Introduction to Deep Learning

Temporal Human Action Recognition

Group 4 - DSAI K65 - HUST

January 2, 2023

Our Team Members

Nguyen Quang Duc
20204876

Le Hong Duc
20204874

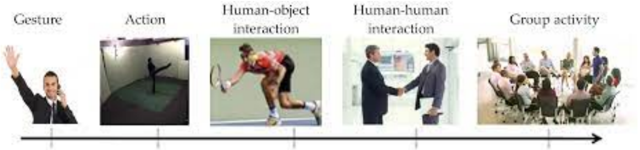
Tran Hoang Quoc Anh
20200044

La Dai Lam
20204918

Luu Trong Nghia
20204888

Table of contents

Introduction



- Temporal human action recognition is one of the most important applications in computer vision
- Our goal is to recognise the action in the short input video
- Using several deep-learning models and training them on two datasets UCF101 and HMDB51
- Challenges: large storage requirements, spatial and temporal combination

Datasets: Overview

Two benchmark datasets: HMDB51 and UCF101

HMDB51 - released in 2011

- Containing 6766 videos, divided into 51 distinct categories of action.
- Each class contains at least 101 videos
- Each video has the fixed width of 240 pixels and the frame rate of 30 frames per second.

UCF101 - released in 2012

- There are 13320 videos, categorized into 101 classes.
- Each type of action contains at least 100 videos
- 25 frames per second, with the identical resolution of 320×240.

Datasets: Examples

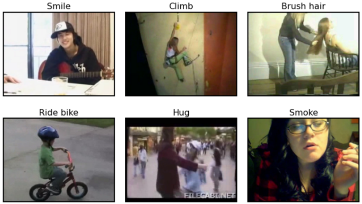


Figure 1: 6 video frames of HMDB51

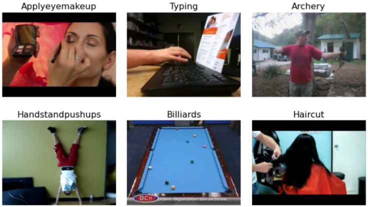


Figure 2: 6 video frames of UCF101

Figure 3: Pull Up video - UCF101

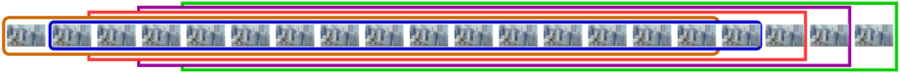
Datasets: Extracting Frames

Raw video

Uniform 5-frame

One 16-frame clip

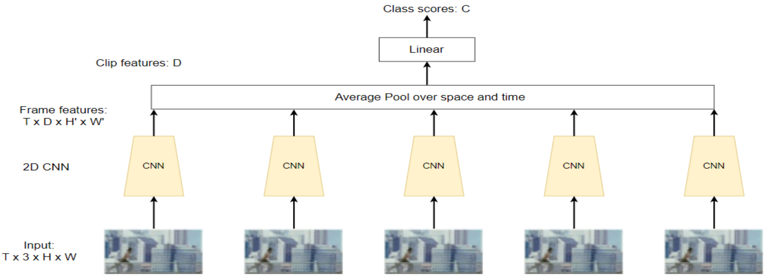
Five 16-frame clips



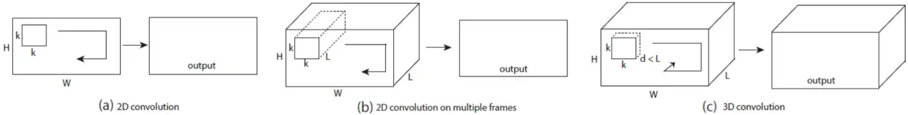
Modeling: Late Fusion

The Baseline Model

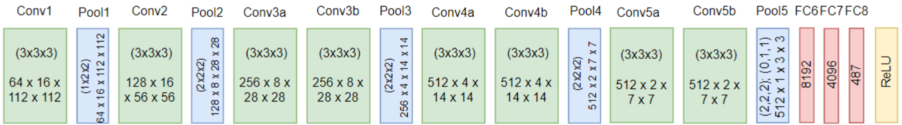
- Backbone 2D CNN architecture: Resnet-152
- The model does not consider temporal relations between frames
- Training time is fast.



