

# Lezione 1 01/10/2024

## Deep learning

Con **machine learning** avevamo le reti neurali, che erano reti molto parallele con relativamente semplici neuroni non lineari che imparano aggiustando la forza dei loro collegamenti.

Però come è possibile per una rete di questo tipo, imparare le rappresentazioni complicate che sono necessarie per task complesse come il riconoscimento di oggetti?

Il **deep learning** cerca una risposta a questa domanda, usando multipli layer di "**activity vectors**" come rappresentazioni, e imparando la forza delle connessioni, seguendo il gradiente stocastico (procedura di ottimizzazione) della funzione obiettivo per trovare il peso ottimale dei collegamenti.

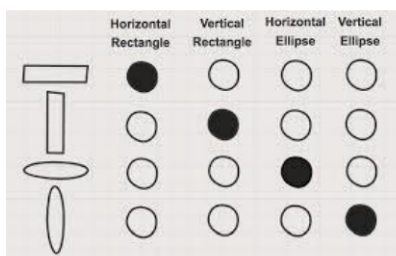
L'ingrediente chiave è quindi la **profondità (depth)**. Avremo quindi tanti layers.

La differenza principale tra machine learning e deep learning è il **representation learning**. Nel deep learning bisogna trovare queste rappresentazioni, non c'è il features engineering, la rappresentazione è imparata. Servono quindi molto più dati.

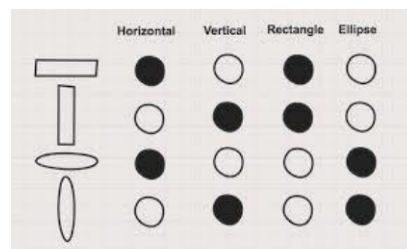
L'obiettivo (nel processo di learning) è di identificare il fattore di variazioni (**factor of variations**) che spiega i dati osservati. Questi fattori potrebbero non essere direttamente osservabili, possono essere inferiti durante il processo. Potrebbero essere dei concetti astratti che aiutano a trovare senso nella variazione dei dati.

Per esempio se vedo una foto di una macchina rossa, il rosso della macchina potrebbe essere più scuro durante la notte. Facendo vedere molte foto durante il giorno e durante la notte, si può far capire al modello come la luce ha effetto sulle immagini.

Non c'è una relazione tra neurone e concetto, il **concetto** è la composizione di molti neuroni. La rappresentazione è distribuita.

