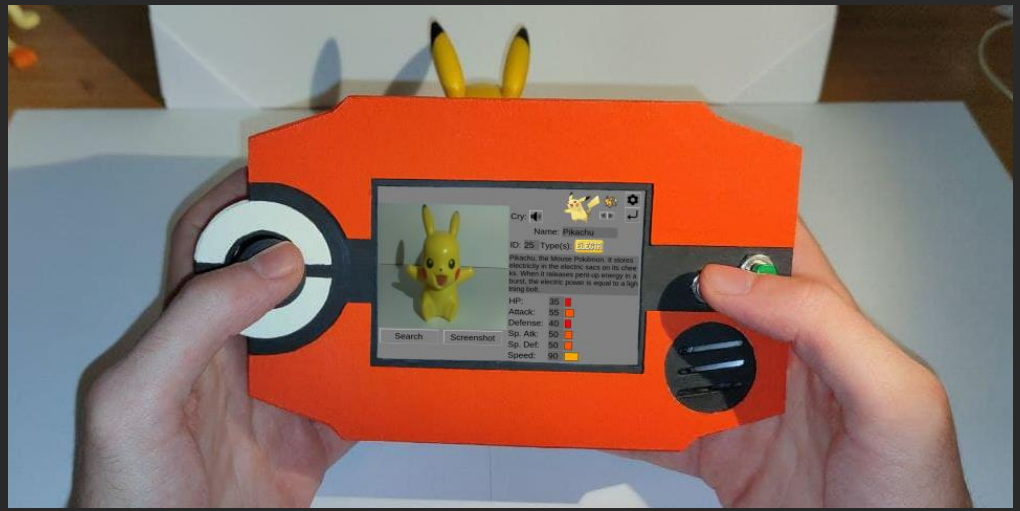
Poké-Pi-Dex

Classificazione di Pokémon tramite l'uso di CNN su dispositivi embedded



Sistemi Digitali M | karina-chichifoi michele-righi TryKatChup mikyll

Indice

Introduzione

Pokémon: Panoramica

Videogioco, gioco carte collezionabili, cartone animato, fumetto

Suddivisi in **tipi**, i.e. ELETT FUOCO ERBA

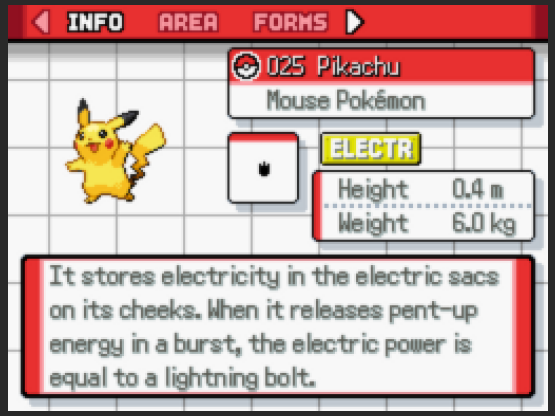
Pocket Monsters: possono essere catturati, allenati e usati in combattimento

Possiedono valori specifici (**statistiche**)

Pokédex

Dispositivo che **tiene traccia** dei Pokémon visti e catturati

Scansiona e **riconosce** il Pokémon catturato o visto in battaglia



Registra le **caratteristiche** del Pokémon catturato

Mostra **statistiche** Pokémon

Obiettivi del Progetto

Realizzazione di un dispositivo embedded simile a un **Pokédex** e **a portata di mano**

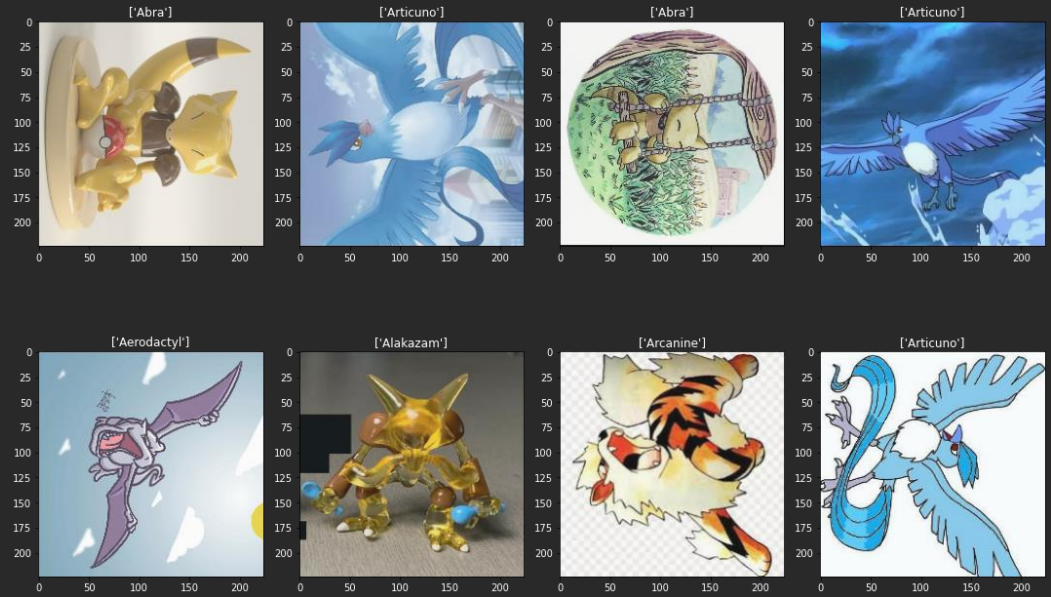
Riconoscimento di **carte**, **peluche**, **immagini** della serie animata, **modelli** 3D, pixel art

