SAGE: A Unified Framework for Generalizable Object State Recognition with State-Action Graph Embedding

Yuan Zang[1] **Zitian Tang**[1] **Junho Cho**[2] **Jaewook Yoo**[2] **Chen Sun**[1]
[1]Brown University [2]Samsung Electronics
https://brown-palm.github.io/SAGE

Abstract

Recognizing the physical states of objects and their transformations within videos is crucial for structured video understanding and enabling robust real-world applications, such as robotic manipulation. However, pretrained vision-language models often struggle to capture these nuanced dynamics and their temporal context, and specialized object state recognition frameworks may not generalize to unseen actions or objects. We introduce SAGE (State-Action Graph Embeddings), a novel framework that offers a unified model of physical state transitions by decomposing states into fine-grained, language-described visual concepts that are sharable across different objects and actions. SAGE initially leverages Large Language Models to construct a State-Action Graph, which is then multimodally refined using Vision-Language Models. Extensive experiments show that our method significantly outperforms baselines, generalizes effectively to unseen objects and actions in open-world settings. SAGE improves the prior state-of-the-art by as much as **14.6%** on novel state recognition with less than **5%** of its inference time.

1 Introduction

If we turn the recipe book *100 Ways of Cooking Eggs* (Filippini, 1892) into videos, can a modern computer vision algorithm successfully understand all of them? The answer to this question is not as straightforward as it might appear. On one hand, the same objects can exhibit a vast range of visual appearances and physical states (e.g., *a whole egg* versus *a scrambled egg*), especially when human actors interact with them. On the other hand, off-the-shelf detection and segmentation systems tend to focus on high-level object categories, largely overlooking their underlying physical states and dynamic transformations. We aim to develop a unified framework to jointly recognize object physical states and their temporal evolutions from visual cues, by learning from unlabeled instructional videos. We believe that understanding object states offers a powerful, object-centric abstraction for modeling world dynamics. This understanding is essential for structured video comprehension of objects, actions, and skills, and for enabling robots to perceive, predict, and plan when interacting with the physical world, thus providing a promising pathway for robots to learn from human behaviors.

A natural initial approach to achieve fine-grained understanding of object states might be to leverage vision-language models (VLMs), which provide detailed language descriptions for visual data from which object states could potentially be extracted. However, Newman et al. (2024) recently demonstrated that a naive application of state-of-the-art VLMs often fails to adequately recognize object physical states. Alternatively, prior work has explored the use of instructional videos, which are rich in object state transformations, to enhance model training. However, these approaches often model object states as discrete categories (Souček et al., 2024), or train specialist models conditioned on actions (Souček et al., 2022; Xue et al., 2024), making it challenging for them to generalize to novel states belonging to unseen objects or actions. This highlights a critical need for models that can learn a more flexible and generalizable representation of object states and their dynamics.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

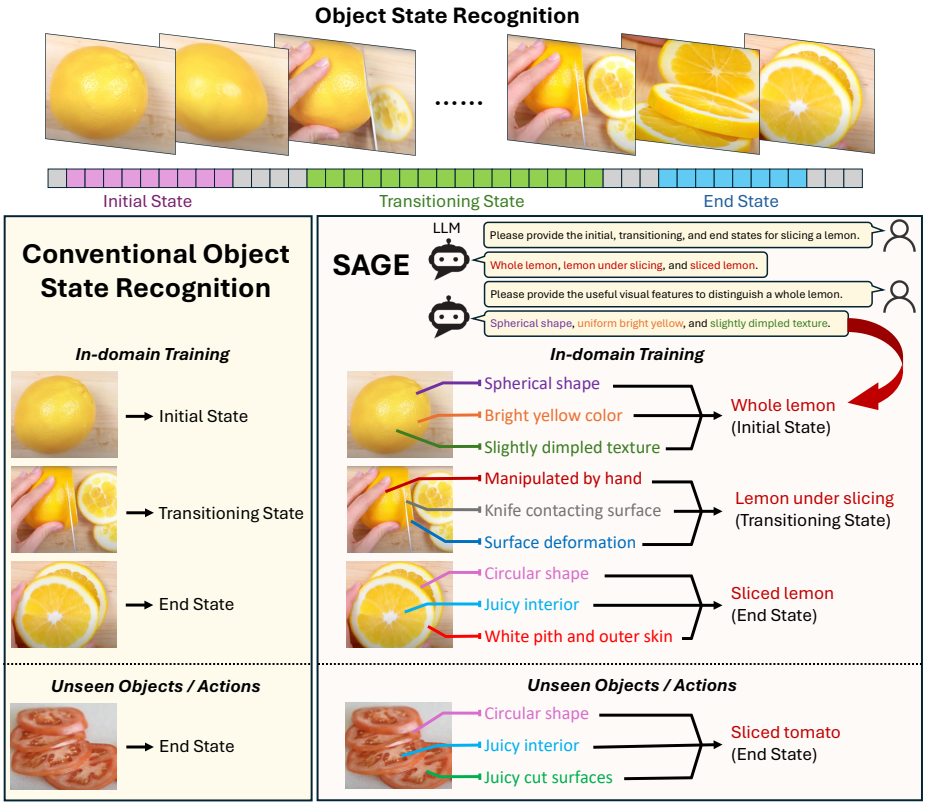

Figure 1: Illustration of state-action graph embedding (SAGE) construction and its application for object state recognition: Given an action, we ask LLM to describe the initial, transitioning, end states and their associated visual concepts. Whereas prior work recognizes states for specific actions, we recognize object states and actions through visual concepts in SAGE. It enables our model to generalize to unseen objects / actions which share similar visual concepts with known ones.

