UNIVERSITÀ
POLITECNICA
DELLE MARCHE

FACOLTA' DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA E DELL'AUTOMAZIONE

Simulazione numerica di un neuro-controllore applicato ad un pendolo inverso

Simulation of neuro-controller applied to an inverse pendulum

Relatore:

Prof. Simone Fiori

Candidata:

Lucia Silla

Anno Accademico 2021-2022

Ringraziamenti

Voglio ringraziare il mio relatore, il Professore Simone Fiori, per la disponibilità e per avermi guidata in questi ultimi mesi di questo percorso.
Vorrei ringraziare, inoltre, tutte le persone che sono state presenti, a partire dai miei genitori, che mi hanno permesso di fare questa esperienza e per avermi sostenuto, soprattutto economicamente, e vorrei anche rassicurarli dicendogli che questa tesi rappresenta il massimo rimborso spese che posso fare in questo momento.
Mia sorella Federica e Vinicio, per essere stati la principale fonte di svago e anche la principale fonte di un Wi-Fi decente durante gli esami on-line, ma soprattutto per il supporto, la pazienza e le notti sul divano.
Ringrazio il mio ragazzo e collega, con il quale ho condiviso questa esperienza, per tutte le volte in cui mi ha ascoltata, lamentandosi, ripetere per un esame e per tutte le volte che ha provato a placare le mie ansie.
I miei amici dell'università, che tra lezioni, aule studio, attese per gli esami e attese per i risultati degli esami, bar dell'università e bar fuori dall'università, abbiamo condiviso bei momenti, che hanno arricchito e reso sicuramente più bella questa esperienza e senza i quali, probabilmente, mi sarei laureata in tempo. Ringrazio, infine, gli amici di sempre, dai quali tornavo ogni volta e con i quali ho festeggiato ogni singola verbalizzazione.

Indice

Introduzione

La progettazione del controllore solitamente implica una complessa analisi matematica e una difficoltà nel controllo delle non linearità. Per aggirare tali difficoltà il numero di nuovi approcci, che utilizzano le reti neurali per compiti di controllo, è aumentato significativamente negli ultimi anni.
L'uso della capacità di apprendimento delle reti neurali aiuta la progettazione del controllore ad essere piuttosto flessibile, soprattutto quando la dinamica del sistema è complessa e altamente non lineare. Questo, dunque, è un vantaggio rispetto al metodo di controllo tradizionale. [1]
Poiché le tecniche di progettazione dei sistemi dinamici sono strettamente correlate alla loro proprietà di stabilità e poiché le condizioni necessarie e sufficienti per la stabilità dei sistemi lineari sono state generate e rese ben note, sono stati stabiliti metodi di progettazione per tali sistemi. Al contrario, per sistemi non lineari, le procedure di progettazione che soddisfano contemporaneamente i requisiti di stabilità, robustezza e buona risposta dinamica, non sono disponibili per grandi classi di questi sistemi.
Poiché esistono pochissimi risultati nella teoria dei sistemi non lineari che possono essere applicati direttamente, si deve esercitare una notevole attenzione alle dichiarazioni dei problemi, la scelta delle strutture identificative e di controllo.
Le due reti neurali che hanno ricevuto notevole attenzione nei compiti di controllo sono:

1) Reti neurali multistrato
2) Reti ricorrenti

Le reti multistrato si sono dimostrate estremamente efficaci nei problemi di riconoscimento di modelli, mentre quelle ricorrenti sono state usate per la soluzione di problemi di ottimizzazione. Durante gli ultimi due decenni, il controllo non lineare dei sistemi dinamici che utilizzano reti neurali ha attirato un enorme interesse negli studi di controllo. Originariamente ispirato dalle capacità degli esseri umani di eseguire con facilità molte attività complicate in ambienti incerti, il controllo neurale si è sviluppato per far fronte alla crescente domanda per il controllo di sistemi complessi, altamente incerti, non lineari in applicazioni industriali.
Dopo la prima fase di sviluppo, in cui le tecniche di ottimizzazione sono state usate principalmente per leggi di adattamento dei parametri, con pochi risultati analitici per la stabilità e le prestazioni, la ricerca attuale si basa principalmente sulla teoria della stabilità di Lyapunov (anche noto come metodo di linearizzazione) nel contesto del controllo neurale adattivo, in cui le caratteristiche principali includono anche la stabilità e le prestazioni del sistema di controllo.