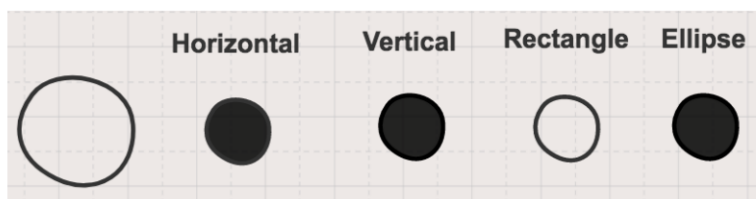


*Sparse or local, non-distributed representation of shapes*



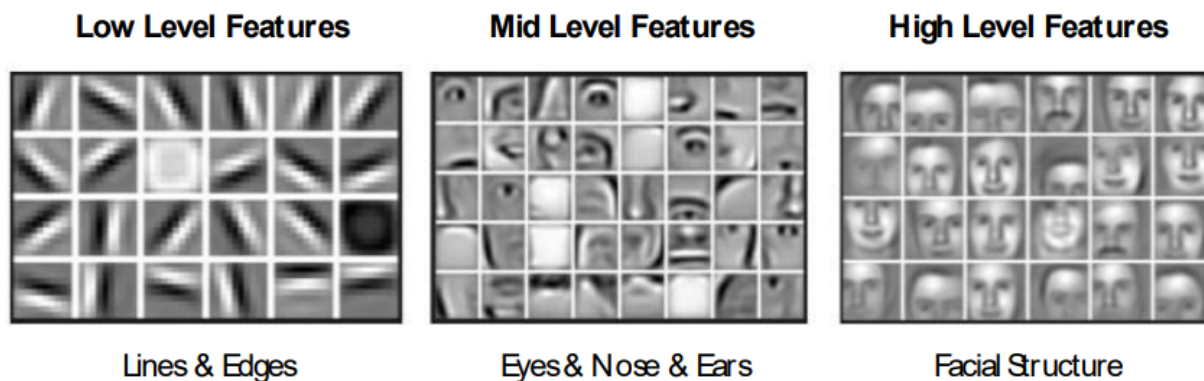
*Distributed representation of shapes*

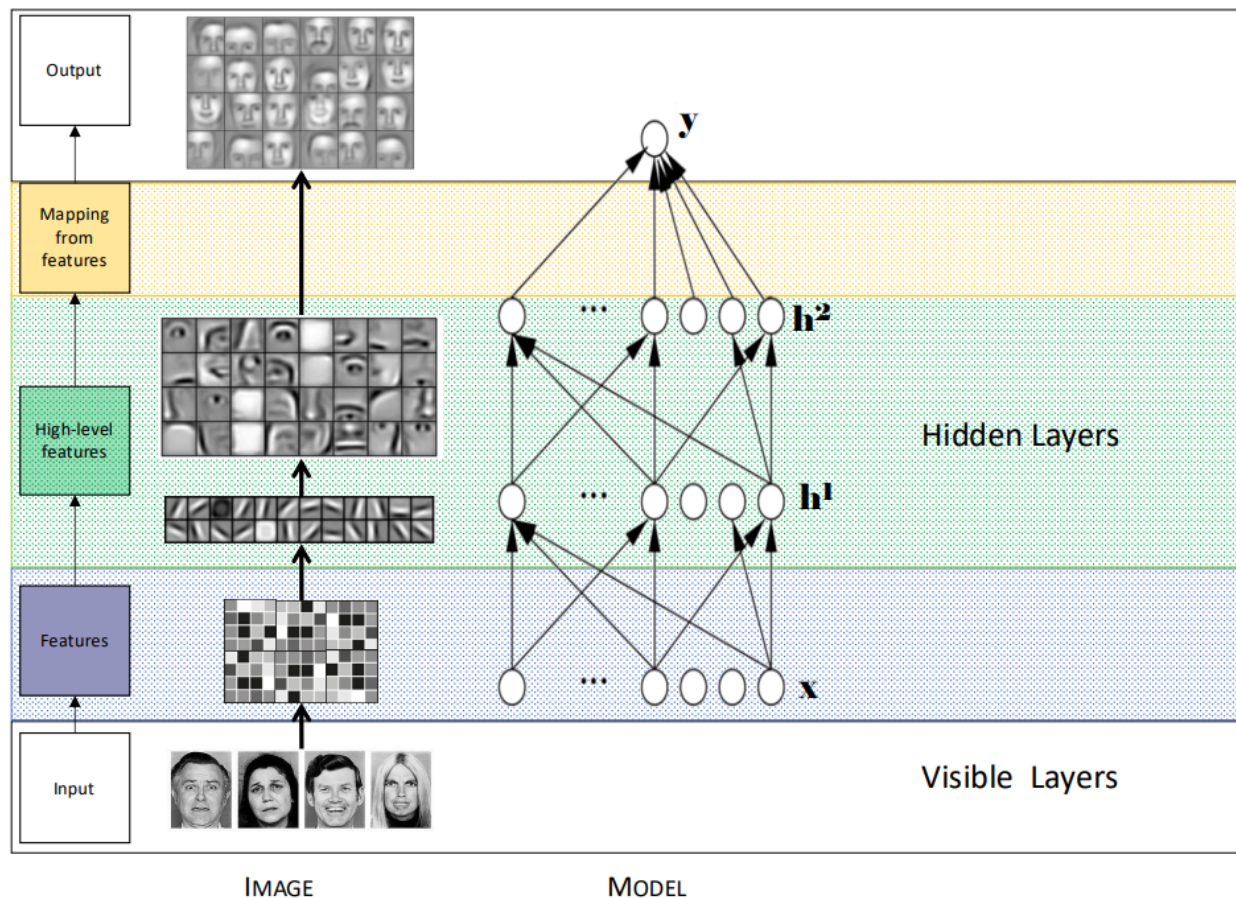
*Images by Garrett Hoffman*



Ho sempre 4 features, il vantaggio è che rappresentato un concetto come composizione di features, posso rappresentare anche un cerchio. Posso quindi rappresentare un altro concetto, con le stesse features.