We introduce SAGE (**S**tate-**A**ction **G**raph **E**mbeddings), a framework that leverages multimodal pre-trained knowledge, as in VLMs, yet is able to learn effectively from unlabeled videos, a strategy adopted by aforementioned work on object state recognition. Our key inspiration is that in order to achieve generalizable object state recognition, the model needs to be specific about how a physical concept is rendered visually. As illustrated in Figure 1, SAGE decomposes an object state into a collection of fine-grained visual concepts with language descriptions. Some of them (e.g., *juicy interior*) are shared across objects and actions, facilitating *generalization*, others (e.g., *white pith*) are unique, capturing fine-grained *nuances* of various physical states for the same objects. In SAGE, individually, each action node (a verb-object pair) is connected to three types of state nodes: initial, transitioning, and end states. Each of these state nodes is, in turn, connected to a set of visual concept nodes that describe it. When considered together, visual concept nodes shared by different actions become connected, forming a comprehensive graph of visual concepts, states, and actions. We embed each concept node with multimodal knowledge from VLMs, where visually similar concepts are embedded nearby, even when some of them are unseen during training. An action node is represented by the direction from the initial state to end state in embedding space. We construct the initial SAGE graph using a pre-trained Large Language Model (LLM), allowing nodes for novel actions or objects to be added automatically. We then refine this graph based on multimodal information from a VLM (*e.g.*, assessing if a concept is visually recognizable) and by prioritizing concepts that are shared across a greater number of actions. Once constructed, we train a video transformer model to predict the text embeddings of these visual concepts from video frames, which are subsequently decoded into a sequence of object state and optionally action predictions.

We conduct comprehensive experiments to evaluate our approach on existing object state recognition benchmarks. To make our observations scientifically rigorous, we carefully re-implement the baseline approaches using the same pre-trained vision encoder, and also compare with their reported results

for reference. We evaluate our method on ChangeIt and HowToChange benchmarks with known and novel objects. To further evaluate the model generalizability, we introduce a more challenging setup where both the action and the object in the video are unseen during training. We perform extensive ablation studies showing the contributions of individual design choices in SAGE. Our method demonstrates strong object state recognition performance, especially when generalized to novel objects and actions. Notably, SAGE outperforms the prior state-of-the-art (Xue et al., 2024) across all benchmarks, yielding as much as 14.6% relative precision improvement on state recognition for objects and actions unseen during training, and requiring less than 5% of its inference time when action is not provided during evaluation. Our project website is `https://brown-palm.github.io/SAGE`.

2 Related Work

Object states are defined as the physical and functional properties of objects (Liu et al., 2017; Newman et al., 2024). Understanding them allows models to capture the compositionality of objects and their attributes (Misra et al., 2017; Isola et al., 2015; Purushwalkam et al., 2019). Recognizing and localizing object states is essential for video understanding, as objects tend to exhibit an even broader range of state variations (Filippini, 1892), and actions can often be represented as state transformations (Wang et al., 2016; Fathi and Rehg, 2013; Alayrac et al., 2017). Additionally, object states provide valuable cues for skill determination (Doughty et al., 2018) and goal completion (Deng et al., 2020), which are crucial for real-world tasks such as robotic manipulation (Gao et al., 2024).

Prior work has introduced foundation models (Radford et al., 2021; Alayrac et al., 2022; Jia et al., 2021; Wang et al., 2022; Xu et al., 2021) for vision-language understanding, pre-trained on large-scale image/video and caption datasets to learn a unified representation of visual features and textual information. These models have demonstrated remarkable performance across various visual recognition and reasoning tasks. However, they struggle with recognizing object states in images (Newman et al., 2024) and videos (Souček et al., 2022; Xue et al., 2024), as their training objectives often neglect object state transformations. To recognize object states in videos, prior work (Souček et al., 2022, 2024; Xue et al., 2024) has proposed to train classifiers to predict the object states for each frame. However, these methods usually require knowing the action or object information during training and struggle with generalizing to novel actions or objects due to lacking unified representation of object states. We solve these issues by proposing State-Action Graph Embeddings to jointly represent object states and actions via visual concepts.

Visual concepts which represent the primitive features (*e.g.*, colors) of objects have been widely utilized in visual computing. The visual concepts can enable visual models generalize compositionally (Farhadi et al., 2009; Nagarajan and Grauman, 2018; Stein et al., 2024) and enhance their interpretability (Koh et al., 2020; Espinosa Zarlenga et al., 2022). Previous research (Menon and Vondrick, 2022; Pratt et al., 2023; Zang et al., 2025) demonstrates that pre-trained VLMs, such as CLIP (Radford et al., 2021), can learn visual concepts and perform zero-shot recognition based on them. The concepts can be discovered by pre-trained LLMs (Yang et al., 2023; Zang et al., 2025). In this work, we explore how visual concepts can be leveraged to represent object states.

3 Method

As illustrated in Figure 2, the input to our framework is a sequence of uniformly sampled video frames encoded by a pre-trained, frozen vision encoder. The action (a verb-object pair) that causes the object state transition can be provided as input (as used by prior work), or otherwise predicted by our framework. The outputs are categorical predictions for all frames, each of which belongs to one of initial state, transitioning state, end state, and background. Unless otherwise specified, we follow the standard setup and assume that each video contains a single action, which can be relaxed when temporal action localization is applied as a pre-processing step.

3.1 Base Model

We first contextualize the encoded visual embeddings v_t from all video frames by first linearly projecting them to obtain h_t, which are fed to a temporal Transformer to produce the output embedding Z_t. An optional token h_{CLS} is reserved for action prediction and transformed into Z_{CLS}. All outputs are projected into the discrete state / action spaces, where cross-entropy losses are computed on

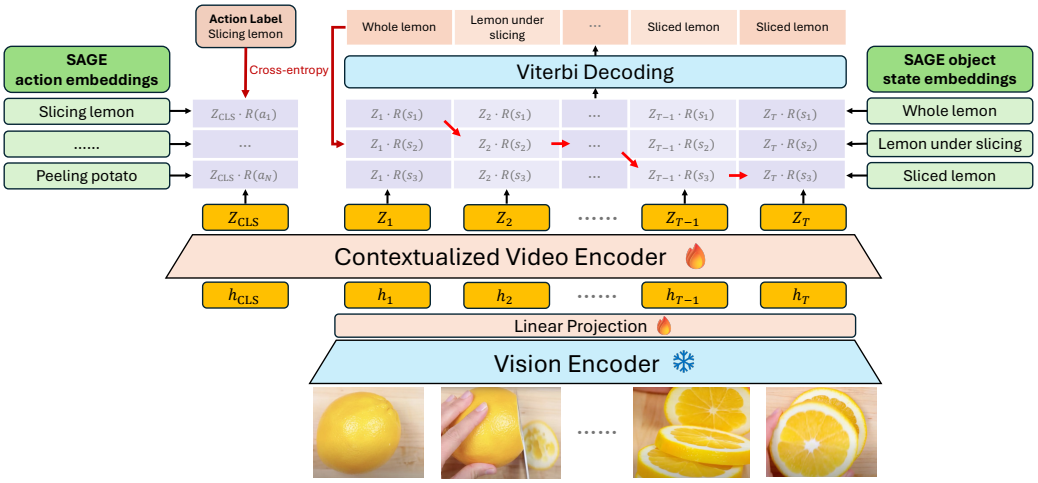

Figure 2: Overview of the training pipeline: Video frames are first encoded individually with a pre-trained and frozen vision encoder, which are then contextualized with a temporal Transformer. We decode the object states by measuring the cosine similarities of the predicted embedding Z_t and the possible state or action embeddings from SAGE.

