

# BiBirra: Deep image retrieval for beer recognition

Matteo Ronchetti

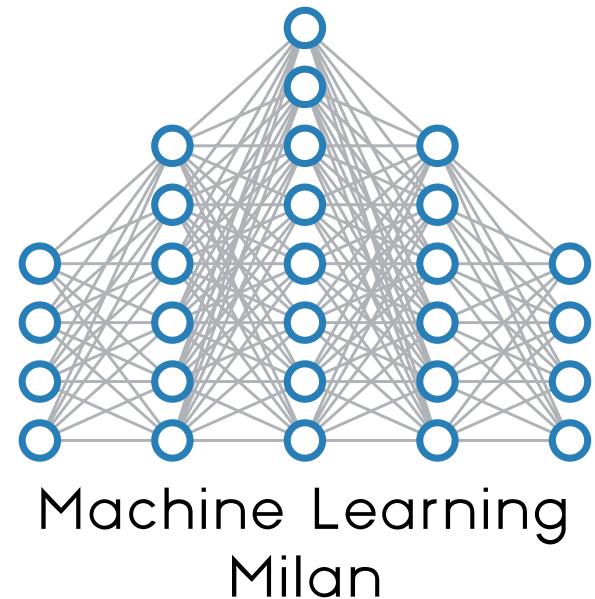
# About Me

- Matteo Ronchetti
- Triennale in Matematica
- Magistrale in Artificial Intelligence a Pisa
- <https://matteo.ronchetti.xyz>
- Organizzatore dei meetup "Machine Learning Milan"



# Machine Learning Milan

- Secondo meetup di ML più grande in Italia
- Meetups serali con talks
- Workshops hands on
- Accesso libero e gratuito
- [www.meetup.com/it-IT/Machine-Learning-Milan](http://www.meetup.com/it-IT/Machine-Learning-Milan)



# Prossimo Meetup

- Reinforcement Learning & Computer Vision
- Ospitato da Google
- Giovedì 7 novembre alle 19:00
- <https://www.meetup.com/it-IT/Machine-Learning-Milan/events/265345835/>