Nel deep learning si risolve questo problema del representation learning, introducendo rappresentazioni che sono espresse in termini di altre **rappresentazioni più semplici**. Questo permette al computer di comporre concetti complessi, partendo da quelli semplici.





Le **features** sono quindi semplicemente i pixels, e poi man mano vengono imparati concetti sempre più complessi. Ovviamente nel modello non ci sono immagini, sono numeri, vettori.

Nel deep learning è importante pulire bene i dati.

Quanto dovrebbe essere profondo un modello di deep learning?

La **profondità** può essere rappresentata:

- dal numero di layers.
- dal numero di istruzioni sequenziali che devono essere eseguite. Questo è solitamente il longest path del flowchart.

Di solito si usa il primo metodo.

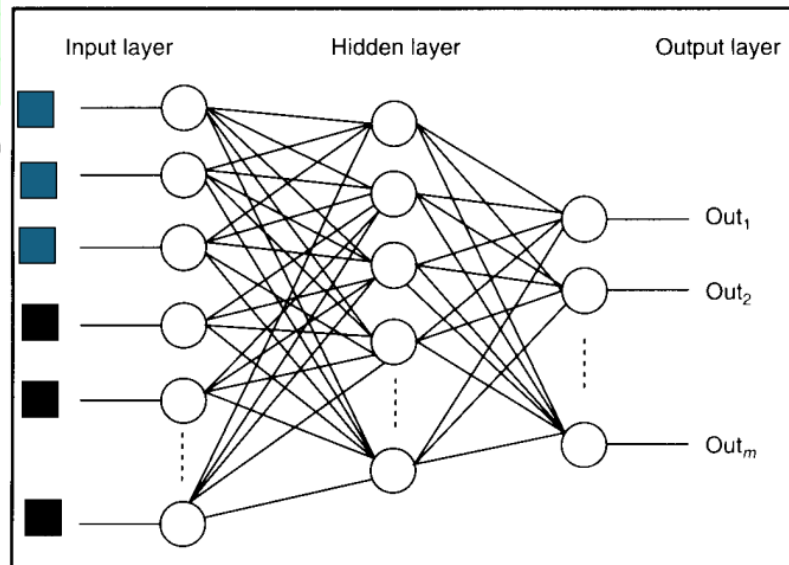
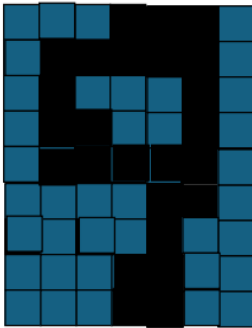
In ogni caso non c'è un numero corretto di layer, e non c'è un consenso su quale numero di layer fa diventare il modello "deep".

