

Learning Compositional Models for Activity Understanding

by

Megha Nawhal

B.Tech.- M.Tech. (Electrical Engineering), Indian Institute of Technology Kanpur, 2015

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

© Megha Nawhal 2023
SIMON FRASER UNIVERSITY
Spring 2023

Copyright in this work rests with the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.

Declaration of Committee

Name: Megha Nawhal

Degree: Doctor of Philosophy (Computing Science)

Title: Learning Compositional Models for Activity Understanding

Committee:

Chair: Angelica Lim
Assistant Professor, Computing Science

Greg Mori
Supervisor
Professor, Computing Science

Manolis Savva
Committee Member
Assistant Professor, Computing Science

Yasutaka Furukawa
Examiner
Associate Professor, Computing Science

Tinne Tuytelaars
External Examiner
Professor, Electrical Engineering
KU Leuven

Abstract

Compositionality serves as a key design principle in artificial intelligence algorithms. In this thesis, we focus on developing compositional models for activity understanding. The core idea of this thesis is to design compositional representations for human activity videos that are specific to the downstream task and are learned using different types of compositional information available at various granularities of the videos. We applied this idea to a diverse set of video tasks aimed at understanding realistic activities. First, we introduce the task of generating human-object interactions in a zero-shot compositional setting and propose a generative model that uses an object-centric spatio-temporal scene graph for generating videos. Second, we work on the problem of temporal action localization and develop an end-to-end learnable transformer model that represents the input video as graphs over video segments and output space of actions as graphs of abstract learnable entities. Third, we focus on the task of long term action anticipation and design a transformer based model trained using two-stage learning approach to employ segment-level and video-level representations for action anticipation. Overall, we demonstrate the benefits of designing compositional representations for human activity videos.

Keywords: Activity Understanding; Video Modeling; Compositional learning

Acknowledgements

I can't express how grateful I am to reach the end of my PhD journey at SFU and there are so many people to thank for making this journey a wonderful one for me.

Firstly, thanks to my advisor Greg Mori. Over the years, I have learned a lot from him - scoping projects, writing papers, presenting research work, evaluating and analyzing problems, the value of constructive feedback, and the importance of a good work-life balance. He has always prioritized my well-being and success, and made me feel confident at every stage of my PhD I feel fortunate and grateful to have him as my advisor. His friendship and his mentorship means a lot to me and I aspire to be the type of friend and mentor to others, as he has been to me.

Thanks to my dissertation and supervisory committee members for taking their time and offering valuable feedback.

Thanks to Utsav Prabhu for my two Google internships. These internships helped nurture my visions and opinions about building practical systems for video understanding. His contagious enthusiasm inspired me to explore challenging problems during both my internships. These internships trained me for working on applied research projects and taught me important skills necessary for cross-team collaboration. Thanks to all other collaborators and mentors at Google - Michael Nechyba, Caroline Pantofaru, Min-hsuan Tsai, Wei-Hong Chuang, Hau Wu, Boqing Gong, Ming-Hsuan Yang, Arsha Nagrani, Tianyi Zhou - for their insightful discussions during these internships. These discussions shaped the way I think about research problems related to video understanding. My conversations with Michael Nechyba and Caroline Pantofaru also played a key role in my career plans after PhD.

Thanks to James Hillis and Karl Ridgeway for my Meta's Reality Labs internship. This internship taught me about designing new tasks and benchmarks for egocentric video understanding in the context of Augmented Reality. Over the course of my internship, my discussions with collaborators at Meta made me realize the importance of scoping problems and designing simple models for practical applications.

Thanks to Cuong Nguyen and Mai Long for my Adobe internship. I still remember the day I visited the Mixed Reality lab at Adobe Research and the feeling of being amazed by it. This internship shaped my thoughts about technical and design problems at the intersection of Augmented Reality and Computer Vision.

Thanks to Andreas Lehrmann and Leonid Sigal for my Borealis AI internships. Over the course of my internships, I learnt a lot from their valuable insights into generative modeling techniques for visual understanding. Leonid's constructive feedback on the video generation work and numerous discussions about research in this area shaped my overall Ph.D trajectory. Andreas' work ethic taught me important skills about organizing, and leading projects.

Thanks to my amazing colleagues - Sachini Herath, Zhiwei Deng, Mengyao Zhai, Lei Chen, Jiawei He, Yu Gong, Thibaut Durand, Fred Tung, Akash Abdu Jyothi, Srikanth Muralidharan, Nazanin Mehrasa, Ruizhi Deng, Mostafa S. Ibrahim - for their warm welcome to SFU and to the lab, the countless research brainstorming sessions, and for easing my transition to Canada. Special thanks to Sachini Herath for always cheering me up, keeping me sane during the COVID-19 lockdowns, and for all the fun conversations.

I am extremely grateful to my mother Kiran Sharma. Her endless support, unconditional love, and continual support has made me who I am today. Her determination to provide quality education to her daughters has been an inspiration to me. After my PhD, I hope to get involved with the cause of educating and empowering women and girls all around the world.

Finally, I owe immense gratitude to my partner Rama Chandra Kota. If not for his patience and support over the years and sacrifices he had made, I would not have made it.

Table of Contents

Declaration of Committee	ii
Abstract	iii
Acknowledgements	iv
Table of Contents	vi
List of Tables	viii
List of Figures	xiii
1 Introduction	1
1.1 Activity Understanding	1
1.2 Compositional Models in Computer Vision	2
1.3 Contributions	5
2 Related Work	7
2.1 Compositional Representations for Videos	7
2.2 Applications	15
3 Learning to Generate Human-Object Interactions	27
3.1 Introduction	27
3.2 Related Work	29
3.3 HOI-GAN	32
3.3.1 Task Formulation	33
3.3.2 Model Description	33
3.4 Experiments	38
3.4.1 Datasets and Data Splits	38
3.4.2 Evaluation Setup	41
3.4.3 Quantitative Evaluation	42
3.4.4 Qualitative Analysis	45
3.5 Limitations	50

3.6	Conclusion	51
4	Learning Activity Graphs for Temporal Action Localization	52
4.1	Introduction	52
4.2	Related Work	55
4.3	Proposed Approach	56
4.3.1	Activity Graph Transformer	57
4.3.2	Implementation Details	64
4.4	Experiments	65
4.4.1	Experimental Setup	65
4.4.2	Quantitative Analysis	71
4.4.3	Qualitative Analysis	77
4.5	Limitations	78
4.6	Conclusion	78
5	Learning Segment-Level Representations for Action Anticipation	80
5.1	Introduction	80
5.2	Related Work	82
5.3	Action Anticipation with ANTICIPATR	84
5.3.1	Stage 1: Segment-level Training	86
5.3.2	Stage 2: Action Anticipation	88
5.4	Experiments	92
5.4.1	Experimental Setup	92
5.4.2	Evaluation.	95
5.4.3	Quantitative Analysis	96
5.4.4	Qualitative Analysis	104
5.5	Limitations	104
5.6	Conclusion	105
6	Conclusion and Future Work	106
6.1	Conclusion	106
6.2	Future Directions	107
	Bibliography	109

List of Tables

Table 3.1	Generation Scenarios. Description of the conditional inputs for the two generation scenarios GS1 & GS2 used for evaluation. ✓ denotes ‘Yes’, ✗ denotes ‘No’.	40
Table 3.2	Quantitative Evaluation. Comparison of HOI-GAN with C-VGAN, C-TGAN, and MoCoGAN baselines. We distinguish training of HOI-GAN with bounding boxes (<i>bboxes</i>) and segmentation masks (<i>masks</i>). Arrows indicate whether lower (↓) or higher (↑) is better. [I: inception score; S: saliency score; D: diversity score]	42
Table 3.3	Ablation Study. We evaluate the contributions of our pixel-centric losses (F,G,V) and relational losses (first block vs. second block) by conducting ablation study on HOI-GAN (masks). The last row corresponds to the overall proposed model.[F: frame discriminator \mathbf{D}_f ; G: gradient discriminator \mathbf{D}_g ; V: video discriminator \mathbf{D}_v ; R: relational discriminator \mathbf{D}_r]	42
Table 3.4	Quantitative Evaluation (Effect of Word Embeddings). Comparison of HOI-GAN with C-VGAN, C-TGAN, and MoCoGAN baselines using one-hot encoded labels instead of embeddings as conditional inputs(default version). (see section 3.4.3). Arrows indicate whether lower (↓) or higher (↑) is better. [I: inception score; S: saliency score; D: diversity score]	43
Table 3.5	Human Evaluation. Human Preference Score (%) for scenarios GS1 and GS2. All the results have p-value less than 0.05 implying statistical significance.	43
Table 4.1	Comparison with state-of-the-art (ActivityNet-1.3). We report the mean average precision at different intersection over union thresholds (mAP@tIoU). These results are reported on the validation set. ↑ indicates higher is better.	64
Table 4.2	Comparison with state-of-the-art (THUMOS14). We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.	65

Table 4.3	Comparison with state-of-the-art (EPIC-Kitchens100). We report mean average precision at different intersection over union thresholds (mAP@tIoU). These results are reported on the validation set. \uparrow indicates higher is better.	66
Table 4.4	Comparison with state-of-the-art (Charades). We report mean average precision (mAP) computed following [213]. \uparrow : higher is better.	66
Table 4.5	Impact of graph based reasoning (ActivityNet-1.3) We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. \checkmark and \times indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. \uparrow indicates higher is better.	67
Table 4.6	Impact of graph based reasoning (THUMOS14) We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. \checkmark and \times indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. \uparrow indicates higher is better.	67
Table 4.7	Impact of graph based reasoning (Charades) We report the mean average precision for ablated versions of our AGT model. \checkmark and \times indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. \uparrow indicates higher is better.	68
Table 4.8	Impact of graph based reasoning (EPIC-Kitchens100) We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. \checkmark and \times indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. \uparrow indicates higher is better.	68
Table 4.9	Ablation Study: Loss function (ActivityNet-1.3) We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . \checkmark and \times indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). \uparrow indicates higher is better.	69

Table 4.10	Ablation Study: Loss function (THUMOS14) We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.	69
Table 4.11	Ablation Study: Loss function (Charades) . We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report mAP to evaluate the performance of the model on Charades dataset. ↑ indicates higher is better.	70
Table 4.12	Ablation Study: Loss function (Epic-Kitchens100) We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.	70
Table 4.13	Impact of number of layers. We report performance of our AGT model with different number of layers in encoder and decoder. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #E indicates number of layers in encoder and #D indicates number of layers in decoder.	71
Table 4.14	Impact of number of heads. We report performance of our AGT model with different number of heads in the attention modules of the transformer network. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #heads indicates number of heads in attention modules of the transformer.	72
Table 4.15	Impact of action query graph size. We report performance of our AGT model with different number of nodes in the action query graph. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #queries indicates number of nodes in the action query graph.	73

Table 5.1	Results (Breakfast and 50Salads). We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance. Note that “Sener <i>et al.</i> [199] (features+labels)” use action labels from a segmentation algorithm as additional input. Baseline results are from respective papers.	93
Table 5.2	Results (EK-55 and EGTEA+). We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance. Baseline results are from respective papers.	94
Table 5.3	Ablation: Loss function (Breakfast and 50Salads). We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance. ✓ and ✗ indicate whether the component of the temporal loss is used or not respectively.	97
Table 5.4	Ablation: Loss function (EK-55 and EGTEA+). We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance. ✓ and ✗ indicate whether the component of the temporal loss is used or not respectively.	97
Table 5.5	Ablation: Anticipation Queries (Breakfast and 50Salads). We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance.	98
Table 5.6	Ablation: Anticipation Queries (EK-55 and EGTEA+). We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance.	98
Table 5.7	Ablation: Segment window length (Breakfast and 50Salads). We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance.	99
Table 5.8	Ablation: Segment window length (EK-55 and EGTEA+). We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance.	99
Table 5.9	Ablation: Sliding windows for Segment Encoder Training. Mean over classes accuracy for different observation/anticipation durations. Higher is better. [BF: Breakfast; 50SL: 50Salads]	100

Table 5.10 **Ablation: Sliding windows for Segment Encoder Training.** mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Higher is better. 100

Table 5.11 **Ablation: Set correspondence (Breakfast & 50Salads).** We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance. 101

Table 5.12 **Ablation: Set correspondence (EK-55 & EGTEA+).** We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance. 101

List of Figures

Figure 2.1	Videos as spatio-temporal region graphs. Illustration of the representation of a video as a spatio-temporal region graph.	8
Figure 2.2	Videos as sequences of unit-actions. Illustration of the representation of a video corresponding to an activity ‘ <i>cooking a meal</i> ’ as sequence of unit-actions wherein each unit-action has a variable temporal extent.	9
Figure 2.3	Videos as graphs of unit-actions. Illustration of the representation of a video depicting the activity of ‘ <i>preparing coffee</i> ’ as an undirected graph of unit-actions.	10
Figure 2.4	Videos as compositions of spatio-temporal scene graphs. Illustration of the representation of a video as composition of spatio-temporal scene graphs.	11
Figure 2.5	Videos as topological graphs. Illustration of the representation of a video as a topological graphs of regions that afford coherent actions.	12
Figure 2.6	Videos as compositions of trajectories. Illustration of the representation of a video as a combination of <i>subject</i> and <i>object</i> trajectories connected with a <i>predicate</i>	13
Figure 2.7	Videos as spatio-temporal graphs of visual concepts. Illustration of the representation of a video as a fully connected spatio-temporal graph of visual concepts (subject ‘S’, object ‘O’, predicate ‘P’ in the figure).	14
Figure 2.8	Action recognition. Overview of the compositional action recognition model (proposed by Ji <i>et al.</i> [100]) for human activity videos based on decomposing videos as compositions of spatio-temporal scene graphs.	17
Figure 2.9	Action recognition. Overview of the compositional action recognition model (proposed by Hussein <i>et al.</i> [93]) for long range videos based on decomposing videos as graphs of unit-actions.	18

Figure 2.10	Group activity recognition. Overview of the group activity action recognition model (proposed by Ibrahim and Mori [94]) for sports videos based on decomposing videos as image regions each containing a person involved in the activity.	19
Figure 2.11	Long term action anticipation. Overview of the long term action anticipation model (proposed by Nagarajan <i>et al.</i> [162]) for egocentric videos based on decomposing videos as topological maps of regions that afford coherent actions.	20
Figure 2.12	Scene affordance learning. Example for affordance prediction in egocentric videos obtained by decomposing videos as topological maps of regions that afford coherent actions (using a model proposed by Nagarajan <i>et al.</i> [162])	21
Figure 2.13	Video captioning. Overview of the video captioning model (proposed by Pan <i>et al.</i> [173]) that decomposing videos as spatio-temporal region graphs and model object interactions in videos.	22
Figure 2.14	Video prediction. Overview of the video prediction model (proposed by Ye <i>et al.</i> [267]) that models interactions between entities in the videos.	23
Figure 2.15	Video prediction. Examples from the compositional model proposed by Ye <i>et al.</i> [267]. The entities that compose the videos are also shown.	23
Figure 2.16	Video relationship reasoning. Examples of predictions using the model (proposed by Shang <i>et al.</i> [203]) for relationship detection (left) and tagging (right) tasks.	25
Figure 3.1	Generation of Zero-Shot Human-Object Interactions. Given training examples ‘wash aubergine’ and ‘put tomato’, an intelligent agent should be able to imagine action sequences corresponding to unseen action-object compositions, <i>e.g.</i> , ‘wash tomato’ and ‘put aubergine’.	28
Figure 3.2	Architecture Overview. The generator network \mathbf{G} is trained using 4 discriminators simultaneously: a frame discriminator \mathbf{D}_f , a gradient discriminator \mathbf{D}_g , a video discriminator \mathbf{D}_v , and a relational discriminator \mathbf{D}_r . Given the word embeddings of an action \mathbf{s}_a , an object \mathbf{s}_o , and a context image \mathbf{s}_I , the generator learns to synthesize a video with background I in which the action a is performed on the object o	32

Figure 3.3	Relational Discriminator. The relational discriminator D_r leverages a spatio-temporal scene graph to distinguish between object layouts in videos. Each node contains convolutional embedding, position and aspect ratio (AR) of the object crop obtained from MaskRCNN. The nodes are connected in space and time and edges are weighted based on their inverse distance. Edge weights of (dis)appearing objects are 0.	35
Figure 3.4	Architecture Details. Model architectures used in our experiments for: (i) Generator, (ii) Video Discriminator, (iii) Frame discriminator (gradient discriminator has similar architecture), (iv) Relational Discriminator. Best viewed in color on desktop.	37
Figure 3.5	Qualitative Analysis (Comparison with Baselines). Samples generated using the baseline models and HOI-GAN for a given composition of action-object (a, o) pair and context image. We provide middle frame of each generated video sample.	44
Figure 3.6	Qualitative Evaluation (GS1). Samples generated using our model in Generation Scenario 1, <i>i.e.</i> , both the target context image and the target action-object (a, o) composition are unseen during training. We provide 3 frames of the generated output and 3 frames of the original video (same context, action, object) from the test set for comparison.	45
Figure 3.7	Qualitative Evaluation (GS2). Samples generated using our model in Generation Scenario 2, <i>i.e.</i> , target action-object composition are unseen during training but target context image is seen with an object different from target object and a same/different action from target action. Thus, the overall target compositions comprising object, action and context are unseen during training. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We provide 5 frames for each generated video sample in the figure.	46
Figure 3.8	Qualitative Evaluation (GS2 - same action, same context, different objects). Samples generated using HOI-GAN in Generation Scenario 2 corresponding to a set of compositions with same context frame, same action and different objects. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We show the context frame with mask on the left in each row and the middle frame of each generated video.	47

Figure 3.9	Qualitative Evaluation (GS2 - same object, same context, different actions). Samples generated using HOI-GAN in Generation Scenario 2 corresponding to a set of compositions with same context frame, same object and different actions. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We show the context frame with mask on the left in each row and the middle frame of each generated video.	47
Figure 3.10	How does HOI-GAN generalize over compositions?. Training samples in the data to illustrate that HOI-GAN leverages the information available during training and learns to combine them in a meaningful way. This ability allows HOI-GAN to generalize over unseen compositions of action, object and context. We provide a few frames for each sample in the figure.	49
Figure 3.11	Failure Cases. Videos generated using HOI-GAN corresponding to the given action-object composition (a, o) and the context frame. We show 4 frames of the videos.	50
Figure 4.1	Main Idea. Given an untrimmed human activity video, we directly predict the set of action instances (label, start time, end time) that appear in the video. We observe that human activity videos contain non-sequential dependencies (illustrated by the overlapping ground truth instances as colored bars). In this work, we propose Activity Graph Transformer that captures this non-sequential structure by reasoning over such videos as graphs. Overall, the network receives a video and directly infers a set of action instances. The network achieves this by transforming a set of graph-structured abstract queries into contextual embeddings which are then used to provide predictions of action instances. It is trained end-to-end using classification and regression losses.	53

Figure 4.2	Model Overview. Activity Graph Transformer (AGT) receives a video as input and directly predicts a set of action instances that appear in the video. The input video is fed into a backbone network to obtain a compact representation. Then, the encoder network receives the compact video-level representation from the backbone network and encodes it to a latent graph representation <i>context graph</i> . The decoder network receives the context graph along with graph-structured abstract query encodings <i>action query graph</i> . The decoder transforms the action query graph to a graph-structured set of embeddings. Each node embedding of the decoder output is fed into a prediction head. The network is trained end-to-end using classification and regression losses for the action labels and timestamps of the action instances respectively.	57
Figure 4.3	Detailed Architecture Architecture of Activity Graph Transformer. ‘Q’, ‘K’, ‘V’ are query, key and value to the self-attention layer as described in [234].	62
Figure 4.4	Effect of Action Instance Durations (THUMOS14). Analysis of segmentation error (L1 loss) with respect to the duration of corresponding ground truth instances. All the values are normalized with respect to the overall video duration. We observe that the action instances of longer durations have lower segmentation errors in their predictions.	74
Figure 4.5	Visualization: Predictions (THUMOS14). Visualization of predictions and groundtruth action instances	75
Figure 4.6	Visualization: Predictions (Epic-Kitchens100). Visualization of predictions and groundtruth action instances	75
Figure 4.7	Visualization: Predictions (ActivityNet-1.3). Visualization of predictions and groundtruth action instances	76
Figure 4.8	Visualization: Predictions (Charades). Visualization of predicted and groundtruth action instances.	77

Figure 4.9	Visualization: Learned Graphs.	Visualizations of embeddings corresponding to the last layer of the decoder and ground truth instances. The thickness of edges show the strength of interaction between the nodes. For ease of visibility, the nodes have been numbered based on the order of their predictions sorted with respect to the start time (<i>i.e.</i> , node 0 represents the instance that starts first). These visualizations demonstrate that the model indeed learns non-linear dependencies between the action instances in a video. The legend below each figure shows the action labels corresponding to the color coded elements.	78
Figure 5.1	Long-Term Action Anticipation.	Given the initial portion of an activity video $(0, \dots, T_o)$ and anticipation duration T_a , the task is to predict the actions that would occur from time $T_o + 1$ to $T_o + T_a$. Our proposed anticipation model receives the observed video and the anticipation duration as inputs and directly predicts a set of future action instances. Here, the action anticipation is <i>long-term</i> – both the observed duration T_o and the anticipation duration T_a are in the order of minutes.	81
Figure 5.2	Learning Approach.	ANTICIPATR uses a two-stage learning approach. In the first stage, we perform segment-level training (refer to Sec 5.3.1). Given a segment as input, we train a segment encoder to predict the set of action labels that would occur at any time after the occurrence of the segment in the activity video. In the second stage, we perform long-term action anticipation (refer to Sec 5.3.2). We use video encoder to obtain video-level representation and segment encoder (trained in the first stage) is used to obtain segment-level representation. The anticipation decoder receives these two representations of the observed video to directly predict a set of action instances that would occur in the future over a given anticipation duration.	85

Figure 5.3	<p>Model Architecture. Our model comprises three networks: <i>segment encoder</i>, <i>video encoder</i> and <i>anticipation decoder</i> and is trained for long-term action anticipation in two stages. (<i>left</i>) Segment-level training (Sec. 5.3.1): The segment encoder receives a segment as input and predicts a set of action labels that would occur at any time in the future (after the occurrence of segment in the video). (<i>right</i>) Action Anticipation (Sec. 5.3.2): The video encoder encodes the observed video to a video-level representation. Concurrently, the video is divided into a sequence of segments and each segment is fed into the segment encoder (trained in first stage)The anticipation decoder receives the two representations along with an anticipation duration as inputs to directly predict a set of future action instances over the given anticipation duration. [MH Attention: Multi-head Attention, FFN: Feed Forward Network.]</p>	86
Figure 5.4	<p>Detailed Architecture. Architecture overview of (a) Segment encoder, (b) Video encoder, and (c) Anticipation Decoder. ‘Q’,‘K’,‘V’ are query, key and value to the self-attention layer as described in [234].</p>	90
Figure 5.5	<p>Evaluation. Visualization showing the outputs from our ANTICIPATR and the corresponding timeline obtained after postprocessing to obtain a sequence of action labels for evaluation.</p>	95
Figure 5.6	<p>Analysis. Quantitative evaluation of the anticipation performance of ablated versions of ANTICIPATR. [SE: segment encoder; VE: video encoder].</p>	96
Figure 5.7	<p>Visualizations (Breakfast). Examples from Breakfast dataset for the case where observation duration is 20% of the video duration and anticipation duration involves predicting actions for 50% of the remaining video.</p>	102
Figure 5.8	<p>Visualizations (50Salads). Examples from 50Salads dataset for the case where observation duration is 20% of the video duration and anticipation duration involves predicting actions for 50% of the remaining video.</p>	102
Figure 5.9	<p>Visualizations (EK-55). Examples from Epic-Kitchens-55 dataset for the case where observation duration is 50% of the video duration. We show the predicted action classes in the visualization – classes in green color are correct predictions, classes in red color are wrong predictions, and classes in gray color are missed classes.</p>	103

Figure 5.10 **Visualizations (EGTEA+)**. Examples from EGTEA Gaze+ dataset for the case where 50% of the video is observed. We show the predicted action classes in the visualization – classes in green color are correct predictions, classes in red color are wrong predictions, and classes in gray color are missed classes. 103

Chapter 1

Introduction

Our world is an endless narrative of things we continually perceive. Today’s Artificial Intelligence (AI) systems aim to comprehend this narrative via videos. The research area of video understanding deals with a set of wide variety of problems focused on automatically extracting useful information from videos.

Human activities are ubiquitous in our perception of the world ranging from the daily activities we do ourselves to digital multimedia we consume via our devices through to our interactions with our surroundings. Therefore, understanding human activity videos play a major role in our pursuit to automatically understand videos. Moreover, modeling humans and interactions is crucial for developing practical assistive AI technologies such as robotic systems, augmented reality applications, and effective surveillance capabilities. Motivated by the diversity of human activities and their applicability to several AI systems, activity understanding is considered to be a key research area in the field of Computer Vision.

Human activity videos often depicts activities that include human(s), objects, and various types of intentional and/or unintentional interactions among them. Thus, we posit the answer to understanding activities lies in modeling videos as *compositions* of different elements that contribute to activities depicted in videos. In this dissertation, we explore the idea of compositional learning for activity understanding and propose compositional models for a diverse set of tasks aimed at understanding human activities in videos.

1.1 Activity Understanding

Over the last decade, there has been growing research interest in building deep neural networks for activity understanding. Early efforts in this area focused on the fundamental task of action recognition which involves identifying action(s) depicted in a given video. Facilitated by availability of large scale datasets [1,19,22,44,72,121,179,219] and advances in deep neural networks [76,115,127,128,234], tremendous progress has been made in designing highly accurate and efficient action recognition models [8,22,30,52,67,82,108,135,214,244,249,291] that operate over short videos of duration of a few seconds. While the progress on

action recognition is promising, the models for this task are trained with videos that are short-form and relatively simple – they often depict a single action (or a few actions) in environments that have no involvement in the action.

The progress in recognition models over several years piqued great research interest in developing benchmarks and models for longer activity videos. Inspired by the advances in sequence modeling and graph modeling techniques, several approaches [92, 93, 98, 143, 243, 244, 248, 250, 262, 278] focused on modeling longer videos as a sequence or graph of segment-level features obtained from a pretrained recognition model. These methods have been developed using datasets [78, 101, 120, 131, 134, 213, 287] containing long-form activity videos spanning a few minutes. Some common benchmark tasks in this direction include temporal action detection, temporal segmentation, and spatio-temporal localization. Despite the success of these methods in capturing long-range dependencies, they were designed for videos depicting scripted activities and often contained irrelevant backgrounds in the scenes making them relatively less suitable for practical systems.

Recent efforts in activity understanding have focused on enhancing the datasets by collecting unscripted human activities conducted in real-world environments resulting in challenging datasets such as Epic-Kitchens [35, 36] and Ego4D [75]. These datasets consist of videos that depict unscripted activities spanning longer time durations of up to a few hours. The videos have multiple overlapping actions involving several objects that interact with the environment or background. Furthermore, these datasets contain multiple types of annotations in videos such as spatio-temporal annotations, segment-level annotations, gaze, and 3D maps. Therefore, relevant annotations corresponding to different granularities of information in videos calls for rethinking videos as compositional entities. Encouraged by the recent developments in large-scale activity video datasets and the promising history of AI algorithms for activity understanding, this dissertation focuses on exploring how the notion of compositional learning can be used to understand complex activity videos.

Another class of work in activity understanding focuses on designing benchmarks and approaches that leverage several modalities of data in videos (*e.g.*, audio and natural language descriptions) [59, 201, 202, 225] and developing benchmark tasks aimed at cross-modal understanding of videos such as dense captioning [104], videoQA [133], video-text retrieval [29], text-guided moment retrieval [129, 159, 281], and speaker diarization [59]. In this dissertation, we focus our efforts on designing compositional algorithms that are solely based on visual information in activity videos.

1.2 Compositional Models in Computer Vision

Compositionality – the ability to break down the world into smaller parts, comprehend the parts and construct flexible compositions of these familiar parts – is a vital feature of human brain [55, 151]. For instance, a sentence is composed of a series of words. When

presented with a new sentence, humans can decompose it into familiar words to understand the sentence. Similarly, human brain can perceive scenes as combinations of multiple familiar objects, objects as aggregations of simpler pieces, events as sequences of known actions. This suggests that smaller structures can be learned from finite amounts of data enabling our learning algorithms to reason over infinite unseen compositions of the learned smaller structures. Therefore, compositionality serves as a key design principle of machine learning algorithms.

There is a rich history of compositional modeling in Computer Vision. Earlier research efforts include part-based [53, 54] or grammar-based [102, 294] approaches for static scenes. Recent compositional techniques to model static images propose semantics-driven representations of (static) scenes such as combinations of pixels [249], scene graphs [103, 119, 146, 264], bounding boxes [69, 81, 85], semantic segmentation masks [85, 144], and class activation maps [217, 292]. Compositional modeling approaches have also been shown to be effective in modeling 3D shapes using primitive geometric shapes [42, 132, 204]. Moreover, effective compositional representations of 3D scenes such as combinations of 3D objects [122], and corresponding 2D object layouts [207] in conjunction with the scene attributes (*e.g.*, pose, depth and layout) [231] have been employed to develop methods for 3D scene understanding.

Building upon the advances in compositional modeling for static scenes, several research efforts aim at designing compositional models for dynamic scenes, *i.e.*, videos. As videos are a sequence of frames, frame-level compositional modeling is a natural choice. Preliminary modeling approaches in this direction [41, 91, 232, 237] are based on decomposing videos at a frame-level resulting in disentangled representations of videos that depicts the object in the frames as *content* (*e.g.*, the person in a video with a man performing different poses) and the change in the content/object over time as *motion* (*e.g.*, pose of the person). These methods utilize independent latent spaces corresponding to motion and content [41, 232], use image residuals to model motion [237], or employ shared latent vectors among frames corresponding to content and independent ones corresponding to motion [91]. While these disentanglement driven modeling techniques demonstrate encouraging results, these approaches are limited to scenes [13, 20, 219] that have strictly linear temporal ordering of frames and involve a single object or entity in the scene.

Realistic videos are more complex in that they have multiple actions that have a weak temporal ordering, involve multiple objects that interact with each other, and have (dis)appearing objects as the scene progresses [22, 36, 72, 120, 213, 268]. The challenging properties of realistic, complex videos have motivated recent research attempts to leverage semantically meaningful disintegration of videos such as spatio-temporal region graphs around objects or persons involved in a dynamic scene [95, 250]; combinations of shorter segments each representing a primitive action in a complex activity [92, 93, 162]; and spatio-temporal scene graphs solely providing semantic relations of objects and primitive actions depicted in a complex video [100, 230]. Such representations of videos rely on different units

to represent the visual information in videos. For instance, a spatio-temporal scene graph corresponding to a video consists of nodes representing label of object present in the video as a unit and a sequence of shorter video segments would have a video segment as a unit. Hereafter, for generality, we refer to a unit in the compositional representation of a video as an *element*.

Designing compositional algorithms offers advantages for all aspects of our learning algorithms ranging from learning representations to inference using a learned model. **(1 - Generalizable models)** By learning smaller structures during training, these models enable reasoning over compositions of learned substructures that are unseen during training and therefore, can generalize well during inference. **(2 - Structured learning)** Compositional designs allow structured modeling of data, thereby, providing a richer understanding of how different elements in the input data interact. **(3 - Generic representations)** They equip models to reason beyond the limited context of the input – they allow models to capture valuable generic information at element-level that is common across the inputs from a given dataset. For instance, consider a dataset that contains multiple recipes that use ‘tomato’ as an ingredient. While the ingredient ‘tomato’ at different stages based on the recipe, certain pieces of information are generic in that they are common across the recipes such as usage of tomato as ingredient, color of tomato, action of cutting tomato. Such element-level information – pertaining to the element (*i.e.*, ingredient) – captures valuable information across recipes regardless of the context provided by any individual recipe.

Despite the promising success of the existing research efforts, compositional modeling for human activity videos remains a challenging and underexplored area. Existing works adopt compositional representations based on the visual information in the scene to encode the video as a whole. While such representations are based on compositional elements, modeling techniques employed to model relations between these elements focus on processing the video as a whole. These approaches do not utilize element-level information and therefore, miss out on leveraging the generic information that is common across different videos at element-level. Such information can potentially be incorporated in the modeling approaches by reasoning over individual compositional elements in addition to the overall video. Furthermore, these methods employ task-agnostic elements while composing a video – the compositional elements are designed based on intuitions about the input and are not necessarily informed of the downstream tasks. It is potentially valuable to imbue task-specificity in the choice of elements that compose a video. This opens up an untapped opportunity to design approaches using different levels of information in videos to learn *task-specific* compositional representations that are informed of the downstream tasks.

In this dissertation, we ask the following questions: (1) Can compositional reasoning of videos be improved by leveraging element-level information in addition to video-level information?, (2) How to design models that effectively incorporate different levels of compositional information in videos?, and (3) Do task-specific compositional representations,

learned using different levels of information in videos, improve the capability of models to perform downstream tasks over complex human activity videos?

1.3 Contributions

This dissertation contributes to the broader research direction of compositional learning for activity understanding. We explore different aspects of compositionality in activity videos to design methods for a diverse set of tasks aimed at understanding human activities. Our contributions are as follows.

- **Learning to Generate Human-Object Interactions [165].** Video generation in realistic scenes is a challenging and open problem. While realistic video generation is a broad research area, we focus on generating human-object interaction (HOI) videos. This is an interesting problem aimed at explicitly modeling interactions between human and objects involved in a realistic scenes and therefore, is an important milestone in the path towards developing methods for activity understanding. To this end, Chapter 3 introduces a novel task of generating HOI videos in a zero-shot compositional setting, *i.e.*, generating videos for action-object compositions that are unseen during training, having seen the target action and target object separately. This setting allows us to study generalization in human activity video generation. It avoids the need to observe every possible action-object combination in training and calls for design of methods to generate videos corresponding to action-object combinations unseen during inference. Thus, we design a novel adversarial framework to generate HOI videos that adopts a task-specific compositional approach to focus on different granularities of information in a video. Overall, this work investigates how compositional models unlock generalization capabilities in generative models for human-object interaction videos and how task-specificity in model designs enhances the ability of methods to generate human-object interactions in complex scenes.
- **Learning Activity Graphs for Temporal Action Localization [164].** The task of temporal action localization involves detecting and localizing actions in untrimmed videos. Given a human activity video, the goal is to predict action instances in the video, *i.e.*, categorical labels, starting, and ending timestamps corresponding to actions that occur in the video. Videos depicting realistic activities contain non-linear temporal structure in the form of: (1) non-linear temporal ordering such as overlapping action instances or re-occurrence of action instances over the course of the video (*i.e.* input), and (2) non-sequential dependencies in the action instances (*i.e.* output). Based on these observations, Chapter 4 proposes a method that encodes the non-linear temporal structure in activity videos and the non-sequential nature of the output space pertaining to the task of temporal action localization. To this end, this

chapter introduces Activity Graph Transformer (AGT) that reasons over the videos and the output set of action instances as non-sequential entities in the form of graphs. AGT is an end-to-end learnable model that receives a video as input and directly predicts a set containing all the action instances that appear in the video. Overall, this work explores the advantages of designing task-specific compositional representations and probes into how compositional learning has benefits for structured modeling of human activity videos.

- **Learning Segment-Level Representations for Action Anticipation [163].** Action anticipation involves predicting future actions having observed the initial portion of a video. While a specific sequence of actions (*i.e.*, segments of a video) in the given portion of the activity video provides temporal context of the ongoing activity, an individual video segment (containing a single action) alone contains generic cross-activity information that is valuable for predicting the future. To capture this relevant segment-level information, Chapter 5 introduces a compositional learning approach ANTICIPATR that leverages segment-level representations in addition to a video-level representation for long-term action anticipation. Our approach trains a transformer-based model in two stages – the first stage involves learning of segment-level representations using individual segments from different activities and the second stage uses the segment-level and video-level representations to directly predict a set of future action instances over any given anticipation duration. Furthermore, to capture segment-level information that is relevant to the downstream task, the segment-level representations are trained on segment-level action anticipation in the first stage. Overall, this work demonstrates the ability of compositional models to enable learning of generic element-level representations for human activity videos. This work also highlights the benefits of learning task-specific element-level representations at different granularities of activity videos.

Chapter 2

Related Work

In this chapter, we discuss the prior work on compositional modeling in videos. We first discuss the existing methods in this direction in Section 2.1 followed by a discussion on the application of compositional models to a diverse set of video-based Computer Vision tasks in Section 2.2. The figures in this chapter are used for illustration purposes and are taken from the original papers.

2.1 Compositional Representations for Videos

In this section, we discuss the compositional modeling approaches in videos focusing on representations of the videos as compositions of its elements and techniques employed to design these representations.

Specifically, we review the canonical compositional representations of videos that have been employed in the prior works on modeling videos in the field of Computer Vision. These representations are referred to using indices R1-R7 listed as follows.

- R1: Videos as spatio-temporal region graphs
- R2: Videos as sequences of unit-actions
- R3: Videos as graphs of unit-actions
- R4: Videos as compositions of spatio-temporal scene graphs
- R5: Videos as topological graphs
- R6: Videos as compositions of trajectories
- R7: Videos as spatio-temporal graphs of visual concepts

Now, for each of these aforementioned representations, we provide a detailed discussion of an existing method that uses the corresponding representation as the underlying compositional representation to model videos.

R1. Videos as spatio-temporal region graphs. Wang and Gupta [250] represent a video as a spatio-temporal region graph wherein a node in the graph depicts a region of

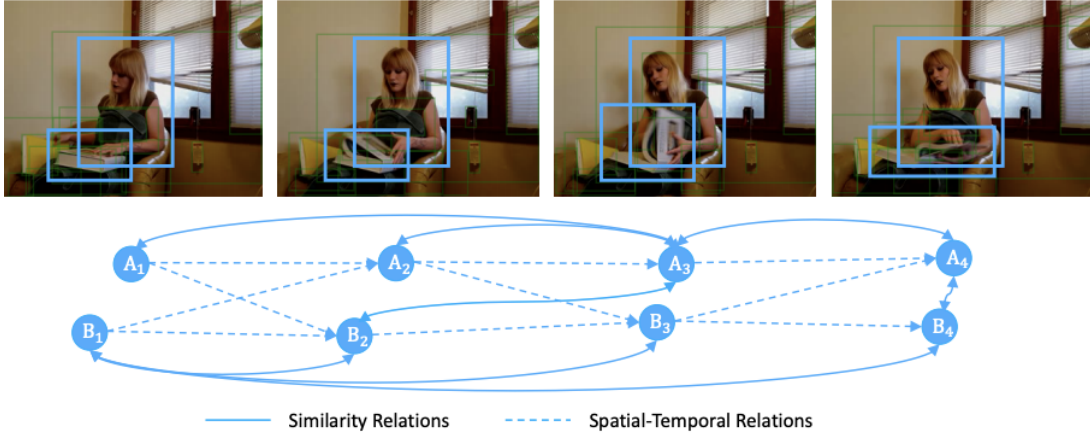


Figure 2.1: **Videos as spatio-temporal region graphs.** Illustration of the representation of a video as a spatio-temporal region graph.

interest in the video (as illustrated in Figure 2.1). This paper uses the class-agnostic object proposals obtained using a region proposal network (RPN) [186] as the regions of interest. In addition, the full video is divided into consecutive clips each containing fixed number of frames to obtain video features from the penultimate layer of a 3D convolutional model, namely, I3D (pretrained on Kinetics dataset [22] and finetuned for the dataset in consideration). Subsequently, to obtain the final region features, i.e., node representations of a fixed size, the paper uses RoIAlign operation [85] to project each of the object proposals on the corresponding video features derived from the appropriate clip.

