

DEGENERAZIONE MACULARE SENILE

L'USO DI **CNN** COME STRUMENTO DI DIAGNOSI

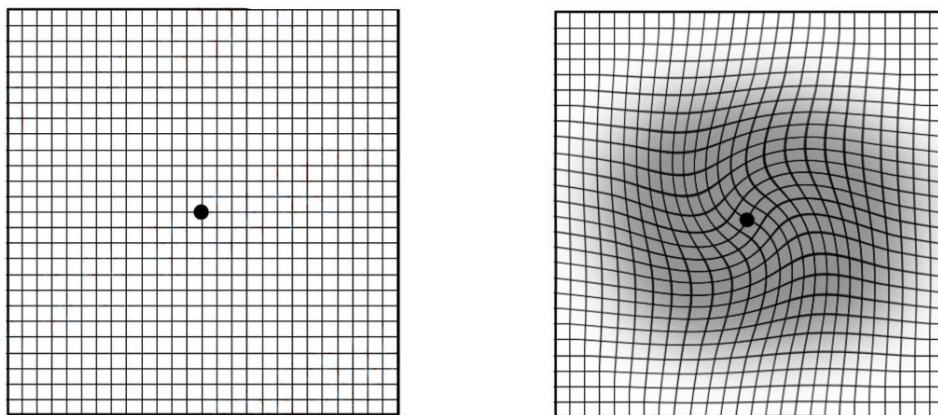


COS'È LA DMS

La degenerazione maculare senile è una **malattia oculare** legata all'invecchiamento

È la principale causa di perdita grave della visione centrale dopo i 55 anni

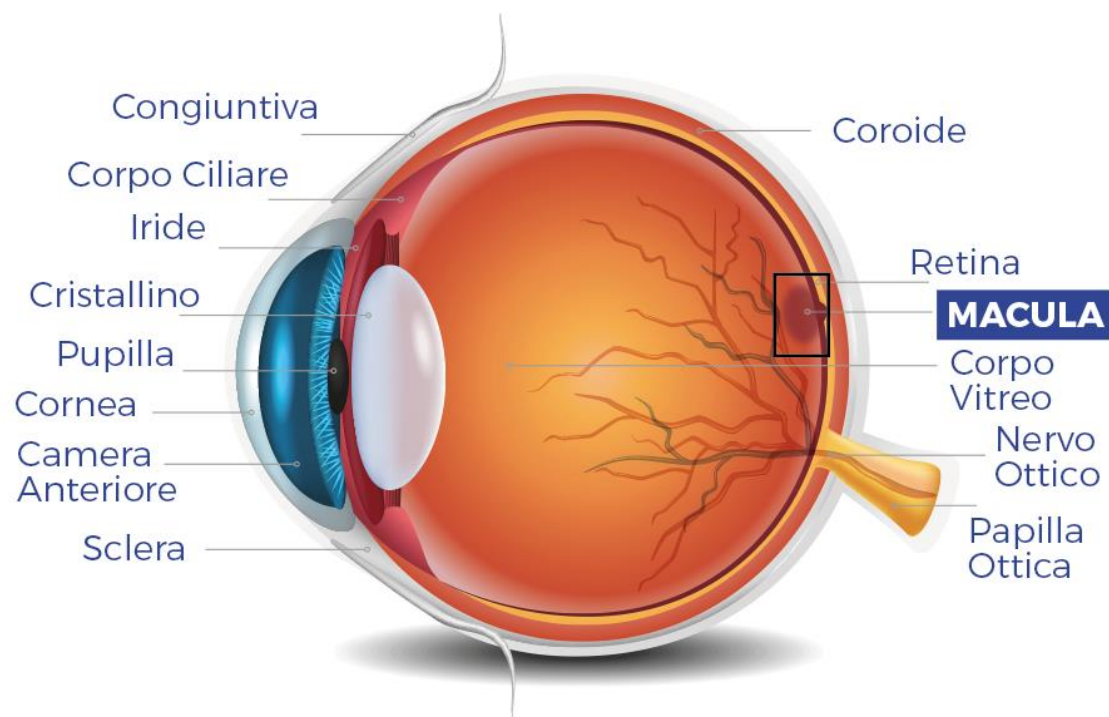
Griglia di Amsler



Se in stato avanzato,
si manifesta
causando **visione offuscata**
o assente al centro del
campo visivo.

COS'È LA DMS

La malattia colpisce la **macula***, ossia la porzione più centrale della retina.



La perdita della vista avviene perché le cellule visive **degenerano** e muoiono nella regione maculare della retina

*Sensibile agli stimoli luminosi, la macula è responsabile della visione a colori ad alta risoluzione, della visione distinta e della percezione dei dettagli

TIPOLOGIE DI PRESENTAZIONE

Forma atrofica

Forma secca:

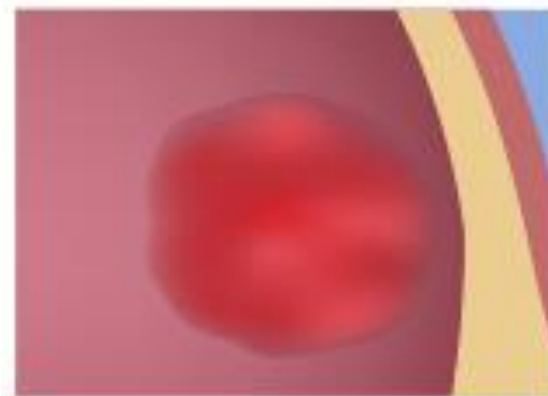
Si tratta di accumuli di scorie cellulari che possono riassorbirsi o calcificare.



