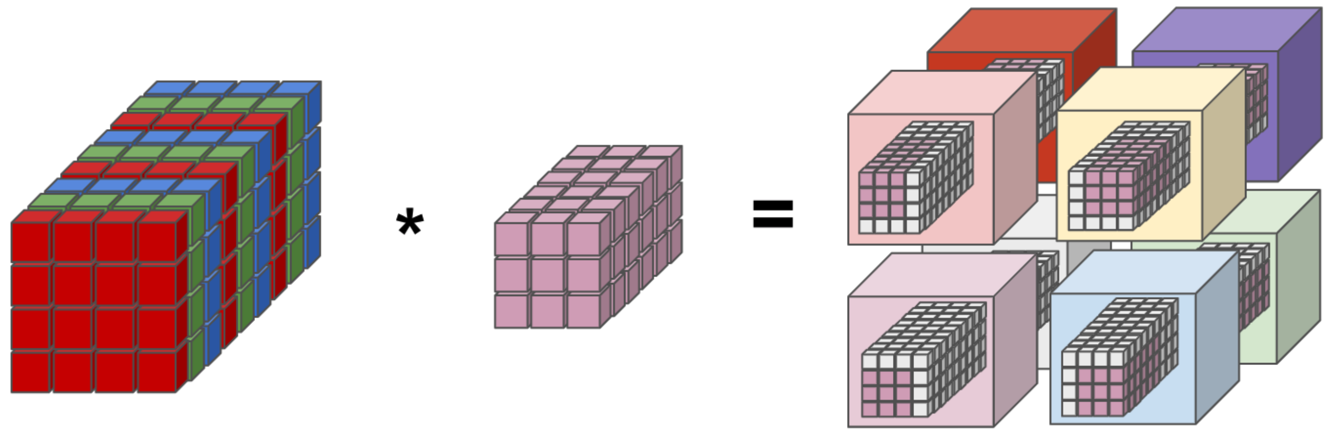
Published in Towards Data Science

Cameron Wolfe Follow
Dec 22, 2021 · 14 min read ★

Deep Learning on Video (Part One): The Early Days…



(created by Author)

Within this series of blog posts, I will explore the topic of deep learning on video data. Beginning from some of the earliest publications on the topic, I aim to outline how this problem was initially approached — prior to the mass popularization of deep learning methodologies— and follow the evolution of video deep learning over time. In doing this, my goal is not only to overview the history of deep learning on video, but also to provide relevant context and understanding for researchers or practitioners looking to become involved in the field.

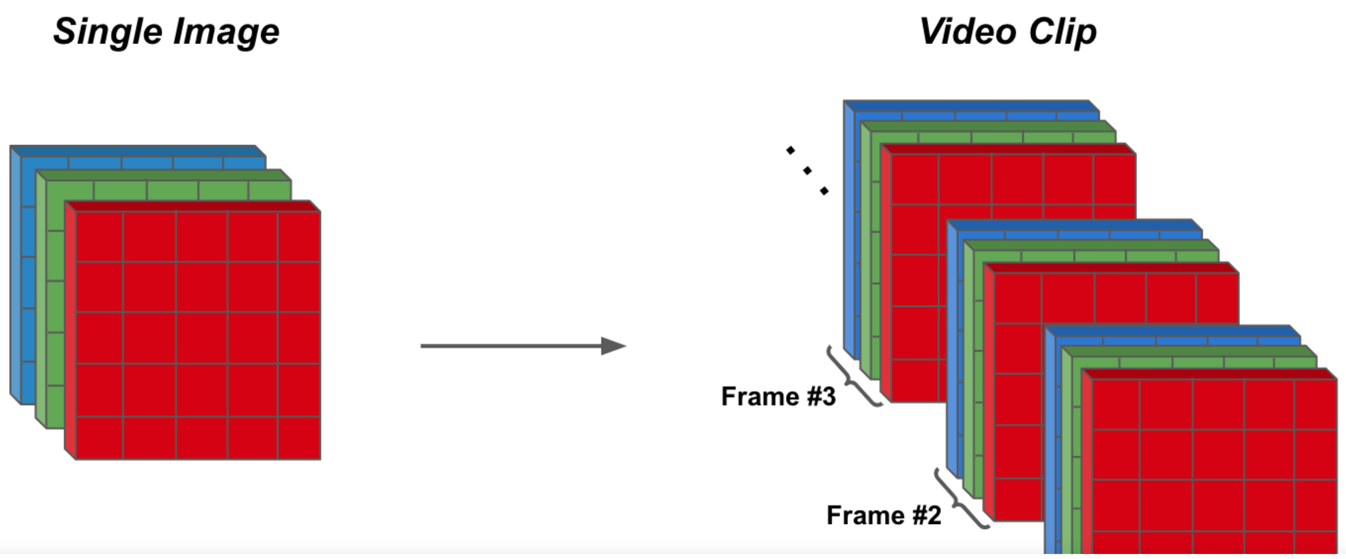
I will separate this topic into several blog posts. This part will overview the "early days" these approaches include two-part architectures that pass hand-crafted video features into 3D convolutional layers, unsupervised learning techniques based upon 3D convolutions, or even 3D convolutional networks that operate directly upon video data.

