VIỆN NGHIÊN CỨU DỮ LIỆU LỚN VINBIGDATA
Chương trình đào tạo kỹ sư AI Vingroup
Khóa 3

BÁO CÁO DỰ ÁN CUỐI KHÓA
MÔN THỊ GIÁC MÁY TÍNH
NỘI DUNG: NHẬN DIỆN DẤU HIỆU BUỒN NGỦ CỦA TÀI XẾ

Ngày 25 tháng 9 năm 2023

Nhóm 7

1. Võ Văn Quang - 3733359
2. Nguyễn Trung Kiên - 3733357
3. Trần Anh Vũ - 3733390
4. Trần Đức Thọ - 3733365
5. Phạm Hoàng Tùng - 3733355

Mục lục

1 Giới thiệu bài toán và phân công công việc

1.1 Bối cảnh

Lái xe trong tình trạng không tỉnh táo là một hành động rất nguy hiểm và là nguyên nhân chính gây ra số lượng lớn những vụ tai nạn giao thông trên đường bộ. Sự thiếu minh mẫn này có thể được gây ra bởi thiếu ngủ, mệt mỏi, tác dụng phụ của thuốc, sử dụng chất có cồn, tiền sử bệnh nền, hay chỉ đơn giản là phải lái xe đêm đường dài. Những người tài xế có dấu hiệu buồn ngủ sẽ có phản ứng chậm, sự đánh giá không chính xác và mất đi khả năng tập trung khi đi trên đường, dẫn đến những vụ tai nạn giao thông không đáng có.

Một báo cáo riêng của Cục An toàn Giao thông Quốc gia Hoa Kỳ (National Highway Traffic Safety Administration) cho biết rằng lái xe khi buồn ngủ đã gây ra khoảng 72.000 vụ tai nạn, 44.000 trường hợp bị thương và 800 người chết vào năm 2013, và những con số này còn tiếp tục tăng trong các năm tiếp theo.

1.2 Mục tiêu

Để ngăn chặn những vụ tai nạn giao thông do cơn buồn ngủ của tài xế gây ra, một hệ thống phát hiện dấu hiệu buồn ngủ trên xe ô tô là thực sự cần thiết. Hệ thống này sẽ có tác dụng cảnh báo với tài xế khi có những dấu hiệu của sự mệt mỏi, thiếu tập trung, từ đấy có thể nhắc họ tập trung hơn, hay là tạm nghỉ giải lao để phục hồi sức lực và lấy lại được sự tỉnh táo, phòng tránh những sự cố tai nạn thương tiếc không đáng xảy ra.

Với mục tiêu muốn giải quyết vấn đề này, qua dự án này, chúng tôi muốn xây dựng một hệ thống phát hiện dấu hiệu buồn ngủ từ khuôn mặt và những đặc điểm khác của người tài xế, sử dụng thị giác máy tính kết hợp nhiều phương pháp học sâu, cấu trúc mạng tích chập và hồi quy khác nhau. Những mô hình này đã được huấn luyện trên một tập video được ghi hình lại bởi những người tài xế khi họ đang buồn ngủ và không buồn ngủ. Đây là một bài toán phân loại video nhị phân (Video Binary Classification) trong học máy.

1.3 Phân công công việc

Tất cả các thành viên trong nhóm đều đã tham gia chia sẻ ý kiến của mình vào xây dựng và viết các mục dưới đây trong bản báo cáo. Ngoài ra, nhóm chúng tôi đã phân chia dự án này thành những khối công việc chính với những người đảm nhận như sau:

- **Nguyễn Trung Kiên**: Đảm nhận train mô hình MTCNN + Video-SwinTransformer
- **Trần Anh Vũ**: Đảm nhận train mô hình Mediapipe + Inception Resnet + LSTM
- **Võ Văn Quang**: Đảm nhận train mô hình MTCNN + Inception ResNet + LSTM
- **Trần Đức Thọ**: Đảm nhận train mô hình Resnet + LSTM
- **Phạm Hoàng Tùng**: Đảm nhận train mô hình EfficientNet + LSTM

Phần code của dự án này có thể được tìm thấy tại link sau:
`https://github.com/TranAnhVu0411/Drowsiness_Detection_CV`

2 Tập dữ liệu và bài toán

2.1 Dữ liệu

Bộ dữ liệu chúng tôi sử dụng để huấn luyện là SUST Driver Drowsiness (SUST-DDD), đến từ Sivas University of Science and Technology [8]. Bộ dữ liệu bao gồm 2 nhóm video, với tổng cộng 19 người lái xe, gồm 3 phụ nữ và 16 nam giới. Những người tham gia đã ghi lại video về những khoảnh khắc lái xe bằng điện thoại riêng của họ trong xe cá nhân vào thời điểm họ mong muốn.

Mỗi video sẽ có nhãn là "buồn ngủ"(drowsy) và "không buồn ngủ"(not drowsy) cho mỗi người tham gia. Video có tên d_X.mp4 là những video trong đó người tham gia thể hiện hành vi buồn ngủ. Video có tên n_X.mp4 là những video trong đó người tham gia thể hiện hành vi không buồn ngủ. Trong quá trình ghi lại, những người tham gia không được yêu cầu phản ứng theo bất kỳ cách nào, và các phản ứng và triệu chứng được ghi lại theo quá trình tự nhiên của họ.

Số lượng video thu được là 2074, với 975 video mang nhãn drowsy và 1099 video mang nhãn not drowsy.

2.2 Bài toán

Từ tập dữ liệu trên, chúng tôi sẽ sử dụng các phương pháp học sâu để giải bài toán học có giám sát phân loại nhị phân về việc đánh giá một đoạn video có chứa dấu hiệu buồn ngủ của tài xế hay không. Mục tiêu sẽ là dự đoán chính xác giá trị của nhãn: 0 với video bình thường và 1 với video có dấu hiệu buồn ngủ.

