Supplementary Material
Chop & Learn: Recognizing and Generating Object-State Compositions

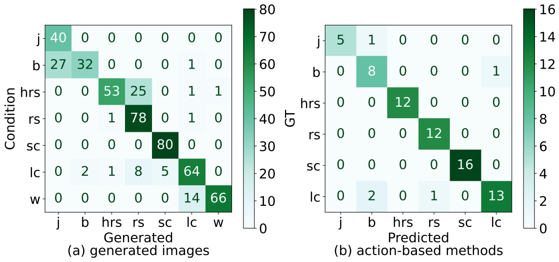
Nirat Saini* Hanyu Wang* Archana Swaminathan Vinoj Jayasundara Bo He
Kamal Gupta Abhinav Shrivastava

University of Maryland, College Park

Contents

1. Scope and Limitations

The objective of ChopNLearn dataset is compositional generation and recognition, using a granular and structural understanding of transferable object states. Terms such as 'slice', 'dice' alone often lead to a loss of granular information. For *e.g.*, a sliced apple can be horizontally or vertically sliced, or cut in the half, and then sliced as semicircles. Hence, we use more specific categories than other traditional state change datasets as shown in Tab. 1. Moreover, the subtle state change understanding is a challenging task on its own merit [3, 14, 15]. Recognizing/segmenting actions in a video is a complementary task and an interesting future direction, but is currently beyond the scope of this work.



(a) generated images
(b) action-based methods

Figure 1. Confusion matrix for generation & action-based tasks.

Moreover, we acknowledge that making the classes more granular can be confusing for the model, which appears similar. To confirm this, in Fig. 1, we show the confusion matrix for generated images (classified by the State-Classifier, and action-based method using the final states (CAF+R3D for Split 3 in Tab. 3). We see baton and julienne, half round slice and round slice, are two difficult pairs for compositional generation. In contrast, the action-based method can classify most states correctly. We hypothesize that since action-based methods use trajectory, and multiple frames for classification, the confusion between similar object-state pairs is significantly reduced.

2. Future Work

2.1. Green Screen Removal (extension).

As described in Section 4.3 of main paper, we chose green screen to focus on the object states, and such that the object pieces can be segmented easily. As some preliminary work, in Figure 2, shows some results on how thresholding using simple opencv library functions works for segmenting the object pieces after styles of cuts are applied. Further, we use Midjourney [1], which is a Stable Diffusion [6] based text to image generator tool, to generate a set of images with chopping boards. For each camera angle, we generate 7-10 images, for red, yellow, blue, white and wooden chop-

[*] First two authors contributed equally.

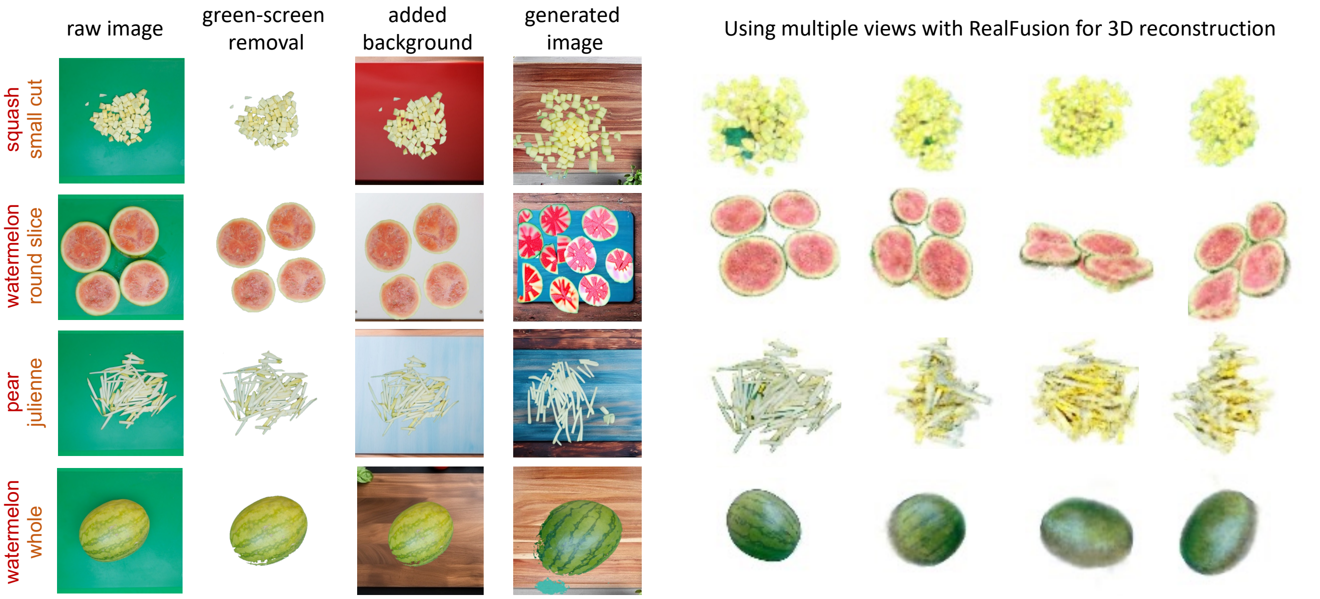


Figure 2. We show different uses of our dataset. The first column shows the raw images. The second column shows basic Python based green screen removal techniques on the dataset. The third column uses generated kitchen chopping board images replacing the green screen using the segmented object pieces. The fourth column shows results of training stable our benchmark SD+TI+FT model with the images without a green screen. Further the rightmost four columns show promising results towards 3D reconstruction of deformable objects, which can be a potential future research problem ChopNLearn can be used for.