This post will be separated into three sections. I will first provide relevant background information (e.g., how 3D convolutions work, methodologies before 3D convolutions, how video data is structured). Then, I will overview the relevant literature, providing a high-level (but comprehensive) understanding of early methodologies for deep learning on video. Finally, I will conclude the post and overview the limitations of such methodologies, looking toward the better-performing approaches that were later developed.

Preliminaries

Before diving into relevant literature, there are several concepts that must be outlined. In describing these concepts, I assume the reader has a basic knowledge of convolution operations and the typical structure of image data within computer vision applications. However, I will still try to provide all relevant information, such that important concepts can be understood with minimal prior knowledge.

How is video data structured?



In developing deep learning applications for video, the first question one may ask themselves is how video data is typically structured (e.g., in comparison to images). For images, this question has a simple answer. In most cases, input images are RGB-format (i.e., each image has three color channels associated with it) and have a certain height and width. For example, within the figure above, the input image has three color channels, each with a height and width of five. Thus, input into an image-based deep learning model will usually be a tensor of size `3 x Height x Width`. Several images can then be stacked into a mini-batch, forming a tensor of size `Batch x 3 x Height x Width`.

For videos, the data structure is not much different. Namely, a video is simply a collection of temporally-ordered images, called frames. When viewed in the correct temporal order, these frames reveal the movement of a scene through time, forming a video. Similar to images, each of these frames will typically be represented in RGB-format and have the same spatial resolution (i.e., all frames have the same height and width). In the figure above, the video is comprised of three frames, each with three color channels and a height and width of five.

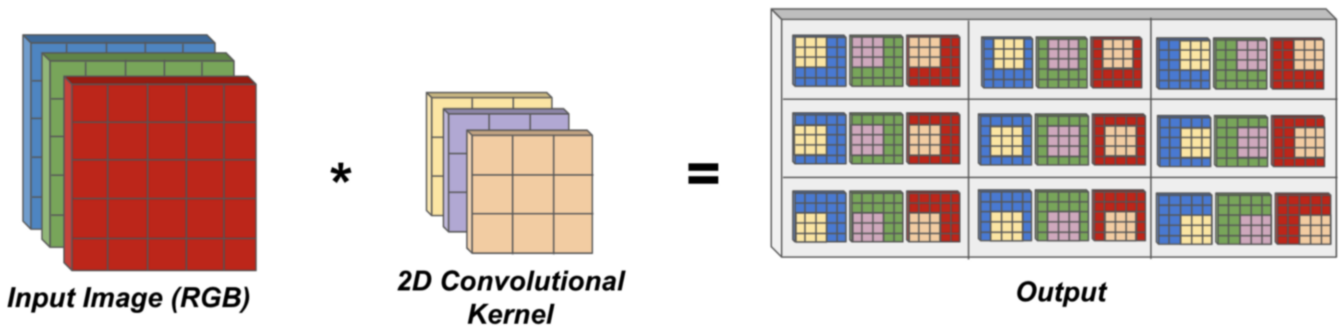
All frames within a video can be combined into a single "volume", forming a tensor of size `3*Frames X Height X Width` by "stacking" all frames on top of each other (i.e., frames can also be aggregated into a tensor of size `Frames X 3 X Height X Width` depending on the application). Combining several videos into a mini-batch must be done carefully, as the number of frames within each video may differ, forming video tensors of different sizes. Nonetheless, different videos can be "chunked" into equal numbers of contiguous frames (or a padding approach could be adopted) to form videos of equal size that can be combined together to form a mini-batch of size `Batch x 3*Frames x Height x Width`.

