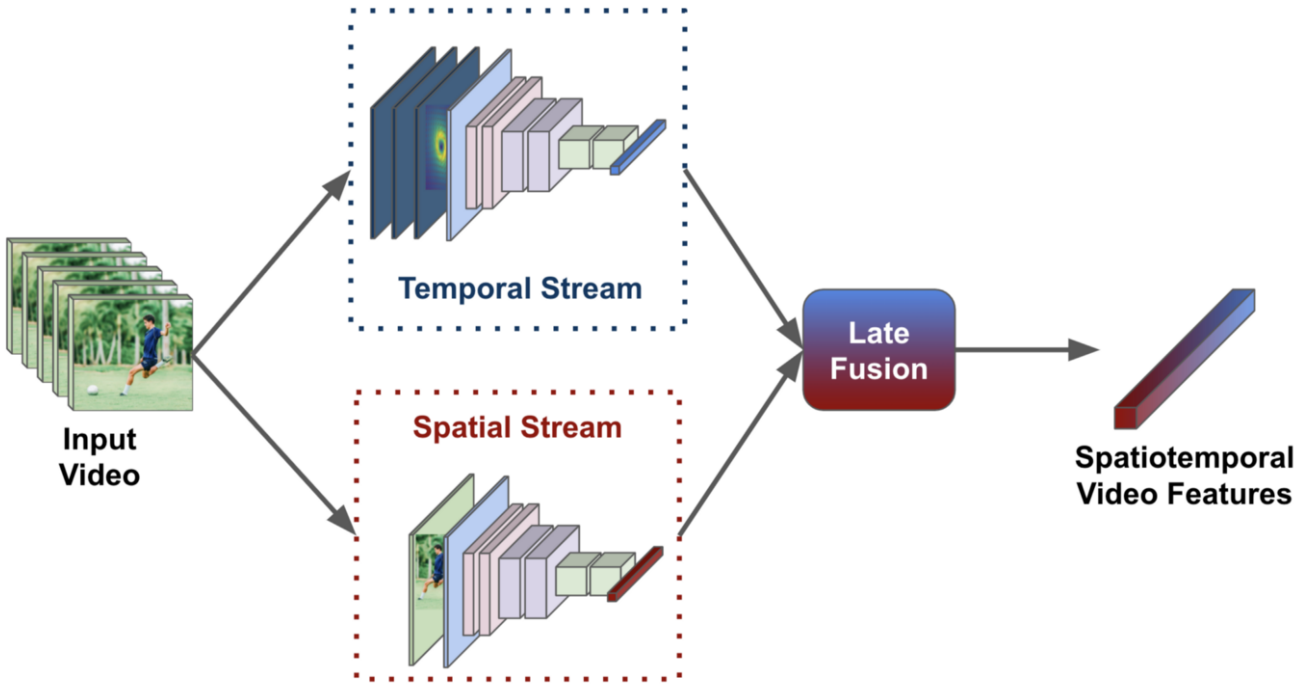
Open in app Get started

You have **1** free member-only story left this month. Sign up for Medium and get an extra one

Cameron Wolfe Follow
Jan 29 · 17 min read ★

Deep Learning on Video (Part Two): The Rise of Two-Stream Architectures



Two-Stream Architecture for Video Understanding (Image by Author)

This post is the second in a series of blog posts exploring the topic of deep learning on video data. The goal of this series of blog posts is to both overview the history of deep learning on video and provide relevant context for researchers or practitioners looking to become involved in the field. In the first post of the series, I overviewed the earliest publications on the topic that used 3D convolutions to extract learnable features from video.

