

ABSTRACT

Title of Dissertation: **GENERATING VISUAL CONTENT:
FROM PIXEL ORDERS TO VIDEOS AND BEYOND**

Hanyu Wang
Doctor of Philosophy, 2025

Dissertation Directed by: **Professor Abhinav Shrivastava**
Department of Computer Science

Visual content generation is a fundamental challenge in computer vision that enables diverse applications across domains. The high-dimensional nature of visual data makes it particularly challenging to achieve both quality and precise control in generation tasks. This thesis investigates visual generation across varying levels of abstraction, ranging from fundamental pixel-level ordering to video synthesis, and extending beyond to the unification of perception and creation within large-scale multimodal systems.

We begin by addressing the foundational challenge of sequentially representing visual data through Neural Space-filling Curves, a data-driven approach that learns context-aware pixel orderings optimized for downstream tasks such as LZW compression. We then explore controlled image generation through two complementary approaches: Chop & Learn, a framework for compositional generation that enables synthesis of novel object-state combinations, and a multimodal style transfer method that effectively combines guidance from both images and text. For video generation, we introduce LARP, a novel tokenization approach with a learned autoregressive prior

that achieves state-of-the-art performance while maintaining computational efficiency. Finally, we present Bridge, a unified framework that equips pre-trained MLLMs with visual generative capabilities. By utilizing a Mixture-of-Transformers architecture to handle conflicting modalities and a novel semantic-to-pixel discrete representation, Bridge enables high-precision visual understanding and high-fidelity generation within a single model, effectively closing the loop between perception and creation.

GENERATING VISUAL CONTENT:
FROM PIXEL ORDERS TO VIDEOS AND BEYOND

by

Hanyu Wang

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:

Professor Abhinav Shrivastava, Chair/Advisor
Professor Shuvra S. Bhattacharyya, Dean's Representative
Professor Furong Huang
Professor Pratap Tokekar
Professor Ruohan Gao

© Copyright by
Hanyu Wang
2025

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Abhinav Shrivastava, for his exceptional guidance throughout my PhD journey. He has always been flexible regarding my research topics and incredibly patient with my progress. This autonomy gave me the freedom to explore various aspects of visual generation, which ultimately made this dissertation possible. Abhinav is insightful and supportive; he consistently provided valuable technical advice when I faced challenges and ensured that I had the essential resources necessary to bring my projects to fruition. I am deeply grateful to have had him as my advisor.

I am also deeply thankful to my dissertation and preliminary exam committee members: Professor Shuvra S. Bhattacharyya, Professor Furong Huang, Professor Pratap Tokekar, Professor Ruohan Gao, Professor Tianyi Zhou, and Dr. Ishan Misra. Their critical feedback and insightful suggestions have significantly contributed to the depth and rigor of this dissertation.

My heartfelt appreciation goes to my colleagues and friends in the Perception & Intelligence lab, including Bo He, Yixuan Ren, Hao Chen, Shuaiyi Huang, Kamal Gupta, Nirat Saini, Saksham Suri, Shishira R. Maiya, Archana Swaminathan, Sharath Girish, Max Ehrlich, Vinoj Jayasundara, Chuong Huynh, Pulkit Kumar, Matthew Gwilliam, Gaurav Shrivastava, Anubhav Gupta, Luyu Yang, Khoi Pham, Soumik Mukhopadhyay, Matthew Walmer, Mara Levy, Alex Hanson, Saketh Rambhatla, Daniel Ekpo, Namitha Padmanabhan, Lillian Huang, Vatsal Agarwal, Prateksha Udhayanan, Moustafa Meshry, for creating such an intellectually stimulating en-

vironment. Our daily discussions, shared insights, and collaborative spirit were instrumental in shaping my research perspective and keeping me motivated.

I would also like to express my sincere gratitude to my mentors and collaborators from industry for their guidance and partnership during my internships. I am especially grateful to Pengxiang Wu, Kevin Dela Rosa, Derek Hao Hu, and Chen Wang at Snap; Chenyang Zhang, Haoyue Tang, and Tian Xie at Meta; and Hao Chen, Jiaming Han, Ziyang Yang, Qi Zhao, Shanchuan Lin, Lu Jiang, and Zhenheng Yang at ByteDance. I am deeply appreciative of the mentorship and the collaborative discussions we shared, which provided me with valuable industrial perspectives.

I am equally grateful to my friends at UMD, including Yiran Xu, Chen Chen, Peihong Yu, Jiayong Lin, Fan Yang, Zhichao Liu, Mingyang Xie, Yue Feng, and Hengduo Li. Their companionship, especially during the difficulties of the pandemic, was an unwavering source of strength and support.

My deepest appreciation goes to my family, who have been my unwavering source of strength. I thank my parents for nurturing the curiosity and perseverance that made this degree possible, and whose steadfast support from across the globe has sustained me throughout this journey. Most importantly, I offer my heartfelt gratitude to my wife, Stephanie Zhou, who has been my anchor. Her patience, understanding, and constant support during the most challenging moments of my doctoral studies were indispensable to the completion of this dissertation.

Finally, while it is impossible to list every individual who has supported this journey, I extend my sincere gratitude to the many friends, colleagues, and mentors not explicitly named here. Their collective encouragement, assistance, and camaraderie were instrumental to the completion of this dissertation, and their contributions are deeply appreciated.

Table of Contents

Acknowledgements	ii
Table of Contents	iv
List of Tables	vii
List of Figures	ix
Chapter 1: Introduction	1
Chapter 2: Generating Pixel Orders: Neural Space-filling Curves	8
2.1 Introduction	8
2.2 Related Work	10
2.3 Approach	12
2.3.1 Overview of Dafner <i>et al.</i> (Single Image-based SFC)	12
2.3.2 Neural Space-filling Curves	14
2.3.3 Weight Generator	15
2.3.4 Objective Functions	17
2.3.5 Weight Evaluator	18
2.3.6 Training	19
2.4 Experiments	20
2.4.1 Datasets	21
2.4.2 Training details	21
2.4.3 Qualitative Evaluation	22
2.4.4 Optimizing Autocorrelation	24
2.4.5 Optimizing Code Length	25
2.4.6 Scaling up SFC	27
2.5 Conclusions and Future Work	27
Chapter 3: Generating Images: Exploring Novel Object-State Compositions	29
3.1 Introduction	29
3.2 Related Work	33
3.3 Chop & Learn	36
3.3.1 Design Choices	36
3.3.2 Data Collection Setup	39
3.4 Compositional Image Generation	40
3.4.1 Methods	42

3.4.2	Evaluation	43
3.4.3	Results and Discussion	44
3.5	Extension: Compositional Action Recognition	47
3.5.1	Dataset Splits	49
3.5.2	Results	50
3.6	Discussion	51
3.7	Conclusion	53
Chapter 4:	Generating Images: Style Transfer Meets Multiple Modalities	54
4.1	Introduction	54
4.2	Related Work	57
4.2.1	Image-guided Image Style Transfer	57
4.2.2	Text-guided Image Manipulation	58
4.2.3	GAN Inversion	59
4.3	Method Overview	60
4.4	Cross-modal GAN Inversion	62
4.4.1	Style-specific CLIP Loss	62
4.4.2	Inversion Algorithm	65
4.5	Multimodality-guided Image Style Transfer	66
4.5.1	Multi-style Boosting	66
4.5.2	Style Transfer Algorithm	67
4.6	Experiments	67
4.6.1	Comparison with TIST Methods	68
4.6.2	MMIST and Cross-modal Style Interpolation	71
4.6.3	Ablation Study	71
4.7	Conclusions and Future Work	73
Chapter 5:	Generating Videos: Video Tokenization with Learned Autoregressive Prior	74
5.1	Introduction	74
5.2	Related Work	78
5.2.1	Discrete Visual Tokenization	78
5.2.2	Visual Generation	79
5.3	Method	79
5.3.1	Preliminary	79
5.3.2	Holistic Video Tokenization	82
5.3.3	Learning an Autoregressive Generative Prior	85
5.4	Experiments	87
5.4.1	Setup	87
5.4.2	Scaling	88
5.4.3	Video Generation Comparison	90
5.4.4	Visualization	90
5.4.5	Ablation Study	92
5.5	Conclusion and Future Work	93

Chapter 6:	Generating with MLLMs: Growing Visual Generative Capacity for Pre-Trained Models	94
6.1	Introduction	94
6.2	Related Work	97
6.3	Method	98
6.3.1	Architecture	98
6.3.2	Semantic-to-pixel Discrete Visual Representation	100
6.3.3	Data Curation	101
6.3.4	Training Procedure	102
6.4	Experiments	104
6.4.1	Main Results	106
6.4.2	Ablation Studies	108
6.5	Conclusion	111
Chapter 7:	Future Work and Conclusion	112
7.1	Future Work	112
7.2	Conclusion	113

List of Tables

2.1	Comparison of performance of lag- k autocorrelation and LZW Encoding length for different orders. For autocorrelation, we consistently outperform both the universal and context-based SFC computation approaches at high values of k . For LZW Encoding length, we measure the average size per frame in bytes as well as the relative improvement compared to the raster scan order, in the case of each of the datasets. We consistently outperform compression performance for other order schemes.	26
3.1	Comparison with other video datasets. This table highlights the distribution of the objects, states and compositions in different datasets. Obj. refers to objects, Comp. is compositions of objects and styles, N refers to the number of compositions that have more than 10 samples, and Styles* refers to grouping of styles: instead of generic names like cut, chop, etc., we use 3 distinct styles (chop/dice, peel, grate) as styles. MIT-States [†] is the only image-based dataset, the rest are video-based datasets. All these data numbers are for edible objects and cutting style actions from respective datasets. Our dataset has uniform distribution for each metric in the table, which makes it suitable for learning objects and their states.	32
3.2	Compositional generation evaluation. FID, user scores, and classifier scores of various generative models. User Realism is on a scale of 1-5. (★) denotes that accuracies are evaluated on a seen data split. Bold represents the best result. . . .	41
3.3	Compositional action recognition results. “Start/End” denote the prediction results for the initial and the final state composition with the corrected object type. Bold and <u>underline</u> represent the top-1 and top-2 results.	45
3.4	Green screen removal evaluation. Both rows employ the SD+FT+TI but are trained using images with varying backgrounds. Classifiers specific to each dataset are trained to assess Classifier Acc. Validation images used to calculate Patch FID differ between the two rows. Patch FID Lower Bound is computed by evaluating the patch FID on one-half of the validation images relative to the other half. For further details, refer to Section 3.4.3.	47
4.1	Comparison between traditional GAN inversion and cross-modal GAN inversion. “Ref.” means reference.	63
4.2	Quantitative user study results on four TIST methods. For each method, we report the number of positive responses received from raters, as well as its percentage (over 161,040 responses in total). Our method outperforms baselines. . .	68

4.3	Speed comparison with other TIST methods. Our method is the fastest. (★) Null-text Inversion does not need per-style training but has a per content inversion time of 80.7s.	70
4.4	Ablation study on different design choices. The performance is evaluated through the user study.	73
5.1	Comparison of video generation results. Results are grouped by the type of generative models. The scores for MAGVIT-AR and MAGVIT-v2-AR are taken from the appendix of MAGVIT-v2 [1]. LARP-L-Long denotes the LARP-L trained for more epochs. Our best results are obtained with a larger AR generator.	89
5.2	Ablation study. All configurations are modified from LARP-B model.	92
6.1	Quantitative Results on Visual Understanding Benchmarks. Models without a specified base (M)LLM are trained from scratch. A dash (–) is used when the corresponding value was not provided in the original work.	104
6.2	Quantitative Results on Text-to-image Generation Benchmarks. † refers to the methods using prompt augmentation, <i>e.g.</i> , LLM rewriter or self-CoT [2]. A dash (–) is used when the corresponding value was not provided in the original work.	105
6.3	Comparison results on ImgEdit. Bridge achieves the best overall performance.	106
6.4	Visual Representation Comparison. Sem: Semantic token. Pix: Pixel token. . .	107
6.5	MLLM Architecture Comparison. Und. denotes the harmonic mean of un- derstanding benchmarks including MME, MMBench, SEED Bench, POPE, and MMMU.	107
6.6	Token Routing Comparison. Visual understanding and generation performance under different token routing schemes. In the Token Routing columns, Und. refers to the understanding expert and Gen. refers to the generation expert. . . .	108

List of Figures

1.1	Overview of visual content generation problems addressed in this thesis. (a) <i>Generating pixel orders.</i> We investigate the generation of optimal space-filling curves (SFCs). These SFCs determine pixel ordering in images and videos, optimized for specific downstream tasks. (b) <i>Generating images.</i> We introduce multiple control mechanisms for image generation. Through our compositional dataset Chop & Learn, we train generative models capable of novel object-state compositions. Additionally, we enable image style transfer guided by arbitrary number of multimodal style representations. (c) <i>Generating videos.</i> We bridge the gap between video tokenization and generation by developing a generation-aware video tokenizer. This is achieved by incorporating a learned autoregressive generative prior during tokenizer training. (d) <i>Generating with MLLMs.</i> We introduce Bridge, a unified framework that equips pre-trained MLLMs with visual generative capabilities. It utilizes a Mixture-of-Transformers architecture to handle conflicting modalities and a novel semantic-to-pixel discrete visual representation to improve visual generation fidelity.	2
2.1	Given a set of images, a gif, or a video, Neural Space-filling Curves (SFC) can provide a more spatially coherent scan order for images as compared to universal scan orders such as S-curve, or Peano-Hilbert curves. As shown in the example of a trouser and a face, the scan line tends to cover the background before moving to the foreground. (SFCs generated here using half-resolution images and resized for clarity. Best viewed in color.)	9
2.2	Cover and Merge Algorithm. (a): An 8×8 image fully covered by 2×2 circuits. (b): The dual graph $\mathcal{G}'$ (black) built on the covering circuits $\mathcal{G}$ (blue). (c): The Minimum Spanning Tree $\mathcal{T}$ (black solid lines) of $\mathcal{G}'$ (all black lines) and the Hamiltonian Circuit (blue) induced by $\mathcal{T}$. (d): A single Hamiltonian circuit merged from the covering circuits. See more details in section 2.3.1.	12
2.3	Neural SFC pipeline. The Neural SFC model consists of a weight generator $F_{\mathcal{G}}$ and a weight evaluator $E_{\mathcal{G}}$. Taking one or more images as input, $F_{\mathcal{G}}$ extracts the deep features of the image and generates the SFC weights. $E_{\mathcal{G}}$ takes both the image(s) and the corresponding SFC weights as input $E_{\mathcal{G}}$ then estimates the negative autocorrelation of the image pixel sequence inferred by the input weights to evaluate the goodness of the input weights with respect to the input image. See more details in section 2.3.2.	14

2.4	Qualitative comparison between Hilbert curves and Neural SFCs. Left: SFC (in red color) overlaid on the image. Right: Image flattened according to the SFC and visualized in 1-dimension. Images in the top two rows are from MNIST, the ones in the middle two rows are from Fashion-MNIST, and the ones in the bottom two rows are from FFHQ Faces. Neural SFCs on images from MNIST and Fashion-MNIST are class-conditional, <i>i.e.</i> , computed for each class. Therefore, for MNIST and Fashion-MNIST, Neural SFCs in the right two columns are the same since the two images have the same class label. In all datasets, Neural SFCs are more spatially coherent and produce fewer clusters when visualized in 1-dimension. Best viewed in color.	23
2.5	We visualize the Neural SFCs training with two objectives considered in this chapter – autocorrelation and LZW encoding.	24
2.6	lag- k autocorrelation for MNIST, Fashion-MNIST and FFHQ datasets. While Dafner <i>et al.</i> provide higher autocorrelation for small lag, <i>i.e.</i> , from $k = 2$ to $k = 4$, Neural SFCs outperform Dafner <i>et al.</i> for $k > 4$ in all the datasets. Note that we trained our model for $k = 6$, and hence this behaviour is expected.	24
2.7	Scaling up SFC. The top row (blue circles) shows a 2×2 crop of an image grid of resolution $n \times n$. The bottom row (red circles) shows how we can scale the SFC path from $n \times n$ grid to $2n \times 2n$ grid. For every incoming SFC to a pixel location, there are 3 possible ‘next’ pixels, straight ahead, left, or right. Column 2 and 3 consider the cases where the SFC goes ‘straight’ or ‘right’. The image on the right shows a toy example of scaling up SFC for a 5×5 image.	27
3.1	We present Chop & Learn (ChopNLearn) , a new dataset and benchmark suite for the tasks of Compositional Image Generation and Compositional Action Recognition. It consists of 1260 video clips and 112 object state combinations captured from multiple viewpoints for 20 objects and 8 cut styles. We also propose two new compositional tasks and benchmarks - (1) Image Generation: given training images of various objects in various states , the goal is to generate images of unseen combinations of objects and states . (2) Action Recognition: training videos are used to recognize objects along with transition from state1 $\rightarrow$ state2 , to generalize on recognizing unseen object-state transitions.	30
3.2	<u>Left</u> : We show examples of cutting styles from popular video datasets (VISOR [3]: chop and peel potato, Youcook2 [4]: chop broccoli, peel radish), image dataset (MIT-states [5]:slice pear, peel orange) and generation pipelines (DALL-E [6]:baton cut apple, half round slices tomato). Most of these are either too noisy to capture subtle differences in objects or do not have the granularity of specific cutting styles. <u>Center</u> : Our 4 camera setup captures videos of one object in 4 different views. <u>Right</u> : We capture 8 styles of object states, which can be derived in a hierarchical manner from larger to small cuts. Each style is of different shape and granularity.	33
3.3	Statistics for ChopNLearn: We show the number of samples for each object-style composition in a color-coded manner: orange represents 12 samples, green represents 8 samples and blue represents 4 samples.	37

3.4	(a) The clip length distribution for one camera (315 unique clips). (b) Preliminary results of using green screen to augment the dataset with different backgrounds. We continue to improve the transfer results by adding shadows and background matting.	37
3.5	Compositional Generation Samples. Ground Truth (GT) real images are shown in the first row for reference. Seven object-state combinations in the test set are displayed, each with two generated samples for each method. Please zoom in to see details.	40
3.6	Video parsing graph: For a given video, we use Grad-CAM[7] on the intermediate frames to identify and visualize the class activation maps corresponding to the most salient states. Top: A training video clip has one transition of orange from large cut → small cut. Bottom: We can learn single transitions from training data, to generalize transitions in a long video with multiple state changes and parse the video as a graph.	51
4.1	Our multimodality-guided image style transfer results. We show style transfer results guided by two text styles (left) and two image styles (right). 15 stylized images are synthesized by evenly interpolating between these four styles.	55
4.2	Failure case of CLIPStyler. We show one content and one style text description, together with the results from CLIPStyler and our method. CLIPStyler adds many small face-like patterns to the stylized images.	57
4.3	The overview of our method. Taking styles from multiple modalities as input, our method generates style representations using cross-modal GAN inversion. With the adapted IIST model, our method can apply the cached styles to any unseen content image in a single forward pass.	61
4.4	Comparison with other TIST methods. Our method delivers more accurate styles than others. Meanwhile, as opposed to other methods which often distort the content or add unreasonable patterns, content information is greatly preserved in results of our method.	66
4.5	MMIST results. The first and second rows show style text descriptions and style images, respectively. The first column shows content images. Our method successfully mixes multimodal styles and applies them to various content images.	70
4.6	MMIST and Style interpolation with four image and text styles. Please zoom in to see the details.	72
5.1	LARP highlights. (a) LARP is a video tokenizer for two-stage video generative models. In the first stage, LARP tokenizer is trained with a lightweight AR prior model to learn an AR-friendly latent space. In the second stage, an AR generative model is trained on LARP’s discrete tokens to synthesize high-fidelity videos. (b) The incorporation of the AR prior model significantly improves the generation FVD (gFVD) across various token number configurations. (c) LARP shows a much smaller gap between its reconstruction FVD (rFVD) and generation FVD (gFVD), indicating the effectiveness of the optimized latent space it has learned.	75
5.2	Method overview. Cubes  represent video patches, circles  indicate continuous embeddings, and squares 	

	denote discrete tokens. (a) Patchwise video tokenizer used in previous works.	
	(b) Left: The LARP tokenizer tokenizes videos in a holistic scheme, gathering information from the video using a set of learned queries. Right: The AR prior model , trained with LARP , predicts the next holistic token, enabling a latent space optimized for AR generation. The AR prior model is forwarded in two rounds per iteration. The red arrow represents the first round, and the purple arrows represent the second round. The reconstruction loss $\mathcal{L}_{\text{rec}}$ is omitted for simplicity.	80
5.3	Scaling LARP tokenizer size and number of tokens.	87
5.4	Video reconstruction comparison with OmniTokenizer [8].	91
5.5	Class-conditional video generation results on the UCF-101 dataset using LARP.	91
5.6	Video frame prediction results on the K600 dataset using LARP.	92
6.1	Method overview. Bridge adopts a Mixture-of-Transformers (MoT) architecture with two experts: a frozen understanding (Und.) expert for text and visual understanding tokens, and a newly trained generation (Gen.) expert for visual generation tokens. Both experts share unified causal attention across all tokens. Visual generation representation are constructed by concatenating short semantic token sequences with longer pixel token sequences, which are modeled jointly with text tokens under a unified next-token prediction objective. Semantic tokens serve as a bridge between text and pixel modalities, substantially improving visual generation quality.	98
6.2	Qualitative Results on text-to-image generation and image editing tasks.	103

Chapter 1: Introduction

Visual content generation sits at the intersection of computer vision, machine learning, and computational creativity. The core challenge lies in developing systems that can synthesize diverse visual content - from images to videos - with high fidelity while maintaining precise control over the generation process. This task demands innovations across multiple technical dimensions: architectural design for neural networks, principled approaches to generative modeling, and effective ways to leverage large-scale visual data. While the field has evolved from constrained scenarios like digit generation to more complex synthesis tasks, fundamental challenges remain in achieving both generality and fine-grained control.

Visual content generation exemplifies the challenge of bridging low-level pixel representations with high-level semantic understanding. While digital images are fundamentally arrays of pixels, human perception interprets them as coherent scenes with objects, textures, and semantic meaning. This gap between raw data and semantic interpretation drives the development of generation approaches that operate at multiple levels of abstraction - from pixel-level manipulations to semantic-aware synthesis. Foundation models and neural architectures have enabled new capabilities in this direction, allowing systems to generate images from text, edit visuals through natural language, and synthesize coherent videos.

This thesis investigates visual content generation across multiple dimensions, exploring

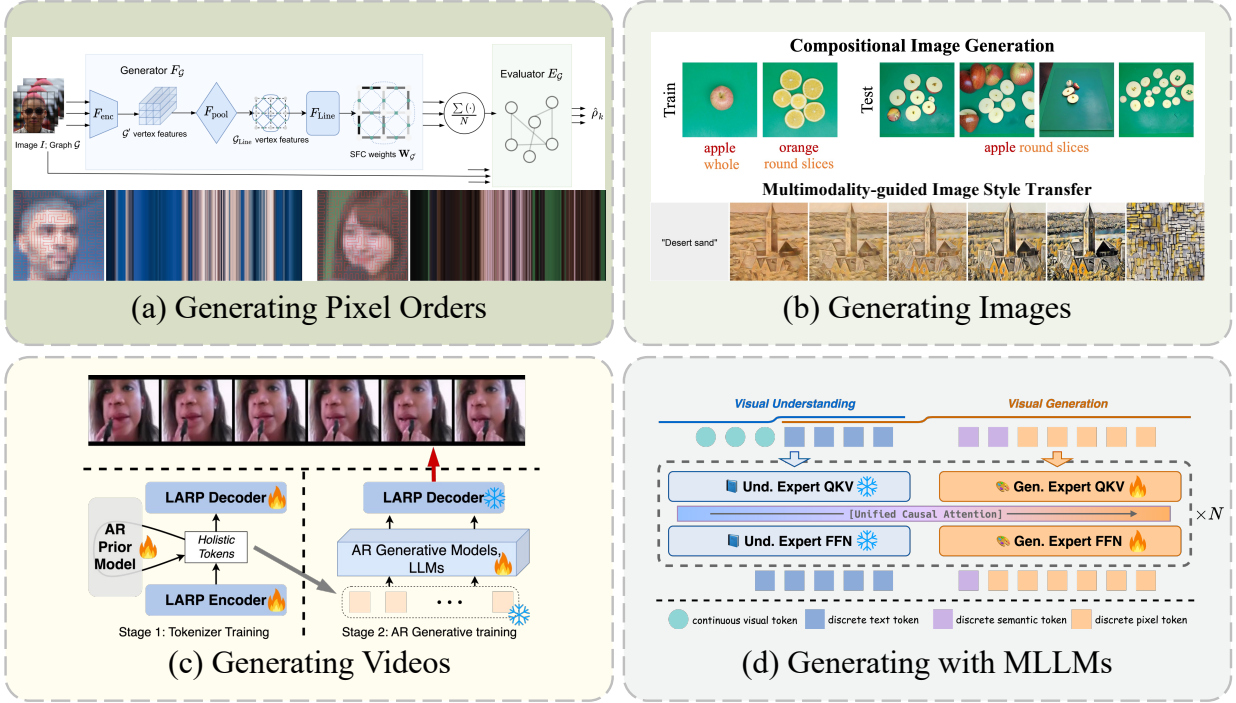


Figure 1.1: **Overview of visual content generation problems addressed in this thesis.** (a) *Generating pixel orders.* We investigate the generation of optimal space-filling curves (SFCs). These SFCs determine pixel ordering in images and videos, optimized for specific downstream tasks. (b) *Generating images.* We introduce multiple control mechanisms for image generation. Through our compositional dataset Chop & Learn, we train generative models capable of novel object-state compositions. Additionally, we enable image style transfer guided by arbitrary number of multimodal style representations. (c) *Generating videos.* We bridge the gap between video tokenization and generation by developing a generation-aware video tokenizer. This is achieved by incorporating a learned autoregressive generative prior during tokenizer training. (d) *Generating with MLLMs.* We introduce Bridge, a unified framework that equips pre-trained MLLMs with visual generative capabilities. It utilizes a Mixture-of-Transformers architecture to handle conflicting modalities and a novel semantic-to-pixel discrete visual representation to improve visual generation fidelity.

challenges of increasing complexity and scope. As illustrated in Figure 1.1, we begin with fundamental questions about optimal pixel ordering for downstream tasks, focusing on minimizing pixel sequence autocorrelation and LZW encoding length for both images and videos. We then advance to the problem of image generation with controls. First, we address compositional image generation by developing the Chop & Learn dataset and framework, enabling generative models to create novel object-state combinations. In the second part, we consider multimodality-guided

image style transfer, presenting a framework that can effectively combine style information from both images and text to guide generation. Next, we extend our investigation to videos by introducing a novel tokenization approach with a learned autoregressive prior. This method significantly improves the quality of generated videos while maintaining computational efficiency. Finally, we propose research projects regarding generating diverse visual content through various methodological approaches.

While images and videos represent the most familiar forms of visual content, fundamental structures like pixel orders, particularly Space-filling Curves (SFCs), play a crucial role in visual data processing. SFCs provide continuous scan orders that traverse all spatial locations in two or higher dimensional signals exactly once, forming a bridge between multi-dimensional visual data and one-dimensional representations.

Traditional SFCs, such as Hilbert curves and Peano curves, are context-agnostic - they generate fixed traversal patterns based solely on the dimensions of the space, without considering the actual content being scanned. While these universal curves have found applications in various domains from image compression to parallel computing, their inability to adapt to specific content limits their effectiveness. In Chapter 2, we introduce Neural Space-filling Curves, a data-driven approach that learns to generate content-aware scanning patterns optimized for specific downstream tasks.

Through Neural SFCs, we demonstrate how SFCs can benefit from learned approaches. By formulating SFC generation as a graph-based optimization problem, our method can learn scanning patterns that minimize objectives such as pixel sequence autocorrelation or LZW compression code length. This adaptive approach not only improves performance on traditional applications like image compression but also opens new possibilities for content-aware scanning in

domains such as video processing and autoregressive generative modeling.

Then, we turn our attention to a core challenge in image generation: compositional generalization. While recent advances in generative models have demonstrated impressive capabilities in synthesizing high-quality images, these models often struggle to combine familiar concepts in novel ways - a capability that humans possess naturally. To study this challenge systematically, we focus on a specific yet ubiquitous domain: the visual changes that occur when objects are cut in different styles.

In Chapter 3, we introduce Chop & Learn, a comprehensive benchmark suite designed to study object compositions and state transformations. The dataset systematically captures 20 different objects undergoing transformation through 8 distinct cutting styles, with each transformation recorded from multiple viewpoints. This carefully structured data collection enables robust learning of the relationships between objects and their possible states. Building on this foundation, we develop novel approaches for compositional image generation that can effectively transfer learned cutting styles across different objects. Our generative models demonstrate the ability to synthesize realistic visualizations of previously unseen object-state combinations. For example, after learning the visual patterns of round-sliced apples and baton-cut carrots separately, our model can successfully generate images of apples cut into batons, despite never encountering this specific combination during training. These results represent significant progress toward achieving true compositional generalization in visual generation systems.

We next explore how to enable more flexible control over the generation process through multiple modalities. While traditional image style transfer methods rely solely on reference images to define styles, and recent text-based approaches use only natural language descriptions, real-world creative processes often benefit from combining multiple forms of guidance. In Chap-

ter 4, we introduce a framework for multimodality-guided image style transfer that can effectively combine style information from both images and text.

At the core of our approach is a novel cross-modal GAN inversion technique that maps style information from different modalities into a unified latent space. This enables our framework to interpret and combine style guidance from multiple sources, whether they are reference images, text descriptions, or a combination of both. For example, an artist could refine a visual style by providing both a reference artwork and textual modifications, allowing for more precise and intuitive control over the generation process.

Our method demonstrates superior performance compared to existing approaches in both traditional image-guided and text-guided style transfer scenarios. More importantly, it enables entirely new workflows by supporting cross-modal style interpolation and combination, expanding the creative possibilities available to users while maintaining high-quality output. This work represents a significant step toward more flexible and intuitive control mechanisms for visual content generation.

Subsequently, we extend our investigation to video generation, addressing the challenge of making autoregressive generation of videos more effective and efficient. While autoregressive models have achieved remarkable success in language modeling and image generation, their application to videos has faced significant challenges due to the inherent complexity and temporal dependencies of video data. To address these challenges, we focus on the two-stage video generation paradigm, which offers compelling advantages in terms of memory efficiency and computational requirements. This paradigm first learns video tokenizers to compress videos into a compact latent space, then performs the generation process within this optimized representation.

In Chapter 5, we introduce LARP (Learned AutoRegressive Prior), a novel video tokenizer

specifically designed for autoregressive video generation. By incorporating a learned autoregressive prior during training, LARP creates a latent space that is inherently optimized for subsequent autoregressive generation, bridging the gap between video compression and generation in a principled way.

LARP differentiates itself from traditional video tokenizers through two key innovations. First, it introduces a holistic tokenization scheme that captures global semantic information rather than being constrained by local patches. Second, it incorporates a lightweight autoregressive prior during training, which helps optimize the discrete latent space for subsequent autoregressive generation. By combining these innovations, LARP achieves state-of-the-art performance in video generation while maintaining computational efficiency, establishing new benchmarks on standard datasets like UCF101 and Kinetics-600.

Finally, we culminate our research by addressing the unification of visual understanding and generation within Multimodal Large Language Models (MLLMs). While recent MLLMs have achieved remarkable success in visual understanding tasks, they typically lack the ability to generate visual content, creating a functional divide between understanding and synthesis.

In Chapter 6, we introduce Bridge, a framework specifically designed to grow visual generative capacity for pre-trained understanding-only MLLMs. Bridge overcomes the challenge of conflicting objectives between continuous understanding embeddings and discrete generation tokens. We propose a Mixture-of-Transformers architecture that augments the pre-trained model with generative capabilities within a single next-token prediction framework. To ensure high generation fidelity and language alignment, we introduce a novel semantic-to-pixel discrete representation that integrates compact semantic tokens with fine-grained pixel tokens. This approach allows Bridge to achieve competitive performance in both visual understanding and generation

benchmarks, effectively closing the loop between perceiving and creating visual content.

Through these major contributions, spanning from fundamental pixel order to complex video and unified multimodal generation, this thesis advances the field’s capability to generate diverse and high-quality visual content. Each component addresses distinct challenges while maintaining practical applicability, demonstrating how learned approaches can significantly improve generation across different visual modalities and levels of abstraction.

Chapter 2: Generating Pixel Orders: Neural Space-filling Curves

2.1 Introduction

In any form of digital communication, information is transmitted via a sequence of discrete symbols. This includes images and videos, even though they are inherently signals with two spatial dimensions (2D). The modus operandi for transmitting such signals is to (1) efficiently encode and quantize their values in the spatial or spectral domain, (2) linearize the signal to a one-dimensional (1D) sequence by using a standard scanning order such as raster, zig-zag, or Hilbert Curve order [9], and finally (3) apply a Shannon [10] style entropy coding technique such as Arithmetic coding [11] or Huffman coding [12] to further compress the 1D sequence. Given the ubiquity of images and videos in our lives, a large amount of effort has gone into optimizing each of these steps of digital communication. The focus of this work is the second step of linearizing the 2D spatial signal to a 1D sequence. A continuous scan order that traverses all spatial locations in two or higher dimensional signals exactly once is also known as the space-filling curve (SFC) [13].