The nodes in the spatio-temporal region graphs are connected with two types of relations incorporating: (1) similarity in visual appearance of the regions referred to as similarity relations, and (2) spatial overlap and temporal proximity in regions referred to as spatio-temporal relations. The similarity relations capture the long and short term dependencies in videos based on the similarity in the visual information of the nodes. On the other hand, spatio-temporal relations capture the relative spatial relations between objects and relative ordering of the change in state of the objects. To ascertain similarity relations, the paper defines the inner product between two different transformations of two nodes as a measure of similarity between them. The similarity score is computed between all pairs of nodes and is higher when nodes are semantically similar. The normalized similarity scores form the overall adjacency matrix referred to a similarity graph. To establish spatio-temporal relations, this paper uses intersection over union (IoU) as a measure of proximity and computes a forward and a backward spatio-temporal graph. The $(i, j)^{\text{th}}$ element of the adjacency matrix in the forward spatio-temporal graph consists of the pairwise IoU between i -th object proposal/bounding box at any time t and j -th object proposal at time $t + 1$, whereas, $(i, j)^{\text{th}}$ element for the backward spatio-temporal graph denotes IoU between i -th object proposal/bounding box at any time $t + 1$ and j -th object proposal at time t .

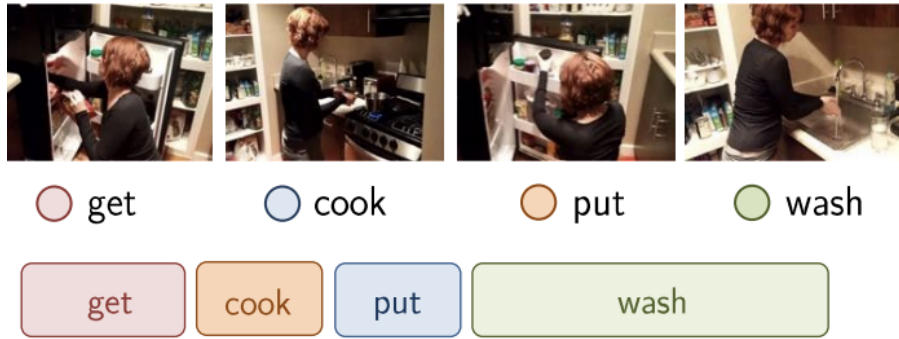


Figure 2.2: **Videos as sequences of unit-actions.** Illustration of the representation of a video corresponding to an activity ‘*cooking a meal*’ as sequence of unit-actions wherein each unit-action has a variable temporal extent.

Finally, the spatio-temporal region graph corresponding to the video is obtained from the combination of the three graphs, namely, similarity graph, forward spatio-temporal graph, and backward spatio-temporal graph. The three graphs are modeled using different Graph Convolutional Network (GCN) [115] modules for similarity graph and spatio-temporal graphs. The output of the GCN modules are summed over to overall aggregate the spatio-temporal information for the video. In summary, this paper proposes to represent videos as spatio-temporal graphs with nodes as regions of interest in the scenes and uses different types of edges/relations to model interactions between the nodes of the graph representing the video.

R2. Videos as sequences of unit-actions. Hussein *et al.* [92] represent a long range video as a sequence of unit-actions (the term originally defined by Keuhne *et al.* [120]) that vary in their temporal extent wherein the events are linked with a temporal order. See Figure 2.2 for an illustration.

To model interactions between unit-actions, this paper introduces a temporal convolutions based module referred to as *Timeception*. Specifically, the Timeception layer in a neural network performs depthwise-separable temporal convolution, *i.e.*, temporal only convolution across the channel dimension of the input tensor. Moreover, in order to capture variable temporal extent of elements, *i.e.*, unit-actions in the video, this paper proposes multi-scale temporal kernels in the Timeception layer. To model cross-channel relations in the overall model, channel grouping is performed before the temporal-only layer and channel shuffling is done after a Timeception layer. To summarize, this paper represents videos as sequences of unit-actions with varying temporal extent and a weak temporal order, and proposes a multi-scale temporal convolutions based model that incorporates long-range dependencies in modeling videos while still capturing short-range details, thereby, effectively modeling relations in the compositional elements in long range videos.

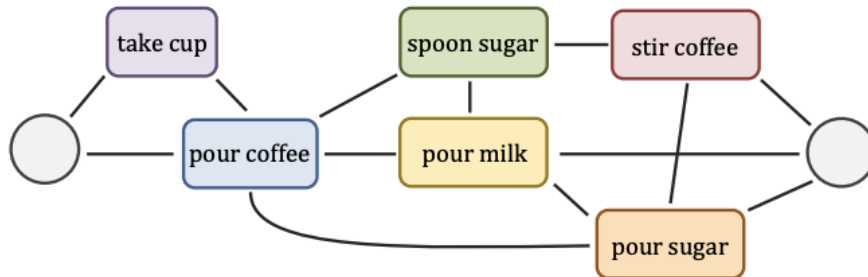


Figure 2.3: **Videos as graphs of unit-actions.** Illustration of the representation of a video depicting the activity of ‘*preparing coffee*’ as an undirected graph of unit-actions.

R3. Videos as graphs of unit-actions. Hussein *et al.* [93] represent a long range video as an undirected graph of short-range unit-actions (the term originally defined by Keuhne *et al.* [120]). Please refer to Figure 2.3 for an illustration. This representation is based on the intuition that the unit-actions in long range videos are related to each other, however, do not necessarily have a sequential relationship.

In this paper, the graph representation of the video consists of nodes representing latent concepts corresponding to the key unit-actions in the video. These nodes are connected with edges capturing the temporal relations between these unit-actions. This paper proposes to break down the video into segments by randomly sampling segments of fixed size. The features corresponding to these segments (obtained using a 3D CNN *e.g.* I3D [22]) are used to learn the nodes and edges for the final video representation. To learn the (fixed) set of latent concepts corresponding to the nodes, this paper employs a node-level attention module that measures the similarity between the features corresponding to the video segments and the latent concepts. This similarity measure is then used to attend to the latent concepts. This attention mechanism enables the algorithm to focus on both individual segments and the video-level context (*i.e.* set of latent concepts).

To learn the edges and to obtain the final structure of the graph corresponding to the video, this paper proposes a graph embedding layer that models two types of relations: (1) relationships between the nodes, and (2) temporal transitions of nodes of the graph. The relationships between the nodes are learned by performing nodewise convolutions on the combined vector representations of all video segments (the output of the node-attention module). The temporal transitions are learned by performing convolutions only on the temporal channel of the combined video representation. In addition, to model the cross-channel relations, the channelwise convolutions are also performed. In summary, this paper performs compositional modeling in videos by representing a video as an undirected graph of unit-actions that are denoted by latent concepts to obtain effective representations for long range videos.

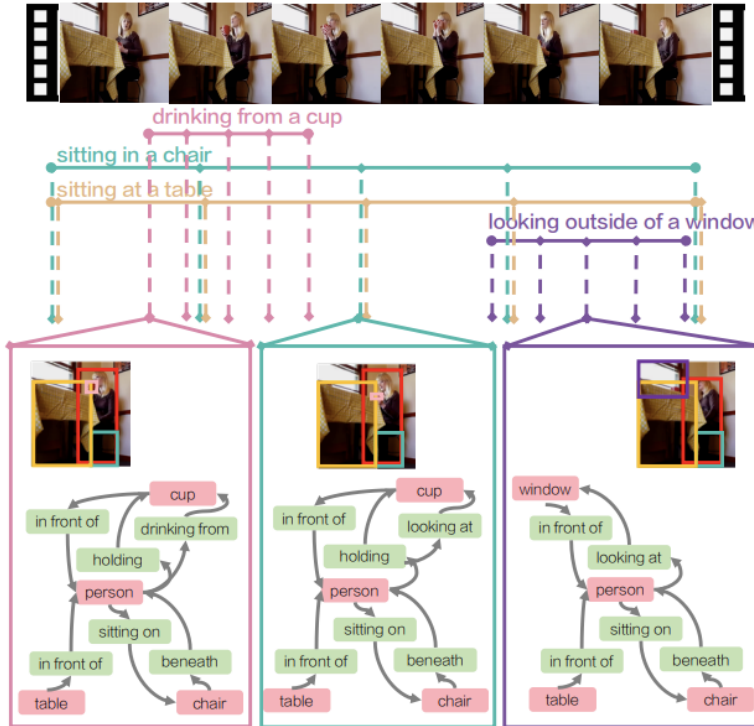


Figure 2.4: **Videos as compositions of spatio-temporal scene graphs.** Illustration of the representation of a video as composition of spatio-temporal scene graphs.

R4. Videos as compositions of spatio-temporal scene graphs. Ji *et al.* [100] represent videos depicting egocentric complex actions as compositions of spatio-temporal scene graphs (see Figure 2.4 for illustration). This representation is based on the observation that complex action videos have hierarchical structure. This paper introduces a large scale dataset Action Genome containing annotated decompositions of actions in videos into spatio-temporal scene graphs comprising prototypical action-object couplets.

Ji *et al.* represent a node in a scene graph as a label corresponding to the category of the object involved in the action. The nodes are connected with edges depicting action labels (referred to as relationships in the paper). This paper categorizes the original action labels provided by Charades [213] dataset (scene graph annotations provided for videos in Charades) into 3 types of relationships: attention, spatial and contact. The attention relationships determine which objects the person in the scene is looking at (*e.g.* ‘*looking at*’, ‘*unsure*’). The spatial relationships are based on the spatial layout of objects (*e.g.* ‘*behind*’, ‘*in front of*’). The contact relationships indicate objects in the scene are manipulated by the person in the scene (*e.g.* ‘*eating*’, ‘*wearing*’, ‘*carrying*’). In summary, this paper introduces the notion of representing videos as compositions of spatio-temporal scene graphs to model complex human activity videos.

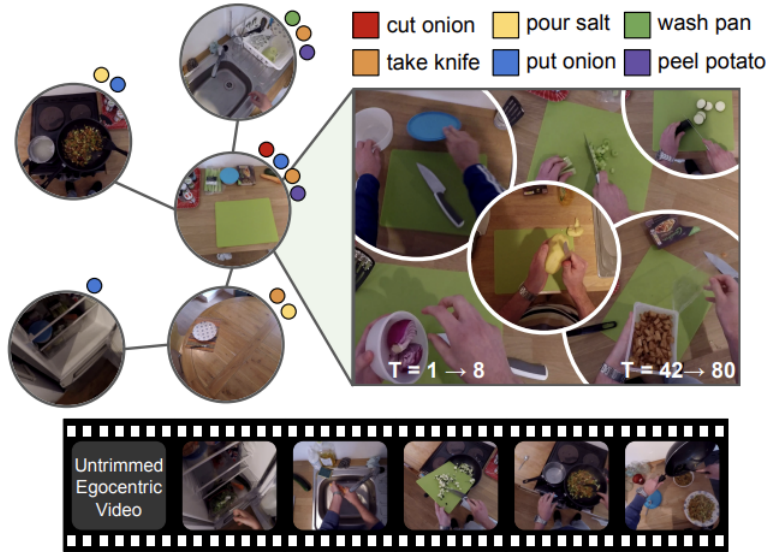


Figure 2.5: **Videos as topological graphs.** Illustration of the representation of a video as a topological graphs of regions that afford coherent actions.

R5. Videos as topological graphs. Nagarajan *et al.* [162] represent egocentric long range videos as topological graphs. In this paper, the topological graphs consists of nodes representing activity ‘zones’ defined as regions of the environment in the scene that afford a coherent set of interactions.

This paper designs the zones by combining the frames that have common physical space and have same functions afforded by a zone. Each node in the graph depicting a zone maintains a collection of ‘visits’, *i.e.*, smaller clips from the video at the location represented by the zone. Please refer to Figure 2.5 for illustration. This paper proposes to train a localization network based on visual similarity and consistency in homographies of frames to discover commonly visited spaces, *i.e.*, zones in the video. The localization network is trained to determine whether a given pair of views contain similar or dissimilar views. Subsequently, this paper employs this (trained) localization network to construct the topological graph corresponding to an unseen unlabeled video.

The overall graph construction process begins with the graph being initialized with a single node containing the first frame of the video which evolves based on the similarity scores (computed using the localization network) between a subsequent frame and the frames in nodes that are already present in the graph. Specifically, if the localization network indicates a high similarity between a subsequent frame and the nodes, then the frame is merged with the most similar node, otherwise a new node is created with an edge connecting the new node with the previously visited node.

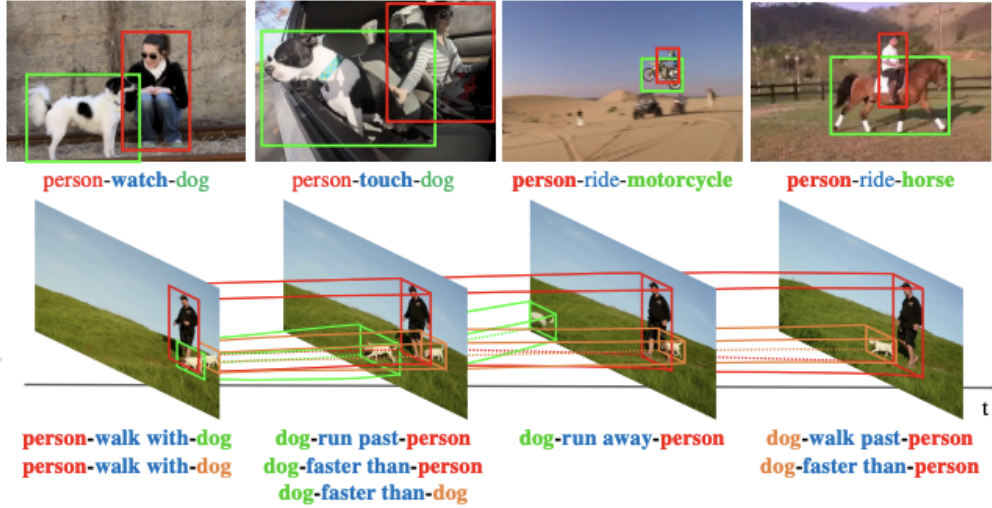


Figure 2.6: **Videos as compositions of trajectories.** Illustration of the representation of a video as a combination of *subject* and *object* trajectories connected with a *predicate*.

Additionally, Ji *et al.* also propose to establish links between zones based on a functional similarity score. They use a pretrained action/object classifier to obtain the distribution of actions and (active) objects that appear in all visits to a node. The functional similarity score between all pairs of nodes in the graph is computed as the KL-divergence of the distributions corresponding to these nodes. Finally, the functional similarity based links are obtained using hierarchical agglomerative clustering. In summary, this paper proposes to represent egocentric videos as topological graphs consisting of nodes depicting zones, *i.e.*, regions that afford coherent actions and edges connecting the zones based on the similarity in likely actions that can be performed in the zones.

R6. Videos as compositions of trajectories. Shang *et al.* [203] represent human activity videos as compositions of trajectories corresponding to *subject* and *object* (involved in an action) linked with a *predicate*. Refer to Figure 2.6 for illustration. Specifically, a human activity video is decomposed based on the observation that a visual entity *subject* (*e.g.* ‘person’, ‘dog’) performs an action described by *predicate* (*e.g.* ‘touch’, ‘walk with’) on another visual entity called *object* (*e.g.* ‘dog’). This paper presents the first dataset containing human annotations for trajectories and category labels of *subject* and *object* along with the labels for *predicate* for 1000 videos from Imagenet-Video dataset [192]. This dataset spans over 35 *subject* categories and 132 *object* categories. In summary, this paper introduces the notion of representing human activity videos as compositions of trajectories of *subject* and *object* related with a *predicate* to reason over the visual entities that appear in the videos.

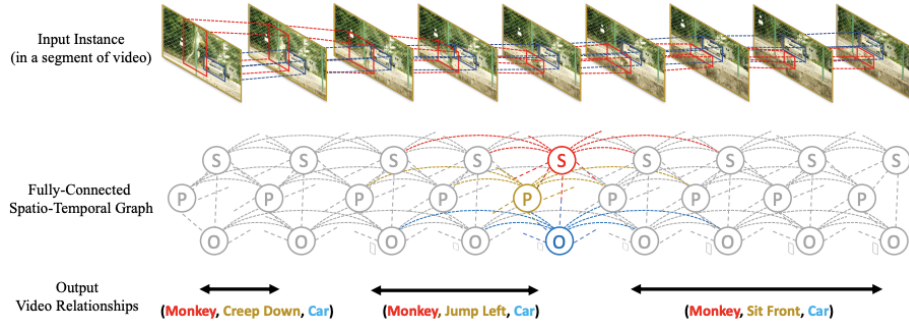


Figure 2.7: **Videos as spatio-temporal graphs of visual concepts.** Illustration of the representation of a video as a fully connected spatio-temporal graph of visual concepts (subject ‘S’, object ‘O’, predicate ‘P’ in the figure).

R7. Videos as spatio-temporal graphs of visual concepts. Tsai *et al.* [230] represent human activity videos as a fully connected spatio-temporal graphs of visual concepts associated with videos. The spatio-temporal graph is defined for a predefined set of visual concepts based on the domain of videos. For instance, *subject, object, predicate* are reasonable visual concepts for human activity videos containing pair of objects [192] and *subject, object, scene* are potential visual concepts for human activity videos with rich and diverse background scenes [213]. Refer to Figure 2.7 for an illustration. This paper proposes to construct the spatio-temporal graph with a node for each visual concept for each frame and all nodes are connected with each other across space and time using weighted edges. The edge weights are derived from learnable, data-dependent functions and are adaptive to the visual information in videos as well as the corresponding visual concepts. To summarize, this paper suggests that representing videos as fully-connected spatio-temporal graphs over visual concepts is an effective means to perform compositional modeling in videos.

Overall, in this section, we discussed various methods that focus on compositional representations of videos. These approaches vary in the type of elements that compose the representations such as (1) only visual elements (image regions in R1 [250], video segments depicting unit-actions in R2 [92] and R3 [93]); (2) only semantic elements (visual concepts pertaining to a video in R7 [230], category labels for actions/objects in a video in R4 [100]); (3) a combination of visual and semantic entities (trajectories linked with labels in R6 [203]); and (4) a combination of visual, semantic and temporal entities (zones depicted by regions (visual) affording coherent actions (semantic) maintaining a record of visits to the regions (temporal) in R5 [162]). Moreover, the approaches discussed in this section are designed for videos with different temporal extents ranging from short video clips (R1, R6, R7) to long range videos (R2, R3, R4, R5). Furthermore, some of these compositional models are specifically designed for egocentric videos (R5) or videos containing pairs of visual entities (R6, R7).

The limitations in the prior work on compositional modeling in videos expose interesting questions related to the impact of domain-specific assumptions on the generalizability of the modeling approaches. Furthermore, the wide variety in the existing compositional models necessitate a study of applicability of these models to several Computer Vision tasks depending on the design of the elements that compose video-level representations and the temporal extent of videos that the models can capture.

2.2 Applications

In this section, we discuss the applications of compositional modeling approaches to various video-based tasks in the field of Computer Vision. Based on the desired output of the learning task at hand, we categorize the applications into two groups: (1) *Joint Modeling based Applications* – applications that require aggregated information at video level and model the inter-element interactions jointly, and (2) *Explicit Modeling based Applications* – applications that require explicit modeling of each element and its interactions with other elements as the video progresses over time.

We review various compositional modeling approaches designed for both these types of applications listed as follows. We select the applications that commonly appear in the prior works for compositional video modeling and are of interest to the Computer Vision community. The selected joint and explicit modeling based applications are referred to using indices J1-J5 and E1-E3 respectively listed as follows.

- **Joint Modeling based Applications**

- J1: Action recognition
- J2: Group activity recognition
- J3: Long term action anticipation
- J4: Scene affordance learning
- J5: Video captioning

- **Explicit Modeling based Applications**

- E1: Video prediction
- E2: Video relationship reasoning
- E3: Spatio-temporal scene graph prediction

Next, we discuss several existing compositional modeling techniques designed for each of these aforementioned applications.

Joint Modeling based Applications

J1. Action recognition. Action Recognition is a standard classification task in the field of Computer Vision which involves providing action label(s) for an input video. The effectiveness of action recognition algorithms are typically evaluated on large-scale human activity

datasets containing complex actions involving diverse set of objects. The performance of the action recognition algorithms is ascertained using classification accuracy and mean average precision (mAP) for multi-label scenarios as evaluation metrics. Some datasets that are commonly used in Computer Vision community are listed below.

- **EPIC-Kitchens [36]** contains unscripted, egocentric videos of activities in several kitchens. Each video clip V is annotated with action label a and object label o (*e.g.* cut apple, open microwave). There are $\sim 40k$ training instances of the form (V, a, o) spanning 352 objects (that commonly appear in kitchen environments) and 125 actions.
- **Something-Something [72]** contains videos of basic activities performed by humans with everyday objects. A video clip is annotated with a label, an action template and object(s) on which the action is applied (*e.g.* ‘putting cup on table’ has action template ‘putting something on something’). There are 220,847 training instances spanning 30,408 unique objects and 174 unique action templates.
- **Charades [213]** contains videos of indoor daily activities such as ‘drinking coffee’, ‘putting on shoes while sitting in a chair’. It is a multi-label, action classification, video dataset containing 157 unique activity classes. There are $\sim 10k$ videos spanning 33 categories of verbs and 38 object categories. On average, each video is 30 seconds.
- **Breakfast [120]** contains unscripted cooking-oriented human activities. The average length of videos is 2.3 minutes. It contains 1712 videos in total depicting 12 categories of breakfast activities wherein each video represents only one activity.
- **MultiTHUMOS [268]** contains untrimmed videos of in-the-wild human activities. It contains 65 action classes and 400 videos. Each video contains a complex action which comprises 11 unit-actions on average.

These datasets cover a wide variety of actions ranging from kitchen activities to daily activities and basic actions to complex activities, therefore, they serve as a reasonable proxy for realistic videos. However, each of the datasets possess implicit dataset biases in terms of shapes of objects (*i.e.*, limited to objects available in kitchen or in indoor environments), thereby, limiting the types of actions that can be performed on these objects. The datasets containing unscripted and in-the-wild videos also represent a constrained set of actions. While the current action recognition algorithms are evaluated on these datasets, the implicit bias in the dataset presents an open question on the generalizability of the algorithms in real world settings.

Several research attempts explore compositional modeling in complex human activity videos in order to capture long range dependencies and encode relations between the visual

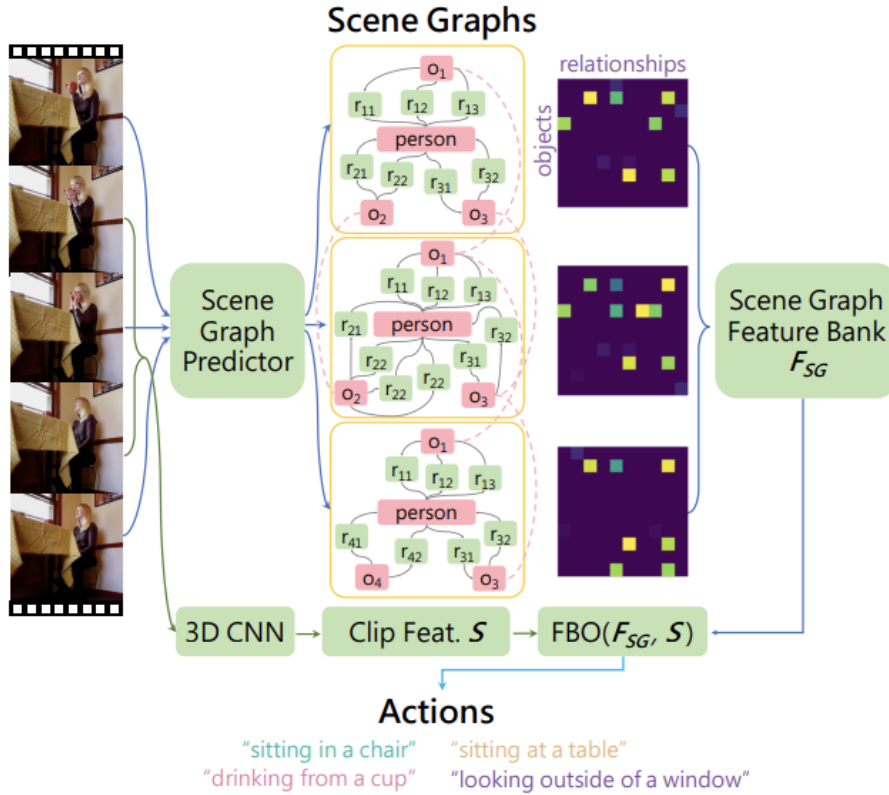


Figure 2.8: **Action recognition.** Overview of the compositional action recognition model (proposed by Ji *et al.* [100]) for human activity videos based on decomposing videos as compositions of spatio-temporal scene graphs.

elements that appear in the videos. Wang and Gupta [250] represent realistic videos (typically having scenes with multiple objects) as spatio-temporal region graphs and employ a Graph Convolution Network (GCN) [115] to model the relations among different image regions in a video. Finally, they aggregate the relational information (using pooling operation) to obtain a video level representation which is then used for classifying input videos. Materzynska *et al.* [154] formulate similar spatio-temporal regions graphs consisting of regions corresponding to only actor and object in videos to perform action recognition. Along the lines of spatio-temporal representations of videos, Ji *et al.* [100] propose to perform action recognition in egocentric videos depicting complex actions by representing videos as compositions of spatio-temporal scene graphs. This paper proposes a model consisting of two modules: (1) frame-level scene graph predictor to obtain scene graphs corresponding to each frame in video, and (2) long term feature bank operator [254] to combine the representations of the predicted scene graphs along with visual features of videos (obtained from 3D CNN *e.g.* I3D [22]) to obtain video level representations. Please refer to Figure 2.8 for an overview.

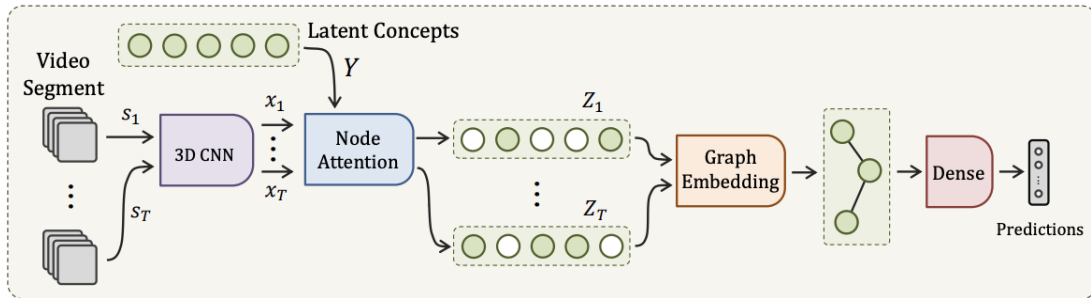


Figure 2.9: **Action recognition.** Overview of the compositional action recognition model (proposed by Hussein *et al.* [93]) for long range videos based on decomposing videos as graphs of unit-actions.

On the other hand, Hussein *et al.* [92, 93] perform action recognition on long range videos by focusing on the temporal variations in the videos. In [92], Hussein *et al.* propose to modularize an input video as a sequence of unit-actions and perform action recognition using a model that consists of temporal only convolutional modules with multi-scale kernels to obtain video level representations. In [93], they represent long range videos as graphs of unit-actions, learn the nodes and edges of the graph corresponding to the video using a node-level attention module, and finally, use multi-layer perceptron to obtain a video level representation for classifying long-range human-activity videos (see Figure 2.9 for an overview).

J2. Group activity recognition. The task of group activity recognition require an activity label as output corresponding to an input multi-person video. Understanding group activities in sports videos is a standard use case of this task, thus, Basketball [183] and Volleyball [95] datasets form a commonly used testbed for group activity recognition algorithms. In addition, in-the-wild multi-person videos such as Collective Activity dataset [31]) are also used for evaluating group activity recognition algorithms. The overall performance of group activity recognition algorithms is determined based on the accuracy of the activity label prediction. The commonly-used datasets are described as follows.

- **Basketball [183]** dataset consists of $\sim 14k$ clips collected from 257 basketball games wherein each clip contains one of 11 group activity labels.
- **Volleyball [95]** dataset consists of 4830 clips collected from 55 volleyball games wherein each clip contains one of 8 group activity labels with only the middle frame of each clip annotated with the bounding boxes and action labels corresponding to the players.
- **Collective Activity [31]** dataset contains 44 short video sequence from 5 group activities (crossing, waiting, queueing, walking and talking) with 6 individual actions

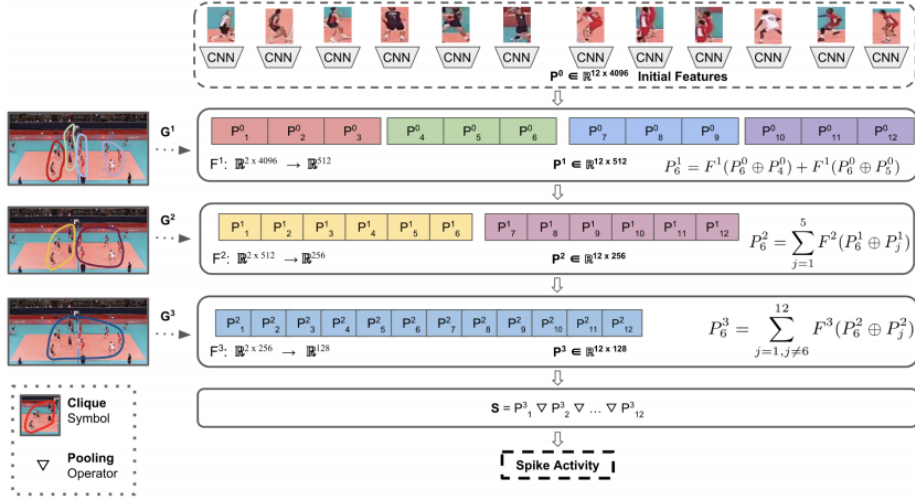


Figure 2.10: **Group activity recognition.** Overview of the group activity action recognition model (proposed by Ibrahim and Mori [94]) for sports videos based on decomposing videos as image regions each containing a person involved in the activity.

(NA, crossing, waiting, queueing, walking and talking) and the group activity label for a frame is defined by the activity in which most people participate.

While these large-scale datasets contain challenging group activities, they still rely on assumptions on how to define a group activity as well as contain limited labels due to sparse annotations corresponding to individual persons involved in an activity.

Several compositional models [94, 95, 183, 212] represent multi-person videos as a compositions of image regions around each person in the scene involved in the group activity depicted in the video. Ibrahim *et al.* [95] propose a hierarchical network using LSTM modules to learn the dynamics of each person as well combine the person-level dynamics to the video-level. Ramanathan *et al.* [183] propose an attention model that attends to key players in a scene depicting group activity. Shu *et al.* [212] employ an object detector to detect persons in the scene and extract their feature representations. They finally connect these representations based on Euclidean distance between the pair of persons and input them to temporal recurrent network along with an energy layer and confidence measure.

Inspired by the advances in relational and graph networks, several research attempts focus on modeling relations between attributes pertaining to the image regions each containing a person involved in the activity depicted in the group activity video. Ibrahim and Mori [94] propose a hierarchical relational networks (inspired by Santoro *et al.* [198]) to encode the inter-person interactions in the scene to finally obtain a video level representations by combining the person-level (relational) representations (refer to Figure 2.10) and demonstrate the effectiveness of the group activity recognition algorithms on sports videos. Wu *et al.* [255] propose GCN [115] based network to model relations between visual entities

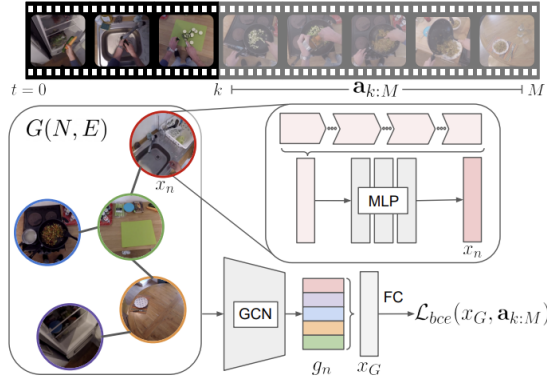


Figure 2.11: **Long term action anticipation.** Overview of the long term action anticipation model (proposed by Nagarajan *et al.* [162]) for egocentric videos based on decomposing videos as topological maps of regions that afford coherent actions.

(*i.e.*, image regions around persons) at a frame level in conjunction with a recurrent network to aggregate the temporal information. Gavriluyk *et al.* [64] use a transformer network to model static and dynamic representations corresponding to each person expressed by features from a 2D pose network and 3D CNN respectively.

J3. Long term action anticipation. The task of long term action anticipation involves prediction of future actions given the past and current observations. To perform long term action anticipation in realistic scenes, Nagarajan *et al.* [162] decompose videos into zones (*i.e.*, regions that afford coherent action) forming topological graphs that maintain a collection of visits to each zone as the video progresses. This paper uses a multi-layer perceptron (MLP) layer to represent each zone, *i.e.*, node. The relational information between these zones are modeled using a Graph Convolution Network [115] followed by another MLP to obtain a combined representation for the given part of the video. These representations are used to predict action labels for the future frames. Refer to Figure 2.11 for an overview. Nagarajan *et al.* demonstrate that compositional modeling is effective by evaluating the task on egocentric video datasets, namely, Epic-Kitchens [36] and EGTEA Gaze+ [134]. EGTEA Gaze+ dataset contains videos of 32 subjects following 7 recipes in a single kitchen, wherein, each video clip depicts a complete dish being prepared (e.g., potato salad, pizza), with interactions (e.g., open drawer, cut tomato). This dataset contains $\sim 14k$ video clips spanning 53 objects and 19 actions. As opposed to EGTEA+, EPIC-kitchens (described earlier in J1) dataset is collected across multiple kitchens and is unscripted. The authors crowd-sourced the annotations for afforded interactions; and collected 1020 instances spanning 75 afforded interactions on EGTEA+, and 1155 instances over 120 afforded interactions on EPIC-Kitchens. They report the multi-label classification performance as mean average precision (mAP) over all action classes. In addition, they also study the effectiveness of



Figure 2.12: **Scene affordance learning.** Example for affordance prediction in egocentric videos obtained by decomposing videos as topological maps of regions that afford coherent actions (using a model proposed by Nagarajan *et al.* [162]) .

the algorithm in low-shot and many-shot settings for rare and frequently occurring action classes in the dataset.

J4. Scene affordance learning. The task of scene affordance learning involves predicting the likely interactions possible in a scene. In the context of egocentric videos, Nagarajan *et al.* [162] define the task of scene affordance learning as predicting the likely human-object interactions in an egocentric video. In this paper, Nagarajan *et al.* highlight the effectiveness of compositional modeling in videos for the task of scene affordance learning. Specifically, they decompose the videos into a topological graphs of zones (regions that afford coherent actions) and maintain record of the visits of scene depicted by the zone in the video. They formulate the task of scene affordance learning as a multi-label classification problem and use the compositional representations as training data by pairing a scene randomly sampled from a zone with its corresponding affordance label.

Nagarajan *et al.* evaluate the task on egocentric video datasets, namely, Epic-Kitchens [36] and EGTEA+ [134] (described in J3). In this paper, the model evaluates the performance of the proposed method using mean average precision (mAP) of the predictions obtained for the likely interactions. In addition, they also study the performance of the algorithm in low-shot and many-shot settings for rare and frequently occurring interaction classes in the dataset (see Figure 2.12 for an example).

J5. Video captioning. Video captioning is a challenging task that requires understanding of visual scenes as well as the scene descriptions. The task of video captioning involves predicting text descriptions given an input video. Compositional method for this task proposed by Pan *et al.* [173] represent input videos as spatio-temporal region graphs (similar to [250]).

This paper proposes a spatio-temporal graph network to model the object interactions in videos. This paper proposes a two-branch model: (1) an object branch that models

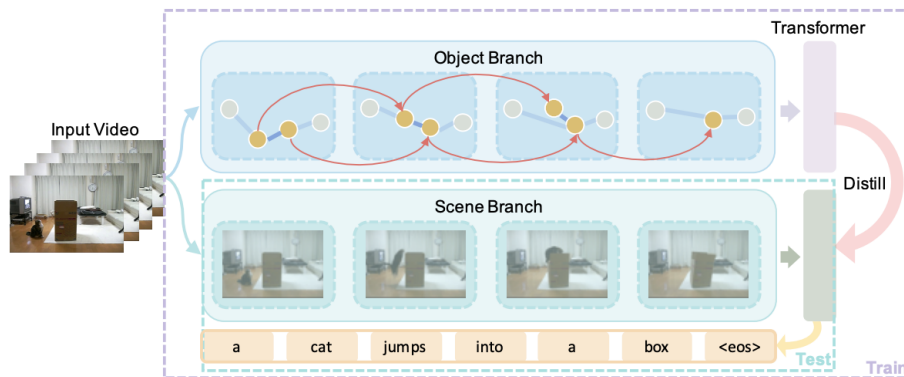


Figure 2.13: **Video captioning.** Overview of the video captioning model (proposed by Pan *et al.* [173]) that decomposing videos as spatio-temporal region graphs and model object interactions in videos.

the spatio-temporal interactions using Graph Convolution Network [115], and (2) a scene branch that provides an overall (visual) information content in videos using any 3D CNN. The two branches are fused using a KL-divergence based knowledge distillation loss from object branch to scene branch. A transformer based module [28] follows both branches to provide the text output. The overall model is trained with the cross-entropy losses for the text output from both branches and a distillation loss. Refer to Figure 2.13 for an overview.

This paper demonstrates the effectiveness of compositional modeling in videos on benchmark datasets, namely, Microsoft Research-Video to Text (MSR-VTT) [261] and Microsoft Video Description Corpus (MSVD) [27] for the task of video captioning. MSR-VTT consists of 10k video clips with each clip annotated with 20 English sentences spanning 20 categories such as sports, gaming, and cooking. MSVD consists of $\sim 2k$ video clips collected from YouTube containing 40 English descriptions per video. These datasets are commonly used for video captioning task, however, generalization to real world videos is limited due to the restricted diversity in the descriptions. This paper evaluates the proposed compositional algorithm using standard video captioning metrics such as BLUE [174], METEOR [10], ROUGE-L [139], and CIDER [235].

Explicit Modeling based Applications

E1. Video prediction. Video prediction is one of the standard tasks in the field of Computer Vision. This task involves predicting a sequence of frames given one (or some) frames as input. Given that the expected output of the algorithm is a sequence of frames, the algorithms require evaluation of the visual quality of the predicted frames. Therefore, metrics such as average ℓ_2 distance or squared error (MSE) between pixel values of the predicted

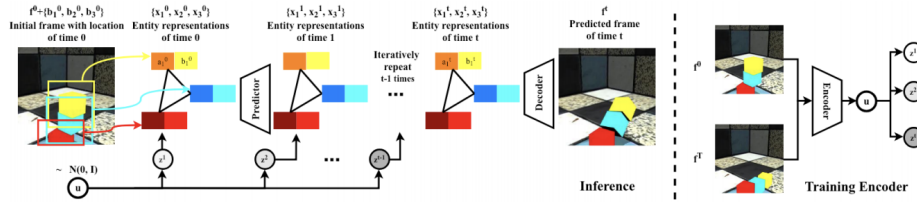


Figure 2.14: **Video prediction.** Overview of the video prediction model (proposed by Ye *et al.* [267]) that models interactions between entities in the videos.

