



POLITECNICO DI BARI

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELL'INFORMAZIONE - DEI
Corso di Laurea Triennale in Ingegneria Informatica e dell'Automazione (D.M. 270/04)

TESI DI LAUREA IN CALCOLO NUMERICO

ANALISI DI IMMAGINI CON RETI NEURALI CONVOLUZIONALI PER LA CLASSIFICAZIONE DEI CETACEI NEL GOLFO DI TARANTO

Relatore:

Chiar.mo Prof. Ing. Tiziano POLITI

Correlatore:

Dott. Ing. Vito RENÒ

Laureando:

Tommaso MONOPOLI

Matricola 568581

Sommario

Il sistema terrestre è sempre stato soggetto alle conseguenze delle attività umane, e la biodiversità degli ecosistemi acquatici e marini è fortemente a rischio. Diversi studi cercano di capire in che modo la perdita della biodiversità possa alterare l'integrità e il funzionamento di tali ecosistemi. Una risposta a questa domanda può essere ricercata negli studi effettuati sulla distribuzione e sullo stato di conservazione dei cetacei, oggetto di numerose ricerche negli ultimi anni.

Un'attività mirata alla raccolta di informazioni rilevanti allo studio dei cetacei è la *foto-identificazione degli individui* di una specie, che prevede il riconoscimento - automatico o manuale - di uno stesso individuo in diverse immagini collezionate nel tempo, mediante l'analisi di particolari segni distintivi (*feature*) presenti nell'immagine.

Questa attività può essere effettuata manualmente, ma con un grande costo in termini di tempo per i ricercatori, che spesso hanno a disposizione diverse migliaia o milioni di fotografie, scattate nel corso di anni. L'evidente difficoltà nell'approccio manuale alla foto-identificazione dei cetacei (tutt'oggi ancora ampiamente operata) suggerisce l'applicazione di metodologie di *Computer Vision* per automatizzare tale attività. L'obiettivo del presente lavoro di tesi è la creazione di classificatori binari che ricevano in input un dataset di immagini bidimensionali collezionate nei pressi delle isole Azzorre (Oceano Atlantico settentrionale) e sappiano suddividere lo stesso dataset in due classi di immagini, a seconda che in ciascuna immagine sia rilevata o meno una *feature* utile ad una successiva foto-identificazione. Nel caso dei cetacei, il criterio di classificazione è la presenza nell'immagine della pinna dorsale dell'individuo.

Le metodologie impiegate sono quelle del *machine learning*; in particolare, si è scelto di utilizzare la tecnica del *transfer learning* per il riuso e il ri-adattamento di modelli pre-addestrati, usati per risolvere task di classificazione diversi da quello in esame. Gli esperimenti condotti su dati reali acquisiti in mare dimostrano l'utilità di tali tecniche di Computer Vision nel campo della foto-identificazione dei cetacei.

Indice

Introduzione	1
1 Teoria	2
1.1 Reti neurali	2
1.2 Immagini digitali	2
1.2.1 Supervised Learning	3
1.3 Classificatore lineare	5
Conclusioni	9
Bibliografia	9

Introduzione

La foto-identificazione è una tecnica largamente impiegata per l'identificazione dei singoli individui a partire da una o più immagini. Il principale vantaggio di questa tecnica è la sua non invasività che la rende particolarmente utile per studiare sia la dinamicità che i movimenti di ogni specie. Tale metodologia risulta essere uno strumento affidabile quando viene applicato nella comprensione dei comportamenti dei cetacei (migrazioni e spostamenti). Tra i delfini, vi sono due specie più adatte a tali studi. Il primo delfino riguarda la specie “*Tursiops truncatus*” (tursiope) o delfino dal naso a bottiglia, mentre la seconda specie riguarda la specie “*Grampus griseus*” (Grampo, o delfino di Risso), avente numerose cicatrici su tutto il corpo, entrambi appartenenti alla famiglia dei Delfinidi.[1]

Capitolo 1

Teoria

Come già anticipato, l'approccio più efficace alla risoluzione del problema della classificazione delle immagini consiste nell'impiego delle reti neurali convoluzionali (CNN, *convolutional neural networks*). In questo capitolo saranno introdotti progressivamente i presupposti teorici matematici e informatici su cui si fondano le reti neurali, partendo dalle definizioni preliminari fino a costruire il modello generale di una CNN.