ping boards. The captions used for this explain the color of board, the view or angle and the surrounds, for instance "empty plastic red colored chopping board in from this view point –style raw " is one of the captions used for one view. Nonetheless, we are aware that achieving precise camera angles during image generation is a challenging task. Many of these models exhibit a bias toward placing certain fruits or vegetables around or atop the chopping board. As a result, we have occasionally supplemented the model with a reference image from the dataset, accompanied by a directive to "generate the same viewpoint and camera angle relative to the chopping board." Despite these efforts, the outcomes we achieve remain suboptimal, particularly when it comes to three-dimensional perspectives. The top view seems to yield the most favorable outcomes for background generation. We utilize these images to substitute the backgrounds in segmented images. However, there are instances where the generated background's view and angle do not align with those of the actual image from which the object segments originated. This occasionally results in sections of the objects appearing to levitate above the chopping board. While we acknowledge that the authenticity of these generated images may sometimes fall short due to occasional inaccuracies in green screen removal and misalignment between object positioning and background image angles, this experiment still holds value in mitigating green screen bias during model training. Additionally, our analysis with respect to green screen removal and background addition sheds light on a significant prospective challenge – the ability to mat and position objects convincingly from diverse angles within backgrounds to achieve a realistic effect. Hopefully, paves the way for new avenues of research and potential applications of ChopNLearn in the realm of detailed background matting.

2.2. 3D reconstruction

Collecting data as well as generating 3D models for deformable objects is still an open problem. We demonstrate results of some preliminary experiments with our dataset for this task. We use RealFusion [8] to recover a promising 3D scene from a single image of our various cut states Figure 2. We believe that with our multi-view camera setup, this direction is worth exploring in future work for more accurate 3D reconstruction and can be an interesting task.

3. Details of User Study

The purpose of conducting a user study was to see if our generative models were able to create images that were of high fidelity and stayed true to capturing the semantic understanding of the object-state composition provided as a text prompt. We chose 20 compositions from the test set, which are unseen as a pair in the training and finetuning of the generative models. These compositions from the test set are also given as a text prompt to five generative models, i) Dreambooth ii) Stable Diffusion iii) Stable Diffusion+Textual Inversion iv) Stable Diffusion finetuned on

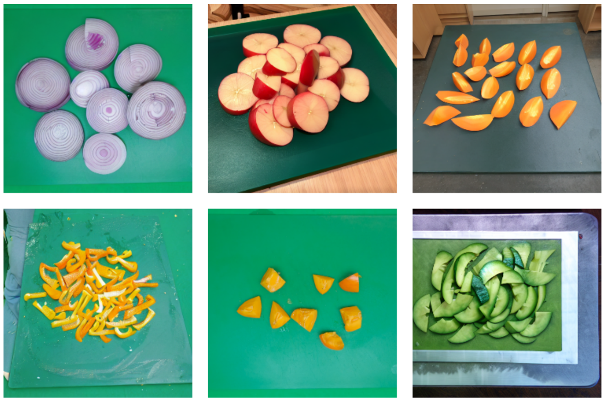
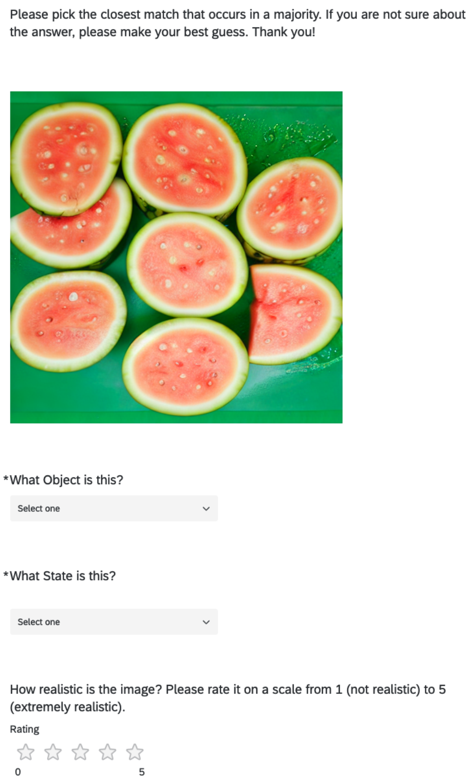


Figure 3. Examples of images from the test set and samples from the generative models presented to participants in the user study



Please pick the closest match that occurs in a majority. If you are not sure about the answer, please make your best guess. Thank you!

*What Object is this?

Select one

*What State is this?

Select one

How realistic is the image? Please rate it on a scale from 1 (not realistic) to 5 (extremely realistic).

