



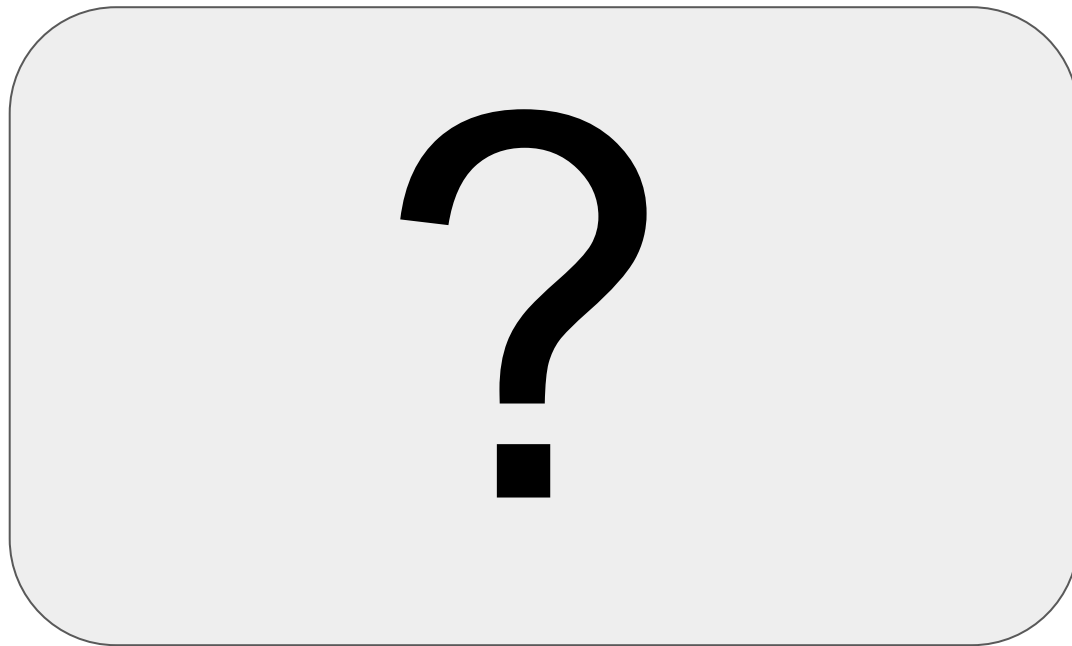
Alligator

By Valerio Conti & Guramrit Singh



Alligatore o coccodrillo?