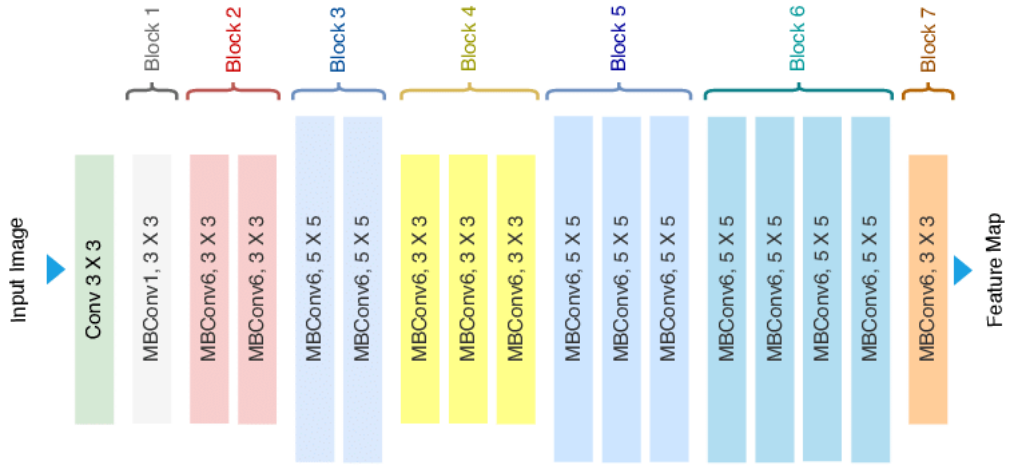
3 Cơ sở lý thuyết

Sau đây chúng tôi xin được phép giới thiệu phần lý thuyết liên quan đến những phương pháp đề xuất về sau của chúng tôi, chủ yếu liên quan đến cấu trúc của những mạng tích chập nổi tiếng

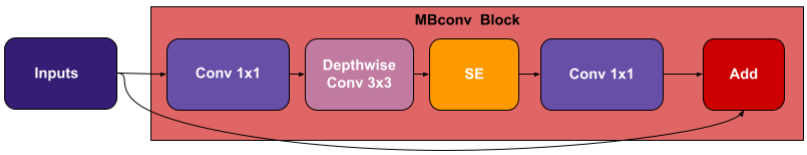
3.1 EfficientNet

Depth (độ sâu) tương ứng với số layers trong mạng, width (độ rộng) liên quan đến số neurons trong mỗi layer hoặc số filters trong mỗi conv layer (số channels của output) và cuối cùng là resolution (độ phân giải) đơn giản là height và width của ảnh đầu vào. EfficientNet được ra đời nhằm giải quyết vấn đề mở rộng mô hình (model scaling) - việc thay đổi các kích thước như depth, width hoặc resolution để tăng độ chính xác. EfficientNet không những tăng accuracy mà còn cải thiện hiệu suất tính toán của model bằng cách giảm đi các siêu tham số, có một mô hình baseline EfficientNet-B0 rất gọn và thực hiện phương pháp mở rộng kết hợp (compound scaling) riêng để tối hưu hóa mô hình [7].

Sau đây là kiến trúc của mạng EfffficientNet

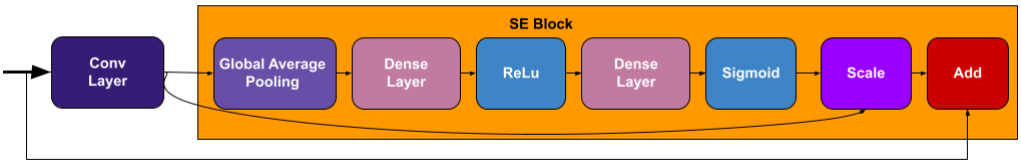


- Lớp MBConv là một khối xây dựng cơ bản của kiến trúc EfficientNet. Nó được lấy cảm hứng từ các khối residual nghịch đảo trong MobileNetV2 nhưng có một số điều chỉnh.



Hình 1: Lớp MBConv trong EfficientNet (SE)

- MBConv bắt đầu bằng một phép tích chập theo chiều sâu, tiếp theo là một phép tích chập điểm (convolution 1x1) để mở rộng số kênh, và cuối cùng là một phép tích chập 1x1 khác để giảm số kênh về số ban đầu. Thiết kế bottleneck này cho phép mô hình học một cách hiệu quả trong khi duy trì mức độ đại diện cao.



- Bên trong các lớp MBConv, EfficientNet còn tích hợp thêm khối Squeeze-Excitation (SE) giúp mô hình học tập trung vào các đặc trưng quan trọng và hạn chế những đặc trưng ít quan trọng hơn. Khối SE sử dụng pooling trung bình toàn cục để giảm kích thước không gian của bản đồ đặc trưng xuống còn một kênh duy nhất, tiếp theo là hai lớp fully connected. Những lớp này cho phép mô hình học các phụ thuộc đặc trưng theo từng kênh và tạo ra trọng số chú ý được nhân với bản đồ đặc trưng gốc, nhấn mạnh thông tin quan trọng.

3.2 ResNet

Resnet, hay có tên gọi đầy đủ khác là Residual Network, là một mạng tích chập CNN nổi tiếng được giới thiệu cho công chúng vào năm 2015 khi đã đoạt vị trí đầu tiên trong cuộc thi ILSVRC 2015 với tỷ lệ lỗi top 5 chỉ 3.57% [1]. Đặc điểm nổi bật nhất của Resnet là sử dụng các kết nối tắt (skip connection) đồng nhất để xuyên qua một hay nhiều lớp để học và giữ lại phần dư, được gọi là Residual Block, thay vì học thẳng từ input ra output để chống hiện tượng đạo hàm bằng 0 (vanishing gradient) khi x được cộng thêm một lượng ở lớp cuối như hình ảnh dưới đây

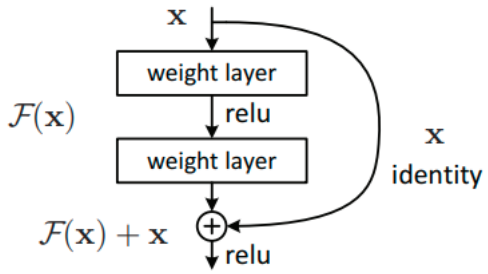
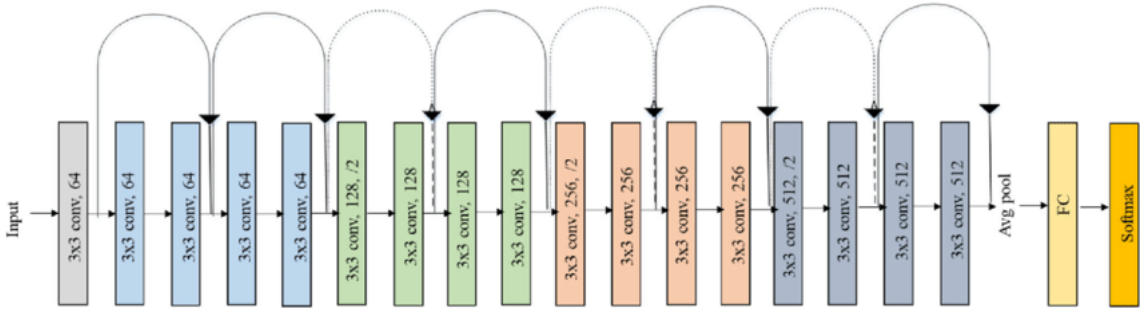


Figure 2. Residual learning: a building block.