Forma essudativa

Forma umida:

Oltre alle lesioni, si presenta la formazione anomala di nuovi vasi sotto la retina.

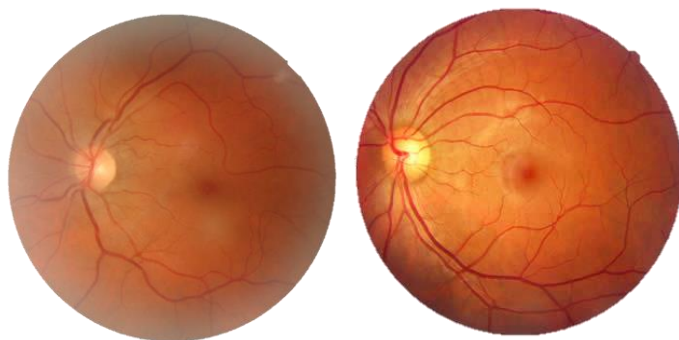


DIAGNOSTICA

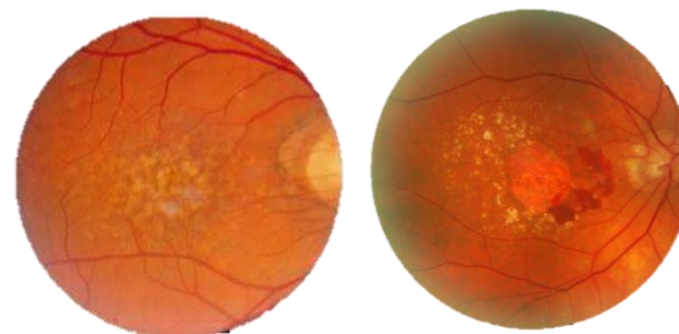
La diagnosi di DMS si effettua mediante visita oculistica con esame del fondo oculare tramite **OCT*** (tomografia ottica a luce coerente).

Ciò permette di ottenere delle **scansioni** della cornea e della retina

SANO



MALATO



*Tecnica di diagnosi basata sull'interferometria a luce bianca o a bassa coerenza, un fascio laser privo di radiazioni nocive che viene impiegato per analizzare le strutture oculari soprattutto retiniche e corneali mediante sezioni ad alta risoluzione.

IL PROBLEMA

196 M

PERSONE IN TUTTO IL
MONDO AFFETTE DA
DMS*

10,4 M

CASI DI COMPROMISSIONE
DELLA VISTA DA MODERATA A
GRAVE*

È evidente come sia sempre più necessario un sistema di elaborazione efficiente ed affidabile in grado di elaborare una **grande quantità di immagini** mediche dalle quali estrapolare una diagnosi accurata



RETI NEURALI: UNA PREZIOSA RISORSA

Le **reti neurali di convoluzione (CNN)** mostrano capacità di riconoscimento delle immagini superiori rispetto ad altri tipi di reti neurali.

Esistono infatti algoritmi basati su CNN in grado di individuare la DMS mediante l'analisi delle **fotografie a colori** del fondo oculare





CNN: COSA SONO

Rappresentano un sottoinsieme del **machine learning** e hanno un ruolo fondamentale negli algoritmi di deep learning.

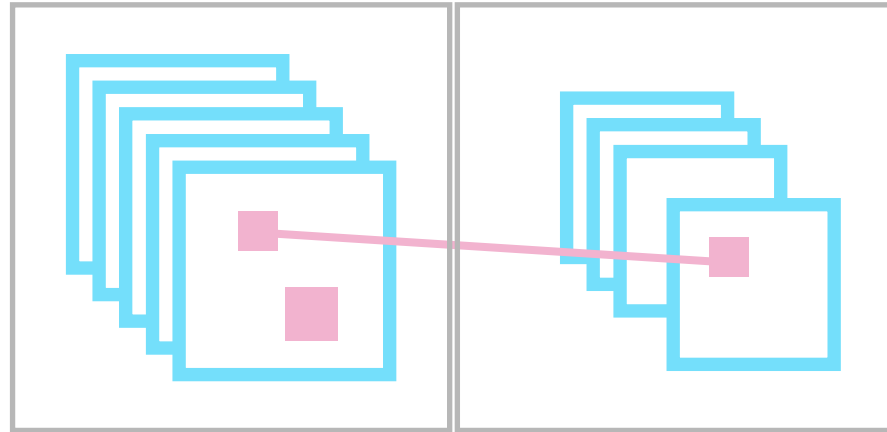
Obiettivo: identificare delle features da un'immagine di input

