What Action Caused the State Change? Learning to Reason About Action Effects

Eric Peh and Paritosh Parmar and Basura Fernando
Centre for Frontier AI Research (CFAR), Agency for Science, Technology and Research (A*STAR),
1 Fusionopolis Way, 16-16 Connexis, Singapore 138632, Republic of Singapore.

Abstract

Given a pair of initial and final states, can a neural model reason about the action responsible for state change? Can models learn the effect of an action? How can we design the task such that the model has to disentangle the effect of an action from the context of an action so that the model may truly understand the meaning (effect) of an action? What models and frameworks are suitable for this task? How can we increase the difficulty level of such a task so that model truly has to learn to reason about human actions? Are the representations learnt for the ubiquitous task of action recognition sufficient to reason about action effects? We address these questions in this paper. Codebase and data will be made publicly available.

1. Introduction

When people use something, they face two gulfs: the Gulf of Execution, where they try to figure out how it operates, and the Gulf of Evaluation, where they try to figure out what happened–Don Norman–[18]. In the gulf of execution, they ask questions such as "how do I work this?" and "what can I do?" In the gulf of evaluation, they ask questions such as "what happened?" and "is this what I wanted?". These questions define what actions humans take in a given scenario. We believe one day AI should be able to do the same.

Humans are good at capturing and understanding causal mechanisms from observations. Children have been shown to develop intuitive physics (concepts of causality, gravity, and goal-directed actions) by the age of two years. Having such capabilities in video understanding systems and robots could be very important. While current video understanding systems have shown outstanding performances on tasks like action recognition, action detection, descriptive question answering, *etc*., modeling or understanding the physical effects of actions still remains to be addressed. Actions have physical effects as consequences. For example, given a whole carrot, if we apply the action of chopping, then it would produce the effect of a chopped carrot. Now, if we replace the carrot with a potato in the starting state, and apply the action of chopping, we would have the resultant state in which the potato is chopped. As another example, given a plastic spoon, if we apply the action of bending, we would have a bent spoon. However, given a carrot, if we apply the action of bending, it would not result in the effect of chopped carrots. While it is easy for us humans to understand this effect of action (i.e. the gulf of evaluation), current video understanding systems fail at this. We hypothesize that this is due to the following reasons: 1) the concept of action-effect understanding is currently lacking; and thus the video understanding models have a limited notion of actions and their effects; 2) current models generally learn a fixed mapping between states/actions and action labels, and not much of on-the-spot/active reasoning.

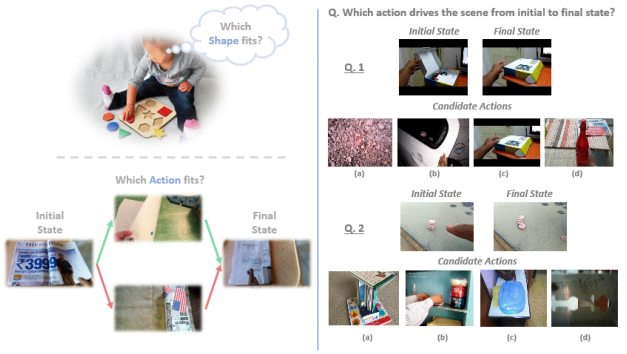

Figure 1: **Concept.** *Please view in AdobeReader to play the embedded videos for better explanation; zoom in to view better.* (Left) We have shown the inspiration for our task. (Bottom left) Dropping a coin action would not change the unfolded newspaper to a folded newspaper, while folding the paper napkin action can. (Right) We have shown two exemplary questions, from our task, alongwith candidate action options. Correct answers are (b) (not (c)) and (a) in questions 1 and 2, respectively.

Towards that end, in this paper, we introduce a new video understanding task — Action Effect Reasoning, inspired from wooden shapes puzzles [2, 1]. In wooden puzzles, we hypothesize that children learn to disentangle colors from shapes, and then use the shape information to perform matching between wooden pegs and holes. To fit the shapes into holes, children need to wriggle (picking up, moving, and twisting) the pieces into the correct position (refer to Fig. 1). Children need to wriggle pieces just the correct amount — if they under- or overdo it then the pieces may not fit. Current literature lacks this kind of disentanglement or reasoning about the degree of action movements (wriggling, as mentioned previously). By introducing the task of piecing together states and actions, our task demands disentanglement of the effect of an action (moving something) from the context of the action (type of object used to move). Our task demands the AI model to learn to reason about the *semantics of the action* that would bring the scene from the initial state to the final state.