What datasets were out there?

Within the literature that will be covered within this post, the main datasets used for evaluation include: TRECVID, KTH, Hollywood2, UCF101, and YouTube Action. All of these datasets are focused upon the problem of human action recognition (HAR), in studied application within the early days of video deep learning. Although later research considered other, more complicated applications, HAR remains a notable problem to this day. Interestingly, however, it can be shown that image-based models that operate independently on frames in a video perform well on HAR [1, 2], revealing that only minimal temporal reasoning is required to provide a reasonable solution.

From 2D to 3D…

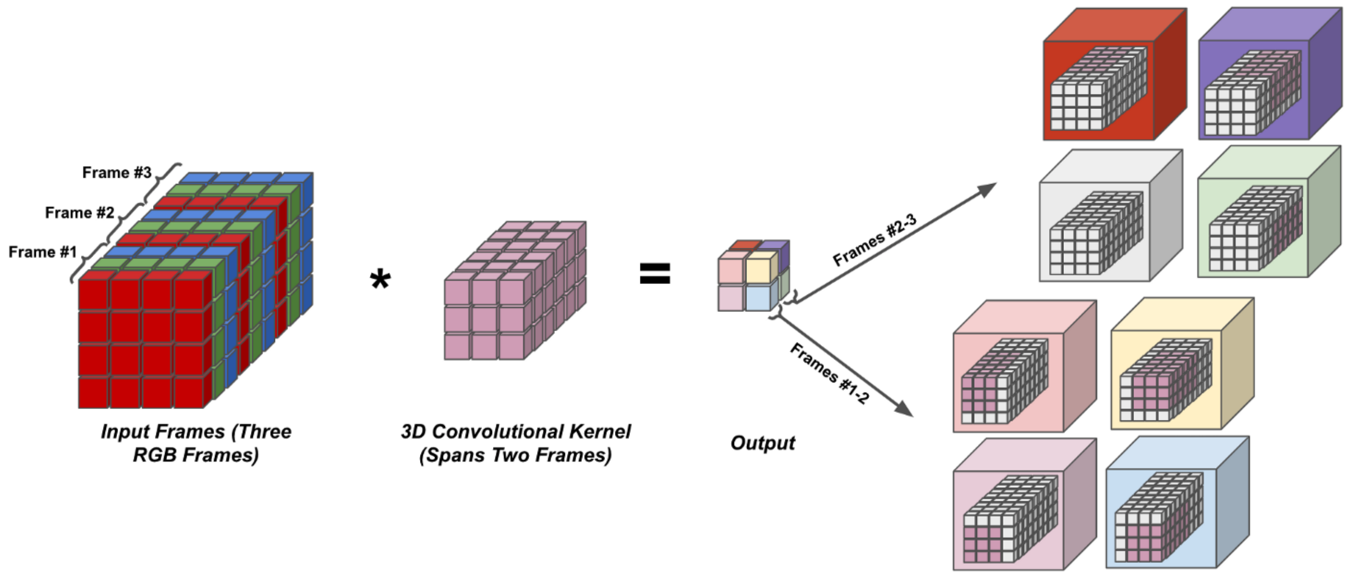
Prior to describing proposed methodologies for deep learning on video, it is useful to consider how constructions from deep learning on images can be generalized to the video domain. In particular, we must consider how 2D convolutions — the workhorse behind convolutional neural networks (CNNs) — can be applied to video. CNNs, which are simply sequences of parallel convolution operations separated by nonlinearities, have long been applied to popular, image-based tasks like image classification, object detection, and key point estimation, yielding great success. As such, it makes sense that early work in deep learning on video would try to extend such a successful approach to operate on video data. A 2D convolution operation is depicted below.