Sono costituite da **livelli di convoluzione** connessi tra loro che identificano la stessa features in aree diverse dell'immagine

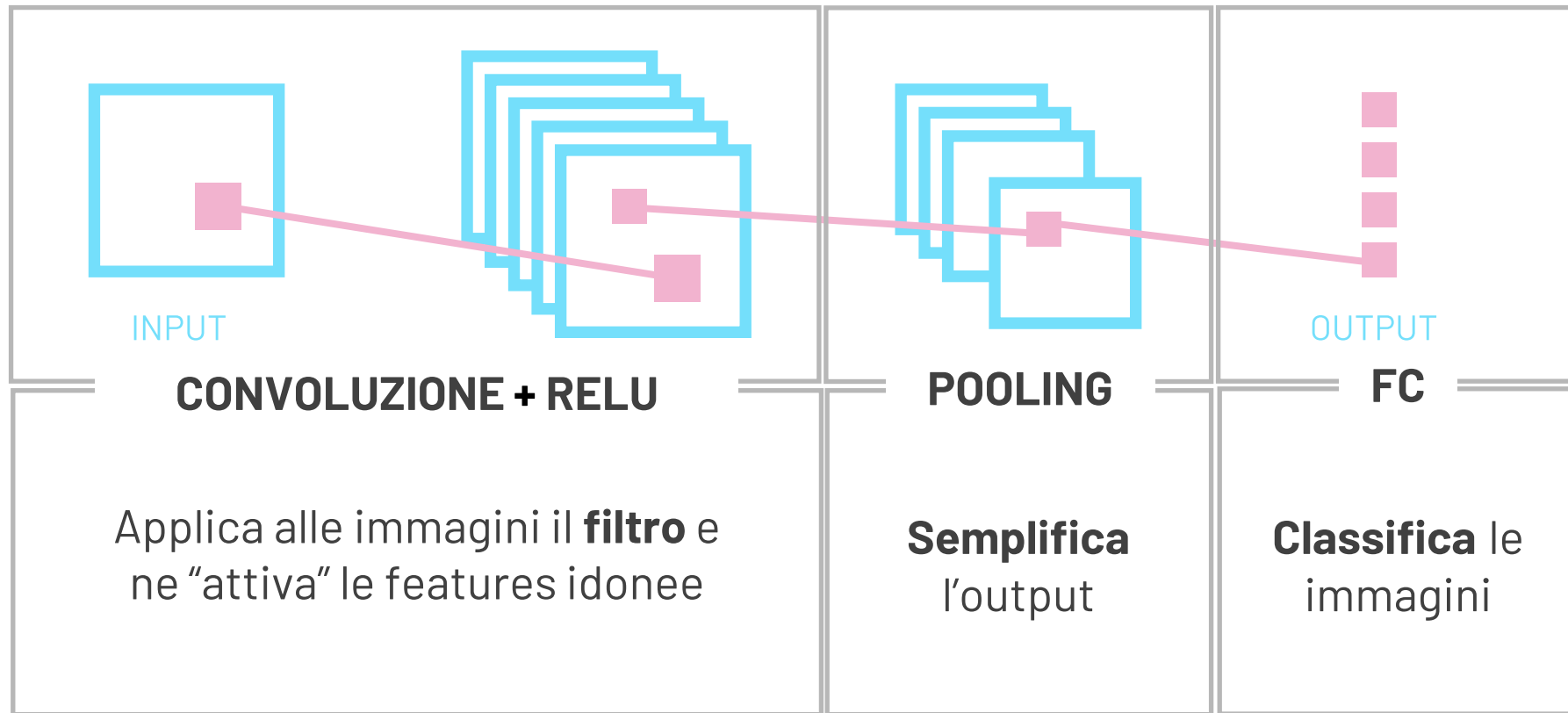


CNN: FUNZIONAMENTO

- 1) Ogni layer **identifica** la propria features tramite un filtro
- 2) L'output di ogni nodo diventa l'input **del nodo successivo**



CNN: FUNZIONAMENTO





L'IDEA

“Sviluppare un algoritmo di riconoscimento tramite CNN per identificare la malattia a partire dalle immagini ottenute da un'esame OCT”

COME?

Addestrando un **modello sequenziale** con un dataset di immagini OCT



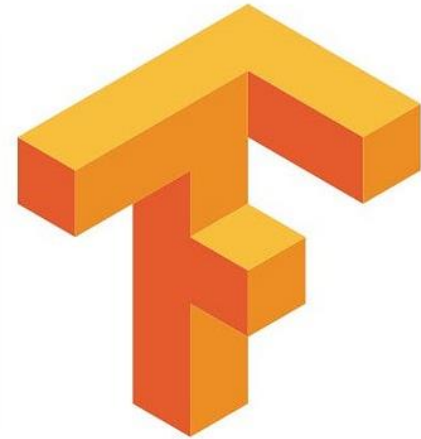
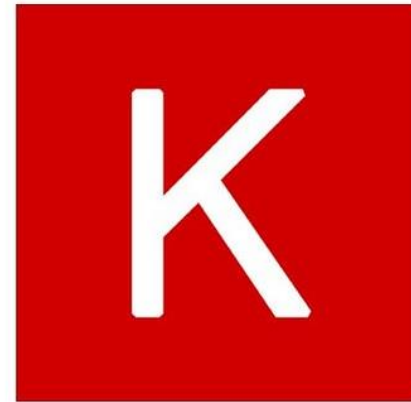
L'ARCHITETTURA: KERAS E TENSORFLOW