all frames and the action prediction for model training. Frames that deemed as not containing the object of interest by a VLM are filtered out as background. The overall training objective of our model is a weighted sum of the object state recognition loss and action recognition loss across all non-background frames, *i.e.*, $\mathcal{L} = \mathcal{L}_{\text{state}} + \alpha\mathcal{L}_{\text{act}}$.

Specialist versus Generalist: As illustrated in the left side of Figure 1, a specialist model is trained to predict the object states for specific actions or objects, hence not generalizable to novel actions by design. A user needs to specify the input action to select which specialist model to use. A generalist, on the other hand, is trained to predict all object states, even for those unseen during training. This may be implemented by treating initial/transitioning/end states for all objects equivalently, which ignores the significant inter-class variations; also alternatively, by dynamically constructing the output projections (i.e., by concatenating template embeddings of all possible object states) given the action information, which is adopted by our proposed framework.

Learning from Unlabeled Videos: While each video in the training dataset is paired with a video-level action label, there is a lack of frame-level annotation necessary for calculating the losses. Following the standard convention, we estimate the noisy "pseudo" object state labels by computing the cosine similarity between frame visual embeddings and state text embeddings using a pre-trained, frozen VLM. We further propose to refine the pseudo labels by applying a temporal constraint (i.e., initial state $\rightarrow$ transitioning state $\rightarrow$ final state), which is implemented as a constrained Viterbi decoding algorithm (Viterbi, 1967). The decoding process is implemented through dynamic programming with restricted state transitions.

3.2 State-Action Graph Embeddings

We introduce State-Action Graph Embeddings (SAGE) to dynamically construct the state and action embeddings used by the base model to decode object states and actions for generalizable recognition. SAGE first decomposes an action into a state transformations, it then describes each state with a collection of visual concepts, forming a tree structure. Intuitively, we want to advocate the selection of visually distinctive concepts, which capture the nuance of fine-grained object states visually, as well as concepts shared by multiple object states, which facilitate the generalization towards unseen objects. The concept nodes shared by multiple states are merged, forming a densely-connected graph. In addition, both the state nodes and action nodes should be embedded, so that they can be used for the base model for prediction.

Graph Construction: To construct a vocabulary of object states, we query an LLM to identify the initial, transitioning, and end states associated with each action in the dataset. For example, the three object states involved in "slicing lemon" include "whole lemon", "slicing lemon", and "sliced lemon".

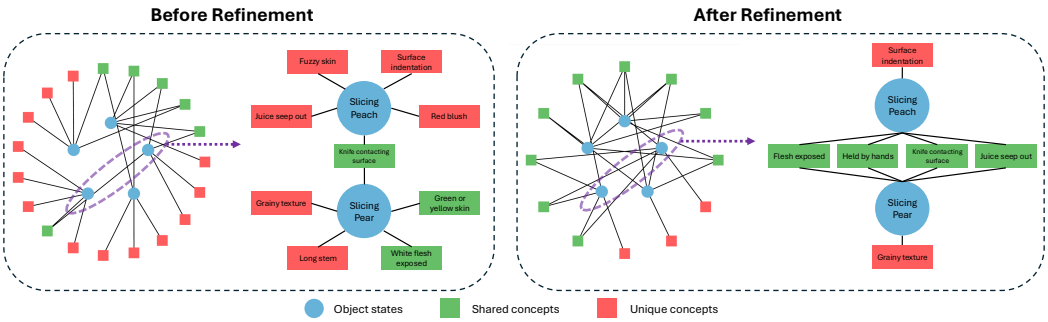

Figure 3: The graph structure of SAGE before and after refinement. We zoom in on a subregion of the graph for demonstration. After refinement, the graph includes more shared visual concepts.

To extract the visual concepts of the object states, we leverage the LLM again to generate descriptive attributes for each object state. Given an object state s and a prompt such as "What are the useful visual features to distinguish a {state name} {object name}?", along with in-context examples, the LLM generates a set of visual concepts $\mathcal{C}_s$ that characterize the object state.

Node Embedding: Instead of directly encoding the object state names and action names, we first derive the embeddings of all the visual concepts and then leverage them to compute embeddings for the object states and actions. Ideally, visually similar concepts should be embedded nearby in the text space, further enhancing the generalizability of our framework. As such, we first encode each visual concept as a distributed embedding using the text encoder E_{text} of a pre-trained VLM. We then represent an object state as the average of its visual concept embeddings, *i.e.*, $R(s) = \frac{1}{|\mathcal{C}_s|}\sum_{c\in\mathcal{C}_s} E_{text}(c)$. Since visual concepts are shared across different object states, this formulation provides a unified embedding space, enabling our model to generalize across various object state transformations, including unseen states. Inspired by prior work on relation encoding (Bordes et al., 2013) and state-based action recognition (Wang et al., 2016), we compute the action embeddings as the difference between the embeddings of its end and initial object states, which naturally accounts for object-specific variations, as the same action might have different visual features for different objects (e.g. "boiling egg" and "boiling pasta").

By integrating object states and actions within the state-action graph, our approach captures both their visual attributes and dynamic transformations. The proposed SAGE embeddings enable a structured representation of actions while preserving the compositional nature of object states. This facilitates a more generalizable understanding of object state transitions across different actions.