2D Convolution Operation on an RGB Image (created by Author)

As can be seen, within a 2D convolution, a single convolutional kernel, which spans all channels of the input (i.e., the three color channels in the image above), is moved over each spatial location of the input to individually compute corresponding output elements. Here, each output element is derived by multiplying pixel values at the correct spatial locations within the input with their corresponding weights in the kernel

When we begin to consider video data, the input into the convolution operation changes, as we now have multiple frames of video instead of a single image. As such, the input is no longer a single RGB image, but a volume of RGB images stacked on top of each other (i.e., this can be viewed as a similar input with a larger number of channels). To account for multiple frames within the input, the convolution operation must span not only spatial dimensions (e.g., our 2D convolution in the image above spans three pixels in height and width), but also temporal dimensions. Namely, a single kernel must consider multiple frames of the input in computing its corresponding output representation. This formulation, which we refer to as a 3D convolution operation, is depicted below.


Frame #3
Frame #2
Frame #1
Input Frames (Three RGB Frames)
3D Convolutional Kernel (Spans Two Frames)
Output
Frames #2-3
Frames #1-2

3D Convolution Operation on Three RGB Frames (created by Author)

As can be seen, the 3D convolution above has a kernel that spans two frames within the input. Initially considering the first two frames, this kernel is moved over each spatial dimension to compute corresponding output elements (i.e., through element-wise multiplication and summing similarly to 2D convolutions). After moving over possible spatial dimensions, the 3D convolution takes a step in time, moving to the next span of frames it must consider, and repeats the process of traversing spatial dimensions. In the image above, the 3D convolution initially considers the first two frames within the

A 3D convolution is a spatio-temporal operation that considers both spatial features within each individual input frame and temporal features within consecutive frames. Although 3D convolutions are not the only way to learn features from video data, they are the main learnable transformation leveraged by early approaches to deep learning on video. Thus, 3D convolutions are a primary focus of this post.

What was used before deep learning?

Although CNNs were widely successful within the image domain, extending their applications to video took time, especially due to the lack of large, diverse video databases (relative to the size and diversity of image-based datasets) for training purposes [1]. As such, one may begin to wonder how HAR was usually tackled prior to the popularization of deep learning-based systems for action recognition.

Initial systems for HAR extracted fixed, hand-crafted features from video data (e.g., using motion history images, SIFT descriptors, HoG features, template matching, filter banks, etc.) to aid in classification. Typically, the video would be first segmented into interest points (i.e., shorter clips with the human cropped from the background). Then, fixed features were extracted from these interest points (i.e., comprised of several cropped frames) and classified with some machine learning model (e.g., a support vector machine). Such methodologies were later extended to form more complex systems that learned features hierarchically through alternating applications of hand-crafted feature extraction and sub-sampling [3, 4].

Although hand-crafted approaches worked relatively well, such features struggle to generalize across datasets and are difficult to adapt to different sensor domains (e.g., video or radar) [5]. Additionally, hand-crafted features tend to make simplifying assumptions about the underlying data distribution (e.g., fixed viewpoint, human at the center of frame, etc.) that degrade performance in real-world applications [6]. Thus, researchers in this domain began to search for alternative feature extraction methodologies that can be learned directly from data. Due to the ability of CNNs to extract useful features from images, using 2D/3D convolutions to learn features from convolutions to extract features from video data in a learnable fashion. The publications that will be covered within this post can be roughly categorized as methods that combine learnable feature extraction with fixed feature extraction, methods that apply learnable feature extraction directly to video data, and unsupervised methods for learning spatio-temporal features in video.

