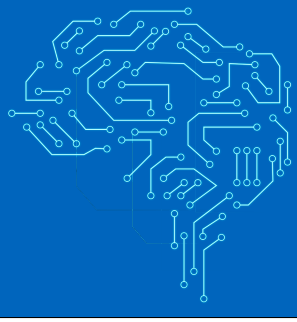
Computer Aided Medical Procedures
Prof. Dr. Nassir Navab



Dissertation

Learning to Learn Neural Representations with Limited Data and Supervision

Azade Farshad

TUM School of Computation,
Information and Technology
Technische Universität München

Technische Universität München
TUM School of Computation, Information and Technology

Learning to Learn Neural Representations with Limited Data and Supervision

Azade Farshad

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung einer

Doktorin der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Niki Kilbertus

Prüfende der Dissertation:
1. Prof. Dr. Nassir Navab
2. Prof. Dr. Ben Glocker
3. Prof. Ehsan Adeli

Die Dissertation wurde am 27.09.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 09.02.2024 angenommen.

Azade Farshad
Learning to Learn Neural
Representations with Limited Data and Supervision
Dissertation, Version 1.0

Technische Universität München
TUM School of Computation, Information and Technology
Lehrstuhl für Informatikanwendungen in der Medizin
Boltzmannstraße 3
85748 and Garching bei München

Abstract

Learning to learn is a powerful paradigm that enables machine learning models to leverage the previously learned features for new tasks and domains more effectively. This approach is especially useful for data-scarce, heterogeneous, and complex domains such as medical imaging analysis. This thesis explores different aspects of learning to learn from data, models, and semantics, and shows how they can enhance various computer vision and medical imaging tasks. In the first part of the thesis, we present novel and fundamental research on learning to learn from data, and in the second part, we investigate the use of high-level semantics in generative models.

In the first part, we study how a model can learn (1) from other model parameters in a distributed setting, (2) from other data instances in a transfer learning setting, and (3) from the inherent structures of the data itself without using external knowledge. We address various challenges that arise in each of these scenarios, and propose novel methods and solutions.

In the second part of this thesis, we introduce the task of semantic image manipulation, which uses scene graphs as a structured representation of the objects and their relationships in an image. Scene graphs can provide rich semantic and spatial information for image generation and manipulation. Within this thesis, we develop several methods that exploit scene graphs for different tasks, such as unconditional image generation, few-shot image generation, and self-supervised image manipulation.

This thesis demonstrates that acquiring larger datasets or designing deeper and more complex neural network architectures is not necessarily the best way to advance machine learning for computer vision and medical imaging. The thesis opens new avenues for efficient representation learning by proposing and developing methods that can benefit from the existing knowledge from other models, other data, and the existing semantic knowledge in the world.

Zusammenfassung

Lernen zu lernen ist ein leistungsfähiges Paradigma, das es Modellen des maschinellen Lernens ermöglicht, die zuvor gelernten Merkmale für neue Aufgaben und Bereiche effektiver zu nutzen. Dieser Ansatz ist besonders nützlich für datenarme, heterogene und komplexe Bereiche wie die medizinische Bildanalyse. In dieser Arbeit werden verschiedene Aspekte des Lernens aus Daten, Modellen und Semantik untersucht und gezeigt, wie sie verschiedene Aufgaben der Computer Vision und der medizinischen Bildgebung verbessern können. Im ersten Teil der Arbeit stellen wir neue und grundlegende Forschungsergebnisse zum Lernen aus Daten vor, und im zweiten Teil untersuchen wir die Verwendung von High-Level-Semantik in generativen Modellen.

Im ersten Teil untersuchen wir, wie ein Modell (1) von anderen Modellparametern in einer verteilten Umgebung, (2) von anderen Dateninstanzen in einer Transfer-Lernumgebung und (3) von den inhärenten Strukturen der Daten selbst lernen kann, ohne externes Wissen zu verwenden. Wir gehen auf verschiedene Herausforderungen ein, die sich in jedem dieser Szenarien ergeben, und schlagen neue Methoden und Lösungen vor.

Im zweiten Teil dieser Arbeit stellen wir die Aufgabe der semantischen Bildmanipulation vor, die Szenegraphen als strukturierte Darstellung der Objekte und ihrer Beziehungen in einem Bild verwendet. Szenegraphen können umfangreiche semantische und räumliche Informationen für die Bilderzeugung und -manipulation liefern. Im Rahmen dieser Arbeit werden mehrere Methoden entwickelt, die Szenegraphen für verschiedene Aufgaben nutzen, z. B. für die unbedingte Bilderzeugung, die Erzeugung von Bildern mit wenigen Aufnahmen und die selbstüberwachte Bildmanipulation.

Diese Arbeit zeigt, dass der Erwerb größerer Datensätze oder die Entwicklung tieferer und komplexerer neuronaler Netzwerkarchitekturen nicht unbedingt der beste Weg ist, um das maschinelle Lernen für Computer Vision und medizinische Bildgebung voranzutreiben. Die Arbeit eröffnet neue Wege für effizientes Repräsentationslernen, indem sie Methoden vorschlägt und entwickelt, die vom vorhandenen Wissen aus anderen Modellen, anderen Daten und dem vorhandenen semantischen Wissen in der Welt profitieren können.

Acknowledgments

I would like to express my sincere gratitude to all the people who supported me throughout my research journey.

First and foremost, I am deeply grateful to my supervisor Nassir, who has been an excellent mentor, guide, and a lovely person. He has provided me with invaluable feedback, advice, and opportunities to grow as a researcher and a person. I thank my partner in life and colleague Yousef, who has been my constant source of inspiration, encouragement, and love and made this journey less challenging. I also thank my parents, Farideh and Hossein, who have always believed in me and supported me in pursuing my dreams.

I would like to thank my colleagues at the CAMP chair, who have made my research journey enjoyable and fruitful. They have been a source of collaboration, discussion, and friendship throughout my Ph.D. I appreciate the contributions and support of Helisa, Ashkan, Shahrooz, Mahdi, Anees, Niko, Schun-cheng, Iro, Christian, Roger, Tariq, Shadi, Thomas, and Federico. I especially acknowledge Vasilis, who inspired me to join the CAMP chair and who has been a great collaborator and supporter. I thank Ehsan and Ahmad, whom I had the pleasure of knowing and working with near the end of my Ph.D., and who are wonderful people. Finally, I would like to thank Martina for her unwavering support in all these years and Leopold for guarding our chair and being a good boy.

Contents

1 List of Authored and Co-authored Publications

2023

[51] **Azade Farshad**, Yousef Yeganeh, and Nassir Navab. "Learning to learn in medical applications: A journey through optimization." *Meta-Learning with Medical Imaging and Health Informatics Applications. Academic Press, 2023. 3-25.*

[194] Abhijit Guha Roy, Shayan Siddiqui, Sebastian Pölsterl, **Azade Farshad**, Nassir Navab, and Christian Wachinger. "Few-shot segmentation of 3D medical images." *Meta-Learning with Medical Imaging and Health Informatics Applications, pp. 161-183. Academic Press, 2023.*

2022

[50] **Azade Farshad***, Yousef Yeganeh*, Helisa Dhamo, Federico Tombari, and Nassir Navab. "DisPositioNet: Disentangled Pose and Identity in Semantic Image Manipulation." *British Machine Vision Conference (BMVC), 2022*.

[55] **Azade Farshad***, Yousef Yeganeh*, Peter Gehlbach, and Nassir Navab. "Y-Net: A Spatiospectral Dual-Encoder Network for Medical Image Segmentation". *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 2022.*

[52] **Azade Farshad***, Anastasia Makarevich*, Vasileios Belagiannis, and Nassir Navab. "MetaMedSeg: Volumetric Meta-learning for Few-Shot Organ Segmentation". *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning, 2022*.

[268] Yousef Yeganeh*, **Azade Farshad***, Johann Boschmann, Richard Gaus, Maximilian Frantzen, and Nassir Navab. "FedAP: Adaptive Personalization in Federated Learning for Non-IID Data." *International Workshop on Distributed, Collaborative, and Federated Learning, Workshop on Affordable Healthcare and AI for Resource Diverse Global Health, pp. 17-27. Springer, Cham, 2022.*

[270] Yousef Yeganeh*, **Azade Farshad***, and Nassir Navab. "Shape-Aware Masking for Inpainting in Medical Imaging." *arXiv preprint arXiv:2207.05787 (2022)*.

2021

[53] **Azade Farshad***, Sabrina Musatian*, Helisa Dhamo, and Nassir Navab. "MIGS: Meta Image Generation from Scene Graphs". *British Machine Vision Conference (BMVC), 2021*.

[63] Sarthak Garg, Helisa Dhamo, **Azade Farshad**, Sabrina Musatian, Nassir Navab, and Federico Tombari. "Unconditional scene graph generation". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021*.

[283] Yang Zhang, Ashkan Khakzar, Yawei Li, **Azade Farshad**, Seong Tae Kim, and Nassir Navab. "Fine-Grained Neural Network Explanation by Identifying Input Features with Predictive Information". *Neural Information Processing Systems (NeurIPS), 2021*.

2020

[42] Helisa Dhamo*, **Azade Farshad***, Iro Laina, Nassir Navab, Gregory D Hager, Federico Tombari and Christian Rupprecht. “Semantic image manipulation using scene graphs”. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), 2020*.

[269] Yousef Yeganeh, **Azade Farshad**, Nassir Navab, and Shadi Albarqouni. “Inverse distance aggregation for federated learning with non-iid data”. *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning, 2020*.

2 Abstracts of Publications not Discussed in this Thesis

Fine-Grained Neural Network Explanation by Identifying Input Features with Predictive Information

Yang Zhang, Ashkan Khakzar, Yawei Li, Azade Farshad, Seong Tae Kim, Nassir Navab

One principal approach for illuminating a black-box neural network is feature attribution, ie identifying the importance of input features for the network's prediction. The predictive information of features is recently proposed as a proxy for the measure of their importance. So far, the predictive information is only identified for latent features by placing an information bottleneck within the network. We propose a method to identify features with predictive information in the input domain. The method results in fine-grained identification of input features' information and is agnostic to network architecture. The core idea of our method is leveraging a bottleneck on the input that only lets input features associated with predictive latent features pass through. We compare our method with several feature attribution methods using mainstream feature attribution evaluation experiments. The code is publicly available.

Neural Information Processing Systems (NeurIPS) 2021

Part I

Introduction

c

3 Learning to Learn

There has been a large amount of research in the field of Artificial Intelligence (AI) in the past decade, transforming it into a central pillar of computer science [196]. Two of the major areas in AI are Computer Vision and Natural Language Processing (NLP). These areas are essential for AI applications, since they enable AI to analyze visual and linguistic data in a similar way to human cognition [66].

In the early days of AI research, traditional methodologies formed the backbone of data analysis and processing. Techniques such as Principal Component Analysis (PCA) were instrumental in reducing the dimensionality of data, making it easier for inference [107]. Simultaneously, research endeavours focused on the development of feature extraction methods from high-dimensional datasets. An exemplary method was the Scale-Invariant Feature Transform (SIFT), a highly engineered yet interpretable method for identifying distinctive keypoints in images [152].

Additionally, Support Vector Machines (SVM) provided a powerful tool for the classification and regression analysis of these extracted features [35]. Furthermore, a host of other techniques, such as minimum spanning trees, superpixels, and graph-cut methods, were utilized to streamline and optimize data processing mechanisms for different tasks such as segmentation [56, 212]. Despite their efficacy, these techniques were restricted by their inherent limitations, predominantly their hand-crafted and manually engineered nature.

With the advent of Neural Networks and Deep Learning, a paradigm shift occurred in AI research [128]. Neural networks, sophisticated enough to be trained for classification tasks, emerged as automatic feature extractors, superseding their traditional counterparts. This change, however, had its drawbacks. Despite their superior performance, the intricate internal workings of these networks were optimized based on simple loss functions, turning them into complex, black-box systems with limited interpretability [276].

This situation underscored the criticality of representation learning in machine learning and AI [11]. The advent of representation learning gave rise to a new frontier in the field of artificial intelligence, setting the stage for ground-breaking advancements like meta-learning, self-supervised representation learning, and foundation models.

Meta-learning, often described as "learning to learn" [205], has emerged as a burgeoning approach in machine learning, aiming to design models that can learn new skills or adapt to new environments rapidly with a few training examples [60, 240]. The ability to adapt to new tasks using minimal data extrapolates the principles of traditional learning methods, offering a glimpse into the potential future of AI. Meta-learning acknowledges the necessity of abstracting from individual task learning, and focuses on understanding the underlying learning process itself, encapsulating this in the learning model. This conceptual shift allows

for the rapid adaptation to new, previously unseen tasks, a capacity that is instrumental in the practical deployment of AI systems in the real world.

Meta-learning has various definitions; In this thesis, we explore topics that go further than few-shot learning, rather towards learning with both limited data and supervision, which could be an ultimate goal in artificial general intelligence or AGI.

In the first part of the thesis, we discuss how our models can learn from the data. This part focuses on different types of learning to learn without blindly increasing the amount of annotated data or increasing the complexity of the neural network architectures.

In the second part, we introduce several approaches that leverage natural language for computer vision tasks. In contrast to images that can contain huge amount of high dimensional information, natural language contains the distilled knowledge perceived through humans cognition. This knowledge can be employed to enrich computer vision models; however, natural language is still not well-structured, and therefore a graph-based semantic representation can be more accurate. This representation, taking the form of scene graphs can be utilized to condition computer vision models more efficiently. In this thesis, we propose novel methods that model the scenes semantically through learning a common representation between images and scene graphs. The semantic scene modeling is employed to enhance the performance of generative models with different levels of supervision.

In the next chapter, we summarize the contributions of this thesis.

4 Contributions

In this thesis, we explore different levels of learning to learn, a paradigm that aims to improve the generalization and adaptation of machine learning models to new tasks and domains. We investigate how learning from other models, other data, the inherent features in the data, and semantics can enhance the performance of various computer vision applications. We divide our contributions into two parts: the first part focuses on fundamental research on learning to learn from data, and the second part focuses on learning from semantics using generative models. Figure 4.1 shows a taxonomy of the works covered or published in the course of this thesis.

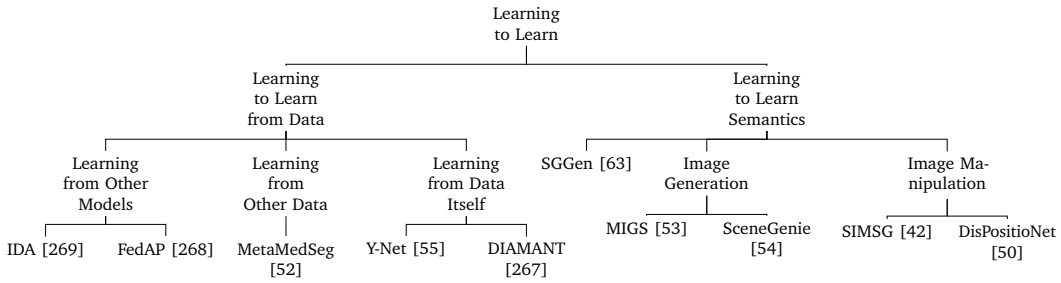


Fig. 4.1. A taxonomy of the published works in this thesis

In the first part, we start with learning from other models in a federated, collaborative and distributed learning setting, where each model learns from the knowledge acquired from other nodes without directly accessing their data. This setting poses several challenges, such as non-iid data distribution, communication efficiency, and privacy preservation. We tackle the problem of non-iid data distribution in this collaborative learning setting in two works. IDA [269] proposes a parameter aggregation approach on the server to optimize the feature learning for all the models, and in FedAP [268] we propose an adaptive personalization approach based on meta-learning and hierarchical clustering to obtain higher performing models at each node.

Then, we focus on learning from other data rather than other models. It is possible to take advantage of other data with abundant labels information to improve the performance on a target domain with scarce labels. This is particularly relevant for medical image segmentation, where annotating images is costly and time-consuming. In MetaMedSeg [52] we optimize the model to learn from other organs by optimizing it in a meta-learning setting for the few-shot learning task. We show that our approach can achieve state-of-the-art results on several medical image segmentation benchmarks.

Learning from other models and other data is nice, but learning from inherent structures inside the data is even more interesting. Such inherent structures can be seen for example in OCT images, where the images encode some high frequency noise or there are structures

such as retinal fluid which are scattered in the image. Therefore, we propose YNet [55] which extracts and combines spectral features with spatial features using Fast Fourier Convolution (FFC) layers. By combining these spectral and spatial features, we can achieve better higher performance in downstream tasks such as segmentation.

In the second part of thesis, we explore ways of using generative models and semantics to learn a representation of the scene. This part mostly focuses on image generation and manipulation using semantics by modeling the scene using a scene graph, a structured representation that captures the objects and their relationships in an image. Scene graphs can provide a rich and flexible way of expressing various visual concepts and scenarios. We propose several methods to leverage scene graphs for different tasks, such as unconditional image generation, few-shot image generation, and self-supervised image manipulation.

In SceneGraphGen [63], we learn the distribution of scene graphs from a large-scale dataset of images and their corresponding scene graphs. We then use this learned distribution to generate new scene graphs that are realistic and diverse. We show that using these generated scene graphs as inputs for an image generation model can produce better quality images than using ground truth scene graphs.

In MIGS [53], we address the problem of few-shot image generation using scene graphs in new scenes that have not been seen by the model during training. We propose a meta-learning framework that can quickly adapt to novel scenes by learning from a few examples. We demonstrate that our method can generate more realistic and faithful images than existing methods.

In SIMSG [42], we tackle the problem of self-supervised image manipulation without paired data. We design a system that can manipulate an image according to a given scene graph by changing the attributes or relations of objects in the image. We do not require any supervision for this task, as we use cycle-consistency and perceptual losses to ensure that the manipulated image matches the desired scene graph.

Finally, in DisPositioNet [50], we learn disentangled scene graphs from images in an unsupervised manner. We propose a novel graph neural network architecture that can separate the appearance and position information of objects in a scene graph. We show that this disentanglement can enable various applications, such as position-aware image generation, position transfer, and position interpolation.

Part II

Learning to Learn from the Data

5 Representation Learning

In recent years, representation learning has become a key component of machine learning, providing an effective method for transforming raw data into meaningful and high-level features [11]. Representation learning, particularly in deep learning-based approaches, has shown outstanding performance in various tasks, such as image classification, natural language processing, and more.

Representation learning is the process of learning meaningful and useful representations of data that can facilitate subsequent tasks such as classification and segmentation [100]. The quality of the learned representations depends on several factors, such as the amount and diversity of the available data, the choice of the learning objective, and the architecture of the model. Depending on these factors, different variations of representation learning can be distinguished, such as supervised, semi-supervised, unsupervised, and self-supervised representation learning.

Supervised representation learning is a technique that uses labeled data to learn representations that are discriminative for a specific task, such as image classification or object detection. The labels provide a direct supervision signal for the model to optimize its parameters and extract relevant features from the data. Supervised representation learning can achieve high performance on the target task, but it requires a large amount of annotated data, which can be costly and time-consuming to obtain. Moreover, supervised representations may not generalize well to other tasks or domains that have different distributions or labels [66].

Semi-supervised representation learning is a technique that uses both labeled and unlabeled data to learn representations that are both discriminative and generalizable. The idea is to leverage the abundant unlabeled data to learn rich features that capture the underlying structure of the data, while using the scarce labeled data to guide the model towards the target task. Semi-supervised representation learning can overcome some of the limitations of supervised representation learning, such as data scarcity and overfitting. However, it also poses some challenges, such as how to effectively combine the information from both types of data, how to deal with noisy or incomplete labels, and how to evaluate the quality of the learned representations.

Few-shot learning (FSL) has similarities to semi-supervised learning and has been a subject of substantial research interest, with the goal of designing learning systems that can generalize well from limited data [51, 242]. This approach is particularly valuable in real-world scenarios where large amounts of annotated data are rarely available. Meta-learning, or learning to learn, is a key concept underpinning few-shot learning, in which models are designed to learn rapidly when exposed to new tasks [232]. Notable meta-learning approaches include MAML (Model-Agnostic Meta-Learning) [60] and Reptile [167], both of which optimize the learning process to adapt quickly to new tasks with limited data. In the extreme case of

few-shot learning, we have zero-shot learning, which aims to perform tasks on classes not seen during the training [256]. The goal is to leverage auxiliary information, such as semantic attributes or text descriptions, to build a bridge between seen and unseen classes. The concept of few-shot and zero-shot learning is now being extended to more complex tasks such as image segmentation. Few-shot segmentation methods aim to segment new object classes with only a few annotated examples [184], while zero-shot segmentation seeks to do so without any labeled instances [249].

Unsupervised representation learning is a technique that uses only unlabeled data to learn representations that are invariant or robust to irrelevant variations in the data, such as translation, rotation, or scaling. The idea is to learn features that capture the intrinsic properties or factors of variation of the data, without relying on any external supervision. Unsupervised representation learning can be useful for discovering latent structures or patterns in the data, such as clusters, manifolds, or hierarchies. However, it also faces some difficulties, such as how to define a suitable learning objective, how to ensure that the learned features are interpretable or meaningful, and how to measure the usefulness of the learned representations for downstream tasks [66].

Self-supervised learning is an unsupervised learning method that generates pseudo-labels from the data itself to enable training without human-annotated labels [43]. It benefits from the vast amounts of unlabeled data available, transforming them into useful, high-dimensional representations. A key technique used in self-supervised learning is contrastive learning, which learns representations by comparing similar and dissimilar instances [69]. The recently proposed DINO (self-distillation with no labels) technique builds upon this contrastive learning approach, focusing on the alignment of global views with local ones [23].

Another important category of self-supervised representation learning is denoising autoencoders, which learn to reconstruct data from a corrupted version of itself, ultimately learning useful data representations in the process [241]. These autoencoders have been pivotal in unsupervised learning, demonstrating the ability to learn powerful representations from raw, unlabeled data and have been a basis for Diffusion Models [192] which are the current state of the art in image synthesis.

In sum, these approaches represent significant advances in the fields of representation learning, few-shot learning, and zero-shot learning. They all strive towards the same goal: to develop more flexible, efficient, and data-economic learning systems. By improving these methodologies, we move closer to creating machine learning models that can perform complex tasks with minimal human supervision or with limited labeled data. In this part of the thesis, we will explore and propose methods that tackle the challenges in representation learning from various types of data.

6 Learning from Other Models

6.1 Introduction

Federated learning (FL) has emerged as a transformative paradigm for collaborative learning that respects data privacy across different clients, drawing increasing interest in recent years [120]. Its capacity to create a consolidated global model while maintaining the confidentiality of individual client data sets it apart as an innovative strategy in data analysis. FL achieves this by employing a server-side model aggregation mechanism, effectively learning from the aggregated knowledge of all clients without compromising the privacy of individual data.

Despite its immense potential, FL presents certain challenges. Most prominently, the data across different clients tends to be widely distributed, non-independent, and non-identically distributed (non-iid), as well as unbalanced, posing significant complexities [83]. These issues are particularly evident in the medical field, where data heterogeneity arises due to various factors such as class imbalance in pathology [156], intra-/inter-scanner variability, intra-/inter-observer variability, and multi-modal data [135]. Consequently, under such heterogeneous conditions, models struggle to learn effectively in a distributed setting with non-iid data distribution.

To address these challenges, we introduce two innovative methods: Inverse Distance Aggregation (IDA) [269] and Adaptive Personalization (FedAP) [268]. Both approaches are designed specifically to manage non-iid and unbalanced data within the context of FL.

In addition to dealing with general FL challenges, it is vital to recognize and address complications resulting from the varying participation rates of clients. A flexible model capable of adapting to fluctuating participation rates is critical for ensuring the robustness and validity of the learning process. Our proposed methods consider this crucial factor and offer solutions capable of performing efficiently even when client participation is less than optimal.

Although the potential of FL is profound, especially in the healthcare sector [210], existing methodologies have failed to adequately consider the role of aggregation mechanisms amid data and system heterogeneity. Our contribution directly confronts this issue, providing strategies capable of effectively handling non-iid data and variations in client participation.

IDA offers an aggregation strategy that computes adaptive weighting based on meta-information derived from the statistical properties of model parameters. This technique diminishes the effect of non-iid and unbalanced data, leading to the creation of a more reliable global model. It helps to alleviate the divergence problem often encountered by FL systems, thereby boosting overall model performance.

On the other hand, FedAP acknowledges the unique data distribution associated with each client, treating each as a separate learning task. By using meta-learning principles, the method is designed to learn how to learn, thereby personalizing the model to each client's unique characteristics. This personalization fosters better model convergence and overall performance, while ensuring robustness to heterogeneity.

Our goal in FedAP is to enhance personalization, and to achieve this, we are treating Federated Learning (FL) as a meta-learning [60, 167] problem. By integrating a meta-learning schema with clustering, we highlight its benefits, including improved personalization, better preservation of specialized data, and expedited model convergence.

Meta-learning, alternatively referred to as learning to learn [239], is the process of effectively learning how to resolve new tasks from a collection of known tasks [60]. In the context of FL, clients can be seen as meta-learning tasks, as each client's data distribution represents a unique problem. Therefore, the application of meta-learning concepts to FL has been successful [24, 47, 98, 116]. Following the MAML approach [60], we cluster clients with analogous distributions into groups of tasks, effectively transforming each cluster into an independent FL problem. This enhances the homogeneity of the data shared among the clients within their respective clusters. Despite the data in FL not being directly accessible, we assess the similarity between clients based on the value differences in model parameters between the clients and the global model from the most recent round. Briggs et al. explored a technique that uses hierarchical clustering to group similar models [17]. We adopted this approach for the personalization of clients' models, merging it with our Adaptive Personalization (FedAP) for the final cluster-based training. We maintain the simplicity of FedAvg [98], yet we differentiate in two key areas: 1) FedAP views the consolidated global update as a meta-level gradient that can be employed with a different optimizer to refresh the global model, thereby introducing a new hyperparameter - the meta-learning rate, and 2) At the completion of the training, FedAP customizes the global model for each client, providing a robust advantage within the FL context [47, 98].

Our solutions are evaluated across multiple datasets, proving their efficacy in handling the challenges associated with *FL*. We demonstrate improved classification accuracy, reduced accuracy variance between clients, and enhanced robustness against non-iid data, positioning our approach as a step forward in the application of *FL*.

Visualizations of IDA and FedAP pipelines are presented in Figure 6.1 and Figure 6.2, respectively. The main contributions of this chapter are as follows:

- We propose IDA and INTRAC, two model aggregation techniques to tackle the non-IID data distribution problem in the federated learning setting.

- We introduce FedAP, a novel method employing hierarchical clustering to enact adaptive personalization within the clusters and for each client, when dealing with non-IID (non-identical or independent) data.

- Our methods register substantial performance enhancements in classification accuracy and in the reduction of accuracy disparity between different clients.

The subsequent sections of this work delve into the detailed methodology of both the IDA and FedAP, followed by a comprehensive showcase of our experimental results. By demonstrating the tangible benefits of our approach in dealing with the realities of non-iid data in federated learning environments, we hope to illuminate a new pathway for the application of *FL*, particularly in medical imaging.

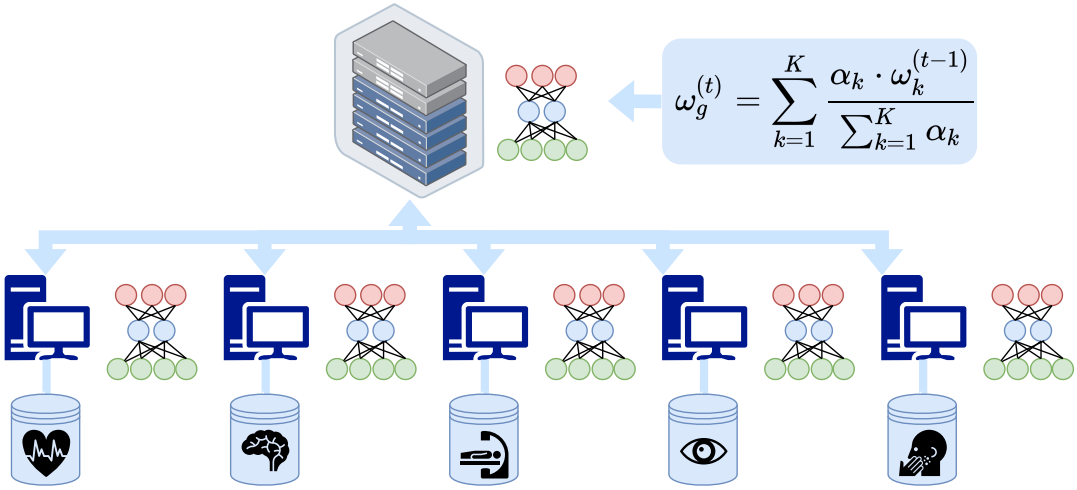


Fig. 6.1. **IDA**. Federated learning with non-iid data - The data has different distributions among clients.

6.2 Related Work

Federated Averaging (*FedAvg*) [158] is one of the most well-known *FL* methods which uses the normalized number of samples in each client to aggregate the models in the server. FedAvg (federated averaging) [158] was proposed as one of the first FL algorithms and has been used as a standard benchmark.

Another aggregation approach using temporal weighting along with a synchronous learning strategy was proposed in [27]. Many recent approaches have been proposed in order to improve the generalization or personalization of the global model using the ideas of knowledge transfer, knowledge distillation, multi-task learning and meta-learning [10, 24, 34, 94, 98, 131, 219].

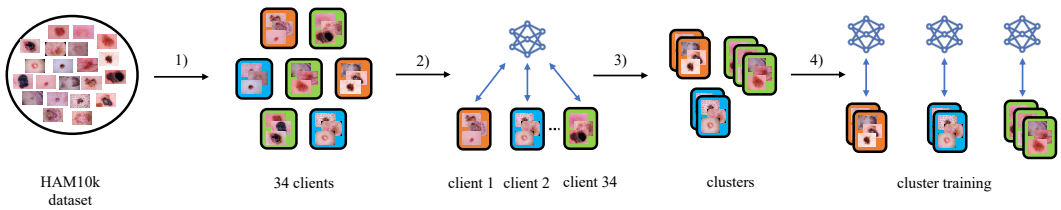


Fig. 6.2. **FedAP**. The pipeline has four steps: 1) splitting the dataset, 2) training all clients with FedAvg for predefined rounds, 3) clustering the clients based on the latest model update, 4) performing FedAvg or FedAP on each separate cluster.

A large number of recent works on *FL* such as [143, 204] are focused on communication efficiency due to its application on edge devices with unstable connections [135], commonly using approaches such as compressed networks or compact features, however FL's most determining aspects in the medical field are data privacy and heterogeneity [109, 190].

Non-IID data in FL Non i.i.d. data is shown to impact both the convergence speed and the final performance of the FedAvg algorithm [138, 204]. [138, 288] tackle data heterogeneity by sharing a limited common dataset. Motivated from classical machine learning techniques [135] introduces a weight regularization term to the local objective function to prevent the divergence between local and global models. Recently, a semi-supervised learning approach for federated learning on the ISIC skin lesion dataset was proposed [8, 9]. Yue *et al*. [275] surveys on the data availability and heterogeneity in the medical domain, concluding that privacy restrictions and missing data pipelines block the full potential of deep Learning which requires big data sets. The described advances in the field of federated learning can help overcome the challenge of medical data privacy and disseminate machine learning techniques in healthcare [209]. Notably, data heterogeneity remains a significant hurdle to this development, so robust techniques towards non-IID data carry considerable future potential [190].

