellazione SD per computer "Simulation of Industrial Management Problems with Lots of Equations" (SIMPLE) e successivamente nel 1959 Phyllis Fox e Alexander Pugh modellizzano la prima versione di "Dynamic Models" (DYNAMO), una versione migliorata di SIMPLE. Nel 1961 esce il primo libro sull’argomento, pubblicato da Forrester col nome "Industrial Dynamics".

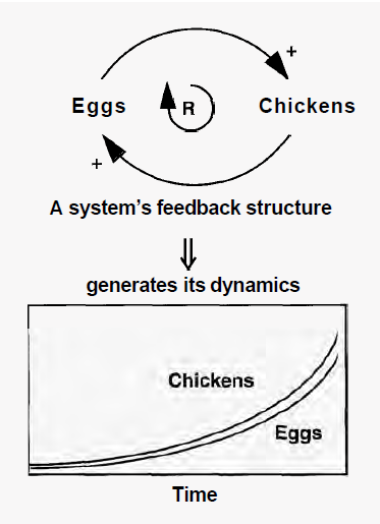
Nel 1968 ci fu il primo allargamento del campo applicativo del System Dynamic: la collaborazione tra l'ex sindaco di Boston John Collins e Forrester porta alla pubblicazione del libro "Urban Dynamics", evento che ha aperto le porte a nuovi utilizzi per System Dynamics. Negli anni System Dynamics ha proseguito la sua evoluzione e i campi di applicazione sono aumentati sempre di più. Un esempio su tutti è il progetto World Dynamics in cui si voleva simulare il sistema socioeconomico mondiale.

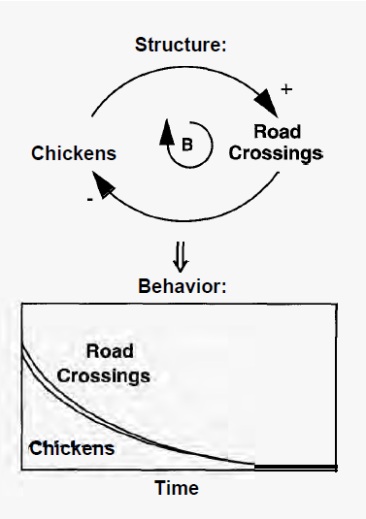
1.1.2 Introduzione ai modelli SD

Il System Dynamics, come già detto, è un approccio alla comprensione del comportamento dei sistemi complessi nel corso del tempo.

Come possiamo ottenere questo risultato?

Innanzi tutto, dobbiamo creare un causal diagram, o diagramma causale. Un diagramma causale è uno schema che ci aiuta a capire quali sono le variabili correlate nel sistema che si influenzano a vicenda. Queste relazioni possono essere positive o negative, ovvero se all'aumentare della prima variabile aumenta la seconda o viceversa. I diagrammi più interessanti da studiare sono i cosiddetti *causal loop diagram,* o abbreviati CLD. Questi ultimi sono dei diagrammi composti da loop, ovvero da cicli di influenza che ogni variabile da all'altra. Questi diagrammi hanno dei feedback, che possono essere positivi, (*self-reinforcing)* o negativi (*self-correcting)*





Self-reinforcing CLD *Self-correcting CLD*

(Business Dynamics, Stermann,2000)

Il feedback positivo: i cicli positivi si auto-rinforzano.

In questo caso, più galline depongono uova, che si schiudono e aumentano il numero di galline, queste si schiudono e si aggiungono alla popolazione di polli, portando a nuove uova e così via. Un diagramma a loop causale o CLD illustra la dipendenza di feedback tra polli e uova. Le frecce indicano le relazioni causali. I segni "+" presenti sulle punte delle frecce indicano che l'effetto è positivamente alla causa: un aumento del numero di galline.

Viceversa, una diminuzione della popolazione di galline fa sì che la deposizione di uova diminuisca.

Questo ciclo è auto-rinforzante, da cui l'identificativo di polarità R. Se questo ciclo fosse l'unico in funzione, il numero di uova deposte ogni giorno sarebbe

Se questo ciclo fosse l'unico in funzione, la popolazione di galline e di uova crescerebbero entrambe in modo esponenziale.

Naturalmente, nessuna quantità reale può crescere all'infinito. Ci sono dei limiti alla crescita. Questi limiti sono creati da feedback negativi.

Feedback negativo: I loop negativi si autocorreggono e contrastano il cambiamento. Man mano che la popolazione di polli cresce, vari loop negativi agiscono per bilanciare la popolazione di polli con la sua capacità di carico.

Un classico feedback è mostrato in figura: più polli ci sono, più attraversano la strada. Se c'è traffico, un maggior numero di attraversamenti stradali porterà a un minor numero di polli (da qui la polarità negativa del collegamento tra attraversamenti stradali e polli). Un aumento della popolazione di polli provoca un maggior numero di attraversamenti stradali rischiosi, che riportano la popolazione di polli in basso.

La B al centro di un anello indica un feedback di bilanciamento. Se il ciclo di attraversamento della strada fosse l'unico in funzione (ad esempio perché l'allevatore vende tutte le uova), il numero di galline diminuirebbe gradualmente fino a scomparire.

Tutti i sistemi, per quanto complessi, sono costituiti da feedback loops positivi e negativi e tutte le dinamiche derivano dall'interazione di questi anelli tra loro.

1.1.3 Stock e flow

Introduciamo ora i concetti di accumulo (*stock*) e flusso (*flow*).

Gli stock rappresentano le quantità immagazzinate in un sistema in un dato momento. Sono come "serbatoi" che raccolgono o rilasciano risorse.

D'altro canto, invece i flussi rappresentano i tassi di cambiamento degli stock. Sono le variabili che determinano l'aumento o la diminuzione di uno stock nel tempo.

L'equazione fondamentale per rappresentare uno stock è:

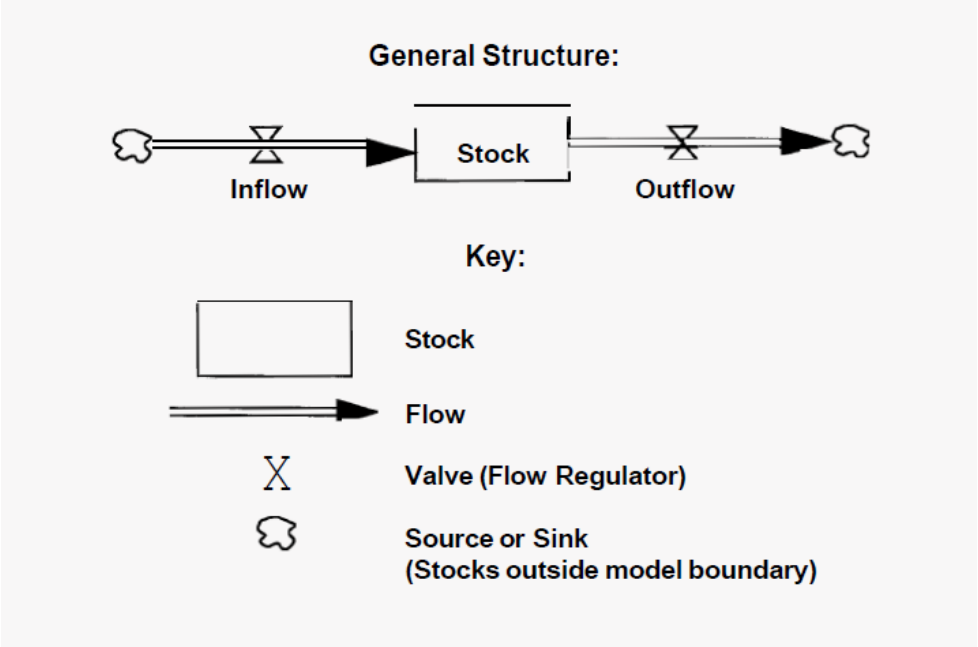
$$Stock(t) = Stock(t-1) + Inflow - Outflow$$

Dove:

- **Stock(t)** è il valore dello stock al tempo **t**.
- **Stock(t-1)** è il valore dello stock al tempo precedente.
- **Inflow** rappresenta l'afflusso che aumenta lo stock.
- **Outflow** rappresenta il deflusso che riduce lo stock.

Nei modelli SD, stock e flussi sono rappresentati con una notazione grafica standard:

- **Stock** = rettangolo 🟥 (es. popolazione, risorse disponibili).
- **Flusso** = frecce con valvole 🔺 (tasso di crescita o riduzione dello stock).
- **Variabili ausiliarie** = cerchi 🔵 (modificano i tassi di flusso).
- **Collegamenti causali** = frecce normali ➡️ (relazioni tra variabili).



(Business Dynamics, Stermann,2000)

Da un punto di vista formale, dato che nei modelli SD vengono utilizzate equazioni differenziali per simulare possiamo scrivere la precedente equazione come:

$$Stock(t) = \int_{t_0}^{t} [Inflow(s) - Outflow(s)]ds + Stock(t_0)$$

dove Inflow(s) rappresenta il valore dell'afflusso in qualsiasi momento s tra il tempo iniziale a e il tempo corrente t. Equivalentemente, il tasso netto di variazione di qualsiasi stock, la sua derivata, è l'afflusso meno il deflusso, che definisce l'equazione differenziale:

$$\frac{d(Stock)}{dt} = Inflow(t) - Outflow(t)$$

In generale, i flussi saranno funzioni dello stock e di altre variabili dello stato e dei parametri. I diagrammi di stock e flusso possono sembrare meno rigorosi delle rappresentazioni con equazioni integrali o differenziali, ma sono esattamente equivalenti e contengono le stesse informazioni. Da qualsiasi sistema di equazioni integrali o differenziali è possibile costruire la corrispondente mappa di stock e flusso, e viceversa.

Il concetto di stock è fondamentale nell'SD per i seguenti motivi:

1. Gli stock di un sistema dicono ai decisori dove si trovano, fornendo loro le informazioni necessarie per agire. Un pilota deve conoscere lo stato dell'aereo, compresa la posizione, la direzione, l'altitudine e il livello di carburante. Senza la conoscenza di questi stati, il pilota vola alla cieca e non sopravviverà a lungo. Allo stesso modo, un'azienda non può impostare il proprio programma di produzione in modo appropriato senza conoscere il portafoglio ordini, le scorte di magazzino, le scorte di ricambi, la forza lavoro e altre scorte. Il bilancio caratterizza la salute finanziaria di un'azienda riportando i valori delle scorte, come la liquidità, le scorte, i debiti e l'indebitamento. Le informazioni su queste scorte influiscono su decisioni quali l'emissione di nuovi debiti, il pagamento dei dividendi e il controllo delle spese attraverso i licenziamenti.

2. Le azioni forniscono ai sistemi inerzia e memoria.
   Le scorte accumulano eventi passati. Il contenuto di una scorta può cambiare solo a causa di un afflusso o di un

deflusso. Senza cambiamenti in questi flussi, l'accumulo passato nello stock persiste. Lo stock di piombo nella vernice delle case popolari americane rimane elevato anche oggi, nonostante la vernice al piombo sia stata vietata nel 1978. Una volta accumulato lo stock di vernice al piombo, l'unico modo per eliminarlo è quello di ricorrere a costosi interventi di deleading o alla demolizione delle abitazioni stesse. Anche in questo caso il piombo rimane, sequestrato in modo sicuro o più probabilmente disperso nell'ambiente sotto forma di polvere, trucioli o lisciviazione di piombo dalle discariche nelle riserve idriche. Allo stesso modo, lo stock di cloro che distrugge l'ozono generato dai CFC rimarrà nell'atmosfera per decenni anche dopo che il tasso di produzione dei CFC sarà sceso a zero, perché il tasso di eliminazione del cloro dalla stratosfera è molto basso. Le scorte non devono essere necessariamente tangibili. I ricordi e le convinzioni sono azioni che caratterizzano i vostri stati mentali. Le vostre convinzioni persistono nel tempo, generando inerzia e continuità nei vostri atteggiamenti e comportamenti. Se si ha una brutta esperienza con una compagnia aerea e non si vola più con quella compagnia, la convinzione della bassa qualità del servizio rimane anche se la compagnia è migliorata.

3. Le scorte sono la fonte dei ritardi.
   Tutti i ritardi coinvolgono le scorte. Un ritardo è un processo il cui output è in ritardo rispetto al suo input. La differenza tra l'input e l'output si accumula in uno stock di materiale in lavorazione. C'è un ritardo tra il momento in cui si spedisce una lettera e quello in cui la si riceve. Durante questo intervallo, la lettera risiede in uno stock di lettere in transito.

Anche la posta elettronica si accumula in uno stock di pacchetti e messaggi non consegnati che risiedono nella memoria dei vari computer tra il mittente e il destinatario. C'è un ritardo di diversi anni tra la decisione di costruire nuovi edifici per uffici e il momento in cui sono pronti per essere occupati. Durante questo intervallo esiste una linea di fornitura di edifici in fase di sviluppo, che comprende uno stock di progetti proposti e uno stock di edifici in costruzione. Per definizione, quando l'input di un ritardo cambia, l'output rimane indietro e continua al vecchio ritmo per qualche tempo. Durante questi aggiustamenti, lo stock che accumula la differenza tra input e output cambia.

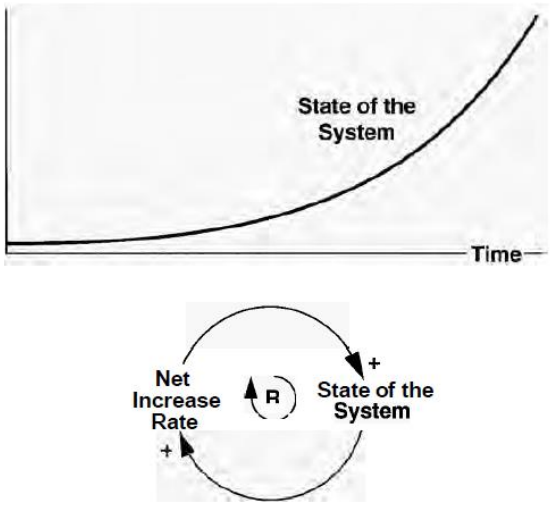
4. Le scorte disaccoppiano i tassi di flusso e creano dinamiche di disequilibrio. Le scorte assorbono le differenze tra afflussi e deflussi, permettendo così di differenziare gli afflussi e i deflussi di un processo. In equilibrio, l'afflusso totale di uno stock è uguale al suo deflusso totale, quindi il livello dello stock è immutabile. Tuttavia, i flussi in entrata e in uscita di solito differiscono perché spesso sono governati da processi decisionali diversi. Il disequilibrio è la regola piuttosto che l'eccezione.

1.1.3 I principali comportamenti dei sistemi

Il comportamento di un sistema dipernde dalla sua struttura, composta da feedback loops, stocks e flows e dalle non-linearità create dall'interazione tra gli agenti decisori che lavorano con essa. In questo capitolo vedremo i più comuni comportamenti che sono di interesse di studio dal nostro punto di vista.

1. La crescita esponenziale.

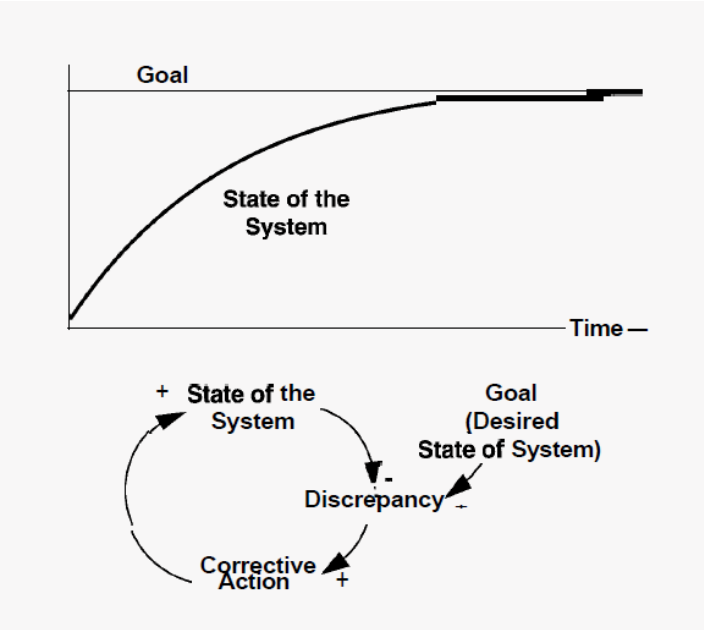
Come già presentata nell'esempio delle galline e delle uova una crescita esponenziale nasce da un feedback positivo (self-reinforcing). La crescita esponenziale ha come proprietà che il *doubling time* (letteralmente il tempo per raddoppiare) sia costante. Ovvero il tempo che ci mette un sistema per crescere da una unità a due è lo stesso tempo che impiega lo stesso sistema a passare da un milione di unità a due milioni.



(Business Dynamics, Stermann,2000)

2. Goal seeking

A differenza della crescita esponenziale, caratterizzata solo da postive loops, la presenza di negative loops portano il sistema a raggiungere un punto di equilibrio



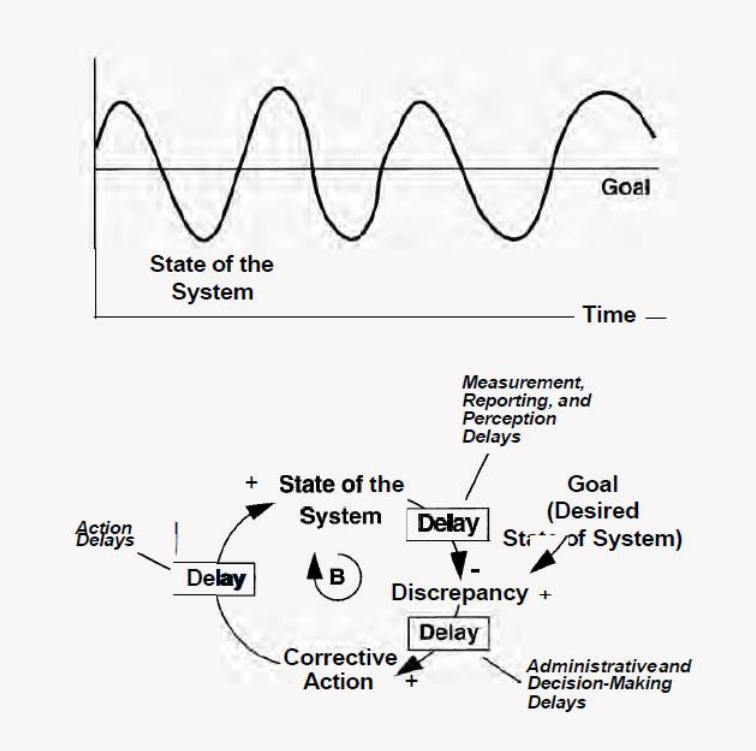
(Business Dynamics, Stermann,2000)

La peculiarità di questo ciclo è che più la discrepanza tra l'obiettivo e lo stato del sistema diminuisce, anche il tasso di crescita diminuisce.

3. Oscillazione

Come nel goal-seeking le oscillazioni sono causate dai feedback loops negativi. Lo stato del sistema, però non è stabile, quindi ci sarà un overshooting dell'obiettivo, per poi invertire la tendenza e andare in

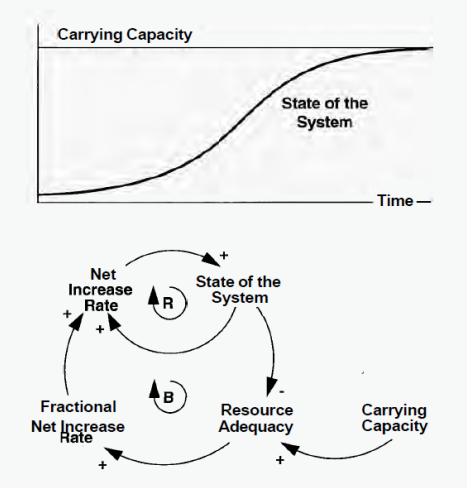
undershooting, continuando in eterno ad “oscillare” intorno al goal. Ciò accade per colpa di un delay nelle azioni correttive dello stato del sistema, cosa che invece non era presente nel caso del goal-seeking. Questo ritardo può essere causato ad esempio da un ritardo sistematico nella misurazione dello stato del sistema.