Rating

☆ ☆ ☆ ☆ ☆

0 5

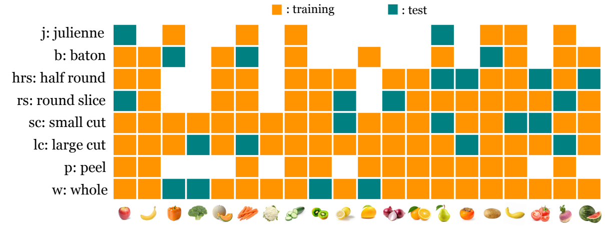
Figure 4. Snapshot of questionnaire presented to participants of the user study

our training dataset v) Stable Diffusion + Textual Inversion finetuned on our training dataset. We evenly chose a distribution of 5 samples per composition, and including the test set + 5 generative models, we had 6 sets to sample images from. The total number of images used for the study was 750 and we asked 30 participants to label each of these images for their object and state as well as rate the realism on a scale of 1-5. We show some examples of images encountered in our user study in Figure 3 and a snapshot of how the user study questionnaire looks like in Figure 4.

4. Compositional Image Generation

4.1. Dataset Split

In the compositional image generation task, we split all (object, state) compositions into a training set consisting of 87 compositions and a test set consisting of 25 compositions. For each composition of object and state present in the test set, the training set includes exactly one of either the object, or the state, but not both. We also ensure that for each (object, state) composition (o, s_i) in the test set, there exists a composition (o, s_j) in the training set, where s_i and s_j belong to the same state related group defined in Section 3 of our main paper. Each combination in our dataset has 8-12 images, resulting in a total of 1032 images in the training set and 296 images in the test set. Figure 5 illustrates the detailed dataset split used in the compositional image generation task. In this figure, training compositions and test compositions are marked with orange and teal, respectively. Unmarked compositions are not included in our dataset. Figure 6 and Figure 7 show some example images in our training set and test set, respectively.



j: julienne	🟦		🟧			🟧		🟧						🟦		🟧	🟧		🟧	
b: baton	🟧	🟧	🟦		🟧	🟦		🟧			🟧			🟧		🟦	🟧		🟧	🟧
hrs: half round	🟧	🟧			🟧	🟧		🟧	🟧	🟧		🟧	🟧	🟦	🟦	🟧	🟧	🟦	🟧	🟦
rs: round slice	🟦	🟧			🟧	🟧		🟧	🟧	🟦		🟦	🟧	🟧	🟧	🟧	🟧	🟧	🟦	🟧
sc: small cut	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟦	🟧	🟧	🟧	🟦	🟧	🟧	🟦	🟦	🟧	🟧
lc: large cut	🟧	🟧	🟧	🟦	🟧	🟦	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟦	🟧	🟧	🟧	🟦	🟧
p: peel	🟧	🟧				🟧		🟧	🟧		🟧	🟧	🟧	🟧	🟧	🟧	🟧		🟧	
w: whole	🟧	🟧	🟦	🟦	🟧	🟧	🟧	🟧	🟦	🟧	🟦	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧

Figure 5. **Dataset Split used in The Compositional Image Generation Task**. Training compositions and test compositions are marked with orange and teal, respectively. Unmarked compositions are not included in our dataset.

4.2. Number of views.

We assess the impact of the number of views on the image generation task in Tab. 1 using the SD+FT+TI setting. Using more views improves training data in terms of both quantity and diversity, yielding results with better patch FID and object accuracy, and maintaining high state accuracy even though generating images in more views is more difficult. The use of 4 cameras also has applications in few-shot 3D reconstruction tasks, which although beyond the scope of current work, are discussed in Section 2.

Table 1. **Number of views ablation results.**

View IDs	Object Acc. (%) ↑	State Acc. (%) ↑	Patch FID ↓
1	42.4	78.2	184.7
1, 2	56.8	81.2	121.4
1, 2, 3	66.2	78.3	115.4
1, 2, 3, 4	67.8	81.4	82.2

4.3. Patch FID Details

We propose patch FID to access the quality of the generated images. In short, it calculates Fréchet Inception Distance on the image patch level. Specifically, we modify the standard FID by sampling 224 × 224 random crops from the real images, as well as the synthetic images. We use 32 patches per image. For each generative model, we compute patch FID using all available real image patches and 16000 generated image patches, and report the average number for the test compositions.

4.4. Object State Classifier Details

As mentioned in our main paper, to automatically evaluate the correctness of the generated images, we train a classifier on real images for classifying objects and states independently. This classifier is built on a CLIP-ViT-B/32 [11]. To classify an input image, it takes this image and texts of all possible labels (all objects or all states) as input. Cosine similarities between the image embedding and text embeddings of all possible labels are computed as the classification logits, which are used to calculate the standard cross entropy loss for classification problems. During hyperparameter-searching, we fine-tune the CLIP model on a different training split that all (object, state) compositions are seen, and report the validation accuracies in the Table 2 of our main paper. One single model is used to predict both object and state.

After deciding on all hyperparameters and training settings, we train our final-version object state classifier on all available data in our dataset to maximize its performance. We keep all parameters in the CLIP model learnable and train it 2000 epochs using a learning rate of $3e-5$. We use a batch size of 128, and a warm-up cosine learning rate schedule [7].