There has been numerous works to handle each of these data assumptions [108]. Sattler et al. [203] propose clustering loss terms and using cosine similarity to overcome the divergence problem when clients have different data distributions. Zhao et al. [288] overcome the non-iid problem by creating a subset of data which is shared globally with the clients. In order to maintain system heterogeneity (affected by their main idea of nonuniform local updates), FedProx [136] proposes a proximal term to minimize the distance between the local and global models. Close to our approach, geometric median is used in [179] to decrease the effect of corrupted gradients on the federated model.

FL in healthcare In the last few years, there has been a growing interest in applying *FL* in healthcare, in particular, to medical imaging. Sheller et al. [210] were among the first works who applied *FL* to multi-institutional data for Brain Tumor Segmentation task. To date, there has been numerous works on *FL* in Healthcare [87, 137, 209, 210, 260].

Personalization A study on live data of millions of users shows that significant improvements can be achieved by personalizing the learning rate and batch size to clients [248]. FedOpt [186] extends the FedAvg algorithm and implements personalization by introducing adjustable gradient update strategies for each client and server. Employing weight decay over training rounds on the server-side is shown to be required to lower the error on non-i.i.d. data [138]. [3] proposes to learn base layers globally while keeping classification layers private on the client-side. FedMD [131] introduces a framework for individually designed by each client. The MOCHA [219] enables efficient meta-learning in the federated environment. Communication efficiency in federated learning is also addressed in [204] suggesting a compression protocol based on quantization.

6.3 Definitions

We are given a set of M clients, where each client $i \in \{1, \ldots, M\}$ only sees its local dataset, which is a subset $\mathcal{S}_i \sim \mathcal{D}_i$ of the global data distribution $\mathcal{D}$ and $x_1, ..., x_{N_i} = \mathcal{S}_i$, where $\mathcal{S} \sim \mathcal{D}$ is the sampled data points and $x_1, ...x_N = \mathcal{S}$ and $\mathcal{D}_i$ is the client's local data distribution.

A neural network with parameters denoted by θ, where θ_{global} corresponds to the global model parameters is shared between the server and thee clients. The objective is to train a global model minimizing the following objective function:

$$\min_{\theta_{global}} \mathcal{L}(\mathcal{S}) = \min_{\theta_{global}} \frac{1}{M} \sum_{i=1}^{M} l_{\theta_{global}}(\mathcal{S}_i) \tag{6.1}$$

where $\mathcal{L}$ is the global loss function and l the local loss function of each client.

6.3.1 Client

A client is selected randomly from a total of M clients, considering the participation rate pr. Each client i receives the global model parameter θ^t_{global} at the communication round t and trains the shared model, initialized by θ^t_{global}, on its training data $\mathcal{S}_i$ for E iterations. This process minimizes the local objective function $f_i(x) = \mathbf{E}\, x \sim \mathcal{S}_i[f(x; \theta^t_i)]$, where θ^t_i are the weight parameters of the client i. Each client's training data is a subset of the entire training data, possibly sampled from different classes of data, denoted by n_{cc}.

6.3.2 Server

The server plays a crucial role in each round t, aggregating the updated local parameters θ^t_i from the clients to form the updated global parameter θ^t_{global}, defined as:

$$\theta^t_{global} = \sum_{i=1}^{M} \alpha_i \cdot \theta^{t-1}_i. \tag{6.2}$$

In this equation, α_i represents the weighting coefficient. This aggregation procedure is repeated for the total communication rounds T.

6.3.3 Federated Averaging

Assume that there are M clients, and $N_i = |\mathcal{D}i|$ denotes the quantity of data samples available with client i. The global and local loss functions can be formulated as:

$$f(\theta) = \sum i = 1^M \frac{N_i}{N} F_i(\theta) \quad \text{where} \quad F_i(\theta) = \frac{1}{N_i} \sum j f_j(\theta) \tag{6.3}$$

In this equation, $Fi(\theta)$ refers to the local objective specific to each client, and $f(\theta)$ represents the global objective, computed as an average across all clients.

The training process using FedAvg incorporates two fundamental updating mechanisms:

- Firstly, local training is carried out at the clients, using a model initially distributed among them. This model is trained on their local data for a predefined number of epochs, denoted by e. In each communication round, client i updates its model parameter θ_i, employing a learning rate α, as specified in the equation,

$$\forall i,\ \theta_i^{t+1} \leftarrow \theta_i^t - \alpha \cdot \Delta\theta \tag{6.4}$$

- Secondly, the server carries out a computation, calculating a weighted average. This calculation is based on the number of data points N_i held by each client i and uses all locally updated model parameters θ_i. This process is captured in the equation:

$$\theta_{global}^{t+1} \leftarrow \sum_{i=1}^{M} \frac{N_i}{N} \theta_i^{t+1} \tag{6.5}$$

In this manner, the FedAvg algorithm effectively combines the benefits of local client-side computation and global server-side updates, enabling efficient collaborative learning.

6.4 Inverse Distance Aggregation (IDA)

To alleviate inconsistencies arising among local parameter updates due to non-IID challenges, we introduce an innovative and robust method for aggregation, termed Inverse Distance Aggregation (**IDA**). This method hinges on the manner in which coefficients α_i are computed. This computation is based on the inverse distance of each client's parameters from the average model of all clients, an approach designed to downplay or dismiss out-of-distribution, or 'poisoning', models.

To achieve this, we utilize the ℓ_1-norm to determine the distance between the parameters of each client θ_i and the average parameters θ_{Avg}, as shown in the equation:

$$\alpha_i = \frac{1}{Z} |\theta^{t-1} Avg - \theta^{t-1} i|^{-1}, \tag{6.6}$$

Here, $Z = \sum_{i \in M} |\theta^{t-1} Avg - \theta^{t-1} i|^{-1}$ serves as a normalization factor. To avert numerical instability, we introduce ϵ to both the numerator and denominator. When a client's parameters align with the average, we get $\alpha_i = 1$, and when $\alpha_i = n_i$, it corresponds to the *FedAvg* [158].

Furthermore, we propose the utilization of the training accuracy of clients in the final weighting, a method we term INTRAC (INverse TRaining ACcuracy). This method penalizes overfitted models and rewards under-trained ones within the aggregated model. For INTRAC, coefficients are determined as $\alpha'_i = \frac{Z'}{max(\frac{1}{M}, acc_i)}$, where the max function ensures all values exceed chance level. In this equation, acc_i stands for the training accuracy of client k, $\alpha' i$ represents the INTRAC coefficient, and $Z' = \sum i \in M max(\frac{1}{M}, acc_i)$ is the normalization

factor. These calculated coefficients, α'_i, are then normalized again to fit within the $(0, 1]$ range. Lastly, to integrate different coefficient values (i.e., INTRAC, IDA, FedAvg), we multiply and then normalize the coefficients to fit within the $(0, 1]$ range.

6.4.1 Experiments and Results

Our proposed method was assessed on a variety of widely used databases to provide a Proof-of-Concept (PoC) demonstration, followed by an illustration of results on a real-world clinical use case. We compare the results obtained from our proposed method, the **IDA**, with the established baseline method, *FedAvg* [158]. The first sequence of PoC trials aims to explore the following aspects:

1. Non-IID vs. IID: Comparative analysis of *FedAvg* and **IDA** under both IID and non-IID conditions, utilizing various datasets and architectural frameworks.

2. Ablation Study: Investigation into the efficacy of IDA in relation to FedAvg.

3. Sensitivity Analysis: A performance comparison under extreme conditions.

Datasets The evaluation results of our method are exhibited across three distinct datasets: cifar-10 [122], fashion-mnist (f-mnist) [257], and HAM10K (an assortment of dermatoscopic images of pigmented lesions) [236]. F-mnist, a recognized derivative of mnist, contains $50k$ images of 28×28 monochromatic clothing items. Cifar-10, another dataset commonly employed in computer vision research, offers $60k$ 32×32 images of various vehicles and animals. For the clinical study, we assessed our method utilizing the HAM10k dataset, which encompasses a total of 10015 images of diverse pigmented skin lesions classified into 7 groups. The classes and their respective sample counts in HAM10k are: Melanocytic nevi: 6705, Melanoma: 1113, Benign keratosis: 1099, Basal cell carcinoma: 514, Actinic Keratoses: 327, Vascular lesions: 142, Dermatofibroma: 115. We selected this dataset due to its pronounced imbalance.

Implementation Details The specific training configurations for each dataset were as follows: For f-mnist, we utilized the LeNet architecture [127] designed for 10 classes, with a batch size of 128, learning rate (lr) of 0.05, and a local iteration value of 1 (E=1). For cifar-10, we employed VGG11 [216] without batch normalization and dropout layers, designed for 10 classes with a batch size of 128, lr of 0.05, and E=1. In the case of the HAM10K dataset, we employed the Densenet-121 architecture [86], designed for 7 classes, with a batch size of 32, lr of 0.016, and E=1. For all experiments, 90% of the images were randomly chosen for training, with the remainder being used for evaluation. All models underwent training for a total of 5000 rounds. Unless specified otherwise, these values were kept consistent across all experiments.

Evaluation Metrics In all experiments, we partitioned a portion of each client's dataset to act as its test set, and we reported the accuracy of the global (aggregated) model on the combined test sets of all clients, as well as the local accuracy of each client on its own test

Tab. 6.1. Comparison to the baselines on cifar10 and f-mnist with different number of classes per client in iid and non-iid scenarios

Dataset	n_{cc}	3c			5c			iid
	Method \ pr	30%	50%	100%	30%	50%	100%	100%
cifar-10	FedAvg	63.20	65.11	69.81	19.68	83.11	80.94	87.77
	IDA	**64.36**	**67.70**	**70.80**	**76.06**	**83.55**	**83.82**	**89.46**
f-mnist	FedAvg	86.23	87.09	**87.45**	87.60	87.81	87.16	86.95
	IDA	**87.64**	**87.61**	87.44	**87.93**	**87.89**	**87.46**	**87.10**

Tab. 6.2. Ablation study of various weighting combinations on the F-MNIST and CIFAR-10 datasets with $n_{cc} = 3$, $pr = 30\%$.

Method \ Settings	f-mnist	cifar-10	
	K=10	K=10	K = 20
Mean	87.47	65.82	84.80
FedAvg	86.23	63.20	22.84
IDA	87.64	64.36	83.98
IDA + FedAvg	86.67	**67.29**	82.14
IDA +INTRAC	**88.33**	64.93	**85.23**

data. This process helps us gauge the representativeness of the global model with respect to the combined dataset. We reported classification accuracy for all experiments.

Proof-of-Concept Experiments

Non-IID vs. IID In this section, we evaluate and compare **IDA** to *FedAvg* on f-mnist and cifar-10 datasets across various data distribution scenarios among clients. Table 6.1 illustrates the outcomes in scenarios of balanced data distribution, where all clients possess an equal or similar number of samples for $n_{cc} \in 3, 5, 10(iid)$ and $pr \in \{30\%, 50\%, 100\%\}$. Our findings indicate that **IDA** exhibits superior or comparable performance to *FedAvg* across all balanced data distribution scenarios.

Ablation Study In this section, we investigate the impact of various components of the weighting coefficients. For this purpose, we undertake an evaluation of all the proposed components on the CIFAR-10 and F-MNIST datasets. Our findings are compared against two baseline methods, specifically *FedAvg* and an alternative baseline where $\alpha_i = 1$, which is referred to as *Mean* and demonstrated in Table 6.2. Furthermore, we assess the fusion of our weighting method with the number of samples per client (**IDA + FedAvg**) as well as the incorporation of each client's training accuracy into the weighting scheme (**IDA + INTRAC**). The results underscore that merging diverse weighting schemes can enhance the performance of the global model within Federated Learning (FL). This validates our hypothesis that, if a subset of clients possess lower quality or malicious models, *FedAvg* would be susceptible. However, our methods can mitigate the influence of poor models (overfitted, low quality or malicious models) thus leading to a more robust final model that performs optimally on the federated dataset.

Sensitivity analysis In real-world applications, the stability of the learning process under less-than-ideal conditions is crucial. In *FL*, there is no obligation for participants to contribute in each round, leading to potentially varying participation rates during different training rounds and the possible presence of suboptimal models at any given time. Moreover, it is

quite plausible that some clients may have a scarce amount of data samples, while others may possess a large volume of data. In this section, we explore the performance of the global model under conditions of low participation rate and severe non-IIDness.

Low Participation Rate in Non-IID Distribution To examine the impact of the participation rate, we employed 1000 clients on the F-MNIST dataset, with parameters set as follows: batch size=30, learning rate=0.016, number of communication rounds=3, and up to 500 samples per client. In this experiment, we find that, despite the relative ease of learning from this dataset, reducing the client participation rate hampers performance (as demonstrated in Figure 6.3). Notably, the model trained using *FedAvg* fails when the participation rate drops to 1%. However, as the participation rate rises to 5%, the model recovers and continues to learn. Both **IDA** and **IDA + FedAvg** exhibit robust performance in both scenarios, underscoring their resilience against fluctuations in participation rates.

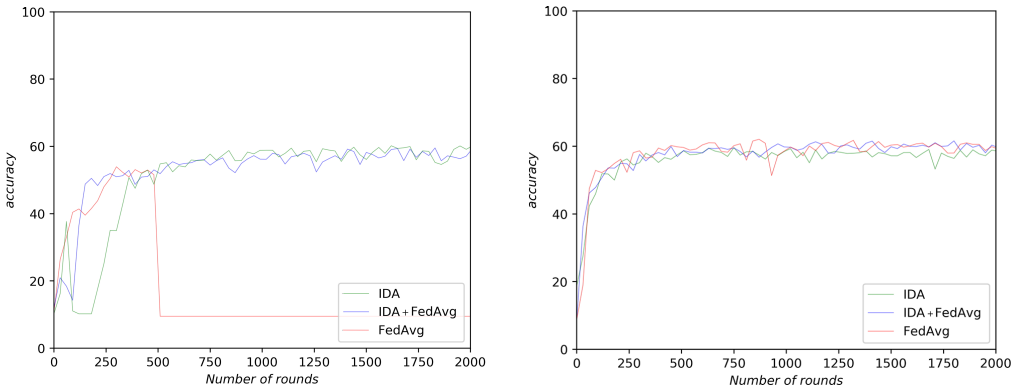


Fig. 6.3. Left: participation rate pr= 0.01; Right: pr = 0.05. The participation rate significantly influences the stability of federated learning. As demonstrated, **IDA** consistently exhibits stable performance, showcasing a marked advantage over FedAvg.

Severity of Non-IID In order to evaluate the influence of non-IID conditions on the performance of our method, we devised an experiment where the number of data samples from underperforming clients was increased.

Initially, we conducted our training according to the parameters established in the preceding sections. Subsequently, we selected the three clients with the lowest accuracy following the initial training and doubled their sample count in the training data distribution. The training was then repeated with this newly generated data distribution.

This experiment was designed to assess the effects of the FedAvg weighting in a scenario where underperforming clients were attributed higher weight. As shown in Figure 6.4, prior to increasing the number of samples, **IDA** outperformed the other methods by a small margin. However, following the sample increase in the three selected clients, FedAvg's performance significantly deteriorated at the start of training.

By considering the performance of the Mean aggregation, we can infer that **IDA** plays a crucial role in the learning process.

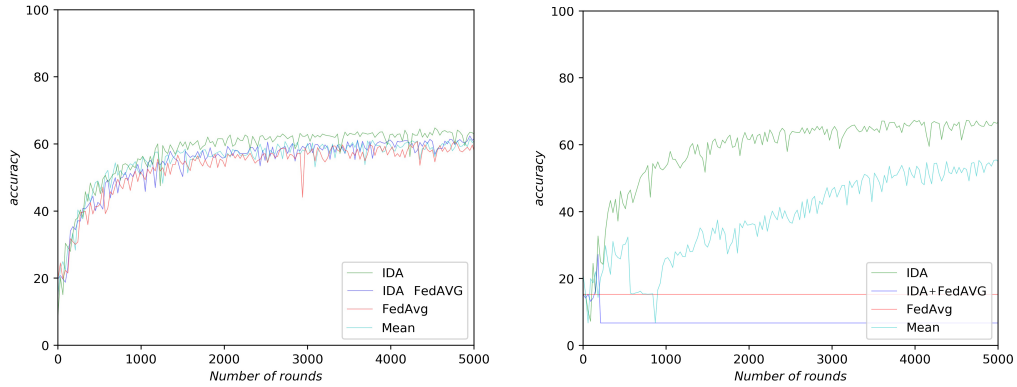


Fig. 6.4. Accuracy of the global model for clients with non-IID data distribution on the CIFAR-10 dataset. On the right, the experiment conditions remain the same as on the left, but with an increased number of samples for three clients that initially had poor performance. The local distribution of data points for those three clients remained unchanged. This experiment was conducted on the CIFAR-10 dataset with $K = 10$ clients, $n_{cc} = 3$, $E = 2$, learning rate of 0.01, and a random number of samples per class per client, up to 1,000 samples.

Clinical Application We applied our proposed method to the HAM10k dataset and presented our findings in Table 6.3. Although the global accuracy of the model utilizing **IDA** is comparable to that of *FedAvg*, it's evident that the local accuracy (accuracy of clients on their own test set) achieved with **IDA** exceeds that of *FedAvg* in every scenario. This demonstrates that **IDA** offers superior generalization and a reduced variance in the local accuracy of clients.

Tab. 6.3. Comparison on an unbalanced data distribution among clients in a federated setting, with five random classes per client, and a random number of samples per client for the HAM10k dataset.

Method	n_{cc}	Global Accuracy	Local Accuracy
FedAvg	1	**69.72**	60.52 ± 9.20
IDA	1	69.16	$\mathbf{61.21 \pm 8.79}$
FedAvg	2	**62.23**	57.14 ± 10.84
IDA	2	61.21	$\mathbf{60.21 \pm 5.48}$
FedAvg	10 (iid)	63.5	52.88 ± 15.73
IDA	10 (iid)	**63.72**	$\mathbf{57.38 \pm 10.56}$

6.5 Federated Adaptive Personalization (FedAP)

In this section, we elucidate the process of our proposed federated adaptive personalization method (FedAP), and detail the integration of hierarchical clustering within our approach. Later, we present and discuss the results of our experiments using FedAP.

Our FedAP methodology commences with the initialization of the global model parameters, denoted as θ_{global}. As each of the k federated rounds starts, these parameters are transmitted to a set of n clients, chosen randomly. This initiates the local training phase in each client i, resulting in the creation of updated local model parameters, θ_i.

Subsequently, an update of θ is carried out using an adaptive meta learning rate, η, which is determined based on the current round number. This rate possesses a unique property - it

diminishes linearly throughout the training process. The equation given below represents this phase of model parameter adjustment:

$$\theta_{global} \leftarrow \theta_{global} + \eta \sum_{i=1}^{n} \frac{N_i}{N} (\theta_i - \theta_{global}) \tag{6.7}$$

Once the final federated round is completed, a personalization step is undertaken for the local model parameters of all clients $i \in 1, \ldots, M$. This involves conducting a fixed number of gradient optimization epochs on the local training data, tailoring the models more closely to the respective client data distributions.

6.5.1 Hierarchical Clustering

In this section, we explain how the personalization of local models can be further enhanced by exploiting the similarities in data across different clients. With this goal in mind, we propose to employ hierarchical clustering, a method previously presented by Briggs et al. [17], after a preliminary phase of global federated learning that includes all the clients. Once FedAvg has been executed for a predetermined number of federated rounds, we conduct hierarchical clustering of clients based on their model updates in the current round. Each cluster of clients that emerges from this process is then treated as an independent federated learning scenario, wherein either FedAvg or our proposed FedAP methodology is applied.

Our underlying hypothesis here is that, even though each distribution $\mathcal{D}i$ is distinct, these can be grouped based on similarity. Drawing from [17], we introduce a unique model θc for each resulting cluster $c \in C$, where C represents the set of all clusters, and θ_C signifies the set of all cluster model parameters.

We denote the data distribution in each cluster by $\mathcal{D}_c$, where $\mathcal{S}_c \sim \mathcal{D}c$ refers to the data points, and $s_1, ..., sN_c = \mathcal{S}_c$. By incorporating these clusters into equation (1), we derive the subsequent global and local loss function for each cluster c:

$$\min_{\theta_C} \mathcal{L}(\mathcal{S}) = \min_{\theta_C} \frac{1}{|C|} \sum_{c \in C} l_{\theta_c}(\mathcal{S}c)\ l\theta_c(\mathcal{S}c) \qquad = \frac{1}{|c|} \sum i \in c l_{\theta_c}(\mathcal{S}_i) \tag{6.8}$$

We can further personalize this model, enabling each client to have its unique model, leading to equation (2):

$$\min_{(\theta_1,\ldots,\theta_M)} \mathcal{L}(\mathcal{S}) = \min_{(\theta_1,\ldots,\theta_n)} \frac{1}{M} \sum_{i=1}^{M} l_{\theta_i}(\mathcal{S}_i) \tag{6.9}$$

Both equation (2) and equation (3) introduce additional flexibility, thereby enabling a more nuanced learning of the sampled training data points from the corresponding distributions,

compared to a single model as demonstrated in equation (1). The resultant inequality can be summarized as:

$$\min_{\theta_{global}} \mathcal{L}(\mathcal{S}) \geq \min_{\theta_C} \mathcal{L}(\mathcal{S}) \geq \min_{(\theta_1,\ldots,\theta_M)} \mathcal{L}(\mathcal{S}) \tag{6.10}$$

While Equation 6.10 establishes the concept of each client learning its own model to diminish the loss on its respective distribution, it's important to note that this approach could potentially limit the amount of data available for training. As illustrated by theoretical research in deep learning, the generalization error of models can increase if the volume of training samples decreases. This is also observed in practical scenarios and is exhibited as the overfitting phenomenon.

To uphold optimal generalization performance, we propose a multi-tiered training approach that captures information from the maximum number of training samples, simultaneously reducing the complexity each model needs to tackle in a step-wise manner. We initiate with a comprehensive training across all clients, aiming to identify a global model, denoted as θ_{global}, designed to learn basic features spanning the entire distribution. Following this, we cluster the clients and launch training within these clusters. The models derived from these clusters, represented as θ_C, are subsequently personalized in the final step by conducting local training on each client i, culminating in the creation of a uniquely tailored model θ_i for each client.

6.5.2 Experiments and Results

In this section, we outline the experimental design and share the findings from our conducted experiments. Detailed discussions concerning the dataset, data preprocessing, and the data split implementation for federated learning can be found in section 6.5.2. Our proposed approach, involving personalized models configured through clustering, is compared to two baseline methodologies. The first is a conventional supervised model designed within a centralized setting, and the second is FedAvg [158]. Additionally, we explore the results of integrating clustering and personalization elements with existing strategies, focusing specifically on the HAM10k skin lesion dataset [236].

Experimental Setup

In this experimental setup, we utilize a pre-trained MobileNetV2 model [199, 202], initially trained on ImageNet, as our foundational classifier for all the baselines as well as our proposed model.

Hyperparameter tuning was carried out with a designated validation set, ensuring no overlap with the test set for the centralized model. For the federated learning experiments, we implemented a two-step process for hyperparameter optimization: Initially, all clients were generated utilizing a consistent random seed, while hyperparameters were optimized against each client's respective test sets. Subsequently, we regenerated all clients using a different random seed for evaluation purposes, and a final learning curve was established based on the previously fixed hyperparameters.

Key hyperparameters that were optimized include the learning rate, batch size, and the number of local training epochs. The final values that were obtained were then consistently applied across all experiments. Models were optimized using the Stochastic Gradient Descent (SGD) optimizer, with a learning rate of 0.001, internal epochs set to $e = 1$, inner personalization epochs set to 7, inner batch size at 16, initial and final meta-learning rates defined as $\eta_0 = 1.0, \eta_i = 0.46$, total federated rounds $k = 220$, and meta batch size $n = 5$.

For the Federated Adaptive Personalization (FedAP) experiments, SGD was also employed for global training, resulting in the global update rule being defined as $\theta \leftarrow \theta + \eta \cdot \Delta\theta$. Furthermore, the adaptive weight was designed to decrease linearly with the number of federated rounds, transitioning from specified initial to final values.

In the experiments involving Hierarchical Clustering (HC), the models underwent training with 20 cluster initialization rounds, and utilized the Euclidean distance metric, a ward linkage mechanism, and a maximum distance of 5.

Dataset and Preprocessing The evaluation of our method employs the HAM10k dataset [236], a collection encompassing 10,015 dermatoscopic images depicting seven distinct types of skin lesions. Ground truth labels in HAM10k are ascertained through multiple methods - histopathology in over half of the cases, alongside follow-up examination, expert consensus, or in-vivo confocal microscopy. The preference for HAM10k is primarily driven by the dataset's inherent non-IID nature and significant imbalance. In order to mitigate the dataset's imbalance, we adopted a strategy of random undersampling [59], thus limiting the training set to a maximum of 500 randomly sampled images from each lesion class.

Federated Data Partitioning Clinical datasets frequently exhibit low statistical heterogeneity [255], a property that poses significant challenges to machine learning methodologies. With this in mind, our clients were designed to exhibit highly non-IID data distributions, effectively representing this ubiquitous problem. The images within each class were segmented into 35 groups, and clients were then randomly allocated two distinct partitions from different classes until all partition pairs were exhausted. This led to the creation of 34 clients, each receiving a total of 70 images from two classes (for a distribution heatmap, see the supplementary material). These 70 images were subsequently randomly partitioned into training and test sets, maintaining an 80:20 split within each client.

Results and Discussions

In this section, we showcase the outcomes of our experimental evaluation and benchmark our proposed model against preceding studies. To accommodate the number of personalization rounds within FedAP and initialization rounds in HC, we base the accuracy values reported in Table 6.4 on the cumulative number of training steps across all models.

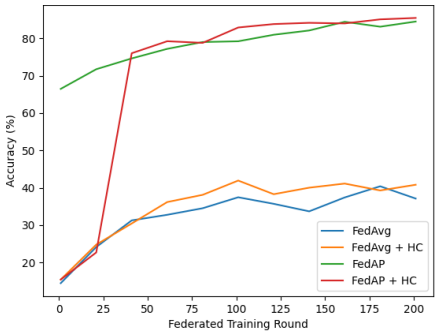


Fig. 6.5. **Test Accuracy Averages Across Various Rounds**. This figure depicts the classification accuracy, averaged over all clients, highlighting the improvement in the convergence rate facilitated by both FedAP and HC.

Method	Accuracy (%)
Centralized	76.8
FedAvg [158]	41.1 ± 34.31
FedAvg + HC [17]	44.1 ± 20.86
FedAP (Ours)	84.1 ± 14.53
FedAP + HC (Ours)	**86.9 ± 12.81**

Tab. 6.4. **Results**. This table offers a comparison of our methods versus the baselines on the non-IID HAM10k dataset. The values reported for federated models are based on the mean and standard deviation across all clients.

A conventional **centralized** training setting was established as a benchmark to measure the results of the more restricted federated learning experiments. In this setup, global training and test datasets were created by consolidating all the training and test data from each client. Given the access to data from all clients, this model yields significantly better performance compared to FedAvg.

FedAvg registered the least satisfactory performance among all the methods. Its learning curve exhibited substantial fluctuation, indicative of the single global model's inability to grasp the necessary features across diverse learning tasks, due to the differing data distributions across clients. This underscores FedAvg's vulnerability in highly non-IID data scenarios, aligning with initial indications from prior studies [138, 204, 288].

FedAvg + HC For the **FedAvg + HC** experiment, several rounds of FedAvg (defined by the "cluster initialization rounds" hyperparameter) were conducted prior to initiating the clustering process. Subsequently, clients were categorized based on their local updates using hierarchical clustering, as suggested by Briggs et al. [17]. Within these individual clusters, traditional *FedAvg* training was carried out. The global accuracy was derived by averaging the test set accuracy across all clients. The application of hierarchical clustering boosted the overall convergence speed of *FedAvg*. A noteworthy surge in accuracy, along with an enhancement in the final model performance, becomes apparent post-clustering in Figure 6.5. This suggests that a pseudo-homogeneous learning setting is reestablished within the clusters, validating the notion that model updates can indeed represent data distribution. Nevertheless, this approach still trails significantly behind *FedAP*, highlighting the indispensable role of model personalization in highly non-IID data settings.

FedAP Similar to *FedAvg*, **FedAP** draws a random sample of n clients at each round. The evaluation of *FedAP* followed the same process as FedAvg, with the exception that after the global model was disseminated, it was personalized to individual clients by executing several gradient optimization steps on their respective local training sets. Following this personalization, the model was assessed on the test set, and a global accuracy score was computed by averaging all local test set accuracies. The global model for *FedAP* demonstrated

continuous learning, while the personalization step facilitated adaptation to the diverse client distributions. This method, particularly the personalization aspect, performed exceptionally well under an extreme non-IID setting with simplistic underlying tasks.

FedAP + HC The **FedAP + HC** experiment mirrored the methodology of the previous one, with the distinction that adaptive personalization was implemented subsequent to hierarchical clustering. As in the *FedAvg + HC* experiment, clusters were created based on the clients' model updates after an initial 20 rounds of FedAvg. The *FedAP + HC* approach achieved the best overall performance in comparison to all other strategies. The learning curve behavior was akin to that observed in *FedAvg + HC*, as it expedited the algorithm's convergence speed, which is evidenced by the immediate surge in balanced accuracy following clustering, as depicted in Figure 6.5. Even though the FedAP + HC model surpassed other federated settings in terms of performance, it was found to be susceptible to overfitting during extended runs. In other words, if both FedAP and FedAP + HC were trained for an additional 500 rounds or so, the performance of FedAP remained constant or saw negligible improvement, whereas the performance of FedAP + HC began to decline. We theorize that this is indicative of the global model's enhanced adaptability to disparate tasks compared to the cluster models. The fundamental concept of meta-learning stems from the ability to learn from a multitude of distinct tasks. Given that clustering clients (which represents the meta-learning task in a federated learning scenario) reduces the diversity of tasks within each cluster to learn from, this could potentially explain the FedAP + HC model's proclivity towards overfitting during extended training periods.

6.6 Conclusion

In this chapter, we delve into two innovative methodologies tailored to federated learning, both designed to confront the issue of non-Independent and Identically Distributed (non-IID) data.

The first methodology introduces a novel weighting scheme for the aggregation of client models. This strategy is designed to operate efficiently in a federated learning context characterized by non-IID and unbalanced data distribution. By calculating weights based on statistical meta-information, this approach assigns higher aggregation weights to clients whose data is closer to the global average. We also propose an additional weighting technique, INTRAC, intended to normalize models and mitigate the influence of overfitted models on the shared model. This method proves particularly resilient to low-quality or poisonous data. Intriguingly, we found that if the majority of clients' models align well, they can counteract out-of-distribution models. This approach proves advantageous over Federated Averaging (FedAvg), which assumes that clients with more data should inherently have more influence on the global model. Our experiments reveal that this proposed method surpasses FedAvg in classification accuracy, particularly in non-IID scenarios.

Our second methodology for federated learning involves an adaptive model parameter weighting strategy combined with hierarchical clustering. This technique allows for better adaptation of the local client models to their unique data distribution, while still benefiting from the

global aggregation of various client model updates. By clustering clients based on the similarity of local model updates, we can approximate an IID setting within respective clusters. This approach has demonstrated significant improvements in classification accuracy when tested on the HAM10k dataset with the MobileNetV2 network, surpassing both the standard supervised learning and the FedAvg baseline. The reduced standard deviation in comparison to previous works indicates the superior generalizability of all client models. Moreover, hierarchical clustering increases the convergence speed and supports the creation of superior global models.