Multimodal Graph Refinement: We propose refining the graph structure to incorporate visual concepts that are reliably recognized by the model to enhance object state recognition accuracy, as well as concepts that are commonly shared across different object states to improve generalizability. Given an object state, we first generate an over-complete concept list with an LLM. We then rank these concepts based on their VL similarity scores with the video frames containing the object state. For a concept c, the score is calculated as $\frac{1}{|F|}\sum_{f\in\mathcal{F}} \text{cos\_sim}(E_{\text{text}}(c), Z_f)$, where F is the set of video frames containing the object state. We select the top-ranked ones as they are most reliably recognized by the model. To further promote generalizability, we prioritize concepts that are shared across multiple object states. We enforce that at least half of the selected concepts are shared. Specifically, to select k concepts, we first select the top-ranked shared concepts until $\lceil \frac{k}{2} \rceil$ are selected and then select remaining top-ranked concepts. In practice, we select the top 5 concepts for each object state and ensure that at least 3 of them are shared. We then construct a new graph with the selected visual concepts. An illustration of SAGE before and after refinement is shown in Figure 3.

3.3 Inference

We follow a two-step inference which first predicts a video-level action label, and then decode the object states with SAGE graph. The first step is skipped when the action is provided as input.

Action Recognition: We first identify the object of interest in the video. Each object o is represented by the average embedding of all its associated states in SAGE. We then compute the similarity

between the object embedding and the embeddings of all video frames. The object of interest is selected as the one with the highest cumulative similarity to the video frames. Next, we predict the action in the video. For all actions potentially associated with the identified object based on the SAGE knowledge graph, we compute the similarity between each action embedding and the video-level embedding Z_{CLS}, and select the action a^* with the highest similarity score.

Object State Recognition: For each frame t, we compute the similarities between its representation Z_t and the state embeddings $R(s)$, and then a probability distribution over object states after softmax. The only difference is that we now narrow our scope of object states to only the initial, transitioning, and end states of the action a^* instead of all the object states in SAGE. We then apply the Viterbi decoding over the state predictions to predict the object states following the temporal order in SAGE.

4 Experiment

4.1 Experimental Setup

Datasets: We evaluate our method on two object state recognition benchmarks, ChangeIt (Souček et al., 2022) and HowtoChange (Xue et al., 2024), which consist of videos depicting single actions. The task is to predict the physical state of the object of interest in every frame given the action. While SAGE can support state recognition from multi-action videos, by replacing the video-level action classification step with temporal action localization, to the best of our knowledge there is no publicly available benchmark with multiple objects and actions for evaluation purposes.

Baselines: We compare our method with LookForTheChange (LFC) (Souček et al., 2022), Multi-TaskChange (MTC) (Souček et al., 2024) and VidOSC (Xue et al., 2024), which are the state-of-the-art methods for object state recognition in single-action videos. LFC and VidOSC train separate specialized models for different objects / actions. MTC trains a unified model for all objects and actions given a known vocabulary. We also evaluate zero-shot VLMs, including CLIP (Radford et al., 2021), VideoCLIP (Xu et al., 2021), and InternVideo (Wang et al., 2022) for object state recognition.

To ensure a fair comparison, we use the same pre-trained VLM, CLIP ViT-L-14 (Radford et al., 2021), which is fine-tuned with the pseudo labels and videos from HowtoChange, as the vision backbone and pseudo-label generator for both the baselines and our method. The fine-tuning helps our method better recognize the objects and actions from videos, by utilizing noisy, automatically generated supervision from the speech modality, which we hope can improve the generalization performance at object state level. To understand the performance benefits when a better video-language model is used, we also train our model with VideoCLIP and compare it with the reported performance of the baselines (Xue et al., 2024; Souček et al., 2022, 2024) in Section 4.5.

Evaluation Metrics: Following Souček et al. (2022, 2024), we evaluate the methods in Precision@1 on the ChangeIt dataset. In this dataset, state precision refers to the precision of initial and end states, while action precision corresponds to the precision of transitioning states. We rename the metric as Trans. Pre@1 to avoid confusion. On the HowToChange dataset, we follow Xue et al. (2024) and evaluate the F-1 score, Precision, and Precision@1. See the Appendix for their detailed definition. While the object and action ground truth labels are available in the test set, they are optionally provided to the model according to the evaluation setup. The use of ground truth action and object annotation as privileged information is indicated in Table 1. When not provided, our method infers them directly from videos as described in Section 3.3. For the remaining tables, we follow the protocol from Xue et al. (2024) where action information is provided but the object label is not.

Implementation: We sample video frames at 1 FPS as in baselines (Xue et al., 2024; Souček et al., 2022, 2024). Frame embeddings are extracted by the vision encoder of the fine-tuned CLIP ViT-L-14 (Radford et al., 2021) (except in Section 4.5) and projected by a linear layer. We then process them using a three-layer Transformer (512-dimensional hidden states, four attention heads) followed by another linear projection layer to obtain frame representations. The action loss weight α is 0.1. We train the models using the AdamW optimizer with $1\text{e}{-4}$ learning rate and $1\text{e}{-4}$ weight decay. We use a batch size of 32 on ChangeIt and 128 on HowToChange. We train our model for 10 epochs on each dataset. These hyper-parameters are optimized according to validation performance. In the state-action graph construction, we use OpenAI `GPT-4o-mini-2024-07-18` as the LLM to generate 5 visual concepts for each object state. We also experiment with an open-weight model, Qwen3-32B, and find the performance differences are negligible compared to GPT-4o.

Table 1: Known object state recognition performance on ChangeIt and HowToChange. SAGE outperforms the baselines across different settings. The unified versions of baseline models suffer substantial performance degradation, whereas our model not only mitigates this issue but even surpasses specialized baseline models. *: Unified version implemented by us.

Methods	Privileged Info		Unified	ChangeIt		HowToChange		
	Action	Object	Model	State Pre@1	Trans. Pre@1	F1	Pre	Pre@1
CLIP (Radford et al., 2021)	✓	✓	✓	0.30	0.63	0.27	0.27	0.48
VideoCLIP (Xu et al., 2021)	✓	✓	✓	0.33	0.59	0.37	0.40	0.48
InternVideo (Wang et al., 2022)	✓	✓	✓	0.27	0.57	0.30	0.31	0.47
LFC (Souček et al., 2022)	✓	✓	✗	0.30	0.63	0.30	0.30	0.36
SAGE (ours)	✓	✓	✓	**0.57**	**0.85**	**0.39**	**0.45**	**0.58**
VidOSC (Xue et al., 2024)	✓	✗	✗	0.52	0.83	0.37	0.40	0.53
SAGE (ours)	✓	✗	✓	**0.53**	0.83	0.37	**0.42**	**0.55**
LFC*	✗	✗	✓	0.25	0.52	0.24	0.26	0.31
VidOSC*	✗	✗	✓	0.41	0.67	0.29	0.32	0.42
MTC (Souček et al., 2024)	✗	✗	✓	0.47	0.75	0.32	0.35	0.45
SAGE (ours)	✗	✗	✓	**0.51**	**0.81**	**0.34**	**0.39**	**0.52**

Table 2: Open-world evaluation results on ChangeIt and HowtoChange. The MTC method cannot generalize to novel actions or objects because its classification heads are fixed for the states of known objects. SAGE shows robust performance in the open-world setting, while baseline models degrade significantly on unseen actions or objects. *: Results from the best specialized models.