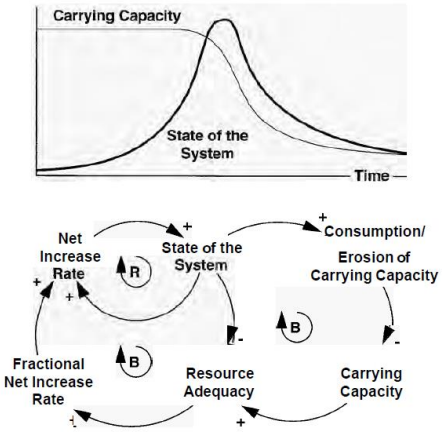


(Business Dynamics, Stermann,2000)

Un’ esempio di questo comportamento può essere osservato ad esempio nella crescita del PIL americano, dove la crescita reale oscilla intorno all’ obiettivo di crescita stabilito in modo irregolare.

La combinazione di questi tre comportamenti fondamentali porta a pattern più complessi. Tra i più importanti osserviamo la crescita S-shaped, e la overshoot con collasso.





(Business Dynamics, Stermann,2000)

1.1.4 I tipi di dati necessari

Come in tutti i modelli, è impossibile poter verificare che un modello sia valido o verificabile, difatti la domanda da porsi non è se il modello sia vero, ma se è utile allo scopo. Trovare il modello giusto che sia utile allo scopo è fondamentale.

Forrester identifica 3 tipi di dati necessari per sviluppare la struttura e le regole decisionali del modello: numerici, scritti e mentali.

I dati numerici sono quelli più familiari, quelli che possono essere trovati nei vari database, mentre i dati scritti includono procedure, grafici organizzatici, reports, e-mail e altri dati di archivio. I dati mentali invece comprendono tutte le informazioni nei modelli mentali delle persone come ad esempio le loro impressioni, le storie che raccontano, la loro comprensione del sistema e come le decisioni sono effettivamente prese. Questi ultimi non possono essere ottenuti come i precedenti due, ma tramite interviste, osservazioni e altri metodi. I dati numerici contengono solo una minima frazione di informazione scritta nei database, portando i dati mentali ad essere cruciali nella capacità descrittiva del mondo. Tutti i dati a nostra disposizione, quindi, risultano distorti. È importante usare il miglior giudizio possibile e i migliori metodi statistici per stimare i parametri.

1.1.5 Applicazioni principali della metodologia

System Dynamics è una metodologia versatile, utilizzata nei più svariati settori. Difatti capire quali sono le cause e gli effetti di un sistema complesso è utile in quasi tutti gli ambiti di ricerca. Tra i più rilevanti di nostro interesse possiamo trovare:

Gestione aziendale e strategica: la metodologia viene impiegata per modellare processi aziendali, supportare la pianificazione strategica e migliorare le performance organizzative. Ad esempio, è possibile analizzare dinamiche di mercato e della concorrenza, ottimizzare la gestione delle scorte o addirittura valutare l'impatto di nuove politiche aziendali (ISBN:978 88 217 3002 3);

Economia e Politiche Pubbliche: System Dynamics può essere utilizzata per analizzare le dinamiche economiche e quindi supportare formulazione di politiche pubbliche efficaci, tramite ad esempio studio dei cicli economici e delle fluttuazioni del mercato, valutare l'impatto di politiche fiscali e monetarie, analizzare dinamiche occupazionali e il mercato del lavoro.

Ingegneria e progettazione di sistemi: può essere applicata nella progettazione e nell'ottimizzazione di sistemi complessi in ambito di ingegneria, simulando ad esempio sistemi meccanici e dinamiche multicorpo,

Logistica: forse il settore di maggiore applicazione, permette infatti di supportare l'analisi e l'ottimizzazione dei processi logistici e della supply chain. Può essere usata per supportare la gestione delle scorte e l'ottimizzazione dei livelli di inventario, analizzare i flussi logistici e delle reti di distribuzione, valutare l'impatto di strategie logistiche sulle performance aziendali.

Portando questi esempi concreti, si evince come la metodologia System Dynamics sia versatile nell'affrontare problematiche complesse nei più svariati settori, offrendo strumenti utili per l'analisi, la simulazione e il supporto decisionale.

1.1.6 Sfide per il futuro

Negli ultimi anni sono stati fatti passi da gigante nella teoria della dinamica non lineare. Esistono infatti numerosi articoli e studi al riguardo, grazie alle nuove innovazioni tecnologiche degli ultimi due decenni. Tuttavia, rimangono ancora delle sfide aperte:

- Mappatura automatica dello spazio dei parametri. Premendo il pulsante di esecuzione si genera una mappa ad alta risoluzione del comportamento del modello su un intervallo di parametri e condizioni iniziali specificato dall'utente.

- Analisi automatica della sensibilità. Il software identifica automaticamente le politiche ad alta leva e i parametri più influenti del modello (in relazione ai criteri specificati dall'utente).

- Test automatizzato delle condizioni estreme. Il modellatore o il cliente specificherà le condizioni estreme che il modello deve soddisfare (ad esempio, assenza di manodopera, assenza di produzione; assenza di scorte, assenza di spedizioni). Il software di simulazione implementerà automaticamente questi “controlli di realtà” e ne riporterà i risultati (vedi Peterson e Eberlein 1994).

- Stima automatica e interattiva dei parametri, calibrazione e ottimizzazione delle politiche. I modellatori possono ora specificare criteri e pesi (eventualmente non lineari) per diverse variabili e vincoli sui valori plausibili dei parametri; il software trova quindi i migliori valori dei parametri e i limiti di confidenza intorno ai valori stimati (utilizzando metodi che vanno dai minimi quadrati ordinari al

filtraggio di Kalman). Tuttavia, il processo è spesso noioso ed è difficile combinare informazioni quantitative e qualitative. I futuri software di simulazione automatizzeranno la calibrazione del modello, consentendo all'utente di esplorare le conseguenze di criteri alternativi in tempo reale. Analogamente, l'ottimizzazione dei modelli dinamici di sistema ha una lunga storia (si veda, ad esempio, Coyle 1985, 1998). I software attualmente disponibili consentono all'utente di specificare una funzione obiettivo multiattributo (possibilmente non lineare) e un insieme di strumenti politici (parametri), per poi cercare nello spazio delle policy la soluzione ottimale. In futuro, questo processo sarà così veloce da svolgersi in modo interattivo e in tempo reale, con l'utente che riceverà un feedback immediato sulle conseguenze di funzioni obiettivo e strumenti di policy alternativi.

- Identificazione automatica dei loop dominanti e della struttura di retroazione. Esistono oggi diversi metodi per identificare i loop dominanti in qualsiasi punto della simulazione, quantificare il contributo di qualsiasi parametro o loop a una determinata modalità e mostrare come le non linearità modifichino la struttura di retroazione dominante (si vedano, ad esempio, Eberlein 1989; Kampmann 1996; Mojtahedzadeh 1997; e N. Forrester 1982). I futuri software di simulazione automatizzeranno e velocizzeranno questi calcoli, in modo che gli utenti possano ottenere un feedback immediato che mostri i loop dominanti nel sistema e come l'influenza dei parametri e delle strutture di retroazione cada e diminuisca con lo svolgersi della dinamica.

- Aiuto automatizzato. Grazie a sistemi esperti, software di riconoscimento dei modelli e altri strumenti di intelligenza

artificiale, il software di simulazione dovrebbe presto essere in grado di fungere da tutor e guida automatica per la costruzione di modelli. Il tutor di modellazione potrebbe, ad esempio, esaminare ogni equazione inserita per verificarne la coerenza dimensionale, la robustezza in condizioni estreme e altri principi fondamentali di una buona formulazione. Il tutor suggerirebbe quindi formulazioni migliori da una libreria di strutture standard, collegamenti a modelli e letteratura correlati e parametri ragionevoli. Immaginate di costruire, ad esempio, un modello di marketing. Il software potrebbe dirvi che la vostra formulazione per la defezione dei clienti verso i concorrenti manca di feedback da parte dello stock di clienti attuali (permettendo allo stock di clienti di diventare negativo), suggerirvi una formulazione migliore e guidarvi attraverso una serie di domande per aiutarvi a specificare i parametri e le relazioni non lineari. Allo stesso modo, il software può suggerire test da condurre e suggerimenti per aiutarvi a comprendere meglio il comportamento del vostro modello.

1.2 Intelligenza Artificiale (AI)

L'intelligenza artificiale (IA) è l'abilità di una macchina di mostrare capacità umane quali il ragionamento, l'apprendimento, la pianificazione e la creatività.

L'intelligenza artificiale permette ai sistemi di capire il proprio ambiente, mettersi in relazione con quello che percepisce e risolvere problemi, e agire verso un obiettivo specifico. Il computer riceve i dati (già preparati o raccolti tramite sensori, come una videocamera), li processa e risponde.

I sistemi di IA sono capaci di adattare il proprio comportamento analizzando gli effetti delle azioni precedenti e lavorando in autonomia.

1.2.1 Storia

L'inizio dello sviluppo della IA come dottrina scientifica inizio nei primi anni '50. Nel 1956, al Darmouth College, nel New Hampshire, si tenne un convegno nel quale presero parte i maggiori esponenti dell'informatica. In quell'occasione si raccolsero i principali contributi sul tema, ponendo anche l'attenzione sugli sviluppi futuri. Durante il convegno, salì alla ribalta Alan Turing, considerato uno dei padri dell'informatica moderna.

Già nel 1936 Alan Turing aveva posto le basi per i concetti di calcolabilità, computabilità e per la macchina di Turing. Nel 1950 scrisse un articolo intitolato Computing machinery and intelligence, in cui teorizzava quello che sarebbe divenuto noto come test di Turing. Il test affermava che una macchina poteva essere considerata intelligente se il suo comportamento, osservato da un essere umano, fosse considerato indistinguibile da quello di una persona.

Grazie a Turing, l'Intelligenza Artificiale ricevette grande attenzione da parte della comunità scientifica e nacquero diversi approcci. Tra i più importanti possiamo ricordare la logica matematica, per la dimostrazione di teoremi e l'inferenza di nuova conoscenza, e le reti neurali. Negli ultimi anni la tecnologia di queste reti è stata implementata e oggi vengono applicate nell'ambito del Deep Learning, un ramo del Machine Learning.

Nei successivi anni l'aspettativa iniziò a crescere. Tuttavia, poiché i macchinari dell'epoca non disponevano di una capacità computazionale adeguata, queste promesse non furono mantenute e ciò portò alla frammentazione dell'Intelligenza Artificiale in distinte aree basate su teorie

diverse. In quel contesto emersero due paradigmi principali: Intelligenza Artificiale Forte e Intelligenza Artificiale Debole.

La prima è stata snobbata dalla comunità scientifica, mentre la seconda è stata oggetto di studi e ricerca. L' IA debole è una intelligenza artificiale che non ha coscienza di sé, ma che è capace di ragionare su problemi complessi e risolverli.

Basandosi sul paradigma dell'Intelligenza Artificiale Debole, a partire dagli anni Ottanta sono state sviluppate le prime applicazioni in ambito industriale. In particolare, la prima Intelligenza Artificiale applicata in ambito commerciale fu R1, sviluppata nel 1982 dall'azienda Digital Equipment per configurare gli ordini di nuovi computer. Questa implementazione permise all'azienda di risparmiare successivamente 40 milioni di dollari all'anno.

Oggi l'Intelligenza Artificiale è uno dei principali ambiti di interesse della comunità scientifica informatica, se non il maggiore, con temi di ricerca come il Machine Learning, l'elaborazione del linguaggio naturale, l'AI Generativa (ad esempio ChatGPT) e la robotica.

Approfondiremo ora le principali tecniche IA che ci torneranno utili per l'integrazione con i modelli SD.

1.2.2 Machine learning:

Il Machine Learning (ML) è una branca dell'Intelligenza Artificiale (IA) che si concentra nello sviluppo di algoritmi e modelli statistici che permettono ai sistemi informatici di apprendere dai dati e migliorare le proprie prestazioni senza essere esplicitamente programmati.

Esistono diverse tipologie di machine learning:

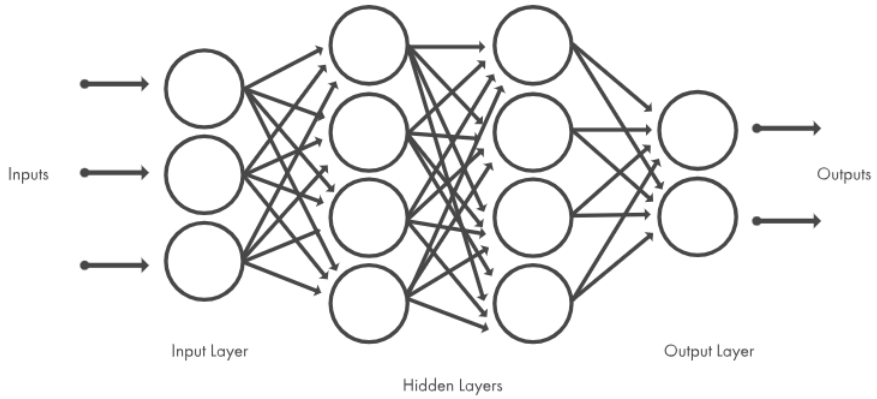
Apprendimento Supervisionato: Gli algoritmi vengono addestrati su un dataset etichettato, dove ogni input è associato a un output desiderato. Questo approccio è utilizzato per compiti come la classificazione e la regressione.

Apprendimento Non Supervisionato: Gli algoritmi lavorano su dati non etichettati e cercano di identificare pattern o strutture nascoste. Tecniche comuni includono il clustering e l'analisi delle componenti principali.

Apprendimento per Rinforzo: Un agente interagisce con un ambiente dinamico e apprende a compiere azioni per massimizzare una ricompensa cumulativa. Questo metodo è spesso applicato in ambiti come la robotica e i giochi.

1.2.3 Deep Learning:

Il Deep Learning è una sottocategoria dell'Intelligenza Artificiale (IA) e del Machine Learning (ML) che si concentra nell'uso di reti neurali artificiali con molteplici strati per analizzare e apprendere da grandi quantità di dati complessi. Questa struttura permette al modello di elaborare i dati attraverso livelli successivi di astrazione, aumentando la capacità di riconoscere pattern complessi.

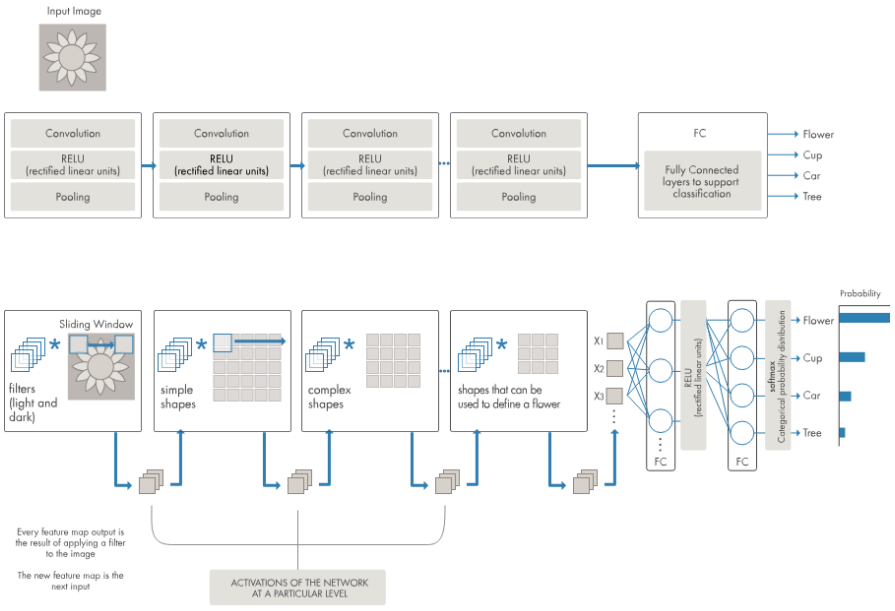


Schema generico di una struttura DL

I modelli di Deep Learning vengono addestrati usando grandi set di dati etichettati e spesso sono in grado di apprendere feature direttamente dai dati, senza dover ricorrere all'estrazione manuale delle feature stesse. Sebbene la prima rete neurale artificiale sia stata teorizzata nel 1958, il Deep Learning richiede un potere computazionale molto alto che è divenuto disponibile a partire dagli anni 2000.

Esistono tre tipi di modelli di Deep Learning, ovvero le reti neurali convoluzionali (CNN), le reti neurali ricorrenti (RNN) e i modelli di trasformatori.

Una Convolutional Neural Network (CNN) analizza i dati di input applicando operazioni di convoluzione per estrarre caratteristiche significative. Grazie ai layer convoluzionali 2D, questa architettura è particolarmente efficace nell'elaborazione di dati bidimensionali, come le immagini. Durante la fase di addestramento su un set di immagini, la rete apprende automaticamente le caratteristiche rilevanti, eliminando la necessità di un'estrazione manuale delle feature. Questo processo consente alle CNN di ottenere elevate prestazioni nei compiti di classificazione delle immagini. Oltre all'elaborazione visiva, queste reti possono essere applicate anche ad altri tipi di dati, come serie temporali e testo, rendendole versatili in diversi ambiti dell'intelligenza artificiale.



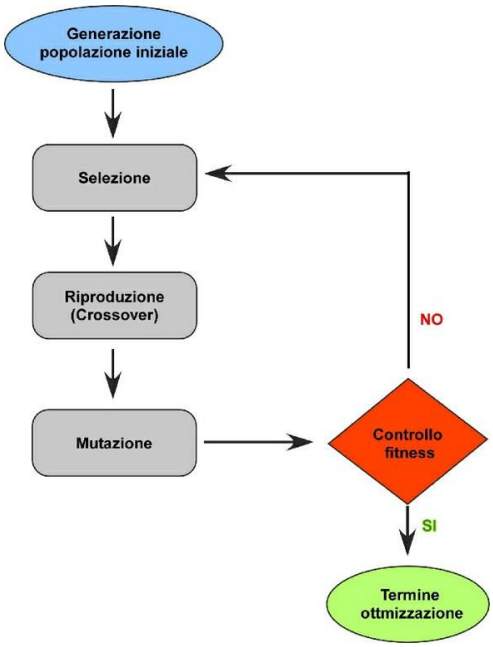
Visualizzazione di un esempio di rete neurale convoluzionale, (Fonte: https://it.mathworks.com/discovery/deep-learning.html)

Una Rete Neurale Ricorrente (RNN) è un'architettura di Deep Learning progettata per analizzare e prevedere dati sequenziali o appartenenti a serie temporali. Grazie alla sua capacità di gestire sequenze di lunghezza variabile, risulta particolarmente efficace in applicazioni che coinvolgono segnali naturali, elaborazione del linguaggio e analisi video. Un'estensione avanzata delle RNN è la Long Short-Term Memory (LSTM), una rete progettata per superare le limitazioni delle RNN standard, migliorando la capacità di apprendere dipendenze a lungo termine nei dati sequenziali.

1.2.4 Algoritmi evolutivi e ottimizzazione:

Gli algoritmi evolutivi sono una classe di metodi stocastici di ottimizzazione ispirati ai processi di evoluzione naturale, come la selezione naturale e la genetica. Questi algoritmi simulano meccanismi evolutivi per risolvere problemi complessi di ottimizzazione, adattandosi progressivamente alle soluzioni migliori attraverso iterazioni successive.

