Learning Goals from Failure

Dave Epstein and Carl Vondrick
Columbia University

Abstract

We introduce a framework that predicts the goals behind observable human action in video. Motivated by evidence in developmental psychology, we leverage video of unintentional action to learn video representations of goals without direct supervision. Our approach models videos as contextual trajectories that represent both low-level motion and high-level action features. Experiments and visualizations show our trained model is able to predict the underlying goals in video of unintentional action. We also propose a method to "automatically correct" unintentional action by leveraging gradient signals of our model to adjust latent trajectories. Although the model is trained with minimal supervision, it is competitive with or outperforms baselines trained on large (supervised) datasets of successfully executed goals, showing that observing unintentional action is crucial to learning about goals in video.

1. Introduction

Goal-directed action is all around us. Even though Figure 1 shows a person performing an unconventional action (heating a wine bottle with a blowtorch), we cannot help but to perceive the action as rational in the context of the goal (to open the bottle).

Predicting the goal of action may seem challenging because future goals are not directly observable in video. However, in a series of papers, development psychologists Amanda Woodward and Michael Tomasello demonstrated that children reason about goals before their second birthday [46, 54], and this reasoning plays a key role in rapid development of communicative skills [47] and mental representations of the world [2]. Despite the relative ease of this task for children, machine recognition of goals has remained challenging.

The hypothesis underlying this paper is that examples of *failure* are key missing pieces in action recognition systems. Without observing unintentional action, we cannot expect models to discriminate goals from actions. Examples demonstrating unintentional action are necessary to decouple these two notions, separating between the visible action and the latent goals. As Efros has been telling us all along, it is all about the data [19], and negative data doubly so [57].

The main observation behind our approach is that natural video will contain abundant and rich examples of both intentional and unintentional action [8], which we can leverage for learning. In our model, video is represented as a trajectory, and goals are encoded as the path for the trajectory. Given examples of videos with variable success, we present a model that learns goal-oriented video representations by discriminating between success and failure. Our model captures both motion and relational features through an attention-based transformer architecture, allowing end-to-end training.

Our experiments show that failure data is crucial for

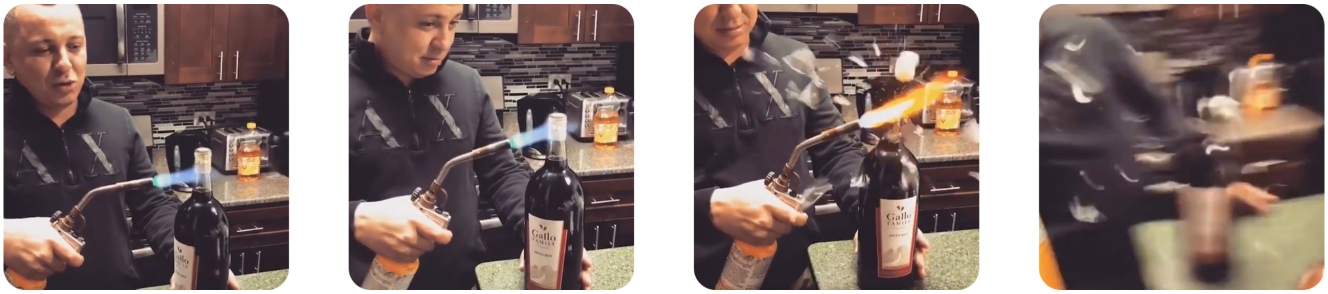


Figure 1: **What are they doing?** While just the action is observable (heating the bottle), we still predict the goal behind the action (to open the bottle). In this paper, we learn from failure examples to learn representations of goals in video.

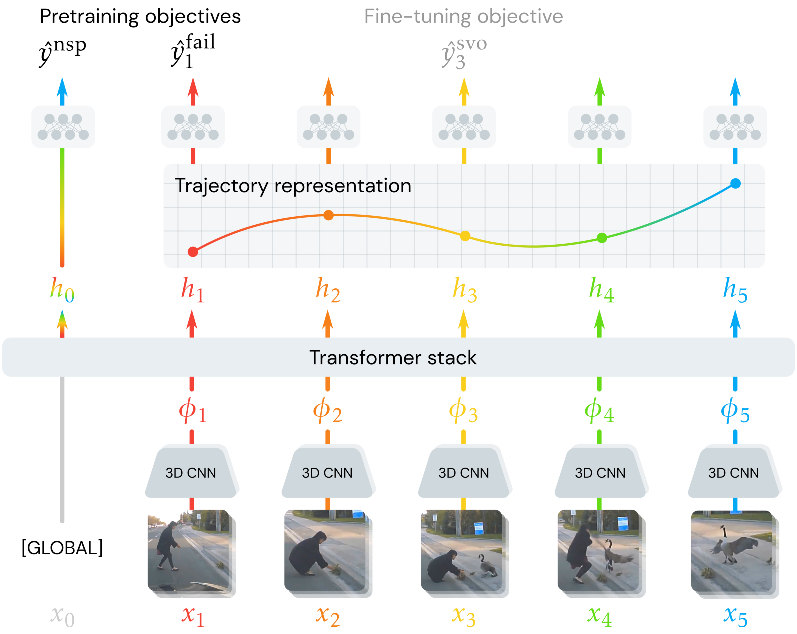


Figure 2: **Learning goal-oriented video representations:** We show an overall view of our approach. First, we embed short clips using a 3D CNN to represent short-term motion features. Then, we run the sequence of CNN embeddings through a stack of Transformers, where they interact with each other to finally form a context-adjusted latent action trajectory. The model is trained end-to-end from scratch, with intentionality and temporal coherence losses (depicted top-left). Points along the resultant trajectory are decoded with linear projections into various spaces (top-middle).

learning representations of goals. We evaluate our model on three goal prediction tasks. First, we experiment on detecting unintentional action in video, and we demonstrate strong performance over baselines on this task. Second, we evaluate the representation at predicting goals with minimal supervision, which we characterize as structured categories consisting of subject, action, and object triplets. Lastly, we use our representation to automatically "correct" unintentional action and decode these corrections by retrieving from other videos or generating categorical descriptions.