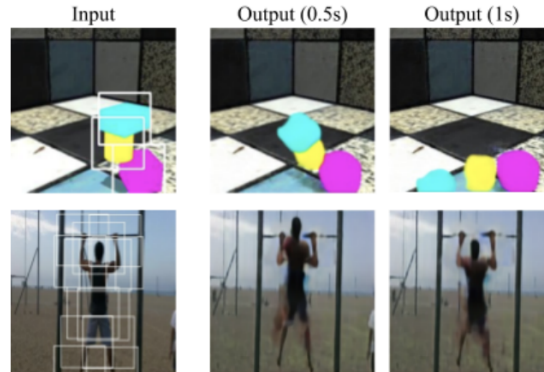


Figure 2.15: **Video prediction.** Examples from the compositional model proposed by Ye *et al.* [267]. The entities that compose the videos are also shown.

and ground truth videos, Peak Signal-to-noise ratio (PSNR), SSIM [252] and LPIPS [283] are commonly used to evaluate video prediction algorithms.

Ye *et al.* [267] demonstrate the effectiveness of compositional modeling for the task of video prediction by employing an object/entity level decomposition of videos. This paper aims at predicting a sequence of frames given a single starting frame along with location of the entities present in the scene. It proposes a stochastic model consisting of two modules: (1) entity predictor that predicts features corresponding to each entities for future timesteps, (2) frame decoder that infers the pixel values for the desired sequence of frames using the entity-level feature predictions. The entity predictor module uses the entity encodings, *i.e.*, entity-level features concatenated with a latent variable as input. The latent variable serves as the global context during prediction and is learned using an encoder based on ground truth video. It predicts the entity features for next timestep taking in the entity encodings for the current timestep using iterative message passing mechanism [115] over the entity-level graph at current timestep. The frame decoder generates the frames using the predicted entity features using a U-Net based convolutional architecture. This paper proposes to train the overall model by maximizing the log-likelihood of the ground truth frame sequence along with an additional loss term to supervise the entity locations (see Figure 2.14 for an overview).

Ye *et al.* evaluate the proposed compositional model on videos on Shapestacks [77] dataset and a subset of Penn Action [284] dataset containing only gym activities. Shapestacks is a synthetic dataset consists of stacked objects that fall under gravity with different objects (such as cubes, cylinders, or balls with different colors) arranged in various configurations. In this paper, the authors denote each of the shapes as entities in the case of Shapestacks videos. Penn Action dataset is a real video dataset of people playing various indoor and outdoor sports with annotations of human joint locations. The authors denote the locations of the pose joints of the person detected in the scene (examples along with entities considered in the video shown in Figure 2.15). Overall, this paper proposes a stochastic compositional model for the task of video prediction that requires explicitly modeling of the interactions between the elements in videos. While the proposed video prediction algorithm is not restricted to these datasets, using compositional algorithms for video prediction task in complex, realistic videos is still an open problem.

E2. Video relationship reasoning. Video relationship reasoning refers to a class of tasks that require understanding of relationships between visual entities that appear in the video. Shang *et al.* [203] propose to represent human activity videos as compositions of trajectories corresponding to *subject* and *object* involved in an action depicted in the videos and connect these trajectories with a *predicate* label. Hence, a human activity video is labeled as a triplet of visual entities, namely, *subject, predicate, and object* depicted in the video. In this paper, the authors also introduce a dataset with human annotations suitable for such compositional representation of human activity videos (described in R6 in Section 2.1). Moreover, Shang *et al.* design several video visual reasoning tasks based on target output pertaining to the elements in the triplets corresponding to the videos. Specifically, they introduce two tasks: (1) *relationship detection*: requires correct subject/object localization and correct prediction for relationship triplet, i.e., $(subject, predicate, object)$, (2) *relationship tagging*: requires correct relationship prediction.

Additionally, Shang *et al.* also propose a method that predicts relation triplets of the form $(subject, predicate, object)$ for smaller segments in videos (created such that there is an overlap between consecutive segments) and uses a greedy association algorithm to merge the short-term detections for the full video. It uses an object tracklet proposal network to obtain tracklet proposals in the form of bounding boxes corresponding to objects detected in the frames of the video segments. These trajectories are processed to extract features that capture both motion and low-level visual information. Finally, these features are used to obtain predictions for relation triplets.

Shang *et al.* use Recall@ K for different values of K and mean average precision (mAP) to evaluate the performance of the model on relationship tagging; and Prec@ K to evaluate the performance of the model on relationship detection. They further evaluate the model in zero-shot compositional setting wherein the relation triplets in test set are unseen during

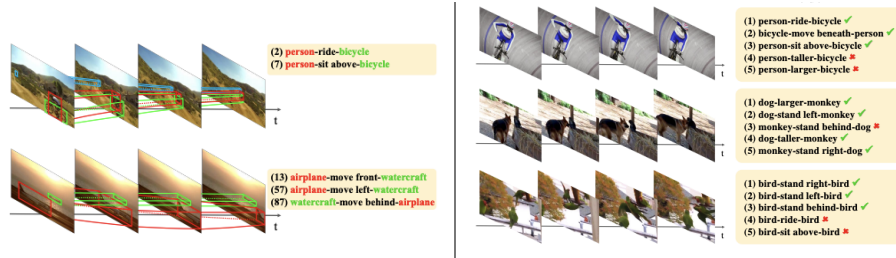


Figure 2.16: **Video relationship reasoning.** Examples of predictions using the model (proposed by Shang *et al.* [203]) for relationship detection (left) and tagging (right) tasks.

training while all the unique elements (*i.e.*, *subject*, *object*, *predicate*) in the triplets are seen individually during training. See Figure 2.16 for examples.

Subsequent to the introduction of dataset, relationship reasoning in videos has gained attention. Dai *et al.* [32] apply conditional random fields (CRFs) on the predictions of individual entities to leverage their statistical correlations. Liang *et al.* [138] propose a deep structured reinforcement learning framework followed by formation of a directed semantic action graph that facilitates predictions in local regions of the image.

Tsai *et al.* [230] extend relationship reasoning in videos to human activity videos with complex scenes, namely, Charades dataset [213] (described in J1) that do not contain annotations for trajectories by representing videos as spatio-temporal graphs of visual concepts ‘*verb*’, ‘*object*’ and ‘*scene*’, wherein ‘*scene*’ describes the environment (*e.g.* ‘bedroom’) in which an action depicted by ‘*verb*’ is performed on the ‘*object*’. This paper formulates the learning task as a structured prediction problem over this spatio-temporal graph variables (*i.e.*, visual concepts in videos) conditioned on the visual information in videos (*e.g.* trajectories of entities in videos [203], and compact features obtained from pretrained 3D CNN [22]). It employs a fully-connected Conditional Random Field (CRF) to model the interactions between the visual concepts in video. It further realizes the edges of the spatio-temporal graph as gating functions in CRF parameterized by the visual information in the videos. The authors validate this approach on different relationship reasoning tasks in large scale video datasets, namely, Charades [213] and Imagenet-Video [192]. They use the same evaluation setup as used by Shang *et al.* [203]. Overall, video relationship reasoning models aim at explicitly modeling each element and its interactions with other elements that compose the videos.

E3. Spatio-temporal scene graph prediction. The task of spatio-temporal scene graph prediction has been recently introduced by Ji *et al.* [100] along with a suitable dataset Action Genome (described in R4 in Section 2.1). The authors define the task as prediction of a spatio-temporal scene graph for a given input video. Inspired by scene graph predictions for images [146], they introduce several tasks to evaluate the effectiveness of the prediction algorithms in capturing different aspects of the videos. Specifically they design three eval-

uation tasks: (1) *scene graph detection* that requires input sequence of images and predicts bounding box locations, object and predicate labels, (2) *scene graph classification* that predicts object categories and predicate labels for input ground truth boxes, and (3) *predicate classification* that predicts predicate labels for input ground truth bounding boxes along with object categories. They use Recall@ K for different values of K to evaluate the performance of scene graph prediction models. Moreover, they benchmark several image-based scene graph prediction models on the task of spatio-temporal scene graph prediction on Action Genome dataset.

Overall, in this section, we discussed the prior work on applications of compositional modeling approaches to various video-based tasks in the field of Computer Vision. The applications discussed in this section vary in the way they model the elements of the compositional representations corresponding to videos depending on the desired output of the task. Specifically, we presented two types of applications: (1) applications that model the inter-element interactions jointly and require aggregated information for a video (J1-J5); (2) applications that require explicit modeling of each element and its interactions with other elements in the representations corresponding to the videos (E1-E3).

In conclusion, the research efforts described in this chapter emphasize the benefits of modeling human activity videos as compositions of elements that represent activities depicted in videos. Additionally, this chapter also highlights that a one-size-fits-all approach is not appropriate for designing compositional representations for videos depicting complex human activities. Instead, the design choices for compositional representations of activity videos depend on several factors: (1) complexity of interactions in activities, (2) level of granularity in annotations provided with the datasets, and (3) downstream task at hand. Based on these observations, this dissertation explores several task-specific compositional representations of human activity videos that leverage information from different granularity levels of videos.

Chapter 3

Learning to Generate Human-Object Interactions

In this chapter, we describe methods for generating human-object interaction videos – making progress towards addressing the important, open problem of video generation in complex scenes. Specifically, this chapter introduces the task of generating human-object interaction videos in a zero-shot compositional setting, *i.e.*, generating videos for action-object compositions that are unseen during training, having seen the target action and target object separately. This setting is particularly important for generalization in human activity video generation, obviating the need to observe every possible action-object combination in training and thus avoiding the combinatorial explosion involved in modeling complex scenes. To generate human-object interaction videos, a novel adversarial framework HOI-GAN which includes multiple discriminators focusing on different aspects of a video is proposed. Moreover, this chapter discusses the findings from our extensive quantitative and qualitative evaluation on challenging datasets to demonstrate the effectiveness of our proposed framework.

3.1 Introduction

Visual imagination and prediction are fundamental components of human intelligence. Arguably, the ability to create realistic renderings from symbolic representations are considered prerequisite for broad visual understanding. Computer vision has seen rapid advances in the field of image generation over the past few years. Existing models are capable of generating impressive results in this static scenario, ranging from hand-written digits [7, 40, 71] to realistic scenes [14, 97, 109, 233, 280]. Progress on *video generation* [11, 84, 196, 232, 239, 245, 246], on the other hand, has been relatively moderate and remains an open and challenging problem. While most approaches focus on the expressivity and controllability of the underlying generative models, their ability to generalize to unseen scene compositions has not received

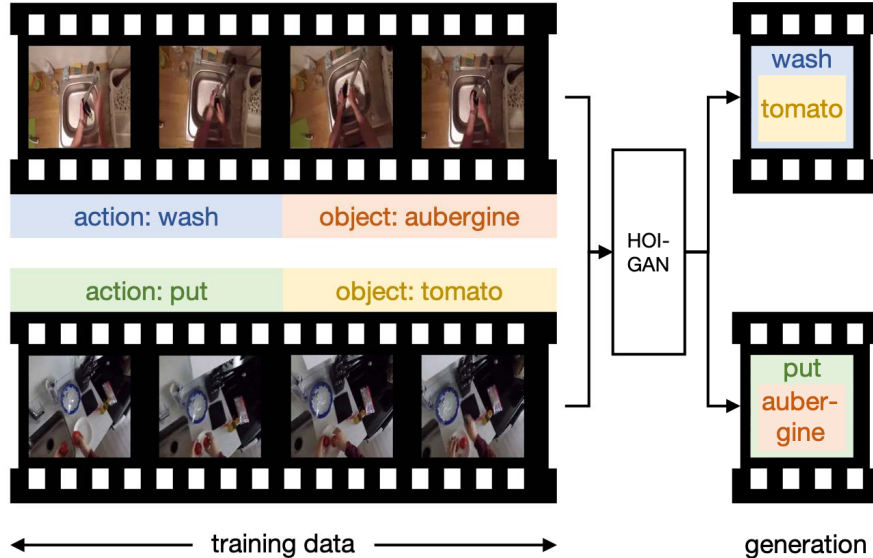


Figure 3.1: **Generation of Zero-Shot Human-Object Interactions.** Given training examples ‘*wash aubergine*’ and ‘*put tomato*’, an intelligent agent should be able to imagine action sequences corresponding to unseen action-object compositions, e.g., ‘*wash tomato*’ and ‘*put aubergine*’.

as much attention. However, such generalizability is an important cornerstone of robust visual imagination as it demonstrates the capacity to reason over elements of a scene.

We posit that the domain of human activities constitutes a rich realistic testbed for video generation models. Human activities involve people interacting with objects in complex ways, presenting numerous challenges for generation – the need to (1) render a variety of objects; (2) model the temporal evolution of the effect of actions on objects; (3) understand spatial relations and interactions; and (4) overcome the paucity of data for the complete set of action-object pairings. The last, in particular, is a critical challenge that also serves as an opportunity for designing and evaluating generative models that can generalize to myriad, possibly unseen, action-object compositions. For example, consider Figure 3.1. The activity sequences for “*wash aubergine*” (action a_1 : “wash”; object o_1 : “aubergine”) and “*put tomato*”(action a_2 : “put”; object o_2 : “tomato”) are observed in the training data. A robust visual imagination would then allow an agent to imagine videos for “*wash tomato*” (a_1, o_2) and “*put aubergine*” (a_2, o_1).

We propose a novel framework for generating human-object interaction (HOI) videos for unseen action-object compositions. We refer to this task as *zero-shot HOI video generation*. To the best of our knowledge, our work is the first to propose and address this problem. In doing so, we push the envelope on conditional (or controllable) video generation and focus squarely on the model’s ability to generalize to unseen action-object compositions. This zero-shot compositional setting verifies that the model is capable of semantic disentanglement of the action and objects in a given context and recreating them separately in other contexts.

The desiderata for performing zero-shot HOI video generation include: (1) mapping the content in the video to the right semantic category, (2) ensuring spatial and temporal consistency across the frames of a video, and (3) producing interactions with the right object in the presence of multiple objects. Based on these observations, we introduce a novel multi-adversarial learning scheme involving multiple discriminators, each focusing on different aspects of an HOI video. Our framework *HOI-GAN* generates a fixed length video clip given an action, an object, and a target scene serving as the context.

Concretely, the conditional inputs to our framework are semantic labels of action and object, and a single start frame with a mask providing the background and location for the object. Then, the model has to create the object, reason over the action, and enact the action on the object (leading to object translation and/or transformation) over the background, thus generating the whole interaction video. During training of the generator, our framework utilizes four discriminators – three pixel-centric discriminators, namely, *frame* discriminator, *gradient* discriminator, *video* discriminator; and one object-centric *relational* discriminator. The three pixel-centric discriminators ensure spatial and temporal consistency across the frames. The novel relational discriminator leverages spatio-temporal scene graphs to reason over the object layouts in videos ensuring the right interactions among objects. Through experiments, we show that our HOI-GAN framework is able to disentangle objects and actions and learns to generate videos with unseen compositions.

In summary, our contributions are as follows:

- We introduce the task of zero-shot HOI video generation. Specifically, given a training set of videos depicting certain action-object compositions, we propose to generate unseen compositions having seen the target action and target object individually, *i.e.*, the target action was paired with a different object and the target object was involved in a different action.
- We propose a novel adversarial learning scheme and introduce our HOI-GAN framework to generate HOI videos in a zero-shot compositional setting.
- We demonstrate the effectiveness of HOI-GAN through empirical evaluation on two challenging HOI video datasets: *20BN-something-something v2* [72] and *EPIC-Kitchens* [35]. We perform both quantitative and qualitative evaluation of the proposed approach and compare with state-of-the-art approaches.

Overall, our work facilitates research in the direction of enhancing generalizability of generative models for complex videos.

3.2 Related Work

Our work builds on prior work in: (1) modeling of human-object interactions and (2) GAN-based video generation. In addition, we also discuss literature relevant to HOI video gener-

ation in a zero-shot compositional setting.

Modeling Human-Object Interactions. Earlier research attempts to study human-object interactions (HOIs) aimed at studying object affordances [73, 117] and semantic-driven understanding of object functionalities [80, 222]. Recent work on modeling HOIs in images range from studying semantics and spatial features of interactions between humans and objects [39, 69, 275] to action information [43, 56, 266]. Furthermore, there have been attempts to create large scale image and video datasets to study HOI [24, 26, 72, 119]. To model dynamics in HOIs, recent works have proposed methods that jointly model actions and objects in videos [107, 110]. Inspired by these approaches, we model HOI videos as compositions of actions and objects.

GAN-based Image & Video Generation. Generative Adversarial Network (GAN) [71] and its variants [7, 40, 288] have shown tremendous progress in high quality image generation. Built over these techniques, conditional image generation using various forms of inputs to the generator such as textual information [185, 263, 280], category labels [161, 170], and images [97, 113, 142, 293] have been widely studied. This class of GANs allows the generator network to learn a mapping between conditioning variables and the real data distribution, thereby allowing control over the generation process. Extending these efforts to conditional video generation is not straightforward as generating a video involves modeling of both spatial and temporal variations. Vondrick et al. [239] proposed the Video GAN (VGAN) framework to generate videos using a two-stream generator network that decouples foreground and background of a scene. Temporal GAN (TGAN) [196] employs a separate generator for each frame in a video and an additional generator to model temporal variations across these frames. MoCoGAN [232] disentangles the latent space representations of motion and content in a video to perform controllable video generation using seen compositions of motion and content as conditional inputs. In our work, we evaluate the extent to which these video generation methods generalize when provided with unseen scene compositions as conditioning variables. Furthermore, promising success has been achieved by recent video-to-video translation methods [11, 245, 246] wherein video generation is conditioned on a corresponding semantic video. In contrast, our task does not require semantic videos as conditional input.

Since our work proposes GAN-based video generation model, our discussion on related work in the area of image and video generation is primarily focused on GAN-based techniques. However, after the acceptance of our paper [165] based on this research, diffusion models [89, 218] based generative techniques have been very successful at generating high resolution images [45, 184, 190, 194, 195, 271] and videos [83, 88, 215, 265]. Specifically, for video generation, recent models such as Imagen-Video [88] and Make-A-Video [215] are capable of producing diverse set of high resolution videos conditioned with any given natural lan-

guage prompt. As the diffusion-based models have been proven to be a powerful generative framework, it would be an interesting future work to explore how diffusion based models can be used as a base network for our compositional approach for human-object interaction generation. Nonetheless, we believe our work provides insights into how to control the process of compositional generation of videos depicting human-object interactions and what type of information is relevant to the task of HOI video generation.

Video Prediction. Video prediction approaches predict future frames of a video given one or a few observed frames using RNNs [221], variational auto-encoders [240,241], adversarial training [137,155], or auto-regressive methods [106]. While video prediction is typically posed as an image-conditioned (past frame) image generation (future frame) problem, it is substantially different from video generation where the goal is to generate a video clip given a stochastic latent space.

Video Inpainting. Video inpainting/completion refers to the problem of correctly filling up the missing pixels given a video with arbitrary spatio-temporal pixels missing [47,74,167,169,205]. In our setting, however, the model only receives a single static image as input and not a video. Our model is required to go beyond merely filling in pixel values and has to produce an output video with the right visual content depicting the prescribed action upon a synthesized object. In doing so, the background may, and in certain cases should, evolve as well.

Zero-Shot Learning. Zero-shot learning (ZSL) aims to solve the problem of recognizing classes whose instances are not seen during training. In ZSL, external information of a certain form is required to share information between classes to transfer knowledge from seen to unseen classes. A variety of techniques have been used for ZSL ranging from usage of attribute-based information [51,123], word embeddings [256] to WordNet hierarchy [4] and text-based descriptions [48,79,130,295]. [257] provides a thorough overview of zero-shot learning techniques. Similar to these works, we leverage word embeddings to reason over the unseen compositions of actions and objects in the context of video generation.

Learning Visual Relationships. Visual relationships in the form of scene graphs, *i.e.*, directed graphs representing relationships (edges) between the objects (nodes) have been used for image caption evaluation [6], image retrieval [105] and predicting scene compositions for images [146,166,259]. Furthermore, in a generative setting, [103] aims to synthesize an image from a given scene graph and evaluate the generalizability of an adversarial network to create images with unseen relationships between objects. Similarly, we leverage spatio-temporal scene graphs to learn relevant relations among the objects and focus on the generalizability of video generation HOI models to unseen compositions of actions and objects. However, our task of zero-shot HOI video generation is more difficult as it requires learning to map the inputs to spatio-temporal variations in a video.

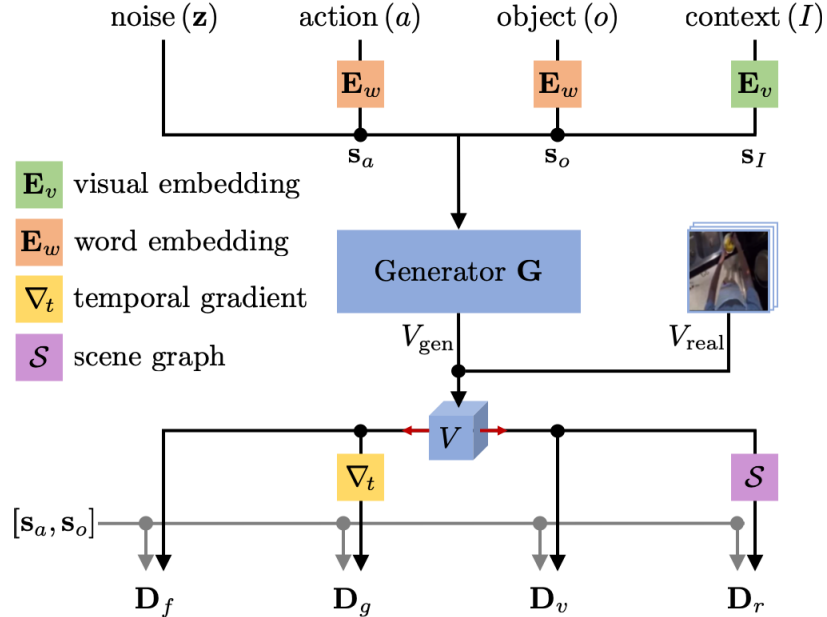


Figure 3.2: **Architecture Overview.** The generator network G is trained using 4 discriminators simultaneously: a frame discriminator D_f , a gradient discriminator D_g , a video discriminator D_v , and a relational discriminator D_r . Given the word embeddings of an action s_a , an object s_o , and a context image s_I , the generator learns to synthesize a video with background I in which the action a is performed on the object o .

Learning Disentangled Representations for Videos. Various methods have been proposed to learn disentangled representations in videos [41, 91, 232], such as, learning representations by decoupling the content and pose [41], or separating motion from content using image differences [237]. Similarly, our model implicitly learns to disentangle the action and object information of an HOI video.

3.3 HOI-GAN

Intuitively, for a generated human-object interaction (HOI) video to be realistic, it must: (1) contain the object designated by a semantic label; (2) exhibit the prescribed interaction with that object; (3) be temporally consistent; and (4 – optional) occur in a specified scene. Based on this intuition, we propose an adversarial learning scheme in which we train a generator network G with a set of 4 discriminators: (1) a frame discriminator D_f , which encourages the generator to learn spatially coherent visual content; (2) a gradient discriminator D_g , which incentivizes G to produce temporally consistent frames; (3) a video discriminator D_v , which provides the generator with global spatio-temporal context; and (4) a relational discriminator D_r , which assists the generator in producing correct object layouts in a video. We use pretrained word embeddings [176] for semantic representations of actions and objects. All discriminators are conditioned on word embeddings of the action

(\mathbf{s}_a) and object (\mathbf{s}_o) and trained simultaneously in an end-to-end manner. Figure 3.2 shows an overview of our proposed framework *HOI-GAN*. We now formalize our task and describe each module in detail.

3.3.1 Task Formulation

Let \mathbf{s}_a and \mathbf{s}_o be word embeddings of an action a and an object o , respectively. Furthermore, let I be an image provided as context to the generator. We encode I using an encoder \mathbf{E}_v to obtain a visual embedding \mathbf{s}_I , which we refer to as a context vector. Our goal is to generate a video $V = (V^{(i)})_{i=1}^T$ of length T depicting the action a performed on the object o with context image I as the background of V . To this end, we learn a function $\mathbf{G} : (\mathbf{z}, \mathbf{s}_a, \mathbf{s}_o, \mathbf{s}_I) \mapsto V$, where \mathbf{z} is a noise vector sampled from a distribution $p_{\mathbf{z}}$, such as a Gaussian distribution.

3.3.2 Model Description

We describe the elements of *HOI-GAN* below. Overall, the four discriminator networks, *i.e.*, frame discriminator \mathbf{D}_f , gradient discriminator \mathbf{D}_g , video discriminator \mathbf{D}_v , and relational discriminator \mathbf{D}_r are all involved in a zero-sum game with the generator network \mathbf{G} .

Frame Discriminator. The frame discriminator network \mathbf{D}_f learns to distinguish between real and generated frames corresponding to the real video V_{real} and generated video $V_{\text{gen}} = \mathbf{G}(\mathbf{z}, \mathbf{s}_a, \mathbf{s}_o, \mathbf{s}_I)$ respectively. Each frame in V_{gen} and V_{real} is processed independently using a network consisting of stacked `conv2d` layers, *i.e.*, 2D convolutional layers followed by spectral normalization [160] and leaky ReLU layers [150] with $a = 0.2$. We obtain a tensor of size $N^{(t)} \times w_0^{(t)} \times h_0^{(t)}$ ($t = 1, 2, \dots, T$), where $N^{(t)}$, $w_0^{(t)}$, and $h_0^{(t)}$ are the channel length, width and height of the activation of the last `conv2d` layer respectively. We concatenate this tensor with spatially replicated copies of \mathbf{s}_a and \mathbf{s}_o , which results in a tensor of size $(\dim(\mathbf{s}_a) + \dim(\mathbf{s}_o) + N^{(t)}) \times w_0^{(t)} \times h_0^{(t)}$. We then apply another `conv2d` layer to obtain a $N \times w_0^{(t)} \times h_0^{(t)}$ tensor. We now perform 1×1 convolutions followed by $w_0^{(t)} \times h_0^{(t)}$ convolutions and a sigmoid to obtain a T -dimensional vector corresponding to the T frames of the video V . The i -th element of the output denotes the probability that the frame $V^{(i)}$ is real. The objective function of the network \mathbf{D}_f is the loss function:

$$L_f = \frac{1}{2T} \sum_{i=1}^T [\log(\mathbf{D}_f^{(i)}(V_{\text{real}}; \mathbf{s}_a, \mathbf{s}_o)) + \log(1 - \mathbf{D}_f^{(i)}(V_{\text{gen}}; \mathbf{s}_a, \mathbf{s}_o))], \quad (3.1)$$

where $\mathbf{D}_f^{(i)}$ is the i -th element of the output of \mathbf{D}_f .

Gradient Discriminator. The gradient discriminator network \mathbf{D}_g enforces temporal smoothness by learning to differentiate between the temporal gradient of a real video V_{real} and a generated video V_{gen} . We define the temporal gradient $\nabla_t V$ of a video V with T frames

$V^{(1)}, \dots, V^{(T)}$ as pixel-wise differences between two consecutive frames of the video. The i -th element of $\nabla_t V$ is defined as:

$$[\nabla_t V]_i = V^{(i+1)} - V^{(i)}, \quad i = 1, 2, \dots, (T - 1). \quad (3.2)$$

The architecture of the gradient discriminator \mathbf{D}_g is similar to that of the frame discriminator \mathbf{D}_f . The output of \mathbf{D}_g is a $(T - 1)$ -dimensional vector corresponding to the $(T - 1)$ values in gradient $\nabla_t V$. The objective function of \mathbf{D}_g is

$$L_g = \frac{1}{2(T - 1)} \sum_{i=1}^{T-1} [\log(\mathbf{D}_g^{(i)}(\nabla_t V_{\text{real}}; \mathbf{s}_a, \mathbf{s}_o)) + \log(1 - \mathbf{D}_g^{(i)}(\nabla_t V_{\text{gen}}; \mathbf{s}_a, \mathbf{s}_o))], \quad (3.3)$$

where $\mathbf{D}_g^{(i)}$ is the i -th element of the output of \mathbf{D}_g .

Video Discriminator. The video discriminator network \mathbf{D}_v learns to distinguish between real videos V_{real} and generated videos V_{gen} by comparing their global spatio-temporal contexts. The architecture consists of stacked `conv3d` layers, *i.e.*, 3D convolutional layers followed by spectral normalization [160] and leaky ReLU layers [150] with $a = 0.2$. We obtain a $N \times d_0 \times w_0 \times h_0$ tensor, where N , d_0 , w_0 , and h_0 are the channel length, depth, width, and height of the activation of the last `conv3d` layer respectively. We concatenate this tensor with spatially replicated copies of \mathbf{s}_a and \mathbf{s}_o , which results in a tensor of size $(\dim(\mathbf{s}_a) + \dim(\mathbf{s}_o) + N) \times d_0 \times w_0 \times h_0$, where $\dim(\cdot)$ returns the dimensionality of a vector. We then apply another `conv3d` layer to obtain a $N \times d_0 \times w_0 \times h_0$ tensor. Finally, we apply a $1 \times 1 \times 1$ convolution followed by a $d_0 \times w_0 \times h_0$ convolution and a sigmoid to obtain the output, which represents the probability that the video V is real. The objective function of the network \mathbf{D}_v is the following loss function:

$$L_v = \frac{1}{2} [\log(\mathbf{D}_v(V_{\text{real}}; \mathbf{s}_a, \mathbf{s}_o)) + \log(1 - \mathbf{D}_v(V_{\text{gen}}; \mathbf{s}_a, \mathbf{s}_o))]. \quad (3.4)$$

Relational Discriminator. In addition to the three pixel-centric discriminators above, we also propose a novel object-centric discriminator \mathbf{D}_r . Driven by a spatio-temporal scene graph, this relational discriminator learns to distinguish between scene layouts of real videos V_{real} and generated videos V_{gen} (Figure 3.3).

Specifically, we build a spatio-temporal scene graph $\mathcal{S} = (\mathcal{N}, \mathcal{E})$ from V , where the nodes and edges are represented by \mathcal{N} and \mathcal{E} respectively. We assume one node per object per frame. Each node is connected to all other nodes in the same frame, referred to as spatial edges. In addition, to represent temporal evolution of objects, each node is connected to the corresponding nodes in the adjacent frames that also depict the same object, referred

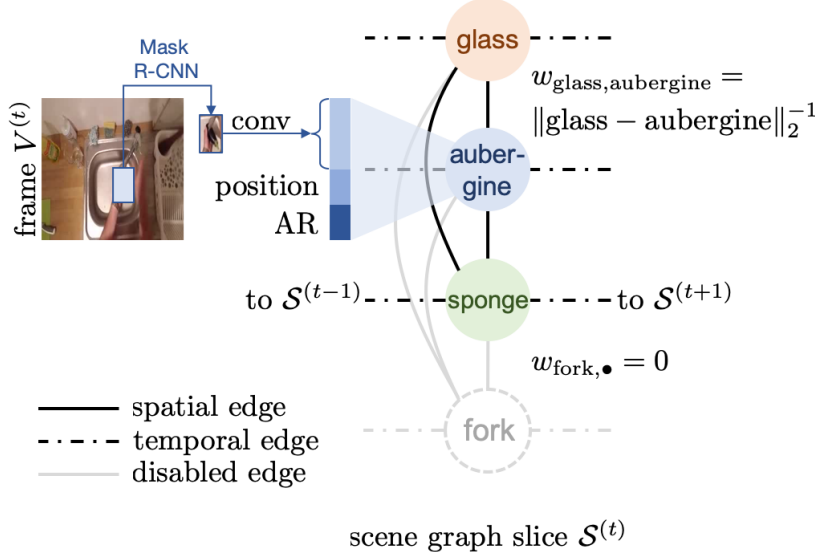


Figure 3.3: **Relational Discriminator.** The relational discriminator \mathbf{D}_r leverages a spatio-temporal scene graph to distinguish between object layouts in videos. Each node contains convolutional embedding, position and aspect ratio (AR) of the object crop obtained from MaskRCNN. The nodes are connected in space and time and edges are weighted based on their inverse distance. Edge weights of (dis)appearing objects are 0.

to as temporal edges. To obtain the node representations, we crop the objects in V using Mask-RCNN [85], compute a convolutional embedding for them, and augment the resulting vectors with the aspect ratio (AR) and position of the corresponding bounding boxes. The weights of spatial edges in \mathcal{E} are given by inverse Euclidean distances between the centers of these bounding boxes corresponding to the object appearing in the frame. The weights of the temporal edges is set to 1 by default. When an object is not present in a frame (but appears in the overall video), spatial edges connecting to the object will be absent by design. This is implemented by setting the weights to 0 depicting distance between the objects as ∞ . Similarly, if an object does not appear in the adjacent frame, the temporal edge is set to 0. In case of multiple objects of the same category, the correspondence is established based on the location in the adjacent frames using nearest neighbour data association.

The relational discriminator \mathbf{D}_r operates on this scene graph \mathcal{S} by virtue of a graph convolutional network (GCN) [115] followed by stacking and average-pooling of the resulting node representations along the time axis. We then concatenate this tensor with spatially replicated copies of \mathbf{s}_a and \mathbf{s}_o to result in a tensor of size $(\dim(\mathbf{s}_a) + \dim(\mathbf{s}_o) + N^{(t)}) \times w_0^{(t)} \times h_0^{(t)}$. As before, we then apply convolutions and sigmoid to obtain the final output which denotes the probability of the scene graph belonging to a real video. The objective function of the network \mathbf{D}_r is given by

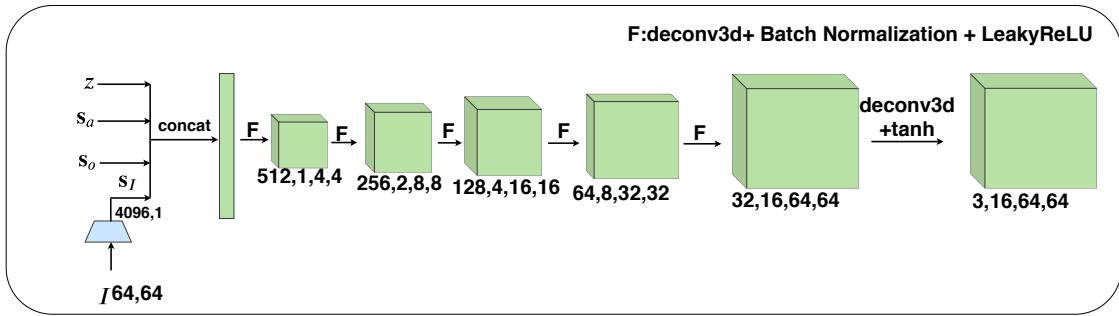
$$L_r = \frac{1}{2} [\log(\mathbf{D}_r(\mathcal{S}_{\text{real}}; \mathbf{s}_a, \mathbf{s}_o)) + \log(1 - \mathbf{D}_r(\mathcal{S}_{\text{gen}}; \mathbf{s}_a, \mathbf{s}_o))]. \quad (3.5)$$

Generator. Given the semantic embeddings \mathbf{s}_a , \mathbf{s}_o of action and object labels respectively, and context vector \mathbf{s}_I , the generator network \mathbf{G} learns to generate video V_{gen} consisting of T frames (RGB) of height H and width W . We concatenate noise \mathbf{z} with the conditions, namely, \mathbf{s}_a , \mathbf{s}_o , and \mathbf{s}_I . We provide this concatenated vector as the input to the network \mathbf{G} . The network comprises stacked `deconv3d` layers, *i.e.*, 3D transposed convolution layers each followed by Batch Normalization [96] and leaky ReLU layers [150] with $a = 0.2$ except the last convolutional layer which is followed by a Batch Normalization layer [96] and a `tanh` activation layer. The network is optimized according to the following objective function:

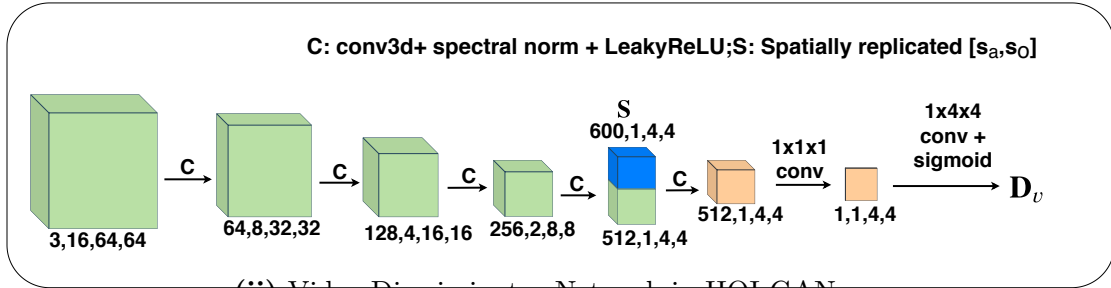
$$\begin{aligned}
L_{gan} = & \frac{1}{T} \sum_{i=1}^T [\log(1 - \mathbf{D}_f^{(i)}(V_{gen}; \mathbf{s}_a, \mathbf{s}_o))] + \\
& \frac{1}{(T-1)} \sum_{i=1}^{T-1} [\log(1 - \mathbf{D}_g^{(i)}(\nabla_t V_{gen}; \mathbf{s}_a, \mathbf{s}_o))] + \\
& \log(1 - \mathbf{D}_v(V_{gen}; \mathbf{s}_a, \mathbf{s}_o)) + \log(1 - \mathbf{D}_r(\mathcal{S}_{gen}; \mathbf{s}_a, \mathbf{s}_o)).
\end{aligned} \tag{3.6}$$

Architecture Details. Our model comprises 5 networks involving a generator network and four discriminator networks. We provide the details of the architectures used in our implementation for the generator network, video discriminator, frame discriminator and relational discriminator in Figure 3.4. The architecture for gradient discriminator is same as that of the frame discriminator.

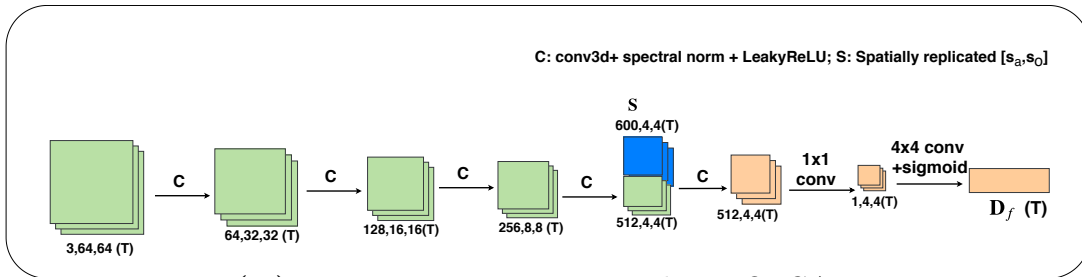
For relational discriminator, we used the final output layer of MaskRCNN, that comprises a list of bounding boxes, a list of segmentation masks and a list of labels corresponding to each detection. We used <https://github.com/facebookresearch/maskrcnn-benchmark> repository to obtain the detection output. The same list of bounding boxes have been used for real and generated. Then, using each bounding box in the output, we crop the visual region from the corresponding frame. These crops will correspond to the nodes of spatio-temporal graph. These cropped visual regions are resized to $3 \times 16 \times 16$ ($C \times H \times W$) and their position (bounding box top-left coordinates normalized with respect to the image size) and their original aspect ratio (bounding box height and width normalized with respect to image size) are collectively used for node feature representation (Refer to Figure 3 for illustration). We used a `conv` module (shared weights for all crops), *i.e.*, convolutional layers (stride=2, kernel size=4) and obtain a convolutional embedding for resized visual regions of size 4096 appended with 4 additional numbers corresponding to position and aspect ratio. We design Graph Convolution Layer using the implementation of Graph Convolution Network (GCN) available at <https://github.com/tkipf/pygcn>. We used 7 such Graph Convolution layers: initial layer converts the feature size to 4096 and output feature size of the node is doubled every two layer in next 6 layers. Until this stage, the node is represented using single dimensional vector. After pooling along the temporal axis, the channel



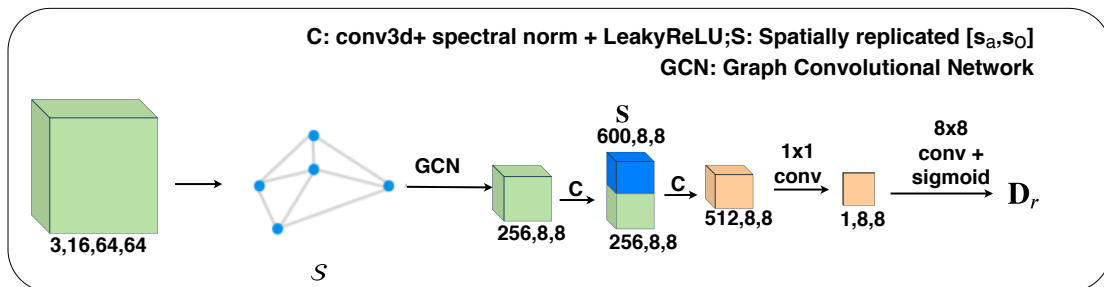
(i) Generator Network in HOI-GAN



(ii) Video Discriminator Network in HOI-GAN



(iii) Frame Discriminator Network in HOI-GAN



(iv) Relational Discriminator Network in HOI-GAN

Figure 3.4: **Architecture Details.** Model architectures used in our experiments for: (i) Generator, (ii) Video Discriminator, (iii) Frame discriminator (gradient discriminator has similar architecture), (iv) Relational Discriminator. Best viewed in color on desktop.

dimension is reshaped to $256 \times 8 \times 8$ and the resulting tensor is of shape $K \times 256 \times 8 \times 8$ where K is the number of crops.