Sau đấy là kiến trúc của mạng ResNet

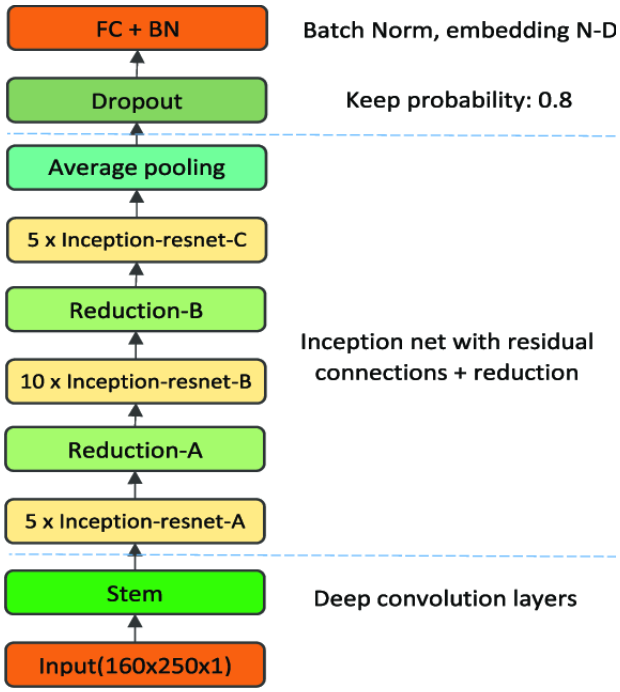


- Chồng các khối phần dư Residual Block

- Mỗi khối có 2 lớp 3x3 convolution
- Tăng gấp đôi số lượng filter với stride = 2 và giảm độ phân giải định kỳ
- Có lớp convolution phụ ở đầu mạng
- Lớp FC 1000 ở cuối để đề xuất kết quả phân loại 1000 lớp, sau đó sẽ được chuẩn hóa bằng softmax để ra xác suất phân loại mỗi lớp

3.3 Inception-ResNet

Kiến trúc mạng nơ-ron Inception-ResNet kết hợp các khái niệm từ mô-đun Inception (còn được gọi là GoogleNet) và kết nối Residual từ ResNet nhằm mục tiêu là đạt được hiệu suất học tốt hơn. Mạng này có tổng cộng 164 lớp và có khả năng phân loại hình ảnh vào 1000 đối tượng khác nhau, chẳng hạn như bàn phím, chuột, bút và nhiều loài động vật khác. Chính vì vậy, Inception-Resnet đã học được biểu diễn đặc trưng phong phú cho hàng loạt các kiểu hình ảnh [6].



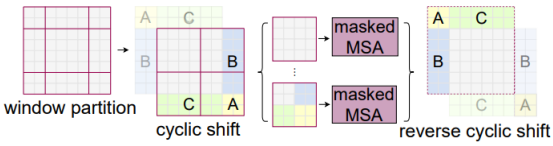
Sau đây là phần mô tả kiến trúc của mạng Inception-Resnet:

- Lớp đầu vào: Đầu vào của mạng thường là một hình ảnh hoặc một bản đồ đặc trưng từ một lớp trước đó.
- Stem: Lớp ban đầu của mạng được gọi là "stem". Thông thường, nó bao gồm một vài lớp tích chập và lớp gộp để trích xuất các đặc trưng cơ bản từ dữ liệu đầu vào.
- Các khối Inception-ResNet: Các khối xây dựng cốt lõi của Inception-ResNet là các khối Inception-ResNet. Mỗi khối chứa một chuỗi các mô-đun Inception được kết nối liên tiếp với các kết nối Residual xung quanh mỗi mô-đun. Dưới đây là cách một khối Inception-ResNet hoạt động:
- Các mô-đun Inception: Trong mỗi khối, có nhiều mô-đun Inception được xếp chồng lên nhau. Các mô-đun này bao gồm các bộ lọc tích chập song song có kích thước khác nhau (ví dụ: 1x1, 3x3, 5x5) và có thể bao gồm các thao tác khác như gộp cực đại. Ý tưởng là để nắm bắt các đặc trưng ở nhiều tỷ lệ cùng một lúc.

- Kết nối Residual: Tương tự như ResNet, có các kết nối Residual (kết nối bỏ qua) xung quanh mỗi mô-đun Inception. Các kết nối này thêm kết quả của một mô-đun vào đầu vào của một mô-đun sau trong cùng khối. Điều này cho phép mạng học các hàm dư thừa, giúp đào tạo mạng sâu một cách dễ dàng hơn.
- Global Average Pooling: Sau khi đi qua nhiều khối Inception-ResNet, các bản đồ đặc trưng được giảm chiều không gian bằng cách sử dụng pooling trung bình toàn cầu (GAP). Điều này bao gồm việc lấy giá trị trung bình của mỗi bản đồ đặc trưng trên các chiều không gian của nó, kết quả là một vector duy nhất cho mỗi bản đồ đặc trưng.
- Fully Connected Layers: Sau GAP, thường có một lớp kết nối đầy đủ cuối cùng. Lớp này có các nơ-ron tương ứng với số lớp đầu ra trong nhiệm vụ phân loại.
- Hàm Kích hoạt Softmax: Lớp kết nối đầy đủ cuối cùng được theo sau bởi hàm kích hoạt softmax. Hàm này chuyển đổi đầu ra của mạng thành xác suất lớp. Lớp có xác suất cao nhất là lớp được dự đoán.
- Đầu ra: Đầu ra cuối cùng của kiến trúc Inception-ResNet là nhãn lớp được dự đoán hoặc một tập hợp các xác suất lớp, tùy thuộc vào nhiệm vụ (ví dụ: phân loại hình ảnh).