Semantics is the study of meaning. To understand the semantics of actions, we need to understand why these actions exist in the first place. What are their roles in our universe of human life? Can the effect of actions given the state uniquely define the semantics of actions? In this vein, we define our task as follows: given an initial state, and an final-state sampled from an action sequence, we input these two states into the model (these are smaller clips of those states). Then, we also input N videos containing several actions to the model and the model has to pick the correct video containing the correct action whose effect could potentially bring us from the initial state to the final state. We call this task Action Effect Reasoning. All $N + 2$ videos may be sampled from different videos. Examples are shown in Fig 1. To solve this task, not only AI models need to understand the semantics of actions (for example the effect of an action) but also the semantic constraints. Semantic constraints are those that rely upon the meaning of the situations to control the set of possible actions [18]. For example, the box in Fig 1 (right) can be opened only from one side. To solve the task effectively, a model needs to understand the semantics of states, actions, and semantic constraints.

Process to solve the task. We hypothesize that in order to solve our task, the framework would need to do the following steps. Step 1: given just the states, imagine what action could have driven the scene from its initial to its final state; Step 2: Make an analogy between this imagined action and the action choices provided to the model. However, we force the network to perform these two steps simultaneously. Due to this, we hypothesize the framework is encouraged to learn an internal model of the underlying causal mechanisms/relationships, rather than only focusing on superficial statistical regularities.

Our contributions can be summarized as follows. We first introduce the problem of Action Effect Reasoning- which requires a model to understand both semantics of actions and semantic constraints pertaining to them. Second, we propose a model that may potentially encourage future research to solve this problem. Third, we design various test benchmarks for our newly introduced task. Our various test benchmarks demand different kinds of reasoning from video understanding models.

2. Related Work

Finn *et al*. [6] learn about physical object motion using a video prediction model that explicitly. While such a model can allow predicting the effect of the action, the emphasis is on the lesser important aspect — faithful construction in the pixel space. Nonetheless, we take inspiration from this work and develop our video generator baseline.

Hong *et al*. [12] introduce a new dataset named TRANCE, which is a synthetic dataset covering three different types of transformations, namely, Basic (or single step) transformation, Event (or multistep transformation), and View (Event transformation, but from a different viewangle). However, such an approach is generally only possible to exploit in an environment where we have a lot of control over the attributes of objects, which is generally satisfied with synthetically created datasets. Furthermore, bifurcating an action into steps is a subjective process — while one person breakdown a natural action sequence into 3 steps, another person might consider the same action sequence comprising of 6 steps. This in-built subjectivity brings an extra challenge in extrapolating such important advances in visual reasoning to the natural action domain. Our novel cross-sample can be seen as a real-world equivalent to View transformation [12] or even more challenging as many other factors are varied. Another difference between our work and TRANCE [12], CLEVR [13], CATER [7], is that they do not contain actions involving humans, while ours has been experimented specifically with human actions in the real-world.

Actions can be viewed as transformations or agents of transformations occurring in the scene or an environment. Based on this, Wang *et al*. [26], proposed to model actions as fixed class-specific transformation matrices, which when applied to a precondition would produce the effect. However, these models are not able to dynamically reason about action effects and they are not able to reason about the semantic constraints of actions. For a long time, attributes were modeled as objects. However, noting that this is ineffective to learn predictable visual prototypes for attributes, Nagarajan and Grauman [17] introduced an approach to model attributes as operators (linear transformations). While parallels can be drawn between attributes and

actions, their work was limited to static images.

One of the most similar work to ours is [25]. It presents a self-supervised model for jointly learning state-modifying actions and object states. They use causal ordering in the video as a free supervisory signal and use it to discover the changing states of objects and state-modifying actions. While this is one of the very few attempts to understand the effects of actions, they do not learn the disentanglement of the effect of the action from the context. In contrast, we aim to learn the semantics of actions (the true meaning of actions) using a new task of Action Effect Reasoning. Their approach also requires learning a class-specific model for every action, whereas our model works across many classes including unseen ones.

Other interesting reasoning problems have been introduced in recent years [27, 10, 15]. However, they are orthogonal to ours.

3. Task Formulation and Approach

3.1. Task formulation

Following the long-standing literature on situation calculus [16, 22], let us denote the initial state and the final state of an action A_k by S_I and S_F, respectively. We view the actions as `operators` that take a world state S_I to S_F and can be written as $S_I \xrightarrow{A_k} S_F$. Indeed, in some literature one may call S_I the precondition state, and S_F as the effect state of an action instance A_k. Given a sampled initial state S_I and a sampled final state S_F from a particular action class (for example, from a large video action dataset), A_k, we ask the model to pick the correct video instance V^* exhibiting action A_k when presented along with another $(N-1)$ number of wrong video instances, *i.e.*, executing different actions. In total, we present the model with N number of video instances and the model has to pick the correct video instance that could potentially change the world state from S_I to S_F. Note that both S_I and S_F are presented to the model as short video clips and all N answer videos are sampled from different videos (different from S_I and S_F video clips). Let us define the answer video set by $\mathcal{V} = \{V_1, V_2, \cdots, V_N\}$. Then the task is to train a model $(\psi(;\theta_\psi))$ that can predict the correct video index (i^*) for the correct video V^*.

