Leaping Into Memories: Space-Time Deep Feature Synthesis

Alexandros Stergiou Nikos Deligiannis
Vrije Universiteit Brussel, Belgium & imec, Belgium
<first>.<last>@vub.be

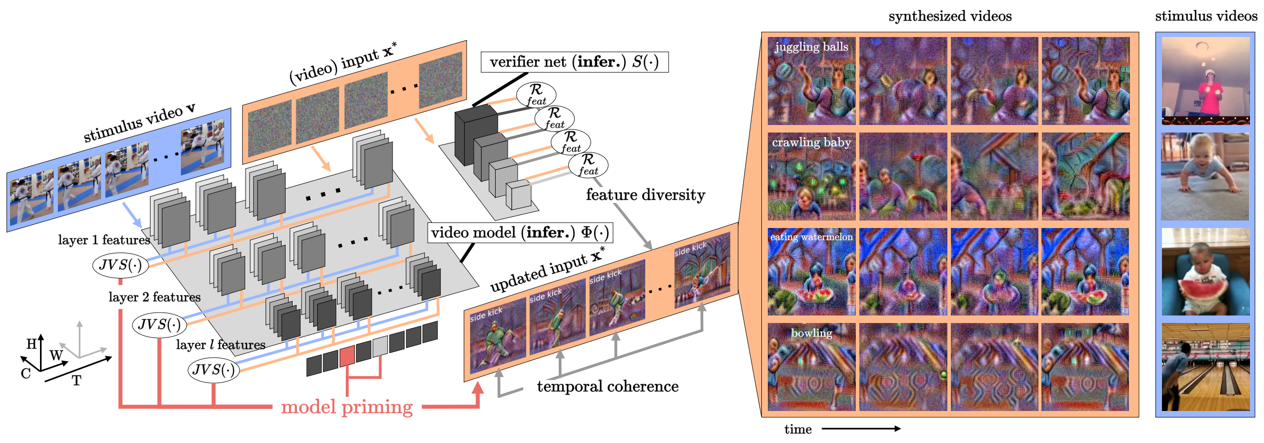


Figure 1: **Illustration of our proposed LEAPS model inversion**. Given a video input $\mathbf{x}^*$ initialized with noise, we use a stimulus video $\mathbf{v}$ of target class y, to prime video model $\Phi(\cdot)$. We iteratively optimize $\mathbf{x}^*$ to synthesize and visually represent the internal spatiotemporal features of $\Phi(\cdot)$. To impose diversity over the synthesized videos, feature statistics from a domain-specific verifier network $S(\cdot)$ are distilled to regularize $\mathbf{x}^*$. To preserve the continuity of motions across frames, $\mathbf{x}^*$ is temporally regularized at each training iteration. Although the illustration visualizes internal representations as $C\times T\times H\times W$ volumes, they can also be flattened patches $CP^3 \times \frac{THW}{P^3}$ of P^3 resolution as in Vision Transformers.

Abstract

*The success of deep learning models has led to their adaptation and adoption by prominent video understanding methods. The majority of these approaches encode features in a joint space-time modality for which the inner workings and learned representations are difficult to visually interpret. We propose **LEA**rned **P**reconscious **S**ynthesis (LEAPS), an architecture-independent method for synthesizing videos from the internal spatiotemporal representations of models. Using a stimulus video and a target class, we prime a fixed space-time model and iteratively optimize a video initialized with random noise. Additional regularizers are used to improve the feature diversity of the synthesized videos alongside the cross-frame temporal coherence of motions. We quantitatively and qualitatively evaluate the applicability of LEAPS by inverting a range of spatiotemporal convolutional and attention-based architectures trained on Kinetics-400, which to the best of our knowledge has not been previously accomplished.* [1]

1. Introduction

Inverting deep networks' learned internal representations has been a difficult task to achieve. The adaptation and deployment of CNNs and more recently Transformers, to a variety of vision tasks has led to significant breakthroughs. The field of video action recognition has experienced drastic growth in recent years through the convergence of models with increased complexities and capacities [2, 4, 32, 63] as well as large accuracy improvements [11, 12, 27, 26, 29]. Despite great progress in their applicability, there is still a large gap in the interpretability of video models. As these models compositionally encode space and time modalities of videos over large feature spaces and require a lot of parameters, conceptualizing their internal representations remains challenging. In this paper, we propose a method to invert learned features of video models associated with specific actions by optimizing a parameterized input to synthesize conceptually and visually coherent representations.

In cognitive science, stages of consciousness include the conscious, the unconscious, and the preconscious. In contrast to the conscious and unconscious, the preconscious is

[1] See alexandrosstergiou/LEAPS for video examples and code.

responsible for learned information currently outside conscious awareness to remain readily available [18]. One way of accessing learned preconscious information and making it part of conscious awareness is through priming [38]. Priming uses a stimulus to activate related learned concepts in memory and make them easily and readily accessible [33, 34]; e.g., one can remember their bedroom if primed with a picture of a bed.

Motivated by visual priming in cognitive science, we demonstrate that learned representations of video models can become accessible through *model priming*. By using a video stimulus and a target action class, we synthesize the dominant learned concepts corresponding to actions. In turn, the visual features of the synthesized videos provide a conceptual view of the models' learned internal representations. We term these features as the *learned preconscious* of the video model associated with a specific action.

We introduce **LEA**rned **P**reconscious **S**ynthesis (LEAPS), illustrated in Figure 1, a spatiotemporal model inversion method that synthesizes interpretable videos by minimizing a classification and priming loss without prior knowledge of the training data. LEAPS uses a video of a target action class as stimulus, to prime a fixed spatiotemporal model. We additionally include two regularization terms. The first term enforces motion coherence across frames by constraining their representations at each update. The second enhances the diversity of the synthesized videos by using a domain-specific verifier network that exploits disagreements between feature statistics similar to [64]. Through the same architecture-independent approach, we show that LEAPS can invert the spatiotemporal features of 3D-CNNs and spatiotemporal Transformers.

Our main contributions are as follows. First, we introduce LEAPS, a general approach for inverting video models, which to the best of our knowledge, is the first attempt to create videos of conceptual representations from jointly encoded space-time features. Second, we use LEAPS on multiple convolution and attention models and compare it to prior image-based inversion methods that we extend to video. Finally, we show that LEAPS can invert both video CNNs and transformers, with the same architecture-independent method without any modifications.

2. Related Work

Approaches for visualizing and interpreting deep models can be divided into three groups which we detail below.

Attribution-based visualizations. These methods have been used to visualize feature contributions over an input. Attribution methods for images have primarily been based on back-propagating activations of classes [8, 48, 61], or individual neurons [3, 5, 49, 52] to localize regions in the input that are informative for a given class or feature. Such approaches include Integrated gradients (IG) [56] which produce pixel-wise attributions from integrating computed backprop gradients. Subsequent works have also included gradient smoothing [51], gradient accumulation in saturated regions [35], and adaptation of the gradient path [23]. Another set of approaches that rely on attribution-based visualizations use perturbations of the input [17, 16, 44]. Given their straightforward applicability, image attribution methods have also been extended to video. The majority of works; e.g., Saliency Tubes [55, 54], STEP [28], video OSA [59], and BOREx [24], have focused on the localization of spatiotemporal salient regions by extending Grad-CAM [48], Extremal Perturbations (EP) [16], Occlusion Sensitivity Analysis (OSA) [66], and Gaussian processes regression (GPR) [6, 37] respectively. In contrast to these approaches that localize salient class features, we offer a visual feature synthesis method to conceptualize the learned representations of video models through priming.