Prior works have proposed various space-filling curves (SFCs) in the last hundred years, most of them context-agnostic, *i.e.*, they are completely defined by the size and dimension of the space without taking into account spatial information of the space, *e.g.*, pixels in the case of two-dimensional images. These universal context-agnostic SFCs are typically defined recursively to

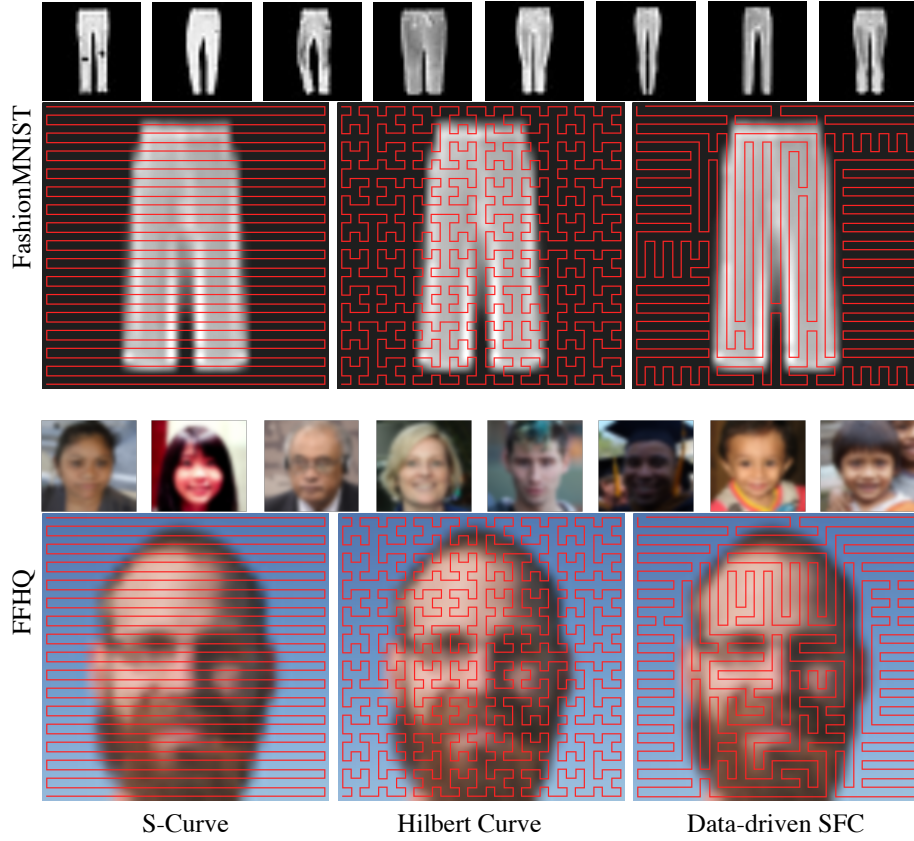


Figure 2.1: Given a set of images, a gif, or a video, Neural Space-filling Curves (SFC) can provide a more spatially coherent scan order for images as compared to universal scan orders such as S-curve, or Peano-Hilbert curves. As shown in the example of a trouser and a face, the scan line tends to cover the background before moving to the foreground. (SFCs generated here using half-resolution images and resized for clarity. Best viewed in color.)

ensure simplicity and scale. Some of the SFCs also have spatial coherence properties and have been used in various image-based applications [14–17].

However, in many applications such as video conferencing, health-care, or social media, the images being transmitted, are often repetitive with a similar layout and content with minor variations transmitted over and over again. GIFs are another great example that consists of highly repetitive content and need to be stored efficiently and often, losslessly. Since universal SFCs do not utilize the intrinsic information of image content, they are far from optimal for a single image or a set of images with repetitive structure (refer to fig. 2.1 for an example). Dafner *et al.* [18]

proposed SFCs that exploit the inherent correlation between pixel values in an image. Our work improves upon Dafner *et al.* , in two aspects.

- Instead of discovering a single SFC for every image independently, we propose a data-driven technique to find optimal SFCs for a set of images. We postulate that context-based SFCs are more suitable for linearizing a group of images (or a short video/gif), since the cost of storing the SFC itself can be amortized by the number of images.
- We devise a novel alternating minimization technique to train an SFC weights generator, which allows us to optimize for any given objective function, even when not differentiable.

To the best of our knowledge, ours is the first work to propose a machine learning method for computing context-based SFCs and opens new directions for future research on optimal scanning of 2D and 3D grid-based data structures such as images, videos, and voxels. We demonstrate both quantitatively and qualitatively the benefit of our approach in various applications.

2.2 Related Work

Space-filling Curves (SFCs), introduced by Peano in 1890 [13] and Hilbert in 1891 [9], are injective functions that map a line segment to a continuous curve in the unit square, cube, or hypercube. Most classic SFCs such as Peano-Hilbert, Sierpinski [19], and Moore [20] curves are defined recursively which allows them to scale up to arbitrary resolution with a number of favorable properties. In fact, [21] showed that the entropy of this pixel sequence asymptotically converges to the two-dimensional entropy of the original image for a large number of images coming from sufficiently random sources. Because of their self-organizing capabilities, Hilbert

curves have found applications in compression [14–17], computing [22], recognition [23, 24], security [25], databases [26], electronics [27], biology [28] and even web-comics [29]. Hilbert curves have also been used to solve multidimensional task allocation problems in parallel processing [30]. This scheme is used in many job schedulers, such as the famous SLURM [31].

Dafner *et al.* [18] proposed the first context-based SFCs. Their work makes use of cover and merge algorithm [25] to compute SFC for an image. Ouni *et al.* [32] employ image gradient based method to compute context-based SFCs, and they apply their SFCs to lossless compression tasks. Zhou *et al.* [33] improve Dafner *et al.* ’s method by considering both data values and location coherency. They also generalize their method to multiscale data via quadtrees and octrees.

However, computing a unique SFC specific to an image has limited applications. Compression techniques such as LZW [21] that exploit the correlation between nearby pixel values during encoding, for example, can do a better job with context-based SFCs, however, the cost of storing an SFC itself for each image gives away the advantage. Universal SFCs, such as Hilbert curves don’t have this disadvantage and are frequently used in compression applications. In this work, we argue that, applications that need to store and transmit images with repetitive structures such as gifs, can benefit greatly from context-based SFC optimized for that set of images.

Finding an optimal SFC is a combinatorial optimization problem. Solving combinatorial problems have a rich history in the field of machine learning. In 1985, [34] first attempted to solve these problems using a neural network. In the deep learning era, various flavors of attention have been proposed [35–37] to reach an approximate solution of NP-hard problems in computer science. Since combinatorial optimization problems are inherently non-differentiable, reinforcement learning (RL) is also a promising alternative for addressing these problems [36–39]. In our initial experiments, we found the performance of RL approach in our use-case to be

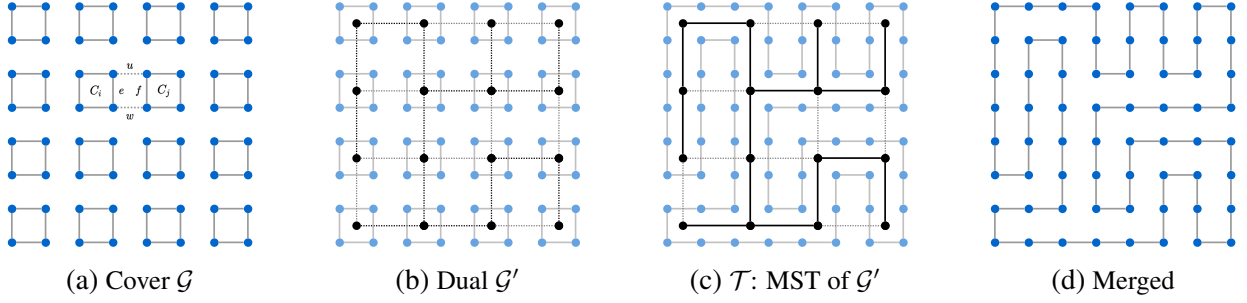


Figure 2.2: **Cover and Merge Algorithm.** (a): An 8×8 image fully covered by 2×2 circuits. (b): The dual graph $\mathcal{G}'$ (black) built on the covering circuits $\mathcal{G}$ (blue). (c): The Minimum Spanning Tree $\mathcal{T}$ (black solid lines) of $\mathcal{G}'$ (all black lines) and the Hamiltonian Circuit (blue) induced by $\mathcal{T}$. (d): A single Hamiltonian circuit merged from the covering circuits. See more details in section 2.3.1.

unstable and inconsistent.

2.3 Approach

We first describe the algorithm for computing SFC for one image as proposed by Dafner *et al.* [18] in Section 2.3.1. We then extend this treatment to a more general setting where we can optimize the SFC for any non-differential objective function for multiple images in Section 2.3.2. The rest of the Section 2.3 describes major components of our framework and training procedure in detail.

2.3.1 Overview of Dafner *et al.* (Single Image-based SFC)

Given an image, [18] represents it as an undirected graph $\mathcal{G}$ whose nodes are pixel locations, and each pixel is connected to its 8 neighboring pixels by an edge. Generating a context-based SFC from the given image is then equivalent to finding a Hamiltonian path in graph $\mathcal{G}$. They use the cover and merge algorithm (initially proposed by Matias *et al.* [25] to scramble a video signal for secure transmission). As the name suggests, the algorithm works by finding a Hamiltonian

path for the image-grid graph $\mathcal{G}$ in two steps - **cover** and **merge**.

In the **cover** step, a dual undirected graph $\mathcal{G}'$ is constructed from $\mathcal{G}$. The vertices of $\mathcal{G}'$ are small disjoint square circuits covering the whole of $\mathcal{G}$ as shown in Figure 2.2a. Each circuit covers 4 pixels. We call these initial circuits $C_1, C_2, \dots, C_k$ and connect (C_i, C_j) if the circuits C_i and C_j are adjacent in the original graph $\mathcal{G}$. Figure 2.2b shows a dual graph example built for an 8×8 image.

In the **merge** step, all circuits are merged to form a single Hamiltonian circuit. To merge the circuits optimally, a weight is assigned to each edge (C_i, C_j) in $\mathcal{G}'$ representing the “cost” of merging circuits C_i and C_j . The weight $w(C_i, C_j)$ of the edge connecting circuits C_i and C_j is defined as the cost of exchanging the edges e and f with the edges u and w in the image graph:

$$w(C_i, C_j) = |u| + |w| - |e| - |f|, \quad (2.1)$$

where $|\cdot|$ corresponds to the absolute difference in pixel values at the two vertices of the edge in $\mathcal{G}$. A minimum spanning tree $\mathcal{T}$ is then constructed using these weights. Figure 2.2c shows what $\mathcal{T}$ might look like for $\mathcal{G}'$. Next, we start merging circuits that are part of $\mathcal{T}$. Merging two circuits corresponds to removing their adjacent edges in $\mathcal{G}$ (e.g., edge e and f in Figure 2.2a), and creating new edges (e.g., edge u and w in Figure 2.2a). Note that only adjacent circuit pairs can be merged. Given the spanning tree, all merging operations on $\mathcal{G}$ can be done in a linear time to obtain a Hamiltonian circuit as in Figure 2.2d. Finally, the SFC or Hamiltonian path can be obtained by cutting the circuit at an arbitrary point.

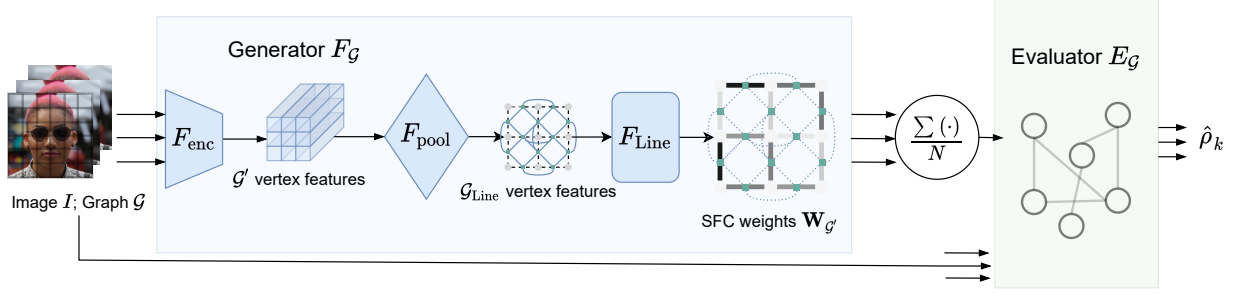


Figure 2.3: **Neural SFC pipeline.** The Neural SFC model consists of a weight generator F_G and a weight evaluator E_G . Taking one or more images as input, F_G extracts the deep features of the image and generates the SFC weights. E_G takes both the image(s) and the corresponding SFC weights as input E_G then estimates the negative autocorrelation of the image pixel sequence inferred by the input weights to evaluate the goodness of the input weights with respect to the input image. See more details in section 2.3.2.

2.3.2 Neural Space-filling Curves

Dafner’s approach [18] is effective at exploiting the local relationships in an image. However, it has a few limitations.

- Since the edge weights are computed using only two adjacent pixels, the receptive field is limited, and the resulting SFC does not take into account the long-range context in the image.
- Dafner’s approach only works for one image at a time. The notion of context-based SFCs can be further generalized to finding an SFC for a set of images.
- In the current form, [18] cannot optimize for arbitrary objective functions. The context-based SFC obtained is closely tied to edge weights defined by Eq. 2.1 which encourage autocorrelation with lag-2 in the pixel sequence obtained.

We address each of these issues in our data-driven approach to infer an optimal SFC for a set of images for any objective function. For brevity, we will optimize for lag- k autocorrela-

tion, however, in our experiments, we will show how we can modify this objective and directly optimize for a downstream application such as compression.

Setup. For the remainder of this section, we use the following notation. We are given a set of N images. Each image (color or grayscale) has a resolution of $H \times W$. For a given image I , we define graph $\mathcal{G}$ over its HW pixel locations. The dual graph $\mathcal{G}'$ consists of 2×2 disjoint circuits covering all the vertices of $\mathcal{G}$ as defined in the previous section. We first note that the whole cover and merge algorithm is context-agnostic barring the step where we assign weights to the edges of the dual graph. Therefore, a context-based SFC can be completely parameterized by these weights, denoted by $\mathbf{W}_{\mathcal{G}'}$. For a given set of weights of image I , the merge operation provides us with a Hamiltonian circuit. A Hamiltonian path, or an SFC, can be obtained by breaking the merged Hamiltonian circuit at any point. Hence, the problem of finding an SFC can be reduced to finding the optimal weights $\mathbf{W}_{\mathcal{G}'}$. We propose to learn a neural network, $F_{\mathcal{G}}$, to approximate $\mathbf{W}_{\mathcal{G}'}$. Once we have the optimized weights, the merge operation is fast and efficient in terms of both memory and speed, and we exploit it to get our desired SFC.

2.3.3 Weight Generator

The weight generator $F_{\mathcal{G}}$ is designed to take as input a single image I , and output the edge weights $\mathbf{W}_{\mathcal{G}'}$ of its dual graph, *i.e.*, $\mathbf{W}_{\mathcal{G}'} = F_{\mathcal{G}}(I)$. While the input dimension is $H \times W$, the output $\mathbf{W}_{\mathcal{G}'}$, has a size equal to the number of edges in the dual graph $\mathcal{G}'$. It is trivial to show that for the dual graph $\mathcal{G}'$ consisting of $\frac{H}{2} \times \frac{W}{2}$ vertices (or circuits), the corresponding number of edges will be $\frac{HW-H-W}{2}$. Further, the edges of $\mathcal{G}'$ do not conform to a 2D grid structure as the

input image I . To address this, we decompose the weight generator $F_{\mathcal{G}}$ further in three modules.

$$\mathbf{W}_{\mathcal{G}'} = F_{\mathcal{G}}(I) = F_{\text{Line}} \circ F_{\text{pool}} \circ F_{\text{enc}}(I). \quad (2.2)$$

Figure 2.3 shows the architecture of the weight generator. The first submodule is a dual graph encoder F_{enc} , which takes I as the input and extracts a deep representation of the vertices of dual graph $\mathcal{G}'$, resulting in a $\frac{H}{2} \times \frac{W}{2} \times d$ dimensional output, where d is the number of output feature maps. In this work, F_{enc} is implemented using a fully convolutional neural network with residual connections.

The second submodule F_{pool} consists of two pooling filters (and no trainable weights) of dimension 1×2 and 2×1 applied sequentially. F_{pool} imitates the graph operations by aggregating the features of vertices, and hence forming edge features of $\mathcal{G}'$. Given a d -dimensional representation of edges, we want to compute a scalar weight for each edge in $\mathcal{G}'$. It is desirable that this scalar weight can exploit not only the edge features, but also long-range relationships among the edges. Hence we construct a Line graph [40] $\mathcal{G}_{\text{Line}}$ to represent the adjacency between edges of $\mathcal{G}'$. Each vertex in $\mathcal{G}_{\text{Line}}$ corresponds to an edge in $\mathcal{G}'$. For every two edges in $\mathcal{G}'$ that have a vertex in common, there is an edge between their corresponding vertices in $\mathcal{G}_{\text{Line}}$.

Using the Line graph, the edge features of $\mathcal{G}'$ becomes the vertex feature of $\mathcal{G}_{\text{Line}}$, and the adjacent relations of edges of $\mathcal{G}'$ are represented by edges of $\mathcal{G}_{\text{Line}}$. To compute the scalar weights on $\mathcal{G}_{\text{Line}}$, we introduce the third submodule, a weights regressor F_{Line} to run on $\mathcal{G}_{\text{Line}}$. F_{Line} can be implemented using Graph Neural Network modules. In this work, we use GCN [41] in MNIST experiments and GAT [42] in FFHQ Faces experiments.

2.3.4 Objective Functions

The weight generator described above can generate edge weights for a given image. For every mini-batch of images, we take an expected value of the weights for all the images to get $\overline{W}_G$. Given $\overline{W}_G$, we can compute an SFC for the mini-batch of images. The quality or ‘goodness’ of an SFC can be different for different applications. In this chapter, we consider two plausible objectives: the 1D autocorrelation and the LZW sequence length. For both of them, the first step is to flatten the given image I to the 1D pixel sequence $\{y_i\}$ based on the SFC defined by $\overline{W}_G$.

Autocorrelation. The 1D autocorrelation measures the internal local similarity of a 1D sequence. Therefore, the smaller the 1D autocorrelation is, the better the SFC is. The lag- k 1D autocorrelation of a pixel sequence $\{y_i\}$ of length HW is defined as

$$\rho_k = \frac{\sum_{i=1}^{HW-k} y_i y_{i+k}}{\sum_{i=1}^{HW} y_i^2}. \quad (2.3)$$

Code Length. Another SFC quality metric, the LZW sequence length, is inspired from the Lempel-Ziv Welch (LZW) encoding [43, 44], which is popularly used to encode GIFs losslessly. Its performance depends on the amount of redundant data in a given sequence. Given a pixel sequence $\{y_i\}$, its LZW length is defined as

$$L = \text{length}(\text{Encode}(\{y_i\})), \quad (2.4)$$

where Encode is the LZW-encoding function, and length measures the length of a sequence.

Note that computing ρ_k or L requires us to first obtain a minimum spanning tree to infer

the SFC from I and $\overline{W}_G$, which is a non-differentiable operation. Therefore, these metrics, by themselves, cannot be directly used to optimize a neural network. Hence we can't simply backpropagate gradient information to update F_G .

To overcome the problem of non-differentiability of SFC computation, we train an evaluator neural network, E_G , as a differentiable proxy to estimate the resulting autocorrelation of SFC weights computed by F_G from the context image(s). By carefully designing the training procedure, our model E_G is able to approximate any non-differentiable metric, hence serving as an effective loss function of the weight generator. Figure 2.3 summarizes our approach.

Also note that, while we refer to the lag- k autocorrelation and the LZW length as our objective functions, we can replace them with any loss (or reward) suitable for a given application, even if it is not differentiable.

2.3.5 Weight Evaluator

The weight evaluator, denoted by E_G , acts as a differentiable proxy to estimate the objective Φ . Depending on the task, we choose $\Phi = -\rho_k$ for minimizing the negative autocorrelation or $\Phi = L$ for minimizing LZW length.

Given the average SFC weights $\overline{W}_G$, and an image I , we compute

$$\hat{\Phi} = E_G(\overline{W}_G, I), \quad (2.5)$$

$$\mathcal{L}_E = \mathbb{E} \left[\|\Phi - \hat{\Phi}\| \right], \quad (2.6)$$

where $\mathcal{L}_E$ denotes the expected value of l_2 error between groundtruth lag- k autocorrelation and the predicted autocorrelation by E_G computed for the entire batch. $\mathcal{L}_E$ serves as the objective

function for training the evaluator.

The inputs to the weight evaluator $E_{\mathcal{G}}$ are $\overline{W}_{\mathcal{G}}$, and I . Similar to section 2.3.3, we again decompose the weight evaluator $E_{\mathcal{G}}$ into submodules,

$$\hat{\Phi} = E_{\mathcal{G}}(\overline{W}_{\mathcal{G}}, I) = E_{\text{Line}}(\overline{W}_{\mathcal{G}}, \| E_{\text{pool}} \circ E_{\text{enc}}(I)), \quad (2.7)$$

where $\|$ denotes the concatenation along the feature dimension. Following the graph encoding procedure of the weight generator $F_{\mathcal{G}}$, E_{pool} and E_{enc} are functionally identical to F_{pool} and F_{enc} , respectively. The final submodule E_{Line} , which takes similar input to F_{Line} , has a different head to predict the estimated objective value $\hat{\Phi}$. The backbone of E_{Line} is also implemented using a GCN or GAT, followed by an average pooling operation and a simple MLP to predict $\hat{\Phi}$ as a single value.

2.3.6 Training

We adopt an alternating optimization procedure for training the core components of our architecture $F_{\mathcal{G}}$ and $E_{\mathcal{G}}$. Algorithm 1 gives an overview of the training schema. The weight evaluator $E_{\mathcal{G}}$ solves the regression task of computing the estimated objective $\hat{\Phi}$ for a given set of SFC weights $\overline{W}_{\mathcal{G}}$. Given the context image I and SFC weights $\overline{W}_{\mathcal{G}}$, we can get the groundtruth objective by first running Prim’s algorithm [45] to get the desired SFC followed by Eq. 2.3 or Eq. 2.4.

Once we have the groundtruth Φ , we optimize a standard L2 loss commonly used in regression methods to train $E_{\mathcal{G}}$, as we described in Eq. 2.6. Since we eventually need to use $E_{\mathcal{G}}$ to train the upstream network F , we would like $E_{\mathcal{G}}$ to be trained on a diverse range of input SFC weights

Algorithm 1: Training Neural SFC

Data: A set of N images each of resolution $H \times W$

Result: SFC weights $\overline{\mathbf{W}}_{\mathcal{G}}$ for the image set

```
1 Randomly initialize  $F_{\mathcal{G}}$  and  $E_{\mathcal{G}}$ ;  
2 repeat  
    // training  $F_{\mathcal{G}}$   
3     Sample a minibatch of  $B$  images;  
4     Forward pass for the weight generator  $F_{\mathcal{G}}$ ,  $\mathbf{W}_{\mathcal{G}'} \leftarrow F_{\mathcal{G}}(I)$ ;  
5     Expected weights for the mini-batch  $\overline{\mathbf{W}}_{\mathcal{G}'} \leftarrow \mathbb{E}(\mathbf{W}_{\mathcal{G}'});$   
6     Forward pass for the weight evaluator  $\hat{\Phi} \leftarrow E_{\mathcal{G}}(\overline{\mathbf{W}}_{\mathcal{G}'}, I)$ ;  
7     SGD update for  $F_{\mathcal{G}}$  keeping  $E_{\mathcal{G}}$  fixed,  $\nabla_{F_{\mathcal{G}}} \mathbb{E} [\hat{\Phi}]$ ;  
  
    // training  $E_{\mathcal{G}}$   
8     For each example in  $B$ , with a probability  $p_1$ , get  $\mathbf{W}_{\mathcal{G}'}$  using Eq. 2.1, with a  
        probability  $p_2$ , sample  $\mathbf{W}_{\mathcal{G}'} \sim \mathcal{N}(0, 1)$  and with a probability  $1 - p_1 - p_2$ , keep  
         $\mathbf{W}_{\mathcal{G}'} = F_{\mathcal{G}}(I)$ ;  
9     Run a forward pass for the weight evaluator  $\hat{\Phi} \leftarrow E_{\mathcal{G}}(\overline{\mathbf{W}}_{\mathcal{G}'}, I)$ ;  
10    For the whole batch, compute the ground truth  $\Phi$  using Eq. 2.3 or Eq. 2.4;  
11    SGD update for  $E_{\mathcal{G}}$  with  $\nabla_{E_{\mathcal{G}}} \mathcal{L}_E$ ;  
12 until  $\mathcal{L}_E \rightarrow 0$ ;
```

$\mathbf{W}_{\mathcal{G}'}$.

The $F_{\mathcal{G}}$ can be trained trivially using a fixed $E_{\mathcal{G}}$. We empirically observed that training them alternately improves the training dynamics thus boosting the SFC quality.

2.4 Experiments

In this section, we evaluate the ability of the proposed training scheme to generate optimized Space-filling Curves (SFC) for a set of images. We further validate the efficacy of the Neural SFCs on real-world applications such as image or gif compression. We compare with standard Raster scan and Hilbert curves. Note that even though Raster scan is not an SFC mathematically speaking, we use it for benchmarking in our experiments due to its prevalence.

2.4.1 Datasets

We trained the Neural SFC model on four different datasets. Both **MNIST** [46] and **Fashion-MNIST** [47] comprise of 60000 training images, and 10000 test images. Each image is a 28×28 grayscale image, which we resize to 32×32 to do a fair comparison with Hilbert curves which can be defined only when the image resolution is a power of 2. Resizing is done by a simple zero padding around the image. We observe a lot of intra-class similarity in the case of both MNIST and Fashion-MNIST, *i.e.*, the images within the same class are similar in layout and content to each other, and hence we train a single SFC for each MNIST and Fashion-MNIST class.

We also consider **FFHQ** [48] dataset. We downsample all the images to size 32×32 using bilinear interpolation. FFHQ is a dataset of celebrity faces and contains less noise compared to datasets like CelebA [49]. We split the dataset into 60000 training and 10000 test images. We train a single SFC for all the images in the data.

Lastly, in order to demonstrate a real-world application of SFCs designed for a set of images, we use a large-scale GIF dataset **TGIF** [50]. The dataset consists of 80,000 training gifs, and 11,360 test gifs. We train a single Neural SFC model that takes a gif as an input and outputs an optimized SFC for the gif. We evaluate it on every gif in the test dataset. Average numbers are reported.

2.4.2 Training details

We consider two different objective functions for training Neural SFC. First, in order to compare our method with Dafner *et al.* [18], we train our model using the autocorrelation objec-

tive function. Since Dafner’s method cannot generate SFCs for an image-set trivially, we compute Dafner’s image-set SFC by taking the expected value of $\mathbf{W}_{\mathcal{G}}$, defined in their method for all the images in the training set. Another possible choice is to calculate an average image for the entire image set, then run Dafner’s method on it to generate an image-set SFC. We use the first setting in all experiments in our main paper because it empirically performs better. In all our experiments, we set $k = 6$, such that lag-6 autocorrelation is used to train the weight evaluator $E_{\mathcal{G}}$.

Second, we also provide quantitative results on the compression of images and gifs using a Lempel-Ziv encoder (LZW) encoding scheme. Specifically, we optimize the SFC separately for each gif, although, it is possible to train a large SFC encoder on the entire gif dataset. We leave the study for evaluating context-based SFC for large- gifs or video datasets for future work.

2.4.3 Qualitative Evaluation

Visualizing the 1D sequence. In Figure 2.4, we show a few examples to illustrate the difference between Neural SFCs and Hilbert Curve on the MNIST, Fashion-MNIST, and FFHQ images. We look at the SFC (red curve overlayed on the image of the digits/clothing/faces). While the Hilbert Curve outputs the same SFC for any 32×32 image in the world, Neural SFC optimizes the SFC for a given set of images. To see this difference, we flatten the SFCs into a 1D sequence (images on the right of the digits/clothing/faces), and observe that Neural SFCs tend to keep better long-range spatial coherence than Hilbert Curve. In both cases, Neural SFCs show pixels in fewer clusters as compared to Hilbert curves. Specifically, Neural SFCs are able to roughly stay in the bright regions until they all get covered. Therefore, bright pixels will mostly gather in one contiguous segment in the 1D sequence inferred from a Neural SFC. In contrast, Hilbert

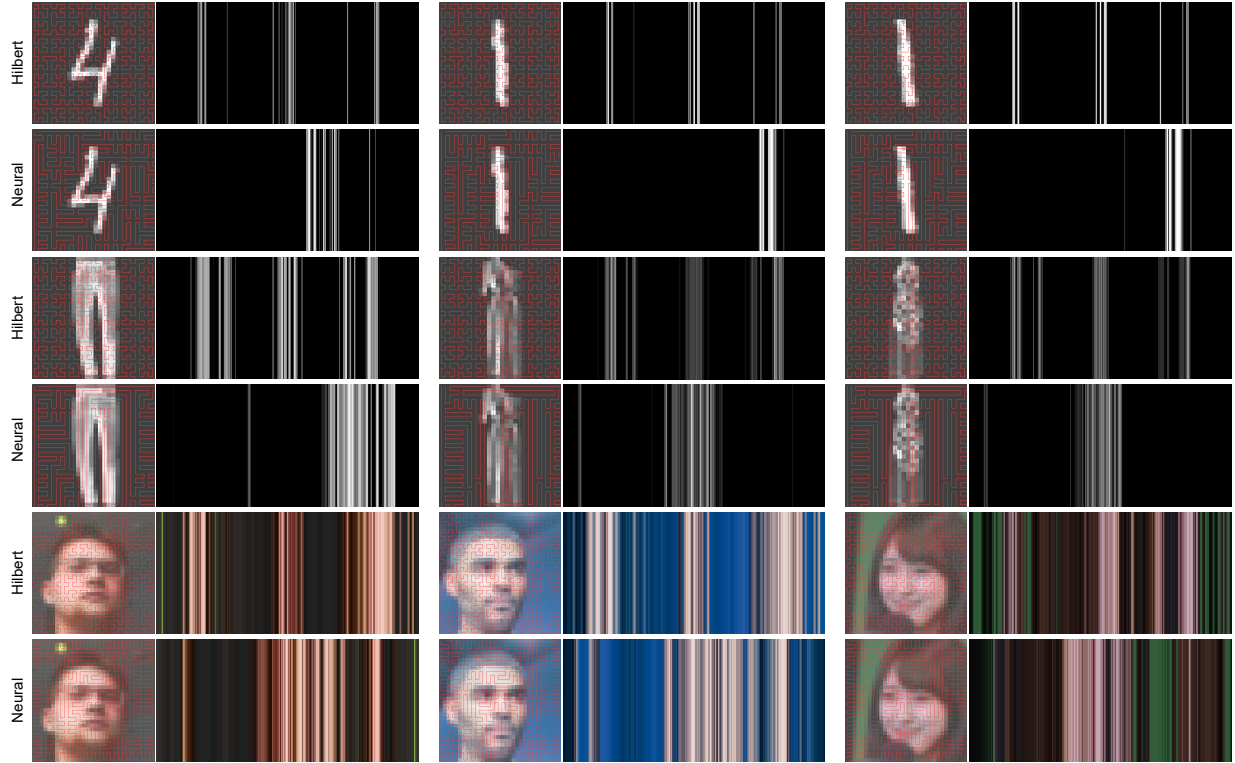


Figure 2.4: Qualitative comparison between Hilbert curves and Neural SFCs. Left: SFC (in red color) overlaid on the image. Right: Image flattened according to the SFC and visualized in 1-dimension. Images in the top two rows are from MNIST, the ones in the middle two rows are from Fashion-MNIST, and the ones in the bottom two rows are from FFHQ Faces. Neural SFCs on images from MNIST and Fashion-MNIST are class-conditional, *i.e.*, computed for each class. Therefore, for MNIST and Fashion-MNIST, Neural SFCs in the right two columns are the same since the two images have the same class label. In all datasets, Neural SFCs are more spatially coherent and produce fewer clusters when visualized in 1-dimension. Best viewed in color.

curves often result in multiple clusters of contiguous structures in the 1D sequence. We provide more examples in the supplementary material.

SFCs obtained with different objectives. Fig. 2.5 shows two different SFCs obtained by our approach when trained with different objective functions. The figure on the left corresponds to SFC optimized for LZW encoding length, while the figure on the right corresponds to SFC optimized for auto-correlation. We observe that generally, SFCs optimized for LZW encoding length are better at short-lag auto-correlation (as compared to SFCs optimized for lag-6 as in

most of our experiments). This results in an SFC with fewer turns and straighter paths.

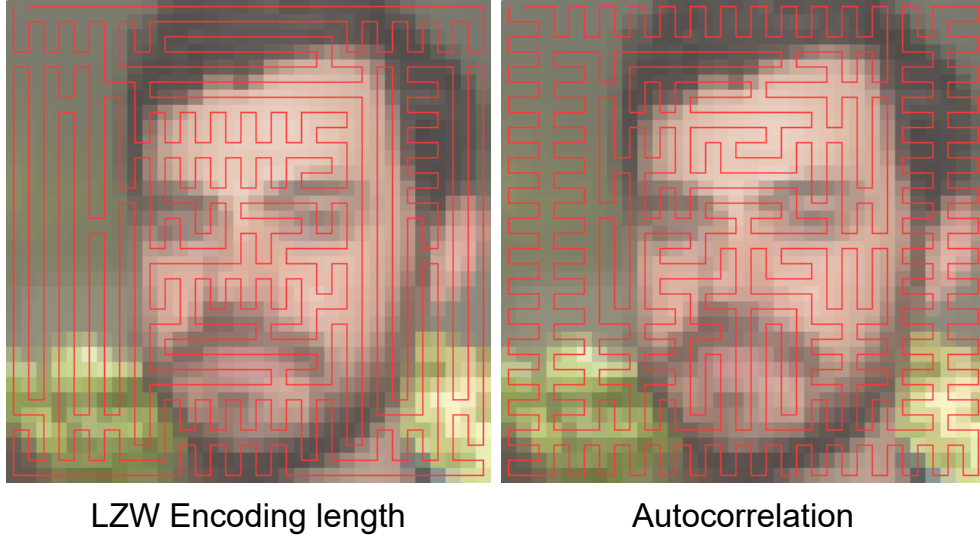


Figure 2.5: We visualize the Neural SFCs training with two objectives considered in this chapter – autocorrelation and LZW encoding.