Our main contribution is an approach that, training on data of unintentional action, learns a goal-directed representation of videos. We show that our model often captures the latent goals behind observed action, performing on par with or better than supervised models trained on large labeled datasets of only intentional action. We also introduce a method to find minimal adjustments to the path and "automatically correct" unintentional action in video. The remainder of this paper will describe this approach in detail. Code, data, and models will be available.

2. Related Work

Recognizing action in video: Previous work explores many different approaches to recognizing action in video. Earlier directions develop hand-designed features to process spatio-temporal information for action recognition [29, 27, 51, 39]. Popular deep learning architectures for images were extended to operate directly on video by modeling time as a third dimension [17, 4, 44, 31, 24]. To deal with variable-length or long video input, previous work frequently takes one of two approaches: pooling or recurrent networks. However, pooling loses spatial and/or temporal connections between different moments of video. Since recurrent networks are sequential, they require selecting important video features ahead of time, without viewing full context. RNNs are also known to struggle to connect between far-apart inputs, which creates significant challenges in modeling long-term video. [45] is most similar to our approach, since they also run clips through 3D CNNs and Transformers, but they freeze 3D CNNs and train on a "masked video modeling" task, ultimately discarding contextually learned temporal dynamics across videos since their goal is to learn information useful for an effective cross-modal representation. To address these drawbacks, we propose a 3D-CNN-Transformer model which allows for short-term, granular motion detection combined with a long-term action representation, trained end-to-end from scratch.

Learning about intention: Evidence in developmental psychology quantifies why humans perceive intention [2], how we perceive it [56, 55, 54], when we begin to do so [32, 33], and what allows us to infer the goals behind others' behavior [42]. Early work in computer vision has investigated assessing the quality of action execution [40, 7, 38], which our work builds upon. However, we view quality from a goal-directed perspective and automatically correct unintentional action with minimal supervision. We take advantage of signals in unconstrained video collections of both intentional and unintentional action [8] to learn about goals from video.

Leveraging adversarial attacks: We use adversarial gradients [12, 28] to find adjustments to learned video representations which "auto-correct" unintentional action back onto the manifold of intentional action. Previous work studied adversarial attacks in steganography [18, 61], software

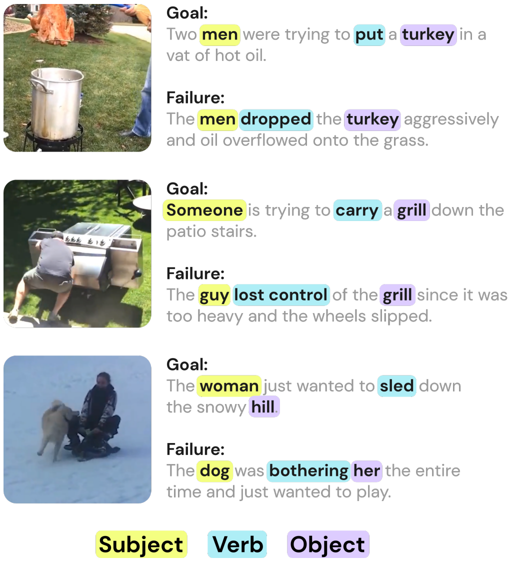

Goal:
Two men were trying to put a turkey in a vat of hot oil.
Failure:
The men dropped the turkey aggressively and oil overflowed onto the grass.
Goal:
Someone is trying to carry a grill down the patio stairs.
Failure:
The guy lost control of the grill since it was too heavy and the wheels slipped.
Goal:
The woman just wanted to sled down the snowy hill.
Failure:
The dog was bothering her the entire time and just wanted to play.
Subject
Verb
Object

Figure 3: **Labeling goals and failures in video:** To evaluate our representation, we annotate the Oops! dataset with short sentences describing the goals and failures. We extract subject-verb-object triples and train a decoder on learned representations. The intentional and unintentional action in the dataset span a diverse range of categories.

bug-finding [41], generating CAPTCHAs [50] to fool modern deep nets [37], generating interesting images [43], creating real-world 3D objects that trick neural networks [60, 1], and in training models more robust to test-time adversarial attacks [36, 12, 35]. [22] extend this concept to generative models, setting a new image output as a target label and perturbing latent space. In video, [26, 53] introduce various methods to fool action recognition networks, often on a 3D CNN backbone. We instead utilize adversarial attacks to manipulate and correct unintentional action.

3. Unintentional Action and Goals Dataset

Similar to how children learn about goals by perceiving failed attempts at executing them [33], we hypothesize that examples of failure are crucial for learning to discriminate between action and goal. Without observing unintentional action, models can not learn the pattern discriminating action and intention. We build on the Oops! dataset [8], which is a large collection of videos containing intentional and unintentional action, to train and evaluate our models. Videos in this dataset are annotated with the moment at which action becomes unintentional. Figure 3 shows some example frames. We also use the Kinetics dataset [3] to evaluate models, since it contains a wide range of successful actions.

We would like to learn a representation of goals that only requires visual information to train. However, evaluating trained models and probing them for an understanding of goals requires gathering labels of goals. Therefore, we expand [8] with textual descriptions of goals and failures in the dataset, and use these annotations to evaluate our (trained, frozen) model in comparison to other representations.