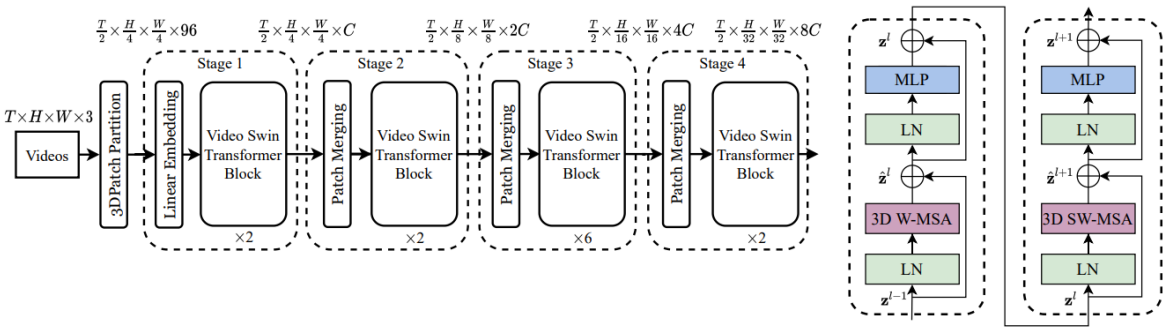
3.4 Video Swin Transformer

[2] VST (Video Swin Transformer) là một phiên bản nâng cấp của Video Vision Transformer (VVT). Nó được thiết kế để cải thiện độ phức tạp tính toán của cơ chế Attention trong VVT từ $O(4hwC^2 + 2(hw)^2C)$ thành $O(4hwC^2 + 2M^2hwC)$ với M là số lượng patches,trong khi hiệu suất được cải thiện một cách đáng kể. Ý tưởng của họ là mô hình sẽ có nhiều stage và trong mỗi stage đầu tiên họ sẽ merge các patches(windows) từ những stage trước. Sau đó thực hiện attention bên trong các patch rồi tiếp tục thực hiện shift toàn bộ ảnh và thực hiện attention lại một lần nữa. Họ cần thực hiện 2 bước như vậy là vì nếu chỉ thực hiện attention bên trong 1 window, chúng ta sẽ bị mất thông tin accros-windows. Vì vậy, để vượt qua vấn đề đó, họ đã đề xuất Shift-Window Attention.



Về cơ bản, phép shift ảnh là phép roll trong lập trình. Tuy nhiên nó sẽ gặp 1 số vấn đề đối với những window ở phía dưới và bên phải cùng của ảnh

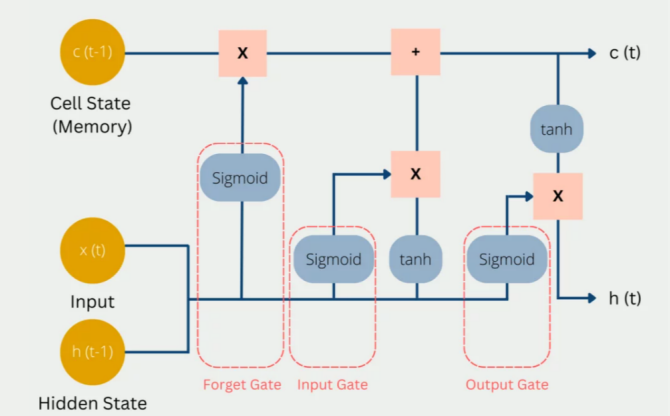
Như ví dụ ở trên, khi thực hiện roll (shfit) ảnh những phần ở trên sẽ chuyển xuống dưới và phần ở trái sẽ chuyển sang phải. Và như vậy, đối với những window ở dòng cuối cùng và cột cuối cùng sẽ gặp phải vấn đề đó là có thể nó sẽ attention với những thông tin không cần thiết. Như ở ví dụ bên trên, sau khi thực hiện shift, phần chứa con bird sẽ được chuyển xuống dưới và nó sẽ được attention với phần đất ở dưới và tương tự với phần bên trái chuyển sang bên phải. Đây là điều chúng ta không mong muốn. Do đó một cái mask đã được đề xuất để giải quyết vấn đề này. Ý tưởng của nó đơn giản chỉ là ngăn chặn vấn đề attention nhầm như đã được trình bày.



Hình 2: Kiến trúc của Video-Swin Transformer

3.5 Long Short Term Memory

LSTM (Long Short-Term Memory) là một biến thể nâng cao của mạng nơ-ron hồi quy (RNN), được thiết kế để dễ dàng lưu trữ thông tin quá khứ vào bộ nhớ của nó. LSTM giải quyết vấn đề Vanishing Gradient hoặc Exploding Gradient của RNN một cách hiệu quả hơn, rất phù hợp để giải quyết các bài toán như phân loại, xử lý và dự đoán chuỗi thời gian có độ dài không xác định. LSTM có lợi thế hơn so với RNN thông thường là LSTM có khả năng nhớ những thông tin quan trọng từ những bước ở xa trong quá khứ, giúp nâng cao khả năng dự đoán so với RNN khi mà RNN chỉ tận dụng được thông tin ở tương lai gần.



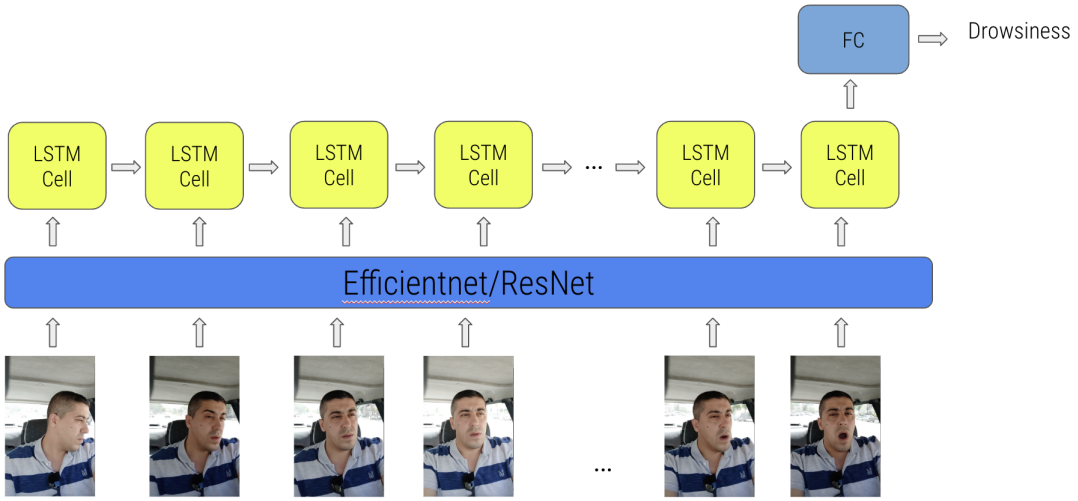
Các bước xử lý trong LSTM (Long Short-Term Memory) bao gồm:

- Input Gate: Cổng này quyết định thông tin nào được đưa vào ô trạng thái bộ nhớ.
- Forget Gate: Nó điều khiển thông tin nào nên bị loại bỏ khỏi ô trạng thái bộ nhớ.
- Memory Cell Update: Ở đây, ô trạng thái bộ nhớ được cập nhật dựa trên quyết định của input và output gate
- Output Gate: Cổng này xác định thông tin nào từ ô trạng thái bộ nhớ nên được xuất ra như dự đoán.