Input synthesis. Network-centric approaches invert models to either visualize particular classes [41, 65, 64] or features [43, 53]. One of the main methods employed for visualizing internal network features is Gradient Ascent (GA) [10], which optimizes the input by increasing the activation of a specific neuron. Activation Maximization (AM) [50] later adapted GA to visualize CNN features. Following works have been built on top of AM by including additional regularizers; e.g., total variation [31], blurring [60], and gradient masking [41]. One of the most popular extensions of AM has been DeepDream [1] which optimizes the input to yield high responses for a chosen class while keeping internal representations constraint-free. The produced images include repetitions of recognizable concepts without representing a coherent whole. In addition to AM, model inversion [9, 19, 30, 64, 66] includes the task of maximizing the classification score of the synthesized image instead of maximizing class activations. The only extension of input synthesis approaches to videos has been introduced by Feichtenhofer *et al*. [13] in which AM is used to create visual representations of class features from two-steam models [14, 62], trained on RGB frames (spatial stream) and optical flow (temporal stream). In this paper, we instead propose a model inversion method for inverting video models concurrently encoding space and time modalities.

Visual feature generation. These methods utilize activation maximization by including an additional generator network [39, 40, 42], with the cost of requiring access to the training data. Huang *et al*. [21] adapted feature generation on video data by modeling the temporal signal with a temporal generator network. Despite the high fidelity of the produced results, these approaches are less suitable for interpreting internal model behaviors, as they do not solely depend on the model under inspection. Instead, they are primarily influenced by the training data used as well as the capacity and complexity of the generator model trained.

3. Learned Preconscious Synthesis

In this section, we overview our LEAPS method, shown in Figure 1. We start by formally introducing model priming for video models in Section 3.1. A stimulus video of a target class is used to initially prime the network. The training process uses the primed representations to update a randomly initialized input alongside two regularizers. The first regularizer is used to enforce temporal coherence, explained in Section 3.2. The second is used to improve synthesized feature diversity, overviewed in Section 3.3. We present the final aggregated function in Section 3.4.

3.1. Model Priming

Priming deep models is influenced by cognitive science [38] in which a stimulus is used to recall prior knowledge. We extend this approach to visualizing the learned preconscious of deep video models associated with a specific action class y. Given a video $\mathbf{v}$ of size $C \times T \times H \times W$, with C channels, T frames, H height, and W width, as a visual cue for action class y, and a randomly initialized input $\mathbf{x}^*$ of $C \times T \times H \times W$ size to optimize. We define a priming loss $\underset{prim}{\mathcal{L}}(\mathbf{x}^*, \mathbf{v})$ between the internal representations $\mathbf{z}^l(\cdot)$ of the optimized input $\mathbf{x}^*$ and stimulus $\mathbf{v}$ across $l \in \mathbf{\Lambda} = \{1, ..., L\}$ layers:

$$\underset{prim}{\mathcal{L}}(\mathbf{x}^*, \mathbf{v}) = \frac{1}{L} \sum_{l \in \mathbf{\Lambda}} \lambda_l \; JVS\left(\mu(\mathbf{z}^l(\mathbf{x}^*)), \mu(\mathbf{z}^l(\mathbf{v}))\right) \quad (1)$$

where $\mu\big(\mathbf{z}^l(\cdot)\big)$ is the C-length spatiotemporal mean vector of representations $\mathbf{z}^l(\cdot)$. $JVS(\cdot)$ denotes the Jaccard vector similarity [15]. To integrate a degree of freedom and avoid hard constraints on the internal representations of $\mathbf{x}^*$, we define $0 < \lambda_l \leq 1$ as priming weight for layer l. An overview of updating $\mathbf{x}^*$ by priming appears in Figure 1.

Due to the vastness of the feature space when optimizing (1), we include two additional regularization terms to constrain the input. Specifically, we apply a temporal coherence regularization $\underset{coh}{\mathcal{R}}$ and a feature diversity regularization $\underset{feat}{\mathcal{R}}$, which we detail below.

3.2. Temporal Coherence Regularization

For the first regularizer, we aim to enforce similarity between representations of consecutive frames in order to enable consistent feature transitions in the synthesized video. Therefore, we include a coherence regularizer $\underset{coh}{\mathcal{R}}$, formulated based on the temporal coherence loss from [36].
Given two spatiotemporal representations $\mathbf{z}^L(\mathbf{x}^*)_{t_1}$ and $\mathbf{z}^L(\mathbf{x}^*)_{t_2}$ at layer L, for temporal locations t_1 and t_2, we use their l_1 norm to enforce similarity if t_1 and t_2 are consecutive in the video. In non-consecutive cases, the divergence between $\mathbf{z}^L(\mathbf{x}^*)_{t_1}$ and $\mathbf{z}^L(\mathbf{x}^*)_{t_2}$ should increase. The

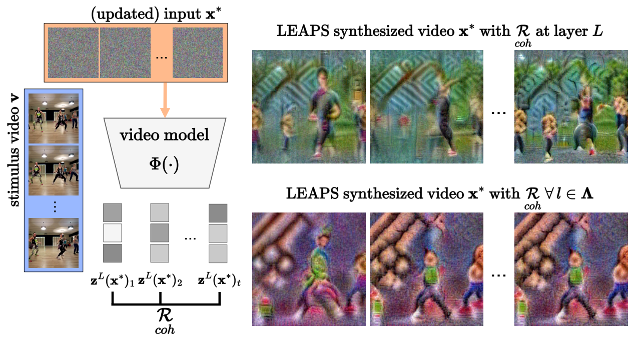

(updated) input $\mathbf{x}^*$
stimulus video $\mathbf{v}$
video model
$\Phi(\cdot)$
$\mathbf{z}^L(\mathbf{x}^*)_1$ $\mathbf{z}^L(\mathbf{x}^*)_2$ $\mathbf{z}^L(\mathbf{x}^*)_t$
$\underset{coh}{\mathcal{R}}$
LEAPS synthesized video $\mathbf{x}^*$ with $\underset{coh}{\mathcal{R}}$ at layer L
LEAPS synthesized video $\mathbf{x}^*$ with $\underset{coh}{\mathcal{R}}\ \forall l \in \mathbf{\Lambda}$

Figure 2: **Temporal Coherence regularization** of *zumba* target action label. Applying temporal coherence regularization on representations from all layers $l \in \mathbf{\Lambda}$ synthesizes static videos (bottom row). Instead, regularizing only the final network layer L synthesizes videos with consistent frame transitions and motions (top row). We note that the video stimulus is shown as a reference as the regularizations are only applied to synthesized video representations.

