Bài báo cáo đồ án CS231

Nhận diện hành động

Nhóm 4MH:

- Nguyễn Tư Thành Nhân – 20520079
- Lê Nhật Minh – 20520070
- Lê Nhật Huy – 20520056

GVHD: Mai Tiến Dũng

Quá trình báo cáo đồ án môn học

1 **STEP**

Giới thiệu bài toán

2 **STEP**

Hướng tiếp cận bài toán

3 **STEP**

Các mô hình thực nghiệm và kết quả nghiệm thu

4 **STEP**

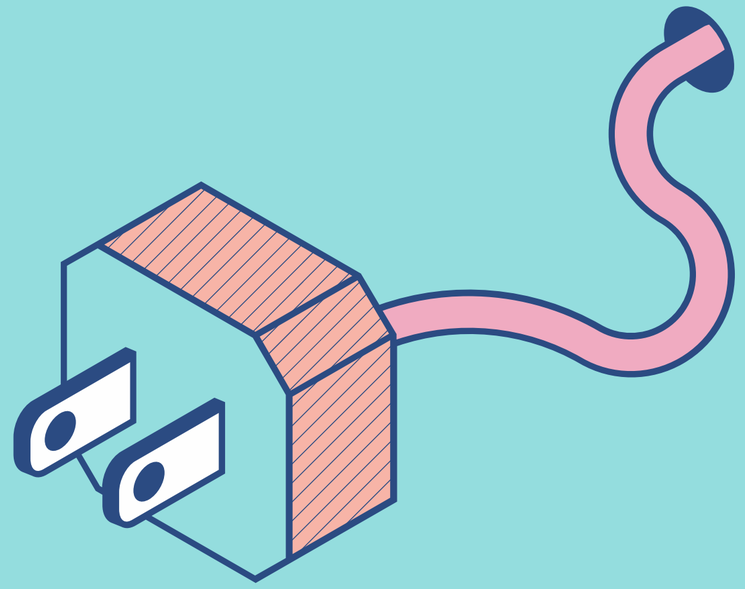
Một số khó khăn khi giải quyết bài toán

5 **STEP**

Các ý tưởng ứng dụng/cải thiện và giải đáp thắc mắc

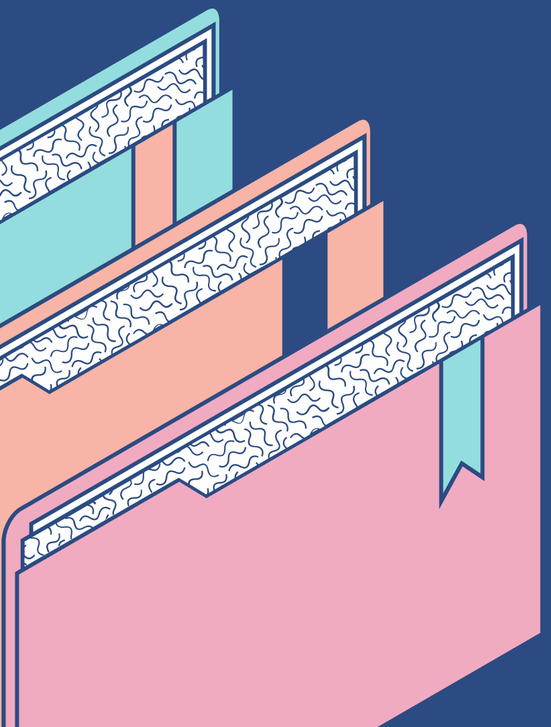
Giới thiệu bài toán

- **Nhận diện hành động là gì?**
- **Các tập dữ liệu**
- **Lựa chọn của chúng tôi**



Giới thiệu

NHẬN DIỆN HÀNH ĐỘNG LÀ GÌ?



Mục tiêu của nhận diện hành động

- Nhận diện được hành động của đối tượng là loại gì
- Nhận diện được mục đích hành động
- Xử lý được đa dạng các loại đối tượng:
  - người-người
  - nhiều người
  - một người
  - người-vật

Tự động hóa nhận diện hành động rất thách thức

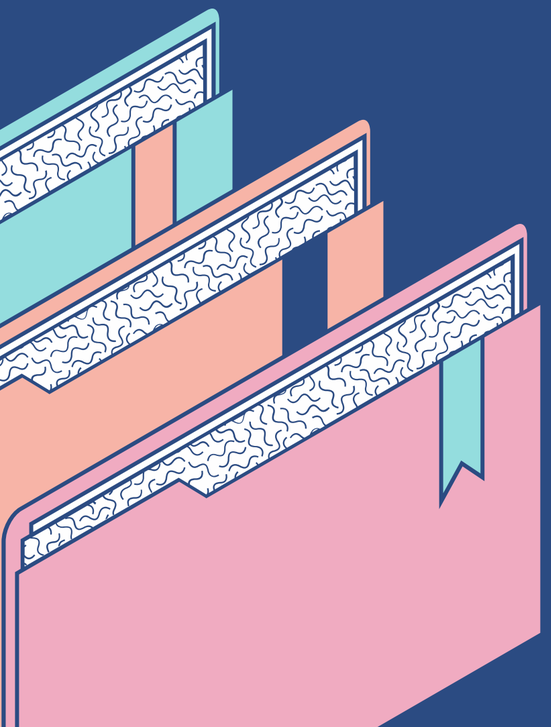
- Một người có thể làm nhiều việc cùng một lúc
- Các hành động xen kẽ nhau: đang nấu ăn phải nghe điện thoại
- Xác định những hành động giống nhau theo các cách khác nhau: mở cánh cửa và lau cánh cửa
- Có nhiều người làm nhiều việc khác nhau trong hình

Ứng dụng tiềm năng của nhận diện hành động

- Trợ giúp người lớn tuổi
- Tương tác người-máy
- Xây dựng các hệ thống giám sát, an ninh

Giới thiệu

MỘT SỐ TẬP DỮ LIỆU DÀNH CHO BÀI TOÁN



UCF-101

- 101 lớp hành động
- Hơn 13000 clips và 27 giờ dữ liệu video
- Các lớp hành động đa dạng, thuộc các loại sau:
  - Tương tác người-vật(Thổi nến)
  - Chuyển động cơ thể(Nhảy cao)
  - Tương tác người-người(Đấm)
  - Sử dụng nhạc cụ(Chơi đàn)
  - Thể thao(Đá bóng)