3.1. Goal and Failure Annotation

Established action datasets in computer vision [13, 30] contain annotations about person and object relationships in scenes, but they do not directly annotate the goal, which we need for evaluation of goal prediction. We collect unconstrained natural language descriptions of a subset of videos in the Oops! dataset (4675 training videos and 3404 test videos), prompting Amazon Mechanical Turk workers[1] to answer “What was the goal in this video?” as well as “What went wrong?”. We then process these sentences[2] to detect lemmatized subject-verb-object triples, manually correcting for common constructions such as “tries to X” (where the verb lemma is detected as “try”, but we would like “X”). The final vocabulary contains 3615 tokens. Figure 3 shows some example annotations. Detailed statistics for processed SVO triples are provided in the Supplementary Material. We use SVO triples to evaluate the video representations.

4. Method

In this section, we introduce our framework to learn goal-oriented trajectory representations of video. Our method accepts as input sequences of video input depicting intentional and/or unintentional action, and learns to represent these sequences as latent trajectories, from which intentionality of action is predicted. We show in Section 5 that, having observed unsuccessful action as well as successful, our trained model learns trajectories which capture the goals latent in the input video.

4.1. Visual Dynamics as Trajectories

A common approach to representing video data is to run each clip through a convolutional network and combine clip representations by pooling to run models on entire sequences [9, 14, 11, 59]. However, these methods do not allow for connections between different moments in video and cannot richly capture temporal relationships, which give rise to goal-directed action. While recurrent networks [20] are more expressive, they require compressing history into a fixed-length vector, which forces models to select relevant visual features without viewing full context and makes reasoning about connections between different moments difficult, especially when they are far apart.

Temporal streams of visual input are highly contextual with both short- and long-term dependencies. We will represent video as a contextually-adjusted trajectory of latent

[1] with $> 10k$ approvals at a $\geq 99\%$ rate

[2] Using the Spacy.io natural language library

representations in a learned space. Figure 2 illustrates this architecture, which has both a motion and action level:

Motion Level: First, we separate video into short clips (or tokens) in order to make initial motion-level observations. Let x be a video, and x_i be a video clip centered at time i. We estimate the motion-level features $\phi_i = f(x_i)$ where f is a 3D CNN [25].

Action Level: Second, we model relations between ϕ_i to construct a contextual trajectory $h_i = g(\phi_i)$ where g is the Transformer [49]. The Transformer accepts as input a sequence of motion-level representations $\{\phi_i\}_{i=1}^n$, repeatedly performs self-attention among them, in the same spirit as the forward pass of a graph neural network, with video clips as nodes [58]. The output of the Transformer is a final latent path $\{h_i\}_{i=1}^n$. Since the self-attention operation can incorporate contributions from both nearby and far away moments in its representations for each clip, the Transformer is well-suited to modeling higher-level connections between the atomic actions recognized at the motion level. The Transformer's output $\{h_i\}_{i=1}^n$ can then be applied in different downstream tasks.

4.2. Learning with Indirect Supervision

We learn the representation with weak, indirect supervision that is accessible at large scales. This supervision is also truer to how humans learn about intention, since we do not require labeled action semantics, but do often receive environmental cues about whether others' action is intentional or not [5]. We use the following two objectives for learning:

Action Intentionality: We train the model to temporally localize when action is unintentional. We assume that the video frame where the action shifts from intentional to unintentional is labeled [8], and note that these labels are a significantly weaker form of supervision than semantic action categories. For each video clip x_i, we set the target $y_i^{\text{fail}} \in \{0, 1, 2\}$ according to whether the labeled frame happens before, during, or after the clip x_i. The model estimates $\hat{y}_i^{\text{fail}} = \text{softmax}(w_1^T h_i)$ with a linear projection where w_1 is a jointly learned projection matrix to $\mathbb{R}^3$. We train with a cross-entropy loss between $\hat{y}^{\text{fail}}$ and y^{fail} where the class weight is set to the inverse frequency of the class label to balance training. We label this loss $\mathcal{L}^{\text{fail}}$.

Temporal Consistency: We also train the model to learn temporal dynamics with a self-supervised consistency loss [14, 34, 10, 52, 23, 6]. Let $y^{\text{nsp}} = 1$ indicate that the sequence is consistent. We predict whether the input sequence is temporally consistent with $\hat{y}^{\text{nsp}} = \sigma(w_2^T h_0)$ where w_2 is a jointly learned projection to $\mathbb{R}$. We train with the binary cross-entropy loss between y^{nsp} and $\hat{y}^{\text{nsp}}$. We label this loss $\mathcal{L}^{\text{nsp}}$ (next sequence prediction). This loss encourages the model to learn longer-term patterns in human action.

We create inconsistent sequences as follows: For each video sequence in the batch, we bisect the sequence into two parts at a random index with probability $p_{\text{split}} = 0.5$. For these sequences, we perturb the video segments with probability $p_{\text{perturb}} = 0.5$. When perturbing, we swap the order of the two sequences with probability $p_{\text{swap}} = 0.3$, otherwise we pick a randomly sized subsequence from another video sequence in the batch to replace one of the two segments.

A large line of recent and concurrent work has tackled the problem of self-supervised representation learning in video (*e.g.* [14, 15, 16, 34, 10]). Our paper focuses on the value of training on data of unintentional action to learn goals, and we use the self-supervised temporal consistency loss to encourage our model to reason about longer sequences of action, especially useful for the automatic correction of unintentional action demonstrated in Section 5.4. Other self-supervised losses could be incorporated into our framework to serve the same purpose.

Training: To train our model, we set the overall loss as $\mathcal{L} = \mathcal{L}^{\text{fail}} + \lambda\mathcal{L}^{\text{nsp}}$, where λ is a hyperparameter controlling the importance of the coherence loss. We set $\lambda = 0.5$ to balance the magnitudes of the losses.