Immagine misteriosa

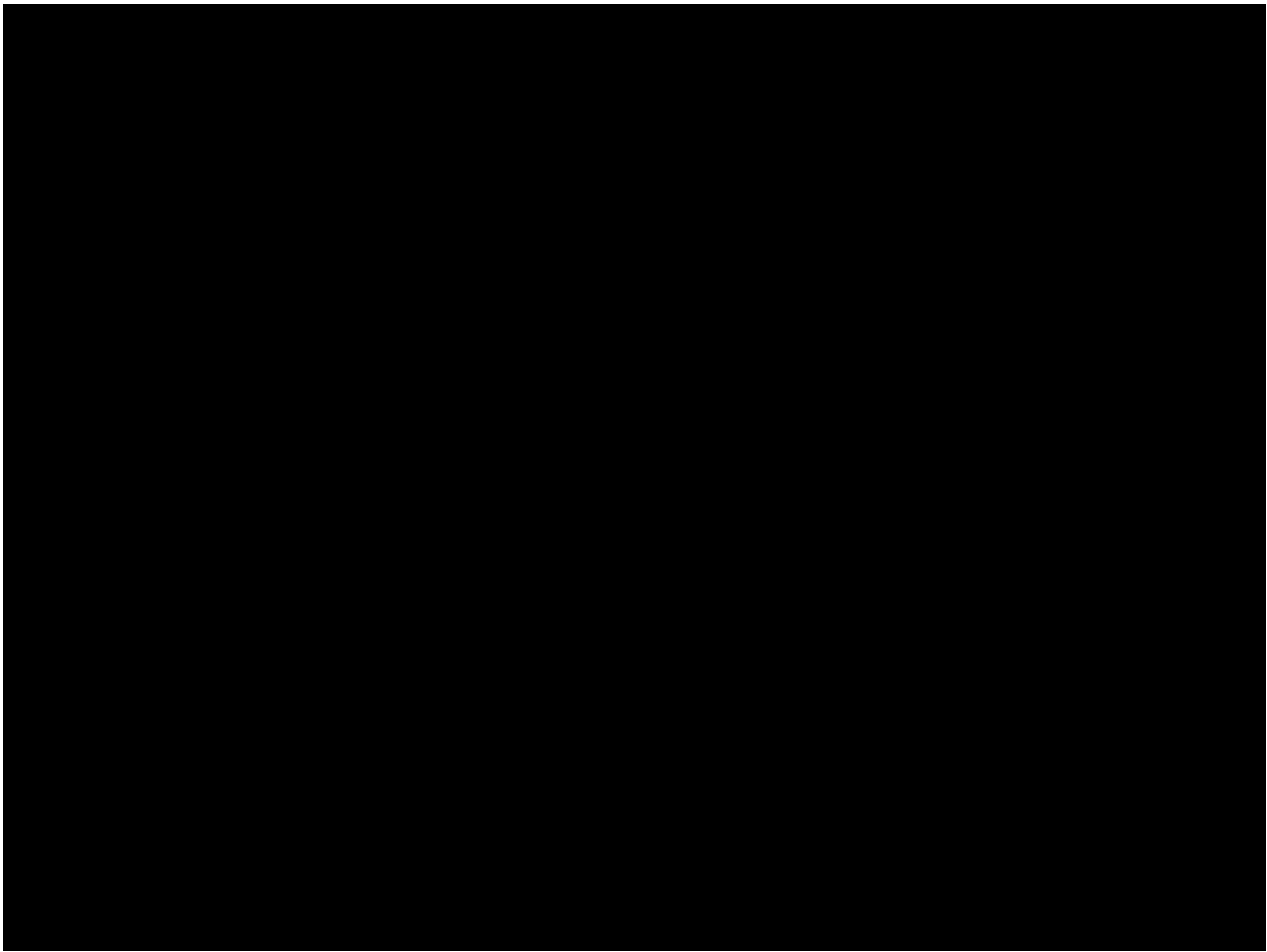


A person in a dark hoodie and a highly detailed, metallic robot stand side-by-side in a dimly lit, industrial setting. The person is on the left, looking slightly to the right. The robot is on the right, facing forward. The scene is bathed in a cool, blue light, creating a futuristic and somewhat somber atmosphere. The robot's head is large and complex, with various sensors and mechanical components visible. The person's hoodie is dark and textured, with the hood pulled up over their head.