3D Convolution Operation



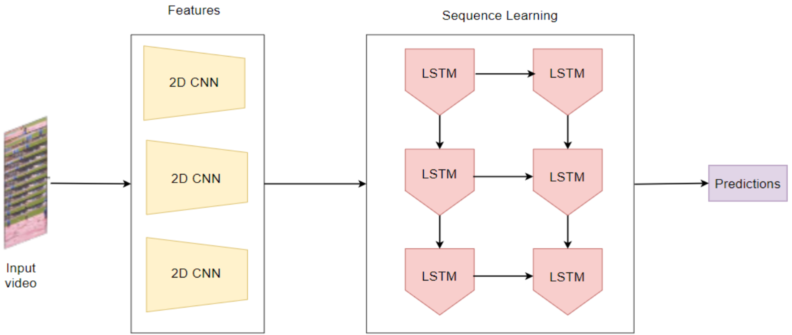
C3D: A 3D version of VGG16



Modeling: LRCN

Long-term Recurrent Convolutional Network

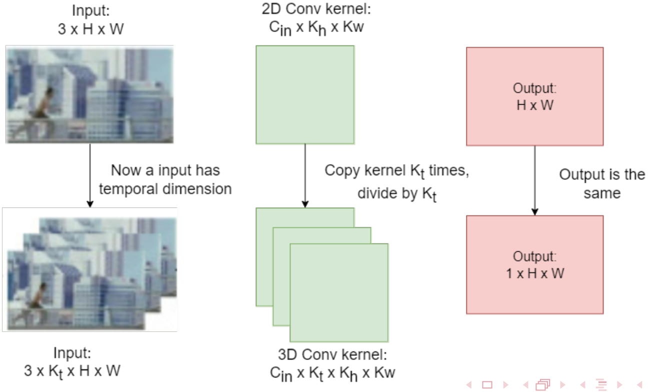
- Backbone 2D CNN architecture: Resnet-152
- Use RNN(s) to handle the temporal relations between frames
- But RNN layers make the training duration longer



Modeling: I3D

Inflated 3D Network: From 2D CNN to 3D CNN

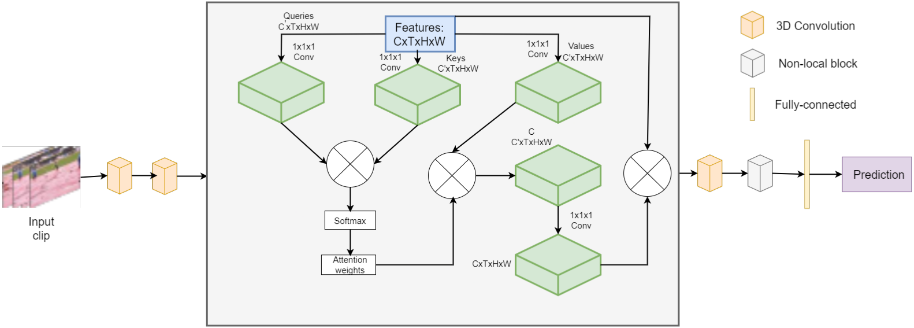
- Reuse the 2D CNN architecture and its pre-train weights
- Change in receptive fields



Modeling: Non-local Neural Networks

Non-local Neural Networks

- Implement self-attention to make Non-local blocks
- Use Non-local blocks inside existing 3D CNN architecture



Experiments: Training Strategies

Data Preparation, Augmentation

- Traning/Validation set: 80% and 20% based on class distribution
- Extracted frames based on models. For examples, sample 5 frames uniform for 2D-based model, 1-clip 16 frames for 3D-based model
- After trials and errors, augmentation technique: only use Resize and Normalization for all models