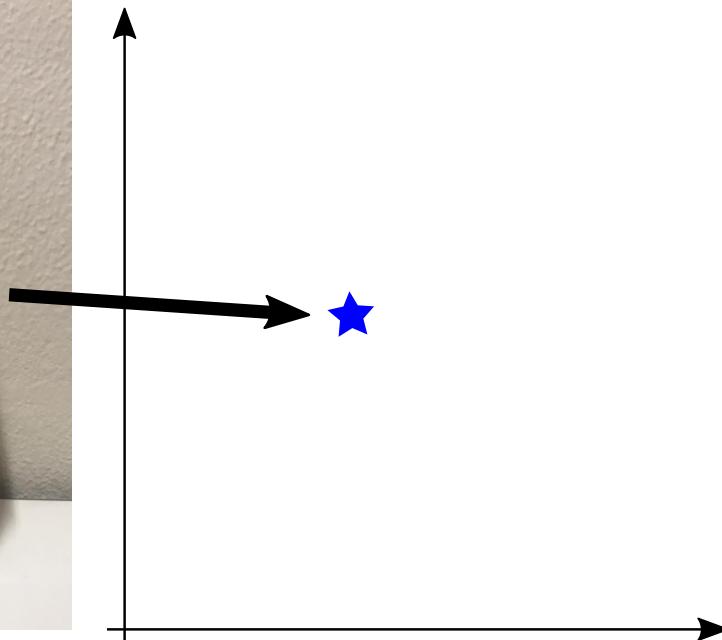
# Riconoscere birre da una foto



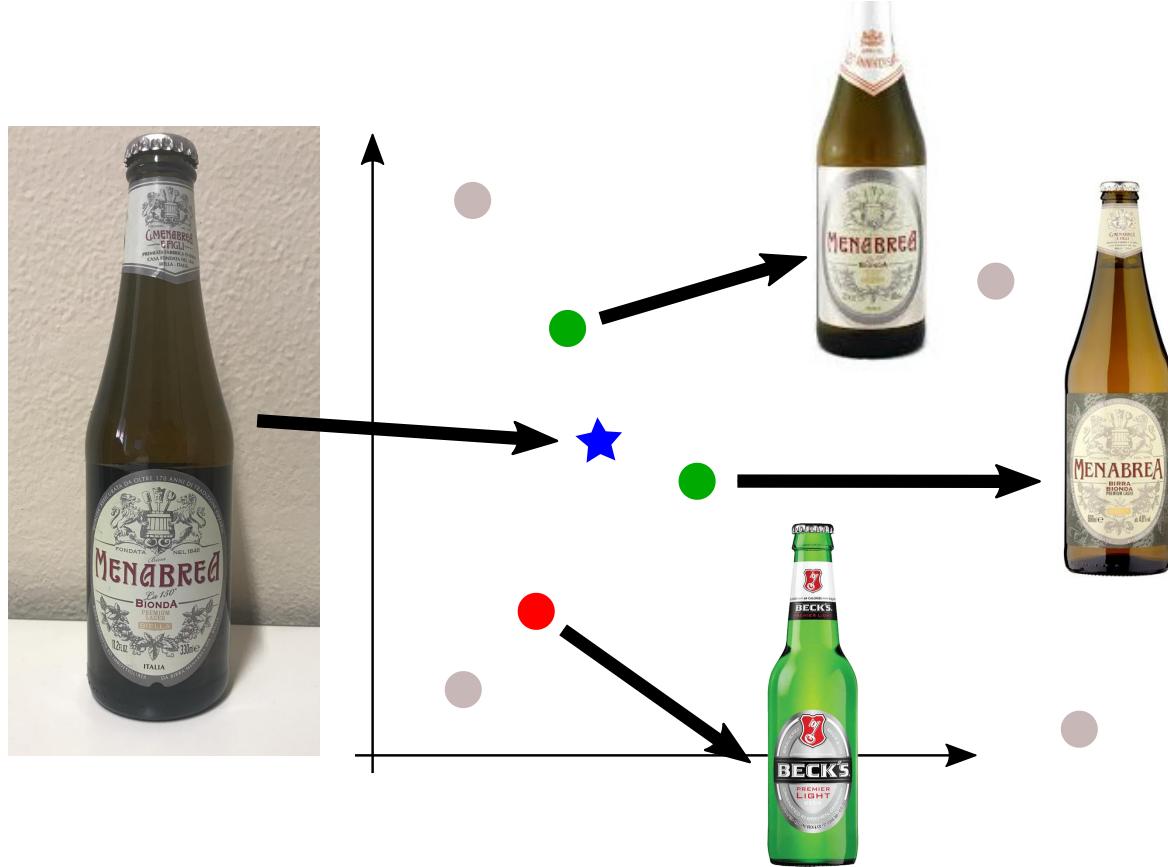
# Bottle Detection



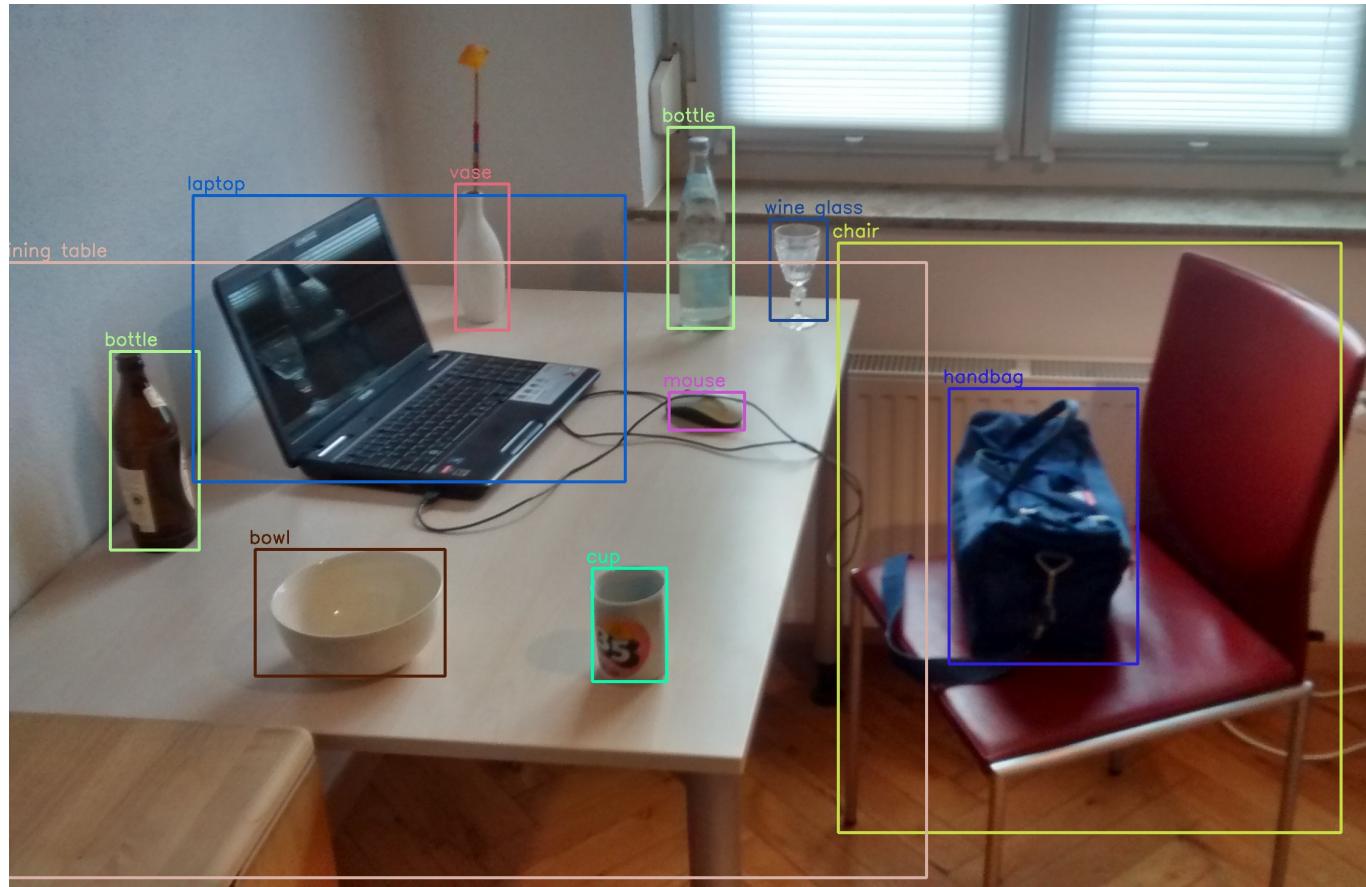
# Bottle Description



# Match Nearest Neighbors



# Detection



# Sliding Window

- Si provano tanti rettangoli di varie forme e in varie posizioni
- Il problema si riduce a capire se la parte d'immagine delimitata dal rettangolo è una birra o meno