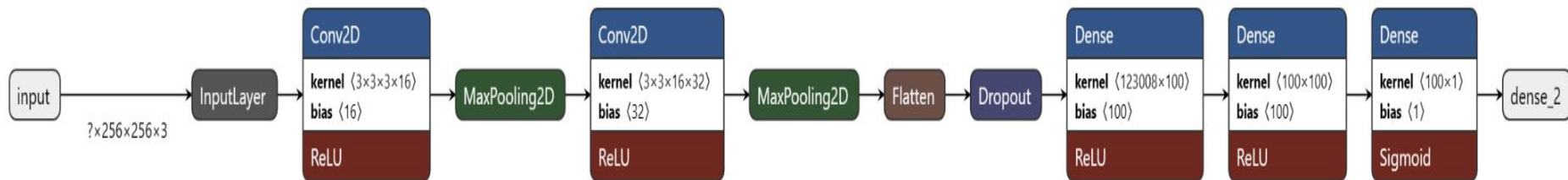
Amico o IA



Rete Neurale

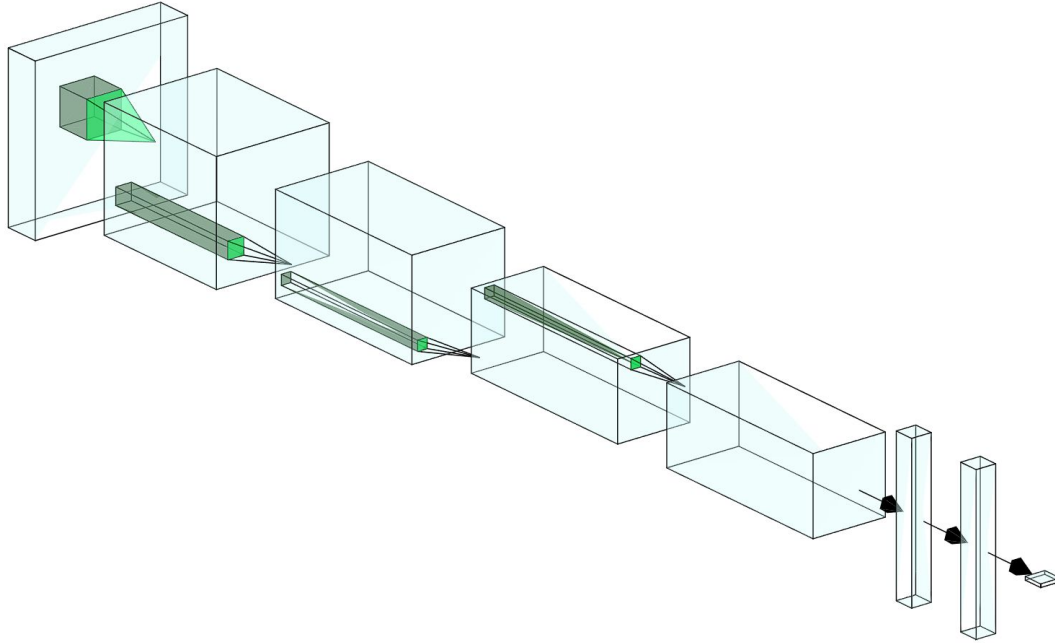


Uno sguardo nel dettaglio



... ma magari così è un po' troppo.

Che cos'è una rete convoluzionale?



Una **Convolutional Neural Network** è un particolare tipo di rete neurale specializzato nel riconoscimento di pattern.

Come riesce a riconoscere dei pattern?



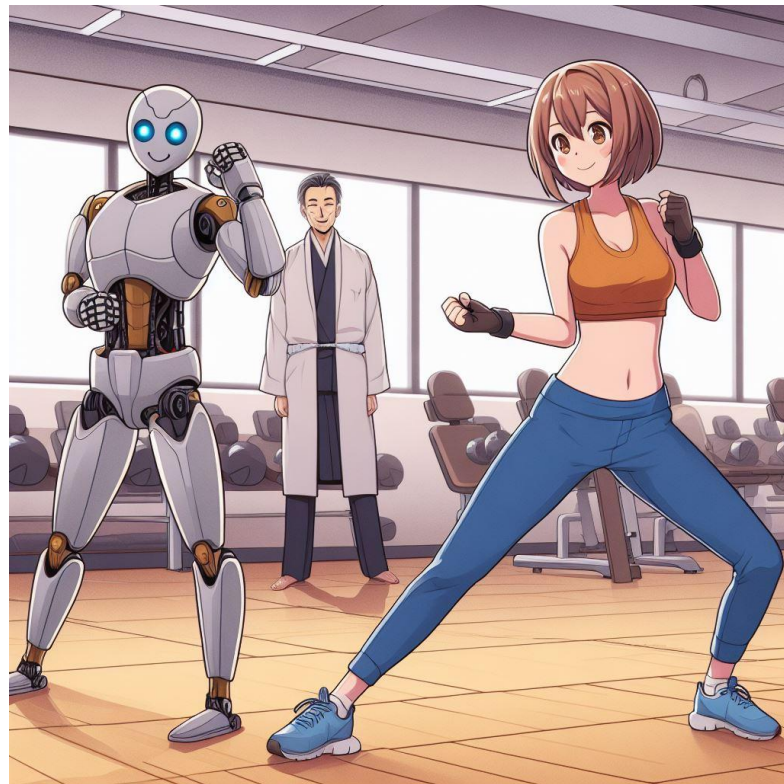
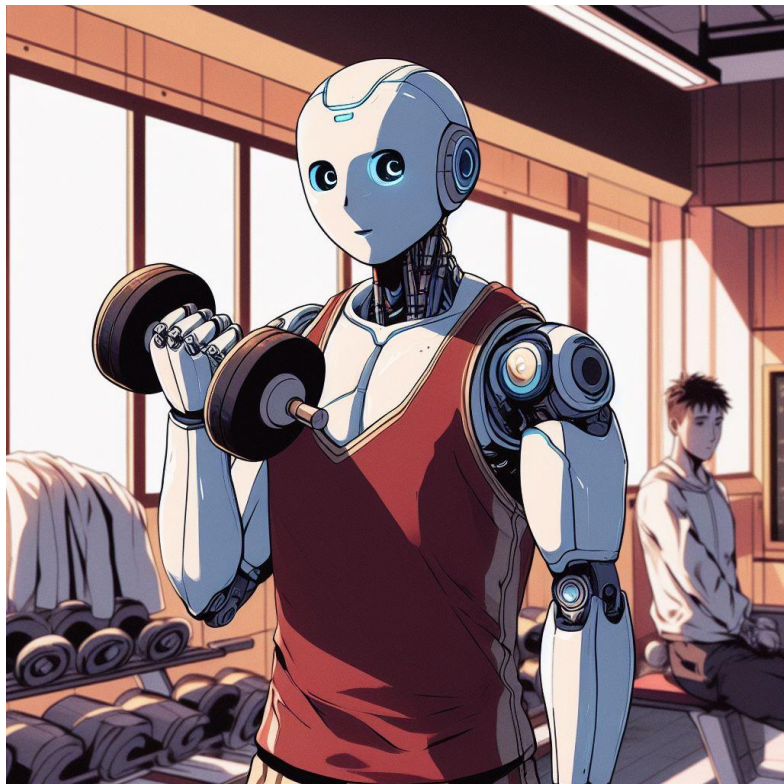
-1.05	-1.05	-1.05
-1.05	8.1	-1.05
-1.05	-1.05	-1.05