5. Experiments

5.1. Experimental Setup

Baselines: We evaluate the 3D CNN from [8] which is trained from scratch on the action intentionality loss (Section 4.2). We also evaluate a 3D CNN pre-trained on Kinetics action recognition, which is frozen unless indicated otherwise. The 3D CNN trained on Kinetics is the current state of the art in video representation learning when transferred to many downstream tasks, and represents a high-water mark for performance when training only on intentional action. Further, to fairly compare the Transformer layer to 3D CNNs which take in one short clip only, we pool 3D CNN predictions locally with neighboring predictions such that both methods have the same effective temporal receptive field.

We evaluate our learned representations by freezing them and then decoding them via retrieval as well as goal and failure prediction.

Retrieval: We perform nearest-neighbor retrieval among one-second long clips in the test sets for the Oops! and Kinetics datasets. While we do not learn a representation using Kinetics data, we include a subset of Kinetics (of the same size as the Oops! validation set) in retrieval, to see if auto-corrected actions match with successfully executed goals in Kinetics rather than failed attempts (see Section 5.4). This decoder maintains a lookup table of all clip representations and computes the k-nearest neighbors from different videos using cosine distance.

Categorization: We also implement a decoder using the textual labels we gathered on the videos. Here, the task is to

Method	Localization		Classification
	01 sec	0.25 sec	Accuracy
Kinetics [4] finetune	**75.9**	**46.7**	64.0
Kinetics frozen + linear	69.2	37.8	53.6
3D CNN only [8]	68.7	39.8	59.4
Our model	**72.4**	**39.9**	**77.9**
Chance	25.9	6.8	33.3

Table 1: **Detecting unintentional action:** We evaluate models on classifying and localizing unintentional action on the Oops! Our model is competitive with Kinetics supervised features on unintentional action localization despite training from scratch, outperforming it on three-way classification. Since our model learns how to relate between different moments in time, instead of naively pooling, it is able to make better use of temporal context to solve these tasks.

Features	Subject		Verb		Object		Average		All three	
	R1	R5	R1	R5	R1	R5	R1	R5	R1	R5
Kinetics [4]	26.8	72.3	27.3	52.7	36.0	**64.6**	30.0	**63.2**	2.1	**16.5**
3D CNN [8]	29.4	72.7	26.4	50.4	44.7	57.9	33.5	60.3	2.9	13.9
Random	23.7	55.7	22.7	45.4	44.8	52.7	30.4	51.3	1.4	8.7
Our Model	**34.3**	**74.5**	**29.7**	**54.2**	**45.0**	58.2	**36.3**	62.3	**3.3**	14.4
Chance	0.1								<0.1	

Table 2: **Comparison of Representations:** To evaluate how well representations encode goals, we freeze them and estimate a linear projection to predict labelled subject-verb-object triples in the Oops! validation set. We evaluate top-1 and top-5 recall (R1, R5). By observing sequences of both intentional and unintentional action, our model performs competitively with others trained on large labeled datasets of successful action.

describe the goals of the input video using the SVO triplets. We train a decoder to predict the *main goal* for clips with intentional action (before the onset of failure), and predict *what went wrong* for clips with unintentional action, using labels gathered as described in Section 3.1. The estimated decoder will describe intentional action in video with descriptions of the goal, for example "athlete wins game" and not "throwing ball", which is an action. Unintentional action, in turn, will be described as "man spills groceries" instead of a generic action category such as "walking". We train a linear layer to output a vector for subject, verb, and object. As ground truth, we use BERT word embeddings [6], calculating scores using dot product and running them through softmax and a cross-entropy loss.

5.2. Unintentional Action Detection

We evaluate the model at detection and temporal localization when action deviates from its goal. We use labels from the test set in [8] as the ground truth. We process videos with our model, sampling continuous one-second clips as tokens, and take the predicted localization as the center of the clip with maximum probability of failure. We also classify each clip according to its label (intentional, transitional, or unintentional). We show results in Table 1. On the former task, our model is competitive with fine-tuning a fully-supervised Kinetics CNN, despite using less data and less supervision. On classification, our network outperforms the Kinetics network by 14%, showing that representing videos as contextual trajectories is effective.

5.3. Goal Prediction

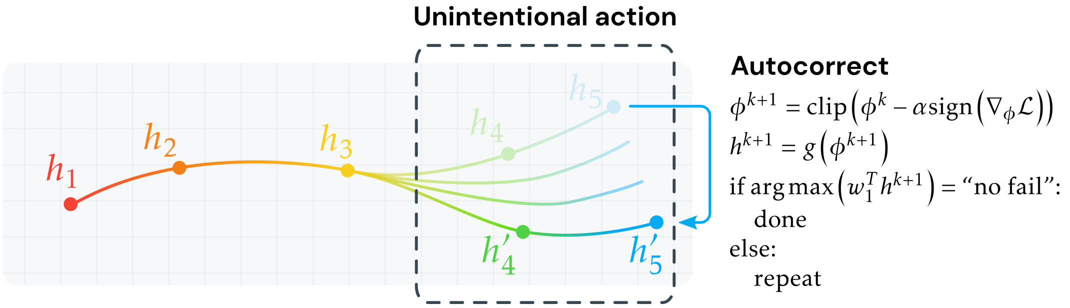
We next evaluate the model at predicting goal descriptions. We train a decoder on the trajectory to read out subject, verb, object triplets. In this task, ground truth is the labeled goal if action is intentional, and the labeled failure if action is unintentional. In training, if sentences have more than one extracted SVO, we randomly select one as ground truth. In testing, we average-pool predictions among all clips with intentional action and unintentional action separately and take the maximum over all sentence SVOs. Each video clip has two pooled predictions: one for video showing intentional action (where ground truth is the labeled goal of the video), and one for video showing unintentional action (where ground truth is the labeled failure). Table 2 shows our model obtains better top-1 accuracy on all metrics than baselines, including the Kinetics-pretrained model, and is competitive on top-5 accuracy, highlighting the importance of observing failure for understanding goals.

