Video Transformer Network

Daniel Neimark   Omri Bar   Maya Zohar   Dotan Asselmann

Theator

{danieln, omri, maya, dotan}@theator.io

Abstract

This paper presents VTN, a transformer-based framework for video recognition. Inspired by recent developments in vision transformers, we ditch the standard approach in video action recognition that relies on 3D ConvNets and introduce a method that classifies actions by attending to the entire video sequence information. Our approach is generic and builds on top of any given 2D spatial network. In terms of wall runtime, it trains $16.1\times$ faster and runs $5.1\times$ faster during inference while maintaining competitive accuracy compared to other state-of-the-art methods. It enables whole video analysis, via a single end-to-end pass, while requiring $1.5\times$ fewer GFLOPs. We report competitive results on Kinetics-400 and Moments in Time benchmarks and present an ablation study of VTN properties and the trade-off between accuracy and inference speed. We hope our approach will serve as a new baseline and start a fresh line of research in the video recognition domain. Code and models are available at: https://github.com/bomri/SlowFast/blob/master/projects/vtn/README.md.

1. Introduction

Attention matters. For almost a decade, ConvNets have ruled the computer vision field [22, 7]. Applying deep ConvNets produced state-of-the-art results in many visual recognition tasks, i.e., image classification [32, 19, 34], object detection [17, 16, 28], semantic segmentation [25], object instance segmentation [18], face recognition [33, 30] and video action recognition [9, 38, 3, 39, 23, 14, 13, 12]. But, recently this domination is starting to crack as transformer-based models are showing promising results in many of these tasks [10, 2, 35, 40, 42, 15].

Video recognition tasks also rely heavily on ConvNets. In order to handle the temporal dimension, the fundamental approach is to use 3D ConvNets [5, 3, 4]. In contrast to other studies that add the temporal dimension straight from the input clip level, we aim to move apart from 3D networks. We use state-of-the-art 2D architectures to learn the spatial feature representations and add the temporal infor-
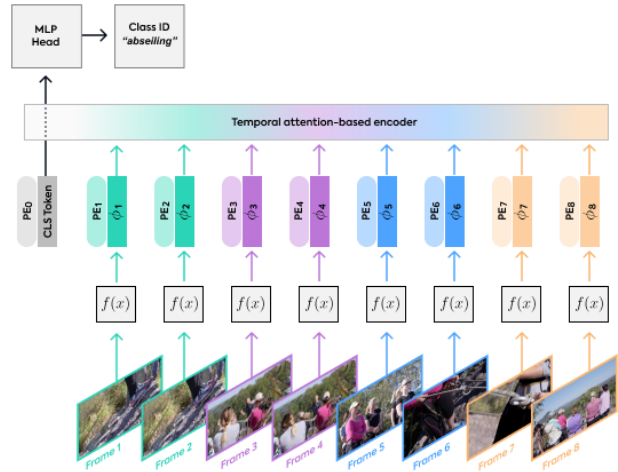

Figure 1. Video Transformer Network architecture. Connecting three modules: A 2D spatial backbone ($f(x)$), used for feature extraction. Followed by a temporal attention-based encoder (Longformer in this work), that uses the feature vectors (ϕ_i) combined with a position encoding. The $[CLS]$ token is processed by a classification MLP head to get the final class prediction.

mation later in the data flow by using attention mechanisms on top of the resulting features. Our approach input only RGB video frames and without any *bells and whistles* (*e.g.*, optical flow, streams lateral connections, multi-scale inference, multi-view inference, longer clips fine-tuning, etc.) achieves comparable results to other state-of-the-art models.

Video recognition is a perfect candidate for Transformers. Similar to language modeling, in which the input words or characters are represented as a sequence of tokens [37], videos are represented as a sequence of images (frames). However, this similarity is also a limitation when it comes to processing long sequences. Like long documents, long videos are hard to process. Even a 10 seconds video, such as those in the Kinetics-400 benchmark [21], are processed in recent studies as short, ~2 seconds, clips.

But how does this clip-based inference would work on much longer videos (i.e., movie films, sports events, or surgical procedures)? It seems counterintuitive that the infor-

Figure 2. Extracting 16 frames evenly from a video of the *abseiling* category in the Kinetics-400 dataset [21]. Analyzing the video's full context and attending to the relevant parts is much more intuitive than analyzing several clips built around specific frames, as many of these frames might lead to false predictions.

mation in a video of hours, or even a few minutes, can be grasped using only a snippet clip of a few seconds. Nevertheless, current networks are not designed to share long-term information across the full video.

