1. \*Dataset UCF101: Được dùng để chạy thí nghiệm ban đầu để so sánh với dataset tự thu thập của nhóm. Mô tả: It consists of 101 action classes, over 13k clips and 27 hours of video data. The database consists of realistic user-uploaded videos containing camera motion and cluttered background.

\*chuẩn bị dataset SEAGS\_V1 theo cấu trúc giống như UCF101

+ Adap với các nghiên cứu đã thực hiện trên dataset UCF101 => thiết kế cấu trúc dataset SEAGS theo cấu trúc như UCF101.

// ảnh cấu trúc

Từng video nguồn sẽ đặt trong thư mục có tên theo tên class và tên file:

v\_<Class name> \_<index>.mp4

File classIdx chứa nội dung như là map <index>-<class label>

File testlist chứa các đường dẫn file video (từ thư mục /data) của dữ liệu tập test

File trainlist chứa các đường dẫn file video (từ thư mục /data) của dữ liệu tập train và class index tương ứng của từng video.

+ Nguồn thu thập là từ video youtube và các nền tảng reel như tiktok. Trong đó nguồn từ youtube đóng vai trò như là dữ liệu cơ sở cho các hành động của từng lớp. Còn nguồn từ các nền tảng reel là các short video có hiệu ứng, text chèn vào có tác dụng như là làm augment cho dữ liệu giúp việc nhận dạng hành động trong các video hiệu ứng ngắn tốt hơn.

+ Video ngắn hiệu ứng thường có độ dài dưới 30 giây sẽ được cắt lấy hết trừ các đoạn intro outro, còn video trên youtube có độ dài khá lớn hay các nền tảng reel khác nhưng có độ dài lớn hơn 30 giây sẽ được chọn cắt thủ công các phân đoạn có hành động tương ứng với lớp, độ dài khoảng từ 5-20 giây, và lấy tối đa 4 phân đoạn cho cùng 1 video để tránh sự lặp lại của dữ liệu.

+ Sau quá trình thu thập dữ liệu của nhóm, kết quả phiên bản 1 của dataset là thu thập được 8 lớp. Mỗi lớp có khoảng 100-150 video, mỗi video dài từ 5-20 giây.

// ảnh

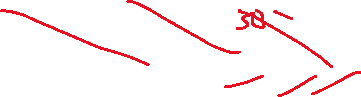
// ảnh

// ảnh

+ Sau khi thu thập, tất cả video của một lớp được trộn và chia ra làm 2 tập Train và Test theo tỉ lệ 7:3, lưu lại trong 2 file testlist và trainlist.

Table

Description automatically generated



1. data augmentation thì t có flip ảnh nữa, ròi m show vài cái cho vui (cho dài) :))

Video ngắn trên các nền tảng Reel thường có nhiều hiệu ứng tác động lên video như zoom cận, cắt góc,… khiến cho cùng một khung cảnh nhưng lại trông khác nhau. Do đó để enrich dữ liệu thì nhóm đã áp dụng thêm thao tác flip ảnh.



1. cnn thì m show là m chạy backbone j, cnn có bnhiu layer, activation của layer cuối là softmax, các figure cho phần cnn

Thực hiện thí nghiệm với dataset SEAGS\_V1 trên 5 backbone: EfficientNet, InceptionNet, ResNet, MobileNet, DenseNet (without embedded motion) thu được kết quả:

// bảng accuracy 5 backbone

Tiếp theo, nhóm chạy thí nhgieejm với EnsembleNet. Backbone này hoạt động bằng cách lấy trung bình score của 4/5 backbone nói trên: EfficientNet, InceptionNet, MobileNet, DenseNet, EnsembleNet (Loại trừ ResNet do có accuracy tương đối thấp).

(3.5) tạo motion image thì thử nghiệm N = bnhiu, interval frames (cái này t chưa đề cập trong method nên m chắc phải ns tí), dùng Gradient Division (xem phần method của t nha, cũng dễ hiểu)

Motion image được tạo với N = 10, interval\_frames = 2. Giải thích thêm về interval frames, mỗi motion image được tạo thành từ 1 batch N frame. Interval frames cho biết khoảng cách (frames) giữa 2 batch kế tiếp nhau, có nghĩa là khoảng cách từ frame đầu tiên của batch thứ k đến frame đầu tiên của batch thứ k+1. Quá trình tạo motion image có sử dụng Gradient Division.

1. training thì m ns là loss function là gì, bnhiu epoch, hoặc là chạy tới ngưỡng val\_acc bnhiu

Lossfunction là cái gì?

Nhóm thực hiện Train trên các backbone với yêu cầu val\_acc đến ngưỡng > 0.9 thì có thể dừng. Trong quá trình training, với tất cả backbone, nếu không dùng phương pháp motion embedded thì chỉ cần đến 10 epoch là đã đạt được val acc yêu cầu, còn ngược lại khi dùng phương pháp motion embedded thì phải cần đến 50 epoch để đạt được 0.9

1. ròi sau đó show 1 cái confusion map cho dataset seags (vì nó là dataset mới), show vài cái nó đúng, vài cái nó sai (kèm theo predictions score lun thì tốt m ạ)
2. show results chạy các backbone vs ảnh normal, motion, 2stream

Bảng tổng kết kết quả thí nghiệm:

//