(a) Evaluation with known actions and novel objects.

Methods	ChangeIt Novel Obj		HowtoChange Novel Obj		
	State Pre@1	Trans. Pre@1	F1	Pre	Pre@1
LFC*	0.25	0.54	0.27	0.27	0.32
VidOSC	0.43	0.71	0.32	0.35	0.48
SAGE (ours)	**0.49**	**0.78**	**0.34**	**0.39**	**0.50**

(b) Evaluation with novel actions and novel objects.

Methods	ChangeIt Novel Obj & Act		HowtoChange Novel Obj & Act		
	State Pre@1	Trans. Pre@1	F1	Pre	Pre@1
LFC*	0.21	0.48	0.23	0.22	0.27
VidOSC*	0.27	0.59	0.25	0.28	0.37
SAGE (ours)	**0.45**	**0.70**	**0.31**	**0.35**	**0.45**

Model Training and Time: We train the model with $8\times$ NVIDIA V100 GPUs. It takes 30 minutes to extract visual embeddings for HowToChange and 3.5 hours for ChangeIt. The training takes about 30 minutes for HowToChange and 8 hours for ChangeIt.

4.2 Evaluation on Object State Recognition

We evaluate our method and baselines on both known and novel objects and actions. Prior work (Souček et al., 2022, 2024; Xue et al., 2024) trains separate specialized models for different objects or actions, and thus struggles to generalize to novel objects and actions. In this work, SAGE enables us to train a unified model for all objects and actions and generalize to novel objects and actions.

Known Objects and Actions: As shown in Table 1, naively extending baseline methods into unified models results in significant performance degradation. We reimplement the unified versions of the baselines by putting state prediction heads for all known actions and objects and on a shared backbone. With SAGE, our unified model with comparable number of parameters significantly outperforms the generalist baselines, and even outperforms the specialists. More importantly, while baseline models require the privileged information of actions and objects as inputs, our model can make precise object state recognition without knowing the action and object in the video during evaluation.

Novel Objects and Actions: We evaluate our model with novel objects as explored in prior work, and propose a more challenging setup where both the action and the object in the video are novel. Among the baseline methods, LFC and MTC cannot generalize to novel objects and actions because they rely on fixed-dimension classification heads trained specifically for seen object state transformations. Similarly, VidOSC cannot generalize to novel actions. To estimate their generalization ability and make a comparison, we follow Xue et al. (2024) and measure their performance upper bounds by enumerating all of their specialist models and pick the one with the best performance using the ground truth labels. Tables 2 reports the model performance on novel objects and novel actions. Our model maintains comparable performance on unseen objects and actions as it does on seen ones, whereas baseline models suffer significant performance drops.

Table 3: Parameter numbers and inference runtime of different methods on the HowtoChange dataset. When actions and objects are unknown, the specialized methods such as LFC and VidOSC must run all expert models and select the best one, which significantly increases the computational cost.

Methods	#Params per Model	#Models	Runtime w/ Known Actions and Objects (s)	Runtime w/ Unknown Actions and Objects (s)
LFC (Souček et al., 2022)	4.2M	409	18.2	7278.4
VidOSC (Xue et al., 2024)	10.5M	20	33.8	612.7
MTC (Souček et al., 2024)	8.1M	1	24.5	24.5
SAGE (Ours)	10.9M	1	29.7	29.7

Table 4: Comparison of SAGE before and after refinement on HowToChange.

Graph	# Concepts	Known			Novel		
		F1	Pre	Pre@1	F1	Pre	Pre@1
SAGE (before refinement)	5	0.37	0.42	0.55	0.31	0.35	0.45
SAGE (before refinement)	15	0.36	0.42	0.54	0.30	0.34	0.44
SAGE (after refinement)	5	**0.38**	**0.42**	**0.58**	**0.33**	**0.39**	**0.50**

Efficiency Analysis: We compare the parameter numbers and runtimes on the HowToChange dataset of our model and baselines. We compare their inference time on $8\times$ NVIDIA V100 GPUs. As shwon in Table 3, when the privileged information of actions and objects is not provided, our method show significant advantages in inference efficiency. Unlike specialized baselines that must run all specialized models and select the best one (following Xue et al. (2024)), our approach uses a single general model for all actions and objects, resulting in substantially reduced computational cost.

These findings demonstrate that our method enables the training of foundation models for object state recognition, paving the way for more scalable and generalizable solutions.

4.3 SAGE Graph Refinement

We evaluate our SAGE refinement strategy for both known and novel object states. To validate the effectiveness of our concept selection method, we compare the performance of original SAGE, SAGE with over-complete concepts and the refined SAGE with selected concepts. As shown in Table 4, the proposed refinement significantly improve the generalizability of SAGE.

In Figure 3, we illustrate the local SAGE graph structures of five object states before and after refinement. The refined graph incorporates more visual concepts that are shared across different object states, enabling the model to recognize novel object states by leveraging common concepts learned from known object states. Especially, before graph refinement, concepts that should be shared by multiple states might not be correctly assigned to these states (*e.g.*, "juice seep out" in Figure 3).

Table 5: Ablation studies on different technical designs in our method. We report the Pre@1 scores and their performance differences ($\triangle$) compared to the complete model.