VTN's temporal processing component is based on a Longformer [1]. This type of transformer-based model can process a long sequence of thousands of tokens. The attention mechanism proposed by the Longformer makes it feasible to go beyond short clip processing and maintain global attention, which attends to all tokens in the input sequence.

In addition to long sequence processing, we also explore an important trade-off in machine learning – *speed vs. accuracy*. Our framework demonstrates a superior balance of this trade-off, both during training and also at inference time. In training, even though wall runtime per epoch is either equal or greater, compared to other networks, our approach requires much fewer passes of the training dataset to reach its maximum performance; end-to-end, compared to state-or-the-art networks, this results in a $16.1\times$ faster training. At inference time, our approach can handle both multi-view and full video analysis while maintaining similar accuracy. In contrast, other networks' performance significantly decreases when analyzing the full video in a single pass. In terms of *GFLOPS x Views*, their inference cost is considerably higher than those of VTN, which concludes to a $1.5\times$ fewer GFLOPS and a $5.1\times$ faster validation wall runtime.

Our framework's structure components are modular (Fig. 1). First, the 2D spatial backbone can be replaced with any given network. The attention-based module can stack up more layers, more heads or can be set to a different Transformers model that can process long sequences. Finally, the classification head can be modified to facilitate different video-based tasks, like temporal action localization.

2. Related Work

Spatial-temporal networks. Most recent studies in video recognition suggested architectures that are based on 3D ConvNets [20, 36]. In [5], a two-stream architecture was used, one stream for RGB inputs and another for Optical Flow (OF) inputs. Residual connections are inserted into

the two-stream architecture to allow a direct link between RGB and OF layers. The idea of inflating 2D ConvNets into their 3D counterpart (I3D) was introduced in [3]. I3D takes 2D ConvNets and expands its layers into 3D. Therefore it allows to leverage pre-trained state-of-the-art image recognition architectures in the spatial-temporal domain and apply them for video-based tasks.

Non-local Neural Networks (NLN) [39] introduced a non-local operation, a type of self-attention, that computes responses based on relationships between different locations in the input signal. NLN demonstrated that the core attention mechanism in Transformers can produce good results on video tasks, however it is confined to processing only short clips. In order to extract long temporal context, [41] introduced a long-term feature bank that acts as the entire video memory and a Feature Bank Operator (FBO) that computes interactions between short-term and long-term features. However, it requires precomputed features, and it is not efficient enough to support end-to-end training of the feature extraction backbone.

SlowFast [14] explored a network architecture that operates in two pathways and different frame rates. Lateral connections fuse the information between the slow pathway, focused on the spatial information, and the fast pathway focused on temporal information.

The X3D study [13] builds on top of SlowFast. It argues that in contrast to image classification architectures, which have been developed via a rigorous evolution, the video architectures have not been explored in detail, and historically are based on expanding image-based networks to fit the temporal domain. X3D introduces a set of networks that progressively expand in different axes, e.g., temporal, frame rate, spatial, width, bottleneck width, and depth. Compared to SlowFast, it offers a lightweight network (in terms of GFLOPS and parameters) with similar performance.

Transformers in computer vision. The Transformers architecture [37] reached state-of-the-art results in many NLP tasks, making it the de-facto standard. Recently, Transformers are starting to disrupt the field of computer vision, which traditionally depends on deep ConvNets. Studies like ViT and DeiT for image classification [10, 35], DETR for object detection and panoptic segmentation [2], and VisTR for

model	pretrain (ImageNet accuracy (%))	top-1	top-5
R50-VTN	ImageNet (76.2 [27])	71.2	90.0
R101-VTN	ImageNet (77.4 [27])	72.1	90.3
DeiT-B-VTN	ImageNet (81.8 [35])	75.5	92.2
DeiT-BD-VTN	ImageNet (83.4 [35])	75.6	92.4
ViT-B-VTN	ImageNet-21K (84.0 [10])	78.6	93.7
ViT-B-VTN†	ImageNet-21K (84.0 [10])	**79.8**	**94.2**

Table 1. VTN performance on Kinetics-400 validation set for different backbone variations. A full video inference is used. We show top-1 and top-5 accuracy. We report what pre-training was done for each backbone and the related single-crop top-1 accuracy on ImageNet. (†) Training with extensive data augmentation.

video instance segmentation [40] are some examples showing promising results when using Transformers in the computer vision field. Binding these results with the sequential nature of video makes it a perfect match for Transformers.

Applying Transformers on long sequences. BERT [8] and its optimized version RoBERTa [24] are transformer-based language representation models. They are pre-trained on large unlabeled text and later fine-tuned on a given target task. With minimal modification, they achieve state-of-the-art results on a variety of NLP tasks.