$$i^* = \texttt{argmax}_{\{i=1,\cdots,N\}} \psi(S_I, S_F, V_i; \theta_\psi) \tag{1}$$

In simplest terms, given the initial and final states of a scene/process, our task involves determining the state change, and based on that, determining what action could have effected that change. Essentially, the task requires the model to recognize the initial state (S_I), final-state (S_F), and deduce the correct action (A_k) that would cause the desired effect. This requires the model to understand the semantic constraints presented in the initial and final states, and find the video V^* having the correct action semantics that matches with the state change expected. We call this task as Action Effect Reasoning.

3.2. Adapting action recognition models for Action Effect Reasoning

A straightforward solution is to directly adapt existing action recognition models for Action Effect Reasoning. Given any action recognition model, $f()$, that takes videos as input and outputs a vector representation can be adapted to solve this task. We can encode the initial and final states as $f(S_I), f(S_F)$ and also the corresponding answer videos as $f(V_i)$ for all i. Then, the correct answer can be generated as follows using the cosine similarity (Sim):

$$score(V_i) = Sim(f(V_i), \frac{f(S_I) + f(S_F)}{2}) \tag{2}$$

$$i^* = \texttt{argmax}_{i=1}^{i=N} \; score(V_i) \tag{3}$$

However, the limitation of such an approach is that there is no explicit reasoning in this approach that the model can use to learn to reason about the effect of actions. In the next section, we present a model suitable for this new task. The architecture diagram of our model is shown in Fig. 2.

3.3. Our Model

Given the initial state video S_I, final state video S_F, we pass them through the network (in our case, a ResNet-3D CNN), $f(;\theta_f)$ to obtain the *state change vector* x_s as follows:

$$z_s = f(S_I); \;\; z_f = f(S_F) \tag{4}$$

$$x_s = g_s(\Delta(z_s, z_f)) \tag{5}$$

where $\Delta(,)$ is the state change operator and $g_s(;\theta_{g_s})$ is a two-layered neural network.

3.4. State change computation scheme

We evaluate several state change operators $\Delta(,)$ as listed in the following.

Concatenation. In this scheme, we concatenate the the initial and final state vectors, $\Delta(z_s, z_f) = [z_s; z_f]$.

Addition. In this scheme, we add the initial and final state vectors, $\Delta(z_s, z_f) = z_s + z_f$.

Subtraction. In this scheme, we subtract the initial state vector from the final state vector, $\Delta(z_s, z_f) = z_f - z_s$.

Multiplication. In this scheme, we multiply (element-wise) the initial and final state vectors, $\Delta(z_s, z_f) = z_f \odot z_s$.

Each answer video V_i is also encoded by the function $f(V_i;\theta_f)$ and then passed through another two-layered neural network $g_a(;\theta_{g_a})$ to obtain its representation, x_i.

$$x_i = g_a(f(V_i);\theta_{g_a}) \tag{6}$$

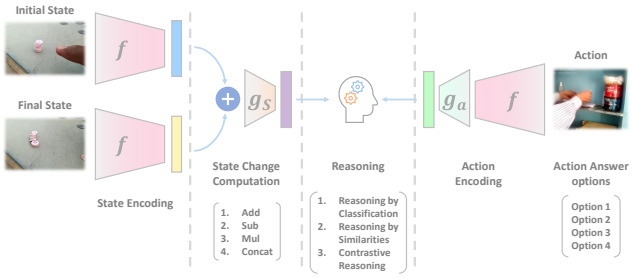

Figure 2: **Architecture diagram of our model**.

In Fig. 2, we depict the vector representing the state change (i.e. x_s) in purple color and the vector representing the answer video V_i (i.e. x_i) in green color.

3.5. Reasoning loss scheme

Now, in order to solve the task presented in Eq. 1, we investigate the following reasoning loss schemes. As we have explained earlier in Eq 5, we have the state change vector x_s and the video representation vector x_i for $i = 1 \cdots N$. As V_i , S_i and S_F are sampled from different videos, the model has to reason about x_s and x_i ($i = 1 \cdots N$) to find the correct answer in Eq. 1. In the following, we describe the reasoning loss schemes that we use in this work.

