

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH



ĐỒ ÁN CUỐI KỲ

CÁC KỸ THUẬT HỌC SÂU VÀ ỨNG DỤNG

*Đề tài :*

*Triển khai mô hình dự đoán tuổi và giới tính lên thiết bị di động*

Giảng viên: T.S Nguyễn Vinh Tiệp

Nhóm sinh viên thực hiện:

Trương Đức Vũ - 18520194

Trần Trung Anh

Vũ Quý San

Lớp: CS431.L21

-- TP.HCM, 06/2021 --

**Mục lục**

**1. Abstract 1**

**2. Giới thiệu 4**

**3. Hướng tiếp cận 4**

**3.1. Bộ dữ liệu 5**

**3.2. Mô hình 5**

**3.2.1. Mô hình dự đoán tuổi 6**

**3.2.1. Mô hình dự đoán giới tính 6**

**3.2.1. Mô hình phát hiện khuôn mặt 6**

**3.3. Tensorflow lite 5**

**3.4. Ứng dụng Android 5**

**4. Kết luận 4**

**5. Tài liệu tham khảo 4**

1. **Abstract**

Trong đồ án này, chúng tôi thực hiện hai mô hình là xác định tuổi và giới tính với mục đích là triển khai trên điện thoại di động android. Ở hai mô hình trên, chúng tôi chọn mạng CNN đơn giản nhưng đủ nhanh và đáp ứng được kết quả tốt. Sử dụng bộ dữ liệu [UTKFace](https://www.kaggle.com/jangedoo/utkface-new) để đào tạo mô hình. Cuối cùng, chúng tôi chuyển model thành dạng TF Lite để phù hợp với cấu hình thiết bị di động.

1. **Giới thiệu**

**Diagram

Description automatically generated**

Deep learning là yếu tố then chốt cho nhiều tiến bộ gần đây trong ứng dụng trí tuệ nhân tạo. Lợi ích lớn hơn sẽ đến khi các công nghệ trí tuệ nhân tạo trở nên phổ biến trong các ứng dụng di động, chẳng hạn như tự động hóa lái xe, robot giá cả phải chăng cho gia đình và hỗ trợ cá nhân thông minh hơn trên điện thoại di động. So với cảm biến di động truyền thống và mô hình điện toán đám mây. Ưu điểm của việc chạy các mô hình trong ứng dụng di động thay vì gửi chúng lên đám mây là giảm độ trễ và khả năng đảm bảo quyền riêng tư dữ liệu cho người dùng. Mặc dù có nhiều thư viện học sâu và các công cụ AI, việc nhúng thành công mô hình học sâu vào ứng dụng di động có thể là một thách thức.

Vì vậy trong đồ án này, chúng tôi sẽ xây dựng một mô hình deep learning với mạng neural network tùy chình giúp giảm số lượng tham số cũng như giảm quy mô của mô hình. Ngoài ra, chúng tôi cũng dùng một số phương pháp để tối ưu mô hình. Cuối cùng, chúng tôi triển khai nó lên điện thoại di động.

1. **Hướng tiếp cận**
   1. **Bộ dữ liệu**

Như yêu cầu đặt ra của bài toán, ta cần một bộ dữ liệu đáp ứng được số lượng cũng như chất lượng cho ba bài toán nhỏ: Phát hiện khuôn mặt, Nhận diện tuổi tác và Nhận diện giới tính. Theo đó, chúng tôi chọn bộ dữ liệu UTKFace.



UTKFace là một dữ liệu lớn bao gồm hơn 20.000 hình ảnh khuôn mặt con người đã được gán nhãn với các thông tin về tuổi tác, giới tính và dân tộc. Bên cạnh đó, các hình ảnh trong bộ dữ liệu này đã được cắt, căn chỉnh vị trí của mỗi khuôn mặt con người, điều này hỗ trợ rất lớn cho bước Phát hiện khuôn mặt của ứng dụng.

Mỗi hình ảnh khuôn mặt con người trong bộ dữ liệu này được gán nhãn trực tiếp trên tên mỗi file, chi tiết như sau:

[age]-[gender]-[race]-[datetime].jpg

Trong đó:

* age: biểu diễn độ tuổi bằng một số nguyên trong khoảng từ 0 đến 116
* gender: biểu diễn giới tính bằng số 0 nếu là Nam, số 1 nếu là Nữ
* race: biểu diễn dân tộc của người trong hình, gồm White, Black, Asian, Indian và Others lần lượt bằng các số 0, 1, 2, 3, 4.
* datetime: biểu diễn thời gian mà bức hình được gán nhãn