Sono stati fatti molti progressi sia in teoria che nelle applicazioni pratiche, tuttavia rimangono alcune questioni da affrontare, ad esempio quali altre proprietà specifiche delle reti neurali debbano essere sfruttate per rendere il controllo neurale distintivo con gli altri metodi di controllo. [2]
In questa tesi viene esposta l'applicazione dell'approccio neurale al controllo di sistemi instabili non lineari. L'architettura di apprendimento è proposta per allenare un neuro controllore in grado di fornire un'appropriata forza necessaria a bilanciare il pendolo inverso e guidare il carrello al centro della pista, dimostrando l'efficacia del metodo di apprendimento.

Capitolo 1

Il problema del pendolo inverso

Il pendolo inverso rappresenta un classico esempio di sistema intrinsecamente instabile. Le sue dinamiche sono fondamentali per i compiti che coinvolgono il mantenimento dell'equilibrio.
Molte tecniche di progettazione di controllo sono state studiate utilizzando il pendolo.
La riuscita di queste tecniche di applicazione richiede una notevole conoscenza del sistema e un'espressione del comportamento desiderato, di solito, sottoforma di funzione obiettivo.
Il problema del controllo del pendolo, presentato di seguito, è trattato nel caso in cui la dinamica del sistema non è nota a priori e la funzione obiettivo non è data. Tutto quello di cui si è a conoscenza, sono i valori e i ranges delle variabili di stato del pendolo e un segnale di fallimento, che deve essere minimizzato nel tempo. [3]
Il problema di bilanciamento che viene presentato, è stato affrontato tramite l'utilizzo di reti neurali multistrato.

1.1 Ambiente del pendolo inverso

L'ambiente del pendolo inverso è costituito da un sistema carrello-asta (figura 1).

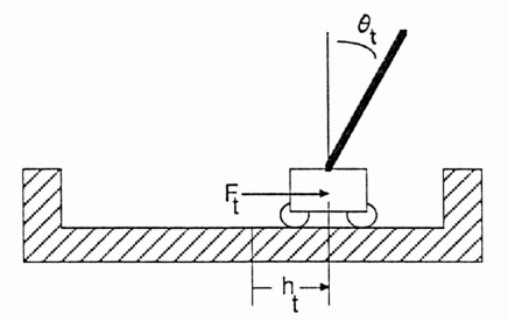

Figura 1 sistema pendolo inverso

Il carello, se lo consideriamo posto in un piano cartesiano, può muoversi sia verso valori positivi che verso valori negativi delle ascisse, restando all'interno di due valori limite che gli vengono assegnati. Mentre l'asta, che è incernierata al carrello, oscilla con valori positivi e negativi dell'angolo, che essa forma con la verticale. Anche gli angoli sono sottoposti a limiti assegnati.

Lo stato del sistema al tempo t è specificato da quattro variabili reali: l'angolo che l'asta forma con la verticale e la velocità angolare (θ, $\dot{\theta}$) e la posizione orizzontale e la velocità del carrello (h, $\dot{h}$).
Il sistema carrello-asta è stato simulato usando le seguenti equazioni del moto:

$$\ddot{\theta}_t = \frac{g * sen\theta_t + cos\theta_t[\frac{-Ft - mp * l * \dot{\theta}_t^{\,2} * sen\theta_t}{mc + mp}]}{l * [\frac{4}{3} - \frac{mp * cos^2\theta_t}{mc + mp}]}$$

$$\ddot{h}_t = \frac{Ft + mp * l * [\dot{\theta}_t^{\,2} * sen\theta_t - \ddot{\theta}_t * cos\theta_t]}{mc + mp}$$

dove θ, h, t sono rispettivamente in radianti, metri e secondi, e dove g è l'accelerazione di gravità (9.8 m/s^2), Ft è la forza applicata al carrello (-/+ 10 Nt) mc la massa del carrello (1.0 kg), mp la massa del pendolo (0.1 kg) e l è la distanza dal centro di massa dell'asta al pivot (0.5 m).

1.2 Soluzione al problema del pendolo inverso

Come già accennato in precedenza, la soluzione al problema del bilanciamento del pendolo è stata applicata tramite una rete neurale multistrato.
Senza una funzione obiettivo, necessaria a valutare stati e azioni, le modifiche al controllore possono essere basate solo sul segnale di fallimento.
Nell'intervallo di tempo tra un fallimento e un altro può, però, verificarsi una

lunga sequenza di azioni con conseguente difficoltà ad assegnare un "giudizio", necessario per descrivere quali azioni della sequenza hanno contribuito al fallimento.
Per risolvere questo problema vengono sviluppate due funzioni: la funzione azione, destinata a mappare lo stato attuale in azioni di controllo, e la funzione valutazione, utilizzata per assegnare crediti alle singole azioni. Queste due funzioni vengono apprese rispettivamente dalla rete d'azione e dalla rete di valutazione.
L'architettura complessiva (figura 2) della rete è data, dunque, dalle due reti, azione e valutazione, che costituiscono il livello di output, e un secondo strato, livello nascosto, a cui le due reti sono collegate.