Both of these methods provide valuable contributions to federated learning, especially in the context of non-IID data.

7 Learning from Other Data

7.1 Introduction

Segmentation of medical images serves as a valuable tool in assisting healthcare professionals with diagnosis. The advent of deep learning has fostered high accuracy in organ and tumor segmentation [160, 193]. However, despite these advancements, conventional supervised learning scenarios often necessitate a significant amount of labeled data. While there can be an ample supply of labeled data for certain organs, such as the liver, it can be drastically limited for others. An initial solution to this constraint was transfer learning, a method where a neural network is pre-trained on a substantial labeled dataset (source domain) and subsequently fine-tuned on a smaller labeled data set (target domain) [191].

Few-shot learning, a technique that aspires to learn from a limited number of examples, has gained substantial popularity as another approach. One recent application of few-shot learning is the detection of COVID-19 using chest X-rays [92]. Few-shot methods can be broadly categorized into augmentation-based learning and task-based meta-learning. In this study, we concentrate on the meta-learning approach, which encompasses various forms: metric learning (for instance, prototypical networks [220, 247, 262]), memory-based learning [85, 245], and gradient-based methods. While few-shot learning for image segmentation has been a dynamic research area [5, 15, 62, 149, 214, 233, 234], there are a handful of works [37, 163] that concentrate on few-shot medical image segmentation. Another strategy to minimize the need for supervision in medical image segmentation involves the use of superpixels as pseudo-labels for self-supervised segmentation [173].

In this study, we adopt the meta-learning approach, specifically utilizing a recent adaptation of the Model-Agnostic Meta-Learning (MAML) algorithm—Reptile [167]. Reptile is a simple yet effective algorithm that affords expansive opportunities for experimentation. Our focus lies on two principal components of gradient-based meta-learning: task definition and gradient aggregation. Here, 'gradient aggregation' refers to the updating of the meta-model's weights, while 'task definition' involves the formulation of tasks by sampling pairs of images and their corresponding segmentation maps based on established criteria. We put forth a volume-based task definition explicitly crafted for volumetric data and introduce a weighting mechanism for the aggregation of gradients in each meta-training step. This mechanism is particularly advantageous when dealing with non-Independent and Identically Distributed (non-IID) data.

This work primarily introduces a novel concept of volumetric task definition. We demonstrate that task-specific optimization of local models can be enhanced by sampling data from a single volume per task, enabling better control over the variability of shots. This stands in contrast

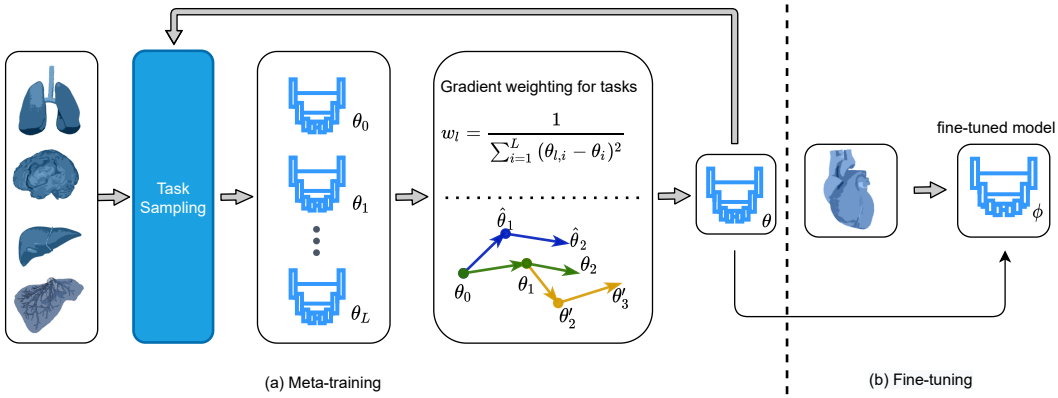


Fig. 7.1. The diagram on the left showcases the meta-training process conducted on source organs. This process involves defining specific meta-training tasks and assigning weights to these tasks according to their relative significance. Here, θ stands for the parameters of the meta-model, while ϕ' symbolizes the model currently under meta-training. Once meta-training is completed, the model, now characterized by the parameters ϕ, undergoes the final fine-tuning phase on the target organ.

to conventional settings where tasks are randomly sampled and may end up with images from similar parts of the volume (e.g., just the initial slices from different volumes). In our approach, we ensure a certain level of diversity among the shots while limiting the overall diversity of the source set, as each volume is linked to a fixed, limited number of shots (for example, 15), and other shots are not involved in training. As shown in [208], this approach can positively influence the training process.

Our second contribution lies in the development of an importance-aware weighting scheme. In the traditional Reptile framework, gradients of sampled tasks are averaged in each meta-epoch. Conversely, in our proposed method, gradients are weighted based on the importance of each task, defined as the distance between a local model trained on a given task and the average of all models trained on other tasks. This weighting mechanism has been previously proposed in the federated learning framework [269] for non-IID data. We argue that by attributing less weight to tasks that significantly deviate from the average model, we minimize the influence of outlier data, potentially averting catastrophic forgetting—a neural network's tendency to lose previously learned information [12]. This method could also serve as a regularization mechanism to prevent overfitting when tasks are similar, and enhance the training when the cross-domain distance is substantial.

Our evaluation demonstrates that the introduced volumetric task definition and weighted gradient aggregation enhance the accuracy of the segmentation. We assess our method under two distinct settings: (1) a few-shot setting, wherein models are fine-tuned using a small number of shots, and (2) a full-data setting, wherein the model is fine-tuned using all available data for the target organ. We juxtapose our results with several baselines: (1) supervised learning with random initialization, (2) supervised learning employing transfer learning initialization, (3) the Reptile baseline, both with and without our proposed volumetric task definition, and (4) few-shot cell segmentation by Dawoud et al. [37], which is the most directly related work, tested both with and without our proposed volumetric task definition. An overview of our method is depicted in Figure 7.1.

In summary, we introduce MetaMedSeg [1], an innovative meta-learning strategy tailored for medical image segmentation. The principal contributions of this research include: (1) The development of a new task definition, rooted in data volumes, specifically designed to address medical scenarios; (2) The implementation of a unique update rule optimized for few-shot learning in contexts with high cross-domain distance; and (3) A substantial enhancement in segmentation performance in comparison to traditional methodologies.

7.2 MetaMedSeg

Consider a dataset $\mathcal{D} = \mathcal{S}, \mathcal{T}$, where $\mathcal{S} = \mathcal{S}_1 \cdots \mathcal{S}_n$ symbolizes the training set or source domain, and $\mathcal{T} = \mathcal{T}_1 \cdots \mathcal{T}_m$ signifies the test set or target domain. Here, n and m represent the total number of organ datasets within the source and target domains, respectively, and there is no intersection between $\mathcal{T}$ and $\mathcal{S}$, hence $\mathcal{T} \cap \mathcal{S} = \emptyset$. Each dataset is made up of paired images and their corresponding segmentation masks. Therefore, we define a task as a subset of k shots, sampled from either $\mathcal{S}_i$ or $\mathcal{T}_j$.

The learning process comprises two stages: meta-training and fine-tuning. The model parameters derived from the meta-training step are represented by θ_i, while the parameters of the fine-tuned model are signified by ϕ. During each meta-training step, each task learns its distinct set of parameters, denoted as θ_l, starting with meta-model weights θ. The network structure employed in this research is the renowned U-Net [193] architecture, frequently used for medical image segmentation. The network accepts a batch of images $\mathcal{I}_b$ as input, and its outputs are the segmentation maps y_b. The ground-truth segmentation maps are signified by y'_b. Instead of batch normalization, we opt for the instance normalization technique [20, 96] for the meta-learning context. The algorithm at the core of our approach is outlined in algorithm 1. Our work's principal components are: 1. *Meta-learning* 2. *Image Segmentation*, each of which is elaborated below.

Algorithm 1: MetaMedSeg for organ segmentation

Input: Meta-training datasets $\mathcal{S} = \mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_n$
Input: Meta-testing dataset T
Initialize: θ
for *meta-epoch = 1,2,...,N* **do**
    Sample L datasets from meta-training datasets $\mathcal{S}$
    for l *= 1,2,...,L* **do**
        Sample K shots from dataset $\mathcal{S}_l$ using rule $\mathcal{R}$
        Train base learner (U-Net) to obtain $\theta_l = g(L(\theta, \mathcal{S}_l))$
    Compute task importance w_l
    Perform meta-update: $\theta \leftarrow \theta + \beta \sum_{l=1}^{L} w_l(\theta_l - \theta)$
Sample K' shots from meta-testing dataset T to generate $T' \in T$
Fine-tune on T'
Compute test IoU on $T'' \in T, T' \cap T'' = \emptyset$

[1] Project page: `http://metamedseg.github.io/`

7.2.1 Meta-learning

As illustrated in fig. 7.1, during each meta-training round, a series of tasks —composed of images and their matching segmentation maps— are selected based on the predetermined rule and fed into the U-Net model. The weights learned from these models are subsequently aggregated according to the specified update rule. The ultimate model serves as the initialization for the fine-tuning stage. In the conventional Reptile algorithm [167], the updates acquired from all tasks are averaged in each meta-epoch. In contrast, we put forth an alternative strategy for weighting these updates, contingent upon the importance of the tasks. The task definition, task sampling, and update rules will be detailed in the subsequent sections.

Task definition

The definition of a task in meta-learning for segmentation can vary. Our initial approach takes its cue from [37], where a task consists of k images and masks from the same dataset. For instance, a task could involve k-shots randomly selected from all available slices of vessel cancer. This method primarily caters to 2D data. While our work also involves 2D slices, there exists additional information that can be exploited. Our data comprises not just a collection of images, but an assemblage of volumes (3D tensors that, when sliced along a chosen direction, yield sets of 2D images). Hence, we put forth a volume-based task definition. We define a task as a group of images originating from the same volume $\mathcal{V}$, sampled with a step size equal to $\left\lceil \frac{|\mathcal{V}|}{K} \right\rceil$. This assures task size balance across datasets. It's noteworthy that the support and query set do not come from the same volume.

Weighted Task Sampling

Certain organs exhibit varied modalities and/or zones (for example, the prostate can be partitioned into peripheral and transitional zones). We treat these as separate datasets. One could anticipate this leading to some organs holding sway during task sampling (for instance, the BRATS dataset alone translates into 12 distinct source datasets). To offset this, we propose using weighted sampling to afford each organ an equitable opportunity to be included in the task set during each meta-epoch. For each organ with z different modalities or zones, we set the sampling rate for each modality/zone to $\frac{1}{z}$ and then normalize them.

Importance-Aware Task Weighting

In our method, we utilize the original Reptile [167] update rule as a baseline for comparison:

$$\theta \leftarrow \theta + \beta \frac{1}{L} \sum_{l=1}^{L} (\theta_l - \theta), \tag{7.1}$$

where θ_l represents the weight vector of the local model, θ stands for the weight vector of the meta-model, L as the number of tasks, and β as the meta learning rate.

We assess the following configurations:

- Average Weighting (AW): Here, all updates are deemed to have equal weights and thus, $(\theta_l - \theta)$ are averaged across tasks.

- Inverse Distance Weighting (IDW): In this case, higher weight is assigned to models that are closer to the meta-model.

The weights for AW update are computed based on the number of tasks sampled in each meta-epoch. With L tasks sampled, the weights for each of these tasks would be: $\frac{1}{L}$. The weight for the inverse distance update is computed as:

$$w_l = \frac{1}{\sum_{i=1}^{L} (\theta_{l,i} - \theta_i)^2}, \tag{7.2}$$

where θ_i denotes the i-th weight of the meta-model, while $w_{l,i}$ represents the i-th weight of the l-th task's model in the current meta-epoch. The weights are normalized so that their sum equals 1, which can be done using $w_l = \frac{w_l}{\sum_j^L w_j}$. As such, the update rule can be written as:

$$\theta \leftarrow \theta + \beta \sum_{l=1}^{L} w_l(\theta_l - \theta), \tag{7.3}$$

7.2.2 Image Segmentation

Binary Cross Entropy (BCE) is commonly employed as a loss function for optimizing image segmentation models. However, in the few-shot setting, our initial trials revealed suboptimal performance. Consequently, we explored alternate loss functions. We utilized a combined approach, incorporating weighted BCE loss and an approximation of the Intersection over Union (IoU) loss [183] for segmentation tasks.

Weighted BCE Loss

For a given input image $\mathcal{I}$, the predicted segmentation map y, and the ground truth segmentation map y', the weighted BCE loss can be computed as follows:

$$BCE(y, y') = -(p_{\text{pos}} y \log(y') + (1 - y) \log(1 - y')) \tag{7.4}$$

Here, p_{pos} represents the weight of positive samples, calculated as the ratio of the sum of object pixels to the sum of background pixels.

IoU Loss

For the IoU loss component, we adopted the IoU approximation suggested by Rahman et al. [183], which is defined as:

$$\begin{aligned} \mathcal{X}(y, y') &= \sum_{i \in \mathcal{P}} y_i * y'_i \\ \mathcal{U}(y, y') &= \sum_{i \in \mathcal{P}} (y_i + y'_i - y_i * y'_i) \\ IoU(y, y') &= \frac{\mathcal{X} + \epsilon}{\mathcal{U} + \epsilon}, \end{aligned} \tag{7.5}$$

The variable $\mathcal{P}$ represents the set of all pixel values in the training images, while y_i and y'_i signify the ground truth pixel value (0 for background, 1 for object) and the probability respectively. The probability is determined by the sigmoid output of the U-Net. The inclusion of ϵ in the equation helps ensure numerical stability.

Ultimately, we also conducted experiments using a combination of BCE and logarithmic DICE loss, applying the following formula (Equation 7.6):

$$Q(L(y, y') = BCE(y, y') - \log\left(\frac{2IoU(y, y')}{IoU(y, y') + 1}\right) \tag{7.6}$$

which is derived in Equation 7.7:

$$\begin{aligned} L(y, y') = \\ & BCE(y, y') - \log(DICE) = \\ & BCE(y, y') - \log\left(\frac{2\sum_{i \in \mathcal{P}} y_i * y'_i}{\sum_{i \in \mathcal{P}} y_i + \sum_{i \in \mathcal{P}} y'_i}\right) = \\ & BCE(y, y') - \log\left(\frac{2\sum_{i \in \mathcal{P}} y_i * y'_i}{\sum_{i \in \mathcal{P}} y_i + \sum_{i \in \mathcal{P}} y'_i - \sum_{i \in \mathcal{P}} y_i * y'_i + \sum_{i \in \mathcal{P}} y_i * y'_i}\right) = \\ & BCE(y, y') - \log\left(\frac{2X}{U + X}\right) = BCE(y, y') - \log\left(\frac{2X * U}{U * (U + X)}\right) = \\ & BCE(y, y') - \log\left(\frac{\frac{2X}{U}}{\frac{X+U}{U}}\right) = BCE(y, y') - \log\left(\frac{2IoU(y, y')}{IoU(y, y') + 1},\right) \end{aligned} \tag{7.7}$$

7.3 Experiments

In this section, we present the results of the experiments using our proposed methodologies and compare them against the related work.

Tab. 7.1. Comparison of the proposed task sampling and weighting schemes to related work in the few-shot setting, fine-tuned on 15 shots on 4 organs. **AW**: Average weighting, **IDW**: Inverse distance weighting.

Update rule	Target organ	IoU ↑	
		Standard	Volume-based
Supervised Learning	Cardiac	58.78	–
Transfer Learning	Cardiac	66.22	–
Dawoud et al. [37]	Cardiac	68.28	67.7
MetaMedSeg + IDW (Ours)	Cardiac	67.49	64.86
MetaMedSeg + AW (Ours)	Cardiac	67.83	**68.33**
Supervised Learning	Spleen	38.81	–
Transfer Learning	Spleen	51.18	–
Dawoud et al. [37]	Spleen	49.29	**58.34**
MetaMedSeg + IDW (Ours)	Spleen	55.64	50.50
MetaMedSeg + AW (Ours)	Spleen	55.98	56.44
Supervised Learning	Prostate Peripheral	7.35	–
Transfer Learning	Prostate Peripheral	10.87	–
Dawoud et al. [37]	Prostate Peripheral	15.99	12.82
MetaMedSeg + IDW (Ours)	Prostate Peripheral	17.15	13.89
MetaMedSeg + AW (Ours)	Prostate Peripheral	16.17	**22.69**
Supervised Learning	Prostate Transitional	38.64	–
Transfer Learning	Prostate Transitional	41.09	–
Dawoud et al. [37]	Prostate Transitional	42.85	46.28
MetaMedSeg + IDW (Ours)	Prostate Transitional	44.25	44.72
MetaMedSeg + AW (Ours)	Prostate Transitional	42.43	**48.33**

Tab. 7.2. Comparison of the proposed task sampling and weighting schemes to related work in full-data setting on 4 different organs. **AW**: Average weighting, **IDW**: Inverse distance weighting.

Update rule	Target organ	IoU ↑	
		Standard	Volume-based
Supervised Learning	Cardiac	90.26	–
Transfer Learning	Cardiac	90.46	–
Dawoud et al. [37]	Cardiac	90.38	92.47
MetaMedSeg + IDW (Ours)	Cardiac	91.38	94.51
MetaMedSeg + AW (Ours)	Cardiac	91.08	**95.55**
Supervised Learning	Spleen	86.74	–
Transfer Learning	Spleen	86.10	–
Dawoud et al. [37]	Spleen	89.65	87.53
MetaMedSeg + IDW (Ours)	Spleen	89.96	91.53
MetaMedSeg + AW (Ours)	Spleen	90.00	**92.27**
Supervised Learning	Prostate Peripheral	39.06	–
Transfer Learning	Prostate Peripheral	39.94	–
Dawoud et al. [37]	Prostate Peripheral	37.05	41.58
MetaMedSeg + IDW (Ours)	Prostate Peripheral	47.58	68.26
MetaMedSeg + AW (Ours)	Prostate Peripheral	46.20	**70.90**
Supervised Learning	Prostate Transitional	68.04	–
Transfer Learning	Prostate Transitional	69.84	–
Dawoud et al. [37]	Prostate Transitional	70.42	71.55
MetaMedSeg + IDW (Ours)	Prostate Transitional	68.98	**79.95**
MetaMedSeg + AW (Ours)	Prostate Transitional	67.68	78.84

We evaluate our methodology using the medical decathlon dataset [217]. This dataset encompasses 3D MRI and CT scans from nine distinct organs: the Brain (with 368 volumes), Hippocampus (260 volumes), Lung (25 volumes), Prostate (32 volumes), Cardiac (20 volumes), Pancreas (279 volumes), Colon (121 volumes), Hepatic Vessels (216 volumes), and Spleen (41 volumes). For the architectural backbone of our model, we employ the U-Net architecture [193], and optimize our model using IoU, BCE, and their hybrid combination as loss functions.

As a benchmark for comparison, we utilize a transfer learning strategy where all available training data are used to derive initial weights for fine-tuning the model.

To validate the robustness of our results and mitigate potential bias from k-shot selection, we fine-tune our model over five different random selections, consistently testing on the same test set of unseen data. Subsequently, the evaluation metric (IoU) is averaged across these five runs.

7.3.1 Experimental Setup

The dataset is prepared for processing by segregating different techniques (for example, T2 and FLAIR) and distinct regions (such as edema and tumor) into separate datasets, leading to a total of 24 distinct datasets. Each dataset has an associated threshold, established based on pixel count and a visual assessment of the results, which determines whether an object is deemed present on the image.

Thresholds for different organs and zones include: 1000 for brain edema, 400 for enhancing tumors, and 600 for non-enhancing tumors; 700 for cardiac; 100 for both anterior and posterior hippocampus; 300 for the peripheral zone and 600 for the transitional zone of the prostate; 1000 for the lung, pancreas and vessel; 600 for the spleen; and 400 for the colon.

All images were resized to a resolution of 256×256 pixels, with the threshold applied post-resizing.

We also ensure volume normalization during slicing by subtracting the mean and dividing by the standard deviation of all the non-zero pixels within the entire volume. The conditions for all fine-tuning experiments are standardized: we train for 20 epochs, applying a weight decay of 3×10^{-5} and a learning rate of $\alpha = 0.005$ with a step learning rate decay of $\gamma = 0.7$ at every alternate step. For full-data training, we use a learning rate of 0.001 and weight decay of $w = 3 \times 10^{-5}$.

For the meta-training phase, we train the model for 100 epochs, sampling 5 tasks with 15 shots and 1 image per shot at each meta-epoch. We employed a learning rate of $\alpha = 0.01$ for both local and meta-model, with a weight decay of $w = 0.003$ and the aforementioned learning rate decay. In the transfer learning phase, we exclude cardiac, prostate, and spleen datasets and train over 20 epochs. Hyperparameter tuning is achieved through the particle swarm optimization approach, with no augmentations performed on the data across any of the settings.

To facilitate comparison with [37], we replicate the original paper's hyperparameters for meta-training. It's important to note that, as per the protocol from [37], we sequentially traverse all available source datasets rather than sampling them. Additionally, for each task, we sample 2 extra tasks from different source datasets to create 2 additional losses. For fine-tuning, we also adhere to the approach outlined in [37], but extend training to 40 epochs instead of 20 to enhance performance, and use IoU loss in place of BCE.

7.3.2 Comparison to Related Work

Table 7.1 presents a comparison of our methodology with baseline methods in a few-shot setting, specifically for 15 shots across four different organs. The performance metrics for models fine-tuned on the entirety of the support set are presented in table 7.2. A selection of segmentation results can be found in fig. 7.2 and fig. 7.3. Additionally, we experimented with an inverse form of the proposed update rule, which penalizes outlier models. However, this alteration did not yield any improvement.

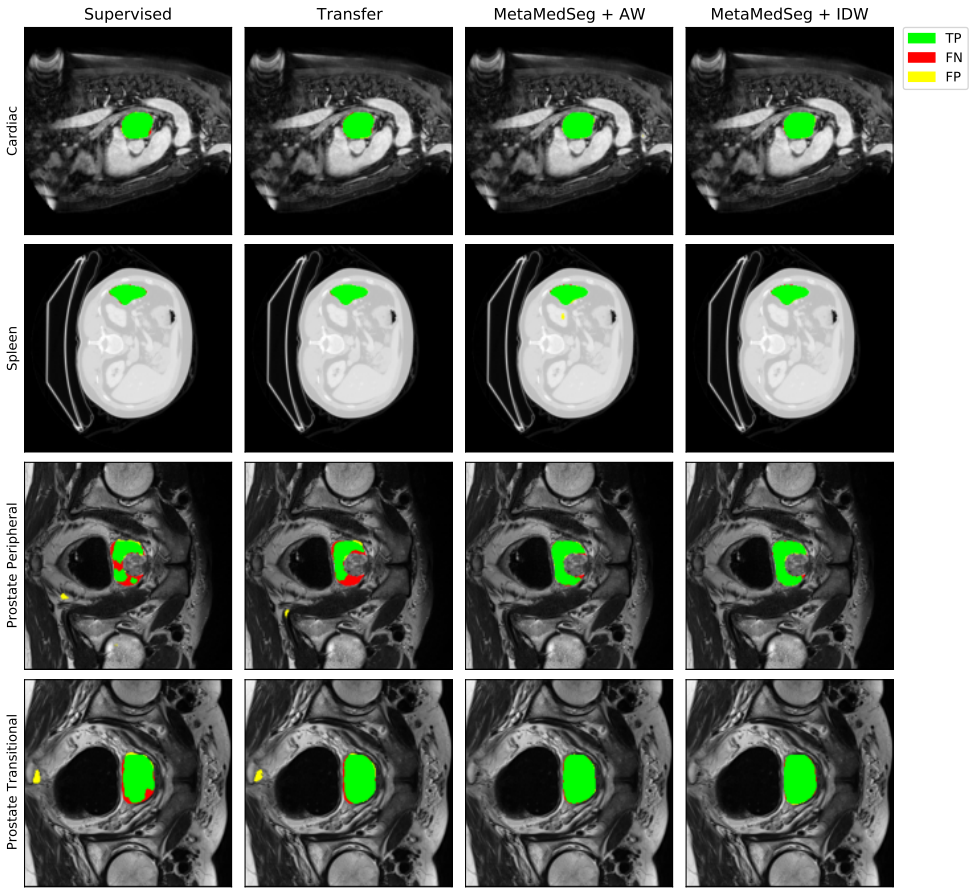


Fig. 7.2. A qualitative comparison of our method to different segmentation baselines on four different target organs in the full-data setting.

7.3.3 Discussion

Our experimental results indicate that volume-based task formulation significantly enhances the performance of segmentation, especially in settings involving full data. The impact of our suggested update rule is marginal in some instances, but notably influential in other organs, like the Prostate Transitional. This differential influence could be due to the variation in data distribution and organ morphology. When organ shapes demonstrate high variability (e.g., the prostate in comparison to other organs), outlier shapes could potentially benefit from the proposed inverse distance update rule, which assigns less weight to gradients farther from the average.

We believe that when the volumetric task definition is employed, all organs are given an equal opportunity to contribute to the final model, resulting in a more balanced setting. Consequently, when the weighted update rule is combined with volumetric tasks, the overall performance experiences a decrease. Another plausible explanation could be that the weighting of updates aids in biasing the model towards tasks that bear greater resemblance to the target.

To better understand the role of image diversity, we construct an adjacency matrix depicting the average Euclidean distances between pairs of randomly selected volumes. As illustrated in Figure 7.4, among our target organs, the distances, from the least to the greatest, are as follows: cardiac, spleen, prostate. The influence of this distance variation is perceptible in the final segmentation performance for each organ.

7.3.4 Ablation Study

Table 7.3 displays the influence of our proposed update rule, various segmentation losses, and the volume-based task formulation employed in this study. The models underwent meta-training with five distinct losses, including Focal Tversky loss [2] and Dice loss [222], followed by fine-tuning using IoU loss. The most effective performance is attained when employing the weighted BCE loss for meta-training and IoU loss for fine-tuning.

Tab. 7.3. Ablation study of various objective functions in few-shot setting for the cardiac segmentation task. **AW**: Average weighting, **IDW**: Inverse distance weighting.

Update rule	Segmentation loss	IoU ↑	
		Standard	Volume-based
AW	IoU	67.82	68.13
IDW	IoU	67.42	64.86
AW	Tversky Focal loss [2]	65.90	65.72
IDW	Tversky Focal loss [2]	63.71	62.78
AW	Dice loss [222]	65.87	66.03
IDW	Dice loss [222]	62.58	62.73
AW	BCE	**67.83**	**68.33**
IDW	BCE	65.09	64.29
AW	BCE + IoU	66.85	67.30
IDW	BCE + IoU	66.98	64.71

7.4 Conclusion

In this work, we have proposed an innovative task definition approach for few-shot learning that caters specifically to volume-based 2D data. Additionally, we've introduced an update rule that is contingent on the importance of tasks during the meta-training phase. We tested our method on four distinct organ types that have minimal data available, specifically the heart, spleen, peripheral prostate, and transitional prostate. Our approach is versatile and applicable not only to organ segmentation, but also to tumor segmentation and other types of density segmentation.

Our results demonstrate that our proposed approach to defining volumetric tasks significantly bolsters the performance of segmentation across all organs tested. The introduction of update rules based on importance also contributes substantially to the improvement of IoU results. The benefits of both the volumetric tasks and the weighted update rule vary according to different scenarios. While the volumetric task definition is consistently advantageous, the use of a weighted update rule is particularly beneficial when the target class data distribution diverges from the source. This is especially the case in the segmentation of new diseases where the available labeled data is scarce.

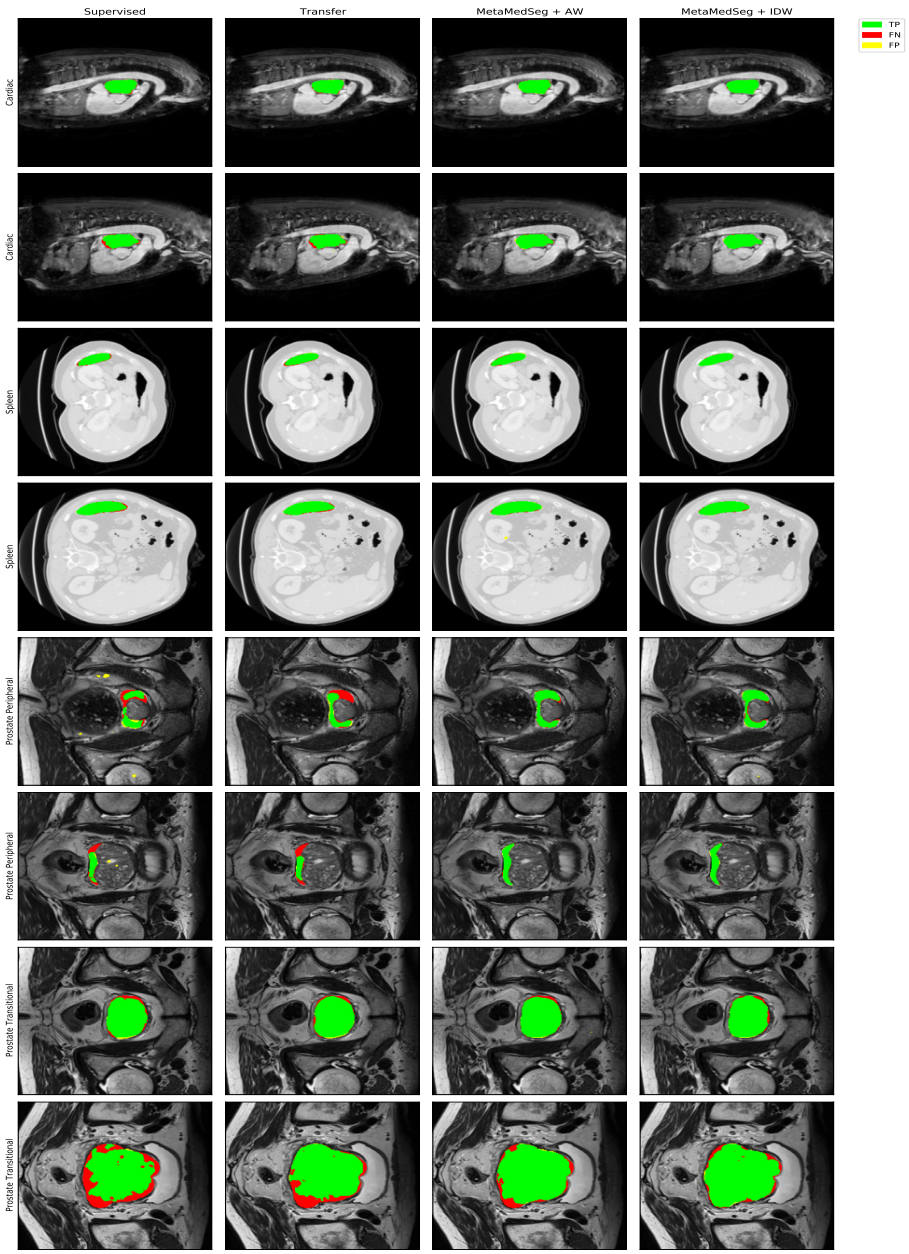


Fig. 7.3. Additional qualitative results for organ segmentation

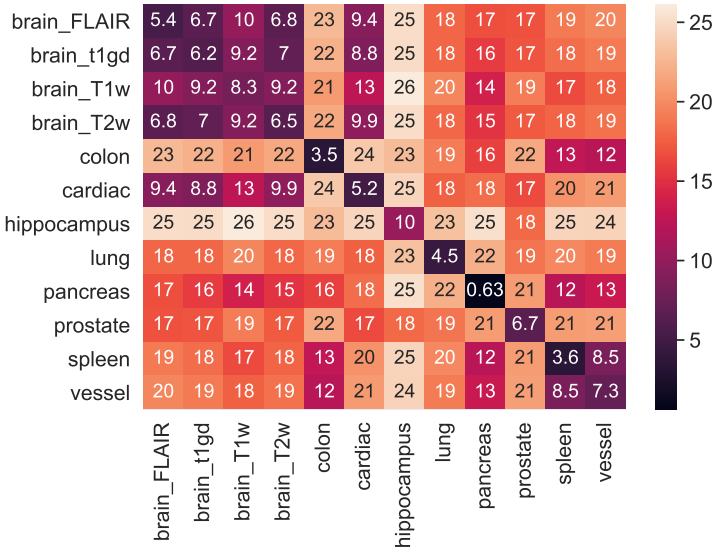


Fig. 7.4. Heatmap of the average distances between pairs of randomly sampled slices from different organs

8 Learning from Within the Data

8.1 Introduction

Optical Coherence Tomography (OCT) of the eye is a frequently used clinical imaging technique in ophthalmology, especially by retina specialists. Segmentation of OCT images aids in diagnosing and treating ocular conditions such as diabetic macular edema (DME) [243] and age-related macular degeneration (AMD). Particularly, the segmentation of intraretinal fluid pockets proves crucial, as it indicates the presence, severity, and treatment response of the retina. Despite the significance of fluid segmentation in OCT images, current methods struggle to efficiently delineate these regions. To address this, our work proposes harnessing spectral domain information due to the presence of spectral features in OCT images, which might be overlooked by existing spatial neural networks. It has been demonstrated in previous research [28] that spatial information often emphasizes local details while overlooking global information distributed across the entire image. Our approach resolves this issue by integrating features from both the spectral and spatial domains.

