Python per il Calcolo Scientifico

Angelo Cardellicchio

Cenni al Deep Learning

- Perché *deep*?
- Deep Learning e Machine Learning
- I layer di una rete neurale
- La funzione di costo
- Un esempio di applicazione: la *object detection*

Perchè *deep*?

- La differenza fondamentale tra reti *shallow* e *deep* sta nel numero di *hidden layer*: le prime ne hanno solitamente solo uno, le seconde più di uno
- Tuttavia, fino a poco più di dieci anni fa, le seconde erano quasi inutilizzate, mentre al momento le prime hanno significato esclusivamente accademico
- Come mai è avvenuto questo cambio di paradigma?

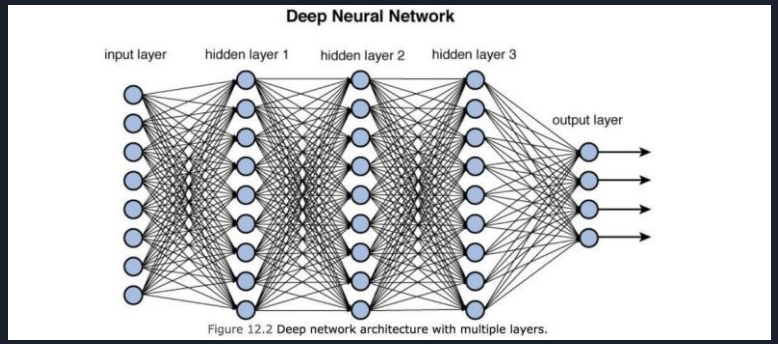
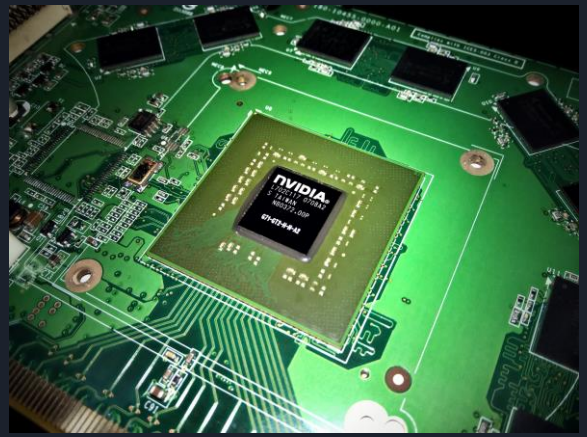


Figure 12.2 Deep network architecture with multiple layers.

Fonte

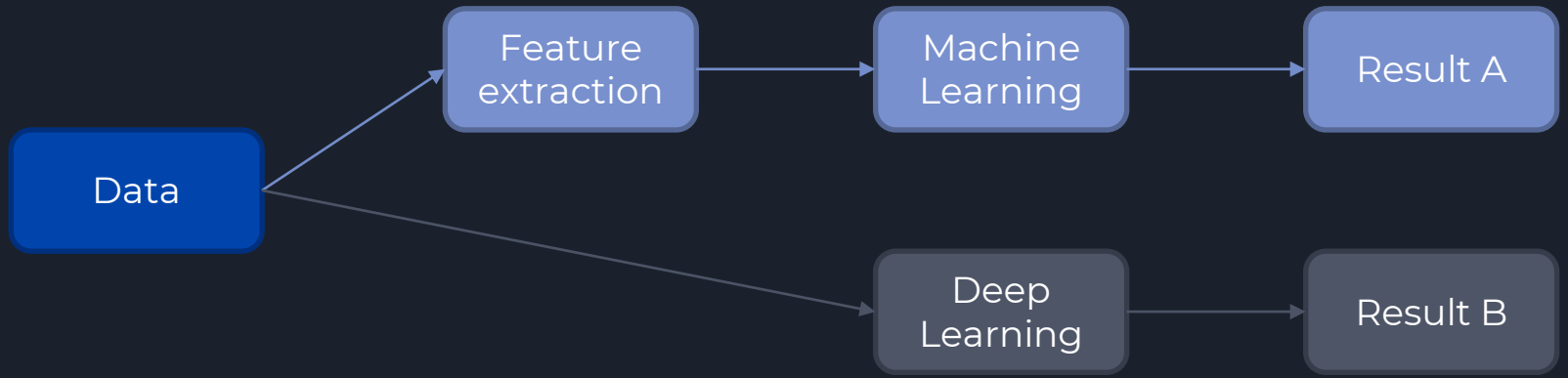
Perchè *deep*?