4.5. Method Details

Stable Diffusion. (SD) We briefly describe classifier-free guidance in diffusion models. Diffusion models generate an image from Gaussian noise via an iterative denoising process. Expected mean square error is used as the denoising objective:

$$\mathcal{L}_{\text{Diff}} = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon},t\sim\mathcal{U}(0,1)}\left[\|\ \boldsymbol{\epsilon} - \epsilon_\theta(\alpha_t\mathbf{x}_0 + \sigma_t\boldsymbol{\epsilon},\mathbf{c})\ \|^2\right] \quad (1)$$

where $\mathbf{x}_0$ is an image and $\mathbf{c}$ is the optional condition from the training data. $\boldsymbol{\epsilon}$ is the additive Gaussian noise. α_t, σ_t are scalar functions of time step t. $\boldsymbol{\epsilon}_\theta$ is the diffusion model with trainable parameters θ. For sampling images from the text condition, SD employs classifier-free guidance [6], such that at every time step (during sampling), predicted noise is adjusted via:

$$\hat{\boldsymbol{\epsilon}_\theta}(\mathbf{x}_t,\mathbf{c}) = \omega\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,\mathbf{c}) + (1-\omega)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t) \quad (2)$$

where ω is the guidance scale. In our experiments, ω is set to be 7.5 in all methods using it.

SD + Textual Inversion (TI). In this method, Equation (1) is used for token embedding optimization. SD weights are kept fixed during training. We use a learning rate of $3e-3$ with a warm-up cosine learning rate schedule [7], a batch size of 4, and train the model for 16000 steps.

DreamBooth. The text prompt we used for DreamBooth fine-tuning is "An image of o_i cut in the [V] style", where o_i is the i^{th} object and [V] is a rare unique identifier representing the state this model is fine-tuned for. The goal of DreamBooth is to overfit a small dataset without drifting too far away from the pre-trained model. Following the available open-source implementation, we use a fixed learning rate of $5e-6$, a batch size of 1, and train the model for 400 steps.

SD + Fine-tuning (FT). We also fine-tune SD while keeping the text encoder fixed. The UNet parameters of the diffusion model are optimized using the diffusion loss defined by Equation (1). We use a learning rate of $5e-6$ with a warmup cosine learning rate schedule [7], a batch size of 4, and train the model for 8000 steps.

SD + TI + FT. When combining SD fine-tuning and Textual Inversion [4] together, we use a learning rate of $5e-6$ for all UNet parameters and a learning rate of $3e-3$ for all added token embeddings. A warmup cosine learning rate schedule [7] is employed for all parameters. We use a batch size of 4, and train the model for 16000 steps.

4.6. Additional Qualitative Results

To better compare the compositional image generation performance of various methods discussed in the main paper, we show additional generated images from them for seven (object, state) compositions in Figure 8 and Figure 9, where the compositions are from the training set and test set, respectively.

5. Compositional Action Recognition

5.1. Dataset Splits

Given the diversity of views and object types and styles, we can construct multiple training and testing splits. In this paper, we present results on three selected splits. For each split, we create training, test and validation set. The validation set is for evaluating the model on training classes, which consists of 10-15% unseen samples for the seen training compositions. Training and test sets have a disjoint set of compositional classes, in an 80-20% ratio. All of the splits in our dataset are created based on object-final state compositions in the videos.

We leverage these related groups defined in Section 3.1 in the main paper, to create different splits for training and testing. All splits use multi-view camera angles and involve creating seen and unseen object-final state compositions in training and testing sets. This ensures cross-view training and test splits, as used in other multi-view datasets [13, 17]. The training set consists of samples from three cameras,

Table 2. Input frames ablation. We do experiments on two settings to demonstrate that taking full video as input is necessary. The first row takes the full length of the video as input. The second row takes the first and last frames of the video as input. Object-final state classification accuracy is reported here.

			Split 1		Split 2		Split 3	
Input	Model	Features	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3
Full	Transformer [16]	I3D [2]	10.9	44.3	14.6	44.2	11.1	44.4
First&Last	Transformer [16]	I3D [2]	6.3	25.6	9.8	31.0	7.9	34.9

Table 3. Other splits: We also present other possible splits of data. All the results are using I3D [2] pre-trained features along with one layer Transformer [16] model. Comp. represents the initial: object-initial state composition and final: object-final state composition results for each split.

	Split 4		Split 5		Split 6		Split 7		Split 8		Split 9	
Comp.	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3
Initial	46.5	78.9	65.2	90.6	37.5	70.8	41.1	76.0	47.2	71.5	41.2	78.3
Final	45.3	75.9	73.9	92.6	37.7	69.2	41.4	72.2	48.9	73.0	42.9	77.6

while the test set includes samples of compositions from one camera whose view is never seen during training.

Similar to the three splits mentioned in the paper, we explore multiple other splits, with different constraints to choose those 3 splits. We find that other splits were not as challenging for the existing baselines, and hence only propose 3 splits that are challenging. In Table 3, we present the results for splits 4-9, which were considered for the data. We only show I3D [2] based Transformer model for these splits. All the splits consider the constraints for the object-final state. The details of each split are as follows:

Split 4: This is the same as the split used for the Compositional Image Generation task (mentioned in Section 4 of the main paper). We use the related groups to split the object-final states, such that objects which are seen with one of the states in a related group in training, are tested on the other related group during testing. This is also similar to Split 2 in the main paper, however, the multi-view constraint is not there. All the camera views are used for training and testing.

