Progetto 2
Riconoscimento di facce generate tramite Taming Transformers

Un recente metodo di generare immagini, chiamata Taming Transformers[1], è in grado di generare volti sintetici estremamente realistici, tanto da essere difficili da distinguere da quelle naturali, vedi Figura 1. Purtroppo, esistono molti usi impropri di immagini del genere e quindi è importante riuscire a rivelarle [3, 2].

Obiettivo di questo progetto è quindi la realizzazione di una tecnica per stabilire se l'immagine di un volto è reale o generata. Per questo progetto adotterete una strategia di fine-tuning utilizzando l'architettura Xception pre-addestrata su ImageNet. In questo progetto i passi da seguire sono:

1. **Creazione del dataset**. Realizzate un dataset costituito da 3000 immagini di volti originali e 3000 immagini di volti sintetici. Potete scaricare i volti originali eseguendo la seguente istruzione su Colab:
   ```
   !wget --user=corso --password=p2021corso http://www.grip.unina.it/download/corso/ffhq_real.zip
   ```
   Ridimensionate tutto i volti originali a 256x256 pixel. Generare 3000 immagini di volti sintetici utilizzate il codice del metodo Taming Transformers[1].

2. **Preparazione dei dati**. Dividete le immagini nei tre set considerando il 66.6% per il training, il 16.7% per la validazione e il 16.7% per il test. Preparate i dati per la classificazione binaria, reale vs sintetico, utilizzando la funzione di Keras `ImageDataGenerator` con il metodo `flow_from_directory`. Per tutti i set, riportate le immagini nel range [-1,1] e ritagliate le immagini al centro di 224×224 pixel. Solo per il set di training, prevedete le seguenti operazioni di *data-augmentation*: rotazione random di angoli multipli di 90 gradi e blurring gaussiano con sigma variabile nel range $[0.0, 3.0]$. Utilizzate il parametro `preprocessing_function` di `ImageDataGenerator` per eseguire le operazioni indicate prima.

3. **Architettura**. Per definire le architetture utilizzate la funzione di Keras `Xception` usando una dimensione di ingresso di 224×224 pixel.

4. **Addestramento**. Per l'addestramento utilizzate l'ottimizzatore Adam tramite la funzione di Keras `keras.optimizers.Adam`, mentre per la loss function adottate la Cross-Entropy. Utilizzate le prestazioni sul set di validation per selezionare i migliori valori per il learning-rate, il batch-size, il numero di epoche e il numero di strati da bloccare della rete.

[1] `https://github.com/CompVis/taming-transformers`

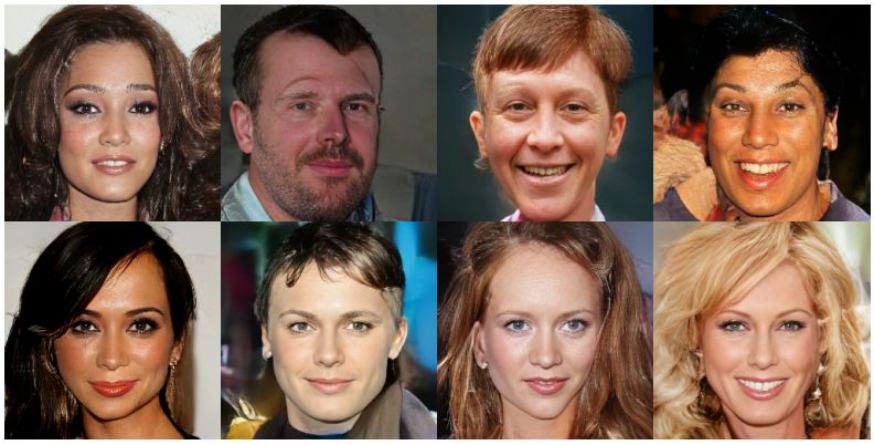

Figure 1: Esempi di volti generati.

5. **Valutazione delle prestazioni**. Utilizzate il test-set per valutare le prestazioni in termini di AUC (Area Under roc Curve) per ogni GAN ed usate la funzione `sklearn.metrics.roc_auc_score`.

References

[1] P. Esser, R. Robin, and B. Ommer, "Taming transformers for high-resolution image synthesis," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[2] D. Gragnaniello, D. Cozzolino, F. Marra, G. Poggi, and L. Verdoliva, "Are GAN generated images easy to detect? A critical analysis of the state-of-the-art," IEEE International Conference on Multimedia & Expo (ICME), 2021.

[3] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. Efros, "CNN-generated images are surprisingly easy to spot... for now," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.