KERAS

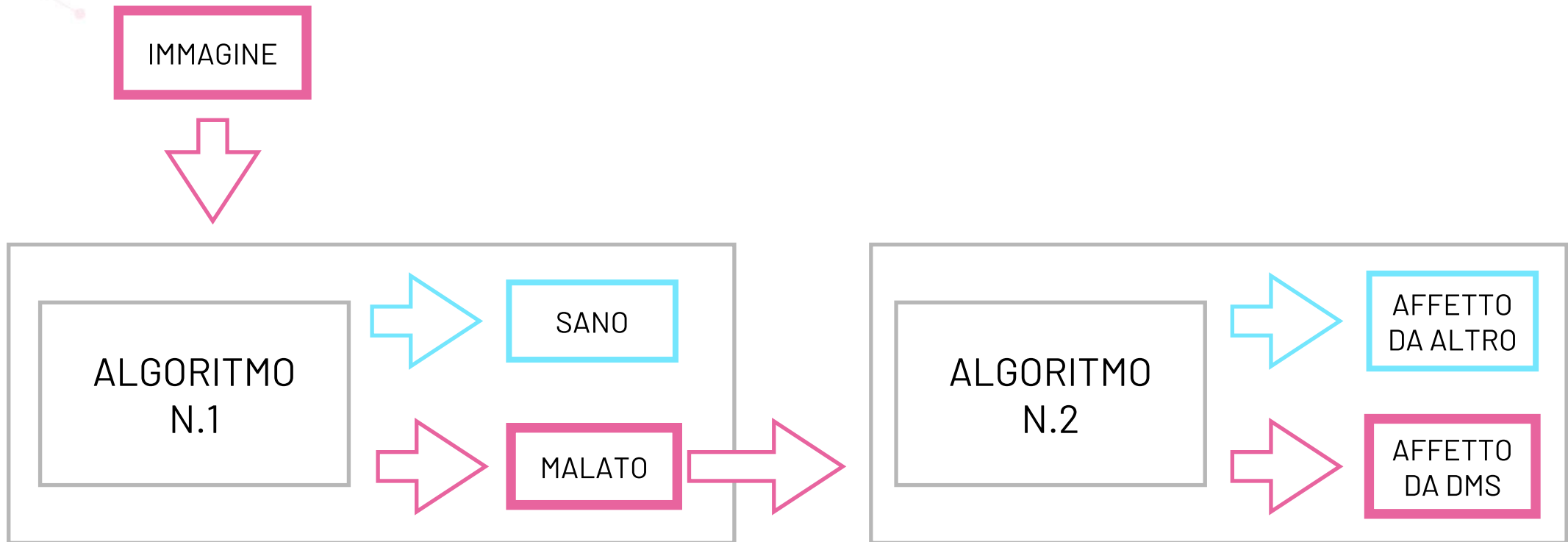
Una libreria di alto livello scritta in Python per l'apprendimento automatico e le reti neurali

TENSORFLOW

API sequenziale di Google basata sulle librerie di Keras

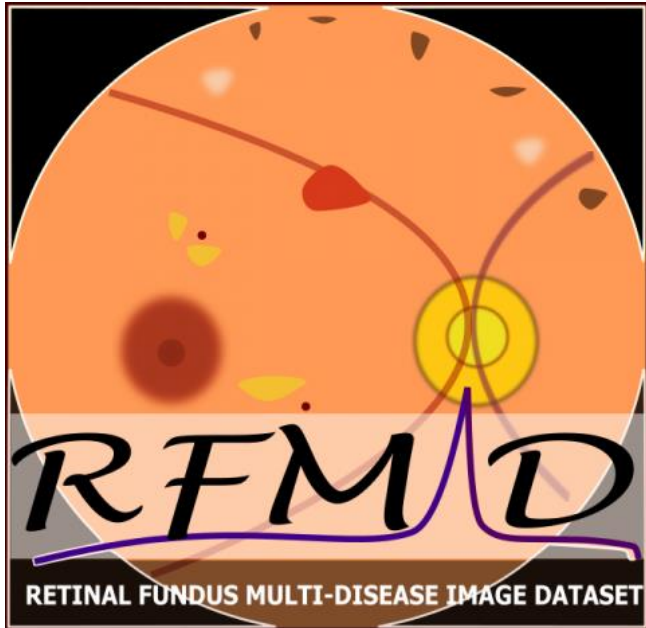


L'ARCHITETTURA: SCHEMA



IL DATASET UTILIZZATO

IEEE*DataPort*TM



3200

SCAN DEL FUNDUS

46

POSSIBILI PATOLOGIE

BILANCIAMENTO DEI DATI

Per evitare **overfitting** nell'addestramento

Data Augmentation

```
data_augmentation = keras.Sequential([
    layers.RandomFlip("horizontal",
                      input_shape=(IMG_SIZE[0],
                                    IMG_SIZE[1],
                                    3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
])
```