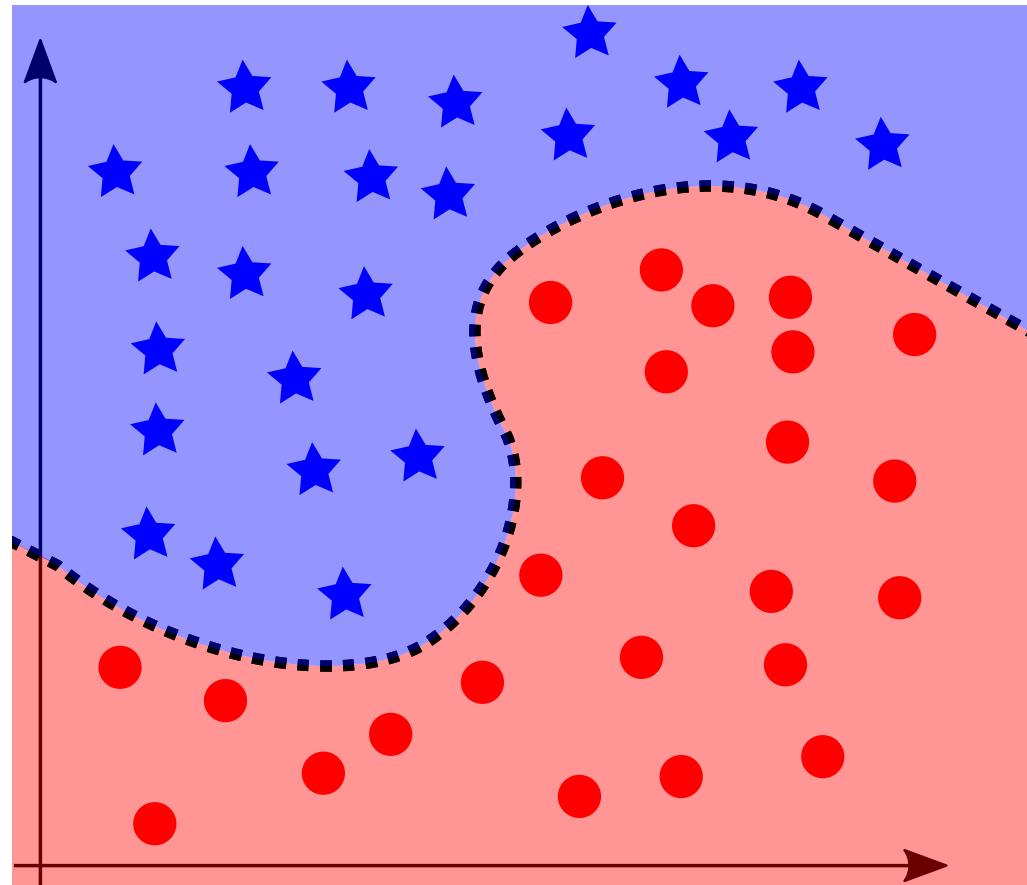
# Classificatore Binario

- Modello di Machine Learning che distingue tra due classi
- Esempio: distingue cane da gatto, bottiglia da non bottiglia, Chihuahua da muffin