Experiments: Training Strategies

Data Preparation, Augmentation

- Traning/Validation set: 80% and 20% based on class distribution
- Extracted frames based on models. For examples, sample 5 frames uniform for 2D-based model, 1-clip 16 frames for 3D-based model
- After trials and errors, augmentation technique: only use Resize and Normalization for all models

Loss Function, Optimizer, Evaluation Metrics

- Loss Function: Cross Entropy Loss
- Evaluation Metrics: Accuracy, Confusion Matrix
- Optimizer and scheduler: Adam optimizers and Exponential Learning Rate scheduler: decrease the learning rate after one epoch

Experiments: Training Strategies

Models and hyperparameters settings

- Tools: Pytorch and Weight & Bias (Wandb)
- All models are pretrained.
- Hyperparameters such as learning rate, chosen through trails and errors

Experiments: Training Strategies

Models and hyperparameters settings

- Tools: Pytorch and Weight & Bias (Wandb)
- All models are pretrained.
- Hyperparameters such as learning rate, chosen through trails and errors

Training two dataset simultaneously

- Make use of all two datasets
- Feed the training model two different batches of data of two dataset and changing the last fully connected layer correspondingly

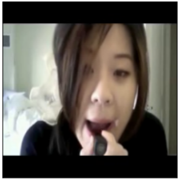
Experiments: Quantitative results

Models	Data Processing	Backbone / Pretrained	UCF101 Val_acc(%)	UCF101 Test_acc(%)	HMDB51 Val_acc(%)	HMDB51 Test_acc(%)
Training on dataset itself						
Late Fusion	5 frames	Resnet-152	95.44	74.19	61.33	41.62
C3D	16 frames clip	Sport-1M[3]	89.99	60.55	47.68	15.98
LRCN	5 frames	Resnet-152	96.61	74.50	65.94	43.87
I3D	16 frames clip	Resnet-50	93.27	66.73	59.59	36.47
Non-local	16 frames clip	Kinetics 400	**95.48**	**83.01**	**67.74**	**53.35**
Training on two datasets						
Late Fusion	5 frames	Resnet-152	92.49	72.85	63.78	37.63
C3D	16 frames clip	Sport-1M	89.18	60.50	56.03	32.34
Non-local	16 frames clip	Kinetics 400	**95.01**	**84.11**	**68.80**	**50.92**

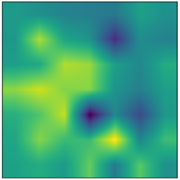
The results for each model for each dataset with two training strategies, on the dataset itself and two datasets

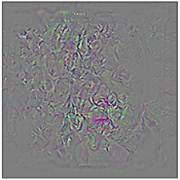
Experiments: Qualitative results

What I3D have learned through three techniques: Activation Map, Grad-Cam, and Guided Backpropagation

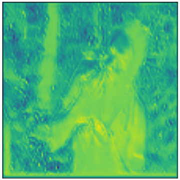

(a) Original image

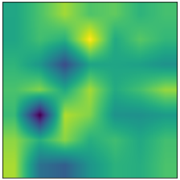

(b) Activation map

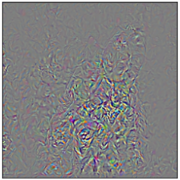

(c) Grad-CAM


(d) Guided Backpropagation


(a) Original image


(b) Activation map


(c) Grad-CAM


(d) Guided Backpropagation

Conclusion

Summary

- By using various methods and tools, we have trained five different models on two dataset benchmark for video recognition problem
- The best model among five ones is Non-local Neural Networks training on two datasets
- Besides quantitative results, we also analyse on qualtitative results based on visual explanation techniques

Possible extensions

- Research and implement new models like ViViT and new data preprocessing methods such as RandAugment
- Try to apply self-supervised and semi-supervised learning for this problem