coherence regularizer is formulated as:

$$\underset{coh}{\mathcal{R}}(\mathbf{x}^*) = \begin{cases} ||\mathbf{z}^L(\mathbf{x}^*)_{t_1} - \mathbf{z}^L(\mathbf{x}^*)_{t_2}||_1, \text{if consecutive} \\ \max\big(0, \delta - ||\mathbf{z}^L(\mathbf{x}^*)_{t_1} - \mathbf{z}^L(\mathbf{x}^*)_{t_2}||_1\big), \text{elsewise} \end{cases} \quad (2)$$

where δ is a margin hyperparameter. Temporal coherence is enforced at layer L as in [36]. Although $\underset{coh}{\mathcal{R}}$ can be applied to any layer $l \in \mathbf{\Lambda}$, in practice, minimizing (2) for all layers enforces a very strong regularization, producing synthesized videos with minimal to no cross-frame variations as shown in Figure 2. We note that for Transformers using patches of P^3 resolution, we first reshape $\mathbf{z}^L(\mathbf{x}^*)$ from $C'P^3 \times \frac{T'H'W'}{P^3}$ to $C' \times T' \times H'W'$ before calculating (2).

3.3. Feature Diversity Regularization

Our second regularization term $\underset{feat}{\mathcal{R}}$ is responsible for improving the diversity of features generated by model priming. Although priming provides a strong signal based on which the input can be updated, the diversity of features is limited compared to observing multiple instances. Thus, class features varying from those in the stimulus, or features not present in the stimulus, may not be explored during optimization. In order to enhance the search space we introduce an additional domain-specific verifier network $\mathcal{S}(\cdot)$. Our goal is to use high- and low-level feature distribution statistics as proposed in [64] incorporating the verifier's prior knowledge during optimization.
Based on input $\mathbf{x}^*$, we run inference on the verifier $\mathcal{S}(\mathbf{x}^*)$ to obtain representations $\mathbf{a}^k(\mathbf{x}^*)$ across each verifier layer $k \in \mathbf{K}$. The feature statistics are then obtained by the C-length space-time mean $\mu\left(\mathbf{a}^k(\mathbf{x}^*)\right)$ and variance $\sigma^2\left(\mathbf{a}^k(\mathbf{x}^*)\right)$ vec-

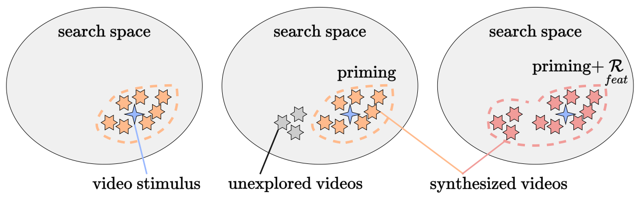


Figure 3: **Feature diversity regularization**. Given the video stimulus in blue, synthesized videos in orange are optimized only from the stimulus. Synthesized videos with feature diversity regularization are shown in red. The regularized videos can also include general class features different from or not existing in the stimulus.

tors. The feature diversity regularizer is defined as:

$$\begin{aligned}\underset{feat}{\mathcal{R}}(\mathbf{x}^*) = \sum_{k \in \mathbf{K}} ||\mu\left(\mathbf{a}^k(\mathbf{x}^*)\right) - \mathbb{E}\big(\mu\left(\mathbf{a}^k(\mathbf{x})\right)|\mathcal{X}\big)||_2 + \\ \sum_{k \in \mathbf{K}} ||\sigma^2\left(\mathbf{a}^k(\mathbf{x}^*)\right) - \mathbb{E}\big(\sigma^2\left(\mathbf{a}^k(\mathbf{x})\right)|\mathcal{X}\big)||_2\end{aligned} \tag{3}$$

where $\mathbb{E}(\cdot)$ corresponds to the expected value of representation $\mathbf{a}(\cdot)$ for video input $\mathbf{x}$ part of video dataset $\mathcal{X}$. Instead of requiring access to the dataset to train over $\mathbf{x} \in \mathcal{X}$ videos, we use the Batch Normalization [22] running mean and variance to approximate the expected mean and variance as in [64]. The mean and variance estimates are then reformulated as $\mathbb{E}\big(\mu\left(\mathbf{a}^k(\mathbf{x})\right)|\mathcal{X}\big) \simeq BN^k(\text{running\_mean})$ and $\mathbb{E}\big(\sigma^2\left(\mathbf{a}^k(\mathbf{x})\right)|\mathcal{X}\big) \simeq BN^k(\text{running\_variance})$. An illustration of the improved search space achieved with the inclusion of feature diversity is shown in Figure 3.

3.4. Aggregation for Model Inversion

We have introduced model priming as a method to obtain a strong signal from a stimulus video with which noise-initialized videos can be optimized. To implicitly enforce transition continuity across frames, we include a temporal coherence regularizer. In addition, as the stimulus does not provide prior intuition about the diversity of features, we use a feature diversity regularizer to enhance the search space. Given their complementary properties, we formulate the final LEAPS objective as the combination of model priming, temporal coherence, and feature diversity regularizers. In line with spatial feature visualization losses that synthesize distinguishable features from noise [1, 36, 47, 64], we include a cross-entropy loss $\underset{CE}{\mathcal{L}}$ from class predictions given the synthesized input $\mathbf{x}^*$:

$$\mathcal{L}(\mathbf{x}^*, \mathbf{v}, y) = \underset{CE}{\mathcal{L}}(\mathbf{x}^*, y) + \underset{prim}{\mathcal{L}}(\mathbf{x}^*, \mathbf{v}) + r\mathcal{R}(\mathbf{x}^*) \tag{4}$$

where the regularizer term combines (2) and (3): $\mathcal{R}(\mathbf{x}^*) = \underset{coh}{\mathcal{R}} + \underset{feat}{\mathcal{R}}$. Respectively, r is a regularizer scaling factor. The final LEAPS objective enables the synthesis of high-fidelity features without being constrained by data availability or architecture types as shown in the results section.

4. Main Results

We first detail the train scheme and implementation settings used in Section 4.1. We then compare our proposed LEAPS method to image-based feature visualization methods that we extend to video and use as baselines in Section 4.2. We also qualitatively and quantitatively investigate the synthesized videos across a variety of video models in Section 4.3. Finally, in Section 4.4 we perform ablation studies for the LEAPS objective as well as different updatable input spatiotemporal resolutions.