HMDB51

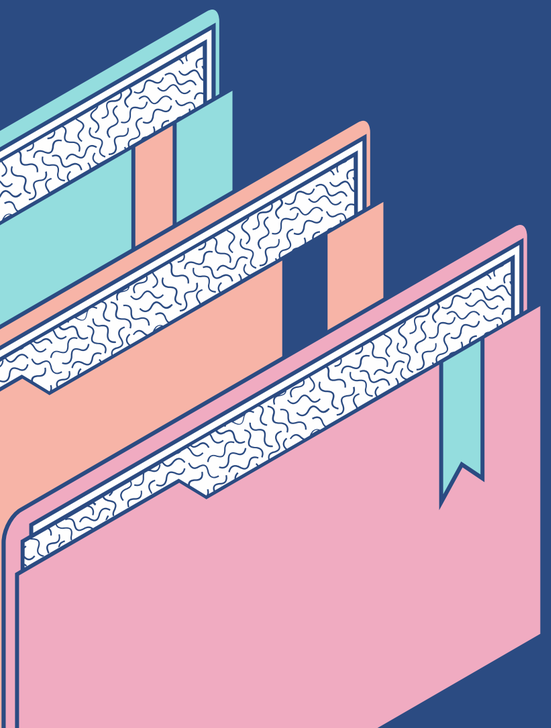
- Là một tập dữ liệu chứa các video thực tế từ nhiều nguồn
- Bao gồm 6849 video clips và 51 lớp hành động
- Mỗi lớp hành động bao gồm tối thiểu 101 clips

KINETICS400

- 400 lớp hành động, với ít nhất 400 video clips cho mỗi hành động
- Mỗi clip kéo dài khoảng 10s
- Các clip được lấy từ các đoạn trong các video khác nhau trên YouTube

Giới thiệu

LỰA CHỌN
CỦA CHÚNG TÔI



Tập dữ liệu

- Một tập dữ liệu nhỏ hơn được lấy từ Kinetics400
- Bao gồm:
  - 4836 videos train, 401 videos test, 771 videos validate
  - 10 class
  - Mỗi class trong có ít nhất gồm 309 videos
- Video có kích thước trung bình 480x360, 30fps, 10s
- Được thành ba tập train, test và validate
- 10 class được chọn bao gồm: bandaging, bowling, breakdancing, ironing, kissing, riding scooter, side kick, tap dancing, texting, washing hair

---

Đầu vào/ra

- Đầu vào: Một video có kích thước 480x360, 30fps, 10s
- Đầu ra: Action class của video

Nền tảng huấn luyện và thử nghiệm

- Google Colaboratory

Giới thiệu

TẬP DỮ LIỆU

Bandaging
Train: 356 videos
Validation: 41 videos
Test: 83 videos

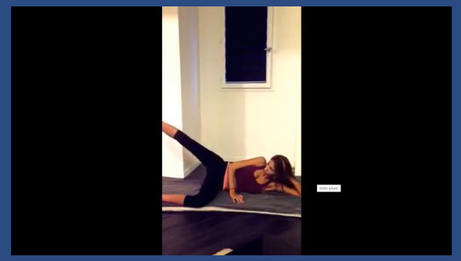
Bowling
Train: 793 videos
Validation: 41 videos
Test: 88 videos

Breakdancing
Train: 671 videos
Validation: 44 videos
Test: 82 videos

Ironing
Train: 300 videos
Validation: 39 videos
Test: 84 videos

Kissing
Train: 265 videos
Validation: 25 videos
Test: 25 videos

Riding scooter
Train: 414 videos
Validation: 39 videos
Test: 86 videos



Side kick
Train: 698 videos
Validation: 47 videos
Test: 89 videos

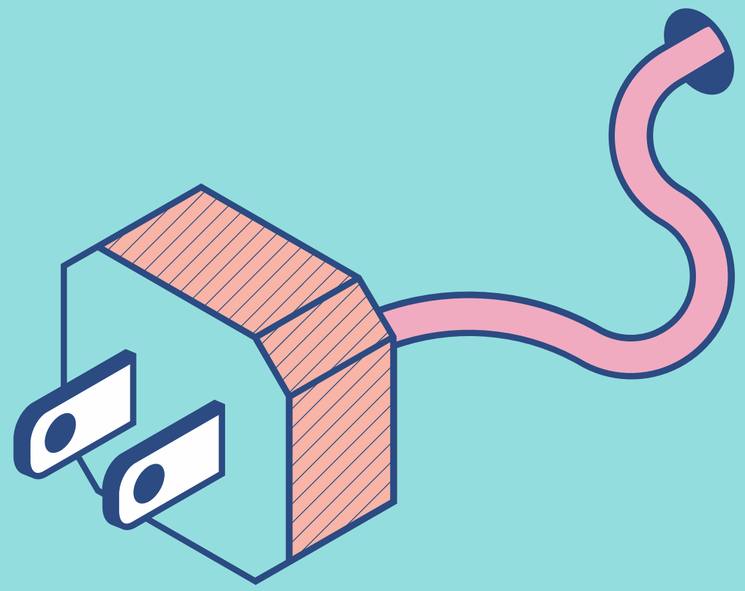
Tap dancing
Train: 699 videos
Validation: 40 videos
Test: 82 videos

Texting
Train: 445 videos
Validation: 43 videos
Test: 80 videos

Washing hair
Train: 195 videos
Validation: 42 videos
Test: 72 videos

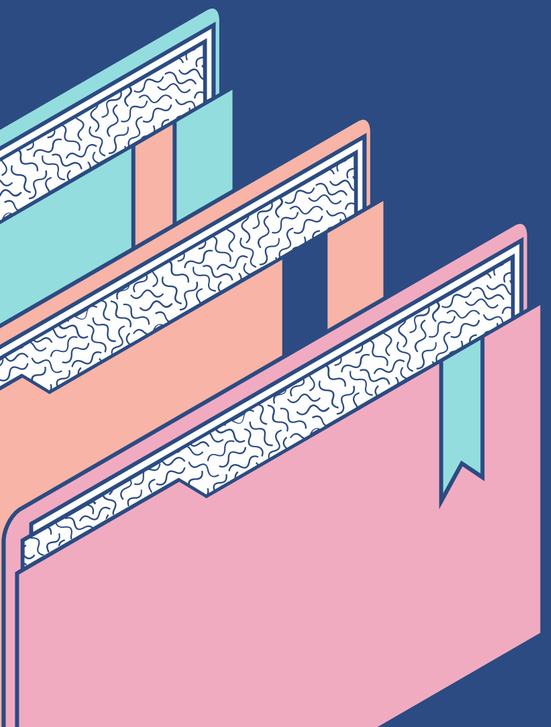
Hướng tiếp cận

- Nhận xét về dataset
- Mô hình giải quyết bài toán
- Kết quả sơ lược của các mô hình



Hướng tiếp cận

NHẬN XÉT DATASET



Tập dữ liệu

- 10 class được chọn bao gồm: bandaging, bowling, breakdancing, ironing, kissing, riding scooter, side kick, tap dancing, texting, washing hair
- Một số hành động tương tác người-vật có thể nhận diện từ vật thể mà con người tương tác: bandaging, bowling, riding scooter, ironing, texting, washing hair
- Các hành động phân biệt dựa vào tương tác người-người, một người: breakdancing, kissing, side kick, tap dancing