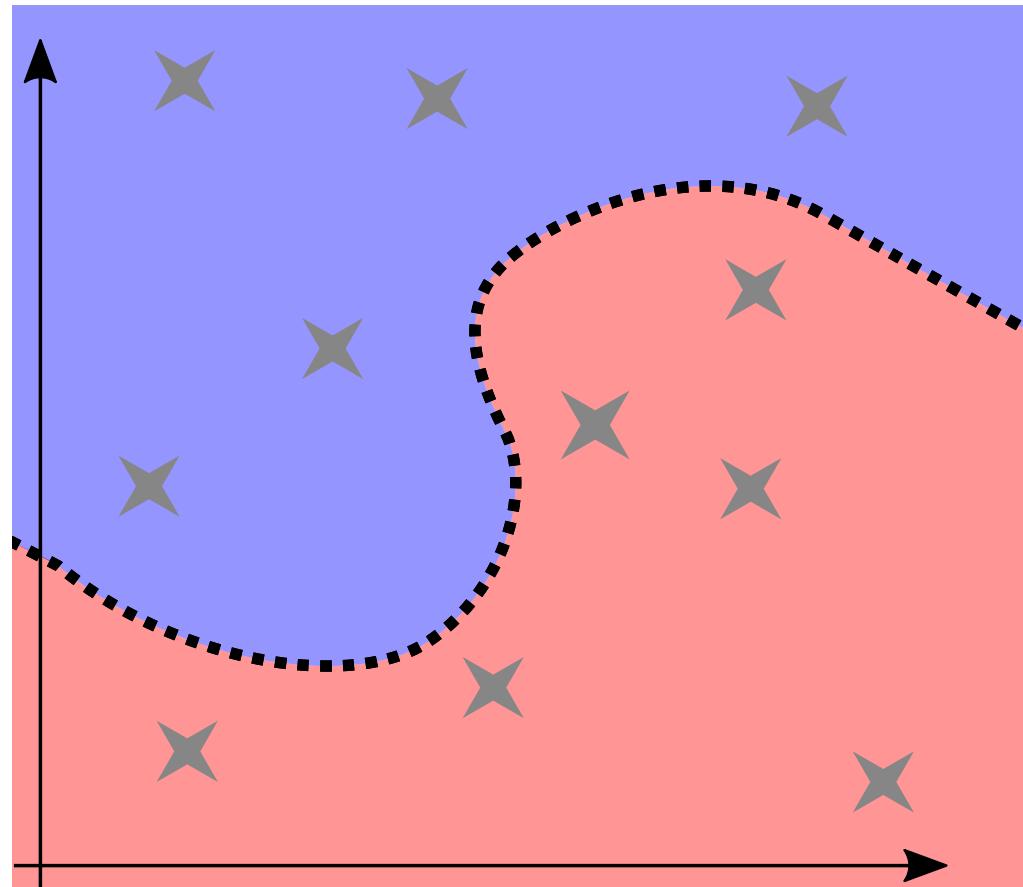


@teenybiscuit

# Classificatore: Training



# Classificatore: Utilizzo

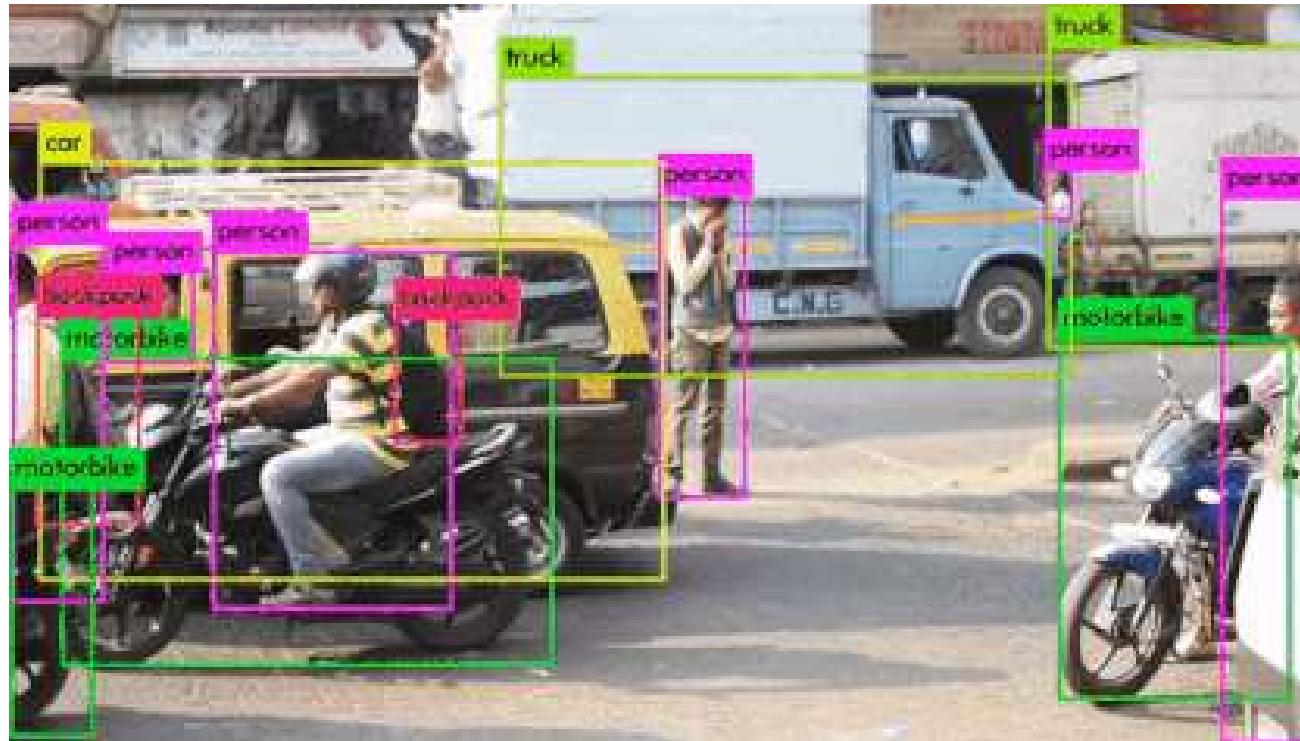


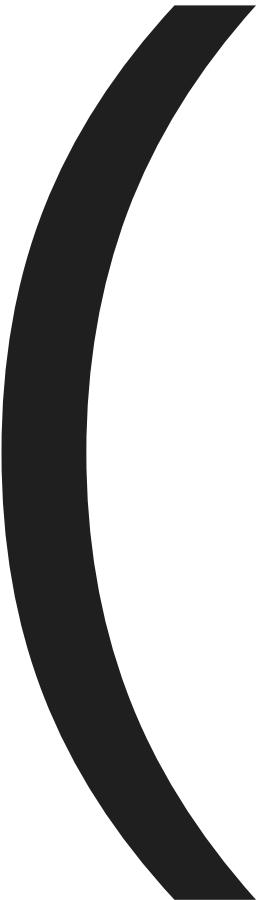
# Sliding Window

- Detection diventa classificazione, quindi più facile
- Lo svantaggio è che provare molti rettangoli può essere dispendioso