4.1. Experimental details

Model details. We invert 3D [20]/(2+1)D [58]/CSN [57] ResNet-50 (R50), X3D [11], TimeSformer [4], Video Swin [29], MViTv2 [27], rev-MViT [32], and UniFormerv2 [26] networks. For all our experiments, we use the official networks available from their respective repositories pretrained on Kinetics-400 [7]. Due to its limited computational overhead and requirement of BN layers by the feature diversity regularizer in (3), we use X3D$_S$ as the verifier network $\mathcal{S}(\cdot)$. We note that only the internal representations and predictions from the inverted and verifier models are used. Both models only run inference and remain fixed throughout the feature synthesis.
Feature synthesis optimization details. Video input $\mathbf{x}^*$ is initialized with size of 8×224^2. For models [11, 4, 29, 27, 32, 26] which use inputs of fixed size, we first interpolate $\mathbf{x}^*$ to match the required size[2]. Priming stimuli are selected from the Kinetics-400 validation set randomly. For transformer models, $\mathbf{x}^*$ and $\mathbf{v}$ are first tokenized. We use Adam [25] with a learning rate of 0.2 and a cosine decrease policy as in [64]. We use a total of 2K gradient updates. As in [36], we set $\delta = 1$. To discover the optimal λ and r hyperparameters for each network we use Mango [46] to perform a simple grid search for 1K gradient updates. A full overview of the hyperparameters used by each model is available in Table S1 in the supplementary material.

4.2. Baseline results

We first assess the quality of the synthesized videos. We invert 3D R50, X3D$_M$, and Video Swin-B models and report the averaged top-1 classification accuracies and Inception Scores (IS) [45] on synthesized videos using each video from the validation set of Kinetics-400 as a stimulus. We extend two prominent image-based feature visualization methods making them applicable to spatiotemporal models:

[2]Due to space limitations in Figures 5 to 8 the last 2 frames of $\mathbf{x}^*$ are not shown.

Visualization method	top-1 (%) model	top-1 (%) ver.	Inception Score (IS) model	Inception Score (IS) verifier
3D R50				
3D Deep Dream [1]	34.5	2.6	1.1 ± 0.1	1.0
3D AM [13]	41.4	5.8	1.4 ± 0.3	1.2 ± 0.2
LEAPS **ours** ($\mathcal{L}_{prim}$)	67.9	53.1	3.9 ± 0.9	3.2 ± 0.6
LEAPS **ours** ($\mathcal{L}_{prim} + \mathcal{R}_{feat}$)	74.3	60.2	5.1 ± 0.8	3.9 ± 1.4
LEAPS **ours (full)**	<u>86.7</u>	<u>68.5</u>	<u>9.0 ± 1.0</u>	<u>5.7 ± 0.7</u>
X3D$_M$				
3D Deep Dream [1]	18.1	1.9	1.1 ± 0.1	1.1 ± 0.1
3D AM [13]	33.2	5.6	1.3 ± 0.3	1.2 ± 0.2
LEAPS **ours** ($\mathcal{L}_{prim}$)	73.4	59.8	5.1 ± 1.1	3.9 ± 0.4
LEAPS **ours** ($\mathcal{L}_{prim} + \mathcal{R}_{feat}$)	81.1	69.3	8.8 ± 1.5	6.3 ± 0.8
LEAPS **ours (full)**	<u>**90.3**</u>	<u>**82.5**</u>	<u>**11.4** ± **0.9**</u>	<u>**8.0** ± **1.4**</u>
Video Swin-B				
3D Deep Dream [1]	15.9	1.4	1.4 ± 0.2	1.1 ± 0.1
3D AM [13]	25.6	2.2	1.6 ± 0.4	1.1 ± 0.1
LEAPS **ours** ($\mathcal{L}_{prim}$)	71.2	58.6	4.4 ± 0.8	3.5 ± 0.6
LEAPS **ours** ($\mathcal{L}_{prim} + \mathcal{R}_{feat}$)	76.0	65.4	5.3 ± 1.5	4.1 ± 1.1
LEAPS **ours (full)**	<u>87.4</u>	<u>74.3</u>	<u>9.8 ± 1.3</u>	<u>6.5 ± 0.9</u>

Table 1: **Quantitative results for mean top-1 accuracy and Inception Score (IS)**. The best results per metric are in **bold** and per architecture are <u>underlined</u>.

DeepDream [1] optimizes the input by a cross-entropy loss. It uses two regularizers including the total variance and l_2 norm on the input to improve convergence.

Activation Maximization (AM) [31] optimizes a random noise image by gradient ascent to maximize the activation of a specific class. We specifically adapt [13] for visualizing concurrent spatiotemporal representations, as it is the only prior method for visualizing features over space and time.

Quantitative evaluation. Table 1 shows the average top-1 accuracies and IS obtained by both the inverted models and verifier when inferring synthesized videos. For DeepDream and AM that do not use priming, the statistics are averaged across 10 runs per class. In LEAPS the statistics are calculated using each video in the Kinetics validation set as stimulus. For both measures, LEAPS yields consistently higher accuracies and IS compared to DeepDream and AM. Notably, it significantly improves the verifier accuracy across all three architectures. This demonstrates the merits of model priming as both DeepDream and AM optimize inputs solely by maximizing feature or class activations, without using the information-rich internal representations provided by a stimulus. This trend is also visible in the IS, as LEAPS regularizes the synthesized features in order to better represent learned temporally coherent motions.

Qualitative examples. Videos for the class *salsa spin* synthesized from different methods are shown in Figure 4. Features synthesized with LEAPS are significantly more visually distinct compared to those of baseline methods. As

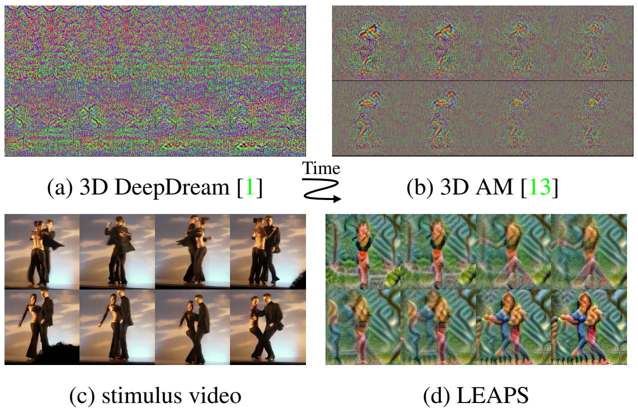


Figure 4: **Different feature synthesis methods for visualizing X3D$_M$ features** corresponding to class *salsa dancing*. Stimulus video is only used for LEAPS.

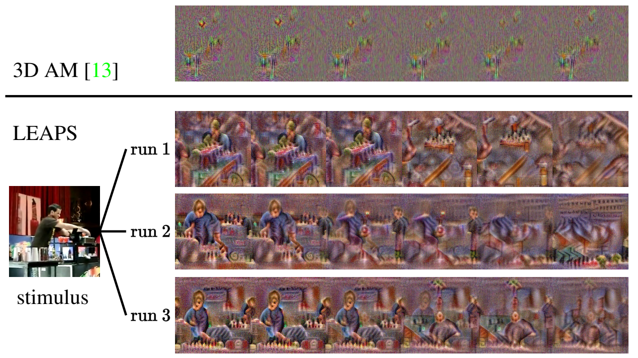


Figure 5: **Feature synthesis over different runs** for action class *bartending*. MViTv2 features are from different runs based on the same priming stimulus.

shown, videos produced by image-based methods extended to videos fail to represent learned spatiotemporal deep features. The visual quality of the synthesized videos also correlates with the accuracies and IS in Table 1.