5.4. Completing Goals by Auto-Correcting Trajectories

We would like to use our learned representation in order to infer the goals of people in scenes of unintentional action. However, since the model is trained with indirect supervision, the trajectories h are not supervised with goal states. We propose to formulate goal completion as a latent trajectory prediction problem. Given an observed trajectory of unintentional action h, we seek to find a new, minimally modified trajectory h' that is classified as intentional. By analogy to how word processors auto-correct a sentence, we call this process **action auto-correct**. We illustrate this process in Figure 4.

We find this correction in feature space, not pixel space, to yield interpretable results. We find a gradient to the features ϕ that switches the prediction $\hat{y}_i^{\text{fail}}$ to be the "intentional" category for all clips i.

We formulate an optimization problem with two soft constraints. Firstly, we want to increase the classification score of intentional action $\mathcal{L}^{\text{fail}}$. Secondly, we want the resulting trajectory to be temporally consistent $\mathcal{L}^{\text{nsp}}$. Without this



```
$\phi^{k+1} = \text{clip}\left(\phi^k - \alpha \text{sign}\left(\nabla_\phi \mathcal{L}\right)\right)$
$h^{k+1} = g\left(\phi^{k+1}\right)$
if $\arg\max\left(w_1^T h^{k+1}\right) =$ “no fail”:
    done
else:
    repeat
```

Figure 4: **Automatically correcting unintentional action:** Starting from an initial trajectory, we use model gradients as a signal to correct the course of points representing unintentional action (Section 5.4). We evaluate corrected trajectories by decoding them into SVO triples and retrieving nearest neighbors from a databank.

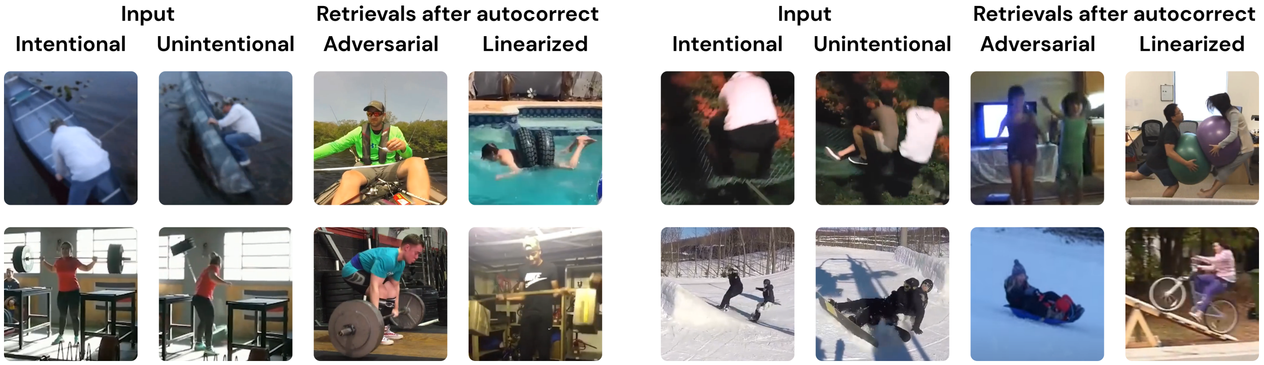


Figure 5: **Retrievals from Auto-corrected Trajectories:** We show the nearest neighbors from auto-corrected action trajectories, using our proposed method and a linearization baseline. The retrievals are computed across both the Oops! and Kinetics datasets, since Kinetics contains many examples of goals being successfully executed, whereas Oops! focuses on unintentional action. The corrected representations yield corrected trajectories that are often embedded close to the goal.

term, the corrected trajectory is not required to be coherent with the initial part of the original trajectory. We minimize this modified cost function with respect to $\phi'_{t:T}$:

$$J = \max\left(0, \mathcal{L}^{\text{nsp}}_{y=1}(\phi') - \mathcal{L}^{\text{nsp}}_{y=1}(\phi)\right) + \lambda \sum_i \mathcal{L}^{\text{fail}}_{y=0}(\phi'_i)$$

where $\mathcal{L}$s are the original loss functions but with target labels y^{fail} overridden to be the intentional class, and $\lambda = 2$ is a scalar to balance the two terms. We only modify ϕ on the clips which the model classifies as unintentional in the first place, which we denote $\phi'_{t:T}$. The coherence loss is also truncated by its original value, causing the optimization to favor a trajectory that is no less temporally coherent than the original one.

To solve this optimization problem, we use the iterative target class method [28], which repeatedly runs the input through the model and modifies it in the direction of the desired loss. For every ϕ_i corresponding to a clip where action is unintentional, we repeat a gradient attack step towards the target $y^{\text{fail}}_i = 0$. The complete update is:[3]

$$\phi^{k+1}_{t:T} = \texttt{clip}\left[\phi^k_{t:T} - \alpha\,\texttt{sign}\left(\nabla_{\phi_{t:T}} J\right), \phi_{t:T} \pm \epsilon\right] \qquad (1)$$

where $\phi^0_{t:T} = \phi_{t:T}$. We repeat this process until the network is “fooled” into classifying the input as intentional action, for at most k_{max} iterations or until $\arg\max \hat{y}^{\text{fail}}_i = 0$. Once the halting condition is satisfied, we run the modified ϕ' vectors through the model, yielding a trajectory of corrected action h' that encodes successful completion of the goal. In other words, goals are the adversarial examples [12] of failed action – instead of viewing adversarial examples as a bug, we view them as a feature [21].