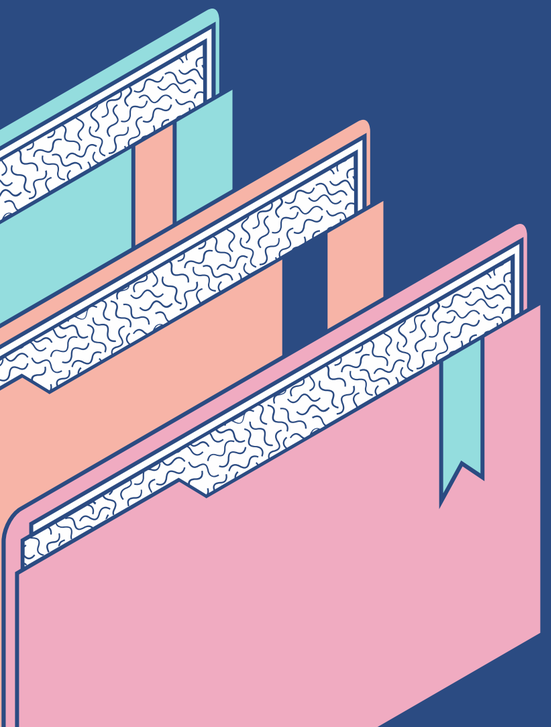
---

Ý tưởng

- Sử dụng các mô hình đã hiệu quả trong trích xuất đặc trưng của đối tượng trong ảnh: Resnet, Densenet với trọng số được train sẵn trên tập ImageNet

Hướng tiếp cận

MÔ HÌNH GIẢI QUYẾT BÀI TOÁN



Quy trình thực hiện

- Video sẽ được tách ra thành các frames kích thước 224 * 224 * 3. Lấy 16 frames mỗi video trong hướng tiếp cận 1 và 2, 64 frames mỗi video trong hướng tiếp cận 3.
- Sử dụng một mô hình chiết xuất đặc trưng ảnh lên 16/64 frames, thu được 16/64 vector đặc trưng với mỗi video
- Sử dụng một mô hình classification để phân loại và đưa ra kết quả

Tại sao mỗi video lấy 16 frames trong hướng tiếp cận 1, 2

- Mỗi video kéo dài 10 giây. Nếu lấy 16 frames mỗi giây, mỗi giây sẽ trung bình có 1.6 frame đại diện. Trong các mô hình liên quan đến video, người ta thường lấy mẫu mỗi giây 2 - 5 frames
- Thời gian để tạo ra feature với DenseNet201 và ResNet50 lần lượt là 3 tiếng và 2.5 tiếng

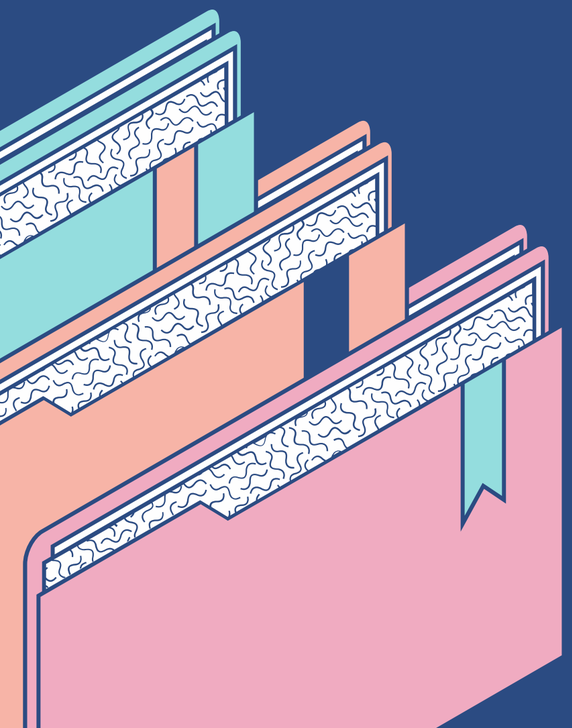
---

Các mô hình đã thí nghiệm

- Chiết xuất ảnh bằng DenseNet201, và phân loại bằng SVM + Biểu quyết đa số
- Chiết xuất ảnh bằng ResNet50/DenseNet201, và phân loại dựa trên các mô hình phân loại chuỗi:
  - Fully Connected
  - LSTM
  - GRU
  - Conv1D
- Sử dụng mô hình chiết xuất đặc trưng video Slowfast(state of the art, 2018) và các lớp fully connected để phân loại

Hướng tiếp cận

KẾT QUẢ SƠ LƯỢC CỦA CÁC MÔ HÌNH



DenseNet + SVM + Biểu quyết đa số

- Accuracy: 78%

ResNet/DenseNet + FC/LSTM/Conv1D

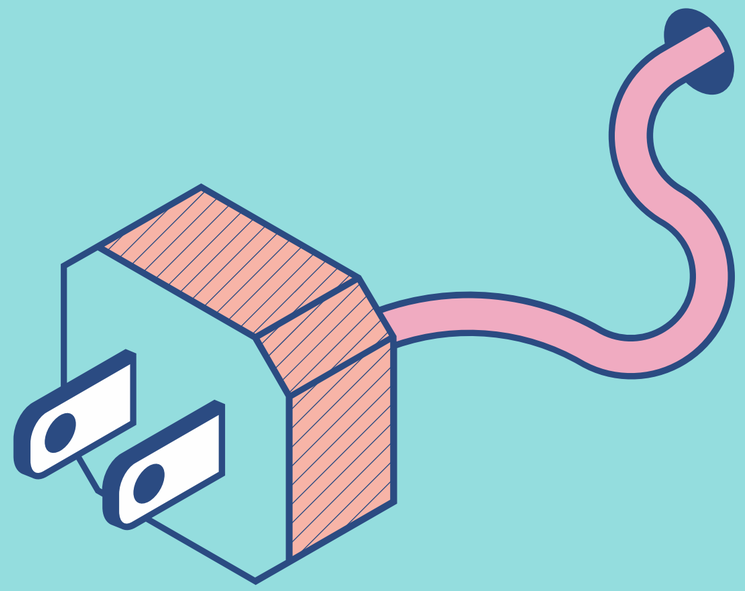
- Accuracy(FC): 29% / 80%
- Accuracy(LSTM, best tried): 38% / 82%
- Accuracy(GRU): 37% / 81%
- Accuracy(Conv1D, best tried): 30% / 80%

(Augmented Data) Slowfast

- Accuracy with non-augmented dataset: 91%
- Accuracy with augmented dataset: 92%