A Hybrid Approach: Combining Hand-Crafted and Learned Features

Although CNNs were known to be capable of learning useful, discriminative features when trained on large image datasets, early extensions of CNNs into the video domain typically combined the use of 3D convolutions with fixed feature extraction instead of applying 3D convolutions directly to the raw video data. In particular, the video data would first be pre-processed by a fixed feature extractor, such as by computing optical flow or vertical and horizontal gradients over the raw video frames. Then, after pre-processing, the resulting representation would be passed through layers of 3D convolutions to extract learnable features for classification.

This approach was adopted by initial HAR systems applied to the TRECVID surveillance dataset [7, 8]. Such methodologies, instead of operating directly on the raw video data, computed optical flow and directional gradients within the input video. These hand-crafted features were then concatenated with the original video frames and passed through several layers of 3D convolutions, which are separated by sub-sampling operations and non-linear activation functions. Within these 3D CNN architectures, each data modality (i.e., optical flow, directional gradient, and normal frames) is processed independently by separate 3D convolutional kernels. Then, the output of the 3D CNN is used for classification, either with a support vector machine [7] or a fully-connected, linear module [8]. Within these works, several different CNN architectures were explored for HAR, leading to the conclusion that combining the output of several, unique 3D CNN architectures typically leads to the best performance [8].

Such approaches were highly-successful at the time, even outperform single-frame CNNs in HAR (i.e., a strong baseline approach) and nearly all baselines that utilize generalize well across datasets or applications. This observation led to the proposal of methodologies that completely eliminate fixed feature extraction in an attempt to extract fully-learnable features that have no dependence upon any hand-crafted heuristic.

Extracting Learnable Video Features with 3D CNNs

To avoid limitations associated with approaches based upon hand-crafted features, later deep learning methodologies for video passed raw input volumes (i.e., groups of several frames) directly into layers of 3D convolutions separated by pooling and activation layers [9]. Such 3D CNNs were structurally similar to image-based CNNs, but modified to contain 3D convolutions [9]. Typically, these approaches extract a small subset of frames from the full video, crop the frames to focus upon the humans within each frame, then pass the sub-sampled and cropped volume into the 3D CNN [9, 10], where the output is then used to perform classification. Interestingly, the performance of hybrid 3D CNNs (i.e., those that combine hand crafted and learned features) can be easily exceeded by such an approach, even with the use of a much smaller network [10].

Notably, because such approaches segment the underlying video into shorter clips (e.g., containing 10–15 frames) that are passed into the 3D CNN, the final classification does not consider relationships between larger or temporally-distant groups of frames within the full video. To solve this shortcoming, the approach described above was combined with recurrent neural networks — a long short-term memory (LSTM) network [11] in particular — to synthesize information extracted with 3D CNNs from each clip in a video. In particular, the video was first segmented into smaller groups of frames, which are again passed into the 3D CNN to extract features. However, instead of performing classification on each of these features independently, the features associated with each video segment are passed, in temporal order, as input to an LSTM, which then performs the final classification [10]. Such a methodology was shown to consider the context of the full video and, as a result, achieve significantly improved performance on HAR.

followed the approach of related deep learning literature for image recognition, choosing to let all features be obtained in a learnable fashion. As such, systems for video learning tasks (e.g., HAR) became less domain-dependent and video-related research could evolve at a faster pace, as the need to constantly determine the best hand-crafted features for each possible dataset was eliminated [12]. However, one major limitation still exists for such approaches — their performance is entirely dependent on the quality and abundance of labeled data for the underlying task.