01 Classificatore

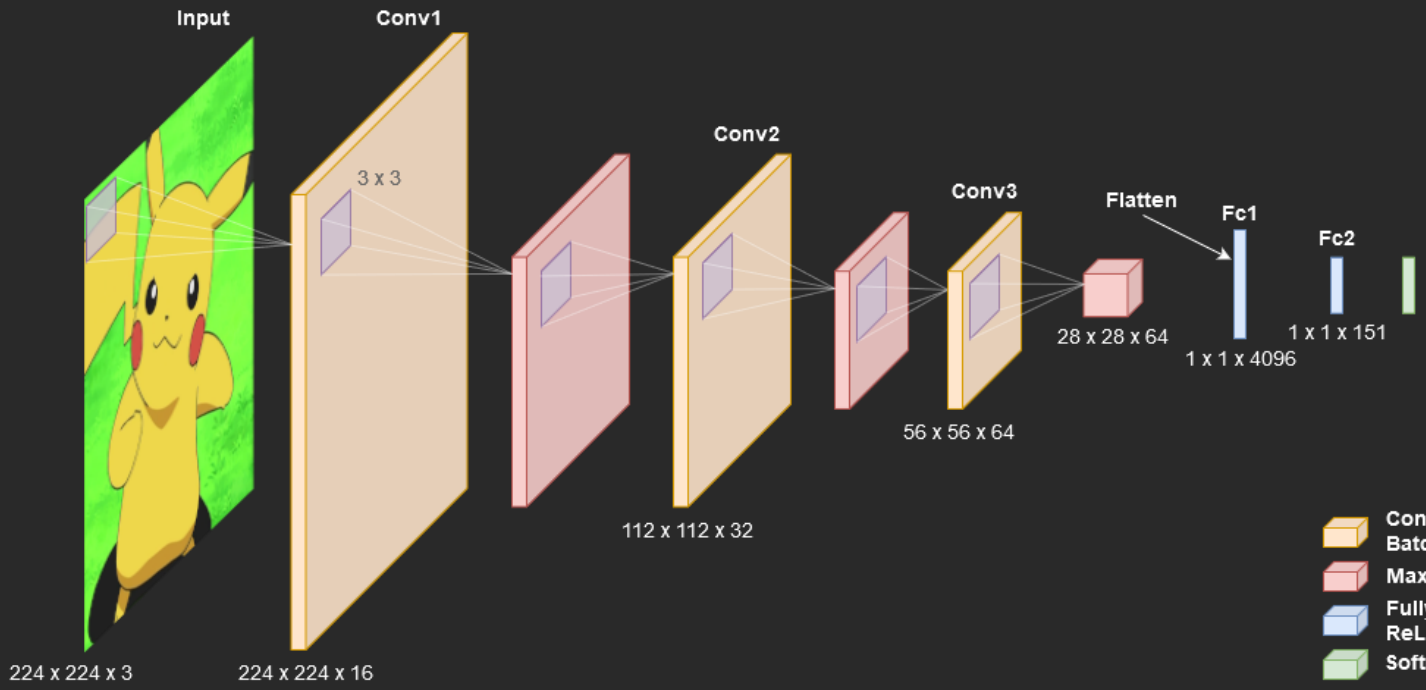
Preprocessing dei Dati

Dataset di **7000 immagini** trovato su Kaggle e opportunamente diviso

Sono state effettuate operazioni di **Data Augmentation**



Architettura della Rete



Allenamento del Modello

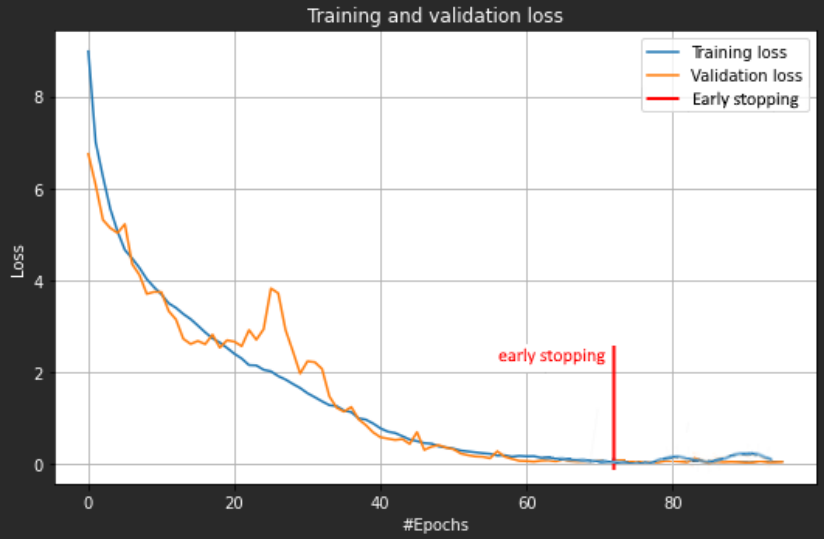
Framework **Tensorflow 2.6** + Keras (integrato)

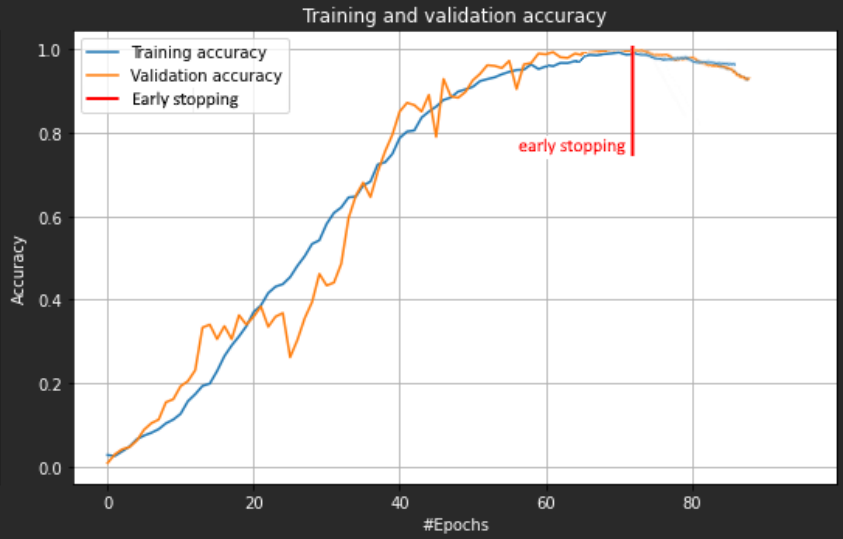