1.1 Reti neurali

TODO: parlare del neurone e dell'idea di replicarlo nella funzione (dopo aver parlato dei classificatori lineari, perché bisogna parlare anche

1.2 Immagini digitali

Caratterizziamo intuitivamente il concetto di immagine dal punto di vista informatico.

Un'**immagine digitale** è una rappresentazione binaria di un'immagine (in generale a colori) a due dimensioni¹; essa può essere definita matematicamente come un tensore $\mathcal{I} \in \mathbb{R}^{h \times w \times c}$, dove h e w sono rispettivamente dette **altezza** e **larghezza** dell'immagine, la coppia (w, h) **risoluzione** mentre c è il numero di *canali di colore*². Nello spazio di colore RGB, ampiamente adoperato, i canali di colore sono rosso (R, Red), verde (G, Green) e blu (B, Blue), quindi $c = 3$. In mancanza di diverse indicazioni, ci si riferirà nel seguito allo spazio di colore RGB.

Un **pixel** $p(i, j)$ è definito come la funzione vettoriale

$$p(i, j) = [r(i, j), g(i, j), b(i, j)]$$

essendo $r, g, b : [0, h] \times [0, w] \rightarrow [0, 255]$ le funzioni scalari che associano ad ogni posizione bidimensionale i, j dell'immagine un valore intero di *intensità luminosa* compreso tra 0 e 255, uno per ciascuno dei tre canali RGB. Ogni pixel definisce

¹Ci riferiamo in questa sede solo alle immagini di tipo raster, tipiche ad esempio delle fotografie digitali in formato jpg.