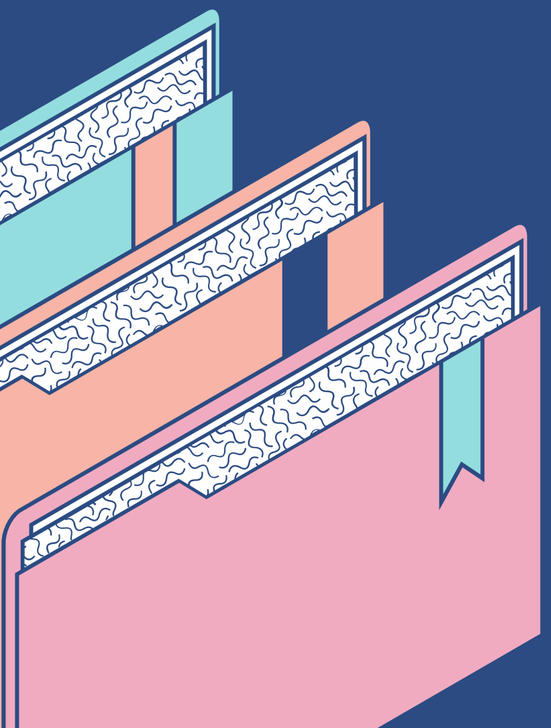
Các mô hình và kết quả nghiệm thu

- Mô hình chiết xuất đặc trưng ảnh
- Mô hình phân loại
- Các cặp mô hình chiết xuất và phân loại
- Mô hình Slowfast



Các mô hình thực nghiệm

MÔ HÌNH TRÍCH XUẤT ĐẶC TRƯNG ẢNH



Các mô hình trích xuất đặc trưng ảnh

- ResNet (ResNet50)
- DenseNet (DenseNet201)

Lí do lựa chọn

- ResNet và DenseNet là hai mô hình nổi tiếng hiểu quả trong việc phát hiện và trích xuất đặc trưng của đối tượng trong ảnh, được ứng dụng trong nhiều cuộc thi, nghiên cứu trong thời gian gần đây và cho hiệu suất tốt.

RESNET

Giới thiệu về ResNet

ResNet (Residual Network) được đề xuất lần đầu năm 2015 bởi nhóm tác giả Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun [*]. ResNet khiến cho việc huấn luyện hàng trăm thậm chí hàng nghìn lớp của mạng nơ ron trở nên khả thi và hiệu quả.

Nhờ khả năng biểu diễn mạnh mẽ của ResNet, hiệu suất của nhiều ứng dụng thị giác máy, không chỉ các ứng dụng phân loại hình ảnh được tăng cường. Một số ví dụ có thể kể đến là các ứng dụng phát hiện đồ vật và nhận dạng khuôn mặt.

[*] https://arxiv.org/abs/1512.03385

RESNET (RESNET-50)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

RESNET (RESNET-50)

```
[40] resnet_model = Sequential()

     pretrained_model= tf.keras.applications.ResNet50(include_top=False,
                       input_shape=(IMAGE_HEIGHT, IMAGE_WIDTH, 3),
                       pooling='avg',
                       weights='imagenet')
     for layer in pretrained_model.layers:
             layer.trainable=False

     resnet_model.add(pretrained_model)
```

```
resnet_model.summary()
```

```
Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

=================================================================
Total params: 23,587,712
Trainable params: 0
Non-trainable params: 23,587,712
_________________________________________________________________
```

Input và Output

- Input: 1 frame kích thước 224 * 224 * 3
- Output: 1 feature vector kích thước 2048

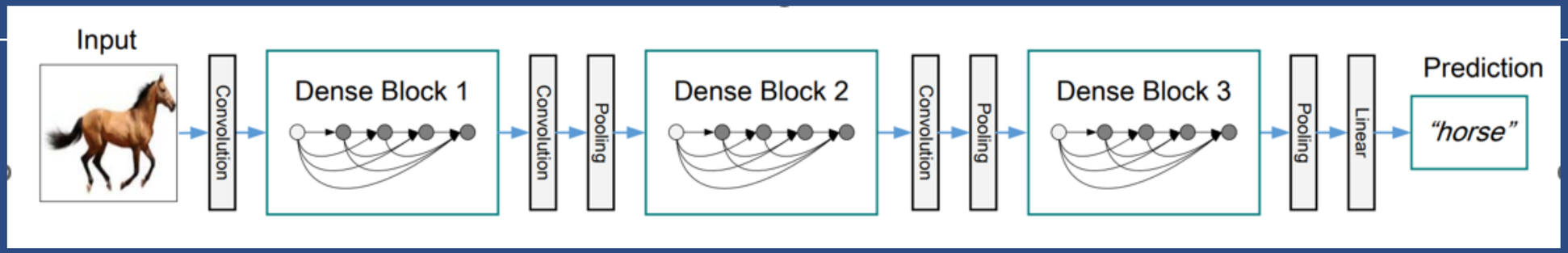
DENSENET

Giới thiệu về DenseNet

Densenet (Dense connected convolutional network) được công bố trong CVPR2017 bởi nhóm tác giả Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger [*], là một trong những netwok mới nhất cho nhận diện đối tượng.

Densenet có cấu trúc gồm các dense block và các transition layers. Các lớp mạng trong một denseblock được kết nối dày đặc với nhau. Các lớp mạng trong Với CNN truyền thống nếu chúng ta có L layer thì sẽ có L connection, còn trong densenet sẽ có L(L+1)/2 connection.

[*] https://arxiv.org/abs/1608.06993



DENSENET (DENSENET-201)

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

DENSENET (DENSENET-201)

```
[38] densenet_model = Sequential()

     pretrained_model= tf.keras.applications.DenseNet201(include_top=False,
                       input_shape=(IMAGE_HEIGHT, IMAGE_WIDTH, 3),
                       pooling='avg',
                       weights='imagenet')
     for layer in pretrained_model.layers:
             layer.trainable=False

     densenet_model.add(pretrained_model)
```