Class Weights

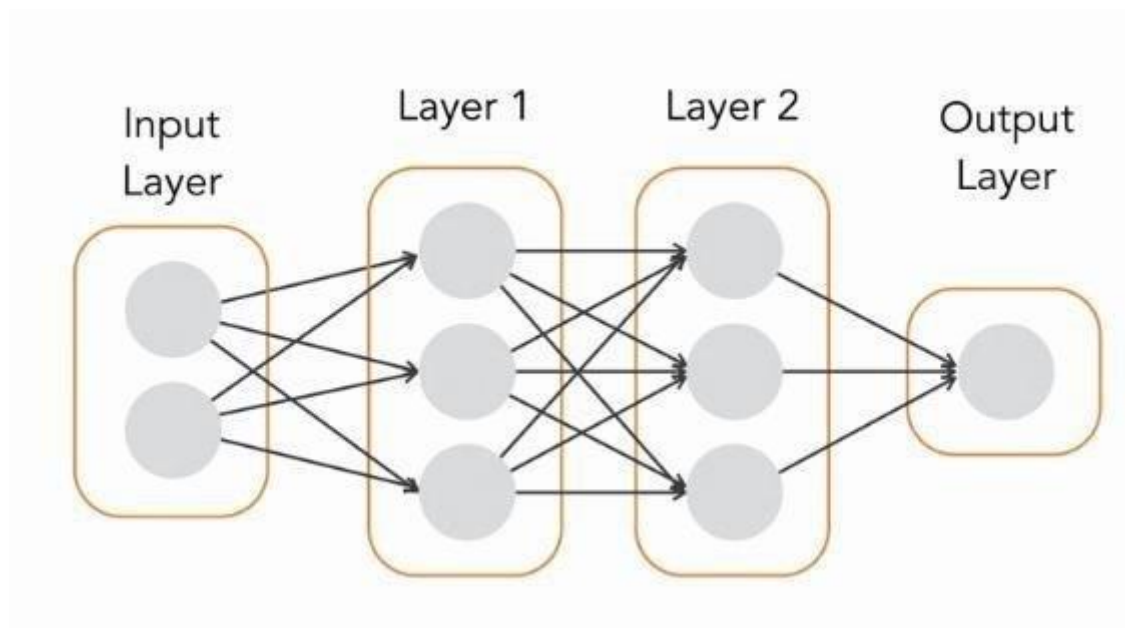
```
imgfolder0 = Path(f"{train_dir}/ARMD")
imgfolder1 = Path(f"{train_dir}/OTHER")
image_count0 = len(list(imgfolder0.glob('*.*png')))
image_count1 = len(list(imgfolder1.glob('*.*png')))
weight_for_0 = (image_count0+image_count1)/image_count0
weight_for_1 = (image_count0+image_count1)/image_count1

class_weight = {0: weight_for_0, 1: weight_for_1}

print('Weight for class 0: {:.2f}'.format(weight_for_0))
print('Weight for class 1: {:.2f}'.format(weight_for_1))
```

COMPOSIZIONE

Modello sequenziale



MODELLO SEQUENZIALE

```
model = Sequential([
    data_augmentation,
    layers.Rescaling(1./255),
    layers.Conv2D(16, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(32, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(64, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Dropout(0.2),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(num_classes)
])
```