Open in app Get started

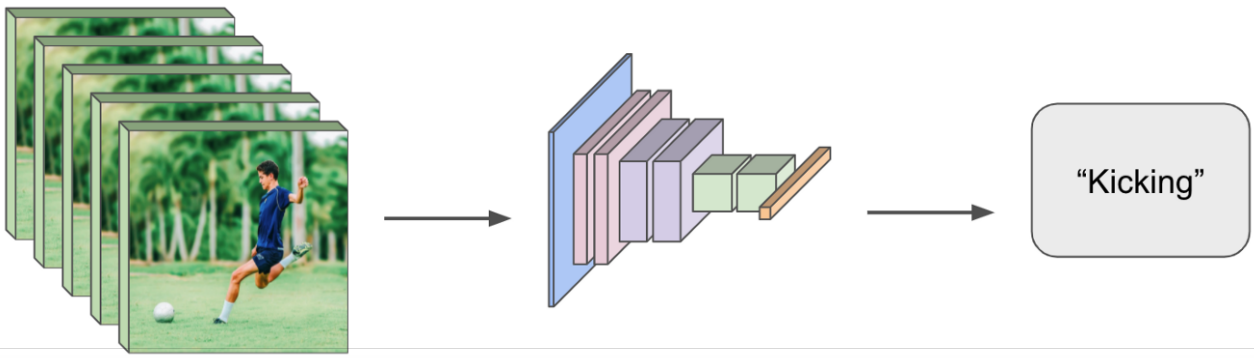
networks (CNNs) — one to handle spatial features and one to handle temporal/motion features. These separate CNNs are typically referred to as the "spatial" and "temporal" networks within the two-stream architecture, and the output of these separate network components can be combined together to form a spatiotemporal video representation. Two-stream architectures yielded massively-improved performance in video action recognition, making them a standard approach to video deep learning for some time.

The post will begin by overviewing relevant preliminary information, such as the definition/formulation of two-stream architectures and the limitations of previous work. I will then overview the literature on two-stream network architectures, including the papers that originally proposed the architecture and later, more complex variants. At the end of the post, I will discuss other, related approaches to video understanding that were proposed during this time period and outline the limitations of two-stream networks, thus motivating improvements that were yet to come.

Preliminaries

Several preliminary concepts must be discussed prior to outlining two-stream approaches to video deep learning. For all details regarding the formulation of 2D/3D convolutions, the structure of video data, and existing approaches to video deep learning prior to two-stream architectures, I refer the reader to the first post in the series. Given an understanding of these concepts, however, I try to overview relevant information in a way that is understandable even with minimal background knowledge.

What problem are we trying to solve?



Open in app Get started

The majority of methodologies overviewed within this post study the problem of video-based human action recognition (HAR). HAR datasets contain a number of variable-length videos that are each associated with a semantic label, corresponding to the action being performed in that video. Typically, the videos within the dataset are focused upon a single entity that is performing the action in question, and the video does not extend far before or after the action is performed. Thus, the goal of the underlying model is to predict the semantic action label given the video as input. HAR was, by far, the most commonly-studied video understanding problem at the time of two-stream network architectures.

Why do we need something new?

In the first post of the series, we overviewed several possible approaches for video deep learning. These approaches typically adopt 3D CNN models (i.e., several consecutive layers of 3D convolutions separated by non-linear activation layers), pass either raw video or hand-crafted video features (e.g., optical flow or directional gradients) as input to these models, and perform supervised training via back propagation based on the semantic label assigned to each video. Given that these methodologies exist, one may begin to wonder why a new approach to video deep learning is needed.

The answer to this question is quite simple — existing models just performed poorly. In fact, earlier deep learning-based approaches to HAR were often outperformed by hand-crafted, heuristic methods and performed comparably to deep learning models that use individual frames as input (i.e., completely ignoring the temporal aspect of video) [1, 2]. Such poor performance was shocking given the massive success of deep learning in the image recognition domain [3]. As such, the research community was left wondering how deep learning could be made more useful for video.

The initially poor performance of 3D CNN architectures was mostly attributed to the lack of large, supervised datasets for video understanding [1]. For example, UCF-101 and HMDB-51, the most commonly-used datasets for HAR at the time, each contain only 13,320 and 7,000 labeled video clips, respectively. In comparison, ImageNet — a widely-used benchmark for image classification — contains ~1.3 million training examples. Although larger datasets (e.g., Sports1M [1]) were proposed for HAR, they were usually collected automatically and quite noisy, leading smaller, curated datasets to be used more



Open in app Get started

more data or a different learning paradigm was needed to enable better performance.

The Two-Stream Network Architecture

The two-stream architecture took the first step towards deep learning-based approaches surpassing the performance of heuristic and single-frame methods for HAR, catalyzing the onset of a new era in video understanding. Put simply, this architecture enabled high-performance video understanding despite a lack of sufficient supervised data by encoding motion information directly into the network's input. We will now overview the basic ins and outs of the two-stream architecture to provide context for the relevant research overviewed throughout the rest of the post.