Reasoning by Classification. Given the initial and final states, classification can be used to reason about what action is likely to take the scene from its initial state to its final state. In this scheme, we first concatenate the state change and action vectors (*i.e.*, $[x_s; x_a]$). Then, process the concatenated vector using a classification head consisting of a two-layered neural network ($g(;\theta_g)$) and a linear classifier ($\psi_c(\theta_c)$). Finally, the network has to infer whether the state vectors and action vectors are correctly paired or not. Binary cross-entropy loss is used to optimize the parameters of the network where ground truth $y_i = 1$ if correct pair and $y_i = 0$ if wrong pair as follows:

$$L_R = \sum_i -(y_i log P(V_i) + (1 - y_i) log(1 - P(V_i)) \quad (7)$$

$$P(V_i) = \sigma(\psi_c(g([x_s; x_i]; \theta_g); \theta_c)) \quad (8)$$

where $\sigma()$ is the Sigmoid activation function. We optimize over all model parameters $\Theta = \{\theta_f, \theta_{g_a}, \theta_g, \theta_c\}$ during training and during inference, we use the following inference formula:

$$i^* = \texttt{argmax}_{\{i=1,\cdots,N\}} \langle x_i, x_s \rangle . \quad (9)$$

where we use the similarity between state vector x_s and video vector x_i. This is because even if the classification model can discriminate between correct and wrong options, their scores ($P(V_i)$) are not that different in inference[1]. Therefore, we resort to Eq. 9 for inference.

Reasoning by Similarities. In this scheme, we bring the state change vector x_s and the correct video vector x_* closer. We minimize the Euclidean distance between x_s and the correct video vector to make them similar. We hypothesize that such a simple method may just focus on the similarities and may not use proper reasoning to arrive at correct answer. During training we minimize the following loss:

$$L_R = ||x_s - x_*||. \quad (10)$$

During inference, we use the following inference formula.

$$i^* = \texttt{argmin}_{\{i=1,\cdots,N\}} ||x_s - x_i||. \quad (11)$$

Contrastive Reasoning. In this scheme, we bring together x_s and positive instance x_*, and force the network to push apart all negative pairs (i.e. x_s and x'_i). This enforces the network to optimize for both the similarities and the differences. For a given state change vector (x_s), a correctly paired action vector is termed as a positive (x_s, x_*), while an incorrectly paired action vector is termed as a negative (x_s, x'_i). During training, all network parameters are optimized using a suitable loss function such as triplet margin loss [23], distance ratio loss [11], or N-pair loss [24]. For example, the distance ratio loss would optimize the following:

$$L_R = log \frac{exp(||x_s - x_*||)}{\sum_{i=1}^{N-1} exp(||x_s - x'_i||)} \quad (12)$$

The inference formula remains same as in Eq. 11.

3.6. Regularization

Action recognition is a good task to learn generalizable representations, and may help our task by providing additional regularization when training our models end-to-end. Therefore, we introduce the additional auxiliary task for classifying answer videos V_i into action classes using cross-entropy loss. This auxiliary action recognition loss is denoted by L_{AR} and is defined as follows:

$$L_{AR} = -\sum log(\texttt{softmax}(g_{ar}(f(V_i; \theta_f)) \quad (13)$$

where g_{ar} is a linear classifier. Then the total loss (L) is given by

$$L = L_R + L_{AR}. \quad (14)$$

4. Benchmarks

In this section, we discuss the construction of our benchmark. We developed our benchmarks on Something-Something [8] v2 dataset. Note that we did not use any

[1] in our experiments, we got very poor results

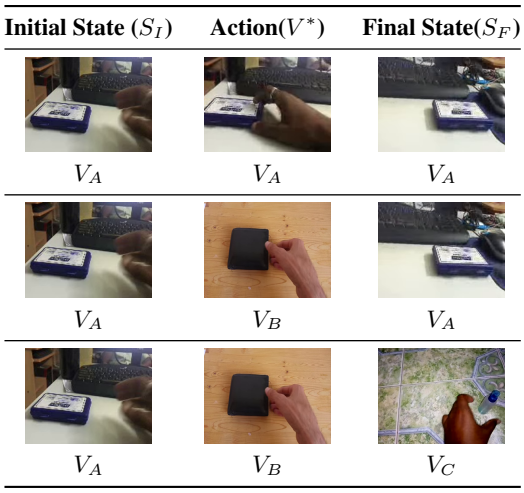

Figure 3: **Illustration of $\{V_A, V_A, V_A\}$ vs $\{V_A, V_B, V_A\}$ vs $\{V_A, V_B, V_C\}$.**

action class labels to train or test our models; we used the labels to construct the question-answer pairs. We considered the following strategies for constructing questions and answers. Different strategies allow us to generate counterfactuals of varying strengths, semantics, and semantic constraints.