# YOLO: Approccio più moderno

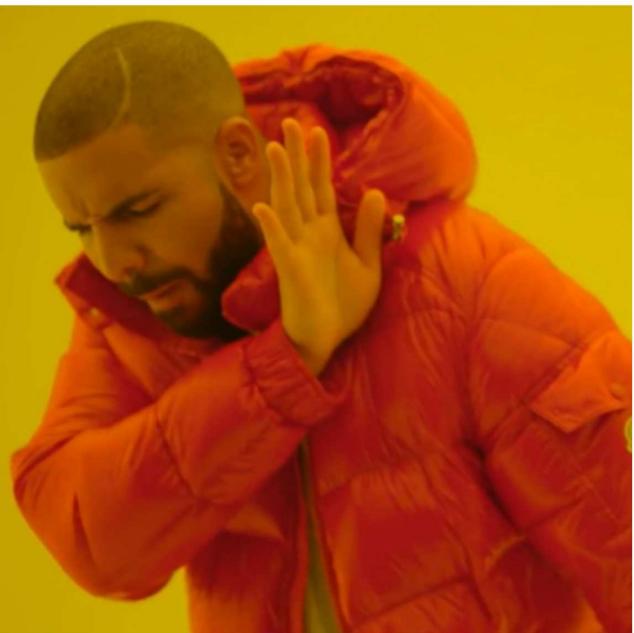




Parentesi su come trattare  
in pratica un modello di  
Machine Learning

# 1. Cercare un modello pretrained

- Si può evitare parte del lavoro
- Si può imparare dall'esperienza di altri
- Se il modello non fosse adeguato può comunque essere utile (transfer learning, knowledge distillation)



# Google

Bottle detection pretrained model

Google Search

I'm Feeling Lucky

Google offered in: [Italiano](#)

# Google

Object detection pretrained model

Google Search

I'm Feeling Lucky

Google offered in: [Italiano](#)

## 2. Cercare un dataset

- Avere un dataset di buone dimensioni e di buona qualità è fondamentale
- Molti dataset pubblici
- Se si è trovato un modello la pubblicazione relativa sicuramente descrive anche il dataset
- Trovato un dataset è facile trovare modelli che lo utilizzano (si torna al punto 1)
- Senza dataset non si può fare nulla, bisogna cercare di essere creativi, automatizzare il più possibile, ma comunque una certa quantità di lavoro manuale è necessaria



**THAT'S A VERY BORING WORK**

**SCRAPING THE INTERNET FOR  
THOUSANDS OF BEER PICTURES**

### 3. Creare il proprio modello

# Tools

- Scikit-Learn per il machine learning classico
- Molti frameworks per deep learning: Pytorch, Keras, Tensorflow...
- Ottimo supporto Linux
- Open source
- Buona documentazione



# CPU/GPU



# Solo GPU NVIDIA

- Librerie propietarie (cuDNN, cuBLAS, ...)
- NVIDIA ha investito molto a riguardo
- Progetti come ROCm devono ancora recuperare terreno



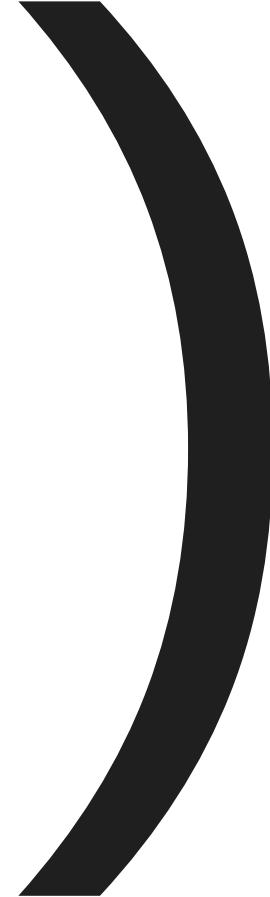
# Cloud

Progetti abbastanza grandi (soprattutto con reti neurali) necessitano di:

- Molto spazio di storage
- Molta potenza di calcolo (a volte CPU, spesso GPU)
- Tempi di lavoro lunghi (giorni di training).

Per questo può essere utile utilizzare servizi cloud dove è possibile noleggiare storage e potenza di calcolo.

Chiusa parentesi



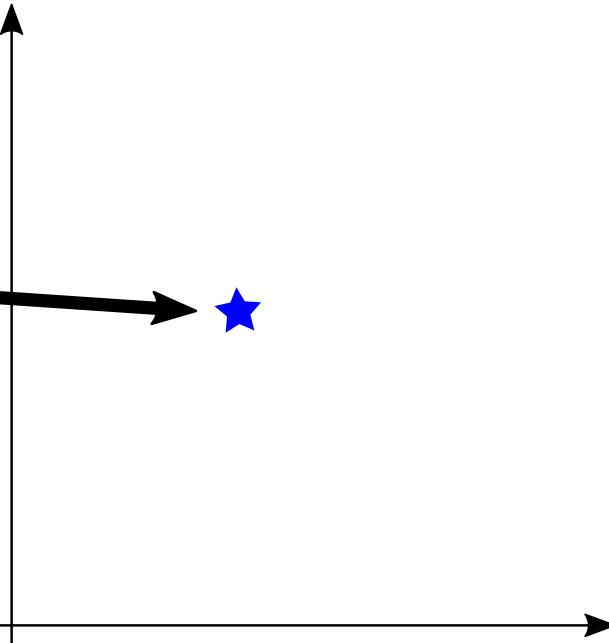
# Riconoscere una bottiglia: usiamo un classificatore?