To summarize, our main contributions are:

- We introduce Y-Net, a dual-encoder autoencoder-based architecture, for automated segmentation of retinal layers and fluids in OCT images.
- Our spectral encoder is specifically designed to extract frequency domain features from images.
- Y-Net surpasses the widely-recognized U-Net [193] architecture and other related work, improving fluid segmentation by at least 13% and dice score by an average of 1.9%.
- Y-Net operates with fewer parameters compared to U-Net.
- We have made the source code for this work publicly accessible at `github.com/azadef/ynet`.

8.2 Related Work

Several early techniques for retinal OCT image segmentation [29] leveraged graph-based strategies (such as graph cut, shortest path). Later works [49, 132] integrated neural networks with graph-based methods to better estimate retinal layer boundaries or fused graph convolutional networks with other neural networks.

He et al. conducted a series of investigations [71, 72, 73] into OCT segmentation, with a focus on OCT scan topology. The utilization of fully convolutional networks (FCN) was examined [74, 124] with the objective of predicting segmentation maps and refining topology based on specific topology criteria.

A recent trend in medical image segmentation emphasizes the use of autoencoder-based deep neural networks [117, 195] for end-to-end segmentation. U-Net [193], one of the earliest and most widely recognized autoencoder-based architectures for 2D medical image segmentation, spurred considerable research interest in the development of U-shaped networks for image segmentation. Several studies, including MDAN-U-Net [150], have sought to enhance the segmentation performance of existing methods through multiscale features or attention mechanisms. Feature Pyramid Networks (FPNs), commonly utilized in computer vision, have also piqued interest in the medical imaging domain for global feature extraction [58, 134]. Other research directions have concentrated on networks specifically tailored for the OCT segmentation task [117, 187, 253], employing techniques like Gaussian process [177], feature alignment [44, 157], or epistemic uncertainty [172].

The application of Recurrent Neural Networks (RNNs) for OCT segmentation has been investigated in [125, 235]. While Kugelman et al. [125] considered sequences across various scans, Tran et al. [235] used natural language to model OCT retinal layers and created an OCT segmentation approach utilizing RNNs for pixel sequence processing.

The concept of an autoencoder network with two encoder branches has been previously deployed for polyp detection [162] leveraging a pre-trained VGG network [216]. However, the goal of this application significantly differs from ours. A combination of U-Net [193] and fast Fourier transforms (FFT) [164] has been explored as a means to reduce the computational demands of convolutional networks. Recently, fast Fourier convolutions [28] were integrated into an image inpainting task [225] by the computer vision community, aiming to utilize global patterns in images potentially overlooked by standard convolutional layers. This inspired us to exploit fast Fourier convolutions for OCT segmentation, given the presence of high-frequency speckles, a characteristic of tissue layers [206]. The existence of these speckles can negatively impact model performance when only spatial features are used. Therefore, we hypothesize that spectral feature extraction from OCT images will enable our network to separate features from different frequency distributions. This will allow the model to focus on more significant frequency ranges in the features using adaptive learnable kernels in FFT Convolutions, and to effectively model the high-frequency variation and distribution within each layer.

8.3 Method

In this section, we delve into the foundational aspects of our work. We initially outline the overall architecture of the segmentation framework. Subsequently, we elucidate the elements of our proposed spectral encoder, with a focus on the Fourier unit that fulfills the spectral feature extraction function. Lastly, we discuss the loss functions employed in this study.

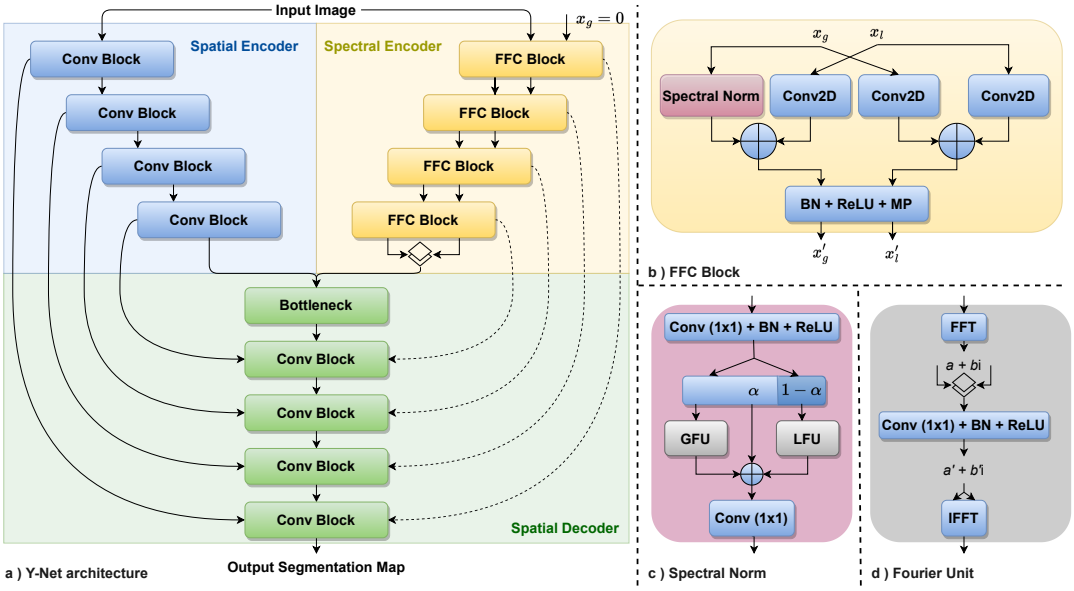


Fig. 8.1. **a) Y-Net:** Our proposed network comprises two branches: one for handling spatial features, akin to previous works, and our proposed branch designed for extracting spectral features. The spectral encoder incorporates four Fast Fourier Convolution (FFC) blocks, which take local and global features x_l, x_g as input and produce processed features x'_l, x'_g. **b) FFC Block:** The FFC blocks extract the local features via Conv2D layers and process the global features utilizing the spectral norm. **c) Spectral Norm:** The global information is partitioned into two parts, each of which is delivered to a Fourier unit. **d) Fourier Unit:** In this step, a fast Fourier transform is applied to the features, followed by a convolutional layer, to extract frequency domain features. Ultimately, the processed features are reverted back to the spatial domain using an inverse FFT.

8.3.1 Segmentation Framework

Our segmentation network takes an input image $x \in \mathbb{R}^{H \times W}$ and its associated segmentation label $y \in \mathbb{Z}^{H \times W}$ to generate the segmentation map $\hat{y}$. Here, H and W denote the image height and width, respectively. As depicted in Figure 8.1-a, the Y-Net segmentation network incorporates two encoder branches, E_c and E_f. The E_c serves as the spatial encoder comprising convolutional blocks, while E_f is our proposed spectral encoder, integrating fast Fourier convolutional (FFC) blocks [28]. The decoder network $G(.)$ gets the fused spatial and spectral features derived from the encoder networks to produce the segmentation map $\hat{y}$, where $\hat{y} = G(E_c(x), E_f(x))$. Resembling U-Net [193], Y-Net possesses an autoencoder-based structure, with skip connections transitioning from spatial encoder blocks to decoder blocks. The primary function of the proposed spectral encoder is to extract and process global features in the frequency domain, which may not have been captured by spatial convolutions. This section further elaborates on each component of our network and the objective functions.

Spatial Encoder

Our network incorporates a spatial encoder identical to the original U-Net [193], which includes four convolutional blocks. Each block is constituted by a convolutional layer, a batch normalization layer (BN), an activation function (ReLU), and a max pooling (MP) layer. The input to the initial convolutional block is the input image, and the output from each block is subsequently fed into the next block, as depicted in Figure 8.1-a.

Spectral Encoder

In this section, we introduce our spectral encoder, designed to extract features from the spectral domain. The spectral encoder receives the same input as the spatial encoder. The input image x serves as local information x_l for the first FFC block. The value of x_g is initialized to zeros for the first block, as the input image pixels are treated as local information, and there are no global features present in the input image. Mirroring the spatial encoder, the spectral encoder comprises a total of four FFC blocks.

Spatial Decoder

The spatial decoder network, denoted as G, is comprised of four up-convolutional blocks in total. It accepts the spectral and spatial features, concatenates them, and then directs them to the bottleneck layer. Following this, the features from the previous decoder block and the features from the skip connections are upscaled using a convolutional block similar to the spatial encoder, followed by transpose convolutional layers. The final segmentation map is produced by the ultimate decoder block. As an optional step, we consider the concatenation of the features from the spectral encoder with the features from the spatial encoder.

8.3.2 Components of the Spectral Encoder

Fast Fourier Convolutional Block

The Fast Fourier Convolutional (FFC) block, as shown in Figure 8.1-b, is designed to accept global and local information, x_g and x_l, as input. This information is then processed through three convolutional layers to extract global and local spatial features, as well as through the spectral norm to extract frequency domain features. Subsequently, batch normalization, a non-linear activation function, and max pooling are applied to these features, thereby generating the global and local features, x'_g and x'_l, for the next FFC block.

Spectral Norm

The spectral norm, depicted in Figure 8.1-c, first applies a convolutional block with a kernel size of 1 to x_g, yielding x''. The channels of x'' are then divided into two portions based on a predefined value α, with α percent of channels designated as global information and the remaining $1 - \alpha$ percent as local information. The divided global and local features are independently fed into Fourier units (FU_g and FU_l) to extract spectral features x''_g and x''_l. Note that FU_g and FU_l share the same architectural design. Lastly, x'' and the output of global and local Fourier units x''_g and x''_l are added and fed into a convolutional layer with a kernel size of 1.

Fourier Unit

As shown in Figure 8.1-d, the Fourier unit accepts a portion of x'' as input, then performs a Fourier transform on those features to obtain real and imaginary parts represented as $a + bi \in \mathbb{C}$. These real and imaginary parts, a and b, are stacked together and then processed through a convolutional layer with a kernel size of 1. The output of this layer is then subjected to an activation layer and a batch normalization layer. The processed output is then divided

into two parts, specifically the real and imaginary parts a' and b', which are fed into the inverse Fourier transform to convert the features back to the spatial domain.

8.3.3 Objective Functions

We utilize a combined version [227] of dice loss [222] and cross-entropy loss to train our models. This particular pairing of losses has been frequently employed in medical image segmentation task.

The loss is computed between each ground truth segmentation label y and its corresponding predicted segmentation map $\hat{y}$ according to Equation 8.1:

$$\mathcal{L}_{Dice}(y, \hat{y}) = 1 - \frac{2y\hat{y} + \epsilon}{y + \hat{y} + \epsilon} \tag{8.1}$$

The dice loss in Equation 8.1, evaluates the intersection over union (IoU) between the predicted and true labels. To ensure numerical stability during the computation of dice loss, we use a very small value, ϵ.

The cross-entropy loss, as outlined in Equation 8.2, strives to maximize the cross-entropy information between the true labels and the predictions.

$$\mathcal{L}_{CE}(y, \hat{y}) = -\frac{1}{N}\sum_{i=0}^{N} y_i \log(\hat{y}_i) \tag{8.2}$$

The total loss is then computed as a weighted sum of the dice loss and the cross-entropy loss, where λ_{Dice} and λ_{CE} are the weighting factors for each respective loss term.

$$\mathcal{L}_{total} = \lambda_{Dice}\mathcal{L}_{Dice} + \lambda_{CE}\mathcal{L}_{CE} \tag{8.3}$$

The combined loss function ensures effective segmentation performance by leveraging the strengths of both individual losses.

8.4 Experiments and Results

In this section, we assess our proposed method's performance by contrasting it with well-established benchmarks and prior research. We begin with an outline of our experimental setup, followed by a comparison of our model's results with those reported in previous studies. Finally, we conduct an ablation study to understand the contribution of individual components of our model. As previously discussed, our focus is on the segmentation of retinal layers and fluid using OCT images.

Tab. 8.1. **Comparison to SOTA on Duke.** Mean and layer-wise Dice Score compared to related works on the Duke OCT dataset [30]

Method	ILM	NFL-IPL	INL	OPL	ONL-ISM	ISE	OS-RPE	Fluid	Mean
RelayNet [195]	0.84	0.85	0.70	0.71	0.87	0.88	0.84	0.30	0.75
Language [235]	0.85	0.89	0.75	0.75	0.89	**0.90**	**0.87**	0.39	0.78
Alignment [157]	0.85	0.89	0.75	0.74	0.90	**0.90**	**0.87**	0.56	0.81
U-Net [193]	0.84	0.89	0.77	**0.76**	0.89	0.89	0.85	0.80	0.836
Y-Net (Ours)	**0.86**	**0.89**	**0.78**	0.75	**0.90**	0.88	0.85	**0.93**	**0.855**

Tab. 8.2. **Comparison to UNet on UMN.** Mean Dice Score and mIoU compared to U-Net on the UMN dataset [185] for fluid segmentation.

Method	DSC	mIoU
U-Net [193]	0.91	0.80
Y-Net (Ours)	**1.0**	**0.86**

Our method is trained and tested on the Duke OCT dataset[1] [30], and the UMN fluid segmentation dataset [185], two publicly available retinal OCT datasets. We benchmark our results against several seminal works on OCT segmentation. Except for U-Net [193], all results reported are derived from the original values published in the respective papers. The results attributed to RelayNet [195] are taken from [235] and are based on a 6-2-2 evaluation split.

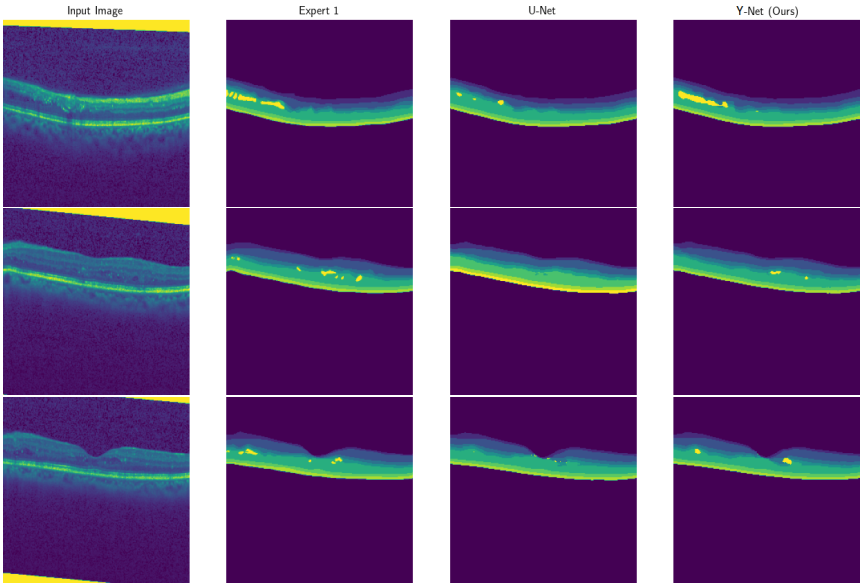


Fig. 8.2. Some qualitative results of Y-Net compared to U-Net [193]

8.4.1 Experimental Setup

Our experimental procedure adheres to the same protocol as previous work [157, 235] when training and evaluating our method on the Duke OCT dataset. This dataset consists of OCT scans from ten patients, annotated by two experts. We utilized scans from the initial six subjects for training, the seventh and eighth subjects for validation, and the last two for

[1] Link: http://people.duke.edu/~sf59/Chiu_BOE_2014_dataset.htm

testing, following the same distribution as prior studies. All models, including ours and U-Net [193], were trained with a batch size of 10, a learning rate of 5×10^{-4}, weight decay of 1×10^{-4}, and the Adam optimizer [118], over a maximum of 80 training epochs. The training epoch count was selected based on peak validation accuracy for all models.

The coefficients λ_{Dice} and λ_{CE}, both set to 1, were determined empirically. We resized images to (224×224) resolution. Evaluation outcomes are presented in terms of Dice score values for all retinal layers, fluid, and their average. U-Net and our Y-Net model comprised 7.76M and 7.46M parameters, respectively.

8.4.2 Results

Table 8.1 highlights our proposed method's Dice score for various retinal layers and corresponding fluid pockets, in comparison to the state-of-the-art methodologies. Our model demonstrates comparable or superior performance to existing methods for retinal layer segmentation, and significantly outperforms other models in fluid segmentation. We believe that this boost in performance is attributable to the presence of features within specific frequency ranges that are associated with fluid pockets.

To validate this hypothesis, we conducted an experiment adjusting the range of frequency values within the Fourier unit. Some qualitative results are also showcased to contrast the fluid segmentation performance of our model and U-Net in Figure 8.2. We note that our model's segmentation of fluid pockets closely resembles the annotations provided by the first expert, while U-Net exhibits shortcomings in segmenting fluid in certain areas.

Ablation Study

In Table 8.3, we conduct an ablation study of the components incorporated into our model. The initial row illustrates the performance of the Y-Net architecture utilizing standard convolutional blocks in the second branch. By evaluating our model with this configuration, we demonstrate that the observed enhancement in average Dice score, particularly in fluid segmentation performance, is not solely due to an increased network size, but rather significantly influenced by the incorporation of FFC blocks.

The remaining rows of the table present the performance of our model with varying α values. As detailed in the methodology section, α delineates the proportion of features within the global and local Fourier units. As presented in Table 8.3, optimal performance is attained with α values of 0.25 and 0.5. Both configurations yield a Dice score of 0.93 for fluid segmentation while maintaining comparable performance with other models in retinal layer segmentation.

We assert that both local and global features provide crucial information that our model can effectively learn. By selecting an α value that is neither too large (1) nor too small (0), our model is capable of correlating the global and local features to accomplish peak performance.

We continue our investigation into the influence of FFC blocks in Table 8.4 by adjusting the frequency ranges processed by the Fourier units. The initial row in Table 8.4 represents the

Tab. 8.3. **Ablation study on the FFC blocks and the value of α.**

FFC Block	α	ILM	NFL-IPL	INL	OPL	ONL-ISM	ISE	OS-RPE	Fluid	Mean
-	-	**0.87**	**0.90**	0.77	0.75	0.89	0.88	**0.86**	0.89	0.851
✓	0	0.86	0.89	0.76	0.75	**0.90**	**0.89**	0.85	0.86	0.845
✓	0.25	0.86	0.89	0.77	0.74	0.89	**0.89**	**0.86**	**0.93**	0.854
✓	0.5	0.86	0.89	**0.78**	0.75	**0.90**	0.88	0.85	**0.93**	**0.855**
✓	0.75	0.84	0.87	0.76	0.73	0.89	0.88	**0.86**	0.90	0.841
✓	1	0.85	0.89	0.77	**0.76**	0.89	0.88	0.85	0.88	0.846

Tab. 8.4. **Ablation Study on the effect of variation in frequency ranges.**

Spectral features range	ILM	NFL-IPL	INL	OPL	ONL-ISM	ISE	OS-RPE	Fluid	Mean
No change	0.85	**0.89**	0.77	**0.75**	0.90	**0.89**	0.85	0.93	0.854
keep(-10,10)	**0.86**	**0.89**	**0.78**	**0.75**	0.90	0.88	0.85	**0.93**	**0.855**
remove(-10,10)	0.84	0.88	0.76	0.74	0.90	0.88	0.85	0.78	0.829

standard model configuration, where the frequency range generally lies between -40 and 40. In the second row, we limit the frequency range to $(-10, 10)$, resulting in a marginal enhancement in the overall segmentation performance. In the final row, we exclude frequencies between $(-10, 10)$ by setting the frequency values from -10 to 0 to -10 and assigning the positive values between 0 and 10 to 10. Under these conditions, the fluid segmentation performance significantly decreases to 0.78, while the retinal layer segmentation performance experiences a minimal reduction. These experiments indicate that the superior fluid segmentation achieved by our model is significantly influenced by spectral domain features and that the features utilized for fluid segmentation belong to a particular frequency range (in this case, approximately $(-10, 10)$).

In our experimental process, we also explored the use of focal frequency loss [97] and attempted to integrate skip connections from the spectral encoder to the spatial decoder. However, these modifications did not yield an enhancement in model performance. We postulate that incorporating skip connections from the spectral domain to the spatial domain does not confer a substantial benefit, as spectral features and global information may not exhibit a strong correlation with the segmentation map.

8.5 Conclusion

In this study, we introduced an end-to-end autoencoder framework for the segmentation of retinal layers and fluid pockets in optical coherence tomography (OCT) images. Our proposed Y-Net incorporates a secondary encoder branch, designed by us, that extracts spectral domain features in conjunction with the spatial encoder utilized in prior works. Given that OCT images contain high-frequency, non-uniform speckles that vary according to tissue type and retinal layers, we posited that our model could enhance OCT segmentation performance by learning these spectral domain features. By recognizing features in the frequency domain, our network is equipped to model and understand the speckle distribution within each layer. Our experimental results demonstrated a significant impact on the model when altering the frequency range within the Fourier units, integral to our Fast Fourier Convolutional (FFC) blocks. This lends support to our hypothesis that certain frequencies in OCT images may

correspond to specific layers or fluid pockets. Our final proposed model was benchmarked against numerous prior works, illustrating that it surpasses existing models by 13% in fluid segmentation, achieving a dice score of 0.93, whilst maintaining a comparable or superior performance in retinal layer segmentation.

Part III

Learning to Learn Semantics

9 Semantic Scene Modelling

In computer vision, a scene is a complex and rich representation of an image that captures its objects, categories, locations, and relationships. Different methods have been proposed to represent scenes, each with varying levels of detail and complexity. In this part of the thesis, we explore the concept of semantic scene understanding using generative models and scene graphs.

A scene graph is a graph-based structure that encodes the elements and relationships of a scene as nodes and edges. Scene graphs can capture the complexity of a scene in a refined and comprehensive manner, making it possible to manipulate the scene semantically. However, constructing scene graphs is challenging and requires extracting features from both the objects and the relationships in the scene.

In this section, we review some of the previous methods for scene representation and feature extraction, and introduce our proposed methods for scene modeling, image generation, and image manipulation using scene graphs.

9.1 Scene Representation

The task of defining a scene has evolved over time in computer vision. Historically, the earliest works conducted image classification tasks by viewing an image as a unified whole and assigning it a categorical label [123]. This method represented a fundamental and essential approach to image analysis but was relatively simplistic and did not account for the intricate complexity of scenes.

To address this, object detection techniques were subsequently introduced that defined a scene in terms of individual object categories contained within bounding boxes [65]. However, this approach still lacked in capturing the detailed intricacies of a scene. To overcome this, semantic segmentation maps were proposed, which provided a more granular view of a scene by assigning labels to each pixel in an image [151]. The notion of semantic segmentation was then extended to instance segmentation, which not only labels each pixel of an image but also differentiates between individual instances of objects within the same category [70].

However, these methods still do not capture the relationships between objects in a scene, which are crucial for understanding the semantics of a scene. To address this, graphs have been utilized to depict a scene [105]. In a semantic scene graph, the nodes are assigned object categories, and the relationships between these objects are encoded as edges of the graph. The difference between a regular graph and a scene graph in literature is the existence of the edge features. A graph is normally a representation of an affinity matrix that defines whether

edges exist between pairs of nodes in the graph or not; while in a scene graph the edges can define semantic relationships between the objects (e.g. riding, sitting), proximity (near, on), etc.

9.2 Learning from Semantics

We aim to explore the potential of generative models for semantic scene understanding, focusing on the task of image generation and manipulation using scene graphs. Scene graphs are structured representations of scenes that capture the objects and their relationships. To generate images from scene graphs, we need to extract useful features from the scene graph and use them to condition a generative model.

We propose to use text embeddings as node and edge features for scene graphs, as they can capture more information and context about the objects and their predicates. These embeddings can be obtained from existing models such as WordNet [159], Glove [178] and GPT-3 [21], or learned from scratch during training. We also propose to use graph neural networks (GNNs) to process these features and update them based on the graph structure. GNNs are neural networks that can operate on graph data, and can learn to propagate information across nodes and edges through message passing mechanisms. GNNs can learn to extract high-level features from scene graphs that can be used for downstream tasks such as image generation or manipulation.

To perform the image generation task, we condition a generative model on the features extracted from the scene graphs. Given pairs of images and scene graphs and the bounding box locations of the objects in the scenes, the goal of the model is to infer images conditioned on the scene graph without using the ground truth bounding box coordinates or any shape information at inference time. Therefore, we need to extract object features from the scene graph and predict the location and shape of the objects in the scene. We use a GNN to extract object features from the scene graphs, and then use these features to predict the spatial layout of the scene. The layout is then used as a conditioning input for the generator.

We investigate the application of meta-learning for improving the quality of image generation [60]. Furthermore, we apply our scene graph to image model for the task of semantic image manipulation. We seek to improve the learned representation by disentangling the latent embeddings and employing neighborhood routing mechanism within the graphs [155]. Semantic image manipulation is a task that allows users to modify images by changing or adding elements in the scene graph. For example, users can change the color or size of an object, or add a new object with a specific relationship to an existing one.

10 Modelling Scenes through Unconditional Scene Graph Generation

10.1 Introduction

Scene graphs provide a detailed depiction of a given scene by denoting categorical object instances as nodes and their categorical relationships as edges. This comprehensive form of representation offers a nuanced understanding of the scene that transcends the simplicity of object-level reasoning. Various methodologies for generating scene graphs from images have been explored within the computer vision discipline [166, 259]. This particular form of representation has demonstrated its utility in various applications such as image retrieval [105] and visual question answering (VQA) [64]. Due to the high level of semantic control scene graphs offer over image components, they serve as an effective means of representation for tasks related to semantically driven image generation [104] and manipulation [42].

The domain of unconditional scene graph generation, in essence, creating a scene graph from a random input rather than a predetermined image, has seen comparatively little exploration. The modelling of such scene graphs holds the potential to facilitate the learning of patterns inherent in real-world scenes, like the co-occurrence of objects, their relative placements, and interactions.

In this chapter, we delve into the unconditional generation of scene graphs using generative models, aiming to produce novel and realistic scene graphs. Prior work has touched upon the generation of relational graphs [246] and probabilistic grammar [110], each designed specifically for a certain domain. However, to the best of our knowledge, our study is the first to investigate the application of a generative model for generating semantic, language-based scene graphs [105, 121]. Since scene graphs provide a detailed account of scenes, it is feasible to transform the generated graph to another domain using advanced models specialized in tasks like graph-to-image translation [104].

In the area of unconditional image generation, recent works have achieved remarkable results, primarily with images that feature a single main subject or exhibit uni-modal distributions, such as datasets of faces or cars. However, these models often struggle when it comes to intricate and diverse scenes containing multiple objects. Our experiments reveal that employing scene graphs for unconditional image scene generation results in more discernible object instances, due to the graphs' ability to grasp complex and frequently abstract semantic concepts like objects, their interactions, and attributes. Moreover, such a generative model can identify scene graphs that fall outside the distribution and complete partial scene graphs.

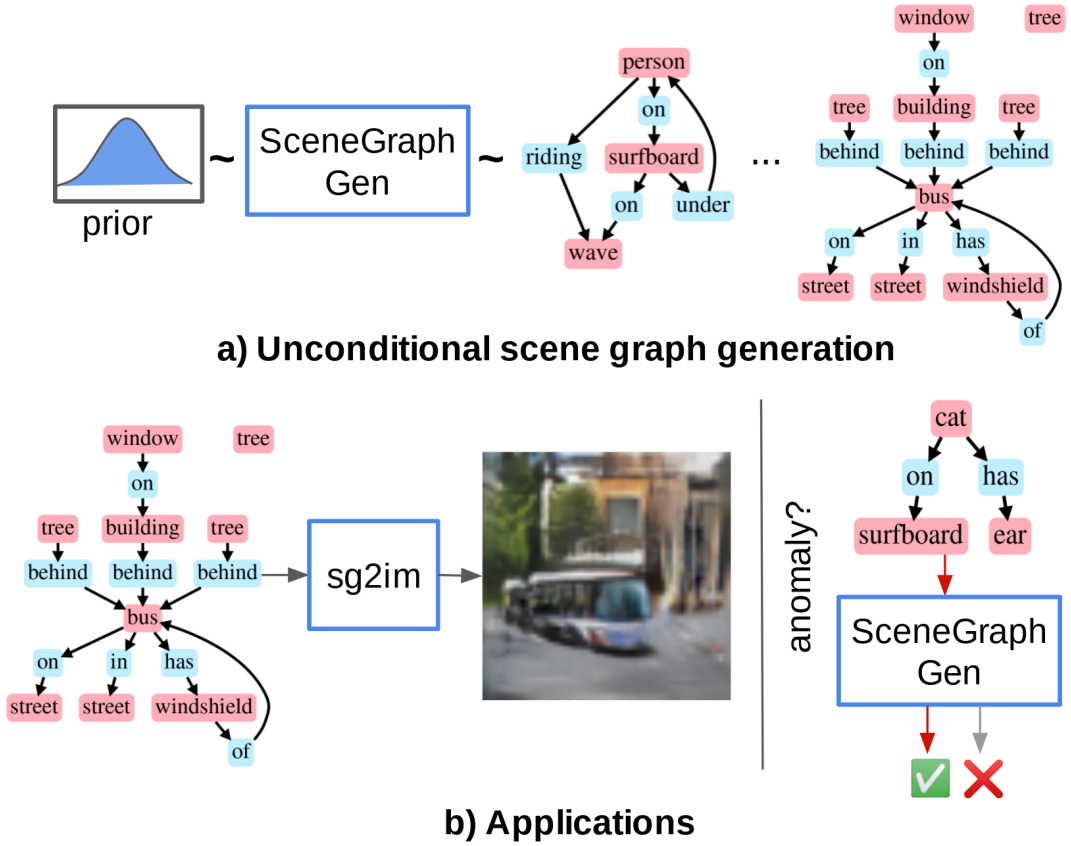


Fig. 10.1. **SceneGraphGen overview. a)** SceneGraphGen generates scene graphs unconditionally from a randomly sampled seed object. **b)** Applications in *Left:* translation of a generated scene graph to an image, using an off-the-shelf graph-to-image network *Right:* detection of out-of-distribution samples

A couple of deep generative models for graph data have been recently introduced [14, 48, 68, 215, 273], with the goal of producing realistic domain-specific graphs while capturing key graph patterns such as degree distribution and clustering. Nevertheless, each model has limitations which render them unsuitable for certain applications. Scene graphs often exhibit significant size variations, object and relationship categories are inherently unbalanced, and the edges are directed.