Ottimizzatore: **Adam**

Early stopping: necessario per fermare l'allenamento in caso di peggioramento di performance del modello

Training con **epochs = 100** e **batch size = 64**

Performance Primo Modello





Loss: 0.0176
Accuracy: 0.9983
Numero di epoche: 75

Osservazioni e Conseguenze

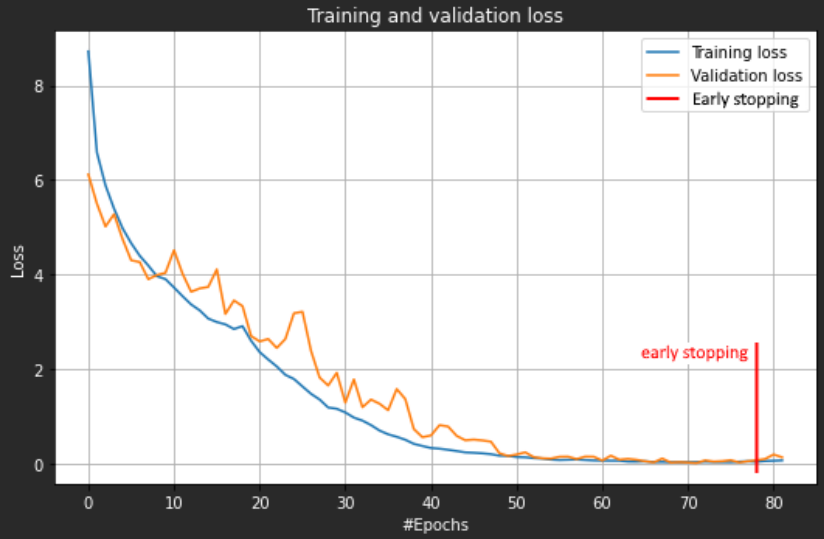
Abbiamo visto che le **performance** sono molto **soddisfacenti**