Figure 5 illustrates synthesized videos from different runs given the same stimulus for class *bartending*. A substantial difference in the quality and representability of the visualizations between LEAPS and the extended 3D AM can be seen across runs. Despite the use of a video stimulus to prime the network, LEAPS visualizations are not constrained by the representations of the stimulus video. Each run of our LEAPS method shows a distinct visual style as feature diversity is encouraged during optimization through the homonym regularizer term. Effectively, LEAPS can be used as a tool for investigating the different class-specific features learned by each model.

4.3. Analysis of LEAPS synthesized video

The generalizability of feature visualization methods across multiple architectures is largely neglected with features from only a small subset of models being visualized.

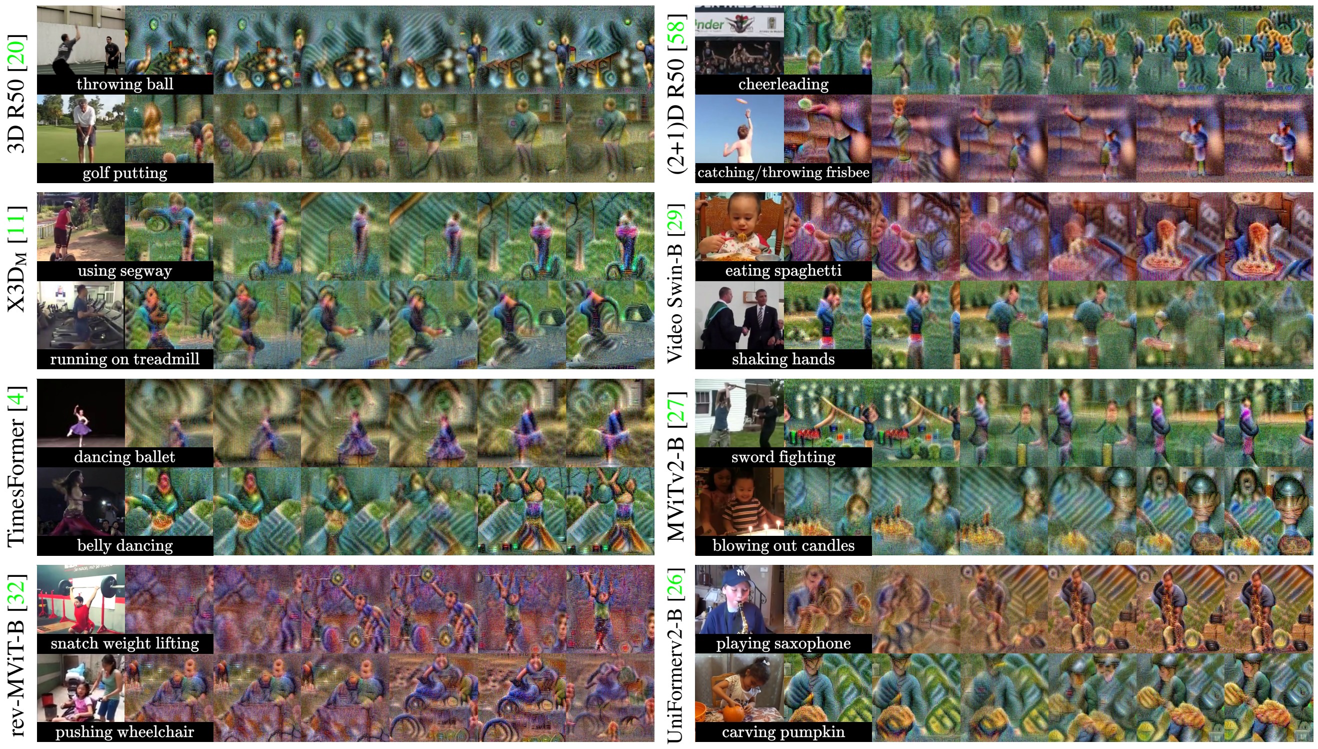
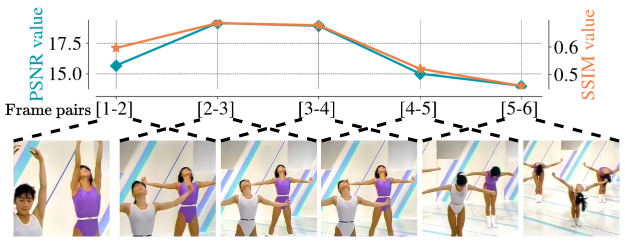
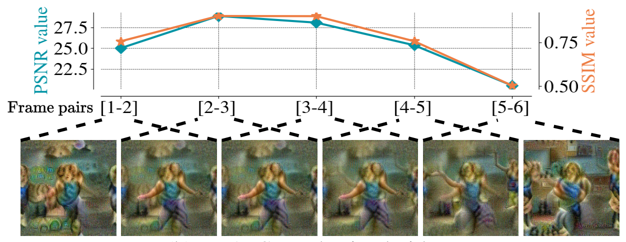


Figure 6: **Qualitative examples of synthesized spatiotemporal features with LEAPS**. Models are primed with a stimulus video, shown on the left of each row of synthesized videos.



(a) stimulus video for *doing aerobics*



(b) LEAPS synthesized video

Figure 7: **SSIM and PSNR statistics over real and synthesized videos** using Video Swin-B features. Lower values correspond to larger cross-frame differences.

To evaluate the architecture-independent nature of LEAPS and better understand the visual fidelity of the synthesized videos, we investigate the applicability of LEAPS over a range of convolutional and attention-based video models.

Generalizability. The quality of the produced feature visualizations may depend on the architecture used, as models vary in terms of their complexities and feature spaces that they employ. We compare our proposed LEAPS visualization method by inverting common video models shown in Figure 6 for an arbitrary number of Kinetics classes. A common theme that arises for all models is the association of objects with specific actions. For example, the feature visualizations for the *throwing ball*, *catching/throwing frisbee*, *eating spaghetti*, and *snatch weight lifting* actions from inverted features of 3D R50, (2+1)D R50, Video Swin-B, and rev-MViT-B respectively, all optimize the video to primarily focus on the objects associated with the specific actions. Instead, for actions that are primarily perceived by motions performed, e.g. *running on treadmill* and *dancing ballet*, the actors/performers of the target actions are shown to be more conceptually influential to the model's learned preconscious of an action, with their associated motions and movements captured by the produced visualizations. In addition, the visualizations for cases *eating spaghetti*, *blowing out candles*, and *sword fighting* reveal that networks also learn to temporally bound distinct spatiotemporal features of certain actions. This is an important ability to be learned by video models as it effectively demonstrates their capacity to filter temporal information alongside the spatial signal. Notably, LEAPS visualizations for both convolutional and attention-based models show that learned features have

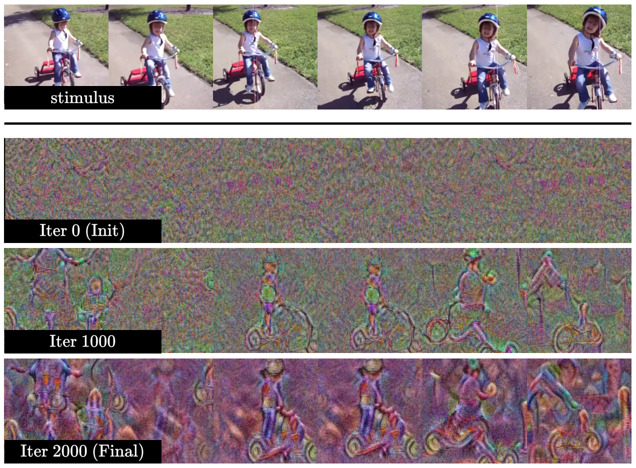


Figure 8: **Video synthesis at different optimization stages for class** *riding on a bike*. X3D$_M$ features are used.