Addressing these challenges, we propose a specialized model dubbed SceneGraphGen[1] [63] (as illustrated in Figure 10.1). The overarching auto-regressive structure borrows inspiration from GraphRNN [273], as it caters to varying graph sizes, unlike models like [22, 48, 215]. More specifically, the model is designed to process categories for nodes and edges and to accommodate the directionality of the edges. In this auto-regressive setup, the scene graphs are modelled as a sequence of sequences. The sequence's history is preserved in a hidden state using a Gated Recurrent Unit (GRU) [31], which is then utilized to generate a categorical distribution over the nodes and edges at each stage, from which the node and edge categories can be sampled. The nodes are generated using a multi-layer perceptron (MLP), while a GRU is employed for sequential edge generation.

[1] https://SceneGraphGen.github.io/

As the task of unconditional scene graph generation is still underexplored, appropriate metrics for evaluating the quality of the generated graphs are yet to be proposed. Thus, following [273], we adopt a Maximum Mean Discrepancy (MMD) metric, adapted with a random-walk graph kernel and a node kernel suitable for the scene graph structure. We validate the effectiveness of these kernels using sets of corrupted datasets.

Our contributions can be encapsulated as follows:

- We propose SceneGraphGen, a novel approach to address the relatively unexplored task of unconditional semantic scene graph generation. We utilize a graph auto-regressive model to effectively process the structural intricacies of scene graphs.
- We showcase the versatility of our learned scene graph model through three key applications - image generation, anomaly detection, and scene graph completion.
- We introduce an innovative MMD metric to assess the quality of the generated scene graphs, which takes into consideration both the node and graph level.

We conduct a thorough evaluation of our model on the Visual Genome dataset [121] and demonstrate its ability to generate semantically plausible scene graphs. Furthermore, we exhibit how these scene graphs can be translated into novel images, leveraging cutting-edge scene graph to image models [104]. Moreover, we demonstrate the model's proficiency in detecting anomalous scene graphs and augmenting incomplete scene graphs.

10.2 Related Work

Scene graphs and their applications Scene graphs, as introduced in [105], provide a semantic interpretation of an image, articulating the relationships between objects through semantic labels. The availability of large-scale datasets such as Visual Genome [121], annotated with scene graphs, has facilitated deep learning tasks. Various studies have investigated scene graph prediction conditioned on an image [139, 166, 259, 263, 277] or point clouds [244, 254]. Most of these efforts involve initially identifying the objects in a scene, followed by reasoning about their interconnections. Our work, in contrast, delves into the realm of unconditional graph generation, where the task is to model the scene graph distribution independent of any input during the testing phase.

Scene graphs have demonstrated their utility in a myriad of tasks. Johnson et al. [104] pioneered the use of scene graphs for image generation, which has subsequently been explored in interactive setups for generation [4] and semantic manipulation [42]. Wang et al. [246] utilized relational graphs for indoor scene planning. Other researchers have harnessed scene graphs for image retrieval that transcends specific domains [105, 244]. Scene graphs have also been integrated with language, exemplified in applications such as Visual Question Answering (VQA) [64, 265], or for predicting the type of an object given its location in the scene [289].

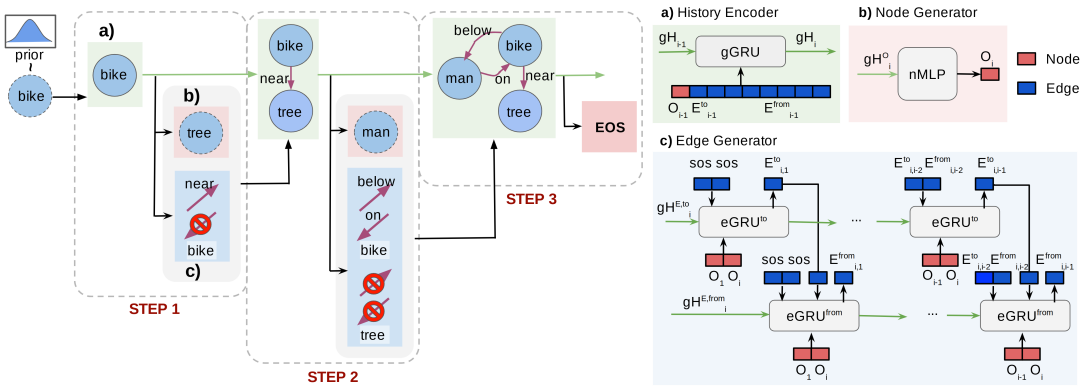


Fig. 10.2. **SceneGraphGen Overview.** *Left:* An overview of the auto-regressive generation process, sequentially considering the current graph sequence at each step to create a new node and a set of associated edges. *Right:* Detailed breakdown of the various modules: **a)** history encoder, **b)** node generator, **c)** edge generator.

Generative models on graphs Historical methods [1, 45, 130, 252] for graph generation, which were engineered to capture specific patterns, often fall short when tasked with generalizing to diverse graph patterns due to their domain-specific nature.

Deep auto-regressive models for graph generation, such as those presented in [88, 142, 144, 273], typically allow flexibility in the number of nodes but require a pre-determined node ordering. In particular, GraphRNN [273] views the graphs as a sequence of sequences and leverages a hierarchical GRU architecture to model the dependencies between nodes and edges. This model, with its $O(N^2)$ complexity, allows for the generation of graphs of varying sizes. The authors also utilize a breadth-first search (BFS) ordering to drastically limit the possible orderings. However, their work is focused solely on the generation of unlabeled graphs.

Other studies have turned to Variational Autoencoders (VAE) which embed the graphs into a vector [215], or a junction tree of node clusters [99], or blend VAE with an auto-regressive approach [148]. Although these models generally enable graph modeling with attributes or categories, they tend to capture imprecise likelihoods and struggle to scale to larger graphs.

Generative Adversarial Network (GAN)-based works, on the other hand, are usually constrained to either a single sample [14], or a small and fixed size graph [22, 48], rendering them unsuitable for scene graphs that vary in size and diversity. They also face challenges in maintaining training stability.

In our study, we investigate an auto-regressive approach that allows for a flexible number of graph nodes. Our method, unlike GraphRNN, supports semantic labels for nodes and edges, models directed edges, and deploys an edge generation GRU that takes into account the node categories.

Although recent studies have delved into generative tasks with comparable scene structures, such as relational graphs [246] and probabilistic grammar [38, 110], our focus lies on semantic graphs that are associated with images taken from real-world scenarios. These

semantic graphs are substantially more diverse in terms of their node and edge labels, and exhibit less regular graph size and connectivity patterns, compared to the synthetic datasets used in other works.

10.3 SceneGraphGen

Given a set of n scene graphs, denoted by $\mathbb{G}s = G_s^1, G_s^2, .., G_s^n$, which we assume that exemplify the data distribution of scene graphs $pdata(G_s)$, our aim is to construct a generative model based on dataset $\mathbb{G}_s$, capable of producing new scene graph samples. A scene graph sample G_s, associated with an image I, is defined by $G_s = (\mathcal{O}, \mathcal{E})$, where $\mathcal{O}$ represents the set of objects (nodes), and $\mathcal{E}$ corresponds to relationships (edges). Each object o_i in the set $\mathcal{O} = o_1, o_2, .., o_m$ holds an object category $o_i \in \mathcal{C}$, with $\mathcal{C} = 1, 2, .., C$. The edges are defined as ordered triplets $\mathcal{E} \subseteq (o_i, r_k, o_j) | o_i, o_j \in \mathcal{O}, o_i \neq o_j$, denoting a directed edge from o_i to o_j, where $r_k \in \mathcal{R}$ represents the category of relationship between the objects, and $\mathcal{R} = 1, 2, .., R$ (*e.g.* `man - wearing - shirt`). We approach this task with an auto-regressive model, providing flexibility in terms of node count and graph connectivity. We provide an overview of the network in section 10.3.1, and delve into the details of each component in the subsequent sections.

10.3.1 Definitions

The SceneGraphGen model aims to learn a distribution $p_\phi(G_s)$ over scene graphs that closely resembles the original data distribution $p_{data}(G_s)$. By adopting an auto-regressive approach, we translate scene graphs into a sequence representation. Given a permutation π, the node set $\mathcal{O}$ is transformed into a sequence $O = (\pi(o_1), \pi(o_2), .., \pi(o_m))$. The edges $\mathcal{E}$ are depicted via two upper triangular sparse matrices E^{from} and E^{to}, with the relationship label r_k located at index (i, j) if either $(\pi(o_i), r_k, \pi(o_j)) \in \mathcal{E}$ or $(\pi(o_j), r_k, \pi(o_i)) \in \mathcal{E}$, respectively. These matrices correspond to the two potential edges (to and from) between any pair of nodes. Consequently, a scene graph G_s can be expressed as a sequence $X = (O, E^{to}, E^{from})$. Each element of sequence X is itself a sequence, given by $X_i = (O_i, E_i^{to}, E_i^{from})$, where O_i is the object node, and $E^{to}i$ and $E^{from}i$ represent sequences of edges connecting O_i and preceding nodes. The goal of learning $p_\phi(G_s)$ is thus translated to learning a sequence distribution $p_\phi(X)$, which can be modeled using an auto-regressive approach. The probability over sequence X is broken down into consecutive conditional probabilities.

$$p_\phi(X) = p(X_1) \prod_{i=2}^{n} p_\phi(X_i | X_{<i}). \tag{10.1}$$

The notation $X_{<i} = (X_1, .., X_{i-1})$ illustrates the partial scene graph sequence up to step i. Each conditional probability $p_\phi(X_i|X_{<i})$ is further dissected into three components, each pertaining to a constituent of X_i, as follows:

$$p_\phi(X) = p(O_1) \prod_{i=2}^{n} p_{\phi_1}(O_i|X_{<i}) p_{\phi_2}(E_i^{to}|O_i, X_{<i}) \\ p_{\phi_3}(E_i^{from}|O_i, E_i^{to}, X_{<i}). \tag{10.2}$$

We propose to model Equation 10.2 to learn the probability distribution over the scene graph sequences, $p_\phi(X)$. Unlike certain models, SceneGraphGen does not make any conditional independence assumptions in its formulation, thus enabling the potential to encapsulate all intricate object and relationship dependencies that exist within the data. We designate $p(O_1)$ as a presumed prior distribution over the initial node, such as the categorical distribution concerning object occurrences. The auto-regressive modeling is undertaken in three distinct phases:

1. History Encoding: This phase focuses on encapsulating the information from prior steps, $X_{<i}$, into a hidden representation h_i, modeling $h_i = \text{Encoder}(X_{<i})$.

2. Node Generation: Utilizing the hidden state, this phase generates the subsequent object node, modeling $p_{\phi_1}(O_i|h_i)$.

3. Edge Generation: Employing both the hidden state and the explicit node data, this phase sequentially generates edges, modeling $p_{\phi_2}(E^{to}i|O_i, h_i)$ and $p\phi_3(E_i^{from}|O_i, E_i^{to}, h_i)$.

Figure 10.2 provides a schematic representation of both the overall architecture and each component of SceneGraphGen. We will now delve into a detailed description of each stage.

10.3.2 History Encoding

To capture the prior information, $X_{<i}$, at each time step i, we utilize a Gated Recurrent Unit (GRU) module. Specifically, we deploy three distinct GRUs, each tailored to process information for the three outputs: O_i, E_i^{from}, and E_i^{to}. This approach facilitates the decoupling of information among the three outputs. It's crucial to note that while each of the three GRUs maintains unique parameters, these parameters are shared throughout all time steps. We denote these networks as gGRU$_O$, gGRU$_{E^{from}}$, and gGRU$_{E^{to}}$.

Initially, the hidden states of the GRUs are set as zero (empty) vectors. In subsequent steps, the input to all three GRUs is the prior sequence X_{i-1}, which is interpreted as the concatenated trio of outputs from the previous step $i-1$—namely, O_{i-1}, E_{i-1}^{from}, and E_{i-1}^{to}. In the first step, the edge inputs are zero (empty), and the node is sampled from a predetermined distribution. This prior distribution is calculated based on the occurrence of the first node category according to the selected ordering strategy.

10.3.3 Node Generation

We employ a multi-layer perceptron (MLP) network—referred to as nMLP—that accepts the hidden state from gGRU_O as an input. This network outputs a categorical distribution over the object categories $\mathcal{C}$, represented as prediction scores θ_i^O. The nMLP's parameters are shared across all time steps. The node O_i is then sampled from this distribution. The generation of new nodes (and subsequently, the scene graph) ceases when an EOS (end of sequence) token appears.

10.3.4 Node-Aware Edge Generation

Each step i produces a *sequence* of edges E_i^{from}, which essentially forms a sequence of sequences, necessitating a sequential model for each step of the sequence. For this hierarchical structure, we utilize a GRU that shares parameters, not only across all time steps i, but also for each step within that sequence i, denoted as j. A similar GRU is used to generate E_i^{to} at each step i. We label these networks as $\text{eGRU}_{E^{from}}$ and $\text{eGRU}_{E^{to}}$.

The initial hidden state of the GRUs is derived from the hidden state outputted by $\text{gGRU}_{E^{from}}$ and $\text{gGRU}_{E^{to}}$ networks, respectively. The initial input to these GRUs is an SOS (start of sequence) token, which triggers the generation process.

For any step j, the input for $\text{eGRU}_{E^{to}}$ is the combination of four elements: the edges from the previous step $E_{i,j-1}^{from}$ and $E_{i,j-1}^{to}$, the node O_i at the current step i, and the node O_j from the j^{th} time step.

This additional node information provides extra conditioning for the edge generation process, inspired by the significant predictability of relationship co-occurrence for a given object pair—that is, additional knowledge of node pairs may enhance relationship predictions.

The GRUs generate categorical distribution parameters over the relation categories $\mathcal{R}$ (and a no-edge category), denoted by $\theta_{i,j}^{E^{to}}$, from this input. The next edge $E_{i,j}^{to}$ is then sampled from the generated categorical distribution.

The inputs of $\text{eGRU}_{E^{from}}$ at step j are identical to those of $\text{eGRU}_{E^{to}}$, in addition to the edge $E_{i,j}^{to}$ generated by $\text{eGRU}_{E^{to}}$ at step j. The next edge $E_{i,j}^{from}$ is produced by $\text{eGRU}_{E^{from}}$ in a similar manner.

This procedure continues for $i-1$ steps, corresponding to the number of previous nodes that can connect to the current node O_i.

10.3.5 Objective Functions

The parameters ϕ of the model $p_\phi(X)$ are learned by maximizing the likelihood of the training data $\mathbb{G}$ with respect to the model. Training is conducted using the teacher forcing technique, where, at each step i, we utilize the actual sequences $O_i, E_i^{to}, E_i^{from}$ instead of sampling

from the model's predictive scores $\theta_i^O, \theta_i^{E_{from}}, \theta_i^{E_{to}}$. The objective function is calculated using the cross-entropy (CE) between the predicted scores and the actual sequence at each step, followed by their summation. For a given scene graph sequence X, the objective function is expressed as follows:

$$\mathcal{L}(X;\phi) = \mathcal{L}_O(O;\phi_1) + \mathcal{L}_E(E_i^{to}, E_i^{from};\phi_2,\phi_3) \tag{10.3}$$

$$\mathcal{L}_O = \sum_{i=2}^{n} CE(\theta_i^O, O_i) \tag{10.4}$$

$$\mathcal{L}_E = \sum_{i=2}^{n}\sum_{j=1}^{i-1} CE(\theta_{i,j}^{E_{to}}, E_{i,j}^{to}) + \sum_{i=2}^{n}\sum_{j=1}^{i-1} CE(\theta_{i,j}^{E_{from}}, E_{i,j}^{from}) \tag{10.5}$$

10.3.6 Inference

After acquiring the distribution over scene graph sequences $p_\phi(X)$, SceneGraphGen can sample new instances of scene graphs. This sampling process essentially adheres to the auto-regressive generation method explained thus far. Initially, we sample the first object node from the prior distribution, thereby creating the initial sequence X_1. This prior distribution is calculated empirically from the training set, based on the frequency of the first node when nodes are ordered according to the strategy chosen during training. In subsequent steps, we utilize the preceding sequence X_{i-1} as input to compute the hidden states with the three gGRUs. These hidden states are then used to generate the next node O_i using nMLP by sampling from the categorical output. Similarly, we generate sequences of edges $E^{to}i$ and E_i^{from} using $\text{eGRU}_{E^{to}}$ and $\text{eGRU}_{E^{from}}$, respectively, from the corresponding categorical outputs. The node and sequence of edges are merged to form the next sequence X_i. This process continues until the node generator produces an EOS token.

Additionally, SceneGraphGen can evaluate the likelihood of a given sample. For a provided scene graph converted into a sequence X, each element of the sequence X_{i-1} is fed into the network to generate the node categorical outputs θ_i^O as well as the edge categorical output sequences $\theta_i^{E_{from}}$ and $\theta_i^{E_{to}}$. We then select the probability of the actual node/edge category for each output, and calculate the negative of the sum of the logarithm over the entire sequence X to determine the negative log-likelihood (NLL). This NLL value illustrates the 'unlikeliness' of the occurrence of a given scene graph sample. We apply the NLL to detect anomalies in section 10.4.2.

10.3.7 MMD Kernels

We propose two MMD kernels, namely the *random walk graph kernel* and the *object set kernel*. We explain them as follows.

Random Walk Graph Kernel

We adopt a generalized formulation for the random walk kernel from [61], allowing flexibility in the selection of appropriate node and edge kernels. When comparing two graphs G_a and G_b, we may want to compare two respective nodes r and s. This comparison can be achieved by juxtaposing all walks of length p in G_a that start from r with all walks of length p in G_b that start from s. The similarity between each pair of walks can be computed by comparing the corresponding nodes and edges encountered in the walks using appropriate kernels. The kernel to compare any two nodes can be written as:

$$k_R^p(G_a, G_b, r, s) = \sum_{\substack{(r_1, e_1, .., e_{p-1}, r_p) \in W_{G_a}^p(r) \\ (s_1, f_1, .., s_{p-1}, f_p) \in W_{G_b}^p(s)}} \Big[k_{node}(r_p, s_p) \prod_{i=1}^{p-1} k_{node}(r_i, s_i) k_{edge}(e_i, f_i) \Big] \tag{10.6}$$

To contrast the entire structure, the kernel in Equation 10.6 is aggregated over all pairs of nodes, and then normalized based on the maximum value of the kernel evaluations for each graph with itself.

$$k_G^p(G_a, G_b) = \sum_{\substack{r \in V_{G_a} \\ s \in V_{G_b}}} k_R^p(G_a, G_b, r, s) \tag{10.7}$$

$$k_G^N(G_a, G_b) = \frac{k_G(G_a, G_b)}{\max(k_G(G_a, G_a), k_G(G_b, G_b))} \tag{10.8}$$

To compare the nodes, we utilize the Kronecker delta function which outputs 1 when the node categories align and 0 otherwise, *i.e.* $k_{node}(r, s) = \delta(r, s)$. Nonetheless, as there can be multiple nodes of the same category, the significance of the nodes in a graph will be lower for a category with a single occurrence and higher for categories with multiple occurrences. Indeed, the importance of a category with multiple occurrences should decrease as the occurrences increase. To address this, the node kernel is normalized with the frequency of occurrence within a graph. The node kernel is derived as:

$$\begin{aligned} k_{node}^N(r, s) &= \sigma(r)\sigma(s)k_{node}(r, s), \\ \text{where} \quad \sigma(s) &= \frac{1}{\sum_{s \in V_{G_s}} k_{node}(r, s)} \end{aligned} \tag{10.9}$$

To compare the edges, we again use the Kronecker delta function, *i.e.* $k_{edge}(p, q) = \delta(p, q)$.

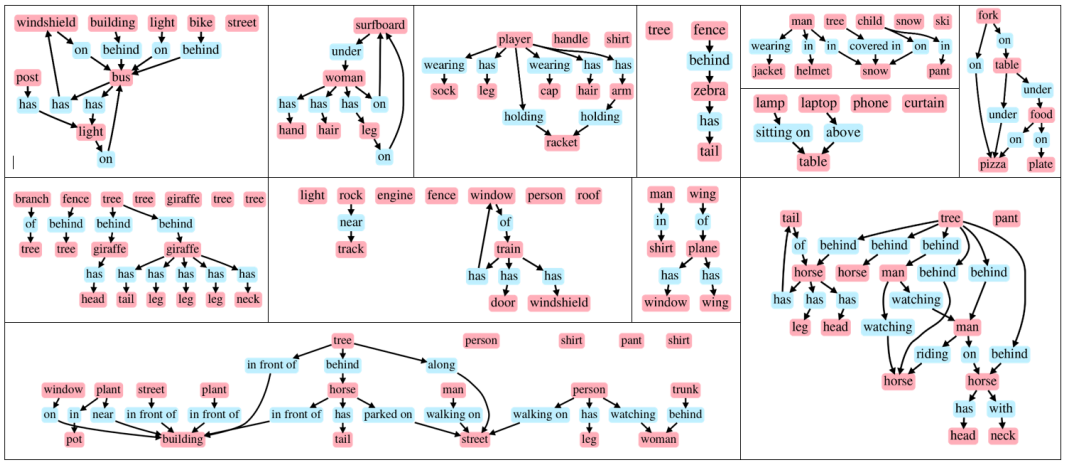


Fig. 10.3. Scene Graph generation visualization by SceneGraphGen on Visual Genome. The size and content of the generated graphs are diverse and the scene contents are reasonable.

Object Set Kernel

We aim to compare two sets of object instances. It is demonstrated in [75] that for a domain set $\mathcal{X}$ and two sets $A, B \in \mathcal{X}$, given positive definite kernels k_{label} and k_{count}, a generalized set kernel between A and B can be defined as:

$$k_{set}(A, B) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{X}} k_{label}(x, y) k_{count}(A(x), B(y)) \tag{10.10}$$

We adopt $k_{label}(x, y) = \delta(x, y)$ as the Kronecker delta function, which yields one when both x and y share the same object categories. Here, $A(x)$ represents the frequency of element x in A, while $B(y)$ denotes the frequency of element y in B. We define k_{count}, which is the generalized t-student kernel [16, 182] as:

$$k_{count}(A(x), B(y)) = \frac{1}{1 + |A(x) - B(y)|} \tag{10.11}$$

This enables us to identify when the two sets share the same object category member and gauge how similar the frequency of those object categories are. In line with the graph kernel above, the object-set kernel is normalized based on the maximum value of the kernel evaluation of each object set and itself.

$$k^N_{set}(A, B) = \frac{k_{set}(A, B)}{\max(k_{set}(A, A), k_{set}(B, B))} \tag{10.12}$$

10.4 Experiments and Results

In this section, we outline the experiments conducted to evaluate the efficacy of our model. We begin by detailing our implementation details, evaluation settings, encompassing evaluation

Model	Ordering	Graph		Image			
		MMD node ($\times 10^3$) ↓	MMD_G ($\times 10^3$) ↓	FID ↓	IS ↑	Precision ↑	Recall ↑
GraphRNN [273]	BFS	2.3	1.3	75.8	4.88	0.680	0.660
	Random	0.39	1.2	74.5	4.85	0.679	0.664
SceneGraphGen	BFS	2.05	1.82	73.3	5.04	0.679	0.690
	Hierarchical	1.85	0.63	72.2	**5.26**	0.717	**0.714**
	Random	**0.37**	**0.11**	**71.2**	4.95	**0.727**	**0.714**
Ground Truth		0.018	0.023	73.0	5.22	0.693	0.707

Tab. 10.1. Quantitative results of the graph samples (left) and image samples (right)

metrics, baselines, and datasets. Subsequently, we deliver both qualitative and quantitative results pertaining to graph generation. Lastly, we demonstrate the practical utility of the trained model across three applications: image generation, anomaly detection, and graph completion.

10.4.1 Experimental Setup

Implementation details We trained SceneGraphGen for 300 epochs using a batch size of 256, from which we sampled 256 batches with replacement per epoch. The initial learning rate was set to 0.001 and was decreased at a rate of 0.95 every 1710 steps. Node and edge inputs were embedded to a size of 64 and 8, respectively. All gGRUs and eGRUs were composed of 4 GRU layers, each having a hidden size of 128. The nMLP was constructed with 2 layers and utilized ReLU activation.

Evaluation Settings Our model is evaluated on the Visual Genome (VG) dataset [121]. In particular, we employ the scene graphs from the VG split popularized by [259]. This split contains 150 object categories and 50 relationship categories, and is divided into training and testing sets, consisting of 58k and 26k samples respectively. However, the VG dataset presents scene graphs with incomplete relationships, for which we supplement the edges using the Unbiased causal TDE model [229]. This model was selected from the SGG benchmark [228] because it demonstrates less bias in PredCls for infrequently occurring relationship categories. Given that various instances of objects (represented as bounding boxes) in scene graphs can be attributed to the same underlying object, we utilize a conservative bounding box intersection-over-union (IoU) of 0.5 along with a membership test in manually curated group categories to identify duplicate objects, randomly selecting one from each group. We somewhat alleviate this issue by employing heuristics to eliminate such relationships.

As there are no existing solutions specifically tailored for unconditional scene graph generation, we draw a comparison between our model and GraphRNN, modified to incorporate node and edge categories along with edge directions. However, SceneGraphGen stands out as it integrates node information during edge generation and conditions E^{from} edges on E^{to} edges. We further subject our model to various node ordering schemes, such as BFS order, hierarchical order, and random order. The BFS node ordering, introduced in GraphRNN [273], visits the graph in a Breadth-First Search sequence. Hierarchical order, on the other hand, classifies nodes based on their relational attributes, for instance, background (`field`), objects/beings (`person`), parts (`foot`).

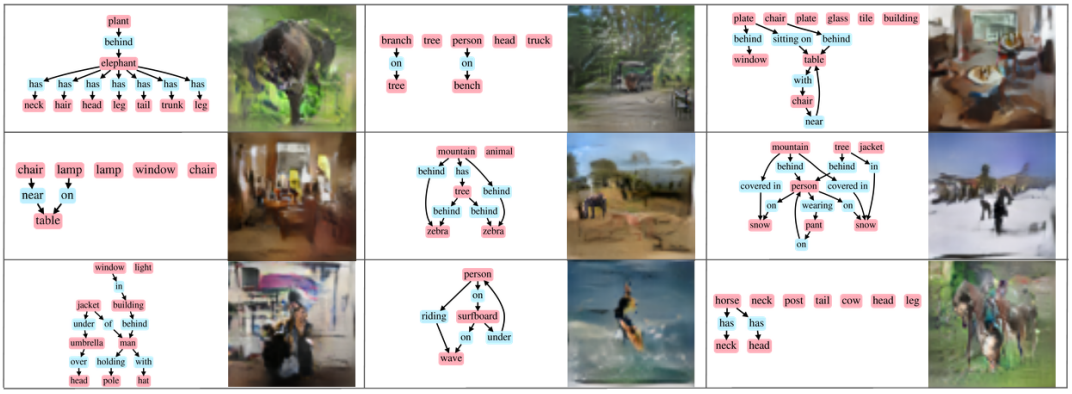


Fig. 10.4. Image synthesis examples with 64x64 resolution using SG2Im conditioned on the scene graphs generated by SceneGraphGen

a) Unconditional StyleGAN2 **b)** SceneGraphGen graphs + sg2im

Fig. 10.5. Image synthesis examples with 64x64 resolution using a.) Unconditional StyleGAN trained on Visual Genome (left) b.) SG2Im on scene graphs generated by SceneGraphGen trained on Visual Genome (right).

Assessing graph generative models based on the quality of samples is a complex endeavor [231], since it necessitates the comparison between the generated set with the test set. To this end, we employ Maximum Mean Discrepancy (MMD) to compare the generated and test datasets. Defined between two distributions p and q for a given kernel k, MMD is expressed as follows:

$$\begin{aligned} \text{MMD}^2(p,q) =& \mathbb{E}_{x,y\sim p}[k(x,y)] - 2\mathbb{E}_{x\sim p, y\sim q}[k(x,y)] \\ &+ \mathbb{E}_{x,y\sim q}[k(x,y)]. \end{aligned} \tag{10.13}$$

Considering MMD has not been previously implemented with kernels for directed and labeled graphs, we conceive two kernels for comparing any two samples of scene graphs:

Random Walk Graph Kernel, drawing inspiration from [61], we measure the similarity between two scene graph samples based on the comparison between the directed random walks generated from the graphs.

Object Set Kernel measure the similarity between the object sets in the scene graphs based on the comparison between the object categories and the quantity of object instances.

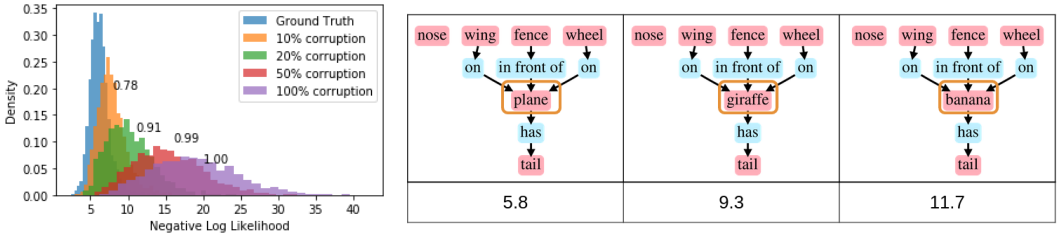


Fig. 10.6. **Anomaly Detection Task** *Left:* The distribution of negative log-likelihood (NLL) for datasets with different levels of corruption is presented. The value adjoining each distribution signifies the area under the ROC curve. *Right:* An instance showcasing an increasing trend in NLL value.

We label these metrics as MMD_G and MMD_N. The MMD_G metric contrasts the distributions based on the holistic graph similarity between samples (such as triplets, larger clusters, etc), while MMD_N contrasts the distributions contingent on the similarity between the corresponding set of objects (considering co-occurrence, number of instances, etc). We corroborate these metrics by examining their results on randomly corrupted datasets as well as the test dataset (refer to Table 10.2). A dataset corruption of x% is executed by randomly selecting x% of the nodes and edges in the graph and substituting them with a random label. As anticipated, the MMD values between two distinct sets of test datasets as well as between two fully corrupted datasets are remarkably low, as the datasets being compared are very alike. As the corruption intensifies, the MMD metrics between the test set and the corrupted set escalate, given that the distributions grow increasingly dissimilar.

Comparison	$MMD_N \downarrow$	$MMD_G \downarrow$
test Vs test	0.018	0.023
100% corrupt Vs 100% corrupt	0.11	0.0098
test Vs 20% corrupt	6.0	3.7
test Vs 50% corrupt	10	6.3
test Vs 100% corrupt	44	25

Tab. 10.2. MMD metrics ($\times 10^3$) sanity check, comparison between a set of ground truth graphs (test) and a randomly corrupted set.