Non è fattibile, ci sono migliaia di bottiglie diverse quindi migliaia di classi ognuna con pochissimi esempi.

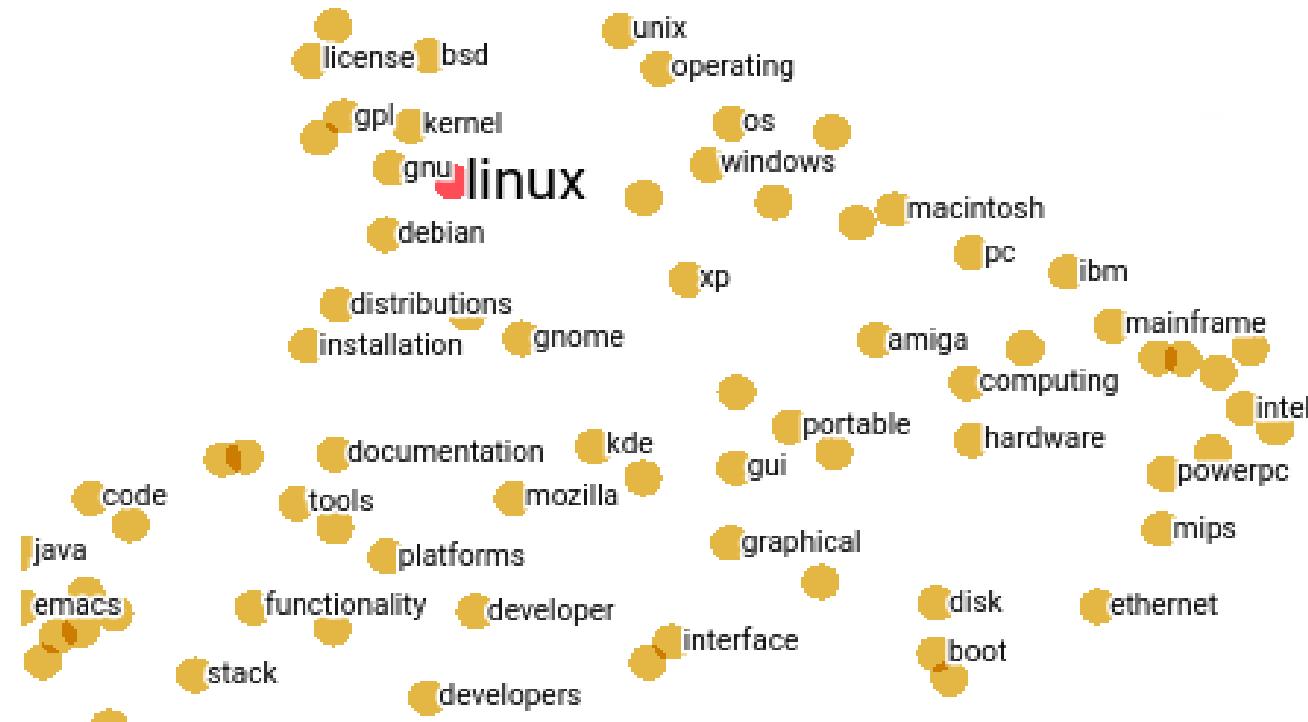
Data un'immagine query vogliamo confrontarla con tutte le immagini nel nostro database per trovare le più simili

# Bottle descriptor

Associamo ad ogni immagine un punto in uno spazio a n dimensioni (un array di n floats), chiamamo questo punto embedding.



# Word embeddings



Source: <https://projector.tensorflow.org>

# Perchè usare embeddings?

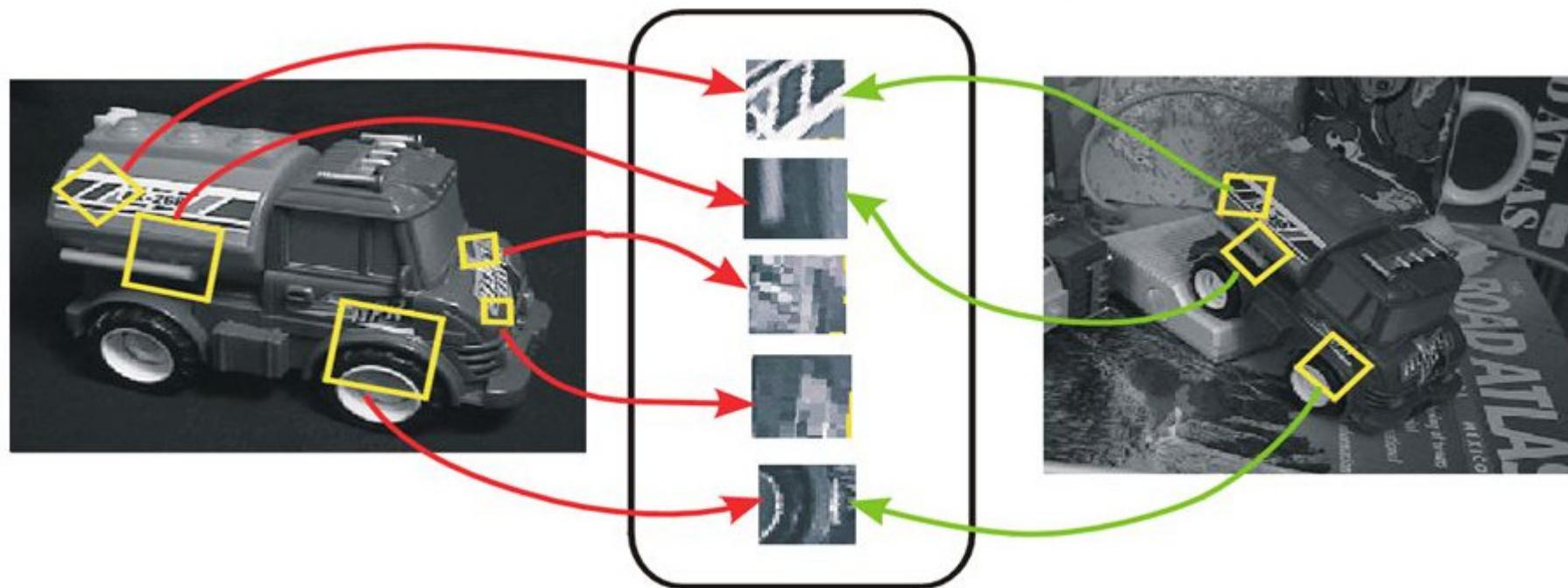
Ci sono molte trasformazioni che cambiano i pixels senza modificare il contenuto: rotazioni, traslazioni, riflessi, piccole variazioni di colore e luminosità...