Split 5: In this split, we have the participant constraint. All samples from participants 1 and 2 are part of the training set, while samples from participant 3 are in the test set.

Split 6: This is a combination of split 4 and 5, which has two constraints: using related groups for splitting object-final states in different splits, and using only participant id 3 for the test set.

Split 7: This split is about multi-camera view. We use Camera 1,2,3 views in the training set, while the camera 4 view is part of the test set. No other constraint regarding related groups for splitting on the basis of object-final states is used.

Split 8: This split is similar to Split 1 in the main paper, without the multi-camera constraint. The object-final state compositions are split randomly into train/test. We use all camera views for both sets, without constraining to distinct views for each set.

Split 9: This split is similar to Split 3 in the main paper, without the multi-camera constraint. The object-final state compositions are split based on random groups for objects and states. We use all camera views for both sets, without constraining to distinct views for each set.

We do not have a split having all constraints, *i.e.* participant constraint, related groups and multi-view constraint, since all of these together end up leaving a total of 400 video clips, which are very few for training and testing. We show only top@1 accuracy for object-final state composition in Table 3.

5.2. First and last segment classification

For compositional action recognition, we emphasize that the model must learn to predict the object-initial state composition and the object-final state composition. Moreover, some works [3, 12] use a similar setup for object state classification and use only the first and last frame/segment for this. Ideally, the first few frames and last few frames should be sufficient for understanding the changes in object states. We also experiment with the first and last segments of videos, for classification. The results for the 3 selected splits (mentioned in the main paper) are in Table 2. We find that using the additional middle frames improves the classification accuracy for the final composition.

5.3. Finetuning Backbone

The results we show in the paper without finetuning any pre-trained features (I3D [2], MIL-NCE [9], R3D [5]). For the sake of completeness, we also show results with finetuning the backbone for R3D features in Table 4. Although the top@1 accuracy is much better, it is still not 100%. Moreover, the dataset is much smaller and overfits very quickly

Table 4. Results of finetuning R3D [5] backbone. "Start/End" denotes the prediction results for the initial and the final state composition with the correct object type.

		Split 1				Split 2				Split 3			
		Start		End		Start		End		Start		End	
Model	Finetune	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3
CAF [10]		53.5	88.7	57.8	88.7	55.1	95.7	58.0	95.7	62.0	93.0	63.4	93.0
CAF [10]	✓	80.3	98.6	87.3	98.6	84.1	98.5	89.9	98.5	88.7	98.6	88.7	98.6

for backbones which are trained on 10x more data. Hence, for sake of benchmarking, we propose not fine-tuning the features for consistency.

6. Project Webpage and License

For more details, results and analysis, please visit our website at: https://chopnlearn.github.io.

License. All files in this dataset are copyright by us and published under the Creative Commons Attribution-NonCommerial 4.0 International License, found at https://creativecommons.org/licenses/by-nc/4.0/. This means that you must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. You may not use the material for commercial purposes.