4.1. Same-sample vs Cross-sample setting

First, we discuss how we generated the questions. We have the choice to draw the states clips (S_I, S_F, and the answer clips $\mathcal{V}$ from the same video or different videos. Let's term these as **same-sample** and **cross-sample** settings. In the same-sample setting S_I, S_F and the correct video answer V^* are obtained from the same action video — see Fig. 3 (top row). We denote this as $\{V_A, V_A, V_A\}$ setup to note that all video clips S_I, S_F, V^* come from same video V_A. In the cross-sample setting, there are two options. First, S_I, S_F are sampled from the same action video, while V^* is sampled from a different video, but still the same action category as S_I, S_F — see Fig. 3 (middle row). We denote this setting by $\{V_A, V_B, V_A\}$. Second, S_I, S_F, V^* are all sampled from different videos containing the same action–see Fig. 3 (bottom row) where we denote it by $\{V_A, V_B, V_C\}$ setting. Now, let's examine how they can affect learning and inference. In the same-sample setting, our model needs to compute the state change and then match the correct action option. Or even worse, in the naive same-sample setting the model may actually take a *shortcut* and solve the task by only matching the background and/or using static appearance features. However, on the other hand, in the cross-sample setting, the model needs to first compute the state change, then predict what action could have taken place that can take the scene from the initial state to the final state, and finally do the reasoning to analyze which of the given action options would suffice. Our default setting is $\{V_A, V_B, V_A\}$.

4.2. RC: Random-Choice Benchmark

Since we formulated the task as a multiple choice question-answering, we also prepared three incorrect choices, i.e. $N = 4$. To obtain incorrect choices, we simply drew video clips from other action classes randomly. We generated the training set on the fly while we used a fixed test set. We will provide the test set for future work and the codes to generate the training set. Our training set consisted of 240,000 question instances. Our fixed test set consisted of 3,000 question instances. Note that there was no overlap between training and test sets. We generated all the questions using 174 action classes while keeping a balance over action categories. These correct and incorrect choices are also known as the positives and negatives when viewed from the perspective of contrastive learning.

4.3. SA: Semantic Augmentation Benchmark

In this benchmark, we created one wrong answer option by semantically augmenting the correct video clip, *i.e.*, the answer in such a way that its meaning is changed, *i.e.*, it can no longer produce the same effect to generate hard incorrect answers. For example, for a pulling action, we applied a 'temporal reversing' augmentation, so that the resultant action would produce the opposite effect of *pushing*, instead of *pulling* while the context remains the same. The remaining two wrong options were drawn from other action classes as in our RC. We used the following temporal and spatial augmentations.

1. Temporal Reversing: can be used for actions such as pushing
2. Flipping (L-R): can be used for videos exhibiting moving an object from left to right
3. Temporally *incomplete* action sequences: in this augmentation, we used only the first 10% of the action sequence. This augmentation can be used for actions such as attaching an object to another object.
4. Static: we repeated the first action frame for the entire duration of the incorrect video clip.

Note that not all classes are suitable for such augmentations. After filtering, we retained 81 suitable action classes. For these selected action classes, we create a list of augmentations that can be applied to each of them. Our Semantic augmentations-based training set consists of 240,000 questions and our fixed test set consists of 3,000 questions.

5. Experiments

Implementation Details Architecture. We used 3D Resnet-18 [9] as the base architecture for encoder function

Reasoning scheme	Perf.
Chance	25.00
Default recognition	33.06
Reasoning by Classification	24.13
Reasoning by Similarity	25.00
Contrastive Reasoning (Margin-based Loss)	52.50
Contrastive Reasoning (Distance Ratio Loss)	53.40
Contrastive Reasoning (multinegative)	**56.00**

Table 1: **Reasoning scheme ablation study**. We use 3D-Resnet18 model.

$f(;\theta_f)$ (Sec. 3). If not mentioned otherwise, the encoders were pretrained on Something-Something-V2 dataset. We used PyTorch library [19] to implement our models and frameworks. We used Adam optimizer [14] with a learning rate of 1e-4, and a batch size of 24 questions.

Answer evaluation. If the model's answer was the same as the ground-truth, then we considered that model correctly answered the question. If all the candidate answers were equally similar, then we considered that model was not able to correctly answer the question. We used accuracy as the performance metric.

5.1. Ablation studies

Unless specified otherwise, we conducted ablation studies on our Random-Choice (RC) benchmark using the $\{V_A, V_B, V_A\}$ cross-sample setting.