Chẳng hạn với một mẫu dữ liệu trong bộ dữ liệu UTKFace,



Tên file: 1\_0\_0\_20161219154724341.jpg

Ta trích xuất được các nhãn như sau:

* Độ tuổi [age]: 1
* Giới tính [gender]: 1 - Nam

Dân tộc [race]: 0 - White

* 1. **Mô hình**

Để thực hiện hai mô hình trên Android, một là dự đoán tuổi và mô hình kia là phân loại giới tính. Vì các mô hình này được triển khai trên điện thoại thông minh Android (có sức mạnh tính toán tương đối thấp), vì vậy chung ta cần nhưng mô hình mà có ít số lượng tham số, do đó thì thời gian suy luận cũng thấp hơn. Bên cạnh đó, mô hình cũng phải tạo ra kết quả khả quan ít nhất là trên tập dữ liệu UTK Face.

Lý do trên thúc đẩy chúng tôi đào tạo trên một kiến trúc neural network tùy chỉnh (custom NN), vì nhưng mô hình hiện có đều có kích thước và số tham số cũng rất lớn vì vậy mà không thể nào tương thích với cấu hình của một điện thoại di động bình thường.

* + 1. **Mô hình dự đoán tuổi**
  1. Tổng quan

**Hình**: Input -> Convolution Block \* n -> Dense layers \* m -> Output

Mô hình đoán tuổi có đầu vào là một ảnh màu 3 kênh RGB kích thước 200x200x3 gương mặt một người, và đầu ra là một con số thực là tuổi dự đoán của người đó. Mô hình có hai thành phần chính: (1) các khối convolution(\*,ghi chú footnote) và (2) các tầng fully-connected. Ảnh đầu vào được đưa qua các khối convolution và các tầng fully-connected để rút trích ra các đặc trưng từ thấp đến cao và cuối cùng là đưa ra số tuổi dự đoán. Mục sau sẽ nói cụ thể hơn về hai thành phần chính này của mô hình.

\*: ghi ở footnote: Trong bài này, các khối convolution không chỉ gồm các phép convolution mà còn có kĩ thuật batch normalization, hàm kích hoạt leaky relu, ... Tương tự vậy với các tầng fully-connected (xem thêm tại mục b).

* 1. Các thành phần của mô hình đoán tuổi

b1. Khối convolution:

 Một khối convolution bao gồm một tầng convolution theo sau đó là việc áp dụng kĩ thuật batch normalization và kết thúc là hàm kích hoạt leaky relu.

(hình ảnh hoặc công thức: conv->batchnorm->leakyrelu)

 Chúng tôi chia mô hình đoán tuổi thành hai phiên bản: một phiên bản mô hình sử dụng phép standard convolution, gọi là mô hình standard, và phiên bản còn lại mô hình sử dụng phép separable convolution, gọi là mô hình lite.

(giải thích separable conv: cái này hình như là phép conv tiết kiệm tham số mà nhóm làm mobilenet có thuyết trình báo cáo môn thầy Dũng. Cái này có vẻ dài nên đọc sau <https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728> )

 Sau tầng convolution, chúng tôi sử dụng kĩ thuật batch normalization, một kĩ thuật chuẩn hóa đầu vào của một tầng, nhằm giảm số epoch cần train và giảm ảnh hưởng của giá trị khởi tạo tham số và tốc độ học lên kết quả train mô hình.

(hình minh họa).

 Tiếp theo đó, leaky relu được chọn làm hàm kích hoạt. Chúng tôi chọn leaky relu thay vì relu như thông thường bởi vì leaky relu giúp giải quyết tình trạng dying-relu trong quá trình train mô hình.

(hình: relu vs leaky relu)

b2. Các tầng fully-connected:

 Một tầng fully-connected bao gồm một tầng mạng nơ-ron fully-connected theo sau là hàm kích hoạt leaky relu và áp dụng thêm kĩ thuật dropout nhằm giảm overfitting và tăng khả năng tổng quát hóa của mô hình. Khi dùng dropout thì quá trình train mô hình sẽ ngẫu nhiên bỏ qua một số đơn vị (unit hay node) trong mạng nơ-ron của mô hình.

**(**hình drop out)

 Tầng cuối cùng của mô hình là một tầng mạng nơ-ron fully-connected có 1 node (unit) cho đầu ra là kết quả dự đoán của mô hình, cụ thể chính là số tuổi mà mô hình dự đoán.

* 1. Train mô hình đoán tuổi

 Chúng tôi chọn mean square error (MAE) làm hàm mục tiêu và sử dụng thuật toán tối ưu Adam để tối ưu tham số mô hình.

