

Progetto: Deep Face Detection

Materie: Machine Learning , Sistemi Intelligenti per Internet

Link al progetto: <https://github.com/paolosilv/ML-SII>

Studente: Paolo Silvestri
Matricola: 521343



Obiettivo del progetto

Lo scopo del progetto consiste nell'ottenere delle immagini in input (da webcam) e riuscire, all'interno di un contesto di video streaming live, a etichettare il volto del soggetto

Breve descrizione del progetto

- Prendere in input (da webcam) delle immagini
- Utilizzo di librerie per etichettare le immagini (il viso)
- Divisione dei dati in train, validation, test
- Data Augmentation per ottenere dati aggiuntivi
- Combinazione di labels e immagini con creazione dei dataset finali
- Creazione del modello
- Definizione della loss e dell'optimizer
- Training, Testing (predictions), Real Time detection

Principali Librerie utilizzate

- Tensorflow e Tensorflow-gpu → per sfruttare modelli, layers e gpu
- Labelme → per etichettare le immagini
- OpenCV-python → per utilizzare la Computer Vision
- Matplotlib → per i grafici
- Albumentations → per sfruttare Data Augmentation
- Numpy → per gestire array e funzioni matematiche

Sviluppo #1

- **Acquisizione delle immagini da webcam**
- **Numero di immagini «reali» utilizzate: 210**
- **Annotazione delle immagini attraverso la libreria «labelme»**
- **L'annotazione consiste nel disegnare un rettangolo intorno al viso**
- **Split manuale dei dati, 210 immagini totali: 147 (70%) per training, 32 (15%) per test e 31 (15%) per validation.**
- **Definizione della pipeline per realizzare data augmentation (sviluppo di 90 immagini a partire da 1 immagine reale)**
- **Compressione delle immagini in 120x120**
- **Combinazione immagini-labels**

