Enhancing Video Transformers for Action Understanding with VLM-aided Training

Hui Lu[1*], Hu Jian[2], Albert Ali Salah[1], Ronald Poppe[1]

[1*]Utrecht University, Utrecht, The Netherlands.
[2]University of Science and Technology of China, Hefei, China.

*Corresponding author(s). E-mail(s): h.lu1@uu.nl;

Abstract

Owing to their ability to extract relevant spatio-temporal video embeddings, Vision Transformers (ViTs) are currently the best performing models in video action understanding. However, their generalization over domains or datasets is somewhat limited. In contrast, Visual Language Models (VLMs) have demonstrated exceptional generalization performance, but are currently unable to process videos. Consequently, they cannot extract spatio-temporal patterns that are crucial for action understanding. In this paper, we propose the Four-tiered Prompts (FTP) framework that takes advantage of the complementary strengths of ViTs and VLMs. We retain ViTs' strong spatio-temporal representation ability but improve the visual encodings to be more comprehensive and general by aligning them with VLM outputs. The FTP framework adds four feature processors that focus on specific aspects of human action in videos: action category, action components, action description, and context information. The VLMs are only employed during training, and inference incurs a minimal computation cost. Our approach consistently yields state-of-the-art performance. For instance, we achieve remarkable top-1 accuracy of 93.8% on Kinetics-400 and 83.4% on Something-Something V2, surpassing VideoMAEv2 by 2.8% and 2.6%, respectively.

Keywords: Video action understanding, Visual Language Models, Vision Transformers
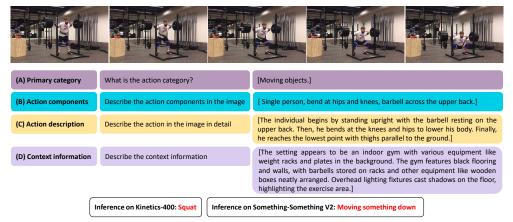
(A) Primary category	What is the action category?	[Moving objects.]
(B) Action components	Describe the action components in the image	[Single person, bend at hips and knees, barbell across the upper back.]
(C) Action description	Describe the action in the image in detail	[The individual begins by standing upright with the barbell resting on the upper back. Then, he bends at the knees and hips to lower his body. Finally, he reaches the lowest point with thighs parallel to the ground.]
(D) Context information	Describe the context information	[The setting appears to be an indoor gym with various equipment like weight racks and plates in the background. The gym features black flooring and walls, with barbells stored on racks and other equipment like wooden boxes neatly arranged. Overhead lighting fixtures cast shadows on the floor, highlighting the exercise area.]

Inference on Kinetics-400: Squat   Inference on Something-Something V2: Moving something down

Fig. 1: **Conceptual idea**. Different domains potentially emphasize other aspects of an action's performance, here reflected in the different label sets used in benchmark datasets. VLMs can provide relevant details of video content that might be insufficiently covered by a pretrained visual encoder. In the FTP framework, we employ the textual descriptions from four prompts to align a ViT's visual embeddings during training. This way, we generate richer, comprehensive video representations. During inference, we don't require the VLM and benefit from the more general feature representations to improve action understanding across domains.

1 Introduction

Video action understanding requires learning rich spatio-temporal representations that reflect human and object appearance and movement, the environment, and interactions between these three aspects. Visual Transformers (ViTs) [1] are currently the best performing backbones for spatio-temporal representation learning. To refine their spatio-temporal representations, researchers have applied different variants of self-attention in the spatial domain of ViTs, including Swin Transformer [2] and local attention [3]. These innovations have effectively reduced computational overhead and mitigated redundancy in the spatial domain. Moreover, various architectures have been proposed to extend Multi-Head Self-Attention (MHSA) along the temporal dimension [4–6] to capture long-term video dependencies [7], thus further enhancing the spatio-temporal modeling capabilities of ViTs.

However, when training and benchmarking ViTs for video action understanding, datasets pose different requirements in terms of the relevant spatio-temporal information that needs to be processed. For example, the Kinetics [8, 9] datasets focus on action types that are predominantly human-centric, while Something-Something V2 (SSV2) [10] emphasizes the relative movement during interactions. Training on a single dataset potentially leads to poor generalization to another. Ideally, video action understanding models should be able to capitalize on all aspects of the action, including the dynamics, interactions, appearance, and context.

Recently, by combining visual information with Large Language Models, visual language models (VLMs) such as GPT-4 [11], BLIP-2 [12], and LLaVA [13] have shown improved generalization on various tasks including image captioning [14] and image understanding [15]. However, current VLMs cannot process videos well due to the shortage of accurately annotated video-text pair, and the computation cost of VLMs during inference is much higher compared to ViTs [16, 17]. This limits their adoption in video action understanding tasks. Recent attempts to extract and concatenate keyframes from videos as the input for VLMs [18] significantly underperform compared to ViTs. One possible reason for this is that temporal information is less readily available from keyframes, especially if these cover a longer time interval.

We argue that ViTs' strength to extract rich spatio-temporal patterns from videos is partly complementary to VLMs' ability to generalize over various contexts. To take advantage of both, we propose to enhance ViTs' visual encoder with information provided by VLMs. By focusing on aspects that are not directly reflected in action labels, such as scene context, we force the ViT to provide a more comprehensive feature representation that generalizes to different domains and datasets. Since the additional information of VLMs is only used during training, this approach only incurs a minimal computational cost during inference.

To demonstrate the potential of this approach, we introduce the Four-Tiered Prompts (FTP) architecture that relies on the additional information obtained from four VLM prompts, along with the visual encoding obtained from a ViT. Contrastive learning is used to train four feature processors that focus on various aspects of actions in videos: action category, action components, action description, and context information, respectively. Together, these processors provide a rich representation of the action performance. By integrating the outputs of these feature processors, the ViT's generalization ability can be significantly improved. Consequently, we can train ViTs that can flexibly accommodate various label sets, see Figure 1 for an example. Our main contributions are:

1. We present a novel Four-Tiered Prompts (FTP) framework that focuses on different aspects of the video action. Because the additional VLM inputs are only required during training, the additional computation cost is negligible.
2. Our work demonstrates the potential of integrating language models into the video domain to enhance model performance.
3. We report strong performance on video action recognition benchmarks, consistently surpassing state-of-the-art methods by clear margins.

We discuss related works, and then detail our method in Section 3. We present our experiments and ablation study in Section 4, and conclude in Section 5.

2 Related Work

For video understanding, early works have explored various ways of extracting spatial-temporal information from videos. Video architectures such as 3D ConvNets [19–21] extend 2D image models to the spatio-temporal domain, handling both spatial

and temporal dimensions in parallel. There are also 2D ConvNets with temporal modules [22–26], designed to extend existing 2D CNN models on the temporal dimension.

Vision Transformers. Recent works have adapted vision transformers to the video domain and achieve superior performance compared to previous CNN-based architectures. For example, VTN [27] adopts ViT [1] to extract spatial features, followed by a Longformer [28] to capture temporal relationships. Both TimeSformer [4] and ViViT [5] further factorize different spatial and temporal attentions for transformer encoders to achieve better performance. Some works explore how to reduce the computational cost of the space-time attention. MViT [29, 30] is a hierarchical transformer with several channel-resolution scale stages, with pooling attention to reduce computation. In contrast, Video Swin Transformer [31] introduces an inductive bias of locality for videos and achieves a favorable speed-accuracy trade-off. Uniformer [32] captures local spatio-temporal context and global token dependency by convolution in early layers and by a transformer in deeper layers, respectively. Video Mobile-Former [33] integrates 3D CNNs and spatial-temporal self-attention mechanisms to enhance efficiency. Despite their ability to learn spatio-temporal representations from videos, the generalization of current ViTs is not ideal. Specially, when the ViT is trained and performs well on one dataset, its performance significantly potentially deteriorates when tested on dataset that focus on different aspects, such as the use of objects, or the importance of the environment.

