Immagine che contiene testo, Carattere, Elementi grafici, logo

Descrizione generata automaticamente

UNIVERSITÀ DEGLI STUDI ROMA TRE

Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

TECNICHE DI MACHINE LEARNING E DEEP LEARNING PER IL RICONOSCIMENTO DELLA DISGRAFIA

Francesco Renzi 547002

Umberto Gentile 547999

Anno Accademico 2023/2024

1. **Introduzione**

La disgrafia è un disturbo dell’apprendimento che si manifesta come difficoltà nella scrittura coerente e leggibile. Questo disturbo può avere un impatto significativo sulla vita di un individuo, influenzando la sua capacità di esprimersi chiaramente attraverso la scrittura. Nonostante la prevalenza di questo problema, la diagnosi della disgrafia rimane una sfida, poiché le tradizionali metodologie spesso richiedono osservazioni manuali e valutazioni da parte di esperti umani, risultando un processo laborioso e soggetto a variazioni interpersonali.

In questo progetto affrontiamo la questione del riconoscimento della disgrafia, studiando alcune tecniche di machine learning e deep learning che possano migliorare l’accuratezza della diagnosi e rendere il processo più efficiente e accessibile. L’obiettivo è quello di allenare dei modelli in maniera tale che, a partire da delle immagini di testo scritto a mano, possano riconoscere la presenza o meno del disturbo. Per l’implementazione di tali modelli, abbiamo utilizzato l’ambiente di sviluppo **Google Colab** e le librerie python: **sklearn**, **tensorflow** e **keras**.

Il lavoro si divide in più fasi:

* Raccolta ed analisi dei dati
* Ampliamento del dataset con tecniche di data augmentation
* Estrazione delle caratteristiche delle immagini con diverse architetture
* Classificazione delle immagini (con e senza augmentation) con tecniche di machine learning

1. **Raccolta dei dati**

Abbiamo considerato due diversi dataset con immagini di testo scritto a mano (**369** immagini in totale)

Il primo comprende 120 immagini, sia di soggetti a cui è stata diagnosticata la disgrafia (57), sia di soggetti che non presentano il disturbo (63), tutte della stessa dimensione. In questo caso, alle immagini è correlata una tabella con i dati anagrafici degli individui a cui è stato chiesto di scrivere il testo.

Il secondo, invece, comprende 249 immagini. Analogamente al primo, abbiamo una suddivisione tra soggetti con disgrafia (114) e soggetti senza (135), ma la dimensione delle immagini è molto più eterogenea e non abbiamo una tabella con i dati anagrafici degli utenti che hanno scritto i testi.

1. **Ampliamento del dataset**

Come visto in precedenza, i dati che abbiamo raccolto sono stati etichettati, ma a causa del numero limitato di campioni per i modelli, abbiamo pensato di impiegare delle tecniche di data augmentation. In questo modo abbiamo potuto aumentare la quantità e la varietà dei dati, nonché testare la variazione delle prestazioni delle architetture.

I metodi utilizzati sono stati:

* Ritaglio
* Capovolgimento orizzontale
* Rotazione
* Traslazione
* Aggiunta di rumore Gaussiano

Ognuna di queste funzioni crea tante immagini quante sono quelle originali (ad eccezione del ritaglio, che ne crea tre nuove per ogni originale), apportando le modifiche casualmente. Alla fine dell’esecuzione abbiamo ampliato il dataset, arrivando ad un totale di **2.935** campioni da utilizzare nelle fasi successive.

1. **Estrazione delle caratteristiche**

Questa è una fase molto importante nel processo di apprendimento automatico. Nel nostro caso specifico abbiamo eseguito l’estrazione delle caratteristiche con due architetture CNN:

* **Xception** (modello profondo 71 strati, pre-addestrato sul dataset ImageNet)
* **EfficientNetB4** (rete convoluzionale pre-addestrata su ImageNet, che ha l’efficienza come punto di forza).