Methods	ChangeIt			HowToChange		
	% Pre@1 ($\triangle$)			% Pre@1 ($\triangle$)		
	Seen	Novel Obj	Novel Obj & Act	Seen	Novel Obj	Novel Obj & Act
W/o Textual Representation	49.8 (-11.6)	N/A	N/A	46.4 (-8.4)	N/A	N/A
W/o Visual Descriptions	56.8 (-4.6)	50.3 (-6.8)	48.4 (-7.0)	50.7 (-4.1)	42.9 (-6.8)	38.8 (-6.5)
W/o Joint Training	62.3 (+0.9)	54.6 (-2.5)	47.9 (-4.1)	55.5 (+0.7)	48.2 (-1.5)	41.9 (-3.4)
W/o Dynamic State Locating	58.9 (-2.5)	53.7 (-3.4)	48.8 (-3.2)	51.8 (-3.0)	45.4 (-4.3)	41.3 (-4.0)

4.4 Ablation Studies

We analyze the effectiveness of our proposed method through ablation studies by removing each component of SAGE at a time. The results are shown in Table 5.

State Text Embedding: We remove the text embeddings of object states by not using the cosine similarity between language and vision representations for state recognition. Instead, we treat object states as discrete categories. The results indicate that removing textual representations significantly degrades model performance and disables it to work on unseen objects and actions.

Table 6: Evaluating SAGE with different VLMs. We observe that our approach is reasonably robust with respect to the choice of vision and text encoders.

	ChangeIt		HowtoChange		
	State Pre@1	Action Pre@1	F1	Pre	Pre@1
SAGE + CLIP	0.51	0.81	0.34	0.39	0.52
SAGE + SigLIP	0.53	0.82	0.34	0.42	0.56
SAGE + MetaCLIP	0.54	0.82	0.36	0.43	0.56

Table 7: Comparison between our model with VideoCLIP as pre-trained VLM and the reported results of baselines. Our model achieves slightly better performance than all baselines on seen objects and significantly outperform all baselines on novel objects.

Methods	ChangeIt		ChangeIt (Open-world)				HowToChange					
	State Pre@1	Action Pre@1	State Pre@1		Trans. Pre@1		F1 (%)		Precision (%)		Pre@1 (%)	
			seen	novel	seen	novel	seen	novel	seen	novel	seen	novel
LFC Souček et al. (2022)	0.35	0.68	0.36	0.25	0.77	0.68	30.3	28.7	32.5	30.0	37.2	36.1
MTC (Souček et al., 2024)	0.49	0.80	0.41	0.22	0.72	0.62	33.9	29.9	38.5	34.1	43.1	38.8
VidOSC (Xue et al., 2024)	0.57	0.84	0.56	0.48	0.89	0.82	**46.4**	43.1	46.6	43.7	60.7	58.2
SAGE (Ours)	**0.60**	**0.89**	**0.59**	**0.55**	0.89	**0.87**	46.4	**44.7**	**47.5**	**46.3**	**63.6**	**61.2**

Visual Concept Descriptions: We replace the visual concept descriptions with object state names to obtain the state embeddings, and keep the pseudo labels consistent for fair comparison. The results show that removing visual concepts leads to the largest performance drop on unseen objects and actions. This suggests that visual descriptions provide crucial reasoning cues for recognizing object states, especially when generalizing to novel objects and actions.

Jointly Training with Action Recognition: We remove the action recognition objective during training. Although this objective does not improve seen state recognition, it enhances the generalizability to unseen objects and actions. This is because unseen object states and actions may share similar relationships to seen ones, which could be learned by joint training with action recognition.

Viterbi Temporal Decoding: Instead of training the model with pseudo labels decoded with temporal constraints, we train it with pseudo labels generated by CLIP as in Xue et al. (2024). The results show that removing Viterbi decoding in pseudo label preparation leads to a significant drop in performance, suggesting that temporal constraints helps improve recognition, particularly for generalization.

Vision-Language Models: Finally, we explore the use of different vision-language models to showcase the robustness of our method. We adopt the visual and text encoders from SigLIP (Zhai et al., 2023), which enhances CLIP's training objective, and from MetaCLIP (Xu et al., 2024), which refines CLIP's training data quality. We evaluate in the most challenging setting where both the action and object information are unknown. In Table 6, we observe moderate improvements when SigLIP and MetaCLIP are used, indicating the image encoder is not the main performance bottleneck.

4.5 Comparison with Reported SOTA Results

For thorough comparison, we train our model by preplacing CLIP with VideoCLIP adopted by prior work, and compare SAGE with the reported results in baselines (Souček et al., 2022, 2024; Xue et al., 2024). Table 7 shows that our unified model can perform consistently better than the reported state-of-the-art results from specialized models.

4.6 Qualitative Results

We demonstrate in Figure 4 the top-1 frame predictions of our model for videos with known and novel objects and states. For each video, we identify the top-1 frame for the initial, transitional, and final object states by selecting the frame with the highest embedding similarity to the corresponding state. We also display the visual concepts with the highest embedding similarity to each top-1 frame. The results suggest that our model can accurately recognize the object states for both known and novel objects and actions.

Figure 4: Examples of the top-1 frames and top-aligned visual concepts predicted by our model for the initial, transitional, and end states in the videos. In the action and object labels, text in *green* indicates known objects or actions and text in *red* indicates novel ones. In the visual concepts, the *green* concepts are shared across different object states while the *red* concepts are wrongly predicted.

5 Conclusion

In this paper, we propose a novel framework to build unified models for recognizing object states in videos. We leverage pre-trained LLMs and VLMs to build State-Action Graph Embeddings (SAGE) that decomposes object states into visual concepts. The graph structure where different objects and actions share the same visual concepts enables the model to generalize to novel objects and actions. Our model outperforms all baselines on two widely used object state transformation benchmarks, especially for open-world settings where both objects and actions are novel.

Limitations: Our approach is evaluated on ChangeIt and HowToChange benchmarks, both contain a single action in each video. Although SAGE may be naturally generalized to handle videos with multiple non-overlapping actions via temporal action localization, it cannot directly support the scenario where multiple objects are undergoing state transformations *concurrently*.

We further envision two natural directions for future work: First, despite our effort to build a unified model for diverse objects and actions in different scenarios, our model is designed solely for object state and action recognition. Integrating the success of SAGE into a vision foundation model would be ideal as recognizing object physical state is a fundamental task of visual perception. Second, it would be highly impactful to apply our approach for downstream tasks such as tracking objects that undergo state transformations, or even temporal abstraction and skill discovery for robotic applications.