One significant limitation of these models, and Transformers in general, is their ability to process long sequences. This is due to the self-attention operation, which has a complexity of $O(n^2)$ per layer (n is sequence length) [37].

Longformer [1] addresses this problem and enables lengthy document processing by introducing an attention mechanism with a complexity of $O(n)$. This attention mechanism combines a sliding window local-context self-attention and task-specific global attention.

Similar to ConvNets, stacking up multiple windowed attention layers results in a larger receptive field. This property of Longformer gives it the ability to integrate information across the entire sequence. The global attention part focuses on pre-selected tokens (like the $[CLS]$ token) and can attend to all other tokens across the input sequence.

3. Video Transformer Network

Video Transformer Network (VTN) is a generic framework for video recognition. It operates with a single stream of data, from the frames level up to the objective task head. In the scope of this study, we demonstrate our approach using the action recognition task by classifying an input video to the correct action category.

The architecture of VTN is modular and composed of three consecutive parts. A 2D spatial feature extraction model (spatial backbone), a temporal attention-based encoder, and a classification MLP head. Fig. 1 demonstrates our architecture layout.

VTN is scalable in terms of video length during inference, and enables the processing of very long sequences. Due to memory limitation, we suggest several types of inference methods. (1) Processing the entire video in an end-to-end manner. (2) Processing the video frames in chunks, extracting features first, and then applying them to the temporal attention-based encoder. (3) Extracting all frames' features in advance and then feed them to the temporal encoder.

3.1. Spatial backbone

The spatial backbone operates as a learned feature extraction module. It can be any network that works on 2D images, either deep or shallow, pre-trained or not, convolutional- or transformers-based. And its weights can be fixed (pre-trained) or trained during the learning process.

3.2. Temporal attention-based encoder

As suggested by [37], we use a Transformer model architecture that applies attention mechanisms to make global dependencies in a sequence data. However, Transformers are limited by the number of tokens they can process at the same time. This limits their ability to process long inputs, such as videos, and incorporate connections between distant information.

In this work, we propose to process the entire video at once during inference. We use an efficient variant of self-attention, that is not all-pairwise, called Longformer [1]. Longformer operates using sliding window attention that enables a linear computation complexity. The sequence of feature vectors of dimension d_{backbone} (Sec. 3.1) is fed to the Longformer encoder. These vectors act as the 1D tokens embedding in the standard Transformer setup.

Like in BERT [8] we add a special classification token ($[CLS]$) in front of the features sequence. After propagating the sequence through the Longformer layers, we use the final state of the features related to this classification token as the final representation of the video and apply it to the given classification task head. Longformer also maintains global attention on that special $[CLS]$ token.

3.3. Classification MLP head

Similar to [10], the classification token (Sec. 3.2) is processed with an MLP head to provide a final predicted category. The MLP head contains two linear layers with a GELU non-linearity and Dropout between them. The input token representation is first processed with a Layer normalization.

3.4. Looking beyond a short clip context

The common approach in recent studies for video action recognition uses 3D-based networks. During inference, due to the addition of a temporal dimension, these networks are

# layers	top-1	top-5
1	**78.6**	93.4
3	**78.6**	**93.7**
6	78.5	93.6
12	78.3	93.3

PE	shuffle	top-1	top-5
learned	-	78.4	93.5
learned	✓	78.8	93.6
fixed	-	78.3	93.7
fixed	✓	78.5	93.7
no	-	78.6	93.7
no	✓	**78.9**	93.7

temporal footprint	# frames	top-1	top-5
2.56	16	78.2	93.4
2.56	32	78.2	**93.6**
5.12	16	**78.6**	93.4
5.12	32	78.5	93.5
10.0	16	78.0	93.3

finetune	top-1	top-5
-	71.6	90.3
✓	**78.6**	**93.7**

(a) **Depth**: Comparing different numbers of attention layers in the Longformer.

(b) **Positional embedding**: Evaluating the impact of different types of PE methods, with and without shuffling the input frames. We conducted this experiment with an older experimental setup using a different learning rate scheduler, so the results of the *learned* without *shuffle* are slightly different from what we report in other tables: 78.4% *vs.* 78.6%.

(c) **Temporal footprint**: Comparing the impact of temporal footprint size (in seconds) and the number of frames in a clip.

(d) **Finetune**: Training a ViT-B-VTN with three attention layers with and without fine-tuning the 2D backbone. All other hyperparameters remain the same.