Le due reti sono molto utilizzate per la classificazione di immagini, riescono a suddividerle in oltre mille categorie. Tuttavia, per il nostro scopo, abbiamo escluso i due strati finali dedicati proprio alla classificazione e abbiamo salvato le caratteristiche, rappresentate in una tabella, in un file csv. Alla fine di ogni tabella è stata aggiunta anche una colonna di label che definisce se l’immagine corrispondente rappresenta o meno un soggetto con disgrafia.

Il procedimento è stato svolto unendo i dati di entrambi i dataset, però abbiamo considerato ogni metodo di augmentation separato dagli altri, ottenendo sei differenti file csv.

1. **Classificazione con modelli di ML**

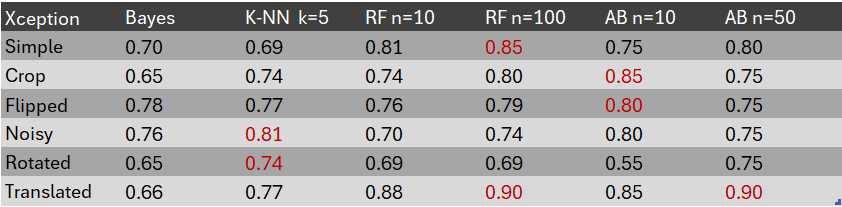
Nella parte della classificazione, abbiamo implementato quattro classificatori integrati all’interno della libreria sklearn:

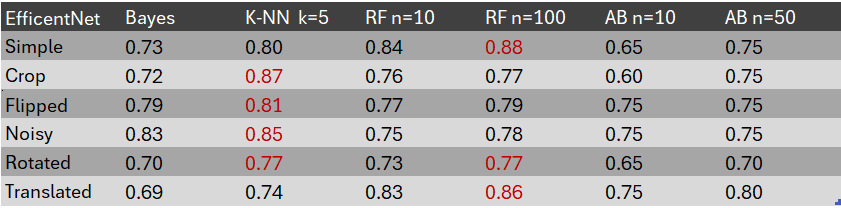
* **Naive Bayes**
* **K-Nearest Neighbours**
* **Random Forest**
* **AdaBoost**

A seguito delle esecuzioni con le diverse immagini dei dataset, abbiamo confrontato le prestazioni dei modelli.

1. **Analisi dei dati**

Per l’addestramento dei classificatori abbiamo optato per una suddivisione fra training set e test set rispettivamente di 80% e 20%, con un random\_state pari a 42. Successivamente abbiamo calcolato l’accuratezza dei modelli per ogni file csv ottenuto dalla feature extraction, evidenziando in rosso il modello con il valore maggiore per ogni dataset analizzato (nelle tabelle, k rappresenta il numero di neighbours considerati, n rappresenta il numero di estimatori):





Analizzando queste tabelle, si nota come la classificazione a partire dalle features estratte dalla rete EfficientNet abbia, in generale, dei valori di accuratezza più elevati rispetto alla rete Xception (fa eccezione il dataset in cui abbiamo implementato la traslazione).

Andando a considerare i singoli dataset, si può notare che i valori massimi si raggiungono con il dataset a cui è stata aggiunta la traslazione, nel caso di Xception, e con il dataset privo di dati aumentati nel caso di EfficientNet.

Invece, dal punto di vista dei singoli modelli notiamo che, mentre con EfficientNet spiccano i risultati dei classificatori K-NN e Random Forest, con Xception abbiamo una distribuzione più omogenea dei risultati più alti, anche su AdaBoost.

1. **Riferimenti**

Il nostro lavoro si trova su GitHub a questo link: <https://github.com/Umberto28/Deep_Learning_Project>

Dataset utilizzati:

<https://www.dropbox.com/scl/fo/6u7z1hcdwdd2ekgobjy8r/h?rlkey=9c8kr7dmbwgffogaojgwcodyi&e=1&dl=0> (contenuto all’interno del progetto)

<https://data.mendeley.com/datasets/39hr8dx76p/1>

Riferimenti per le tecniche di data augmentation:

<https://www.tensorflow.org/tutorials/images/data_augmentation?hl=it#using_tfimage>

<https://github.com/vxy10/ImageAugmentation>

Documentazione dei modelli di ML per la classificazione:

<https://scikit-learn.org/stable/supervised_learning.html#supervised-learning>