## Feature Detectors

# Feature detectors

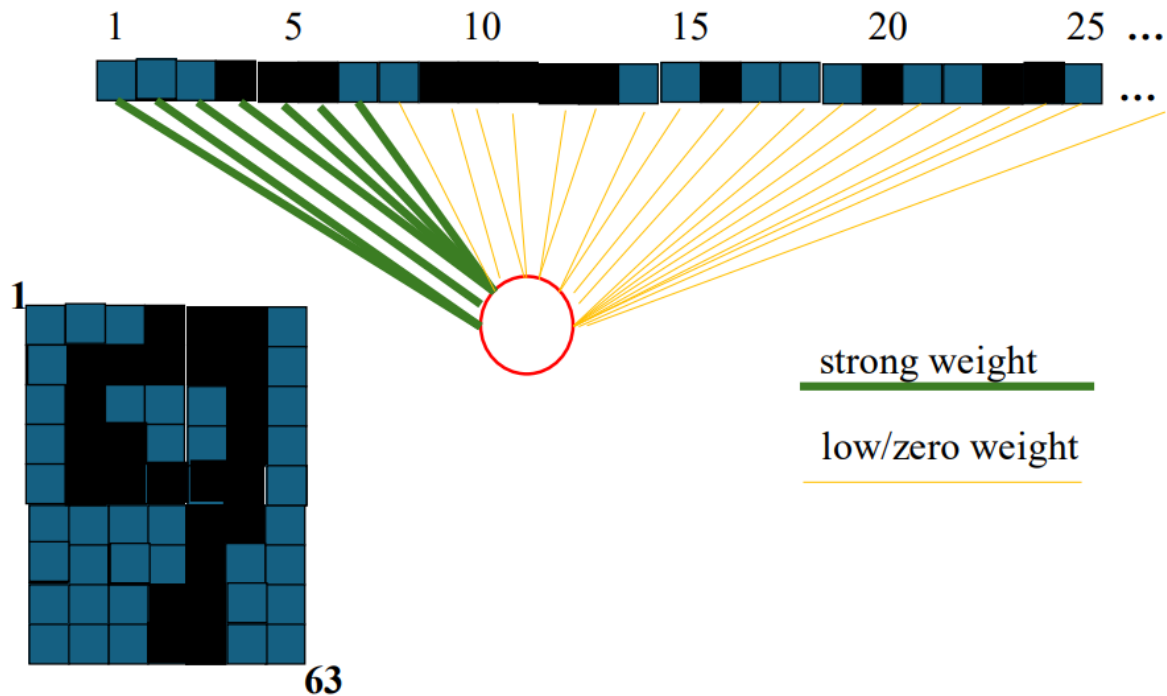


Figure 1.2: Examples of handwritten digits from postal envelopes.

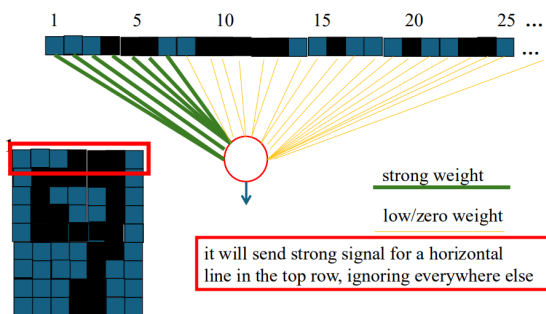


Nel caso di un testo scritto a mano, nei layer più bassi i neuroni si specializzano per capire per esempio le righe orizzontali e verticali. A livello superiore si compone la feature che riconosce il numero.

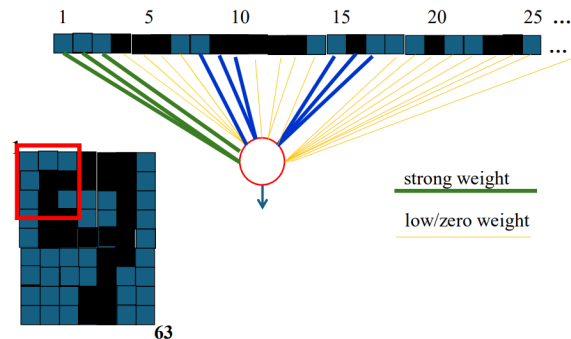
# Hidden layer units become *self-organised feature detectors*

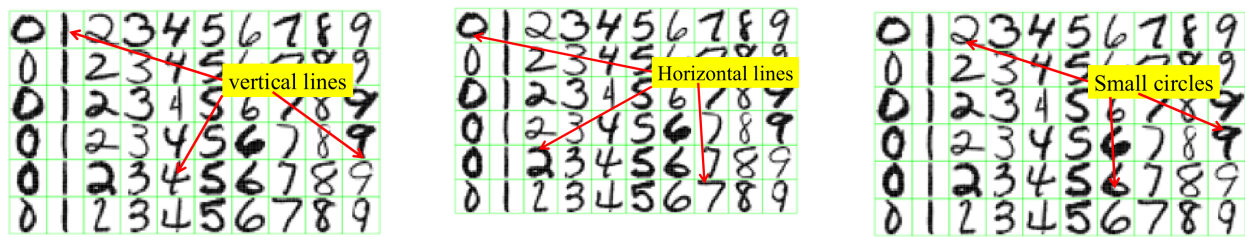


What does this unit detect?