Table 2. Ablation experiments on Kinetics-400. The results are top-1 and top-5 accuracy (%) on the validation set using the full video inference approach.

limited by memory and runtime to clips of a small spatial scale and a low number of frames. In [3], the authors use the whole video during inference, averaging predictions temporally. More recent studies that achieved state-of-the-art results processed numerous, but relatively short, clips during inference. In [39], inference is done by sampling ten clips evenly from the full-length video and average the softmax scores to achieve the final prediction. SlowFast [14] follows the same practice and introduces the term *"view"* – a temporal clip with a spatial crop. SlowFast uses ten temporal clips with three spatial crops at inference time; thus, 30 different views are averaged for the final prediction. X3D [13] follows the same practice, but in addition, it uses larger spatial scales to achieve its best results on 30 different views.

This common practice of multi-view inference is somewhat counterintuitive, especially when handling long videos. A more intuitive way is to "look" at the entire video context before deciding on the action, rather than viewing only small portions of it. Fig. 2 shows 16 frames extracted evenly from a video of the *abseiling* category. The actual action is obscured or not visible in several parts of the video; this might lead to a false action prediction in many views. The potential in focusing on the segments in the video that are most relevant is a powerful ability. However, full video inference produces poor performance in methods that were trained using short clips (Table 3 and 4). In addition, it is also limited in practice due to hardware, memory, and runtime aspects.

4. Video Action Recognition with VTN

In order to evaluate our approach and the impact of context attention on video action recognition, we use several spatial backbones pre-trained on 2D images.

ViT-B-VTN. Combining the state-of-the-art image classification model, ViT-Base [10], as the backbone in VTN. We use a ViT-Base network that was pre-trained on ImageNet-21K. Using ViT as the backbone for VTN produces an end-to-end transformers-based network that uses attention both for the spatial and temporal domains.

R50/101-VTN. As a comparison, we also use a standard 2D ResNet-50 and ResNet-101 networks [19], pre-trained on ImageNet.

DeiT-B/BD/Ti-VTN. Since ViT-Base was trained on ImageNet-21K we also want to compare VTN by using similar networks trained on ImageNet. We use the recent work of [35] and apply DeiT-Tiny, DeiT-Base, and DeiT-Base-Distilled as the backbone for VTN.

4.1. Implementation Details

Training. The spatial backbones we use were pre-trained on either ImageNet or ImageNet-21k. The Longformer and the MLP classification head were randomly initialized from a normal distribution with zero mean and 0.02 std. We train the model end-to-end using video clips. These clips are formed by choosing a random frame as the starting point, then sampling 2.56 or 5.12 seconds as the video's temporal footprint. The final clip frames are subsampled uniformly to a fixed number of frames $N (N = 16, 32)$, depending on the setup.

For the spatial domain, we randomly resize the shorter side of all the frames in the clip to a $[256, 320]$ scale and randomly crop all frames to 224×224. Horizontal flip is also applied randomly on the entire clip.

The ablation experiments were done on a 4-GPU machine. Using a batch size of 16 for the ViT-VTN (on 16 frames per clip input) and a batch size of 32 for the R50/101-VTN. We use an SGD optimizer with an initial learning rate of 10^{-3} and a different learning rate reduction policy, steps-based for the ViT-VTN versions and cosine

Figure 3. Illustrating all the single-head first attention layer weights of the $[CLS]$ token *vs.* 16 frames pulled evenly from a video. High weight values are represented by a *warm* color (yellow) while low values by a *cold* color (blue). The video's segments in which *abseiling* category properties are shown (*e.g.*, shackle, rope) exhibit higher weight values compared to segments in which non-relevant information appears (*e.g.*, shoes, people). The model prediction is *abseiling* for this video.
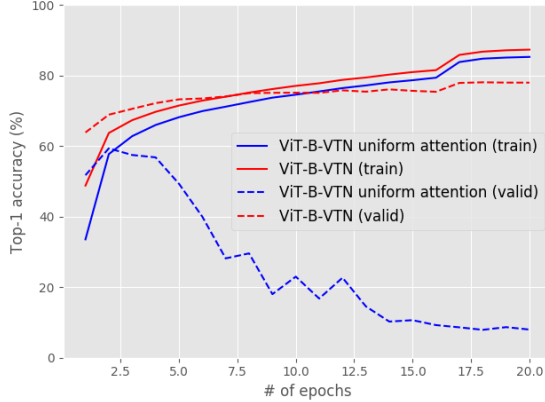

Figure 4. Evaluating the influence of attention on the training (solid line) and validation (dashed line) curves for Kinetics-400. A similar ViT-B-VTN with three Longformer layers is trained for both cases, and we modify the attention heads between a learned one (red) and a fixed uniform version (blue).

schedule decay for the R50/101-VTN versions. In order to report the wall runtime, we use an 8-V100-GPU machine.