The two-stream network architecture [2] is motivated by the two-stream hypothesis for the human visual cortex in biology [4], which states that the brain has separate pathways for recognizing objects and motion. Attempting to mimic this this structure, the two-stream network architecture for video understanding utilizes two separate network components that are dedicated to processing spatial and motion information, respectively. Thus, the two-stream architecture delegates the tasks of object recognition and motion understanding to separate network components, forming different pathways for spatial and temporal cues.

The input to the two stream architecture is typically centered around a single frame within the input video, which is directly passed as input into the network's spatial stream (i.e., no adjacent frames are considered). As input to the temporal stream, `L` consecutive frames (centered around the frame passed as input to the spatial stream) are selected. The horizontal and vertical optical flow fields are then computed for each of the adjacent frames in this group, forming an input of size `H x W x 2L` (i.e., `H` and `W` are just the height and width of the original image). Then, this stack of optical flow fields is passed as a fixed-size input into the network's temporal stream.

From here, the spatial and temporal streams process the frame and optical flow input using separate convolutional networks with similar structure — the only difference between the respective networks is that the temporal stream is adapted to accept input with a larger number of channels (i.e., `2L` channels instead of 3). Once the output of each stream is



Open in app Get started

simple, “late” fusion strategy. This formulation of the two-stream architecture is illustrated below.

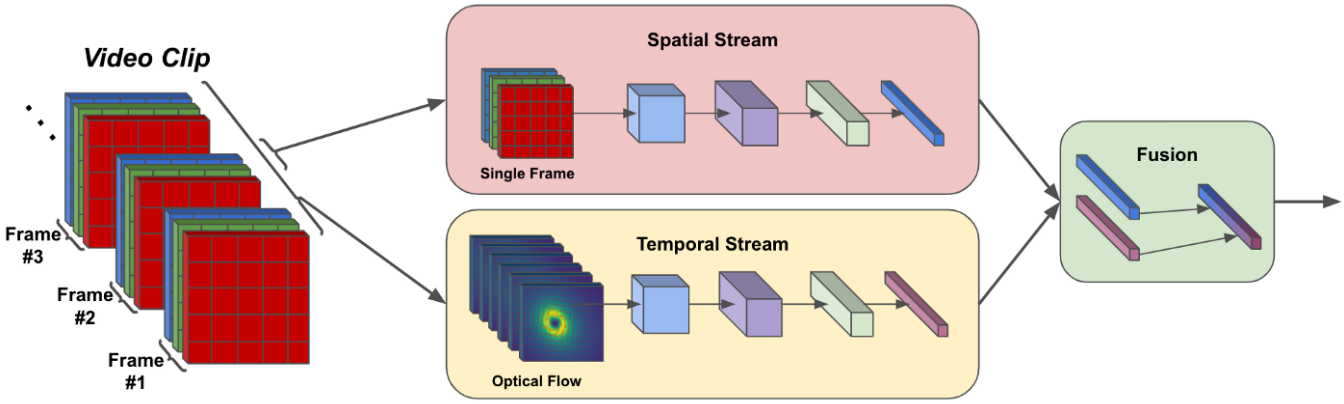


Illustration of a Basic, Two-Stream Network Architecture (Image by Author)

As shown above, the input to the two-stream architecture only contains a single frame for the spatial stream and a fixed-size group of optical flow maps for the temporal stream. Although one may argue this approach is limited because it only looks at a fixed-size, incomplete portion of the video, this issue can be mitigated by sampling several of these fixed-size clips from the underlying video and averaging their output to yield a final prediction. Furthermore, the clips used as input to the two-stream architecture could be sampled with a stride (i.e., instead of sampling adjacent frames, sample those those at consecutive intervals of two, three, four, etc. frames) so that the network considers a larger temporal extent within the underlying video.

