Video Similarity Search

Ryan Bae, Hemant Jain, Nimisha Sharath
MS in Data Science, University of Washington

INTRODUCTION

BACKGROUND

Video content is becoming increasingly common and rising as the preferred media due to its ability to engage the user and communicate complex information in short periods of time. While websites such as YouTube allow search through millions of videos via keywords or queries, no such product exists that can search a video database by means of an input video.

OBJECTIVE

- Explore different methods for calculating video similarity by performing a vigorous literature survey and use machine learning to solve this problem.
- Create a prototype video search similarity engine that integrates these methods that we explored for a practical demonstration.

DATA

UCF101 - 13,200 videos dataset with 101 class labels. This is our main video corpus used for benchmarking video classification accuracies and similarity search.

In addition to the main query dataset, other datasets were used to train various models in our pipeline:

AudioSet - The dataset used to train Google's VGGish model for audio feature extraction. The dataset contains audio clips of around 10 seconds extracted from 2,084,320 YouTube videos with 527 class labels.

MSVD - This comprises of 1,970 YouTube clips with human annotated sentences.

Kinetics 400 - This dataset contains 500,000 videos with 400 class labels of various actions. The 3D CNN model was trained using this dataset.

ARCHITECTURE

- All the approaches taken require extracting individual frames from videos followed by extracting image features from each frame. This is a computationally expensive task which takes many hours on a CPU. (We use **ffmpeg** for this)
- The data preparation and model training for the supervised approach has been on an EC2 instance of type p3.xlarge of the Deep Learning AMI suite. **Python** has been used for the entire pipeline with **PyTorch, Keras, TensorFlow, faiss** and **OpenCV** for the neural network and image processing modules respectively.
- The web-app demo for the search engine was build using **flask**. The demo for inference can be run locally on the CPU and GPU (significantly faster).

PRACTICAL APPLICATIONS

There are many practical real-world applications of video similarity search technology. We list some of them below:

1. **Security Surveillance**: Automate the search of surveillance footage for suspicious activities and crimes.
2. **Medicine**: Detecting anomalous/similar activity in patients. E.g.: monitoring periods when the fetus behaves in a certain way from a sonogram.
3. **Media and Entertainment**: Find video which are similar to one another - recommendation engines for videos.
4. **Detecting Duplicate**: By detecting near duplicate videos to put an end to piracy and copyright infringement.
5. **Reverse Video Search:** Similar to Reverse-Image Search.

METHODS

Unsupervised Approach

The unsupervised approach is divided into 4 verticals, each based on different type of features extracted.

1. ResNet Features (Object level)

This method uses a pre-trained CNN model called *ResNet-50* (*PyTorch* implementation) to extract CNN features (**2048 dim.**) per frame sampled from the video (**8** frames). The similarity search is conducted across each frame, and then aggregated for the corresponding videos.

2. 3D ResNet Features (Action level)

This method uses a model that extends ResNet to work on 3D tensors instead of 2D (*PyTorch* implementation) by performing the convolution and pooling operations using 3D blocks. The model builds a 3D tensor by sampling **8** frames uniformly from a video and extracts a **512 dim.** feature vector.

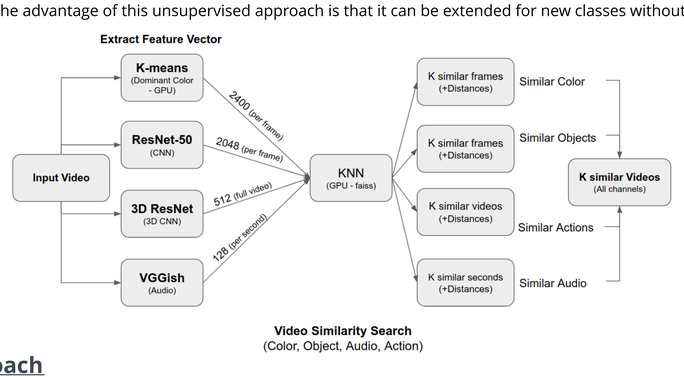
3. Dominant Color Features (Color level)

Feature reduction is done by creating a histogram for the **k** dominant colors per frame sampled from the video (**8**). We perform *k-means clustering* on each frame treating each pixel (RGB) as a data-point. These **k** (**10**) cluster centroids are the k dominant colors in that frame. We *weight* each dominant color by the size of the corresponding cluster and sort them (desc.) by cluster size. Each such histogram is merged to form a single image representation (**8x100x3 dim.**) for the video.