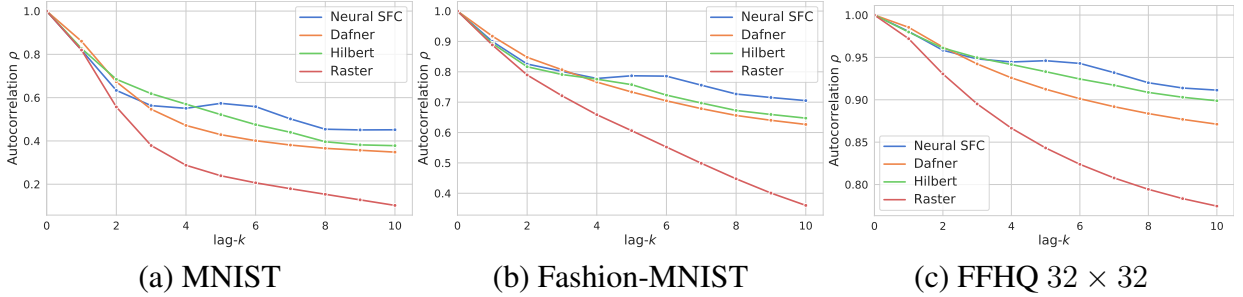


Figure 2.6: lag- k autocorrelation for MNIST, Fashion-MNIST and FFHQ datasets. While Dafner *et al.* provide higher autocorrelation for small lag, *i.e.*, from $k = 2$ to $k = 4$, Neural SFCs outperform Dafner *et al.* for $k > 4$ in all the datasets. Note that we trained our model for $k = 6$, and hence this behaviour is expected.

2.4.4 Optimizing Autocorrelation

To quantitatively evaluate the generated SFCs, we plot lag- k autocorrelations of pixel sequences obtained from test sets of three different datasets MNIST, Fashion-MNIST, and FFHQ.

Note that even though, we trained our models only to optimize the lag-6 autocorrelation, we plot

them for a range of values of lag- k . From Figure 2.6, we first observe that the trend is somewhat consistent across multiple datasets, even though they are very different in nature. Neural SFC performs the same or slightly worse than other SFCs at lower values of k . However, for $k > 4$, it outperforms other SFCs by a wider margin. This is intuitive since our model was optimized to increase the autocorrelation for a large value of lag. This also reflects our model’s ability to capture long-range and global information. We believe that higher gains can be obtained if we train and test using the same k value.

In the second and the third columns of Table 2.1, we show the autocorrelation values for MNIST, Fashion-MNIST, and FFHQ. We observe that the performance of Dafner’s SFCs is even worse than the Hilbert Curve in most cases. This indicates that the naive average of several good SFCs may not be a good way to compute the SFC for a set of images.

2.4.5 Optimizing Code Length

In this section, we study how Neural SFC can improve image compression results. Specifically, we use the LZW length objective to optimize E_g . This means we set $\Phi = L$ in Algorithm 1, where L is defined in Eq. 2.4. We evaluate our model’s performance on all 4 datasets. On MNIST and Fashion-MNIST, Neural SFC model is trained for each class label. FFHQ has no labels, therefore all images are used together to learn a Neural SFC. On the TGIF dataset, Neural SFC model is trained on all gifs but generates a unique SFC for each gif, *i.e.*, all frames in a gif share the same Neural SFC.

The last column of Table 2.1 shows the average file size required to store an image in the test data and its relative improvement compared to the Raster scan. Specifically, in the TGIF

Table 2.1: Comparison of performance of lag- k autocorrelation and LZW Encoding length for different orders. For autocorrelation, we consistently outperform both the universal and context-based SFC computation approaches at high values of k . For LZW Encoding length, we measure the average size per frame in bytes as well as the relative improvement compared to the raster scan order, in the case of each of the datasets. We consistently outperform compression performance for other order schemes.

Dataset	Method	$\rho_6 \uparrow$	$\rho_{10} \uparrow$	Size in bytes (Δ) $\downarrow$
MNIST	Raster	0.206	0.102	175.4
	Hilbert	0.475	0.378	182.7 (+7.3)
	Dafner [18]	0.401	0.348	-
	Neural SFC (Ours)	0.558	0.451	171.1 (-4.3)
FMNIST	Raster	0.552	0.360	425.8
	Hilbert	0.723	0.647	427.3(+1.5)
	Dafner [18]	0.704	0.627	-
	Neural SFC (Ours)	0.786	0.705	412.4 (-13.4)
FFHQ	Raster	0.824	0.775	688.0
	Hilbert	0.924	0.899	689.6(+1.6)
	Dafner [18]	0.901	0.871	-
	Neural SFC (Ours)	0.943	0.911	678.3 (-9.7)
TGIF	Raster	-	-	563.9
	Hilbert	-	-	567.0 (+3.1)
	Neural SFC (Ours)	-	-	556.9 (-7.0)

section, the number represents the average size to save a single frame instead of the entire gif file. It is worth noting that this size does not include the order itself. One order can be shared by many images, thus the cost of it can be amortized. We observe that the Neural SFC results in smaller a sequence size on all 4 datasets, showing consistent improvement on the Raster scan or Hilbert Curve. Interestingly, as compared to the Raster scan, even though Hilbert Curve results in higher autocorrelation (see Figure 2.6), it is always worse in terms of LZW encoding length. This finding suggests that there is no simple relation between an order’s autocorrelation performance and LZW encoding length performance.

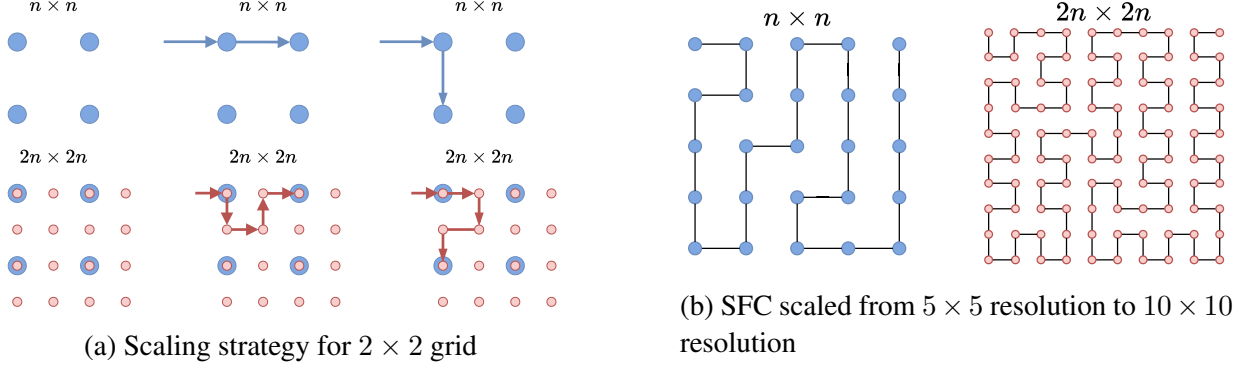


Figure 2.7: **Scaling up SFC.** The top row (blue circles) shows a 2×2 crop of an image grid of resolution $n \times n$. The bottom row (red circles) shows how we can scale the SFC path from $n \times n$ grid to $2n \times 2n$ grid. For every incoming SFC to a pixel location, there are 3 possible ‘next’ pixels, straight ahead, left, or right. Column 2 and 3 consider the cases where the SFC goes ‘straight’ or ‘right’. The image on the right shows a toy example of scaling up SFC for a 5×5 image.

2.4.6 Scaling up SFC

Although in the current work, we only train Neural SFC models for 32×32 images to demonstrate their effectiveness, there are no theoretical restrictions of our approach for higher resolution images. It is trivial to either (1) train SFC for higher resolution itself or (2) scale an SFC computed for a low-resolution image to a high-resolution image, preserving the locality properties as shown in fig. 2.7.

2.5 Conclusions and Future Work

We propose Neural SFC, the first data-driven approach to finding a context-based SFC for a set of images. We parameterize the SFCs as a set of weights over a graph defined using an image grid, and train a neural network to generate the weights. Neural SFC can be trained for any objective function defined for a set of images, even when not differentiable. We show the performance of Neural SFC on four real-world datasets on two different objective functions - (1)

Pixel autocorrelations in the pixel sequences obtained from an image (2) Compressing a set of images or a short video such as a gif using LZW encoding.

While our work takes the first steps towards finding 1D sequences in 2D data, it opens up a number of directions for future research such as learning SFC for higher dimensional data such as 3D objects or in the case when the 2D space is in a latent space instead of pixels (VQ-VAE [51, 52], dVAE [53]). Applying Neural SFCs to large video compression tasks [54, 55] is also promising given their success on gifs. We will explore these exciting directions and more in future work.

Chapter 3: Generating Images: Exploring Novel Object-State Compositions

3.1 Introduction

Objects often exist in different shapes, colors, and textures in the real-world. These visually discernible properties of objects, also known as states or attributes, can be inherent to an object (*e.g.*, color) or be a result of an action (*e.g.*, chopped). Generalization to unseen properties of objects remains an Achilles heel of current data-driven recognition models (*e.g.*, deep networks) that assume robust training data available for exhaustive object properties. However, humans (and even animals) [56, 57] can innately imagine and recognize a large number of objects with varying properties, by composing a few known objects and their states. This ability to synthesize and recognize new combinations from finite concepts, called *compositional generalization* is often absent in modern deep learning models [58].

Several recent works have been proposed to study composition in terms of the disentanglement of objects and the states in images [5, 59–61] as well as videos [62–68]. A few works have attempted to improve open-world text-to-image generation models [69, 70] for the task of compositional generation. However, current suite of datasets lacks either granular annotations for object states or enough data to study how object states evolve under different conditions. Therefore, measuring the compositional generalizability of these models on different tasks remains an open challenge.

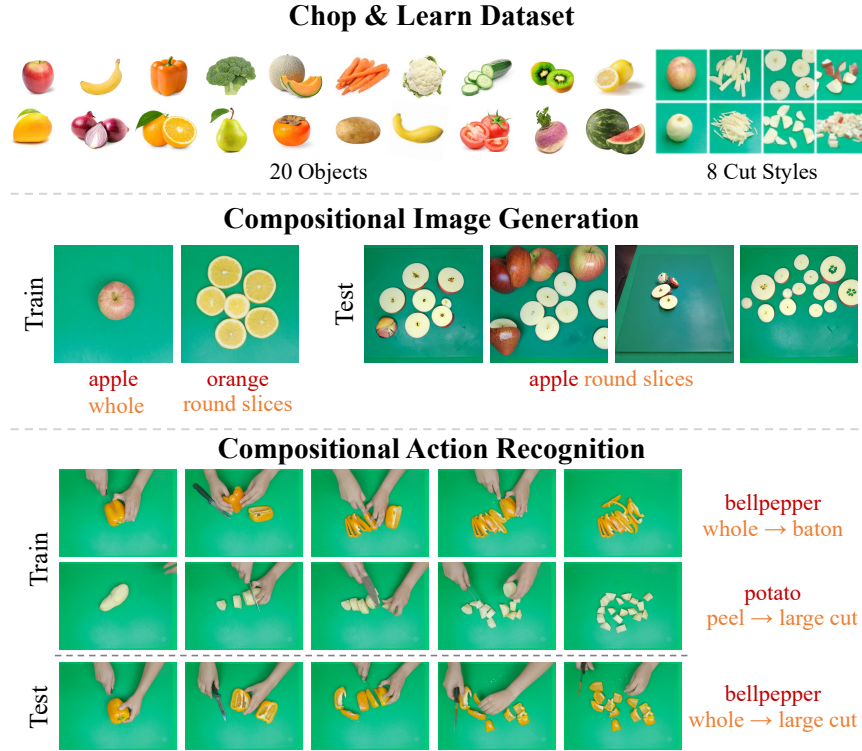


Figure 3.1: We present **Chop & Learn (ChopNLearn)**, a new dataset and benchmark suite for the tasks of Compositional Image Generation and Compositional Action Recognition. It consists of 1260 video clips and 112 object state combinations captured from multiple viewpoints for 20 objects and 8 cut styles. We also propose two new compositional tasks and benchmarks - (1) Image Generation: given training images of various **objects** in various **states**, the goal is to generate images of unseen combinations of **objects** and **states**. (2) Action Recognition: training videos are used to recognize **objects** along with transition from **state1** → **state2**, to generalize on recognizing unseen object-state transitions.

In this chapter, we propose a new dataset, **Chop & Learn (ChopNLearn)** collected to support studying compositional generalization, the ability to recognize and generate unseen compositions of objects in different states. To focus on the compositional aspect, we limit our study to a common task in our daily lives – cutting fruits and vegetables. When using different styles of cutting, these objects undergo different transformations and the resulting states are easily recognizable by humans. Our goal is to study how these different styles can be applied to a variety of objects for recognizing unseen object states. More specifically, we select *twenty* objects and *seven* commonly used styles of cuts (plus whole object) which results in object-state pairs with

different granularity and sizes (Figure 3.1). We collect videos of these objects being from *four* different viewpoints, and label different object states in each video. Each style of cut changes the visual appearance of different objects in different ways. To study and understand object appearance changes, we propose two new benchmark tasks of Compositional Image Generation and Compositional Action Recognition, with a focus on unseen compositions.

The objective of the first task is to generate an image based on an (object, state) composition that was not seen during training. As shown in Figure 3.1, during training, a generative model is provided with images of an (apple, whole) as well as an (orange, round slices). At the test time, the model has to synthesize a new unseen composition (apple, round slices). We propose to adapt large-scale text-to-image generative models for this task. Specifically, by using text prompts to represent the object-state composition, we benchmark several existing methods such as Textual Inversion [70] and DreamBooth [69]. We also propose a new method by introducing new tokens for objects and states and simultaneously fine-tuning language and diffusion models. Lastly, we discuss the challenges and limitations of prior works as well as the proposed generative model with an extensive evaluation.

In the second task, we extend an existing task of Compositional Action Recognition [77]. While the focus of prior work [77] is on long-term activity tracking in videos, we aim to recognize subtle changes in object states which is a crucial first step for activity recognition. By detecting the initial state and final object state compositions, our task allows the model to learn unseen object state changes robustly. We benchmark multiple recent baselines for video tasks on the ChopNLearn dataset.

Finally, we discuss various other applications and tasks that can use our dataset in image and video domains. To summarize, our contributions are threefold:

Table 3.1: **Comparison with other video datasets.** This table highlights the distribution of the objects, states and compositions in different datasets. Obj. refers to objects, Comp. is compositions of objects and styles, N refers to the number of compositions that have more than 10 samples, and Styles* refers to grouping of styles: instead of generic names like cut, chop, etc., we use 3 distinct styles (chop/dice, peel, grate) as styles. MIT-States[†] is the only image-based dataset, the rest are video-based datasets. All these data numbers are for edible objects and cutting style actions from respective datasets. Our dataset has uniform distribution for each metric in the table, which makes it suitable for learning objects and their states.

Datasets	Total # of				Avg. # of Samples			N	# of Views
	Samples	Obj.	Comp.	Styles*	/Obj.	/Comp.	/Style		
MIT-States [†] [71]	1676	27	52	4	62.07	32.23	419	48	1
Youcook2 [4]	714	160	313	3	7.3	2.2	166.7	26	1
VISOR [3]	301	58	122	3	5.2	2.5	42.9	3	1
COIN [72]	390	6	7	2	65	55	195	6	1
Ego4D [73]	216	12	12	3	18.2	18	54.5	8	1
50Salads [74]	904	5	6	2	182	152	457	6	1
ChangeIt [62]	264	8	14	4	46.3	26.4	96	14	1
CrossTask [75]	1150	7	8	2	164.3	143.7	575	8	1
Breakfast [76]	1055	3	4	2	351.7	263.8	527.5	4	1
ChopNLearn	1260	20	112	8	74.2	11.8	185.5	112	4

- We propose a new dataset ChopNLearn, consisting of a large number of images and videos of diverse object-state compositions with multiple camera views.
- We introduce the task of Compositional Image Generation, which goes beyond the common conditional image generation benchmarks, and focuses on generating images for unseen object and state compositions.
- We introduce a new benchmark for the task of Compositional Action Recognition, which aims at understanding and learning changes in object states over time and across different view-points.

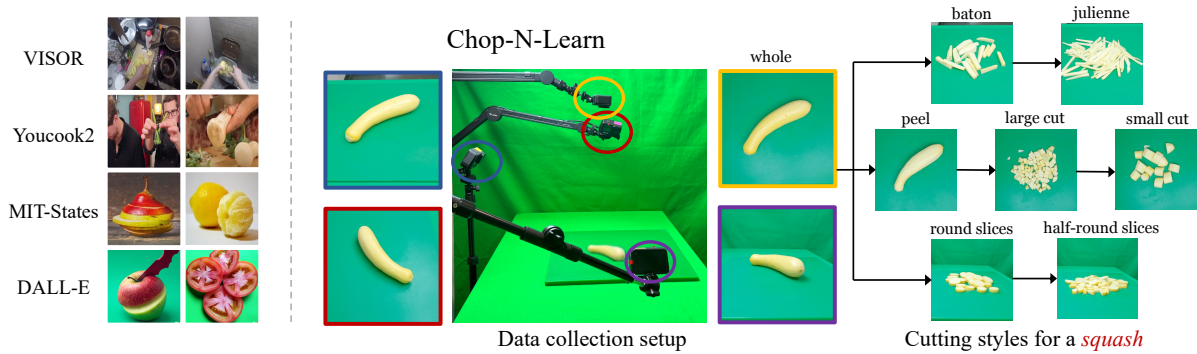


Figure 3.2: Left: We show examples of cutting styles from popular video datasets (VISOR [3]: chop and peel potato, Youcook2 [4]: chop broccoli, peel radish), image dataset (MIT-states [5]: slice pear, peel orange) and generation pipelines (DALL-E [6]: baton cut apple, half round slices tomato). Most of these are either too noisy to capture subtle differences in objects or do not have the granularity of specific cutting styles. Center: Our 4 camera setup captures videos of one object in 4 different views. Right: We capture 8 styles of object states, which can be derived in a hierarchical manner from larger to small cuts. Each style is of different shape and granularity.

3.2 Related Work

Object states or attributes have recently received significant attention for recognition tasks, in images and videos. Some of the common works and their dissimilarities with the proposed dataset are mentioned here.

Attributes of Objects. In the image domain, states are often referred to as attributes for Compositional Learning of attribute-object pairs. Attributes describe the visual properties of objects, such as shape, color, structure and texture. The common datasets used are MIT-states [5], UT-Zappos [59], COCO-attributes [78], CGQA [79] and VAW [80]. All of these datasets consist of web-scraped images of various types of objects (from furniture to shoes and clothes to food items), which makes the variety of states very diverse. Most of the prior works [60, 61, 79, 81–87] focus on attribute-object recognition tasks using compositional learning but do not expand to image generation tasks due to the diversity in background and attributes. Some works in

compositional zero-shot learning of attributes show visual disentanglement of attributes from objects [61, 88], however, they only hallucinate compositions of unseen attribute-object pairs in the feature space, rather than the image space. Moreover, even newer large vision-language models such as CLIP [89], DALL-E [6] fail to capture the subtle attributes of objects which are visually discernible [90, 91]. Therefore, the image generation task for objects with different attributes is still unexplored, which is a major focus of our work.

States for Action Recognition. Detecting object states and corresponding actions from videos is explored in supervised [64–67] and self-supervised manners [62, 63, 92]. While some works focus on recognizing actions using states [64–67], others discover states as the future frames in the videos in [92, 93]. Some works [62, 63] also detect the exact frames of state 1, state 2 and the action that causes transition from state 1 $\rightarrow$ 2. Another recent work (Ego4D [73]) also proposes new tasks like point-of-return state-change prediction for object state transition detection. Hence, object states so far have been used as a signal for detecting and localizing actions. We focus on extending this understanding of states to generalize across different objects with limited seen object-state transition videos.

Compositional Action Recognition. In contrast to randomly assigning samples for training and testing, [77] presented a new task of Compositional Action Recognition. The premise of this task is: actions are split based on objects they apply on. During training, only a set of objects are seen corresponding to set of objects, while during testing, unseen object appear for seen action labels. Following studies [94–98] used relationship between objects and states bounding boxes to model the compositional aspect, where the evaluation is performed on how well the composition of unseen object and state is recognized. We propose a similar task, where videos are trained on seen compositions and tested on unseen compositions.

Comparison with existing Datasets. The existing image datasets such as MIT-states [5], UT-Zappos [59], COCO-attributes [78], CGQA [79] and VAW [80], are not suitable for image generation tasks for two reasons: 1) there are very few transferable objects and attributes, 2) the images are web-scraped and very diverse with varied background. Due to this, generative models latch on background details rather than understanding subtle changes in objects. In video domain, there have been various video datasets with procedural and kitchen activities that capture object and state transformations, such as Epic-Kitchens [99] with object and hand bounding box annotation version VISOR [3], Youcook2 [4], Ego4D [73], COIN [72], HowTo100M [100], Breakfast [76], 50Salads [74], CrossTask [75] and ChangeIt [62]. There are a few common problems across these datasets: (1) Most of these datasets lack annotations for the granularity of cutting styles. The styles labeled are *cut*, *chop*, *slice*, *dice*, *peel*, *grate*, *julienne*, which only comprises of three broader styles of transformations, *i.e.* *chop/dice*, *peel* and *grate*. (2) The compositions of different objects and states are highly skewed and similar to image datasets. Some datasets have a long-tail distribution of objects, which can make it challenging for models to learn per-object-based states when there is only one sample available in the dataset. And lastly (3), the frames are noisy with lots of objects and attributes that object states changes are harder to capture (as shown in left side of Figure 3.2). For most datasets, the ground truth is also not annotated for object detection, which makes it even harder to look for object of interest. Using an object detector to remove the background is an option, however with deformable objects, most Faster-RCNN [101] based object detectors fail to capture the object itself, and latch onto smaller pieces instead. In Table 3.1, we show statistics of data available in different datasets. The # of clips from other datasets that has granular annotations of object-state pairs and can be used for compositional tasks. For instance, COIN [72] has 180 categories with 10000 videos, but clips

that have cutting styles as labels were only 390. Further, these clips only cover cut/peel actions, and cannot be categorized further based on granularity and shape of pieces. Our proposed dataset ChopNLearn is designed to capture various objects and their cut styles, with uniformly distributed samples for 20 objects and 8 styles (including whole, 7 other cut styles Figure 3.2).

3.3 Chop & Learn

Our main objective with Chop & Learn (ChopNLearn) is to understand and learn granular object states, specifically styles of cuts which can be applied to diverse variety of objects. With this in focus, we collect object state transition videos, as well as images of object in various states, with 4 different camera views (Figure 3.2). We discuss the design choices and motivation below.

3.3.1 Design Choices

Selection of States (styles of cuts). Fruits and vegetables are commonly cut in specific styles based on the need of the recipes. For instance, for eating an apple, we slice it in relatively large pieces while for using it in a pie, we might cut smaller or round slices of it. We select 8 common styles of cuts, *i.e.*, `large cut`, `small cut`, `baton`, `julienne`, `round slices`, `half round slices`, `peel`, and `whole` for our study. These are the most common styles of cuts for vegetables and fruits, which do not require any additional training to learn apart from common kitchen operation and knife handling skills. These styles of cuts can also have similarities with respect to shapes, yet are different in granularity. For example, `baton` (french-fries style cut) and `julienne` are similar in shape (long pieces), but `julienne` is more finely cut than `baton`. Similarly, `large cut` is a coarser version of `small cut`, and `half round`



Figure 3.3: **Statistics for ChopNLearn:** We show the number of samples for each object-style composition in a color-coded manner: orange represents 12 samples, green represents 8 samples and blue represents 4 samples.



Figure 3.4: **(a)** The clip length distribution for one camera (315 unique clips). **(b)** Preliminary results of using green screen to augment the dataset with different backgrounds. We continue to improve the transfer results by adding shadows and background matting.

`slice` is one step from `round slices` (as shown in Figure 3.2). We also have annotated the states `whole` and `peel`, which are the base states of objects.

Selection of Objects. We want to learn to transfer styles of cuts to different objects. To ensure consistency in transfer, we also consider the base state, *i.e.*, `whole` state of objects. For instance, it is hard to visualize `large cut` of carrots, if the seen data only includes rounder objects like `oranges`. Hence, we consider some fruits and vegetables with similar colors, textures and shapes to include consistency across visual similarities after chopping. In this study, we used seasonal fruits and vegetables categorised on the basis on their shapes, colors and textures: `round small objects`: [`apple`, `pear`, `mango`, `potato`, `turnip`, `onion`, `kiwi`], `citrus fruits` [`lemon`, `orange`], `flower-like textured objects`: [`cauliflower`, `broccoli`], `larger round objects`: [`cantaloupe`, `watermelon`], `textured from inside objects`: [`bellpepper`, `tomato`, `persimmon`], and `long objects`: [`cucumber`, `carrot`, `squash`, `banana`]. This consists of 10 fruits and 10 vegetable items, with at least one pair of similar objects presents in the dataset.

Related Groups. One of the key aspects of this dataset is transferability of cut styles to a variety of objects. We set up some constraints and create related groups for objects and styles. These related group enable us with structural and visual style transfer abilities. If an object is seen from related group A with a particular style, we should be able to transfer that style to another object from the same related group A and vice-versa. In other words, we group sets of objects and cut styles which are visually similar (based on color, shape and texture) together to create related groups for objects and states separately. For states, we combine [`baton`, `julienne`], [`round slices`, `half-round slices`], and [`large cut`, `small cut`] together as related groups. For objects, we define seven groups with related objects: [`apple`, `pear`, `mango`],

[lemon, orange], [cauliflower, broccoli], [cantaloupe, watermelon, kiwi], [bellpepper, tomato, persimmon], [potato, turnip, onion], and [cucumber, carrot, squash, banana].

3.3.2 Data Collection Setup

We collect data using four GoPro cameras [102] positioned at different angles, with three participants (Figure 3.2). We use a green screen and green chopping board for minimum distraction in the background, such that the objects and their cut pieces are easily segmented for each view.

Granularity of styles. For ease and consistency across participants, the size of cut pieces can be defined as the shape and ratio of one piece with respect to the whole object. For more details, please refer to the appendix. Given a set of n states and m objects, we can have at most $m \times n$ compositions. However, our dataset does not include some compositions which are not commonly found in real world. For instance, due to the texture of `onions`, it is not feasible to cut `onions` in `baton` or `julienne` style, since the layers of the `onion` do not stay intact, so we do not have a sample of `[baton, onion]`.

Video Recording. We primarily collect video data, and derive state change frames from long videos. Each video consists of 2-3 object states, which are annotated while data collection process using the highlight features of GoPros. For synchronizing across different cameras, we initially start with a clapper to make a clap sound for indicating the beginning of the video. Then, we highlight the frames in one of the GoPro as the first/initial state. The participant then walks up the object and starts cutting the object. After the object is cut in one style, the participant steps back and we highlight another frame as the next state. The participant performs at least 2 styles of

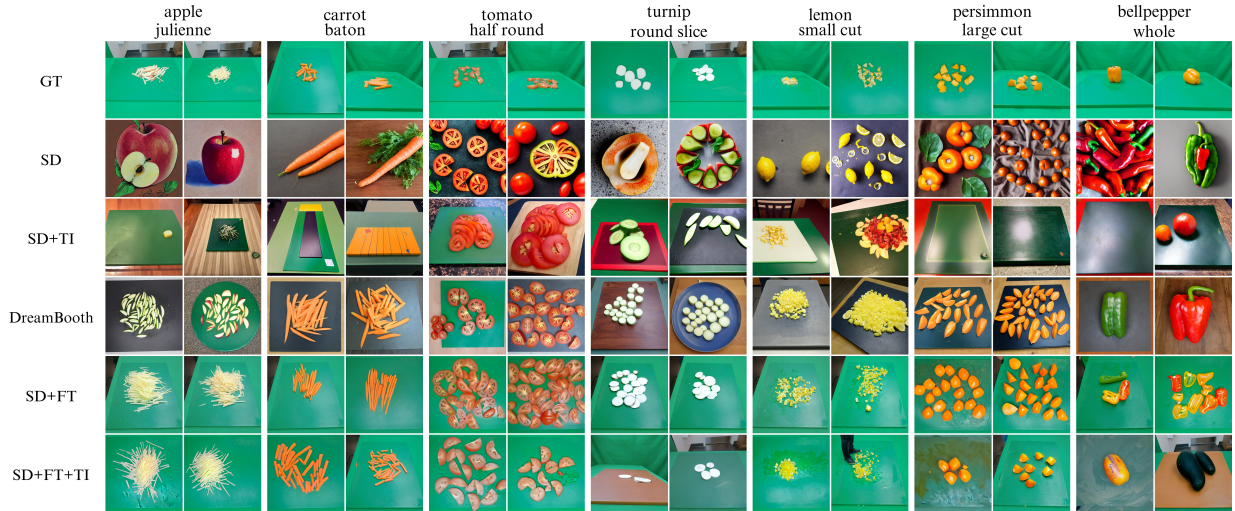


Figure 3.5: **Compositional Generation Samples.** Ground Truth (GT) real images are shown in the first row for reference. Seven object-state combinations in the test set are displayed, each with two generated samples for each method. Please zoom in to see details.

cut in each video, which can be done consecutively. For instance, we can first cut an object with `large cuts`, and then do `small cuts` subsequently. The video ends with another clap for the end of video detection and synchronization across different cameras. Henceforth, we collect video data along with annotated states for each participant, without extra effort of annotations. More details and statistics of dataset are shown in Figure 3.3. Average video clip length (one state change for an object) is 1m40s. The distribution is shown in Fig. 3.4(a).

3.4 Compositional Image Generation

Large-scale deep generative models [103–105] trained on open-world big datasets have made significant breakthroughs in image generation in the last couple of years. These models, are typically conditioned using a text encoder and also support tasks such as zero-shot image generation, inpainting, image editing, and super-resolution without explicit training on these tasks. However, the performance of these models significantly degrades when it comes to composi-

Table 3.2: **Compositional generation evaluation.** FID, user scores, and classifier scores of various generative models. User Realism is on a scale of 1-5. (★) denotes that accuracies are evaluated on a seen data split. **Bold** represents the best result.

Method	Patch	User	Classifier Acc. (%)		User Acc. (%)	
	FID ↓	Realism ↑	Object ↑	State ↑	Object ↑	State ↑
Real Images	-	4.65	87.5★	92.0★	73.6	84.0
SD	178.0	3.41	73.1	27.9	81.6	28.8
SD+TI	145.0	2.58	23.6	37.7	21.6	43.2
DreamBooth	139.9	3.56	53.5	74.2	61.6	72.8
SD+FT	88.9	3.78	70.5	67.7	72.0	65.6
SD+FT+TI	82.2	3.47	67.8	81.4	67.2	79.2

tional generation [106]. Our dataset, consisting of 112 real-world object and state combinations, is well-suited to test the compositional capabilities of generative models.

Task Description. The goal of the task is to either train from scratch or fine-tune an existing generative model using the (object, state) pairs provided in the training, and generate images from unseen compositions. We consider all 20 objects, each object captured in up to 7 different states, *i.e.*, all the states excluding `peel`. We split the (object, state) combinations into a training set consisting of 87 combinations and a test set consisting of 25 combinations. The training set covers all objects and states used in our dataset, but it does not overlap with the test set in terms of (object, state) combinations. In other words, for each combination of object and state present in the test set, the training set includes exactly one of either the object, or the state, but not both. We also ensure that for each (object, state) combination (o, s_i) in the test set, there exists a combination (o, s_j) in the training set, where s_i and s_j belong to the same state related group defined in Section 3.3.1. This setting ensures that all object and state information are available in the training set. Each combination in our dataset has 8-12 images, resulting in a total of 1032 images in the training set and 296 images in the test set. The exact split is provided in the appendix along with some examples.

3.4.1 Methods

Stable Diffusion. (SD) We evaluate a popular open-source text-to-image generative model Stable Diffusion (SD) [103]. For details on the SD, refer to the original work [103]. Here we briefly describe the sampling process. Diffusion models generate an image from Gaussian noise via an iterative denoising process. SD uses classifier-free guidance [107] for sampling. This means given a text prompt $\mathbf{c}$, we encode the prompt using CLIP’s text classifier [89] and recursively update a Gaussian noise sample with

$$\omega \epsilon_{\theta}(\mathbf{x}_t, \mathbf{c}) + (1 - \omega) \epsilon_{\theta}(\mathbf{x}_t) \quad (3.1)$$

where $\mathbf{x}_t$ is the denoised sample at the time step t and ϵ_{θ} is SD. With each time step, we try to move the denoised sample using the guidance provided by the text prompt. The strength of the guidance is defined by ω .

As our first baseline approach, we sample zero-shot images from SD with a text prompt “An image of o_i cut in s_j style”, where o_i is the i^{th} object and s_j is the j^{th} state of the object. Zero-shot generation with a pre-trained SD model doesn’t work as intended as shown in Figure 3.5, and the generated images often perform poorly in capturing the object state. Several recent works have shown that it is possible to extend models such as SD to achieve high-quality customized generations [69, 70, 108]. We evaluate several methods that have been proposed for compositional generation in the recent literature. We also propose a simple yet strong baseline by fine-tuning a Stable Diffusion (SD) model [103] along with textual inversion.