We further attest the performance of SceneGraphGen by evaluating the quality of images produced by the scene graphs that it generates. Specifically, we use SceneGraphGen to generate scene graphs that are then input into the SG2Im model [104] to produce 64×64 images. To measure the quality of the generated samples, we employ established metrics commonly used in image generation tasks, such as Frechet Inception Distance (FID) [79], Inception Score (IS) [201], and the Precision ($F_{1/8}$) and Recall (F_8) scores [200]. The produced images are then contrasted with the original images from the Visual Genome dataset to quantify their quality.

10.4.2 Results

We evaluate our proposed scene graph generation model in various downstream tasks. In this section, we present their results.

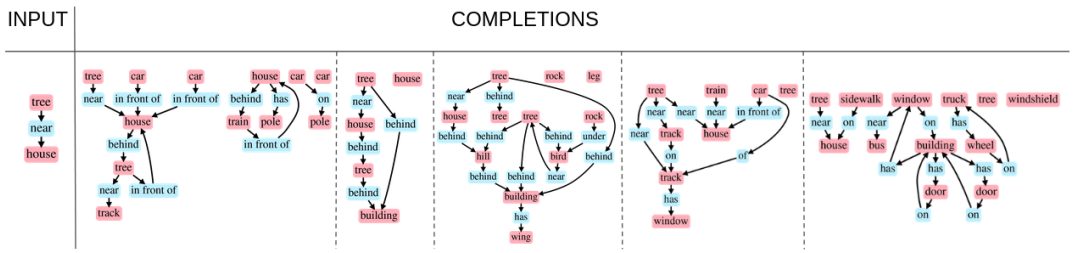


Fig. 10.7. Diversity in the scene graph graph completion task from a partial input scene graph.

Graph Generation Our model, SceneGraphGen, is capable of generating plausible and diverse scene graph samples. Visual illustrations of generated graphs can be found in Figure 10.3. The model successfully generates graphs that represent both indoor environments (`bed`, `chair`) and outdoor scenes (`tree`, `giraffe`, `road`), and forms logical co-occurrences of objects and meaningful relationships between them. When compared to GraphRNN, SceneGraphGen proves superior in both MMD_G and MMD_N metrics (Table 10.1, left), suggesting that the inclusion of node information significantly enhances edge generation. We also note that a random ordering scheme outperforms other methods for scene graph data. BFS and hierarchical ordering are predisposed to certain node orderings due to their reliance on node connectivity and semantics respectively. This bias is not beneficial for a dataset like Visual Genome which exhibits significant structural variation and little regularity. We surmise that random ordering performs better due to the absence of such bias.

Image Generation from Created Graphs We aim to demonstrate that SceneGraphGen-generated graphs can facilitate the synthesis of unique images. We utilize SG2Im [104], a method for image generation from scene graphs, to convert the generated graphs into images, a method we name SG2Im-SGG. Figure 10.4 provides examples of the novel scenes generated from their respective novel scene graphs. Quantitative assessment of the generated images (Table 10.1, right) illustrates that SceneGraphGen surpasses GraphRNN across all metrics and closely aligns with the ground truth, *i.e.* images generated using the test set of the ground truth data.

We further contrast these images with those produced by a leading model in unconditional image generation, namely StyleGAN2 [113]. To maintain fairness, we train the StyleGAN2 model on the same Visual Genome training split used to train SG2Im. The results for both methods are displayed in Figure 10.5. We find that while StyleGAN2 generates high-quality images for simpler configurations (e.g., landscapes), it struggles to maintain semantic properties in more intricate scenes (e.g., multiple instances of the same object or indistinct objects). Conversely, the images generated from our generated graphs exhibit more consistent compositions, as directed by the scene graphs. To further this comparison, we detect objects in StyleGAN and SG2Im-SGG images using a Faster-RCNN model trained on COCO. This detector identified 50 object categories in SG2Im-SGG images, as opposed to 40 categories in StyleGAN images, illustrating that using scene graphs as an intermediate representation contributes to the generation of semantically diverse scenes. Additionally, when comparing object occurrences to the ground truth test set from VG, SG2Im-SGG aligns more closely with the ground truth than the StyleGAN2 model does, with an average error of 1.2 compared to 1.4.

Anomaly Detection Our model can also be leveraged to detect anomalies in scenes. We calculate the likelihood over the test dataset using SceneGraphGen, identifying anomalies as scene graphs whose likelihoods deviate from the overall likelihood distribution. The utility of likelihood in anomaly detection is illustrated by computing the negative log-likelihood (NLL) over datasets subject to varying degrees of node and edge corruption. We measure the effectiveness of NLL in anomaly detection by using NLL scores to classify corrupted samples as anomalies, and then calculating the area under the ROC curve (AUROC) [76]. The distribution of NLL along with AUROC values can be seen in Figure 10.6 (left). The equivalent GraphRNN plot is available in the supplementary material. An example of an anomaly, where unexpected samples lead to a higher NLL value, is shown in Figure 10.6 (right).

A comparison of SceneGraphGen with the GraphRNN baseline on the NLL plot, is depicted in Figure 10.8. We believe that our model displays greater sensitivity to the level of dataset corruption compared to the baseline, i.e., the NLL gap between different corruption levels is more substantial than for the baseline.

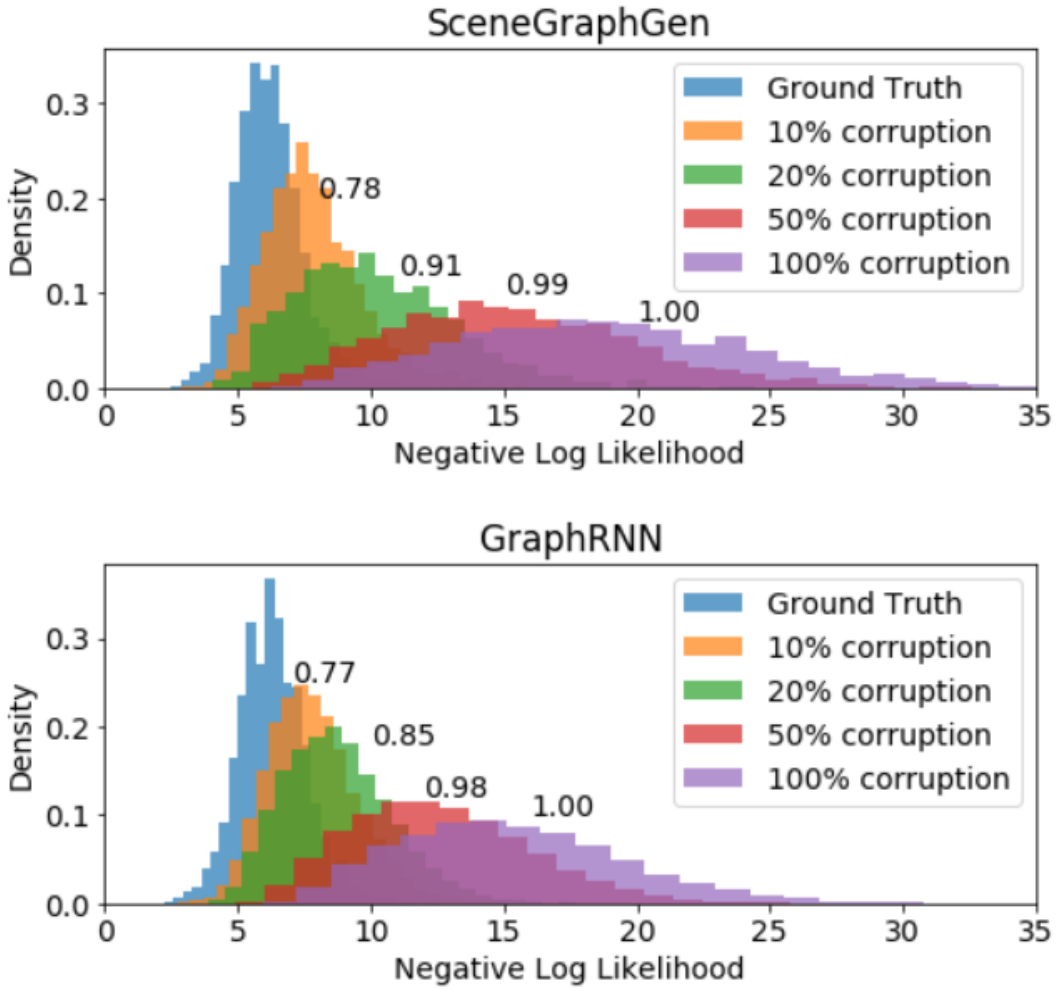


Fig. 10.8. NLL Distribution with various corruption levels

Overfitting Analysis Examples of generated graphs and their respective nearest counterparts in the training data, determined through graph kernel comparison, are presented in Fig-

ure 10.9. The graphs are distinct from each other, indicating that the model is not simply duplicating examples from the training set.

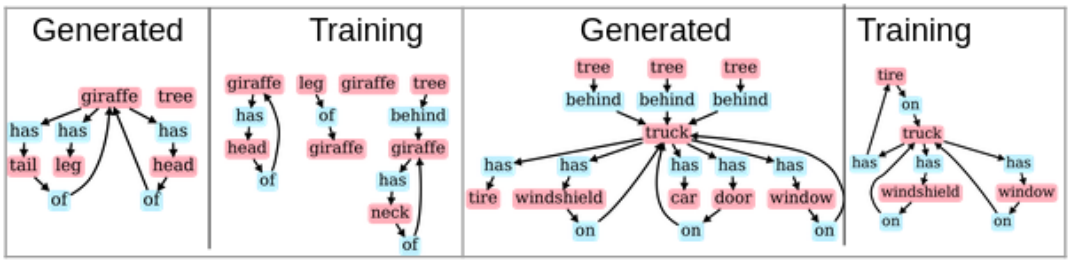
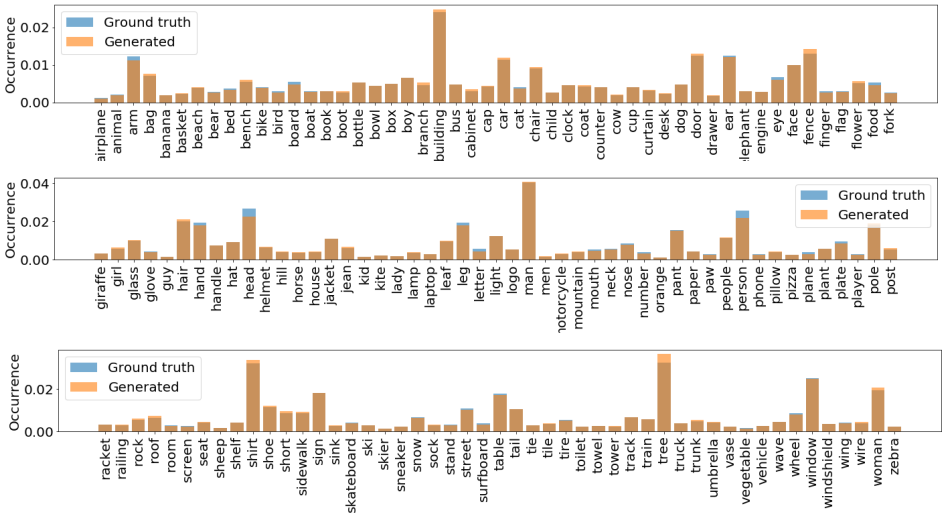


Fig. 10.9. Graph Generation Examples based with the closest sample from the training set.

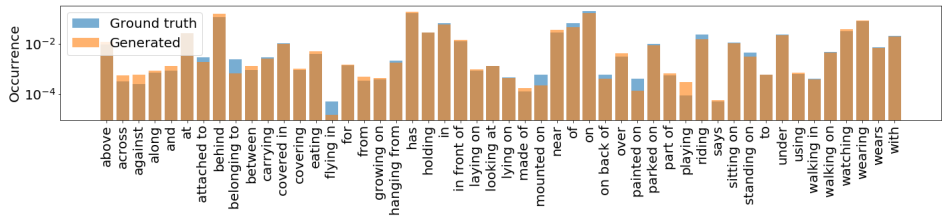
Scene Graphs Completion SceneGraphGen also has the capability to generate a scene graph based on partial graph input, allowing for a more controlled generation of scenes. Starting from an incomplete graph, we stochastically sample a sequence of nodes/edges (X_i) that is conditioned on the preceding sequence (X_{i-1}), just like in graph generation. This leads to an accumulation of randomness at each step, resulting in diverse graph completions. Figure 10.7 illustrates how the model is able to generate a range of possible scene graph completions starting from the same triplet `tree - near - house`.

Dataset Statistics Beyond the MMD metrics, which provide a comparison at the sample level, we provide statistics to compare generated and test samples at the dataset level. We contrast 20k samples of scene graphs generated by SceneGraphGen with the ground truth, *i.e*. the test dataset from Visual Genome. Figure 10.10 presents various patterns such as object occurrence (a), relationship occurrence (b), and object co-occurrence (c). Object occurrence calculates the probability of each object label appearing throughout the entire dataset. Similarly, relationship occurrence estimates the probability of each relationship label occurring across the full dataset. Object co-occurrence is determined by the normalized frequency at which two distinct object labels appear within the same scene graph (scene). For better visibility of co-occurrence patterns, we employ a maximum threshold of 0.05. In all measurements presented in Figure 10.10, we discern analogous patterns between the generated and ground truth datasets. Given that each object category can appear multiple times in an image (instances), we compare the distribution of counts (1, 2, and so on) for each object category using the Kullback-Leibler (KL) divergence of the generated dataset from the test dataset. Figure 10.11 displays the KL divergence of the object count distribution for each object category, showcasing a low average value of 0.048.

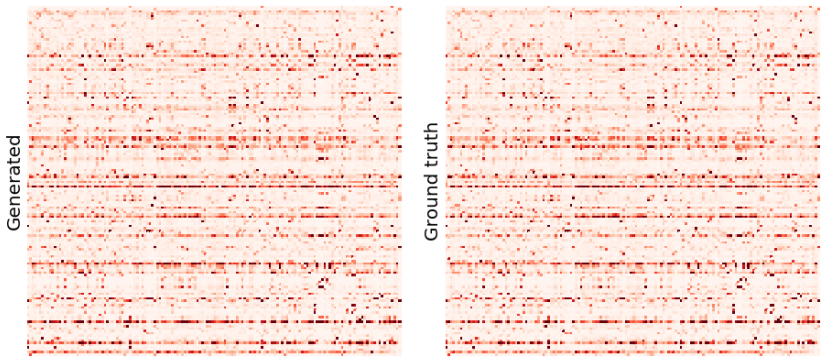
Object Occurrences in Synthesized Images Figure 10.12 illustrates the occurrence of objects detected (utilizing FasterRCNN) in the images generated by StyleGAN (unconditional) versus those produced by SG2Im based on scene graphs created by SceneGraphGen (SG2Im-SGG). The results show that SG2Im-SGG is more efficient in generating images with a higher number of detectable objects and superior object statistics compared to StyleGAN. However, StyleGAN's FID on Visual Genome is 66.3, which surpasses SG2Im-SGG. This disparity is attributed to the image generative models in use rather than the quality of the input scene graph.



(a) Object occurrence



(b) Relationship occurrence



(c) Object Co-Occurrence

Fig. 10.10. Dataset Statistics Comparison of generated scene graphs versus the ground truth scene graphs from Visual Genome. a.) Object Occurrence, b.) Relationship Occurrence, c.) Object Co-Occurrence

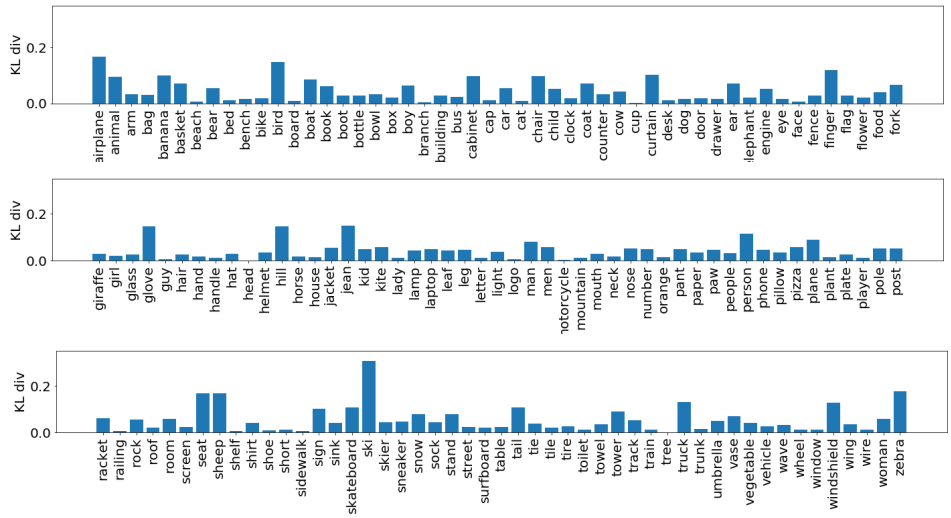


Fig. 10.11. Generated data KL divergence from test dataset, comparing the object frequency distribution for each object category

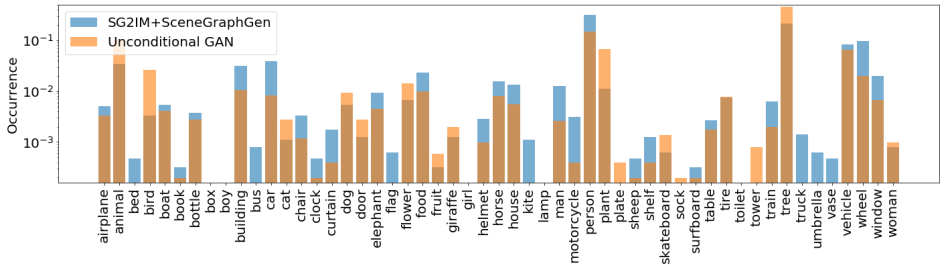


Fig. 10.12. Object Occurrence Comparison based on the detections of Faster R-CNN on the generated images by Unconditional-GAN (StyleGAN2) vs. SG2Im+SceneGraphGen model

Additional Applications Examples We offer additional examples of image generation in Figure 10.13, and examples of scene graph completion can be found in Figure 10.14.

10.5 Conclusions

In this chapter, we presented SceneGraphGen, a model explicitly designed for the generation of unconditional scene graphs. Our results exhibit the model's proficiency in discerning semantic patterns prevalent in real-world scenes and its ability to create diverse, yet plausible scene graphs. When benchmarked against established baselines, our model demonstrated superior performance on both graph-centric and image-centric metrics. Furthermore, we showcased the utility of our model and the generated graphs through a variety of applications, including image generation, identification of out-of-distribution samples, and scene graph completion. In terms of future work, we are keen to explore conditional variants of our model, for instance, using textual descriptions as a way to guide the creation of specific scene categories.

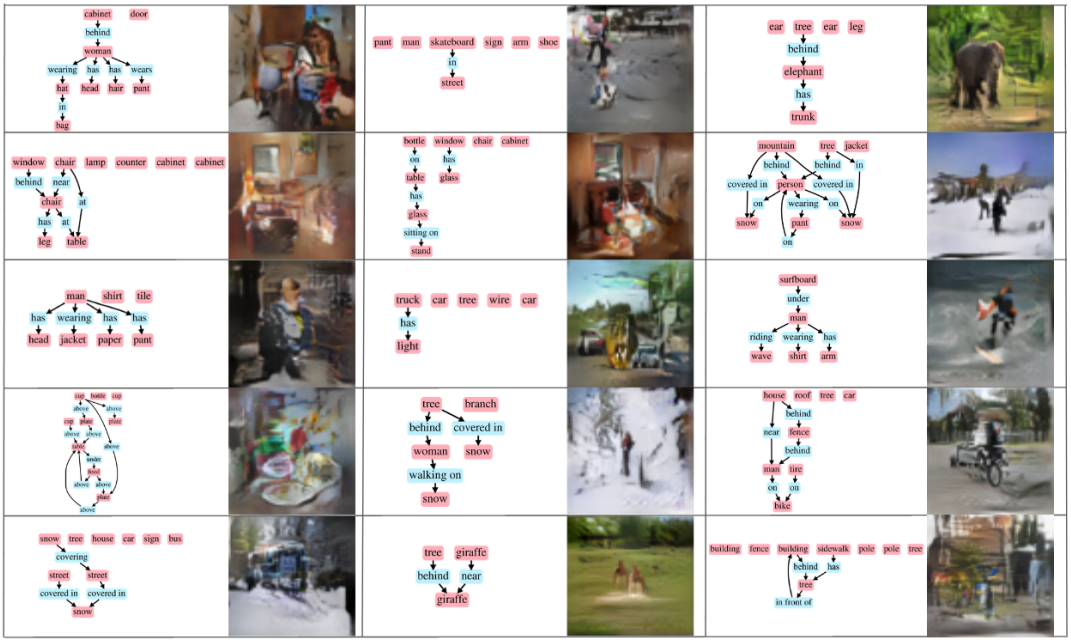


Fig. 10.13. Additional Image Synthesis examples with 64×64 resolution using SG2Im conditioned on the scene graphs generated by SceneGraphGen

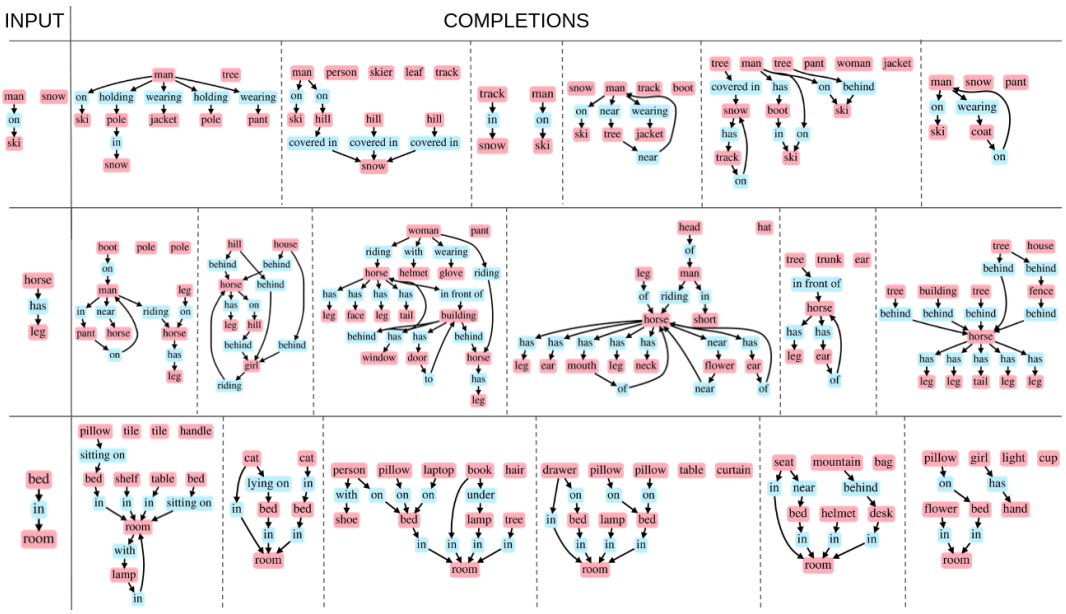


Fig. 10.14. Additional Scene Graph Completion Results from a partial scene graph using SceneGraphGen

11 Meta Image Generation from Scene Graphs

Fig. 11.1. Image generation samples of MIGS in various scene attributes on the BDD dataset [207] in a 10-shot setting.

11.1 Introduction

The field of image generation and manipulation has drawn significant research interest over the past years. Recent breakthroughs in deep learning for unconditional image synthesis [111, 113, 114, 115] have given rise to high-quality images that often appear real to human observers. Impressive outcomes have also been achieved in class-conditional image generation [18] and image generation from semantic segmentation masks [174]. Although the latter allows pixel-level control over the image, acquiring a segmentation map might be challenging in practical scenarios. A more straightforward alternative for representing image semantics is a scene graph [105], i.e., a graph where nodes symbolize the objects, and edges denote the relationships between them. Image generation from such graphs is an attractive concept as they offer complete control over the semantics and are easy to modify for scene editing. Image generation from scene graphs was first introduced in [104]. While the results were promising, the images generated by this model showed a lack of quality when trained on datasets with diverse scenes, as the network struggled to learn meaningful representations to accommodate these discrepancies. To tackle this issue, we propose using meta-learning to assist the network in focusing on specific tasks during training. Such a model can quickly adapt to a broad range of tasks during testing, with only a few training samples available. Furthermore, our approach allows us to introduce the task of few-shot learning for the scene graph to image generation problem, an appealing scenario that can assist in generating semantically meaningful images in applications with limited data. The main contributions of this work are as follows:

- We propose MIGS (Meta Image Generation from Scene Graphs), a novel meta-learning approach for the generation of images from scene graphs, outperforming previous methods.

- We introduce the concept of few-shot learning to the problem of scene graph to image generation.

- We present a new task sampling method for wild scenes that can benefit other image generation scenarios involving meta-learning.

Our proposed method is evaluated on automatically generated scene graphs for Berkeley Deep Drive [207], Action Genome [95], and Visual Genome [121] datasets, showing superior results compared to the baselines, both qualitatively and quantitatively. This superiority is further confirmed by a user study on the quality of the images. The project page and source code for this work can be accessed at `https://migs2021.github.io/`.

11.2 Related Work

Image Generation Generative models, particularly Generative Adversarial Networks (GANs) [67] and Diffusion Models [168], have significantly improved the quality of image generation in recent years. There is an array of work focusing on generative models for unconditional image generation [111, 113, 114, 115]. Conditional image generation models have also been investigated, leveraging a variety of priors such as semantic segmentation maps[25, 174, 250], natural language descriptions[141, 188, 280, 284], or transforming one image domain to another using paired [90] or unpaired data [291]. Among these, methods generating images from scene graphs [104] are most pertinent to our research.

Scene Graph-based Image Generation Scene graphs [105] are representational tools that describe images where nodes symbolize objects, and edges represent relationships between them. With the advent of large-scale scene graph datasets, such as Visual Genome [121], a wide range of scene graph-related tasks have been explored. Several works propose strategies for generating scene graphs from images [78, 166, 259]. Johnson *et al*. [104] pioneered the reverse task of image generation from scene graphs, employing a 2D layout as an intermediary between graphs and images. These layouts were then decoded into images using a Cascade Refinement Network (CRN)[25] architecture. Similar architectures were subsequently examined for interactive image generation[4] and semantic image manipulation [42]. Herzig *et al*. [77] introduced a model utilizing canonical scene graphs to enhance robustness against graph size and noise.

In section 10, we proposed a method [63] to unconditionally generate scene graphs and subsequently synthesize images from the produced graphs. Other related studies investigate image generation directly from layouts [224, 226, 285] or delve into 3D scene graphs [39, 244].

Advances in Meta-learning As discussed earlier in section 7, Model-agnostic Meta-Learning [60] (MAML) and Reptile [167] are well-known meta-learning approaches commonly used for the few-shot classification task. A combination of meta-learning and GANs was presented in [282], utilizing adversarial training for few-shot image classification. However, the application of meta-learning to the problem of few-shot image generation has been a relatively under-explored topic. FIGR [33] employed meta-learning for few-shot image generation, focusing on smaller datasets with black and white images, while few-shot image-to-image translation [147] generates images by transferring inputs from a source domain to a target domain. Even though meta-learning has been used for image generation in the past, its application has been limited to datasets with restricted diversity and low resolution. As far as we are aware, this is the first work on the few-shot generation of high-resolution scenes 'in the wild'.

11.3 MIGS

11.3.1 Problem Definition

In the topic of few-shot image classification meta-learning, a task is defined as a set of image-label pairs. In our work, we adapt this definition to the problem at hand and define a task as a pair of scene graphs and images. Given a set $\mathcal{D}$ of image $\mathcal{I}$ and scene graph $\mathcal{G}$ pairs, our preliminary dataset $\mathcal{D} = \mathcal{I}, \mathcal{G}$ is delineated. The dataset is further divided into various tasks based on a predefined task definition. During each iteration of the training phase, a task is randomly selected and the parameters of the scene graph-to-image model are optimized according to the selected task. In the testing phase, the trained parameters are then utilized to fine-tune the model on specific target tasks. Our methodology, comprised of the image generation and meta-learning components, is elaborated in the subsequent sections.

11.3.2 Image Generation

In order to address the task of generating images from scene graphs, we leverage the SG2Im [104] architecture as a the baseline. SG2Im receives the scene graph $\mathcal{G}$, with the nodes representing objects and the edges defining their relationships. The scene graph is parsed by a graph convolutional network (GCN) which operates on subject-predicate-object triplets, enabling the dissemination of information along the edges. This results in processed per-node features, wherein each object is attributed with an embedding vector that encodes information about the object itself as well as relationships with interconnected objects. These embeddings are subsequently used to predict a set of bounding boxes and segmentation masks for each object. The predicted boxes and masks are amalgamated to project the GCN features onto the image space and to generate a scene layout. Following this, an image generator receives the scene layout and fabricates an image corresponding to the defined semantics. To compel the network to generate images that are not only visually realistic but also semantically accurate, two image discriminators are applied to the generated image. The first discriminator discerns individual objects in a local context, while the second one classifies the entire image.

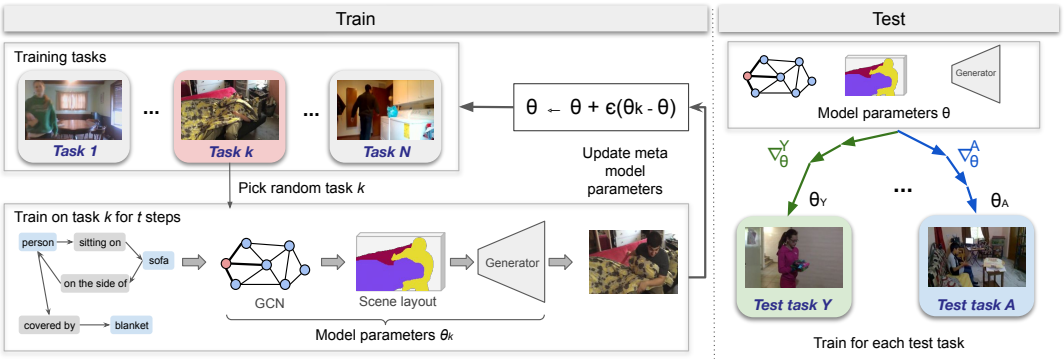


Fig. 11.2. **MIGS Overview**. Our methodology comprises two phases: Meta-training and Testing. During the meta-training phase, the model parameters, represented by θ, are updated on a randomly sampled task l in each iteration. One task represents a set of image and scene graph pairs that are collectively grouped according to specific criteria. We route a scene graph through a GCN to generate features from the node embeddings for creating a scene layout. This layout is relayed to the generator, which synthesizes the ultimate image. During the testing phase, we fine-tune the model θ for a predetermined number of shots on each specific task, resulting in the generation of our final images.