Since we use 2D models as the spatial backbone, we can manipulate the input clip shape $x_{\text{clip}} \in \mathbb{R}^{B \times C \times T \times H \times W}$ by stacking all frames from all clips within a batch to create a single frames batch of shape $x \in \mathbb{R}^{(B \cdot T) \times C \times H \times W}$. Thus, during training, we propagate all batch frames in a single forward-backward pass.

For the Longformer, we use an *effective attention window* of size 32, which was applied for each layer. Two other hyperparameters are the dimensions set for the *Hidden size* and the *FFN inner hidden size*. These are a direct derivative of the spatial backbone. Therefore, in R50/101-VTN we use 2048 and 4096, respectively, and for ViT-B-VTN we use 768 and 3072, respectively. In addition, we apply *Attention Dropout* with a probability of 0.1. We also explore the impact of the number of Longformer layers.

The positional embedding (PE) information is only relevant for the temporal attention-based encoder (Fig. 1). We explore three positional embedding approaches (Table 2b): (1) *Learned positional embedding* - since a clip is repre-

sented using frames taken from the full video sequence, we can learn an embedding that uses as input the frame location (index) in the original video, giving the Transformer information regarding the position of the clip in the entire sequence; (2) *Fixed absolute encoding* - we use a similar method to the one in DETR [2], and modified it to work on the temporal axis only; and (3) *No positional embedding* - no information is added in the temporal dimension, but we still use the global position to mark the special $[CLS]$ token position.

Inference. In order to show a comparison between different models, we use both the common practice of inference in multi-views and a full video inference approach (Sec. 3.4).

In the multi-view approach, we sample 10 clips evenly from the video. For each clip, we first resize the shorter side to 256, then take three crops of size 224×224 from the left, center, and right. The result is 30 views per video, and the final prediction is an average of all views' softmax scores.

In the full video inference approach, we read all the frames in the video. Then, we align them for batching purposes, by either sub- or up-sampling, to 250 frames uniformly. In the spatial domain, we resize the shorter side to 256 and take a center crop of size 224×224.

5. Experiments

5.1. Ablation Experiments on Kinetics-400

Kinetics-400 dataset. The original Kinetics-400 dataset [21] consists of 246,535 training videos and 19,761 validation videos. Each video is labeled with one of 400 human action categories, curated from YouTube videos. Since some YouTube links are expired, we could only download 234,584 of the original dataset, thus missing 11,951 videos from the training set, which are about 5%. This leads to a slight drop in performance of about 0.5%[1].

In the validation set, we are missing one video. To test our data's validity and compare it to previous studies, we

---

[1]https://github.com/facebookresearch/
video-nonlocal-net/blob/master/DATASET.md

model	training wall runtime (minutes)	# training epochs	validation wall runtime (minutes)	inference approach	params (M)	top-1	top-5
I3D*	30	-	84	multi-view	28	73.5 [11]	90.8 [11]
NL I3D (our impl.)	68	50	150	multi-view	54	74.1	91.7
NL I3D (our impl.)	68	50	31	full-video	54	72.1	90.5
SlowFast-8X8-R50*	70	196 [14]	140	multi-view	35	77.0 [11]	92.6 [11]
SlowFast-8X8-R50*	70	196 [14]	26	full-video	35	68.4	87.1
SlowFast-16X8-R101*	220	196 [14]	244	multi-view	60	78.9 [11]	93.5 [11]
R50-VTN	62	40	32	full-video	168	71.2	90.0
R101-VTN	110	40	32	full-video	187	72.1	90.3
DeiT-Ti-VTN (3 layers)	52	60	30	full-video	10	67.8	87.5
ViT-B-VTN (1 layer)	107	25	48	full-video	96	78.6	93.4
ViT-B-VTN (3 layers)	130	25	52	full-video	114	78.6	93.7
ViT-B-VTN (3 layers)†	130	35	52	full-video	114	**79.8**	**94.2**