Implementation Details. In our experiments, the convolutional layers in all networks, namely, $\mathbf{G}, \mathbf{D}_f, \mathbf{D}_g, \mathbf{D}_v, \mathbf{D}_r$ have kernel size 4 and stride 2. We generate a video clip consisting of $T = 16$ frames having $H = W = 64$. The noise vector \mathbf{z} is of length 100. The parameters $w_0 = h_0 = 4$, $d_0 = 1$ and $N = 512$ for \mathbf{D}_v and $w_0^t = h_0^t = 4$ and $N^{(t)} = 512$ for $\mathbf{D}_f, \mathbf{D}_g$, and \mathbf{D}_r . To obtain the semantic embeddings \mathbf{s}_a and \mathbf{s}_o corresponding to action and object labels respectively, we use Wikipedia-pretrained GLoVe [176] embedding vectors of length 300. For training, we use the Adam [114] optimizer with learning rate 0.0002 and $\beta_1 = 0.5$, $\beta_2 = 0.999$. We train all our models with a batch size of 32. We use dropout (probability = 0.3) [197] in the last layer of all discriminators and all layers (except first) of the generator.

3.4 Experiments

We conduct quantitative and qualitative analysis to demonstrate the effectiveness of the proposed framework HOI-GAN for the task of zero-shot generation of human-object interaction (HOI) videos.

3.4.1 Datasets and Data Splits

We use two datasets for our experiments: EPIC-Kitchens [35] and 20BN-Something-Something V2 [72]. Both of these datasets comprise a diverse set of HOI videos ranging from simple translational motion of objects (*e.g.* push, move) and rotation (*e.g.* open) to transformations in state of objects (*e.g.* cut, fold). Therefore, these datasets, with their wide ranging variety and complexity, provide a challenging setup for evaluating HOI video generation models.

EPIC-Kitchens [35] contains egocentric videos of activities in several kitchens. A video clip V is annotated with action label a and object label o (*e.g.* open microwave, cut apple, move pan) along with a set of bounding boxes \mathcal{B} (one per frame) for objects that the human interacts with while performing the action. There are around 40k instances in the form of (V, a, o, \mathcal{B}) across 352 objects and 125 actions. We refer to this dataset as EPIC hereafter.

20BN-Something-Something V2 [72] contains videos of daily activities performed by humans. A video clip V is annotated with a label l , an action template and object(s) on which the action is applied (*e.g.* ‘hitting ball with racket’ has action template ‘hitting something with something’). There are 220,847 training instances of the form (V, l) spanning 30,408 objects and 174 action templates. To transform l to action-object label pair (a, o) , we use NLTK POS-tagger. We consider the verb tag (after stemming) in l as action label a . We observe that all instances of l begin with the present continuous form of a which is acting upon the subsequent noun. Therefore, we use the noun that appears immediately

after the verb as object o . Hereafter, we refer to the transformed dataset in the form of (V, a, o) as SS.

Splitting by Compositions. We believe it is reasonable to only generate combinations that are semantically feasible, and do so by only using action-object pairs seen in the original datasets. We use a subset of action-object pairs as testing pairs – these pairs are not seen during training but are present in the original dataset, hence are semantically feasible. To make the dataset training / testing splits suitable for our zero-shot compositional setting, we first merge the data samples present in the default train and validation sets of the dataset. We then split the combined dataset into training set and test set based on the condition that all the unique object and action labels in appear in the training set, however, any composition of action and object present in the test set is absent in training set and vice versa. The details specific to eat dataset is provided as follows.

EPIC: Processing and Splits. The EPIC-Kitchens dataset originally consists of 39,594 video samples of the form (V, a, o) , *i.e.*, video V with action label a and object label o , spanning 125 unique actions and 352 unique objects. We further filtered the dataset to ensure that the video samples contain both ground truth bounding box annotation and MaskRCNN output (NMS threshold = 0.7) in the frames uniformly sampled from a video. We interpolated the sequence if the number of such frames is less than 16. We then split the filtered dataset by action-object compositions to obtain train and test splits suitable for the zero-shot compositional setting, *i.e.*, all the unique object and action labels in combined dataset appear independently in the train split, however, a certain pair of action and object present in the test split is absent in train split and vice versa. Subsequently we obtained two splits: (1) train split containing 19,895 videos that overall depict 1,128 unique action-object compositions, and (2) test split containing 7,805 videos (568 unique action-object compositions). The final splits consist of compositions spanning 204 unique actions and 63 unique objects.

SS: Processing and Splits. The 20BN-Something-Something V2 dataset originally consists of 220,847 video samples of the form (V, l) , *i.e.*, video V having a label l . To transform the dataset instances to the form (V, a, o) , we applied NLTK POS-tagger on l and obtained verb a and noun o . In particular, we considered the verb tag (after stemming) in l as action label a . We observe that all instances of l begin with the present continuous form of a which is acting upon the subsequent noun. Therefore, we used the noun that appears immediately after the verb as object o . We merged the train and validation split of the transformed dataset. We further filtered the dataset to ensure that the video samples contain objects that can be detected using MaskRCNN (NMS threshold = 0.7) in the frames uniformly sampled from a video. We then split the transformed dataset by compositions of action a and object o to obtain the train and test splits suitable for the zero-shot compositional setting (same as EPIC). Subsequently, we obtained two splits: (1) train split containing 23,511 videos overall that overall depict 671 unique action-object compositions,

Table 3.1: **Generation Scenarios.** Description of the conditional inputs for the two generation scenarios GS1 & GS2 used for evaluation. ✓ denotes ‘Yes’, ✗ denotes ‘No’.

Target Conditions	GS1	GS2
Target action a seen during training	✓	✓
Target object o seen during training	✓	✓
Background of target context I seen during training	✗	✓
Object mask in target context I corresponds to target object o	✓	✗
Target action a seen with target context I during training	✗	✓/ ✗
Target object o seen with target context I during training	✗	✗
Target action-object composition ($a-o$) seen during training	✗	✗

and (2) test split containing 3,515 videos overall (135 unique action-object compositions). The final splits consist of compositions spanning 48 unique actions and 62 unique objects.

Generation Scenarios. Recall that the generator network in the HOI-GAN framework (Fig. 3.2) has 3 conditional inputs, namely, action embedding, object embedding, and context frame I . The context frame serves as the background in the scene. Thus, to provide this context frame during training, we apply a binary mask $M^{(1)}$ corresponding to the first frame $V^{(1)}$ of a real video as $I = (\mathbb{1} - M^{(1)}) \odot V^{(1)}$, where $\mathbb{1}$ represents a matrix of size $M^{(1)}$ containing all ones and \odot denotes elementwise multiplication. This mask $M^{(1)}$ contains ones in regions (either rectangular bounding boxes or segmentation masks) corresponding to the objects (non-*person* classes) detected using MaskRCNN [85] and zeros for other regions. Intuitively, this helps ensure the generator learns to map the action and object embeddings to relevant visual content in the HOI video.

During testing, to evaluate the generator’s capability to synthesize the right human-object interactions, we provide a background frame as described above. This background frame can be selected from either the test set or training set, and can be suitable or unsuitable for the target action-object composition. To capture these possibilities, we design two different generation scenarios. Specifically, in *Generation Scenario 1 (GS1)*, the input context frame I is the masked first frame of a video from the test set corresponding to the target action-object composition (unseen during training). In *Generation Scenario 2 (GS2)*, I is the masked first frame of a video from the training set which depicts an object other than the target object. The original action in this video could be same or different than the target action. See Table 3.1 for the contrast between the scenarios.

As such, in GS1, the generator receives a context that it has not seen during training but the context (including object mask) is consistent with the target action-object composition it is being asked to generate. In contrast, in GS2, the generator receives a context frame that it has seen during training but is not consistent with the action-object composition it is being asked to generate. Particularly, the object mask in the context does not correspond to the target object. Although the background is seen, the model has to evolve the background

in ways different from training samples to make it suitable for the target composition. Thus, these generation scenarios help illustrate that the generator indeed generalizes over compositions.

3.4.2 Evaluation Setup

Evaluation of image/video quality is inherently challenging, thus, we use both quantitative and qualitative metrics.

Quantitative Metrics. Inception Score (**I-score**) [197] is a widely used metric for evaluating image generation models. For images x with labels y , I-score is defined as:

$$I = \exp(\mathbf{KL}(\rho(y|x)||\rho(y))), \quad (3.7)$$

where $\rho(y|x)$ is the conditional label distribution of an ImageNet [192]-pretrained Inception model [227]. We adopted this metric for video quality evaluation. We fine-tune a Kinetics [22]-pretrained video classifier ResNeXt-101 [258] for each of our source datasets and use it for calculating I-score (higher is better). It is based on one of the state-of-the-art video classification architectures. We used the same evaluation setup for the baselines and our model to ensure a fair comparison.

In addition, we believe that measuring realism explicitly is more relevant for our task as the generation process can be conditioned on any context frame arbitrarily to obtain diverse samples. Therefore, in addition to *I-score*, we also analyze the first and second terms of the KL divergence separately. We refer to these terms as: (1) Saliency score or **S-score** (lower is better) to specifically measure realism, and (2) Diversity score or **D-score** (higher is better) to indicate the diversity in generated samples. A smaller value of S-score implies that the generated videos are more realistic as the classifier is very confident in classifying the generated videos. Specifically, the saliency score will have a low value (low is good) only when the classifier is confidently able to classify the generated videos into action-object categories matching the conditional input composition (action-object), thus indicating realistic instances of the required target interaction. In fact, even if a model generates realistic-looking videos but depicts an action-object composition not corresponding to the conditional action-object input, the saliency score will have high values. Finally, a larger value of D-score implies the model generates diverse samples.

Human Preference Score. We conduct a user study for evaluating the quality of generated videos. In each test, we present the participants with two videos generated by two different algorithms and ask which among the two better depicts the given activity, *i.e.*, action-object composition (*e.g.* lift fork). We evaluate the performance of an algorithm as the overall percentage of tests in which that algorithm’s outputs are preferred. This is an aggregate measure over all the test instances across all participants.

Table 3.2: **Quantitative Evaluation.** Comparison of HOI-GAN with C-VGAN, C-TGAN, and MoCoGAN baselines. We distinguish training of HOI-GAN with bounding boxes (*bboxes*) and segmentation masks (*masks*). Arrows indicate whether lower (\downarrow) or higher (\uparrow) is better. [I: inception score; S: saliency score; D: diversity score]

Model	EPIC						SS					
	GS1			GS2			GS1			GS2		
	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow
C-VGAN [239]	1.8	30.9	0.2	1.4	44.9	0.3	2.1	25.4	0.4	1.8	40.5	0.3
C-TGAN [196]	2.0	30.4	0.6	1.5	35.9	0.4	2.2	28.9	0.6	1.6	39.7	0.5
MoCoGAN [232]	2.4	30.7	0.5	2.2	31.4	1.2	2.8	17.5	1.0	2.4	33.7	1.4
(ours) HOI-GAN (bboxes)	6.0	14.0	3.4	5.7	20.8	4.0	6.6	12.7	3.5	6.0	15.2	2.9
(ours) HOI-GAN (masks)	6.2	13.2	3.7	5.2	18.3	3.5	8.6	11.4	4.4	7.1	14.7	4.0

Table 3.3: **Ablation Study.** We evaluate the contributions of our pixel-centric losses (F,G,V) and relational losses (first block vs. second block) by conducting ablation study on HOI-GAN (masks). The last row corresponds to the overall proposed model.[F: frame discriminator \mathbf{D}_f ; G: gradient discriminator \mathbf{D}_g ; V: video discriminator \mathbf{D}_v ; R: relational discriminator \mathbf{D}_r]

Model	EPIC						SS					
	GS1			GS2			GS1			GS2		
	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow
HOI-GAN (F)	1.4	44.2	0.2	1.1	47.2	0.3	1.8	34.7	0.4	1.5	39.5	0.3
\mathcal{R}_\downarrow HOI-GAN (F+G)	2.3	25.6	0.7	1.9	30.7	0.5	3.0	24.5	0.9	2.7	28.8	0.7
HOI-GAN (F+G+V)	2.8	21.2	1.3	2.6	29.7	1.7	3.3	18.6	1.2	3.0	20.7	1.0
HOI-GAN (F)	2.4	24.9	0.8	2.2	26.0	0.7	3.1	20.3	1.0	2.9	27.7	0.9
\mathcal{R}_\downarrow HOI-GAN (F+G)	5.9	15.4	3.5	4.8	21.3	3.3	7.4	12.1	3.5	5.4	19.2	3.4
HOI-GAN (F+G+V)	6.2	13.2	3.7	5.2	18.3	3.5	8.6	11.4	4.4	7.1	14.7	4.0

Baselines. We compare HOI-GAN with three state-of-the-art video generation approaches: (1) VGAN [239], (2) TGAN, [196] and (3) MoCoGAN [232]. We develop the conditional variants of VGAN and TGAN from the descriptions provided in their papers. We refer to the conditional variants as C-VGAN and C-TGAN respectively. We observed that these two models saturated easily in the initial iterations, thus, we added dropout in the last layer of the discriminator network in both models. MoCoGAN focuses on disentangling motion and content in the latent space and is the closest baseline. We use the code provided by the authors.

3.4.3 Quantitative Evaluation

Next, we discuss the results of our quantitative evaluation.

Table 3.4: **Quantitative Evaluation (Effect of Word Embeddings)**. Comparison of HOI-GAN with C-VGAN, C-TGAN, and MoCoGAN baselines using one-hot encoded labels instead of embeddings as conditional inputs(default version). (see section 3.4.3). Arrows indicate whether lower (\downarrow) or higher (\uparrow) is better. [I: inception score; S: saliency score; D: diversity score]

Model	EPIC						SS					
	GS1			GS2			GS1			GS2		
	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow	I \uparrow	S \downarrow	D \uparrow
C-VGAN [239]	1.1	52.1	0.4	1.1	52.1	0.4	2.1	45.6	0.8	1.9	45.1	0.5
C-TGAN [196]	1.6	65.4	0.4	2.2	28.1	0.5	2.4	36.2	1.1	1.7	42.8	0.6
MoCoGAN [232]	2.6	25.4	1.0	2.0	34.9	1.0	2.9	22.8	1.3	2.4	27.4	1.5
(ours) HOI-GAN (bboxes)	3.8	18.5	2.1	3.2	24.1	2.4	4.9	26.2	2.7	4.0	25.2	2.4
(ours) HOI-GAN (masks)	4.3	16.5	2.5	3.9	20.2	1.6	5.8	15.8	3.0	4.5	23.7	2.8

Table 3.5: **Human Evaluation**. Human Preference Score (%) for scenarios GS1 and GS2. All the results have p-value less than 0.05 implying statistical significance.

Ours / Baseline	GS1	GS2
HOI-GAN / MoCoGAN	71.7/28.3	69.2/30.8
HOI-GAN / C-TGAN	75.4/34.9	79.3/30.7
HOI-GAN / C-VGAN	83.6/16.4	80.4/19.6

Comparison with Baselines. As shown in Table 3.2, HOI-GAN with different conditional inputs outperforms C-VGAN and C-TGAN by a wide margin in both generation scenarios. In addition, our overall model shows considerable improvement over MoCoGAN, while MoCoGAN has comparable scores to some ablated versions of our models (where gradient discriminator and/or relational discriminator is missing). Furthermore, we varied the richness of the masks in the conditional input context frame ranging from bounding boxes to segmentation masks obtained corresponding to non-*person* classes using MaskRCNN framework [85]. We observe that providing masks during training leads to slight improvements in both scenarios as compared to using bounding boxes (refer to Table 3.2).

Ablation Study. To illustrate the impact of each discriminator in generating HOI videos, we conduct ablation experiments (refer to Table 3.3). We observe that the addition of temporal information using the gradient discriminator and spatio-temporal information using the video discriminator lead to improvement in generation quality. In particular, the addition of our scene graph based relational discriminator leads to considerable improvement in generation quality resulting in more realistic videos (refer to second block in Table 3.3).

Effect of Word Embeddings. In our approach, we use word embeddings for the action and object labels to share information among semantically similar categories during training. To demonstrate the impact of using embeddings, we also trained HOI-GAN using one-hot


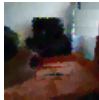
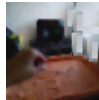
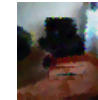
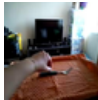



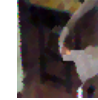


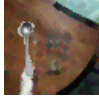
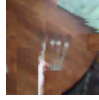
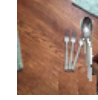
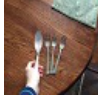

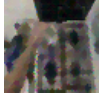

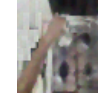







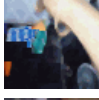
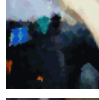
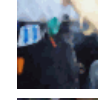
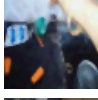

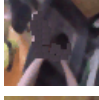
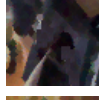
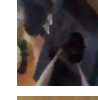
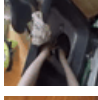

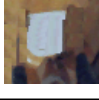
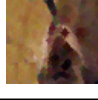
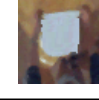
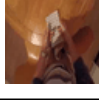
(a, o)	Context	C-VGAN	C-TGAN	MoCoGAN	HOI-GAN
lift fork					
bend carrot					
put spoon					
open lid					
cover banana					
fall carrot					
brush pan					
fold cloth					

Figure 3.5: **Qualitative Analysis (Comparison with Baselines).** Samples generated using the baseline models and HOI-GAN for a given composition of action-object (a, o) pair and context image. We provide middle frame of each generated video sample.

encoded labels corresponding to both actions and objects. We observe that these models perform worse than the models trained using semantic embeddings (refer last two rows of Table 3.2 in the main paper and Table 3.4). Nevertheless, our models still outperform the baselines (refer to Table 3.4).

Human Evaluation. We recruited 15 sequestered participants for our user study. We randomly chose 50 unique categories and chose generated videos for half of them from generation scenario GS1 and the other half from GS2. For each category, we provided three instances, each containing a pair of videos; one generated using a baseline model and the other using HOI-GAN. For each instance, at least 3 participants (ensuring inter-rater reliability) are asked to choose the video that best depicts the given category. The (aggregate) human preference scores for our model versus the baselines range between 69-84% for both generation scenarios (refer Table 3.5). These results indicate that HOI-GAN generates more realistic videos than the baselines.

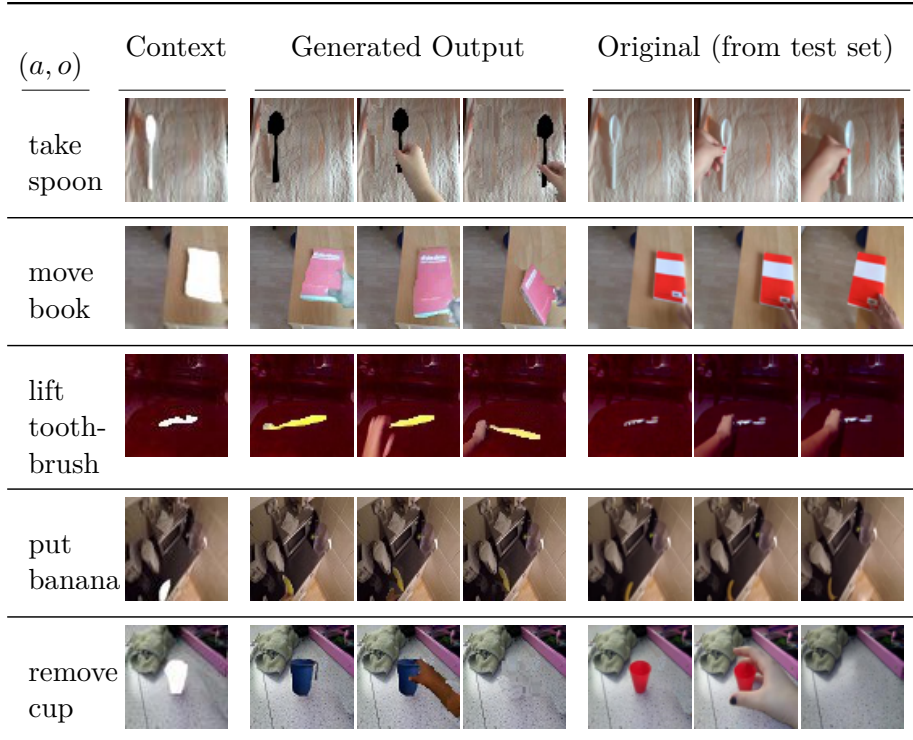


Figure 3.6: **Qualitative Evaluation (GS1)**. Samples generated using our model in Generation Scenario 1, *i.e.*, both the target context image and the target action-object (a, o) composition are unseen during training. We provide 3 frames of the generated output and 3 frames of the original video (same context, action, object) from the test set for comparison.

3.4.4 Qualitative Analysis

In this section, we visualize different types of scenarios and analyzed some of the videos generated using HOI-GAN.

Qualitative Analysis (Comparison with Baselines). For more qualitative evaluation, Figure 3.5 shows a comparison between samples generated from these baselines and HOI-GAN. The results clearly show that our HOI-GAN is able to synthesize more realistic videos. Moreover, this supports the quantitative evaluation in Table 3.2.

Qualitative Analysis (GS1). We present samples generated using our HOI-GAN in generation scenario 1 (GS1). In GS1 setting, the target context image and the target action-object composition are unseen during training. Thus, the context image is from the test set (obtained in zero-shot compositional setting) and the object mask in the context image corresponds to the target object. As shown in Figure 3.6, our model is able to create objects and enact the prescribed action on the object in the given context.

Figure 3.6 also shows the real videos from the test set corresponding to the given compositions and context frame. The results clearly demonstrate that our model is able to generate realistic videos depicting the given action-object in the given context. The visual








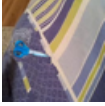
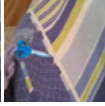
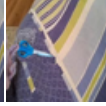
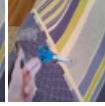
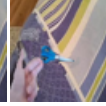
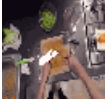
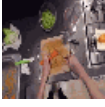
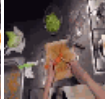
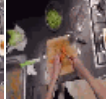
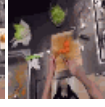
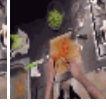




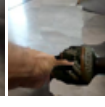









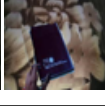
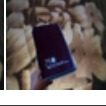
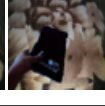
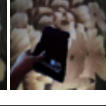




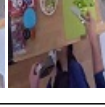
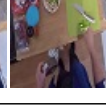






Action-object labels	Context	Generated output				
G: lift apple O: hold banana						
G: push scissors O: pull spoon						
G: cut carrot O: cut celery						
G: turn vase O: move bottle						
G: spin bottle O: spin remote						
G: move book O: open handbag						
G: move broccoli O: take carrot						
G: put apple O: put bowl						

Figure 3.7: **Qualitative Evaluation (GS2)**. Samples generated using our model in Generation Scenario 2, *i.e.*, target action-object composition are unseen during training but target context image is seen with an object different from target object and a same/different action from target action. Thus, the overall target compositions comprising object, action and context are unseen during training. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We provide 5 frames for each generated video sample in the figure.

appearance of objects and actions (hand movements) are somewhat different in the generated videos compared to the corresponding real video because the model had to generalize based on other depictions of the object and action that were seen separately in training. Nevertheless, the results show that the generated video is also a realistic depiction of the target composition showing the target action on the target object in the target context.

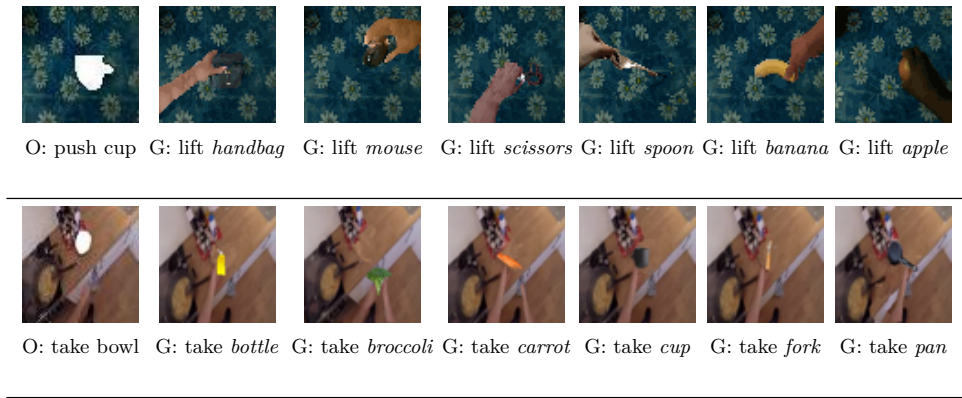


Figure 3.8: **Qualitative Evaluation (GS2 - same action, same context, different objects)**. Samples generated using HOI-GAN in Generation Scenario 2 corresponding to a set of compositions with same context frame, same action and different objects. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We show the context frame with mask on the left in each row and the middle frame of each generated video.

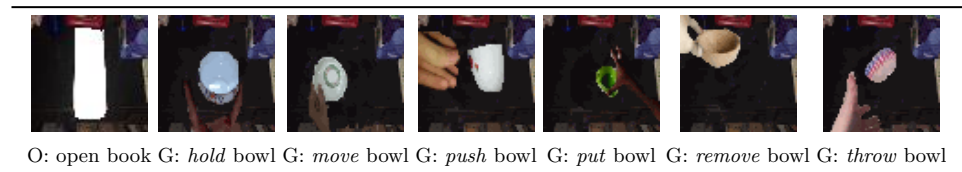


Figure 3.9: **Qualitative Evaluation (GS2 - same object, same context, different actions)**. Samples generated using HOI-GAN in Generation Scenario 2 corresponding to a set of compositions with same context frame, same object and different actions. ‘G’ indicates the target action-object composition and ‘O’ indicates the action-object composition of the video (in the training set) from which the context image is chosen. We show the context frame with mask on the left in each row and the middle frame of each generated video.

Qualitative Analysis (GS2). We present samples generated using our HOI-GAN in generation scenario 2 (GS2). In GS2 setting, the target context background is seen during training while the target action-object composition is unseen. Specifically, the context image is from a video in the training set and the object mask in the context image corresponds to an object different than the target object. Also, the action corresponding to the context image may or may not be same as the target action. As such, the background may or may not be fully amenable for the target action-object composition. As shown in Figure 3.7, our model is able to create the required objects and enact the prescribed actions on the objects in the given context background. Moreover, our model is also able to modify the background as and when needed based on the target composition to be generated. The results clearly demonstrate that our model is able to generate realistic videos depicting the given action-object in the given context. In particular, the *move book* sample provides a comparison with its corresponding sample of *move book* in the GS1 setting (see Figure 3.6).

In the GS2 setting seen here, the mask in the context frame corresponds to a handbag. The model is able to align the orientation of the book with the provided mask of the handbag and fit the object *book* in the mask. In contrast, the size of *book* with respect to the mask in this case is different from that seen in the GS1 example. In addition to showing the diversity in generated samples, we also generate videos corresponding to various sets of compositions with the same target context, same target action and different target objects. Samples in Figure 3.8 indicate our model is able to synthesize videos with the same action in the same context being performed on multiple objects differently. For instance, hand(s) appear from different directions and look different. Our model is also able to scale the objects appropriately based on the mask (see *lift handbag* in Figure 3.8). Furthermore, we also generate videos corresponding to various sets of compositions with the same target context, same target object and different target actions. Samples in Figure 3.9 indicate that our model is able to synthesize videos with different actions being performed on same object. In particular, the model is able to generate the same object with different and diverse set of visual appearances (*e.g.* the *bowls* in Figure 3.9 look different) and perform the different actions upon them.

How does HOI-GAN generalize over compositions? Recall, the generation in this paper is performed in a zero-shot compositional setting, i.e., actions and objects are seen independently in certain compositions during training but the target action-object compositions are unseen during training. Intuitively, during this process, our model is able to effectively disentangle actions and objects. Therefore, when given a previously unseen target action-object composition for generation, our model is able to bring together or combine the information (distributed over the training set) in a meaningful way to synthesize realistic videos corresponding to the unseen composition. Consider the video corresponding to *lift handbag* in Figure 3.10, the model has seen different handbags in different contexts with different actions (other than *lift*), and has also seen different instances of the action *lift* being performed on objects other than *handbag* in different contexts. Given all this information, our model is able to combine the relevant information and synthesizes a video corresponding to a handbag being lifted in the given context. Similarly, we show two other compositions and the corresponding training samples of the action and object that might have helped the model during the particular generations.

Failure Cases. We show some qualitative examples where our framework fails to generate realistic HOI videos in Figure 3.11. For “*open microwave*”, we observe that although HOI-GAN is able to generate conventional colors for a microwave, it shows limited capability to hallucinate such large objects. For “*cut peach*” (Figure 3.11), the generated sample shows that our model can learn the increase in count of partial objects corresponding to the action cut and yellow-green color of a peach. However, as the model has not observed the interior of a peach during training (as *cut peach* was not in training set), it is unable to create



Figure 3.10: **How does HOI-GAN generalize over compositions?**. Training samples in the data to illustrate that HOI-GAN leverages the information available during training and learns to combine them in a meaningful way. This ability allows HOI-GAN to generalize over unseen compositions of action, object and context. We provide a few frames for each sample in the figure.

realistic transformations in the state of *peach* that show the interior clearly. Particularly, for *open microwave*, while the model is able to generate a microwave object having seen it

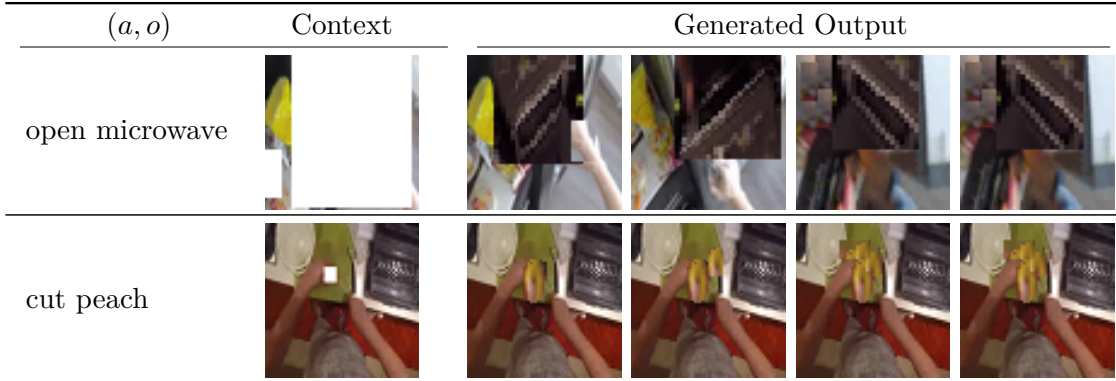


Figure 3.11: **Failure Cases.** Videos generated using HOI-GAN corresponding to the given action-object composition (a, o) and the context frame. We show 4 frames of the videos.

in training, it is not able to blend it into the background context. This is because the mask covers most of the background and the model gets very little information about the context. In the case of *cut peach*, the model is unable to generate the pieces well because the interior of a peach differs from its exterior. This is in contrast to *cut carrot* (see Figure 3.7) wherein the interior of the carrot is similar to its exterior, and hence the model is able to generate the pieces properly.

3.5 Limitations

HOI-GAN is designed to generate human-object interactions in the form of videos with fixed duration. However, in real-world, human-object interactions depend on the complexity of the action being performed with the object and are not of the same duration. This limitation can potentially be addressed by incorporating duration of an interaction in the learning scheme and by training a generator network (*e.g.* autoregressive model) that is able to generate sequences of variable length. Furthermore, to evaluate the generalizability of HOI-GAN, we only generate interactions that are semantically feasible by repurposing the train-test split of datasets. We believe this serves as a reasonable benchmark to evaluate generalizability of models for human-object interactions generation. However, to test out-of-domain generalization, more complex benchmarks could be designed by engineering the input prompts corresponding to action and object labels that together characterize a human-object interactions. This can be addressed by discerning the feasibility of prescribed interactions based on external knowledge bases or large language models such as GPT-3 [15] that are trained using large natural language corpus.

3.6 Conclusion

In this chapter, we introduced the task of zero-shot HOI video generation, *i.e.*, generating human-object interaction (HOI) videos corresponding to unseen action-object compositions, having seen the target action and target object independently. Towards this goal, we proposed the HOI-GAN framework that relies on a compositional adversarial learning scheme involving multiple discriminators each focusing on a different granularity of information in videos. Our compositional learning approach leverages object-based spatio-graph scene graph as a means to encode specific information pertaining to the task of compositional generation of HOI videos. We show that the object-level relational adversaries are effective for GAN-based generation of interaction videos. We demonstrated the effectiveness of our proposed approach on challenging HOI datasets.

Future research in human activity video generation can benefit from our idea of using compositional information at different granularities and relational information among different elements depicted in the scenes to synthesize more realistic videos. We believe modeling relational information can prove to be relevant beyond video generation in tasks such as layout-to-image translation.

Chapter 4

Learning Activity Graphs for Temporal Action Localization

This chapter focuses on the task of temporal action localization in human activity videos. This task involves predicting *which* actions occur *when* in a given activity video. The dominant paradigms in the literature process videos temporally to either propose action regions or directly produce frame-level detections. However, sequential processing of videos is problematic when the action instances have non-sequential dependencies and/or non-linear temporal ordering, such as overlapping action instances or re-occurrence of action instances over the course of the video. Thus, this chapter introduces Activity Graph Transformer, an end-to-end learnable model for temporal action localization, that receives a video as input and directly predicts a set of action instances that appear in the video. The key idea is to address this by capturing such non-linear temporal structure by reasoning over the videos as non-sequential entities in the form of graphs. Finally, this chapter discusses our findings from the extensive evaluation of the proposed model on challenging benchmark datasets used for the task of temporal action localization.

4.1 Introduction

Visual understanding of human activities in untrimmed videos involves reasoning over multiple action instances with varying temporal extents. This problem has been formally studied in the setup of temporal action localization, *i.e.*, given a human activity video, the goal is to predict a set of action labels and their corresponding start and end timestamps indicating their occurrence in the video. Reasoning over untrimmed human activity videos for action localization is particularly challenging due to the idiosyncrasies of the videos such as: (1) overlap - the action instances may have overlaps in their temporal extents indicating non-sequential temporal ordering of the instances; (2) non-sequential dependencies - some action instances may have temporal dependencies but are separated by other unrelated action instances and/or durations of no action; and (3) re-occurrence - instances belonging to

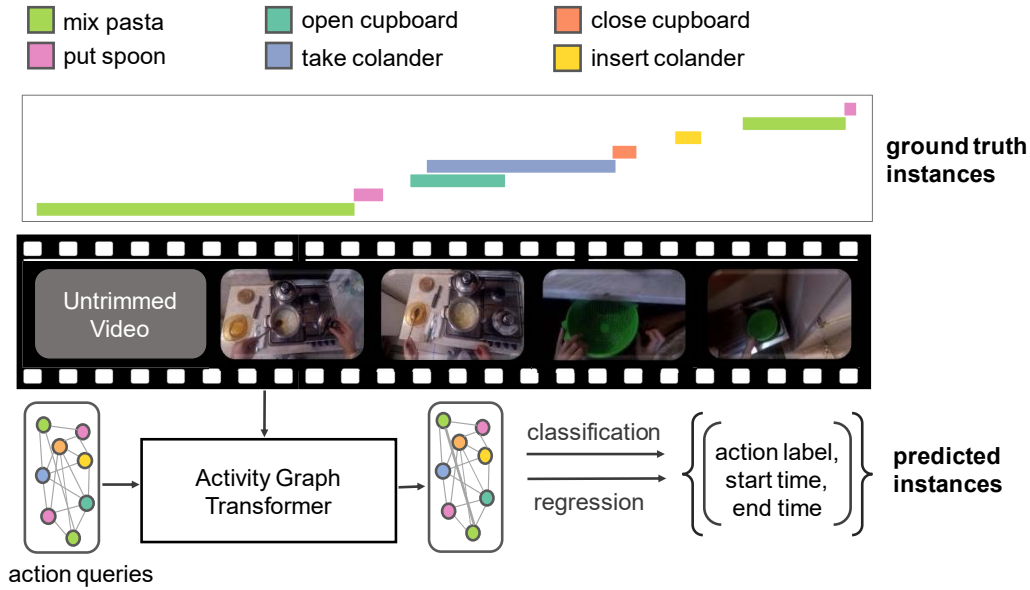


Figure 4.1: **Main Idea.** Given an untrimmed human activity video, we directly predict the set of action instances (label, start time, end time) that appear in the video. We observe that human activity videos contain non-sequential dependencies (illustrated by the overlapping ground truth instances as colored bars). In this work, we propose Activity Graph Transformer that captures this non-sequential structure by reasoning over such videos as graphs. Overall, the network receives a video and directly infers a set of action instances. The network achieves this by transforming a set of graph-structured abstract queries into contextual embeddings which are then used to provide predictions of action instances. It is trained end-to-end using classification and regression losses.

same category may appear more than once over the course of the video. In this work, we propose a novel end-to-end learnable model for temporal action localization that receives a video as an input and directly predicts the set of action instances that appear in the video.

Existing approaches for the task of temporal action localization predominantly fall into two paradigms. First is the local-then-global paradigm where the video-level predictions are obtained by postprocessing of local (*i.e.* frame-level or snippet-level) predictions using sequence modeling techniques such as recurrent neural networks, temporal convolutions and temporal pooling [38, 108, 126, 148, 180, 180, 187, 216, 269, 273]. Second is the proposal-then-classification paradigm which involves generation of a sparse set of class agnostic segment proposals from the overall video followed by classification of the action categories for each proposal using either two-stage learning [16, 18, 49, 86, 209, 211, 289, 290] or end-to-end learning [25, 33, 62, 260, 277].