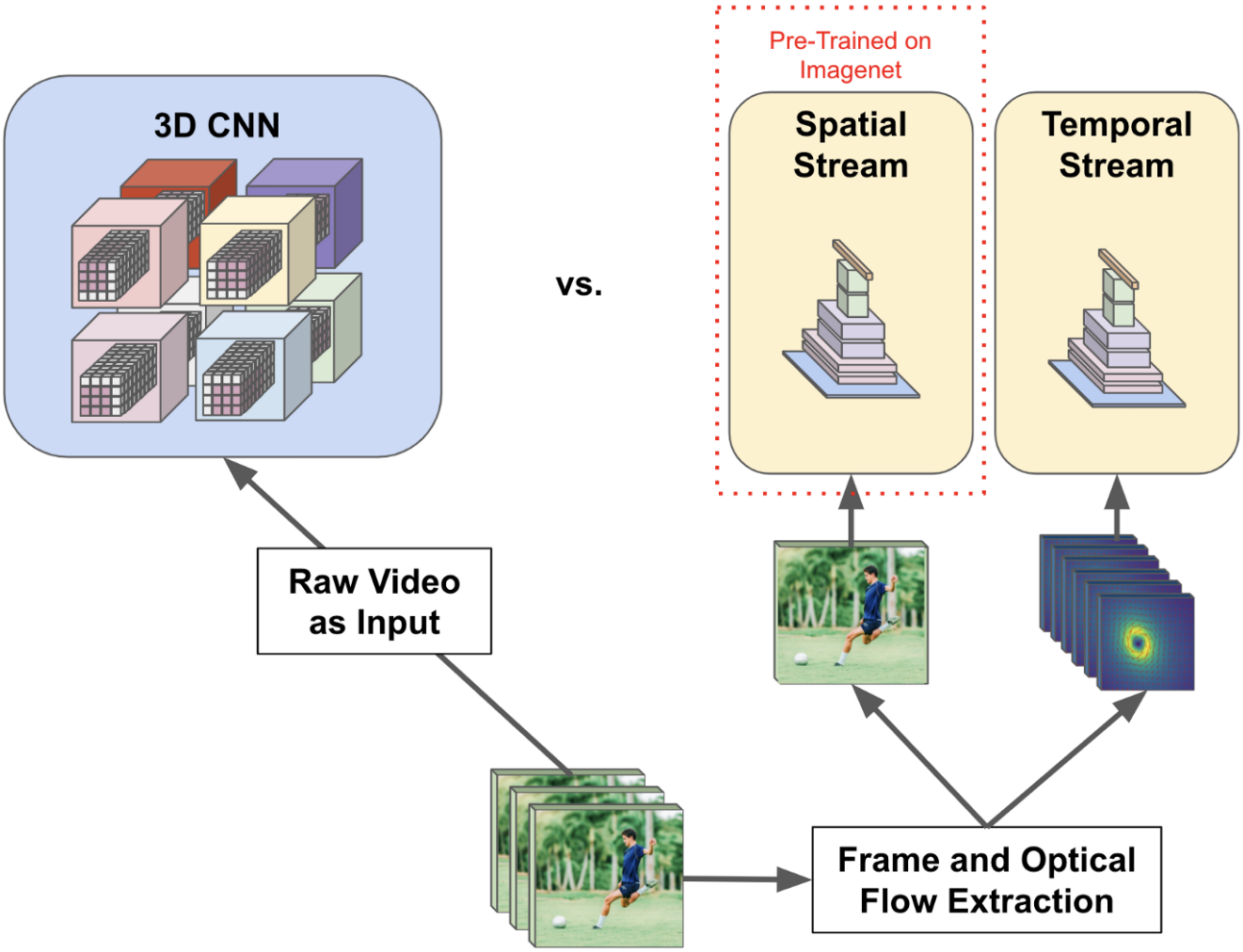
Why does this work?

After providing the formulation of a basic two-stream network architecture, one may begin to wonder why such an architecture would be superior to something like a 3D CNN. After all, 3D CNNs have very high representational capacity (i.e., lots of parameters), so they should be able to learn good spatial and temporal features, right?

Recall, however, that the amount of supervised data for video understanding was limited at the time two-stream architectures were proposed. As such, the two-stream approach provides a few major benefits that enable them to exceed the performance of 3D CNNs.

Open in app Get started

large image-classification datasets (e.g., ImageNet), which provides a massive performance benefit. These points of differentiation between two-stream architectures and 3D CNNs are depicted below.