²Spesso si scrive che \mathcal{I} è un'immagine $w \times h \times c$

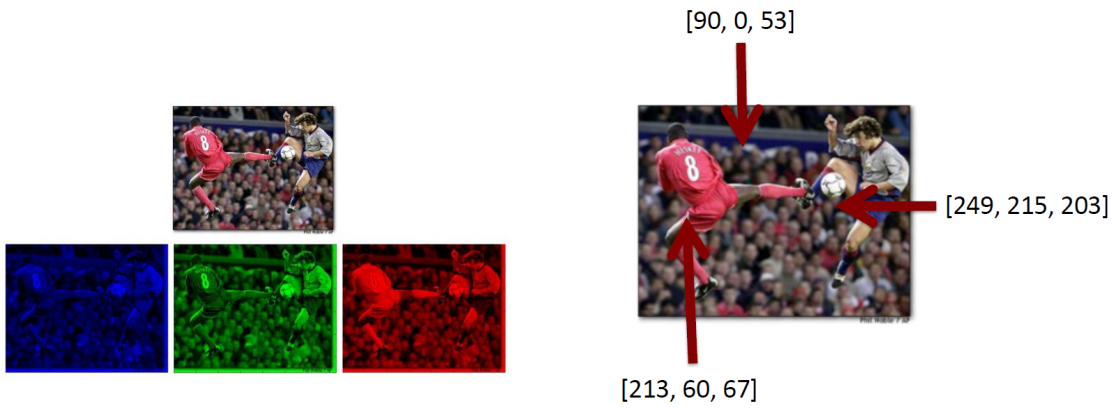


Figura 1.1: Canali RGB di un'immagine

Figura 1.2: Pixel di un'immagine

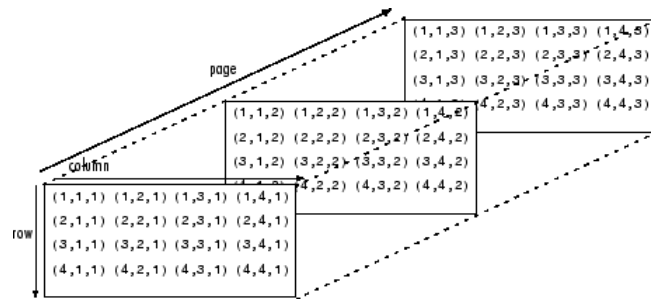


Figura 1.3: Rappresentazione grafica di un tensore tridimensionale; in ogni posizione compaiono gli indici del tensore

univocamente un colore nello spazio RGB, il quale può rappresentare in tutto 256^3 colori diversi, cioè circa 17 milioni.

Si può immaginare il tensore immagine \mathcal{I} come una "pila" di tre matrici, una per ogni canale di colore, come mostrato in figura 1.3.

1.2.1 Supervised Learning

Il paradigma dell'**apprendimento supervisionato** (*supervised learning*) consiste nel creare un algoritmo in grado di apprendere una funzione che mappi un input all'output corretto, sulla base di una serie di esempi ideali, costituiti da coppie di input e dei relativi output attesi, che gli vengono inizialmente forniti per addestrarlo[1]. Un algoritmo di apprendimento supervisionato analizza i dati di addestramento e inferisce una funzione che può essere usata per mappare nuovi input ai corretti output. Ciò richiede all'algoritmo la capacità di trovare una funzione che sappia generalizzare efficacemente dai dati di training, al fine di adattarsi bene a nuovi dati (cioè di poterne mappare correttamente quanti più possibile).

Molti problemi pratici, come ad esempio la regressione e la classificazione, possono essere formulati ricorrendo al concetto di funzione matematica

$$\mathcal{F} : X \rightarrow Y$$

che associa ad ogni elemento nello spazio degli input X (dataset) uno ed un solo elemento dello spazio degli output; ogni elemento di X è correttamente associato ad un solo elemento di Y . Il problema consiste allora nel cercare una funzione \mathcal{F} in grado di ottenere esattamente tale associazione.

È evidente che questo tipo di problemi ben si presta ad essere approcciato con algoritmi di apprendimento supervisionato.

Prima di analizzare in dettaglio il problema di classificazione delle immagini oggetto della presente tesi, è necessario inquadrare il problema partendo da alcune definizioni preliminari.

Un **dataset** X è una generica collezione di N dati

$$X = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$$

Ogni dato $\mathbf{x}^{(i)}$ è chiamato **esempio** (o **data point**). I data point possono essere anche non omogenei tra loro (cioè avere dimensioni differenti). Ciascun esempio si può caratterizzare come un vettore $\mathbf{x}^{(i)} \in \mathbb{R}^D$, in cui ciascun elemento x_i è detto **feature** e rappresenta una caratteristica di un oggetto o un evento misurato. D è il numero di feature in ogni esempio, o **dimensione** dell'esempio. In caso di esempi omogenei (cioè aventi stessa dimensione D) un dataset può essere descritto attraverso una matrice detta **design matrix**, in cui ogni riga corrisponde ad un particolare esempio e ogni colonna corrisponde ad una precisa feature. Un dataset di cardinalità N e in cui ogni esempio ha D feature ha quindi una design matrix di dimensione $N \times D$

In un problema di classificazione delle immagini, orientato all'*object recognition* (riconoscimento di un oggetto in un immagine), sussiste la seguente caratterizzazione:

- X : un insieme di N immagini digitali
- Y : un insieme di K classi predefinite di oggetti che possono essere individuati all'interno di un'immagine (possono essere dei "descrittori" testuali o, equivalentemente, dei numeri interi)

Un elemento di Y è solitamente chiamato **etichetta** o **categoria** (in inglese **label** o **class**); si dice quindi che ogni immagine $\mathbf{x}^{(i)} \in X$ può essere *descritta da un'etichetta* (o *associata ad una categoria*) $\mathbf{y}^{(i)} \in Y$ tramite una funzione di associazione f .³ Nella pratica, TODO f non può essere trovata esattamente. (vd gianvito)

TODO: scrivere ora o in un paragrafo a parte i tipi di dato per gestire le immagini messi a disposizione da matlab.