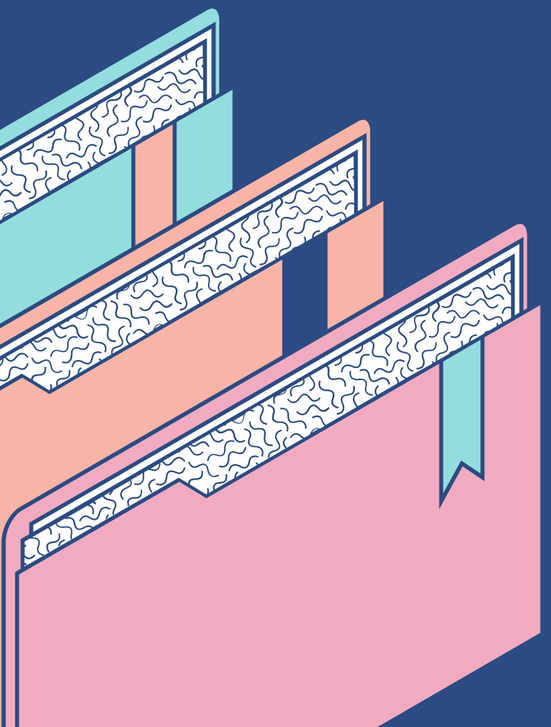
```
[39] densenet_model.summary()

     Model: "sequential_10"
     _________________________________________________________________
      Layer (type)                 Output Shape              Param #
     =================================================================
      densenet201 (Functional)     (None, 1920)              18321984

     =================================================================
     Total params: 18,321,984
     Trainable params: 0
     Non-trainable params: 18,321,984
     _________________________________________________________________
```

Input và Output

- Input: 1 frame kích thước 224 * 224 * 3
- Output: 1 feature vector kích thước 1920

Các mô hình thực nghiệm

MÔ HÌNH PHÂN LOẠI



Các mô hình phân loại đã được thí nghiệm trong hướng tiếp cận 1 và 2

- Hướng tiếp cận 1:
  - SVM + Biểu quyết đa số
- Hướng tiếp cận 2:
  - RNN:
    - 1 lớp LSTM + Các lớp Fully Connected
    - 1 lớp GRU + Các lớp Fully Connected
    - 3 lớp LSTM
  - Conv1D:
    - 1 lớp Conv1D + Các lớp Fully Connected
    - Nhiều lớp Conv1D + 1 lớp Fully Connected

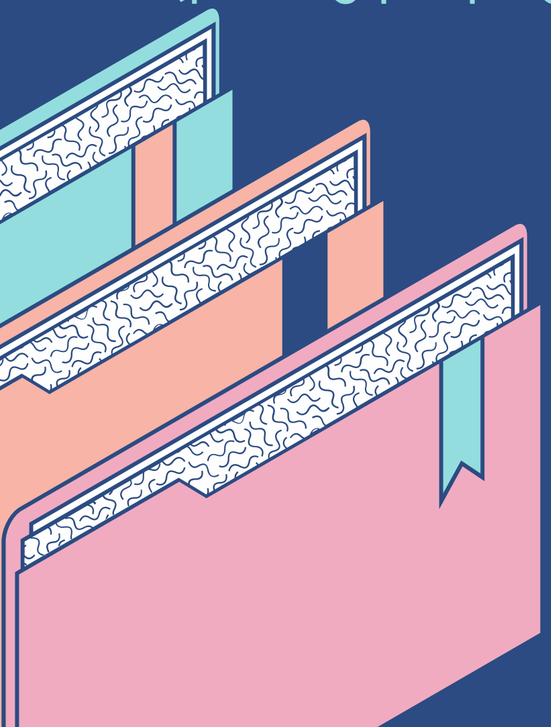
Notes

- Cấu trúc cũng như hyperparameter của các mô hình chưa được tới ưu nhiều

---

Các mô hình thực nghiệm

MÔ HÌNH PHÂN LOẠI: SVM+Biểu quyết đa số (phương pháp ngây thơ)



Ý tưởng

- Dựa vào quan sát trên tập dữ liệu, nhận thấy có thể dễ dàng phân loại một số video kiểu tương tác người vật thuộc lớp nào chỉ dựa trên vật
- Từ ý tưởng này ta có thể đưa bài toán nhận diện hành động về bài toán phân loại hình ảnh quen thuộc
- Vì còn nhiều hành động khác phức tạp hơn, nên mô hình phân loại nên sử dụng một loại phi tuyến tính

Quá trình thực hiện

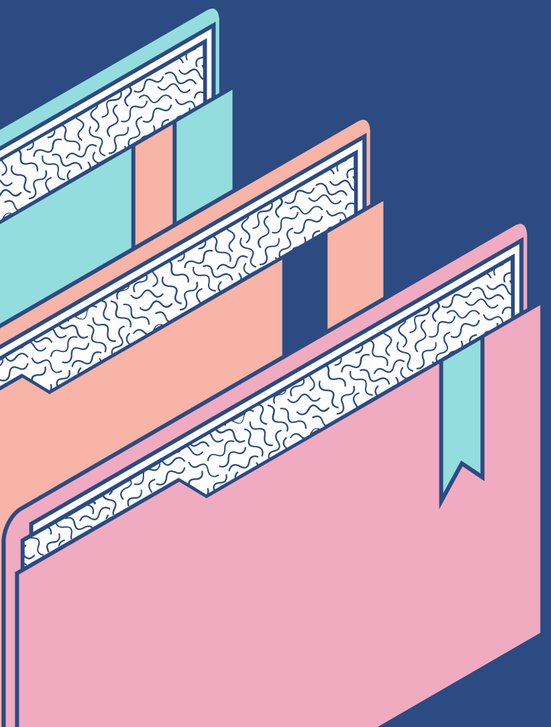
- Gán mỗi ảnh trong tập các video với nhãn là nhãn của video chứa ảnh
- Đưa các ảnh qua mô hình chiết xuất đặc trưng ảnh(DenseNet) và cho mô hình phân loại học
- Sử dụng mô hình phân loại để phân loại từng ảnh(hoặc một lượng ảnh nhất định) trong video, nhãn nào được mô hình chọn nhiều nhất sẽ là nhãn của video.

Hạn chế và hướng cải thiện

- Khó phân loại được các nhãn thuộc chuyển động cơ thể
- Không tận dụng được yếu tố thời gian của mô hình
- Nếu có thể xem quá trình biểu quyết đa số như là một RNN với memory là bảng đếm, thì cận dưới của phương pháp sử dụng RNN là đếm.