 Thêm vào đó chúng tôi sử dụng hai kĩ thuật thường được dùng trong quá trình train mô hình là:

Early stopping: kỹ thuật này giúp dừng việc train khi mô hình bắt đầu overfit dữ liệu train. Cụ thể, sau p epoch liên tiếp mà độ lỗi trên tập validation không giảm thì dừng train mô hình. Hyperparameter p ở đây trong early stopping thường được gọi là patience.

(hình minh họa early stopping)

Learning rate schedule: kỹ thuật này thay đổi learning rate trong quá trình train mô hình. Thực nghiệm cho thấy so với việc giữ nguyên learning rate thì thay đổi learning rate thường giúp mô hình đạt kết quả tốt hơn đồng thời giảm ảnh hưởng của giá trị khởi tạo learning rate đến kết quả train mô hình. Có nhiều cách để thay đổi learning rate (tăng dần, giảm dần, tăng dần đến ngưỡng nhất định rồi giảm dần, v.v ), chúng tôi chọn cách giảm dần learning rate sau mỗi lần cập nhật tham số.

(hình minh họa các đồ thị learning rate schedule khác nhau + hình so sánh fixed learning rate vs scheduled learning rate nếu có)

* + 1. **Mô hình dự đoán giới tính**
  1. Tiền xử lý dữ liệu

Trước khi sử dụng dữ liệu để huấn luyện mô hình, ta cần thực hiện các bước tiền xử lý như sau:

* Đọc hình ảnh từ file dữ liệu và lưu dưới dạng Numpy array, ở đây mỗi pixel được biểu diễn dưới dạng 3 giá trị RGB nên mỗi hình ảnh sẽ được lưu dưới dạng Numpy array với kích thước Chiều rộng Chiều dài 3. Trong giai đoạn này, chúng tôi chọn kích thước hình cố định là Chiều dài Chiều rộng = 128 128.
* Trích xuất nhãn Giới tính [gender] từ tên file tương ứng. Vì mỗi hình ảnh trong bộ dữ liệu sau khi được căn chỉnh được gán nhãn trực tiếp trên tên file.
* Biểu diễn 2 giá trị Giới tính dưới dạng One-hot.

Sau khi thực hiện xong, ta nhận được bộ dữ liệu với mỗi mẫu dữ liệu với kích thước [1281283]và nhãn Giới tính tương ứng với kích thước [12].

Phân chia dữ liệu thành hai tập Huấn luyện và tập Kiểm thử với tỉ lệ 7:3.

* 1. Thiết kế mô hình:

Để có thể thực hiện Nhận diện độ tuổi trên các thiết bị di động, mục tiêu chính của chúng tôi là thiết kế được một mạng CNN cơ bản với số lượng tham số ít hơn nhưng vẫn cố gắng đảm bảo độ chính xác của mô hình.

Theo đó, mỗi khối Convolution bao gồm một chuỗi các layer sau:

Conv2D BatchNorm LeakyReLU

Lớp Conv2D ở đây có thể sử dụng Convolution cơ bản, tuy nhiên với mục tiêu giảm bớt số lượng tham số, chúng tôi thêm một tùy chọn để có thể thay thế Convolution cơ bản bằng lớp Separable Convolution.

Ý tưởng chính của lớp Separable Convolution là phân tách kernel gốc thành các kernel nhỏ hơn, từ đó thực hiện ít phép tính hơn. Chẳng hạn với kernel quen thuộc Sobel, ta có có thực hiện như sau:

A picture containing diagram

Description automatically generated

Từ đó, thay vì thực hiện Convolution 9 phép tính thì chúng ta chỉ cần thực hiện 6 phép tính nên mô hình sẽ giảm được đáng kể thời gian huấn luyện và độ phức tạp.

BatchNorm (Batch Normalization) là một kỹ thuật chuẩn hóa dữ liệu đầu vào để tránh

trường hợp mô hình cố gắng tối gắng tối ưu một kết quả thay đổi liên tục, từ đó giúp ổn định và giảm thời gian huấn luyện.

Hàm activation Leaky ReLU giúp tránh trường hợp Dying-ReLU khi mà có một số dữ liệu đầu vào làm cho các đơn vị ReLU không hoạt động.

Sau đó, chúng tôi định nghĩa thêm một khối Dense để huấn luyện mô hình bằng các đặc trưng trích xuất từ khối Convolution, đồng thời thêm một lớp Dropout để giảm thiểu overfitting. Ở lớp Dropout, chúng tôi cài đặt tham số rate giảm dần để phù hợp với quá trình huấn luyện ở lớp Dense khi mà số lượng đơn vị tính toán cũng giảm dần sau mỗi epoch. Cuối cùng, chúng tôi thêm một lớp Dense và áp dụng hàm activation softmax để tính toán phân phối cho hai nhãn đầu ra: Nam và Nữ.