³Teoricamente una stessa immagine potrebbe essere descritta da più di un'etichetta o addirittura da nessuna, coerentemente col fatto che in essa potrebbero essere presenti più oggetti o nessun oggetto tra quelli previsti in Y . Tuttavia nella presente tesi questa ambiguità non può sussistere: la classificazione riduce qualsiasi immagine ad una di due categorie mutualmente esclusive e di cui almeno una deve essere ammessa, cioè la presenza o meno di una pinna nell'immagine.

1.3 Classificatore lineare

Il classificatore lineare è una tra le più semplici funzioni di classificazione.⁴ Ipotizziamo di avere un insieme di N immagini $\mathbf{x}^{(i)}$ (*data points*), ciascuna con risoluzione fissa $w \times h$ e in formato RGB ($c = 3$), e un insieme di K distinte categorie di oggetti (*labels*). Un **classificatore lineare** è definito dalla funzione

$$f(\mathbf{x}^{(i)}; \mathbf{W}, \mathbf{b}) = \mathbf{W}\mathbf{x}^{(i)} + \mathbf{b} \quad (1.1)$$

In questa espressione stiamo assumendo che $\mathbf{x}^{(i)}$ sia un vettore colonna di dimensione $D = hwc$ ottenuto incolonnando una ad una le righe dell' i -esima immagine di tutti e tre i canali di colore, \mathbf{W} una matrice detta **matrice dei pesi** (*weights matrix*) di dimensione $K \times D$ e \mathbf{b} un vettore colonna detto **vettore dei bias** (*bias vector*) di dimensione K . I pesi e i bias sono parametri della funzione f .

Ogni riga j -esima di \mathbf{W} e il relativo j -esimo valore di \mathbf{b} serve a calcolare la combinazione (lineare a meno del bias) $\mathbf{w}_j \cdot \mathbf{x}^{(i)} + b_j$. Ognuna delle K combinazioni calcolate è un numero reale che si può interpretare come un "punteggio" registrato dall' i -esima immagine in ogni classe di oggetti in Y (*class score*): l' i -esima immagine è classificata con l'etichetta $\mathbf{y}_j \in Y$ se l'elemento j -esimo del vettore output $f(\mathbf{x}^{(i)}; \mathbf{W}, \mathbf{b})$ è il massimo del medesimo vettore.

L'esempio in figura 1.4 mostra la classificazione di un'immagine di un gatto con $|Y| = 3$ classi (*gatto, cane, barca*). Per semplicità, l'immagine input è ipotizzata 2×2 e composta da un unico canale di colore ($c = 1$) (quindi \mathbf{x} , scritta come vettore colonna, è 4×1).