(a) **3D R50**.

Metric	Distance function			
	l_2	l_1	cos	JVS
top-1 (m)	84.3	82.4	72.1	**86.7**
top-1 (v)	65.8	63.7	46.1	**68.5**

(b) **X3D$_M$**.

Metric	Distance function			
	l_2	l_1	cos	JVS
top-1 (m)	88.1	86.8	78.9	**90.3**
top-1 (v)	79.7	78.4	70.2	**82.5**

(c) **Video Swin-B**.

Metric	Distance function			
	l_2	l_1	cos	JVS
top-1 (m)	85.5	83.0	69.2	**87.4**
top-1 (v)	72.8	71.6	41.4	**74.3**

(d) **MViTv2-B**.

Metric	Distance function			
	l_2	l_1	cos	JVS
top-1 (m)	83.5	81.9	70.7	**85.9**
top-1 (v)	72.4	69.7	45.6	**73.1**

Table 2: **Top-1 accuracies of the inverted model (m) and verifier (v)** on synthesized videos over different priming distance functions. Best results are in **bold**.

a good correspondence to their classes across architectures.

Temporal coherence. As we show in Figure 7, LEAPS can visualize motions performed over different speeds (slow/fast). We report the Peak Signal Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) for consecutive frame pairs to analyze the variations observed by both fast and slow motions within a video. We observe that the speeds with which motions are performed within the same video can be fundamentally different with both large and small changes occurring between frames. In contrast to image-based methods extended to video, LEAPS shows the ability to represent such variations in the produced feature visualizations. As shown, the PSNR and SSIM statistics from the stimulus video of class *doing aerobics* in Figure 7a, follow similar trends as those produced by the synthesized LEAPS video in Figure 7b.

Video synthesis optimization. We visualize the resulting synthesized video $\mathbf{x}^*$ at different iteration steps during optimization in Figure 8. General features such as the outlines of objects and actors as well as their movements are synthesized first by our proposed method. Interestingly, later iterations show to refine the visualized features by further synthesizing visual details. This demonstrates a level of learned hierarchy by video models as to the types of spatiotemporal learned features that are associated with a specific action. Our proposed use of model priming in tandem with temporal coherence and feature diversity regularization terms shows to enable the visualization of models' spatiotemporal representations of actions, at a finer quality and detail.

4.4. Ablation studies

In this section, we conduct ablation studies reporting model statistics over synthesized videos. We initially consider the effect of the distance function used during model priming. Additionally, we report statistics over different combinations of priming and introduced regularizers across the tested architectures. Finally, we present quantitative results when using inputs of different spatiotemporal sizes alongside the resulting averaged latency times.

Priming distance methods. In Table 2, we evaluate the impact of the distance functions used during model priming at (1). We test three different magnitude-based methods including l_2, l_1, and JVS, as well the cosine similarity cos. Across all four spatiotemporal models, 3D R50, X3D$_M$, Video Swin-B, and MViTv2-B, the magnitude-based methods perform favorably over the cosine similarity. This is due to cos not taking into account the magnitude of the feature activation vectors from the synthesized and stimulus videos, which in turn limits the ability to synthesize representations relevant to the stimulus. Across magnitude-based methods, an average improvement of $+2.2\%$ and $+4.1\%$ to the inverted model's (m) accuracy is observed with JVS compared to l_2 and l_1 respectively. The same trend also holds true for the verifier (v), with $+1.9\%$ and $+3.7\%$ accuracy improvements from l_2 and l_1 when using JVS. Comparatively to l_2 and l_1, JVS uses a combination of vector magnitudes and angles [15], providing a more balanced approach than magnitude-only or angle-only metrics. Therefore, we adopt JVS as the priming distance method used between stimulus and synthesized feature vectors.

Regularizers. In Table 3 we provide comparisons over different priming and regularizer combinations for our LEAPS objective. We report the top-1 accuracies of the inverted model (m), and verifier (v), as well as the IS of the inverted model across a range of spatiotemporal convolutional and attention-based architectures. We note that in the case of inverting X3D$_S$ the same model is used for both model inversion and as the verifier, evidently resulting in matching verifier/inverted model accuracies. Overall, the priming objective $\underset{prim}{\mathcal{L}}$ combined with either regularizer term yields clear improvements compared to the sole use of model priming. Modest improvements are observed with the combination of priming and the feature diversity regularizer over priming with temporal coherence. We believe that this is due to the

Metric	Video model architectures and variants															
	R50			X3D [11]				TS [4]	Video Swin [29]			MViTv2 [27]		rev-MViT-B [32]	UniFormerv2 [26]	
	3D	(2+1)D [58]	CSN [57]	XS	S	M	L		T	S	B	S	B		B	L
LEAPS $\underset{prim}{\mathcal{L}}$																
top-1 (m)	67.9	63.8	72.4	67.4	68.5	73.4	73.8	64.9	69.1	69.5	71.2	70.6	71.5	64.5	72.3	73.0
top-1 (v)	53.1	49.5	55.8	51.9	68.5	59.8	60.4	52.6	52.3	53.5	57.6	54.6	56.7	50.7	56.4	56.9
IS	3.9±0.9	2.4±0.5	4.2±0.6	4.0±1.0	4.3±0.6	5.1±1.1	5.5±1.3	2.7±0.7	4.1±1.6	4.2±1.1	4.4±0.8	4.3±1.2	4.6±0.7	3.1±1.7	4.3±0.8	4.5±1.2
LEAPS $\underset{prim}{\mathcal{L}} + \underset{coh}{\mathcal{R}}$																
top-1 (m)	70.4	65.7	76.1	74.8	75.6	78.0	78.5	73.1	72.8	73.2	74.5	73.7	74.2	69.2	74.9	75.3
top-1 (v)	55.8	54.3	60.1	61.0	75.6	65.6	70.0	63.4	62.9	63.3	63.8	63.5	63.9	52.9	63.7	64.5
IS	4.6±1.2	3.5±0.8	4.8±1.6	4.3±0.9	5.0±1.5	7.2±1.3	7.5±1.0	3.9±0.9	4.5±0.7	4.8±1.3	5.4±0.6	5.0±0.8	5.3±1.0	3.3±2.1	5.5±0.6	5.9±0.4
LEAPS $\underset{prim}{\mathcal{L}} + \underset{feat}{\mathcal{R}}$																
top-1 (m)	74.3	67.4	76.2	78.9	79.4	81.1	81.5	71.9	74.1	74.7	76.0	75.6	76.3	70.4	75.8	76.6
top-1 (v)	60.2	56.9	64.3	65.8	79.4	69.3	70.8	67.2	63.2	64.5	65.4	66.7	68.0	58.2	69.0	69.8
IS	5.1±0.8	4.2±1.4	5.6±1.2	7.3±1.1	7.6±0.8	8.8±1.5	9.1±1.2	3.8±1.2	4.7±1.1	4.9±0.5	5.3±1.5	5.1±0.6	5.5±1.2	4.0±1.8	5.7±1.0	6.2±0.9
LEAPS (full)																
top-1 (m)	**86.7**	**78.0**	**88.3**	**86.2**	**87.0**	**90.3**	**90.8**	**83.6**	**85.7**	**86.2**	**87.4**	**85.1**	**85.9**	**82.5**	**87.1**	**88.3**
top-1 (v)	**68.5**	**65.2**	**71.6**	**76.4**	87.0	**82.5**	**83.7**	**69.7**	**71.9**	**73.5**	**74.3**	**72.4**	**73.1**	**67.3**	**75.4**	**76.2**
IS	**9.0±1.0**	**6.4±1.3**	**9.7±0.7**	**9.6±0.4**	**10.4±1.2**	**11.4±0.9**	**11.9±1.5**	**7.5±1.6**	**8.5±0.7**	**9.4±1.5**	**9.8±1.3**	**8.7±1.3**	**9.3±0.8**	**7.1±2.6**	**9.1±1.3**	**9.6±1.2**