Unsupervised Representation Learning for Video

Despite the incredible value provided by 3D CNN models, their performance is limited by the amount of data available for training. Especially during the early days of deep learning, the availability of labeled video data was minimal [1]. As such, the abilities of 3D CNN models to learn high-quality features was always somewhat limited due to a lack of sufficient data. Such a limitation led to the simultaneous development of unsupervised deep learning methodologies for extracting meaningful video features [12, 13].

The first of such unsupervised learning methodologies for video drew upon a gated, restricted Boltzmann machine (GRBM) architecture to learn features from adjacent frames in a video. Initially, the GRBM architecture could not handle processing high-resolution images (i.e., it was too computationally expensive), due to the flattening of images into a vector before being passed as input to the model. To mitigate this problem, the GRBM was modified in [13] to use convolution operations, allowing for higher-resolution images to be processed at a lower computational cost. This convolutional GRBM model could then be trained on adjacent video frames in an unsupervised manner to learn useful spatial and temporal information, thus allowing high-quality features to be extracted from the video without the need for labeled data. From here, it was then shown that the features extracted from the convolutional GRBM could be passed into a 3D CNN to solve tasks like HAR, where performance was greatly improved by the unsupervised pre-training of the GRBM [13].

learning methodology, based upon independent component analysis, that was previously used to extract meaningful features from static images and can be trained in an unsupervised manner. Similar to GRBMs, ISA was too computationally inefficient to be applied to high-resolution images. To solve this issue, [12] proposed that ISA be first trained (i.e., in an unsupervised manner) on smaller image patches from the frames of a video. Then, the dimensionality of ISA output on these initial frame patches could be reduced — using principal component analysis (PCA) — enough to be passed through another ISA module. Then, the output of these modules could be fed into a 3D CNN to extract features for classification [12]. Such an approach greatly improved the efficiency of unsupervised representation learning on video and achieved state-of-the-art performance on nearly all datasets for HAR.

Conclusions and Looking Forward

Within this post, we overviewed early methodologies that utilized 3D convolution operations to extract learnable features from video. Originally, such work adopted a hybrid approach, in which hand-crafted features were extracted and mixed with learnable features from 3D CNNs. Later, all hand-crafted components were eliminated, and video features were extracted in a completely learnable fashion by applying 3D CNNs directly to raw video data. Then, in parallel, several unsupervised approaches for learning meaningful video features were proposed, thus mitigating dependence upon labeled video data in solving common learning tasks (e.g., HAR).

Although early deep learning approaches for video worked relatively well, they were quite inefficient and did not match the impressive performance achieved by deep learning on image-based tasks. The approaches outlined in this work only narrowly outperformed single-frame baseline models and were even occasionally outperformed by hand-crafted baseline methodologies on certain datasets. As such, a more performant approach was needed that maintains or improves the computational efficiency of the 3D CNN methodologies outlined within this post. Eventually, this need led to the proposal of two-stream 3D CNN architectures, which will be overviewed in you'd like to follow my future work, you can follow me on Medium or check out the content on my personal website. This series of posts was completed as part of my background research as a research scientist at Alegion. If you enjoy this post, feel free to check out the company and any relevant, open positions — we are always looking to discuss with or hire motivated individuals that have an interest in deep learning-related topics!

Bibliography

[1]https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42455.pdf

[2] https://ieeexplore.ieee.org/document/1495508

[3] https://mcgovern.mit.edu/wp-content/uploads/2019/01/04069258.pdf

[4] https://www.cs.tau.ac.il/~wolf/papers/action151.pdf

[5] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5995496

[6] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6165309

[7]https://www.researchgate.net/publication/229045898_Detecting_Human_Actions_in_Surveillance_Videos

[8] https://ieeexplore.ieee.org/document/6165309

[9] https://link.springer.com/content/pdf/10.1007/978-3-540-72393-6_85.pdf

[10] https://link.springer.com/content/pdf/10.1007%2F978-3-642-25446-8_4.pdf

[11] https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[12] https://ieeexplore.ieee.org/document/5995496