The local-then-global paradigm does not utilize the overall temporal context provided by the activity in the video as the local predictions are solely based on visual information confined to the frame or the snippet. For instance, consider the example in Figure 4.1, these approaches would miss out on important relevant information provided by ‘*mix pasta*’ in

predicting ‘*put spoon*’ or may produce imprecise predictions when the temporal extents of instances ‘*take colander*’ or ‘*open cupboard*’ overlap.

Alternatively, the proposal-then-classification paradigm generates a subset of proposals by processing the video as a sequence. As a result, these approaches suffer from limited receptive field for incorporating temporal information, and do not capture non-sequential temporal dependencies effectively. This problem is further aggravated in the case of overlapping action instances. For instance, in the example in Figure 4.1, ‘*open cupboard*’ and ‘*close cupboard*’ share information but are separated by other, potentially overlapping, action instances such as ‘*take colander*’. Due to such ordering, when generating proposals corresponding to ‘*close cupboard*’, these approaches are unlikely to capture the dependency with the visual information pertaining to ‘*open cupboard*’. Furthermore, these approaches use heuristics to perform non-maximal suppression of proposals that might result in imprecise localization outcomes when the action instances vary widely in their temporal extents.

As such, both these types of approaches process videos sequentially to either generate direct local predictions or action proposals and are problematic when action instances reoccur, overlap, or have non-sequential dependencies. These observations suggest that although a video has a linear ordering of frames, the reasoning over the video need not be sequential. We argue that modeling the non-linear temporal structure is a key requirement for effective reasoning over untrimmed human activity videos. In this work, we seek a temporal action localization model that: (1) captures the temporal structure in complex human activity videos, (2) does not rely on heuristics or postprocessing of the predictions, and (3) is trained end-to-end.

Towards this goal, we formulate temporal action localization as a direct set prediction task. We propose a novel temporal action localization model, *Activity Graph Transformer* (AGT), an end-to-end learnable model that receives a video as input and predicts the set of action instances that appear in the video. In order to capture the non-linear temporal structure in videos, we reason over videos as non-sequential entities, *i.e.*, learnable graphs. Particularly, we map the input video to graph embeddings using an encoder-decoder transformer that operates using graph attention. A feed forward network then uses these embeddings to directly predict the action instances. As such, we propose a *streamlined and end-to-end trainable* approach that does not rely on any heuristic or postprocessing.

To summarize, we propose Activity Graph Transformer (AGT) that reasons over videos as graphs and can be trained end-to-end, and we demonstrate effectiveness of our approach on challenging human activity datasets: ActivityNet-1.3 [19], THUMOS14 [101], EPIC-Kitchens100 [36], and Charades [213].

4.2 Related Work

In this section, we discuss the prior work relevant to temporal action localization and graph based modeling in videos.

Temporal Action Localization. Early methods for temporal action localization used computationally inefficient approach of using temporal sliding windows for all possible sizes and locations a video and designed hand-crafted features to classify action with each window [60, 99, 171, 228, 272].

A body of recent temporal localization work falls into local-then-global paradigm. These methods rely on obtaining temporal boundaries of actions based on local predictions and perform video-level reasoning using temporal modeling techniques such as modeling of action durations or transitions [187, 273], recurrent networks [38, 148, 216, 269], temporal pooling [108], temporal convolutions [126, 180], and temporal attention [180]. These approaches do not utilize the overall temporal context of the videos as local predictions are computed using only the frame/snippet information.

Alternatively, another body of work belongs to proposal-then-classification paradigm that formulates temporal action localization as the mirror problem of object detection in the temporal domain. These methods generate a set of class-agnostic segment proposals and classify a subset of the proposals. Several methods employ a two-stage training framework with most methods focusing on improving the proposal generation stage [9, 16, 18, 49, 86, 140, 141, 262, 289] and a few aiming to improve classification stage [209, 290]. Inspired by improvements in object detection methods [68], recent methods also propose end-to-end trainable models [25, 33, 62, 260, 277]. This class of models suffer from: (1) limited temporal receptive field due to sequential processing, thus, are unable to capture non-sequential dependencies, and (2) error-prone localization due to heuristics based non-maximal suppression in case of high variations in the temporal extents. However, non-sequential dependencies and high variations in temporal extents of actions are commonplace in activity videos and need to be addressed. In contrast to the above approaches, we propose a streamlined action localization approach which does not rely on heuristics or postprocessing and can be trained end-to-end. Our proposed approach formulates temporal action localization as a direct set prediction task – it receives a video as input and gives a set of actions instances as output. Based on the intuition that the reasoning over videos is not necessarily sequential, we model the video (input) as well as action instances (output) as undirected graphs.

With the advances in transformer-based architectures, recent methods employ transformer based designs for temporal action localization [143, 182, 206, 247, 279] wherein a video is modeled as sequence of segment-level features. In contrast, we employ a transformer architecture that operates over graphs and models interactions using graph-based attention mechanisms. The self-attention mechanisms operate over the graphs corresponding to the

full videos, thereby capturing the full context of the video, *i.e.*, all possible interactions and their corresponding intensities (as learnable weights) in the videos.

Action Recognition. Action recognition methods [34,98,125,214,229,242,243,244,250,278] operate on short video clips that are trimmed such that a single action instance spans the video duration and, hence, are not suitable for untrimmed videos containing multiple actions. Nonetheless, models pretrained for the task of action recognition provide effective feature representations for tasks that focus on untrimmed videos. In this work, we use pretrained I3D [214] models for feature extraction.

Graph-based Modeling for Videos. The advances in graph convolutional networks (GCNs) [115] have inspired several recent approaches for video based tasks [162,173,250]. Most of the graph based approaches for videos represent either the input space (*i.e.* videos or derived visual information) as graphs [93,162,173,250,262,286] or the output space (*i.e.* labels) as graphs [230]. In contrast, we design our model based on the insight that both the input space (*i.e.* features derived from videos) and the output space (*i.e.* set of action instances) potentially contain non-sequential relations, thus, are graph-structured. Specifically, in our approach, the model learns the mapping between the graph-structured input and output spaces by learning the graph structure (*i.e.* both nodes and edges) from the data using self-attention mechanisms.

Transformers in Computer Vision. Inspired by advances in transformer architectures in NLP tasks [191,234], transformer based models have gathered attention in computer vision. Methods such as DETR [21] for object detection and segmentation; ViTR [46] for image classification; Video Action Transformer [65] for spatio-temporal localization; VisTR [251] for video instance segmentation; Trackformer [158] for multi-object tracking; ActionFormer [279], ReACT [206] for temporal action localization are some successful examples of application of transformers for both image and video based tasks. In this work, we propose a graph transformer model for temporal action localization in videos. Our model is inspired by DETR [21] in that we use a transformer model for a set prediction task. However, by design, our model AGT is different from [21]: (1) it operates over input space and output space as graphs, (2) it relies on graph based self-attention mechanisms to model dependencies in data. Our work here demonstrates a novel transformer architecture for videos, with graphs as an effective means to capture dependencies and a streamlined end-to-end trained localization pipeline.

4.3 Proposed Approach

The task of temporal action localization involves prediction of the category labels as well as start and end timestamps of the actions that occur in a given video. In this work, we formulate this task as a direct set prediction problem, wherein each element in the predicted

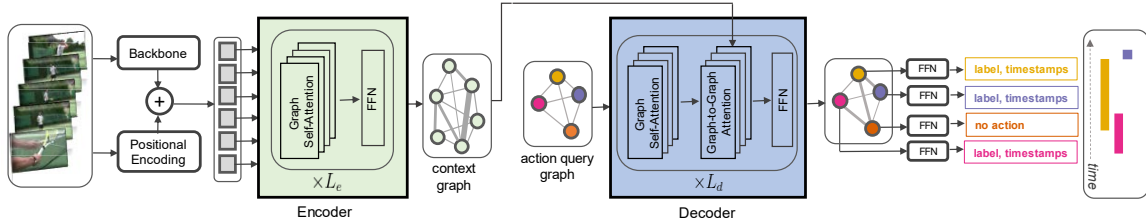


Figure 4.2: **Model Overview.** Activity Graph Transformer (AGT) receives a video as input and directly predicts a set of action instances that appear in the video. The input video is fed into a backbone network to obtain a compact representation. Then, the encoder network receives the compact video-level representation from the backbone network and encodes it to a latent graph representation *context graph*. The decoder network receives the context graph along with graph-structured abstract query encodings *action query graph*. The decoder transforms the action query graph to a graph-structured set of embeddings. Each node embedding of the decoder output is fed into a prediction head. The network is trained end-to-end using classification and regression losses for the action labels and timestamps of the action instances respectively.

set denotes an action instance in the video. Specifically, given a video V , the goal is to predict a set \mathcal{A} where the i -th element $a^{(i)} = (c^{(i)}, t_s^{(i)}, t_e^{(i)})$ denotes an action instance in the video depicting action category $c^{(i)}$ that starts at time $0 \leq t_s^{(i)} \leq T$, ends at time $0 \leq t_e^{(i)} \leq T$, for $i \in \{1, 2, \dots, |\mathcal{A}|\}$. Here, $|\mathcal{A}|$ is the number of action instances present in the video and T is the duration of the video. Thus, $|\mathcal{A}|$ and T vary based on the input video.

Towards this goal, we propose *Activity Graph Transformer* (AGT), an end-to-end learnable model that receives a video as input and directly infers the set of action instances in the video. Our approach consists of: (1) a network that predicts a set of action instances in a single forward pass; and (2) a loss function to train the network by obtaining a unique alignment between the predicted and ground truth action instances. We contend that effective reasoning over untrimmed human activity videos requires modeling the non-linear temporal structure in the videos. In our approach, we seek to capture this structure by employing graphs. Specifically, we propose a novel graph transformer network that leverages graph based self-attention to reason over videos. We describe the details of our approach below.

4.3.1 Activity Graph Transformer

As shown in Figure 4.2, Activity Graph Transformer (AGT) consists of three components: (1) backbone network to obtain a compact representation of the input video; (2) transformer network consisting of an encoder and a decoder network that operates over graphs; and (3) prediction heads for the final prediction of action instances of the form (label, start time, end time). We employ an encoder-decoder transformer network because the length of the video is not directly related to number of action instances.

The encoder network receives the video representation from the backbone network and encodes it to a latent graph representation, referred to as *context graph*. The decoder network receives graph-structured abstract query encodings (referred to as *action query graph*) as input along with the context graph and transforms the action query graph to a graph-structured set of embeddings. Each node embedding of this decoder output is fed into a feedforward network to obtain predictions of action instances. The overall AGT network is trained end-to-end using a combination of classification and regression losses for the action labels and the timestamps respectively. We provide description of the components below.

Backbone. To obtain a compact representation for the input video V containing T frames, any 3D convolutional network can be used to extract the features. In our implementation, we chunk the videos into short overlapping segments and use an I3D model [22] dataset to extract features of dimension C ($= 2048$) from the segments, resulting in video-level feature $\mathbf{v} = [\mathbf{v}^{(1)}, \mathbf{v}^{(2)} \dots \mathbf{v}^{(N_v)}]$ where N_v is the number of chunks used in the feature extraction.

Transformer Encoder. The backbone simply provides a sequence of local features and does not incorporate the overall context of the video or the temporal structure in the video. Therefore, we use an encoder network that receives the video-level feature as input and encodes this video representation to a graph (referred to as the *context graph*). Intuitively, the encoder is designed to model the interactions among the local features using self-attention modules.

The context graph is initialized with video-level feature $\mathbf{v}^{(i)}$ (of dimension $C = 2048$) as the i -th node for $i \in \{1, 2, \dots, N_v\}$. Usually, transformer networks use fixed positional encoding [175] to provide position information of each element in the input sequence. In contrast, in our setting, we contend that the video features have a non-linear temporal structure. Thus, we provide the positional information using learnable positional encodings \mathbf{p}_v as additional information to the video feature \mathbf{v} . The positional encoding $\mathbf{p}_v^{(i)}$ corresponds to the i -th node in the graph and is of the same dimension as the node. Next, the graph nodes are from the same video and hence, they are related to each other. However, their connection information (edges) is not known a priori. Thus, we model the interactions among these nodes as learnable edge weights. This is enabled by the graph self-attention module (described below).

We design the transformer encoder network \mathbf{E} as a sequence of L_e blocks, wherein, an encoder block \mathbf{E}_ℓ for $\ell \in \{1, 2, \dots, L_e\}$ consists of a graph self-attention module followed by a feed forward network. The output of the encoder network is the context graph $\mathbf{h}_{L_e} = [\mathbf{h}_{L_e}^{(1)}, \mathbf{h}_{L_e}^{(2)} \dots \mathbf{h}_{L_e}^{(N_v)}]$ where $\mathbf{h}_{L_e}^{(i)}$ is the i -th node and is of dimension d (same for each block). The output of the ℓ -th encoder block \mathbf{h}_ℓ and the final output of the encoder \mathbf{h}_{L_e} are defined

as:

$$\begin{aligned}
\mathbf{h}_0 &= \mathbf{v} \\
\mathbf{h}_\ell &= \mathbf{E}_\ell(\mathbf{h}_{\ell-1}, \mathbf{p}_v) \\
\mathbf{h}_{L_e} &= \mathbf{E}_{L_e} \circ \dots \circ \mathbf{E}_1(\mathbf{v}, \mathbf{p}_v).
\end{aligned} \tag{4.1}$$

Graph Self-Attention. This module aims to model interactions among graph structured variables along with learnable edge weights. Here, we describe the graph self-attention module in the context of the $(\ell + 1)$ -th encoder block $\mathbf{E}_{\ell+1}$. For simplicity of notation, let \mathbf{x} be the output of the ℓ -th block of the encoder, *i.e.*, $\mathbf{x} = \mathbf{E}_\ell(\mathbf{h}_{\ell-1}, \mathbf{p}_v)$. \mathbf{x} is a graph contains N_v nodes $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N_v)}$ which are connected using learnable edge weights. The graph self-attention module first performs graph message passing (as described in [236]) to produce the output \mathbf{x}' , with the i -th node of the output defined as

$$\mathbf{x}'^{(i)} = \mathbf{x}^{(i)} + \left\| \sum_{k=1}^K \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}_g^k \mathbf{x}^{(j)} \right) \right\|, \tag{4.2}$$

where $\left\| \right\|$ represents concatenation operator, K is the number of parallel heads in the self-attention module, σ is a non-linear function (leaky ReLU in our case), \mathcal{N}_i represents the set of neighbours of the i -th node, \mathbf{W}_g^k is the learnable transformation weight matrix. α_{ij}^k are the self attention coefficients computed by the k -th attention head described as:

$$\alpha_{ij}^k = \frac{\exp(f(\mathbf{w}_{a,k}^T [\mathbf{W}_g^k \mathbf{x}^{(i)} \parallel \mathbf{W}_g^k \mathbf{x}^{(j)}]))}{\sum_{m \in \mathcal{N}_i} \exp(f(\mathbf{w}_{a,k}^T [\mathbf{W}_g^k \mathbf{x}^{(i)} \parallel \mathbf{W}_g^k \mathbf{x}^{(m)}]))}, \tag{4.3}$$

where \cdot^T represents a transpose operator, f is a non-linear activation (leaky ReLU in our case) and $\mathbf{w}_{a,k}$ is the attention coefficients. α_{ij}^k is the attention weight and denotes the strength of the interaction between i -th and j -th node of the input graph of the module. Subsequent to the message passing step, we apply batch normalization and a linear layer. This is then followed by a standard multi-head self-attention layer (same as in [234]). Overall, the graph self-attention module models interactions between the nodes, *i.e.*, local features derived from the video.

Transformer Decoder. Based on the observation that the action instances in the video have a non-linear temporal structure, we design the decoder to learn a graph-structured set of embeddings which would subsequently be used for predicting the action instances. Intuitively, the output graph provided by the decoder serves as the latent representation for the set of action instances depicted in the video. The inputs of the transformer decoder are: (1) graph-structured abstract query encodings, referred to as action query graph \mathbf{q} , containing N_o nodes wherein each node is a learnable positional encoding of dimension d (same as the dimension used in the encoder); and (2) the context graph \mathbf{h}_{L_e} containing

N_v nodes (obtained from the encoder). Here, the nodes in the action query graph are just learnable positional encodings that are fully connected. Regardless of the input video, the action query graph contains a fixed number of nodes (which is a hyperparameter). Thus, whether during training or testing, the only input to the model is the video. During testing, the action query graph (input to decoder) is unchanged since the (positional) encoding layer is frozen.

We assume that the number of nodes in the action query graph N_o is fixed and is sufficiently larger than the maximum number of action instances per video in the dataset. This idea of using representations of prediction entities as positional query encodings is inspired from [21]. To learn the interactions among the graph-structured query embeddings, we use graph self-attention modules (same module as used in transformer encoder). Additionally, we use graph-to-graph attention module (described below) to learn interactions between the context graph, *i.e.*, latent representation of the input video, and graph-structured query embeddings, *i.e.*, latent representations of the action queries.

The overall decoder network \mathbf{D} consists of L_d blocks, wherein, a block $\mathbf{D}_{\ell'}$ for $\ell' \in \{1, 2, \dots, L_d\}$ consists of a graph self-attention module followed by a graph-to-graph attention module, and then a feed forward network. The block $\mathbf{D}_{\ell'}$ has the output $\mathbf{y}_{\ell'}$ and final output of the decoder $\mathbf{y}_{L_d} = [\mathbf{y}_{L_d}^{(1)}, \mathbf{y}_{L_d}^{(1)}, \dots, \mathbf{y}_{L_d}^{(N_o)}]$. They are defined as:

$$\begin{aligned} \mathbf{y}_0 &= \mathbf{q} \\ \mathbf{y}_{\ell'} &= \mathbf{D}_{\ell'}(\mathbf{y}_{\ell'-1}, \mathbf{h}_{L_e}) \\ \mathbf{y}_{L_d} &= \mathbf{D}_{L_d} \circ \dots \circ \mathbf{D}_1(\mathbf{q}, \mathbf{h}_{L_e}) \end{aligned} \tag{4.4}$$

Graph-to-Graph Attention. The graph-to-graph attention module aims to learn the interactions between two different graphs referred to as a source graph and a target graph. Here, we describe this module in the context of the decoder block $\mathbf{D}_{\ell'+1}$. The input to this block is the output $\mathbf{y}_{\ell'}$ of the previous decoder block $\mathbf{D}_{\ell'}$. This is fed to the graph self-attention module in the block $\mathbf{D}_{\ell'+1}$, and the output is used as the target graph for the graph-to-graph attention module. The source graph for this module (in any decoder block) is the context graph \mathbf{h}_{L_e} . For simplicity of notation, let \mathbf{x}_s denote the source graph (*i.e.* \mathbf{h}_{L_e}) and \mathbf{x}_t denote the target graph. Here, the source and target graphs may contain different number of nodes. In our case, \mathbf{x}_s contains N_v nodes and \mathbf{x}_t contains N_o nodes. The graph-to-graph attention module performs message passing from source graph to target graph to provide an output \mathbf{x}'_t , with the i -th node defined as

$$\mathbf{x}'_t^{(i)} = \mathbf{x}_t^{(i)} + \left\| \sum_{k=1}^K \sigma \left(\sum_{j \in \mathcal{N}_i} \beta_{ij}^k \mathbf{W}_s^k \mathbf{x}_s^{(j)} \right) \right\|, \tag{4.5}$$

where \mathbf{W}_s^k is the learnable transformation weight matrix for the source graph. Other symbols denote the same entities as in graph self-attention section. β_{ij}^k is the attention coefficient

for k -th attention head between i -th node of the source graph and j -th node of the target graph computed as:

$$\beta_{ij}^k = \frac{\exp(f(\mathbf{w}_{a,k}^{stT}[\mathbf{W}_s^k \mathbf{x}_s^{(i)} \parallel \mathbf{W}_t^k \mathbf{x}_t^{(j)}]))}{\sum_{m \in \mathcal{N}_i} \exp(f(\mathbf{w}_{a,k}^{stT}[\mathbf{W}_s^k \mathbf{x}_s^{(i)} \parallel \mathbf{W}_t^k \mathbf{x}_t^{(m)}]))} \quad (4.6)$$

where $\mathbf{w}_{a,k}^{st}$ is the graph-to-graph attention coefficients, and \mathbf{W}_s^k and \mathbf{W}_t^k are the learnable transformation weight matrices for source and target graphs respectively. Other symbols denote the same entities as in graph self-attention. Similar to the transformer encoder, subsequent to the message passing step, we apply batch normalization and a linear layer. This is then followed by a standard multi-head self-attention layer (same as in [234]). Overall, the graph-to-graph attention module models the interactions between the latent representations of the input video and the action queries.

Prediction Heads. The decoder network provides a set of embeddings where the embeddings serve as the latent representations for the action instances in the video. This output graph \mathbf{y}_{L_d} contains N_o nodes. We use these N_o node embeddings to obtain predictions for N_o action instances using prediction heads. The prediction heads consist of a feed forward network (FFN) with ReLU activation which provides the start time and end time of the action instance normalized with respect to the overall video duration. Additionally, we use a linear layer with a softmax function to predict the categorical label corresponding to the action instance.

Therefore, when provided with the i -th node embedding $\mathbf{y}_{L_d}^{(i)}$, the prediction head provides prediction $\tilde{a}^{(i)} = (\tilde{c}^{(i)}, \tilde{t}_s^{(i)}, \tilde{t}_e^{(i)})$ where $\tilde{c}^{(i)}$, $\tilde{t}_s^{(i)}$ and $\tilde{t}_e^{(i)}$ are the category label, start time and end time for the i -th action instance for $i \in \{1, 2, \dots, N_o\}$. Note that the ground truth set contains a variable number of action instances, whereas N_o is larger than the maximum number of action instances per video in the dataset. To suppress irrelevant predictions, we introduce an additional class label \emptyset indicating no action (similar to [21]). Typically, existing models perform non-maximal suppression (NMS) typically performed using heuristics [25]. In contrast, NMS is learnable in our model.

Loss functions. To train the overall network, we align the predictions with the ground truth action instances using a matcher module which optimizes a pair-wise cost function. This provides a unique matching between the predicted and ground truth action instances. Subsequently, our model computes losses over these matched pairs of predicted and ground truth action instances to train the network end-to-end.

Matcher. The matcher module finds an optimal matching between the predicted set of action instances (fixed number of elements for every video) and the ground truth set of action instances (variable number of elements depending on the video). To obtain this matching, we design a matching cost function and use the Hungarian algorithm to obtain the optimal matching between these two sets following [224].

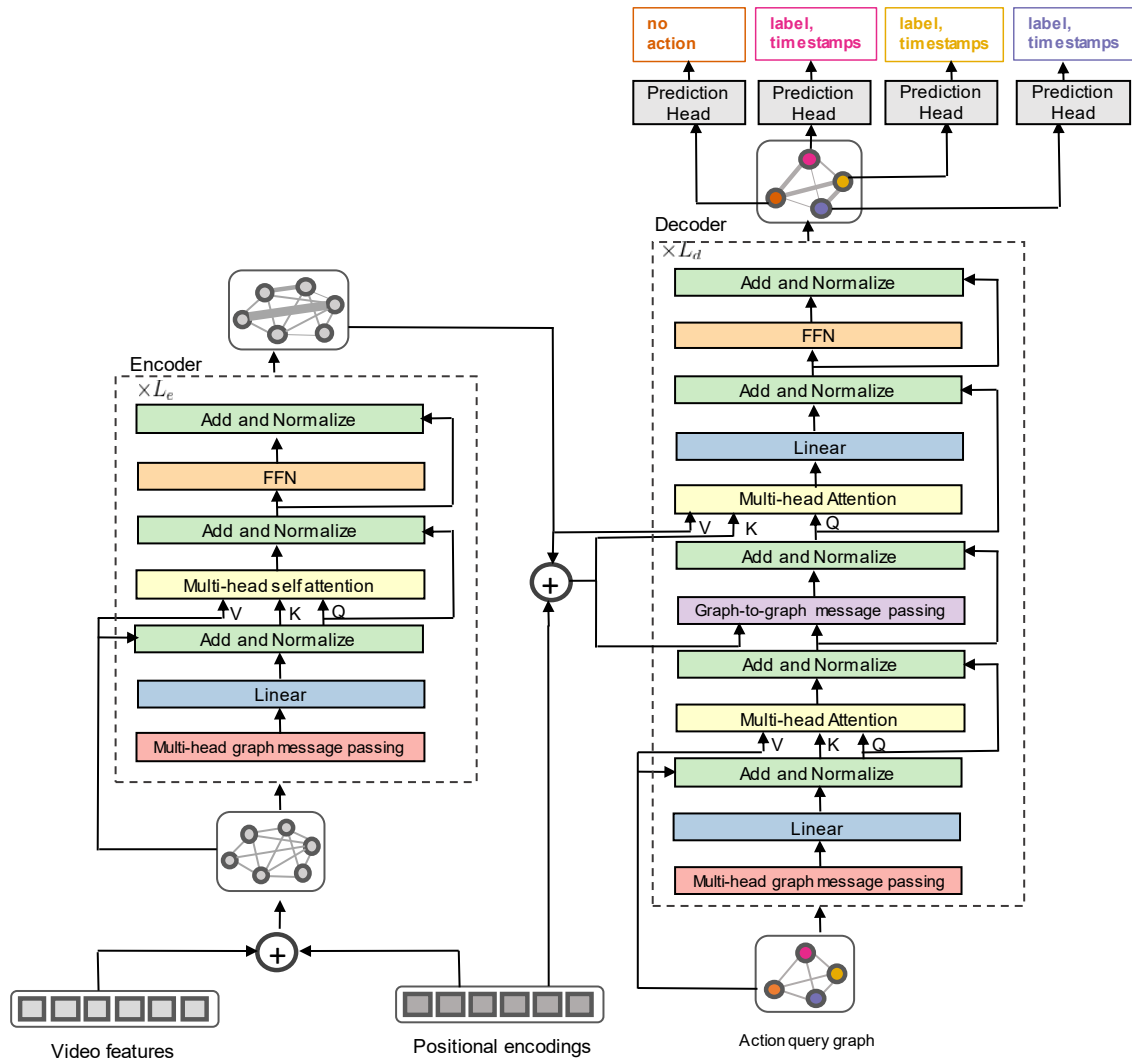


Figure 4.3: **Detailed Architecture** Architecture of Activity Graph Transformer. ‘Q’, ‘K’, ‘V’ are query, key and value to the self-attention layer as described in [234].

Formally, let \mathcal{A} be the ground truth set of action instances $\mathcal{A} = \{a^{(i)}\}_{i=1}^{|\mathcal{A}|}$, where $a^{(i)} = (c^{(i)}, t_s^{(i)}, t_e^{(i)})$ and $\tilde{\mathcal{A}}$ be the predicted set of action instances $\tilde{\mathcal{A}} = \{\tilde{a}^{(i)}\}_{i=1}^{N_o}$ where $\tilde{a}^{(i)} = (\tilde{c}^{(i)}, \tilde{t}_s^{(i)}, \tilde{t}_e^{(i)})$.

In our model, we assume that N_o is larger than the number of actions in any video in the dataset. Therefore, we assume that ground truth set \mathcal{A} also is a set of size N_o by padding the remaining $(N_o - |\mathcal{A}|)$ elements with \emptyset element indicating no action. The optimal bipartite matching provides a permutation of N_o elements $\hat{\phi}$ from the set of all possible permutations Φ_{N_o} that results in lowest value of the matching cost function \mathcal{L}_m . Thus, $\hat{\phi} = \operatorname{argmin}_{\phi \in \Phi_{N_o}} \mathcal{L}_m(a^{(i)}, \tilde{a}^{(\phi(i))})$, where $\mathcal{L}_m(a^{(i)}, \tilde{a}^{(\phi(i))})$ is the matching cost function between ground truth $a^{(i)}$ and prediction with index $\phi(i)$. The matching cost function incorporates the class probabilities of the action instances and the proximity between predicted and ground truth timestamps. Specifically, we define the cost function as:

$$\begin{aligned} \mathcal{L}_m(a^{(i)}, \tilde{a}^{(\phi(i))}) &= -\mathbb{1}_{\{c^{(i)} \neq \emptyset\}} \tilde{p}_{\phi(i)}(c^{(i)}) \\ &\quad + \mathbb{1}_{\{c^{(i)} \neq \emptyset\}} \mathcal{L}_s(s^{(i)}, \tilde{s}^{(\phi(i))}), \end{aligned} \quad (4.7)$$

where $s^{(i)} = [t_s^{(i)}, t_e^{(i)}]$ and $\tilde{s}^{(\phi(i))} = [\tilde{t}_s^{(\phi(i))}, \tilde{t}_e^{(\phi(i))}]$, and $\tilde{p}_{\phi(i)}(c^{(i)})$ is the probability of the class $c^{(i)}$ for prediction $\phi(i)$ and \mathcal{L}_s represents segment loss that measures proximity in the timestamps of the instances. The segment loss is defined as a weighted combination of an L_1 loss (sensitive to the durations of the instances) and an IoU loss (invariant to the durations of the instances) between the predicted and ground-truth start and end timestamps. It is expressed as:

$$\mathcal{L}_s = \lambda_{iou} \mathcal{L}_{iou}(s^{(i)}, \tilde{s}^{(\phi(i))}) + \lambda_{L1} \|s^{(i)} - \tilde{s}^{(\phi(i))}\|_1, \quad (4.8)$$

where $\lambda_{iou}, \lambda_{L1} \in \mathbb{R}$ are hyperparameters. Here, the IoU loss is expressed as:

$$\mathcal{L}_{iou}(s^{(i)}, \tilde{s}^{(\phi(i))}) = 1 - \frac{|s^{(i)} \cap \tilde{s}^{(\phi(i))}|}{|s^{(i)} \cup \tilde{s}^{(\phi(i))}|}, \quad (4.9)$$

where $|\cdot|$ is the duration of the instance, *i.e.*, difference between end and start timestamp.

Subsequent to obtaining the optimal permutation $\hat{\phi}$, we compute the Hungarian loss \mathcal{L}_H over all the matched pairs as follows:

$$\mathcal{L}_H = \sum_{i=1}^{N_o} \left[-\log \tilde{p}_{\hat{\phi}}(c^{(i)}) + \mathbb{1}_{\{c^{(i)} \neq \emptyset\}} \mathcal{L}_s(s^{(i)}, \tilde{s}^{(\hat{\phi}(i))}) \right]. \quad (4.10)$$

This loss is used to train our AGT model end-to-end.

In summary, our proposed Activity Graph Transformer performs temporal action localization using an encoder-decoder based architecture leveraging graph based attention modules. We jointly optimize all parameters of our model to minimize the regression loss

Table 4.1: **Comparison with state-of-the-art (ActivityNet-1.3).** We report the mean average precision at different intersection over union thresholds (mAP@tIoU). These results are reported on the validation set. \uparrow indicates higher is better.

Method	mAP@tIoU \uparrow			
	0.5	0.75	0.95	Average
SCC [17]	40.00	17.90	4.70	21.70
CDC [209]	45.30	26.00	0.20	23.80
R-C3D [260]	26.8	-	-	-
SSN [290]	39.12	23.48	5.49	23.9
BSN [141]	46.45	29.96	8.02	30.03
TAL-Net [25]	38.2	18.3	1.3	20.2
PGCN [277]	48.26	33.16	3.27	31.1
BMN [140]	50.07	34.78	8.29	33.85
GTAD [262]	50.4	34.6	9.0	34.1
VSGN [286]	52.4	35.2	8.3	34.7
ActionFormer [279]	54.7	37.8	8.4	36.6
PRN [247]	59.7	-	-	42.0
TCANet [182]	56.7	41.1	12.2	39.7
AGT(Ours)	52.36	35.51	9.62	35.68

for the start and end timestamps of the action instances and the cross entropy losses for the corresponding action labels.

4.3.2 Implementation Details

In this section, we provide additional implementation details of our proposed method.

Detailed Architecture. Figure 4.3 presents the architecture of our AGT in detail showing architecture of all the three components: (1) backbone network to obtain features corresponding to the input video; (2) transformer network consisting of an encoder network and a decoder network that operates over graphs; and (3) prediction heads for the final prediction of action instances of the form (label, start time, end time).

Positional Encoding. Positional encoding layer consists of a layer that retrieves encodings based on an integer index provided to it. In our case, given a video feature $\mathbf{v} = [\mathbf{v}^{(1)}, \mathbf{v}^{(2)} \dots, \mathbf{v}^{(N_v)}]$, the positional encoding layer receives input i and provides an embedding $\mathbf{p}_v^{(i)}$ corresponding to the i -th element of the video feature $\mathbf{v}^{(i)}$ where $i \in \{1, 2, \dots, N_v\}$. In our implementation, the embedding size is same as that of the video feature so as to allow addition of the positional encodings and input video features. Since the weights of the layer are learnable during training, the positional encoding layer is learnable. We use `torch.nn.Embedding` in Pytorch to implement it. This layer initialization requires maximum possible value of N_v in the features corresponding to the video.

Table 4.2: **Comparison with state-of-the-art (THUMOS14)**. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). \uparrow indicates higher is better.

Method	mAP@tIoU \uparrow				
	0.1	0.2	0.3	0.4	0.5
Richard <i>et al.</i> [187]	39.7	35.7	30.0	23.2	15.2
Shou <i>et al.</i> [211]	47.7	43.5	36.3	28.7	19.0
SST [16]	-	-	37.8	-	23.0
CDC [209]	-	-	40.1	29.4	23.3
Yeung <i>et al.</i> [269]	48.9	44.0	36.0	26.4	17.1
Yuan <i>et al.</i> [273]	51.0	45.2	36.5	27.8	17.8
TURN-TAP [62]	60.1	56.7	50.1	41.3	31.0
R-C3D [260]	54.5	51.5	44.8	35.6	28.9
SSN [290]	66.0	59.4	51.9	41.0	29.8
BSN [141]	-	-	53.5	45.0	36.9
TAL-Net [25]	59.8	57.1	53.2	48.5	42.8
PGCN [277]	69.5	67.8	63.6	57.8	49.1
BMN [140]	56.0	47.4	38.8	29.7	20.5
GTAD [262]	-	-	66.4	60.4	51.6
VSGN [286]	-	-	66.7	60.4	52.4
TCANet [182]	-	-	60.6	53.2	44.6
ActionFormer [279]	-	-	82.1	77.8	71.0
ReACT [206]	-	-	69.2	65.0	57.1
AGT(Ours)	72.1	69.8	68.4	62.3	52.1

Action Query Graph. Similar to positional encoding layer, the N_o encodings in the action query graph \mathbf{q} is obtained using an embedding layer. Specifically, the layer receives i as input to provide i -th node $\mathbf{q}^{(i)}$ of the query graph where $i \in \{1, 2, \dots, N_o\}$. In our implementation, we use `torch.nn.Embedding` in Pytorch to implement this. The weights of this layer are learnable during training.

4.4 Experiments

We conducted several experiments to demonstrate the effectiveness of our proposed approach. In this section, we report the results of our evaluation.

4.4.1 Experimental Setup

Datasets. We use four benchmark datasets for evaluation. They vary in their extent of overlap in action instances, the number of action instances per video, and the number of action categories in the dataset. Thus, these datasets together serve as a challenging testbed.

- **ActivityNet-1.3** [19] contains around 10K training videos and 5K validation videos spanning 200 action categories. The dataset contains 1.65 instances per video on average.

Table 4.3: **Comparison with state-of-the-art (EPIC-Kitchens100)**. We report mean average precision at different intersection over union thresholds (mAP@tIoU). These results are reported on the validation set. \uparrow indicates higher is better.

Method	Task	mAP@tIoU \uparrow				
		0.1	0.2	0.3	0.4	0.5
Damen <i>et al.</i> [36]	Verb	10.51	9.24	7.67	6.40	5.12
	Noun	10.71	8.73	6.75	5.05	3.35
	Action	6.78	6.03	4.94	4.04	3.35
GTAD [262]	Verb	12.1	11.0	9.4	7.10	5.6
	Noun	11.0	10.0	8.6	6.5	5.4
AGT (Ours)	Verb	12.01	10.25	8.15	7.12	6.14
	Noun	11.63	9.33	7.05	6.57	3.89
	Action	7.78	6.92	5.53	4.22	3.86

Table 4.4: **Comparison with state-of-the-art (Charades)**. We report mean average precision (mAP) computed following [213]. \uparrow : higher is better.

Method	mAP \uparrow
Predictive-corrective (Dave <i>et al.</i> [38])	8.9
Two-stream (Siggurdson <i>et al.</i> [213])	8.9
Two-stream + LSTM (Siggurdson <i>et al.</i> [213])	9.6
R-C3D (Xu <i>et al.</i> [260])	12.7
SSN (Zhao <i>et al.</i> [290])	16.4
I3D baseline [178]	17.2
Super-events (Piergiovanni <i>et al.</i> [180])	19.4
TGM (Piergiovanni <i>et al.</i> [180])	22.3
Mavroudi <i>et al.</i> [156]	23.7
3D ResNet-50 + super-events (Piergiovanni <i>et al.</i> [179])	25.2
AGT (Ours)	28.6

- **THUMOS14** [101] contains 200 videos in training set and 213 videos in testing set. This dataset has 20 action categories and contains an average of 15 action instances per video with an average of 8% overlapping instances.
- **EPIC-Kitchens100** [36] contains 700 egocentric videos of daily kitchen activities. This dataset contains 289 noun and 97 verb classes. The dataset has 128 action instances per video with an average of 28% overlapping instances.
- **Charades** [213] is large scale dataset containing 9848 videos of daily indoor activities. This dataset has 157 action categories. Videos in the dataset contain an average of 6 action instances per video with an average of 79% of overlapping instances in a video. This dataset is challenging because of the high degree of overlap in the action instances.

Table 4.5: **Impact of graph based reasoning (ActivityNet-1.3)** We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. ✓ and ✗ indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. ↑ indicates higher is better.

Method	mAP@tIoU ↑			
	0.5	0.75	0.95	Average
E: ✗/ D: ✗	49.8	32.9	8.9	32.1
E: ✗/ D: ✓	50.6	33.4	9.0	32.6
E: ✓/ D: ✗	51.8	34.1	9.2	33.6
E: ✓/ D: ✓	52.4	35.5	9.6	35.7

Table 4.6: **Impact of graph based reasoning (THUMOS14)** We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. ✓ and ✗ indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. ↑ indicates higher is better.

Method	mAP@tIoU ↑					Average
	0.1	0.2	0.3	0.4	0.5	
E: ✗/ D: ✗	64.6	60.8	59.1	51.2	40.3	57.6
E: ✗/ D: ✓	65.1	62.4	60.3	52.4	41.3	58.3
E: ✓/ D: ✗	67.1	64.4	62.5	53.6	44.9	60.3
E: ✓/ D: ✓	72.1	69.8	68.4	62.3	52.1	65.1

Evaluation Metrics. To measure the performance of a model, we use a widely used metric mean Average Precision (mAP). We compute the mean of average precision (AP) per class. A predicted segment is considered true positive if their Intersection over Union (IoU) is greater than or equal to a given threshold.

Training Details. We describe the details of the setup used for training AGT for each of the datasets.

Data augmentation. To prevent severe overfitting, we perform data augmentation to train our model on the features directly obtained from I3D model (described above). We use a hyperparameter N_v^{max} as the maximum size of temporal channel used for training. This helps in stabilizing the training as the video datasets contain high variance in their duration. If the size of temporal channel of the video tensor T' is less than N_v^{max} , we repeat each element in the temporal channel γ times ($\gamma = 4$) in our implementation to obtain a modified tensor of size $\gamma T' \times 2048$ and then randomly sample T' elements from the modified tensor. If the size of temporal channel of the video tensor T' is more than N_v^{max} , we just randomly sample T' elements from the modified tensor. Note that, positional encoding is applied on this feature of size $N_v = \min(T', N_v^{max})$.