Visual language models. Recently, by training on large numbers of image-text pairs, visual language models (VLMs) such as GPT-4 [11], BLIP-2 [12], and LLaVA [13] have shown excellent generalization performance on computer vision tasks. Current VLMs, however, are currently not capable of processing video data, resulting in subpar performance when dealing with sequential image inputs [16]. To address this deficit, Lin et al. [34] extract the most important keyframe and prompt VLMs to recognize the action. Himakunthala et al. [18] instead extract multiple keyframes and concatenate them side-by-side as a single image to prompt VLMs. While this approach can reveal various phases of action performance, dynamics cannot be properly extracted. As a result, and despite a significantly higher computation cost during inference, the performance of these methods falls short in comparison to ViTs.

3 Four-Tiered Prompts Framework

We first provide an overview of our proposed four-tiered prompt (**FTP**) framework, before discussing the two-stage training process.

3.1 Framework

The FTP architecture is shown in Figure 2(a). Taking a ViT as the basis, we add four feature processors after the visual encoder. Each processor is trained to focus on specific aspects of the video contents. The outputs of the four processors are integrated and processed to produce the classification.
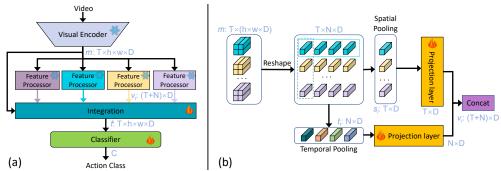
Fig. 2: **Architecture of the FTP framework**. (a) Four feature processors attend to different aspects of the video contents. Their outputs are integrated and subsequently classified. (b) Architecture of a feature processor, with spatial and temporal pooling and projection. The outputs are concatenated to produce the output v_i.

Visual encoder. The visual encoder processes the input video and generates feature map $m \in \mathbb{R}^{T \times h \times w \times D}$, with T the number of frames, h and w the height and width of the feature map, and D the depth of the feature map.

Feature processor. We employ four feature processors i ($1 \leq i \leq 4$) to process m and to attend to specific aspects of the video. Refer to Figure 2(b) for an overview of the processing steps. In each processor, feature map m is spatially unfolded into a one-dimensional vector of size $N = h \times w$. We then apply pooling along the spatial dimension to produce video-level spatial representation $s_i \in \mathbb{R}^{T \times D}$. In parallel, we apply temporal pooling and obtain a temporal representation $t_i \in \mathbb{R}^{N \times D}$. Both pooled representations are projected onto their input dimensions with a trainable projection layer, and subsequently concatenated to obtain spatio-temporal feature representation $v_i \in \mathbb{R}^{(T+N) \times D}$. The dimensionality of v_i matches that of [35], which is crucial for the visual-textual alignment that we perform during training (see Section 3.2).
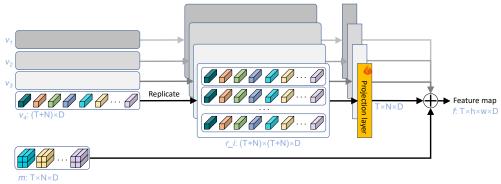

Fig. 3: **Integration process for feature processor and visual encoder outputs**. Each feature processor output v_i is replicated, projected, and finally element-wise summed with visual encoder output m.

5

Integration and classification. We first integrate the outputs of the feature processors v_i $(1 \leq i \leq 4)$ with feature map m from the pretrained visual encoder. To match the dimensions of v_i and m, we use an integration layer, shown in Figure 3. We first replicate each feature v_i to produce r_i. Subsequently, we use a trainable projection layer to project the video-level feature into the visual encoder's embedding space. The integrated output of the four feature processors is then element-wise summed with the feature map from the visual encoder, yielding a feature map $f \in \mathbb{R}^{T \times h \times w \times D}$ of the same dimension as m.

The final layers consist of two sequential transformer blocks, each containing a multi-head self-attention layer and a feed-forward network layer. After an average pooling layer, the final action classification is performed by a linear layer that maps the activations to the action class outputs. Architecture details appear in the supplementary material A.1

Compared to the original ViT architecture, the FTP framework only adds computation overhead for the four feature processors. Since these are light-weight, the framework incurs a negligible additional computation cost.

3.2 Training

We need to train the feature processors, their integration, and the classifier. The visual encoder of the ViT remains untouched, which significantly reduces the computational burden of the training. Training proceeds in two stages.
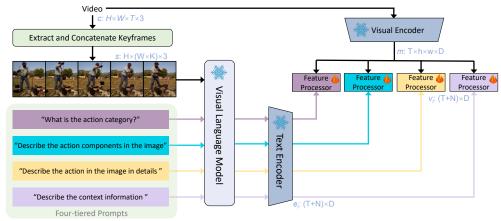

Fig. 4: **Training stage 1**. Concatenated keyframe images are processed by a VLM using four prompts. The outputs pass through a text encoder to yield text embeddings. The feature processors are trained with contrastive loss to project the visual embeddings onto the text embeddings. VLM, text encoder, and visual encoder are frozen.

Stage 1: Visual-text alignment. We first train the feature processors using supervision from a VLM and text encoder, shown in Figure 4. The goal of this stage is to be able to generate rich and general encodings for action videos. This stage only

needs to be performed once, to serve model development for a range of target domains. Both VLM and text encoder are used off-the-shelf.

Only during training stage 1, we augment the architecture with an additional processing pipeline that includes the VLM, see Figure 4. Because VLMs currently cannot process videos directly, we represent a video as a side-by-side concatenation of K keyframes extracted at equal time intervals, similar to [18]. The result is a single image $c \in \mathbb{R}^{H \times (W \times K) \times 3}$, with H and W the height and width of the video. We use GPT-4 [11] (alternatives tested in Section 4.3) to generate text descriptions for the keyframe image, according to four specific prompts: (A) "What is the action category?", (B) "Describe the action components in the image", (C) "Describe the action in the image in detail", (D) "Describe the context information". The four textual outputs are then processed by the pretrained text encoder from [35] to produce text embeddings $e_i \in \mathbb{R}^{(T+N) \times D}$ with prompt index i ($1 \leq i \leq 4$).

In parallel, we obtain for the sequence of T frames of the same training video a visual embedding m from the ViT's visual encoder. This embedding passes through the four feature processors i ($i \leq i \leq 4$), that produce the FTP-based visual encoding v_i. We now train the corresponding four feature processors using contrastive learning to transform each visual encoding v_i in such a way that it matches its corresponding textual embeddings e_i. Following previous works [35, 36], we use InfoNCE loss in the training process.

Stage 2: Model fine-tuning. Based on the availability of rich video encodings, we fine-tune the integration of the four feature processors and subsequent layers for a specific dataset or target domain (Figure 2(a)). Apart from a different set of action classes, each domain potentially differs in the importance of different aspects of the human action, such as the dynamics, objects, or the scene. In this stage, we do not require VLM and text encoder, and only train the integration of the feature processors and the classification layers. Owing to the richness of the video encodings that are produced in training stage 1, we learn to prioritize those features that are most meaningful to the target domain.

Specifically, with visual encoder and feature processors frozen, we fine-tune only the projection layer in the feature integration, and the two transformer blocks and final fully-connected layer in the classifier, using cross-entropy loss.

4 Experiments

We evaluate our FTP framework on several video action recognition benchmarks. We detail the experimental setup in Section 4.1, present and discuss the main results in Section 4.2, before presenting an ablation study in Section 4.3.