SD + Textual Inversion (TI). Textual Inversion [70] introduces new tokens in the vocabulary and

optimizes their embedding from the given images keeping SD frozen. We adapt the method for our task by introducing new tokens for the objects $\{o_i\}$ and the states $\{s_j\}$, and jointly optimize the embeddings of $\{o_i\} \cup \{s_j\}$ by providing (image, prompt) pairs from our training data. As before, the prompt is simply constructed as “An image of o_i cut in s_j style”.

DreamBooth. Next, we adapt DreamBooth [69], which fine-tunes the diffusion model along with the state-specific tokens. In our experiments, we fine-tune one model for each state in the dataset, where only the state token is learned. Original DreamBooth optimizes the diffusion loss as well as a prior preservation loss [69]. We observed that the latter significantly deteriorates the performance thus we skip it.

SD + Fine-tuning (FT). We also fine-tune SD. In this baseline, only the parameters in the UNet of the diffusion model are optimized while keeping the text encoder fixed. **SD + TI + FT.** Finally, we combine SD fine-tuning and Textual Inversion [70]. Specifically, on top of our SD + Fine-tuning baseline, we also adapt Textual Inversion by introducing new object tokens and state tokens and optimizing their embeddings along with the UNet parameters.

3.4.2 Evaluation

We use both qualitative and quantitative measures to evaluate the capabilities of different methods. This section explains the details of different evaluation metrics we used:

Patch FID. Fréchet Inception Distance (FID) [109] is a commonly used metric to assess the quality of generative models. Given a set of real images and a set of generated images, FID compares the mean and std of Inception-v3 features of the two sets. For each composition and generative model, we compute patch FID using all real and 16000 generated patches, and report the average number for the test pairs. We hypothesize that using patch FID gives more weight

to the object-state patches, rather than the whole image, which includes almost 50% background pixels. We further calculate the lower bound for patch FID score by computing it between two sets of real images. Any score lower than that for this dataset can be disregarded as irrelevant. The determined lower bound for the patch FID score is 37.2.

Object/State Accuracy using a Classifier. To evaluate the correctness of objects and states in the generated images, we train a classifier on real images for classifying objects and states independently. This classifier is built on top of CLIP-ViT-B/32 [89]. Classification logits are obtained by computing the cosine similarity between the image embedding and text embeddings of all possible state labels or object labels. To ensure the reliability of the classifier’s results, we train it on the training set from a different dataset split, where all (object, state) combinations are present.

User Study. We conducted a user study to evaluate the generated images. We took images from the test set as well as samples from our generative models and present them to 30 users. Each user was presented with 25 distinct images, randomly sampled with an even distribution from our models and the test set. After giving a tutorial to the users about the different objects and states present in our experiments, the users were asked to choose an appropriate object name and state label, as well as rate the image for realism on a scale of 1-5. We report the object and state accuracies as well as realism score in Table 3.2. The details of our user study design can be found in the appendix.

3.4.3 Results and Discussion

Qualitative Results. fig. 3.5 displays the generated images from various methods for seven (object, state) combinations in the test set. The first row of the figure exhibits the ground truth

Table 3.3: **Compositional action recognition results.** “Start/End” denote the prediction results for the initial and the final state composition with the corrected object type. **Bold** and underline represent the top-1 and top-2 results.

Model	Features	Split 1				Split 2				Split 3			
		Start		End		Start		End		Start		End	
		acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3	acc@1	acc@3
AvgPool	I3D [110]	9.5	23.7	4.7	14.2	8.3	21.9	5.2	19.8	15.9	28.5	4.8	22.3
LSTM [111]	I3D [110]	14.2	36.2	5.7	29.8	12.5	29.2	6.2	26.0	17.5	34.9	6.3	23.7
Transformer [112]	I3D [110]	23.7	49.0	10.9	44.3	27.5	46.2	14.6	44.2	20.6	42.9	11.1	44.4
AvgPool	MIL-NCE [113]	11.1	31.6	4.8	28.4	9.4	17.7	5.2	13.5	14.2	41.4	12.8	41.4
LSTM [111]	MIL-NCE [113]	15.9	36.5	6.4	36.6	11.9	36.7	9.8	36.7	18.9	39.6	8.0	25.4
Transformer [112]	MIL-NCE [113]	<u>50.9</u>	85.7	47.7	76.2	56.2	82.3	<u>52.7</u>	88.5	41.1	74.6	42.9	77.7
STLT [94]	–	2.8	15.5	1.4	8.4	1.4	13	1.4	11.6	4.2	14.1	1.4	11.3
Transformer [112]	R3D [114]	45.1	<u>85.9</u>	<u>52.1</u>	<u>85.9</u>	<u>55.1</u>	<u>94.2</u>	58.0	<u>92.8</u>	<u>59.1</u>	<u>85.9</u>	<u>56.3</u>	<u>85.9</u>
CAF [94]	R3D [114]	53.5	88.7	57.8	88.7	<u>55.1</u>	95.7	58.0	95.7	62.0	93.0	63.4	93.0

real images for reference. We observe that vanilla SD often generates correct objects in random states, while SD+TI frequently synthesizes images without displaying the object. DreamBooth performs better than SD+TI, but worse than a simple finetuning of SD. SD+FT and SD+FT+TI perform well in terms of state generation.

Quantitative Results. Table 3.2 displays the performance of all baseline methods evaluated according to the metrics outlined in Section 3.4.2. Assessing image realism is a crucial evaluation metric for generative models; however, defining and measuring it can be challenging. Note that the patch FID values and user realism ratings do not align well. This is due to the disparity between the distribution of images in our dataset and that of typical occurrence of those objects in the real world. The patch FID metric measures the similarity between the generated images with those in our dataset, instead of the ones most typical in real world. In particular, our results indicate that SD achieves the worst patch FID score since it has not encountered our dataset before, whereas its user realism rating is more satisfactory. SD+TI has the lowest user realism rating and a poor patch FID score, which suggests that only training object/state embeddings is inadequate for generating high-quality images. DreamBooth receives a good user realism rating

but a poor patch FID, indicating that the images it generates are realistic but not very similar to those in our proposed dataset. Finally, fine-tuning via both SD+FT and SD+FT+TI achieve better results for patch FID and user realism.

We next evaluate the accuracy of objects and states in generated images. It is worth noting that the classification task on our dataset is intrinsically difficult, which leads to imperfect user accuracy on real images. In general, the accuracy scores from classifier closely align with one from users, indicating that the proposed classifier is suited for evaluating compositional generation.

Our results show that SD achieves the best object accuracy but the worst state accuracy. This is possibly due to the lack of state variations in most existing large image datasets. SD+TI is the worst performer due to its limited learning capacity. On the other hand, DreamBooth, SD+FT, and SD+FT+TI attain better state accuracy. Among them, DreamBooth’s object accuracy is slightly worse as it is particularly trained for states. SD+FT achieves high object accuracy, and SD+FT+TI attains the best state accuracy with the help of fine-tuning and textual inversion together.

Green Screen Removal. One of the main challenges for understanding fine-grained object-state pairs with existing datasets such as MIT-states [71] is diverse backgrounds. Using them for training often leads to the model latching on to unwanted background details and missing out on the state understanding. Hence, we collected ChopNLearn with a clean green screen background for the benchmark tasks. While we acknowledge the limitations it poses to our trained models, we highlight that the green screen can potentially enhance our ability to generalize to diverse scenes. This can be achieved by segmenting out images and placing various backgrounds, along with scaled and rotated object-state images (Figure 3.4). As a proof-of-concept, we train a SD+FT+TI model on background-augmented images, and report the Patch FID, classifier object accuracy

and state accuracy in table 3.4. Note that here we employ a newly trained classifier that uses background-augmented images, and the patch FID scores are also computed based on these images. We further reference the lower bound of the patch FID as defined in Section 3.4.2. Due to the complex backgrounds introduced, the object accuracy and the patch FID of the new model are slightly compromised. However, it maintains a high and even improved state accuracy. This demonstrates the potential of the background-augmented ChopNLearn in enhancing fine-grained compositional image generation.

Table 3.4: **Green screen removal evaluation.** Both rows employ the SD+FT+TI but are trained using images with varying backgrounds. Classifiers specific to each dataset are trained to assess Classifier Acc. Validation images used to calculate Patch FID differ between the two rows. Patch FID Lower Bound is computed by evaluating the patch FID on one-half of the validation images relative to the other half. For further details, refer to Section 3.4.3.

Data Background	Classifier Acc. (%)		Patch FID ↓	Patch FID Lower Bound
	Object ↑	State ↑		
Green Screen	67.8	81.4	82.2	37.2
Various	46.3	82.3	133.6	46.4

3.5 Extension: Compositional Action Recognition

Human actions often change object states and different objects can have diverse visual transitions even when subjected to the same action type. To investigate this problem in a more intuitive manner, [77] introduced a new task of compositional action recognition, which targets at improving the robustness of models to handle the same actions with different objects involved. For example, given an action of ‘taking something out from something’, the model is trained on a limited set of objects and is tested on unseen types of objects to access its generalizability. Hence, despite the same underlying action, the object and visual features can be quite diverse. Similarly, the composition of the same action with different object types can look very distinctive. For

instance, although cutting an `carrot` and a `apple` require similar knife movements, the resulting visual changes are distinct, with the former changing from a whole `apple` to a peeled `apple`, and the latter changing from a whole `carrot` to a peeled `carrot`. Therefore, we propose to use our dataset for the task of compositional action recognition, which can also be referred to as Compositional Zero-Shot Action Recognition, as the compositions of objects and states are unseen during training.

Task Description. For this task, we consider each clip of a video as containing a single object with a single state transition. From the raw videos, which typically contain 2-3 transitions of object states per video, we segment the clips into isolated ones with only one transition. Examples of transitions include changing from a `whole` object to a `peeled` object or from a `peeled` object to a `baton cut` object. Similar to [77], we divide all object-final state compositions into two sets: seen compositions, which are used for training, and unseen compositions, which are used for testing. Following the approach used in the Compositional Image Generation task, we ensure that each object and state are seen at least once individually in the training set, but not together as a composition. The objective of the task is to predict the correct labels for the initial object-state composition (o_i, s_j) and the final composition (o_i, s_k) , given a clip containing an object o_i transitioning from an initial state s_j to a final state s_k . Note that the clip is considered correctly classified only if both the object and state labels are correct for both the initial and final compositions.

3.5.1 Dataset Splits

We create 3 different dataset splits as follows (more details are in the Appendix). All splits have disjoint train, test and validation samples, and are created with different constraint combinations:

- **Split 1:** This split is a random selection of object-final state compositions with cross-view condition. We do not use any information from related groups.
- **Split 2:** In this split, we use related group information for states, along with cross-view. based on related groups, if `baton carrots` is seen in training set, then `julienne carrots` can be part of test set. Since `baton` and `julienne` are part of the same related group, we can learn an object in one style and can generalize to another style from the same group in Section 3.3.1.
- **Split 3:** This split includes information from both related groups for states and objects. We want to ensure that even if an object is not seen in its related group, a similar object is seen in the related group. For example, if `broccoli` is seen with `large cuts`, then `cauliflower` with large or small cuts can be in the test set.

Hence different splits represent different complexity levels for compositional action recognition.

Evaluation. We evaluate the accuracy of predicting both the initial and final compositions of objects and states in the test set. Only when the object and state are both correct, it is counted as a correct prediction. Specifically, we use two separate prediction heads for objects and states. We emphasize the need to evaluate composition as a whole, rather than just predicting the state, as the way an `apple` is cut can differ significantly from the way a `bellpepper` is cut. There-

fore, accurately recognizing both the object and state is crucial for tasks related to understanding and generating videos of object states. We also recognize the importance of top@3 accuracy, since object states can sometimes be visually similar, leading to confusion in detecting the correct composition. For example, `julienne apple` can be visually very similar to `julienne potato`.

3.5.2 Results

To evaluate our proposed method, we establish baselines using both traditional architectures and features for video action classification, as well as comparing with recent works in compositional action recognition. As shown in Table 3.3, in the first section, we use pre-extracted I3D[110] features and conduct experiments by comparing simple average pooling, LSTM, and multi-layer Transformer [112] model. It shows that the Transformer model performs the best among these variants due to the great capacity of temporal modeling ability. In the second section, we also experiment with more recent pre-trained features MIL-NCE [113] along with transformer models, which outperforms I3D features. MIL-NCE [113] features are pre-trained on HowTo100M [100] with multimodal (RGB+narrations) setup, which is more robust for video downstream tasks.

In the final section of Table 3.3, we employ the state-of-the-art compositional video recognition model proposed in [94] and use pseudo labels of bounding boxes for each hand and object, as there are no ground-truth hand and object trajectories available. Specifically, the Spatial-Temporal Layout Transformer (STLT) [94] takes in the spatio-temporal locations and class labels for each bounding box as input, uses positional embeddings to project coordinates and class labels

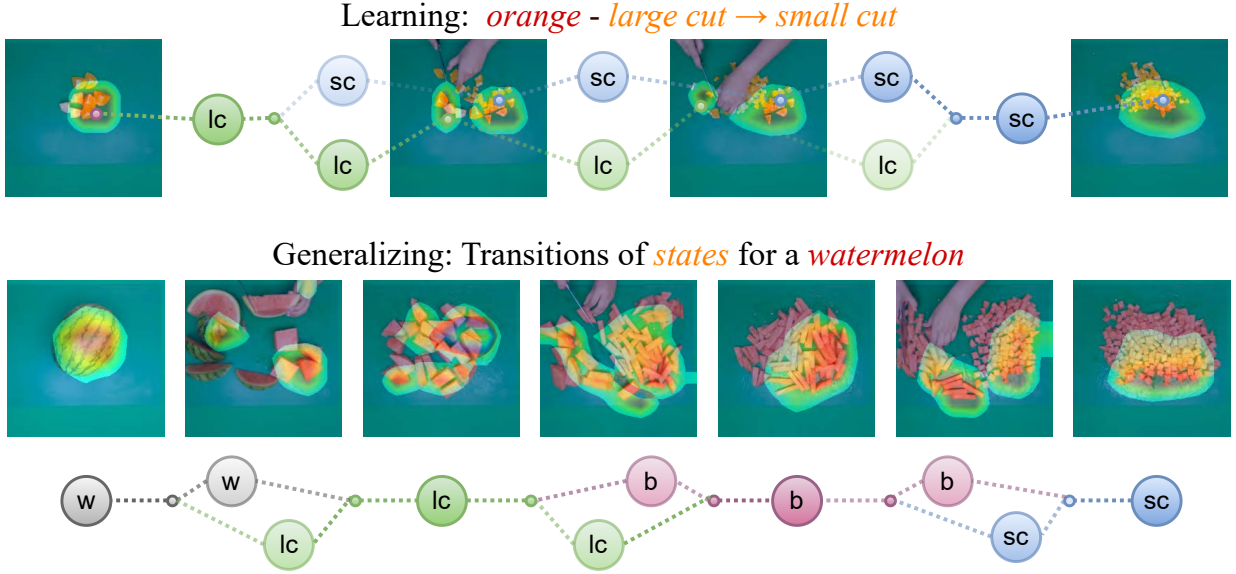


Figure 3.6: **Video parsing graph:** For a given video, we use Grad-CAM[7] on the intermediate frames to identify and visualize the class activation maps corresponding to the most salient states. Top: A training video clip has one transition of *orange* from *large cut* $\rightarrow$ *small cut*. Bottom: We can learn single transitions from training data, to generalize transitions in a long video with multiple state changes and parse the video as a graph.

into features, and adds transformer layers to model spatial-temporal relationships. However, without any appearance information, STLT achieves low performance on all metrics. On the other hand, with the appearance features, which are extracted by inflated 3D ResNet50 [115] (R3D), it can achieve much higher performances than STLT. Finally, Cross-Attention Fusion (CAF) applies cross-attention [116] to fuse the layout (STLT) and appearance (R3D) branch embeddings, achieving the best results. It demonstrates that combining the layout and appearance information together can help predict object and state types more accurately.

3.6 Discussion

We discuss the potential future use of ChopNLearn, while addressing the limitations and scope as well.

Long-term Video Parsing. We use compositional state recognition to further understand the

temporal dynamics [92, 117–119] with the aid of a video parsing graph construction as previously explored in Ego-Topo [120] and VideoGraph [121]. Each clip in the training set has one state transformation (top example in Figure 3.6). We visualize the class activation maps corresponding to the most salient intermediate state transitions with Grad-CAM [7], to learn the transition in each frame of the video for training data. This is illustrated as a graph for a training video. Having learned multiple single transformations, we can now extend this knowledge to understand long activities, with multiple transitions. As shown in fig. 3.6, we can learn state changes for `orange from large cut → small cut` using our training clip. Given a long unseen video with multiple clips, we can construct a state-transition graph to represent changes in state for a `watermelon`. Hence, by using an extensive array of videos, the process of learning transitions between individual states can be extended to encompass transitions between multiple states. This enables the creation of a self-supervised transition knowledge graph for comprehensive long-term video comprehension, as demonstrated in [92, 122].

Limitations. With advent of foundation models, few-shot generalization is an increasingly important task. In this work, we explore the potential of ChopNLearn for the research in compositional generation and recognition for highly complex and interdependent concepts. Admittedly, ChopNLearn is a small scale dataset with green screen background, which restricts the models trained on it to have specific biases. Nonetheless, this is the first attempt to understand how fine-grained states (cut styles) can be transferred to diverse objects. We explore this by using ChopNLearn as a test set for larger models, fine-tuning these models using ChopNLearn and trying them with or without a green screen background. We further see the potential of using ChopNLearn for benefiting the community in even more challenging tasks such as 3D reconstruction, video frame interpolation, state change generation, *etc.*.

3.7 Conclusion

In this chapter, we propose ChopNLearn, a new dataset for measuring the ability of models to recognize and generate unseen compositions of objects in different states, a skill known as compositional generalization. We also introduce two tasks, Compositional Image Generation and Compositional Action Recognition, and benchmark the performance of state-of-the-art generative models and video recognition methods on these tasks. We show the challenges with the existing approaches and their failure in some cases in their ability to generalize to new compositions. However, these two tasks are just the tip of the iceberg. Understanding object states is important for multiple image and video tasks such as 3D reconstruction, future frame prediction, video generation, summarization, and parsing of long-term video. We hope that our dataset will help the computer vision community to propose and learn new compositional tasks for images, videos, 3D, and beyond.

Chapter 4: Generating Images: Style Transfer Meets Multiple Modalities

4.1 Introduction

As a research topic at the intersection of computer vision and art, Image Style Transfer (IST) aims to apply certain style patterns to a given content image. The seminal work of Gatys *et al.* [123] proposed to transfer the style of one image to another content image by optimizing the pixel values using both style and content losses, inspiring many subsequent works in this field. To speed up the style transfer process, [124] trained a feed-forward neural network for each style to transfer it to different contents. Going beyond single style transfer, [125] introduced the idea of arbitrary style transfer and aimed to transfer arbitrary styles to any content in a single forward pass. Based on this formulation, [126–133] improved [125] on multiple aspects.

The above-mentioned methods can be classified as *Image-guided Image Style Transfer (IIST)*. They rely on reference style images, which are not always accessible in real-world scenarios. For example, artists may conceive novel styles that can be easily described via texts but never exist in previous artworks. Such a dependence on reference style images limits the application of IIST methods [134].

Recently, based on the large-scale image-text pretrained model CLIP [135], several methods proposed to edit images purely conditioned on text descriptions, achieving *Text-guided Image Style Transfer (TIST)* [134, 136–138]. Notably, by training a lightweight U-Net on a single



Figure 4.1: **Our multimodality-guided image style transfer results.** We show style transfer results guided by two text styles (left) and two image styles (right). 15 stylized images are synthesized by evenly interpolating between these four styles.

content image using CLIP loss, CLIPStyler [134] can synthesize stylized images from arbitrary content images and style text descriptions, setting a state-of-the-art (SOTA) performance on this task. However, although the styles of transferred images by CLIPStyler are generally consistent with the corresponding text descriptions, CLIPStyler often adds undesirable local patterns to the stylized images, distorting the original content severely, as shown in Figure 4.2. This indicates CLIPStyler fails to disentangle the style and content information from both text and image.

To address these issues, we propose a novel framework to better manipulate images based on reference style texts. Meanwhile, to offer more flexibility than IIST and TIST methods, our framework is designed to accept style guidance from multiple sources and modalities, enabling *MultiModality-guided Image Style Transfer (MMIST)*. The ability to exploit multimodal style references can be useful in many scenarios. For example, an artist may design new artistic styles by modifying styles of existing artworks; such modified styles can be easily defined by combining text descriptions and existing art images, yet are difficult to describe with text or image reference

only.

To realize MMIST, we propose a novel cross-modal GAN inversion method which generates diverse style representations according to multi-modal style inputs (*e.g.*, text and image). Such generated style representations allow us to generalize any IIST methods to tackle the problem of MMIST. Specifically, we leverage a pretrained GAN model and invert style text descriptions and/or style images into GAN’s latent space to get the corresponding style reference images. In this process, style-specific CLIP-based guidance is used to connect the domains of text and image. After obtaining the style reference images, we then feed them into an existing IIST approach which is adapted to take multiple style references as input. To further enhance the quality of stylized images, we propose a novel multi-style boosting strategy which enriches the style patterns. Similar to learned model parameters, the style representations can be reused at test time, allowing our method to stylize arbitrary contents in a single forward pass.

We evaluate our framework on 44 style text descriptions and 61 content images, which result in 2,684 style-content combinations. Both qualitative and user study results clearly show the improvement of our framework over previous methods on TIST task. Furthermore, extensive experiments also confirm the effectiveness of our framework on MMIST task and cross-modal style interpolation.

Our main contributions can be summarized as follows:

- We introduce a more general problem than IIST and TIST, *i.e.*, MultiModality-guided Image Style Transfer (MMIST), and solve it with a novel framework. The proposed framework can transfer styles from arbitrary number of reference images/texts to arbitrary content, a task which is not feasible for all existing methods to the best of our knowledge.

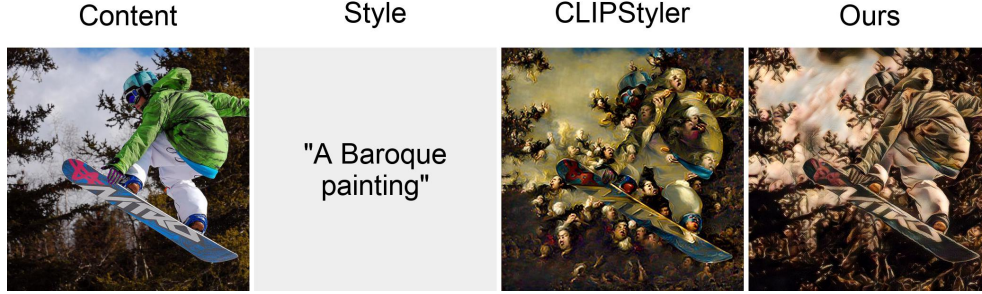


Figure 4.2: **Failure case of CLIPStyler.** We show one content and one style text description, together with the results from CLIPStyler and our method. CLIPStyler adds many small face-like patterns to the stylized images.

- We propose a novel cross-modal GAN inversion method to distill styles from different modalities. This inversion procedure also enables our method to interpolate between different styles arbitrarily.
- Extensive experiments and large-scale user studies (5,041 users) confirm the effectiveness of our model in terms of both qualitative results and user preference.

4.2 Related Work

4.2.1 Image-guided Image Style Transfer

Image-guided Image Style Transfer (IIST) has become a popular research topic since the seminal work of Gatys *et al.* [123]. Using a deep feature-based style loss and content loss, Gatys *et al.* [123] directly optimized pixel values to obtain decent style transfer results. To address the slow optimization problem in this method, [124] and several subsequent works [139–141] proposed to train feed-forward neural networks that can apply the pretrained style to arbitrary images in a single forward pass. Given its success, this idea was further developed by allowing one single trained model to store multiple styles. For example, [142] used multiple

convolutional filter banks to explicitly represent multiple styles; [143] proposed conditional instance normalization to achieve the same goal; and [144] introduced a selector structure to support incremental learning for new styles.

Recently, arbitrary IIST started to attract widespread attention due to its effectiveness, efficiency, and flexibility. [125] proposed AdaIN to perform IST by adaptively aligning the mean and variance of content features with those of style features. With similar motivation, [126–128, 130] introduce new loss functions or novel mechanisms to improve the style transfer quality. [129] adaptively performs attentive normalization on a per-point basis. Besides quality, some works focus on other properties of IST methods, *e.g.*, domain-awareness [145] or brushstroke-level optimization [146].

4.2.2 Text-guided Image Manipulation

Text-guided image manipulation aims to manipulate the input image based on a text description while preserving text-irrelevant parts in the original image. [147] employed a GAN-based encoder-decoder model to achieve this goal. [148] further introduced a text-adaptive discriminator to ensure that only text-related regions are modified. By extending GAN-based text-to-image generators [149–151], Li *et al.* [152] proposed ManiGAN to manipulate images in a multi-stage manner. [153] utilized GAN inversion, visual-linguistic similarity learning, and instance-level optimization to build a unified framework for multimodal image generation and manipulation with text.

Recently, the success of CLIP [135] in connecting the domains of image and text has inspired a new direction to achieve text-guided image manipulation. By modifying the latent space

of StyleGAN [48, 154, 155] using CLIP guidance, StyleCLIP [136] can perform text-guided manipulation in three different ways. VQGAN-CLIP [156] achieved the same goal by using CLIP loss to optimize the latent space of VQGAN [157]. StyleGAN-NADA[138] proposed a directional CLIP loss to optimize a GAN model instead of the latent space, resulting in more accurate manipulation effects. More recently, diffusion models [103, 158–160] have been combined with CLIP to obtain better performance [137, 161].

Most of these works are intended for content or attribute editing. Although some of them [137, 138] can be applied to style editing or transfer, the quality they can achieve is far from desirable. By contrast, CLIPStyler [134] is specifically designed to solve the task of TIST, outperforming all previous image manipulation methods on this task. However, CLIPStyler still suffers from certain drawbacks, as illustrated in Figure 4.2. Similar to CLIPStyler, [162] proposed to transfer an image’s style conditioned on an artist’s name. In this chapter, we generalize the tasks of TIST and IIST to MMIST, and solve it under a unified framework.

4.2.3 GAN Inversion

Traditional GAN inversion aims to invert a given image back into the latent space of a pretrained GAN generator. It emphasizes the accuracy and fidelity of the reconstructed image. GAN inversion can be applied to a wide range of downstream tasks, including image manipulation [136, 138, 153], image interpolation [163], image generation [153], etc. Learning-based [164], optimization-based [163], or hybrid [165] methods have been developed to invert a GAN. All of them use a reconstruction loss such as L_2 loss or LPIPS loss [166].

In this chapter, we propose cross-modal GAN inversion. Different from traditional methods

that pursue a perfect reconstruction of the whole input image, our cross-modal GAN inversion only reconstructs partial information of the input, *i.e.*, style, which is defined by inputs of multiple modalities such as text and image.

4.3 Method Overview

Given a set of style images $\{I_i\}_{i=1}^{N_I}$ and a set of style text descriptions $\{T_i\}_{i=1}^{N_T}$, our framework applies these specified styles to a set of content images $\{C_i\}_{i=1}^{N_C}$ and synthesizes a corresponding set of stylized images $\{Y_i\}_{i=1}^{N_C}$. Different from previous image editing methods that directly optimize the stylized image [134], the latent of the content image [136, 156], or parameters of a generative model [137, 138], we train a model for certain styles in a content-agnostic manner.

Our key insight is that MMIST can be achieved with the aid of style representations that comply with the input style text descriptions and image patterns. More specifically, we can generalize IIST methods to leverage such style representations and thereby create stylized images guided by multiple modalities. To this end, we propose a novel cross-modal GAN inversion method to map all input multimodal style references into the $\mathcal{W}^+$ space [163] of a pretrained StyleGAN3 [155] generator G , and thereby generate intermediate style representations $\{\mathcal{S}_i\}_{i=1}^{N_S}$ in the image space. As is shown in Figure 4.3, the style representations are images consisting of style *patterns* without meaningful contents. The cross-modal GAN inversion ensures that $\{\mathcal{S}_i\}_{i=1}^{N_S}$ summarizes and combines the style information from all input styles $\{I_i\}_{i=1}^{N_I}$ and $\{T_i\}_{i=1}^{N_T}$. To leverage $\{\mathcal{S}_i\}_{i=1}^{N_S}$, we adapt an existing IIST method to make it compatible with multiple style inputs. Denote the adapted IIST method by M . We use M to stylize the content images $\{C_i\}_{i=1}^{N_C}$

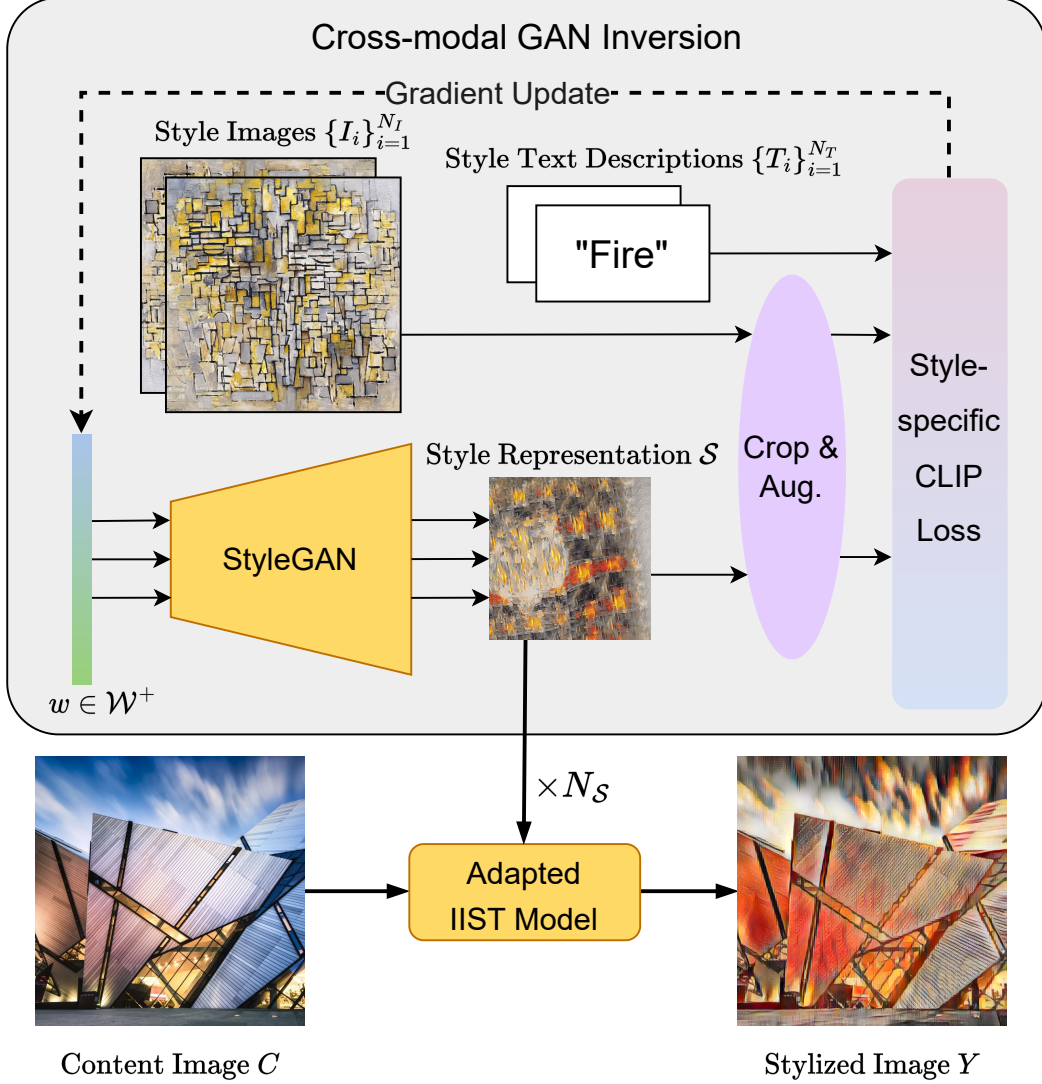


Figure 4.3: **The overview of our method.** Taking styles from multiple modalities as input, our method generates style representations using cross-modal GAN inversion. With the adapted IIST model, our method can apply the cached styles to any unseen content image in a single forward pass.

with intermediate style representations $\{S_i\}_{i=1}^{N_S}$, producing stylized outputs $\{Y_i\}_{i=1}^{N_C}$.

Separating style representation generation from stylized image synthesis is the key to the success of our framework. When performing TIST task, previous methods [134, 137, 156] always deal with style alignment and content preservation simultaneously, resulting in distorted content or irrelevant artifacts that appear in the results. In contrast, by leveraging the strong style-content

disentangling ability of IIST approaches, our method can put the entire focus on style representation generation for creating high-quality stylized images. Besides, with generated intermediate style representations, only a single forward pass is needed for our method to apply a learned style to any unseen content image.

4.4 Cross-modal GAN Inversion

To generate style representations from modalities other than image or text only, we propose cross-modal GAN inversion. In Table 4.1, we compare it with traditional GAN inversion. The goal of traditional GAN inversion is to faithfully reconstruct the original input image. Naturally, this only works for the image modality, and only one image at a time. Since it is targeting pixel-wise reconstruction, all information from the input image is supposed to be stored in the latent space of the GAN generator. However, the goal of cross-modal GAN inversion is completely different. It aims to combine different styles together to generate intermediate style representations. Therefore, the inversion should be able to accept multiple inputs from different references and modalities. Besides, only the style components of inputs are required to be inverted as their content parts are irrelevant to the downstream task.