We find such data augmentation during training to be crucial to prevent overfitting and obtain good performance of our model, especially for smaller datasets such as THUMOS14 and Epic-Kitchens100. During testing, if the size of temporal channel of the video tensor T'

Table 4.7: **Impact of graph based reasoning (Charades)** We report the mean average precision for ablated versions of our AGT model. ✓ and ✗ indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. ↑ indicates higher is better.

Method	mAP (↑)
E: ✗ / D: ✗	18.2
E: ✗ / D: ✓	19.2
E: ✓ / D: ✗	22.5
E: ✓ / D: ✓	28.6

Table 4.8: **Impact of graph based reasoning (EPIC-Kitchens100)** We report the mean average precision at different intersection over union thresholds (mAP@tIoU) for ablated versions of our AGT model. ✓ and ✗ indicates whether a component (encoder E or decoder D) contains graph message passing module or not respectively. ↑ indicates higher is better.

Task	Method	mAP@tIoU ↑					Average
		0.1	0.2	0.3	0.4	0.5	
Verb	E: ✗ / D: ✗	9.4	6.9	5.2	4.5	2.5	5.7
	E: ✗ / D: ✓	9.9	7.5	5.5	4.9	2.7	6.1
	E: ✓ / D: ✗	11.4	9.0	6.9	6.3	3.4	7.4
	E: ✓ / D: ✓	12.0	10.3	8.2	7.1	6.1	8.7
Noun	E: ✗ / D: ✗	8.9	5.4	4.9	3.6	1.7	4.9
	E: ✗ / D: ✓	9.2	6.0	5.1	4.2	2.0	5.3
	E: ✓ / D: ✗	10.1	8.0	6.8	5.2	2.3	6.3
	E: ✓ / D: ✓	11.6	9.3	7.1	6.6	3.9	7.7
Action	E: ✗ / D: ✗	4.8	4.1	2.9	2.1	1.5	3.0
	E: ✗ / D: ✓	5.1	4.3	3.2	2.3	1.8	3.3
	E: ✓ / D: ✗	7.3	6.1	5.0	3.9	3.7	5.1
	E: ✓ / D: ✓	7.8	6.9	5.5	4.2	3.9	5.9

is less than N_v^{max} , we don’t perform any augmentation. If the size of temporal channel of the video tensor T' is more than N_v^{max} , we uniformly sample T' elements from the feature in order to match the maximum index of the positional encoding layer.

Furthermore, to perform training in minibatches, we apply 0-padding to ensure all elements have the same size as the largest element of the batch. For training efficiency and minimizing the amount of 0-padding, we sort all the dataset based on the duration of the video. We observe that this type of batch formation leads to improvement in training speed without affecting the model performance.

Feature Extraction. For THUMOS14 and ActivityNet-1.3 datasets, we use the features provided by [262]. For Epic-Kitchens100 datasets, we use the TSN features corresponding to the RGB and Flow streams used by [36]. For Charades, we first divide the video into short overlapping segments of 8 frames and use an I3D model pretrained on the Kinetics [22] dataset to extract features. In our implementation, we obtain two-stream features (both RGB and flow stream) with overlap of 4 frames, i.e., we obtain features for $T' = \lfloor \frac{T-4}{4} \rfloor$ chunks to obtain a tensor of size $T' \times 2048$. The features from each stream are

Table 4.9: **Ablation Study: Loss function (ActivityNet-1.3)** We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.

	mAP@tIoU ↑			
	0.5	0.75	0.95	Average
L_1 : ✓/ \mathcal{L}_{iou} : ✗	51.3	33.7	9.2	34.3
L_1 : ✗/ \mathcal{L}_{iou} : ✓	51.0	33.4	9.0	34.1
L_1 : ✓/ \mathcal{L}_{iou} : ✓	52.4	35.5	9.6	35.7

Table 4.10: **Ablation Study: Loss function (THUMOS14)** We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.

Method	mAP@tIoU ↑					Average
	0.1	0.2	0.3	0.4	0.5	
L_1 : ✓/ \mathcal{L}_{iou} : ✗	71.0	68.7	67.8	62.0	51.2	61.3
L_1 : ✗/ \mathcal{L}_{iou} : ✓	70.6	68.5	67.1	61.8	50.6	59.6
L_1 : ✓/ \mathcal{L}_{iou} : ✓	72.1	69.8	68.4	62.3	52.1	65.1

concatenated along the channel dimension. Here, the length of the video T depends on the duration of the video, and, hence the size of the temporal channel (*i.e.* T') of the feature tensor varies based on the input.

Hyperparameters. We train all our models using AdamW optimizer [145] with a learning rate of 1e-4 and a weight decay of 1e-5 for 5000k steps. We reduce the learning rate by factor of 10 after 3500k steps. The hyperparameters in the loss functions λ_{L_1} and λ_{iou} are set to 5 and 3 respectively for all our experiments. All the learnable weights are initialized using Xavier initialization. We mention the dataset specific hyperparameters below.

ActivityNet-1.3. We use dropout with default probability 0.1. We use maximum number of nodes in the context graph N_v^{max} equal to 30. The size of the action query graph is 10 for our experiments (except when conducting ablation on the size of action query graph). We use base model dimension in the transformer as 512 and set the number of encoder and decoder layers as 4 (except when conducting ablation on the number of layers). We use an effective batch size of 64 for training models on this dataset.

THUMOS14. We do not use dropout for this dataset. We use maximum number of nodes in the context graph N_v^{max} equal to 256. The size of the action query graph is 300 for our experiments (except when conducting ablation on the size of action query graph). We use base model dimension in the transformer as 2048 and set the number of encoder and

Table 4.11: **Ablation Study: Loss function (Charades)**. We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report mAP to evaluate the performance of the model on Charades dataset. ↑ indicates higher is better.

Method	mAP (↑)
L_1 : ✓/ \mathcal{L}_{iou} : ✗	26.0
L_1 : ✗/ \mathcal{L}_{iou} : ✓	25.3
L_1 : ✓/ \mathcal{L}_{iou} : ✓	28.6

Table 4.12: **Ablation Study: Loss function (Epic-Kitchens100)** We train the model with a combination of cross-entropy loss and segment loss containing L_1 loss and/or IoU loss \mathcal{L}_{iou} . ✓ and ✗ indicate whether the specific component of the segment loss is used or not respectively. We report the mean average precision at different intersection over union thresholds (mAP@tIoU). ↑ indicates higher is better.

Task	Model	mAP@tIoU ↑					Average
		0.1	0.2	0.3	0.4	0.5	
Verb	L_1 : ✓/ \mathcal{L}_{iou} : ✗	11.5	9.2	8.0	6.6	5.5	7.1
	L_1 : ✗/ \mathcal{L}_{iou} : ✓	11.2	9.0	7.8	6.2	5.4	6.4
	L_1 : ✓/ \mathcal{L}_{iou} : ✓	12.0	10.3	8.2	7.1	6.1	8.7
Noun	L_1 : ✓/ \mathcal{L}_{iou} : ✗	10.8	8.9	6.8	6.3	3.5	6.3
	L_1 : ✗/ \mathcal{L}_{iou} : ✓	10.7	8.8	6.6	6.1	3.4	5.0
	L_1 : ✓/ \mathcal{L}_{iou} : ✓	11.6	9.3	7.1	6.6	3.9	7.7
Action	L_1 : ✓/ \mathcal{L}_{iou} : ✗	7.4	6.2	4.1	3.7	3.2	4.8
	L_1 : ✗/ \mathcal{L}_{iou} : ✓	7.3	6.1	3.7	3.3	3.0	3.7
	L_1 : ✓/ \mathcal{L}_{iou} : ✓	7.8	6.9	5.5	4.2	3.9	5.9

decoder layers as 4 (except when conducting ablation on the number of layers). We use an effective batch size of 32 for training models on this dataset.

Epic-Kitchens100. We do not use dropout for this dataset. We use maximum number of nodes in the context graph N_v^{max} equal to 1024. The size of the action query graph is 1200 for our experiments (except when conducting ablation on the size of action query graph). We use base model dimension in the transformer as 512 and set the number of encoder and decoder layers as 4 (except when conducting ablation on the number of layers). We use an effective batch size of 16 for training models on this dataset.

Charades. We use dropout with default probability 0.1. We use maximum number of nodes in the context graph N_v^{max} equal to 64. The size of the action query graph is 100 for our experiments (except when conducting ablation on the size of action query graph). We use base model dimension in the transformer as 512 and set the number of encoder and decoder layers as 4 (except when conducting ablation on the number of layers). We use an effective batch size of 32 for training models on this dataset.

Table 4.13: **Impact of number of layers.** We report performance of our AGT model with different number of layers in encoder and decoder. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #E indicates number of layers in encoder and #D indicates number of layers in decoder.

Dataset	#E	#D	mAP
ActivityNet-1.3	4	2	34.9
	4	4	35.7
	2	4	35.0
THUMOS14	4	2	64.0
	4	4	65.1
	2	4	64.4
EPIC (A)	4	2	5.5
	4	4	5.9
	2	4	5.6
EPIC (V)	4	2	8.5
	4	4	8.7
	2	4	8.6
EPIC (N)	4	2	7.2
	4	4	7.7
	2	4	7.3
Charades	4	2	28.0
	4	4	28.6
	2	4	28.2

4.4.2 Quantitative Analysis

In this section, we discuss our experiments and provide quantitative analysis.

Comparison with state-of-the-art. We compare the performance of our proposed AGT with the state-of-the-art methods. We use mean average precision as the metric to evaluate the model. To ensure fair comparison, we use the same evaluation protocol as used by state-of-the-art methods for each of the datasets. The results in Table 4.1 (ActivityNet-1.3), Table 4.2 (THUMOS14), Table 4.3 (Epic-Kitchens100), and Table 4.4 (Charades) show that our proposed AGT performs competitively with state-of-the-art methods for all the benchmark datasets.

Impact of graph based reasoning. To demonstrate the importance of reasoning over videos as graphs, we conducted ablation studies by removing the graph based reasoning components from either the encoder or the decoder or both (*i.e.* overall transformer network) in our model. Specifically, this is implemented by removing the graph message passing layers from the attention modules (*i.e.*, graph self-attention module and graph-to-graph attention

Table 4.14: **Impact of number of heads.** We report performance of our AGT model with different number of heads in the attention modules of the transformer network. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #heads indicates number of heads in attention modules of the transformer.

Dataset	#heads	mAP
ActivityNet-1.3	8	35.7
	4	35.1
THUMOS14	8	65.1
	4	63.4
EPIC (A)	8	5.9
	4	5.2
EPIC (V)	8	8.7
	4	8.4
EPIC (N)	8	7.7
	4	7.1
Charades	8	28.6
	4	26.4

module) in the encoder and/or decoder blocks in the network. Intuitively, when the graph message passing module is removed from the whole transformer network, the transformer encoder treats the input as a sequence and the transformer decoder treats the action queries as independent. The results shows the performance of the ablated versions of our model by reporting mAP at specific IoU thresholds for ActivityNet-1.3 in Table 4.5, THUMOS14 in Table 4.6, Epic-Kitchens100 in Table 4.8 as well as the average mAP for Charades dataset in Table 4.7.

The results clearly indicate that removing the graph-based reasoning module hurts the localization performance. The results also suggest that graph-based modeling is more useful in the encoder than in the decoder. We believe this is because the graph reasoning performed by the encoder is more effective in capturing the dependencies as it operates directly on the video features. In non-graph versions, the encoder-to-decoder attention, which is essential to mapping input space (videos) to the output space (actions), is performed using conventional multi-head attention. In contrast, our AGT performs this mapping using graph-to-graph attention which enables a richer mapping between the input and the output - the influence of an input node on an output node depends not just on the input node, but also on the connections of that input node with other neighboring nodes in the input space. Thus, the structure of the context graph is also conveyed to the output resulting in more effective grounding of input video to the actions.

Table 4.15: **Impact of action query graph size.** We report performance of our AGT model with different number of nodes in the action query graph. We report mAP for evaluation performance (higher is better). EPIC (A), EPIC (V), EPIC (N) indicates task ‘Action’, ‘Verb’, ‘Noun’ classification on EPIC-Kitchens100. #queries indicates number of nodes in the action query graph.

Dataset	#queries	mAP
ActivityNet-1.3	3	34.0
	10	35.7
	50	35.8
THUMOS14	150	60.2
	300	65.1
	900	65.1
EPIC (A)	900	4.3
	1500	5.9
	2000	6.1
EPIC (V)	900	7.0
	1500	8.7
	2000	8.7
EPIC (N)	900	6.1
	1500	7.7
	2000	7.8
Charades	30	24.0
	50	28.6
	100	28.6

Furthermore, the graph message passing layer in encoder as well as decoder blocks explicitly models pairwise dependencies. In every encoder or decoder block of our AGT, this message passing layer works in conjunction with a multi-head self-attention layer that captures global dependencies across all the nodes. These two types of dependencies effectively model the relations among the entities in the input space (videos) and the output space (actions). In contrast, the non-graph versions only rely on global dependencies across all entities in input space and output space, thereby not explicitly modeling pairwise dependencies. Our results demonstrate that the explicitly modeling of pairwise dependencies in conjunction with global dependencies are effective at capturing dependencies in data.

Ablation Study (Loss function). Note that for any version of the loss function, the model requires cross entropy loss to be able to classify the action labels. The model also requires some form of regression loss to produce predictions pertaining to the start and end timestamps of an action instance. Our overall loss (Eq. 4.10) is a combination of cross-entropy loss and segment loss \mathcal{L}_s with two components: L_1 loss and IoU loss \mathcal{L}_{iou} . We present the results of the performance of our model when trained with ablated versions of the segment loss by reporting mAP at specific IoU thresholds for ActivityNet-1.3 in Table 4.9),

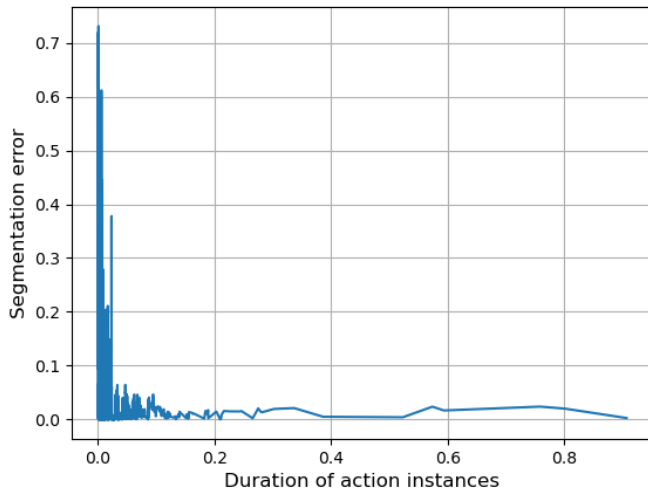


Figure 4.4: **Effect of Action Instance Durations (THUMOS14)**. Analysis of segmentation error (L1 loss) with respect to the duration of corresponding ground truth instances. All the values are normalized with respect to the overall video duration. We observe that the action instances of longer durations have lower segmentation errors in their predictions.

THUMOS14 in Table 4.10), and Epic-Kitchens100 in Table 4.12) and by reporting average mAP for Charades in Table 4.11. The results indicate that the models trained with only L_1 loss perform better than the ones trained with only IoU loss \mathcal{L}_{iou} . Additionally, models trained with both losses are better than the ones trained with only one of the losses. We only provide the mAP values averaged over the various intersection-over-union thresholds (tIoU).

Impact of number of layers Table 4.13 shows the results of the performance of our model with different number of layers in encoder and decoder component of the transformer. While increase in number of layers increases the training time, we did not observe much difference in the performance of the model with increased depth of the transformer components. We only provide the mAP values averaged over the various intersection-over-union thresholds (tIoU) for ActivityNet-1.3, THUMOS14 and Epic-Kitchens100.

Impact of number of heads. Table 4.14 shows the results of the performance of our AGT model with different number of heads in the attention modules of the transformer. The results suggest a slight improvement with more number of heads in the transformer network. We only provide the mAP values averaged over the various intersection-over-union thresholds (tIoU) for ActivityNet-1.3, THUMOS14 and Epic-Kitchens100.

Impact of action query graph size. Table 4.15 shows the results of the performance of our AGT model with different number of node encodings in the action query graph. Intuitively, a very large size of action query graph implies the model will require more time



Figure 4.5: **Visualization: Predictions (THUMOS14)**. Visualization of predictions and groundtruth action instances

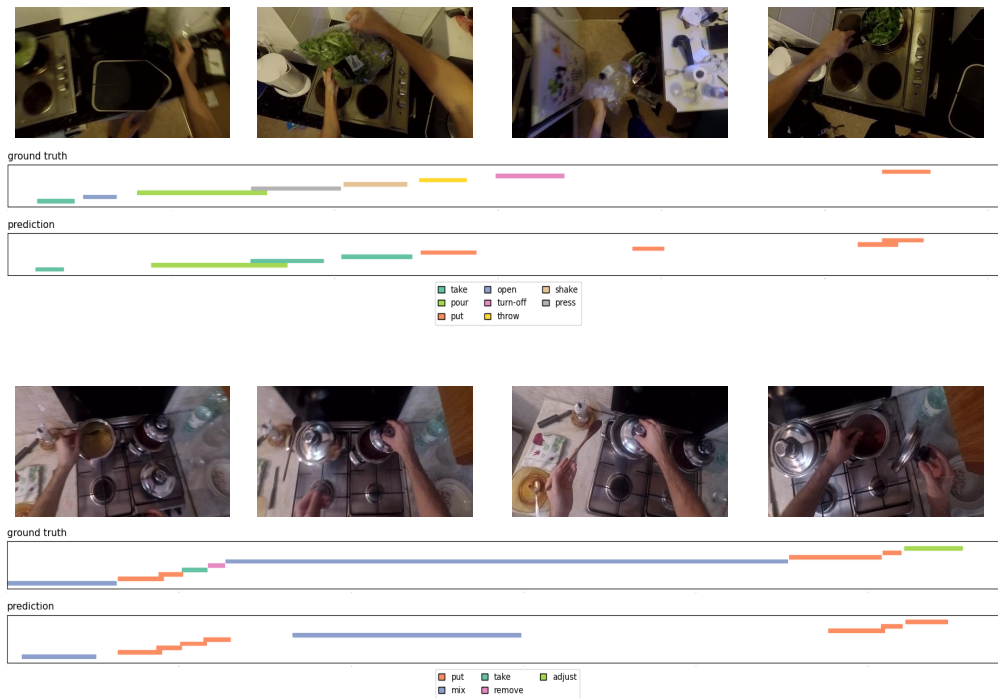


Figure 4.6: **Visualization: Predictions (Epic-Kitchens100)**. Visualization of predictions and groundtruth action instances

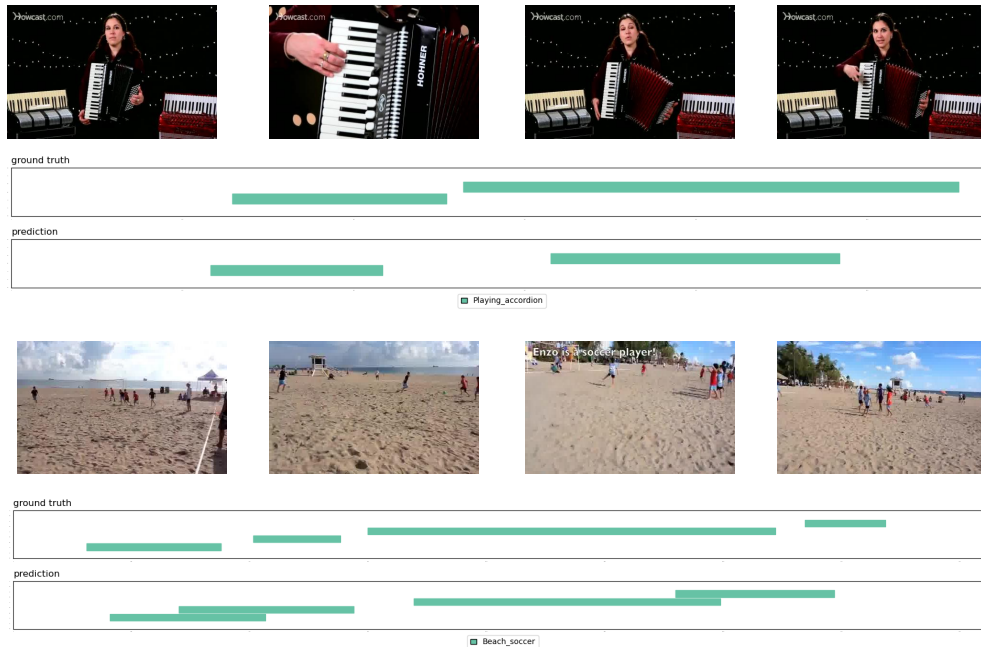


Figure 4.7: **Visualization: Predictions (ActivityNet-1.3).** Visualization of predictions and groundtruth action instances

to learn the non-maximal suppression of the irrelevant predictions. On the other hand, a very small size of action query graph might limit the ability of model to learn complex structure in the action instances. Note that, any value used for our experiments is higher than the maximum number of action instances per video in the dataset. The results suggest minor improvement with more number of nodes in the action query graph, however, the models with more number of nodes require longer training times. Our experiments also suggest that when the size of the action query graph is reduced, the localization performance of our model degrades. We only provide the mAP values averaged over the various intersection-over-union thresholds (tIoU) for ActivityNet-1.3, THUMOS14 and Epic-Kitchens100.

Effect of Action Instance Durations. We conduct further analysis to study the performance of our model in terms of the durations of the action instances. Figure 4.4 shows the trend of segmentation error, *i.e.*, L_1 norm computed between the ground truth and predicted timestamps of actions instances plotted against the duration of the ground truth instances (normalized with respect to the video duration). The error is computed over normalized values of the timestamps. This analysis indicates that action instances with larger durations (with respect to the whole video duration) have lower segmentation errors in their predictions as compared to the instances with smaller durations.

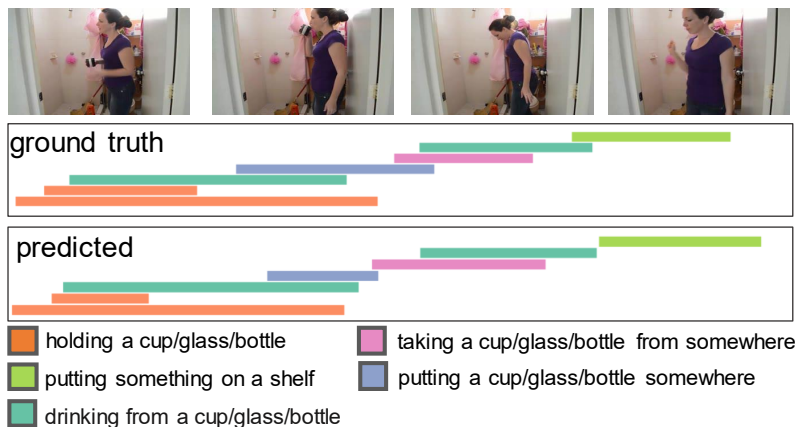


Figure 4.8: **Visualization: Predictions (Charades)**. Visualization of predicted and groundtruth action instances.

4.4.3 Qualitative Analysis

In addition to the quantitative evaluation of our model, we also visualize the predictions of our AGT for qualitative analysis.

Visualization (Predictions). We provide additional visualizations of the predictions of our AGT on several diverse samples in Figure 4.5, Figure 4.6, Figure 4.7, and Figure 4.8. The visualizations indicate that our model is able to predict the correct number of action instances as well as most of the correct action categories with minimal errors in start and end timestamps for videos containing overlapping instances with varying temporal extents. We observe the errors in timestamps predictions are near the boundaries. This is potentially because action is mostly concentrated away from the boundaries of some action instances and the video content around boundaries has no or very less role in the action depicted in those instances.

Visualization (Learned Graphs). We visualize the learned action query graph in Figure 4.9. by observing the graph embeddings obtained from the last layer of decoder. For better visibility, we do not plot the nodes (or their edges) that are classified as no action (*i.e.* class label \emptyset) by the prediction head. Note that the edge matrix is also learnable in our model. For the purpose of this visualization, we obtained the edge weights from the attention coefficients in the self-attention based graph message passing module . The visualizations demonstrate that the model indeed learns non-linear dependencies among the action instances that appear in the video.

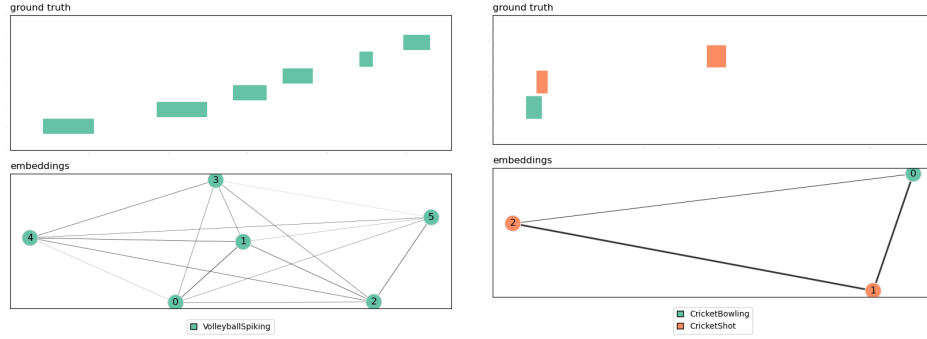


Figure 4.9: **Visualization: Learned Graphs.** Visualizations of embeddings corresponding to the last layer of the decoder and ground truth instances. The thickness of edges show the strength of interaction between the nodes. For ease of visibility, the nodes have been numbered based on the order of their predictions sorted with respect to the start time (*i.e.*, node 0 represents the instance that starts first). These visualizations demonstrate that the model indeed learns non-linear dependencies between the action instances in a video. The legend below each figure shows the action labels corresponding to the color coded elements.

4.5 Limitations

AGT is designed to directly predict a set of action instances. In our implementation, for each dataset, the size of the predicted set is treated as a hyperparameter and is typically higher than the maximum number of action instances per video in the dataset. While our results show that this approach has benefits for the existing benchmark datasets, this approach may not be as effective for datasets that are skewed in terms of number of instances per videos. For example, in case of a dataset with just a few video samples containing large number of actions and rest of the videos containing considerably low number of actions, this approach due might result in high number of false positives. Furthermore, predicting a fixed number of action instances might also restrict the capabilities of this model in online learning or active learning setting where the nature of the video data is not known beforehand.

4.6 Conclusion

In this chapter, we proposed a novel, streamlined, and end-to-end learnable graph transformer model for the task of temporal action localization in untrimmed human activity videos. Our approach relies on a key task-specific observation that videos depicting realistic activities contain non-linear temporal structure in both visual information and localized action instances. Based on this observation, our model adopts a graph-based compositional representation for the input video and output set of action instances and learns to detect and localize actions in videos using graph self-attention mechanisms. We evaluated

the effectiveness of our approach by presenting the experimental evaluation on challenging human activity datasets. Overall, this work highlights the importance of reasoning over activity videos (both input and output spaces) as non-sequential entities and shows that graph-based transformers are an effective technique to model complex activity videos.

Future work in temporal action localization can benefit from our idea of using graph-based compositional modeling for videos. We believe the notion of designing compositional representations for both the input and output spaces allows the model to capture richer information pertaining to the downstream task and can be relevant beyond temporal localization in tasks such as spatio-temporal localization and action prediction.

Chapter 5

Learning Segment-Level Representations for Action Anticipation

This chapter focuses on the task of action anticipation. Given an initial portion of a video, this task involves predicting future actions. Typically, the observed video is processed as a whole to obtain a video-level representation of the ongoing activity in the video, which is then used for future prediction. In this chapter, we introduce ANTICIPATR which performs long-term action anticipation leveraging segment-level representations learned using individual segments from different activities, in addition to a video-level representation. We present a two-stage learning approach to train a novel transformer-based model that uses these two types of representations to directly predict a set of future action instances over any given anticipation duration. Finally, this chapter discusses the findings of our experimental valuation on a diverse set of long-term action anticipation benchmarks.

5.1 Introduction

The ability to envision future events is a crucial component of human intelligence which helps in decision making during our interactions with the environment. We are naturally capable of anticipating future events when interacting with the environment in a wide variety of scenarios. Similarly, anticipation capabilities are essential to practical AI systems that operate in complex environments and interact with other agents or humans (*e.g.*, wearable devices [220], human-robot interaction systems [118], autonomous vehicles [149, 270]).

Existing anticipation methods have made considerable progress on the task of near-term action anticipation [35, 37, 57, 58, 63, 66, 152, 238] that involves predicting the immediate next action that would occur over the course of a few seconds. While near-term anticipation is a valuable step towards the goal of future prediction in AI systems, going beyond short time-horizon prediction has applicability in a broader range of tasks that involve long-term interactions with the environment. The ability to anticipate actions over long time-horizons

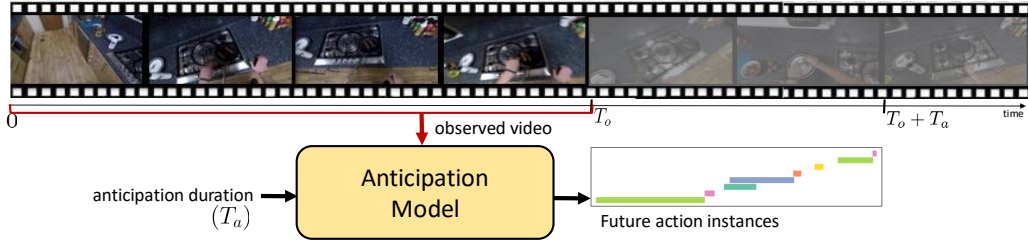


Figure 5.1: **Long-Term Action Anticipation.** Given the initial portion of an activity video ($0, \dots, T_o$) and anticipation duration T_a , the task is to predict the actions that would occur from time $T_o + 1$ to $T_o + T_a$. Our proposed anticipation model receives the observed video and the anticipation duration as inputs and directly predicts a set of future action instances. Here, the action anticipation is *long-term* – both the observed duration T_o and the anticipation duration T_a are in the order of minutes.

is imperative for applications such as efficient planning in robotic systems [23, 61] and intelligent augmented reality systems.

In this paper, we focus on long-term action anticipation. Figure 5.1 illustrates the problem – having observed an initial portion of an untrimmed activity video, we predict *what* actions would occur *when* in the future.

Long-term anticipation methods [3, 50, 61, 70, 111, 199] predict future actions based on the information in the observed video (*i.e.*, an initial portion of an untrimmed activity video) that partially depicts the activity in the video. Current approaches rely on encoding the observed video (input) as a whole to obtain *video-level representations* to perform action anticipation.

We propose a novel approach that leverages segment-level and video-level representations for the task of long-term action anticipation. Consider the example in Figure 5.1. The video depicts the activity *person making pasta* spanning several minutes. This activity has segments with actions such as *slice onion*, *put pesto*, *put courgette*, *add cheese*. One of these segments such as *put pesto* tends to co-occur with actions involving objects such as *courgette*, *onion*, or *cheese* in a specific order. However, other videos with a different activity, say, *person making pizza*, could potentially have a similar set and/or sequence of actions in a different kitchen scenario. As such, while a specific sequence of actions (*i.e.*, segments of a video) help denote an activity, an individual video segment (containing a single action) alone contains valuable information for predicting the future. Based on this intuition, we introduce an approach that leverages segment-level representations in conjunction with video-level representations for the task of long-term action anticipation. In so doing, our approach enables reasoning beyond the limited context of the input video sequence.

In this work, we propose ANTICIPATR that consists of a two-stage learning approach employed to train a transformer-based model for long-term anticipation (see Figure 5.2 for an overview). In the first stage, we train a *segment encoder* to learn segment-level

representations. As we focus on action anticipation, we design this training task based on co-occurrences of actions. Specifically, we train the segment encoder to learn *which future actions are likely to occur after a given segment?* Intuitively, consider a video segment showing a pizza pan being moved towards a microwave. Irrespective of the ongoing activity in the video that contains this segment, it is easy to anticipate that certain actions such as *open microwave*, *put pizza* and *close microwave* are more likely to follow than the actions *wash spoon* or *close tap*.

In the second stage, we utilize both the segment-level and video-level representations for long-term action anticipation. We design a transformer-based model that contains two encoders: (1) the segment encoder to derive representations corresponding to segments in the observed video, and (2) a *video encoder* to derive the video-level representations of the observed video. These encoded representations are then fed into an *anticipation decoder* that predicts actions that would occur in the future. Our model is designed to directly predict a set of future action instances, wherein, each element of the set (*i.e.*, an action instance) contains the start and end timestamps of the instance along with the action label. Using direct set prediction, our approach predicts the actions at all the timestamps over a given anticipation duration in a single forward pass.

To summarize, this paper makes the following contributions: (1) a novel learning approach for long-term action anticipation that leverages segment-level representations and video-level representations of the observed video, (2) a novel transformer-based model that receives a video and anticipation duration as inputs to predict future actions over the specified anticipation duration, (3) a direct set prediction formulation that enables single-pass prediction of actions, and (4) quantitative and qualitative evaluation on a diverse set of anticipation benchmarks: Breakfast [120], 50Salads [223], Epic-Kitchens-55 [35], and EGTEA Gaze+ [134].

Overall, our work highlights the benefits of learning representations that capture different aspects of a video, and particularly demonstrates the value of such representations for action anticipation.

5.2 Related Work

Action Anticipation. Action anticipation is generally described as the prediction of actions before they occur. Prior research efforts have used various formulations of this problem depending on three variables: (1) anticipation format, *i.e.*, representation format of predicted actions, (2) anticipation duration, *i.e.*, duration over which actions are anticipated, and (3) model architectures.

Current approaches span a wide variety of anticipation formats involving different representations of prediction outcomes. They range from pixel-level representations such as frames or segmentations [12, 137, 147, 155] and human trajectories [5, 37, 87, 98, 116, 153] to

label-level representations such as action labels [50, 57, 58, 63, 111, 124, 177, 188, 189, 199, 208, 238, 276, 282] or temporal occurrences of actions [3, 61, 136, 152, 157, 226] through to semantic representations such as affordances [118] and language descriptions of sub-activities [200]. We focus on label-level anticipation format and use ‘action anticipation’ to refer to this task hereafter.

Existing anticipation tasks can be grouped into two categories based on the anticipation duration: (1) near-term action anticipation, and (2) long-term action anticipation. In this paper, we focus on long-term action anticipation.

Near-term anticipation involves predicting label for the immediate next action that would occur in the range of a few seconds having observed a short video segment of duration of a few seconds. Prior work propose a variety of temporal modeling techniques to encode the observed segment such as regression networks [238], reinforced encoder-decoder network [63], TCNs [274], temporal segment network [35], LSTMs [57, 58, 172], VAEs [157, 253] and transformers [66].

Long-term anticipation involves predicting action labels over long time-horizons in the range of several minutes having observed an initial portion of a video (observed duration of a few minutes). A popular formulation of this task involves prediction of a sequence of action labels having observed an initial portion of the video. Prior approaches encode the observed video as a whole to obtain a video-level representation. Using these representations, these approaches either predict actions recursively over individual future time instants or use time as a conditional parameter to predict action label for the given single time instant. The recursive methods [3, 50, 61, 177, 199] accumulate prediction error over time resulting in inaccurate anticipation outcomes for scenarios with long anticipation duration. The time-conditioned method [111] employs skip-connections based temporal models and aims to avoid error accumulation by directly predicting an action label for a specified future time instant in a single forward pass. However, this approach still requires multiple forward passes during inference as the task involves predicting actions at all future time instants over a given anticipation duration. Additionally, sparse skip connections used in [111] do not fully utilize the relations among the actions at intermediate future time instants while predicting action at a given future time instant. A recent transformer-based method [70] attempts to address this by predicting future actions using an encoder-decoder approach wherein, the encoder is used to a video-level representation and the decoder predicts the output sequence of future action labels in a single forward pass. In contrast to these approaches based on video-level representations, our approach leverages segment-level representations (learned using individual segments across different activities) in conjunction with video-level representations. Both these representations are utilized to directly predict action instances corresponding to actions at all the time instants over a given anticipation duration in a single forward pass.

An alternate formulation of long-term anticipation proposed in [162] focuses on predicting a set of future action labels without inferring when they would occur. [162] extracts a graph representation of the video based on frame-level visual affordances and uses graph convolutional network to encode the graph representation to predict a set of action labels. In contrast, our approach leverages both the segment-level and video-level representations of the input video and a transformer-based model to predict action instances - both action labels and their corresponding timestamps.

Other methods design approaches to model uncertainty in predicting actions over long time horizons [2, 168, 177] and self-supervised learning [181].

Early action detection. The task of early action detection [90, 148, 193, 210] involves recognizing an ongoing action in a video as early as possible given an initial portion of the video. Though the early action detection task is different from action anticipation (anticipation involves prediction of actions *before* they begin), the two tasks share the inspiration of future prediction.

Transformers in computer vision. The transformer architecture [234], originally proposed for machine translation task, has achieved state-of-the-art performance for many NLP tasks. In recent years, there has been a flurry of work on transformer architectures designed for high-level reasoning tasks on images and videos. Examples include object detection [21], image classification [46], spatio-temporal localization in videos [65], video instance segmentation [251], action recognition [8, 285], action detection [143, 164, 182, 206, 279], multi-object tracking [158], next action anticipation [66], human-object interaction detection [112, 296]. DETR [21] is a transformer model for object detection, wherein, the task is formulated as a set prediction problem. This work has since inspired transformer designs for similar vision tasks – video instance segmentation [251] and human-object interaction detection [296]. Inspired by these works, we propose a novel transformer architecture that uses two encoder to encode different representations derived from the input video and a decoder to predict the set of future action instances in a single pass. Our proposed decoder also receives anticipation duration as an input parameter to control the duration over which actions are predicted.

5.3 Action Anticipation with ANTICIPATR

In this section, we first describe our formulation of long-term action anticipation and then describe our approach.

Problem Formulation. Let \mathbf{v}_o be an observed video containing T_o frames. Our goal is to predict the actions that occur from time $T_o + 1$ to $T_o + T_a$ where T_a is the anticipation duration, *i.e.*, the duration over which actions are predicted. Specifically, we predict a set $\mathcal{A} = \{a^i = (c^i, t_s^i, t_e^i)\}$ containing future action instances. The i -th element denotes an action

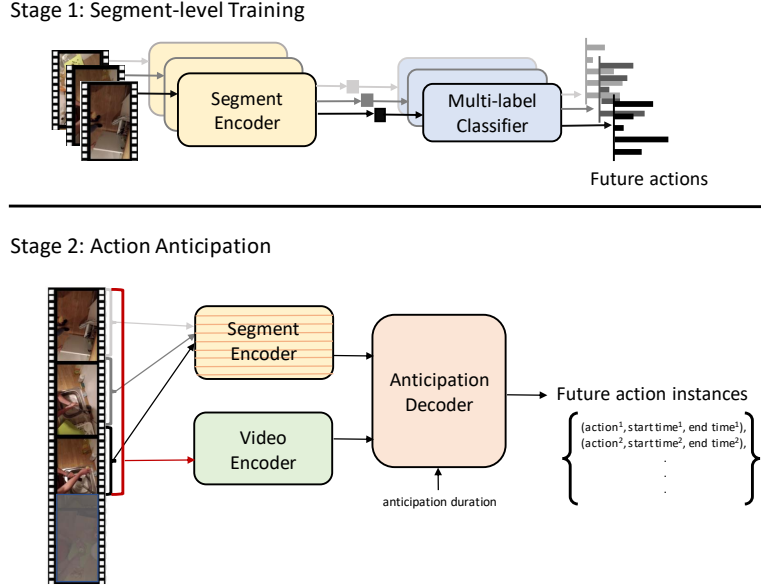


Figure 5.2: **Learning Approach.** ANTICIPATR uses a two-stage learning approach. In the first stage, we perform segment-level training (refer to Sec 5.3.1). Given a segment as input, we train a segment encoder to predict the set of action labels that would occur at any time after the occurrence of the segment in the activity video. In the second stage, we perform long-term action anticipation (refer to Sec 5.3.2). We use video encoder to obtain video-level representation and segment encoder (trained in the first stage) is used to obtain segment-level representation. The anticipation decoder receives these two representations of the observed video to directly predict a set of action instances that would occur in the future over a given anticipation duration.

instance a^i depicting action category c^i occurring from time t_s^i to t_e^i where $T_o < t_s^i < t_e^i \leq T_o + T_a$. Here, $c^i \in \mathcal{C}$ where \mathcal{C} is the set of action classes in the dataset.