Acknowledgments

We would like to thank Calvin Luo, Nate Gillman, Shijie Wang, Tian Yun, and Zilai Zeng for helpful discussions. This project was supported by Samsung. Our research was conducted using computational resources at the Center for Computation and Visualization at Brown University.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736.

Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Simon Lacoste-Julien. 2017. Joint discovery of object states and manipulation actions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2127–2136.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Xinke Deng, Yu Xiang, Arsalan Mousavian, Clemens Eppner, Timothy Bretl, and Dieter Fox. 2020. Self-supervised 6d object pose estimation for robot manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3665–3671. IEEE.

Hazel Doughty, Dima Damen, and Walterio Mayol-Cuevas. 2018. 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.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. 2022. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems*, 35:21400–21413.

Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing objects by their attributes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1778–1785. IEEE.

Alireza Fathi and James M Rehg. 2013. Modeling actions through state changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2586.

Alexander Filippini. 1892. *One Hundred Ways of Cooking Eggs*. 1. CL Webster.

Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. 2024. Physically grounded vision-language models for robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12462–12469. IEEE.

Phillip Isola, Joseph J Lim, and Edward H Adelson. 2015. Discovering states and transformations in image collections. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1383–1391.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR.

Yang Liu, Ping Wei, and Song-Chun Zhu. 2017. Jointly recognizing object fluents and tasks in egocentric videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2924–2932.

Sachit Menon and Carl Vondrick. 2022. Visual classification via description from large language models. In *The Eleventh International Conference on Learning Representations*.

Ishan Misra, Abhinav Gupta, and Martial Hebert. 2017. From red wine to red tomato: Composition with context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1792–1801.

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

Kaleb Newman, Shijie Wang, Yuan Zang, David Heffren, and Chen Sun. 2024. Do pre-trained vision-language models encode object states? *arXiv preprint arXiv:2409.10488*.

Sarah Pratt, Ian Covert, Rosanne Liu, and Ali Farhadi. 2023. What does a platypus look like? generating customized prompts for zero-shot image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15691–15701.

Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc'Aurelio Ranzato. 2019. Task-driven modular networks for zero-shot compositional learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3593–3602.

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

Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. 2022. Look for the change: Learning object states and state-modifying actions from untrimmed web videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. 2024. Multi-task learning of object states and state-modifying actions from web videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Adam Stein, Aaditya Naik, Yinjun Wu, Mayur Naik, and Eric Wong. 2024. Towards compositionality in concept learning. *arXiv preprint arXiv:2406.18534*.

Andrew Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269.

Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. 2016. Actions˜ transformations. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2658–2667.

Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, et al. 2022. Internvideo: General video foundation models via generative and discriminative learning. *arXiv preprint arXiv:2212.03191*.

Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. 2021. Videoclip: Contrastive pre-training for zero-shot video-text understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6787–6800.

Hu Xu, Saining Xie, Xiaoqing Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. 2024. Demystifying clip data. In *The Twelfth International Conference on Learning Representations*.

Zihui Xue, Kumar Ashutosh, and Kristen Grauman. 2024. Learning object state changes in videos: An open-world perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18493–18503.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. 2023. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19187–19197.

Yuan Zang, Tian Yun, Hao Tan, Trung Bui, and Chen Sun. 2025. Pre-trained vision-language models learn discoverable visual concepts. *Transactions on Machine Learning Research*.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We discuss in detail the contribution and scope of this paper in the Abstract and Introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of the work in conclusions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We disclose the implementation details including model architectures, hyper-parameters and pre-trained models we used in Section 4.1. Researchers can reproduce our results with the provided information.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We submit the code in supplementary documents with the README file describing the steps to reproduce the results. We use the public datasets for experiments. The link of these datasets are provided.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the hyperparameters and optimizers information in Section 4.1. We provide the details of the datasets in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We perform the Student T-test to evaluate the statistical significance of the main results. The results are provided in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We describe the compute resources we use for training and inference in Section 4.1 and Section 4.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform with the NeurIPS Code of Ethics. It does not include human subjects. The datasets are publicly available and do not have ethic concerns. The research does not have potential harmful consequences.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss in the introduction how the proposed method can potentially benefit real-world machine learning applications such as robotics. We do not identify any potential negative impacts the proposed method can bring.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: This paper uses pre-trained Vision-Language models and Large-Language Models. All of these models are properly cited. The datasets this paper uses are publicly available and properly cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We describe in detail how we use the LLM to construct the core component (SAGE) of this model in Section 3.2.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Evaluation Metrics

On the ChangeIt dataset, we follow Souček et al. (2022, 2024) and report State Precision@1 for initial and end states, and Transition Precision@1 for the transitioning state. Note that the transition precision was referred to as *action* precision in prior works (Souček et al., 2022, 2024). We rename it here to avoid confusion. For each of the initial/transitioning/end states, the video frame with the top one probability predicted by a model is retrieved, where the precision is calculated as the percentage of corrected retrieved frames across all videos. Since the frame sampling strategy would affect the collection of candidate frames, we follow the same 1-FPS uniform sampling strategy as prior work so that the results are comparable.

Additionally, on the HowToChange dataset, we follow Xue et al. (2024) to evaluate the F-1 score, Precision, and Precision@1. The definition of Precision@1 is the same as Precision@1 in ChangeIt , except that the initial/transitioning/end states are jointly considered in this metric. Since Precision@1 only considers the top retrieved frames, F-1 score and Precision are used, both of which are computed over all sampled videos frames. We calculate the F-1 score and Precision for each of the initial, transitioning, and end states and report their average over the three states. Similarly, we use the same frame sampling strategy since the metrics are defined with respect to all sampled frames.

B SAGE Construction Details

We construct SAGE by querying the LLM to first provide the name of initial, transitioning and end states for an action and then provide the visual concepts for each state. In practice, we use OpenAI `GPT-4o-mini-2024-07-18`[1] as the LLM and add in-context examples in the prompts to guide it. In the following, we provide an instance of the prompts for generating the states and visual concepts. We skip the in-context examples below for conciseness.