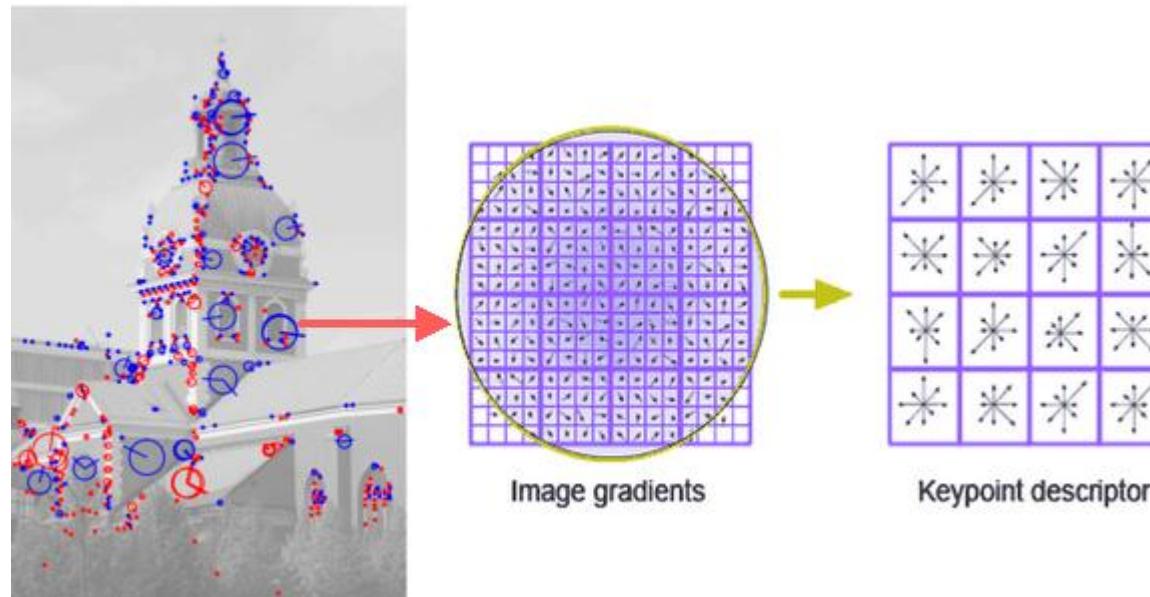
# Perchè usare embeddings?

- Un'immagine contiene moltissime informazioni superflue
- Estrarre solo le informazioni importanti
- Risparmiare memoria e queries più veloci