Ablation of Reasoning loss schemes. In this experiment, we compared the various reasoning schemes as discussed in Sec. 3.5. We trained our models end-to-end with Aux loss with concatenation as the state change operator. The results are presented in Table 1. Since we had one correct option out of a total of four options, the chance performance is 25%. When we tested the default recognition approach presented in Sec. 3.2, we got a performance of 33.06%. Interestingly, this model did not have an explicit action effect reasoning module, yet, using the power of learned action semantics, it was able to obtain decent results. The classification-based reasoning model was found to perform almost the same as taking a guess. We observed a similar inferior performance for the similarity-based reasoning scheme as well. We hypothesize that such inferior performance stemmed from representations collapsing due to the loss functions in Eq. 7 and Eq.11. Next, we experimented with contrastive-reasoning models. In particular, we experimented with different loss functions and a different number of negatives for contrastive learning. All of our contrastive reasoning-based models outperformed all the other models. Amongst our contrastive reasoning-based models, distance ratio loss performed better than margin-based loss. Furthermore, we found that contrasting the state change with multiple negatives (use of all 3 negatives) instead of a single negative also improved the performance even further. We conclude that contrastive reasoning is more effective for Action Effect Reasoning. For generality, unless mentioned otherwise, we consider distance ratio loss-based contrastive reasoning in the following. We have also provided per class question-answer accuracy in the supplementary material.

State change operator $\Delta(,)$	Performance
Subtraction	49.40
Addition	49.63
Multiplication	49.80
Concatenation	**53.40**

Table 2: **State change computation scheme ablation study.**

Training	Testing	Performance
$\{V_A, V_A, V_A\}$		41.23
$\{V_A, V'_A, V_A\}$		36.63
$\{V_A, V''_A, V_A\}$	$\{V_A, V_B, V_A\}$	27.97
$\{V_A, V_B, V_A\}$		**53.40**
$\{V_A, V_B, V_C\}$		52.40

Table 3: **Importance of disentanglement obtained through our cross-sample training**.

Ablation of state change computation schemes. We compare the various state change computation schemes introduced in Sec. 3.4 in Table 2. We empirically observed that arithmetic operators yielded similar performances, while the concatenation operation performed better than the arithmetic operators. This may be because using raw state features and letting the network process them directly is better than processing the resultant output of arithmetic operators.

Impact of same-sample vs cross-sample. In this experiment, we compared the same-sample vs cross-sample settings presented in Sec. 4.1 in Table 3. Specifically, we trained using various settings, and tested on our Random-Choice benchmark (RC). Firstly, we found that same-sample training ($\{V_A, V_A, V_A\}$) performed significantly poorly than our cross-sample trained model ($\{V_A, V_B, V_A\}$). This may be, as we discussed Sec. 4.1, due to the model focusing on the background and/or static features co-occurring in states and action clips. A way to resolve such issues is through applying augmentations to action clips to discourage the model to latch on to irrelevant parts/factors. Noting that applying augmentations is a quite popular and effective strategy, *e.g.*, as explored in

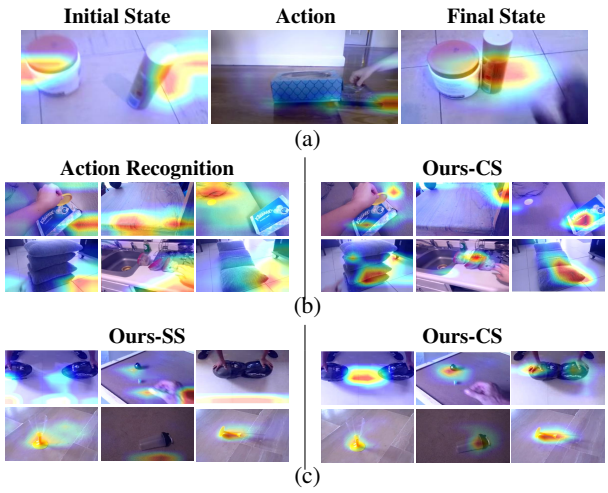

Figure 4: **Qualitative results**. *Please zoom in to view better*. (a) We probe into where our model is attending when computing state changes and analyzing the action video V. Here the action is '*putting something close to something*'. (b) We compare attention of our model vs that of action recognition model fine-tuned in SSv2. Our model focuses on state changes and driving action and effects; notice the avocado and coin being tracked. Action recognition without reasoning module seems to be doing simple matching based on texture. (c) Same sample (SS) setting is attending to background/irrelevant parts; compared to cross sample (CS) setting where model attend on actual state changes and regions of the actions.