4.4.1 Style-specific CLIP Loss

We employ CLIP [135] to connect image with other modalities, as well as to extract style components. However, naively applying CLIP cosine similarity loss does not result in accurate style representation, since the content components are also entangled in the CLIP embedding space.

Table 4.1: **Comparison between traditional GAN inversion and cross-modal GAN inversion.** “Ref.” means reference.

Method	Ref. Modality	Number of Ref.	Inversion Target
Traditional	Image Only	Single	Original Image
Cross-modal	Multiple	Multiple	Style

Following [134] and [138], we employ patch-wise CLIP loss to address this problem. Formally, denote the pretrained CLIP image encoder by E_I and text encoder by E_T . For each style text description T_i , we use the text-image patch-wise directional CLIP loss proposed by [134], *i.e.*,

$$\begin{aligned}
\mathcal{S} &= \mathbf{G}(w), \\
\{\mathcal{S}^j\}_{j=1}^{N_{\text{crop}}} &= \text{aug}(\text{crop}(\mathcal{S})), \\
\Delta \mathcal{S}^j &= E_I(\mathcal{S}^j) - E_I(I_{\text{src}}), \\
\Delta T &= E_T(T_i) - E_T(T_{\text{src}}), \\
L_{T_i} &= \frac{1}{N_{\text{crop}}} \sum_{j=1}^{N_{\text{crop}}} \left(1 - \frac{\Delta \mathcal{S}^j \cdot \Delta T}{\|\Delta \mathcal{S}^j\| \|\Delta T\|} \right),
\end{aligned} \tag{4.1}$$

where $w \in \mathcal{W}^+$ is a vector in StyleGAN3 latent space that we optimize, $\mathcal{S}$ is the style representation generated by $\mathbf{G}$ from w . $\text{aug}(\cdot)$ is the augmentation function, $\text{crop}(\cdot)$ is the patch crop function, and N_{crop} is the number of cropped patches. T_{src} and I_{src} are the source text and source image used to compute CLIP embedding directions, respectively. For simplicity, T_{src} is set to be “a photo” following [134], whereas I_{src} is an arbitrary photo-realistic image.

Eq. 4.1 effectively measures the style similarity between the input text T_i and the generated style $\mathcal{S}$. However, we want our model to handle style inputs from the image modality as well. To

Algorithm 2: Cross-modal GAN Inversion

Data: A set of style images $\{I_i\}_{i=1}^{N_I}$, a set of style text descriptions $\{T_i\}_{i=1}^{N_T}$, and corresponding style weights $\{\alpha_i^I\}_{i=1}^{N_I}$, $\{\alpha_i^T\}_{i=1}^{N_T}$.

Result: The generated style representation $\mathcal{S}$ and its corresponding latent $w^* \in \mathcal{W}^+$.

```
1 Randomly initialize  $w$ ;  
2 repeat  
3    $\mathcal{S} \leftarrow \mathbf{G}(w)$ ;  
4   Run aug and crop on  $\mathcal{S}$  and  $\{I_i\}_{i=1}^{N_I}$  to obtain  $\{\mathcal{S}^j\}_{j=1}^{N_{\text{crop}}}$ ,  $\{I_i^k\}_{i,k=1,1}^{N_i, N_{\text{crop}}}$ ;  
5   Calculate  $L_{\text{sty}}$  using Eq. 4.1, Eq. 4.2, and Eq. 4.3;  
6   Adam update for  $w$  with  $\nabla_w L_{\text{sty}}$ ;  
7 until  $L_{\text{sty}}$  is converged;  
8 return  $(\mathcal{S}, w^*)$ ;
```

this end, we propose an image-image patch-wise directional CLIP loss as below:

$$\begin{aligned} \{I_i^k\}_{k=1}^{N_{\text{crop}}} &= \text{aug}(\text{crop}(I_i)), \\ \Delta I_i^k &= E_I(I_i^k) - E_I(I_{\text{src}}), \\ L_{I_i} &= \frac{1}{N_{\text{crop}}^2} \sum_{j=1}^{N_{\text{crop}}} \sum_{k=1}^{N_{\text{crop}}} \left(1 - \frac{\Delta \mathcal{S}^j \cdot \Delta I_i^k}{\|\Delta \mathcal{S}^j\| \|\Delta I_i^k\|} \right), \end{aligned} \tag{4.2}$$

where I_i is the input style image, and $\Delta \mathcal{S}^j$ is calculated using Eq. 4.1. Specifically, for every image I_i or $\mathcal{S}$ involved in the CLIP embedding computation, we first randomly crop a large number of patches and then augment them. After computing the loss for each patch, we average them together to obtain the final loss value. By computing the averaged cosine similarity between each direction pair, Eq. 4.2 accurately estimates the style similarity between the input image I_i and the generated style $\mathcal{S}$.

In the general case where multiple style images $\{I_i\}_{i=1}^{N_I}$ and style text descriptions $\{T_i\}_{i=1}^{N_T}$ are given, we calculate the style-specific CLIP loss L_{sty} , and solve the following optimization

Algorithm 3: Multimodality-guided Style Transfer

Data: Input styles $\{I_i\}_{i=1}^{N_I}$, $\{T_i\}_{i=1}^{N_T}$, style weights $\{\alpha_i^I\}_{i=1}^{N_I}$, $\{\alpha_i^T\}_{i=1}^{N_T}$, and a set of content images $\{C_i\}_{i=1}^{N_C}$.

Result: A set of stylized images $\{Y_i\}_{i=1}^{N_C}$.

```
1 if The aggregated feature  $F$  is not cached then
2   Run Algorithm 2  $N_S$  times to obtain  $\{\mathcal{S}_i\}_{i=1}^{N_S}$ ;
3    $\{F_i\}_{i=1}^{N_C} \leftarrow \mathbf{M}_f(\{\mathcal{S}_i\}_{i=1}^{N_S})$ ;
4    $F \leftarrow \text{aggregate}(\{F_i\}_{i=1}^{N_C})$ ;
5 end
6  $\{Y_i\}_{i=1}^{N_C} \leftarrow \mathbf{M}_t(\{C_i\}_{i=1}^{N_C}, F)$ ;
7 return  $\{Y_i\}_{i=1}^{N_C}$ ;
```

problem:

$$w^* = \arg \min_{w \in \mathcal{W}^+} L_{\text{sty}} = \arg \min_{w \in \mathcal{W}^+} \sum_{i=1}^{N_I} \alpha_i^I L_{I_i} + \sum_{i=1}^{N_T} \alpha_i^T L_{T_i}, \quad (4.3)$$

where $\{\alpha_i^I\}_{i=1}^{N_I}$ and $\{\alpha_i^T\}_{i=1}^{N_T}$ are the style weights.

4.4.2 Inversion Algorithm

With the style-specific CLIP guidance, it is straightforward to run our cross-modal GAN inversion algorithm. As shown in Algorithm 2, after initializing w , we repetitively calculate L_{sty} and use Adam optimizer [167] to update w , until L_{sty} has converged.

Note that in this algorithm, both $\{I_i\}_{i=1}^{N_I}$ and $\{T_i\}_{i=1}^{N_T}$ are optional. If only $\{T_i\}_{i=1}^{N_T}$ is given, it degenerates to text-guided style generation, converting our framework to a method for TIST. Similarly, if only $\{I_i\}_{i=1}^{N_I}$ is given, it degenerates into mixing multiple style images, *i.e.*, generating a mixture of styles represented by multiple input images. When both $\{I_i\}_{i=1}^{N_I}$ and $\{T_i\}_{i=1}^{N_T}$ are provided, cross-modal style interpolation, *e.g.*, interpolating a style between a given text and a given image, can be naturally achieved by adjusting the style weights $\{\alpha_i^I\}_{i=1}^{N_I}$ and $\{\alpha_i^T\}_{i=1}^{N_T}$.

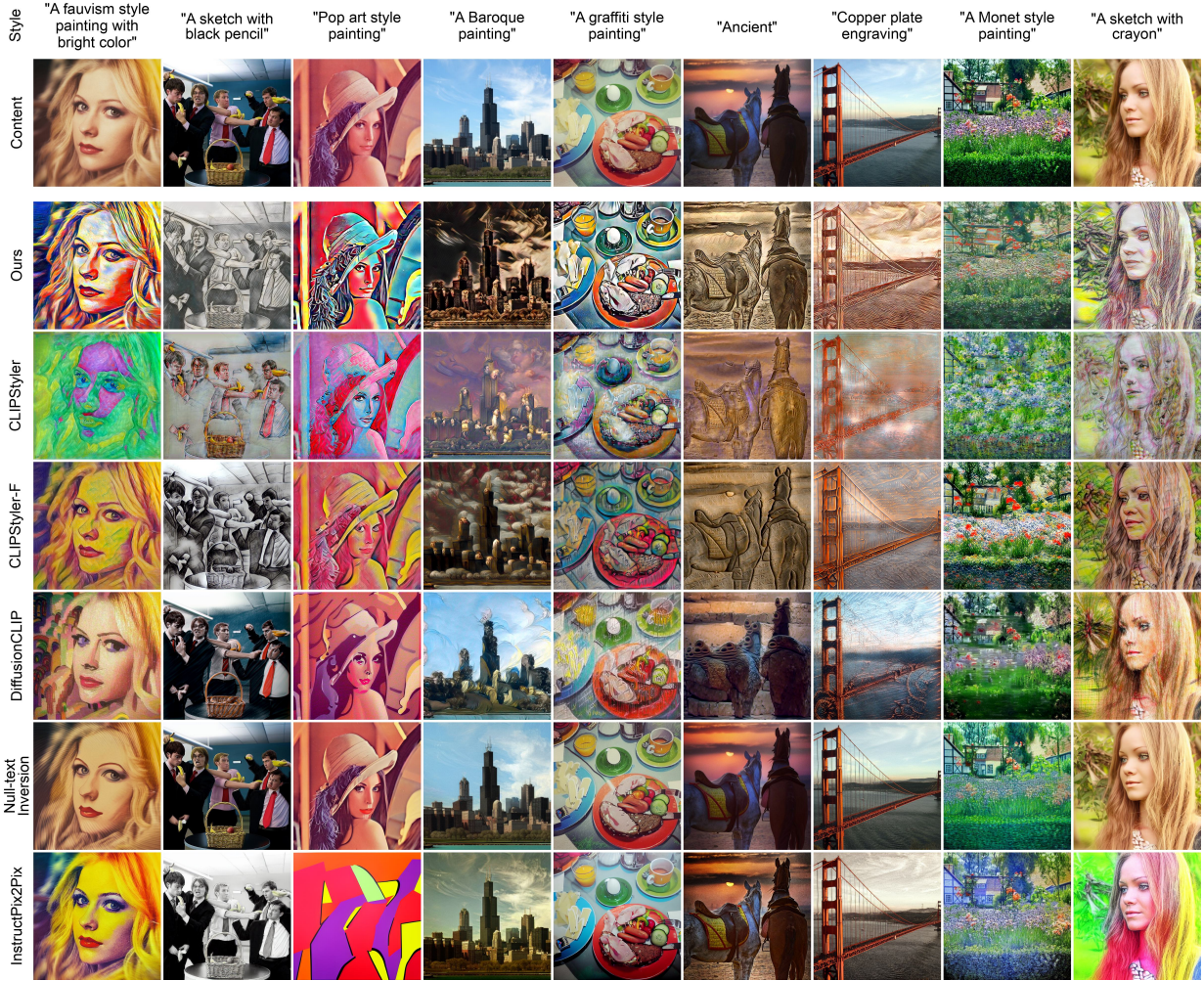


Figure 4.4: **Comparison with other TIST methods.** Our method delivers more accurate styles than others. Meanwhile, as opposed to other methods which often distort the content or add unreasonable patterns, content information is greatly preserved in results of our method.

4.5 Multimodality-guided Image Style Transfer

4.5.1 Multi-style Boosting

Due to the internal randomness of GAN inversion, one single intermediate style representation may not cover all style patterns specified by the input references, impairing the style transfer quality of final results. To address this problem, we propose a multi-style boosting algorithm. We aim to enrich the intermediate style representations, while keeping them compatible with the

adapted IIST model $\mathbf{M}$. Specifically, for each set of style inputs, we run cross-modal GAN inversion multiple times, resulting in a set of style representations $\{\mathcal{S}_i\}_{i=1}^{N_S}$. Then we feed them into $\mathbf{M}$ separately, and aggregate the outputs together to exploit $\{\mathcal{S}_i\}_{i=1}^{N_S}$. The aggregation strategy depends on the specific implementation of $\mathbf{M}$. We describe one instance of $\mathbf{M}$ and the aggregation strategy in the supplementary material.

4.5.2 Style Transfer Algorithm

We detail our style transfer method in Algorithm 3. For brevity, we assume the adapted IIST model $\mathbf{M}$ can be decomposed into a feature extraction network $\mathbf{M}_f$ and a style transfer module $\mathbf{M}_t$, *i.e.*, $\mathbf{M}(C, \mathcal{S}) = \mathbf{M}_t(C, \mathbf{M}_f(\mathcal{S}))$. Similar to Algorithm 2, Algorithm 3 can be used for TIST by only providing $\{T_i\}_{i=1}^{N_T}$, or MMIST by providing both $\{I_i\}_{i=1}^{N_I}$ and $\{T_i\}_{i=1}^{N_T}$. In particular, if only one T_0 is given, our method degenerates to the IIST approach it generalizes since cross-modal GAN inversion is not necessary in this case. The aggregated style feature F from each unique set of input styles can be cached for later use once it is produced. With cached F , when new content images arrive with the same input styles, only the style transfer part $\mathbf{M}_t$ needs to be executed. Since $\mathbf{M}_t$ is a feed-forward network, our method runs significantly faster than previous methods [134, 137, 156] that require ad-hoc optimization for each pair of style and content.

4.6 Experiments

We consider three tasks to evaluate our method: (1) TIST; (2) MMIST with one style image and one style text description; and (3) MMIST with style interpolated from multiple references in different modalities. Note that IIST is trivial and unnecessary to consider because our

Table 4.2: **Quantitative user study results on four TIST methods.** For each method, we report the number of positive responses received from raters, as well as its percentage (over 161,040 responses in total). Our method outperforms baselines.

Method	Style (%) $\uparrow$	Content (%) $\uparrow$	Overall (%) $\uparrow$
CLIPStyler	10631 (39.6)	9686 (36.1)	8966 (33.4)
CLIPStyler-F	14626 (54.5)	13453 (50.1)	11776 (43.9)
DiffusionCLIP	9768 (36.4)	7929 (29.5)	7395 (27.6)
Null-text Inversion	7375 (27.5)	16876 (62.8)	7166 (26.7)
InstructPix2Pix	10580 (39.4)	17453 (65.0)	8268 (30.8)
Ours	17151 (63.9)	18061 (67.3)	15125 (56.4)

method degenerates to the IIST approach it includes, as mentioned in Section 4.5.2. We employ AdaAttN [129] as our adapted IIST method in all our experiments. Please see the supplementary material for more implementation details.

4.6.1 Comparison with TIST Methods

Qualitative comparison. We conducted a qualitative comparison between our method and several TIST methods. In this task, style transfer is conditioned on a single text description. Therefore, we simply set $N_T = 1$ and $N_I = 0$ in our method. We consider CLIPStyler [134], CLIPStyler-F [134], DiffusionCLIP [137], Null-text Inversion [168], and InstructPix2Pix [169] as our baselines. These methods either exclusively focus on this task [134] and achieve state-of-the-art performance, or treat TIST as one of their practical applications. It is worth noting that CLIPStyler and CLIPStyler-F are proposed in the same paper, and the latter is an extension of the former and achieves TIST in a single forward pass for learned style text descriptions. By caching style representations, the speed of our method is similar to CLIPStyler-F, which is significantly faster than CLIPStyler and DiffusionCLIP.

Figure 4.4 shows comparison results on 9 content-style pairs. We list the style text descrip-

tions and the content image in the first and the second rows, respectively. Style transfer results of different methods are listed in the remaining rows. As is shown, CLIPStyler often breaks or distorts the original contents (1st and 7th columns). For results generated by DiffusionCLIP, their styles do not match the corresponding text descriptions well (1st, 2nd, and 7th columns). In addition, these methods tend to add undesirable local patterns to stylized images (4th and 9th columns). The absence of an original image caption likely explains why Null-text Inversion’s outcomes closely mirror the content images, displaying minimal style variations. InstructPix2Pix sometimes compromises the content (3rd column) or introduces inaccurate styles (2nd and 7th columns). In contrast, our method delivers more accurate styles while greatly preserving the content information.

Quantitative user study. We conduct a large-scale quantitative user study to better understand the performance of our method. We still use CLIPStyler [134], CLIPStyler-F [134], DiffusionCLIP [137], Null-text Inversion [168], and InstructPix2Pix [169] as our baselines. We apply 44 distinctive text-described styles to 61 different content images, giving 2,684 stylized images. For each of them, we ask 10 different raters to evaluate it from three aspects: style consistency, content preservation, and overall quality. For each aspect, raters are asked if the stylized image *respects the aspect well* (positive) or *not* (negative). In total we obtain 161,040 responses where each method receives 26,840 responses and each stylized image receives 10 responses. The total number of raters involved is 5,041. We report the number and the percentage of positive responses. Table 4.2 shows that our method outperforms all baselines in every aspect. Interestingly, CLIPStyler-F received more positive responses than CLIPStyler, although the former is designed to be a fast extension in [134]. This user study result is consistent with the qualitative results shown in Figure 4.4.

Table 4.3: **Speed comparison with other TIST methods.** Our method is the fastest. (★) Null-text Inversion does not need per-style training but has a per content inversion time of 80.7s.

Method	Per-style Training Time (s) ↓	Stylization FPS ↑
CLIPStyler	0	0.03
CLIPStyler-F	44.1	4.40
DiffusionCLIP	1885	0.05
Null-text Inversion	★	0.07
InstructPix2Pix	0	0.08
Ours	12.6	6.00

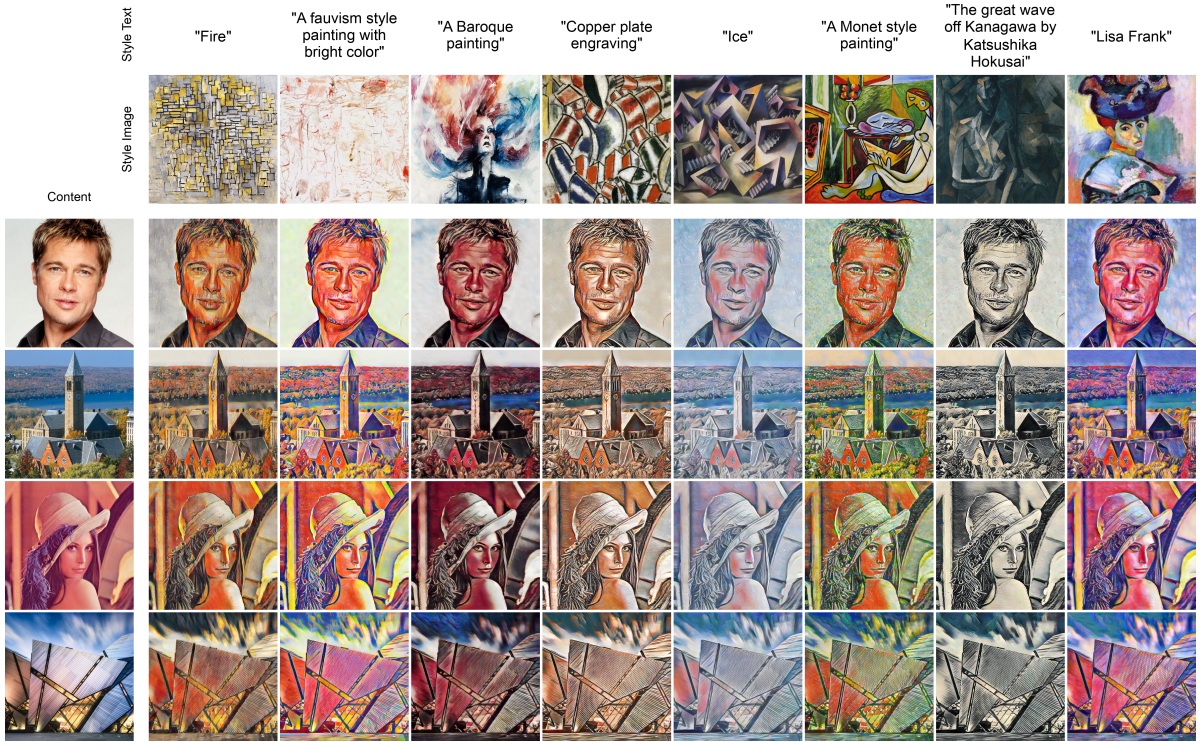


Figure 4.5: **MMIST results.** The first and second rows show style text descriptions and style images, respectively. The first column shows content images. Our method successfully mixes multimodal styles and applies them to various content images.

Running speed. Our pre-computed style representations enable *fast stylization* at *test time*. We compare speed of our method with other TIST methods in Table 4.3. All methods are under their fastest setting and run on an RTX A6000.

4.6.2 MMIST and Cross-modal Style Interpolation

Qualitative results of MMIST with one style image and one style text description. To the best of our knowledge, MMIST is infeasible for all the existing methods. However, under our unified framework, it can be easily performed by providing style references from more than one modality. We first consider the case where input style is specified through one style image and one style text description. Figure 4.5 shows MMIST results of our method on 8 different text-image styles and 4 content images. Our method can successfully summarize styles from a pair of style image and text description, and apply it to various contents. Moreover, contents are consistently preserved in all stylized images.

Qualitative results of MMIST with style interpolated from multiple references in different modalities. As mentioned in Section 4.5.2, our method is able to handle arbitrary number of style inputs from different modalities. To finer control the mixture degree of two or more given styles, we interpolate the styles with various ratios. This is done by setting different $\{\alpha_i^I\}_{i=1}^{N_I}$ and $\{\alpha_i^T\}_{i=1}^{N_T}$, while keeping their summation fixed. Note that under our unified framework, style interpolation between any number or kinds of modalities can be achieved in the same way. Figures 4.1 and 4.6 demonstrate the style interpolation results defined by different mixtures of style images and text descriptions. Our method produces decent and reasonable stylized images. More results can be found in the supplementary material.

4.6.3 Ablation Study

We investigate a few design choices of our method and quantitatively measure their effects in practice. To this end, we consider the TIST task. In our experiment, we randomly pair the

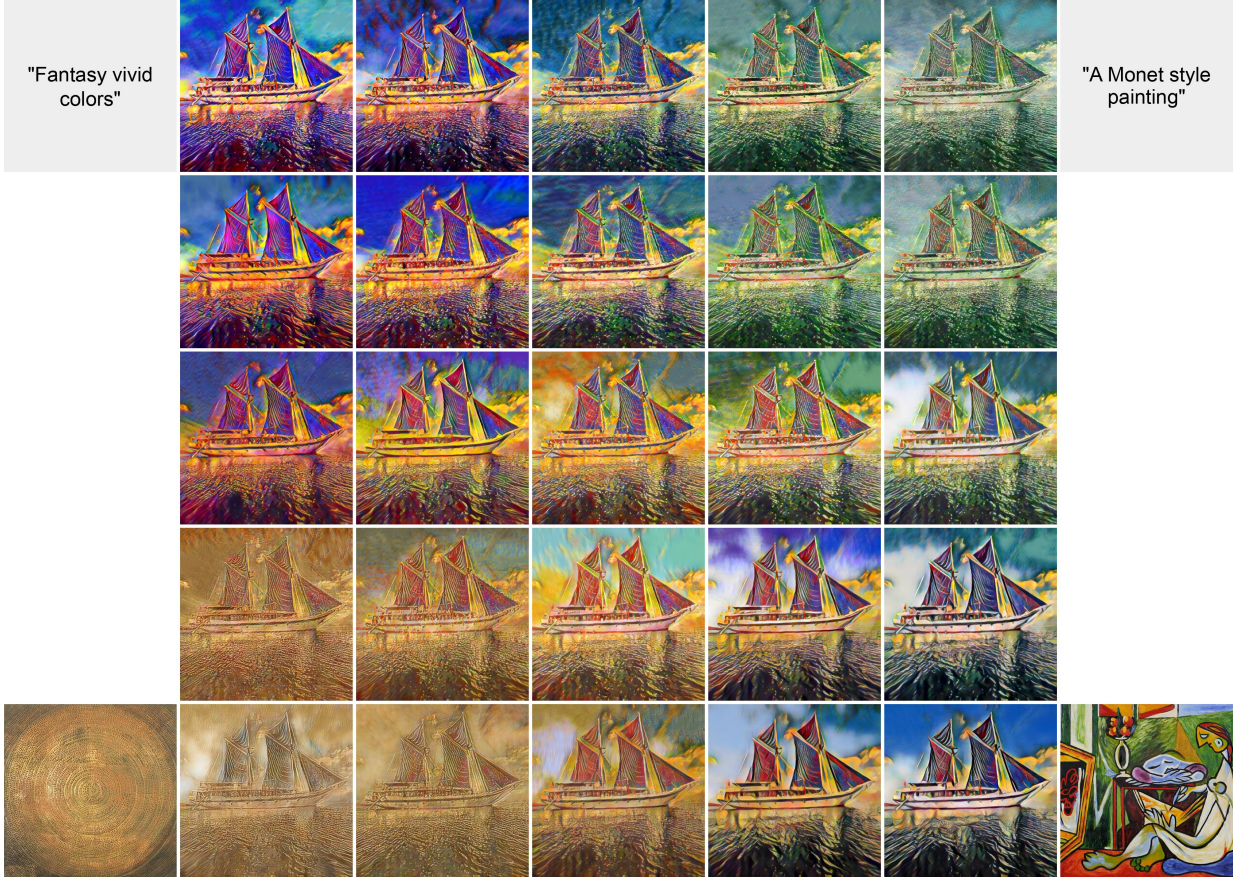


Figure 4.6: **MMIST and Style interpolation with four image and text styles.** Please zoom in to see the details.

style text descriptions and content images, and obtain 1,012 pairs for stylized image generation. For each specific pair, 10 raters are asked to pick their preferred stylized image between the one generated by a certain design choice and the one generated by our full method. We report the user preference percentage for all design choices in Table 4.4. In the table, CropSize128 means we crop 128×128 patches when calculating style-specific CLIP loss. NoCrop means we do not crop or augment the image, and directly apply the loss. NoAug means we do not apply augmentation to the cropped patches. PatchLoss500 and PatchLoss2500 mean we set $\alpha_0^T = 500$ and $\alpha_0^T = 2500$, respectively. NoBoosting means we do not use multi-style boosting when applying the style to contents. We observe that proper crop size and multi-style boosting are critical to the

Table 4.4: **Ablation study on different design choices.** The performance is evaluated through the user study.

Setting	Preference % $\uparrow$	Setting	Preference % $\uparrow$
CropSize128	39.5	PatchLoss500	49.9
NoCrop	41.9	PatchLoss2500	48.2
NoAug	47.9	NoBoosting	34.5

good performance of our method, whereas the effects of augmentation and style weight are relatively minor to our method. Changing the style weight from 500 to 2500 almost does not affect the performance, indicating that our method is quite robust to this hyperparameter. Qualitative ablation study results are available in the supplementary material.

4.7 Conclusions and Future Work

In this chapter, we present a unified style transfer framework to transfer styles defined by multiple modalities. The proposed cross-modal GAN inversion enables our framework to combine different styles and faithfully transfer them to arbitrary images. Extensive experiments demonstrate that our method achieves SOTA performance on TIST. In addition, the proposed method handles the new MMIST problem and cross-modal style interpolation task effectively.

While our work only considers style information from image and text, there is no theoretical restriction on our method to obtain styles from other modalities, *e.g.*, audio. We leave the exploration of this direction as future work.

Chapter 5: Generating Videos: Video Tokenization with Learned Autoregressive Prior

5.1 Introduction

The field of generative modeling has experienced significant advancements, largely driven by the success of autoregressive (AR) models in the development of large language models (LLMs) [170–175]. Building on AR transformers [112], these models are considered pivotal for the future of AI due to their exceptional performance [176, 177], impressive scalability [178–180], and versatile flexibility [171, 172].

Inspired by the success of LLMs, recent works have begun to employ AR transformers for visual generation [8, 52, 157, 181–184]. Additionally, several recent developments have extended LLMs to handle multimodal inputs and outputs [185, 186], further demonstrating the promising potential of AR models in visual content generation. All of these methods employ a **visual tokenizer** to convert continuous visual signals into sequences of discrete tokens, allowing them to be autoregressively modeled in the same way as natural language is modeled by LLMs. Typically, a visual tokenizer consists of a visual encoder, a quantization module [1, 181], and a visual decoder. The generative modeling occurs in the quantized discrete latent space, with the decoder mapping the generated discrete token sequences back to continuous visual signals. It is

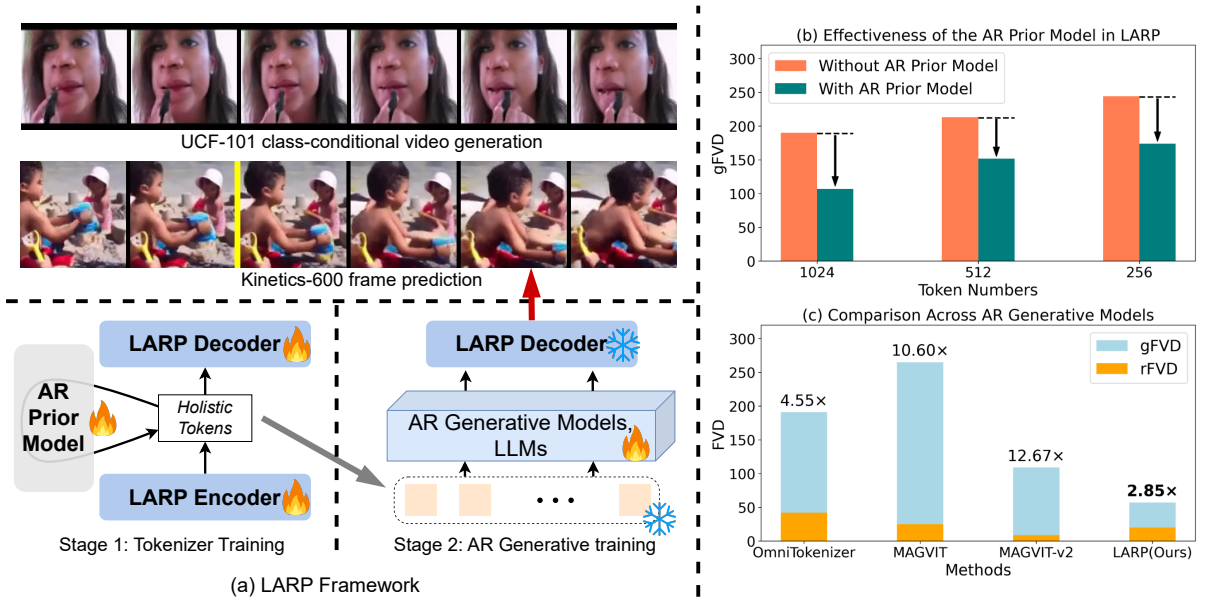


Figure 5.1: **LARP highlights.** (a) LARP is a video tokenizer for two-stage video generative models. In the first stage, LARP tokenizer is trained with a lightweight AR prior model to learn an AR-friendly latent space. In the second stage, an AR generative model is trained on LARP’s discrete tokens to synthesize high-fidelity videos. (b) The incorporation of the AR prior model significantly improves the generation FVD (gFVD) across various token number configurations. (c) LARP shows a much smaller gap between its reconstruction FVD (rFVD) and generation FVD (gFVD), indicating the effectiveness of the optimized latent space it has learned.

evident that the visual tokenizer plays a pivotal role, as it directly influences the quality of the generated content. Building on this insight, several works have focused on improving the visual tokenizer [1, 187], making solid progress in enhancing the compression ratio and reconstruction fidelity of visual tokenization.

Most existing visual tokenizers follow a patchwise tokenization paradigm [1, 8, 157, 181], where the discrete tokens are quantized from the encoded patches of the original visual inputs. While these approaches are intuitive for visual data with spatial or spatiotemporal structures, they restrict the tokenizers’ ability to capture global and holistic representations of the entire input. This limitation becomes even more pronounced when applied to AR models, which rely on sequential processing and require locally encoded tokens to be transformed into linear 1D

sequences. Previous research [157] has demonstrated that the method of flattening these patch tokens into a sequence is critical to the generation quality of AR models. Although most existing works adopt a raster scan order for this transformation due to its simplicity, it remains uncertain whether this is the optimal strategy. In addition, there are no clear guidelines for determining the most effective flattening order.

On the other hand, although the reconstruction fidelity of a visual tokenizer sets an upper bound on the generation fidelity of AR models, the factors that determine the gap between them remain unclear. In fact, higher reconstruction quality has been widely reported to sometimes lead to worse generation fidelity [188, 189]. This discrepancy highlights the limitations of the commonly used reconstruction-focused design of visual tokenizers and underscores the importance of ensuring desirable properties in the latent space of the tokenizer. However, very few works have attempted to address this aspect in improving image tokenizers [188, 190], and for video tokenizers, it has been almost entirely overlooked.

In this chapter, we present LARP, a video tokenizer with a **L**earned **A**uto**R**egressive generative **P**rior, designed to address the underexplored challenges identified in previous work. By leveraging a ViT-style spatialtemporal patchifier [191] and a transformer encoder architecture [112], LARP forms an autoencoder and employs a stochastic vector quantizer [181] to tokenize videos into holistic token sequences. Unlike traditional patchwise tokenizers, which directly encode input patches into discrete tokens, LARP introduces a set of learned queries [192, 193] that are concatenated with the input patch sequences and then encoded into holistic discrete tokens. An illustrative comparison between the patchwise tokenizer and LARP is shown in Figure 5.2 (a) and the left part of Figure 5.2 (b). By decoupling the direct correspondence between discrete tokens and input patches, LARP allows for a flexible number of discrete tokens,

enabling a trade-off between tokenization quality and latent representation length. This design also empowers LARP to produce more holistic and semantic representations of video content.