Table 3. To measure the overall time needed to train each model, we observe how long it takes to train a single epoch and how many epochs are required to achieve the best performance. We compare these numbers to the validation top-1 and top-5 accuracy on Kinetics-400 and the number of parameters per model. To measure the training wall runtime, we ran a single epoch for each model, on the same 8-V100-GPU machine, with a 16GB memory per GPU. The models marked by (*) were taken from the *PySlowFast* GitHub repository [11]. We report the accuracy as written in the *Model Zoo*, which was done using the 30 multi-view inference approach. To measure the wall runtime, we used the code base of *PySlowFast*. To calculate the SlowFast-16X8-R101 time on the same GPU machine, we used a batch size of 16. The number of epochs is reported, when possible, based on the original model paper. All other models, including the NL I3D, are trained using our codebase and evaluated with a full video inference approach. (†) The model in the last row was trained with extensive data augmentation.
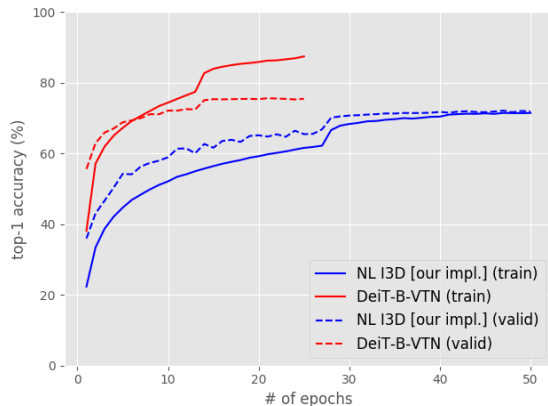

Figure 5. Kinetics-400 learning curves for our implementation of NL I3D (blue) *vs.* DeiT-B-VTN (red). We show the top-1 accuracy for the train set (solid line) and the validation set (dash line). Top-1 accuracy during training is calculated based on a single random clip, while during validation we use the full video inference approach. DeiT-B-VTN shows high performance in every step of the training and validation process. It reaches its best accuracy after only 25 epochs compared to the NL I3D that needs 50 epochs.

evaluated the SlowFast-8X8-R50 model, published in *PySlowFast*[11], on our validation data. We got 76.45% top1-accuracy *vs.* the reported 77%, thus a drop of 0.55%. This drop might be related to different *FFmpeg* encoding and rescaling of the videos. From this point forward, when comparing to other networks, we report results taken from the original studies except when we evaluate them on the full video inference in which we use our validation set. All our approach results are reported based on our validation set.

Spatial backbone variations. We start by examining how different spatial backbone architectures impact VTN performance. Table 1 shows a comparison of different VTN variants and the pretrain dataset the backbone was first trained on. ViT-B-VTN is the best performing model and reaches 78.6% top-1 accuracy and 93.7% top-5 accuracy. The pretraining dataset is important. Using the same ViT backbone, only changing between DeiT (pre-trained on ImageNet) and ViT (pre-trained on ImageNet-21K) we get an improvement in the results.

Longformer depth. Next, we explore how the number of attention layers impacts the performance. Each layer has 12 attention heads and the backbone is ViT-B. Table 2a shows the validation top-1 and top-5 accuracy for 1, 3, 6, and 12 attention layers. The comparison shows that the difference in performance is small. This is counterintuitive to the fact that deeper is better. It might be related to the fact that Kinetics-400 videos are relatively short, around 10 seconds. We believe that processing longer videos will benefit from a large receptive field obtained by using a deeper Longformer.

Longformer positional embedding. In Table 2b we compare three different positional embedding methods, focusing on learned, fixed, and no positional embedding. All versions are done with a ViT-B-VTN, a temporal footprint of 5.12 seconds, and a clip size of 16 frames. Surprisingly, the one without any positional embedding achieved slightly better results than the fixed and learned versions.

model	inference approach	# frames per inference view	# of views	test crop size	inference GFLOPs	top-1
NL I3D (our impl.)	multi-view	32	30	224	2,625	74.1
NL I3D (our impl.)	full-video	250	1	224	1,266	72.2
SlowFast-8X8-R50*	multi-view	32	30	256	1,971	77.0 [11]
SlowFast-8X8-R50*	full-video	250	1	256	517	68.4
SlowFast-16X8-R101*	multi-view	64	30	256	6,390	78.9 [11]
SlowFast-16X8-R101*	full-video	250	1	256	838	-
R50-VTN (3 layers)	multi-view	16	30	224	2,106	70.9
R50-VTN (3 layers)	full-video	250	1	224	1,059	71.2
R101-VTN (3 layers)	multi-view	16	30	224	3,895	72.0
R101-VTN (3 layers)	full-video	250	1	224	1,989	72.1
ViT-B-VTN (1 layer)	multi-view	16	30	224	8,095	78.5
ViT-B-VTN (1 layer)	full-video	250	1	224	4,214	78.6
ViT-B-VTN (3 layers)	multi-view	16	30	224	8,113	78.6
ViT-B-VTN (3 layers)	full-video	250	1	224	4,218	78.6

Table 4. Comparing the number of GFLOPs during inference. The models marked by (*) were taken from the *PySlowFast* GitHub repository [11]. We reproduced the SlowFast-8X8-R50 results by using the repository and our Kinetics-400 validation set and got 76.45% compared to the reported value of 77%. When running this model using a full video inference approach, we get a significant drop in performance of about 8%. We did not run the SlowFast-16X8-R101 because it was not published. The inference GFLOPs is reported by multiplying the number of views with the GFLOPs calculated per view. ViT-B-VTN with one layer achieves 78.6% top-1 accuracy, a 0.3% drop compared to SlowFast-16X8-R101 while using $1.5\times$ fewer GFLOPS.