Un algoritmo genetico è un algoritmo euristico usato per risolvere problemi di ottimizzazione con ordine di complessità NP. Si chiamano algoritmi genetici perché applicano dei meccanismi simili a quelli dei processi chimici dell'evoluzione nella genetica. Sono degli algoritmi che valutano diverse soluzioni di partenza, le ricombinano e aggiungendo degli elementi di disordine producono nuove soluzioni che vengono valutate scegliendo le migliori, raggiungendo soluzioni di ottimo.



Schema di un algoritmo genetico

1.2.5 Reinforcement Learning:

Il Reinforcement Learning (RL) è una metodologia di Machine Learning in cui un sistema intelligente, chiamato agente, apprende a eseguire un compito interagendo ripetutamente con un ambiente dinamico attraverso un processo di tentativi ed errori (trial-and-error). Questo metodo consente all'agente di prendere decisioni successive in modo autonomo, con l'obiettivo di massimizzare una ricompensa, senza la necessità di una programmazione esplicita o di un intervento diretto da parte dell'uomo.

Il Reinforcement Learning (RL) è una sottocategoria del Machine Learning che si distingue dagli approcci supervisionati e non supervisionati per il suo metodo di apprendimento. A differenza di questi ultimi, il RL non si basa su un dataset statico, ma opera in un ambiente dinamico, apprendendo direttamente dalle esperienze acquisite durante l'interazione con l'ambiente stesso. Durante il processo di addestramento, l'agente raccoglie informazioni attraverso un sistema di tentativi ed errori (trial-and-error), adattando progressivamente il proprio comportamento. Questo approccio riduce la necessità di operazioni preliminari come la raccolta, la preelaborazione e l'etichettatura dei dati, che sono invece fondamentali negli altri metodi di apprendimento automatico. In sostanza, con un adeguato sistema di ricompensa, un modello di Reinforcement Learning può sviluppare autonomamente schemi decisionali, senza richiedere una supervisione umana diretta.

Il processo di addestramento nel Reinforcement Learning (RL) riflette molte dinamiche del mondo reale. Un esempio pratico è l'addestramento di un animale domestico attraverso il rinforzo positivo. In questo

contesto, il cane rappresenta l'agente, mentre l’ambiente in cui si muove comprende sia il contesto circostante sia la persona che lo addestra.

Il processo di apprendimento inizia quando l’addestratore fornisce un comando o un segnale, che il cane percepisce (osservazione). In risposta, l'animale esegue un’azione. Se questa azione si avvicina al comportamento desiderato, riceverà una ricompensa (come un croccantino o un gioco); in caso contrario, non otterrà alcun premio.

Durante le prime fasi dell’addestramento, il cane potrebbe rispondere in modo casuale, eseguendo movimenti non pertinenti, come rotolarsi quando gli viene chiesto di sedersi. Questo avviene perché sta ancora cercando di associare determinati stimoli a specifiche azioni e alle rispettive ricompense. Questo processo di apprendimento porta alla costruzione di una politica, ovvero una strategia che guida l’agente nell’associare osservazioni a risposte adeguate.

L’obiettivo finale dell’addestramento con il Reinforcement Learning è affinare questa politica in modo che l’agente (il cane) apprenda il comportamento ottimale per massimizzare la ricompensa. Alla fine del percorso di apprendimento, il cane dovrebbe essere in grado di interpretare correttamente i segnali e agire di conseguenza, ad esempio sedendosi quando gli viene dato il comando "seduto". A quel punto, anche se i premi rimangono graditi, non saranno più essenziali per eseguire correttamente il comportamento appreso.

Un aspetto importante da considerare nel Reinforcement Learning (RL) è la sua bassa efficienza nell’uso dei campioni (sample efficiency). Questo significa che, per addestrare efficacemente un modello, è

necessario un elevato numero di interazioni tra l'agente e l'ambiente, al fine di raccogliere dati sufficienti per l'apprendimento.

Ad esempio, AlphaGo, il primo sistema AI in grado di sconfiggere un campione mondiale di Go, ha richiesto un addestramento intensivo e ininterrotto per diversi giorni, durante i quali ha disputato milioni di partite, accumulando una conoscenza equivalente a millenni di esperienza umana. Anche in applicazioni più semplici, i tempi di addestramento possono variare da pochi minuti a diverse ore o giorni, a seconda della complessità del problema.

Inoltre, la definizione del problema nel RL può risultare complessa, poiché richiede una serie di scelte di progettazione e più iterazioni prima di ottenere risultati soddisfacenti. Tra gli aspetti critici da ottimizzare vi sono:

Selezione dell'architettura della rete neurale più adatta al problema.

Regolazione degli iperparametri, che influenzano le prestazioni e la velocità di apprendimento.

Definizione del segnale di ricompensa, essenziale per guidare il comportamento dell'agente verso l'obiettivo desiderato.

Di conseguenza, il successo del Reinforcement Learning dipende non solo dalla potenza computazionale, ma anche dalla capacità di modellare correttamente il problema e ottimizzare i vari elementi del processo di apprendimento.

L'applicazione del Reinforcement Learning (RL) segue un processo strutturato in cinque fasi principali, che guidano la creazione, l'addestramento e la validazione dell'agente.

1. Definizione dell'ambiente

Il primo passo consiste nel progettare l'ambiente in cui opererà l'agente RL, stabilendo l'interfaccia che regolerà le interazioni tra i due. L'ambiente può essere sia un modello simulato sia un sistema fisico reale, ma generalmente si preferisce iniziare con simulazioni, poiché offrono un ambiente di test più sicuro e permettono di sperimentare senza rischi.

2. Impostazione della ricompensa

Successivamente, è fondamentale definire il segnale di ricompensa, che misura le prestazioni dell'agente rispetto agli obiettivi assegnati. La progettazione di un sistema di ricompensa efficace può risultare complessa e potrebbe richiedere diverse iterazioni per perfezionarlo e ottenere il comportamento desiderato.

3. Creazione dell'agente

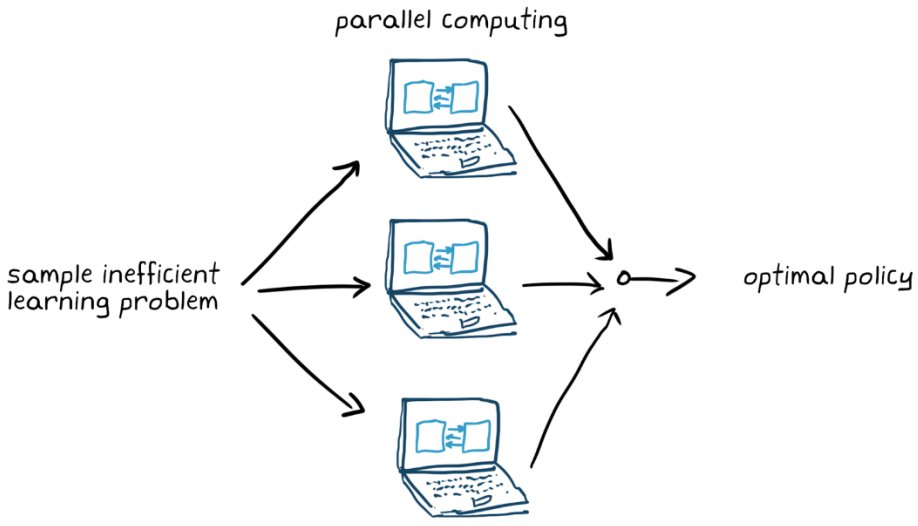
A questo punto, si procede con la realizzazione dell'agente, che include la sua politica e l'algoritmo di apprendimento. Questo passaggio prevede:

- Scelta della rappresentazione della politica, che può essere implementata tramite reti neurali o tabelle di look-up.
- Selezione dell'algoritmo di addestramento più adatto al problema. Molti algoritmi RL moderni utilizzano reti neurali, poiché permettono di gestire spazi di stato e azione complessi.

4. Addestramento e validazione

L’addestramento dell’agente prevede la configurazione di diversi parametri di apprendimento, come i criteri di arresto, e l’ottimizzazione della politica attraverso interazioni ripetute con l’ambiente. Una volta completato l’addestramento, è essenziale verificare la validità della politica ottenuta. Se i risultati non sono soddisfacenti, si possono rivedere aspetti critici come il segnale di ricompensa o l’architettura della politica, per poi ripetere il processo.

Poiché il Reinforcement Learning non è particolarmente efficiente nell'uso dei dati (sample efficiency), il tempo di addestramento può variare da pochi minuti a diversi giorni, a seconda della complessità dell’applicazione. Nei problemi più avanzati, l’utilizzo del calcolo parallelo su CPU, GPU o cluster di computer accelera il processo, facilitando la ricerca della politica ottimale.



Schema di parallel computing (Fonte: https://it.mathworks.com/discovery/reinforcement-learning.html)

5. Distribuzione della politica

L'ultima fase consiste nel distribuire la politica addestrata, implementandola attraverso codice ottimizzato, ad esempio utilizzando CUDA o C/C++. A questo punto, la politica diventa un sistema decisionale autonomo, in grado di operare senza bisogno di ulteriore addestramento.

Poiché il Reinforcement Learning è un processo iterativo, le scelte fatte nelle fasi finali possono rendere necessario tornare a una fase precedente per migliorare le prestazioni dell'agente. Se l'addestramento non portasse alla convergenza su una politica efficace in un tempo ragionevole, potrebbe essere necessario apportare modifiche a uno o più elementi del modello prima di eseguire un nuovo ciclo di apprendimento. Tra gli aspetti da rivedere vi sono:

- Parametri di addestramento, per migliorare l'efficienza del processo.
- Configurazione dell'algoritmo RL, adattando le impostazioni per una migliore convergenza.
- Struttura della politica, scegliendo una rappresentazione più efficace.
- Definizione del segnale di ricompensa, affinché guidi meglio il comportamento dell'agente.
- Osservazioni e azioni, regolando il modo in cui l'agente percepisce e interagisce con l'ambiente.
- Dinamiche dell'ambiente, per riflettere con maggiore accuratezza le condizioni reali in cui il sistema opererà.

Questo approccio iterativo assicura che l'agente di Reinforcement Learning possa adattarsi e migliorare progressivamente fino a raggiungere le prestazioni desiderate.

2. Metodologia di ricerca

In questo capitolo approfondiremo la metodologia applicata nella ricerca di letteratura relativa al nostro scopo, spiegando come sono stati selezionati gli articoli pertinenti, quali database sono stati consultati e quali sono stati i criteri di selezione.

2.1 Fase di ricerca

Per identificare e selezionare gli articoli pertinenti alla causa della revisione ho utilizzato la metodologia PRISMA (Preferred Reporting Items for Systematic reviews and Meta-Analyses), definita appositamente per facilitare la trasparenza e la completezza delle revisioni sistematiche.

La metodologia ha una direttiva pubblicata sul proprio sito, dove vengono fornite un insieme di linee guida da seguire nella fase di ricerca.

Seguendo le direttive, troviamo come prima fase l'identificazione degli articoli potenzialmente rilevanti al nostro scopo. Dopo questa prima fase iniziale si esegue una fase di screening, dove vengono rimossi tutti gli articoli duplicati e dove vengono letti gli abstract per una prima scrematura. Successivamente avviene una lettura completa degli articoli per verificarne l'effettiva pertinenza rispetto ai nostri criteri di inclusione/esclusione, per poi fare uno screening finale dove creiamo una lista di studi da includere nella ricerca.

Per questa ricerca, il database accademico principalmente usato è stato Scopus, anche se è stato integrato da altre fonti più specializzate nel settore, dato che la presenza di articoli rilevanti allo scopo della nostra ricerca, era scarsa, per questo sono stati presi in considerazione tutti gli articoli trovabili a tal riguardo.

Successivamente è stato usato l'archivio della System Dynamics Society, dove ho applicato altri metodi di ricerca.

Ho applicato come filtri la ricerca per titolo, includendo tutte le metodologie IA che potrebbero essere applicate nel caso di specie

Type of Algorithm	Name of Algorithm	Application
Supervised Learning	Naive Bayes	Predict project risks
Supervised Learning	Logistic Regression	Classify tasks as 'on track' or 'at risk'
Supervised Learning	K-Nearest Neighbor (KNN)	Find similar past projects
Supervised Learning	Random Forest	Forecast project success probabilities
Supervised Learning	Support Vector Machine (SVM)	Categorize tasks by priority
Supervised Learning	Decision Tree	Simplify decision-making for milestones
Supervised Learning	Simple Linear Regression	Predict costs over time
Supervised Learning	Multivariate Regression	Estimate outcomes based on multiple factors
Supervised Learning	Lasso Regression	Focus on critical project variables
Unsupervised Learning	K-Means Clustering	Group similar team members
Unsupervised Learning	DBSCAN Algorithm	Identify bottlenecks in team performance
Unsupervised Learning	Principal Component Analysis	Reduce task complexity
Unsupervised Learning	Independent Component Analysis	Spot hidden patterns in timelines
Unsupervised Learning	Frequent Pattern Growth	Detect recurring task combinations
Unsupervised Learning	Apriori Algorithm	Uncover dependencies between tasks
Unsupervised Learning	Z-Score Algorithm	Identify outlier tasks disrupting timelines
Semi-Supervised Learning	Self-Training	Predict outcomes using partially labeled data
Semi-Supervised Learning	Co-Training	Merge data sources for resource predictions
Reinforcement Learning	Policy Optimization	Improve workflows by learning from past mistakes
Reinforcement Learning	Q-Learning	Adjust schedules for efficiency
Reinforcement Learning	Learn the Model	Adapt to changing project needs
Reinforcement Learning	Given the Model	Create optimized plans using known patterns

Lista di algoritmi cercati.

Oltre ad articoli di letteratura, su questa fonte, sono riuscito anche a trovare delle presentazioni di conferenza a riguardo.

2.2 Criteri di selezione

Il tema centrale della ricerca è stato: *Come è possibile implementare le nuove metodologie IA nell'ambito del System Dynamics?*

A questo scopo ho sottoposto Scopus a diverse query avanzate, più o meno specifiche, per trovare degli articoli a riguardo:

```
TITLE-ABS-KEY(("Naive Bayes" OR "Logistic Regression" OR "K-Nearest Neighbor" OR "Random Forest" OR
"Support Vector Machine" OR "Decision Tree" OR "Linear Regression" OR "Multivariate Regression" OR "Lasso
Regression" OR "K-Means Clustering" OR "DBSCAN" OR "Principal Component Analysis" OR "Independent
Component Analysis" OR "Frequent Pattern Growth" OR "Apriori Algorithm" OR "C4.5 Algorithm" OR "Self-
Training" OR "Co-Training" OR "Policy Optimization" OR "Q-Learning" OR "Reinforcement Learning") AND
("System Dynamics" OR "Dynamic Systems" OR "Project Dynamics" OR "Complex Systems")) AND PUBYEAR >
2018 AND DOCTYPE("ar") AND SUBJAREA("ENGI" OR "COMP" OR "MATH") AND ( LIMIT-TO (
EXACTKEYWORD,"System Dynamics" ) )
```

Questo è un esempio di query applicata nella ricerca. Possiamo notare come includa tutti gli algoritmi di Machine learning interessati nell'implementazione in SD, concetti chiave come System Dynamics, Dynamic Systems, Project Dynamics e Complex Systems. Inoltre, ho inserito dei filtri come anno di pubblicazione dopo il 2018, documenti per articoli di ricerca, e come materia di riferimento ingegneria informatica e matematica. Successivamente, a causa di scarsi risultati, ho allargato la ricerca anche ad articoli con anno di pubblicazione generico

Inoltre, ho posto come parole chiave System *Dynamics* e *Artificial Intellience.*

Non è stato possibile invece utilizzare la tecnica dello snowbailing dal primo campione di studi trovati, dato che come precedentemente affermato, gli articoli a riguardo del tema sono estremamente ridotti.

3. Risultati della revisione

3.1 Analisi degli articoli selezionati

Come precedentemente anticipato, purtroppo la letteratura non supporta a pieno questo argomento, dato che è un campo molto applicativo e soprattutto innovativo, difatti esistono molte issue aperte che sono in cerca di soluzione.

Tuttavia, sono riuscito ad ottenere dei risultati rilevanti nella ricerca.

Il corpus da me esaminato consiste di 14 articoli e una presentazione di conferenza, ritenuta attendibile e presentata in una conferenza del system dynamics society.

Gli argomenti sono così composti:

Possiamo notare come le date di pubblicazioni degli articoli sono in crescita negli anni, dato che è un argomento di interesse crescente nel tempo. Inoltre, possiamo notare come, ad oggi, anche se in modo impercettibile,

l’argomento viene trattato in modo più pratico che teorico, presentando quindi dei modelli veri e propri già sviluppati, pronti per essere integrati con le nuove metodologie di IA.

I contenuti di questi articoli sono tra i più disparati, e trattano tutti l’applicazione di algoritmi, tecnologie o metodologie innovative nei più disparati campi, dalla finanza all’impatto ambientale dell’impronta carbonica. Entrerò ora nel dettaglio di queste nuove innovazioni, cercando di risassumere in modo critico e rigoroso i contenuti presentati da questi articoli.

3.2 Analisi delle metodologie individuate

3.2.1 Random forest per la stima dei parametri

La prima metodologia che andremo ad analizzare è l'applicazione del Machine learning (in particolare l'algoritmo *random forest*) per stimare le variabili regolatorie di un modello SD.

In un modello SD, infatti, è difficile determinare i parametri delle equazioni differenziali che governano il sistema, per diversi motivi: sono difficili da determinare manualmente, dipendono da più variabili con interazioni non lineari e, soprattutto, possono essere influenzate da bias soggettivi nella loro selezione.

Ci sono diversi modi di applicare l'algoritmo *random forest* per risolvere questo problema, ma in una buona parte degli articoli selezionati, ci sono delle fasi. Inizialmente deve essere definito il modello SD, definendo stock, flow, feedback loops e ritardi, scrivendo le equazioni differenziali che caratterizzano il sistema. Successivamente vanno individuati i parametri chiave da stimare, ovvero le variabili che influenzano il comportamento del sistema.

Nello step successivo bisogna creare un dataset storico, ovvero raccogliere dati reali su variabili rilevanti per il sistema, per poi "pulirli", cioè trasformare i dati per garantirne l'omogeneità.

Deve essere poi addestrato un modello Random Forest Regressor dove come input avremo le variabili indipendenti storiche, mentre come output avremo i valori previsti dei parametri da stimare nel modello SD. Vengono poi selezionate le variabili con il maggiore peso nella previsione dei parametri SD e rimosse quelle meno influenti.

L’uso di questo algoritmo, oltre a risolvere i problemi sopracitati, riesce anche a migliorare la precisione dei parametri, permettendo simulazioni più realistiche.

3.2.2 Applicazione di GPT-4 per sviluppare modelli SD

Un'altro approccio interessante dal punto di vista della modellazione SD è lo studio della recente tecnologia GPT-4 per lo sviluppo dei modelli SD. GPT-4 (*Generative Pre-trained Transformer 4*) è un modello avanzato di Intelligenza Artificiale sviluppato da OpenAI. Si basa su reti neurali di tipo Transformer ed è progettato per comprendere e generare testo in un linguaggio naturale con un livello di precisione ed elaborazione molto avanzato. È la tecnologia su cui si basa il famoso ChatGPT.

Questa tecnologia è stata testata in uno studio, cercando di capire se potesse supportare la costruzione e l'espansione dei modelli SD, identificare errori e suggerire miglioramenti nei modelli, convertire modelli SD in codice Python per simulazioni computazionali, e valutare limiti e potenzialità dell'IA nel processio di modellazione.