4.1 Experiment setup

Datasets. We evaluate our approach on four video action recognition and one action detection datasets: (a) **Kinetics** is among the most widely used action recognition benchmarks. We report on Kinetics-400 (**K400**) [8] with 400 classes and Kinetics-600 (**K600**) [37] with 600 classes. The sets comprise 240K/20K and 390K/30K training/validation videos with an average duration of 10 seconds, respectively. (b)

Method	Pretrain data	Input	Crops	Params	FLOPs	Top-1	Top-5
TimeSformer-L [4]	IN-21K	96×224^2	10×3	121M	7.1T	80.7	94.7
Mformer-HR [41]	IN-21K	16×336^2	10×3	109M	28.8T	81.1	95.2
UniFormerV1-B [32]	IN-1K	32×224^2	4×3	50M	3.1T	83.0	95.4
Video Swin-L [31]	IN-21K	32×384^2	10×5	200M	105.4T	84.9	96.7
ViViT-H [5]	JFT-300M	32×224^2	4×3	654M	47.8T	84.9	95.8
TokenLearner-L/10 [42]	JFT-300M	64×224^2	4×3	450M	48.9T	85.4	96.3
MViTv2-L [30]	IN-21K	40×312^2	5×3	218M	42.2T	86.1	97.0
VideoMAE-H [43]	IN-21K	16×224^2	5×3	633M	17.9T	86.6	97.1
MAE-H [44]	IN1K+K400+K600-1.9M	16×224^2	4×3	632M	25.1T	86.8	97.2
MaskFeat-L [45]	IN-21K	64×224^2	4×3	218M	45.5T	87.0	97.4
EVL-L/14 [46]	CLIP-400M	32×336^2	3×3	67M	19.1T	87.7	—
X-CLIP-L/14 [47]	CLIP-400M	16×336^2	4×3	453M	37.0T	87.7	—
MTV-H [6]	IN-21K+WTS-60M	32×224^2	4×3	1120M	44.5T	89.1	98.2
MTV-H [6]	IN-21K+WTS-60M	32×280^2	4×3	1330M	73.6T	89.9	98.3
VideoMAE V2-g [48]	Hybrid-1.35M[1]	64×266^2	2×3	1050M	160.3T	90.0	98.4
TubeVit-H [49]	IN-1K	32×224^2	4×3	635M	17.6T	90.9	98.9
InternVideo [50]	CLIP-400M+Hybrid-12M[2]	64×224^2	16×4	1300M	86.2T	91.1	98.9
UniFormerV2-B/16 [7]	CLIP-400M+K710-0.66M	8×224^2	4×3	115M	1.8T	85.6	97.0
UniFormerV2-L/14 [7]	CLIP-400M+K710-0.66M	32×224^2	2×3	354M	16.0T	89.3	98.2
UniFormerV2-L/14 [7]	CLIP-400M+K710-0.66M	64×336^2	2×3	354M	75.3T	90.0	98.4
FTP-UniFormerV2-B/16 (ours)	CLIP-400M+K710-0.66M	8×224^2	4×3	136M	2.2T	91.9	98.9
FTP-UniFormerV2-L/14 (ours)	CLIP-400M+K710-0.66M	32×224^2	2×3	402M	18.4T	93.4	99.3
FTP-UniFormerV2-L/14 (ours)	CLIP-400M+K710-0.66M	64×336^2	2×3	402M	78.7T	**94.3**	**99.4**

Table 1: Performance on K400. —: numbers not reported. Best results in **bold**. We report crops (temporal × spatial) and TFLOPs for inference.
[1] K710+SSV2+WebVid2M+AVA V2.2, and videos crawled from Instagram
[2] WebVid2M/10M+HowTo100M+K710+SSV2+AVA V2.2, and self-collected videos.

Something-Something V2 (SSV2) [10] includes 169K training and 25K validation videos with an average duration of 4 seconds. They are categorized into 174 motion-centric classes of humans performing actions with objects. (c) **UCF-101** [**38**] is a relatively small dataset, consisting of 9.5K training videos and 3.5K validation videos. (d) **HMDB51** [**39**] is also a compact video dataset, containing 3.5K training videos and 1.5K validation videos. We adhere to common evaluation protocols for UCF-101 and HMDB51, averaging results on the three training/validation splits. (e) **AVA V2.2** [40] is a challenging dataset for spatio-temporal localization of human actions with 211K training and 57K validation video segments. Together, these datasets present a significant variation in action performance, label sets, and complexity of the videos, thus allowing for a thorough investigation of the merits of our approach. More experiment results are shown in the supplementary material B.

Implementation details. Our FTP framework can be implemented with a range of ViTs. For our main results, we use the well-performing visual encoder from UniformerV2 [7]. It is based on CLIP-ViT [51], with different capacities (ViT-B, ViT-L), and pretrained on CLIP-400M. We insert the global UniBlocks in the last four layers to perform the multi-stage fusion. For our resulting FTP-UniFormerV2-B/16 and FTP-UniFormerV2-L/14, we use $T = 16$ and $T = 32$ franes, respectively. We use a resolution of 224^2, and additionally report on K400 with a resolution of 336^2 and $T = 64$. The number of parameters of these models ranges from 136–402, with TFLOPs between 2.2–78.7.

Following [7, 50], when reporting on K400/K600, we train on Kinetics-710 (K710). K710 combines the Kinetics-400/600/700 datasets, while removing test videos. For all other datasets, we fine-tune the pretrained models on the training set and report on

Method	Input	Crops	Params (M)	TFLOPs	Top-1	Top-5
TimeSformer-HR [4]	(16×448^2)	16×3	121	5.1	62.4	86.3
SlowFast R101 [21]	(64×224^2)	8×8	53	0.3	63.1	87.6
ViViT-L FE [5]	(32×224^2)	1×3	612	47.6	65.4	89.8
EVL-L/14 [46]	(32×336^2)	3×3	67	9.6	66.7	—
Mformer-L [41]	(32×224^2)	1×3	109	3.6	68.1	91.2
VIMPAC [53]	(10×256^2)	10×3	307	—	68.1	—
TDN R101 [54]	(24×224^2)	10×3	88	0.6	68.2	91.6
MTV-B [6]	(32×320^2)	4×3	310	11.2	68.5	90.4
Video Swin-B [31]	(32×224^2)	1×3	89	1.0	69.6	92.7
UniFormerV1-B [32]	(32×224^2)	1×3	50	0.8	71.2	92.8
BEVT [55]	(32×224^2)	—	88	1.0	71.4	—
MViTv2-B [30]	(32×224^2)	1×3	51	0.7	72.1	93.4
MaskFeat-L [45]	(64×312^2)	4×3	218	8.5	75.0	95.0
VideoMAE-L [43]	(32×224^2)	1×3	305	4.3	75.4	95.2
TubeViT-L [49]	(32×224^2)	4×3	311	9.5	76.1	95.2
VideoMAE V2-g [48]	(64×266^2)	5×3	1050	160.3	77.0	95.9
InternVideo [50]	(64×224^2)	16×4	1300	86.2	77.2	95.9
UniFormerV2-B/16 [7]	(16×224^2)	1×3	163	0.6	69.5	92.3
UniFormerV2-L/14 [7]	(32×224^2)	1×3	574	5.2	73.0	94.5
FTP-UniFormerV2-B/16 (ours)	(16×224^2)	1×3	183	1.1	77.3	96.7
FTP-UniFormerV2-L/14 (ours)	(32×224^2)	1×3	620	6.6	**79.8**	**98.9**

Table 2: Performance on SSV2. —: numbers not reported. Best results in **bold**.

the validation set. For action detection, we additionally predict bounding boxes before classification. We use Faster-RNN as in [7].

In training stage 1, we set the base learning rate to 1.0×10^{-5}, repeated sampling to 1, batch size to 512, and train for 800 epochs. We adopt sparse sampling [52]. The number of keyframes is set to $K = 5$. For training stage 2, we keep these settings but set the learning rate to 1.5×10^{-6} and train for 40 epochs. More details appear in the supplementary material A.2. All experiments are performed on 16 NVIDIA A100 GPUs.

4.2 Main results

We first address action recognition, before moving to action detection.

Kinetics-400. Results on K400 are summarized in Table 1. Our FTP models outperform previous top-perfoming methods by a clear margin. The best performing FTP-UniFormerV2-L/14 with 64×336^2 inputs achieves state-of-the-art performance with an unprecedented 94.3%. But even the significantly smaller FTP-UniFormerV2-B outperforms the previous best InternVideo [50], despite having only 10.5% of the parameters, $T = 8$ instead of 64 input frames, and having seen a significantly lower number of training videos. Compared to their vanilla counterparts, FTP-UniFormerV2 models achieve 4.3-6.1% higher top-1 on the already saturated results.