Ma in realtà...

$$\sum_{i=0}^n (in_i * k_i)$$

Momento del training arc



File Edit Selection View Go Run Terminal Help

EXPLORER

- AI-LLIGATOR
 - dataset
 - testing
 - Alligator.jpg
 - Crocodile.jpg
 - FinalTest.jpg
 - training
 - validation
 - .gitignore
 - best_model_64.h5
 - best_model.h5
 - dataset_scort.zip
 - first_model.h5
 - LICENSE
 - ListGPU.py
 - Prediction.py
 - README.md
 - requirements.txt
 - TrainingArc.py

TrainingArc.py > ...

```
51 # decay_steps=10000,
52 # decay_rate=0.9)
53
54 # Impostazione dell'ottimizzatore
55 #opt = keras.optimizers.Adam(learning_rate=lr_schedule)
56
57 # Compilo il modello specificando la funzione di perdita (loss), l'ottimizzatore e la metrica da monitorare
58 model.compile(
59     loss='binary_crossentropy', # La funzione di perdita per la classificazione binaria
60     optimizer='adam', # L'ottimizzatore Adam che adatta il tasso di apprendimento in base al gradiente
61     metrics=['accuracy'] # La metrica da monitorare è l'accuratezza della classificazione
62 )
63
64 # Creo una callback che salva il modello migliore in base alla metrica di validazione
65 checkpoint = keras.callbacks.ModelCheckpoint(
66     filepath='best_model.h5', # Il nome del file in cui salvare il modello
67     monitor='val_accuracy', # La metrica da monitorare è l'accuratezza di validazione
68     save_best_only=True, # Salvo solo il modello che ha il valore migliore della metrica
69     mode='max', # Il valore migliore della metrica è il massimo
70     verbose=1 # Mostro un messaggio quando salvo il modello
71 )
72
73 # Addestro il modello usando i generatori di immagini e la callback
74 with tf.device('/GPU:0'):
75     model.fit(
76         train_generator, # Il generatore di immagini di training
77         epochs=10, # Il numero di epoche da eseguire
78         validation_data=validation_generator, # Il generatore di immagini di validazione
79         callbacks=[checkpoint] # La lista delle callback da usare
80     )
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Remus\Documents\AI-lligator>

powerShell
Python
Python
Python

OUTLINE
TIMELINE

Ln 77, Col 18 Spaces: 4 UTF-8 LF Python 3.9.18 (venv, conda: conda)

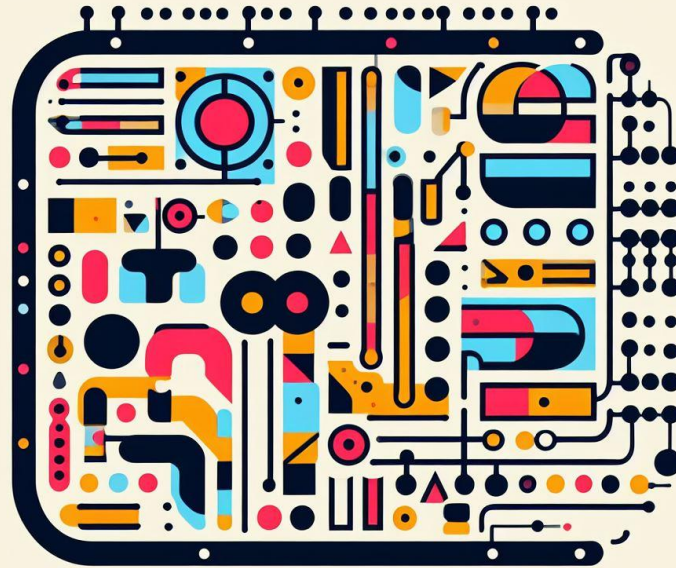
Binary Crossentropy - la nostra funzione di perdita

$$L(y,p) = -[y \log(p) + (1-y) \log(1-p)]$$



Antagoniste

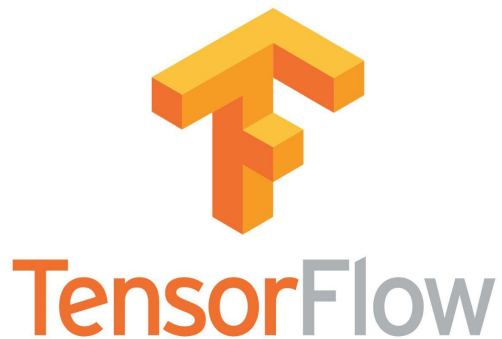
Dataset



TRAINING DATA FOR AI

Codice

Librerie scelte



Modello