Prompt template for generating state names:

```
Q: What are the initial, transitional, and end states of a(n) {object}
during the action {action}?
```

For example:

```
Q: What are the initial, transitional, and end states of a lemon during the
action slicing lemon?
A:
- whole lemon
- lemon under slicing
- sliced lemon
```

Prompt for generating visual concepts:

```
Q: What are the visual features for distinguishing a(n) {initial state
name}, a(n) {transitional state name} and a(n) {end state name}?
```

For example:

```
Q: What are the visual features for distinguishing a whole lemon, a lemon
under slicing and a sliced lemon?
A:
Whole lemon:
- spherical shape
- bright yellow color
- slightly dimpled texture
- smooth surface
- evenly distributed color
Lemon during slicing:
- manipulated by hands
```

[1] https://platform.openai.com/docs/models/gpt-4o-mini

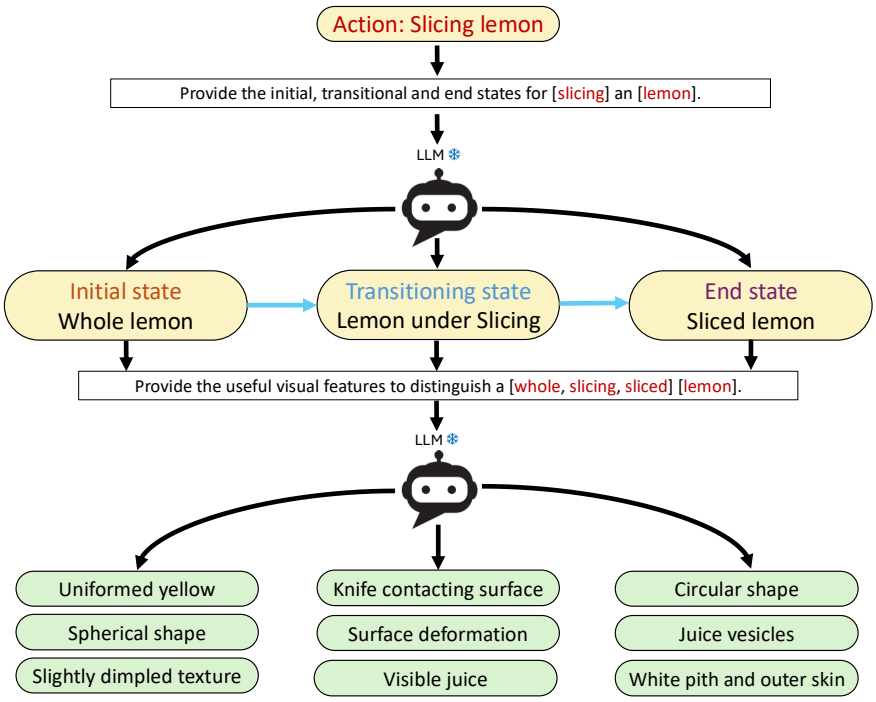

Figure A1: Illustration of how the state-action graph is constructed for a single action *slicing lemon* with a frozen LLM. See details in Section B.

```
- knife contacting surface
- surface deformation
- juice beginning to escape
- distinct cut line forming
Lemon after slicing:
- circular shape
- juicy interior
- white pith and outer skin
- exposed flesh glistening
- broken surface texture
```

We can obtain the object state names and visual concepts by parsing the LLM generated answers and collecting the visual concepts into a list. We observe that the answers consistently follow the in-context examples so individual concepts can be extracted by splitting over the dash ("-") sign. We further merge similar concepts according to the CLIP text embeddings. Two concepts are considered as similar if their cosine similarity over the text embeddings is higher than 0.9. We merge greedily until no concepts have similarity higher than 0.9. We use the pre-trained, frozen CLIP `ViT-L-14` text encoder, and normalize the extracted embeddings. As discussed in the method section, object state embeddings are calculated by averaging their corresponding visual concept embeddings. We normalize the state embedding again after taking the average.

C Dataset Details

We provide the details of the ChangeIt and HowToChange datasets in Table A8. These datasets consist of instructional videos for daily tasks beyond cooking (e.g., dyeing T-shirt). HowToChange contains a diverse set of objects, including a subset of novel objects that do not appear in the training set, making it suitable for evaluating the generalization ability of models. ChangeIt features a wider variety of actions and longer video sequences.

Table A8: Statistics of ChangeIt and HowToChange datasets.

Datasets	#Objects	#Actions	#Videos	#Training	#Evaluation	Avg. Duration (s)
ChangeIt	42	27	35,095	34,428	667	276
HowToChange	134	20	41,499	36,075	5,424	41

Table A9: Statistical significance of the object state recognition performance between our method and the baseline methods. We report p-values and statistical significance. Results marked with ✓ indicate statistical significance ($p < 0.05$).

Methods	ChangeIt		ChangeIt (Open-world)				HowToChange					
	State Pre@1	Action Pre@1	State Pre@1		Trans. Pre@1		F1 (%)		Precision (%)		Pre@1 (%)	
			seen	novel	seen	novel	seen	novel	seen	novel	seen	novel
SAGE v.s. LFC Souček et al. (2022)	6.1e-20 (✓)	1.0e-20 (✓)	4.1e-17 (✓)	4.9e-29 (✓)	5.4e-09 (✓)	9.6e-17 (✓)	1.3e-66 (✓)	5.9e-67 (✓)	3.1e-57 (✓)	2.2e-68 (✓)	1.9e-166 (✓)	9.3e-151 (✓)
SAGE v.s. MTC (Souček et al., 2024)	5.5e-05 (✓)	5.6e-06 (✓)	4.9e-11 (✓)	3.2e-35 (✓)	4.7e-15 (✓)	1.1e-25 (✓)	3.0e-40 (✓)	3.5e-57 (✓)	2.9e-21 (✓)	2.1e-38 (✓)	1.3e-101 (✓)	2.2e-120 (✓)
SAGE v.s. VidOSC (Xue et al., 2024)	2.7e-01 (✗)	7.5e-03 (✓)	2.7e-01 (✗)	1.1e-02 (✓)	1.0e+00 (✗)	1.2e-02 (✓)	1.0e+00 (✗)	9.3e-02 (✗)	3.5e-01 (✗)	6.5e-03 (✓)	1.8e-03 (✓)	1.4e-03 (✓)

D Statistical Significance

We conduct Students' T-test to evaluate the statistical significance of the result of our main experiments in Table 6. The results are shown in Table A9. Our model significantly outperforms LFC and MTC for both known and novel objects and significantly outperforms VidOSC for novel objects.