Per testare l'utilità di GPT-4 nella modellazione SD sono stati condotti una serie di esperimenti strutturati in queste fasi:

- Viene chiesto a GPT-4 di generare un modello SD semplice, ad esempio nel caso di un articolo la crescita della popolazione, assicurando condizioni di equilibrio iniziale.
- Successivamente si prova ad espandere il modello con nuove variabili, monitorando come GPT-4 ha gestito i nuovi parametri e se le nuove variabili siano attendibili.
- Viene chiesto a GPT-4 di individuare e correggere eventuali errori nei parametri e nella logica del modello.
- Si richiede di generare idee per espandere la simulazione, con nuove variabili, stock, flussi o eventuali ritardi.
- Alla fine, viene chiesto al sistema di tradurre la simulazione in codice Python, permettendone l'integrazioni con altri sistemi.

La prestazione dell'IA alla fine di questi test, è riuscita a generare un modello base funzionante con condizioni di equilibrio corrette, ha espanso correttamente il modello, è riuscito a stabilizzare la simulazione regolando il rapporto tra stock ed è riuscita a proporre delle idee di espansione valide. Tuttavia, ha riscontrato difficoltà nell'individuare eventuali errori, ed ha generato a volte delle risposte errate o incomplete, da questo capiamo che GPT-4 ha bisogno di prompt chiari e iterazioni multiple per migliorare i risultati.

Sicuramente questa metodologia ci permette di accelerare il processo di modellazione, facilita l'apprendimento per i nuovi modellatori ed è molto comoda per convertire il modello direttamente in codice Python, senza dover riprogrammare da 0, ma comunque non può ancora sostituire l'expertise umana a causa a volte della sua imprecisione. Delle soluzioni a questo problema nel futuro potranno essere lo sviluppo di IA specifiche nello sviluppo di modellazione SD, e studiare l'uso di GPT-4 per la generazione automatica di diagrammi causali.

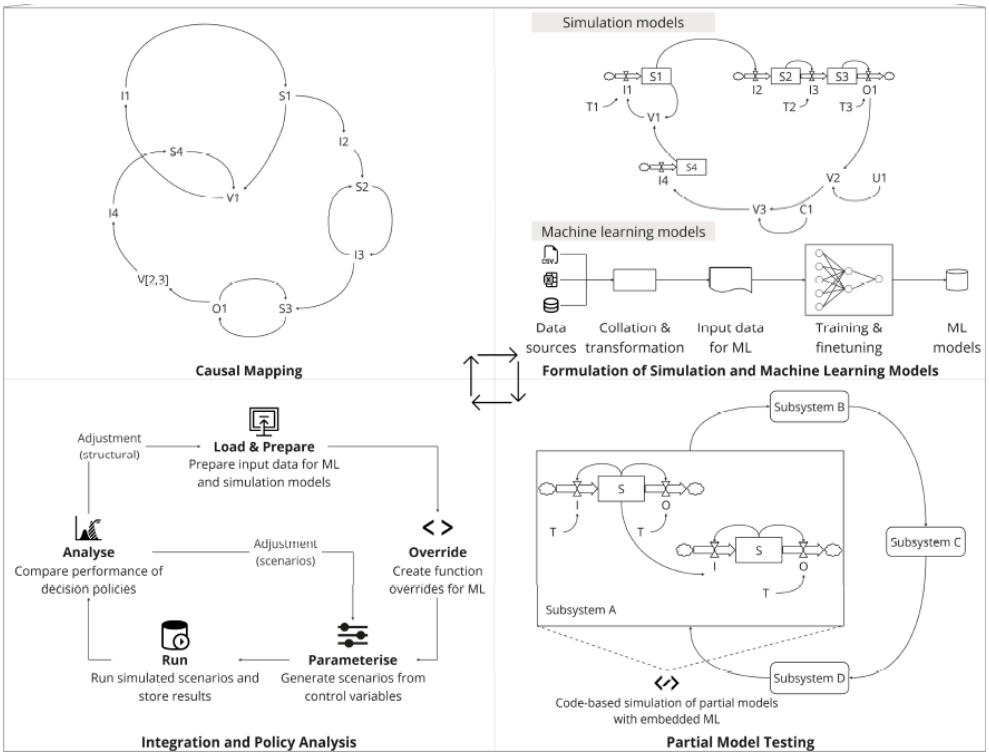
3.2.3 Caso studio New York Citi Bike

In questo studio viene proposto un esempio pratico di modello dinamico per valutare e migliorare le decisioni prese congiuntamente da esseri umani e sistemi di machine learning (ML). Risulta molto interessante ai fini della nostra ricerca, dato che è una prima applicazione pratica di un'interazione tra un modello SD e le tecniche di IA.

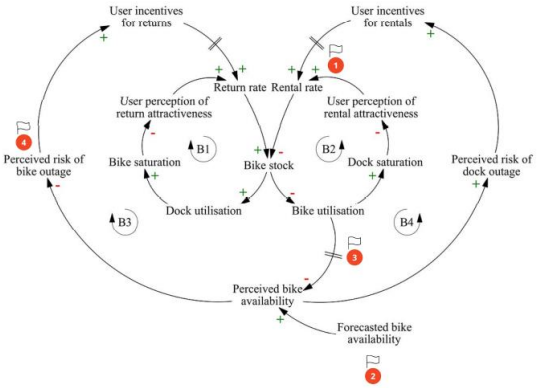
L'obiettivo principale è capire come l'integrazione di algoritmi predittivi e giudizio umano possa ottimizzare il bilanciamento dell'inventario nelle flotte di bike-sharing, utilizzando Citi Bike di New York come caso di studio.

Nel caso studio analizzato osserviamo come i sistemi bike-sharing con stazioni fisse soffrono di squilibri nell'inventario: alcune stazioni esauriscono le biciclette mentre altre rimangono sovraccariche. Citi Bike utilizza incentivi per incoraggiare gli utenti a spostare le biciclette (Bike Angels Program), ma l'efficacia di questi incentivi non è sempre ottimale. Il problema principale è la discrepanza tra la domanda prevista e gli incentivi forniti, che porta a inefficienze operative e insoddisfazione degli utenti.

L'approccio usato si vasa su quattro fasi principali: innanzitutto viene usata una Mappatura Causale dove vengono identificate le relazioni tra le variabili chiave nel sistema di bike-sharing, successivamente vengono creati dei modelli predittivi e dinamici tramite la formulazione di modelli di simulazione e Machine Learning, viene poivfatto un test parziale del modello verificandone la coerenza dei sotto-moduli (flussi,stock e previsione della domanda), per poi infine fare un'analisi delle policy per simulare degli scenari reali per valutare strategie di incentivazione più efficaci.



*Schematizzazione del framework sopradescritto. (*Sankaran*, G.; Palomino, M.A.; Knahl, M.; Siestrup, G. Towardsa System Dynamics Framework forHuman–Machine Learning Decisions: A Case Study of New York Citi Bike.Appl. Sci. 2024, 14, 10647)*



*Modello SD usato nello studio. (*Sankaran*, G.; Palomino, M.A.; Knahl, M.; Siestrup, G. Towardsa System Dynamics Framework forHuman–Machine Learning Decisions: A Case Study of New York Citi Bike.Appl. Sci. 2024, 14, 10647)*

Da notare come il modello SD sia stato implementato usando Vensim e PySD, una libreria Python che permette di combinare modelli di SD con algoritmi di Machine Learning. Nel nostro caso PySD è stato usato per collegare il modello con il modello di Machine Learning basato su LSTM.

Per gestire e migliorare l'inventario, infatti, è stato implementato un modello di Long Short-Term Memory (LSTM), un tipo di rete neurale ricorrente adatta alla previsione di serie temporali. Il modello è stato usato per due scopi principali:

- Previsione della domanda: L'LSTM è stato addestrato utilizzando dati storici di Citi Bike per prevedere la domanda futura di biciclette in ogni stazione.
- Stima dell'Impatto degli Incentivi: Per capire l'efficacia degli incentivi, è stato utilizzato un modello di inferenza causale basato sull'approccio di Brodersen,

Previsione domanda

La previsione della domanda è stata una delle componenti chiave del modello proposto nello studio, utilizzata per ottimizzare la gestione dell'inventario di biciclette e il posizionamento degli incentivi. Per questo, è stato adottato un modello di Machine Learning basato su reti neurali ricorrenti LSTM (Long Short-Term Memory), particolarmente adatto alla modellazione di serie temporali. L'obiettivo è stimare il numero di noleggi e restituzioni in ogni stazione di Citi Bike per i prossimi intervalli temporali, tenendo conto di fattori stagionali, condizioni metereologiche, effetto degli incentivi, dinamiche di utilizzo storiche. L'accuratezza della previsione è cruciale perché influenza il bilanciamento dell'inventario e la gestione degli incentivi.

Per realizzare un modello accurato, il processo è stato suddiviso in 7 passaggi: vengono raccolti e puliti i dati, vengono create variabili utili per il modello (*feature engineering*), normalizziamo i dati per garantire una convergenza stabile del modello, definiamo il modello LSTM, lo addestriamo e validiamo i parametri verificando le prestazioni, e lo integriamo con il modello di System Dynamics. LSTM è una variante delle reti neurali ricorrenti (RNN) che può memorizzare informazioni a lungo termine, il che è utile per dati sequenziali come le serie temporali. L'utilizzo di questo algoritmo ha migliorato le previsioni in modo eccellente e, confrontato con altri algoritmi quali Exponential Smoothing e Naive Forecasting, ha migliorato del 26% la previsione rispetto al primo e ridotto l'errore medio del 15-20% rispetto al secono.

Impatto sugli incentivi

Per valutare l'efficacia degli incentivi nella redistribuzione delle biciclette, lo studio utilizza un modello di inferenza causale basato sul metodo di Causal Impact proposto da Brodersen et al. (2015).

L'obiettivo è stimare il reale effetto degli incentivi confrontando la domanda osservata con una domanda controfattuale (cosa sarebbe successo senza incentivi).

Il metodo utilizzato è basato su una modifica della regressione bayesiana strutturata che costruisce un controfattuale, ovvero una stima di quale sarebbe stata la domanda senza incentivi, confronta la domanda effettuata contro quella stimata senza incentivi e fornisce un intervallo di confidenza.

Nel caso presentato, poiché non abbiamo un esperimento con gruppo di controllo, lo studio seleziona delle stazioni simili a quella incentivata, basandosi sulla somiglianza nell'andamento storico della domanda.

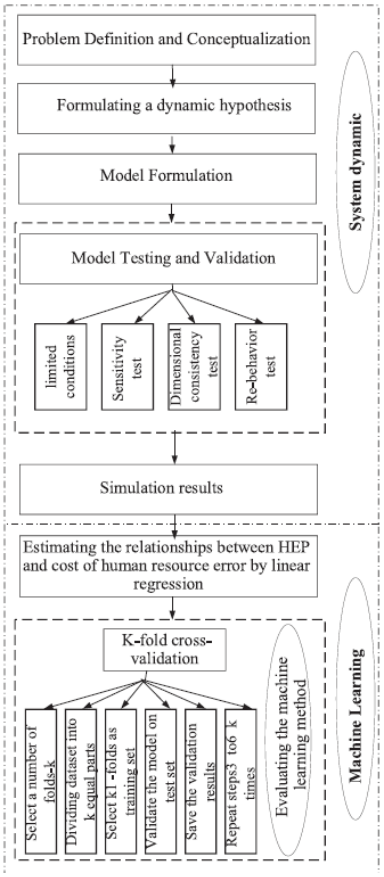
Come risultato, otteniamo che gli incentivi hanno aumentato la domanda di noleggio, tuttavia il 40% degli incentivi è stato applicato in momenti sbagliati, portando ad uno spreco delle risorse.

Come insegnamento da questo caso studio possiamo riportare come l'utilizzo del Machine Learning, soprattutto nella stima dei parametri dei modelli SD sia importantissima al fine di ottenere una maggiore accuratezza nella stima della simulazione.

3.2.4 Caso studio con uso di regressione polinomiale

Ora presenterò un altro articolo, molto interessante nella nostra ricerca, dove viene presentata una implementazione del Machine Learning nel modello SD usato per migliorare le previsioni con dati reali, ottimizzare i costi e supportare il decision making dei manager, capaci ora di simulare scenari molto più realistici rispetto al passato.

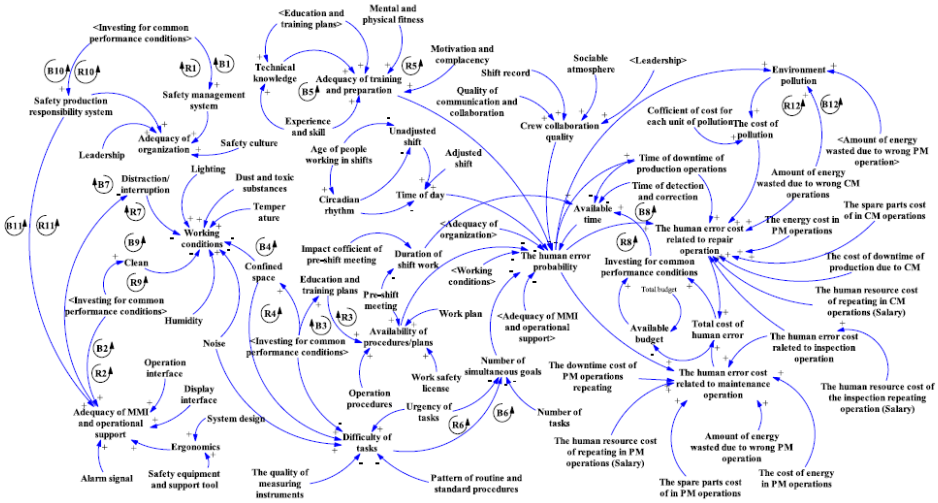
Nel caso in questione viene analizzata la probabilità di errore umano (HEP) nei compiti di manutenzione utilizzando un approccio integrato basato sulla dinamica dei sistemi (SD) e sull'apprendimento automatico (ML).



*Flow chart usato nel caso studio. (*Emroozi *V.B, Kaziemi M., Pooya A, Doostparast M, Evaluating human error probability in maintenance task: An integrated system dynamics and machine learning approach 2024)*

In primo luogo, definiamo il problema e identifichiamo le variabili chiave del modello, formuliamo l'ipotesi dinamica, trovando le realzioni tra variabili, formuliamo il modello Stock & Flow e lo validiamo.

Otteniamo così il diagramma causale:



*Causal loop diagram (*Emroozi *V.B, Kaziemi M., Pooya A, Doostparast M, Evaluating human error probability in maintenance task: An integrated system dynamics and machine learning approach 2024)*

Successivamente viene usata l'AI per estrarre relazioni funzionali tra le variabili simulate nel modello SD e i dati reali aziendali ed è costruito con un approccio di regressione polinomiale cubica per stimare la relazione tra l'HEP e i costi associati.

Una volta raccolti i dati dal modello è stato scelto il modello di regressione polinomiale cubica perché ha fornito i migliori risultati in termini di errore quadratico medio (MSE) e R-quadro.

Successivamente viene usata la tecnica del K-Fold Cross-Validation per valutare la robustezza del modello, divedendo il dataset in 10 gruppi dove 9

erano destinati al training e l'ultimo per il test, ripetuto 10 volte. Così è stato scelto il modello con MSE minimo per garantire la maggiore accuratezza.

Ricapitolando con l'SD generiamo simulazioni dinamiche sull'evoluzione dell'HEP e dei costi, il ML apprende la relazione tra HEP e costi aziendali, affinanfo la previsione con la regressione polinomiale, ottimizzando poi i parametri e trovando un valore di HEP ottimale che minimizza costi e incidenti. Le previsioni ML vengono successivamente riutilizzate nel modello SD migliorando la qualità delle simulazioni per futuri scenari decisionali.

La ricerca ha portato buoni risultati, ma per uno sviluppo futuro, l'utilizzo dell'algoritmo random forest per sviluppare previsioni, sarà più accurato e più attuabile.

3.2.5 Modellazione SD tramite random forest

Ricollegandoci alle osservazioni fatte nel capitolo precedente analizziamo ora l'applicazione dell'algoritmo Random Forest. L'utilizzo di questa metodologia IA nell'implementazione SD permette grossi vantaggi dal punto di vista applicativo. Rispetto ad un modello tradizionale l'analisi viene accelerata, riducendo il numero di simulazioni nescessarie, automatizzando la classificazione dei pattern di comportamento, generando regole interpretabili per comprendere meglio il sistema e guidare le decisioni, identificando rapidamente i parametri chiave, evitando analisi di sensibilità manuali e rendendo la modellazione SD più accessibile a chi non ha competenze avanzate di dinamica dei sistemi.

Questa metodologia, non può sostituire la modellazione SD, ma può potenziarla in modo esponenziale rendendo l'analisi più efficiente e interpretabile.

Come è possibile implementare quest'algoritmo nella modellazione?

Le metodologie sono molteplici, tuttavia dalla ricerca risalta un'applicazione in particolare

In un modello SD esistente vengono selezionati i parametri più rilevanti per l'analisi e si studia come la variazione di questi parametri impatti i risultati del modello SD.

Dopo aver selezionato i parametri viene creato un dataset di input-output eseguendo simulazioni del modello SD con combinazioni diverse di input. Il dataset sarà quindi composto da variabili di input e risposte del sistema in base agli input dati.

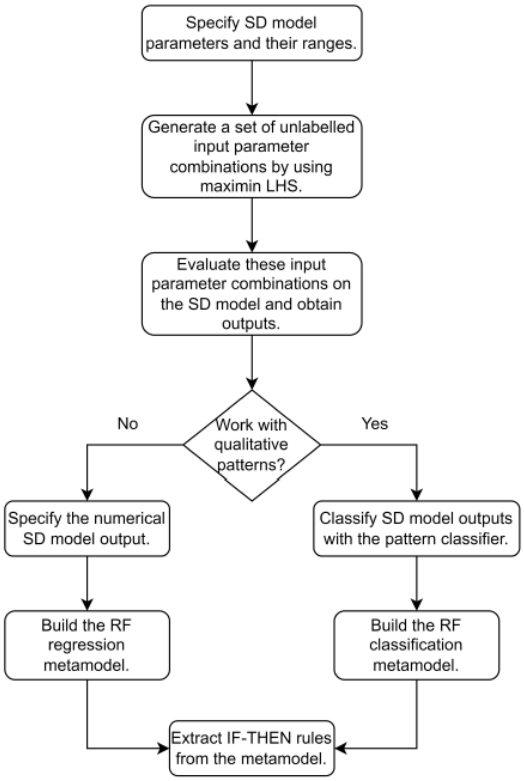
Il dataset viene generato tramite un campionamento Latin Hypercube Sampling (LHS), una metodologia di campionamento molto avanzata che migliora l'efficienza rispetto ad un normale campionamento casuale, creando un set diversificato di valori di input. Ogni set di parametri diinput viene simulato nel modello SD per ottenere l'output corrispondente, il quale può essere qualitativo, ovvero può classificare la tipologia di sistema dinamico, quale oscillativo, crescita esponenziale, stabilizzazione, oppure quantitativo ovvero trova la deviazione rispetto al valore target tramite una regressione.

Una volta ottenuto il dataset, si procede all'addestramento del modello RF. Il dataset deve essere diviso in un set di training e in un set di test, dove il primo viene usato per addestrare il modello RF, mentre il secondo verifica l'accuratezza della previsione generata. Il modello Random Forest viene addestrato per imparare le relazioni tra i parametri di input e i corrispondenti output generati dal modello SD. Un esempio di random forest può essere come quello riportato nello studio di Mert Edali. Il modello RF è composto da 100 alberi decisionali, e per ogni nodo, viene selezionato un sottoinsieme casuale dei parametri di input per determinare la migliore suddivisione dei dati. L'output viene poi determinato per i modelli qualitativi tramite la predizione della classe più votata tra gli alberi, mentre per i modelli quantitativi tramite la predizione della media delle previsioni di tutti gli alberi.

Dopo l'addestramento, si valuta l'accuratezza del modello RF confrontandolo con il test set.