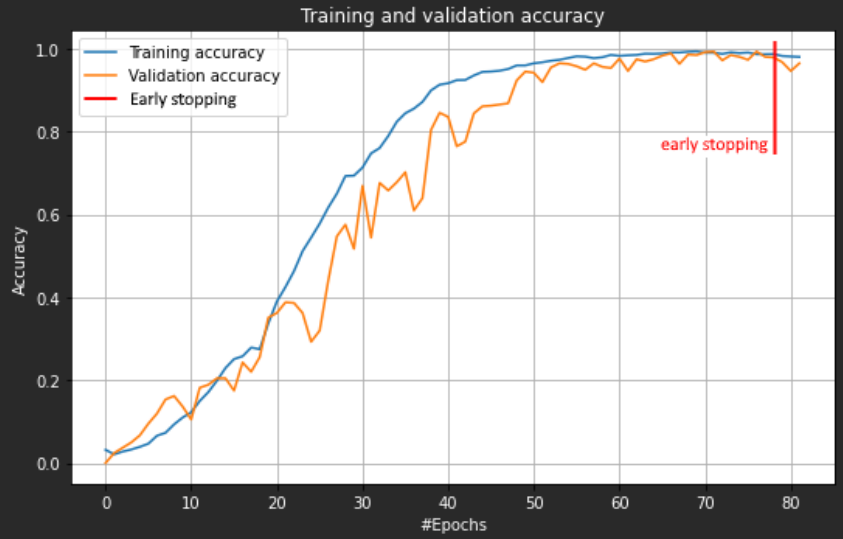
Provando sul dispositivo abbiamo visto che **non riconosceva molti Pokémon**

Dataset di qualità scarsa, con numerosi errori

Abbiamo ricreato un **nuovo dataset da 11945** immagini, di cui soltanto 8238 sono state usate

Performance Secondo Modello





Loss: 0.0262
Accuracy: 0.9933
Numero di epoche: 72

Conversione in TFLite

Il modello ha un peso di 1.2 GB. Per questi motivi:

È stato esportato in **Tensorflow Lite** per limiti di memoria e di spazio su sistemi embedded

È stato compresso tramite tecniche di **quantizzazione**

Peggioramento di accuracy pari all'**1%** rispetto al modello originale

Peso finale: **93 MB**

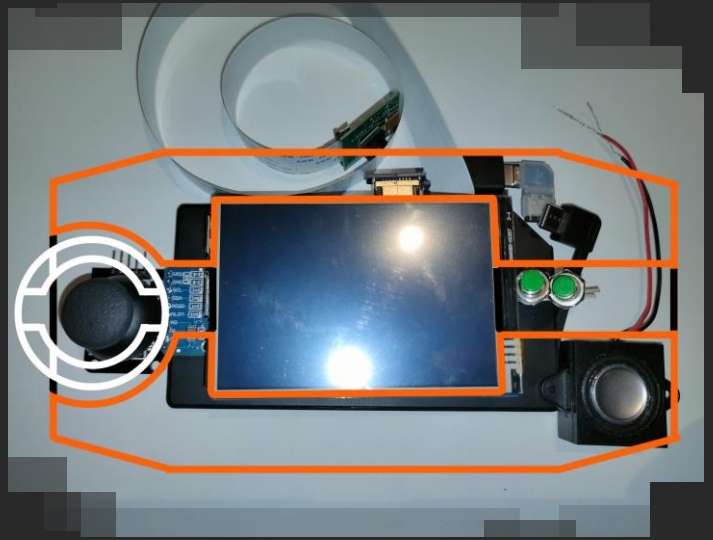
02 Architettura

Calcolatore e Componenti

- **Raspberry Pi4** Model B
- Powerbank ultrasottile con capacità 5000mAh e uscita 5V/2A
- **Display LCD 3.5”** HDMI con resistive touch screen
- **Pi Camera** Rev 1.3 (5MP, 1080p)
- Mini Speaker
- Push button e joystick analogico