Something-Something V2. Table 2 presents comparisons to the state-of-the-art methods on SSV2. Our smaller FTP-UniFormerV2-B/16 model performs on par with the previous best approach InternVideo [50] while having only 1.3% of the TFLOPs and 14.1% of the parameters. Our increased performance thus is not due to additional model capacity or computation cost. Our FTP-UniFormerV2-L/14 model achieves

state-of-the-art performance of 79.8%, outperforming InternVideo and VideoMAE V2-g [48] by 2.6–2.8%.

UCF-101 and HMDB51. From Table 3, for UCF-101, we observe that our FTP-UniFormerV2-L/14 performs on par with the top model VideoMAE V2 [48], despite having much less parameters. For HMDB51, our method outperforms all tested methods, including the previous best VideoMAE V2 [48] (+3.4%) while requiring only 38.3% of the parameters.

Method	Params	UCF-101	HMDB51
VideoMoCo [56]	15M	78.7	49.2
MemDPC [57]	32M	86.1	54.5
Vi²CLR [58]	9M	89.1	55.7
VIMPAC [53]	307M	92.7	65.9
CORP [59]	32M	93.5	68.0
XDC [60]	33M	94.2	67.1
CVRL [61]	328M	94.4	70.6
GDT [62]	33M	95.2	72.8
MMVFAC [63]	94M	95.2	75.0
VideoMAE [43]	87M	96.1	73.3
MVD-L [64]	87M	97.5	79.7
VideoMAE V2 [48]	1050M	99.6	88.1
UniFormerV2-B/16 [7]	115M	96.8	80.0
UniFormerV2-L/14 [7]	354M	98.2	86.2
FTP-UniFormerV2-B/16 (ours)	136M	98.7	83.9
FTP-UniFormerV2-L/14 (ours)	402M	**99.7**	**91.5**

Table 3: Results on UCF-101/HMDB51.

Method	FLOPs	Param	mAP
SlowFast R101 [21]	138G	53M	23.8
HIT [65]	622G	198M	32.6
MViTv2-L [30]	2828G	213M	34.4
ST-MAE-H [44]	1193G	632M	36.2
VideoMAE-L [43]	597G	305M	37.0
MaskFeat [45]	2828G	218M	37.5
VideoMAE-H [43]	1192G	633M	39.5
InternVideo [50]	8733G	1300M	41.0
MVD-H [64]	1192G	633M	41.1
VideoMAE V2 [48]	4220G	1050M	42.6
LART-MViT [66]	3780G	1640M	42.6
Hiera-H [67]	1158G	672M	43.3
UniFormerV2-B/16 [7]	406G	115M	38.4
UniFormerV2-L/14 [7]	833G	354M	40.3
FTP-UniFormerV2-B/16 (ours)	482G	136M	43.9
FTP-UniFormerV2-L/14 (ours)	970G	402M	**46.2**

Table 4: Results on AVA V2.2.

AVA V2.2. Also with an additional challenge of localizing the action, our approach shows remarkable improvement over previous methods, shown in Table 4. While our smaller FTP-UniFormerV2-B/16 already performs better than the previous best Hiera-H [67] (+0.6%), the larger FTP-UniFormerV2-L/14 achieves a 2.9% performance gain. Our performance is currently limited due to the presence of multiple targets in a sequence, because our visual encoder is trained on single targets. This complicates the alignment of the feature processor with the text descriptions generated by the VLM.

4.3 Ablation study

We have observed consistent improvements when using FTP. In this section, we investigate the role of the VLM, visual encoder, and the prompts. We also provide qualitative analyses. Additional results appear in the supplementary material B.

Effect of VLM. Various VLMs can be used to generate text descriptions for keyframe images. We experiment with text representations generated by Open-Flamingo [68], MiniGPT-4 [69], MiniGPT-v2 [70], LLaVA [13], BLIP-2 [12], and GPT-4 [11]. For GPT-4, we directly use their web API. For all other models, we employ their publicly available code and model weights. For training, we use the FTP-UniFormerV2-L/14 with 32×224^2 inputs, with the same settings as before and calculate the model accuracy on various video action datasets.

The results are shown in Table 5. At a first glance, the performance of Open-Flamingo is somewhat lower. Compared to Flamingo, an overall relative performance of 80-89% is reported [68], which shows to be insufficient to compete with the other VLMs. GPT-4 shows the best performance, likely because its more comprehensive scene descriptions, and its more extensive training.

VLM	K400		K600		SSV2		UCF-101	HMDB51	AVA V2.2
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-1	mAP
OpenFlamingo [68]	91.2	98.9	91.3	98.9	70.2	91.8	98.4	88.5	41.9
BLIP-2 [12]	92.8	99.0	91.7	98.9	77.1	95.4	99.1	90.7	43.6
MiniGPT-4 [69]	93.1	99.2	93.2	99.1	73.3	94.1	99.1	89.7	43.2
MiniGPT-v2 [70]	93.3	**99.3**	93.6	99.2	73.6	94.0	99.6	**91.5**	43.6
LLaVA [13]	**93.4**	**99.3**	93.4	99.2	79.6	98.6	99.4	90.3	44.7
GPT-4 [11]	**93.4**	**99.3**	**93.8**	**99.4**	**79.8**	**98.9**	**99.7**	**91.5**	**46.2**

Table 5: Results on FTP-UniFormerV2-L/14 for different VLMs. Best results in **bold**.

ViT	K400		K600		SSV2		AVA V2.2
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	mAP
ViT [1]	81.6	95.1	83.5	96.2	67.5	91.0	28.7
ViT + FTP	84.8 (+3.2)	95.8	85.7 (+2.2)	96.5	68.9 (+1.4)	91.4	30.0 (+1.3)
DeiT III [71]	82.7	95.5	84.0	96.5	69.6	92.7	29.0
DeiT III + FTP	85.7 (+3.0)	97.0	86.2 (+2.2)	97.3	70.6 (+1.0)	92.0	30.8 (+1.8)
VideoMAE-L [43]	85.2	96.8	86.1	97.0	75.4	95.2	37.0
VideoMAE-L + FTP	88.3 (+3.1)	97.6	89.0 (+2.9)	98.2	77.0 (+1.6)	95.9	39.5 (+2.5)
X-CLIP-L [47]	87.7	97.4	88.3	97.7	72.2	93.7	36.1
X-CLIP-L + FTP	89.6 (+1.9)	98.3	90.1 (+1.8)	98.3	73.1 (+0.9)	94.5	38.2 (+2.1)
ILA [72]	88.7	97.8	89.2	98.2	70.2	91.8	36.0
ILA + FTP	91.0 (+2.3)	98.7	90.6 (+1.4)	98.5	71.5 (+1.3)	93.5	37.9 (+1.9)
MTV-H [6]	89.1	98.2	89.6	98.3	67.6	90.1	31.2
MTV-H + FTP	91.4 (+2.3)	98.7	90.9 (+1.3)	98.5	69.1 (+1.5)	92.6	33.2 (+2.0)
UniformerV2-L [7]	89.3	98.2	89.5	98.3	73.0	94.5	40.3
UniformerV2-L + FTP	**93.4** (+4.1)	**99.3**	**93.8** (+4.3)	**99.4**	**79.8** (+6.8)	**98.9**	**46.2** (+5.9)

Table 6: Results for different ViTs with and without FTP. Best results in **bold**.

The difference between the top five VLMs on K400 is only 0.6%. We hypothesize that the similar performance is mainly due to the human-centric nature of the Kinetics videos. Object and scene information might be less relevant for the action labels. In turn, differences in the quality and richness of the VLM outputs might have had only a limited effect. We also see relatively similar performances across VLMs for human-centric datasets K600, UCF-101, and HMDB51.