As a comparison, we implement a simple baseline where we linearly extrapolate the trajectory of observed intentional action: if the unintentional action in a sequence of clips $\{x_i\}_{i=0}^n$ begins at clip j, we extend the trajectory for a clip $x_k \in \{x_j, \dots, x_n\}$ by setting $h_k = h_j + (k-j)\frac{h_j - h_0}{j}$. We find this baseline to outperform other naive ones such as the identity function (*i.e.* leaving the representation untouched)

[3] We found $k_{max} = 25, \alpha = 0.03, \epsilon = 1$ to be reasonable values.

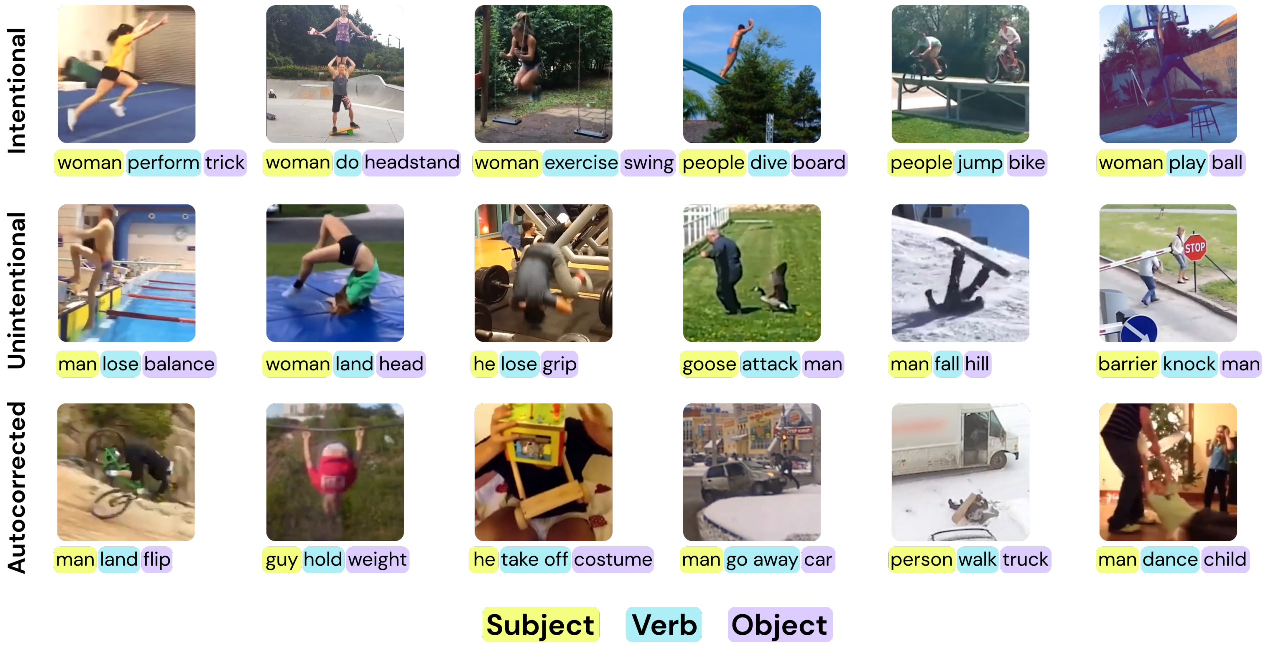


Figure 6: **Decoding the Trajectories:** We run our trained subject-verb-object decoder on different segments of Oops! videos. Row 1 shows clips of intentional action, and the trained decoder predicts the latent goal. Row 2 shows unintentional action, and the trained decoder now predicts failures instead. The final row also shows unintentional videos, but we run our auto-correction algorithm before predicting SVOs. The trained decoder returns to predicting goals, suggesting the auto-correct procedure shifts the failed trajectories towards successful ones.

and using the representation of the last moment before unintentional action.

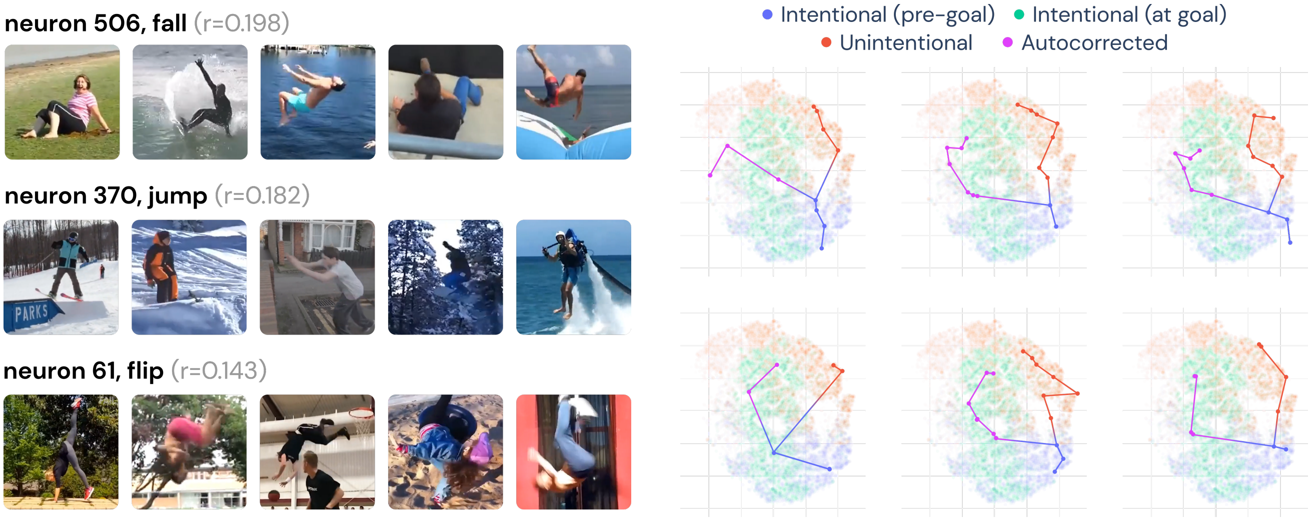
Figure 5 shows examples of nearest neighbor retrievals of the corrected latent vectors, computing over the Oops! and Kinetics test sets. Despite not training on Kinetics (i.e. on videos with completed goals), our representation can adjust video trajectories such that their nearest neighbors are goals being successfully executed. We also examine the effects of auto-correction on the frozen SVO decoder. Table 3 shows these results. For decoders trained on all models, rankings of intentional action SVOs increase while those of unintentional SVOs decrease. However, the changes are greatest for our model. Figure 6 visualizes the output of a frozen SVO decoder on auto-corrected actions, demonstrating the auto-correct process' ability to encode completed goals in its output trajectories.