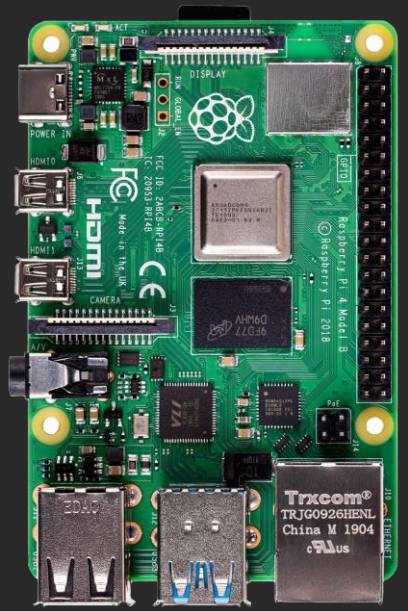
Calcolatore e Componenti

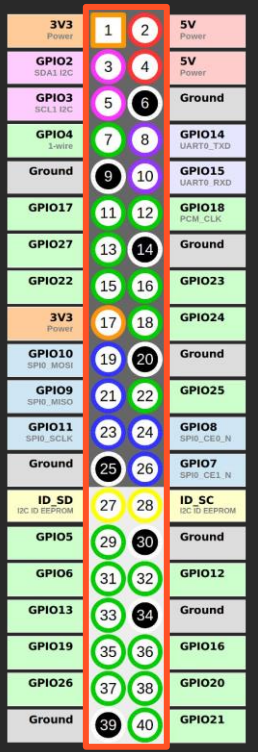
- **Raspberry Pi4** Model B
- Powerbank ultrasottile con capacità 5000mAh e uscita 5V/2A
- **Display LCD 3.5”** HDMI con resistive touch screen
- **Pi Camera** Rev 1.3 (5MP, 1080p)
- Mini Speaker
- Push button e joystick analogico
- **Case**



Installazione Componenti

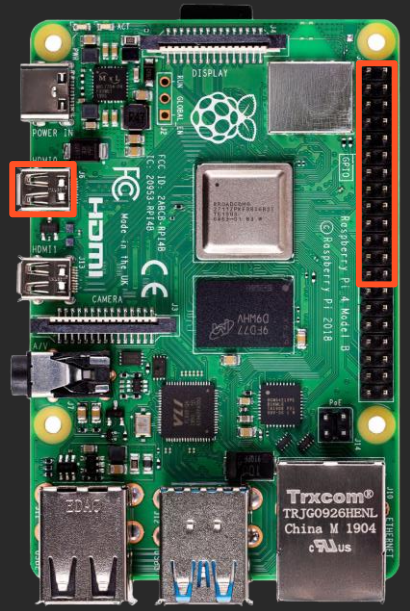
40 GPIO disponibili:

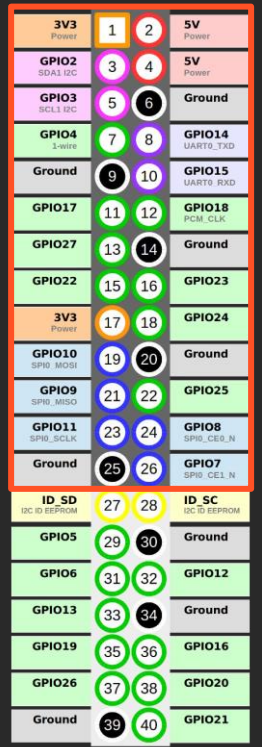




Installazione Componenti

- **40 GPIO** disponibili:
  - Display LCD,





Installazione Componenti

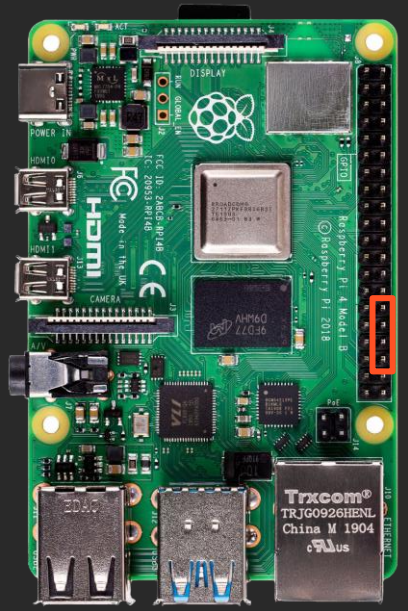
40 GPIO disponibili:

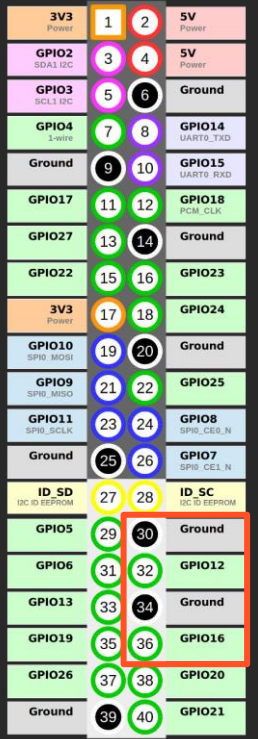
- Display LCD,
- Pi Camera

Installazione Componenti

40 GPIO disponibili:

- Display LCD,
- Pi Camera
- Push button

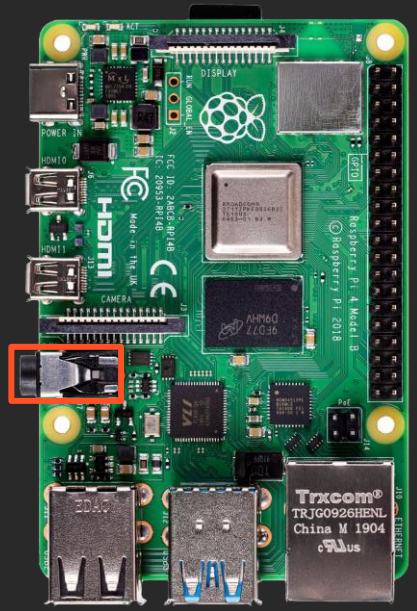




Installazione Componenti

40 GPIO disponibili:

- Display LCD,
- Pi Camera
- Push button
- Speaker



Case e Prototipo

Poké-Pi-Dex:

- Ispirato al Pokédex di **terza generazione** (Hoenn)
- Cartoncino **riciclato** da quaderni quablock finiti

Realizzazione:

Case e Prototipo

Poké-Pi-Dex:

- Ispirato al Pokédex di **terza generazione** (Hoenn)
- Cartoncino **riciclato** da quaderni quablock finiti

Realizzazione:

- Suddivisione in 3 parti: **Base**, **Sezione Intermedia**, **Coperchio**

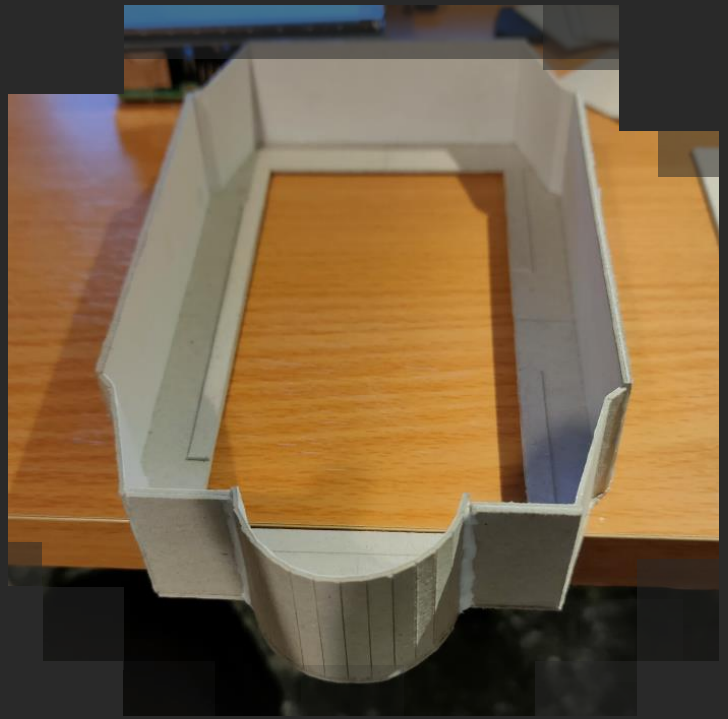
Case e Prototipo

Poké-Pi-Dex:

- Ispirato al Pokédex di **terza generazione** (Hoenn)
- Cartoncino **riciclato** da quaderni quablock finiti

Realizzazione:

- Suddivisione in 3 parti: **Base**, **Sezione Intermedia**, **Coperchio**



Case e Prototipo

Poké-Pi-Dex:

- Ispirato al Pokédex di **terza generazione** (Hoenn)
- Cartoncino **riciclato** da quaderni quablock finiti

Realizzazione:

- Suddivisione in 3 parti: **Base**, **Sezione Intermedia**, **Coperchio**

Case e Prototipo

Poké-Pi-Dex:

- Ispirato al Pokédex di **terza generazione** (Hoenn)
- Cartoncino **riciclato** da quaderni quablock finiti

Realizzazione:

- Suddivisione in 3 parti: **Base**, **Sezione Intermedia**, **Coperchio**
- Verniciatura con colori **acrilici**

