Predictive Coding per ottimizzazione di reti neurali ricorrenti in System Identification

Negli ultimi anni, l'applicazione del machine learning, in particolare delle reti neurali e del deep learning, ha significativamente influenzato il campo della System Identification[1]. In tale contesto, si fa frequentemente ricorso a reti neurali ricorrenti (RNN) standard, quali le Long Short-Term Memory (LSTM)[2] o le Gated Recurrent Unit (GRU)[3].

L'addestramento di queste architetture neurali avviene tipicamente attraverso la minimizzazione di una funzione di costo, come l'errore quadratico medio (MSE). Questo processo di ottimizzazione impiega comunemente algoritmi basati sul gradiente, tra cui il gradient descent e le sue varianti stocastiche. Un aspetto cruciale di tali metodi è il calcolo del gradiente della funzione di costo rispetto ai pesi sinaptici della rete, che viene effettuato mediante l'algoritmo della **backpropagation**. Quest'ultimo si fonda sull'applicazione della chain rule nel contesto della differenziazione automatica.

Nonostante la backpropagation rappresenti un meccanismo ampiamente consolidato ed efficace per l'addestramento delle reti neurali artificiali, è importante sottolineare come il suo principio di funzionamento non trovi una diretta corrispondenza nei processi di apprendimento osservati nelle reti neurali biologiche, a differenza invece della topologia di alcune reti artificiali che si ispirano a quelle biologiche.

Nel campo delle neuroscienze computazionali[4], il **predictive coding**[5] emerge come un promettente paradigma di apprendimento bio-ispirato. A differenza della backpropagation, che usa la retropropagazione di un errore globale per aggiornare i pesi, il predictive coding apprende minimizzando gli errori di predizione tra l'input atteso e quello ricevuto per ogni singolo neurone individualmente. Questo processo è spesso visto come la minimizzazione di una funzione di energia. Un aspetto distintivo è la **località** dell'aggiornamento dei pesi: le modifiche sinaptiche dipendono solo da informazioni locali (attività del neurone e errore ricevuto dal post-sinaptico), rendendolo più plausibile biologicamente e potenzialmente vantaggioso per l'apprendimento online e la gestione di scenari complessi, contribuendo anche a mitigare il catastrophic forgetting.

---

[1] Ljung et al., 2020, "Deep Learning and System Identification"
[2] Hochreiter and Schmidhuber, 1997, "Long Short-Term Memory"
[3] Cho et al., 2014, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation"
[4] Richards et al.,2019, "A deep learning framework for neuroscience"
[5] Song et al., 2024, "Inferring neural activity before plasticity as a foundation for learning beyond backpropagation"

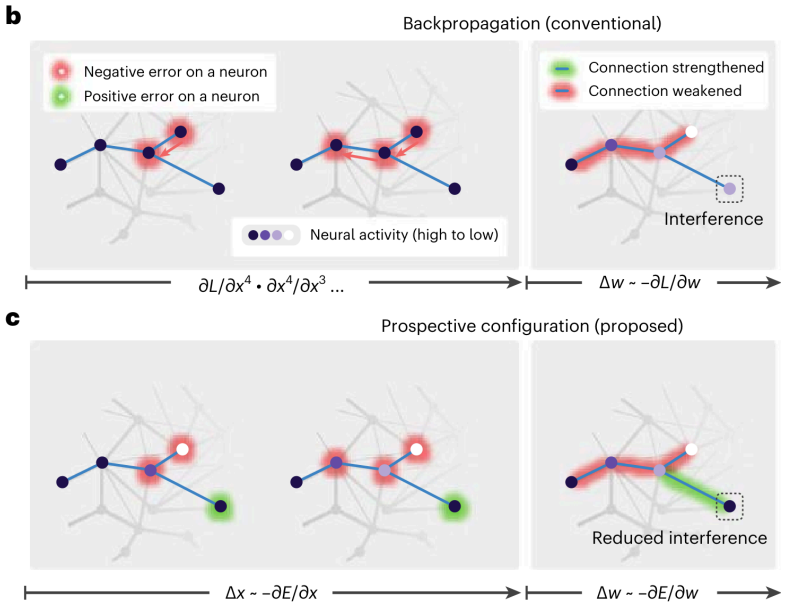

Figura da Song et al., 2024, "Inferring neural activity before plasticity as a foundation for learning beyond backpropagation". Confronto schematico tra l'apprendimento tramite Backpropagation convenzionale (b) e l'approccio basato sulla Configurazione Prospettica/Predictive Coding (c). Mentre nel primo caso il gradiente per l'aggiornamento dei pesi deriva dalla minimizzazione dell'errore globale della funzione di costo, nel secondo l'aggiornamento (basato sulla minimizzazione dell'energia) è guidato dall'errore locale a livello di singolo neurone.

Obiettivo della tesi

Questa tesi esplora il Predictive Coding come alternativa alla backpropagation per ottimizzare reti neurali ricorrenti in System Identification. Gli obiettivi chiave sono:

1. **Implementare il Predictive Coding**: sviluppare un modello RNN basato sul Predictive Coding.
2. **Valutare le prestazioni**: confrontare le performance del modello proposto con la backpropagation su compiti di System Identification (accuratezza, velocità di convergenza, robustezza in presenza di rumore, …).
3. **Analisi Comparativa**: analisi dei meccanismi di apprendimento, esplorando come i processi di inferenza locale nel Predictive Coding influenzino l'adattabilità e la capacità della rete di generalizzare.

La tesi può essere svolta singolarmente o in coppia.

L'implementazione sarà principalmente in MATLAB.

Ci si concentrerà su architetture LSTM e GRU, con possibilità di utilizzare feedforward o RNN classiche per confronti/semplicità.

I modelli verranno testati su dataset di benchmark pubblici (i.e. NISB), eventualmente integrati con simulazioni.