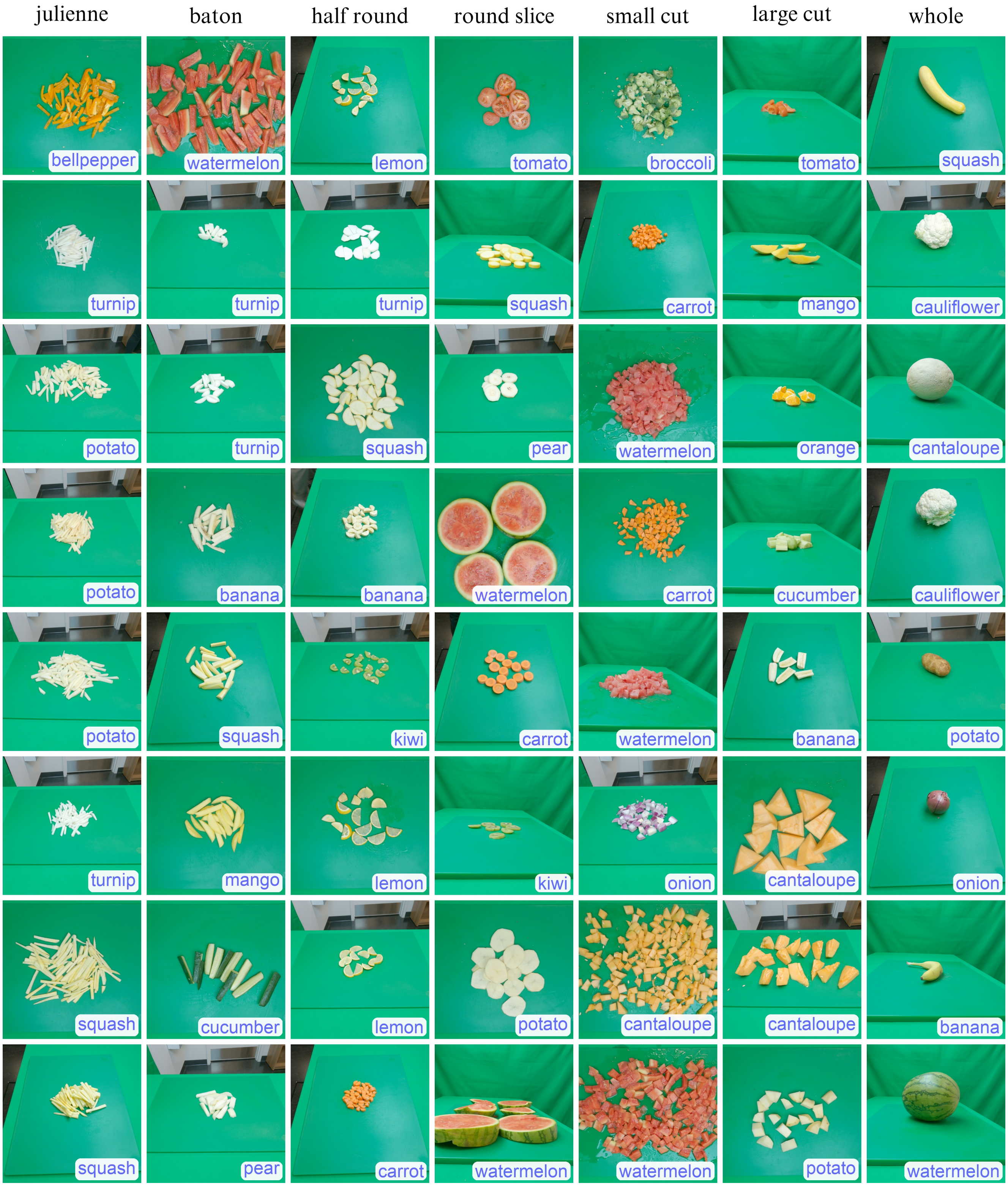


Figure 6. **Example Images In The Training Set**. Eight example images are shown in a column for each state. State labels are shown in the first row. Object labels are marked on the bottom right corner of each image.