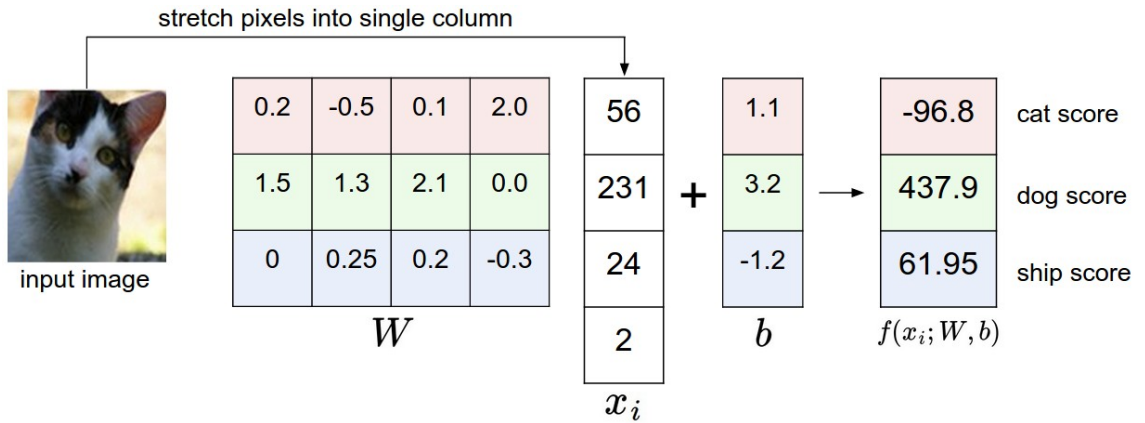


Figura 1.4: Mappatura di un'immagine ai punteggi di ogni classe mediante un classificatore lineare. Si noti che i pesi di \mathbf{W} non costituiscono un buon set di parametri: il punteggio assegnato alla classe "cane" (sbagliata) è alto e quello totalizzato dalla classe "gatto" (corretta) è basso. Il classificatore "è convinto" di aver classificato l'immagine di un cane.

⁴La fonte principale per gli argomenti trattati in questo paragrafo è [2]

Interpretare un classificatore lineare

Poiché le immagini sono "schiate" in vettori colonna hwc -dimensionali, si possono immaginare le immagini di un dataset come dei punti nello spazio \mathbb{R}^{hwc} . Analogamente, quindi, il dataset può essere pensato come una collezione di punti multidimensionali. Ovviamente non possiamo visualizzare questi spazi altamente dimensionali, ma se immaginiamo di "comprimere" tutte le dimensioni in sole due dimensioni otteniamo una visualizzazione del tipo in figura 1.5.