Các mô hình thực nghiệm

MÔ HÌNH PHÂN LOẠI: RNN / Conv1D



Ý tưởng

- Sử dụng các loại mô hình Fully Connected, RNN và Conv1D nhằm để mô hình hóa trục thời gian
- Nếu sử dụng mô hình RNN/Conv1D thì có thể mô hình hóa được tính gần xa giữa các frame theo trục thời gian

---

Quá trình thực hiện

- Thử thay thế quá trình đếm bằng một số mạng với các cấu trúc và tham số khác nhau
- Chọn ra các cấu trúc cho độ chính xác cao nhất trên tập test
- Chạy thử và lấy độ chính xác trên tập validation

Hạn chế và hướng cải thiện

- Khó phân loại được các nhãn thuộc chuyển động cơ thể

Các mô hình thực nghiệm

DenseNet + SVM + BQ đa số

Label	Precision	Recall	F1
Bandaging			79%
Bowling			88%
Breakdancing			69%
Ironing			85%
Kissing			58%
Riding scooter			86%
Side kick			78%
Tap dancing			82%
Texting			83%
Washing hair			72%

Các mô hình thực nghiệm

DenseNet + Fully Connected (SGD optimizer generalize better)

```
classification_model = Sequential([
                                    Flatten(),
                                    Dense(512, activation='relu'),
                                    Dense(256, activation='relu'),
                                    Dense(10),
                                    Softmax()])
```

Label	Precision	Recall	F1
Bandaging	85%	87%	86%
Bowling	95%	88%	92%
Breakdancing	68%	73%	71%
Ironing	92%	81%	86%
Kissing	51%	76%	61%
Riding scooter	78%	88%	83%
Side kick	77%	67%	72%
Tap dancing	78%	68%	73%
Texting	86%	81%	83%
Washing hair	78%	88%	82%

Các mô hình thực nghiệm

DenseNet + 1 layer LSTM + Fully Connected

```
classification_model = Sequential([LSTM(1024),
                                   Flatten(),
                                   Dense(512, activation='relu'),
                                   Dense(256, activation='relu'),
                                   Dense(10),
                                   Softmax()])
```

Label	Precision	Recall	F1
Bandaging	86%	84%	85%
Bowling	95%	93%	94%
Breakdancing	64%	74%	69%
Ironing	91%	87%	89%
Kissing	60%	84%	70%
Riding scooter	80%	86%	83%
Side kick	79%	70%	75%
Tap dancing	79%	70%	74%
Texting	87%	85%	86%
Washing hair	85%	83%	84%

Các mô hình thực nghiệm

DenseNet + GRU +
Fully connected

```
classification_model = Sequential([GRU(1024),
                                   Flatten(),
                                   Dense(512, activation='relu'),
                                   Dense(256, activation='relu'),
                                   Dense(10),
                                   Softmax()])
```

Label	Precision	Recall	F1
Bandaging	85%	88%	86%
Bowling	93%	93%	93%
Breakdancing	60%	76%	67%
Ironing	82%	89%	86%
Kissing	76%	76%	76%
Riding scooter	81%	87%	84%
Side kick	80%	64%	71%
Tap dancing	78%	71%	74%
Texting	92%	81%	86%
Washing hair	88%	85%	87%

Các mô hình thực nghiệm

DenseNet + 1 layer Conv1D + Fully connected

```
classification_model = Sequential([BatchNormalization(),
                                   Conv1D(128, 5, padding='same'),
                                   Flatten(),
                                   Dense(512, activation='relu'),
                                   Dense(256, activation='relu'),
                                   Dense(10),
                                   Softmax()])
```

Label	Precision	Recall	F1
Bandaging	85%	81%	83%
Bowling	97%	88%	93%
Breakdancing	66%	77%	71%
Ironing	88%	86%	87%
Kissing	54%	80%	65%
Riding scooter	82%	87%	84%
Side kick	66%	66%	66%
Tap dancing	77%	67%	72%
Texting	93%	82%	88%
Washing hair	81%	83%	82%

Về độ chính xác của ResNet50 ở cách tiếp cận 2

Ở cách tiếp cận 2, chúng tôi đã sử dụng ResNet để chiết xuất đặc trưng và kết quả của nó thật sự tệ nên chúng tôi có hai giả thuyết cho vấn đề này

Hoặc là mô hình phân loại của chúng tôi không có khả năng phân loại, hoặc là feature của ResNet50 không phù hợp với bài toán này?

Cùng một kiến trúc phân loại, mô hình dùng DenseNet có thể phân loại được với độ chính xác lên tới 83%

Về độ chính xác của ResNet50 ở cách tiếp cận 2

Pre-Activation ResNet is used in detailed comparison.

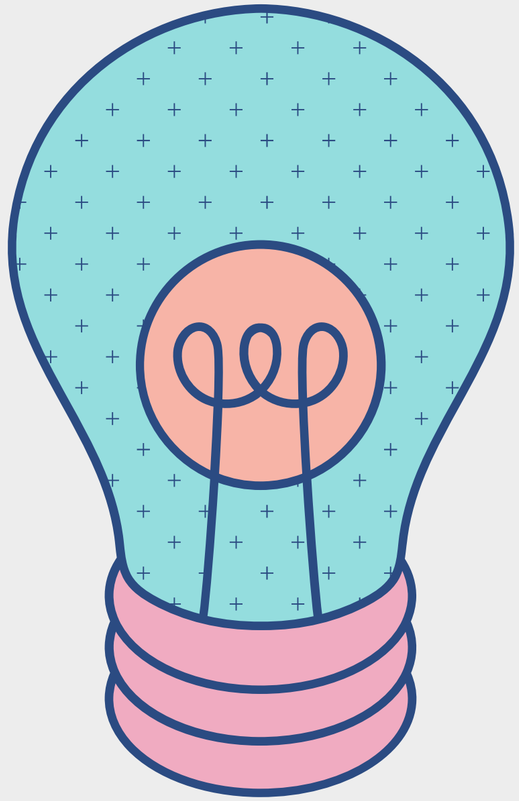
With data augmentation (C10+), test error:

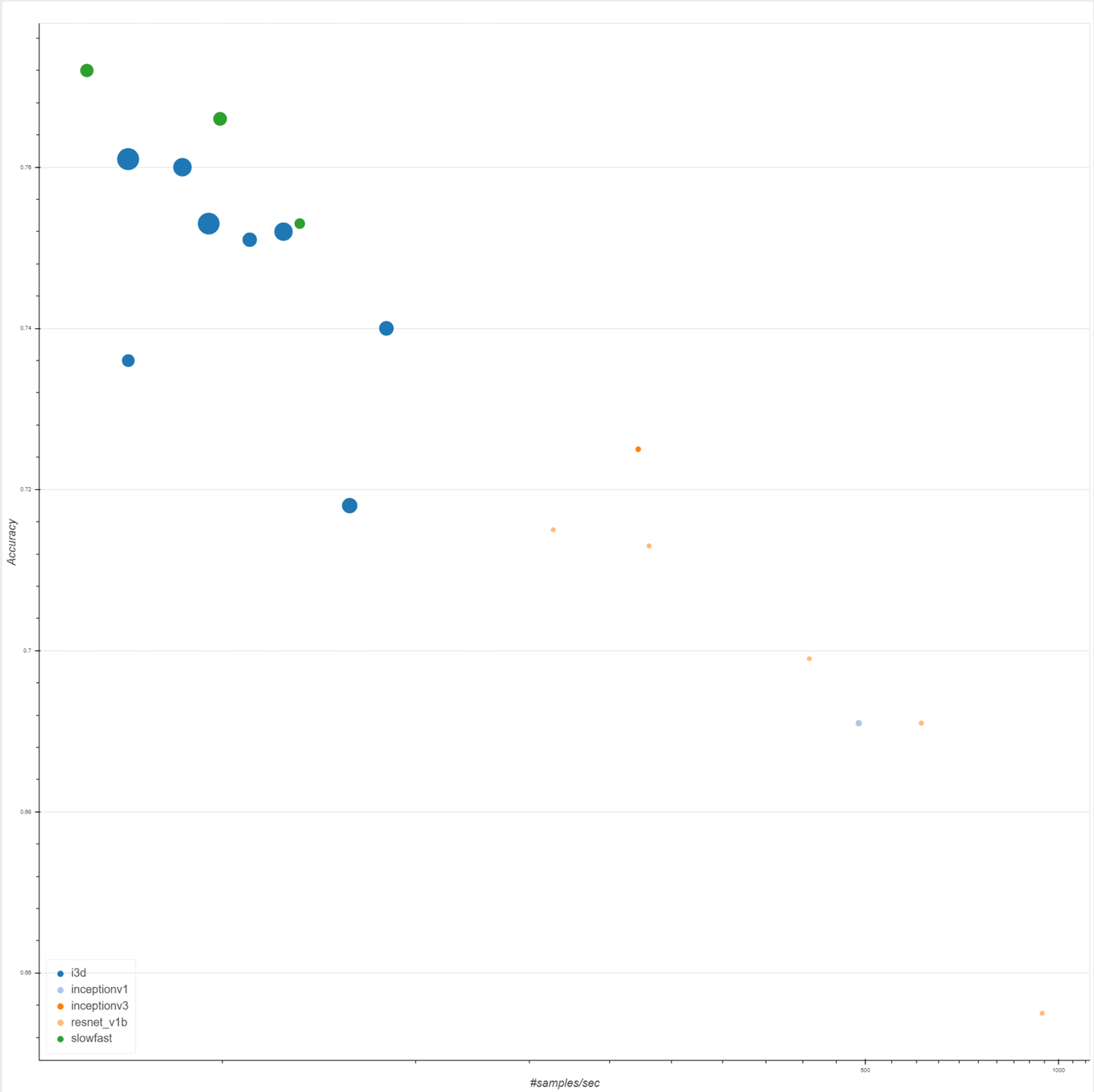
- Small-size ResNet-110: 6.41%
- Large-size ResNet-1001 (10.2M parameters): 4.62%
- State-of-the-art (SOTA) 4.2%
- Small-size DenseNet-BC ($L=100$, $k=12$) (Only 0.8M parameters): 4.5%
- Large-size DenseNet ($L=250$, $k=24$): 3.6%

Without data augmentation (C10), test error:

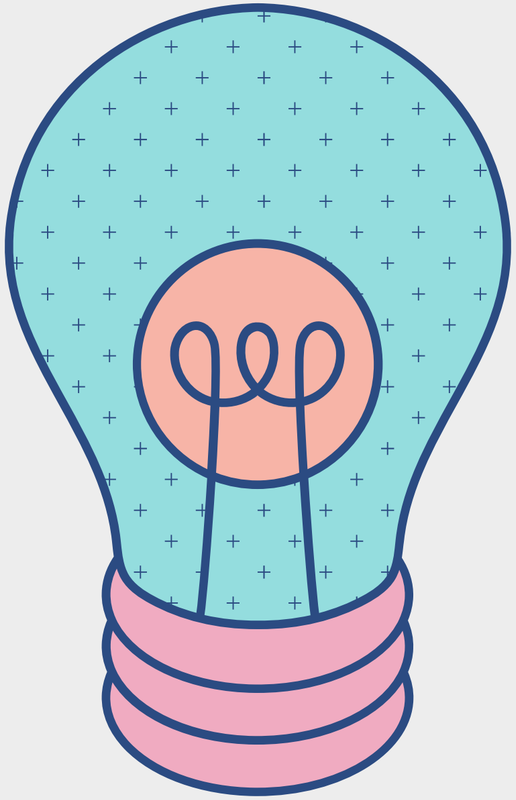
- Small-size ResNet-110: 11.26%
- Large-size ResNet-1001 (10.2M parameters): 10.56%
- State-of-the-art (SOTA) 7.3%
- Small-size DenseNet-BC ($L=100$, $k=12$) (Only 0.8M parameters): 5.9%
- Large-size DenseNet ($L=250$, $k=24$): 4.2%

SlowFast




0.76
0.74
0.72
0.7
0.68
0.66
Accuracy
500
1000
#samples/sec
i3d
inceptionv1
inceptionv3
resnet_v1b
slowfast

SlowFast



Input và Output của SlowFast

- Input: Tập sample có kích thước 224 * 224 * 3 * 64
- Output: Tập sample có kích thước 2304

Một số ưu điểm của SlowFast

- SlowFast được train trên tập Kinetic400 nên kết quả trích xuất tốt hơn
- SlowFast có độ chính xác khá cao và tốc độ khá nhanh so với các SOTA ra mắt trước đó
- Sử dụng mạng 2 luồng: Fath pathway và Slow pathway

Một số nhược điểm

- Chưa finetune được SlowFast mà chỉ sử dụng lại trích xuất đặc trưng được pretrain trên tập Kinetics400 (do tài nguyên có hạn)

SlowFast

stage	*Slow* pathway	*Fast* pathway	output sizes $T \times S^2$
raw clip	-	-	64×224^2
data layer	stride 16, 1^2	stride **2**, 1^2	*Slow* : 4×224^2 *Fast* : 32×224^2
conv_1	1×7^2, 64 stride 1, 2^2	5×7^2, 8 stride 1, 2^2	*Slow* : 4×112^2 *Fast* : 32×112^2
pool_1	1×3^2 max stride 1, 2^2	1×3^2 max stride 1, 2^2	*Slow* : 4×56^2 *Fast* : 32×56^2
res_2	$\begin{bmatrix} 1 \times 1^2, 64 \\ 1 \times 3^2, 64 \\ 1 \times 1^2, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 3 \times 1^2, 8 \\ 1 \times 3^2, 8 \\ 1 \times 1^2, 32 \end{bmatrix} \times 3$	*Slow* : 4×56^2 *Fast* : 32×56^2
res_3	$\begin{bmatrix} 1 \times 1^2, 128 \\ 1 \times 3^2, 128 \\ 1 \times 1^2, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 3 \times 1^2, 16 \\ 1 \times 3^2, 16 \\ 1 \times 1^2, 64 \end{bmatrix} \times 4$	*Slow* : 4×28^2 *Fast* : 32×28^2
res_4	$\begin{bmatrix} 3 \times 1^2, 256 \\ 1 \times 3^2, 256 \\ 1 \times 1^2, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 3 \times 1^2, 32 \\ 1 \times 3^2, 32 \\ 1 \times 1^2, 128 \end{bmatrix} \times 6$	*Slow* : 4×14^2 *Fast* : 32×14^2
res_5	$\begin{bmatrix} 3 \times 1^2, 512 \\ 1 \times 3^2, 512 \\ 1 \times 1^2, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 3 \times 1^2, 64 \\ 1 \times 3^2, 64 \\ 1 \times 1^2, 256 \end{bmatrix} \times 3$	*Slow* : 4×7^2 *Fast* : 32×7^2
global average pool, concate, fc			# classes