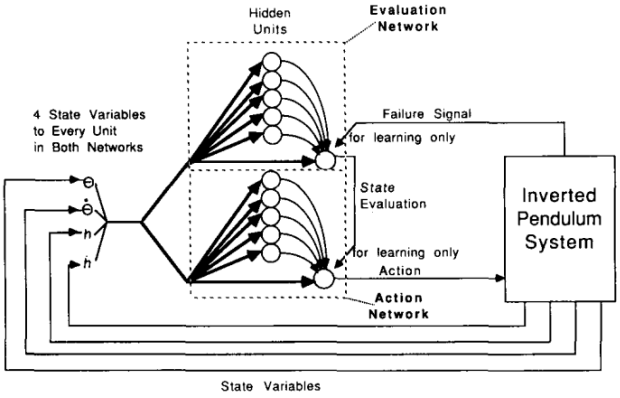

Figura 2 architettura della rete multistrato

1.2.1 Rete Azione e Rete Valutazione

Per assegnare crediti alle singole azioni di una sequenza che precede il fallimento, abbiamo visto che la rete di valutazione deve apprendere una funzione. Per fare questo viene usata una generalizzazione del metodo di apprendimento AHC (adaptive heuristic critic) [4], un metodo della classe di predizione chiamata "differenze-temporali" [5].
AHC sviluppa la funzione di valutazione, dove il suo valore per un dato stato rappresenta una predizione di un futuro fallimento.
Il valore della funzione di valutazione v viene combinato con il segnale di fallimento r, che può assumere solamente due valori (es: 0 e -1), formando $\hat{r}$.

L'AHC rappresenta un tipo di apprendimento supervisionato, dove $\hat{r}$ gioca il ruolo dell'errore. Valori positivi di $\hat{r}$ rappresentano un incremento della valutazione dello stato precedente e, allo stesso modo, valori negativi di $\hat{r}$ rappresentano un decremento della valutazione dello stato precedente.
Per quanto riguarda le unità nascoste della rete di valutazione, la procedura delle modifiche è simile al metodo della back-propagation, dove gli errori assegnati alle unità di primo livello si basano su $\hat{r}$.
L'apprendimento supervisionato non può essere utilizzato per la rete d'azione in quanto non si conosce l'azione corretta, sono, infatti, possibili molte azioni che non portano ad un conseguente fallimento.
Per la rete d'azione viene utilizzato l'apprendimento per rinforzo nel seguente modo: il livello di output viene modificato in modo tale da incrementare la probabilità di un'azione seguita da un valore positivo di $\hat{r}$, e in maniera analoga decrementare la probabilità d'azione seguita da un valore negativo di $\hat{r}$. Il cambiamento della probabilità è proporzionale al valore di $\hat{r}$ e alla differenza tra l'azione e il valore atteso dell'azione.
Le unità nascoste, della rete di azione, sono aggiornate, come nel caso della rete di valutazione, con il metodo della back-propagation dove in questo caso il ruolo dell'errore è dato dal prodotto di $\hat{r}$ e la differenza tra l'azione e il suo valore atteso. [6]

Capitolo 2

Simulazione del problema

Il successo del compito consiste nell'applicare una sequenza di forze a destra o a sinistra, di grandezza fissa, al carrello in maniera tale da bilanciare l'asta e in modo da mantenere il carrello entro i limiti della pista. La forza di grandezza nulla non è consentita.
L'unica informazione per valutare il successo del compito consiste nel segnale di fallimento, che si verifica quando il carrello urta i limiti della pista, posti a $\pm 2.4\ metri$, e quando l'angolo θ supera i $\pm 0.21\ radianti$.
Questo segnale viene definito come segue:

$$\mathrm{r[t]} = \begin{cases} -1 & , se\ |\theta[t]|\ >\ 0.21\ rad\ o\ |h[t]|\ > 2.4m \\ 0 & altrimenti \end{cases}$$

Lo stato del sistema carrello-asta è stato presentato al sistema di apprendimento come una versione scalata delle variabili di stato:

- $x_1[t] = \frac{1}{4.8}(h[t] + 2.4)$
- $x_2[t] = \frac{1}{3}\left(\dot{h}[t] + 1.5\right)$
- $x_3[t] = \frac{1}{0.42}(\theta[t] + 0.21)$
- $x_4[t] = \frac{1}{4}\left(\dot{\theta}[t] + 2\right)$
- $x_5[t] = 0.5$

dove h[t], $\dot{h}[t]$, $\theta[t]$, $\dot{\theta}[t]$, sono discretizzate nel tempo.
Gli input $x_1[t]$, $x_3[t]$, possono assumere valori in un range da 0 a 1, mentre $x_2[t]$, $x_4[t]$, possono assumere valori che eccedono tale range. Questo è dovuto alle imposizioni fatte sui limiti dell'angolo e della posizione del carrello.
L'input $x_5[t]$ rappresenta la costante di bias.