[20, 5], we also explored it for our task. We use V'_A to denote applying random rotation augmentation; while V''_A to denote applying both random rotation and translation. However, applying these augmentations to the correct answer during training did not bring any improvements. In fact, we observed that as we incorporated more augmentations $\{V_A, V''_A, V_A\}$, the performance worsened. This indicates that regular augmentations may not help with our task and cross-sample training is an effective strategy for Action Effect Reasoning. We see that cross-sample setting is a much more effective data augmentation technique — it possibly enables our model to learn to disentangle the action and effect from other factors such as background and objects. See Fig. 4 for qualitative results. We have provided more qualitative results in the supplementary materials.

Impact of pretraining. We conducted an ablation study to determine how action recognition pretraining on various datasets affects the performance on our task. We considered Kinetics-400 [3] and Something-Something V2 (SSv2) [8] datasets for pretraining. While SSv2 has more emphasis on motion compared to Kinetics dataset, the Kinetics dataset has more action classes captured under diverse environments. We trained both models end-to-end on our task and benchmark after pretraining. We found that SSv2 pretraining helped improve the performance significantly over the Kinetics-trained model. This may be, in part, because our test benchmark is based on SSv2 dataset. We also noticed that even our Kinetics pretrained model outperformed the naive action recognition baseline trained on SSv2 (Table 1) (35.67 compared to 33.06). This study indicates that while our task is different from action recognition, action recognition does provide significant support to our task.

Encoder Pretraining	**Perf.**
Kinetics	35.67
SSv2	**53.40**

Table 4: **Effect of pretraining on performance.**

Training	**Perf.**
Default Recognition (No training)	33.06
Heads (g_s, g_a)	50.40
End-to-End (f, g_s, g_a)	41.11
End-to-End (f, g_s, g_a) + Aux	**53.40**

Table 5: **Are action recognition representations sufficient?**

Are action recognition representations sufficient? In this experiment we wanted to determine if representations learnt for the task of action recognition — representative task for video understanding in literature — are sufficient to solve our task. For this, we considered two scenarios: 1) in this case, froze the backbone and trained only the heads (g_s, g_a); 2) we trained both the backbone (f) and the heads (g_s, g_a) in an end-to-end fashion. So, essentially, while we used the action recognition representations in the first case, we learnt representations tailored for our task in the second case. Furthermore, we considered a third case, where we learnt the representations for our task in an end-to-end fashion while incorporating the auxiliary task of action recognition as discussed in Sec. 3.6 which is denoted as *Aux*. From the results summarized in Table 5, we can see that the default action recognition model performed reasonably well (33.06%). We got a significant improvement when we used action recognition features (fixed) as base representations and had state change computation $g_s()$ and reasoning functions (reasoning by contrast) operate on top of those features. However, when we trained our model without action recognition Aux loss (Eq. 14), the performance dropped to 41.11%. However, when we combined end-to-end training with Aux loss, we saw that performance improved to 53.40%. Clearly, the action recognition loss is important to solve this problem. Although, action recognition alone would not be able to solve the task of Action Effect Reasoning, with the help of state change computations, and reasoning modules, we are able to obtain better results.

5.2. Impact of counterfactual selection strategies

In this experiment, we trained and tested on our Semantic augmentations-based counterfactual (SA) benchmark. We

Training	Testing	
	RC	SA
RC	56.10	48.70
SA	52.90	54.80

Table 6: **Impact of Counterfactual selection strategies.**

Model Type	Performance
TranceNet-Adapted [12]	10.97
Actions $\sim$ Transformations [26]	25.55
PPIV [4]	25.00
Ours Video Generator	23.33
Ours	**53.40**

Table 7: **Performance comparison** with baselines.

Model Type	Performance
Humans	**81.33**
Ours	59.00

Table 8: **Performance comparison against humans**.

also noted the performance of our model trained on the Random-Choice (RC) benchmark as the baseline. Results are summarized in Table 6. the best results on RC and SA are obtained when they trained on RC and SA, respectively. RC performed reasonably well even when tested on unseen semantic augmentations. This may indicate that our models learn the mechanisms and effects of actions, which is why they are able to handle unseen semantic augmentations.

5.3. Comparison with SoTA

We compared our method with other state-of-the-art methods for visual reasoning on our task. We have provided further details on these methods in the Supplementary material. Results are presented in Table 7. We believe that the existing methods did not perform well because they either require more explicit information [12] about object states and actions; or are fixed transformations-based [26] — while our task requires more adaptive reasoning; or is less-focused on reasoning, but more focused on less important aspects like faithful video reconstruction in pixel space (video generator inspired from [6]). In comparison, our approach worked better as it is more suitable for natural videos, with a focus on adaptive reasoning.

Human baseline. We prepared a smaller subset of 100 questions and observed the performance of 3 undergraduate students on that. We compared the average human performance with our model's performance on this subset — Table 8. We observed that humans performed significantly better than our model.