1. Huấn luyện mô hình:

Để thực hiện quá trình huấn luyện mô hình Nhận diện giới tính, chúng tôi tiến hành cài đặt mô hình trên với các thông tin sau:

* Hàm loss là CategoricalCrossentropy
* Thuật toán tối ưu tham số Adam.
* Độ đo Accuracy để đánh giá mô hình
* Áp dụng kỹ thuật Early Stopping để dừng quá trình huấn luyện nếu mô hình bị overfit, cụ thể là Accuracy của mô hình có dấu hiệu giảm sau mỗi epoch huấn luyện.
* Một số hàm callback để lưu lại mô hình tại một số thời điểm huấn luyện.
  + 1. **Mô hình phát hiện khuôn mặt**

ML Kit là một SDK di động mang kiến ​​thức chuyên môn về máy học của Google cho các ứng dụng Android và iOS trong một gói mạnh mẽ nhưng dễ sử dụng. Cho dù bạn là người mới hay đã có kinh nghiệm về học máy, bạn có thể triển khai chức năng mình cần chỉ trong một vài dòng mã. Không cần phải có kiến ​​thức sâu về mạng nơ-ron hoặc tối ưu hóa mô hình để bắt đầu. Mặt khác, nếu bạn là một nhà phát triển máy học có kinh nghiệm, ML Kit cung cấp các API tiện lợi giúp bạn sử dụng các mô hình TensorFlow Lite tùy chỉnh trong các ứng dụng dành cho thiết bị di động của mình. Vì vậy chúng tôi gọi ML Kit Face Detection để dùng luôn.

* 1. **Tensorflow Lite**

TensorFlow Lite là một bộ công cụ cho phép học máy trên thiết bị bằng cách giúp các nhà phát triển chạy mô hình của họ trên các thiết bị di động, nhúng và IoT. Vì vậy, chúng tôi quyết định sử dụng mô hình của TensorFlow Lite

Mô hình TensorFlow Lite được thể hiện ở định dạng di động hiệu quả đặc biệt. Nó cung cấp một số lợi thế so với định dạng mô hình thường của TensorFlow như giảm kích thước ít tham số, suy luận nhanh hơn mà không làm giảm đáng kể hiệu suất của mô hình. Điều đó, cho phép mô hình TensorFlow Lite thực thi hiệu quả trên các thiết bị có tài nguyên bộ nhớ và tính toán hạn chế như thiết bị điện thoại di động.

|  |  |  |
| --- | --- | --- |
| Mô hình suy luận tuổi | model | Lite model |
| Số lượng tham số | **1.069.297** | **200.956** |
| MAE | **0.0205** | **0.0458** |