# Features locali



# SIFT: Scale Invariant Feature Transform



# Hardnet: Approccio più moderno

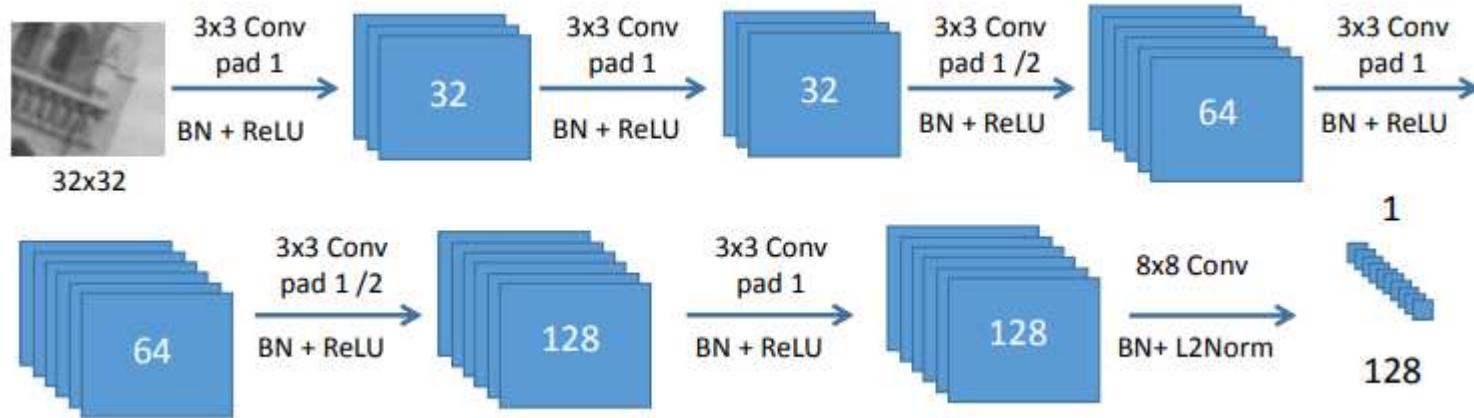


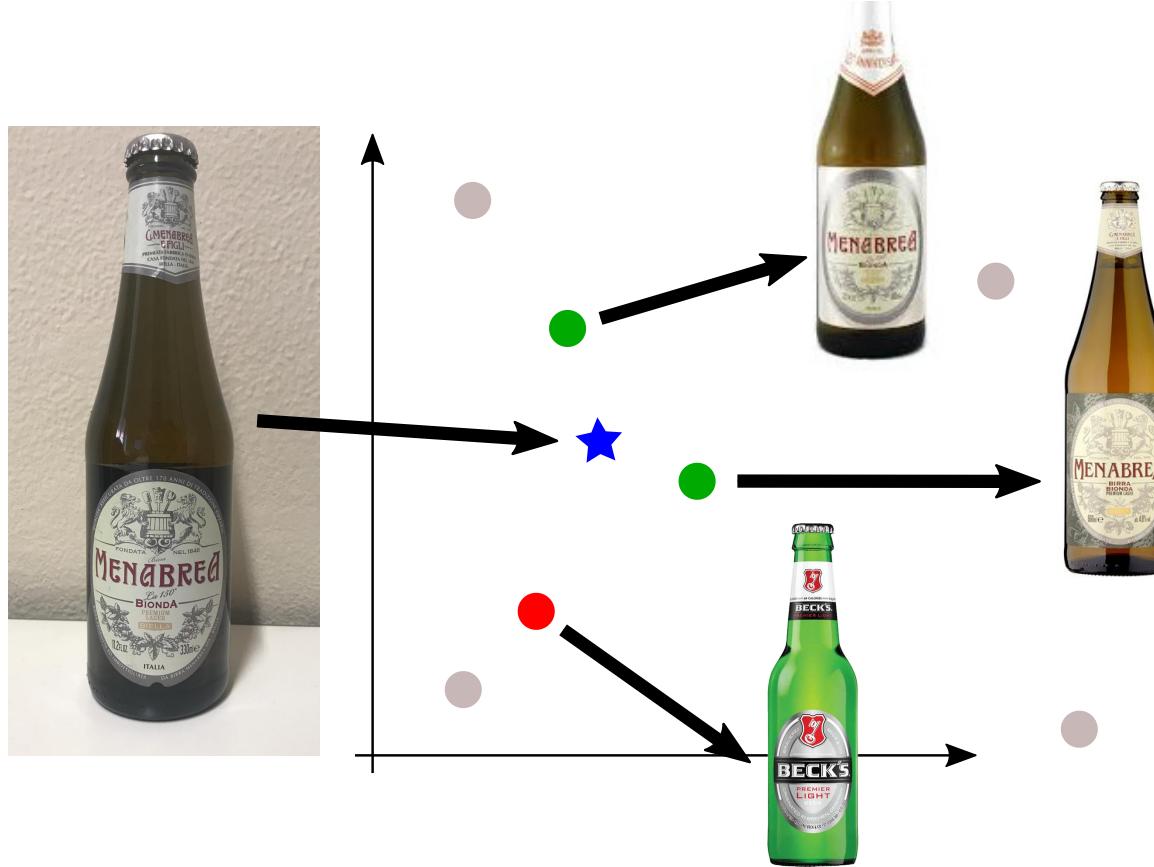
Image taken from the paper Working hard to know your neighbor's margins: Local descriptor learning loss , A. Mishchuk et al. (2017)

Github repo

# Aggregazione features locali

- Media (non è una buona idea)
- Fisher vectors
- VLAD (vector of locally aggregated descriptors)
- Kernel approximation

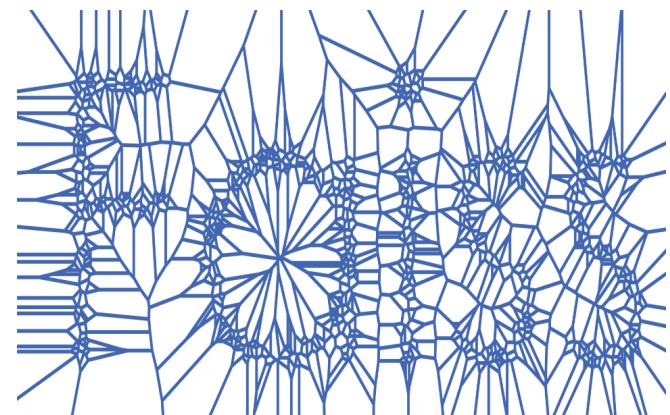
# Match Nearest Neighbors



# Indicizzazione per velocizzare la ricerca

Vari tools open source:

- Faiss (Facebook AI Similarity Search)
- Annoy (Approximate Nearest Neighbors Oh Yeah)
- OpenCV contiene alcuni indici



# Domande?