Depiction of the Differences between Two-Stream and 3D CNN Networks (Image by Author)

While 3D CNNs represent space and time as equivalent dimensions (i.e., this contradicts the two-stream hypothesis in biology) [1], two-stream architectures enable better performance because *i)* motion information is encoded directly in the input (i.e., no longer any need to learn this from data) and *ii)* large amounts of image classification data can be leveraged to train the spatial network. In the low-data regime, this basic two-stream architecture took a large step towards surpassing the performance of the best hand-crafted, heuristic methods for video understanding.

Open in app Get started

introduced, I will explore some of the literature behind the proposal and development of two-stream architectures for video understanding. I will begin with early papers that explored the topic, followed by more advanced architectural variants — still following the two-stream approach — that later emerged.

Early Approaches

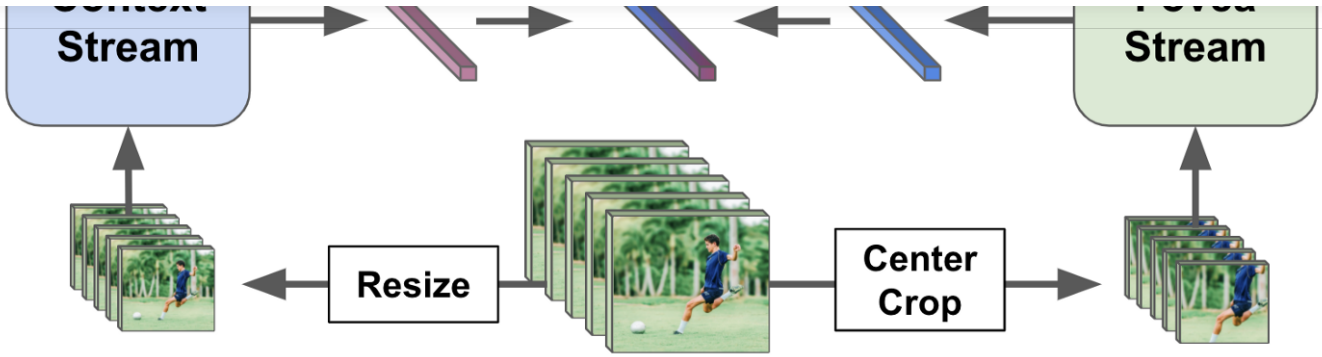
Context and Fovea Streams [1]. The concept of two-stream architectures, although popularized more formally by a later paper [2], was loosely explored in [1]. Within this paper, the authors created two separate processing streams for input data: context and fovea streams. Each of these separate stream share the same network architecture and take the same number of frames as input.

To improve computational efficiency, frames are reduced to 50% of their original area prior to being provided as input to each of the streams, but the context and fovea stream adopt different approaches for reducing the size of the input. Namely, frames within the context stream are just resized, while frames in the fovea stream are center cropped. Put simply, such an approach ensures that context and fovea streams receive low and high-resolution input, respectively — one network sees full frames at low resolution, while the other sees only the center of each frame but at full resolution.

Notice that, unlike the original description provided for two-stream architectures, this approach does not explicitly try to separate motion and spatial recognition into separate processing streams. Instead, each stream is given the same group of raw frames — as opposed to a single frame and optical flow stack — as input (i.e., just resized/cropped differently) and passes such frames through identical, but separate network architectures. Then, the outputs of the two streams are combined prior to prediction. See the figure below for a depiction.



Open in app Get started



Context and Fovea Streams for Video Understanding (Image by Author)

Given that both streams are responsible for detecting both spatial and temporal features, one must determine how to best incorporate temporal information within the streams. We cannot just adopt 2D CNNs within each stream, as this will never consider relationships between adjacent frames. To determine how to best fuse spatial and temporal information, the authors test several possibilities for the CNN architectures of each stream:

- *Early Fusion:* change the first convolutional layer of each stream to a 3D convolution.
- *Late Fusion:* use 2D CNNs for each stream, compute their output on two frames that are 15 frames apart, then merge the final output for both frames in each stream.
- *Slow Fusion:* change all convolutional layers within each stream to be 3D convolutions with a smaller temporal extent (i.e., kernel size is smaller in time) in comparison to early fusion.