As this is an interesting result, we also use the same trained models and evaluate them after randomly shuffling the input frames only in the validation set videos. This is done by first taking the unshuffled frame embeddings, then shuffle their order, and finally add the positional embedding. This raised another surprising finding, in which the shuffle version gives better results, reaching 78.9% top-1 accuracy on the no positional embedding version. Even in the case of learned embeddings it does not have a diminishing effect. Similar to the Longformer depth, we believe that this might be related to the relatively short videos in Kinetics-400, and longer sequences might benefit more from positional information. We also argue that this could mean that Kinetics-400 is primarily a static frame, appearance based classification problem rather than a motion problem [31].

Temporal footprint and number of frames in a clip. We also explore the effect of using longer clips in the temporal domain and compare a temporal footprint of 2.56 *vs.* 5.12 seconds. And also how the number of frames in the clip impact the network performance. The comparison is done on a ViT-B-VTN with one attention layer in the Longformer. Table 2c shows that top-1 and top-5 accuracy are similar, implying that VTN is agnostic to these hyperparameters.

Finetune the 2D spatial backbone. Instead of fine-tuning the spatial backbone, by continuing the back-propagation process, when training VTN, we can use a *frozen* 2D network solely for feature extraction. Table 2d shows the validation accuracy when training a ViT-B-VTN with three attention layers with and without also training the backbone. Fine-tuning the backbone improves the results by 7% in Kinetics-400 top-1 accuracy.

Does attention matter? A key component in our approach is the impact of attention functionally on the way VTN perceives the full video sequence. To convey this impact we train two VTN networks, using three layers in the Longformer, but with a single head for each layer. In one network the head is trained as usual, while in the second network instead of computing attention based on query/key dot products and softmax, we replace the attention matrix with a hard-coded uniform distribution that is not updated during back-propagation.

Fig. 4 shows the learning curves of these two networks. Although the training has a similar trend, the learned attention performs better. In contrast, the validation of the uniform attention collapses after a few epochs demonstrating poor generalization of that network. Further, we visualize the $[CLS]$ token attention weights by processing the same video from Fig. 2 with the single-head trained network and depicted, in Fig. 3, all the weights of the first attention layer aligned to the video's frames. Interestingly, the weights are much higher in segments related to the *abseiling* category. In Appendix A. we show a few more examples.

Training and validation runtime. An interesting observation we make concerns the training and validation wall runtime of our approach. Although our networks have more parameters, and therefore, are longer to train and test, they are actually much faster to converge and reach their best performance earlier. Since they are evaluated using a single view of all video frames, they are also faster during validation. Table 3 shows a comparison of different models and several VTN variants. Compared to the state-of-the-art SlowFast model, our ViT-B-VTN with one layer achieves almost the same results but completes an epoch faster while requiring

model	pretrain	MiT version	inputs	top-1	top-5
ResNet50	ImageNet	v1	RGB	27.2 [26]	51.7 [26]
I3D	ImageNet	v1	RGB + OF	29.5 [26]	56.1 [26]
AssembelNet-50 [29]	Kinetics400	v1	RGB + OF	33.9	60.9
AssembelNet-101 [29]	Kinetics400	v1	RGB + OF	34.3	62.7
I3D	ImageNet	v2	RGB	28.4*	54.5*
NL I3D (our impl.)	Kinetics400	v2	RGB	30.1	57.3
ViT-B-VTN	Kinetics400	v2	RGB	**37.4**	65.3
ViT-B-VTN (w/ shuffle)	Kinetics400	v2	RGB	**37.4**	**65.4**

Table 5. Comparison with the state-of-the-art on MiT-v1 and MiT-v2. The results marked by (*) are based on MiT GitHub repository[3].

fewer epochs. This accumulates to a $16.1\times$ faster end-to-end training. The validation wall runtime is also $5.1\times$ faster due to the full video inference approach.