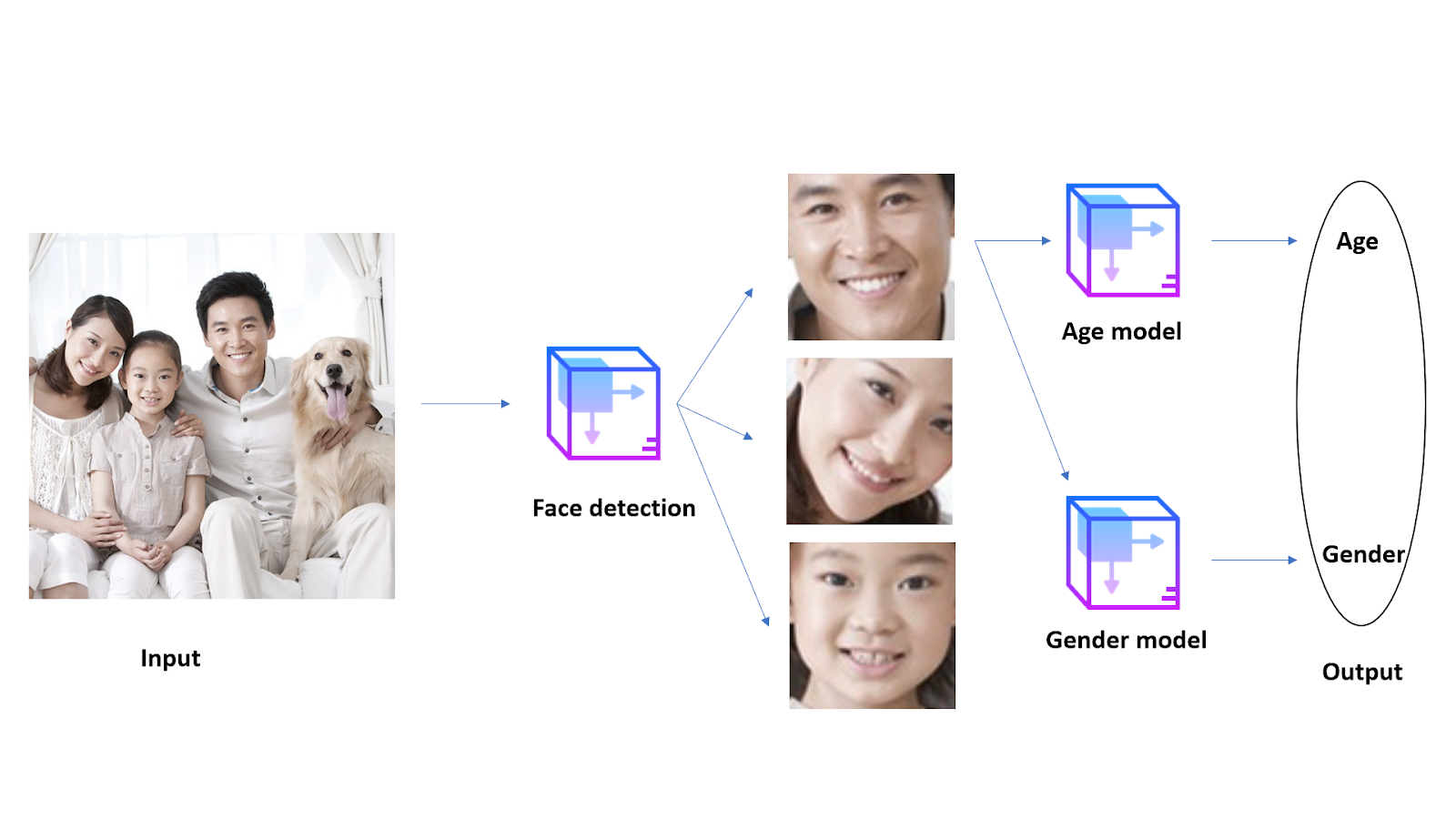
|  |  |  |
| --- | --- | --- |
| Mô hình phân loại giới tính | model | Lite model |
| Số lượng tham số | **675.090** | **328.733** |
| Accuracy (%) | **99.8** | **96.4** |

* 1. **Ứng dụng Android**

Để trực quan hóa, cũng như cho mọi người có thể sử dụng hai mô hình trên (kể cả người không biết gì về lập trình), chúng tôi đã triển khai mô hình lên một ứng dụng trên Android có tên là “Đoán tuổi-giới tính”.

Chúng tôi sử dụng ngôn ngữ lập trình Kotlin. Giống với Java, Kotlin chạy trên Java Virtual Machine và sử dụng được các công cụ và thư viện có sẵn trên Java. Kotlin có code ngắn gọn, giảm thiểu các đoạn không cần thiết, cấu trúc ngắn, trực quan, dễ học, dễ làm, dễ sử dụng, tiết kiệm thời gian. Đó cũng chính là lý do chúng tôi lựa chọn Kotlin để viết ứng dụng Android.

Ứng dụng “Đoán tuổi-giới tính” có một số chức năng như tùy chỉnh mô hình trước khi suy luận, nút chụp ảnh, nút tải ảnh lên và khoảng để hiển thị hình ảnh và kết quả.



Để suy luận ra được độ tuổi và giới tính tương ứng.



* Bước thứ nhất, hình ảnh chụp được hoặc hình ảnh tải lên được cho qua model Face Detection, kết quả đầu ra là một loạt khuôn mặt được, lấy khuôn mặt đầu tiên cắt ra rồi làm đầu vào của bước hai.
* Bước thứ hai, ảnh khuôn mặt được detect ở bước trước được cho vào hai model và thực hiện suy luận tuổi, giới tính.
* Bước cuối cùng là hiển thị kết quả như hình dưới. (Tuổi đúng của huấn luận viên Park Hang-seo là 63)

Trong phần xây dựng ứng dụng android, chúng tôi cũng đã tìm hiểu về GPU trên android nhưng do kết qua khi có GPU và không có GPU không có khác biệt gì nhiều. Bên cạnh đó, khi sử dụng GPU thì ứng dụng nhanh bị thoát ra vì không đủ RAM. Vì vậy, chúng tôi quyết định không thêm phần GPU khi suy luận nữa.

1. **Kết luận**
2. **Tài liệu tham khảo**