Input Layer

**Convolutional e Pooling
layers**

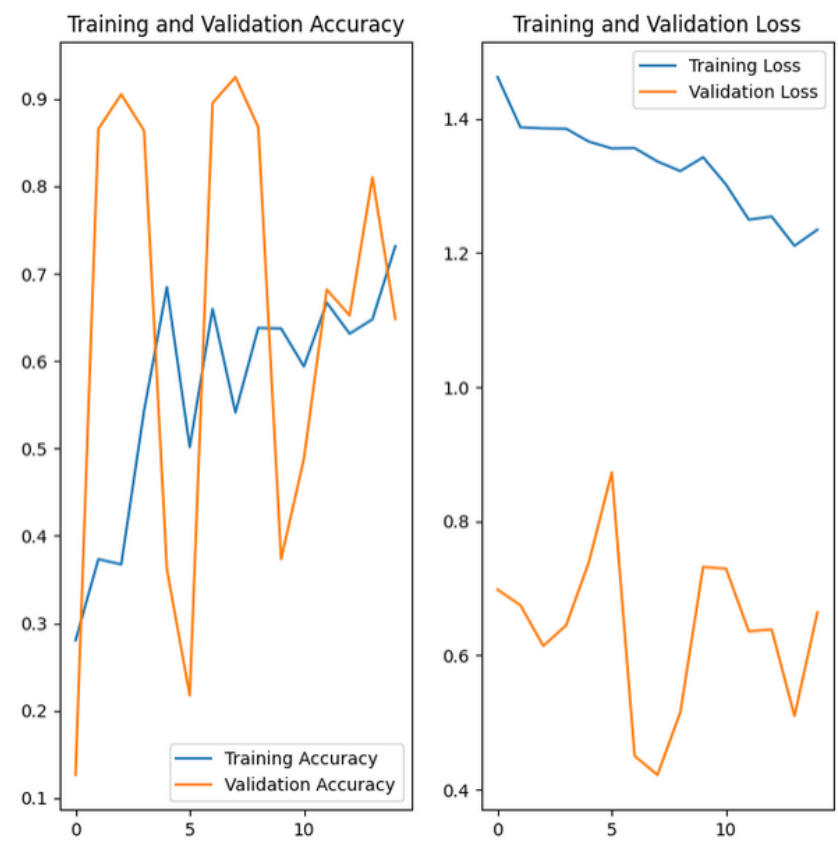
Output Layer

RISULTATI

Primo modello



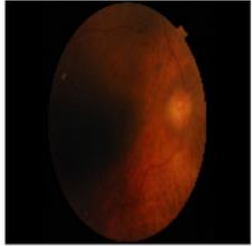
Secondo modello



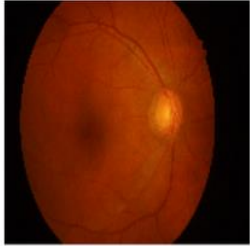
RISULTATI

Primo modello

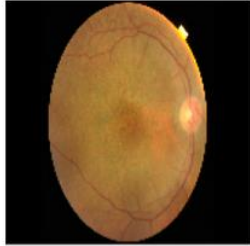
Prediction:Malate Truth:Malate



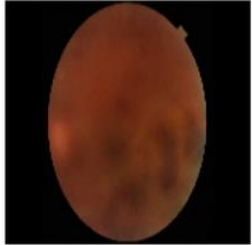
Prediction:Malate Truth:Sane



Prediction:Malate Truth:Malate



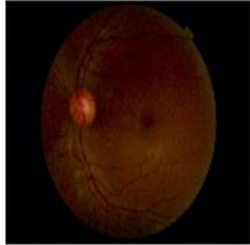
Prediction:Sane Truth:Malate



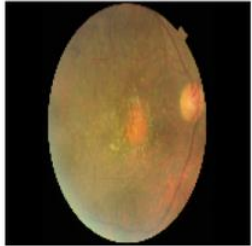
Prediction:Sane Truth:Sane



Prediction:Malate Truth:Sane



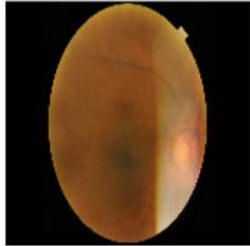
Prediction:Sane Truth:Malate



Prediction:Malate Truth:Malate

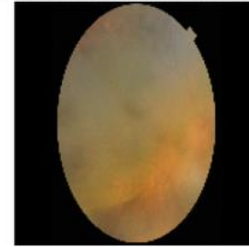


Prediction:Malate Truth:Malate

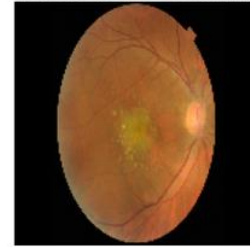


Secondo modello

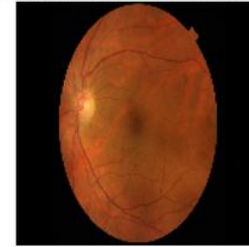
Prediction:OTHER Truth:OTHER



Prediction:ARMD Truth:ARMD



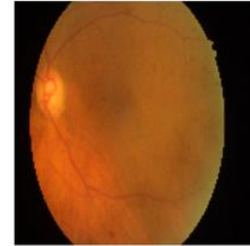
Prediction:OTHER Truth:OTHER



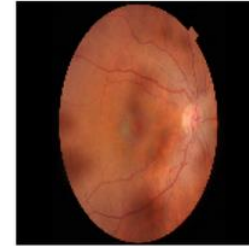
Prediction:ARMD Truth:OTHER



Prediction:ARMD Truth:OTHER



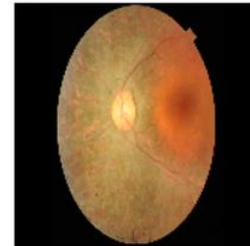
Prediction:OTHER Truth:OTHER



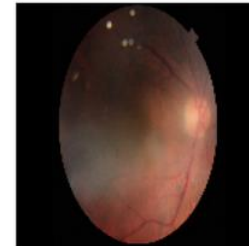
Prediction:OTHER Truth:OTHER



Prediction:ARMD Truth:OTHER



Prediction:OTHER Truth:OTHER



CARATTERISTICHE

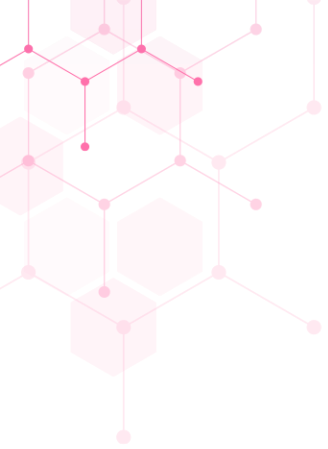
Punti di forza

- **Facile** implementazione
- Poca potenza di **calcolo**

Punti di debolezza

- **Dataset** ridotto
- Richiede immagini alla stessa **risoluzione**

Di seguito, valutiamo delle possibili modifiche e **miglioramenti** volti ad ovviare a queste limitazioni.