For SSV2, we obtain similar performances for GPT-4 (79.8) and LLaVA (79.6%), while MiniGPT-4 (73.6%) and MiniGPT-v2 (73.3%) show poorer performance than BLIP-2 (77.1). The VLM's ability to understand relative movement is crucial to classify SSV2 videos. We conclude that some VLMs are less able to extract such patterns from the concatenated keyframe images.

For action detection results on AVA V2.2, the difference between VLMs is also significant. Compared to GPT-4 (46.2%), other VLMs show lower performance, somewhat modest for LLaVA (-1.5%) but larger for MiniGPT-v2 (-2.6%), BLIP-2 (-2.6%), MiniGPT-4 (-3.0%), and OpenFlamingo (-4.3%). We suspect that this is because the AVA V2.2 dataset involves multiple targets and the action types present in the clips are more diverse. Consequently, a more fine-grained content description is required. Differences in how comprehensive a textual output is, has a larger effect on the final classification performance.

11

Group	Prompts A	B	C	D	K400 Top-1	Top-5	K600 Top-1	Top-5	SSV2 Top-1	Top-5	UCF-101 Top-1	HMDB51 Top-1	AVA V2.2 mAP
1					87.5	97.6	89.0	98.0	66.7	91.3	97.8	80.5	39.5
2	✓				88.6	98.2	89.9	98.2	69.4	92.1	98.2	83.8	41.1
3		✓			89.0	98.4	90.5	98.8	71.2	92.8	98.6	86.0	42.2
4			✓		88.8	98.2	89.4	98.0	72.0	93.9	98.4	86.8	42.6
5				✓	88.5	98.2	90.0	98.2	71.0	92.8	98.2	85.6	42.0
6	✓	✓			91.3	98.9	91.7	99.0	72.3	93.9	99.0	87.2	42.8
7	✓		✓		90.9	98.9	91.3	99.0	73.3	94.2	99.0	88.3	43.4
8	✓			✓	90.2	98.6	90.7	98.8	71.9	93.8	98.9	86.8	42.6
9		✓	✓		92.3	99.0	92.7	99.1	75.3	95.2	99.3	89.5	44.6
10		✓		✓	91.5	98.9	91.9	99.0	74.7	94.8	99.2	89.1	44.2
11			✓	✓	91.9	98.9	92.3	99.1	75.0	95.0	99.2	89.2	44.4
13	✓	✓	✓		92.5	99.0	92.9	99.3	78.9	98.5	99.4	90.7	45.8
14	✓	✓		✓	93.2	**99.3**	93.3	99.3	77.8	97.3	99.6	89.7	44.9
15	✓		✓	✓	92.9	99.1	93.0	99.3	78.0	97.4	99.5	90.3	45.2
16		✓	✓	✓	92.6	99.1	92.9	99.3	79.1	98.6	99.4	90.8	45.8
17	✓	✓	✓	✓	**93.4**	**99.3**	**93.8**	**99.4**	**79.8**	**98.9**	**99.7**	**91.5**	**46.2**

Table 7: Results on FTP-UniFormerV2-L/14 for different combinations of prompts.

Effect of visual encoder. We apply our approach with visual encodings from different ViTs: vanilla ViT [1], DeiT III [71], VideoMAE [43], X-CLIP [47], ILA [72], and MTV-H [6]. We again use inputs of 32×224^2 and keep all other settings. In Table 6, we summarize the results for the various ViTs with and without our FTP framework.

The use of FTP consistently improves the performance across datasets and ViTs. We obtain performance gains in the range of 0.9–6.8%. The largest gains are for UniFormerV2, which already provided the best results across the tested ViTs for K400, K600, and AVA V2.2. Without FTP, VideoMAE-L produced the best results on SSV2. This could be attributed to the limited generality of the original UniformerV2-L, which was solely trained on K710. When the FTP framework is used to improve the visual encodings, the performance gain is significant at 6.8%, and makes FTP-UniFormerV2-L the top performing model.

Effect of prompt types. In the FTP framework, we use four feature processors that each attend to a specific aspect of the video contents. In training stage 1, they are aligned with text encodings of four corresponding prompts (refer to Section 3 and Figure 1). While the prompts have been chosen to cover various perspectives, we have not investigated their relative contribution until now. For these experiments with different combinations of the four prompts, we use the FTP-UniFormerV2-L/14 with 32×224^2 input and keep all other settings as before. We report the models' accuracy on various video action datasets in Table 7. We refer to the various combinations with their group number.

Group 1 only relies on the visual encoder, without additional inputs coming from the VLM. Because of architectural differences in the classification layers between our FTP-UniFormerV2-L/14 and the vanilla UniFormerV2-L/14 from [7], the performance of the two models differs.

Overall, we see a trend of increasing performance when we use more prompts. From zero to four prompts, there is a 5.9% and 13.1% increase for K400 and SSV2, respectively. We observe a similar increasing trend for other datasets. When using

a single prompt, we observe the largest increase for prompt B on K400, K600, and UCF-101, while SSV2, HMDB51 and AVA V2.2 benefit most from prompt C. This fact seems to suggest that different datasets focus on different characteristics of the action and its depiction. When two prompts are used in groups 6–11, we see superior performance when prompts B and C are used together.

With three prompts, we notice that leaving out either prompt B or C produces the best results for K400, K600, and UCF-101, but the lowest for the other tested datasets. We thus see the same division as before, again confirming our hypothesis that different datasets concentrate on different aspect of the action.

Finally, when all four prompts are used, we obtain the best results. Overall, each prompt provides complementary information, with a combination of all four prompts to consistently yield the best results. This analysis suggests that adding more prompts is generally beneficial. Future work should be aimed at understanding which aspects of the action performance or depiction we are currently not captured, and how to include the missing information. The FTP framework for combining including VLM textual outputs into the visual encodings coming from the ViT, presented in this paper, provides an effective tool to conduct such investigations.

Fig. 5: Example classifications for two different combinations of prompts, with softmax outputs. Top row is misclassified, bottom row is correctly classified.

Qualitative analysis. We show several frames of example test videos from K400 that are classified differently when using another combination of prompts. The VLM is not used during inference so we cannot analyze the textual descriptions for each prompt. Still, we can try to understand how additional prompts may have helped to fix misclassifications.

For example, the leftmost two examples in Figure 5 benefit from the additional input from the feature processor that has been aligned to encode context information. In the third example, the additional availability of the action components could reveal the movements of all people. The rightmost example could benefit from additional granularity encoded by the third feature processor. The supplementary material B contains more quantitative and qualitative analyses.

5 Conclusion and Future Work

We have introduced Four-Tiered Prompts (FTP), a framework to include semantic information into the visual encodings of Video Transformers (ViTs). During training only, we leverage a visual language model (VLM) to produce textual descriptions of a video, with four prompts that focus on different aspects of the video content. We train feature processors to align the output of the visual encoder with these text descriptions, to produce more comprehensive visual embeddings. Owing to these richer representations, we then fine-tune for action understanding in a broad range of domains, and with different requirements such as an emphasis on objects or dynamics. The feature processors incur a minimal computational overhead during inference. Moreover, VLM, text encoder, and visual encoder are used off-the-shelf, which makes the approach flexible.

Experiments on Kinetics-400/600, Something-Something V2, HMDB51, UCF-101, and AVA V2.2 consistently demonstrate state-of-the-art performance, while the computation cost is typically lower than recent methods. We have further validated the influence of the VLM, ViT, and the combination of prompts.

There are several limitations and avenues for future work. First, the additional feature processors operate on the output of the ViT's visual encoder, which typically has a limited dimensionality and potentially overlooks important aspects of the video contents. We therefore expect that integration of the textual embeddings at an earlier stage is beneficial for the performance. Second, our FTP framework employs four prompts that focus on different aspects of action performance but a more systematic analysis of the number of type of prompts could reveal which aspects are not sufficiently covered. We argue that such investigations will further improve the performance of the FTP framework. Together with its flexibility, we expect that the framework will be effective in a range of application domain, potentially also beyond action understanding.