Mert Edali presenta due casi, uno qualitativo nel modello di regolazione della temperatura e uno quantitativo, nel modello Inventory-Workforce. Nel primo caso l'errore medio del modello RF è dell'8% quindi il metamodello RF predice correttamente la dinamica in oltre il 92% dei casi, mentre nel

secondo l'errore medio MAPE (Mean Absolute Percentage Error) è del 4,06%, indicando un'elevata precisione nel prevedere l'output.



Flow chart del modello presentato (Edali M, Pattern-oriented analysis of system dynamics models via random forests, 2022)

Un'altra peculiarità è la possibilità di estrarre regole interpretative dal modello RF. Sebbene il RF sia più interpretabile di una rete neurale, è comunque un modello black-box che rende difficile comprendere quali combinazioni di input generano determinati comportamenti. Per risolvere questo problema si usa un algoritmo di estrazione di regole IF-THEN che analizza gli alberi decisionali della RF e seleziona un sottoinseìieme ottimale di regole. Ciò permette di tradurre il comportamento del modello SD in regole interpretabili, utilissime per il policy-making e l'ottimizzazione del sistema.

In conclusione l'applicazione di Random Forest in un modello SD offre diversi vantaggi rispetto ai modelli tradizionali, sia nell'ambito della modellazione predittiva che in quello della System Dynamics (SD), tra cui maggiore robustezza e accuratezza maggiore, minore sensibilità al rumore dei dati, maggiore capacità di lavorare con dati complessi, identificazione automatica delle variabili più rilevanti, scalabilità e velocità maggiore, interpretabilità e generazione di regole IF-THEN e adattabilità a diversi tipi di problemi.

3.2.5 Applicazione del Reinforcement Learning all'SD

In questa sezione analizzeremo le metodologie usate per implementare il Reinforcement Learning nella modellazione SD. Useremo un modello citato dall'articolo di Fabian Bussieweke, Josefa Mula & Francisco Campuzano-Bolarin, dove viene affrontato il problema delle disruption nelle supply chain globali causate da eventi imprevisti, come la pandemia da COVID-19 e i conflitti militari. L'obiettivo principale di questo modello è quello di sviluppare strategie di recupero ottimali utilizzando una combinazione di System Dynamics (SD) e Reinforcement Learning (RL), per mitigare l'effetto a cascata senza avere necessariamente dati completi sulla disruption.

Il modello di System Dynamics (SD) utilizzato nell'articolo è una simulazione di una supply chain a quattro livelli, progettata per studiare l'effetto delle interruzioni (ripple effect) e testare politiche di recupero ottimali. Include quattro attori principali, Fornitore Produttore Distributore e Rivenditore e l'interazione tra questi livelli avviene tramite flussi di ordini e materiali, regolati da livelli di inventario e backlog.

Il modello può simulare tre tipi di disruption: disruption della capacità di trasporto, disruption della domanda, disruption della fornitura. Tutti e tre i tipi sono basati sulle interruzioni date dal COVID-19, dove appunto si sono verificate in molte aziende tutte e tre le tipologie di interruzioni. L'effetto a cascata (*ripple effect*) si verifica quando una disruption colpisce una parte del sistema, propagando il suo impatto lungo tutta la catena di approvvigionamento.

Con l'uso del Reinforcement Learning nel modello otteniamo una integrazione innovativa che permette di ottimizzare le politiche di recupero della supply chain in caso di disruption.

Come è utilizzato il RL nel modello?

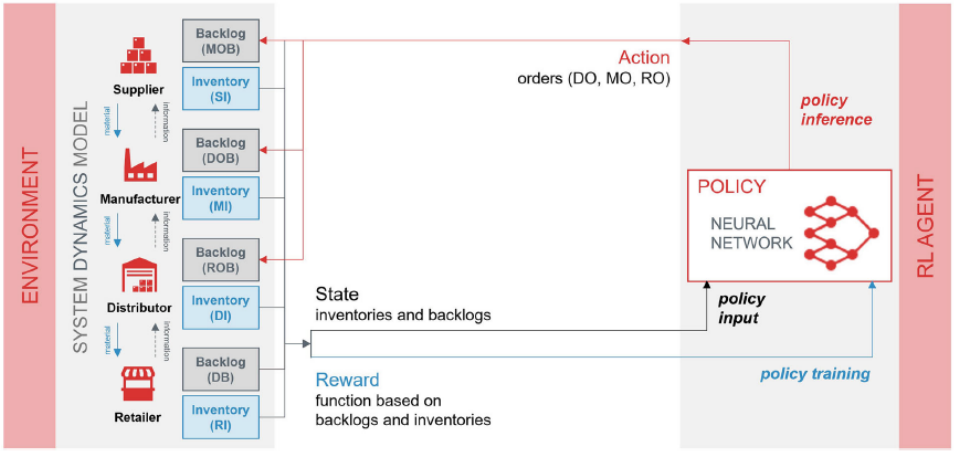
Il modello SD funge da ambiente in cui l'agente RL interagisce e apprende. Ogni ciclo di simulazione rappresenta un episodio di apprendimento.

Innanzitutto, l'agente RL apprende a decidere la quantità di ordini per i Distributori, Produttori e Rivenditori ad ogni timestep t, basandosi sulle condizioni attuali della supply chain. L'agente poi osserva i livelli di inventario lungo la supply chain, il backlog degli ordini a ogni livello, la domanda attuale e la capacità di trasporto disponibile se colpita da disruption. L'agente RL può poi modificare gli ordini da piazzare ad ogni livello, e questi valori vengono appresi e aggiornati dinamicamente. L'obiettivo dell'RL è ridurre le variazioni negli inventari e backlog per evitare stockout e overstocking. Viene quindi definito il reward come:

$$R_t = -\sqrt{\sum_{l_t \in L_t} (l_t - \mu_D)^2}$$

Dove L_t è il set di variabili di livello (inventari e backlog), μ_D è la domanda media attesa. L'agente è penalizzato se ci sono forti deviazioni rispetto ai livelli desiderati di inventario e backlog.

Sono stati testati due algoritmi RL: il Proximal Policy Optimization (PPO) e l'Advantage Actor-Critic (A2C). Il primo è un algoritmo policy-based che aggiorna in modo stabile la policy evitando aggiornamenti troppo grandi, viene utilizzata una rete neurale per mappare lo stato alle azioni. Si è rivelato più efficace e stabile nei risultati, mentre il secondo è un algoritmo che apprende separatamente la politica e il valore dello stato, che comunque ha dato dei buoni risultati ma con maggiore varianza rispetto a PPO.



*Visualizzazione di come funziona il modello, (*Fabian *Bussieweke, Josefa Mula & Francisco Campuzano-Bolarin (06 Aug2024): Optimisation of recovery policies in the era of supply chain disruptions: a systemdynamics and reinforcement learning approach, International Journal of Production Research, DOI: 10.1080/00207543.2024.2383293*

I risultati dell'applicazione dell'RL rispetto ad un modello tradizionale offre numerosi vantaggi nel contesto dell'ottimizzazione delle strategie di recupero delle supply chain. Innanzitutto, aumenta l'adattabilità a scenari di disruption Variabili: L'RL può infatti apprendere politiche di riordino ottimali anche senza conoscere in anticipo le caratteristiche della disruption (tempo, durata, localizzazione), può essere applicato a supply chain complesse con incertezze operative e logistiche e funziona anche quando le disruption sono randomizzate, migliorando la resilienza generale. Permette quindi una maggiore capacità di reazione anche in contesti incerti o con dati incompleti.

Come risultato principale dobbiamo citare la consistevole riduzione dell'effetto a cascata (Ripple effect). Vengono infati ottimizzate le decisioni sugli ordini, riducendo l'accumulo di eccesso di inventario o ritardi nelle consegne e minimizza le fluttuazioni negli stock e nei backlog, garantendo

una catena di approvigionamento più stabile, portando minori ritardi operativi e minori costi legati alla gestione delle scorte.

Possiamo anche notare una maggiore efficienza rispetto a metodi tradizionali. Viene confrontato anche questo sistema con la metaeuristica Powell's Method e dimostra che utilizzare RL offre migliori prestazioni nella stabilizzazione degli stock, una maggiore flessibilità e adattabilità alle disruption e migliori risultati a lungo termine rispetto a metodi statici.

Inoltre, possiamo constatare la riduzione di necessità di grandi dataset storici per prevedere le disruption, dato che RL non ha bisogno di dati passati, perché apprende iterativamente integrando con il modello SD, il che rende il modello ideale per situazioni nuove e impreviste, dove i dati storici sono scarsi o assenti.

L'integrazione del Reinforcement Learning e System Dynamics offre un metodo innovativo e adattabile per gestire disruption nella supply chain con vantaggi significativi in termini di flessibilità, efficienza, automazione e robustezza. Risulta un approccio rivoluzionario sopratutto grazie alla capacità di reagire in tempo reale.

3.2.6 Usare il Deep learning per la stima dei parametri

Discuteremo ora di come sia possibile applicare algoritmi di Deep Learning per stimare i parametri di un modello SD.

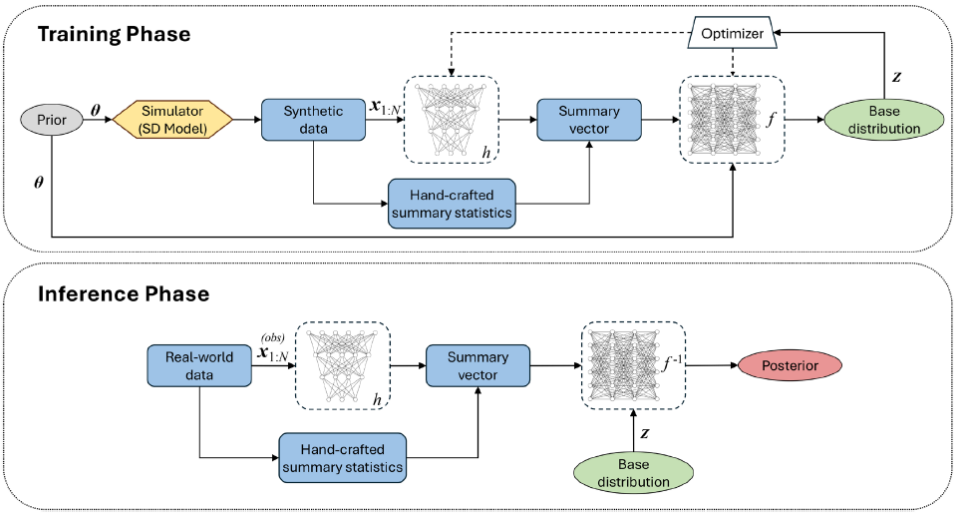
La stima dei parametri e dei loro intervalli nei modelli System Dynamics è un argomento importantissimo per il continuo sviluppo della metodologia. Più le stime sono accurate, più i risultati attesi della simulazione sono attendibili. Il Deep Learning riesce ad approssimare funzioni complesse e, a differenza dei metodi di calibrazione tradizionali, riesce ad apprendere in modo efficiente da grandi quantità di dati, rendendo la stima più precisa.

Nei modelli SD, stimare i parametri significa trovare valori numerici che permettono al modello di generare output che siano coerenti con i dati osservati. Questo è un processo complesso a causa della nonlinearità dei modelli, dei dati incompleti che ci vengono formiti, dall'impossibilità di calcolare funzioni di verosimiglianza e dell'onere computazionale che la calibrazione tradizionale richede. Il Deep learning aiuta a superare questi problemi con l'uso di reti neurali per approssimare la relazione tra parametri e dati osservati.

Una delle tecniche più promettenti a tal riguardo è l'*Amortized Bayesian Inference* (**ABI**), che sfrutta il Deep Learning per rendere il processo più efficiente. L'Amortized Bayesian Inference è una tecnica avanzata che permette di stimare i parametri di un modello in modo scalabile ed efficiente, superando i limiti delle classiche tecniche di inferenza bayesiana.

Inizialmente vengono creati dei dati sintetici tramite molte simulazioni del modello SD con parametri casuali estratti da una distribuzione a priori, per poi raccogliere le serie temporali simulate per creare un dataset di training. Procediamo poi ad addestrare la rete neurale, facendole apprendere la relazione tra i dati osservati e i parametri sottostanti tramite il metodo

Neural Posterior Estimation (NPE), un metodo che approssima direttamente la distribuzione a posteriori dei parametri. Una volta addestrata, la rete può stimare istantaneamente i parametri da nuovi dataset senza bisogno di iterazioni lente di ottimizzazione tramite un'inferenza rapida sui nuovi dati.



Visualizzazione della fase di traning e di inferenza (Rahmandad,H., Akhavan A., Jalali M.S., Incorporating Deep Learning Into System Dynamics:Amortized Bayesian Inference for Scalable Likelihood-FreeParameter Estimation,2024)

L'architettura di rete è composta da due elementi principali, la summary network che permette l'estrazione di feature dai dati osservati e la inference Network che stima la distribuzione dei parametri in base delle feature estratte.

La summary network trasforma le serie temporali osservate in una rappresentazione compatta (un vettore di caratteristiche) che verrà poi usato per l'inferenza dei parametri. È composta da:

Reti convoluzionali 1D (CNN) che operano in modo gerarchico, dove strati più profondi apprendono pattern temporali più complessi e sono usate per catturare i pattern locali nei dati temporali e applicare filtri convoluzionali sulle serie temporali per estrarre feature di basso e alto livello.

Long Short-Term Memory (LSTM) usate per catturare dipendenze a lungo termine nelle serie temporali e per gestire il problema della sparizione del gradiente, tipico delle reti ricorrenti standard.

Summary Vector è l'output finale della Summary Network ed è un vettore di caratteristiche dove vengono riassunte le informazioni utili sulle serie temporali e viene passato alla Inference Network.

La Inference Network ha invece come obiettivo approssimare la distribuzione a posteriori dei parametri. È composta da Normalizing Flows utilizzati per modellare distribuzioni di probabilità complesse. Consentono di trasformare una distribuzione semplice tipo una Gaussiana standard nella distribuzione a posteriori dei parametri.

Una volta addestrata, l'architettura, dopo aver osservato un nuovo dataset x, lo fa passare attraverso la Summary Network. Il vettore di feature risultante viene dato in input alla Inference Network, che genera campioni della distribuzione a posteriori dei parametri $p(\theta|x)$, dando come risultato una stima istantanea della distribuzione dei parametri del modello SD.

Sono numerosi i vantaggi rispetto alle metodologie tradizionali per la stima dei parametri.

Mentre ad esempio la calibrazione numerica tradizionale richiede iterazione lente e ripetute su ogni nuovo dataset, oppure nei metodi bayesiani classici vanno campionati ripetutamente la distribuzione a posteriori, rendendo il processo molto costoso dal punto di vista computazionale, con l'ABI l'addestramento della rete è costoso inizialmente, ma una volta completato, l'inferenza su nuovi dataset è istantanea. Non è necessario, infatti, nessun bisogno di riaddestramento: una volta appreso il mapping tra dati e parametri, il modello generalizza a nuobi dataset senza nuove interazioni, il che rende il modello ideale anche per applicazioni in tempo reale o con

grandi dataset. Il fatto che il modello debba eseguire il training una sola volta, gli permette di essere altamente scalabile e generalizzabile.

Come detto in precedenza nei modelli SD esiste enorme difficoltà nel calcolare una funzione di verosimiglianza, a causa dell'alta non linearità di questi modelli, e a volte è impossibile calcolarla, forzando la semplificazione del modello. Con la metodologia ABI non c'è necessità di una funzione di verosimiglianza, dato che utilizza solo un motore simulativo (il modello SD stesso) per generare dati sintetici, il che è un enorme vantaggio nei modelli complessi e stocastici.

È possibile anche quantificare l'incertezza in modo più affidabile. Nei metodi tradizionali, infatti, o viene fornita una stima puntuale come nella calibrazione numerica, o con altri metodi bayesiani forniscono distribuzioni, ma sono lenti e difficili da validare. Con il sistema ABI viene data una stima delle distribuzioni completa (a posteriori) per ogni parametro invece di un singolo valore, generando intervalli di credibilità, dando un'idea della qualità delle stime, ed è facile validare l'inferenza tramite Simulation-Based Calibration (SBC).

3.2.7 Stima dei parametri tramite ANN SVM e RF

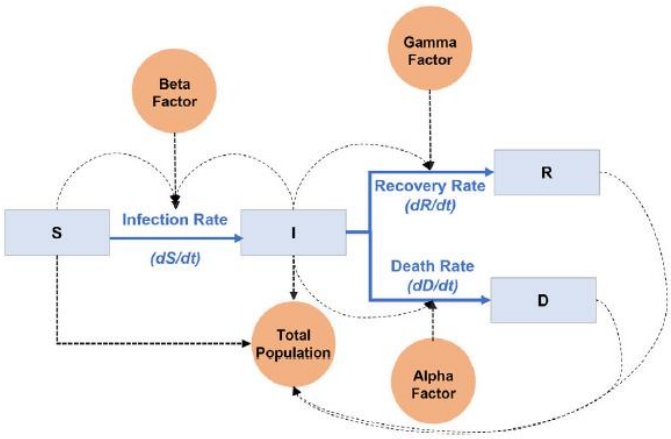
Analizzeremo ora un'altra metodologia per stimare i parametri usando diversi tipi di algoritmi AI confrontando i risultati ottenuti e valutando eventuali differenze negli approcci e trovando il migliore.

Le ANN (*Artificial Neural Networks*) sono modelli di intelligenza artificiale ispirati al funzionamento del cervello umano. Sono composte da neuroni artificiali organizzati in strati (layers) che elaborano i dati attraverso connessioni ponderate. Possiamo considerarle simili all'approccio analizzato nel precedente paragrafo.

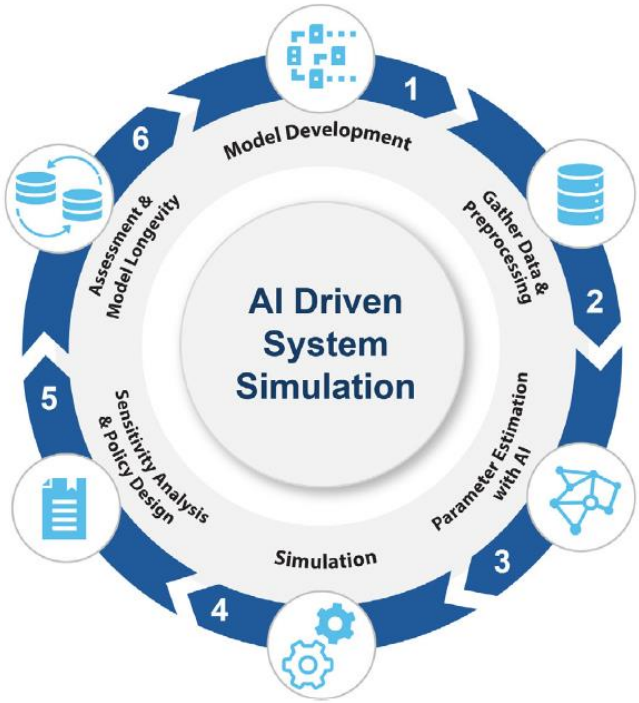
Le SVM (*Support Vector Machines*) sono algoritmi di apprendimento supervisionato usati per classificazione e regressione. Sono particolarmente efficaci nel separare dati in classi attraverso un iperpiano ottimale.

Le RF (*Random Forest*) sono un insieme di alberi decisionali utilizzati per classificazione e regressione. Sono già state presentate ampiamente nei capitoli precedenti.

Useremo come esempio di caso studio un modello SIR per la diffusione del COVID-19



Il semplice modello SD usato per il test (Marshall J., Gadewadikar J., Amethodology for parameter estimation in system dynamicsmodels using artificial intelligence, 2023)



I passaggi necessari a creare un modello SD integrato con AI (Marshall J., Gadewadikar J., Amethodology for parameter estimation in system dynamicsmodels using artificial intelligence, 2023)

Iniziamo ad analizzare come le ANN siano utilizzate per stimare i parametri del modello SIR(Susceptible-Infected-Recovered), in particolare per prevedere le funzioni di infezioni, guarigione e mortalità.