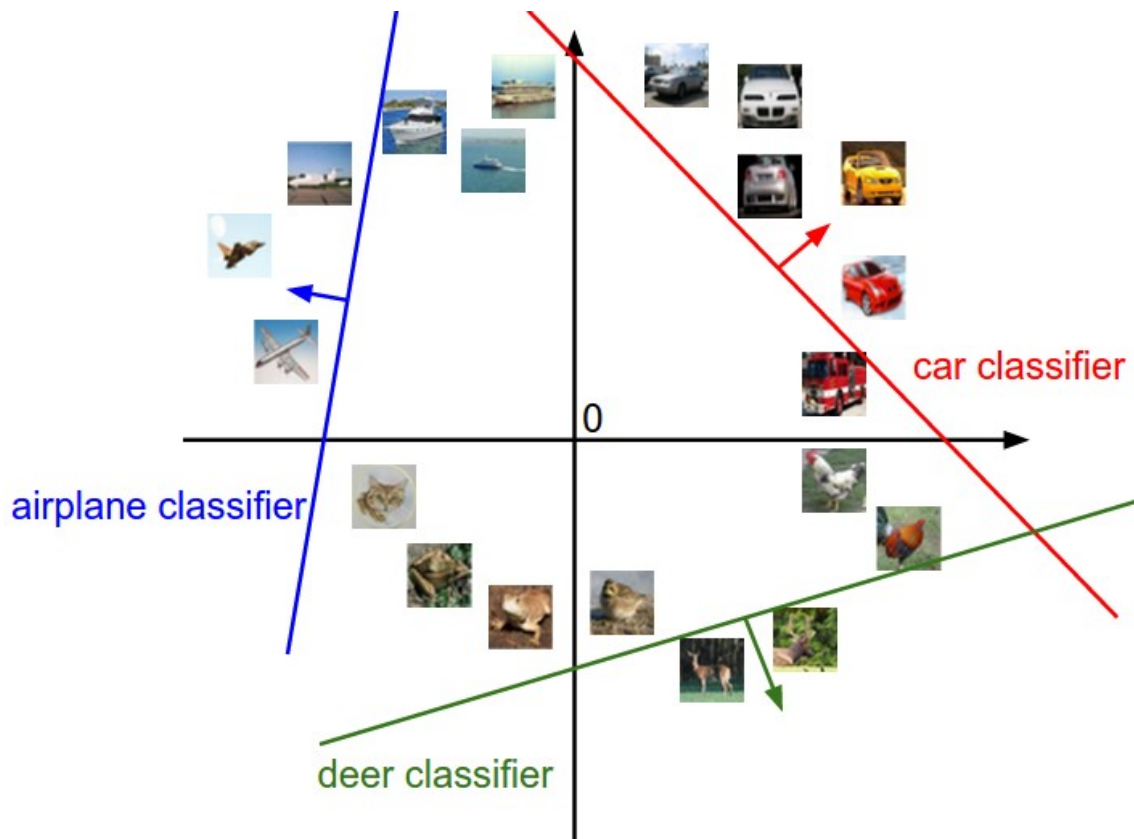


Figura 1.5: Visualizzazione di tre righe di un classificatore lineare, una per ciascuna delle classi "aereo", "auto", "cervo".

Le rette in figura devono in realtà essere pensate come degli iperpiani ($hwc-1$)-dimensionali, associati a ciascuna classe di Y (cioè a ciascuna riga di \mathbf{W} e \mathbf{b}), e il piano come l'iperspazio \mathbb{R}^{hwc} . Sussistono le seguenti interpretazioni geometriche:

- Le immagini sono dei punti nel piano. Ogni retta è il luogo dei punti che totalizzano un punteggio nullo per la classe associata a quella retta (la classe è scritta in figura accanto ad ogni retta). La freccia nella figura indica la direzione seguendo la quale i punti del piano aumentano (linearmente) il punteggio realizzato per quella classe.
- Modificare i pesi di \mathbf{W} significa regolare l'inclinazione delle rette (cioè ruotarle rispetto al punto di intercetta).

- Modificare i bias di \mathbf{b} significa regolare l'intercetta delle rette (cioè traslarle verticalmente).

Un altro modo di interpretare i pesi \mathbf{W} può essere quello di far corrispondere ogni riga di \mathbf{W} a un **prototipo** (in inglese **template**) per una delle classi. In questa interpretazione, il punteggio realizzato per ogni classe da un'immagine è ottenuto attraverso l'operazione di prodotto matriciale tra il prototipo della classe j (\mathbf{w}_j) e l'immagine da classificare ($\mathbf{x}^{(i)}$). Usando la terminologia introdotta, possiamo affermare che ciò che sta facendo il classificatore lineare è un'operazione di *template matching*, dove i *templates* sono oggetto di apprendimento da parte del classificatore⁵.

Ad esempio, analizziamo il dataset *CIFAR-10* [3]. Esso contiene immagini 32×32 ciascuna appartenente ad una di 10 classi. Visualizzando⁶ i pesi (e quindi i 10 templates) di un classificatore lineare addestrato su CIFAR-10 si ottengono i risultati in figura seguente:



Figura 1.6: Visualizzazione dei templates di un classificatore addestrato sul dataset CIFAR-10

Si possono fare alcune interessanti osservazioni.

Ad esempio, il prototipo della classe "barca" è composto da molti pixel blu disposti perlopiù lungo i margini, come ci si potrebbe aspettare dal momento che molte immagini di barche in CIFAR-10 mostrano queste in mare aperto. Questo template allora assegnerà un punteggio alto quando l'immagine che si vuole classificare è una barca in mare aperto. In altre parole, un'immagine realizzerà un punteggio tanto più alto in una certa classe quanto più essa è *simile* al template che il classificatore lineare *ha imparato* per quella classe.