4 Các phương pháp đề xuất

4.1 Phát hiện buồn ngủ sử dụng toàn bộ frame

Với mỗi video, chúng tôi sẽ lấy một số lượng frame nhất định, resize nó thành những bức ảnh cố định có chiều 224x224 rồi sau đó chuẩn hóa từng channel trong ảnh. Sau đó, chúng tôi sẽ sử dụng những cấu trúc mạng tích chập mạnh mẽ khác nhau như EfficientNet, ResNet để véc-tơ hóa những bức ảnh này trước khi cho chúng vào mô hình Long Short Term Memory (LSTM). Lí do mô hình LSTM được đưa vào bước cuối của các model là nhằm xử lý dữ liệu dạng chuỗi, tại vì mỗi video là một chuỗi của các frame.



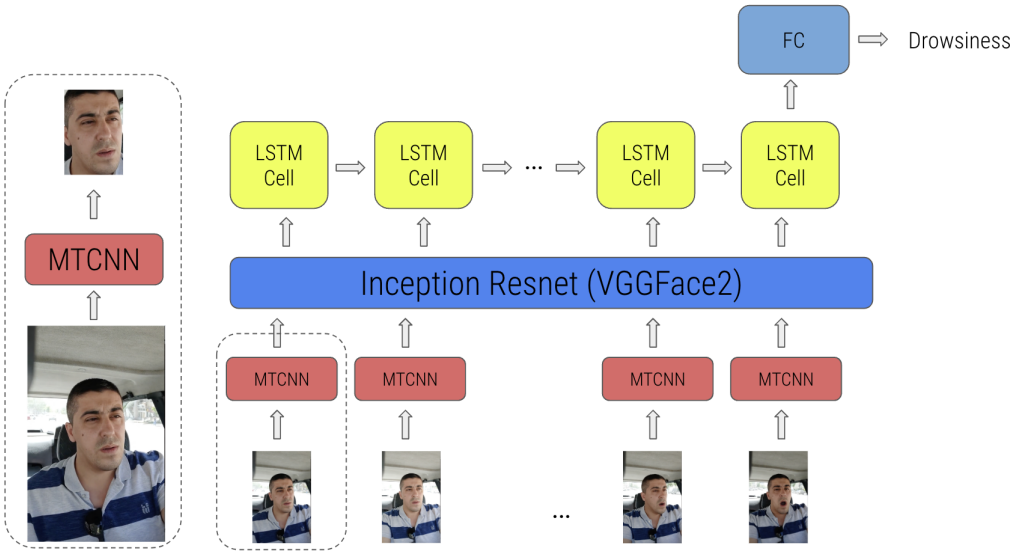
Hình 3: Kiến trúc mô hình sử dụng EfficentNet/ResNet + LSTM cho từng frame

4.2 Phát hiện buồn ngủ sử dụng ảnh khuôn mặt

4.2.1 MTCNN + Resnet + LSTM

Để có thể phát hiện buồn ngủ một cách chính xác hơn, thay vì sử dụng toàn bộ frame như hướng đề xuất đầu tiên, frame đầu vào sẽ được đi vào bộ xác định khuôn mặt, cụ thể ở đây chúng tôi dùng mô hình MTCNN [9] để trích xuất khuôn mặt.

Sau đó chúng tôi thực hiện tương tự như hướng đề xuất trên, đưa các ảnh khuôn mặt vào mô hình CNN để véc-tơ hoá khuôn mặt, và đưa tập véc-tơ đã mã hoá vào mô hình LSTM. Tuy nhiên, một điểm khác ở đây so với hướng đề xuất đầu tiên là chúng tôi sử dụng mô hình Inception Resnet với trọng số sử dụng của VGGFace2 thay vì các mô hình pretrained như EfficientNet và ResNet, do mô hình trên chuyên được sử dụng cho bài toán Face Recognition [5].

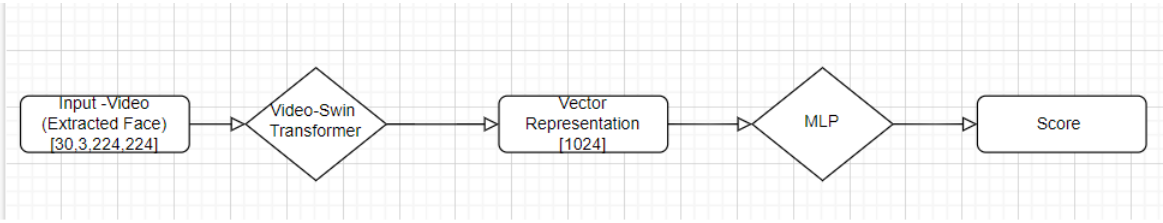


Hình 4: Kiến trúc mô hình sử dụng MTCNN để trích rút khuôn mặt và Inception Resnet + LSTM cho từng ảnh khuôn mặt được trích rút

4.2.2 MTCNN + Video Swin Transformer

Mối liên hệ giữa các bộ phận trên khuôn mặt có thể sẽ là 1 dấu hiệu tương đối ổn giúp phát hiện buồn ngủ. Ví dụ như khi người lái xe ngáp thì thường mắt họ sẽ thu hẹp lại hơn so với bình thường. Vì thế chúng tôi muốn thử thực hiện embed toàn bộ video thành 1 vector biểu diễn bằng video swin transformer sau đó đưa qua bộ phân loại MLP để đưa ra dự đoán.

Cụ thể, với mỗi video 10s, chúng tôi sử dụng MTCNN để trích xuất khuôn mặt của 30 frames trong video đó và Resize lại về kích thước (224,224) thu được tensor với kích cỡ $[30, 3, 224, 224]$. Sau khi cho qua Video Swin Transformer ta thu được 1 vector 1024 chiều biểu diễn cho tensor đấy. Sau đó một lớp MLP với sigmoid function ở cuối được thêm vào để đưa ra điểm số cho video đó.