Table 3: **Top-1 accuracies and Inceprion Scores (IS) over different objectives** across different architectures and model variants. X3D$_S$ is used as the verifier (v) for all experiments. The best results per variant are in **bold**. The verifier accuracy is denoted in gray for the case of using X3D$_S$ for both model inversion and as the verifier.

Model	temp.×spatial^2	top-1		IS	Latency (secs)
		m	v		(↓I / ↑ B)
3D R50	8×182^2	84.3	67.2	8.1±1.3	**0.591 / 0.913**
	8×224^2	86.7	68.5	9.7±1.0	0.832 / 1.205
	16×224^2	**87.0**	**68.7**	**9.8±1.6**	1.140 / 1.681
(2+1)D R50	8×182^2	75.4	62.9	4.2±1.8	**1.684/2.325**
	8×224^2	78.0	65.2	6.4±1.3	1.858/2.793
	16×224^2	**78.5**	**65.4**	**6.6±1.5**	2.343/3.176

Table 4: **Synthesized video size comparisons** based on the top-1 accuracies of the inverted model, and verifier, IS, and latency times for inference (↓I) and backprop (↑B). Best settings per architecture are in **bold**.

enhanced search space of $\underset{prim}{\mathcal{L}} + \underset{feat}{\mathcal{R}}$ as more diverse class features not in the stimulus are also explored. The combination of the priming objective with both regularizer terms for our proposed LEAPS consistently achieves the best results by a large margin compared to all other settings. LEAPS improves accuracy, for both the inverted model and verifier, as well as the quality of the generated videos based on the IS. These results further demonstrate that our proposed LEAPS optimization can be used across a range of architectures as a general spatiotemporal feature visualization method. Further qualitative examples for each network over different Kinetics classes are shown in Figures S1 to S4 in the supplementary alongside embeddings projections of LEAPS with different regularizer regimes in Section S3.

Synthesized video resolution. Finally, we compare the accuracies, IS, and latency times when optimizing video inputs of different spatiotemporal sizes. Due to the majority of architectures requiring fixed-size inputs, we use 3D R50 and (2+1)D R50 as they are input size independent. From the top-1 (m/v) accuracies and IS summarized in Table 4, the best-performing setting across metrics is obtained with 16×224^2-sized inputs. We observe comparable performance on the temporally-reduced 8×224^2 setting with a significantly more balanced performance-to-latency. Notable decreases in accuracies and IS are observed with the potentially limited spatial resolution 8×182^2 setting, making it less suitable. Due to the significant improvements in the per-iteration latency of the 8×224^2 inputs, we adopt this setting throughout our experiments.

5. Conclusions

We have introduced LEAPS, a novel spatiotemporal model inversion method for visualizing the learned internal representations of networks through video synthesis. LEAPS uses a stimulus video to prime a model and iteratively optimize an input by minimizing the classification and priming loss. The resulting synthesized video visualizes learned concepts that are associated with classes without prior knowledge of the training data. During optimization, LEAPS uses two regularizers. The first enforces temporal coherence between feature transitions across frames in the updatable input. The second improves the diversity of the synthesized features through a domain-specific verifier network to enhance the search space. The proposed architecture-independent method has shown qualitatively and quantitatively that it can produce high-quality and visually coherent synthesized videos over a wide range of spatiotemporal convolutional and attention-based video models. The high classification scores and synthesized video quality metrics make LEAPS a generalizable and effective spatiotemporal feature visualization method. We believe that this first step towards learned spatiotemporal representations synthesis is a promising direction in understanding video models.

Acknowledgments. We use publicly available datasets and models. Research is funded by imec.icon Surv-AI-llance project and FWO (Grant G0A4720N).

References

[1] Mordvintsev Alexander, Olah Christopher, and Tyka Mike. Inceptionism: Going deeper into neural networks. *Google Research Blog*, 2015. 2, 4, 5

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *International Conference on Computer Vision (ICCV)*, pages 6836–6846. IEEE, 2021. 1

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 2

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *International Conference on Machine Learning (ICML)*, pages 813–824. PMLR, 2021. 1, 4, 6, 8

[5] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks (ICANN)*, pages 63–71. Springer, 2016. 2

[6] Michael Burke. Leveraging gaussian process approximations for rapid image overlay production. In *ACM Multimedia Workshop (ACMMW)*, pages 21–26. ACM, 2017. 2

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6299–6308. IEEE, 2017. 4

[8] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847. IEEE, 2018. 2

[9] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4829–4837. IEEE, 2016. 2

[10] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. Technical Report 1341-3, University of Montreal, 2009. 2

[11] Christoph Feichtenhofer. X3D: Expanding architectures for efficient video recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 203–213. IEEE, 2020. 1, 4, 6, 8

[12] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *International Conference on Computer Vision (ICCV)*, pages 6202–6211. IEEE, 2019. 1

[13] Christoph Feichtenhofer, Axel Pinz, Richard P Wildes, and Andrew Zisserman. Deep insights into convolutional networks for video recognition. *International Journal of Computer Vision*, 128:420–437, 2020. 2, 5

[14] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1933–1941. IEEE, 2016. 2

[15] Basura Fernando and Samitha Herath. Anticipating human actions by correlating past with the future with jaccard similarity measures. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13224–13233. IEEE, 2021. 3, 7