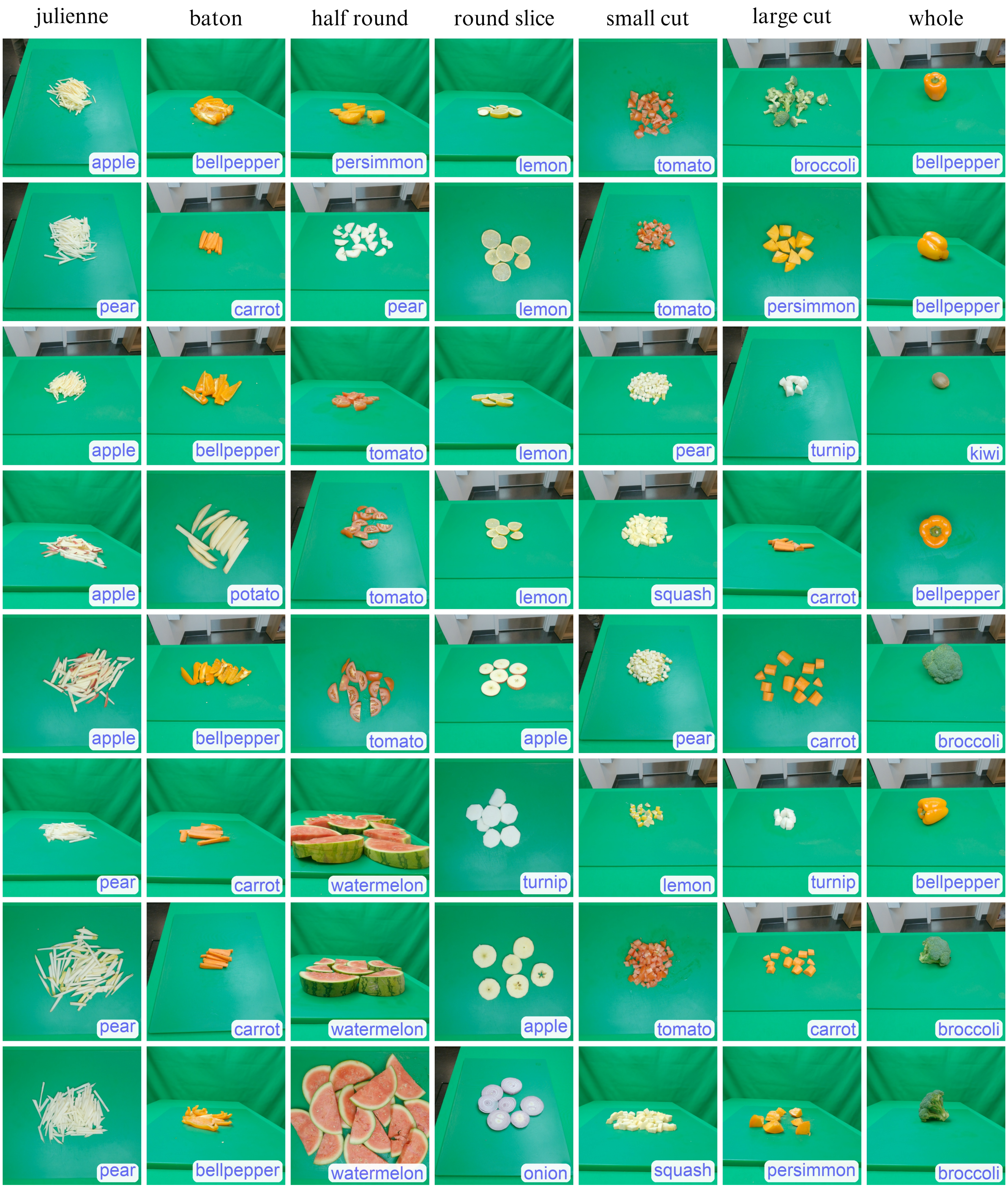


Figure 7. **Example Images In The Test Set**. Eight example images are shown in a column for each state. State labels are shown in the first row. Object labels are marked on the bottom right corner of each image.

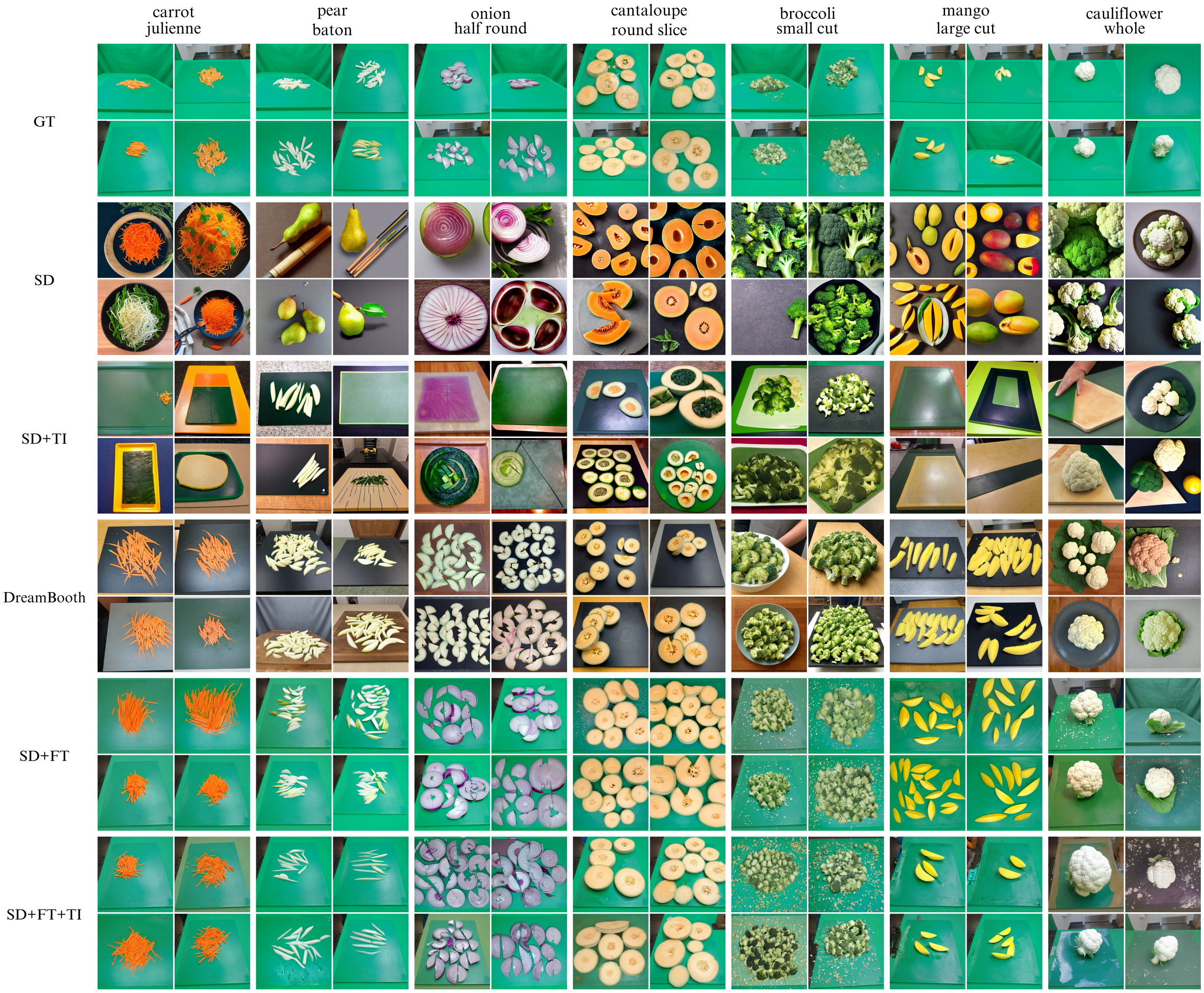


Figure 8. **Additional Compositional Generation Samples Using Training Compositions** Ground Truth (GT) real images are shown in the first row for reference. Seven object-state compositions in the training set are displayed, each with four generated samples for each method. Please zoom in to see the details.