		Intentional SVO		Unintentional SVO	
Method	**Features**	**Δ R5**	**Δ Rank**	**Δ R5**	**Δ Rank**
Adversarial	Ours	**+1.6**	**+15.8M**	**-3.3**	**-9.3M**
	Kinetics [4]	+0.4	+0.3M	-0.3	-1.2M
	3D CNN [8]	+0.3	+0.1M	-0.3	-0.6M
Linearized	Ours	+0.6	+1.0M	-0.5	-1.7M

Table 3: **Evaluating Autocorrection:** We freeze the trained SVO decoder and run it on trajectories of unintentional action, before and after auto-correction. We run our algorithm based on adversarial attacks in various feature spaces as well as a linearization baseline. Using our algorithm, the frozen decoder more often predicts the ground truth goal SVO instead of the failure, indicating that our representation – crucially trained on unintentional and intentional action – captures the goals latent to video.

5.5. Analysis of Learned Representation

We finally probe the model's learned representation to analyze how trajectories are encoded. We measure Spearman's rho correlation between the activation of neurons in the output vectors $h \in \mathbb{R}^{512}$ and words in the SVO vocabulary. Each video is an observation containing neuron activations and an indicator variable for whether each word is present in ground truth. Many neurons have significant correlation, and we show the top 3 in Figure 7a, along with the 5 clips that activate them most. These neurons appear to discover common intentions in the Oops! dataset, despite being trained without any labels other than the moment of onset of unintentional action. Note that the neurons are often invariant to action class and capture shared underlying intention. We also visualize trajectories of some videos using t-SNE (Figure 7b), before and after autocorrect. Our model often adjusts trajectories from unintentional action to



(a) Top neuron-SVO correlations

(b) Trajectories in t-SNE

Figure 7: **Analyzing the Representation:** We probe the learned trajectories. (**a**) shows the neurons with highest correlation to the words in the SVO vocabulary, along with their top-5 retrieved clips. Neurons that detect intentions across a wide range of action and scene appear to emerge, despite only training with binary labels on the intentionality of action. (**b**) We show six randomly sampled video trajectories in t-SNE space, before and after auto-correct, superimposed over the embeddings for intentional and unintentional action. Visualizations suggest our approach tends to adjust unintentional action in the direction of successful, intentional action.

the region of embedding space with Kinetics videos, shown in the figure as “at goal" action.

We evaluate our model’s ability to classify action intentionality, predict goals, and automatically correct unintentional action. We train from scratch using the Oops! dataset [8] as described above.

6. Implementation Details

To train our model, we randomly sample sequences of clips $\{x_i\}_{i=1}^n$, where each clip x_i consists of $k = 16$ frames at $r = 16$ fps. In training, the length of these sequences n is randomly drawn between $[n_{lo}, n_{hi}] = [6, 10]$, so the model trains on video segments up to 10 seconds long (due to GPU memory limitations). Each clip is input to a 3D CNN f_{cnn} (we use the R2+1D-18 architecture [48]) which gives a video token embedding $\phi_i = f_{cnn}(x_i) \in \mathbb{R}^d$, where $d = 512$ is the hidden representation dimension. This is analogous to the word token embedding common in language modeling (*e.g.* [6]), where we separately learn embeddings for special tokens used to delimit input sequences.

In addition to video and special token embeddings, an additional function embeds each token’s position in the input sequence, since the Transformer’s attention-based computations do not otherwise encode input positions. This embedding is fixed to a combination of trigonometric functions as in [49], and is added to the CNN output. This allows the network to learn to generalize to unseen sequence lengths at test time, crucial to allow inference on very long videos (which would not fit in the GPU during training due to computational graph overhead). Input token embeddings are then fed to a 4-layer Transformer network with 8 attention heads per layer. For more details, please see Supplementary Material.

At test time, we feed entire videos through our model, sampled in continuous one-second intervals. If running auto-correct, we automatically split the model into two sequences at the clip where unintentional action is predicted to begin. Otherwise, we keep the entire video intact and represent it as a full trajectory.

7. Conclusion

We introduce an approach to learn about goals in video. By encoding action as a trajectory, we are able to perform several different tasks, such as decoding to categorical descriptions or manipulating the trajectory. Our experiments show that learning from failure examples, not just successful action, is crucial for learning rich visual representations of goals.