Training set (%)	Rec.	Fixed Xform	Ours
50	33.06	25.44	51.20
20		26.32	50.03
10		25.09	49.40
1		25.86	43.43

Table 9: **Fewshot experiment**.

Training set (%)	CLiP[21]	Ours
90	31.60	50.53
80		50.17
70		49.83
60		49.80
50		49.70

Table 10: **Zeroshot experiment**

Fewshot training. In this experiment, we trained our model with increasingly fewer training samples, while keeping the test set constant. We observed that the performance of our model dropped, however, the drop was very minor — see Table 9. We also found that our model outperformed default recognition (Rec.) and fixed transformation models [26] under all settings.

Zeroshot testing. Here we removed some action classes from the training set and added them to the test set; so that the test set altogether comprised of unseen action classes. In particular, we chose 18 action classes (10% of all classes) for unseen/zeroshot testing. We keep the zero-shot test set size constant, while we reduced the number of training action classes (from 90% to 50%). Results are reported in Table 10. We observed that the performance of our model remained quite stable while CLiP [21] zero-shot model obtains 31.60. We hypothesize that better generalizability in fewshot/zeroshot stems from how our model learns from the proposed task; it potentially learns action effect representation agnostic to action classes, perhaps using some means of resolving the causal mechanisms. Thus, it can learn from fewer observations and generalize to unseen actions.

6. Discussion

In this paper, we have presented a new video reasoning task called Action Effect Reasoning. in which a model has to find the correct video answer for a multiple-choice video question that requires the model to learn about action effects and use reasoning. As our task requires disentangling the effect of an action from the context of the action, it is extremely challenging for current models to tackle this task. We found that action recognition models are quite useful for our task as base encoders, but in order to solve our task successfully, dedicated state change computation and reasoning modules are necessary. We hope this interesting task and the provided solution would open new research direction in action understanding and action effect reasoning. There is a significant opportunity to improve and reach human-level performance on our reasoning task.

References

[1] Learning with puzzles. 2

[2] Wooden shapes puzzle (for toddlers 18 months and older), jigsaw puzzles - amazon canada. 2

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

[4] Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. Procedure planning in instructional videos. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI*, pages 334–350. Springer, 2020. 8

[5] Ishan Dave, Rohit Gupta, Mamshad Nayeem Rizve, and Mubarak Shah. Tclr: Temporal contrastive learning for video representation. *Computer Vision and Image Understanding*, 219:103406, 2022. 7

[6] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *Advances in neural information processing systems*, 29, 2016. 2, 8

[7] Rohit Girdhar and Deva Ramanan. Cater: A diagnostic dataset for compositional actions and temporal reasoning. *arXiv preprint arXiv:1910.04744*, 2019. 2

[8] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017. 4, 7

[9] 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 (CVPR)*, pages 6546–6555, 2018. 5

[10] Jack Hessel, Jena D Hwang, Jae Sung Park, Rowan Zellers, Chandra Bhagavatula, Anna Rohrbach, Kate Saenko, and Yejin Choi. The abduction of sherlock holmes: A dataset for visual abductive reasoning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVI*, pages 558–575. Springer, 2022. 3

[11] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, October 12-14, 2015. Proceedings 3*, pages 84–92. Springer, 2015. 4

[12] Xin Hong, Yanyan Lan, Liang Pang, Jiafeng Guo, and Xueqi Cheng. Transformation driven visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6903–6912, 2021. 2, 8

[13] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017. 2

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[15] Chen Liang, Wenguan Wang, Tianfei Zhou, and Yi Yang. Visual abductive reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15565–15575, June 2022. 3

[16] John McCarthy. Situations, actions, and causal laws. Technical report, STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, 1963. 3

[17] Tushar Nagarajan and Kristen Grauman. Attributes as operators: factorizing unseen attribute-object compositions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 169–185, 2018. 2

[18] Don Norman. *The design of everyday things: Revised and expanded edition*. Basic books, 2013. 1, 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 6

[20] Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6964–6974, 2021. 7

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 8

[22] Raymond Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. *Artificial and Mathematical Theory of Computation*, 3, 1991. 3

[23] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 4

[24] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016. 4

[25] Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. Look for the change: Learning object states and state-modifying actions from untrimmed web videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13956–13966, 2022. 3

[26] Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions˜ transformations. In *Proceedings of the IEEE conference*

on Computer Vision and Pattern Recognition, pages 2658–2667, 2016. 2, 8

[27] Bo Wu, Shoubin Yu, Zhenfang Chen, Joshua B Tenenbaum, and Chuang Gan. Star: A benchmark for situated reasoning in real-world videos. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 3