To further align LARP’s latent space with AR generative models, we incorporate a lightweight AR transformer as a prior model. It autoregressively models LARP’s latent space during training, providing signals to encourage learning a latent space that is well-suited for AR models. Importantly, the prior model is trained simultaneously with the main modules of LARP, but it is discarded during inference, adding zero memory or computational overhead to the tokenizer. Notably, by combining holistic tokenization with the co-training of the AR prior model, LARP automatically determines an order for latent discrete tokens in AR generation and optimizes the tokenizer to perform optimally within that structure. This approach eliminates the need to manually define a flattening order, which remains an unsolved challenge for traditional tokenizers.

To evaluate the effectiveness of the LARP tokenizer, we train a series of Llama-like [174, 175, 194] autoregressive (AR) generation models. Leveraging the holistic tokens and the learned AR generative prior, LARP achieves a Frechét Video Distance (FVD) [195] score of 57 on the UCF101 class-conditional video generation benchmark [196], establishing a new state-of-the-art among all published video generative models, including proprietary and closed-source approaches like MAGVIT-v2 [1]. To summarize, our key contributions are listed as follows:

- We present LARP, a novel video tokenizer that enables flexible, holistic tokenization, allowing for more semantic and global video representations.
- LARP features a learned AR generative prior, achieved by co-training an AR prior model, which effectively aligns LARP’s latent space with the downstream AR generation task.
- LARP significantly improves video generation quality for AR models across varying to-

ken sequence lengths, achieving state-of-the-art FVD performance on the UCF101 class-conditional video generation benchmark and outperforming all AR methods on the K600 frame prediction benchmark.

5.2 Related Work

5.2.1 Discrete Visual Tokenization

To enable AR models to generative high resolution visual contents, various discrete visual tokenization methods have been developed. The seminal work VQ-VAE [52, 181] introduces vector quantization to encode continuous images into discrete tokens, allowing them to be modeled by PixelCNN [197]. VQGAN [157] improves visual compression rate and perceptual reconstruction quality by incorporating GAN loss [198] in training the autoencoder. Building on this, several works focus on improving tokenizer efficiency [199] and enhancing generation quality [188, 190, 200]. Leveraging the powerful ViT [191] architecture, ViT-VQGAN [201] improves VQGAN on image generation tasks.

Inspired by the success of image tokenization, researchers extend VQGAN to videos using 3D CNNs [183, 202, 203]. C-ViT [204] employs the temporal-causal ViT architecture to tokenize videos, while more recent work, MAGViT-v2 [1], introduces lookup-free quantization, significantly expanding the size of the quantization codebook. OmniTokenizer [8] unifies image and video tokenization using the same tokenizer model and weights for both tasks.

It is worth noting that all of the above tokenizers follow the patchwise tokenization paradigm discussed in Section 5.1, and are therefore constrained by patch-to-token correspondence. Very recently, a concurrent work [189] proposes a compact tokenization approach for images. How-

ever, it neither defines a flattening order for the discrete tokens nor introduces any prior or regularization to improve downstream generation performance.

5.2.2 Visual Generation

Visual generation has been a long-standing area of interest in machine learning and computer vision research. The first major breakthrough comes with the rise of Generative Adversarial Networks (GANs) [48, 154, 198, 205], known for their intuitive mechanism and fast inference capabilities. AR methods are also widely applied in visual generation. Early works [197, 206, 207] model pixel sequences autoregressively, but are limited in their ability to synthesize high-resolution content due to the extreme length of pixel sequences. Recent advancements in visual tokenization make AR generative models for visual content more practical. While all tokenizers discussed in Section 5.2.1 are suitable for AR generation, many focus on BERT-style [208] masked visual generation [209], such as in [1, 189, 203]. Diffusion models [210–212] have recently emerged to dominate image [160] and video synthesis [213], delivering impressive visual generation quality. By utilizing VAEs [214] to reduce resolution, latent diffusion models [103, 215] further scale up, enabling multimodal visual generation [105, 216–218].

5.3 Method

5.3.1 Preliminary

Patchwise Video Tokenization. As discussed in Section 5.1, existing video tokenizers adopt a patchwise tokenization scheme, where latent tokens are encoded from the spatiotemporal patches of the input video. Typically, a patchwise video tokenizer consists of an encoder $\mathcal{E}$, a decoder

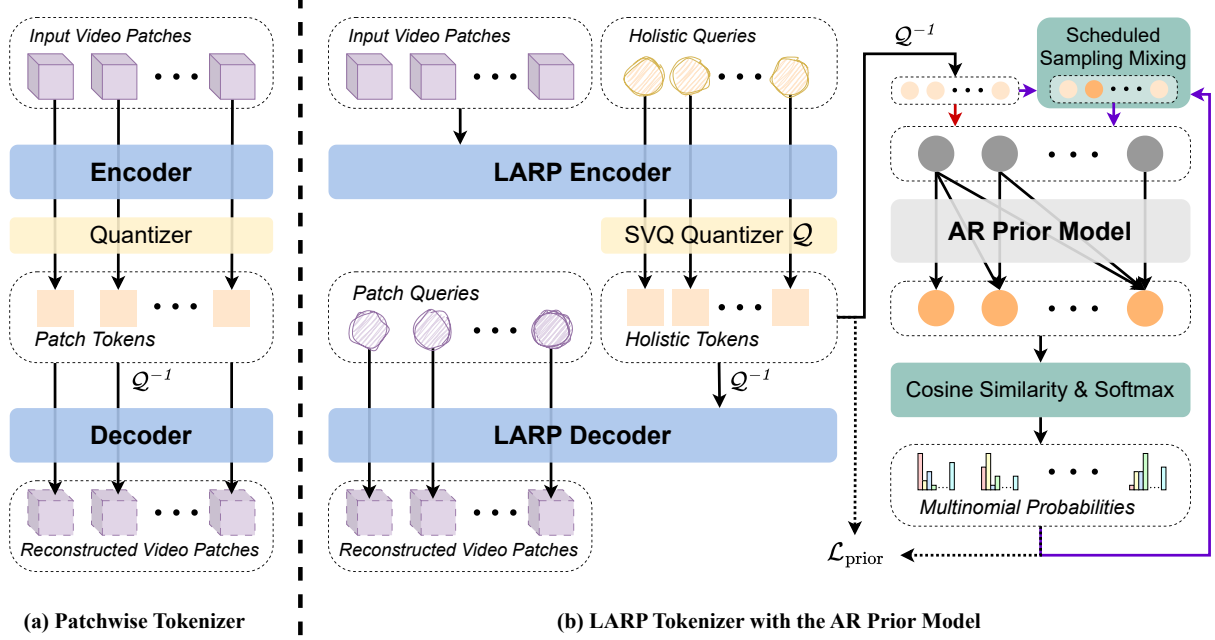


Figure 5.2: **Method overview.** Cubes $\boxtimes$ represent video patches, circles $\circ$ indicate continuous embeddings, and squares $\square$ denote discrete tokens. **(a) Patchwise video tokenizer** used in previous works. **(b) Left: The LARP tokenizer** tokenizes videos in a holistic scheme, gathering information from the video using a set of learned queries. **Right: The AR prior model**, trained with LARP, predicts the next holistic token, enabling a latent space optimized for AR generation. The AR prior model is forwarded in two rounds per iteration. The red arrow represents the first round, and the purple arrows represent the second round. The reconstruction loss $\mathcal{L}_{\text{rec}}$ is omitted for simplicity.

$\mathcal{D}$, and a quantizer $\mathcal{Q}$. Given a video input $\mathbf{V} \in \mathbb{R}^{T \times H \times W \times 3}$, it is encoded, quantized, and reconstructed as:

$$\mathbf{Z} = \mathcal{E}(\mathbf{V}), \quad \mathbf{X} = \mathcal{Q}(\mathbf{Z}), \quad \hat{\mathbf{V}} = \mathcal{D}(\mathbf{X}), \quad (5.1)$$

where $\mathbf{Z} \in \mathbb{R}^{\frac{T}{f_T} \times \frac{H}{f_H} \times \frac{W}{f_W} \times d}$ refers to the spatialtemporally downsampled video feature maps with d latent dimensions per location, $\mathbf{X} \in \mathbb{N}^{T' \times H' \times W'}$ denotes the quantized discrete tokens, and $\hat{\mathbf{V}}$ is the reconstructed video. f_T, f_H, f_W are the downsampling factors for the spatialtemporal dimensions T, H, W , respectively.

Despite different implementations of the encoder $\mathcal{E}$, decoder $\mathcal{D}$, and quantizer $\mathcal{Q}$, all patch-wise tokenizers maintain a fixed downsampling factor for each spatiotemporal dimension. The latent vector $\mathbf{Z}_{i,j,k,:} \in \mathbb{R}^d$ at each position is typically the direct output of its spatiotemporally corresponding input video patch (*e.g.* same spatiotemporal location in CNNs, or token position in transformers). While this design is intuitive for 3D signals like video, it limits the discrete tokens to low-level patch features, hindering their ability to capture higher-level, holistic information. Moreover, this formulation introduces the challenge of flattening patch tokens into a unidirectional sequence, which is critical for AR generation.

Autoregressive Modeling. Given a sequence of discrete tokens $\mathbf{x} = (x_1, x_2, \dots, x_n)$, we can train a neural network to model the probability distribution $p_\theta(\mathbf{x})$ autoregressively as follows:

$$p_\theta(\mathbf{x}) = \prod_{i=1}^n p_\theta(x_i \mid x_1, \dots, x_{i-1}, \theta), \quad (5.2)$$

where θ denotes the neural network parameters. This model can be conveniently trained by optimizing the negative log-likelihood (NLL) of $p_\theta(\mathbf{x})$. During inference, it iteratively predicts the next token x_i by sampling from $p_\theta(x_i \mid x_1, \dots, x_{i-1}, \theta)$, based on the previously generated tokens.

While autoregressive modeling imposes no direct constraints on data modality, it does require the data to be both *discrete* and *sequential*, which necessitates the use of a visual tokenizer when applied to images or videos.

5.3.2 Holistic Video Tokenization

Patchify. LARP employs the transformer architecture [112] due to its exceptional performance and scalability. Following the ViT framework [191], we split the input video into spatialtemporal patches, and linearly encode each patch into continuous transformer patch embeddings. Formally, given a video input $\mathbf{V} \in \mathbb{R}^{T \times H \times W \times 3}$, the video is linearly patchified as follows:

$$\mathbf{P} = \mathcal{P}(\mathbf{V}), \quad \mathbf{E} = \text{flatten}(\mathbf{P}), \quad (5.3)$$

where $\mathcal{P}$ denotes the linear patchify operation, $\mathbf{P} \in \mathbb{R}^{\frac{T}{f_T} \times \frac{H}{f_H} \times \frac{W}{f_W} \times d}$ is the spatialtemporal patches projected onto d dimensions, and $\mathbf{E} \in \mathbb{R}^{m \times d}$ is the flattened d -dimensional patch embeddings. Here, f_T, f_H, f_W are the downsampling factors for dimensions T, H, W , respectively, and $m = \frac{T}{f_T} \times \frac{H}{f_H} \times \frac{W}{f_W}$ is the total number of tokens. Importantly, the patch embeddings $\mathbf{E}$ remain local in nature, and therefore cannot be directly used to generate holistic discrete tokens.

Query-based Transformer. To design a holistic video tokenizer, it is crucial to avoid directly encoding individual patches into discrete tokens. To achieve this, we adapt the philosophy of [192, 193] to learn a set of fixed input queries to capture the holistic information from the video, as illustrated in the left section of Figure 5.2 (b). For simplicity, LARP employs a transformer encoder* architecture, as opposed to the transformer encoder-decoder structure used in [192]. In-context conditioning is applied to enable information mixing between different patch and query tokens.

Formally, we define n learnable holistic query embedding $\mathbf{Q}_L \in \mathbb{R}^{n \times d}$, where each embed-

*Here and throughout this paper, “transformer encoder” refers to the specific parallel transformer encoder architecture defined in [191]

ding is d -dimensional. These query embeddings are concatenated with the patch embeddings $\mathbf{E}$ along the token dimension. The resulting sequence, now of length $(n + m)$, is then input to the LARP encoder $\mathcal{E}$ and quantizer $\mathcal{Q}$ as follows:

$$\mathbf{Z} = \mathcal{E}(\mathbf{Q}_L \parallel \mathbf{E}), \quad \mathbf{x} = \mathcal{Q}(\mathbf{Z}_{1:n,:}), \quad (5.4)$$

where $\parallel$ denotes the concatenation operation, $\mathbf{Z}$ is the latent embeddings, and $\mathbf{x} = (x_1, \dots, x_n)$ denotes the quantized discrete tokens. Note that only $\mathbf{Z}_{1:n,:}$, *i.e.* the latent embeddings corresponding to the queries embeddings, are quantized and used. This ensures that each discrete token x_i has equal chance to represent any video patch, eliminating both soft and hard local patch constraints.

The LARP decoder is also implemented as a transformer encoder neural network. During the decoding stage, LARP follows a similar approach, utilizing m learnable patch query embeddings $\mathbf{Q}_P \in \mathbb{R}^{m \times d}$. The decoding process is defined as:

$$\hat{\mathbf{Z}} = \mathcal{Q}^{-1}(\mathbf{x}), \quad \hat{\mathbf{V}} = \text{reshape}(\mathcal{D}(\mathbf{Q}_P \parallel \hat{\mathbf{Z}})_{1:m,:}), \quad (5.5)$$

where $\mathcal{Q}^{-1}$ denotes the de-quantization operation that maps discrete tokens $\mathbf{x}$ back to the continuous latent embeddings $\hat{\mathbf{Z}} \in \mathbb{R}^{n \times d}$. These embeddings are concatenated with the patch query embeddings $\mathbf{Q}_P$, and the combined sequence of length $m + n$ is decoded into a sequence of continuous vectors. The first m vectors are reshaped to reconstruct the video $\hat{\mathbf{V}} \in \mathbb{R}^{T \times H \times W \times 3}$.

Crucially, although the latent tokens $\mathbf{x}$ are now both holistic and discrete, no specific flattening order is imposed due to the unordered nature of the holistic query set and the parallel

processing property of the transformer encoder. As a result, $\mathbf{x}$ is not immediately suitable for AR modeling.

Stochastic Vector Quantization. While vector quantization (VQ) [181] has been widely adopted in previous visual quantizers [157, 183], its deterministic nature limits the tokenizer’s ability to explore inter-code correlations, resulting less semantically rich codes. To address these limitations, LARP employs a stochastic vector quantization (SVQ) paradigm to implement the quantizer $\mathcal{Q}$. Similar to VQ, SVQ maintains a codebook $\mathbf{C} \in \mathbb{R}^{c \times d'}$, which stores c vectors, each of dimension d' . The optimization objective $\mathcal{L}_{\text{SVQ}}$ includes a weighted sum of the commitment loss and the codebook loss, as defined in [181]. The key difference lies in the look-up operation. While VQ uses an $\arg \min$ operation to find the closest code by minimizing the distance between the input vector $\mathbf{v} \in \mathbb{R}^{d'}$ and all codes in $\mathbf{C}$, SVQ introduces stochasticity in this process. Specifically, SVQ computes the cosine similarities $vs.$ between the input vector $\mathbf{v}$ and all code vectors in $\mathbf{C}$, interprets these similarities as logits, and applies a softmax normalization to obtain the probabilities $\mathbf{p}$. One index x is then sampled from the resulting multinomial distribution $P(x)$. Formally, the SVQ process $x = \mathcal{Q}(\mathbf{v})$ is defined as:

$$vs. = \frac{\mathbf{v} \cdot \mathbf{C}_i}{\|\mathbf{v}\| \|\mathbf{C}_i\|}, \quad \mathbf{p} = \text{softmax}(vs.), \quad (5.6)$$

$$x \sim P(x) = \prod_{j=1}^n \mathbf{p}_i^{\mathbf{1}_{x=j}}, \quad (5.7)$$

where $\mathbf{1}$ denotes the indicator function. To maintain the differentiability of SVQ, we apply the straight-through estimator [219]. The de-quantization operation is performed via a straightfor-

ward index look-up, $\hat{\mathbf{v}} = \mathcal{Q}^{-1}(x) = \mathbf{C}_x$, similar to the standard VQ process.

Reconstructive Training. Following [157, 183, 203], the reconstructive training loss of LARP, $\mathcal{L}_{\text{rec}}$, is composed of L_1 reconstruction loss, LPIPS perceptual loss [166], GAN loss [198], and SVQ loss $\mathcal{L}_{\text{svq}}$.

5.3.3 Learning an Autoregressive Generative Prior

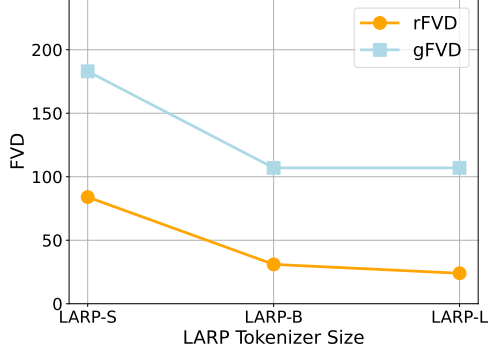
Continuous Autoregressive Transformer. To better align LARP’s latent space with AR generative models, we introduce a lightweight AR transformer as a prior model, which provides gradients to push the latent space toward a structure optimized for AR generation. A key challenge in designing the prior model lies in its discrete nature. Simply applying an AR model to the discrete token sequence would prevent gradients from being back-propagated to the LARP encoder. Furthermore, unlike the stable discrete latent spaces of fully trained tokenizers, LARP’s latent space is continuously evolving during training, which can destabilize AR modeling and reduce the quality of the signals it provides to the encoder. To address these issues, we modify a standard AR transformer into a continuous AR transformer by redefining its input and output layers, as depicted in the right section of Figure 5.2 (b).

The input layer of a standard AR transformer is typically an embedding look-up layer. In the prior model of LARP, this is replaced with a linear projection that takes the de-quantized latents $\hat{\mathbf{Z}}$ as input, ensuring proper gradient flow during training. The output layer of a standard AR transformer predicts the logits of the next token. While this does not block gradient propagation, it lacks awareness of the vector values in the codebook, making it unsuitable for the continuously evolving latent space during training. In contrast, the output layer of LARP’s AR prior model

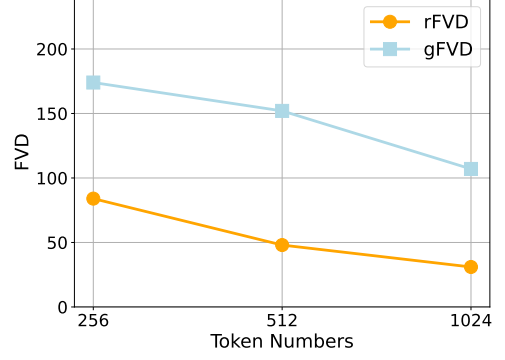
makes predictions following the SVQ scheme described in Section 5.3.2. It predicts an estimate of the next token’s embedding, $\bar{v} \in \mathbb{R}^{d'}$, which has the same shape as a codebook vectors C_i . Similar to SVQ, the predicted embedding $\bar{v}$ is used to compute cosine similarities with all code vectors in C , as described in Equation (5.6). These similarities are then softmax-normalized and interpreted as probabilities, which are used to compute the negative log-likelihood (NLL) loss with the input tokens as the ground truth. To predict the next token, a sample is drawn from the resulting multinomial distribution using Equation (5.7). This output layer design ensures that the AR prior model remains aware of the continuously evolving codebook, enabling it to make more accurate predictions and provide more precise signals to effectively train the LARP tokenizer.

Scheduled Sampling. Exposure bias [220] is a well-known challenge in AR modeling. During training, the model is fed the ground-truth data to predict the next token. However, during inference, the model must rely on its own previous predictions, which may contain errors, creating a mismatch between training and inference conditions. While the AR prior model in LARP is only used during training, it encounters a similar issue: as the codebook evolves, the semantic meaning of discrete tokens can shift, making the input sequence misaligned with the prior model’s learned representations. To address this problem, we employ the scheduled sampling technique [221, 222] within the AR prior model of LARP. Specifically, after the first forward pass of the prior model, we randomly mix the predicted output sequence with the original input sequence at the token level. This mixed sequence is then fed into the AR prior model for a second forward pass. The NLL loss is computed for both rounds of predictions and averaged, helping to reduce exposure bias and ensure more robust training.

Integration. Although the AR prior model functions as a standalone module, it is trained jointly with the LARP tokenizer in an end-to-end manner. Once the NLL loss $\mathcal{L}_{\text{prior}}$ is computed, it is



(a) Scaling LARP tokenizer sizes.



(b) Scaling number of discrete tokens.

Figure 5.3: **Scaling LARP tokenizer size and number of tokens.**

combined with the reconstructive loss $\mathcal{L}_{\text{rec}}$ to optimize the parameters of both the prior model and the tokenizer. Formally, the total loss is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \alpha \mathcal{L}_{\text{prior}}, \quad (5.8)$$

where α is the loss weight, and $\mathcal{L}_{\text{rec}}$ is defined in Section 5.3.2. Since α is typically set to a small value, we apply a higher learning rate to the parameters of the prior model to ensure effective learning. Importantly, the prior model is used solely to encourage an AR-friendly discrete latent space for LARP during training. It is discarded at inference time, meaning it has no effect on the inference speed or memory footprint.

5.4 Experiments

5.4.1 Setup

Dataset. We conduct video reconstruction and generation experiments using the Kinetics-600 (K600)[223] and UCF-101[196] datasets. In all experiments, we use 16-frame video clips with a

spatial resolution of 128×128 for both training and evaluation following [1, 183, 203].

Implementation Details. LARP first patchifies the input video. In all experiments, the patch sizes are set to $f_T = 4$, $f_H = 8$, and $f_W = 8$, respectively. As a result, a $16 \times 128 \times 128$ video clip is split into $4 \times 16 \times 16 = 1024$ video patches, which are projected into 1024 continuous patch embeddings in the first layer of LARP. For the SVQ quantizer, we utilize a factorized codebook with a size of 8192 and a dimension of $d' = 8$, following the recommendations of [201]. The softmax normalization in Equation (5.6) is applied with a temperature of 0.03. The AR prior model in LARP is adapted from a small GPT-2 model [172], consisting of only 21.7M parameters. Scheduled sampling for the AR prior model employs a linear warm-up for the mixing rate, starting from 0 and reaching a peak of 0.5 at 30% of the total training steps. We set AR prior loss weight $\alpha = 0.06$ in our main experiments, and use a learning rate multiplier of 50.

We employ a Llama-like [174, 175, 194] transformer as our AR generative model. One class token [cls] and one separator token [sep] are used in the class-conditional generation task on UCF101 and frame prediction task on K600, respectively.

Frechét Video Distance (FVD) [195] serves as the main evaluation metric for both reconstruction and generation experiments.

5.4.2 Scaling

To explore the effect of scaling the LARP tokenizer, we begin by varying its size while keeping the number of latent tokens fixed at 1024. As shown in Figure 5.3 (a), we compare the reconstruction FVD (rFVD) and generation FVD (gFVD) for three scaled versions of LARP : LARP-L, LARP-B, and LARP-S, with parameter counts of 173.0M, 116.3M, and 39.8M, respec-

Method	#Params		#Tokens	rFVD↓	gFVD↓	
	Tokenizer	Generator			K600	UCF
<i>Diffusion-based generative models with continuous video tokenizers</i>						
VideoFusion [224]	-	2B	-	-	-	173
HPDM [225]	-	725M	-	-	-	66
<i>MLM generative models with discrete video tokenizers</i>						
MAGVIT-MLM [203]	158M	306M	1024	25	9.9	76
MAGVIT-v2-MLM [1]	-	307M	1280	8.6	4.3	<u>58</u>
<i>AR generative models with discrete video tokenizers</i>						
CogVideo [182]	-	9.4B	2065	-	109.2	626
TATS [183]	32M	321M	1024	162	-	332
MAGVIT-AR [203]	158M	306M	1024	25	-	265
MAGVIT-v2-AR [1]	-	840M	1280	8.6	-	109
OmniTokenizer [8]	82.2M	650M	1280	42	32.9	191
LARP-L (Ours)	173M	343M	1024	24	6.2	107
LARP-L-Long (Ours)	173M	343M	1024	<u>20</u>	6.2	102
LARP-L-Long (Ours)	173M	632M	1024	<u>20</u>	<u>5.1</u>	57

Table 5.1: **Comparison of video generation results.** Results are grouped by the type of generative models. The scores for MAGVIT-AR and MAGVIT-v2-AR are taken from the appendix of MAGVIT-v2 [1]. LARP-L-Long denotes the LARP-L trained for more epochs. Our best results are obtained with a larger AR generator.

tively. All results are reported on the UCF-101 dataset. Interestingly, while rFVD consistently improves as the tokenizer size increases, gFVD saturates when scaling from LARP-B to LARP-L, suggesting that gFVD can follow a different trend from rFVD. Notably, as shown in Figure 5.1 (c), LARP has already achieved the smallest gap between rFVD and gFVD, further demonstrating the effectiveness of the optimized latent space it has learned.

One of LARP’s key features is its holistic video tokenization, which supports an arbitrary number of latent discrete tokens. Intuitively, using more tokens slows down the AR generation process but improves reconstruction quality. Conversely, using fewer tokens significantly speeds up the process but may lead to lower reconstruction quality due to the smaller information bot-

tleneck. To evaluate this trade-off, we use LARP-B and the default AR model, scaling down the number of latent tokens from 1024 to 512 and 256. The corresponding rFVD and gFVD results on the UCF-101 dataset are reported in Figure 5.3 (b). It is expected that both rFVD and gFVD increase when fewer tokens are used to represent a video. However, the rate of degradation in gFVD slows down when reducing from 512 to 256 tokens compared to rFVD, indicating improved generative representation efficiency.

5.4.3 Video Generation Comparison

For video generation, we compare LARP with other state-of-the-art published video generative models, including diffusion-based models, Masked Language Modeling (MLM) methods, and AR methods. We use the UCF-101 class-conditional generation benchmark and the K600 frame prediction benchmark, where the first 5 frames are provided to predict the next 11 frames in a 16-frame video clip. As shown in Table 5.1, LARP outperforms all other video generators on the UCF-101 dataset, setting a new state-of-the-art FVD of 57. Notably, within the family of AR generative models, LARP significantly surpasses all other AR methods by a large margin on both the UCF-101 and K600 datasets, including the closed-source MAGVIT-v2-AR [1]. Moreover, the last two rows of Table 5.1 demonstrate that using a larger AR generator can significantly improve LARP’s generation quality, highlighting the scalability of LARP’s representation.

5.4.4 Visualization

Video Reconstruction. In Figure 5.4, we compare video reconstruction quality of LARP with OmniTokenizer [8]. LARP consistently outperforms OmniTokenizer, particularly in complex

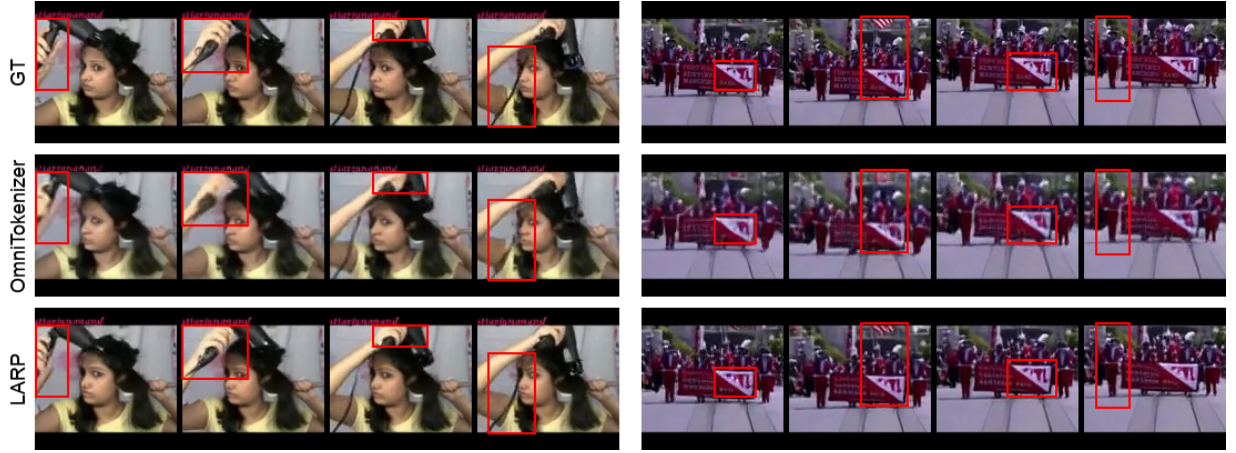


Figure 5.4: **Video reconstruction comparison** with OmniTokenizer [8].

scenes and regions, further validating the rFVD comparison results shown in Table 5.1.

Class-Conditional Video Generation. We present class-conditional video generation results in Figure 5.5. LARP constructs a discrete latent space that better suited for AR generation, which enables the synthesis of high-fidelity videos, not only improving the quality of individual frames but also enhancing overall temporal consistency. Additional results are provided in the appendix.

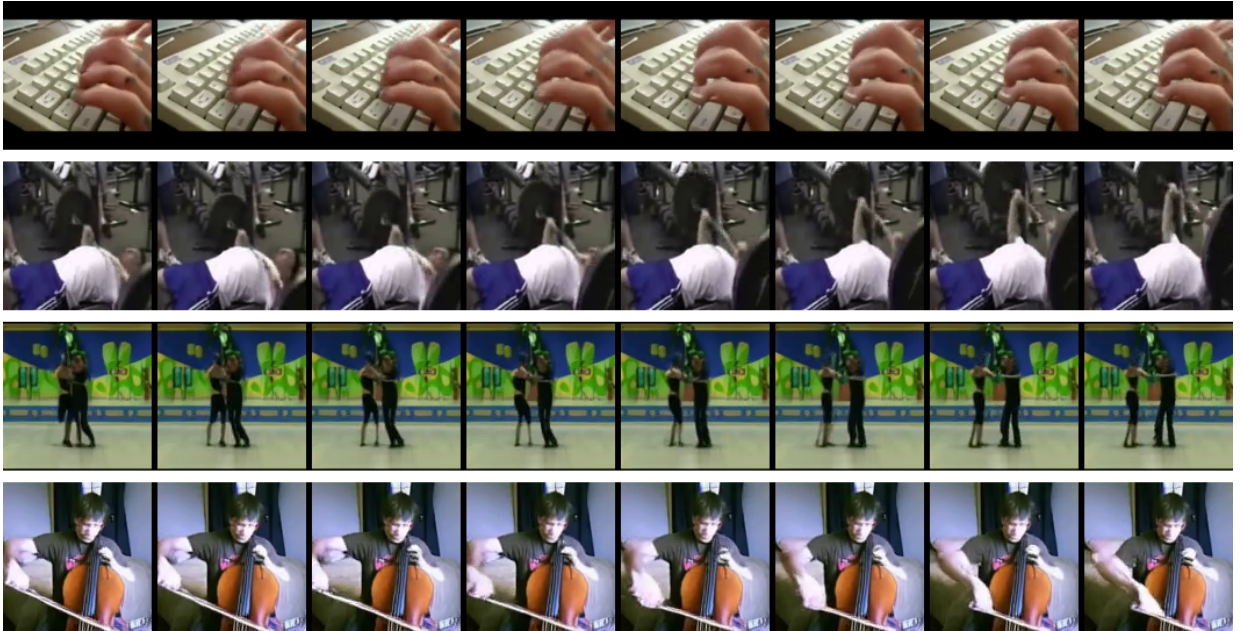


Figure 5.5: **Class-conditional video generation results** on the UCF-101 dataset using LARP.

Video Frame Prediction. Video frame prediction results are displayed in Figure 5.6. The vertical yellow line marks the boundary between the conditioned frames and the predicted frames. We use 5 frames as input to predict the following 11 frames, forming a 16-frame video clip, which is temporally downsampled to 8 frames for display. It is evident that LARP effectively predicts frames with diverse scenes and natural motions. Additional results are provided in the appendix.



Figure 5.6: **Video frame prediction results** on the K600 dataset using LARP.

5.4.5 Ablation Study

To assess the impact of the different components proposed in Section 5.3, we perform an ablation study, with results shown in Table 5.2. Clearly, the AR prior model contributes the most to the exceptional performance of LARP. As further validated in Figure 5.1 (b), the improvement from using the AR prior model remains consistent across different token numbers. The scheduled sampling for the AR prior model and the use of SVQ are also critical, as both are closely tied to

Configuration	PSNR $\uparrow$	LPIPS $\downarrow$	rFVD $\downarrow$	gFVD $\downarrow$
LARP-B	27.88	0.0855	31	107
No AR prior model	27.95	0.0830	23	190
No scheduled sampling in AR prior model	27.85	0.0856	27	142
Deterministic quantization	27.65	0.0884	27	149
Small AR prior model loss weight ($\alpha = 0.03$)	27.83	0.0866	28	120
No CFG	27.88	0.0855	31	121

Table 5.2: **Ablation study.** All configurations are modified from LARP-B model.

the AR prior model’s effectiveness. The loss weight of the AR prior model and the use of CFG have relatively minor effects on the generative performance. Interestingly, the model without the AR prior achieves the best reconstruction results but the worst generation results, highlighting the effectiveness of the AR prior model in *enhancing LARP’s discrete latent space for generative tasks*.