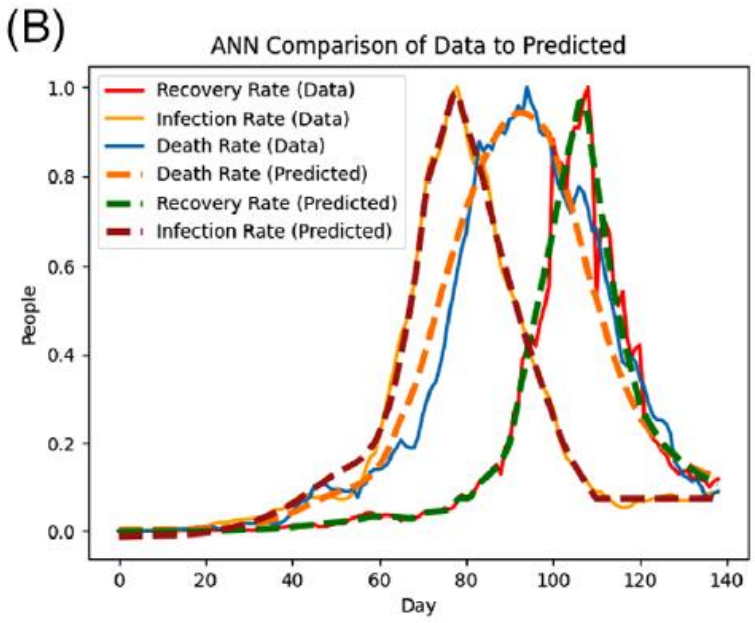
Le ANN sono state usate per sostituire le equazioni tradizionali del modello SIR con una previsione basata sui dati, avendo come obiettivo la previsione del tasso di infezione $\frac{dS}{dt}$ in funzione del numero di suscettibili S e infetti I, il tasso di guarigione $\frac{dR}{dt}$ in funzione del numero di infetti I e il tasso di mortalità $\frac{dD}{dt}$ in funzione del numero di infetti I.

Prima di allenare la rete neurale, pre-elaboriamo i dati per garantire che siano adatti al modello ANN scegliendo dei dati di input ovvero il numero di suscettibili e degli infetti e di output, ovvero i tassi sopracitati. Successivamente applichiamo una media mobile per ridurre il rumore nei dati raccolti e scaliamo i dati tra 0 e 1 per migliorare la stabilità dell'addestramento della rete neurale.

L'ANN viene quindi costruita con la libreria PyTorch, una libreria presente in PySD un software di cui già abbiamo parlato in precedenza riguardo la modellazione di SD. L'architettura di quest'ultima utilizza due neuroni per il tasso di infezione, mentre uno rispettivamente per il tasso di guarigione e per il tasso di mortalità.

La rete viene quindi addestrata e testata su nuovi dati. Nel nostro caso studio viene calcolato il coefficiente di determinazione R-squared per misurare la precisione delle previsioni, riportando un risultato molto positivo: per il tasso di infezione è risultato pari a 0.995, per il tasso di guarigione 0.972, mentre per il tasso di mortalità 0.965.

Integriamo quindi nel sistema dinamico e, al posto di usare le equazioni tradizionali, utilizziamo la rete neurale per prevedere i valori dei tassi. I risultati sono stati molto simili ai dati reali della pandemia.



Paragone tra la simulazione usando ANN e i dati reali ((Marshall J., Gadewadikar J., Amethodology for parameter estimation in system dynamicsmodels using artificial intelligence, 2023)

Passando ora alle SVM sono state utilizzate oer stimare la relazione tra le variabili del modello. A differenza dell'ANN, le SVM trasformano i dati in uno spazio di dimesioni superiori, dove la relazione diventa più lineare, e applicano un'analisi di regressione. Poiche le SVM sono sensibili alla scala dei dati, i valori sono stati scalati tra 0 e 10 per migliorare la precisione delle previsioni.

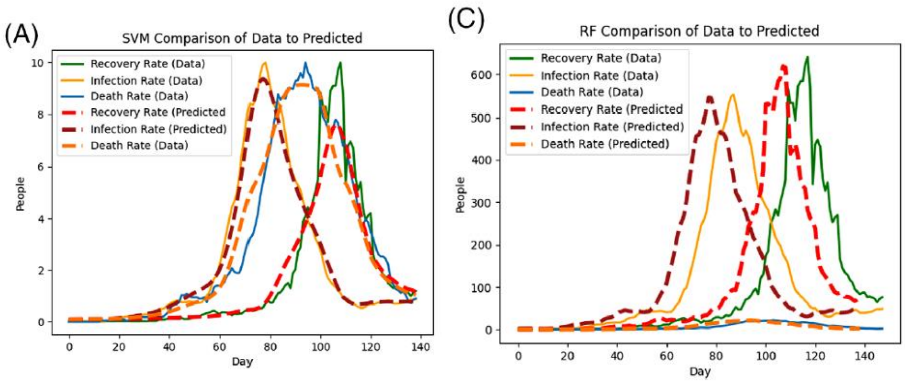
Dividiamo il dataset in training set (80% del dataset) e validation set (20%). Grazie a Questa metodologia siamo riusciti ad ottenere risultati piuttosto accurati, con un coefficiente R-squared sufficientemente elevato, ottenendo per il tasso di infezione un valore 0.984, per il tasso di guarigione un tasso 0.866 e per il tasso di mortalità 0.952.

Le Random Forest (RF) sono un metodo di regressione basato su alberi decisionali ensemble. Sono state utilizzate per stimare i flussi del modello SIR, analogamente alle SVM e alle ANN.

Ogni albero decisionale è stato addestrato su un sottoinsieme casuale del dataset ed è stata utilizzata una tecnica chiamata Bootstrap Aggregating (Bagging) per ridurre l'overfitting, ottenendo come output la media delle previsioni dei singoli alberi (molto simile alla metodologia presentata nel paragrafo 3.1.5). Il modello RF ha ottenuto un buon livello di accuratezza con gli R-squared pari a 0.993 per il tasso di infezione, 0.972 per il tasso di guarigione e 0.925 per il tasso di mortalità. Tuttavia, le RF hanno mostrato ritardi nelle previsioni rispetto ai dati reali e sopratutto il modello ha avuto difficoltà a generalizzare dati fuori dal range di addestramento, portando a previsioni costanti in alcuni scenari.

Alla fine del test le ANN hanno ottenuto i risultati migliori, ma richiedono più dati e più tempo di addestramento, le SVM hanno mostrato buone prestazioni, ma con alcune difficoltà per dati con valori molto bassi, mentre le RF sono state le meno affidabili nel generalizzare i dati e hanno mostrato previsioni costanti in alcuni casi.

Nella simulazione del modello SD le ANN e le SVM hanno prodotto simulazioni molto simili ai dati reali, mentre le RF hanno avuto problemi di stabilità e hanno previsto un tasso di cambiamento costante nelle simulazioni a lungo termine. Tuttavia, in conclusione le SVM hanno mostrato risultati migliori nei primi giorni della simulazione, mentre le ANN hanno avuto un miglior adattamento generale.



Paragone tra la simulazione usando SVM e RF e i dati reali ((Marshall J., Gadewadikar J., Amethodology for parameter estimation in system dynamicsmodels using artificial intelligence, 2023)

3.2.8 Algoritmo k-means

In questo paragrafo parleremo di come sia possibile implementare un algoritmo K-means per migliorare il modello previsionale System Dynamics per migliorare l'efficienza economica dell'economia portuale.

L'algoritmo K-means è un metodo di clustering non supervisionato, utilizzato per raggruppare dati simili in un numero K di cluster. Viene ampiamente usato in analisi dei dati, machine learning e ottimizzazione dei sistemi. L'algoritmo segue un processo iterativo in 4 fasi principali. Vengono selezionati dei centri iniziali, ovvero presi K punti del dataset di base casualmente, sono utilizzati come centroidi iniziali. Successivamente ogni punto del dataset viene assegnato al cluster più vicino in base alla distanza euclidea, applicando la formula base della distanza euclidea:

$$d(x, c) = \sqrt{\sum_{i=1}^{n} (x_i - c_i)^2}$$

dove x è un punto dati e c è il centroide del cluster.

In seguito, vengono aggiornati i centroidi, calcolando quelli nuovi di ciascun cluster come media dei punti assegnati:

$$c_j = \frac{1}{|C_j|} \sum_{x \in C_j} x$$

Vengono iterati i passaggi 2 e 3 fin quando i centroidi non cambiano più o la variazione tra iterazioni è minima.

Il modello SD che utilizzaeremo per illustrare l'implementazione di questo algoritmo, simula l'evoluzione della popolazione, dell'ambiente, delle risorse e dell'economia nei porti, fornendo supporto decisionale per la sostenibilità. Questo modello si basa su interazioni dinamiche tra quattro

sottosistemi chiave, utilizzando equazioni matematiche e feedback loops per prevedere scenari futuri.

I quattro sottosistemi principali che rappresentano le dimensioni fondamentali dello sviluppo portuale sono la popolazione, che determina la forza lavoro e l'espansione della città portuale, l'Ambiente, dove vengono analizzate l'impatto ambientale dello sviluppo portuale, le Risorse, dove sono misurate la sostenibilità dell'uso delle risorse naturali, l'economia, dove viene valutata la crescita economica e la trasformazione dei settori produttivi. Tutti e quattro i sottosistemi sono interconnessi e influenzano reciprocamente il comportamento globale del sistema portuale.

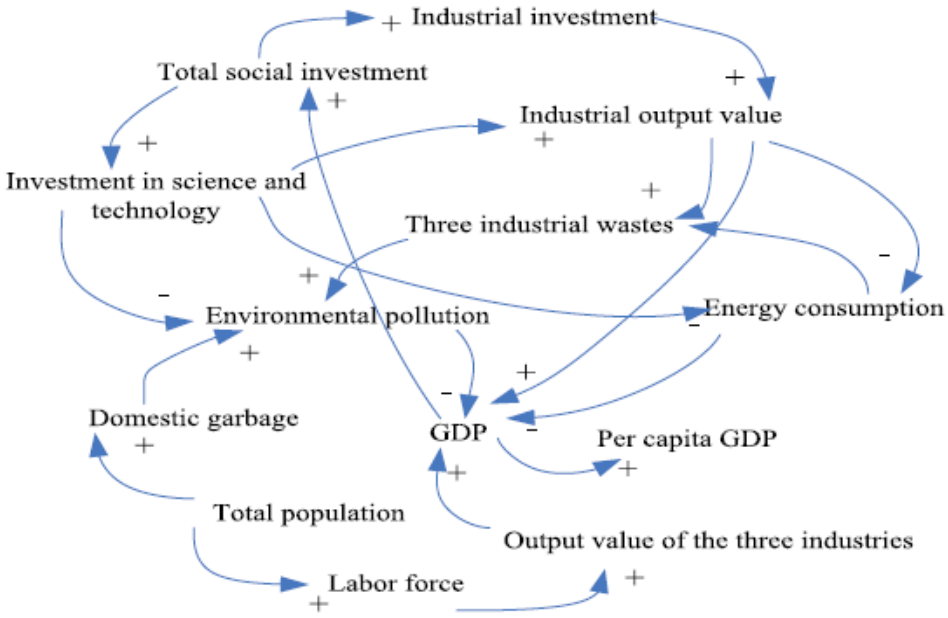


Diagramma causale del modello, Guoyi Xiu, Zexu Zhao, Sustainable Development of Port Economy Based on Intelligent System Dynamics, 2021

L'IA viene applicate nel modello principalmente attraverso due tecnologie, l'algoritmo di clustering K-means ottimizzato, sopradescritto, e sensori intelligenti per la raccolta dati. Questi strumenti migliorano la capacità del

modello SD di analizzare dati complessi e fornire previsioni più precise sullo sviluppo sostenibile dei porti.

L'algoritmo K-means viene integrato per migliorare l'elaborazione e l'analisi dei dati all'interno del modello SD. Il suo utilizzo ha due obiettivi: segmentare e classificare dati complessi e ottimizzare la simulazione.

Il primo obiettivo viene raggiunto suddividendo i dati in gruppi omogenei, formando i cluster, per identificare schemi e correlazioni nei quattro sottosistemi del porto, il secondo invece tramite l'algoritmo vengono ridotte il numero di iterazioni necessarie alla simulazione e velocizza il processo decisionale.

Rispetto all'algoritmo descritto precedentemente, per migliorare la precisione del clustering, viene applicata una strategia basata sulla chiusura dell'insieme dei dati (*closure-based K-means*). In pratica si analizzano solo i punti attivi, ovvero quelli che cambiano cluster durante l'iterazione, per ridurre il numero di calcoli e si usa la distanza euclidea ottimizzata per migliorare l'assegnazione dei dati ai cluster.

I sensori intelligenti (*networked smart sensors*) vengono utilizzati per raccogliere dati in tempo reale da diverse fonti ovvero dati ambientali tramite emissioni industriali, qualità dell'aria, consumo di energia, dati economici tramite movimentazione delle merci e efficienza logistica del porto, dati demografici, variazioni della popolazione e dell'occupazione.

I dati raccolti vengono poi elaborati dal modello SD tramite dynamic feedbacks. I sensori inviano continuamente informazioni aggiornate, il modello ricalcola in tempo reale l'impatto delle politiche economiche e ambientali, gli output vengono usati per adattare le strategie di sviluppo

L’uso del modello SD arricchito con AI e sensori intelligenti offre diversi vantaggi chiave rispetto a un modello tradizionale di analisi economica e ambientale, che spesso si basa su dati storici statici e metodi predittivi meno dinamici.

Possiamo notare una maggiore accuratezza delle previsioni, dato che al posto di affidarsi alle stime statistiche e regressioni su dati storici, i sensori forniscono continuamente aggiornamenti di dati e l’intelligenza artificiale analizza i dati e aggiorna il modello dinamico. Ciò permette anche al modello di rispondere rapidamente ai cambiamenti, rendendolo capace di simulare strategie di mitigazione immediata.

In un modello tradizionale, le variabili sono analizzate separatamente, non considerando le interdependenze tra i vari fattori. Con questo nuovo modello abbiamo un approccio sistematico dove vengono analizzate anche le interazioni tra i sottosistemi, simulando gli effetti a lungo termine di ogni cambiamento, permettendo nel nostro caso di bilanciare sviluppo economico e sostenibilità ambientale.

Rispetto ad altri algoritmi IA il K-means ottimizzato permette di cluserizzare automaticamente i dati economici e ambientali, migliorando le simulazioni, l’IA adatta le simulazioni in base a nuove informazioni, riduce il numero di calcoli necessari, velocizzando l’elaborazione, rendendo il modello più veloce e adattivo, migliorando la qualità delle decisioni strategiche.

3.2.9 Algoritmi genetici per ottimizzazione

In questo paragrafo analizzeremo come applicare gli algoritmi genetici per ottimizzare la selezione delle policies. In un caso tradizionale, infatti, è molto dispendioso utilizzare un modello SD per rappresentare un sistema, dato l'alto livello di conoscenza della materia necessario e l'enorme quantità di tentativi per calibrare il modello.

Introduciamo gli algoritmi genetici come delle procedure di ricerca altamente parallele, matematiche e adattive che appunto, come è intuibile dal nome, sono basati sui processi di selezione naturale genetica nota nel mondo naturale. In pratica consiste nell'applicare il concetto evolutivo della "sopravvivenza del più adatto".

Rispetto ai metodi tradizionali di ottimizzazione nei modelli SD gli algoritmi genetici si differenziano in tre punti principali:

1. Gli algoritmi genetici lavorano con una codifica del set di parametri, non direttamente con i parametri del modello. Ad esempio, un parametro del modello il cui valore può variare da 0 a 31 viene codificato come una stringa binaria di cinque cifre per essere utilizzato.

2. Gli algoritmi genetici effettuano la ricerca da una popolazione di stringhe binarie, non da un singolo punto. Ogni membro della popolazione viene valutato per la sua idoneità prima di procedere con il prossimo ciclo di ottimizzazione. I membri più idonei hanno maggiori probabilità di generare figli per la generazione successiva.

3. Gli algoritmi genetici utilizzano regole di transizione probabilistiche piuttosto che deterministiche. La probabilità gioca un ruolo nel determinare i migliori membri di una popolazione per la riproduzione, il crossover e la mutazione. Sebbene la probabilità sia centrale per l'efficienza operativa dei GAs, questi non sono semplicemente una ricerca casuale attraverso lo spazio delle soluzioni. Le generazioni successive in un'ottimizzazione GA mostrano un miglioramento costante delle prestazioni del sistema nel corso dell'ottimizzazione, riflettendo più di una semplice casualità.

Presentiamo ora due casi in cui è possibile applicare questo tipo di ottimizzazione. Nel primo proveremo ad ottimizzare i valori di quattro parametri costanti, nel secondo invece si mostrerà come è possibile usare gli algoritmi genetici per generare dei dati per le tavole dei modelli. Verrà utilizzato il modello di Kaibab Plateau per testare l'algoritmo. Il Kaibab Plateau Model è un caso di studio ecologico che illustra gli effetti della gestione umana sulle popolazioni animali, in particolare sulla dinamica della popolazione dei cervi nel Plateau Kaibab, situato nel nord dell'Arizona, vicino al Grand Canyon.

Nel primo caso determiniamo i tassi ottimali di caccia per cervi e predatori, oltre a definire gli anni di inizio della caccia, per mantenere una popolazione stabile di 10000 cervi nel tempo. L'algoritmo genetio ottimizza quattro variabili, che sono il tasso di caccia ai cervi, il tasso di caccia ai predatori, l'anno di inizio della caccia ai cervi e l'anno di inizio della caccia ai predatori. Viene quindi calcolata la fitness misurando la deviazione tra la popolazione simulata e il valore target di 10000 cervi, dove ovviamente un valore più vicino al valore target corrisponde a una fitness più alta. In questo test l'algoritmo genetico identifica i parametri ottimali per garantire che la

popolazione di cervi raggiunga le unità prestabilite nel 1910 e rimanesse stabile fino al 1950.

Nel secondo test invece abbiamo come obiettivo la minimizzazione degli effetti negativi di politiche passate e prevenire un disastro ecologico a causa del boom e crash della popolazione dei cervi, avendo uno scenario di partenza dove la popolazione di cervi cresce senza controllo fino a 20000 unità nel 1920, causando un collasso dell'ecosistema con una carestia e un successivo crollo della popolazione. Con l'algoritmo genetico ottimizziamo i tassi di caccia ai cervi in base alla densità della popolazione imponendo un tasso di caccia ai predatori come valore fisso, mentre il tasso di caccia ai cervi viene definito come una funzione tabellare ottimizzata dal GA per adattarsi alla popolazione. La fitness viene quindi asata sulla stabilità della popolazione nel tempo senza picchi di sovrappopolazione o collassi. Puntiamo come obiettivo quello di raggiungere 40000 cervi entro il 1924 e mantenere questa popolazione fino al 1950. Grazie all'algoritmo riusciamo a trovare un tasso di caccia variabile che previene il crash della popolazione, dimostrando quindi che l'algoritmo può gestire politiche adattive correggendo errori del passato e minimizzando l'impatto di politiche errate.

Valutando complessivamente i vantaggi e gli svantaggi di questa applicazione possiamo considerare l'automatizzazione del processo di ricerca delle migliori policy, l'elevata accessibilità al processo e la capacità di trovare soluzioni globali evitando minimi locali come dei grandi vantaggi nell'applicazione. Tuttavia, sono presenti diverse limitazioni, dato che il processo è computazionalmente molto intensivo, a causa della necessità di molte simulazioni per convergere, e necessita di tradurre i parametri in codici binari, il che può introdurre errori se non fatto correttamente.