- La teoria alla base delle reti neurali risale agli anni ‘50
- Tuttavia, sono sempre mancati due fattori fondamentali, ovvero la capacità computazionale ed i dati
- Questo è però cambiato negli ultimi anni, grazie all’avvento delle ***General Purpose GPU*** ed alla facile reperibilità di grosse quantità di dati (***Big Data***)

Deep Learning e Machine Learning

- Molto spesso, si parla in maniera quasi “intercambiabile” di machine learning e deep learning
- In realtà, esistono una grossa differenza tra i due approcci: il deep learning infatti automatizza la fase di ingegnerizzazione delle feature



I layer di una rete neurale

- Esistono diversi tipi di layer, adatti a diversi scopi
- Ad esempio, i ***layer convoluzionali*** sono adatti all'estrazione ed identificazione di feature all'interno delle immagini
- Invece, i ***layer completamente connessi*** sono usati in ambiti più eterogenei, ed in generale quando abbiamo feature di tipo vettoriale
- Altri tipi di layer che è comunemente possibile individuare sono quelli di ***dropout*** e ***pooling***

La funzione di costo

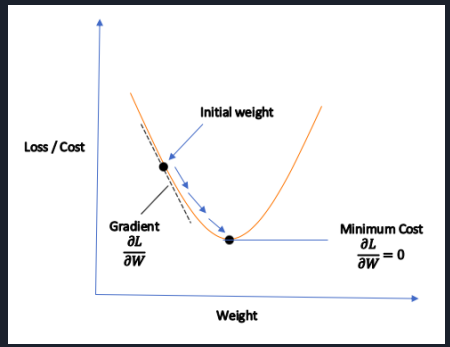
- La funzione di costo (***loss function***) ha il compito di valutare l'efficacia dell'algoritmo.
- La si può pensare come una funzione che ha il compito di minimizzare (o massimizzare) una metrica rappresentativa dell'efficacia dell'algoritmo.
- Molto usata è la ***cross-entropy***.

Intermezzo sulla cross-entropy

- ***Informazione***: numero di bit necessari a codificare un evento
  - Un evento a bassa probabilità contiene più informazione di uno ad alta probabilità
- ***Entropia***: informazione richiesta per codificare un evento casuale estratto da una certa distribuzione
  - L'entropia necessaria alla codifica di un evento estratto da una distribuzione perfettamente bilanciata è maggiore rispetto a quella necessaria a codificare un evento estratto da una distribuzione sbilanciata
- ***Cross-entropia***: informazione necessaria a codificare un evento estratto da una distribuzione *p* usando un modello relativo ad una distribuzione *q*
  - Minimizzarla ci permette di ottenere un modello che, seppur relativo a *q*, riesca a caratterizzare la distribuzione *p*.

La funzione di ottimizzazione

- Le reti neurali utilizzano funzioni di ottimizzazione per minimizzare (o massimizzare) il valore assunto dalla funzione di costo
- Una funzione di ottimizzazione ‘studia’ quali parametri della rete modificare per ottimizzare il risultato
- Molto usato in tal senso è il *gradiente* della funzione di costo
- Quando la funzione è molto complessa, si usano delle approssimazioni, selezionando in maniera causale sottoinsiemi di derivate parziali da ottimizzare (***discesa stocastica del gradiente***)



Fonte

Per approfondire

- Overview sugli ottimizzatori
- Introduzione alla cross-entropy
- Layer di una CNN