We utilize various loss functions for training the model. While most of the losses are borrowed from [105], we also introduce additional loss terms, outlined below. To prevent the model from generating trivial solutions, we implement the perceptual loss [106] $\lambda_p \mathcal{L}_p$ using the VGG network. The GAN losses are defined as: $\mathcal{L}_{\text{GAN,global}}$ for the whole image and $\mathcal{L}_{\text{GAN,obj}}$ to ensure the realism of individual objects. The auxiliary classification loss $\mathcal{L}_{\text{aux,obj}}$ is used to maintain the quality of the generated objects. The loss for predicting the bounding boxes, denoted by $\mathcal{L}_{box}$, is calculated using the $\mathcal{L}_1$ loss between the predicted and ground truth bounding boxes. Finally, the image loss $\mathcal{L}_{\text{im}}$, which is the $\mathcal{L}_1$ distance between the predicted image and the ground truth image, is used. The task loss $\mathcal{L}_\tau$ definition is presented in Equation 11.1.

$$\begin{aligned}\mathcal{L}_\tau = \lambda_b \mathcal{L}_{box} + \lambda_g \min_G \max_D \mathcal{L}_{\text{GAN,global}} \\ + \lambda_o \min_G \max_D \mathcal{L}_{\text{GAN,obj}} + \lambda_a \mathcal{L}_{\text{aux,obj}} \\ + \lambda_p \mathcal{L}_p + \lambda_{\text{im}} \mathcal{L}_{\text{im}},\end{aligned} \tag{11.1}$$

where the weighting factors are λ_b,λ_g, λ_o, λ_a, λ_p, λ_{im}.

$$\mathcal{L}_{\text{GAN}} = \mathop{\mathbb{E}}_{q \sim p_{\text{real}}} \log D(q) + \mathop{\mathbb{E}}_{q \sim p_{\text{fake}}} \log(1 - D(q)), \tag{11.2}$$

Here, p_{real} designates the real data distribution derived from the ground truth, and p_{fake} denotes the distribution of the generated faux images or objects. The discriminator's input is denoted by q.

To enhance the image quality produced by SG2Im and address a few-shot learning scenario, we modify the Reptile [167] algorithm for GANs, following the approach used in FIGR [33]. However, FIGR is designed for the problem of unconditional image generation and it experiments with images that contain a single object. In contrast, our setup is conditioned

on a scene graph with multiple objects, posing a more substantial challenge compared to FIGR. We meta-train all components of the SG2Im pipeline on an array of diverse scenes and demonstrate that this process facilitates high-quality image generation from the scene graph.

11.3.3 MIGS

In this section, we present our Meta Image Generation from Scene Graphs (MIGS) method and its constituent parts. Our primary aim is to rapidly and efficiently adapt a model, trained on a variety of images, to a specific task using just a few training shots. To address this problem, we refer to the meta-learning models [60, 167] and their application in a few-shot learning context. To our knowledge, MIGS is the first method introducing a few-shot learning scenario for the scene graph to image generation.

Task Definition Meta-learning models require a set of tasks T, composed of various learning problems τ. Each image generation from the scene graph task is represented by τ and contains a set of image-graph pairs $\mathcal{D}_\tau = \mathcal{I}, \mathcal{G}_\tau$. These pairs have been grouped into one task based on certain unifying criteria, such as the type of objects in the scene, specific background surroundings, or any other attributes of either the graphs or the images. Nevertheless, the criteria for splitting should be consistent across the entire set of tasks T. The task-splitting criteria depend on the dataset characteristics. It can be defined based on the scene attributes such as the time of day, the context, or by merely clustering the images into a set of clusters based on their visual attributes using an unsupervised clustering method such as [237].

Meta-learning We denote a loss function on a task τ as $\mathcal{L}_\tau$. For the sake of simplicity, we define $\mathcal{L}_\tau$ as a combination of all generator $\mathcal{L}_{\tau,G}$ and discriminator $\mathcal{L}_{\tau,D}$ losses in the SG2Im model. Our meta-learning objective is to identify such initial model parameters θ that for a randomly selected task τ, the loss $\mathcal{L}_\tau$ will be minimized after k iterations with only a few available data points. In brief, such an objective is defined as:

$$\min_\theta [\mathcal{L}_\tau \Big(U_\tau^k(\theta) \Big)], \tag{11.3}$$

where $U_\tau^k(\theta)$ represents an operator that uses image-graph pairs from $\mathcal{D}_\tau$ to update the weights θ, k times.

To determine such parameters θ, the models are trained using the Reptile algorithm [167], where it consists of inner and outer loops. In the inner loop, k iterations of the operator U are executed on the locally copied weights θ_l for a randomly sampled training task l. In the outer loop, the weight vector of the meta-model θ is updated in the inner loop using the calculated difference between θ and θ_l. This update is summarized as:

$$\theta \leftarrow \theta + \beta \frac{1}{L} \sum_{l=1}^{L} (\theta_l - \theta), \tag{11.4}$$

Here L defines the number of the tasks and β defines the meta-learning rate. These updates are performed separately for the image generator model and the two discriminators.

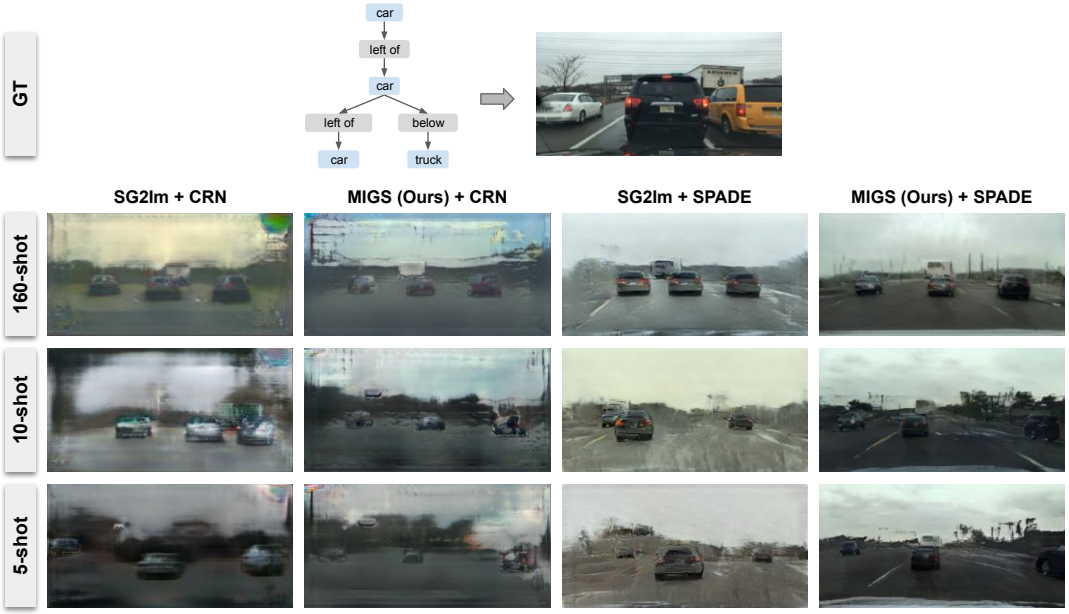


Fig. 11.3. **Qualitative Results on BBD.** This figure displays examples of images generated from the BDD dataset, corresponding to a task characterized by: daytime, rainy weather, and highway driving scenario. All images have been generated from the scene graph provided at the top. The MIGS model not only generates more lifelike images, but also shows higher accuracy when cross-verified with the given scene graph. Specifically, the MIGS + SPADE model is the only one where the truck is distinctly visible in all three scenarios.

Inference For the evaluation of the meta-model during inference, first we fine-tune the trained weights θ on the training split of each specific test task l to obtain the final weights θ_l for this task. Subsequently, we generate images from the (unseen) test set scene graphs of each task.

In order to make a fair comparison of our meta-models with baseline methods, we apply transfer learning to each of the non-meta models' weights Φ , where they are fine-tuned on each task to obtain the corresponding Φ_l. Then, we evaluate the images generated by θ_l and Φ_l on each task l.

11.4 Experiments and Results

Our method is evaluated using three datasets: Berkeley Deep Drive (BDD) [207], Action Genome (AG) [95], and Visual Genome (VG) [121]. We benchmark our model against various baselines, demonstrating its independence from the generator architecture. Performance improvements are noticeable across two different generator architectures, specifically CRN [25] and SPADE [174]. To quantify the quality and realism of the images generated by our method, we provide the Fr'echet Inception Distance (FID) [79], Kernel Inception Distance (KID) [13], and precision and recall metrics [200]. Additionally, we compare the image samples generated by our method with those from related work across various scenarios. We also provide the architectural details of CRN, SPADE, and GCN networks in the appendix.

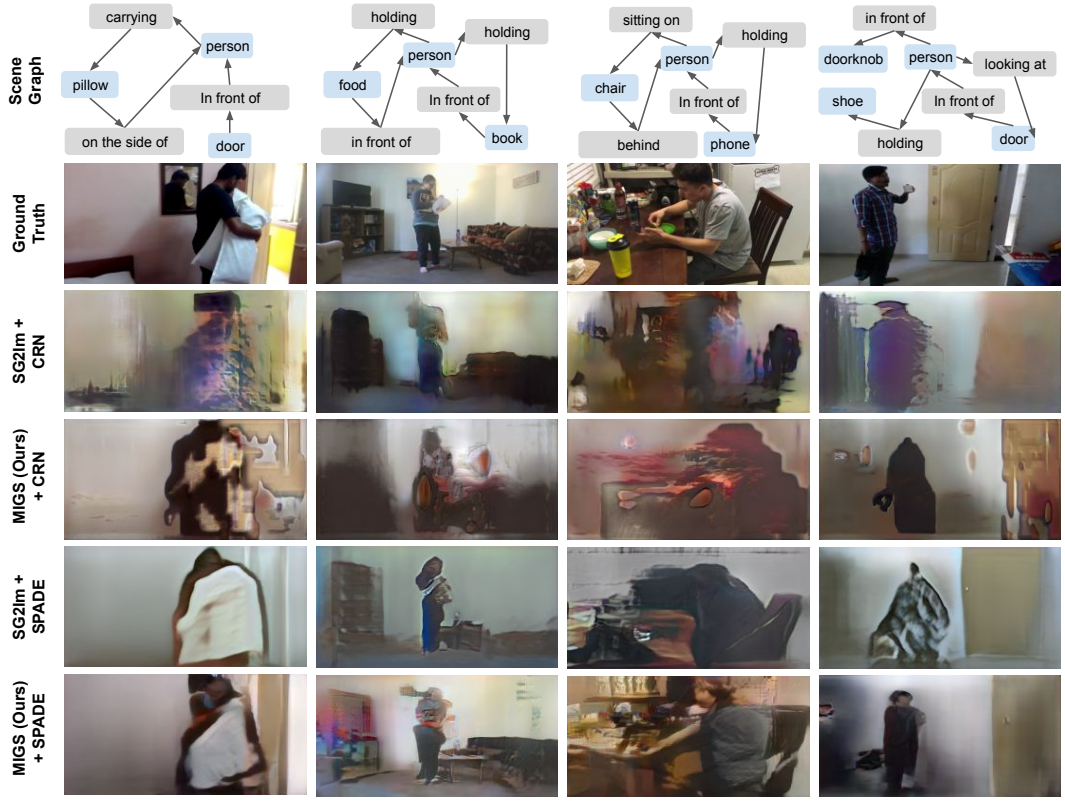


Fig. 11.4. **Qualitative Results on Action Genome.** This figure showcases examples of images generated from the Action Genome dataset. Each column corresponds to one task (i.e., a video sequence) on which each model was fine-tuned. As can be seen, the MIGS results on each video sequence contain a significant level of detail when compared to their baseline counterparts.

11.4.1 Datasets

Considering the lack of standardized datasets for meta-learning in conditional image generation, we sought out datasets that naturally lend themselves to categorization into multiple tasks.

Berkeley Deep Drive (BDD) [207] This dataset comprises images taken in city streets, residential areas, and highways under diverse scene conditions. Given that BDD does not include associated scene graphs with images, we generate spatial scene graphs automatically using the provided ground truth bounding boxes. This process results in six mutually exclusive spatial relationships: `left of`, `right of`, `above`, `below`, `inside`, and `surrounding`. To focus primarily on the objects and their relationships, we pre-process the images and crop the areas of interest that contain all objects with minimal background.

Based on the meta-learning objectives, we divide the BDD dataset into tasks based on the given image attributes such as time of day, weather conditions, and driving scenarios. We exclude tasks that have fewer than 500 images, resulting in a total of 23 distinct meta-learning tasks. Of these, 20 tasks are designated for training and validation, and the remaining 3 for testing.

Action Genome (AG) [95] Originally developed for action recognition, the Action Genome dataset comprises video frames showing human interactions with objects within a scene. We directly use human-object relationship labels from Action Genome to generate semantically meaningful scene graphs.

Given the large volume of videos in the AG dataset, each featuring different actions, we use these videos as criteria for task division. We remove frames from videos that either do not feature humans, or only feature humans with no other objects, as these do not allow for the construction of meaningful scene graphs. We then select only horizontal videos that have at least 30 frames annotated by humans for our tasks. Following this process, we are left with 735 tasks, from which 662 are assigned to training and validation, while 73 are reserved for testing.

Visual Genome (VG) [121] The Visual Genome dataset is fitting for image generation from scene graphs, as it provides images along with their corresponding semantic scene graphs and bounding box annotations. Given the absence of specific scene attribute annotations in the VG dataset, we construct tasks by classifying all images into 100 clusters using the SCAN model [237] pre-trained on the ImageNet dataset in an unsupervised manner. The first 60 clusters are used for the pre-training phase, and the final 40 clusters are allocated for evaluation.

11.4.2 Experimental Setup

Our models are trained to generate images of dimensions 128×256 for BDD and AG, and 64×64 for VG. All meta-learning models utilize an inner learning rate of 0.0001 and are trained for 10 inner iterations with the Adam optimizer. The outer loop employs a learning rate of 1 and makes use of SGD. The total number of training iterations depends on the specific model and dataset; for instance, as AG is more diverse than BDD, it requires a longer convergence time. Meta-learning models on AG are trained for 40000 outer loop iterations, while those on BDD are trained for 30000.

The baseline model, SG2Im [104], undergoes training on the same data as MIGS (across all training tasks) during the pre-training phase. During the testing phase, the pre-trained model serves as the initial point and is fine-tuned and tested in a similar few-shot setting to MIGS. In all datasets, both models are trained until they reach convergence. The SG2Im model undergoes training for 250k iterations on AG, 200k iterations on BDD, and 40k iterations on VG. Conversely, the MIGS model is trained for 40k and 30k iterations on AG and BDD respectively, and 8k iterations on VG.

11.4.3 Results

The quantitative outcomes of our experimental work are presented in Table 11.1, Table 11.4, Table 11.2, Table 11.5. We set our method against two baselines: the SG2Im model with its original CRN decoder, and an alternative version of SG2Im that utilizes the SPADE network as the decoder to enhance the quality of generation.

Method	Decoder	FID ↓	KID $\cdot 10^3$ ↓	FID ↓	KID $\cdot 10^3$ ↓	FID ↓	KID $\cdot 10^3$ ↓
		160-shot		10-shot		5-shot	
SG2Im [104]	CRN	194	210	176	186.5	196.8	224.2
MIGS (Ours)	CRN	158.5	156.4	157	158.4	183.5	187.6
SG2Im	SPADE	66.1	42.2	70.6	48.3	95.2	73.1
MIGS (Ours)	SPADE	**49.5**	**26.7**	**46.1**	**24**	**53.5**	**30.7**

Tab. 11.1. Quantitative results using FID and KID trained and fine-tuned on BDD100k using 5,10 and 160 shots.

Method	Decoder	F_8 ↑	$F_{1/8}$ ↑	F_8 ↑	$F_{1/8}$ ↑	F_8 ↑	$F_{1/8}$ ↑
		160-shot		10-shot		5-shot	
SG2Im	CRN	0.101	0.135	0.063	0.131	0.06	0.057
MIGS (Ours)	CRN	0.06	0.176	0.052	0.123	0.05	0.06
SG2Im	SPADE	0.486	0.612	0.462	0.45	0.329	0.438
MIGS (Ours)	SPADE	**0.7**	**0.86**	**0.79**	**0.854**	**0.74**	**0.823**

Tab. 11.2. Quantitative results using Precision and Recall trained and fine-tuned on BDD100k using 5,10 and 160 shots.

BDD Results The performance of the aforementioned models on the BDD dataset is demonstrated with a range of shot values from 5 to 160, both quantitatively in Table 11.1 and Table 11.2 and qualitatively in Figure 11.3. Across all three experiments, we notice that all metrics, including FID, KID and Precision / Recall display approximately a twofold improvement compared to their counterparts in the model without meta-learning. A user study was conducted on BDD images, wherein users were asked to rank the quality of images (generated by different methods from the same scene graph) and evaluate whether the scene reflects the designated attribute. According to the user study, MIGS + SPADE was commonly chosen as the most realistic method, followed by MIGS + CRN, and then SG2Im + SPADE and SG2Im + CRN, ranked third and fourth respectively. The specific ranking percentages are outlined in Table 11.3. The table represents the percentages of users who selected a specific method as a particular rank based on image quality. Regarding attribute representation, MIGS + SPADE and MIGS + CRN secured the 1st and 2nd ranks, while SG2Im + SPADE and SG2Im + CRN were ranked 3rd and 4th respectively.

Figure 11.3 depicts examples of scene graphs and images generated using the baselines and our method for a single test task from BDD. These images clearly illustrate that our method surpasses all baselines, being able to generate highly detailed and realistic-looking images, even in exceptionally challenging scenarios where only 5 frames are available for training. Furthermore, Figure 11.1 exhibits sample images generated for a diverse set of training tasks. Our method effectively captures the variations in scenes associated with changes in daytime and driving scenarios.

Our method's quantitative performance using precision ($F_{1/8}$) and recall (F_8) metrics, in comparison to the baselines, is presented on both the BDD (Table 11.2) and the AG (Table 11.5) datasets.

We present further qualitative results from applying MIGS to the BDD dataset under the conditions of 160-shot learning (Figure 11.5), 10-shot learning (Figure 11.6), and 5-shot

learning (Figure 11.7). The model consistently generates persuasive results across a wide range of tasks. All images were produced using the MIGS model equipped with a SPADE generator. Both the 160 and 10-shot learning models demonstrate their capacity to generate a variety of images within a single task. The quality of their output is comparable, and both are adept at rendering intricate details, such as cloud formations in an overcast task or the reflective glares of rain on a road. The 5-shot learning model is also capable of producing realistic images for many tasks. However, it encounters difficulty when asked to produce significantly varied images within a single task, and it is less successful at rendering detailed features compared to the 160 and 10-shot learning models. This outcome is to be expected, given the minimal amount of training images available to the model, which limits the diversity it can accurately represent.

User Study For the user study, we generated 600 images at random, divided evenly across three different scene attributes: daytime, dawn/dusk, and night. Each image was evaluated by three different participants. In each instance, the participant was presented with four randomly ordered images—each representing one of our four studied methods—and asked to rank them. Additionally, we offered a checkbox for each image, allowing the user to indicate whether a particular attribute was fulfilled.

Method	Decoder	Rank			
		1 (%)	2 (%)	3 (%)	4 (%)
SG2Im [104]	CRN	15.79	23.34	26.55	**35.58**
MIGS (Ours)	CRN	24.46	**25.57**	25.16	24.81
SG2Im	SPADE	25.36	24.60	**28.29**	21.74
MIGS (Ours)	SPADE	**34.72**	26.48	20.0	17.79

Tab. 11.3. User study ranking results on randomly sampled images from the BDD dataset. The provided results represent the percentages of users who ranked the method in the specified position.

AG Results On the Action Genome (AG) dataset, the model is trained on all training images for each test task, which consists of around 30 frames, and is evaluated on the frames extracted from the complete videos that were not used for training. The test set includes approximately 65,000 images.

The AG dataset poses a considerable challenge for image generation from scene graphs. It contains labels for only a few selected objects in an image, and these objects are often quite small, such as a phone or book. Example images generated by our model and the baselines on the AG dataset are displayed in Figure 11.4, while the quantitative performance on AG is reported in Table 11.4. Given the complexity of the dataset in use, it's not surprising that the results differ from those on BDD. However, even under these conditions, our method generates images with more detail compared to the respective baseline. This improvement is further confirmed quantitatively by the substantial reduction in both FID and KID scores.

In Table 11.5, we present the quantitative results using the precision and recall metrics.

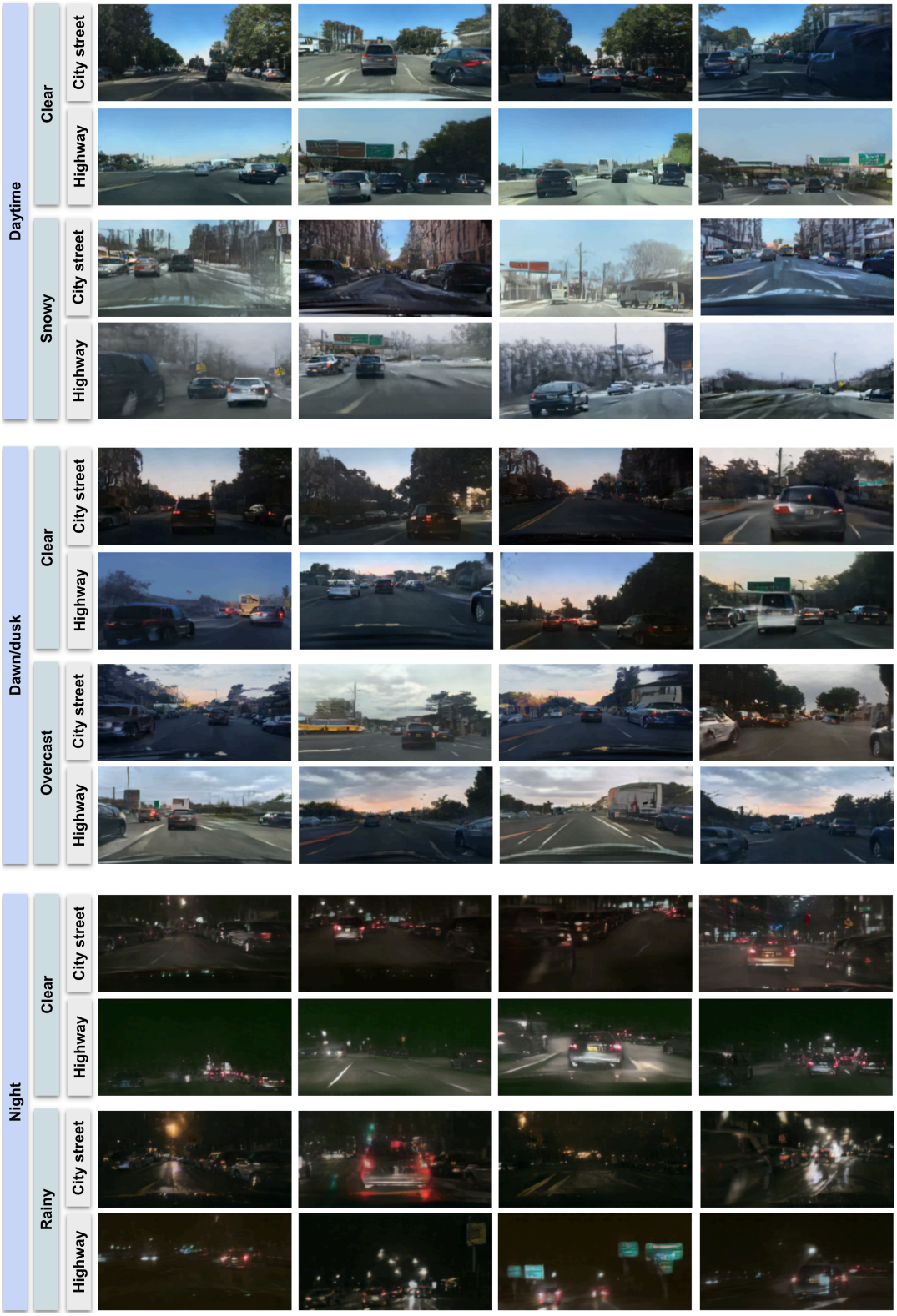


Fig. 11.5. Additional quantitative results of 160-shot learning on Berkeley Deep Drive (BDD) with MIGS + SPADE model.

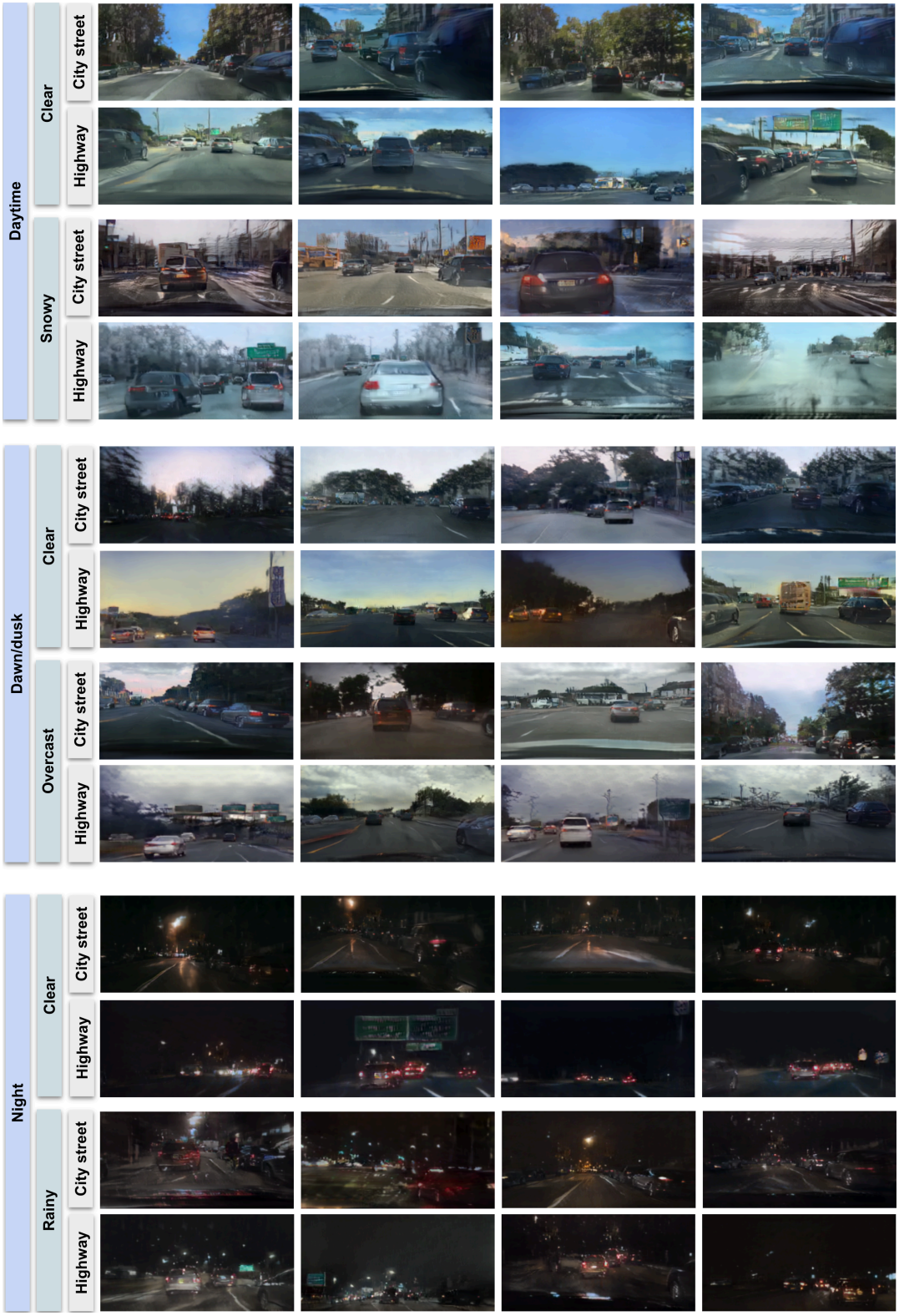


Fig. 11.6. Additional quantitative results of 10-shot learning results on Berkeley Deep Drive (BDD) with MIGS + SPADE model.

Fig. 11.7. Additional quantitative results of 5-shot learning results on Berkeley Deep Drive (BDD) with MIGS + SPADE model.

Method	Decoder	FID ↓	KID $\cdot 10^3$ ↓
SG2Im [104]	CRN	198	163.4
MIGS (Ours)	CRN	174.5	137.8
SG2Im	SPADE	141.3	76.3
MIGS (Ours)	SPADE	**98.1**	**47.4**

Tab. 11.4. Quantitative results on the Action Genome dataset, contrasted with related work.

Method	Decoder	F_8 ↑	$F_{1/8}$ ↑
SG2Im	CRN	0.217	0.116
MIGS (Ours)	CRN	0.167	0.09
SG2Im	SPADE	0.59	0.31
MIGS (Ours)	SPADE	**0.6**	**0.5**

Tab. 11.5. Additional quantitative results on the Action Genome dataset contrasted with related work.

We offer further qualitative results from the AG dataset on Figure 11.8. These additional examples underline how our method accurately captures the semantic relationships between objects, as defined by the scene graph, in a variety of circumstances.

Additionally, we showcase how our model, when trained on a singular video from the AG dataset, can create a variety of semantically different images. These generated images mimic different frames from the video while consistently maintaining the same style (Figure 11.9).

VG Results The comparative performance of MIGS and SG2Im [104] on the VG dataset is presented in Table 11.6 and illustrated in Figure 11.10. The results demonstrate that MIGS consistently surpasses the baseline across all metrics for various shot values, even with a reduced number of training epochs. Despite the wide diversity and wild nature of the images in the VG dataset, MIGS exhibits the ability to generate images that appear more lifelike than those generated by the baseline model.

Ablation Study On Memorization In Figure 11.11, we illustrate some training examples for a 5-shot training task along with generated images from the same task's test set, accompanied by their corresponding ground truth images. As can be observed in the figure, the generated images present several differences from the five training samples in aspects like car colors and shapes. Interestingly, the model has managed to generate cars in colors distinct from those it was trained on. As indicated by the Precision and Recall metric in Table 11.2, and FID and KID metrics, the diversity of the generated images surpasses that of the original SG2Im model. During the testing phase, for each task composed of distinct images and their corresponding

Method	Decoder	FID ↓	KID $\cdot 10^3$ ↓	FID ↓	KID $\cdot 10^3$ ↓	FID ↓	KID $\cdot 10^3$ ↓
		160-shot		10-shot		5-shot	
SG2Im [104] (All epochs)	SPADE	55.20	35.54	81.42	59.39	91.79	68.52
MIGS (Ours, 1/3 epochs)	SPADE	54.83	34.21	76.56	52.02	84.87	59.38
MIGS (Ours, All epochs)	SPADE	**54.24**	**29.00**	**75.96**	**50.69**	**83.54**	**55.28**

Tab. 11.6. Quantitative comparison of MIGS and SG2Im on the VG dataset, fine-tuned on 5, 10, and 160 shots.