References

[1] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)

[2] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV, pp. 10012–10022 (2021)

[3] Hu, H., Zhang, Z., Xie, Z., Lin, S.: Local relation networks for image recognition. In: CVPR, pp. 3464–3473 (2019)

[4] Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML, pp. 813–824 (2021)

[5] Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: ICCV, pp. 6836–6846 (2021)

[6] Yan, S., Xiong, X., Arnab, A., Lu, Z., Zhang, M., Sun, C., Schmid, C.: Multiview transformers for video recognition. In: CVPR, pp. 3333–3343 (2022)

[7] Li, K., Wang, Y., He, Y., Li, Y., Wang, Y., Wang, L., Qiao, Y.: Uniformerv2: Spatiotemporal learning by arming image vits with video uniformer. arXiv preprint arXiv:2211.09552 (2022)

[8] Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the Kinetics dataset. In: CVPR, pp. 6299–6308 (2017)

[9] Carreira, J., Noland, E., Hillier, C., Zisserman, A.: A short note on the Kinetics-700 human action dataset. arXiv preprint arXiv:1907.06987 (2019)

[10] Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., *et al.*: The "Something Something" video database for learning and evaluating visual common sense. In: ICCV, pp. 5842–5850 (2017)

[11] OpenAI: GPT-4 Technical Report (2023)

[12] Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597 (2023)

[13] Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. arXiv preprint arXiv:2304.08485 (2023)

[14] Ming, Y., Hu, N., Fan, C., Feng, F., Zhou, J., Yu, H.: Visuals to text: A comprehensive review on automatic image captioning. IEEE CAA J. Autom. Sin. **9**(8), 1339–1365 (2022)

[15] Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., Kalyan, A.: Learn to explain: Multimodal reasoning via thought chains for science question answering. NeurIPS **35**, 2507–2521 (2022)

[16] Wu, W., Sun, Z., Song, Y., Wang, J., Ouyang, W.: Transferring vision-language models for visual recognition: A classifier perspective. IJCV **132**, 92–409 (2024)

[17] Zhao, Y., Misra, I., Krähenbühl, P., Girdhar, R.: Learning video representations from large language models. In: CVPR, pp. 6586–6597 (2023)

[18] Himakunthala, V., Ouyang, A., Rose, D.P., He, R., Mei, A., Lu, Y., Sonar, C., Saxon, M., Wang, W.Y.: Let's think frame by frame with VIP: A video infilling and prediction dataset for evaluating video chain-of-thought. In: EMNLP (2023)

[19] Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. In: ICCV, pp. 4489–4497 (2015)

[20] Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: ECCV, pp. 305–321 (2018)

[21] Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: ICCV, pp. 6202–6211 (2019)

[22] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR, pp. 2625–2634 (2015)

[23] Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: ICCV, pp. 7083–7093 (2019)

[24] Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. NeurIPS **27** (2014)

[25] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks for action recognition in videos. IEEE TPAMI **41**(11), 2740–2755 (2018)

[26] Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: Deep networks for video classification. In: CVPR, pp. 4694–4702 (2015)

[27] Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. In: ICCV, pp. 3163–3172 (2021)

[28] Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150 (2020)

[29] Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: ICCV, pp. 6824–6835 (2021)

[30] Li, Y., Wu, C.-Y., Fan, H., Mangalam, K., Xiong, B., Malik, J., Feichtenhofer, C.: Mvitv2: Improved multiscale vision transformers for classification and detection. In: CVPR, pp. 4804–4814 (2022)

[31] Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. In: CVPR, pp. 3202–3211 (2022)

[32] Li, K., Wang, Y., Peng, G., Song, G., Liu, Y., Li, H., Qiao, Y.: Uniformer: Unified transformer for efficient spatial-temporal representation learning. In: ICLR (2022)

16

[33] Wang, R., Wu, Z., Chen, D., Chen, Y., Dai, X., Liu, M., Zhou, L., Yuan, L., Jiang, Y.-G.: Video mobile-former: Video recognition with efficient global spatial-temporal modeling. arXiv preprint arXiv:2208.12257 (2022)

[34] Lin, W., Karlinsky, L., Shvetsova, N., Possegger, H., Kozinski, M., Panda, R., Feris, R., Kuehne, H., Bischof, H.: Match, expand and improve: Unsupervised fine-tuning for zero-shot action recognition with language knowledge. arXiv preprint arXiv:2303.08914 (2023)

[35] Xu, H., Ghosh, G., Huang, P.-Y., Okhonko, D., Aghajanyan, A., Metze, F., Zettle-moyer, L., Feichtenhofer, C.: Videoclip: Contrastive pre-training for zero-shot video-text understanding. arXiv preprint arXiv:2109.14084 (2021)

[36] Li, Y., Liang, F., Zhao, L., Cui, Y., Ouyang, W., Shao, J., Yu, F., Yan, J.: Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. arXiv preprint arXiv:2110.05208 (2021)

[37] Carreira, J., Noland, E., Banki-Horvath, A., Hillier, C., Zisserman, A.: A short note about Kinetics-600. arXiv preprint arXiv:1808.01340 (2018)

[38] Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)

[39] Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: ICCV, pp. 2556–2563 (2011)

[40] Gu, C., Sun, C., Ross, D.A., Vondrick, C., Pantofaru, C., Li, Y., Vijaya-narasimhan, S., Toderici, G., Ricco, S., Sukthankar, R., et al.: AVA: A video dataset of spatio-temporally localized atomic visual actions. In: CVPR, pp. 6047–6056 (2018)

[41] Patrick, M., Campbell, D., Asano, Y., Misra, I., Metze, F., Feichtenhofer, C., Vedaldi, A., Henriques, J.F.: Keeping your eye on the ball: Trajectory attention in video transformers. NeurIPS **34**, 12493–12506 (2021)

[42] Ryoo, M., Piergiovanni, A., Arnab, A., Dehghani, M., Angelova, A.: Tokenlearner: Adaptive space-time tokenization for videos. NeurIPS **34**, 12786–12797 (2021)

[43] Tong, Z., Song, Y., Wang, J., Wang, L.: VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. NeurIPS **35**, 10078–10093 (2022)

[44] Feichtenhofer, C., Fan, H., Li, Y., He, K.: Masked autoencoders as spatiotemporal learners. NeurIPS **35**, 35946–35958 (2022)

[45] Wei, C., Fan, H., Xie, S., Wu, C.-Y., Yuille, A., Feichtenhofer, C.: Masked feature prediction for self-supervised visual pre-training. In: CVPR, pp. 14668–14678

(2022)

[46] Lin, Z., Geng, S., Zhang, R., Gao, P., Melo, G., Wang, X., Dai, J., Qiao, Y., Li, H.: Frozen clip models are efficient video learners. In: ECCV, pp. 388–404 (2022)

[47] Ni, B., Peng, H., Chen, M., Zhang, S., Meng, G., Fu, J., Xiang, S., Ling, H.: Expanding language-image pretrained models for general video recognition. In: ECCV, pp. 1–18 (2022)

[48] Wang, L., Huang, B., Zhao, Z., Tong, Z., He, Y., Wang, Y., Wang, Y., Qiao, Y.: VideoMAE V2: Scaling video masked autoencoders with dual masking. In: CVPR, pp. 14549–14560 (2023)

[49] Piergiovanni, A., Kuo, W., Angelova, A.: Rethinking video vits: Sparse video tubes for joint image and video learning. In: CVPR, pp. 2214–2224 (2023)

[50] Wang, Y., Li, K., Li, Y., He, Y., Huang, B., Zhao, Z., Zhang, H., Xu, J., Liu, Y., Wang, Z., et al.: Internvideo: General video foundation models via generative and discriminative learning. arXiv preprint arXiv:2212.03191 (2022)

[51] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., *et al.*: Learning transferable visual models from natural language supervision. In: ICML, pp. 8748–8763 (2021). PMLR

[52] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV, pp. 20–36 (2016)

[53] Tan, H., Lei, J., Wolf, T., Bansal, M.: Vimpac: Video pre-training via masked token prediction and contrastive learning. arXiv preprint arXiv:2106.11250 (2021)

[54] Wang, L., Tong, Z., Ji, B., Wu, G.: Tdn: Temporal difference networks for efficient action recognition. In: CVPR, pp. 1895–1904 (2021)