The authors find that slow fusion consistently performs best. As a result, the final network architecture adopts a two-stream approach (loosely), where each stream is a 3D CNN that takes a group of frames as input. The only distinction between these streams is their input — frames are resized within the context stream (i.e., lower resolution) and center cropped within the fovea stream (i.e., higher resolution). Although this approach is efficient in comparison to previous 3D CNNs (i.e., due to reducing the dimensionality of input images) and performs comparably, it only performs slightly better than single-frame, 2D CNNs on HAR and is often outperformed by hand-crafted, heuristic methods. Thus, this approach had to be extended and improved upon.

Open in app Get started

spatial and temporal features separately within each stream by passing, as input, a frame or a stack of optical flow maps to the spatial and temporal streams, respectively. As a result, the two-stream architecture was one of the first to make an explicit effort at capturing motion information within the underlying video. By adopting late fusion as described within the preliminaries, the spatial and temporal network output could be combined to form highly robust spatiotemporal features.

The two-stream architecture, as originally proposed, was the first deep learning-based methodologies to achieve consistently improved performance in comparison to single-frame and heuristic baseline methodologies on HAR benchmarks. Thus, it became a standard for video understanding that was heavily studied, utilized, and extended in later work. The dependence of the two-stream architecture upon hand-crafted optical flow features as input (as well as several other previously-discussed aspects of the architecture's design) enabled better performance in the face of limited data, but this dependence upon hand crafted-features (i.e., optical flow) as input was eventually seen as a limitation to the architecture's design.

Best Practices [5]. In addition to the main papers that originally proposed and explored the two-stream network architecture, following work adopted this architecture and explored the best practices for achieving optimal performance. In particular, [5] explored deeper variants of the original two-stream architecture, finding that using CNN backbones with more layers (e.g., VGG [6] and inception-style networks [7]) within each stream of the architecture can yield significant performance benefits if trained properly. The authors claim that the improvement in performance from deeper two-stream architectures comes from the increased representational capacity of the underlying network, which is beneficial for complex tasks like HAR.

To yield the best possible performance, the spatial and temporal streams are ***both*** pre-trained (i.e., as opposed to just pre-training the spatial stream) using image classification and optical flow data (i.e., generated from an image recognition dataset), respectively. Then, the model is trained with a low learning rate with high levels of data augmentation and regularization, yielding final performance that exceeds that of previous implementations of two-stream architectures on HAR.



Open in app Get started

proposed variants of the architecture (with slight modifications) that yielded massively-improved performance. These more advanced variants maintained the same general network architecture and input schema, but added supplemental modules or connections within the network to improve the representation of temporal information. These modification were motivated by the fact that original two-stream networks relied heavily upon spatial information and did not represent temporal data well.

Improved Fusion for Two-Stream Architectures [8]. Initial criticisms to the two-stream architecture claimed that the existing formulation did not properly synthesize spatial and temporal information. Namely, because temporal and spatial features were only fused at the output layer of the two streams, the model does not learn to utilize temporal information properly and relies mostly upon spatial information to generate a correct classification. Additionally, the temporal "scale" of the two-stream architecture was limited, as it only considered a fixed-size subset of frames as input to the temporal stream (i.e., as opposed to the full video).