5.5 Conclusion and Future Work

In this chapter, we introduce LARP, a novel video tokenizer tailored specifically for autoregressive (AR) generative models. By introducing a holistic tokenization scheme with learned queries, LARP captures more global and semantic video representations, offering greater flexibility in the number of discrete tokens. The integration of a lightweight AR prior model during training optimizes the latent space for AR generation and defines an optimal token order, significantly improving performance in AR tasks. Extensive experiments on video reconstruction, class-conditional video generation, and video frame prediction demonstrate LARP’s ability to achieve state-of-the-art FVD scores. The promising results of LARP not only highlight its efficacy in video generation tasks but also suggest its potential for broader applications, including the development of multimodal large language models (MLLMs) to handle video generation and understanding in a unified framework.

Chapter 6: Generating with MLLMs: Growing Visual Generative Capacity for Pre-Trained Models

6.1 Introduction

Inspired by the success of large language models (LLMs) [170–175], multimodal large language models (MLLMs) [226, 227] have been developed to jointly process and understand visual and textual information within a next-token prediction framework. Leveraging pre-trained visual encoders such as CLIP [135] or SigLIP [228, 229], existing approaches typically project the extracted visual features into the latent space of the LLM and fine-tune the model to interpret visual signals through these embeddings. Despite their success across a wide range of visual understanding applications, these kinds of MLLMs are not truly “multimodal”, as they can only **understand** but **not generate** visual signals. This limitation not only prevents them from generating images as desired by users, but also hinders their ability to reason through the visual modality [230] when tackling more complex multimodal tasks.

To address these limitations, unified MLLMs have been designed to jointly learn visual understanding and generation within a single system. Broadly, existing unified MLLMs fall into two categories. *Hybrid MLLMs* [231–233] represent visual tokens with continuous embeddings and incorporate generative objectives such as diffusion or flow matching, generating images through

an iterative refinement process. In contrast, *pure autoregressive MLLMs* [234–238] rely on discrete visual tokens encoded by vector-quantized (VQ) tokenizers [157, 181] and are trained solely with negative log-likelihood loss, thereby following a clean next-token prediction paradigm for both language and vision generation.

Both understanding MLLMs and unified MLLMs are typically initialized from pure language LLMs in order to inherit strong linguistic capabilities. However, while unified MLLMs are functionally a superset of understanding MLLMs, most existing unified MLLMs do not inherit the visual understanding ability already established in understanding MLLMs. This gap can be attributed to differences in architectural choices and training objectives. For instance, unified MLLMs sometimes discard the pre-trained visual encoders [234, 235] used in understanding-oriented MLLMs, or employ non-autoregressive generative objectives [231, 232, 239] that deviate substantially from the training paradigm of understanding-oriented models.

In this chapter, we propose **Bi-modality Routing via Dual-branch Generative Experts (Bridge)**, a pure autoregressive unified MLLM that performs both image understanding and generation through the same next-token prediction framework. Unlike prior unified MLLMs, Bridge is directly built upon existing understanding-oriented MLLMs, thereby inheriting their strong visual comprehension capabilities. Crucially, by adopting a dual-branch modeling design, our approach preserves the inherited model’s visual understanding ability while enabling generative capacity, ensuring that comprehension is not compromised. This design reduces dependence on large-scale, high-quality visual understanding datasets, while also decreasing training time and enhancing overall training efficiency.

To further enhance visual generation quality, we propose a *semantic-to-pixel discrete representation* for images. This representation integrates two complementary levels of tokenization:

compact, high-level semantic tokens that capture global structure and meaning, and fine-grained pixel tokens that preserve detailed visual information such as textures and edges. By combining semantic abstraction with pixel-level precision, our discrete visual representation achieves both strong alignment with language modeling and accurate reconstruction of visual details. Remarkably, this richer representation requires only 7.9% increase in per-image token length compared to only using pixel tokens, yet it delivers significantly improved generation fidelity across diverse tasks.

To summarize, our key contributions are as follows:

- We propose **Bridge**, a pure autoregressive unified MLLM that expands pre-trained MLLMs with visual generative capacity while strictly preserving their inherited visual understanding ability. This design enables efficient unification of multimodal understanding and generation under a single next-token prediction framework.
- We introduce a *semantic-to-pixel discrete visual representation*, which combines compact high-level semantic tokens with fine-grained pixel tokens. This dual-level representation provides strong semantic alignment with language models while preserving detailed visual fidelity, significantly boosting generation quality with minimal increase in sequence length.
- We conduct extensive experiments across multimodal benchmarks, showing that our method achieves competitive or superior performance in both understanding and generation tasks, while requiring less training data, shorter training time, and overall improved efficiency compared to prior unified MLLMs.

6.2 Related Work

Unified Multimodal Large Language Models. Recent advances in multimodal large language models (MLLMs) aim to move beyond visual understanding to also support visual generation within a unified framework. Existing approaches can be broadly divided into two families. The first employs continuous visual embeddings combined with diffusion or flow-based objectives for image generation, as in Transfusion [232], BAGEL [231], Show-o2 [233], JanusFlow [240], and ILLUME [241]. While capable of producing high-quality synthesis, these methods break the clean autoregressive paradigm, thereby complicating training and integration. The second line of work adopts discrete vision tokens produced by VQ-based tokenizers [157, 181], enabling unified next-token prediction across both text and image tokens. Within this family, models such as Emu3 [236], Chameleon [242], and Janus [238, 243] achieve more elegant unification, but their reliance on pixel-level VQ tokenizers [157, 194] often limits semantic alignment with language models and weakens visual comprehension [234]. Other variants, such as Tar [234], UniTok [237], and VILA-U [235], attempt to bridge this gap through semantic quantization, but still suffer from reduced pixel-level fidelity. A few recent works [244–246] attempt to extend pre-trained MLLMs with generative capability, yet they heavily depend on external diffusion or flow-based decoders, and thus cannot be considered truly unified MLLMs. In contrast, Bridge relies solely on next-token prediction for image generation, without the need for any external generative models.

6.3 Method

In this section, we present the design of our pure autoregressive unified MLLM, **Bridge**. We begin with the architecture of Bridge in Section 6.3.1, describing how it is built on top of a pre-trained understanding MLLM. In Section 6.3.2, we introduce our semantic-to-pixel discrete visual representation, which enhances visual generation quality significantly. We then detail the data usage in Section 6.3.3, and finally outline the full training procedure in Section 6.3.4.

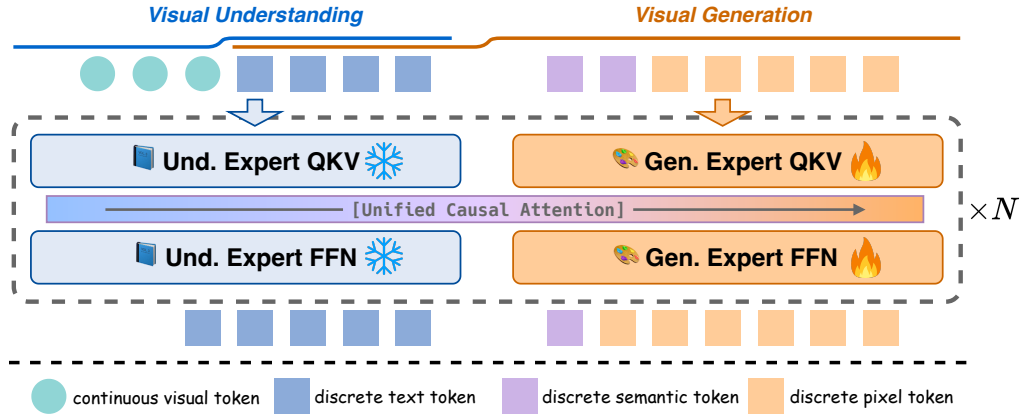


Figure 6.1: **Method overview.** Bridge adopts a Mixture-of-Transformers (MoT) architecture with two experts: a frozen understanding (Und.) expert for text and visual understanding tokens, and a newly trained generation (Gen.) expert for visual generation tokens. Both experts share unified causal attention across all tokens. Visual generation representation are constructed by concatenating short semantic token sequences with longer pixel token sequences, which are modeled jointly with text tokens under a unified next-token prediction objective. Semantic tokens serve as a bridge between text and pixel modalities, substantially improving visual generation quality.

6.3.1 Architecture

Bridge is built upon a pre-trained decoder-only MLLM. We adopt InternVL3 [247] with Qwen2.5 [248] model architecture as the backbone, selected for its competitive performance in both language modeling and visual understanding. To preserve these capabilities, we keep a

complete copy of the InternVL3 model, including its continuous vision encoder, frozen within Bridge. This ensures that the strong language modeling and visual understanding ability of the underlying MLLM remains intact.

To endow the model with visual generation capacity, we introduce additional trainable modules. Unlike prior works such as MetaQueries [245] or Qwen-Image [249], which treat the MLLM as a frozen text encoder and rely on external diffusion transformers [212, 250] for image synthesis, our goal is to construct a unified MLLM capable of both understanding and generating multimodal information entirely within the next-token prediction paradigm.

To achieve this, we leverage a Mixture-of-Transformers (MoT) design [231, 251, 252], as illustrated in Figure 6.1. Specifically, we copy the LLM backbone to create a generation expert and combine it with the frozen understanding expert. In contrast to Mixture-of-Experts (MoE) architectures, where only the feed-forward networks differ across experts, both the understanding and generation experts in Bridge are implemented as complete transformer blocks with separate parameter sets. The two experts interact at every attention layer, where their tokens are concatenated and processed jointly using standard causal masking, enabling seamless integration of understanding and generation within a unified autoregressive framework.

Hard routing is employed to dispatch tokens between the two experts. Text tokens are passed directly into the understanding expert. Images for understanding tasks are first encoded by the continuous vision encoder inherited from the backbone MLLM and then fed into the understanding expert. In contrast, image generation tokens are routed to the generation expert. During training, ground-truth images are converted into discrete visual tokens and provided to the generation expert for loss computation. At inference, the generation expert autoregressively predicts each discrete visual token, which is then fed back into the same expert to generate the

next token.

6.3.2 Semantic-to-pixel Discrete Visual Representation

A central challenge in pure autoregressive unified MLLMs lies in designing effective visual representations for generation. Most existing approaches [236, 238, 242, 243] rely on pixel-level tokens produced by VQGANs [157]. While these tokens capture fine-grained details, their heavy focus on low-level reconstruction makes them poorly aligned with language tokens, often leading to suboptimal image generation quality. In contrast, semantic-level tokens [234, 235], which are encoded by text-aligned visual encoders, exhibit strong alignment with language but lack sufficient precision for representing detailed visual content.

To combine the strengths of both representations, Bridge adopts a *semantic-to-pixel discrete visual representation*. Each image is represented as a sequence that begins with semantic tokens and is followed by pixel tokens:

$$\langle \text{BOI} \rangle \langle \text{SEM}_0 \rangle \langle \text{SEM}_1 \rangle \dots \langle \text{PIX}_0 \rangle \langle \text{PIX}_1 \rangle \dots \langle \text{EOI} \rangle,$$

where $\langle \text{BOI} \rangle$ and $\langle \text{EOI} \rangle$ denote special tokens marking the beginning and end of an image, $\langle \text{SEM}_i \rangle$ represents the i -th semantic token, and $\langle \text{PIX}_i \rangle$ represents the i -th pixel token.

The semantic tokens, placed at the front of the sequence, provide high-level structure and holistic information about the image. Their strong alignment with language tokens makes them easier to generate and helps bridge the modality gap. The subsequent pixel tokens supply fine-grained details necessary for accurate reconstruction. Although pixel tokens alone remain difficult to be modeled, the presence of preceding semantic tokens reduces this difficulty, aligning the entire sequence more effectively with text and thereby improving generation quality. This

design is conceptually similar to the Chain-of-Thought (CoT) mechanism [2] in LLMs, where intermediate reasoning steps improve the final result.

Concretely, we employ TA-Tok [234] as the semantic encoder and LlamaGen-VQGAN [194] as the pixel encoder. Instead of using TA-Tok’s de-tokenizer, we rely solely on the LlamaGen-VQGAN de-tokenizer to reconstruct images from the generated pixel tokens. Importantly, we find that a short, coarse semantic sequence is sufficient to capture holistic image semantics and connect text and pixel tokens. Thus, we use $3\times$ spatially downsampled semantic tokens from TA-Tok, resulting in 81 semantic tokens. When combined with 1024 pixel tokens for representing 512×512 images, this increases the total token length by less than 10%, while yielding significantly improved visual generation fidelity, as demonstrated in Section 6.4.2.

6.3.3 Data Curation

Our training data consists of image-to-text, text-to-image and interleaved multimodal datasets. Among them, our main goal is to collect and filter large-scale high-quality data for visual generation tasks. **(1) Filtering and recaptioning.** We first apply the aesthetic score and resolution filtering to the webdataset LAION-5B [253], where images with an aesthetic score >5 and resolution $>512\text{px}$ will be preserved. Then we leverage open-sourced VLMs [247, 254] to generate more accurate captions for these images. To enhance the model’s generation capability on text rendering and human face, we also apply text and human face detection pipeline to curate a subset. **(2) Synthetic Images.** To further improve the model’s generation quality, we collect open-sourced datasets generated by GPT-4o-Image [239, 255–257] and FLUX [234]. Besides, we also leverage Seedream 3.0 [258] to curate million scale text-to-image datasets using prompts

from other datasets [259, 260] and users [261, 262].

6.3.4 Training Procedure

Since Bridge represents images entirely with discrete tokens, we train it using a unified negative log-likelihood loss across all modalities. This formulation allows text and image tokens to be modeled consistently under the same autoregressive objective. To fully exploit diverse multimodal data, we adopt a multi-stage training procedure inspired by prior LLMs and MLLMs. Below, we describe the details of each stage.

Stage 1: Unified Multimodal Pre-training. The objective of this stage is to establish robust multimodal generation capability. We pre-train Bridge on large-scale heterogeneous datasets that include text-to-image pairs [253], interleaved multimodal documents [263, 264], and a small proportion of image-to-text data [265, 266]. During this stage, all input images, including those from image-to-text samples, are routed through the newly introduced generation encoder and processed by the generation expert. In the latter case, textual outputs are still produced by the frozen understanding expert but are conditioned on latent representations provided by the generation expert. This setup encourages the generation expert to learn rich and discriminative visual features, thereby improving downstream tasks that require image-conditioned generation, such as image editing and inpainting. The full pre-training stage uses 410M text-to-image pairs for visual generation, 57M image-to-text pairs for visual understanding, and 29M interleaved multimodal samples.

Stage 2: Continued Pre-training. In this stage, we focus on further enhancing the visual generation ability of Bridge by training on high-quality text-to-image datasets [234, 253]. Compared

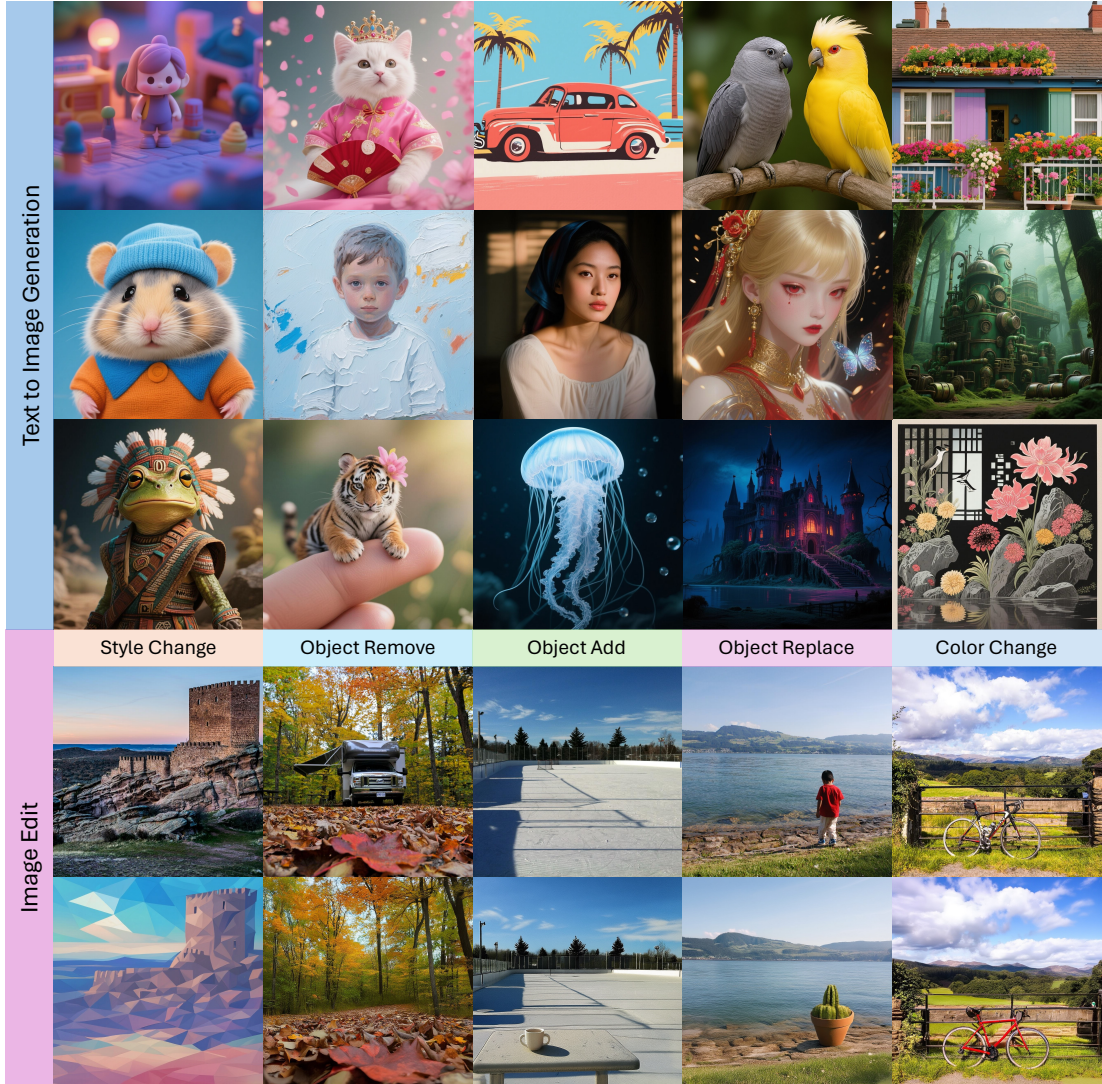


Figure 6.2: **Qualitative Results** on text-to-image generation and image editing tasks.

to the large but diverse corpus used in Stage 1, this data has higher aesthetic quality and includes more challenging cases, such as images with OCR contents or human faces. This stage exposes the model to approximately 60M multimodal examples (partially filtered from the pre-training corpus), enabling it to refine generation quality while improving robustness on visually complex and semantically demanding scenarios.

Stage 3: Supervised Fine-tuning. The supervised fine-tuning (SFT) stage uses the smallest amount of data, but of the highest quality. We leverage carefully curated text-to-image

Table 6.1: **Quantitative Results on Visual Understanding Benchmarks.** Models without a specified base (M)LLM are trained from scratch. A dash (–) is used when the corresponding value was not provided in the original work.

Model	Base (M)LLM	POPE $\uparrow$	MME-P $\uparrow$	MME-C $\uparrow$	MMB $\uparrow$	SEED $\uparrow$	MMMU $\uparrow$
<i>Understanding Only Models</i>							
LLaVA-v1.5	Vicuna-7B	85.9	1511	-	64.3	58.6	35.4
Qwen-VL	Qwen-7B	-	1488	-	60.6	58.2	-
LLaVA-NeXT	Vicuna-7B	86.5	1519	-	67.4	64.7	35.1
DeepSeek-VL	DeepSeek-7B	88.1	-	-	73.2	70.4	36.6
LLaVA-OV	Qwen2-7B	87.2	1580	418	80.8	75.4	48.8
<i>Unified Models</i>							
ILLUME	Vicuna-7B	88.5	1445	-	65.1	72.9	38.2
Chameleon	-	-	-	-	-	-	22.4
LWM	LLaMA2-7B	75.2	-	-	-	-	-
Emu3	-	85.2	-	-	58.5	68.2	31.6
Liquid	GEMMA-7B	81.1	1119	-	-	-	-
UniTok	LLaMA2-7B	83.2	1448	-	-	-	-
VILA-U	LLaMA2-7B	85.8	1402	-	-	59.0	-
Janus-Pro	DeepSeek-7B	87.4	1567	260	79.2	72.1	41.0
TokenFlow-XL	Qwen-2.5-14B	87.8	1551	371	76.8	72.6	43.2
MetaMorph	LLaMA-3.1-8B	-	-	-	75.2	71.8	41.8
Tar	Qwen2.5-7B	87.8	1571	355	74.4	73.0	39.0
Show-o2	Qwen2.5-7B	-	1621	-	79.3	69.8	48.9
BAGEL	Qwen2.5-7B	-	<u>1687</u>	-	85.0	-	55.3
LMFusion	LLaVA-NeXT-8B	-	1604	-	72.1	72.5	41.7
MetaQuery-XL	LLaVA-NeXT-8B	-	1685	-	83.5	76.9	58.6
UniWorld-V1	Qwen2.5-VL-7B	-	-	-	83.5	-	58.6
BLIP3-o	Qwen2.5VL-7B	-	1683	<u>647</u>	83.5	77.5	50.6
Bridge (Ours)	InternVL3-8B	<u>88.4</u>	1730	677	<u>84.4</u>	<u>77.4</u>	<u>57.4</u>

datasets [239, 255, 256, 259–262] and instructional image editing data [246, 257, 267, 268] to align the model’s outputs with human preferences. In total, this stage involves approximately 28M training samples, providing precise guidance for controllable and instruction-following generation.

6.4 Experiments

In all experiments, we use InternVL3-8B [247] as the base MLLM. Our semantic-to-pixel discrete representation is built upon two components: the visual encoder of TA-Tok [234] for semantic tokens and LlamaGen-VQGAN [194] for pixel tokens. The TA-Tok encoder takes

Table 6.2: **Quantitative Results on Text-to-image Generation Benchmarks.** † refers to the methods using prompt augmentation, *e.g.*, LLM rewriter or self-CoT [2]. A dash (–) is used when the corresponding value was not provided in the original work.

Method	DPG Bench			GenEval			WISE		
	Entity	Relation	Overall↑	Two Obj.	Color Attr.	Overall↑	Time	Space	Overall↑
<i>Generation Only Model</i>									
SDXL	82.43	86.76	74.65	0.74	0.23	0.55	0.48	0.47	0.43
Playground v2.5	82.59	84.08	75.47	-	-	-	0.58	0.55	0.49
Hunyuan DiT	80.59	74.36	78.87	-	-	-	-	-	-
DALLE3	89.61	90.58	83.50	0.87	0.45	0.67	-	-	-
SD3-Medium	91.01	80.70	84.08	0.94	0.60	0.74	0.44	0.48	0.42
SANA-1.5	-	-	84.70	0.93	0.65	0.81	-	-	-
NextStep-1	-	-	85.28	-	-	0.63 / 0.73†	0.54	0.61	0.54 / 0.79 †
<i>Unified Model</i>									
Chameleon	-	-	-	-	-	0.39	-	-	-
LWM	-	-	-	0.41	0.15	0.47	-	-	-
Emu3	86.68	90.22	80.60	0.71	0.21	0.54 / 0.66†	0.45	0.48	0.39
SEED-X-13B	-	-	-	0.58	0.14	0.49	-	-	-
Transfusion	-	-	-	-	-	0.63	-	-	-
ILLUME	-	-	-	0.86	0.28	0.61	-	-	-
Janus-Pro-7B	88.90	89.32	84.19	0.89	0.66	0.80	0.37	0.49	0.35
Tar-7B	88.62	93.98	84.19	0.92	0.65	<u>0.84</u>	-	-	-
Show-o2-7B	91.78	91.81	86.14	0.87	0.62	0.76	-	-	-
MetaQuery-XL	-	-	82.05	-	-	0.80†	0.55	0.62	0.55
BAGEL	-	-	-	0.94	0.63	0.82 / 0.88 †	0.55	0.68	0.52 / <u>0.70</u> †
UniWorld-V1	-	-	-	0.93	0.70	0.80	0.55	0.73	0.55
BLIP3-o-8B	-	-	81.60	-	-	<u>0.84</u>	-	-	0.62
Bridge (Ours)	90.10	92.27	<u>85.51</u>	0.93	0.66	0.74 / 0.82†	0.56	0.65	0.53 / 0.69†

images of size 384×384 as input, encodes and pools the features, and quantizes them into 81 discrete tokens selected from a codebook of size 65,536. The LlamaGen-VQGAN encoder uses a downsampling ratio of 16, producing $32 \times 32 = 1024$ pixel tokens from 512×512 images, drawn from a codebook of size 16,384. Concatenating the 81 semantic tokens with the 1024 pixel tokens yields a total of 1105 tokens per image. During training, we consistently resize and center-crop images to 512×512 before feeding them into the model. Besides, to further enhance the visual quality, we also develop an *optional* upscale module using Lumina-Accessory [269], raising the output resolution to 1024px.

Table 6.3: **Comparison results on ImgEdit.** Bridge achieves the best overall performance.

Model	Add	Adjust	Extract	Replace	Remove	Background	Style	Hybrid	Action	Overall $\uparrow$
Instruct-P2P	2.45	1.83	1.44	2.01	1.50	1.44	3.55	1.20	1.46	1.88
AnyEdit	3.18	2.95	1.88	2.47	2.23	2.24	2.85	1.56	2.65	2.45
UltraEdit	3.44	2.81	2.13	2.96	1.45	2.83	3.76	1.91	2.98	2.70
Step1X-Edit	3.88	3.14	1.76	3.40	2.41	3.16	4.63	2.64	2.52	3.06
BAGEL	3.56	3.31	1.70	3.30	2.62	3.24	4.49	2.38	4.17	3.20
UniWorld-V1	3.82	3.64	2.27	3.47	3.24	2.99	4.21	2.96	2.74	3.26
Bridge (Ours)	3.49	2.64	2.93	3.45	3.48	3.45	4.14	3.09	3.85	3.39

6.4.1 Main Results

Visual Understanding. We evaluate Bridge on a suite of visual understanding benchmarks, including POPE [270], MME [271], MMBench [272], Seed-Bench [273], and MMMU [274]. As reported in Table 6.1, thanks to the strong visual understanding ability inherited from the base MLLM, Bridge achieves state-of-the-art or near state-of-the-art results across these benchmarks. Notably, even on datasets where Bridge does not rank first, its performance remains very close to the best reported results, demonstrating the robustness of its inherited understanding capacity.

Text-to-image Generation. We evaluate visual generation performance on three commonly used text-to-image benchmarks, including GenEval [275], DPG Benchmark [276], and WISE [277]. Results are reported in Table 6.2. Our Bridge achieves consistently strong results across all three benchmarks. On DPG Bench, it obtains an overall score of 85.51, surpassing most prior unified model and generation-only methods. On GenEval, Bridge achieves 0.82 overall score, which is competitive with recent state-of-the-art unified models like BAGEL. On WISE, our model delivers a solid 0.69, reaching the best reported numbers (0.70) of unified models. These results demonstrate that Bridge achieves highly competitive performance across diverse evaluation settings.

Table 6.4: **Visual Representation Comparison.** Sem: Semantic token. Pix: Pixel token.

#Sem.	#Pix.	GenEval	DPG	ImgEdit
0	1024	0.48	71.7	2.93
729	0	0.57	73.1	3.16
729	1024	0.60	73.4	3.35
81	0	0.56	73.7	2.82
81	1024	0.61	75.8	<u>3.33</u>

Table 6.5: **MLLM Architecture Comparison.** Und. denotes the harmonic mean of understanding benchmarks including MME, MMBench, SEED Bench, POPE, and MMMU.

Arch.	Und.	GenEval	DPG
Dense	90.1	0.53	77.2
MoT	108.0	0.63	76.3

Instructional Image Editing. We leverage the ImgEdit benchmark [278] to systematically evaluate the instructional image editing performance of Bridge. Comparisons are made against a series of specialist editing models [169, 268, 279, 280] as well as unified MLLMs [231, 244]. Table 6.3 summarizes the results. As shown, Bridge not only surpasses the leading editing specialist model Step1X-Edit [280], but also outperforms recent unified MLLMs such as BAGEL [231] and UniWorld-V1 [244]. In particular, Bridge achieves significant gains on the *Extract*, *Remove*, *Background*, and *Hybrid* categories, underscoring its ability to follow fine-grained instructions while preserving global consistency.

Visualization. As shown in Figure 6.2, our model can generate a wide range of images including cartoon-style scenes, fantasy illustrations, photorealistic portraits, stylized characters and animals *etc.* In addition, it also supports image editing tasks such as style transfer, object remove or addition, and replacement.

Table 6.6: **Token Routing Comparison.** Visual understanding and generation performance under different token routing schemes. In the Token Routing columns, **Und.** refers to the understanding expert and **Gen.** refers to the generation expert.

Token Routing		Und. Benchmarks						Gen. Benchmarks	
Und. Img	Text	MMBench	MME-P	MME-C	SEED	POPE	MMMU	GenEval	DPG
Und.	Und.	84.3	1730	677	77.4	88.4	57.4	0.63	76.3
Gen.	Und.	80.9	1621	541	71.0	88.6	52.6	0.60	75.6
Und.	Gen.	79.0	1479	564	75.2	85.3	46.0	0.59	77.1
Gen.	Gen.	74.5	1228	475	71.4	80.2	42.1	0.53	77.2

6.4.2 Ablation Studies

In this section, we present ablation studies to examine several key design choices of our method. Unless otherwise noted, all experiments use 25M high-quality text-to-image samples to establish the visual generation capability of the model. For ablations that also require training on visual understanding and language modeling, we additionally incorporate image understanding and text-only data, maintaining a sampling ratio of 6 : 2 : 1 across text-to-image, image understanding, and text data.

Effectiveness and Efficiency of Semantic-to-pixel Visual Representation. In Bridge, visual representations are constructed by sequentially combining semantic tokens with pixel tokens. To assess the effectiveness of this design, we compare several alternatives, including using only pixel tokens, only semantic tokens, or varying semantic sequence lengths. Results are shown in Table 6.4, where #Sem. denotes the number of semantic tokens and #Pix. denotes the number of pixel tokens. All models are trained for one epoch on text-to-image datasets, and further continued on 1.4M high-quality image editing samples. For variants without pixel tokens, generation terminates after all semantic tokens are sampled, and images are decoded with TA-Tok’s [234] autoregressive de-tokenizer. Evaluation is performed using GenEval [275] and DPG [276] for

text-to-image generation, and ImgEdit [278] for image editing.

The results reveal several key trends. First, using only pixel tokens yields the poorest scores, highlighting their limited alignment with language tokens. Second, incorporating semantic tokens consistently improves performance, demonstrating their role in bridging the gap between language and vision. Finally, when comparing variants that use the same number of semantic tokens, our approach of natively generating pixel tokens outperforms those that rely on external generative de-tokenizers on all text-to-image and editing benchmarks, with the advantage becoming more pronounced when the semantic sequence is short (e.g., 81 tokens). This configuration not only achieves the best overall performance but also increases the sequence length by only $\sim 7.9\%$, indicating that a small set of semantic tokens is necessary and sufficient for effective in-context cross-modal alignment.

Effectiveness of the MoT Architecture. Bridge adopts a Mixture-of-Transformers (MoT) architecture to develop visual generation ability on top of pre-trained understanding-oriented MLLMs. To assess the necessity of this design, we compare it with a straightforward dense architecture, where a single transformer backbone inherited from the same pre-trained MLLM is continually trained on a mixture of text-to-image, image understanding, and pure language data. As shown in Table 6.5, after continued training, the dense variant achieves a slightly higher DPG score but performs significantly worse on GenEval and multiple understanding benchmarks. This result suggests that generative training inevitably degrades the visual understanding ability of the base MLLM, whereas the MoT architecture effectively isolates and preserves understanding while adding generative capacity, validating the importance of our design.

Token Routing Scheme. In Bridge, visual generation ability is introduced by routing newly

added generation image tokens to the generation expert, while all understanding image tokens and text tokens remain routed to the understanding expert. This design preserves the original comprehension capability of the base MLLM while extending it with generative capacity. To examine alternative routing strategies, we experiment with all combinations of routing for understanding image tokens and text tokens, and report results in Table 6.6. Note that generation image tokens are always routed to the generation expert, since they cannot be modeled by the frozen understanding expert. For cases where understanding image tokens are assigned to the generation expert, they are still encoded by the inherited continuous vision encoder to ensure consistency.

The results highlight several key trends. Our default routing scheme (row 1), where only generation tokens are handled by the generation expert, achieves the best overall balance: strong visual generation performance and excellent visual understanding ability. Moving understanding tokens into the generation expert (row 2) causes interference between semantic and pixel representations, leading to degradation in both understanding and generation benchmarks. Routing text tokens to the generation expert (row 3) is counterintuitive and unsurprisingly results in worse visual understanding. Finally, sending all tokens to the generation expert (row 4) collapses to the dense model setting, where limited capacity yields the weakest visual understanding performance. Interestingly, this variant also achieves the lowest GenEval score but slightly outperforms other schemes on DPG.

In summary, keeping understanding image tokens and text tokens routed to the understanding expert, while restricting the generation expert to newly introduced generation tokens, delivers the best overall trade-off between visual understanding and generation.

6.5 Conclusion

In this chapter, we present Bridge, a pure autoregressive unified MLLM that augments pre-trained models with visual generative capability while rigorously preserving their visual understanding, enabling seamless multimodal understanding and generation within a single next-token prediction paradigm. Central to our approach is a semantic-to-pixel discrete visual representation that fuses compact high-level semantic tokens with fine-grained pixel tokens, achieving strong language alignment and high-fidelity synthesis with only a modest increase in sequence length. Extensive experiments across diverse benchmarks demonstrate competitive or superior performance in both understanding and generation, alongside reduced data requirements, shorter training time, and overall improved efficiency compared to prior unified MLLMs.