[16] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *International Conference on Computer Vision (ICCV)*, pages 2950–2958. IEEE, 2019. 2

[17] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *International Conference on Computer Vision (ICCV)*, pages 3429–3437. IEEE, 2017. 2

[18] Sigmund Freud and James Ed Strachey. Vol. XIX The Ego and the Id and other works (1923-1925). In *The Standard Edition of the Complete Psychological Works of Sigmund Freud*. the Hogarth Press, 1959. 2

[19] Amin Ghiasi, Hamid Kazemi, Steven Reich, Chen Zhu, Micah Goldblum, and Tom Goldstein. Plug-in inversion: Model-agnostic inversion for vision with data augmentations. In *International Conference on Machine Learning (ICML)*, pages 7484–7512. PMLR, 2022. 2

[20] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6546–6555. IEEE, 2018. 4, 6

[21] De-An Huang, Vignesh Ramanathan, Dhruv Mahajan, Lorenzo Torresani, Manohar Paluri, Li Fei-Fei, and Juan Carlos Niebles. What makes a video a video: Analyzing temporal information in video understanding models and datasets. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7366–7375. IEEE, 2018. 2

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456. PMLR, 2015. 4

[23] Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5050–5058. IEEE, 2021. 2

[24] Atsushi Kikuchi, Kotaro Uchida, Masaki Waga, and Kohei Suenaga. Borex: Bayesian-optimization–based refinement of saliency map for image-and video-classification models. In *Asian Conference on Computer Vision (ACCV)*, pages 2092–2108, 2022. 2

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 4

[26] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Limin Wang, and Yu Qiao. Uniformerv2: Spatiotemporal learning by arming image vits with video uniformer. *arXiv preprint arXiv:2211.09552*, 2022. 1, 4, 6, 8

[27] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for

classification and detection. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4804–4814. IEEE, 2022. 1, 4, 6, 8

[28] Zhenqiang Li, Weimin Wang, Zuoyue Li, Yifei Huang, and Yoichi Sato. Towards visually explaining video understanding networks with perturbation. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 1120–1129. IEEE, 2021. 2

[29] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3202–3211. IEEE, 2022. 1, 4, 6, 8

[30] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196. IEEE, 2015. 2

[31] Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120:233–255, 2016. 2, 5

[32] Karttikeya Mangalam, Haoqi Fan, Yanghao Li, Chao-Yuan Wu, Bo Xiong, Christoph Feichtenhofer, and Jitendra Malik. Reversible vision transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10830–10840. IEEE, 2022. 1, 4, 6, 8

[33] Anthony J Marcel. Conscious and unconscious perception: An approach to the relations between phenomenal experience and perceptual processes. *Cognitive psychology*, 15(2):238–300, 1983. 2

[34] Anthony J Marcel. Conscious and unconscious perception: Experiments on visual masking and word recognition. *Cognitive psychology*, 15(2):197–237, 1983. 2

[35] Vivek Miglani, Narine Kokhlikyan, Bilal Alsallakh, Miguel Martin, and Orion Reblitz-Richardson. Investigating saturation effects in integrated gradients. In *International Conference on Machine Learning Workshops (ICMLW)*. PMLR, 2020. 2

[36] Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *International Conference on Machine Learning (ICML)*, pages 737–744. PMLR, 2009. 3, 4

[37] Mamuku Mokuwe, Michael Burke, and Anna Sergeevna Bosman. Black-box saliency map generation using bayesian optimisation. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020. 2

[38] James H. Neely. Priming. In *Encyclopedia of Cognitive Science*. Nature Publishing Group, 2003. 2, 3

[39] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4467–4477. IEEE, 2017. 2

[40] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in Neural Information Processing Systems (NeurIPs)*, pages 3395–3403, 2016. 2

[41] Anh Nguyen, Jason Yosinski, and Jeff Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. In *International Conference of Machine Learning Workshops (ICMLW)*. PMLR, 2016. 2

[42] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International Conference on Machine Learning (ICML)*, pages 2642–2651. PMLR, 2017. 2

[43] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017. 2

[44] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *British Machine Vision Conference (BMVC)*. BMVA, 2018. 2

[45] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2234–2242. PMLR, 2016. 4

[46] Sandeep Singh Sandha, Mohit Aggarwal, Igor Fedorov, and Mani Srivastava. Mango: A python library for parallel hyperparameter tuning. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3987–3991. IEEE, 2020. 4

[47] Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1262–1273, 2019. 4

[48] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision (ICCV)*, pages 618–626. IEEE, 2017. 2

[49] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning (ICML)*, pages 3145–3153. PMLR, 2017. 2

[50] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2

[51] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 2

[52] J Springenberg, Alexey Dosovitskiy, Thomas Brox, and M Riedmiller. Striving for simplicity: The all convolutional net. In *International Conference on Learning Representations Workshops (ICLRW)*, 2015. 2

[53] Alexandros Stergiou. The mind's eye: Visualizing class-agnostic features of cnns. In *International Conference on Image Processing (ICIP)*, pages 2738–2742. IEEE, 2021. 2

[54] Alexandros Stergiou, Georgios Kapidis, Grigorios Kalliatakis, Christos Chrysoulas, Ronald Poppe, and Remco Veltkamp. Class feature pyramids for video explanation. In *International Conference on Computer Vision Workshop (ICCVW)*, pages 4255–4264. IEEE, 2019. 2

[55] Alexandros Stergiou, Georgios Kapidis, Grigorios Kalliatakis, Christos Chrysoulas, Remco Veltkamp, and Ronald Poppe. Saliency tubes: Visual explanations for spatio-temporal convolutions. In *International Conference on Image Processing (ICIP)*, pages 1830–1834. IEEE, 2019. 2

[56] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, pages 3319–3328. PMLR, 2017. 2

[57] Du Tran, Heng Wang, Lorenzo Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. In *International Conference on Computer Vision (ICCV)*, pages 5552–5561. IEEE, 2019. 4, 8

[58] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6450–6459. IEEE, 2018. 4, 6, 8

[59] Tomoki Uchiyama, Naoya Sogi, Koichiro Niinuma, and Kazuhiro Fukui. Visually explaining 3D-CNN predictions for video classification with an adaptive occlusion sensitivity analysis. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 1513–1522. IEEE, 2023. 2

[60] Feng Wang, Haijun Liu, and Jian Cheng. Visualizing deep neural network by alternately image blurring and deblurring. *Neural Networks*, 97:162–172, 2018. 2

[61] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 24–25. IEEE, 2020. 2

[62] 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 (ECCV)*, pages 20–36. Springer, 2016. 2

[63] Shen Yan, Xuehan Xiong, Anurag Arnab, Zhichao Lu, Mi Zhang, Chen Sun, and Cordelia Schmid. Multiview transformers for video recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3333–3343. IEEE, 2022. 1

[64] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deep-inversion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8715–8724. IEEE, 2020. 2, 3, 4

[65] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In *International Conference of Machine Learning Workshops (ICMLW)*. PMLR, 2015. 2

[66] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision (ECCV)*, pages 818–833. Springer, 2014. 2