[55] Wang, R., Chen, D., Wu, Z., Chen, Y., Dai, X., Liu, M., Jiang, Y.-G., Zhou, L., Yuan, L.: Bevt: Bert pretraining of video transformers. In: CVPR, pp. 14733–14743 (2022)

[56] Pan, T., Song, Y., Yang, T., Jiang, W., Liu, W.: Videomoco: Contrastive video representation learning with temporally adversarial examples. In: CVPR, pp. 11205–11214 (2021)

[57] Han, T., Xie, W., Zisserman, A.: Memory-augmented dense predictive coding for video representation learning. In: ECCV, pp. 312–329 (2020)

[58] Diba, A., Sharma, V., Safdari, R., Lotfi, D., Sarfraz, S., Stiefelhagen, R., Van Gool, L.: Vi2CLR: Video and image for visual contrastive learning of

representation. In: CVPR, pp. 1502–1512 (2021)

[59] Hu, K., Shao, J., Liu, Y., Raj, B., Savvides, M., Shen, Z.: Contrast and order representations for video self-supervised learning. In: ICCV, pp. 7939–7949 (2021)

[60] Alwassel, H., Mahajan, D., Korbar, B., Torresani, L., Ghanem, B., Tran, D.: Self-supervised learning by cross-modal audio-video clustering. NeurIPS **33**, 9758–9770 (2020)

[61] Qian, R., Meng, T., Gong, B., Yang, M.-H., Wang, H., Belongie, S., Cui, Y.: Spatiotemporal contrastive video representation learning. In: CVPR, pp. 6964–6974 (2021)

[62] Patrick, M., Asano, Y.M., Fong, R., Henriques, J.F., Zweig, G., Vedaldi, A.: Multi-modal self-supervision from generalized data transformations. arXiv preprint arXiv:2003.04298 (2020)

[63] Alayrac, J.-B., Recasens, A., Schneider, R., Arandjelović, R., Ramapuram, J., De Fauw, J., Smaira, L., Dieleman, S., Zisserman, A.: Self-supervised multimodal versatile networks. NeurIPS **33**, 25–37 (2020)

[64] Wang, R., Chen, D., Wu, Z., Chen, Y., Dai, X., Liu, M., Yuan, L., Jiang, Y.-G.: Masked video distillation: Rethinking masked feature modeling for self-supervised video representation learning. In: CVPR, pp. 6312–6322 (2023)

[65] Faure, G.J., Chen, M.-H., Lai, S.-H.: Holistic interaction transformer network for action detection. In: WACV, pp. 3340–3350 (2023)

[66] Rajasegaran, J., Pavlakos, G., Kanazawa, A., Feichtenhofer, C., Malik, J.: On the benefits of 3d pose and tracking for human action recognition. In: CVPR, pp. 640–649 (2023)

[67] Ryali, C., Hu, Y.-T., Bolya, D., Wei, C., Fan, H., Huang, P.-Y., Aggarwal, V., Chowdhury, A., Poursaeed, O., Hoffman, J., et al.: Hiera: A hierarchical vision transformer without the bells-and-whistles. arXiv preprint arXiv:2306.00989 (2023)

[68] Awadalla, A., Gao, I., Gardner, J., Hessel, J., Hanafy, Y., Zhu, W., Marathe, K., Bitton, Y., Gadre, S., Sagawa, S., et al.: OpenFlamingo: An open-source framework for training large autoregressive vision-language models. arXiv preprint arXiv:2308.01390 (2023)

[69] Zhu, D., Chen, J., Shen, X., Li, X., Elhoseiny, M.: MiniGPT-4: Enhancing vision-language understanding with advanced large language models. arXiv preprint arXiv:2304.10592 (2023)

[70] Chen, J., Zhu, D., Shen, X., Li, X., Liu, Z., Zhang, P., Krishnamoorthi, R., Chandra, V., Xiong, Y., Elhoseiny, M.: MiniGPT-V2: Large language model as a unified interface for vision-language multi-task learning. arXiv preprint arXiv:2310.09478 (2023)

[71] Touvron, H., Cord, M., Jégou, H.: Deit III: Revenge of the ViT. In: ECCV, pp. 516–533 (2022)

[72] Tu, S., Dai, Q., Wu, Z., Cheng, Z.-Q., Hu, H., Jiang, Y.-G.: Implicit temporal modeling with learnable alignment for video recognition. arXiv preprint arXiv:2304.10465 (2023)

[73] Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)

Stage	FTP-UniFormerV2-L
Visual encoder	[3,4,8] Uni.V2 Blocks
Feature processor	torch.transpose(features, 1, -1) torch.reshape(-1, depth) maxpool2d(features, kernelsize=1, stride=1) torch.nn.Conv1d(inchannels, outchannels, kernelsize) torch.cat((features1, features2), dim=1)
Integration	features.repeat(1, 1, 1, num copies) DepthwiseSeparableConv2d((T+N), D, kernelsize=(T,N), padding=0) torch.add(features1, features2) nn.Conv2d(kernelsize=3, padding=1, stride=1)
Successive layers and classifier	2 × [MHSA+nn.Conv2d(kernel size=3, padding=0, stride=1)] torch.nn.AvgPool2d(kernelsize=2, stride=2) torch.nn.Linear(input, output) Softmax

Table 1: **Architecture of FTP-UniFormerV2-L**.

Appendix A   Implementation

A.1   Model architecture

We show the architecture details of FTP-UniFormerV2-L/14 in Table 1. We use the UniformerV2 [7] as the video encoder for better and more efficient temporal modeling, and insert the global UniBlocks in the last four layers to perform the multi-stage fusion. For the feature processor, we first reshape the features, and then obtain the spatio-temporal feature through temporal pooling, spatial pooling, and subsequent concatenation. To realize the integration of visual encoder and feature processor, we first replicate the spatio-temporal feature, then use the feature processor to adjust the shape. Features are then summed up element-wise and then passed to the CNN layers to generate the final classification.

Parameter	Value
optimizer	AdamW
base learning rate	1.e-5
weight decay	0.05
optimizer momentum	$\beta1, \beta2 = 0.9, 0.95$
batch size	512
learning rate schedule	cosine decay
warmup epoch	5
epoch	800
repeated augmentation	1
flip augmentation	no
augmentation	MultiScaleCrop

Table A2: **Parameter settings for training stage 1**: video-text alignment process.

A.2 Training details

Training stage 1: Video-text alignment. We prompt GPT-4 to generate textual descriptions when provided with a concatenated keyframe image and a specific prompt. The VLM and text encoder remain frozen, we only fine-tune the parameters of the feature processors. We implement the training process of video-text alignment with 40 epochs on the K710 dataset with 16 NVIDIA 80G-A100 GPUs. We adopt repeated augmentation to reduce the video loading overhead. The parameter settings are summarized in Table A2.

Parameter	Value
optimizer	AdamW
base learning rate	1.e-5
weight decay	0.05
optimizer momentum	$\beta1, \beta2 = 0.9, 0.999$
batch size	128
learning rate schedule	cosine decay
warmup epoch	5
epoch	40
α	0.7
repeated augmentation	2
label smoothing	0.1
mixup	0.8
flip augmentation	yes
augmentation	MultiScaleCrop

Table A3: **Parameter settings for training stage 2**: model fine-tuning.

Training stage 2: Model fine-tuning We fine-tune the model trained in stage 1 to adjust the weights so that the model can fit in the integrated features from feature processor and visual encoder. The parameters of the fine-tuning process are shown in Table A3.

Appendix B Additional Results on Kinetics

We first report the qualitative results on Kinetics-600, and present additional insights into the improvements on Kinetics-400 when using our FTP framework.

B.1 Results on Kinetics-600

We report comparisons on Kinetics-600 with several FTP-UniformerV2 models in Table B4. Compared to other models including UniFormerV2, VideoMAE V2, our approach shows consistent improvement. For example, compared to UniFormerV2-B/16, our approach brings 4.8% improvement. Moreover, when increasing the input size, our model further improves the recognition results. For example, when increasing the spatial input size from 224^2 to 336^2, our FTP-UniFormerV2-L/14 shows 0.6% improvement.