Chapter 7: Future Work and Conclusion

7.1 Future Work

The research presented in this dissertation establishes a foundation for controllable, efficient, and unified visual content generation. Building on these contributions, we outline a clear path forward. Our future agenda focuses on tighter integration between low-level generative representations and high-level multimodal reasoning to develop systems that better meet the real-world demands of visual content generation, while improving capability and robustness.

One key direction is to further bridge the objective gap between visual tokenization and generative modeling. As demonstrated in our work with LARP, tokenizers optimized solely for reconstruction fidelity can yield latent spaces that are suboptimal for downstream generation. To address this misalignment, we propose integrating a learned diffusion prior directly into the tokenizer training process. By jointly training a lightweight diffusion model alongside the tokenizer and backpropagating the denoising loss through the encoder, we can encourage a latent space that is inherently compatible with iterative generation. This approach goes beyond pixel-level reconstruction by ensuring that the fundamental units of visual representation are optimized for the generative tasks they are intended to support.

This stronger representational foundation motivates the next step in system design: the development of truly unified Multimodal Large Language Models (MLLMs). Although recent

unified models [231–238] have made progress toward combining understanding and generation, they often rely on loosely coupled components, which limits cross-modal synergy and robustness. Future work should develop multimodal tokenizers and training strategies that enable understanding and generation to actively reinforce each other, rather than operating as parallel capabilities. Achieving meaningful unification also requires expanding beyond 2D static vision to a broader set of modalities, including video, 3D, audio, and touch, which will be essential for building more general-purpose systems. Finally, to ensure these powerful unified models are safe and dependable in real-world settings, robust post-training and reinforcement learning methods will be necessary to align model behavior with human values.

7.2 Conclusion

This dissertation examined visual content generation across a spectrum of abstraction levels, from low-level pixel organization to high-level semantic synthesis. By analyzing generation through these complementary perspectives, we developed principled approaches that improve control, fidelity, and generalization in modern computer vision systems. We began by addressing the foundational representation of visual data through Neural Space-filling Curves, showing that data-driven, context-aware pixel orderings substantially outperform universal scans on downstream tasks such as compression. This establishes that even the choice of pixel traversal can meaningfully influence efficiency.

Advancing toward semantic manipulation, we introduced the Chop & Learn framework to study compositional generalization. Through its structured object-state design, the benchmark enables the synthesis of novel combinations and demonstrates how curated, factorized data sup-

ports transferable visual reasoning. To enhance controllability in image generation, we proposed a unified framework for multimodality-guided style transfer. Supported by cross-modal GAN inversion, this approach enables precise style manipulation using arbitrary mixtures of text and image references, overcoming the inherent limitations of single-modality guidance.

Extending from static images to temporal signals, we developed LARP, a video tokenizer equipped with a learned autoregressive prior. Its autoregressive-aligned tokenization scheme optimizes the latent space specifically for generation, effectively bridging the divide between video compression and high-fidelity synthesis. Finally, we unified visual understanding and generation within Multimodal Large Language Models through the Bridge framework. By combining a semantic-to-pixel discrete representation with a Mixture-of-Transformers architecture, Bridge equips pre-trained understanding-focused models with visual generative capabilities while preserving their interpretive strengths.

Collectively, these contributions advance the ability to synthesize diverse forms of visual content. They demonstrate that integrating learned priors, structured representations, and unified generative–understanding architectures is central to the continued evolution of generative AI.

Bibliography

- [1] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, et al. Language model beats diffusion—tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.
- [2] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [3] Ahmad Darkhalil, Dandan Shan, Bin Zhu, Jian Ma, Amlan Kar, Richard E. L. Higgins, Sanja Fidler, David F. Fouhey, and Dima Damen. Epic-kitchens visor benchmark: Video segmentations and object relations. *ArXiv*, abs/2209.13064, 2022.
- [4] Luowei Zhou, Nathan Louis, and Jason J. Corso. Weakly-supervised video object grounding from text by loss weighting and object interaction. In *British Machine Vision Conference*, 2018.
- [5] Phillip Isola, Joseph J. Lim, and E. Adelson. Discovering states and transformations in image collections. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1383–1391, 2015.
- [6] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *ArXiv*, abs/2102.12092, 2021.
- [7] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [8] Junke Wang, Yi Jiang, Zehuan Yuan, Binyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. *arXiv preprint arXiv:2406.09399*, 2024.
- [9] David Hilbert. Über die stetige abbildung einer linie auf ein flächenstück. In *Dritter Band: Analysis· Grundlagen der Mathematik· Physik Verschiedenes*, pages 1–2. Springer, 1935.
- [10] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

- [11] Ian H Witten, Radford M Neal, and John G Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.
- [12] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.
- [13] Giuseppe Peano. Sur une courbe, qui remplit toute une aire plane. *Mathematische Annalen*, 36(1):157–160, 1890.
- [14] Bongki Moon, Hosagrahar V Jagadish, Christos Faloutsos, and Joel H. Saltz. Analysis of the clustering properties of the hilbert space-filling curve. *IEEE Transactions on knowledge and data engineering*, 13(1):124–141, 2001.
- [15] Seiichiro Kamata, Richard O Eason, and Eiji Kawaguchi. An implementation of the hilbert scanning algorithm and its application to data compression. *IEICE TRANSACTIONS on Information and Systems*, 76(4):420–428, 1993.
- [16] A Ansari and A Fineberg. Image data compression and ordering using peano scan and lot. *IEEE Trans. on Consumer Electronics*, 38(3):436–445, 1992.
- [17] KS Thyagarajan and S Chatterjee. Fractal scanning for image compression. In *Conference Record of the Twenty-Fifth Asilomar Conference on Signals, Systems & Computers*, pages 467–468. IEEE Computer Society, 1991.
- [18] Revital Dafner, Daniel Cohen-Or, and Yossi Matias. Context-based space filling curves. In *Computer Graphics Forum*, volume 19, pages 209–218. Wiley Online Library, 2000.
- [19] Wacław Sierpiński. Sur une nouvelle courbe continue qui remplit toute une aire plane. *Bull. Acad. Sci. Cracovie (Sci. math. et nat. Serie A)*, pages 462–478, 1912.
- [20] Eliakim Hastings Moore. On certain crinkly curves. *Transactions of the American Mathematical Society*, 1(1):72–90, 1900.
- [21] Abraham Lempel and Jacob Ziv. Compression of two-dimensional data. *IEEE transactions on information theory*, 32(1):2–8, 1986.
- [22] Michael Bader. *Space-filling curves: an introduction with applications in scientific computing*, volume 9. Springer Science & Business Media, 2012.
- [23] VV Alexandrov, AI Alexeev, and ND Gorsky. A recursive algorithm for pattern recognition. *Proc. IEEE Intl. Conf. Pattern Recognition*, pages 431–433, 1982.
- [24] Jia-Hong Lee and Yuang-Cheh Hsueh. Texture classification method using multiple space filling curves. *Pattern Recognition Letters*, 15(12):1241–1244, 1994.
- [25] Yossi Matias and Adi Shamir. A video scrambling technique based on space filling curves. In *Conference on the theory and application of cryptographic techniques*, pages 398–417. Springer, 1987.

- [26] Jonathan K Lawder. Calculation of mappings between one and n-dimensional values using the hilbert space-filling curve. *School of Computer Science and Information Systems, Birkbeck College, University of London, London Research Report BBKCS-00-01 August, 2000.*
- [27] Jinhui Zhu, Ahmad Hoorfar, and Nader Engheta. Bandwidth, cross-polarization, and feed-point characteristics of matched hilbert antennas. *IEEE Antennas and Wireless Propagation Letters*, 2:2–5, 2003.
- [28] Erez Lieberman-Aiden, Nynke L Van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R Lajoie, Peter J Sabo, Michael O Dorschner, et al. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *science*, 326(5950):289–293, 2009.
- [29] Randall Munroe. xkcd: Map of the internet. <https://xkcd.com/195>, 2006-12-11. Accessed: 2021-11-16.
- [30] Maciej Drozdowski. *Scheduling for parallel processing*, volume 18. Springer, 2009.
- [31] Andy B Yoo, Morris A Jette, and Mark Grondona. Slurm: Simple linux utility for resource management. In *Workshop on job scheduling strategies for parallel processing*, pages 44–60. Springer, 2003.
- [32] Tarek Ouni, Arij Lassoued, and Mohamed Abid. Gradient-based space filling curves: Application to lossless image compression. In *2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pages 437–442. IEEE, 2011.
- [33] Liang Zhou, Chris R Johnson, and Daniel Weiskopf. Data-driven space-filling curves. *IEEE transactions on visualization and computer graphics*, 27(2):1591–1600, 2020.
- [34] John J Hopfield and David W Tank. “neural” computation of decisions in optimization problems. *Biological cybernetics*, 52(3):141–152, 1985.
- [35] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *NIPS*, 2015.
- [36] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *ICLR*, 2015.
- [37] Michel Deudon, Pierre Cournut, Alexandre Lacoste, Yossiri Adulyasak, and Louis-Martin Rousseau. Learning Heuristics for the TSP by Policy Gradient. In *International conference on the integration of constraint programming, artificial intelligence, and operations research*, pages 170–181. Springer, 2018.
- [38] Hanjun Dai, Elias B Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *NIPS*, 2017.
- [39] Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! *ICLR*, 2019.

- [40] Frank Harary and Robert Z Norman. Some properties of line digraphs. *Rendiconti del Circolo Matematico di Palermo*, 9(2):161–168, 1960.
- [41] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2016.
- [42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.
- [43] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding. *IEEE transactions on Information Theory*, 24(5):530–536, 1978.
- [44] Terry A. Welch. Technique for high-performance data compression. *Computer*, 1984.
- [45] Robert Clay Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957.
- [46] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [47] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [48] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.
- [49] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- [50] Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. Tgif: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4641–4650, 2016.
- [51] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *NIPS*, 2017.
- [52] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.
- [53] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *ICML*, 2021.
- [54] Max Ehrlich, Jon Barker, Namitha Padmanabhan, Larry Davis, Andrew Tao, Bryan Catanzaro, and Abhinav Shrivastava. Leveraging bitstream metadata for fast and accurate video compression correction. *arXiv preprint arXiv:2202.00011*, 2022.

- [55] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021.
- [56] Noam Chomsky. A minimalist program for linguistic theory. *The view from building*, 20, 1993.
- [57] Kate Arnold and Klaus Zuberbühler. Semantic combinations in primate calls. *Nature*, 441(7091):303–303, 2006.
- [58] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.
- [59] A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 192–199, 2014.
- [60] Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2021.
- [61] Nirat Saini, Khoi Pham, and Abhinav Shrivastava. Disentangling visual embeddings for attributes and objects. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13648–13657, 2022.
- [62] Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. Look for the change: Learning object states and state-modifying actions from untrimmed web videos. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13936–13946, 2022.
- [63] Tomáš Souček, Jean-Baptiste Alayrac, Antoine Miech, Ivan Laptev, and Josef Sivic. Multi-task learning of object state changes from uncurated videos. *ArXiv*, abs/2211.13500, 2022.
- [64] Nirat Saini, Bo He, Gaurav Shrivastava, Sai Saketh Rambhatla, and Abhinav Shrivastava. Recognizing actions using object states. In *ICLR2022 Workshop on the Elements of Reasoning: Objects, Structure and Causality*, 2022.
- [65] Jean-Baptiste Alayrac, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Joint discovery of object states and manipulation actions. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2146–2155, 2017.
- [66] Alireza Fathi and James M Rehg. Modeling actions through state changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2586, 2013.
- [67] Nachwa Abou Bakr, James L. Crowley, and Rémi Ronfard. Recognizing manipulation actions from state-transformations. *ArXiv*, abs/1906.05147, 2019.

- [68] Bo He, Xitong Yang, Zuxuan Wu, Hao Chen, Ser-Nam Lim, and Abhinav Shrivastava. Gta: Global temporal attention for video action understanding. *British Machine Vision Conference (BMVC)*, 2021.
- [69] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022.
- [70] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.
- [71] Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *CVPR*, 2015.
- [72] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1207–1216, 2019.
- [73] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Q. Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh K. Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Z. Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Christian Fuegen, Abrahm Gebreselasie, Cristina González, James M. Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Yu Heng Khoo, Jáchym Kolár, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran K. Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Yunyi Zhu, Pablo Arbeláez, David J. Crandall, Dima Damen, Giovanni Maria Farinella, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard A. Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the world in 3,000 hours of egocentric video. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18973–18990, 2021.
- [74] Sebastian Stein and Stephen J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, page 729–738, 2013.
- [75] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3537–3545, 2019.

- [76] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of Computer Vision and Pattern Recognition Conference (CVPR)*, 2014.
- [77] Joanna Materzynska, Tete Xiao, Roei Herzig, Huijuan Xu, Xiaolong Wang, and Trevor Darrell. Something-else: Compositional action recognition with spatial-temporal interaction networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1046–1056, 2019.
- [78] Genevieve Patterson and James Hays. Coco attributes: Attributes for people, animals, and objects. In *ECCV*, 2016.
- [79] Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Learning graph embeddings for open world compositional zero-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 2022.
- [80] Khoi Pham, Kushal Kafle, Zhe Lin, Zhi Ding, Scott D. Cohen, Quan Tran, and Abhinav Shrivastava. Learning to predict visual attributes in the wild. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13013–13023, 2021.
- [81] Tushar Nagarajan and K. Grauman. Attributes as operators: Factorizing unseen attribute-object compositions. In *ECCV*, 2018.
- [82] Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc’Aurelio Ranzato. Task-driven modular networks for zero-shot compositional learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3592–3601, 2019.
- [83] Muli Yang, Cheng Deng, Junchi Yan, Xianglong Liu, and Dacheng Tao. Learning unseen concepts via hierarchical decomposition and composition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10245–10253, 2020.
- [84] Yong-Lu Li, Yue Xu, Xiaohan Mao, and Cewu Lu. Symmetry and group in attribute-object compositions. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11313–11322, 2020.
- [85] Ziwei Xu, Guangzhi Wang, Yongkang Wong, and Mohan S Kankanhalli. Relation-aware compositional zero-shot learning for attribute-object pair recognition. *IEEE Transactions on Multimedia*, 2021.
- [86] Khoi Pham, Kushal Kafle, Zhe Lin, Zhihong Ding, Scott Cohen, Quan Tran, and Abhinav Shrivastava. Improving closed and open-vocabulary attribute prediction using transformers. In *European Conference on Computer Vision*, pages 201–219. Springer, 2022.
- [87] Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Constrained semi-supervised learning using attributes and comparative attributes. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part III 12*, pages 369–383. Springer, 2012.

- [88] Xin Wang, Fisher Yu, Trevor Darrell, and Joseph E Gonzalez. Task-aware feature generation for zero-shot compositional learning. *arXiv preprint arXiv:1906.04854*, 2019.
- [89] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.
- [90] Tian Yun, Usha Bhalla, Elizabeth-Jane Pavlick, and Chen Sun. Do vision-language pre-trained models learn primitive concepts? *ArXiv*, abs/2203.17271, 2022.
- [91] Moustafa Meshry, Yixuan Ren, Larry S Davis, and Abhinav Shrivastava. Step: Style-based encoder pre-training for multi-modal image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [92] Dave Epstein, Jiajun Wu, Cordelia Schmid, and Chen Sun. Learning temporal dynamics from cycles in narrated video. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1460–1469, 2021.
- [93] Dinesh Jayaraman, Frederik Ebert, Alexei A. Efros, and Sergey Levine. Time-agnostic prediction: Predicting predictable video frames. *ArXiv*, abs/1808.07784, 2018.
- [94] Gorjan Radevski, Marie-Francine Moens, and Tinne Tuytelaars. Revisiting spatio-temporal layouts for compositional action recognition. *British Machine Vision Conference (BMVC)*, abs/2111.01936, 2021.
- [95] X. Wang and Abhinav Kumar Gupta. Videos as space-time region graphs. *ArXiv*, abs/1806.01810, 2018.
- [96] Tae Soo Kim and Gregory Hager. Safcar: Structured attention fusion for compositional action recognition. *ArXiv*, abs/2012.02109, 2020.
- [97] Jian Ma and Dima Damen. Hand-object interaction reasoning. *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8, 2022.
- [98] Rui Yan, Peng Huang, Xiangbo Shu, Junhao Zhang, Yonghua Pan, and Jinhui Tang. Look less think more: Rethinking compositional action recognition. *Proceedings of the 30th ACM International Conference on Multimedia*, 2022.
- [99] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. *ArXiv*, abs/1804.02748, 2018.
- [100] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2630–2640, 2019.

- [101] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.
- [102] Hero9 black 5k video 20mp photo streaming camera and bundles. <https://gopro.com/en/us/shop/cameras/hero9-black/CHDHX-901-master.html>.
- [103] 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*, pages 10684–10695, 2022.
- [104] 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.
- [105] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
- [106] Miranda Dixon-Luinenburg. What dall-e 2 can and cannot do, 2023.
- [107] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [108] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.
- [109] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.
- [110] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, 2017.
- [111] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [112] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [113] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9876–9886, 2019.

- [114] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3154–3160, 2017.
- [115] Hirokatsu Kataoka, Tenga Wakamiya, Kensho Hara, and Yutaka Satoh. Would mega-scale datasets further enhance spatiotemporal 3d cnns? *arXiv preprint arXiv:2004.04968*, 2020.
- [116] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.
- [117] Bo He, Xitong Yang, Hanyu Wang, Zuxuan Wu, Hao Chen, Shuaiyi Huang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. Towards scalable neural representation for diverse videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- [118] Bo He, Xitong Yang, Le Kang, Zhiyu Cheng, Xin Zhou, and Abhinav Shrivastava. Asmloc: Action-aware segment modeling for weakly-supervised temporal action localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [119] Bo He, Jun Wang, Jieli Qiu, Trung Bui, Abhinav Shrivastava, and Zhaowen Wang. Align and attend: Multimodal summarization with dual contrastive losses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14867–14878, 2023.
- [120] Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 160–169, 2020.
- [121] Noureldien Hussein, Efstratios Gavves, and Arnold W. M. Smeulders. Videograph: Recognizing minutes-long human activities in videos. *ArXiv*, abs/1905.05143, 2019.
- [122] Donglai Wei, Joseph J. Lim, Andrew Zisserman, and William T. Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [123] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.
- [124] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.
- [125] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.
- [126] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3809–3817, 2019.

- [127] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5880–5888, 2019.
- [128] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4369–4376, Apr. 2020.
- [129] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6649–6658, 2021.
- [130] Haibo Chen, Zhizhong Wang, Huiming Zhang, Zhiwen Zuo, Ailin Li, Wei Xing, Dongming Lu, et al. Artistic style transfer with internal-external learning and contrastive learning. *Advances in Neural Information Processing Systems*, 34:26561–26573, 2021.
- [131] Zijie Wu, Zhen Zhu, Junping Du, and Xiang Bai. Ccpl: contrastive coherence preserving loss for versatile style transfer. In *European Conference on Computer Vision*, pages 189–206. Springer, 2022.
- [132] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Bjorn Ommer. A style-aware content loss for real-time hd style transfer. In *proceedings of the European conference on computer vision (ECCV)*, pages 698–714, 2018.
- [133] Yuxin Zhang, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, Tong-Yee Lee, and Changsheng Xu. Domain enhanced arbitrary image style transfer via contrastive learning. In *ACM SIGGRAPH*, pages 1–8, 2022.
- [134] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18062–18071, 2022.
- [135] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [136] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Style-clip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021.
- [137] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022.
- [138] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021.

- [139] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. *arXiv preprint arXiv:1603.03417*, 2016.
- [140] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6924–6932, 2017.
- [141] Xin Wang, Geoffrey Oxholm, Da Zhang, and Yuan-Fang Wang. Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5239–5247, 2017.
- [142] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1897–1906, 2017.
- [143] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.
- [144] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Diversified texture synthesis with feed-forward networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3920–3928, 2017.
- [145] Kibeom Hong, Seogkyu Jeon, Huan Yang, Jianlong Fu, and Hyeran Byun. Domain-aware universal style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14609–14617, 2021.
- [146] Dmytro Kotovenko, Matthias Wright, Arthur Heimbrecht, and Bjorn Ommer. Rethinking style transfer: From pixels to parameterized brushstrokes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12196–12205, 2021.
- [147] Hao Dong, Simiao Yu, Chao Wu, and Yike Guo. Semantic image synthesis via adversarial learning. In *Proceedings of the IEEE international conference on computer vision*, pages 5706–5714, 2017.
- [148] Seonghyeon Nam, Yunji Kim, and Seon Joo Kim. Text-adaptive generative adversarial networks: manipulating images with natural language. *Advances in neural information processing systems*, 31, 2018.
- [149] 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 international conference on computer vision*, pages 5907–5915, 2017.
- [150] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan++: Realistic image synthesis with stacked generative

- adversarial networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1947–1962, 2018.
- [151] 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 conference on computer vision and pattern recognition*, pages 1316–1324, 2018.
 - [152] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip HS Torr. Manigan: Text-guided image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7880–7889, 2020.
 - [153] Weihao Xia, Yujiu Yang, Jing-Hao Xue, and Baoyuan Wu. Tedigan: Text-guided diverse face image generation and manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2256–2265, 2021.
 - [154] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.
 - [155] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.
 - [156] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. *arXiv preprint arXiv:2204.08583*, 2, 2022.
 - [157] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.
 - [158] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.
 - [159] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.
 - [160] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.
 - [161] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.
 - [162] Zhi-Song Liu, Li-Wen Wang, Wan-Chi Siu, and Vicky Kalogeiton. Name your style: An arbitrary artist-aware image style transfer. *arXiv preprint arXiv:2202.13562*, 2022.

- [163] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4432–4441, 2019.
- [164] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.
- [165] Yuval Alaluf, Omer Tov, Ron Mokady, Rinon Gal, and Amit Bermano. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18511–18521, 2022.
- [166] 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 conference on computer vision and pattern recognition*, pages 586–595, 2018.
- [167] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [168] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023.
- [169] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.
- [170] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- [171] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.
- [172] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [173] Gemini Team Google, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [174] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

- [175] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [176] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.
- [177] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.
- [178] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.
- [179] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [180] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.
- [181] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [182] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.
- [183] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In *European Conference on Computer Vision*, pages 102–118. Springer, 2022.
- [184] Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Rachel Hornung, Hartwig Adam, Hassan Akbari, Yair Alon, Vighnesh Birodkar, et al. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023.
- [185] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In *The Eleventh International Conference on Learning Representations*, 2022.
- [186] Sipeng Zheng, Bohan Zhou, Yicheng Feng, Ye Wang, and Zongqing Lu. Unicode: Learning a unified codebook for multimodal large language models. *arXiv preprint arXiv:2403.09072*, 2024.

- [187] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022.
- [188] Jiahui Zhang, Fangneng Zhan, Christian Theobalt, and Shijian Lu. Regularized vector quantization for tokenized image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18467–18476, 2023.
- [189] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *arXiv preprint arXiv:2406.07550*, 2024.
- [190] Yuchao Gu, Xintao Wang, Yixiao Ge, Ying Shan, and Mike Zheng Shou. Rethinking the objectives of vector-quantized tokenizers for image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7631–7640, 2024.
- [191] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2020.
- [192] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.
- [193] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.
- [194] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.
- [195] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.
- [196] K Soomro. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.
- [197] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixcnn decoders. *Advances in neural information processing systems*, 29, 2016.
- [198] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

- [199] Shiyue Cao, Yueqin Yin, Lianghua Huang, Yu Liu, Xin Zhao, Deli Zhao, and Kaigi Huang. Efficient-vqgan: Towards high-resolution image generation with efficient vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7368–7377, 2023.
- [200] Chuanxia Zheng, Tung-Long Vuong, Jianfei Cai, and Dinh Phung. Movq: Modulating quantized vectors for high-fidelity image generation. *Advances in Neural Information Processing Systems*, 35:23412–23425, 2022.
- [201] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.
- [202] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.
- [203] Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, et al. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10459–10469, 2023.
- [204] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual descriptions. In *International Conference on Learning Representations*, 2022.
- [205] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3626–3636, 2022.
- [206] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning*, pages 1747–1756. PMLR, 2016.
- [207] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.
- [208] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [209] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.
- [210] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

- [211] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
- [212] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.
- [213] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.
- [214] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.
- [215] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.
- [216] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science*. <https://cdn.openai.com/papers/dall-e-3.pdf>, 2(3):8, 2023.
- [217] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.
- [218] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. *Technical Report*, 2024.
- [219] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.
- [220] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.
- [221] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.
- [222] Tsvetomila Mihaylova and André FT Martins. Scheduled sampling for transformers. *arXiv preprint arXiv:1906.07651*, 2019.
- [223] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.
- [224] Zhengxiong Luo, Dayou Chen, Yingya Zhang, Yan Huang, Liang Wang, Yujun Shen, Deli Zhao, Jingren Zhou, and Tieniu Tan. Videofusion: Decomposed diffusion models for high-quality video generation. *arXiv preprint arXiv:2303.08320*, 2023.

- [225] Ivan Skorokhodov, Willi Menapace, Aliaksandr Siarohin, and Sergey Tulyakov. Hierarchical patch diffusion models for high-resolution video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7569–7579, 2024.
- [226] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.
- [227] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024.
- [228] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.
- [229] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.
- [230] Ethan Chern, Zhulin Hu, Steffi Chern, Siqi Kou, Jiadi Su, Yan Ma, Zhijie Deng, and Pengfei Liu. Thinking with generated images. *arXiv preprint arXiv:2505.22525*, 2025.
- [231] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.
- [232] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.
- [233] Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved native unified multimodal models. *arXiv preprint arXiv:2506.15564*, 2025.
- [234] Jiaming Han, Hao Chen, Yang Zhao, Hanyu Wang, Qi Zhao, Ziyan Yang, Hao He, Xiangyu Yue, and Lu Jiang. Vision as a dialect: Unifying visual understanding and generation via text-aligned representations. *arXiv preprint arXiv:2506.18898*, 2025.
- [235] Yecheng Wu, Zhuoyang Zhang, Junyu Chen, Haotian Tang, Dacheng Li, Yunhao Fang, Ligeng Zhu, Enze Xie, Hongxu Yin, Li Yi, et al. Vila-u: a unified foundation model integrating visual understanding and generation. *arXiv preprint arXiv:2409.04429*, 2024.
- [236] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yuezhe Wang, Zhen Li, Qiyang Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

- [237] Chuofan Ma, Yi Jiang, Junfeng Wu, Jihan Yang, Xin Yu, Zehuan Yuan, Bingyue Peng, and Xiaojuan Qi. Unitok: A unified tokenizer for visual generation and understanding. *arXiv preprint arXiv:2502.20321*, 2025.
- [238] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.
- [239] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.
- [240] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7739–7751, 2025.
- [241] Chunwei Wang, Guansong Lu, Junwei Yang, Runhui Huang, Jianhua Han, Lu Hou, Wei Zhang, and Hang Xu. Illume: Illuminating your llms to see, draw, and self-enhance. *arXiv preprint arXiv:2412.06673*, 2024.
- [242] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.
- [243] Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12966–12977, 2025.
- [244] Bin Lin, Zongjian Li, Xinhua Cheng, Yuwei Niu, Yang Ye, Xianyi He, Shenghai Yuan, Wangbo Yu, Shaodong Wang, Yunyang Ge, et al. Uniworld: High-resolution semantic encoders for unified visual understanding and generation. *arXiv preprint arXiv:2506.03147*, 2025.
- [245] Xichen Pan, Satya Narayan Shukla, Aashu Singh, Zhuokai Zhao, Shlok Kumar Mishra, Jialiang Wang, Zhiyang Xu, Jiuhai Chen, Kunpeng Li, Felix Juefei-Xu, et al. Transfer between modalities with metaqueries. *arXiv preprint arXiv:2504.06256*, 2025.
- [246] Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ze Liu, Ziyi Xia, Chaofan Li, Haoze Deng, Jiahao Wang, Kun Luo, Bo Zhang, Defu Lian, Xinlong Wang, Zhongyuan Wang, Tiejun Huang, and Zheng Liu. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025.
- [247] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

- [248] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025.
- [249] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.
- [250] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.
- [251] Weixin Liang, Lili Yu, Liang Luo, Srinivasan Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen-tau Yih, Luke Zettlemoyer, et al. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *arXiv preprint arXiv:2411.04996*, 2024.
- [252] Weijia Shi, Xiaochuang Han, Chunting Zhou, Weixin Liang, Xi Victoria Lin, Luke Zettlemoyer, and Lili Yu. Lmfusion: Adapting pretrained language models for multimodal generation. *arXiv preprint arXiv:2412.15188*, 2024.
- [253] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.
- [254] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- [255] Junying Chen, Zhenyang Cai, Pengcheng Chen, Shunian Chen, Ke Ji, Xidong Wang, Yunjin Yang, and Benyou Wang. Sharegpt-4o-image: Aligning multimodal models with gpt-4o-level image generation, 2025.
- [256] Junyan Ye, Dongzhi Jiang, Zihao Wang, Leqi Zhu, Zhenghao Hu, Zilong Huang, Jun He, Zhiyuan Yan, Jinghua Yu, Hongsheng Li, et al. Echo-4o: Harnessing the power of gpt-4o synthetic images for improved image generation. *arXiv preprint arXiv:2508.09987*, 2025.
- [257] Yuhan Wang, Siwei Yang, Bingchen Zhao, Letian Zhang, Qing Liu, Yuyin Zhou, and Cihang Xie. Gpt-image-edit-1.5m: A million-scale, gpt-generated image dataset, 2025.
- [258] Yu Gao, Lixue Gong, Qiushan Guo, Xiaoxia Hou, Zhichao Lai, Fanshi Li, Liang Li, Xiaochen Lian, Chao Liao, Liyang Liu, et al. Seedream 3.0 technical report. *arXiv preprint arXiv:2504.11346*, 2025.

- [259] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [260] DrawThingsAI. Megalith-10m dataset. <https://huggingface.co/datasets/drawthingsai/megalith-10m>, 2024. Accessed: 2025-05-09.
- [261] Keqiang Sun, Junting Pan, Yuying Ge, Hao Li, Haodong Duan, Xiaoshi Wu, Renrui Zhang, Aojun Zhou, Zipeng Qin, Yi Wang, et al. Journeydb: A benchmark for generative image understanding. *Advances in neural information processing systems*, 36:49659–49678, 2023.
- [262] Ben Egan, Alex Redden, XWAVE, and SilentAntagonist. Dalle3 1 Million+ High Quality Captions, May 2024.
- [263] Qingyun Li, Zhe Chen, Weiyun Wang, Wenhai Wang, Shenglong Ye, Zhenjiang Jin, Guanzhou Chen, Yinan He, Zhangwei Gao, Erfei Cui, et al. Omnicorpus: A unified multimodal corpus of 10 billion-level images interleaved with text. *arXiv preprint arXiv:2406.08418*, 2024.
- [264] Leigang Qu, Feng Cheng, Ziyan Yang, Qi Zhao, Shanchuan Lin, Yichun Shi, Yicong Li, Wenjie Wang, Tat-Seng Chua, and Lu Jiang. Vincie: Unlocking in-context image editing from video. *arXiv preprint arXiv:2506.10941*, 2025.
- [265] Shuhao Gu, Jialing Zhang, Siyuan Zhou, Kevin Yu, Zhaohu Xing, Liangdong Wang, Zhou Cao, Jintao Jia, Zhuoyi Zhang, Yixuan Wang, Zhenchong Hu, Bo-Wen Zhang, Jijie Li, Dong Liang, Yingli Zhao, Yulong Ao, Yaoqi Liu, Fangxiang Feng, and Guang Liu. Infinity-mm: Scaling multimodal performance with large-scale and high-quality instruction data, 2024.
- [266] Qiying Yu, Quan Sun, Xiaosong Zhang, Yufeng Cui, Fan Zhang, Yue Cao, Xinlong Wang, and Jingjing Liu. Capsfusion: Rethinking image-text data at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14022–14032, 2024.
- [267] Cong Wei, Zheyang Xiong, Weiming Ren, Xinrun Du, Ge Zhang, and Wenhui Chen. Omniedit: Building image editing generalist models through specialist supervision. *arXiv preprint arXiv:2411.07199*, 2024.
- [268] Qifan Yu, Wei Chow, Zhongqi Yue, Kaihang Pan, Yang Wu, Xiaoyang Wan, Juncheng Li, Siliang Tang, Hanwang Zhang, and Yueting Zhuang. Anyedit: Mastering unified high-quality image editing for any idea. *arXiv preprint arXiv:2411.15738*, 2024.
- [269] Alpha-VLLM Team. Lumina-accessory github page, 2025.
- [270] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023.

- [271] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2024.
- [272] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023.
- [273] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023.
- [274] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.
- [275] Dhruva Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.
- [276] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.
- [277] Yuwei Niu, Munan Ning, Mengren Zheng, Weiyang Jin, Bin Lin, Peng Jin, Jiaqi Liao, Chaoran Feng, Kunpeng Ning, Bin Zhu, et al. Wise: A world knowledge-informed semantic evaluation for text-to-image generation. *arXiv preprint arXiv:2503.07265*, 2025.
- [278] Yang Ye, Xianyi He, Zongjian Li, Bin Lin, Shenghai Yuan, Zhiyuan Yan, Bohan Hou, and Li Yuan. Imgedit: A unified image editing dataset and benchmark. *arXiv preprint arXiv:2505.20275*, 2025.
- [279] Haozhe Zhao, Xiaojian Shawn Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. Ultraedit: Instruction-based fine-grained image editing at scale. *Advances in Neural Information Processing Systems*, 37:3058–3093, 2024.
- [280] Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, et al. Step1x-edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025.