4. Discussione e conclusioni

In questo ultimo capitolo procederò a presentare i risultati finali del mio lavoro di tesi, sottolinenando i benefici riscontrati nell'integrazione AI-SD, i vantaggi rispetto ai metodi tradizionali, sfide tecniche, gap di ricerca individuati e eventuali opportunità di ricerca future.

4.1 Benefici dell'integrazione AI-SD

Rispetto ad un approccio tradizionale, da questa revisione della letteratura, possiamo osservare come l'implementazione di tecniche AI nei modelli SD permette di ottenere grandi vantaggi. Dagli articoli analizzati infatti sono risultati enormi passi avanti dal punto di vista dell'automatizzazione di processi che altrimenti sarebbero stati da compiere da un utente umano, ad esempio nella stima dei parametri.

In pratica questo approccio permette di migliorare la modellazione, la previsione e l'ottimizzazione dei sistemi complessi.

Abbiamo visto infatti come la modellazione sia stata migliorata dall'AI, grazie alla sua capacità di individuare più facilmente e rapidamente pattern nascosti, automatizzando la calibrazione dei parametri dei modelli SD.

Possiamo osservare anche come le simulazioni SD sono basate su equazioni differenziali e ipotesi semplificate. Grazie all'IA è possibile utilizzare degli algoritmi per il fitting dei dati reali, affinandone le previsioni di modelli SD adattandoli dinamicamente a nuovi dati.

Nel caso specifico del Reinforcement Learning è possibile testare strategie decisionali nei modelli SD, identificando policies ottimali per la gestione dei sistemi complessi.

È inoltre possibile rispetto ad un modello tradizionale, basato principalmente su modelli teorici e strutture causali, intervenire integrando strutture Big Data, migliorando la rappresentazione del sistema e riducendo l'errore di modellazione, e permettendo al modello di gestire grandi quantità di dati.

Non da sottovalutare anche la capacità del sistema integrato di avere una maggiore adattabilità ai cambiamenti del sistema. In situazioni dove le simulazioni devono essere rapide, a causa dell'alta aleatorietà dei parametri in gioco, grazie all'AI è possibile per i modelli adattarsi automaticamente ai cambiamenti, cosa che sarebbe impossibile da fare tramite il solo intervento umano.

Siamo riusciti anche ad osservare come grazie alle recentissime tecnologie, si riesca a generare un modello completamente da zero tramite un semplice input. Anche se ancora non avanzata, questa tecnologia già da ora sta rivoluzionando il modo di modellizzare, rendendo più accessibile questo tipo di simulazione anche alle persone meno esperte.

Inoltre, usando determinate metodologie, è possibile anche usare dati stocastici al posto dei classici dati deterministici, il che permette al modello di utilizzare dati reali e riuscire a dare una stima reale rispetto ad una semplice simulazione. L'AI infatti aiuta ad estrarre, organizzare e integrare dati da diverse fonti, migliorando la qualità e la coerenza del dataset finale, grazie ad algoritmi di raccolta automatici, datacleaning e implementazione di dati mancanti.

Come ultimo, ma non meno importante aspetto, l'Intelligenza Artificiale permette di migliorare la simulazione e la visualizzazione di modelli SD, permettendo anche di creare dashboard interattive, che aiutano i decisori a comprendere meglio le dinamiche del sistema rispetto alle visualizzazioni classiche.

4.2 Sfide tecniche riscontrate nell'integrazione

Come prima sfida individuiamo dei problemi riguardanti la metodologia e la teoria. Difatti AI e SD si basano su principi fondamentali diversi, dove la prima si focalizza sull'apprendimento dei dati, rilevando pattern e correlazioni senza necessariamente comprendere le relazioni causa-effetto, mentre la seconda si concentra proprio sulla modellazione causale dei sistemi, simulando dinamiche complesse nel tempo attraverso equazioni differenziali e interdipendenze strutturali.

Proprio questa differenza è la causa di difficoltà nell'integrazione dei due approcci che sono una prettamente empirica, mentre l'altra più concettuale e strutturale, creando quindi anche una necessità di costruire un linguaggio comune per i due approcci che però non perda il rigore scientifico che entrambe le metodologie hanno singolarmente, il che porterebbe i modelli ad avere il rischio di risultare meno interpretabili.

Dal punto di vista tecnologico, invece, possiamo constatare che l'integrazione tra AI e SD richiede un'elevata potenza di calcolo per elaborare grandi quantità di dati in tempo reale, simulando sistemi dinamici complessi su scale temporali lunghe, unendo le caratteristiche predittive dell'AI con simulazioni SD interattive. Nella realtà, dobbiamo anche valutare che la maggior parte dei modelli SD non sono progettati per lavorare con dati dinamici in tempo reale, anche se tuttavia l'AI necessita di una costante alimentazione di dati aggiornati.

Applicare l'ibridazione dei due sistemi causerebbe la trasformazione dei modelli SD in dei modelli computazionalmente pesanti, il che rallenterebbe enormemente le tempistiche di simulazione. Da valutare anche l'enorme differenza nella velocità di elaborazione dati delle due metodologie, il che

porterebbe un sistema complesso con entrambe le metodologie applicate ad avere problemi di sincronizzazione dati.

Dal punto di vista sociale invece, l'implementazione di questa nuova metodologia, integrandola con i processi aziendali che già conosciamo e usiamo richiederebbe dei cambiamenti strutturali notevoli nelle tecnologie e nelle competenze dei team. I decisori aziendali, infatti, potrebbero anche decidere di non fidarsi delle nuove tecnologie, a causa degli elevati costi di implementazione di un sistema di tale calibro e dell'incertezza dell'efficacia su una tale innovazione.

Oltre agli elevati costi di investimento sarebbero necessari anche altri corsi di training del personale che dovrebbero comprendere come utilizzare al meglio questo nuovo strumento, nonché l'eventuale necessità di dover riorganizzare tutta la struttura dati aziendale, a causa della necessità da parte del sistema di avere dei dati affidabili per eseguire le simulazioni.

Possibili soluzioni a queste sfide potrebbero essere lo sviluppo di framework ibridi, ovvero sviluppare modelli che combinano la causalità della SD con l'adattabilità dell'AI. Ad esempio, in un caso di epidemiologia, l'AI potrebbe analizzare i dati storici dei contagi, mentre la SD potrebbe simulare gli scenari di diffusione

Per le sfide tecnologiche, invece si potrebbe utilizzare cloud computing per simulazioni SD su larga scala e training AI intensivo, integrare tecniche Big Data con i modelli SD, ad esempio dei data lakes per archiviare i dati provenienti da AI e SD, di conseguenza, migliorandone la loro interoperabilità.

Per le sfide sociali invece una possibile soluzione potrebbe essere sviluppare interfacce API per facilitare l'integrazione tra i modelli e i sistemi aziendali esistenti, oppure creare delle strutture modulari e scalabili che consentano di aggiornare singoli componenti, senza dover rifare l'intero sistema. Inoltre,

sarebbe necessario cambiare i programmi di formazione, crendone degli appositi, per formare il personale ad utilizzare il nuovo sistema, nonché sarebbe necessario creare un Framework di Data Governance per definire uno standard per garantire la qualità e la coerenza dei dati tra AI e SD.

4.3 Gap di ricerca

Dalla ricerca emergono diversi gap nella letteratura inerente all'integrazione tra le metodologie System Dynamics e l'Intelligenza Artificiale, sia a livello teorico che applicativo.

Come primo aspetto valutiamo una mancanza di convergenza strutturata tra i due approcci. Nonostante, infatti, entrambe le metodologie affrontino la complessità dei sistemi, mancano dei veri e propri approcci integrati. Gli studi esistenti infatti sono spesso paralleli, sfruttando ad esempio l'AI da un lato per prevedere e classificare dati e dall'altro l'utilizzo dell'SD per sfruttare la modellazione causale, ma non esiste una vera e propria comunicazione tra le due parti. Ciò accade principalmente per due motivi:

- Origini disciplinari diverse: l'SD nasce infatti nell'ingegneria gestionale, mentre l'AI è propriamente una branca dell'informatica
- La mancanza di framework teorici condivisi: la mancanza infatti di metodologie condivise, fa si che non si riescano a trovare strumenti che uniscano le due discipline efficacemente.

Possiamo di fatto parlare dell'assenza di Hard Convergence tra le due metodologie. L'hard convergence, infatti, si verificherà quando l'IA non si limiterà a supportare le analisi del System Dynamics, ma interverrà direttamente nella progettazione, nella modifica e nella ri-configurazione dinamica dei modelli di sistema. In pratica l'Intelligenza Artificiale non sarà più uno strumento analitico e predittivo implementato nell'SD, ma diventerà parte attiva nel costruire o adattare modelli SD, contribuendo di fatto a riscrivere le regole del sistema.

Nella ricerca abbiamo sottolineato come esistano davvero pochi studi e un limitato uso di tecnologie moderne volte a strutturare il modello in sé, difatti

l’AI viene utilizzata più a supporto rispetto a una vera e propria implementazione nel modello.

Un’altra mancanza nello stato dell’arte è l’assenza di studi che cerchino un linguaggio epistemologico comune tra le due metodologie. L’Intelligenza Artificiale si basa su correlazioni nei dati e su predizioni statistiche, senza necessariamente comprendere le cause sottostanti; invece, il System Dynamics è basato su modellazione causale, cercando di rappresentare la struttura e il comportamento di sistemi complessi. La prima utilizza funzioni di ottimizzazione, regressione, reti neurali che sono approcci spesso non interpretabili ovvero dei modelli black-box, mentre l’SD usa equazioni differenziali, loop causali e stock & flow, il che fornisce un approccio più chiaro, ma meno flessibile nell’aggiornamento automatico. Inoltre, mentre l’AI usa un approccio bottom-up, analizzando dati grezzi senza iipotesi a priori, l’SD è top-down partendo appunto da un modello teorico della realtà e simulandolo.

Tutti questi fattori generano problemi sia pratici che teorici. Non esistono infatti framework che uniscano correlazioni AI con i modelli causali SD, senza considerare la diffidenza tra le discipline, dove i ricercatori SD vedono l’AI come una black-box priva di spiegabilità, mentre gli esperti di AI vedono i modelli SD come troppo rigidi.

Un’altro principale gap di ricerca evidenziato nello sudio riguarda la mancanza di applicazioni concrete di questa nuova metodologia nel decision-making aziendale e pubblico. In teoria, l’unione di AI e SD dovrebbe fornire strumenti più potenti per la previsione e la gestione strategica, ma nella pratica pochi modelli sono adottati da aziende e policy makers a causa della elevata difficoltà nell’uso e per la mancanza di interfacce intuitive e strumenti di supporto decisionale.

Molti modelli sono infatti complessi da interpretare e usare, richiedendo competenze avanzate in simulazione e machine learning. Ad esempio, un manager che vuole prevedere la domanda di mercato con un modello AI-SD potrebbe trovarlo troppo tecnico e poco utilizzabile rispetto a strumenti più semplici come, ad esempio, Excel o Power BI.

Proprio l'elevata facilità di utilizzo unita alla trasparenza e spiegabilità di questi ultimi strumenti li fanno preferire, dando più fiducia nei risultati. Inoltre, l'assenza di veri e propri casi studi e di validazione empirica di questo strumento, dato che la maggior parte delle applicazioni sono teoriche e sperimentali, è causa di una non percepita importanza di sviluppo da parte delle aziende che ne gioverebbero.

Un'ulterione gap trovato è a rigurdo del mancato approfondimento di temi emergenti. Questi temi sono cruciali per il futuro della ricerca e dell'innovazione, ma sono ancora poco esplorati nella letteratura AI-SD. Possiamo elencare tra i maggiori i temi la gestione della conoscenza in azienda, l'impatto della tecnologia sul lavoro e l'automazione, l'educazione e la formazione digitale e la gestione dei bias nei dati e nella governance dell'AI.

Le aziende e le organizzazioni hanno grandi quantità di conoscenza non strutturata, che è difficile da organizzare e diffondere efficacemente. La letteratura AI si concentra su la knowledge extraction, ma non su come la conoscenza viene utilizzata e diffusa nei sistemi complessi.
L'approfondimento di questo aspetto sarebbe possibile attuarlo tramite la strutturazione di framework che combinano AI per l'analisi della conoscenza e SD per la modellazione dei processi di apprendimento organizzativo.

L'AI e l'automazione stanno trasformando il mercato del lavoro, ma le conseguenze sistemicje di questo cambiamento sono difficili da prevedere.

L’AI può analizzare trend occupazionali, identificando quali settori sono più a rischio di automazione, mentre l’SD può simulare l’evoluzione del mercato del lavoro modellando scenari in cui l’AI rimpiazza il lavoro umano simulando gli effetti su salari, occupazione e formazione.

I sistemi educativi tradizionali non riescono a tenere il passo con le rapide trasformazioni tecnologiche e i nuovi metodi di apprendimento digitale. L’AI può personalizzare l’apprendimento, analizzando il comportamento degli studenti e suggerendo percorsi educativi adatti a ciascuno, mentre l’SD può modellare i sistemi educativi adatti a ciascuno, simulando l’impatto delle nuove tecnologie sul rendimento degli studenti e sulla distribuzione delle competenze nel tempo. Ad esempio, un’istituzione scolastica potrebbe usare un modello AI-SD per simulare l’effetto dell’introduzione di corsi di coding nelle scuole, valutando come questa innovazione influenzerebbe il mercato del lavoro nel lungo periodo. L’AI nella letteratura relativa all’educazione è infatti molto usata per il tutoring intelligente, ma non viene integrata con SD per simulare strategie di riforma educativa a livello sistemico. Gli studi SD in ambito educativo sono ancora basati su modelli statici e non sfruttano il potenziale predittivo dell’AI.

L’AI spesso riproduce e amplifica bias presenti nei dati, generando discriminazioni e decisioni ingiuste. Ma l’impatto sistemico dei bias non è ben studiato. La nuova struttura implementata AI-SD può calmierare questo problema. L’AI può correggere bias nei dati utilizzando modelli di fairness e bias detection, mentre l’SD può simulare l’impatto dei bias, modellando come decisioni distorte dell’AI influenzano l’economia, la società e la politica.

Nonostante l'interesse crescente per l'integrazione tra AI e SD, la mancanza di una base metodologica solida e condivisa rappresenta un ostacolo rilevante allo sviluppo di una ricerca coerente e applicabile su larga scala.

Ogni studio che integra AI e SD tende a utilizzare approcci unici, ad hoc, spesso non replicabili né generalizzabili. Non esistono linee guida metodologiche standard su come e quando usare l'AI in un modello SD, su come interpretare l'impatto dell'AI sulla struttura dei modelli SD e su come bilanciare causalità sulla predizione. Ad esempio, è possibile che vengano utilizzate reti neurali per stimare i parametri di un modello SD, mentre altri potrebbero usare decision trees per generare variabili, ma non esiste un linguaggio comune e nemmeno un protocollo scientifico condiviso.

Tutti i modelli integrati oggettto di studio nella nostra ricerca sono stati costruiti per uno specifico caso di studio, ma mancano di una documentazione chiara, di un codice open-source, di dataset pubblici e di metodologie replicabili, risultando in una difficile validazione o miglioria dei lavori esistenti, frenando l'evoluzione della disciplina.

AI e SD, inoltre, utilizzano ambienti diversi e scarsamente interoperabili, e non esistono piattaforme native che combinano facilmente simulazione SD e algoritmi AI, difatti costringendo i ricercatori a costruire soluzioni "artigianali", complicando lo sviluppo, la condivisione e la manutenzione dei modelli.

Non esistono poi criteri standard per valutare l'efficacia dell'integrazione AI-SD. È molto difficile misurare la qualità dei modelli ibridi, rendendo impossibile confrontarli tra di loro e capire in modo univoco quali possano essere i vantaggi di una tale implementazione. Ogni autore propone le proprie metriche, ma ciò rende i risultati non empirici.

Tutti questi aspetti rendono molto complicato andare avanti nella ricerca in questo ambito, nonostante ci siano delle ottime premesse e diverse motivazioni valide per sviluppare queste nuove tecnologie.

4.4 Opportunità di ricerca futura

Le opportunità di ricerca sono varie, ma sono strettamente collegate ai gap di ricerca identificate. Queste opportunità si collocano su diversi livelli: metodologico, teorico, applicativo e interdisciplinare. In questo paragrafo cercheremo di sviscerare a pieno le varie tematiche.

Iniziamo parlando dello sviluppo di framework ibridi. Attualmente, come già spiegato, le due metodologie operanno in parallelo, ma raramente in modo profondamente integrato. Un framework ibrido fornirebbe una struttura coerente per combinare l'apprendimento dai dati con la modellazione strutturale e causale, offrire predizioni accurate e spiegabili e affrontare i problemi complessi nei sistemi aziendali, sociali, ambientali e tecnologici in modo più robusto. In altre parole, un framework ibrido permetterebbe di usare al meglio i due mondi.

Un buon framework dovrebbe integrare l'AI nel ciclo di modellazione SD, stimando parametri, variabili latenti o comportamenti per poi permettere al SD di usare queste informazioni per simulare dinamiche di lungo termine. Dovrebbe poi permettere l'adattamento dinamico dei modelli facendo aggiornare dall'AI il modello SD in base ai nuovi dati e facendo seguire a quest'ultimo la logica causale e strategica delle simulazioni. Inoltre, dovrebbe supportare l'interpretabilità e la validazione del modello, evitando che quest'ultimi restino black-box e fornendo strumenti per spiegare i risultati sia ai tecnici che ai decisori.

Le componenti di un framework Ibrido AI-SD dovrebbe essere composto da uno strato System Dynamics dove inseriamo i diagrammi causali, le struttuure di Stock & Flow e le equazioni dinamiche che descrivono le relazioni tra le variabili nel tempo, uno strato AI dove possiamo implementare il machine learning per stimare parametri o variabili

dinamiche, la causal AI per supportare la costruzione della struttura causale, il reinforcement learning per apprendere le strategie ottimali in ambienti simulati, l'NLP/Computer Vision per integrare dati non strutturati nei modelli, e infine una interfaccia di scambio AI-SD, con dei meccanismi che permettono all'AI di fornire input dinamici al modello SD, correggere o suggerire modifiche strutturali al modello e aggiornare automaticamente i parametri sulla base dei dati.

Un'altra opportunità di ricerca interessante è sicuramente la costruzione di modelli SD potenziati da AI in contesti dinamici e incerti. Sviluppare modelli SD in grado di adattarsi automaticamente a contesti in rapido cambiamento, porterebbe sicuramente enormi vantaggi dal punto di vista di accuratezza delle simulazioni. L'AI può essere utilizzata per estrarre dati in tempo reale, adatttare i parametri dei mdelli SD e identificare i pattern non previsti che modificano la struttura del sistema. Un esempio portante dell'utilità di questa innovazione potrebbe essere il modello SD dell'evoluzione di una pandemia, aggiornando i tassi di contagio in tempo reale, in modo da rendere più realistica la simulazione. Oppure immaginiamo dei sistemi aziendali che si riorganizzano automaticamente in base a dati di performance o domanda.

Un'altro percorso di ricerca molto interessante potrebbe essere l'utilizzo dell'AI per identificare, apprendere e aggiornare anelli di feedback. Si potrebbe usare l'AI per scoprire nuove strutture causali o ottimizzare quelle esistenti nei modelli SD. Questo rappresenta un passo verso la "hard convergence", citata nel capitolo precedente, dove l'AI contribuisce a costruire la logica del modello, non solo i dati in input. Sarebbe possibile studiare questa strada applicando tecniche di causal discovery per identificare relazioni causa-effetto a partire dai dati, meta-learning per adattare le strutture dei modelli SD o il reinforcement learning per aggiornare i loop di feedback in funzione dei risultati. Ad esempio,

potremmo considerare un'AI che apprende dai dati aziendali e suggerisce la creazione di nuovi loop tra variabili chiave, oppure un modello SD che auto-corregge i suoi feedback in base alle performance predittive.