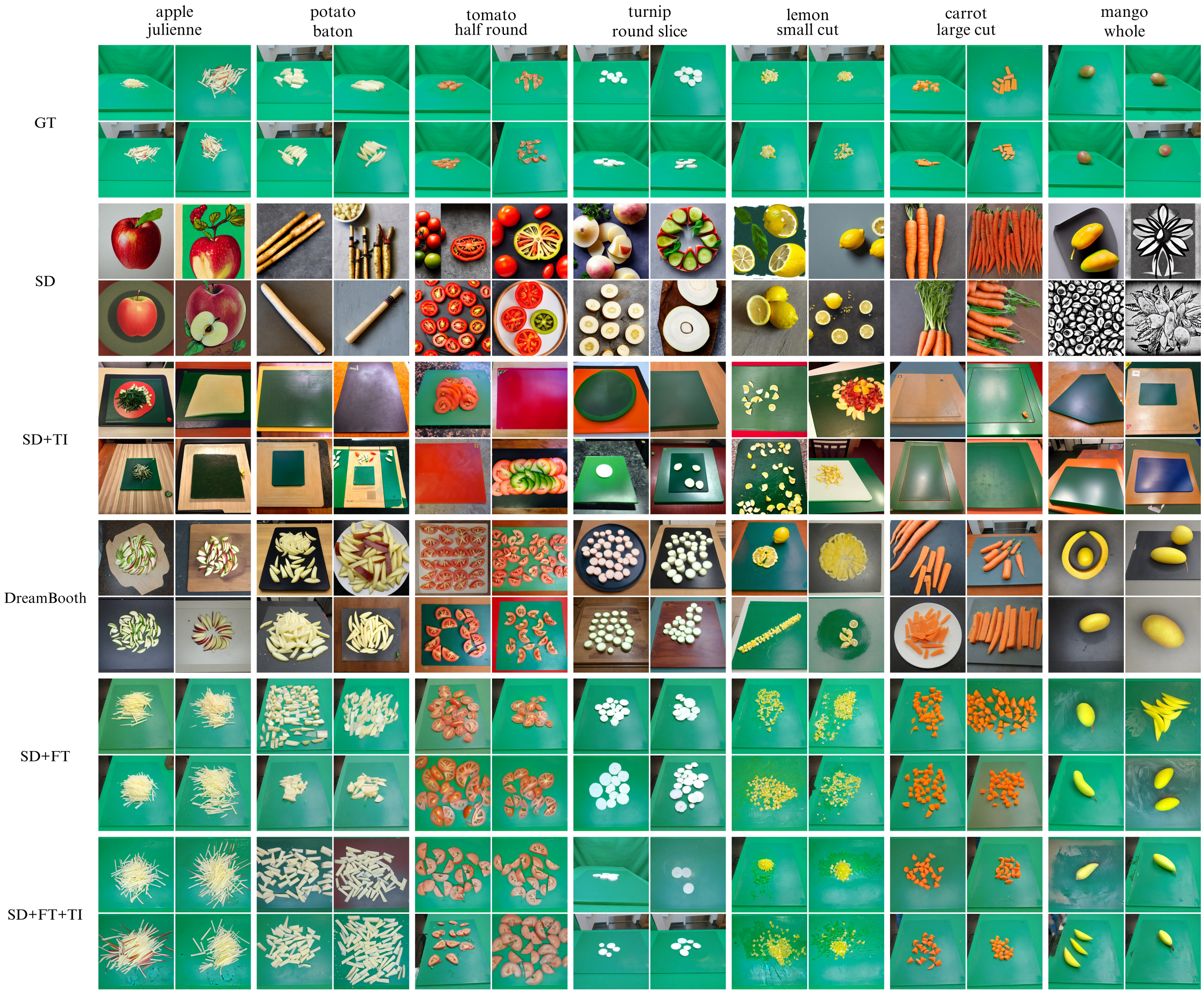


Figure 9. **Additional Compositional Generation Samples Using Test Compositions** Ground Truth (GT) real images are shown in the first row for reference. Seven object-state compositions in the test set are displayed, each with four generated samples for each method. Please zoom in to see the details.

References

[1] Midjourney v5. https://www.midjourney.com/home/?callbackUrl= 1

[2] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, 2017. 5

[3] Alireza Fathi and James M Rehg. Modeling actions through state changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2586, 2013. 1, 5

[4] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 4

[5] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3154–3160, 2017. 5, 6

[6] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 1, 4

[7] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 4

[8] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Realfusion: 360 reconstruction of any object from a single image. In *CVPR*, 2023. 2

[9] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9876–9886, 2019. 5

[10] Gorjan Radevski, Marie-Francine Moens, and Tinne Tuytelaars. Revisiting spatio-temporal layouts for compositional action recognition. *British Machine Vision Conference (BMVC)*, abs/2111.01936, 2021. 6

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 4

[12] Nirat Saini, Bo He, Gaurav Shrivastava, Sai Saketh Rambhatla, and Abhinav Shrivastava. Recognizing actions using object states. In *ICLR2022 Workshop on the Elements of Reasoning: Objects, Structure and Causality*, 2022. 5

[13] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and G. Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1010–1019, 2016. 4

[14] 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. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13936–13946, 2022. 1

[15] Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. Multi-task learning of object state changes from uncurated videos. *ArXiv*, abs/2211.13500, 2022. 1

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 5

[17] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. Cross-view action modeling, learning, and recognition. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2649–2656, 2014. 4