Intuitively, for action anticipation, the observed video as a whole helps provide a broad, video-level representation of the ongoing activity depicted in the video. However, the observed video is composed of several segments that individually also contain valuable information about future actions and provide an opportunity to capture the video with segment-level representations. Using this intuition, in this paper, we propose ANTICIPATR that leverages these two types of representations of the observed video for the task of long-term anticipation.

ANTICIPATR employs a two-stage learning approach to train a transformer-based model that takes an observed video as input and produces a set of future action instances as output. See Figure 5.2 for an overview. Also, refer to Figure 5.4 for the detailed architecture of each of the components. In the first stage, we train a *segment encoder* that receives a segment (sequence of frames from a video) as input and predicts the set of action labels that would occur at any time in the future after the occurrence of the segment in the video. We refer to this stage as segment-level training (described in Sec. 5.3.1). As the segment encoder

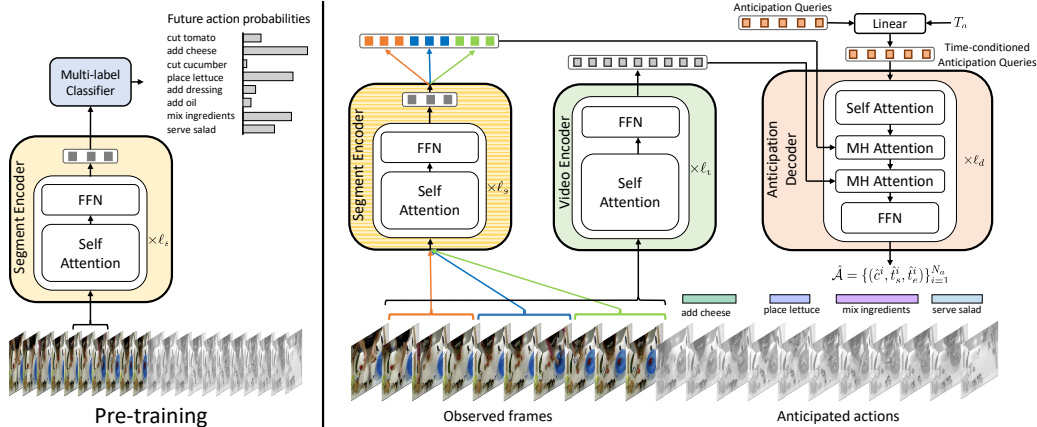


Figure 5.3: **Model Architecture.** Our model comprises three networks: *segment encoder*, *video encoder* and *anticipation decoder* and is trained for long-term action anticipation in two stages. (*left*) Segment-level training (Sec. 5.3.1): The segment encoder receives a segment as input and predicts a set of action labels that would occur at any time in the future (after the occurrence of segment in the video). (*right*) Action Anticipation (Sec. 5.3.2): The video encoder encodes the observed video to a video-level representation. Concurrently, the video is divided into a sequence of segments and each segment is fed into the segment encoder (trained in first stage)The anticipation decoder receives the two representations along with an anticipation duration as inputs to directly predict a set of future action instances over the given anticipation duration. [MH Attention: Multi-head Attention, FFN: Feed Forward Network.]

only operates over individual segments, it is unaware of the broader context of the activity induced by a specific sequence of segments in the observed video.

In the second stage, we train a *video encoder* and an *anticipation decoder* to be used along with the segment encoder for long-term action anticipation. The video encoder encodes the observed video to a video-level representation. The segment encoder (trained in the first stage) is fed with a sequence of segments from the observed video as input to obtain a segment-level representation of the video. The anticipation decoder receives the two representations along with the anticipation duration to predict a set of future action instances over the given anticipation duration in a single pass. The video encoder and anticipation decoder are trained using classification losses on the action labels and two temporal losses (L_1 loss and temporal IoU loss) on the timestamps while the segment encoder is kept unchanged. We refer to this second stage of training as action anticipation (see Sec. 5.3.2).

5.3.1 Stage 1: Segment-level Training

In this stage, the segment encoder is trained on a segment-level prediction task to learn representations for individual segments. See Figure 5.3 (*left*) for an overview and Figure 5.4 for the detailed architecture of each of the components of the model.

Segment Encoder. We design the segment encoder network E_s as a sequence of ℓ_s transformer blocks containing a multi-head self-attention module followed by layernorm and a feed forward network [234]. This network is trained on the task of segment-level action anticipation.

Training. During training, the segment encoder receives a segment (sequence of frames from a video) as input and predicts the set of action labels that would occur at any time in the future (starting from the temporal boundary, *i.e.*, end of the segment until the end of that video) without inferring when they would occur. Depending on the segment, there could be multiple actions occurring between the end of segment and end of video. Thus, we formulate this training task as a multi-class multi-label classification.

The training data for the segment encoder is derived from the training set in the original video dataset containing videos with action annotations. These input segments are obtained using the action boundaries provided in the training set. We do not require any additional annotations. Formally, given a video \mathbf{v} containing T frames, a segment $\mathbf{v}_s^{(t', t'')}$, spanning time indices t' to t'' where $0 \leq t' < t'' < T$, is taken as input. For this segment, the target is a binary vector \mathbf{c}_s (dimension $|\mathcal{C}|$) corresponding to the action labels that occur after the temporal boundary of the segment until the end of the video ($[\mathbf{v}^{t''+1}, \dots, \mathbf{v}^T]$).

The segment encoder E_s receives the segment $\mathbf{v}_s^{(t', t'')}$ along with positional encodings $\mathbf{p}_s^{(t', t'')}$. These positional encoders are intended to provide temporal information in the segment using the sinusoidal positional encodings (c.f. Vaswani *et al.* [234]) based on timestamps corresponding to the features of input segment. Specifically, for each input feature of each embedding we independently use sine and cosine functions with different frequencies. We then concatenate them along the channel dimension to get the final positional encoding. In our implementation, the embedding size is same as that of the segment feature so that they can be combined by simple addition of the positional encodings and segment features.

The output of the encoder is an embedding $\mathbf{h} = [\mathbf{h}^1, \dots, \mathbf{h}^{t''-t'+1}]$ of dimension $(t'' - t' + 1) \times d_s$ where d_s is the channel dimension. The output embeddings are then averaged along time dimension and fed into a linear layer F followed by a sigmoid activation σ to obtain future action probabilities $\hat{\mathbf{c}}_s$ of dimension $|\mathcal{C}|$, expressed as:

$$\begin{aligned} \mathbf{h} &= E_s(\mathbf{v}_s^{(t', t'')}, \mathbf{p}_s^{(t', t'')}) \\ \hat{\mathbf{c}}_s &= \sigma \left(F \left(\frac{1}{t'' - t' + 1} \sum_{i=1}^{t''-t'+1} \mathbf{h}^i \right) \right). \end{aligned} \quad (5.1)$$

Here, $\hat{\mathbf{c}}_s$ is the output of a multi-label classifier where each element c_s^j of $\hat{\mathbf{c}}_s$ denotes probability of corresponding action category $j \in \mathcal{C}$. This network is trained using binary cross entropy loss between the prediction vector $\hat{\mathbf{c}}_s$ and target vector \mathbf{c}_s . Once trained, the linear layer F is discarded and the segment encoder E_s is used to obtain segment-level representations for the action anticipation stage.

5.3.2 Stage 2: Action Anticipation

In the second stage of our approach, we use an encoder-decoder model that contains two encoders: (i) the segment encoder from the first stage, and (ii) a video encoder that encodes the observed video as a whole. The outputs of these two encoders along with an anticipation duration are fed into an anticipation decoder which uses the representations from the two encoders to predict a set of future action instances over the given anticipation duration. See Figure 5.3 (*right*).

Video Encoder. The video encoder receives an observed video containing T_o frames. We denote the input as $\mathbf{v}_o = [\mathbf{v}^1, \dots, \mathbf{v}^{T_o}]$. We design the encoder network E_v as a sequence of ℓ_v transformer blocks [234] containing a multi-head self-attention module followed by layernorm and feed forward network. The encoder receives the features corresponding to the observed video \mathbf{v}_o as input. As the self-attention module is permutation-invariant, we provide additional information about the sequence in the form of sinusoidal positional encodings [234] $\mathbf{p}_o = [\mathbf{p}^1, \dots, \mathbf{p}^{T_o}]$. Specifically, for each input feature of each embedding we independently use sine and cosine functions with different frequencies. We then concatenate them along the channel dimension to get the final positional encoding. In our implementation, the embedding size is same as that of the video feature.

Here, each element in the positional encoding sequence is added to the corresponding element in the video features and then fed into the encoder block. The encoder models temporal relationships in the observed video and transforms the input sequence to a contextual representation $\mathbf{h}_v = [\mathbf{h}_v^1, \dots, \mathbf{h}_v^{T_o}]$, expressed as:

$$\mathbf{h}_v = E_v(\mathbf{v}_o, \mathbf{p}_o). \quad (5.2)$$

Encoding Video Segments. Concurrent to the video encoder, the input video is divided into a sequence of segments using temporal sliding windows. Specifically, a temporal window of size k starting from frame index i obtains a segment $[\mathbf{v}^i, \dots, \mathbf{v}^{i+k-1}]$, which is fed to the segment encoder to obtain the outputs $\mathbf{h}_s^i, \dots, \mathbf{h}_s^{i+k-1}$. The starting index i slides across time with $i \in \{1, k+1, 2k+1, \dots, (T_o-k+1)\}$ generating the temporal windows, where the window size k is a hyperparameter. The outputs of the segment encoder for all temporal windows are concatenated to obtain $\mathbf{h}_s = [\mathbf{h}_s^1, \dots, \mathbf{h}_s^{T_o}]$. During implementation, the representations can still be obtained in one forward pass of the segment encoder by stacking segments along the batch dimension of the input. This segment-level representation of the video is complementary to the video-level representation that encodes the ongoing activity in the video.

Anticipation Decoder. Given the video-level and the segment-level representations, the decoder aims to predict a set of future action instances over a given anticipation duration. The predicted set contains action instances of the form (label, start time, end time). The

anticipation decoder receives the following inputs: (i) *anticipation queries* \mathbf{q}_0 , (ii) anticipation duration T_a over which actions are to be predicted, (iii) encoded representation \mathbf{h}_v from video encoder E_v , and (iv) encoded representation \mathbf{h}_s from segment encoder E_s .

The anticipation queries contain N_a elements, *i.e.*, $\mathbf{q}_0 = [\mathbf{q}_0^1, \dots, \mathbf{q}_0^{N_a}]$, wherein each query is a learnable positional encoding. The positional encoding layer is designed as learnable embedding layer that receives integer index i as input corresponding to i -th anticipation query and provides an embedding \mathbf{q}_0^i where $i \in \{1, \dots, N_a\}$. In our implementation, we use `torch.nn.Embedding` in Pytorch to implement this. The weights of the layer are learnable during training, thus, the positional encoding layer is also learnable. The initialization of this layer requires maximum possible value of the index, *i.e.*, N_a in our case. We consider N_a as a hyperparameter that is constant for a dataset and is sufficiently larger than the maximum number of action instances to be anticipated per video in the overall dataset. Each query \mathbf{q}_0^i is then fed into a linear layer (weights shared for all values of i) along with the anticipation duration T_a to obtain time-conditioned anticipation queries \mathbf{q}_a^i for $i = 1, \dots, N_a$. This time conditioning enables the anticipation decoder to predict actions over any specified anticipation duration.

The decoder network D consists of ℓ_d blocks, wherein, each block contains a cascade of attention layers. The first attention layer is the multi-head self-attention block which models relations among the anticipation queries. The second attention layer is a multi-head encoder-decoder attention layer that maps the queries and the segment-level representations from the segment encoder. And, the third attention layer is another multi-head encoder-decoder attention layer that maps the output of previous layer to the video-level representation corresponding to the input. This third attention layer is followed by a feedforward network. The output of the decoder $\mathbf{y} = [\mathbf{y}^1, \dots, \mathbf{y}^{N_a}]$ serves as a latent representation of the action instances in the videos, expressed as:

$$\mathbf{y} = D(\mathbf{q}_a, \mathbf{h}_v, \mathbf{h}_s) \quad (5.3)$$

The decoder output is used to predict the set of action instances $\hat{\mathcal{A}} = \{\hat{a}^i = (\hat{c}^i, \hat{t}_s^i, \hat{t}_e^i)\}_{i=1}^{N_a}$. Each element in decoder output \mathbf{y}^i is fed into a linear layer followed by softmax to obtain prediction probabilities $\hat{p}^i(c)$ where $c = 1, \dots, |\mathcal{C}| + 1$ and \hat{c}^i is the class corresponding to maximum probability. The number of queries N_a is larger than the maximum number of action instances per video in the dataset. Thus, we introduce an additional class label \emptyset indicating no action. \mathbf{y}^i is also fed into another feedforward network with ReLU to obtain corresponding start timestamps \hat{t}_s^i and end timestamps \hat{t}_e^i .

Training. To compute the loss, we first align the predictions with the groundtruth set of action instances. This alignment is necessary as there is no fixed prior correspondence between the predicted and the groundtruth set of action instances. Here, the predicted set for any video contains N_a action instances, but the size of groundtruth set \mathcal{A} varies based

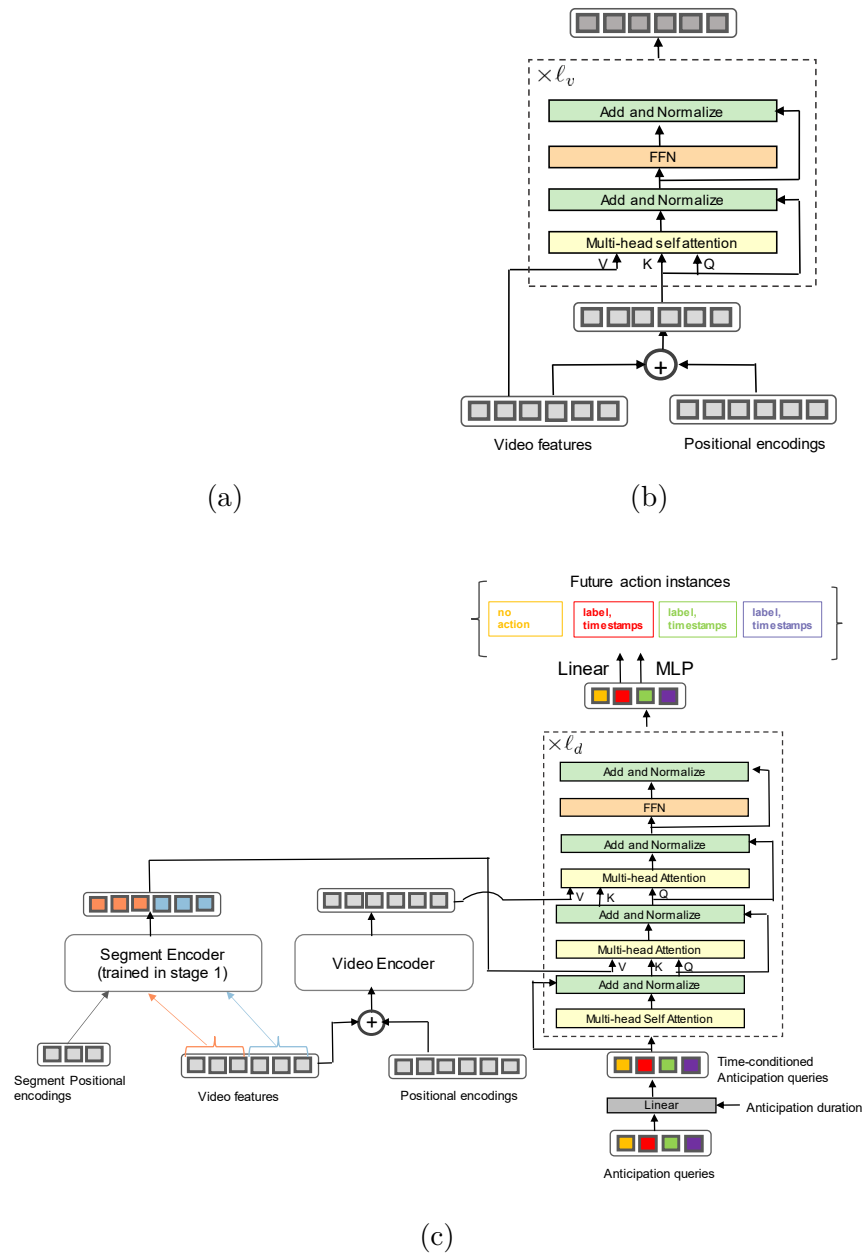


Figure 5.4: **Detailed Architecture.** Architecture overview of (a) Segment encoder, (b) Video encoder, and (c) Anticipation Decoder. ‘Q’, ‘K’, ‘V’ are query, key and value to the self-attention layer as described in [234].

on the video and is smaller than the predicted set. Thus, we first pad the groundtruth set to make it the same size as the predicted set by adding $N_a - |\mathcal{A}|$ elements with label \emptyset indicating no action. Then, we use a pair-wise greedy correspondence algorithm to align the groundtruth and predicted sets. Intuitively, the objective is to correctly align actions at as many future time instants as possible. Therefore, starting with the groundtruth instance having the longest duration, we match each groundtruth instance with the unmatched predicted instance that has the maximum temporal overlap with the groundtruth instance. To start the iterative process, we first sort the action instances groundtruth set based on the descending order of the duration of the instances. We then begin the alignment process with the groundtruth instance having the maximum duration. We lookup the predicted set to find the predicted instance that has maximum temporal overlap with this groundtruth instance. Since the predicted set is designed to represent a single action instance, the alignment between groundtruth and predicted set is one-to-one. Thus, to continue the alignment process, the matched groundtruth instance and predicted instance are removed from the corresponding sets. In this way, this process is repeated until the groundtruth set is empty. As the predicted set is of size larger than groundtruth set, the remaining predicted instances are mapped to \emptyset denoting no action. This results in a one-to-one mapping for loss computation.

Consider the output of the set correspondence module as γ denoting the permutation of the predicted set of instances, *i.e.*, the groundtruth action instance a^i is matched to predicted instance $\hat{a}^{\gamma(i)}$ for $i = 1, \dots, N_a$. Given this alignment, we compute loss \mathcal{L} over all the matched pairs as a weighted combination of cross-entropy loss for classification, and two temporal losses: L_1 loss and IoU loss (\mathcal{L}_{iou}) for prediction of segment timestamps, defined as:

$$\mathcal{L} = \sum_{i=1}^{N_a} \left[-\log(\hat{p}^{\gamma(i)}(c^i)) + \mathbb{1}_{\{c^i \neq \emptyset\}} \lambda_{L_1} \|s^i - \hat{s}^{\gamma(i)}\|_1 + \mathbb{1}_{\{c^i \neq \emptyset\}} \lambda_{iou} \mathcal{L}_{iou}(s^i, \hat{s}^{\gamma(i)}) \right], \quad (5.4)$$

where $\lambda_{iou}, \lambda_{L_1} \in \mathbb{R}^+$ are hyperparameters, $s^i = [t_s^i, t_e^i]$, $\hat{s}^{\gamma(i)} = [\hat{t}_s^{\gamma(i)}, \hat{t}_e^{\gamma(i)}]$ and $\hat{p}^{\gamma(i)}(c^i)$ is the probability of the groundtruth class c^i for prediction $\gamma(i)$. The L_1 temporal loss is sensitive to the absolute value of the duration of the segments. The IoU loss \mathcal{L}_{iou} is invariant to the duration of the segments. Thus, these two losses together are designed to incorporate different aspects of segment prediction. \mathcal{L}_{iou} is described as

$$\mathcal{L}_{iou}(s^i, \hat{s}^{\gamma(i)}) = 1 - \frac{|s^i \cap \hat{s}^{\gamma(i)}|}{|s^i \cup \hat{s}^{\gamma(i)}|}, \quad (5.5)$$

where $|\cdot|$ is the duration of the instance, *i.e.*, difference between end and start timestamp.

The video encoder and anticipation decoder are jointly trained to minimize the loss \mathcal{L} described in Eq. 5.4. We do not fine-tune the segment encoder in this stage.

Inference. During inference, the video encoder takes the observed video as input and the segment encoder takes the chunked video (*i.e.*, non-overlapping segments of fixed length) as input. The inputs to the decoder are: (i) anticipation queries $\mathbf{q}_0 = 1, \dots, N_a$ (a constant, regardless of input), (ii) anticipation duration T_a (varies based on the input video and the anticipation requirement), (iii) output representation from the video encoder, and (iv) output representation from the segment encoder. The decoder predicts a set of action instances. Thus, our approach allows us to build a model that can anticipate actions over any future duration in a single pass by simply controlling the input T_a to the decoder as shown by results in Table 5.1.

In summary, ANTICIPATR uses a two-stage learning approach to train a transformer-based model (consisting of two encoders and one decoder) to predict a set of future action instances over any given anticipation duration. Our approach aims to perform action anticipation with segment-level representations learned using individual video segments in conjunction with video-level representations learned by encoding input video as a whole. Our model anticipates actions at all time instants over a given anticipation duration in a single forward pass by directly predicting a set of future action instances.

5.4 Experiments

We conducted extensive experiments and analysis to demonstrate the effectiveness of our proposed approach.

5.4.1 Experimental Setup

Datasets. We evaluate on four established benchmarks for this task. These datasets of untrimmed videos vary in scale, diversity of labels and video duration.

- **Breakfast** [120] contains 1,712 videos each depicting one of 10 breakfast activities and annotated with action instances spanning 48 different action classes. On average, a video contains 6 action instances and has a duration of 2.3 minutes. For evaluation, we report the average across 4 splits from the original dataset.
- **50Salads** [223] contains 50 videos, each showing a person preparing a salad. On average, there are 20 action instances per video spanning 17 action classes and duration is 6.4 minutes. Following the original dataset, we report the average across 5-fold cross-validation in our evaluation.
- **EGTEA Gaze+ (EGTEA+)** [134] contains egocentric videos of 32 subjects following 7 recipes in a single kitchen. Each video depicts the preparation of a single dish.

Table 5.1: **Results (Breakfast and 50Salads)**. We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance. Note that ‘‘Sener *et al.* [199] (features+labels)’’ use action labels from a segmentation algorithm as additional input. Baseline results are from respective papers.

Observation (β_o) \rightarrow		20%				30%			
Anticipation (β_a) \rightarrow		10%	20%	30%	50%	10%	20%	30%	50%
Breakfast	RNN [3]	18.1	17.2	15.9	15.8	21.6	20.0	19.7	19.2
	CNN [3]	17.9	16.3	15.3	14.5	22.4	20.12	19.7	18.7
	RNN [3] + TCN	5.9	5.6	5.5	5.1	8.9	8.9	7.6	7.7
	CNN [3] + TCN	9.8	9.2	9.1	8.9	17.6	17.1	16.1	14.4
	Ke <i>et al.</i> [111]	18.4	17.2	16.4	15.8	22.7	20.4	19.6	19.7
	Farha <i>et al.</i> [50]	25.9	23.4	22.4	21.5	29.7	27.4	25.6	25.2
	Qi <i>et al.</i> [181]	25.6	21.0	18.5	16.0	27.3	23.6	20.8	17.3
	Sener <i>et al.</i> [199] (features)	24.2	21.1	20.0	18.1	30.4	26.3	23.8	21.2
	Sener <i>et al.</i> [199] (features+labels)	37.4	31.8	30.1	27.1	39.8	34.2	31.9	27.9
	FUTR [70]	27.7	24.5	22.8	22.0	32.3	29.9	27.5	25.9
	ANTICIPATR (Ours)	37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4
50Salads	RNN [3]	30.1	25.4	18.7	13.5	30.8	17.2	14.8	9.8
	CNN [3]	21.2	19.0	15.9	9.8	29.1	20.1	17.5	10.9
	RNN [3] + TCN	32.3	25.5	19.1	14.1	26.1	17.7	16.3	12.9
	CNN [3] + TCN	16.0	14.7	12.1	9.9	19.2	14.7	13.2	11.2
	Ke <i>et al.</i> [111]	32.5	27.6	21.3	15.9	35.1	27.1	22.1	15.6
	Farha <i>et al.</i> [50]	34.8	28.4	21.8	15.2	34.4	23.7	18.9	15.9
	Sener <i>et al.</i> [199](features)	25.5	19.9	18.2	15.1	30.6	22.5	19.1	11.2
	Sener <i>et al.</i> [199](features+labels)	34.7	26.3	23.7	15.7	34.5	26.1	22.7	17.1
	Qi <i>et al.</i> [181]	37.9	28.8	21.3	11.1	37.5	24.1	17.1	09.1
	Piergiovanni <i>et al.</i> [177]	40.4	33.7	25.4	20.9	40.7	40.1	26.4	19.2
	FUTR [70]	39.5	27.5	23.3	17.8	35.2	24.9	24.2	15.2
ANTICIPATR (Ours)	41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6	

Each video is annotated with instances depicting interactions (*e.g.*, open drawer), spanning 53 objects and 19 actions.

- **EPIC-Kitchens-55 (EK-55)** [35] contains videos of daily kitchen activities. It is annotated for interactions spanning 352 objects and 125 actions. It is larger than the aforementioned datasets, and contains unscripted activities.

Training Details. We describe the details of the setup used for training ANTICIPATR for each of the datasets. We train all our models using AdamW [145] optimizer on 4 Nvidia V100 32GB GPUs. We initialize all the learnable weights using Xavier initialization.

For training of first stage, we use dropout probability of 0.1. For the segment encoder, we use base model dimension as 2048 and set the number of encoder layers as 3 with 8 attention heads. We use an effective batch size of 64 for training segment encoder on this dataset. For training in the second stage, we use base model dimension in the video encoder and anticipation decoder as 2048 and set the number of encoder and decoder layers as 3 with 8 heads. We provide dataset-specific hyperparameters as follows.

Table 5.2: **Results (EK-55 and EGTEA+)**. We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance. Baseline results are from respective papers.

Method	EK-55			EGTEA+		
	ALL	FREQ	RARE	ALL	FREQ	RARE
RNN	32.6	52.3	23.3	70.4	76.6	54.3
I3D [22]	32.7	53.3	23.0	72.1	79.3	53.3
ActionVLAD [67]	29.8	53.5	18.6	73.3	79.0	58.6
Timeception [92]	35.6	55.9	26.1	74.1	79.7	59.7
VideoGraph [93]	22.5	49.4	14.0	67.7	77.1	47.2
EGO-TOPO [162]	38.0	56.9	29.2	73.5	80.7	54.7
ANTICIPATR(Ours)	39.1	58.1	29.1	76.8	83.3	55.1

Breakfast. We represent input videos as I3D features. We choose N_a (anticipation queries) to be 150. We use an effective batch size of 16 for training the video encoder and anticipation decoder on this dataset on the long-term anticipation task. We train our models with a learning rate of $1e-4$ and a weight decay of 0. The model is trained for 4000k steps. We use a dropout probability of 0.1. We set $\lambda_{L1} = 3$ and $\lambda_{iou} = 5$. To obtain segment-level representation of the observed video during action anticipation, we use a temporal window of length $k = 16$.

50Salads. We represent input videos as Fisher vectors. We choose N_a (anticipation queries) to be 80. We use an effective batch size of 16 for training the video encoder and anticipation decoder on this dataset on the long-term anticipation task. We use a learning rate of $1e-5$ and a weight decay of $1e-5$. We train the model for 3000k steps and reduce the learning rate by factor of 10 after 1500k steps. We don't use dropout for this dataset. We set $\lambda_{L1} = 3$ and $\lambda_{iou} = 5$. To obtain segment-level representation of the observed video during action anticipation, we use a temporal window of length $k = 48$.

EPIC-Kitchens-55. We represent input videos as I3D features. We use an effective batch size of 16 for training the video encoder and anticipation decoder in the second stage. We choose N_a (anticipation queries) to be 900. We use a learning rate of $1e-4$ and a weight decay of $1e-5$. We train the model for 6000k steps and reduce the learning rate by factor of 10 after 4000k steps. We use a dropout probability of 0.1. We set $\lambda_{L1} = 5$ and $\lambda_{iou} = 8$. To obtain segment-level representation of the observed video during action anticipation, we use a temporal window of length $k = 32$.

EGTEA Gaze+. We represent input videos as I3D features. We use an effective batch size of 16 for training the video encoder and anticipation decoder in the second stage. We choose N_a to be 600. We use a learning rate of $1e-5$ and a weight decay of $1e-5$. We train the model for 4000k steps and reduce the learning rate by factor of 10 after 3000k steps. We use a dropout probability of 0.1. We set $\lambda_{L1} = 3$ and $\lambda_{iou} = 5$. To obtain segment-level

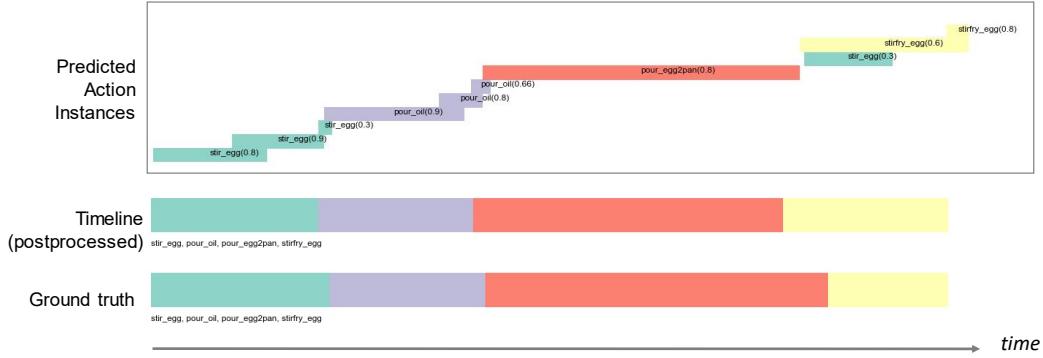


Figure 5.5: **Evaluation.** Visualization showing the outputs from our ANTICIPATR and the corresponding timeline obtained after postprocessing to obtain a sequence of action labels for evaluation.

representation of the observed video during action anticipation, we use a temporal window of length $k = 24$.

5.4.2 Evaluation.

To measure the performance of our model, we adopt the evaluation protocol followed by state-of-the-art methods for these benchmark datasets.

For Breakfast and 50Salads, we report the mean over classes accuracy averaged over all future timestamps in the specified anticipation duration, *i.e.*, dense prediction evaluation as defined in [3, 50, 111]. We use $\beta_o\%$ of a full video as observation duration and predict the actions corresponding to following $\beta_a\%$ of the remaining video. As per the benchmarks, we sweep the values of $\beta_o \in \{20, 30\}$ and $\beta_a \in \{10, 20, 30, 50\}$ denoting different observation and anticipation durations respectively. Note that a single trained model is used for predicting at all these values of β_o and β_a by just varying the anticipation duration input to the decoder. Since the metric is computed over a dense anticipation timeline, we first convert our model predictions (set of action instances) into a timeline and then compute mean over classes accuracy. We refer to the timeline as a sequence of action labels for time instants in the anticipation duration, *i.e.*, between $T_o + 1, \dots, T_o + T_a$ with a single action class assigned to each time instant. Thus, we iterate over the predicted set to assign class labels to this timeline. Specifically, for each action instance in the predicted set, we assign the predicted action class to the time instants that are within the predicted segment (determined by predicted start and end timestamp). When predicted action instances overlap at certain time instants, we assign the action class with highest probability score among the overlapping predictions. Once the timeline is constructed, we compute mean over classes accuracy [199] to evaluate the model performance. We are constructing this timeline only during evaluation to follow the benchmark evaluation protocols. See Figure 5.5 for an example.

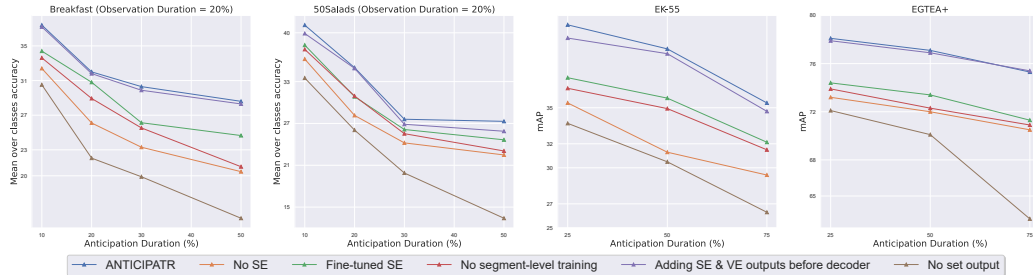


Figure 5.6: **Analysis.** Quantitative evaluation of the anticipation performance of ablated versions of ANTICIPATR. [SE: segment encoder; VE: video encoder].

For EK-55 and EGTEA+, we compute a multi-label classification metric (mAP) over the target action classes as defined in [162]. $\alpha_o\%$ of each untrimmed video is given as input to predict all action classes in the future $(100 - \alpha_o)\%$ of the video, *i.e.*, until the end of the video. We sweep values of $\alpha_o \in \{25, 50, 75\}$ representing different observation durations. Since the metric is computed only over the future action classes, we take the union of the class labels of predicted action instances to compute mAP. Specifically, we perform a union over all the action classes (except \emptyset class) in the predicted set of instances to obtain a set of future action classes. We use this set of action classes to compute mAP as described in benchmark [162].

5.4.3 Quantitative Analysis

In this section, we report the findings of the quantitative evaluation of ANTICIPATR.

Comparison with state-of-the-art. Table 5.1 shows the results for Breakfast and 50Salads datasets in the ‘*no groundtruth labels*’ setting [111, 199]. The results show that our approach outperforms existing methods by a considerable margin for different observation/anticipation durations. For these benchmarks, the most similar approach to ours is Sener *et al.* [199] where they propose self-attention methods for temporal aggregation for long-term video modeling. In the setting similar to ours where they use only visual features as input, our approach outperforms [199] with up to 13% improvement. Moreover, when they also use action labels from a segmentation algorithm as input, our approach is still competitive despite not using such additional inputs. In addition, the benefit of our approach is more apparent when the anticipation duration is longer.

Table 5.2 shows results on the long-term action anticipation benchmarks for EK-55 and EGTEA+ datasets, as defined by [162]. The results show that our model achieves competitive results with the state-of-the-art method [162]. While this benchmark only considers prediction of future action labels, our results demonstrate that the segment prediction in our model acts as a beneficial auxiliary task for label prediction.

Table 5.3: **Ablation: Loss function (Breakfast and 50Salads)**. We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance. ✓ and ✗ indicate whether the component of the temporal loss is used or not respectively.

	Method	$\beta_o \rightarrow$ $\beta_a \rightarrow$	20%				30%			
			10%	20%	30%	50%	10%	20%	30%	50%
			Breakfast	$L_1: \text{✗}; \mathcal{L}_{iou}: \text{✓}$	36.2	30.7	28.6	26.4	38.7	33.9
	$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✗}$	36.5	31.1	29.1	28.2	39.2	34.2	31.7	28.1	
	$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✓}$	37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4	
50Salads	$L_1: \text{✗}; \mathcal{L}_{iou}: \text{✓}$	40.2	33.9	26.8	26.0	41.9	41.4	27.6	23.3	
	$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✗}$	40.8	34.5	27.1	26.8	42.1	41.6	27.9	23.4	
	$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✓}$	41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6	

Table 5.4: **Ablation: Loss function (EK-55 and EGTEA+)**. We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance. ✓ and ✗ indicate whether the component of the temporal loss is used or not respectively.

Method	EK-55			EGTEA+		
	ALL	FREQ	RARE	ALL	FREQ	RARE
$L_1: \text{✗}; \mathcal{L}_{iou}: \text{✓}$	34.9	56.4	27.3	75.2	82.1	53.8
$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✗}$	37.7	57.8	28.4	76.0	82.7	54.6
$L_1: \text{✓}; \mathcal{L}_{iou}: \text{✓}$	39.1	58.1	29.1	76.8	83.3	55.1

Impact of Segment-level Training. Our two-stage learning approach separately learns video-level representations and segment-level representations. To analyze the impact of such two-stage training, we design following experiments.

(i) **Fine-tuned Segment Encoder.** In this experiment, we also fine-tune the segment encoder while training video encoder and decoder during the anticipation stage (Sec 5.3.2). The results in Figure 5.6 (‘Fine-tuned SE’) indicate that fine-tuning the segment encoder hurts the anticipation performance. We believe fine-tuning the segment encoder with anticipation loss (Eq. 5.4) perturbs the segment-level representation learned during first stage of training.

(ii) **No Segment-level Training.** In this experiment, we do not train the segment encoder network in a separate stage. Instead, we train all three networks (*i.e.*, segment encoder, video encoder and anticipation decoder) jointly for the task of long-term action anticipation using the anticipation loss function (Eq. 5.4). Here, the segment encoder receives videos chunked into short segments (same as the proposed two-stage training). However, it is directly tasked with solving a more difficult problem of simultaneously encoding segment-level representation and inferring its usage for long-term anticipation. The results for all datasets presented in Figure 5.6 (‘No Segment-level Training’) illustrate that eliminating

Table 5.5: **Ablation: Anticipation Queries (Breakfast and 50Salads)**. We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance.

Method	$\beta_o \rightarrow$ $\beta_a \rightarrow$	20%				30%			
		10%	20%	30%	50%	10%	20%	30%	50%
Breakfast	$N_a = 50$	32.6	28.2	26.4	24.3	35.8	31.4	28.7	25.3
	$N_a = 150$	37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4
	$N_a = 500$	36.6	31.5	29.4	27.3	38.5	34.4	31.3	28.3
50Salads	$N_a = 20$	38.4	33.2	24.2	23.6	39.1	35.6	25.5	24.2
	$N_a = 80$	41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6
	$N_a = 320$	40.5	34.2	26.0	25.6	41.3	40.9	27.4	23.3

Table 5.6: **Ablation: Anticipation Queries (EK-55 and EGTEA+)**. We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance.

Dataset		ALL	FREQ	RARE
EK-55	$N_a = 300$	34.3	55.6	24.2
	$N_a = 900$	39.1	58.1	29.1
	$N_a = 2700$	38.2	56.9	28.3
EGTEA+	$N_a = 200$	70.2	79.5	49.7
	$N_a = 600$	76.8	83.3	55.1
	$N_a = 1800$	75.3	82.4	53.3

training of the segment encoder worsens the anticipation performance. This shows the value of learning the segment-level representations independently without being influenced by the overall activity in the input video.

In summary, these experiments demonstrate the importance of the two-stage learning approach and suggest that the two representations should be learned separately to serve their individual purposes during anticipation.

Impact of Segment Encoder. To evaluate the impact of learning segment-level representation, we conducted experiments without the segment encoder network. This ablated version only contains the video encoder and the anticipation decoder and is trained in a single-stage using the anticipation loss (Eq. 5.4). The results in Figure 5.6 (‘No SE’) show that removing the segment-level representations considerably hurts the anticipation performance. This performance degradation is worse than just removing the segment-level training stage (‘No segment-level training’ in Figure 5.6). Thus, this experiment validates the benefit of the segment-level stream of information for action anticipation.