Hình 5: MTCNN + Video-Swin Transformer

4.3 Phát hiện buồn ngủ sử dụng các đặc trưng của khuôn mặt

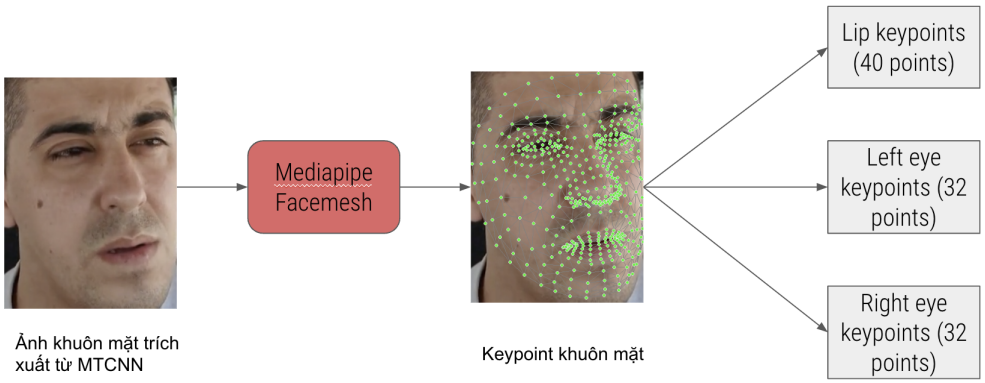
Các dấu hiệu của buồn ngủ thường thể hiện rõ nét thông qua các đặc trưng của khuôn mặt như mắt, miệng và hướng ngiêng của đầu, vì vậy hướng đề xuất này sẽ tập trung vào sử dụng các đặc trưng trên.

4.3.1 Phát hiện buồn ngủ sử dụng các keypoint của mắt và miệng

Trong hướng này, sau khi phát hiện khuôn mặt sử dụng MTCNN, chúng tôi tiến hành xác định các keypoint của mắt và miệng sử dụng Mediapipe [3]. MediaPipe là tập hợp của một loạt

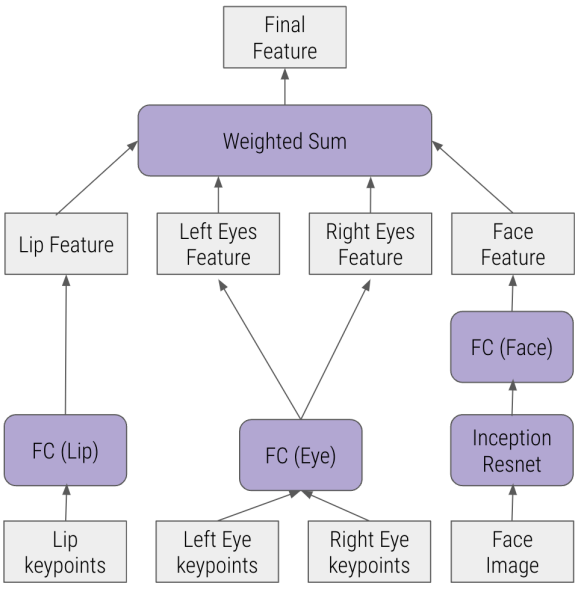
các giải pháp Machine Learning đa nền tảng, có thể can thiệp được và cực kỳ lightweight được phát triển bởi Google, hầu hết các bài toán nổi bật trong lĩnh vực Computer Vision - Thị giác máy tính, đều được Google cài đặt trong MediaPipe như Face Detection, Face Mesh, ...

Cụ thể, trong phạm vi dự án này, chúng tôi sử dụng giải pháp FaceMesh của Mediapipe để xác định 16 keypoint của từng mắt và 40 keypoint của miệng trên khuôn mặt (Kết quả trả về gồm các toạ độ điểm x và y đã được normalize theo ảnh):



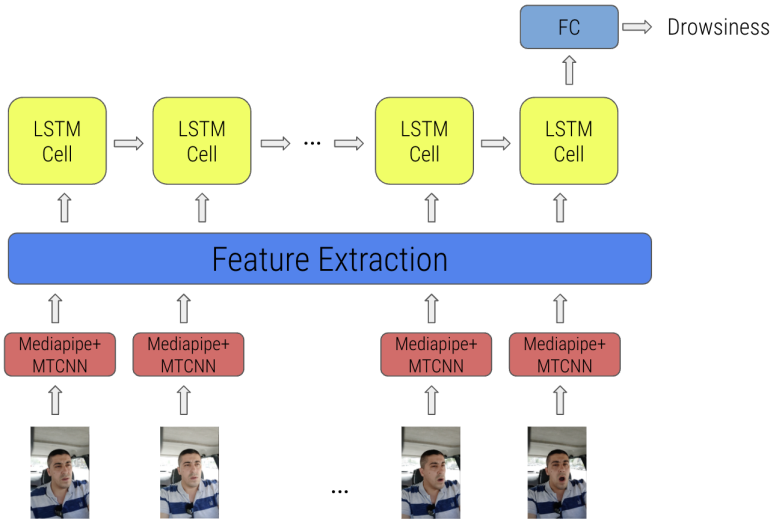
Hình 6: Mediapipe trích xuất keypoint

Sau khi có thông tin keypoint của mắt và miệng, các thông tin này sẽ được trải phẳng thành các vector 1 chiều, cụ thể vector ứng với các keypoint của mắt sẽ có kích thước là 16*2, và của miệng sẽ có kích thước là 40*2. Sau đó các vector này sẽ được đi qua tầng fully connected layer để mã hoá thông tin thành các vector kích thước 512. Cuối cùng các vector mã hoá của keypoints và vector mã hoá ảnh khuôn mặt sẽ được tổng hợp lại bằng phép cộng trọng số để thu về véc-tơ mã hoá cuối cùng của từng khuôn mặt trên từng frame.



Hình 7: Mã hoá thông tin keypoint + khuôn mặt

Cuối cùng, các véc-tơ mã hoá của các frame sẽ được đi qua LSTM, tương tự như các phương pháp đã đề xuất. Hình vẽ dưới tổng kết lại hướng đề xuất này:



Hình 8: Kiến trúc mô hình sử dụng ảnh khuôn mặt kết hợp thông tin keypoint

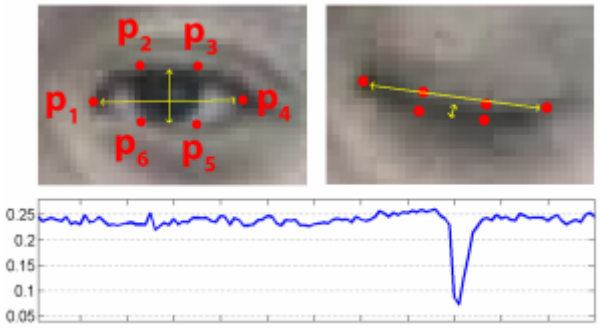
4.3.2 Phát hiện buồn ngủ sử dụng các đặc trưng đặc biệt của mắt

Trong hướng này, thay vì sử dụng keypoint thô như hướng trên, chúng tôi tổng hợp đặc trưng có ý nghĩa hơn từ các keypoint trên, cụ thể ở đây là đặc trưng Eye Aspect Ratio (EAR).