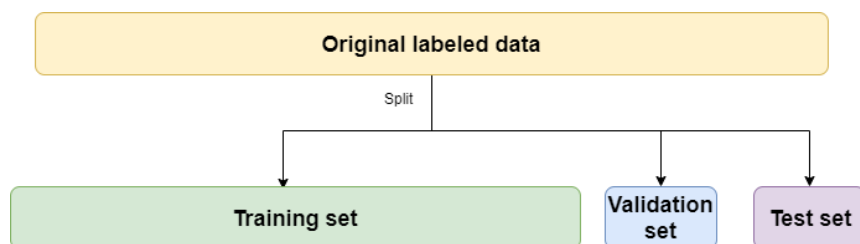


1) VALIDAZIONE

cambiare **strategia di validazione** risolve i problemi dati da un Dataset ridotto

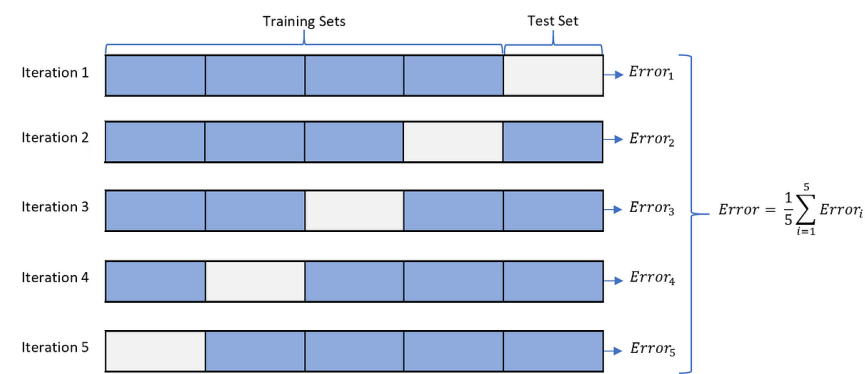
BLINDFOLD VALIDATION

divide il dataset in 3 (training set, validation set e test set)



TEN-FOLD CROSS-VALIDATION

sfruttare le stesse immagini per più operazioni e calcolarne la **media** alla fine.



VALIDAZIONE

Risultati da un dataset ridotto (600 campioni ca.)

BLINDFOLD VALIDATION Fase di training

Dai **106s** a **110s**



Efficienza **91.17%**

TEN-FOLD CROSS- VALIDATION Fase di training

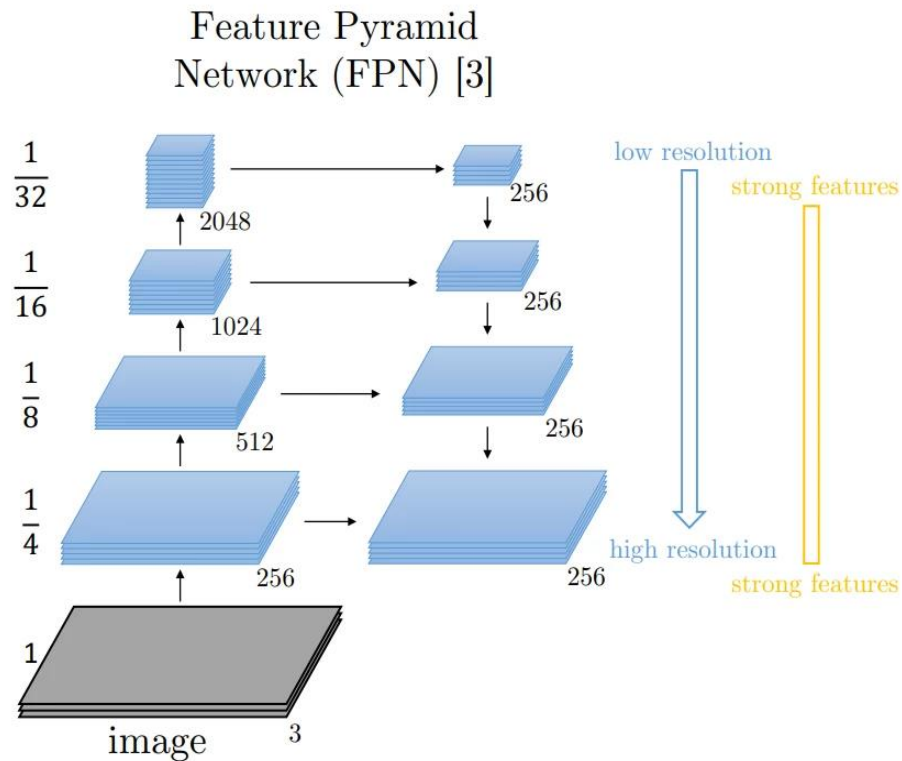
Dai **176s** a **190s**



Efficienza **95.45%**

2) STRUTTURA PIRAMIDALE

Integrare una **struttura piramidale delle caratteristiche** delle immagini risolve i problemi dati da immagini di risoluzioni diverse



Obiettivo: Mantenere forti le features nonostante diverse risoluzioni o dimensioni delle immagini

L'integrazione avviene dopo il livello di pooling



STRUTTURA PIRAMIDALE

Risultati da 120.000 campioni ca.