Questo ridimensionamento è necessario per due motivi; poiché gli algoritmi di apprendimento coinvolgono i termini di inputi $x_i[t]$, come fattori nelle equazioni dell'aggiornamento dei pesi, termini con grandezze superiori ad altri avranno una maggiore influenza sull'apprendimento. Per far sì che ciò non avvenga, i termini di input sono stati ridimensionati affinché si trovino tutti nello stesso intervallo di valori. In secondo luogo, poiché i valori delle variabili di stato sono centrati a zero, e a causa della natura lineare delle unità della rete, l'azione corretta per valori positivi di θ e $\dot{\theta}$ li trasferirebbe a θ e $\dot{\theta}$ negativi nel giusto modo e viceversa. [6]

L'ambiente del sistema è stato, inoltre, simulato approssimando le equazioni del moto usando il metodo di Eulero con time step $\tau = 0.02$:

$$h[t+1] = h[t] + \tau\dot{h}[t]$$
$$\dot{h}[t+1] = \dot{h}[t] + \tau\ddot{h}[t]$$
$$\theta[t+1] = \theta[t] + \tau\dot{\theta}[t]$$
$$\dot{\theta}[t+1] = \dot{\theta}[t] + \tau\ddot{\theta}[t]$$

2.1 Simulazione Rete Valutazione

In primo luogo, vengono calcolate le uscite dalle unità nascoste come segue:

$$y_i[t_1, t_2] = g\left(\sum_{j=1}^{5} a_{ij}[t_1]x_j[t_2]\right)$$

per i = 1,2,3,4,5.
La funzione g, rappresenta la sigmoide, che è definita nel seguente modo:

$$g(s) = \frac{1}{1+e^{-s}}$$

La dipendenza dal doppio tempo è dovuta al fatto che nel confrontare un valore di y con un valore precedente, si deve fare attenzione ad evitare l'instabilità nella crescita dei valori dei pesi (le equazioni verranno presentate in seguito). Se il calcolo di y per il passo [t-1] usa a_{ij}[t-1] mentre y

per il passo t, usa a_{ij}[t], allora un cambiamento in y da un passo temporale al successivo potrebbe essere causato da un cambiamento nei valori dei pesi piuttosto che dall'incontro di uno stato con una diversa aspettativa di rinforzo. Per evitare questo, la coppia di y successivi si basa su un unico insieme di valori dei pesi, cioè, la differenza tra y per il passo t-1 e per il passo t è dovuta solo alla variazione da x_i[t-1] a x_i [t], poiché entrambe le y sono calcolate utilizzando a_{ij} [t-1]. [7]

Calcolato l'output delle unità nascoste, che rappresentano l'input della rete valutazione, l'uscita dalla rete è data da:

$$v[t_1, t_2] = \sum_{i=1}^{5} b_i[t_1] x_i[t_2] + \sum_{i=1}^{5} c_i[t_1]\, y_i[t_1, t_2]$$

il valore di v viene combinato con il segnale di fallimento per formare $\hat{r}$, che verrà poi utilizzato nelle equazioni per l'aggiornamento dei pesi:

$$\hat{r}[t+1] = \begin{cases} 0\ , & \textit{se lo stato al tempo } t+1 \text{ è } \textit{lo stato iniziale.} \\ r[t+1] - v[t,t]\ , & \textit{se lo stato al tempo } t+1 \text{ è } \textit{di fallimento} \\ r[t+1] + \gamma v[t,t+1] - v[t,t]\ , & \textit{altrimenti} \end{cases}$$

La rete valutazione, in generale, assume la seguente forma:

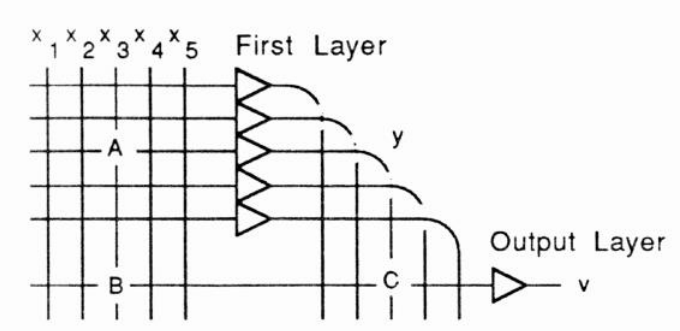

Figura 3 rete valutazione

A, B e C sono le matrici dei pesi e i singoli vengono indicati rispettivamente come a_{ij}, b_i, c_i.
Le equazioni che regolano i pesi sono date da:

$$b_i[t+1] = b_i[t] + \beta\hat{r}[t+1]x_i[t]$$
$$c_i[t+1] = c_i[t] + \beta\hat{r}[t+1]y_i[t,t]$$
$$a_{ij}[t+1] = a_{ij}[t] + \beta_h\hat{r}[t+1]y_i[t,t](1 - y_i[t,t])sgn(c_i[t])x_j[t]$$

Con $0 < \Upsilon \leq 1$ e $\beta, \beta_h > 0$.
Sgn rappresenta la funzione segno.

2.2 Simulazione Rete Azione

Come nel caso della rete di valutazione, il primo step consiste nel calcolare l'uscita dalle unità nascoste:

$$z_i[t] = g(\sum_{j=1}^{5} d_{ij}[t] * x_j[t])$$

Con i = 1,2,3,4,5.
Data $z_i[t]$, possiamo ricavare il valore di p[t], necessario a determinare la probabilità con cui deve essere effettuata un'azione. p[t] assume la seguente forma:

$$p[t] = g(\sum_{i=1}^{5} e_i[t] * x_i[t] + \sum_{i=1}^{5} f_i[t] * z_i[t])$$

da cui:

$$q[t] = \begin{cases} 1, & con\ probabilità\ p[t] \\ 0, & con\ probabiità\ 1 - p[t] \end{cases}$$

$$F[t] = \begin{cases} 10, & se\ q[t] = 1 \\ -10, & se\ q[t] = 0 \end{cases}$$

Dunque, supponendo, ad esempio, che il valore di p al tempo t sia 0.6 (essendo soggetto alla funzione sigmoide abbiamo un range di valori limitato) avremo una probabilità del 60% che la spinta del carrello venga effettuata verso destra e una probabilità del 40% che la spinta avvenga verso sinistra.
Inizialmente, prima che venga appresa la funzione valutazione, le due azioni sono equiprobabili.
La struttura della rete d'azione è analoga a quella di valutazione, in questo caso però le matrici dei pesi vengono indicate con D, E, F.

Le equazioni che regolano i pesi sono date da:

$$e_i[t+1] = e_i[t] + \rho * \hat{r}[t+1] * (q[t] - p[t]) * x_i[t]$$

$$f_i[t+1] = f_i[t] + \rho * \hat{r}[t+1] * (q[t] - p[t]) * z_i[t]$$

$$d_{ij}[t+1] = d_{ij}[t] + \rho_h * \hat{r}[t+1] * z_i[t] * (1 - z_i[t]) * sgn(f_i[t]) * (q[t] - p[t]) * x_j[t]$$

$\rho, \rho_h > 0$.

Capitolo 3

Risultati

Ogni esperimento consiste in un numero di runs, che differiscono tra loro per i valori pseudocasuali. Ogni run consiste in un numero di "prove", le quali iniziano con il sistema carrello-asta settato casualmente e terminano con la comparsa del segnale di fallimento.
All'inizio di ogni run le matrici dei pesi sono inizializzate a valori casuali compresi tra -0.1 e 0.1.
L'inizializzazione a valori pseudocasuali viene fatta per far sì che le due azioni, spinta a destra e spinta a sinistra, risultino, inizialmente, equiprobabili.
Dopo ogni fallimento, viene reimpostato a valori random lo stato del sistema ma non le matrici dei pesi. Ad esempio, se al tempo $t = 10$ si verifica un fallimento, dovuto o al carrello che urta i limiti della pista o al pendolo che supera il valore assegnato dell'angolo, le variabili di stato $h, \dot{h}, \theta, \dot{\theta}$, saranno casuali per il passo $t = 11$.
Per i parametri, presenti nelle equazioni di aggiornamento dei pesi, sono stati utilizzati i seguenti valori:

$\rho = 1.0$ (learning rate per l'unità di output della rete azione)

$\rho_h = 0.2$ (learning rate per le unità nascoste della rete azione)

$\beta = 0.2$ (learning rate per l'unità di output della rete valutazione)

$\beta_h = 0.05$ (learning rate per le unità nascoste della rete valutazione)

