# App chống giả mạo khuôn mặt

## Mục tiêu của đồ án tốt nghiệp

* Xây dựng một phần mềm chống giả mạo khuôn mặt (sử dụng **AI/Deep Learning**) trên app với hệ điều hành **Android** theo thời gian thực

(có thể làm thêm phần nhận diện khuôn mặt trong trường hợp có đủ thời gian, nhận biết người đó là ai)

* Học được lý thuyết và thực hành code với **AI / Deep Learning**
* Học được lý thuyết và thực hành code với **App Android**
* Viết báo cáo đồ án tốt nghiệp theo format của trường

## Ngôn ngữ lập trình và công cụ sử dụng

### 2.1. Công nghệ

* Ngôn ngữ sử dụng cho viết App: Java / Kotlin
* Ngôn ngữ sử dụng cho viết model, thuật toán dùng để chống giả mạo khuôn mặt: Python, C++

### 2.2. Công cụ sử dụng

* Android Studio
* Visual Studio Code
* Google Colab (dùng để train model)

## Quy trình thực hiện

### 3.1. Tìm hiểu về một số cách tấn công giả mạo và data tham khảo

* Hiện nay có rất nhiều phương pháp tấn công giả mạo những sẽ chủ yếu tập trung vào 2 loại sau là tấn công bằng 2D và 3D
* 2D attacks: một cái ảnh, một tấm hình in, một video chứa mặt người đó, những ảnh được thay liên tiếp để tạo sự chuyển động giả…
* 3D attacks: đeo mặt nạ giả, hình in 3D, tượng mặt người, makeup hoặc thậm chí là một con robot có khuôn mặt được điều chỉnh mô phỏng khuôn mặt ai đó (hơi ảo một chút nhỉ, nhưng khó mà biết tội phạm có thể làm ra đến mức nào)
* Trên thực tế, 2D diễn ra thường xuyên hơn, và chúng ta cũng sẽ tập trung xử lý những case phổ biến hơn. Vì vậy, các phương pháp mình đề cập trong bài này chủ yếu tập trung vào chống giả mạo cho 2D attacks. => chỉ làm đối với **những loại tấn công 2D trong đồ án tốt nghiệp**
* Link dataset tham khảo
* <https://www.kaggle.com/datasets/attentionlayer241/celeba-spoof-for-face-antispoofing>
* <https://www.kaggle.com/datasets/faber24/lcc-fasd>
* <https://www.kaggle.com/datasets/hlly34/liveness-detection-zalo-2022>

### 3.2. Nghiên cứu và phát triển các model để phát hiện khuôn mặt đó có phải giả mạo hay không

3.2.1. Model detect (phát hiện vị trí của khuôn mặt trên camera điện thoại)

* Hiện tại do chạy trên app và camera điện thoại sát mặt nên chỉ cần model SSD là được (nhẹ và độ chính xác khá cao trong trường hợp đưa sat vào camera)

3.2.2. Model classification (model dùng để nhận diên xem đối tượng đó là thật, giả mạo)

* Do phải thực thi trên app Android nên không thể lấy những model nặng như (VGG, Resnet, Efficient) đẻ chạy được (không tối ưu về mặt xử lý trên thiết bị di động) => Sử dụng các model lightweight ví dụ MobileNetv1, Mobilev2, Mobilenetv3.
* Train các mô hình dựa vào những tập data ở mục 3.1
* Đánh giá mô hình sau khi train được thông qua các metrics sau APCER, BPCER, Accuracy, HTER, EER
* Link đọc thêm về các chỉ số:
* <https://viblo.asia/p/tong-quan-ve-face-anti-spoofing-bai-toan-chong-gia-mao-khuon-mat-1Je5E6oYKnL>
* <https://arxiv.org/pdf/2106.14948.pdf> (page 6)

### 3.3. Viết code giao diện cho app Android

* Quay được camera trước và sau
* Vẽ hình chữ nhật xung quanh khuôn mặt trong trường hợp phát hiện được khuôn mặt
* Hiện thị text sau khi nhận diện người đó là **real** hay **fake**

### 3.4. Code chuyển thuật toán Python xuống C++

* Chuyển model detect và model classification sau khi train được dưới dạng (pth, pt, .v.v) xuống model (ncnn, onnx) (Mục đich chỉnh là để tăng tốc độ xử lý của mô hình

Link:

<https://github.com/Tencent/ncnn>

<https://github.com/onnx/onnx>

* Chuyển đổi code python -> c++ (ví dụ như đọc ảnh, xử lý box cho khuôn mặt, v.v)

### 3.5. Ghép code C++ vào app Android => tăng tốc độ thực thi

* Ghép code c++ v kotlin và java sử dụng **jni**