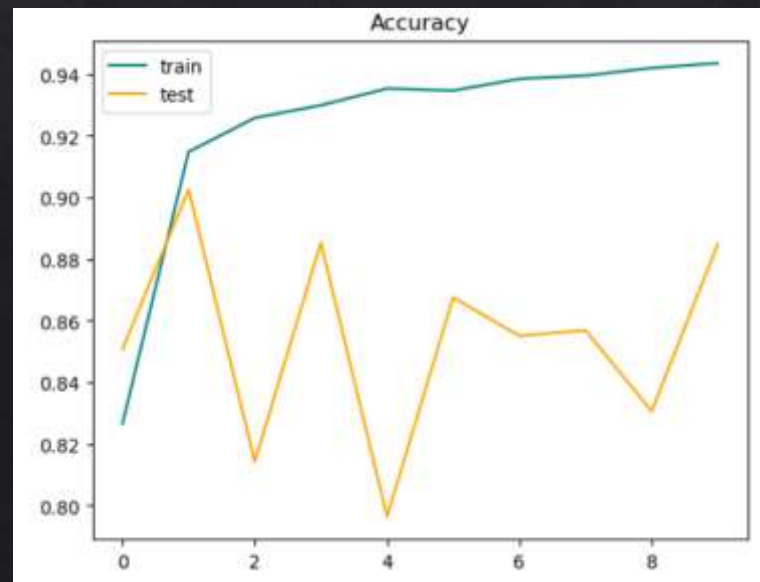
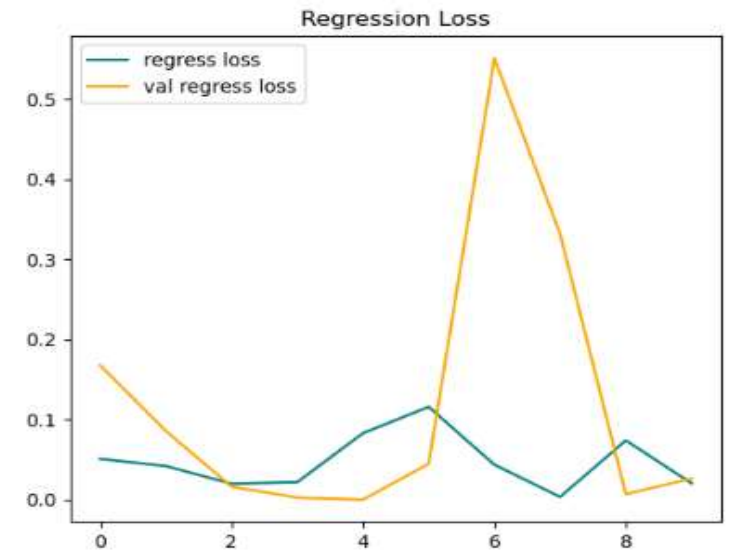
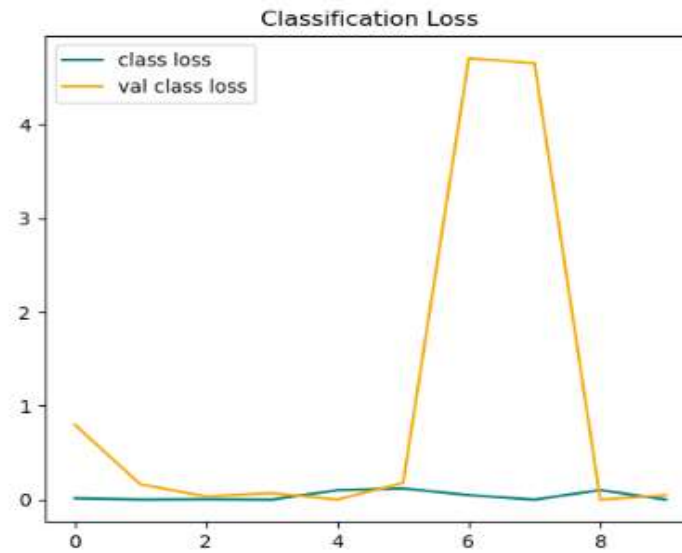
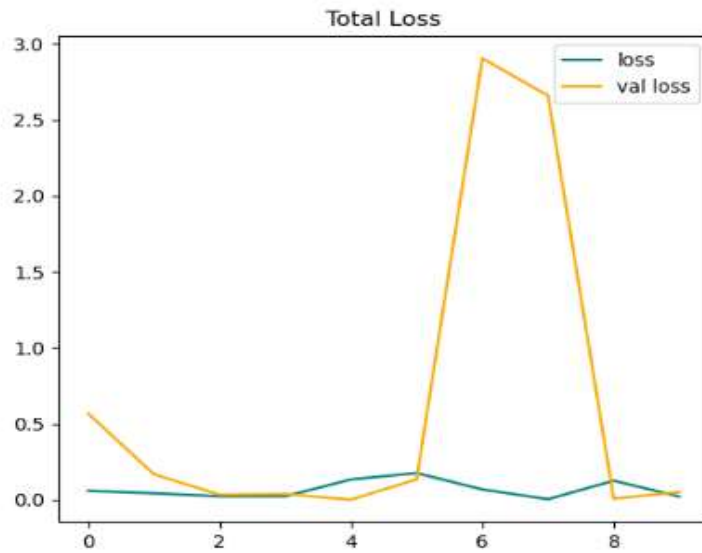
Sviluppo #2

- Creazione dei dataset finali
- Import del modello VGG16 che verrà utilizzato nel modello principale
- Creazione del modello (16.8 milioni di parametri)
- Definizione delle losses (localization e classification loss)
- Definizione dell'optimizer (Adam)
- Training della rete neurale (10 epoche) → circa 3240 secondi (324 secondi/epoca)
- Predizioni sul test set
- Salvataggio del modello
- Real Time detection

Problemi riscontrati

- Sviluppato il training utilizzando solo la CPU, ma era troppo lento
- Grazie alla GPU risparmiati circa 1000 secondi/epoca
- La `validation_classification_loss` (per la label) e la `validation_regression_loss` (per i 4 punti del rettangolo) presentano un andamento non lineare, ma spigoloso.

Risultati ottenuti



Conclusione

**Il progetto è stato molto stimolante, ho
imparato il funzionamento di diverse
tecnologie e librerie che non conoscevo**

Grazie per l'attenzione