References

[1] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017. 3

[2] John Barresi and Chris Moore. Intentional relations and social understanding. *Behavioral and brain sciences*, 19(1):107–122, 1996. 1, 2

[3] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018. 3

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 2, 5, 7

[5] Livia Colle, Divide Mate, Marco Del Giudice, Chris Ashwin, and Simon Baron Cohen. Childrens understanding of intentional vs. non-intentional action. *Journal of Cognitive Science*, 8(1):39–68, 2007. 4

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 4, 5, 8

[7] Hazel Doughty, Dima Damen, and Walterio Mayol-Cuevas. Who's better? who's best? pairwise deep ranking for skill determination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6057–6066, 2018. 2

[8] Dave Epstein, Boyuan Chen, and Carl Vondrick. Oops! predicting unintentional action in video. *arXiv preprint arXiv:1911.11206*, 2019. 1, 2, 3, 4, 5, 7, 8

[9] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6202–6211, 2019. 3

[10] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017. 4

[11] Ruohan Gao, Tae-Hyun Oh, Kristen Grauman, and Lorenzo Torresani. Listen to look: Action recognition by previewing audio. *arXiv preprint arXiv:1912.04487*, 2019. 3

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2, 3, 6

[13] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018. 3

[14] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 3, 4

[15] Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. *arXiv preprint arXiv:2008.01065*, 2020. 4

[16] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. *Advances in Neural Information Processing Systems*, 33, 2020. 4

[17] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 2

[18] Jamie Hayes and George Danezis. Generating steganographic images via adversarial training. In *Advances in Neural Information Processing Systems*, pages 1954–1963, 2017. 2

[19] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (TOG)*, 26(3):4–es, 2007. 1

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[21] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 125–136. Curran Associates, Inc., 2019. 6

[22] Ali Jahanian, Lucy Chai, and Phillip Isola. On the"steerability" of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019. 3

[23] Dinesh Jayaraman and Kristen Grauman. Slow and steady feature analysis: higher order temporal coherence in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3852–3861, 2016. 4

[24] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as composition of spatio-temporal scene graphs. *arXiv preprint arXiv:1912.06992*, 2019. 2

[25] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2012. 4

[26] Linxi Jiang, Xingjun Ma, Shaoxiang Chen, James Bailey, and Yu-Gang Jiang. Black-box adversarial attacks on video recognition models. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 864–872, 2019. 3

[27] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. 2008. 2

[28] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016. 2, 6

[29] Ivan Laptev. On space-time interest points. *International journal of computer vision*, 64(2-3):107–123, 2005. 2

[30] Ang Li, Meghana Thotakuri, David A Ross, João Carreira, Alexander Vostrikov, and Andrew Zisserman. The ava-kinetics localized human actions video dataset. *arXiv preprint arXiv:2005.00214*, 2020. 3

[31] Diogo C Luvizon, David Picard, and Hedi Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5137–5146, 2018. 2

[32] Andrew N Meltzoff. Understanding the intentions of others: re-enactment of intended acts by 18-month-old children. *Developmental psychology*, 31(5):838, 1995. 2

[33] Andrew N Meltzoff, Alison Gopnik, and Betty M Repacholi. Toddlers' understanding of intentions, desires and emotions: Explorations of the dark ages. 1999. 2, 3

[34] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016. 4

[35] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016. 3

[36] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015. 3

[37] Margarita Osadchy, Julio Hernandez-Castro, Stuart Gibson, Orr Dunkelman, and Daniel Pérez-Cabo. No bot expects the deepcaptcha! introducing immutable adversarial examples, with applications to captcha generation. *IEEE Transactions on Information Forensics and Security*, 12(11):2640–2653, 2017. 3

[38] Paritosh Parmar and Brendan Tran Morris. Learning to score olympic events. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 2

[39] Hamed Pirsiavash and Deva Ramanan. Parsing videos of actions with segmental grammars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 612–619, 2014. 2

[40] Hamed Pirsiavash, Carl Vondrick, and Antonio Torralba. Assessing the quality of actions. In *European Conference on Computer Vision*, pages 556–571. Springer, 2014. 2

[41] Dongdong She, Kexin Pei, Dave Epstein, Junfeng Yang, Baishakhi Ray, and Suman Jana. Neuzz: Efficient fuzzing with neural program smoothing. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 803–817. IEEE, 2019. 3

[42] Thomas R Shultz, Diane Wells, and Mario Sarda. Development of the ability to distinguish intended actions from mistakes, reflexes, and passive movements. *British Journal of Social and Clinical Psychology*, 19(4):301–310, 1980. 2

[43] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 3

[44] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 2

[45] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Contrastive bidirectional transformer for temporal representation learning. *arXiv preprint arXiv:1906.05743*, 2019. 2

[46] Michael Tomasello. The usage-based theory of language acquisition. In *The Cambridge handbook of child language*, pages 69–87. Cambridge Univ. Press, 2009. 1

[47] Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and brain sciences*, 28(5):675–691, 2005. 1

[48] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In

Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 6450–6459, 2018. 8

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4, 8

[50] Luis Von Ahn, Manuel Blum, Nicholas J Hopper, and John Langford. Captcha: Using hard ai problems for security. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 294–311. Springer, 2003. 3

[51] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR 2011*, pages 3169–3176. IEEE, 2011. 2

[52] Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018. 4

[53] Xingxing Wei, Siyuan Liang, Ning Chen, and Xiaochun Cao. Transferable adversarial attacks for image and video object detection. *arXiv preprint arXiv:1811.12641*, 2018. 3

[54] Amanda L Woodward. Infants' grasp of others' intentions. *Current directions in psychological science*, 18(1):53–57, 2009. 1, 2

[55] Amanda L Woodward, Jessica A Sommerville, Sarah Gerson, Annette ME Henderson, and Jennifer Buresh. The emergence of intention attribution in infancy. *Psychology of learning and motivation*, 51:187–222, 2009. 2

[56] Amanda L Woodward, Jessica A Sommerville, and Jose J Guajardo. How infants make sense of intentional action. *Intentions and intentionality: Foundations of social cognition*, pages 149–169, 2001. 2

[57] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017. 1

[58] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020. 4

[59] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE international conference on computer vision*, pages 5783–5792, 2017. 3

[60] Zhe Zhou, Di Tang, Xiaofeng Wang, Weili Han, Xiangyu Liu, and Kehuan Zhang. Invisible mask: Practical attacks on face recognition with infrared. *arXiv preprint arXiv:1803.04683*, 2018. 3

[61] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 657–672, 2018. 2