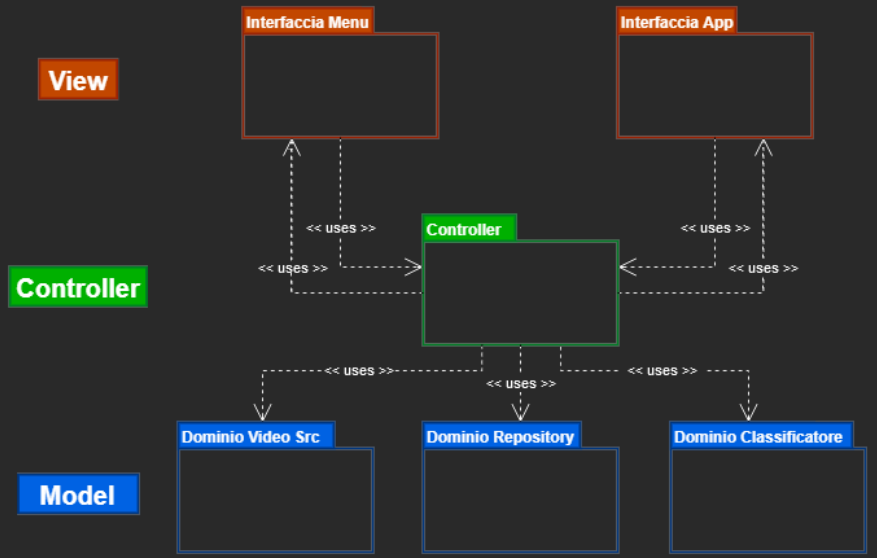
03 Applicazione

Architettura Logica

- **Python** 3.x
- Pattern **MVC**

Principali package utilizzati:

- **Tkinter**
- **OpenCV**
- Pillow
- Pygame
- Gpiozero
- Numpy
- Sklearn
- **Tf-lite_runtime**



Interfacce Grafiche

Interfaccia Menu:

- **Vista Menu**

Interfaccia Pokédex:

Interfacce Grafiche

Interfaccia Menu:
- **Vista Menu**
- Vista Informazioni

Interfaccia Pokédex:

Interfacce Grafiche

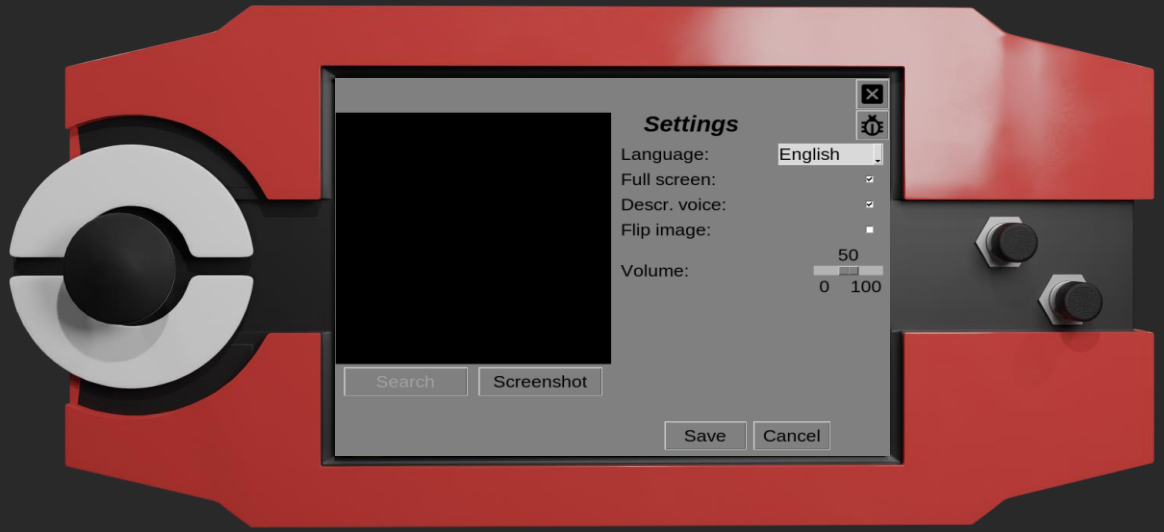
Interfaccia Menu:
- **Vista Menu**
- Vista Informazioni

Interfaccia Pokédex:
- Vista Video
- **Vista Dettagli**

Interfacce Grafiche

Interfaccia Menu:
- **Vista Menu**
- Vista Informazioni

Interfaccia Pokédex:
- Vista Video
- **Vista Dettagli**
- Vista Impostazioni



Interfacce Grafiche

Interfaccia Menu:

- **Vista Menu**
- Vista Informazioni

Interfaccia Pokédex:

- Vista Video
- **Vista Dettagli**
- Vista Impostazioni
- Vista Debug

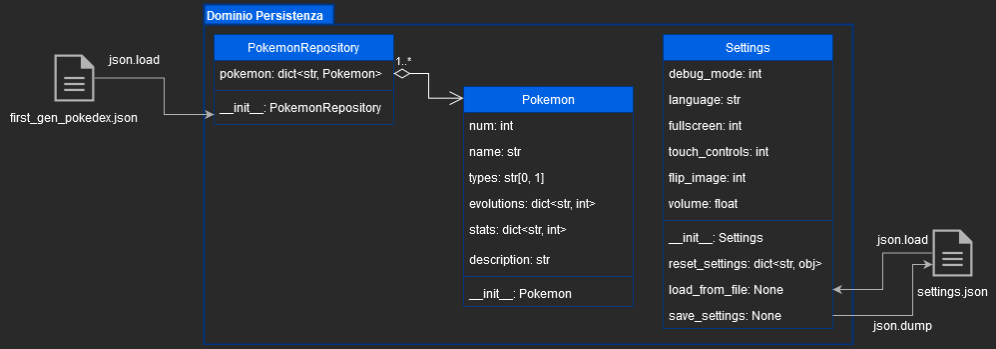
Persistenza

Dati Pokémon:

- File **JSON** per i dettagli (nome, id, tipi, descrizione, evoluzioni, statistiche)
- Immagini di anteprima, sprite, file audio per verso e descrizioni

Impostazioni:

- Classe ausiliaria
- JSON per renderle persistenti
- Scelta **lingua**



Classificatore nell'App

Utilizzo dell'**interprete** di **TFLite** per fare inferenza e ottenere nuove predizioni

Installazione dell'interprete separata rispetto al framework Tensorflow:
- **Risparmio risorse**
- **Efficienza** app

Distorsione Immagini

Tipi di distorsione:

- **Radiale**

Parametri:

Radiale:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Distorsione Immagini

Tipi di distorsione:

- **Radiale**
- **Tangenziale**

Parametri:

Radiale:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Tangenziale:

$$x_{distorted} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

Distorsione Immagini

Tipi di distorsione:

- **Radiale**
- **Tangenziale**

Parametri:

- **Coefficienti** di **Distorsione**

Radiale:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Tangenziale:

$$x_{distorted} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

$$Coefficienti\ di\ distorsione = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

Distorsione Immagini

Tipi di distorsione:
- **Radiale**
- **Tangenziale**

Parametri:
- **Coefficienti di Distorsione**
- **Matrice della Fotocamera**

Radiale:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Tangenziale:

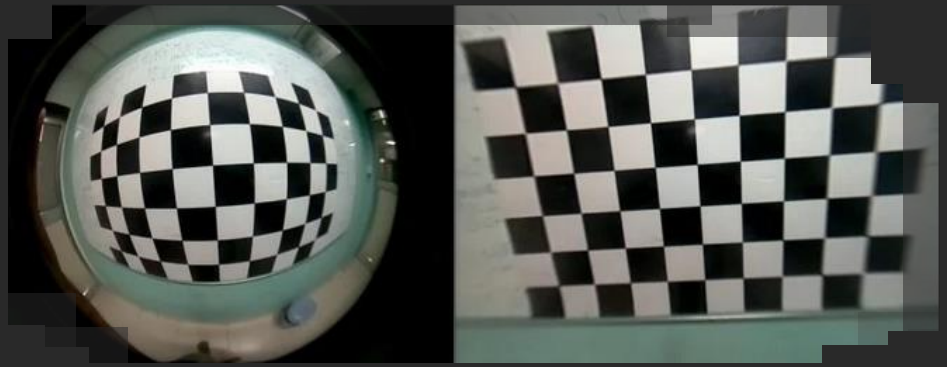
$$x_{distorted} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

$$Coefficienti\ di\ distorsione = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

$$Matrice\ della\ fotocamera = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Calibrazione Fotocamera

Calibrazione fotocamera e **rettificazione** immagini con OpenCV, utilizzando una scacchiera



Calibrazione Fotocamera

Calibrazione fotocamera e **rettificazione** immagini con OpenCV, utilizzando una scacchiera

Calibrazione Fotocamera

Calibrazione fotocamera e **rettificazione** immagini con OpenCV, utilizzando una scacchiera

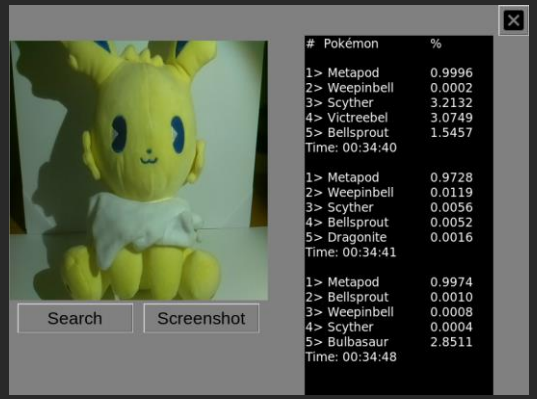
$$C_{dist} = (\,0.2876 \quad -1.7175 \quad -0.0102 \quad 0.0038 \quad 2\;3.066)$$

$$Matrice\ della\ fotocamera = \begin{bmatrix} 630.72 & 0 & 312.61 \\ 0 & 630.54 & 226.20 \\ 0 & 0 & 1 \end{bmatrix}$$

04 Conclusioni

Conclusioni

Riconoscimento in circa **200ms**



85% accuratezza della predizione in casi reali

Predizione **semi-errata** per **Pokémon molto simili** o con sfondi fraintendibili

È necessario **illuminare sufficientemente** la zona inquadrata

https://github.com/TryKatChup/Poke-Pi-Dex