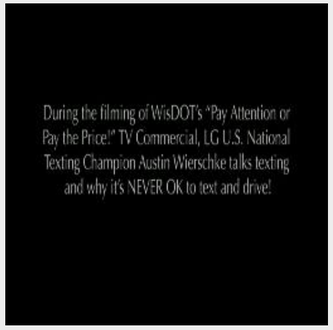
SlowFast

Predict: Riding scooter
Ground true: Texting

Predict: Breakdancing
Ground true: Side kick

Predict: Side kick
Ground true: Ironing

Predict: Ironing
Ground true: Bowling

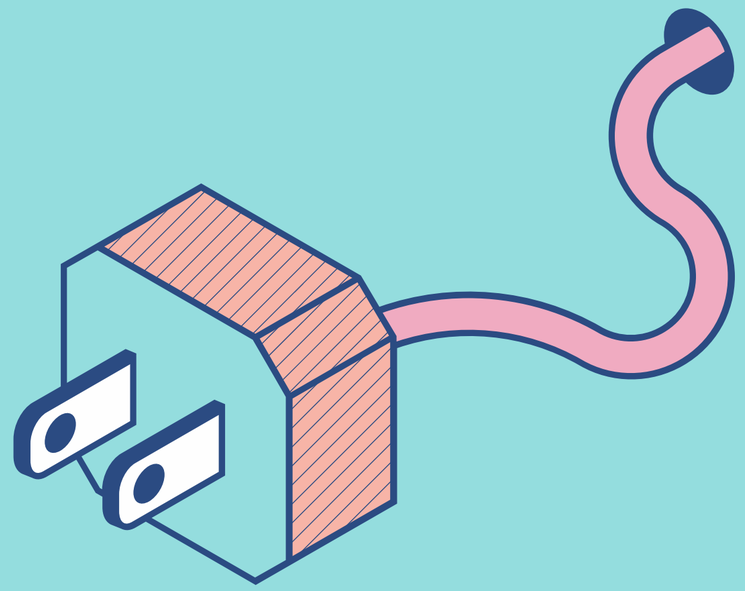


Predict: Kissing
Ground true: Texting

Predict: Tap dancing
Ground true: Kissing

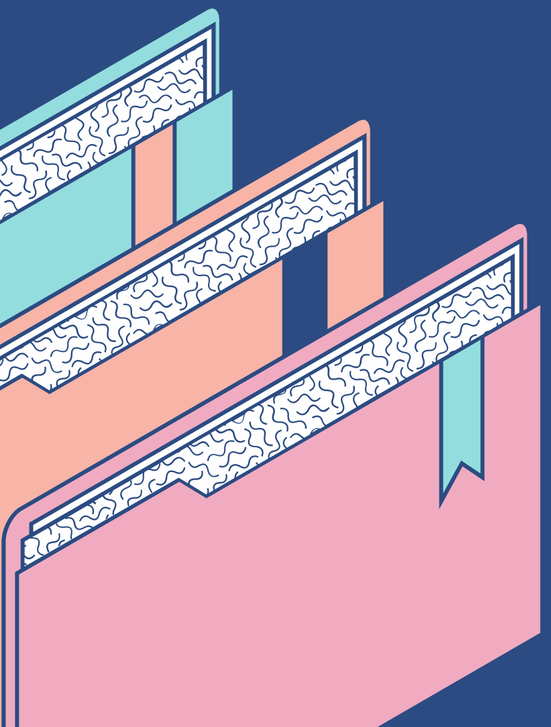
Một số khó khăn

- **Tài nguyên: bộ nhớ và thời gian**



Một số khó khăn

Tài nguyên



Mô hình chiết xuất đặc trưng ảnh trả về tensor kích thước quá lớn

- Sử dụng một lớp GlobalAveragePooling2D để tổng hợp lại các tensor

Không đủ RAM và thời gian để train SVM

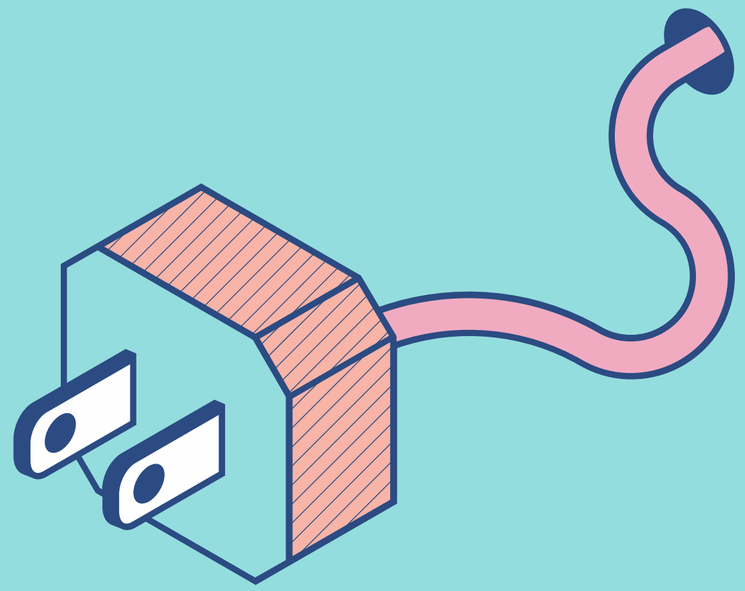
- Thời gian để huấn luyện mô hình tăng theo hàm bậc ba so với số lượng test
- Thu nhỏ tập train và có thể ensemble nhiều mô hình được huấn luyện bởi các tập train khác nhau.

Không đủ tài nguyên để fine tune lại bộ chiết xuất đặc trưng

- Không cần fine tune lại bộ chiết xuất đặc trưng
- Mỗi lần chạy qua hết các video mất 3 tiếng. Cần khoảng 30 epochs để đến được độ chính xác tối ưu

Hướng cải thiện

- Phát triển mô hình để xử lý chuỗi frame không đồng đều trên trục thời gian
- Sử dụng các phương pháp Ensemble như XGB và Light GBM để tăng độ chính xác
- Fine-tuning lại mô hình feature extraction ở phương pháp 1, 2 và Slowfast
- Kết hợp với đặc trưng từ Pose Estimation hay Body Segmentation



Do you have any questions?

Send it to us! We hope you learned something new.