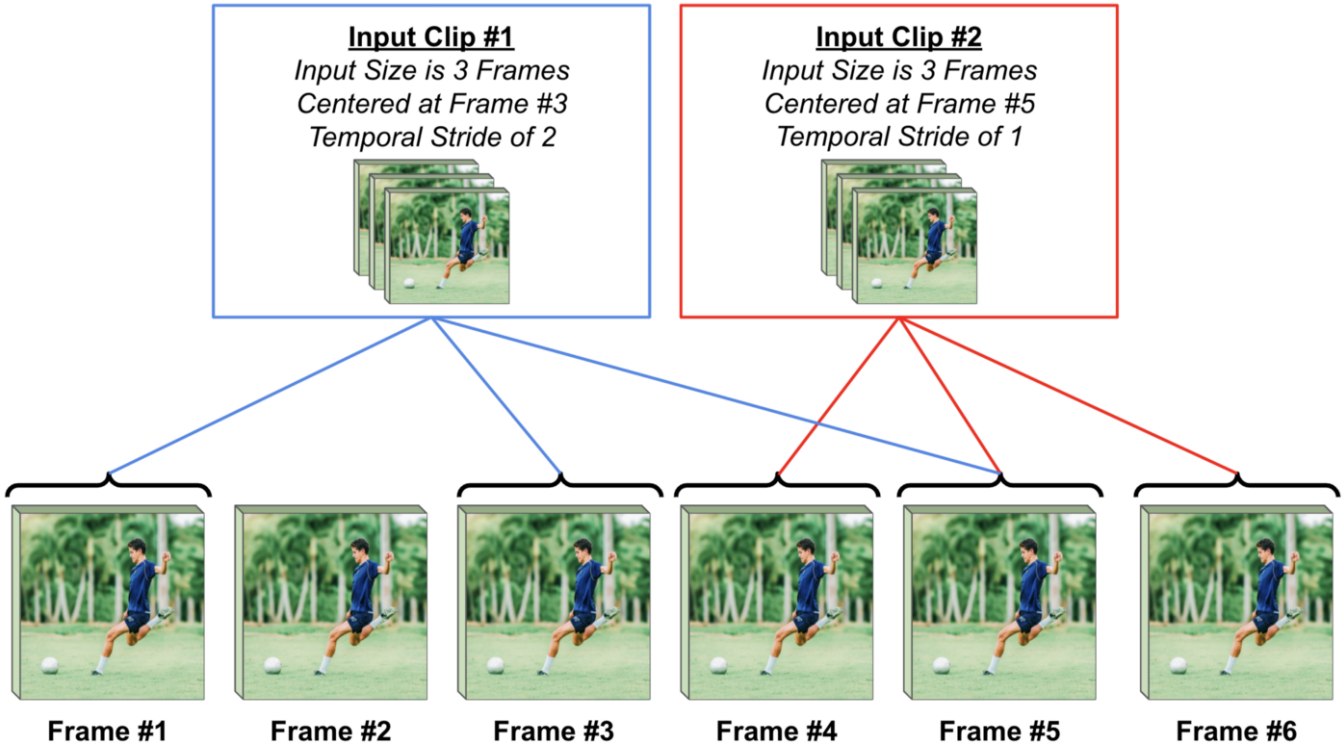
To improve the fusion of temporal information within the two-stream architecture, the authors of [8] explored numerous methods of fusion between feature representations of spatial and temporal streams within the two-stream architecture. As recommended by previous work [5], deeper VGG networks are adopted as the backbone for each stream. Then, the authors consider the following fusion types: sum, max, concatenate, convolutional (i.e., concatenate feature maps then convolve with a bank of 1x1 filters), and bilinear (i.e., compute matrix outer product over features at each pixel and sum over pixel locations to output a single vector). After testing each of these fusion types at different layers within the two-stream network, the authors find that adopting convolutional-style fusion combined with a temporal pooling operation after the last convolutional layer (i.e., before ReLU) of the VGG networks yields the best performance.

Beyond developing a better fusion methodology within the two-stream network, the authors also propose a sampling methodology that allows the underlying network to consider frames across the entire video. Namely, several different "clips" are sampled throughout the video, each of which have a different temporal stride; see the image below. By sampling clips at numerous different locations in the video and utilizing both small and



Open in app Get started

the authors are able to set a new state-of-the-art performance in HAR benchmarks.



Clip Sampling from an Underlying Video with Varying Temporal Stride (Image by Author)

Residual Two-Stream Architectures [9]. Shortly after the proposal of improved fusion techniques, the two-stream architecture was adapted to utilize a ResNet-style CNN architecture within each of its streams. Such a modification was motivated by the incredible success of the ResNet family of CNN architectures for image recognition [10], which remain a widely-used family of architectures to this day. Beyond the popularity of the ResNet architecture, however, many best practices for achieving optimal performance with CNN architectures for image recognition had emerged (e.g., batch normalization [11], maximizing the receptive field, avoiding information bottlenecks, etc.) that were not yet used within video deep learning. Thus, authors of [9] attempted to introduce these numerous improvements, including the ResNet architecture, to two-stream networks.

The ResNet architectures used for the spatial and temporal streams within [9] are modified slightly from their original formulation. Namely, supplemental residual connections are added *i)* in between the spatial and temporal streams (i.e., the authors claim that this

Open in app Get started

a result of these supplemental residual connections, the resulting architecture has a large spatiotemporal receptive field (i.e., the full extent of the video is considered in both space and time).