Il prototipo per la classe "cavallo" sembra essere l'immagine di un cavallo a due teste; similmente, quello per la classe "auto" sembra una miscela di rappresentazioni di un'auto vista da più direzioni diverse. Ciò è coerente col fatto che il classificatore lineare è stato addestrato su immagini di cavalli visti rispetto a entrambi i fianchi e su immagini di auto raffigurate in tante direzioni diverse. Inoltre, il template per l'auto sembra rappresentare un'auto di colore rosso: evidentemente in CIFAR-10 la maggior parte delle automobili rappresentate sono di quel colore.

Come si vedrà nel seguito, questa operazione di *template matching* presenta una forte analogia con il funzionamento di una *Fully Connected Layer* di una rete neurale convoluzionale.

⁵Si introdurranno gli algoritmi di apprendimento (supervisionato) nel capitolo ??.

⁶TODO Per i dettagli su come "visualizzare" i pesi si veda <https://it.mathworks.com/help/deeplearning/examples/visualize-activations-of-a-convolutional-neural-network.html>.

Bias trick

Concludiamo questo capitolo menzionando un "trucco" matematico molto utilizzato per rappresentare \mathbf{W} e \mathbf{b} come un'unica matrice, semplificando di molto la notazione 1.1. Possiamo aggiungere il vettore dei bias in coda alla matrice dei pesi e aggiungere un "1" al vettore che rappresenta l'immagine. In questo modo, il classificatore lineare è rappresentato dalla funzione di associazione

$$f(\mathbf{x}^{(i)}; \mathbf{W}) = \mathbf{W}\mathbf{x}^{(i)} \quad (1.2)$$

In questa maniera, f calcola solo combinazioni lineari (un singolo prodotto matriciale), poiché il vettore dei bias è stato eliminato. Tale utile passaggio, noto come *bias trick*, è visualizzato nella seguente figura

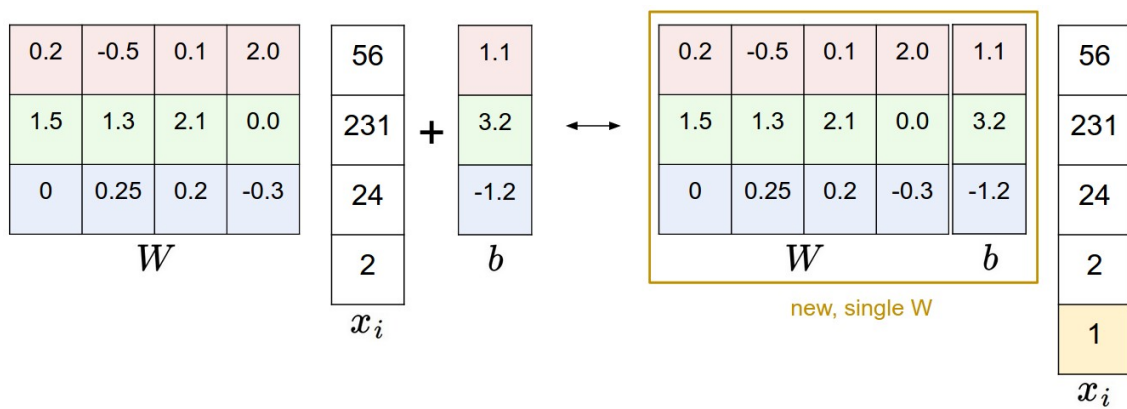


Figura 1.7: Bias trick

TODO: loss functions.

Bibliografia

- [1] Stuart Russell e Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 2009.
- [2] Andrej Karpathy. *Lectures of Stanford class CS231n: Convolutional Neural Networks for Visual Recognition*. Appunti del corso Stanford CS class C231n. 2019.
- [3] Alex Krizhevsky. «Learning Multiple Layers of Features from Tiny Images». In: (2009).
- [4] Flavio Forenza. *Tecniche innovative di Computer Vision per la Foto-Identificazione dei cetacei*. Tesi di laurea triennale, a.a. 2017/2018, rel. prof. Giovanni Dimauro, correl. dr. Vito Renò.
- [5] Ian Goodfellow, Yoshua Bengio e Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.