Trong paper [4] của Tereza Soukupova và Jan ´ Cech đã tìm ra được một công thức để xác định xem mắt nhắm hay mở dựa theo ngưỡng, cụ thể công thức có dạng:

$$EAR = \frac{||\overrightarrow{p2} - \overrightarrow{p6}|| + ||\overrightarrow{p3} - \overrightarrow{p5}||}{2||\overrightarrow{p1} - \overrightarrow{p4}||} \quad (1)$$

Trong đó p1,p2,p3,p4,p5,p6 là các keypoint của một mắt, nếu giá trị EAR nhỏ hơn một ngưỡng nào đó thì sẽ được coi là nhắm mắt:

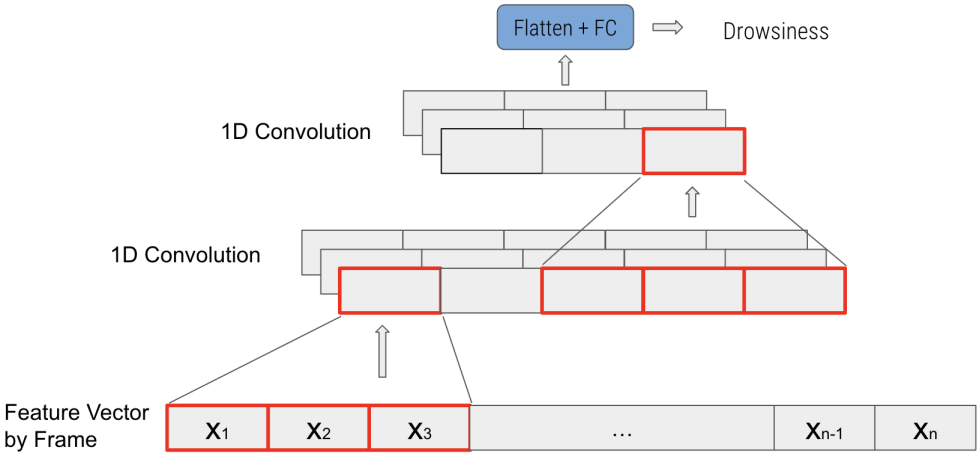


Hình 9: EAR và đồ thị biểu diễn giá trị EAR theo thời gian

Chúng tôi tổng hợp các giá trị này và tạo ra feature vector 2 chiều (EAR mắt trái và EAR mắt phải) ứng với mỗi frame.

Cuối cùng, chúng tôi sử dụng các mạng CNN 1D xếp chồng để huấn luyện bộ vector này. Lý do chúng tôi sử dụng CNN trong trường hợp này thay vì LSTM: các chỉ số EAR, MAR, MOE hay HPD trong phần lớn thời gian sẽ ghi nhận giá trị khá ổn định. Những giá trị biến thiên mạnh thường sẽ xảy ra trong khoản thời gian rất ngắn. Tuy nhiên, nếu những giá trị ngoại lai

này lại duy trì trong thời gian dài (ví dụ EAR thấp liên tục trong hơn 1s), rất có thể đây là chỉ báo cho dấu hiện buồn ngủ. Chúng tôi cho rằng CNN (với cấu trúc cửa sổ trượt) sẽ phù hợp hơn LSTM trong việc nhận diện các tín hiệu bất thường này.



Hình 10: Kiến trúc mô hình CNN sử dụng đặc trưng trích xuất từ keypoint

5 Kết quả và đánh giá

5.1 Độ đo và phương pháp đánh giá

Chúng tôi sẽ tiến hành chia tập dữ liệu thành 5 folds, mỗi fold sẽ chứa video từ 3 đến 4 người lái và mỗi người lái chỉ nằm trong 1 fold duy nhất. Chúng tôi sẽ đánh giá bằng phương pháp 5-fold cross validation: train trên 4 folds và dự đoán trên fold còn lại, và lặp điều này trong 5 lần rồi lấy kết quả trung bình của 5 lần này.

Với bài toán này, chúng tôi vẫn sẽ báo cáo kết quả của các tiêu chí liên quan đến bài toán phân loại như accuracy, precision, recall, f1-score. Tuy nhiên, chúng tôi sẽ chú trọng nhiều hơn vào video recall (độ phủ), tiếp theo đến accuracy (độ chính xác) bởi chiến thuật "giết nhầm còn hơn bỏ xót": hậu quả của việc không phát hiện được dấu hiệu buồn ngủ khi tài xế thực sự buồn ngủ là rất nguy hiểm, có thể gây tai nạn chết người, còn hậu quả của việc phát hiện nhầm dấu hiệu buồn ngủ khi tài xế đang tỉnh táo (giả sử như nhắm mắt hoặc há mồm) gần như là không có (tài xế có thể cảm thấy phiền phức khi bất cứ hành động nào của mình cũng có thể bị mô hình báo động). Chúng tôi huấn luyện mô hình với tiêu chí recall trong đầu, nhằm mục đích thà phát hiện nhầm một người buồn ngủ, dự đoán buồn ngủ nhiều hơn, để không bị bỏ lỡ bất kỳ trường hợp buồn ngủ thực sự nào mà không bị phát hiện.

5.2 Kết quả và đánh giá các mô hình

Key Feature	Mô hình	Accuracy	Precision	Recall	F1-score	Inference Time (ms/video)
Whole Frame	ResNet + LSTM	0.545	0.342	0.4154	0.415	88.12
	EfficentNet + LSTM	0.5583	0.5533	0.2932	0.3834	90.21
Cropped Face	MTCNN + Inception Resnet + LSTM	0.6516	0.6312	0.6289	0.6301	97.36
	MTCNN + Video-Swin-Transformer	0.6412	0.62	0.64	0.6298	98.34
Cropped faces + facial keypoints	Mediapipe + Inception Resnet + LSTM	0.6125	0.5868	0.7279	0.6498	161.29
Keypoints feature	EAR + CNN	0.56	0.5315	0.5653	0.5479	3.98

Bảng 1: Kết quả của các mô hình

Nhận xét kết quả

Từ kết quả này, chúng tôi nhận thấy các các mô hình sử dụng đầu vào là toàn bộ Frame (Chúng tôi gọi các mô hình này là Baseline) trả về kết quả F1 khá thấp, trong khi đó các mô hình chỉ sử dụng ảnh khuôn mặt cho ra kết quả khá tốt.