$\Upsilon = 0.9$ (discount rate)

Gli esperimenti sono stati effettuati con quattro simulazioni, di durata differente, e per ogni simulazione sono stati eseguiti 10 runs.
L'obiettivo di questi esperimenti è che il sistema impari a generare azioni, F[t], che massimizzano il numero di time step tra un fallimento e l'altro.
Le quattro simulazioni sono state eseguite con valori di t pari a 9milioni, 26milioni, 43milioni e 60milioni. Di seguito sono riportiti i risultati, di uno solo dei dieci run, relativi alle simulazioni di più lunga e breve durata:

- Per t = 9milioni
  Nei primi step, da t = 0 a 4500000, la media dei passi senza fare fallimento è di 183, mentre negli ultimi time step la media sale a 2139, con un numero totale di fallimenti pari a 26340.

- Per t = 60milioni
  Nei primi 30milioni di step, la media dei passi senza fare fallimento è di 285, per poi salire nella seconda metà dell'intervallo di tempo a 1198, con un totale di fallimenti pari a 126097.

Analizzando i 10 runs e facendo un confronto tra i diversi time step, il numero di passi che si ha senza fare fallimento cresce in maniera lineare:

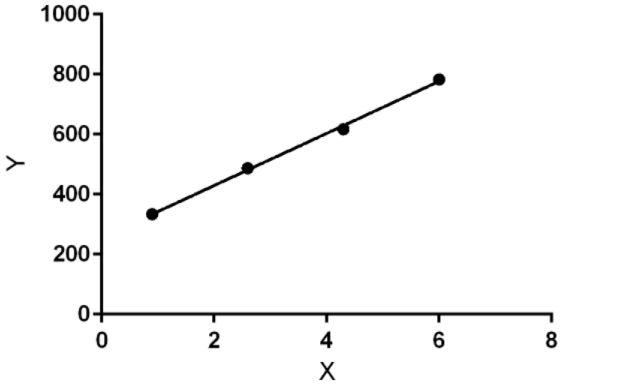

Figura 4 regressione lineare

La regressione lineare (figura 4) rappresenta la relazione tra la durata della simulazione e l'intervallo di tempo tra un fallimento e l'altro.
Sull'asse X è rappresentato il numero di time step, ovvero la durata della simulazione, diviso per 10milioni, cioè, ad esempio, il valore 2 corrisponde a una durata della simulazione pari a 20milioni di time step.
Sull'asse Y, invece, viene rappresentata la media dei passi senza fallimento dei 10 runs relativi ad una simulazione. Pertanto, analizzando il grafico, si ha una media di circa 340 passi senza fallimento nei 10 runs relativi alla simulazione di durata 9milioni di time step (0.9 nell'asse X), una media di circa 460 passi per la simulazione di durata 26milioni di time step (2,6), una media di circa 600 passi per la simulazione di durata 43milioni di time step (4,3) e una media di circa 780 passi per la simulazione di durata 60milioni di time step (6 nell'asse x).

La formula per calcolare la media dei passi in funzione della durata della simulazione è data da:

$$Y = 255.5 + 86.88*X$$

Con la rete neurale multistrato si può, dunque, ottenere una buona performance, dopo che le caratteristiche utili (features) sono state formate. Questo spiega il lento miglioramento nel comportamento del sistema nei primi time step, ovvero la presenza di numerosi fallimenti all'inizio di ogni simulazione. Una volta che le caratteristiche utili sono state formate, il comportamento migliora significativamente.
Per comprendere meglio il comportamento del sistema, e quindi cosa apprende la rete, in figura, viene mostrato un plot della simulazione:

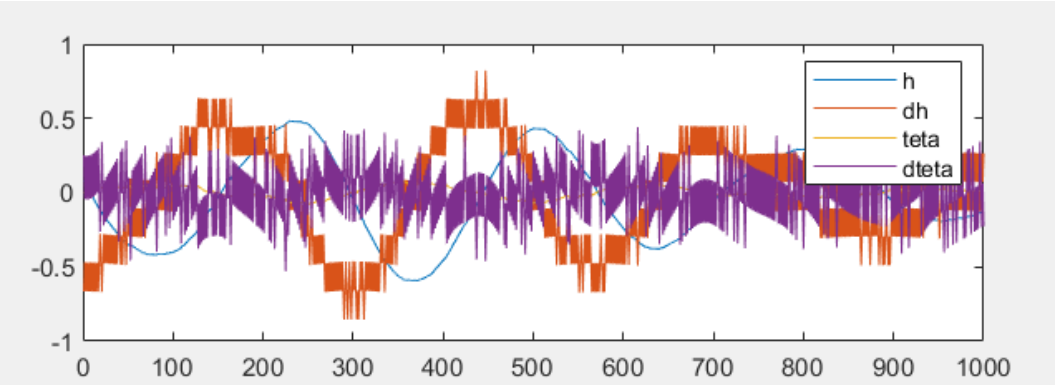

Figura 5 plot di simulazione

sulle ascisse sono rappresentati gli ultimi mille time step della simulazione, mentre sulle ordinate vengono rappresentati i valori della velocità e posizione del carrello e velocità angolare e angolo dell'asta.
La funzione in blu rappresenta lo spostamento del carrello, quella in rosso la sua velocità, mentre le funzioni in giallo e viola indicano rispettivamente l'angolo e la velocità angolare dell'asta.
Se l'angolo θ assume valori positivi ,con velocità angolare positiva, (nell'intervallo 0-150), la posizione del carrello si porta man mano da valori negativi a valori positivi con un'azione di spinta a destra e velocità del carrello positiva (dh).
Quando θ assume valori negativi (150-300) la velocità del carrello diminuisce e l'azione di spinta a sinistra porta il carrello verso valori negativi di h e θ si porta a valori positivi.
La posizione dell'angolo, così come quella del carrello, oscilla in maniera

sinusoidale all'interno del dell'intervallo -0.5 e 0.5, ottenendo in questo modo l'equilibrio del sistema pendolo.

Oltre a regolare la posizione dell'asta, il sistema deve essere in grado di mantenere il carrello entro un certo limite della pista, come espresso dal segnale di fallimento, anche in questo caso lo si ottiene applicando correttamente l'azione al sistema. Per analizzare meglio questo tipo di comportamento, facciamo rifermento ad un plot diverso, relativo ad un'altra simulazione, in cui questo risulta più evidente:

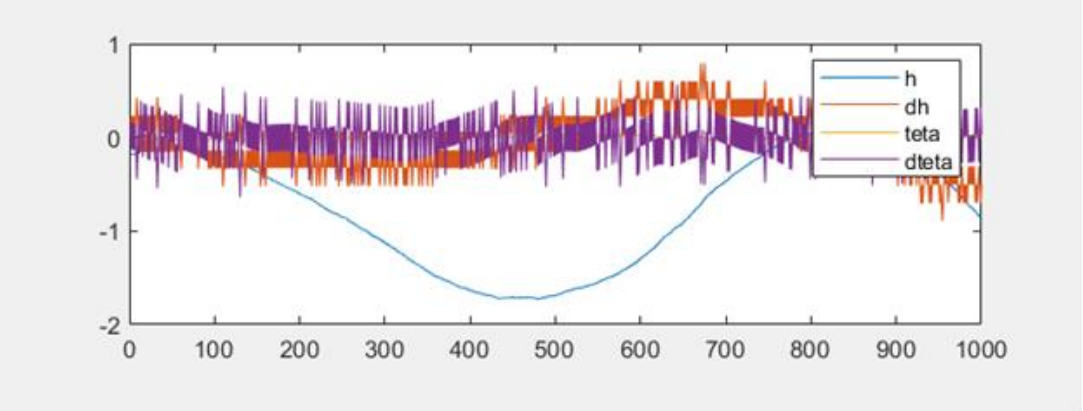

Figura 6 plot di simulazione

Seguendo la linea in blu, che rappresenta la posizione del carrello, vediamo come all'avvicinarsi del limite della pista (-2.4 sull'asse delle ordinate), il sistema riceve una spinta verso destra, e questo lo si può vedere dall'aumento della velocità del carrello, rappresentata dalla funzione in rosso (dh), in modo da riportarlo lontano dai limiti e quindi dal fallimento.
In questa rappresentazione possiamo inoltre vedere come i valori dell'angolo e della velocità angolare oscillano tra valori intorno allo zero.

Il sistema, dunque, riesce ad apprendere, dopo molta esperienza, la giusta azione da applicare, per far sì che il pendolo sia bilanciato. Tuttavia, anche dopo l'apprendimento la probabilità di selezionare l'azione sbagliata, che porta ad un conseguente fallimento, esiste.