Un'ulteriore strada da approfondire potrebbe essere lo sviluppo di modelli predittivo-simulativi per la gestione della complessità. Si possono infatti costruire modelli che combinano la potenza predittiva dell'AI con la capacità simulativa della SD, per gestire sistemi complessi, integrando AI per prevedere eventi futuri o trend, usare SD per simulare diversi scenari e testare strategie a lungo termine. Ad esempio, un sistema predittivo-simulativo per la gestione dell'energia elettrica che prevede la domanda tramite AI e simula l'impatto delle politiche energetiche con SD, oppure dei modelli integrati per la resilienza urbana, che combinano dati IoT con simulazioni di comportamento urbano

Ancora, si potrebbe pensare alla progettazione di strumenti decisionali AI-SD utilizzabili da manager e policy maker. Si potrebbe tradurre l'integrazione AI-SD in strumenti pratici, accessibili a utenti non tecnici per supportare il processo decisionale in contesti complessi, sviluppando dashboard interattive che mostrano i risultati delle simulazioni e predizioni in modo chiaro, e integrare il framework con sistemi aziendali esistenti. Ad esempio, un sistema di supporto alle politiche sanitarie che mostra in tempo reale gli effetti a lungo termine di diverse scelte, sulla base di dati AI e modelli SD, oppure una dashboard per la gestione dei rischi aziendali, che prevede shock di mercato e simula piani di risposta.

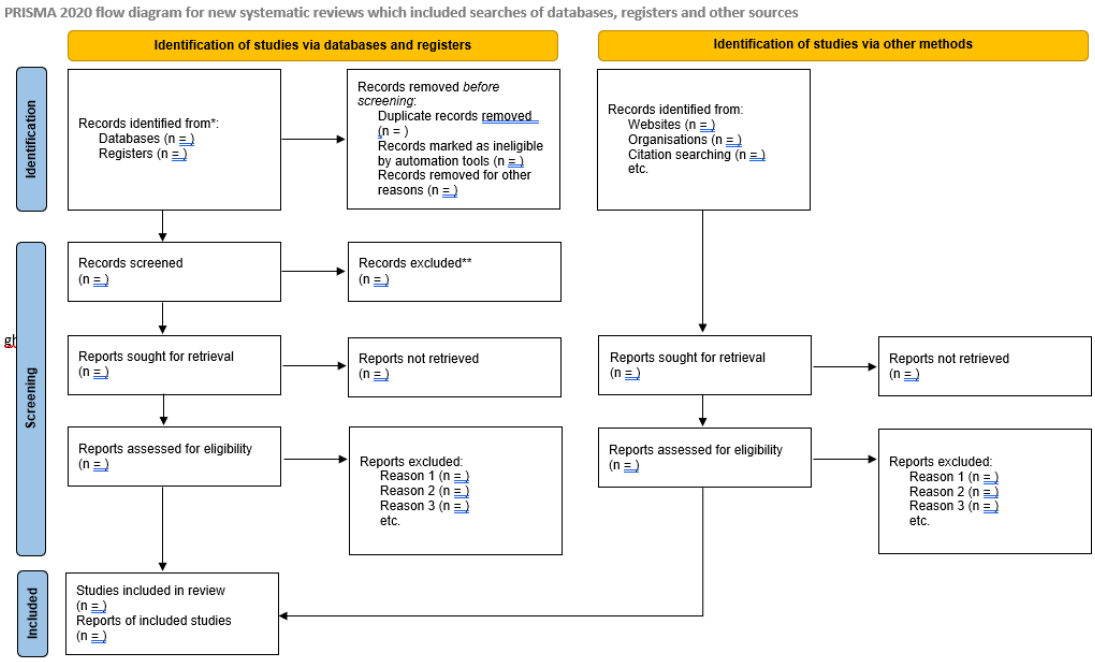
Come ultimo aspetto da poter approfondire nella ricerca futura può essere la costruzione di strumenti e ambienti integrati per la modellazione AI-SD, per quindi superare la frammentazione tra le due metodologie, creando ambienti di sviluppo unificati. Sarebbe il caso quindi di sviluppare librerie open-

source o API che permettano di collegare facilmente modelli SD e AI, creare linguaggi di modellazione ibridi o interfacce grafiche condivise.

In conclusione, l'integrazione tra AI e SD non rappresenta solo una frontiera metodologica, ma anche una leva strategica per affrontare la complessità dei sistemi contemporanei. Affinché tale convergenza si realizzi pienamente, sarà essenziale promuovere standard condivisi, linguaggi comuni e collaborazione tra comunità scientifiche oggi ancora troppo distanti.

5. Appendici

5.1 Diagramma prisma



*Consider, if feasible to do so, reporting the number of records identified from each database or register searched (rather than the total number across all databases/registers).
**If automation tools were used, indicate how many records were excluded by a human and how many were excluded by automation tools.

Source: Page MJ, et al. BMJ 2021;372:n71. doi: 10.1136/bmj.n71.

5.2 Tabella estesa degli articoli approfonditi

Titolo	**Autori**	**Anno di pubblicazione**
Modeling the effects of artificial intelligence (AI)-based innovation on sustainable development goals (SDGs): Applying a system dynamics perspective in a cross-country setting	Sharmin Nahar	2024
Multi-scenario prediction and path optimization of industrial carbon unlocking in China	Feifei Zhao, Zheng Hu, Xu Zhao	2023
AI USEFULNESS IN SYSTEMS MODELLING AND SIMULATION: GPT-4 APPLICATION	C. du Plooy, R. Oosthuizen	2023
Exploration of Artificial Intelligence-oriented Power System Dynamic Simulators	Tannan Xiao, Ying Chen, Jianquan Wang, Shaowei Huang, Weilin Tong, Tirui He	2022
AI Concepts for System of Systems Dynamic Interoperability	Jacob Nilsson, Saleha Javed, Kim Albertsson, Jerker Delsing , Marcus Liwicki and Fredrik Sandin	2024
Towards a System Dynamics Framework for Human–Machine Learning Decisions: A Case Study of New York Citi Bike	Ganesh Sankaran, Marco A. Palomino, Martin Knahl and Guido Siestrup	2024
Optimizing System Behavior using Genetic Algorithms	Robert M Sholtes	1994
Evaluating human error probability in maintenance task: An integrated system dynamics and machine learning approach	Vahideh Bafandegan Emroozi, Mostafa Kazemi, Alireza Pooya, Mahdi Doostparast	2024

Zooming in and out the landscape: Artificial intelligence and system dynamics in business and management	Stefano Armenia, Eduardo Franco, Francesca Iandoloc, Giuliano Maielli, Pietro Vito	2023
Optimisation of recovery policies in the era of supply chain disruptions: a system dynamics and reinforcement learning approach	Fabian Bussieweke, Josefa Mula & Francisco Campuzano-Bolarin	2024
Sustainable Development of Port Economy Based on Intelligent System Dynamics	GUOYI XIU, ZEXU ZHAO	2021
Evaluating system dynamics models of risky projects using decision trees: alternative energy projects as an illustrative example	Burcu Tan, Edward G. Anderson Jr., James S. Dyera and Geoffrey G. Parker	2009
Pattern-oriented analysis of system dynamics models via random forests	Mert Edali	2022
Incorporating Deep Learning Into System Dynamics: Amortized Bayesian Inference for Scalable Likelihood-Free Parameter Estimation	Hazhir Rahmandad, Ali Akhavan, Mohammad S. Jalali	2024
Amethodology for parameter estimation in system dynamics models using artificial intelligence	Jyotirmay Gadewadikar, Jeremy Marshall	2023

Bibliografia

Aamodt, A., Nygård, M., 1995. Different roles and mutual dependencies of data, information, and knowledge—an AI perspective on their integration. Data Knowl. Eng. 16 (3), 191–222.

Adya, M., Collopy, F., 1998. How effective are neural networks at forecasting and prediction? A review and evaluation. J. Forecast. 17 (5-6), 481–495.

Akaev, A., Devezas, T., Ichkitidze, Y., Sarygulov, A., 2021. Forecasting the labor intensity and labor income share for G7 countries in the digital age. Technol. Forecast. Soc. Chang. 167, Article 120675 https://doi.org/10.1016/j.techfore.2021.120675.

Alkaldy, E.A.H., Albaqir, M.A., Hejazi, M.S.A., 2019. A new load forecasting model considering planned load shedding effect. Int. J. Energy Sect. Manag. 13 (1), 149–165.

Antons, D., Kleer, R., Salge, T.O., 2016. Mapping the topic landscape of jpim, 1984–2013: in search of hidden structures and development trajectories. J. Prod. Innov. Manag. 33 (6), 726–749.

Armenia, S., 2019. Smart model-based governance: taking decision making to the next level by integrating data analytics with systems thinking and system dynamics. In: New Challenges in Corporate Governance: Theory and Practice, pp. 41–42. https:// doi.org/10.22495/ncpr_10.

Armenia, S., Ferreira, Franco E., Mecella, M., Onori, R., 2017. Smart model-based governance: from big-data to future policy making. In: Proceedings of the BSLab- SYDIC Workshop 2017, Rome (Italy). ISBN 9788890824258.

Armenia, Stefano, Eduardo Franco, Francesca Iandolo, Giuliano Maielli, e Pietro Vito. 2024. “Zooming in and Out the Landscape: Artificial Intelligence and System Dynamics in Business and Management.” Technological Forecasting & Social Change 2024: 123131. https://doi.org/10.1016/j.techfore.2023.123131.

Azadeh, A., Darivandi Shoushtari, K., Saberi, M., & Teimoury, E. (2014). An integrated artificial neural network and system dynamics approach in support of the viable system model to enhance industrial intelligence: the case of a large broiler industry. Syst. Res. Behav. Sci., 31(2), 236–257.

Badakhshan, E., Humphreys, P., Maguire, L., & McIvor, R. (2020). Using simulation- based system dynamics and genetic algorithms to reduce the cash flow bullwhip in the supply chain. Int. J. Prod. Res., 58(17), 5253–5279.

Bafandegan Emroozi, Vahideh, Mostafa Kazemi, Alireza Pooya, e Mahdi Doostparast. 2024. "Evaluating Human Error Probability in Maintenance Task: An Integrated System Dynamics and Machine Learning Approach." Human Factors and Ergonomics in Manufacturing & Service Industries 35 (1): e21057. https://doi.org/10.1002/hfm.21057.

Bussieweke, Fabian, Josefa Mula, and Francisco Campuzano-Bolarin. 2024. "Optimisation of Recovery Policies in the Era of Supply Chain Disruptions: A System Dynamics and Reinforcement Learning Approach." International Journal of Production Research, published August 6, 2024. https://doi.org/10.1080/00207543.2024.2383293.

Caponio, G., Massaro, V., Mossa, G., Mummolo, G., 2015. Strategic energy planning of residential buildings in a smart city: a system dynamics approach. Int. J. Eng. Bus. Manag. 7, 20

Chung, B., 2018. System dynamics modelling and simulation of the Malaysian rice value chain: effects of the removal of price controls and an import monopoly on rice prices and self-sufficiency levels in Malaysia. Syst. Res. Behav. Sci. 35 (3), 248–264.

du Plooy, Corne, e Rikus Oosthuizen. 2023. "AI Usefulness in Systems Modelling and Simulation: GPT-4 Application." South African Journal of Industrial Engineering 34 (3): 286–303. https://doi.org/10.7166/34-3-2944.

Dyner, I., Smith, R.A., Pe˜na, G.E., 1995. System dynamics modelling for residential energy efficiency analysis and management. J. Oper. Res. Soc. 46, 1163–1173.

Dyson, B., Chang, N.B., 2005. Forecasting municipal solid waste generation in a fast- growing urban region with system dynamics modeling. Waste Manag. 25 (7), 669–679

Edali, Mert. 2022. "Pattern-Oriented Analysis of System Dynamics Models via Random Forests." System Dynamics Review 38 (2): 135–166. https://doi.org/10.1002/sdr.1706.

Ford, A., 1997. System dynamics and the electric power industry. Syst. Dyn. Rev. 13 (1), 57–85.

Forrester, J.W., 2007. System dynamics—the next fifty years. Syst. Dyn. Rev. 23 (2–3), 359–370.

Gadewadikar, Jyotirmay, and Jeremy Marshall. 2023. "A Methodology for Parameter Estimation in System Dynamics Models Using Artificial Intelligence." Systems Engineering 27 (2): 253–266. https://doi.org/10.1002/sys.21718.

Ghaffarzadegan, N., Lyneis, J., Richardson, G.P., 2011. How small system dynamics models can help the public policy process. Syst. Dyn. Rev. 27 (1), 22–44.

Huang, L., Zimmerm, B., Hasan, J., 2007. System dynamics model for renewable enerpry: case from a country. In: Proceedings of the 2007 Conference on Systems Science, Management Science and System Dynamics: Sustainable Development and Complex Systems, vols 1-10, pp. 793–799.

Huang, L., Zimmerm, B., Hasan, J., 2007. System dynamics model for renewable enerpry: case from a country. In: Proceedings of the 2007 Conference on Systems Science, Management Science and System Dynamics: Sustainable Development and Complex Systems, vols 1-10, pp. 793–799.

Kabir, C., Sharif, M.N., Adulbhan, P., 1981. System dynamics modeling for forecasting technological substitution. Comput. Ind. Eng. 5 (1), 7–21.

Kunc, M., Mortenson, M.J., Vidgen, R., 2018. A computational literature review of the field of System Dynamics from 1974 to 2017. J. Simul. 12 (2), 115–127.

Kunsch, P., Springael, J., 2008. Simulation with system dynamics and fuzzy reasoning of a tax policy to reduce CO2 emissions in the residential sector. Eur. J. Oper. Res. 185 (3), 1285–1299. https://doi.org/10.1016/j.ejor.2006.05.048.

Liu, C., Mak, V., Rapoport, A., 2015. Cost-sharing in directed networks: experimental study of equilibrium choice and system dynamics. J. Oper. Manag. 39, 31–47.

Lyneis, J.M., 2000. System dynamics for market forecasting and structural analysis. Syst. Dyn. Rev. 16 (1), 3–25.

Nahar, Sharmin. 2024. “Modeling the Effects of Artificial Intelligence (AI)-Based Innovation on Sustainable Development Goals (SDGs): Applying a System Dynamics Perspective in a Cross-Country Setting.” Technological Forecasting and Social Change 201: 123203. https://doi.org/10.1016/j.techfore.2023.123203.

Nilsson, Jacob, Saleha Javed, Kim Albertsson, Jerker Delsing, Marcus Liwicki, e Fredrik Sandin. 2024. “AI Concepts for System of Systems Dynamic Interoperability.” Sensors 24 (9): 2921. https://doi.org/10.3390/s24092921.

Rahmandad, Hazhir, Ali Akhavan, and Mohammad S. Jalali. 2025. “Incorporating Deep Learning into System Dynamics: Amortized Bayesian Inference for Scalable Likelihood-Free Parameter Estimation.” System Dynamics Review 41 (1): e1798. https://doi.org/10.1002/sdr.1798.

Rashwan, W., Abo-Hamad, W., Arisha, A., 2015. A system dynamics view of the acute bed blockage problem in the Irish healthcare system. Eur. J. Oper. Res. 247 (1), 276–293.

Reddi, K.R., Moon, Y.B., 2011. System dynamics modeling of engineering change management in a collaborative environment. J. Adv. Manuf. Technol. 55 (9–12), 1225–1239

Sankaran, Ganesh, Marco A. Palomino, Martin Knahl, e Guido Siestrup. 2024. “Towards a System Dynamics Framework for Human–Machine Learning Decisions: A Case Study of New York Citi Bike.” Applied Sciences 14 (22): 10647. https://doi.org/10.3390/app142210647.

Sholtes, Robert M. 1994. “Optimizing System Behavior Using Genetic Algorithms.” Paper presented at the 1994 International System Dynamics Conference, Sterling, Scotland, July 1994.

Shrestha, Y.R., Krishna, V., von Krogh, G., 2021. Augmenting organizational decision- making with deep learning algorithms: principles, promises, and challenges. J. Bus. Res. 123, 588–603.

Sterman, John D. Business Dynamics: Systems Thinking and Modeling for a Complex World. Boston: Irwin/McGraw-Hill, 2000.

Tan, Burcu, Edward G. Anderson Jr., James S. Dyer, and Geoffrey G. Parker. 2010. “Evaluating System Dynamics Models of Risky Projects Using Decision Trees: Alternative Energy Projects as an Illustrative Example.” System Dynamics Review 26 (1): 1–17. https://doi.org/10.1002/sdr.433.

Toorajipour, R., Sohrabpour, V., Nazarpour, A., Oghazi, P., Fischl, M., 2021. Artificial intelligence in supply chain management: a systematic literature review. J. Bus. Res. 122, 502–517. https://doi.org/10.1016/j.jbusres.2020.09.009.

Wirth, N., 2018. Hello marketing, what can artificial intelligence help you with? Int. J. Mark. Res. 60 (5), 435–438. https://doi.org/10.1177/1470785318776841.

Xia, M., Stallaert, J., Whinston, A.B., 2005. Solving the combinatorial double auction problem. Eur. J. Oper. Res. 164 (1), 239–251. https://doi.org/10.1016/j. ejor.2003.11.018.

Xiao, Tannan, Ying Chen, Jianquan Wang, Shaowei Huang, Weilin Tong, e Tirui He. 2023. "Exploration of Artificial Intelligence-Oriented Power System Dynamic Simulators." arXiv preprint arXiv:2110.00931. https://doi.org/10.48550/arXiv.2110.00931.

Xiu, Guoyi, and Zexu Zhao. 2021. "Sustainable Development of Port Economy Based on Intelligent System Dynamics." IEEE Access 9: 14070–14077. https://doi.org/10.1109/ACCESS.2021.3051065.

Xu, J.J., Babaian, T., 2021. Artificial intelligence in business curriculum: the pedagogy and learning outcomes. Int. J. Manag. Educ. 19 (3) https://doi.org/10.1016/j. ijme.2021.100550.

Zhang, L.X., Pentina, I., Fan, Y.H., 2021. Who do you choose? Comparing perceptions of human vs robo-advisor in the context of financial services. J. Serv. Mark. 35 (5), 628–640. https://doi.org/10.1108/jsm-05-2020-0162.

Zhao, Feifei, Zheng Hu, e Xu Zhao. 2023. "Multi-Scenario Prediction and Path Optimization of Industrial Carbon Unlocking in China." Journal of Cleaner Production 421: 138534. https://doi.org/10.1016/j.jclepro.2023.138534.

Zhao, J., Wu, G., Xi, X., Na, Q., Liu, W., 2018. How collaborative innovation system in a knowledge-intensive competitive alliance evolves? An empirical study on China, Korea and Germany. Technol. Forecast. Soc. Chang. 137, 128–146.

Sitografia

IBM. 2024. “Cos’è il Machine Learning?” IBM. https://www.ibm.com/it-it/topics/machine-learning

MathWorks. 2024. “Deep Learning.” MathWorks Italia. https://it.mathworks.com/discovery/deep-learning.html

MathWorks. 2024. “Reinforcement Learning.” MathWorks Italia. https://it.mathworks.com/discovery/reinforcement-learning.html

Osservatori Digital Innovation. 2023. “Storia dell’Intelligenza Artificiale: Dalle Origini a Oggi.” Osservatori.net. https://blog.osservatori.net/it_it/storia-intelligenza-artificiale

Parlamento Europeo. 2020. “Che Cos'è l'Intelligenza Artificiale e Come Viene Usata.” Parlamento Europeo. https://www.europarl.europa.eu/topics/it/article/20200827STO85804/che-cos-e-l-intelligenza-artificiale-e-come-viene-usata

PRISMA Statement. 2024. “Transparent Reporting of Systematic Reviews and Meta-Analyses.” PRISMA. https://www.prisma-statement.org/