Impact of Set-based Output Representation. In our approach, we model the anticipation output as a set of action instances. We empirically validate this design by comparing

Table 5.7: **Ablation: Segment window length (Breakfast and 50Salads)**. We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance.

	Method	$\beta_o \rightarrow$	20%				30%			
			$\beta_a \rightarrow$	10%	20%	30%	50%	10%	20%	30%
Breakfast		$k = 4$	35.9	30.6	26.3	26.1	38.4	33.6	30.8	28.2
		$k = 16$	37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4
		$k = 64$	37.4	31.7	29.9	28.1	39.1	35.0	31.7	28.7
50Salads		$k = 12$	39.0	33.5	25.8	25.4	39.6	38.4	26.4	21.5
		$k = 48$	41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6
		$k = 192$	41.0	34.8	27.2	26.8	42.6	42.1	27.5	22.8

Table 5.8: **Ablation: Segment window length (EK-55 and EGTEA+)**. We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance.

Dataset		ALL	FREQ	RARE
EK-55	$k = 8$	37.9	57.2	27.4
	$k = 32$	39.1	58.1	29.1
	$k = 128$	38.8	58.0	28.7
EGTEA+	$k = 6$	75.4	81.7	53.9
	$k = 24$	76.8	83.3	55.1
	$k = 96$	76.3	82.9	54.8

with an alternative approach where the output is a sequence of action labels corresponding to the individual future time instants. We implement this by changing the anticipation queries (decoder input) during the anticipation stage – we provide positional encodings corresponding to each time instant over anticipation duration and directly predict the labels corresponding to these time instants. While the prediction for all time instants still happens in a single pass, the decoder is required to transform a large number of anticipation queries. The results in Figure 5.6 (‘No Set Output’) show poor performance that worsen further as anticipation duration increases. This is largely because the number of queries is too high for the decoder for effective modeling.

Fusion of Encoder Outputs. To combine the representation from segment encoder and video encoder, our model uses two encoder-decoder attention layers in the decoder blocks. We tested an alternative approach wherein we fused the representations using a simple addition along temporal dimension before feeding into the decoder. Here, we modify the decoder blocks to contain a single encoder-decoder attention layer. The results in Figure 5.6 (‘Adding SE & VE before decoder’) indicate that this fusion approach leads to a slight decrease in anticipation performance. We believe adding the representations before decoder forces the

Table 5.9: **Ablation: Sliding windows for Segment Encoder Training.** Mean over classes accuracy for different observation/anticipation durations. Higher is better. [BF: Breakfast; 50SL: 50Salads]

Observation (β_o) \rightarrow		20%				30%			
Anticipation (β_a) \rightarrow		10%	20%	30%	50%	10%	20%	30%	50%
Breakfast	Sliding windows	35.9	30.7	28.0	26.4	37.8	33.5	29.9	25.2
	ANTICIPATR(Full)	37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4
50Salads	Sliding windows	37.2	33.5	26.3	25.8	37.9	37.0	26.1	24.5
	ANTICIPATR(Full)	41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6

Table 5.10: **Ablation: Sliding windows for Segment Encoder Training.** mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Higher is better.

Method	EK-55			EGTEA+		
	ALL	FREQ	RARE	ALL	FREQ	RARE
Sliding Windows	37.6	56.5	27.4	74.8	81.2	53.0
No Video Encoder	30.9	51.8	21.2	70.2	79.9	50.1
ANTICIPATR(Full)	39.1	58.1	29.1	76.8	83.3	55.1

computation of encoder-decoder attention weights by considering both information streams at once. In contrast, our ANTICIPATR approach of computing attention one-by-one enables it to first filter out the relevant information from segment-level representations learned across different activities and then contextualize them into the specific context of the input video.

Ablation: Loss function. The training loss function defined in Eq. (4) in the main paper contains three components (cross-entropy loss and two temporal losses). We conduct ablation experiments by removing one of the temporal losses. Note that we always need cross entropy loss for the classification task. Results in Table 5.3 and Table 5.4 show that models trained with overall loss perform better than the ones trained with the ablated versions. Moreover, the models trained with only L_1 temporal loss perform better than the ones trained with only \mathcal{L}_{iou} .

Ablation: Anticipation queries. The number of anticipation queries discerns the maximum number of action instances the model is supposed to predict. Results in Table 5.5 and Table 5.6 shows the performance of our model with different number of anticipation queries. The results suggest minor improvement with higher number of anticipation queries, however, the models with more number of queries require longer training times. Intuitively, a very large number of anticipation queries implies the model will require more time to learn the non-maximal suppression of the irrelevant predictions. On the other hand, when the number of anticipation queries is reduced, the anticipation performance of our model degrades. A very small number of anticipation queries implies less number of action are

Table 5.11: **Ablation: Set correspondence (Breakfast & 50Salads)**. We report the mean over classes accuracy for different observation/anticipation durations. Higher values indicate better performance.

	Method	$\beta_o \rightarrow$	20%				30%			
			$\beta_a \rightarrow$	10%	20%	30%	50%	10%	20%	30%
Breakfast	Hungarian		36.8	32.0	30.5	28.4	39.2	35.4	31.9	29.6
	Greedy		37.4	32.0	30.3	28.6	39.9	35.7	32.1	29.4
50Salads	Hungarian		41.3	35.1	27.4	26.8	42.9	42.0	28.4	23.8
	Greedy		41.1	35.0	27.6	27.3	42.8	42.3	28.5	23.6

Table 5.12: **Ablation: Set correspondence (EK-55 & EGTEA+)**. We report mAP values for ALL classes, FREQUENT classes (> 100 action instances) and RARE class (< 10 action instances). Following [162], we report the mAP values averaged over different observation durations. Higher values implies better performance.

Method	EK-55			EGTEA+		
	ALL	FREQ	RARE	ALL	FREQ	RARE
Hungarian	39.0	58.4	28.4	76.7	83.5	55.0
Greedy	39.1	58.1	29.1	76.8	83.3	55.1

anticipated. Thus, for very complex video with many future action instances, the model would miss several action instances resulting in poor anticipation performance. Additionally, as shown in Table 5.5, the anticipation error increases over time. This is because there are more actions to be anticipated and the model is limited by the number of anticipation queries.

Ablation: Segment window length. Results in Table 5.7 and Table 5.8 shows the performance of our model with different values of temporal window lengths used to extract segment-level representations during action anticipation. The results suggest that neither a very small window length nor a very large window is helpful. The segment encoder is trained to predict future actions given a video segment depicting a single action. During the action anticipation stage, when the segment encoder is used to extract segment-level representations, the observed video is divided into a series of non-overlapping segment using temporal sliding windows as the action boundaries are not known. Intuitively, when the temporal sliding window is very small, the individual segments do not have enough information to obtain effective representations. On the other hand, when the window is very large, the segments contain more than one action and potentially results in segment-level representations with overlapping semantic content. We observe that the drop in performance with models that use smaller window lengths is larger as compared to the ones with larger window lengths.

Ablation: Sliding Windows for Segment Encoder Training. Instead of using action boundaries we used sliding temporal windows of length= k (same as used during stage 2) to



Figure 5.8: **Visualizations (50Salads)**. Examples from 50Salads dataset for the case where observation duration is 20% of the video duration and anticipation duration involves predicting actions for 50% of the remaining video.

obtain segments for segment-level training. Results in Table 5.9 and Table 5.10 show that this approach results in a slightly lower performance than our proposed training approach. This is possibly due to increased noise in the segment-level representations from this training approach.

Ablation: Set correspondence. To compute the anticipation loss, we use a greedy algorithm to align groundtruth and predicted set of action instances. Another commonly employed set correspondence algorithm is Hungarian matcher algorithm used in prior works [21, 112]. For completeness, we also conducted experiments with Hungarian matcher optimized

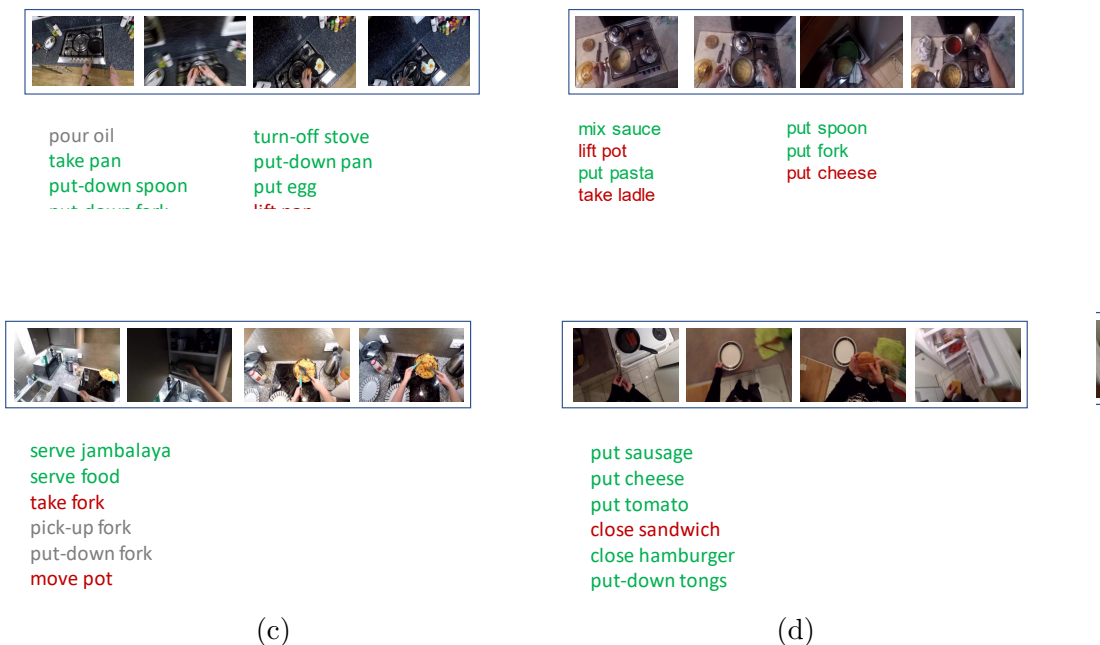


Figure 5.9: **Visualizations (EK-55)**. Examples from Epic-Kitchens-55 dataset for the case where observation duration is 50% of the video duration. We show the predicted action classes in the visualization – classes in green color are correct predictions, classes in red color are wrong predictions, and classes in gray color are missed classes.

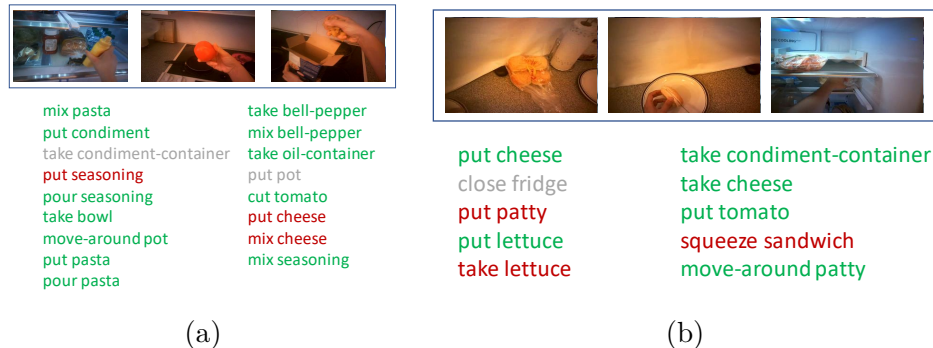


Figure 5.10: **Visualizations (EGTEA+)**. Examples from EGTEA Gaze+ dataset for the case where 50% of the video is observed. We show the predicted action classes in the visualization – classes in green color are correct predictions, classes in red color are wrong predictions, and classes in gray color are missed classes.

over the cost function with all three terms (classification loss and two temporal losses) following [164]. We didn't observe any significant difference in performance of the models trained using either of the two matchers as shown in Table 5.11 and Table 5.12.

5.4.4 Qualitative Analysis

Visualizations in Figure 5.7 and Figure 5.8 show that our model is generally able to anticipate correct actions at future time instants long anticipation durations for Breakfast and 50Salads benchmarks respectively. Moreover, examples in Figure 5.9 and Figure 5.10 show that our model is able to effectively predict future action classes for EK-55 and EGTEA benchmarks respectively.

Failure Cases. We observe that the action boundaries in some cases are not exactly aligned with the groundtruth even though the class labels are predicted accurately (See Figure 5.7 and Figure 5.8). We believe this could be because the visual information pertaining to the information is limited or negligible towards the beginning and end of the action instance.

Most classification errors result from the model getting confused among semantically similar classes. Some such cases from our examples are *'take ladle'* and *'pick-up ladle'* in Figure 5.9(b)); *'close sandwich'* and *'close hamburger'* in Figure 5.9(d)); *'put seasoning'* and *'pour seasoning'* in Figure 5.10(a)). Moreover, our model sometimes misses rare actions during predictions such as *'pour oil'* in Fig 5.9(a) and *'close fridge'* in Figure 5.10(b).

Additionally, we also observe that having seen certain objects in the observed video, the model predicts objects that are likely to co-occur with the seen objects. See the scenario in Figure 5.8(d). The model does not predict *'cut_cheese'* and *'place_cheese_into_bowl'* after the action *'place_cucumber_into_bowl'*. Instead, the model predicts *'cut_tomato'* and *'place_tomato_into_bowl'*. While the prediction is not correct for this specific activity, it is still a reasonable sequence of actions as there are several other salad recipe videos in the dataset that only use *cucumber* and *tomato*. In another scenario in Figure 5.10(b), having seen *'pasta'* in the observed video, the model anticipates action classes with *'cheese'* noun. While *'cheese'* does not appear in this particular video, it is a reasonable prediction since the nouns *'pasta'* and *'cheese'* often appear together in activity videos in this dataset.

5.5 Limitations

In this chapter, we demonstrate the effectiveness of our model on minutes-long activity videos. Handling longer videos with durations in hours or days (common in surveillance or monitoring scenarios) would be interesting future work. Furthermore, our approach assumes that the videos have an overall context provided by the ongoing long-term activity. We show that modeling interactions among segments (and, in turn, segment-level representation) is an effective technique for such activity videos as the video segments are indeed related. However, such approaches may not be directly applicable to videos that are just a montage of several unrelated content such as videos containing clips from different movies. Nonetheless, the core idea of learning representations that captures generic information using relevant clips from videos is relevant for videos with clips of unrelated content and might inspire

future work to design novel approaches to compositionally such videos using visual and semantic change points.

5.6 Conclusion

In this chapter, we introduced a novel approach for long-term action anticipation to leverage segment-level representations learned from individual segments across different activities in conjunction with a video-level representation that encodes the observed video as a whole. We proposed a novel two-stage learning approach to train a transformer-based model that receives a video and an anticipation duration as inputs and predicts a set of future action instances over the given anticipation duration. Results showed that our approach achieves state-of-the-art performance on long-term action anticipation benchmarks for Breakfast, 50Salads, Epic-Kitchens-55, and EGTEA Gaze+ datasets. Overall, our work highlights the benefits of learning representations that capture information across different activities for action anticipation.

Future work on long-term action anticipation can benefit from our idea of learning compositional representations to leverage generic information present at different granularities of human activity videos.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this dissertation, we focused on developing compositional models for activity understanding. The central idea proposed in this dissertation is to design compositional representations for human activity videos that are specific to the downstream task and are learned using different types of compositional information available at various granularities of the videos. We applied this idea to a diverse set of video tasks aimed at understanding realistic activities, namely, generation of human-object interaction videos, temporal action localization, and action anticipation. In doing so, we highlighted how compositionality effectively contributes to the following aspects of learning algorithms.

- **Generalizable models.** By learning smaller structures during training, compositional models enable reasoning over compositions of learned substructures that are unseen during training and therefore, can generalize well during inference. We focused on this aspect of compositional modeling in Chapter 3 by exploring how compositional models unlock generalization capabilities in generative models for human-object interaction videos in complex scenes. Specifically, we introduced a task of generating human-object interaction videos in a zero-shot compositional setting, *i.e.*, generating videos for action-object compositions that are unseen during training. We designed a novel adversarial framework to generate human-object interaction videos that employs multiple discriminators to focus on different aspects of a video.
- **Structured learning.** Compositional models allow structured modeling of data, thereby, providing a richer understanding of how different elements in the input data interact. We investigated this in Chapter 4 for the task of temporal action localization in human activity videos. Specifically, we proposed a method that encodes the non-linear temporal structure in activity videos and the non-sequential nature of the output space pertaining to the downstream task (*i.e.* set of action instances). Our pro-

posed method learns to detect and localize actions in videos by reasoning over the videos and the action instances as non-sequential entities in the form of graphs.

- **Generic representations.** Compositional learning equips models to reason beyond the limited context of the input – they allow models to capture valuable generic information at element-level that is common across the inputs in a given dataset. Chapter 5 probed into this observation in the context of long-term action anticipation. Specifically, while a specific sequence of actions (*i.e.*, segments of a video) in the given portion of the activity video provides temporal context of the ongoing activity, an individual video segment (containing a single action) alone contains generic cross-activity information that is valuable for predicting the future. Therefore, we introduced ANTICIPATR which performs long-term action anticipation leveraging segment-level representations learned using individual segments from different activities, in addition to a video-level representation. We presented a two-stage learning approach to train a novel transformer-based model that uses these two types of representations to directly predict a set of future action instances over any given anticipation duration.

6.2 Future Directions

Encouraged by the findings in this dissertation, I am interested in further exploring the broader direction of compositional models for activity understanding. My future plans will be focused on the following perspectives.

- **Compositional models for multiple tasks.** While there is value in learning task-specific models in constrained settings, it is limiting for practical systems aimed at activity understanding to employ separate models for every single downstream task. The area of activity understanding has a plethora of tasks that are related. For instance, the tasks action detection, spatio-temporal localization, temporal segmentation, and video object segmentation are all aimed at spatially and/or temporally localizing entities. This implies we could potentially leverage compositional information at different granularities of activity videos pertaining to different tasks and learn a single model that can handle several related tasks. Furthermore, such a learning paradigm is also promising in providing insights into how different tasks interact and could contribute to the research aimed at learning large-scale models aimed at learning general-purpose representations for videos.
- **Self-supervised learning of compositional representations.** Research on self-supervised representation learning for multimodal video understanding [59, 201, 202, 225] offers interesting insights into benefits of using self-supervised algorithms leveraging different streams of information in videos. Such approaches consider videos as compositions of multiple modalities of information. With the recent availability of

large-scale activity video datasets [36, 75], there is a research opportunity to design compositional representations for activities. These datasets contain annotations at different granularities and various types of derived information from videos (*e.g.*, gaze, 3D maps). Thus, in the context of holistic activity understanding, there are potentially interesting questions that emerge along this research direction: Can we design self-supervised algorithms that effectively incorporates different types of compositional information in activity videos? Can these techniques learn foundation models for holistic video understanding? Can such foundation models generalize to other domains?

Bibliography

- [1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.
- [2] Yazan Abu Farha and Juergen Gall. Uncertainty-aware anticipation of activities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2019.
- [3] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what?-anticipating temporal occurrences of activities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2018.
- [4] Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [5] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [6] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
- [7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.
- [8] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [9] Yueran Bai, Yingying Wang, Yunhai Tong, Yang Yang, Qiyue Liu, and Junhui Liu. Boundary content graph neural network for temporal action proposal generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [10] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005.

- [11] Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-gan: Unsupervised video retargeting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [12] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Bayesian prediction of future street scenes using synthetic likelihoods. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.
- [13] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2005.
- [14] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.
- [15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems (NeurIPS)*, 2020.
- [16] Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2017.
- [17] Fabian Caba Heilbron, Wayner Barrios, Victor Escorcia, and Bernard Ghanem. Sec: Semantic context cascade for efficient action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [18] Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2016.
- [19] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [20] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [21] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [22] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

- [23] Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. Procedure planning in instructional videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [24] Yu-Wei Chao, Yunfan Liu, Xieyang Liu, Huayi Zeng, and Jia Deng. Learning to detect human-object interactions. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.
- [25] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [26] Yu-Wei Chao, Zhan Wang, Yugeng He, Jiaxuan Wang, and Jia Deng. Hico: A benchmark for recognizing human-object interactions in images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.
- [27] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, 2011.
- [28] Ming Chen, Yingming Li, Zhongfei Zhang, and Siyu Huang. Tvt: Two-view transformer network for video captioning. In *Proceedings of the Asian Conference on Machine Learning (ACCV)*, 2018.
- [29] Shizhe Chen, Yida Zhao, Qin Jin, and Qi Wu. Fine-grained video-text retrieval with hierarchical graph reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [30] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Multi-fiber networks for video recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [31] Wongun Choi, Khuram Shahid, and Silvio Savarese. What are they doing?: Collective activity classification using spatio-temporal relationship among people. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshop on Visual Surveillance*, 2009.
- [32] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [33] Xiyang Dai, Bharat Singh, Guyue Zhang, Larry S Davis, and Yan Qiu Chen. Temporal context network for activity localization in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [34] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

- [35] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [36] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021.
- [37] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Msr-gen: Multi-scale residual graph convolution networks for human motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [38] Achal Dave, Olga Russakovsky, and Deva Ramanan. Predictive-corrective networks for action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [39] Vincent Delaitre, David F Fouhey, Ivan Laptev, Josef Sivic, Abhinav Gupta, and Alexei A Efros. Scene semantics from long-term observation of people. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [40] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.
- [41] Emily L Denton et al. Unsupervised learning of disentangled representations from video. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.
- [42] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [43] Chaitanya Desai and Deva Ramanan. Detecting actions, poses, and objects with relational phraselets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [44] Ali Diba, Mohsen Fayyaz, Vivek Sharma, Manohar Paluri, Jürgen Gall, Rainer Stiefelhagen, and Luc Van Gool. Large scale holistic video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [45] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [46] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words:

- Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.
- [47] Mounira Ebdelli, Olivier Le Meur, and Christine Guillemot. Video inpainting with short-term windows: application to object removal and error concealment. In *IEEE Transactions on Image Processing*, 2015.
- [48] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.
- [49] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
- [50] Yazan Abu Farha, Qihong Ke, Bernt Schiele, and Juergen Gall. Long-term anticipation of activities with cycle consistency. In *Proceedings of the German Conference on Pattern Recognition (GCPR)*, 2020.
- [51] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [52] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [53] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.
- [54] Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial structures for object recognition. *International journal of computer vision (IJCV)*, 2005.
- [55] Jerry A Fodor and Ernest Lepore. The compositionality papers. *Oxford University Press*, 2002.
- [56] David F Fouhey, Vincent Delaitre, Abhinav Gupta, Alexei A Efros, Ivan Laptev, and Josef Sivic. People watching: Human actions as a cue for single view geometry. In *International Journal of Computer Vision (IJCV)*, 2014.
- [57] Antonino Furnari and Giovanni Farinella. Rolling-unrolling lstms for action anticipation from first-person video. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.
- [58] Antonino Furnari and Giovanni Maria Farinella. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2019.
- [59] Valentin Gabeur, Paul Hongsuck Seo, Arsha Nagrani, Chen Sun, Karteek Alahari, and Cordelia Schmid. Avatar: Unconstrained audiovisual speech recognition. *Interspeech*, 2022.

- [60] Adrien Gaidon, Zaid Harchaoui, and Cordelia Schmid. Temporal localization of actions with actoms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.
- [61] Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Forecasting future action sequences with neural memory networks. *Proceedings of the British Machine Vision Conference (BMVC)*, 2019.
- [62] Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [63] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Red: Reinforced encoder-decoder networks for action anticipation. *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.
- [64] Kirill Gavriluk, Ryan Sanford, Mehrsan Javan, and Cees GM Snoek. Actor-transformers for group activity recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [65] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [66] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [67] Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [68] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.
- [69] Georgia Gkioxari, Ross Girshick, Piotr Dollár, and Kaiming He. Detecting and recognizing human-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [70] Dayoung Gong, Joonseok Lee, Manjin Kim, Seong Jong Ha, and Minsu Cho. Future transformer for long-term action anticipation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [71] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.
- [72] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

- [73] Helmut Grabner, Juergen Gall, and Luc Van Gool. What makes a chair a chair? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [74] Miguel Granados, Kwang In Kim, James Tompkin, Jan Kautz, and Christian Theobalt. Background inpainting for videos with dynamic objects and a free-moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [75] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [76] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 2016.
- [77] Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [78] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [79] Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond Mooney, Trevor Darrell, and Kate Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.
- [80] Abhinav Gupta and Larry S Davis. Objects in action: An approach for combining action understanding and object perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [81] Abhinav Gupta, Aniruddha Kembhavi, and Larry S Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2009.
- [82] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [83] William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *arXiv preprint arXiv:2205.11495*, 2022.

- [84] Jiawei He, Andreas Lehrmann, Joseph Marino, Greg Mori, and Leonid Sigal. Probabilistic video generation using holistic attribute control. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [85] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [86] Fabian Caba Heilbron, Wayner Barrios, Victor Escorcia, and Bernard Ghanem. Sec: Semantic context cascade for efficient action detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2017.
- [87] Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [88] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.
- [89] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [90] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *International Journal of Computer Vision (IJCV)*, 2014.
- [91] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.
- [92] Nouredien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Timeception for complex action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [93] Nouredien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Videograph: Recognizing minutes-long human activities in videos. In *Proceedings of the ICCV Workshop on Scene Graph Representation and Learning*, 2019.
- [94] Mostafa S Ibrahim and Greg Mori. Hierarchical relational networks for group activity recognition and retrieval. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.
- [95] Mostafa S Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A hierarchical deep temporal model for group activity recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [96] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

- [97] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceeding of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [98] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [99] Mihir Jain, Jan Van Gemert, Hervé Jégou, Patrick Bouthemy, and Cees GM Snoek. Action localization with tubelets from motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [100] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as composition of spatio-temporal scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [101] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. <http://crcv.ucf.edu/THUMOS14/>, 2014.
- [102] Ya Jin and Stuart Geman. Context and hierarchy in a probabilistic image model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.
- [103] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [104] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Denscap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [105] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [106] Nal Kalchbrenner, Aäron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.
- [107] Vicky Kalogeiton, Philippe Weinzaepfel, Vittorio Ferrari, and Cordelia Schmid. Joint learning of object and action detectors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2017.
- [108] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [109] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

- [110] Keizo Kato, Yin Li, and Abhinav Gupta. Compositional learning for human object interaction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [111] QiuHong Ke, Mario Fritz, and Bernt Schiele. Time-conditioned action anticipation in one shot. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [112] Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, and Hyunwoo J Kim. Hotr: End-to-end human-object interaction detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [113] Taeksoo Kim, Moon-su Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.
- [114] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.
- [115] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.
- [116] Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [117] Hedvig Kjellström, Javier Romero, and Danica Kragić. Visual object-action recognition: Inferring object affordances from human demonstration. In *Computer Vision and Image Understanding (CVIU)*, 2011.
- [118] Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.
- [119] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision (IJCV)*, 2017.
- [120] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [121] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

- [122] Nilesh Kulkarni, Ishan Misra, Shubham Tulsiani, and Abhinav Gupta. 3d-relnet: Joint object and relational network for 3d prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [123] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [124] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.
- [125] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision(IJCV)*, 2005.
- [126] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [127] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
- [128] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2010.
- [129] Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [130] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.
- [131] Ang Li, Meghana Thotakuri, David A Ross, João Carreira, Alexander Vostrikov, and Andrew Zisserman. The ava-kinetics localized human actions video dataset. *arXiv preprint arXiv:2005.00214*, 2020.
- [132] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [133] Linjie Li, Jie Lei, Zhe Gan, Licheng Yu, Yen-Chun Chen, Rohit Pillai, Yu Cheng, Lu-wei Zhou, Xin Eric Wang, William Yang Wang, et al. Value: A multi-task benchmark for video-and-language understanding evaluation. *arXiv preprint arXiv:2106.04632*, 2021.
- [134] Yin Li, Miao Liu, and James M Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

- [135] Zhenyang Li, Kirill Gavriluk, Efstratios Gavves, Mihir Jain, and Cees GM Snoek. Videolstm convolves, attends and flows for action recognition. *Computer Vision and Image Understanding (CVIU)*, 2018.
- [136] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G Hauptmann, and Li Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [137] Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P Xing. Dual motion gan for future-flow embedded video prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [138] Xiaodan Liang, Lisa Lee, and Eric P Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [139] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post2Conference Workshop of ACL*, 2004.
- [140] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [141] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [142] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems (NIPS)*, 2017.
- [143] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Shiwei Zhang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *IEEE Transactions on Image Processing (TIP)*, 2022.
- [144] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [145] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.
- [146] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *Proceedings of the European conference on computer vision (ECCV)*, 2016.
- [147] Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

- [148] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [149] Yuexin Ma, Xinge Zhu, Sibozhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2019.
- [150] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.
- [151] Edouard Machery, Markus Werning, and Gerhard Schurz. The compositionality of meaning and content. volume ii. applications to linguistics, psychology and neuroscience. *Frankfurt aM [ua]: Ontos*, 2005.
- [152] Tahmida Mahmud, Mahmudul Hasan, and Amit K Roy-Chowdhury. Joint prediction of activity labels and starting times in untrimmed videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [153] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [154] Joanna Materzynska, Tete Xiao, Roei Herzig, Huijuan Xu, Xiaolong Wang, and Trevor Darrell. Something-else: Compositional action recognition with spatial-temporal interaction networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [155] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.
- [156] Effrosyni Mavroudi, Benjamín Béjar Haro, and René Vidal. Representation learning on visual-symbolic graphs for video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [157] Nazanin Mehrasa, Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. A variational auto-encoder model for stochastic point processes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [158] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021.
- [159] Niluthpol Chowdhury Mithun, Sujoy Paul, and Amit K Roy-Chowdhury. Weakly supervised video moment retrieval from text queries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

- [160] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.
- [161] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.
- [162] Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [163] Megha Nawhal, Akash Abdu Jyothi, and Greg Mori. Rethinking learning approaches for long term action anticipation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [164] Megha Nawhal and Greg Mori. Activity graph transformer for temporal action localization. *arXiv preprint arXiv:2101.08540*, 2021.
- [165] Megha Nawhal, Mengyao Zhai, Andreas Lehrmann, Leonid Sigal, and Greg Mori. Generating videos of zero-shot compositions of actions and objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [166] Alejandro Newell and Jia Deng. Pixels to graphs by associative embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [167] Alasdair Newson, Andrés Almansa, Matthieu Fradet, Yann Gousseau, and Patrick Pérez. Video inpainting of complex scenes. In *SIAM Journal on Imaging Sciences*, 2014.
- [168] Yan Bin Ng and Basura Fernando. Forecasting future action sequences with attention: a new approach to weakly supervised action forecasting. *IEEE Transactions on Image Processing (TIP)*, 2020.
- [169] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [170] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.
- [171] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. Action and event recognition with fisher vectors on a compact feature set. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.
- [172] Nada Osman, Guglielmo Camporese, Pasquale Coscia, and Lamberto Ballan. Slowfast rolling-unrolling lstms for action anticipation in egocentric videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [173] Boxiao Pan, Haoye Cai, De-An Huang, Kuan-Hui Lee, Adrien Gaidon, Ehsan Adeli, and Juan Carlos Niebles. Spatio-temporal graph for video captioning with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

- [174] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics (ACL)*, 2002.
- [175] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.
- [176] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.
- [177] AJ Piergiovanni, Anelia Angelova, Alexander Toshev, and Michael S Ryoo. Adversarial generative grammars for human activity prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [178] AJ Piergiovanni and Michael Ryoo. Temporal gaussian mixture layer for videos. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.
- [179] AJ Piergiovanni and Michael Ryoo. Avid dataset: Anonymized videos from diverse countries. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [180] AJ Piergiovanni and Michael S Ryoo. Learning latent super-events to detect multiple activities in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [181] Zhaobo Qi, Shuhui Wang, Chi Su, Li Su, Qingming Huang, and Qi Tian. Self-regulated learning for egocentric video activity anticipation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021.
- [182] Zhiwu Qing, Haisheng Su, Weihao Gan, Dongliang Wang, Wei Wu, Xiang Wang, Yu Qiao, Junjie Yan, Changxin Gao, and Nong Sang. Temporal context aggregation network for temporal action proposal refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [183] Vignesh Ramanathan, Jonathan Huang, Sami Abu-El-Haija, Alexander Gorban, Kevin Murphy, and Li Fei-Fei. Detecting events and key actors in multi-person videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [184] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.
- [185] Scott Reed, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text-to-image synthesis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.
- [186] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

- [187] Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [188] Ivan Rodin, Antonino Furnari, Dimitrios Mavroeidis, and Giovanni Maria Farinella. Untrimmed action anticipation. *arXiv preprint arXiv:2202.04132*, 2022.
- [189] Cristian Rodriguez, Basura Fernando, and Hongdong Li. Action anticipation by predicting future dynamic images. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.
- [190] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [191] Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics (ACL)*, 2020.
- [192] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision (IJCV)*, 2015.
- [193] Michael S Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2011.
- [194] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [195] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022.
- [196] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [197] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems (NIPS)*, 2016.
- [198] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems (NIPS)*, 2017.

- [199] Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [200] Fadime Sener and Angela Yao. Zero-shot anticipation for instructional activities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2019.
- [201] Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. End-to-end generative pretraining for multimodal video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [202] Paul Hongsuck Seo, Arsha Nagrani, and Cordelia Schmid. Look before you speak: Visually contextualized utterances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [203] Xindi Shang, Tongwei Ren, Jingfan Guo, Hanwang Zhang, and Tat-Seng Chua. Video visual relation detection. In *Proceedings of the ACM International Conference on Multimedia (MM)*, 2017.
- [204] Gopal Sharma, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhansu Maji. Csgnet: Neural shape parser for constructive solid geometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [205] Yuping Shen, Fei Lu, Xiaochun Cao, and Hassan Foroosh. Video completion for perspective camera under constrained motion. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2006.
- [206] Dingfeng Shi, Yujie Zhong, Qiong Cao, Jing Zhang, Lin Ma, Jia Li, and Dacheng Tao. React: Temporal action detection with relational queries. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [207] Yifei Shi, Angel X Chang, Zhelun Wu, Manolis Savva, and Kai Xu. Hierarchy denoising recursive autoencoders for 3d scene layout prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [208] Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [209] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2017.
- [210] Zheng Shou, Junting Pan, Jonathan Chan, Kazuyuki Miyazawa, Hassan Mansour, Anthony Vetro, Xavier Giro-i Nieto, and Shih-Fu Chang. Online detection of action start in untrimmed, streaming videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

- [211] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2016.
- [212] Tianmin Shu, Sinisa Todorovic, and Song-Chun Zhu. Cern: confidence-energy recurrent network for group activity recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [213] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
- [214] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.
- [215] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.
- [216] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [217] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don’t judge an object by its context: Learning to overcome contextual bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [218] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.
- [219] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.
- [220] Bilge Soran, Ali Farhadi, and Linda Shapiro. Generating notifications for missing actions: Don’t forget to turn the lights off! In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.
- [221] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *Proceedings of the International conference on Machine Learning (ICML)*, 2015.
- [222] Louise Stark and Kevin Bowyer. Achieving generalized object recognition through reasoning about association of function to structure. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1991.

- [223] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2013.
- [224] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end people detection in crowded scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [225] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [226] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Rahul Sukthankar, Kevin Murphy, and Cordelia Schmid. Relational action forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [227] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [228] Kevin Tang, Bangpeng Yao, Li Fei-Fei, and Daphne Koller. Combining the right features for complex event recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.
- [229] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [230] Yao-Hung Hubert Tsai, Santosh Divvala, Louis-Philippe Morency, Ruslan Salakhutdinov, and Ali Farhadi. Video relationship reasoning using gated spatio-temporal energy graph. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [231] Shubham Tulsiani, Saurabh Gupta, David F Fouhey, Alexei A Efros, and Jitendra Malik. Factoring shape, pose, and layout from the 2d image of a 3d scene. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [232] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [233] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems (NIPS)*, 2016.
- [234] 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 (NIPS)*, 2017.

- [235] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [236] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.
- [237] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.
- [238] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [239] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances in neural information processing systems (NIPS)*, 2016.
- [240] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
- [241] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [242] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.
- [243] Limin Wang, Yu Qiao, and Xiaou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [244] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.
- [245] Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Jan Kautz, and Bryan Catanzaro. Few-shot video-to-video synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [246] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Advances in neural information processing systems (NeurIPS)*, 2018.
- [247] Xiang Wang, Zhiwu Qing, Ziyuan Huang, Yutong Feng, Shiwei Zhang, Jianwen Jiang, Mingqian Tang, Changxin Gao, and Nong Sang. Proposal relation network for temporal action detection. *arXiv preprint arXiv:2106.11812*, 2021.

- [248] Xiang Wang, Shiwei Zhang, Zhiwu Qing, Yuanjie Shao, Zhengrong Zuo, Changxin Gao, and Nong Sang. Oadtr: Online action detection with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [249] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [250] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [251] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [252] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 2004.
- [253] Bohan Wu, Suraj Nair, Roberto Martin-Martin, Li Fei-Fei, and Chelsea Finn. Greedy hierarchical variational autoencoders for large-scale video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [254] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [255] Jianchao Wu, Limin Wang, Li Wang, Jie Guo, and Gangshan Wu. Learning actor relation graphs for group activity recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [256] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [257] Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning-the good, the bad and the ugly. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [258] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [259] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

- [260] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [261] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [262] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [263] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [264] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.
- [265] Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation. *arXiv preprint arXiv:2203.09481*, 2022.
- [266] Bangpeng Yao and Li Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [267] Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani. Compositional video prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [268] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *International Journal of Computer Vision (IJCV)*, 2017.
- [269] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [270] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [271] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.
- [272] Jun Yuan, Bingbing Ni, Xiaokang Yang, and Ashraf A Kassim. Temporal action localization with pyramid of score distribution features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

- [273] Zehuan Yuan, Jonathan C Stroud, Tong Lu, and Jia Deng. Temporal action localization by structured maximal sums. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [274] Olga Zatsarynna, Yazan Abu Farha, and Juergen Gall. Multi-modal temporal convolutional network for anticipating actions in egocentric videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [275] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [276] Kuo-Hao Zeng, William B Shen, De-An Huang, Min Sun, and Juan Carlos Niebles. Visual forecasting by imitating dynamics in natural sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [277] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [278] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. Real-time action recognition with enhanced motion vector cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [279] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [280] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [281] Hao Zhang, Aixin Sun, Wei Jing, Guoshun Nan, Liangli Zhen, Joey Tianyi Zhou, and Rick Siow Mong Goh. Video corpus moment retrieval with contrastive learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [282] Haotong Zhang, Fuhai Chen, and Angela Yao. Weakly-supervised dense action anticipation. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2021.
- [283] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [284] Weiyu Zhang, Menglong Zhu, and Konstantinos G Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.

- [285] Yanyi Zhang, Xinyu Li, Chunhui Liu, Bing Shuai, Yi Zhu, Biagio Brattoli, Hao Chen, Ivan Marsic, and Joseph Tighe. Vidtr: Video transformer without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [286] Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [287] Hang Zhao, Antonio Torralba, Lorenzo Torresani, and Zhicheng Yan. Hacs: Human action clips and segments dataset for recognition and temporal localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [288] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.
- [289] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [290] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2017.
- [291] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.
- [292] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [293] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [294] Song-Chun Zhu, David Mumford, et al. A stochastic grammar of images. *Foundations and Trends® in Computer Graphics and Vision*, 2007.
- [295] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [296] Cheng Zou, Bohan Wang, Yue Hu, Junqi Liu, Qian Wu, Yu Zhao, Boxun Li, Chengguang Zhang, Chi Zhang, Yichen Wei, et al. End-to-end human object interaction detection with hoi transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.