3.1 Algoritmo Adaptive Heuristic Critic (AHC)

Di seguito vengono riportati i passi dell'algoritmo applicato all'esperimento [8]:

- Fase di inizializzazione

    1) Settare i pesi della rete azione e valutazione a valori compresi tra -0.1 e 0.1

- Fase di iterazione

    2) Ripetere un numero di volte pari al numero di time step (t)

        a) La rete di valutazione calcola la valutazione v_t dello stato corrente

        b) La rete d'azione calcola la probabilità di spinta a destra o a sinistra.

        c) Si applica l'azione al sistema carrello-asta generando un nuovo stato.

        d) La rete di valutazione calcola la valutazione v_{t+1} per il nuovo stato.

        e) Si calcola la differenza temporale tra la valutazione del nuovo stato e la valutazione dello stato precedente

        f) Si aggiornano i pesi della rete di valutazione

        g) Si aggiornano i pesi della rete d'azione

        h) Se occorre il fallimento, resettare a random lo stato e il contatore si passi a zero

    3) Riportare i risultati di ogni "prova".

Capitolo 4

Discussione e Conclusioni

4.1 Discussione

In molte situazioni del mondo reale, un obiettivo di controllo non può essere espresso come una funzione definita su tutti gli stati, ma solo per un sottoinsieme relativamente piccolo di stati. Per alcuni compiti di controllo, un obiettivo così definito è forse desiderabile. Richiedere a un controller di portare il valore di una variabile di stato il più vicino possibile a zero quando il vero obiettivo è solo quello di evitare valori estremi potrebbe interferire con il controllo di altre variabili di stato.
Le reti neurali che imparano tramite apprendimento per rinforzo ed i metodi alle differenze temporali, si occupano di questo problema simultaneamente imparando una funzione probabilistica di azione e una funzione di valutazione. Sono stati applicati altri tipi di approccio al pendolo inverso come, ad esempio, apprendimento supervisionato fornendo esplicitamente azioni corrette.
Le difficoltà di alcune attività di controllo di basso livello possono portare all'applicazione accurata di metodi di apprendimento mediante reti neurali. La capacità delle reti neurali di gestire variabili di input e output multipli e funzioni non lineari, giustificano ulteriori indagini sui metodi di apprendimento delle reti neurali nei domini di controllo.
Gli esperimenti nel controllo dell'apprendimento con le reti neurali possono far luce su come affrontare le incertezze del mondo reale e le complessità nel controllo. Tuttavia, molte questioni rimangono irrisolte, come ad esempio le prestazioni dei metodi di apprendimento possano arrivare fino a compiti più grandi e complessi di quelli attualmente studiati. [3]

4.2 Conclusioni

Precedentemente fu svolto un simile esperimento utilizzando un solo livello della rete, i cui risultati furono il bilanciamento del pendolo ma non il mantenimento del carrello all'interno della pista. L'aggiunta di unità nascoste, ovvero di un secondo livello, alla rete ha portato a risultati migliori, in quanto in questo caso, il sistema di apprendimento impara sia a bilanciare l'asta che a mantenere il carrello entro i limiti della pista.
La strategia iniziale di selezionare in maniera casuale l'azione è stata trasformata in una scelta quasi deterministica di azione migliore per ogni stato, grazie anche all'apprendimento di nuove funzionalità. Un'azione è stata scelta rapidamente calcolando l'output della rete d'azione e la regola di controllo viene applicata sulle variabili di stato che vengono generate dalla forza applicata e non sulla forza stessa.
Questo approccio si è dunque rivelato adatto a perfezionare il compito di controllo e l'identificatore neurale è stato in grado di rappresentare la dinamica del pendolo inverso.

Bibliografia

[1] Young-Moon Park, Myeon-Song Choi and K. Y. Lee, "An optimal tracking neuro-controller for nonlinear dynamic systems," in *IEEE Transactions on Neural Networks*, vol. 7, no. 5, pp. 1099-1110, Sept. 1996, doi: 10.1109/72.536307.

[2] Cong Wang and D. J. Hill, "Learning from neural control," in *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 130-146, Jan. 2006, doi: 10.1109/TNN.2005.860843.

[3] C.W. Anderson. "Learning to control an inverted pendulum using neural networks" *IEEE Control Systems Magazine*, 9 (1989), pp. 31-37

[4] Samuel,A.L. Some studies in machine learning using the game of checkers. IBM *journal on research and development*, (1959) ,3,210-229.

[5] Sutton,R.S, "Learning to predict by the methods of temporal differences (technical report TR87-509.1),Waltham,MA; GTE laboratories incorporated. (1987)

[6] C. W. Anderson." Strategy Learning with Multilayer Connectionist Representations", Colorado State University.

[7] C.W. Anderson, Learning and problem solving with multilayer connectionist systems, PhD thesis, Computer and Information Science, University of Massachusetts, 1986

[8] Whitley, D., Dominic, S., Das, R. *et al.* Genetic reinforcement learning for neurocontrol problems. *Mach Learn* 13, 259–284 (1993). https://doi.org/10.1007/BF00993045