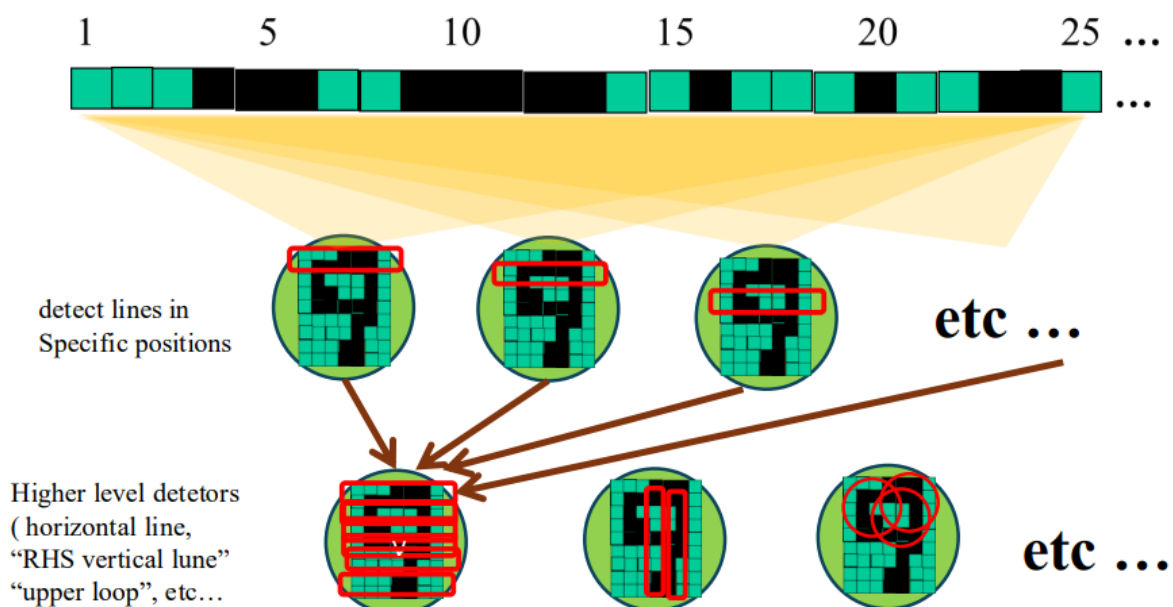


What does this unit detect?



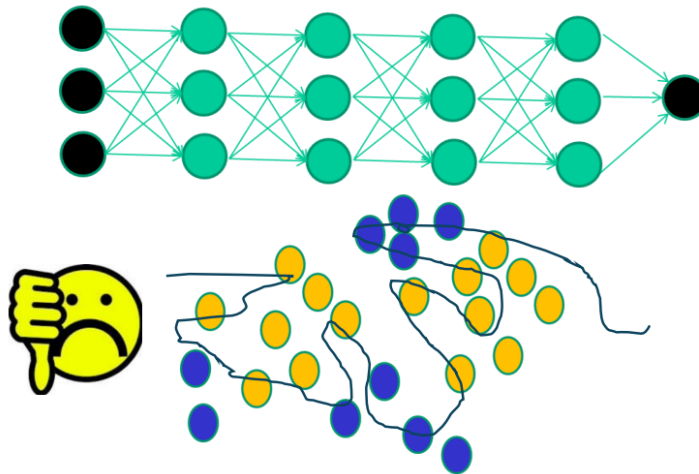


Con un dataset di questo tipo, le features che vengono trovate dalla rete neurale sono queste righe verticali ed orizzontali, cerchi.



E per quanto riguarda la posizione invece? Queste informazioni di più alto livello verranno imparate da successivi layer.

Questo è proprio il motivo per cui abbiamo bisogno di **layers multipli**.



In passato reti neurali con layer multipli non funzionavano.

Il nuovo metodo di training che funziona per le reti neurali multilivello allena prima il primo layer, poi il secondo, poi il terzo... e infine il nodo di output.

Quindi ciascun layer è allenato come se fosse un **auto-encoder** (quel tipo di rete neurale che ha come output lo stesso input, avendo una strozzatura al centro che rappresenta l'encoding), forzando la rete ad imparare features che descrivono ciò che arriva dal layer precedente.

Se un layer ha quindi meno nodi di quello precedente, il layer viene forzato a diventare un buon **feature detector**.

Il nodo di output viene allenato per predire la classe, in base all'output del layer precedente.