4. Audio Embeddings/Features (Audio level)

Google's Audioset project (*TensorFlow* implementation) for audio classification converts audio signals to images called ***log mel spectrograms***. They trained a special model - VGGish (inspired by VGG-Net) to classify these images and extract feature representations. This pre-trained model (open-sourced) is used to extract second-level feature vector (**128 dim.**) representations for the video.

After extracting the features, similarity search is conducted across each feature vertical using k-nearest neighbor (**KNN**) using the **faiss** (Facebook AI Similarity Search) library. The advantage of this unsupervised approach is that it can be extended for new classes without manual annotation.

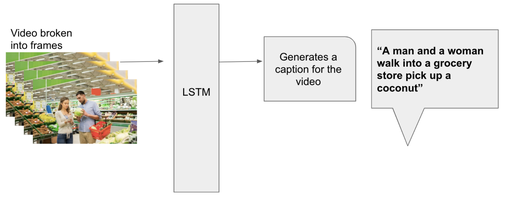


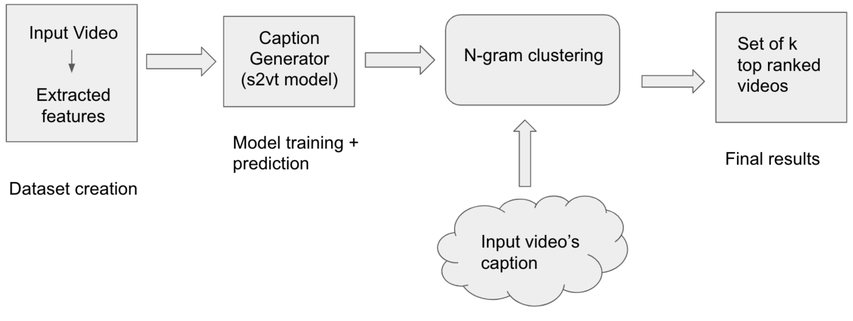
Video Similarity Search
(Color, Object, Audio, Action)

Supervised Approach

- This sequence to sequence model works with the MSVD dataset which has videos with descriptions.
- Uses the features from every image frame to generate text.
- The sequence of these image features are the input to an LSTM which will then produce a word at every time period.
- <video clip, sentence> becomes <[image features], [words]>
- The generated sentence for the input video is then clustered with the descriptions in the dataset using n-gram clustering
- Top k similar descriptions are found and the videos corresponding of each of these are pulled up as results
- Quantitative evaluation of the models are performed using the METEOR metric. The METEOR score is computed based on the alignment between a given hypothesis sentence and a set of candidate reference sentences.

  METEOR compares exact token matches, stemmed tokens, paraphrase matches.





RESULTS

Measuring performance of a similarity search was difficult, as it is an unsupervised problem. Because the UCF-101 dataset had labels of the videos, we used **top-3 classification accuracy** of the returned video labels to measure model performance.

In addition we measured the **feature extraction time for a single video** and **KNN query time** as additional performance metrics.

Method Vertical	Top-3 Classification Accuracy (101 Classes)	Per Video Feature Extraction Time (s)	KNN Query Time (s)
ResNet-50	91.6%	~1	0.16
3D ResNet	74.7%	~5	0.2
Dominant Colors	~20%	~7	0.3
VGGish Audio	63.5%	~2.5	0.2

CHALLENGES

- **Size of datasets**: all the datasets are > 30 GB in size. This certainly poses memory constraints on the machines initially available to us.
- **None of these datasets have features extracted** from them. Features are the values of the penultimate layers of common model architectures (e.g. VGG16) and this means we have to spend compute power on this.
- Adobe **videos are not annotated**. They have no descriptions and hence cannot be used for most of our study. The MSVD dataset is diverse and has descriptions which come close to the diverse nature of the videos present on the Adobe website.
- **Quality of annotations** are good, but not for all the videos. Since there are hundreds of clips and we cannot visually inspect all we have to accommodate for some noise in the model.

REFERENCES

[1] *"Sequence to Sequence - Video to Text"*, Venugopalan, Subhashini and Rohrbach, Marcus and Donahue, Jeff and Mooney, Raymond and Darrell, Trevor and Saenko, Kate, Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015

[2] Hara, Kensho, Hirokatsu Kataoka, and Yutaka Satoh. "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?." Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2018.

[3] Gemmeke, Jort F., et al. "Audio set: An ontology and human-labeled dataset for audio events." 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2017.

ACKNOWLEDGEMENTS

While this project was independently driven, we would like to thank Prof. Megan Hazen for guiding us. We would also like to thank the University of Washington for helping us with cloud credits and our fellow classmates for their helpful feedback.