Methods	Input	Crops	TFLOPs	Top-1	Top-5
SlowFast R101-NL [21]	64×224^2	10×3	7.02	81.8	95.1
TimeSformer-L [4]	96×224^2	10×3	7.14	82.2	95.6
ViViT-L FE [5]	32×224^2	1×3	11.94	82.9	94.6
MTV-B [6]	32×320^2	4×3	4.79	83.6	96.1
MViT-B [29]	32×224^2	3×3	4.10	83.8	96.3
MViTv2-B [30]	32×312^2	5×3	1.03	85.5	97.2
MaskFeat [45]	64×224^2	4×3	3.77	86.4	97.4
MViTv2-L [30]	40×352^2	5×3	45.48	87.9	97.9
MaskFeat [45]	40×312^2	4×3	33.94	88.3	98.0
VideoMAE V2-H [48]	16×224^2	2×3	17.88	88.3	98.1
VideoMAE V2-g [48]	64×266^2	2×3	38.16	88.8	98.2
UniFormerV2-B/16	8×224^2	4×3	1.8	87.4	97.9
UniFormerV2-L/14	32×224^2	2×3	16.0	89.5	98.3
FTP-UniFormerV2-B/16 (Ours)	8×224^2	4×3	2.2	92.2	98.9
FTP-UniFormerV2-B/16 (Ours)	8×280^2	4×3	8.1	93.0	99.0
FTP-UniFormerV2-L/14 (Ours)	32×224^2	2×3	18.4	93.8	99.4
FTP-UniFormerV2-L/14 (Ours)	32×336^2	2×3	78.7	**94.4**	**99.3**

Table B4: **Performance on K600**. Best results in **bold**. We report crops (temporal × spatial) and TFLOPs for inference.

B.2 Class accuracy improvement on Kinetics-400

We now investigate the relative class performance when using our FTP-UniForm-er V2-L/14 model, compared to a UniFormerV2-L/14 baseline. We show the relative performance for all classes of Kinetics-400 in Figure B1. For over 85% of the classes, our method shows improvement. Although there is a lower performance for certain classes, the decrease is typically limited. We further analyze these categories in the following section and investigate the possible reasons for the performance decline.

Top-10 classes with highest improvement. We show the top-10 classes from Kinetics-400 that have seen the most significant improvement when using our FTP framework in Figure B2. These classes correspond to the leftmost 10 bars in Figure B1.

These 10 classes are all challenging and require the integration of action objects, context, and other information. For instance, the "auctioning" class typically involves identifying the context of objects and actions within the video, and our method yields a +9.3% improvement compared to UniformerV2.

Top-10 classes with biggest performance decline. Application of our method performs worse than a baseline Uniformerv2 model on a minority of the classes of Kinetics-400. We analyze the bottom-10 classes in terms of relative improvement in Figure B3. We see several action in which the scene or specific recording setting might be introducing distracting elements into the textual representations. For example, long jump and triple jump take place in stadiums with many sports occurring simultaneously. Parkour, capoeira, and breakdancing can be performed in scenes that vary significantly.

Because the VLM is presented only with several key frames, motion information is not emphasized. As a result, the motion information that is initially present in the
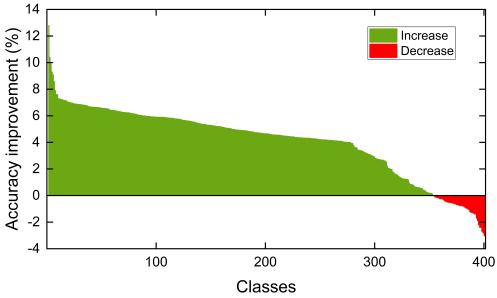
Fig. B1: **Relative accuracy per class on Kinetics-400** by comparing FTP-UniFormerV2-L/14 to a baseline UniFormerV2-L/14 model. The classes are sorted by relative performance.

visual encoder might be diluted in the presence of the textual encodings. This might have deteriorated the performance for classes such as salsa dancing and gymnastics tumbling.

Visualization of embeddings. To further analyze the effect of our prompts, we use t-SNE [73] and visualize the distribution of features from different classes. We use the 2048D feature activations of the first layer of the classifier.

In Figure B4, we randomly select 100 samples from the "riding a bike" and "biking through snow" category. The distribution of features from each class is much more compact and discriminative after applying our FTP approach.

In Figure B5, we randomly select 5 samples of each of the 400 categories of Kinetics-400. After applying our FTP approach, we observe that the intra-class variation of features becomes smaller, while the inter-class variation increases. This indicates that the classes are better isolated, consequently benefitting classification.

Appendix C   Effect of Number of Keyframes K to VLM

In our experiments in the main paper, we used $K = 5$ as default to prompt the VLM. To evaluate the effect of using less or more keyframes, we conducted experiments using FTP-UniFormerV2-L/14 with different values of K. We kept the experiment settings consistent with those used in the previous experiments on Kinetics-400 and summarize the results in Table C5.
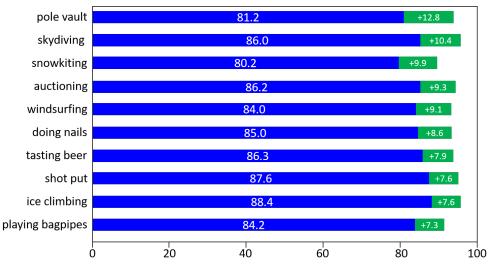
Fig. B2: **Top-10 Kinetics-400 classes with most improvement** when using the FTP framework.
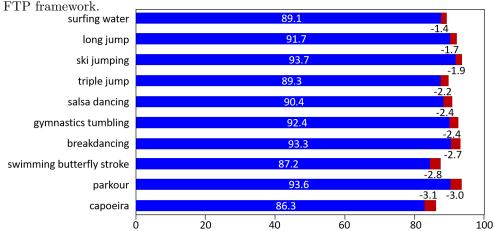

Fig. B3: **Top 10 Kinetics-400 classes with least improvement** when using the FTP framework. These classes have a lower score when using our FTP framework.

We can observe that when $K = 1$, the performance is the worst, with a top-1 accuracy of 91.9%. As K increases to 3, the top-1 accuracy improves to 92.5%. The best performance of 93.4% is achieved for $K = 5$. However, as we continue to increase K, the performance starts to decline, reaching 93.2% for $K = 7$. This could be because, at this point, the image descriptions generated by GPT become overly complex, and potentially focus on less consistent aspects of the video. Consequently, this might introduce some noise in the text encodings that affects the performance.
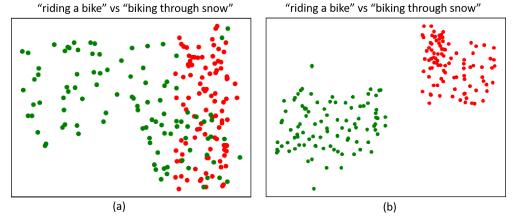
Fig. B4: **t-SNE visualization of feature distribution** of "riding a bike" vs "biking through snow" (a) without applying the FTP framework, and (b) when applying FTP.
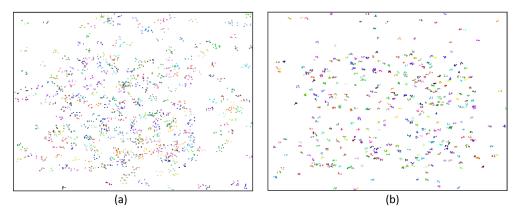

Fig. B5: **t-SNE visualization of feature distribution of all Kinetics-400 classes** (a) without applying the FTP framework, and (b) when applying FTP.

	$K = 1$	$K = 3$	$K = 5$	$K = 7$
Top-1 accuracy	91.9	92.5	**93.4**	93.2

Table C5: **Effect of number of concatenated keyframes** K on the performance on Kinetics-400 (in %).