Interestingly, the parameters of both network streams are initialized using pre-trained weights from ImageNet. Then, 3D convolutions are initialized in a way that uses the same weights as the original corresponding 2D convolution for the center frame, but forms residual connections through time (i.e., just an identity operation) to each of the adjacent frames within the 3D convolution's receptive field. This residual, two-stream architecture (i.e., including the supplemental residual connections between streams and through time) is shown to learn and extract features that better represent the evolution of spatial concepts through time, thus further improving upon the performance of previous approaches for HAR.

What else did people try?

Although the two-stream architecture was a very popular choice for deep learning on video, not all research on video understanding during this time leveraged such an approach. In fact, many other interesting algorithms were developed in parallel to the two-stream architecture that were able to achieve impressive performance on HAR (or other video understanding benchmarks) despite using a completely different approach.

New 3D CNN Variants [12]. Another popular architecture was the C3D network [12], which adopts a convolutional architecture that is fully-composed of 3D convolutions (i.e., `3x3x3` kernels in every layer). In particular, this network takes a fixed-length set of frames as input and passes them through a sequence of convolutional and pooling layers, followed by two fully-connected layers at the end of the network. To train this network, authors use the massive Sports1M dataset for HAR [1], thus enabling the network to learn discriminative features over a large dataset (i.e., recall that previous attempts at 3D CNNs performed poorly due to a lack of sufficient supervised data). Nonetheless, this architecture was outperformed by more advanced two-stream variants and criticized for only considering a limited temporal window within the video. As a result, C3D gained less popularity than two-stream architectural variants.



Open in app Get started

action brings upon the environment. Inspired by this idea, authors developed a method for splitting the underlying video into "pre-condition" and "effect" states, representing portions of the video before and after the action takes places. Then, these groups of frames are passed through separate CNN architectures to extract feature representations for each. From here, an action is modeled as a linear transformation (i.e., a matrix multiplication) that transforms the pre-condition features into the effect features (i.e., this can be measured with the cosine distance between predicted and actual effect feature vectors). Interestingly, this entire siamese network architecture (including the action transformations) could be trained using an expectation maximization procedure to achieve competitive performance on HAR benchmarks.

Other stuff… Some work on video understanding studied more efficient representations of 3D convolutions, finding that robust spatiotemporal relations could be learned by factoring 3D convolutions into separate 2D spatial and 1D temporal convolutions that are applied in sequence [14]. The resulting architecture contains significantly fewer parameters than corresponding 3D CNN architectures and, thus, can achieve better performance in the limited-data regime. Additionally, concurrent work went beyond the HAR problem domain and considered the problem of action detection [15], where actions must be both identified/classified and localized within the underlying video. By adopting a region proposal and feature extraction approach that utilizes early, two-stream architecture variants [1], impressive performance could be achieved on action detection benchmarks.

Conclusions and Future Directions…

Though many approaches to video understanding were explored concurrently, the impressive performance of the two-stream approach led to popularization of the technique. Nonetheless, the two-stream architecture was still — at its core — dependent upon hand-crafted features that are extracted from the underlying video. In particular, it relied upon optical flow maps that were extracted from the underlying video and passed as input to the temporal stream. Although such features make minimal assumptions about the underlying video (i.e., just smoothness and continuity assumptions), this reliance upon hand-crafted optical flow features would be criticized by later work, leading to the



Open in app Get started

Thank you so much for reading this post! I hope you found it helpful. If you have any feedback or concerns, feel free to comment on the post or reach out to me via twitter. If 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://arxiv.org/abs/1406.2199

[3] https://arxiv.org/abs/1803.01164

[4] https://pubmed.ncbi.nlm.nih.gov/1374953/

[5] https://arxiv.org/abs/1507.02159

[6] https://arxiv.org/abs/1409.1556

[7] https://arxiv.org/abs/1409.4842

[8] https://arxiv.org/abs/1604.06573

[9] https://arxiv.org/abs/1611.02155

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

[11] https://arxiv.org/abs/1502.03167

[12] https://arxiv.org/abs/1412.0767

[13] https://arxiv.org/abs/1512.00795

Open in app Get started