```
model = keras.Sequential([  
    layers.Conv2D(16, (3, 3), activation='relu', input_shape=(256, 256, 3)), # Layer convoluzionale  
    layers.MaxPooling2D((2, 2)), # Layer di pooling  
    layers.Conv2D(32, (3, 3), activation='relu'), # Un altro layer convoluzionale  
    layers.MaxPooling2D((2, 2)), # Un altro layer di pooling  
    layers.Flatten(), # Un layer che appiattisce le feature map in un vettore unidimensionale  
    layers.Dropout(0.5), # Layer di dropout  
    layers.Dense(100, activation='relu'), # Primo hidden layer  
    layers.Dense(100, activation='relu'), # Secondo hidden layer  
    layers.Dense(1, activation='sigmoid') # Layer di output  
])
```


Callback

```
# Creo una callback che salva il modello migliore in base alla metrica di validazione  
checkpoint = keras.callbacks.ModelCheckpoint(  
    filepath='best_model.h5', # Il nome del file in cui salvare il modello  
    monitor='val_accuracy', # La metrica da monitorare è l'accuratezza di validazione  
    save_best_only=True, # Salvo solo il modello che ha il valore migliore della metrica  
    mode='max', # Il valore migliore della metrica è il massimo  
    verbose=1 # Mostro un messaggio quando salvo il modello  
)
```

Training

```
# Addestro il modello
```

```
with tf.device('/GPU:0'):
```

```
    model.fit(
```

```
        train_generator, # Il generatore di immagini di training
```

```
        epochs = 100, # Il numero di epoche da eseguire
```

```
        validation_data = validation_generator, # Il generatore di immagini  
di validazione
```

```
        callbacks = [checkpoint] # La lista delle callback da usare
```

```
    )
```

Ma quindi...
chi ha ragione?

Momento del test finale




File Edit Selection View ... AI-lligator

EXPLORER

- AI-LLIGATOR
 - dataset
 - testing
 - Alligator.jpg
 - Crocodile.jpg
 - FinalTest.jpg
 - training
 - validation
 - .gitignore
 - best_model_64.h5
 - best_model.h5
 - dataset_scorta.zip
 - first_model.h5
 - LICENSE
 - ListGPU.py
 - Prediction.py M
 - README.md
 - requirements.txt
 - TrainingArc.py M
- OUTLINE
- TIMELINE

dataset > testing > FinalTest.jpg



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Remus\Documents\AI-lligator> & C:/Users/Remus/anaconda3/envs/venv_conda/python.exe  
e c:/Users/Remus/Documents/AI-lligator/Prediction.py dataset\testing\FinalTest.jpg
```

main* 0:51 0 0 0 0 18% 1024x1024 176.12KB

Grazie per l'attenzione, buon Linux Day 2023!



Un ringraziamento speciale a **Marco Guerriero**, collega che ci ha assistito nella realizzazione del progetto <3.



GitHub