Khi xem xét kĩ hơn các mô hình sử dụng đầu vào là khuôn mặt, chúng tôi nhận thấy mô hình sử dụng thông tin keypoint đạt kết quả F1 tốt hơn so với các mô hình không sử dụng keypoint, nhưng khi xem xét đến accuracy thì kết quả lại không bằng so với các mô hình không sử dụng keypoint. Có khả năng thông tin keypoint chúng tôi sử dụng trong mô hình vẫn chưa được tận dụng một cách hợp lý trong bài toán này.

Mô hình sử dụng thông tin trích rút từ keypoint đạt kết quả tốt hơn so với các mô hình baseline, nhưng lại tệ hơn so với mô hình sử dụng ảnh khuôn mặt, một phần lý do là các đặc trưng tổng hợp khá là ít (vector đặc trưng chỉ có 2 chiều đại diện cho EAR của 2 mắt).

Nhận xét thời gian xử lý

Khi xét đến thời gian xử lý, thì các mô hình sử dụng ảnh khuôn mặt làm đầu vào có thời gian xử lý lâu hơn so với các mô hình baseline, lý do nằm ở chỗ các mô hình này cần có thêm một khoảng thời gian để trích rút khuôn mặt từ mô hình MTCNN nên thời gian xử lý lâu. Đặc biệt, mô hình sử dụng thêm keypoint có thời gian xử lý lâu nhất, do quá trình lấy đặc trưng từ các điểm keypoint mất nhiều thời gian.

Tuy nhiên, khi xét đến mô hình chỉ sử dụng thông tin trích rút từ keypoint, mô hình về cơ bản có thời gian xử lý nhanh hơn nhiều so với các mô hình còn lại vì 2 lý do: thời gian xử lý dữ liệu ngắn (Mediapipe hỗ trợ xác định keypoint trực tiếp trên frame) và mô hình rất đơn giản (CNN có cấu trúc đơn giản hơn và có khả năng xử lý song song).

6 Kết luận và Đề xuất

6.1 Kết luận

Chúng tôi đã khảo sát các hướng đề xuất giải quyết bài toán Phát hiện buồn ngủ và cho ra kết quả khá khả quan. Nhìn chung, các mô hình baseline sử dụng cả bức ảnh sẽ cho ra kết quả accuracy và recall kém hơn là những mô hình nâng cao hơn tập trung vào khuôn mặt và sử dụng keypoints của khuôn mặt để dự đoán. Mô hình MTCNN + Inception ResNet + LSTM đã cho ra kết quả accuracy cao nhất, nhưng Mediapipe + Inception ResNet + LSTM thì cho kết quả recall tốt nhất. Về thời gian dự đoán, các mô hình đòi hỏi trích lọc đặc trưng sẽ có thời gian chạy lâu hơn so với những mô hình baseline (trừ keypoins feature), do tốn thêm thời gian trích xuất keypoints đúng như những gì chúng tôi mong đợi.

Việc có accuracy đa phần đều trên 0.5 và recall đều trên 0.6 (cho những model tốt) là một điều khá tích cực, vì đến cả con người khi làm công việc dự đoán dấu hiệu buồn ngủ cũng chỉ đoán đúng được khoảng 50% số ảnh. Các mô hình của ta đã đoán chính xác hơn mắt người bình thường đoán, thể hiện một sự tiến bộ hơn khi các mô hình học máy đã có khả năng tinh vi tính toán và trích xuất các đặc trưng cụ thể hơn mắt người. Việc recall của chúng ta đạt được ở mức 0.7 cho model tốt nhất thể hiện rằng cứ tầm 10 người buồn ngủ trên xe, ta sẽ dự đoán đúng được khoảng 7 người và cứu họ được khỏi tai nạn. Ta cân bằng recall này với accuracy để tránh khỏi việc mô hình đạt được recall tối đa bằng cách lúc nào cũng dự đoán buồn ngủ, dẫn đến chuông báo động, gây phiền toái liên tục cho người lái xe.

6.2 Đề xuất

Do thời gian và nguồn lực có hạn, chúng tôi đã chưa kịp thử hết một vài phương pháp và cách xử lý dữ liệu tối ưu hơn. Nếu có thời gian, chúng tôi sẽ tiến hành thử thêm những ý tưởng sau:

- Thêm các đặc trưng của khuôn mặt: Ngoài mắt ra thì có một số đặc trưng khác của khuôn mặt cũng thể hiện được sự mệt mỏi như các vết nhăn trên trán, miệng hoặc góc nghiêng của đầu, chúng tôi sẽ có thể tận dụng những thông tin đó để cải thiện thêm chất lượng mô hình.
- Thay đổi chiến lược xây dựng dữ liệu huấn luyện: Hiện tại dữ liệu đang được xử lý lấy theo số lượng frame, tuy hiên, phương pháp xử lý này về cơ bản vẫn chưa thực sự hợp lý do các video được quay trên các thiết bị khác nhau có frame rate cũng sẽ khác nhau. Chúng tôi cũng khảo sát một số bài báo thay vì sử dụng frame làm đầu vào cho mô hình LSTM, thì các tác giá của các bài báo đó sử dụng thông tin nháy mắt làm đầu vào, và thông tin đó hoàn toàn độc lập với thông số của thiết bị.
- Có thể áp dụng các mô hình Action Recognition cho bài toán phát hiện buồn ngủ.
- Tạo ra sản phẩm thực tế.

Tài liệu tham khảo

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[3] C Lugaresi, J Tang, H Nash, C McClanahan, E Uboweja, M Hays, F Zhang, CL Chang, MG Yong, J Lee, et al. Mediapipe: A framework for building perception pipelines. arxiv 2019. *arXiv preprint arXiv:1906.08172*, 5, 2019.

[4] Bhargava Reddy, Ye-Hoon Kim, Sojung Yun, Chanwon Seo, and Junik Jang. Real-time eye blink detection using facial landmarks. *IEEE CVPRW*, 2017.

[5] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[7] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[8] Esra Kavalcı Yılmaz and M Ali Akcayol. Sust-ddd: A real-drive dataset for driver drowsiness detection. In *Conference of Open Innovations Association (FRUCT)*, volume 6, 2022.

[9] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE signal processing letters*, 23(10):1499–1503, 2016.