To better demonstrate the fast convergence of our approach, we wanted to show an apples-to-apples comparison of different training and evaluating curves for various models. However, since other methods use the multi-view inference only post-training, but use a single view evaluation while training their models, this was hard to achieve. Thus, to show such comparison and give the reader additional visual information, we trained a NL I3D (pre-trained on ImageNet) with a full video inference protocol during validation (using our codebase and reproduced the original model results). We compare it to DeiT-B-VTN which was also pre-trained on ImageNet. Fig. 5 shows that the VTN-based network converges to better results much faster than the NL I3D and enables a much faster training process compared to 3D-based networks.

Data augmentation. Recent studies showed that data augmentation significantly improves the performance of transformers-based models [35]. To demonstrate its impact on VTN, we apply extensive data augmentation as suggested in DeiT [35] and RandAugment [6]. Table 3 shows that our method reaches 79.8% top-1 accuracy, a 1.2% improvement *vs.* the same model trained without such augmentations. Training with augmentations requires 10 more epochs but didn't impact the training wall runtime.

Final inference computational complexity. Finally, we examine what is the final inference computational complexity for various models by measuring GFLOPs. Although other models need to evaluate multiple views to reach their highest performance, ViT-B-VTN performs almost the same for both inference protocols. Table 4 shows a significant drop of about 8% when evaluating the SlowFast-8X8-R50 model using the full video approach. In contrast, ViT-B-VTN maintains the same performance while requiring, end-to-end, fewer GFLOPs at inference.

5.2. Experiments on Moments in Time

The Moments in Time (MiT) dataset is a large-scale collection of short (3 seconds) videos [26]. MiT is a chal-

lenging dataset, with state-of-the-art results just above 34% top-1 accuracy [29]. In this work, we use MiT-v2, consisting of 727,305 training videos and 30,500 validation videos. Each video is labeled with one of 305 classes of dynamic events. Although previous studies worked on MiT-v1 (802,264 training videos, 33,900 validation videos, 339 classes), this dataset is no longer available. In Table 5, we show the results of various models on MiT-v1 and MiT-v2. Since the relation between v1 and v2 in terms of performance was not established and thus unknown, we also trained our implementation of NL I3D on MiT-v2 using RGB inputs and achieved comparable results to those of I3D (RGB+OF) published on MiT-v1 [26]. Furthermore, ViT-B-VTN achieves the highest top-1 accuracy on MiT-v2 while using only RGB frames as input.

6. Conclusion

We presented a modular transformer-based framework for video recognition tasks. Our approach introduces an efficient way to evaluate videos at scale, both in terms of computational resources and wall runtime. It allows full video processing during test time, making it more suitable for dealing with long videos. VTN is especially useful for video editing tasks, in which long-term video understanding is of great importance. Various video editing-related tasks, such as video shortening and summarization, rely on the ability to process very long videos in a single full-video pass efficiently. Although current video classification benchmarks are not ideal for testing long-term video processing ability, hopefully, in the future, when such datasets become available, models like VTN will show even larger improvements compared to 3D ConvNets. Thus, enabling assisted video editing using advanced machine learning and AI technologies.

---

[3]https://github.com/zhoubolei/moments_models

References

[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. Long-former: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[4] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018.

[5] R Christoph and Feichtenhofer Axel Pinz. Spatiotemporal residual networks for video action recognition. *Advances in Neural Information Processing Systems*, pages 3468–3476, 2016.

[6] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[11] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. https://github.com/facebookresearch/slowfast, 2020.

[12] Linxi Fan, Shyamal Buch, Guanzhi Wang, Ryan Cao, Yuke Zhu, Juan Carlos Niebles, and Li Fei-Fei. Rubiksnet: Learn-able 3d-shift for efficient video action recognition. In *European Conference on Computer Vision*, pages 505–521. Springer, 2020.

[13] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 203–213, 2020.

[14] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211, 2019.

[15] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019.

[16] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[17] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2012.

[21] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[23] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7083–7093, 2019.

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[26] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown,

Quanfu Fan, Dan Gutfreund, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):502–508, 2019.

[27] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[29] Michael S. Ryoo, AJ Piergiovanni, Mingxing Tan, and Anelia Angelova. Assemblenet: Searching for multi-stream neural connectivity in video architectures. In *International Conference on Learning Representations*, 2020.

[30] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[31] Laura Sevilla-Lara, Shengxin Zha, Zhicheng Yan, Vedanuj Goswami, Matt Feiszli, and Lorenzo Torresani. Only time can tell: Discovering temporal data for temporal modeling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 535–544, 2021.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[33] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[34] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[35] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[36] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc., 2017.

[38] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[39] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[40] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020.

[41] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 284–293, 2019.

[42] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8739–8748, 2018.