Prima fase di apprendimento

Efficienza

senza piramide **87.2% \pm 2.5%**

con piramide **92% \pm 1.6%.**

Seconda fase di apprendimento

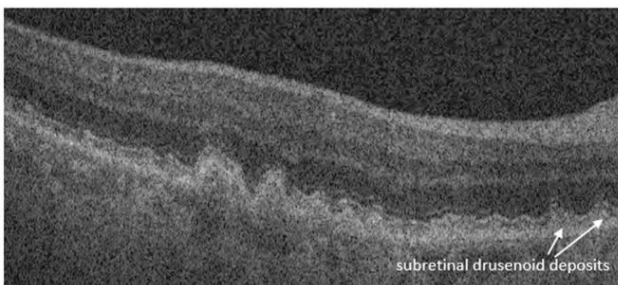
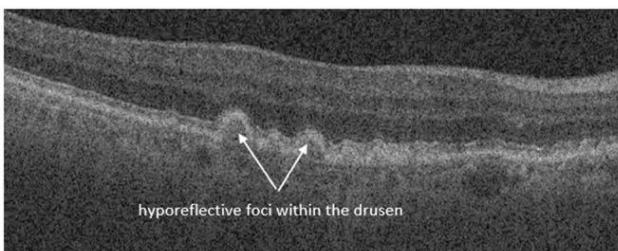
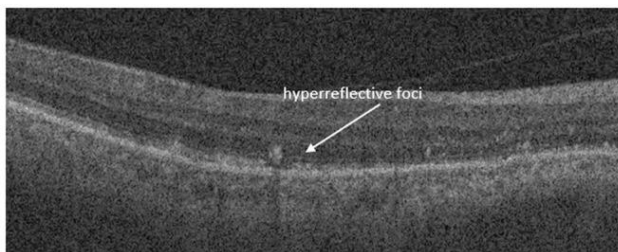
Efficienza

senza piramide **92.0% \pm 1.6%**

con piramide **93.4% \pm 1.4%.**



3) BIOMARCATORI



Questo metodo si applica ricercando sulle immagini specifici **biomarcatori** (indicatori di una possibile predisposizione alla malattia)

Su 20.000 campioni ca.

Efficienza **87%**

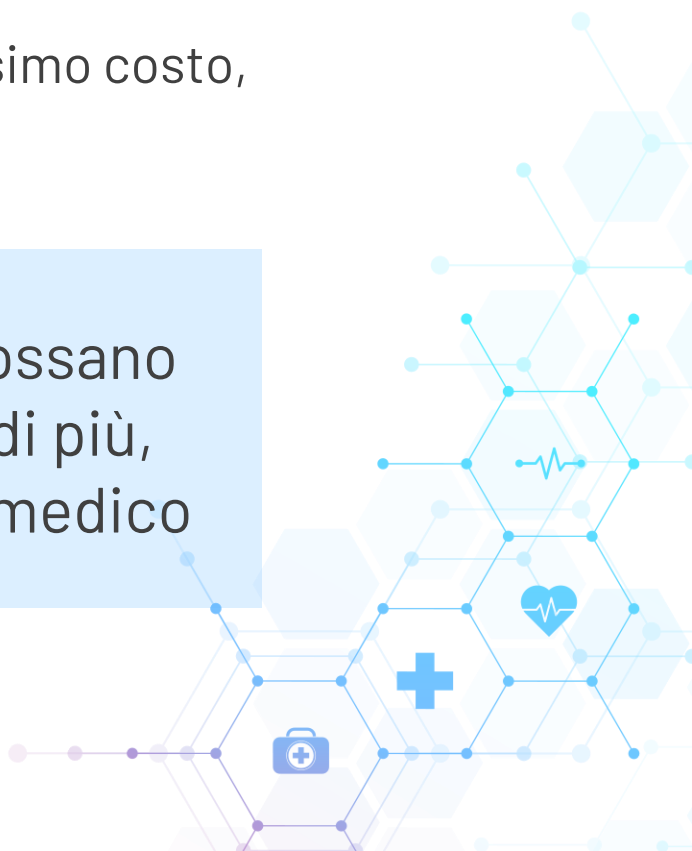


CONCLUSIONI

Un dataset ridotto e un hardware altrettanto limitato non consentono di raggiungere le prestazioni di algoritmi più complessi e forniti.

Tuttavia, è possibile ottenere un'iniziale classificazione e screening velocemente e a bassissimo costo, ottenendo comunque un'efficienza del 65%.

Ciò dimostra come tecnologie anche semplici possano portare a **risultati** molto validi e come, sempre di più, esse siano una preziosa risorsa anche in ambito medico





**GRAZIE PER
L'ATTENZIONE**