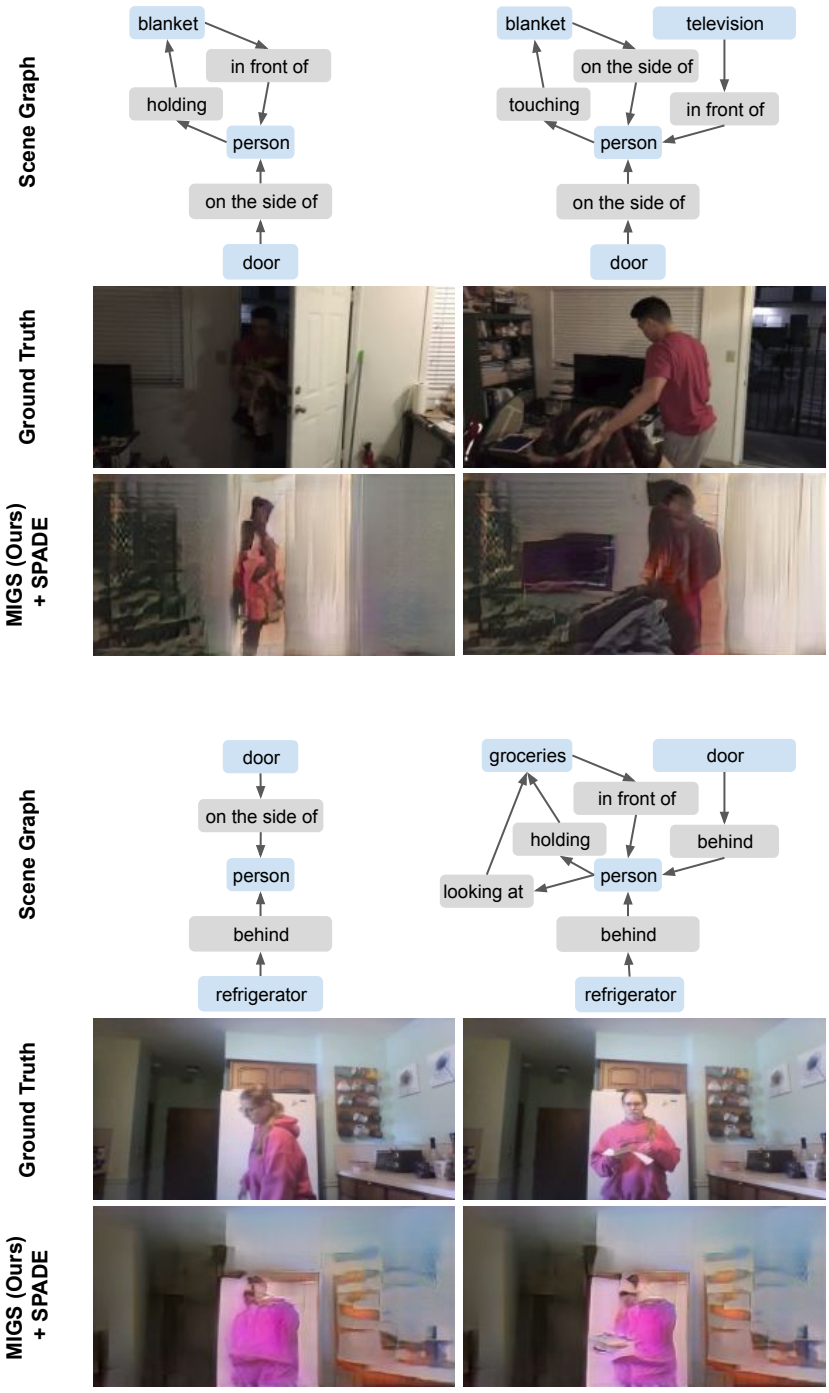


Fig. 11.8. Extra examples of images produced by the MIGS + SPADE model, trained on individual video classes from the Action Genome.

Fig. 11.9. Example images generated with the MIGS + SPADE model, trained on a specific video class from the Action Genome. These examples highlight how our approach successfully interprets the semantic relationships between objects, as indicated by the scene graph.

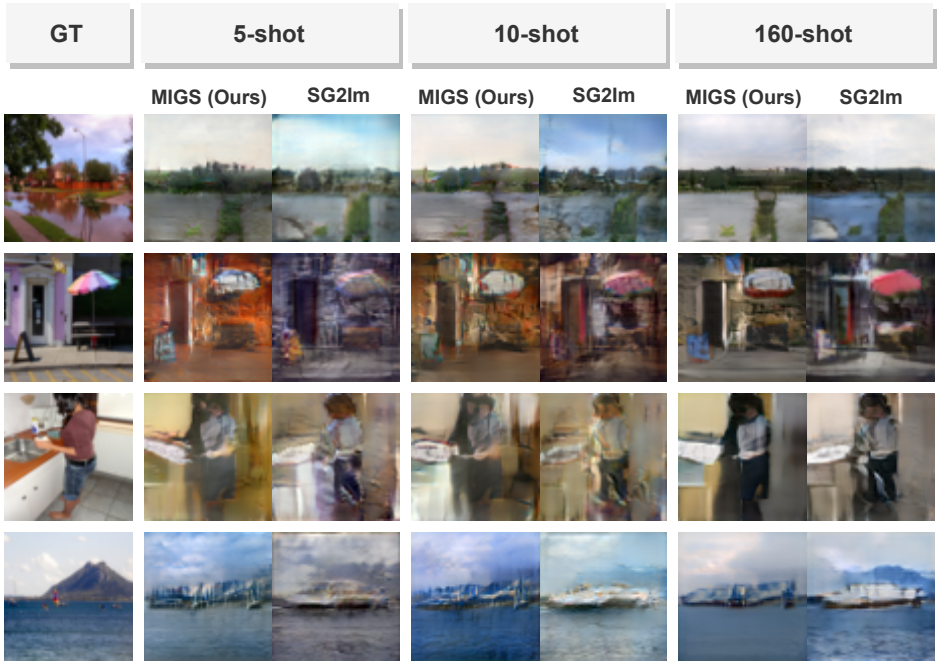


Fig. 11.10. Sample images generated using MIGS + SPADE and SG2Im + SPADE on the Visual Genome dataset.

scene graphs, five separate images (in the case of 5-shot) are sampled for model fine-tuning. Subsequently, other images from the same task possessing similar attributes are sampled. Their scene graph is employed to generate images, which are then compared with their ground truth counterparts.

11.4.4 Discussion

Our experimental results demonstrate that meta-learning applied to image generation from scene graphs remarkably outperforms the corresponding baselines, nearly doubling the effec-

Fig. 11.11. This figure demonstrates 5-shot learning results on the Berkeley Deep Drive (BDD) using the MIGS + SPADE model, highlighting the effect of memorization.

tiveness based on the metrics employed, when compared to the model without meta-learning integration.

In the cases of both BDD and AG datasets, MIGS + SPADE drastically surpasses the respective baseline, with almost double the precision and recall. This enhancement signifies that images generated through the meta-learning technique look more realistic (precision) and cover a broader range of the real data distribution (recall). The outcomes achieved with MIGS + CRN are relatively equivalent to those of the baseline.

Our proposed method proves to be advantageous across all contexts, even when restricted to merely 5 training samples. Despite the significant performance boost rendered by meta-learning, we noticed the emergence of considerable artifacts in images, such as grayish or blurred sections, when the model was trained with a higher number of shots. We associate this issue with overfitting, a behavior also noted in non-meta-learning models when subjected to prolonged training. We have also found that meta-learning is particularly susceptible to overfitting, a phenomenon underscored in Table 11.1, where 10-shot training outperforms the 160-shot setup. Therefore, it may be beneficial to monitor and mitigate overfitting by limiting the number of samples or implementing model regularization.

The qualitative results garnered from the BDD dataset vary considerably, a factor attributable not only to AG's more complex environment but also its unsuiteness for the image generation task. Given its design for action recognition in videos, only objects interacting with the person are annotated. This often leads to instances where certain object frames are neglected until an action is performed or remain unannotated throughout the entire video if the main actor doesn't interact with them.

Such inconsistencies can confound the model, resulting in subpar quality. Additionally, this dataset includes a multitude of small objects in relation to the frame size. We posit that a better-constructed dataset for semantically meaningful scene graphs should enable the model to produce results akin to BDD. The findings on the VG dataset suggest that incorporating a straightforward yet effective task construction scheme, like clustering (which can be applied to any other dataset), with the meta-learning approach, can enhance image generation performance. The impact of task construction on the performance of meta-learning presents a fascinating area for future research exploration.

11.5 Conclusion

In this chapter, we introduced MIGS, a novel meta-learning method for generating images from scene graphs, marking the first attempt at few-shot image generation of diverse real-world scenes. This adaptable method can be implemented across various generator architectures and datasets. Evaluation results from three distinct datasets demonstrate that our proposed meta-learning strategy significantly enhances the quality of generated images in all contexts. Our method proves that it's feasible to produce high-quality images with as few as 5 shots of data. The superior performance of our method compared to earlier works is demonstrated both quantitatively and qualitatively in the results.

12 Semantic Image Manipulation using Scene Graphs

12.1 Introduction

The objective of image understanding is to extract meaningful and comprehensive information from an image. Over the years, various tasks such as object recognition [189], visual relationship detection [153], and image captioning [103] have greatly benefited from deep representation techniques. Additionally, image synthesis tasks, like generating realistic images from semantic layouts [25, 250, 285] or natural language descriptions [81, 141, 188, 280, 284], heavily rely on image understanding.

Despite these advancements, image *manipulation*, involving alterations, additions, or removal of image elements, has received comparatively less attention. Traditional image manipulation typically involves pixel-level changes using photo editing software and low-level tools like in-painting. Some methods target high-level manipulations, focusing on specific objects, like facial modifications or reenactment.

Deep generative models, especially Generative Adversarial Networks (GANs) [67], have also been applied to image manipulation. However, the existing approaches often require user interfaces that may involve tedious segmentation maps for image editing [82, 169], requiring pixel-level changes on semantic segmentation maps.

A more efficient approach to image manipulation, based on semantic understanding, encompassing objects, their relationships, and attributes, could offer an easier and more user-friendly image editing experience, reducing the need for extensive manual effort from the user.

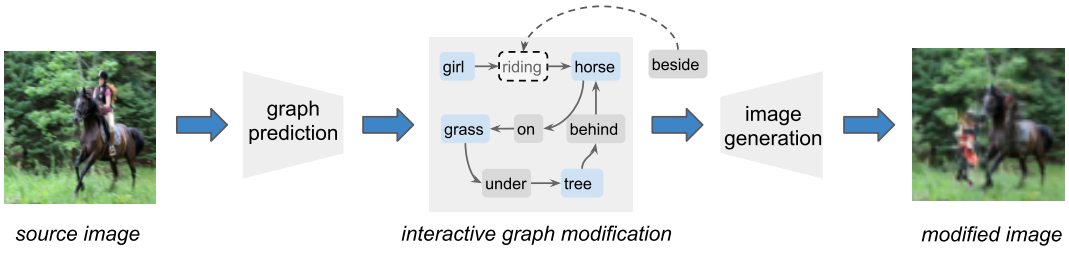


Fig. 12.1. **Semantic Image Manipulation.** Our approach begins with an input image, and from that, we predict a corresponding scene graph that represents the objects, their attributes, and their interactions within the image. The user can then interact with this scene graph by making changes to its nodes or edges. Based on the user's edits, we generate a new image that incorporates the modifications made to the scene graph. This enables the user to seamlessly edit the image's content and relationships by manipulating the underlying scene graph, resulting in a visually updated image reflecting the desired changes.

Motivated by the mentioned limitations, we propose a framework for semantic image editing that allows the user to modify a scene graph, which is a representation of the objects, attributes and interactions in the image (Figure 12.1). [1]

Scene graphs define a scene by using nodes to represent objects and edges to represent relationships between them. This formulation enables the user to perform different editing functions by simply changing the nodes or edges in the graph. For example, the user can delete unwanted tourists in a holiday photo by removing the corresponding <person> nodes, instead of manually segmenting, deleting and in-painting them. The user can also replace graph nodes with different semantic categories, for example replacing <clouds> with <sky>. Moreover, the user can re-arrange the spatial composition of the image by swapping object nodes on the image canvas. To the best of our knowledge, this is the first approach to image editing that also enables semantic relationship changes, for example changing *a person walking in front of the sunset*" to *a person jogging in front of the sunset*" to create a more scenic image. Semantic image editing based on scene graphs is not only useful for photo editing, but also for other domains, such as robotics. For example, a robot tasked to tidy up a room can manipulate the scene graph of the perceived scene by moving objects to their designated spaces, changing their relationships and attributes: *clothes lying on the floor*" to *folded clothes on a shelf*", to obtain a realistic view of the room after the action is performed.

Previous research efforts have predominantly focused on two separate tasks: generating a scene graph from an image [139, 166] or generating an image from a given scene graph [4, 104]. However, our work addresses the unique challenge of combining these tasks into a single problem. Specifically, we aim to generate an image while simultaneously manipulating it based on the provided scene graph. This manipulation requires preserving the identity of certain elements, such as objects, in the scene while modifying the relationships between them. For instance, changing the relationship between a "boy" and "grass" from "sitting on" to "standing on" necessitates generating an image with the same "boy" while adjusting the positioning according to the new relationship.

Creating a fully supervised dataset that contains pairs of images before and after manipulation, along with their corresponding scene graphs, is highly challenging and resource-intensive. However, our approach does not rely on such extensive supervision. Instead, we propose a method that leverages existing training data comprising image and scene graph pairs. Through this semi-automatic process, users interact indirectly with the image by modifying the nodes and edges of the scene graph. This allows for flexible changes in the visual entities' interactions, both semantically and spatially, without the need for manual image editing.

Our contribution is an innovative approach that enables various types of edits using a single model. This includes modifying the semantic relationships between objects while preserving the original content of the image. The resulting images accommodate user-specified changes and seamlessly integrate new or modified content as desired. Our method, known as SIMSG [42], achieves this manipulation by employing masking techniques on specific data components, such as object features or bounding box information, depending on the manipulation mode.

[1] https://he-dhamo.github.io/SIMSG/

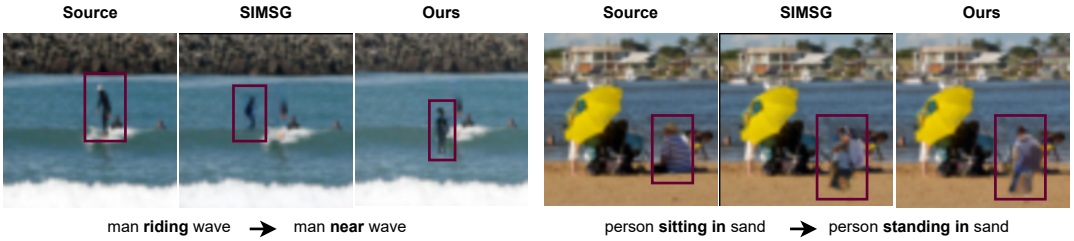


Fig. 12.2. DisPositioNet preserves the object features by disentangling them in the latent space, leading to more realistic objects compared to SIMSG [42] in semantic image manipulation.

Despite the promising results achieved by SIMSG, there exists a limitation in the way it learns object features for manipulation. Specifically, the learned object features encode both pose and appearance simultaneously, making it challenging to preserve one aspect while modifying the other. This issue becomes apparent in scenarios where we aim to change object relationships while maintaining specific visual attributes. For instance, in Figure 12.2, when the relationship of a `man` changes from `riding` to `near` the `wave` (left), or from `sitting` to `standing` in the `sand` (right), SIMSG (middle column) may lose some visual characteristics of the man during the adaptation to the new pose, such as changes in body shape or outfit color.

To address this limitation, we propose a novel network called **DisPositioNet**, which stands for **Dis**entangled **Pos**e and **I**dentity in Semantic Image Manipula**tion**. DisPositioNet aims to disentangle object features using a self-supervised variational approach, employing two branches to encode pose and appearance features separately in the latent space. By disentangling features in the image manipulation process, we hypothesize that the model will preserve visual attributes more reliably, leading to more meaningful and accurate results. Additionally, to further enhance feature disentanglement and ensure compatibility with the variational embedding, we propose **DSGN**, a disentangled scene graph neural network for extracting disentangled features from scene graphs.

Our proposed models, DisPositioNet and SIMSG, are thoroughly evaluated on standard benchmarks for image manipulation, including CLEVR [102], Visual Genome [121] and Microsoft COCO [145]. The evaluation demonstrates the superior performance of our models compared to previous approaches, both quantitatively and qualitatively. In particular, DisPositioNet outperforms SIMSG [42] in cases where preserving object appearance while changing pose is crucial.

To summarize our contributions:

- A method for semantic image manipulation that allows for constellation changes without the need for pairs of before and after modifications.

- A self-supervised approach for disentangling pose and appearance features in semantic image manipulation, which does not require labeled information for the disentanglement task.

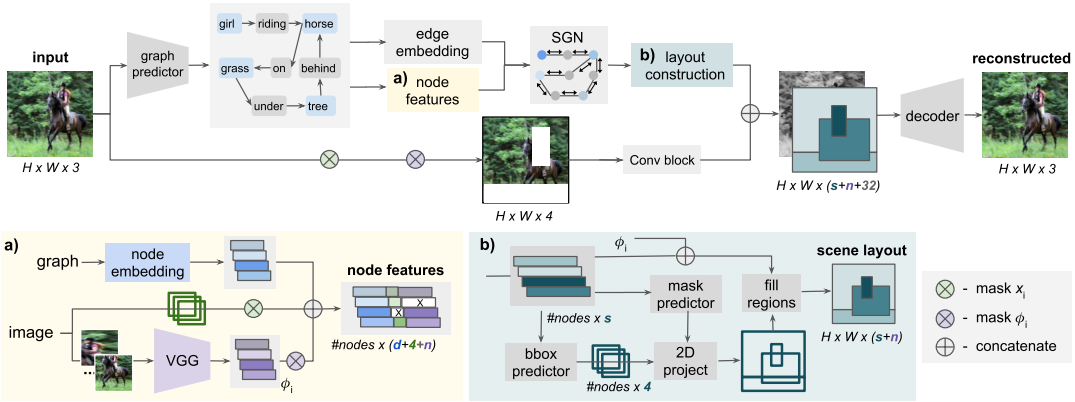


Fig. 12.3. **Training Overview.** *Top:* We employ a two-step process in our approach. First, given an input image, we predict its corresponding scene graph, capturing the objects and their relationships within the image. Next, we reconstruct the input image using a masked representation. a) Within the scene graph, the graph nodes o_i (depicted in blue) are enriched with bounding boxes x_i (shown in green) and visual features $z_{I,i}$ (highlighted in violet) obtained from cropped objects. To create the reconstruction, we randomly mask certain boxes x_i, object visual features $z_{I,i}$, and parts of the source image. The model then reconstructs the same scene graph and image utilizing the remaining information. b) To further visualize the process, we project the per-node feature vectors to a 2D space using the bounding box predictions from SGN (Scene Graph Network). This projection facilitates a clearer representation of the features in a reduced-dimensional space.

- A disentangled scene graph neural network, DSGN, that efficiently extracts disentangled features from scene graphs.
- A variational latent representation that offers higher diversity in image manipulation.
- Superior quantitative and qualitative performance compared to state-of-the-art methods on three well-established public benchmarks.

These contributions collectively demonstrate the effectiveness of our approach and highlight its potential for advancing the field of semantic image manipulation. The source code and additional information for SIMSG and DispoNet can be found at our project page [2].

12.2 Related Work

Image generation Deep generative models [67, 119, 171, 181, 238] have played a pivotal role in advancing (un)conditional image synthesis techniques. The progress in image generation has been largely driven by Generative Adversarial Networks [67] and diffusion models [168]. Among these developments, conditional variants [161] have been extensively explored, allowing image generation based on different input conditions.

Conditional image generation involves modeling the conditional distribution of images given specific prior information. Practical tasks, such as denoising or inpainting, can be viewed as generating images from noisy or partially available inputs. Existing literature has investigated various conditional models for different use cases, including conditioning on image labels

[2] https://scenegenie.github.io/DispositioNet/

[161, 170], attributes [261], lower-resolution images [129], semantic segmentation maps [25, 174, 250], natural language descriptions [141, 188, 280, 284], or performing domain translation using paired [90] or unpaired data [291]. Pix2Pix [90] and CycleGAN [291] are notable examples of models that enable general image translation between different domains, with the latter relaxing the requirement for paired training data. Additionally, some works focus on unconditional image generation within specific domains, such as faces [112, 113].

Semantic image generation, where an image is generated from an input semantic map, has also been addressed in the literature [25, 174, 250]. In these approaches, the input semantic map provides essential information for image synthesis.

Another relevant line of research involves generating images from layout information, represented as sets of bounding boxes and class labels for each scene instance [81, 224, 285]. More closely related to our work are methods that generate images conditioned on a scene graph [4, 53, 91, 104]. In such approaches, the layout serves as an intermediate step to translate the graph structure into the image space. For instance, Sg2im [104] was the pioneering method that tackled this task using a combination of object-level and image-level GAN loss in a supervised manner. Subsequent works have sought to enhance the performance on this challenging task by incorporating per-object neural image features to increase diversity [4], leveraging meta-learning for improved learning on highly diverse datasets (MIGS) [53], and utilizing contextual information to refine the layout (CoLoR) [91].

Image manipulation Unconditional image synthesis remains a challenging task, especially when dealing with complex scenes. In contrast, image manipulation focuses on specific parts of the image, allowing for the generation of higher-quality samples. This task involves partial image generation and often involves a user interface to indicate the subject of change [133].

Early works in scene-level image editing took a hand-crafted approach, where some image parts were replaced with sample patches from a database [84].

Image manipulation based on semantics has mostly been confined to object-centric scenarios. For instance, face editing has been automated using attributes [32, 126, 286] or accomplished through manual edits with paintbrushes and scribbles [19, 290]. Image composition, which also involves individual objects, faces the challenge of decoupling appearance and geometry [6, 279].

At the scene level, generative models have been used for tasks like inpainting [176] with additional guidance from semantics [271] or edges [165, 274]. Other works have allowed user-specified content [101, 287] and object removal [41, 211].

A learned model on a semantic layout representation was employed by Hong et al. [82], enabling users to make changes to the image by adding, moving, or removing bounding boxes. Similarly, SESAME [169] enables users to draw masks with semantic labels on images to indicate changed pixels. EditGAN [146] offers detailed object part segmentation map modifications to alter object appearance.

Image synthesis from semantics also facilitates interactive editing through changes applied to the semantic map [250]. In contrast, our approach follows a semi-automatic methodology, encompassing all these scenarios using a single general-purpose model, where edits are incorporated via a scene graph. Additionally, Hu *et al*. [84] proposed a hand-crafted image editing approach using graphs for library-driven patch replacement. While [84] focused on copy-paste tasks, our framework enables high-level semantic edits and addresses object deformations.

Our proposed framework is trained by reconstructing the input image, thereby avoiding the need for paired data. A similar idea was explored by Yao *et al*. [266] for 3D-aware modification of scenes (specifically 3D object pose) by disentangling semantics and geometry. However, their approach is limited to specific scene and object types (streets and cars) and requires CAD models. In contrast, our approaches tackle semantic changes of objects and their relationships in natural scenes, achieved through scene graphs. Recently, Su et al. [221] proposed an improvement to SIMSG by utilizing masks instead of bounding boxes for object placement.

Images and scene graphs Scene graphs serve as structured, abstract representations of image content, capturing objects, their attributes, and relationships. Initially defined by Johnson et al. [105], a scene graph is represented as a directed graph, where objects constitute the nodes, and their interactions are expressed by the edges.

Numerous methods have been proposed to generate scene graphs from images [78, 139, 140, 166, 180, 223, 229, 259, 263, 277], and more recently, even from point clouds [244, 254]. The main goal is to identify objects and their visual relationships within the scene. Diverse approaches have been explored for this task, including iterative message-passing [259], sub-graph decomposition [139], and attention mechanisms [263]. SceneGraphGen [63] has recently addressed this task unconditionally by employing an auto-regressive model for scene graph generation.

Scene graph generation inherently depends on successful object detection [189] and visual relationship detection [36, 93, 153, 198, 272], which enable the identification of visual entities and their interactions in the image.

Conversely, the reverse and more challenging problem involves generating an image from its scene graph. Johnson et al. tackled this challenge using a graph convolution network (GCN) to decode the scene graph into a layout and subsequently translate it into an image [104]. Scene graphs have also been employed effectively for conditional scene generation [40, 104, 154].

We extend this architecture and introduce additional mechanisms for information transfer from an image to serve as conditioning when the objective is image editing, rather than free-form generation. Related works have explored image generation directly from layouts [285]. Recent research focuses on interactive image generation from scene graphs [4] or layouts [224]. However, these methods differ from ours in two main aspects. First, while [4, 224] process a graph/layout to generate multiple variants of an image, our approach manipulates an *existing* image. Second, we address complex semantic relationship editing, while they use graphs with

simplified spatial relations, such as `left of` or above in [4], or without relations at all, as in the layout-only approach of [224].

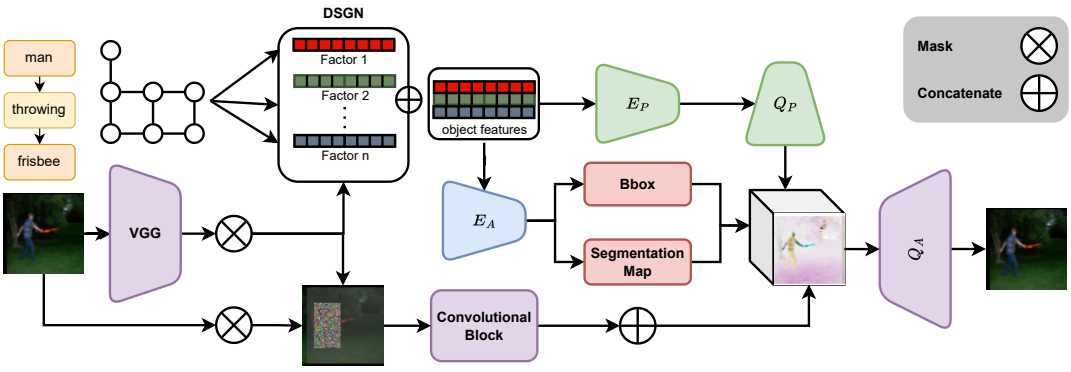


Fig. 12.4. **DisPositioNet Overview.** Given a source image and its corresponding scene graph, we obtain a set of node features. The node features along with the edge features are passed through the Disentangled GNN to produce the object features. The disentangled object features are then fed to two branches, i.e., the pose encoder E_P and the appearance encoder E_A. The scene layout is computed by embedding object features at the spatial locations from predicted object bounding boxes. Finally, the predicted transformation by the pose decoder Q_P is applied to feature patches of the layout for each object, and the final image is generated by passing the transformed layout to the decoder network Q_A.

Disentangled Representation Learning Disentangled representation learning has been extensively studied using variational autoencoders [46, 230], as demonstrated in tasks like changing digits and handwriting in the MNIST dataset. Early works on disentangled representation learning [26] focused on maximizing variational mutual information or reducing the channel capacity of the variational autoencoder [80]. More recent approaches explore disentanglement using deformable networks [213, 258], contrastive learning [7, 175], or disentangling identity and pose for face manipulation [278]. However, many of these methods require labeled information to condition the model on specific attributes for disentangling. Moreover, they often disentangle the data without explicitly identifying which feature corresponds to which factor. Recently, an unsupervised approach called [218] was proposed to disentangle pose and appearance. It employed two branches to predict the image's transformation parameters and then applied the learned transformation to the appearance features.

Disentanglement in graph neural networks (GNN) has also been explored in previous works such as [155, 264], where the graph features are decomposed into different factors to aid in disentangling the latent representation. However, these approaches primarily operate on regular graphs, considering only neighboring nodes when computing features. In contrast, GNNs designed for scene graphs incorporate the modulation of edge features into the GNN network, providing a more comprehensive representation.

12.3 Method

The main focus of this chapter is to perform semantic manipulation of images without direct supervision for image edits, meaning that we do not require paired data consisting of original and modified content. The process starts with an input image I from which we generate its

scene graph $\mathcal{G}$ that serves as the interaction interface with a user. Subsequently, a new image $\tilde{I}$ is generated based on the user-modified graph representation $\tilde{\mathcal{G}}$ and the original content of I. An overview of the method is depicted in Figure 12.1.

Our method consists of three interconnected parts. The first step is scene graph generation, where we encode the image contents into a spatio-semantic scene graph that is designed to be easily manipulated by a user. During inference, the user is able to modify the scene graph by making changes to object categories, locations, or relations, directly interacting with the nodes and edges of the graph. Finally, the output image is generated based on the modified graph. The three components and their connections are illustrated in Figure 12.3. We extend this methodology by disentangling the pose and identity features in the latent space of the semantic image manipulation model.

An inherent challenge in this problem lies in obtaining training data, as it requires matching pairs of source and target images along with their corresponding scene graphs. To address these limitations, we propose a method that learns the task through unsupervised image reconstruction, leveraging readily available training data. In contrast, graph prediction is learned with full supervision due to the availability of labeled data.

12.3.1 Definitions

We are given an input image I and its corresponding scene graph $\mathcal{G} = \mathcal{O}, \mathcal{R}$, where $\mathcal{O}$ represents the set of objects (nodes) in the scene and $\mathcal{R}$ denotes the set of relationships (edges) between the objects, our objective is to obtain a modified image $\tilde{I}$ based on an altered version of the scene graph $\tilde{\mathcal{G}}$. To achieve this, we formulate the image manipulation task with a reconstruction proxy objective, eliminating the need for paired images with explicit changes for training.

For fine-grained control over specific object attributes, we extend the semantic graph representation to obtain an augmented graph. Each node in this graph contains a semantic class embedding, a bounding box x, and a neural visual feature z_I. During training, we randomly mask object regions in the image, visual features, or bounding boxes using a noise vector, and the model's objective is to reconstruct the masked parts using the information from the scene graph $\mathcal{G}$ and the remaining regions in the image.

In mathematical terms, $\mathcal{G}$ defines a set of triplets $\mathcal{G}_i = (s_i, r_i, o_i)$, where s_i, r_i, and o_i represent the subject, predicate, and object, respectively. Each object in the graph belongs to a class of object categories $\mathcal{C} = c_1, c_2, ..., c_n$. We extract image features z_I from the input image using a pre-trained classifier network, such as VGG16 [216]. The graph triplets are then passed through a scene graph neural network (SGN) with parameters Φ, facilitating message passing between the nodes. Consequently, $z_\mathcal{G}$ is obtained from SGN, which processes the scene graph $\mathcal{G}$, the input bounding boxes x, and the visual features z_I.

12.3.2 Semantic Image Manipulation using Scene Graphs

Scene Graph Generation

The problem of generating a scene graph from an image has been extensively studied [139, 166, 259, 277]. It involves representing the image with a directed graph $\mathcal{G} = (\mathcal{O}, \mathcal{R})$ consisting of objects $\mathcal{O}$ (nodes) and their relations $\mathcal{R}$ (edges).

To achieve this, we adopt a state-of-the-art scene graph prediction method (F-Net) [139] and use its output as the foundation for our work. Since the desired output of our system is a generated image, we aim to encode as much image information as possible into the scene graph, in addition to semantic relationships.

We define each pair of objects and the relationship between them as a triplet $o_i = (c_i, z_{I,i}, x_i) \in \mathcal{O}$, where $c_i \in \mathbb{R}^d$ is a d-dimensional learned embedding of the i-th object category, and $x_i \in \mathbb{R}^4$ represents the four values defining the object's bounding box. The visual feature encoding of the object is denoted as $z_{I,i} \in \mathbb{R}^n$, which can be obtained from a pre-trained convolutional neural network (CNN) used for image classification. Similarly, for each relationship between two objects i and j, we learn an embedding ρ_{ij} of the relation class $r_{ij} \in \mathcal{R}$.

Conceptually, one can view our graph representation as an augmentation of a simple graph containing only object and predicate categories, enriched with image features and spatial locations. By incorporating this additional information into the graph, our model ensures the preservation of object identity and appearance even when corresponding locations and/or relationships are modified.

Graph Feature Extraction

The core of our approach centers around the spatio-semantic scene graph network (SGN), which plays a pivotal role in handling the user modified graph. The SGN is designed to learn a graph transformation that facilitates the smooth flow of information between objects and their relationships. Its primary objective is to acquire resilient object representations that will be employed in the image reconstruction process. To achieve this, the SGN conducts a sequence of convolutional operations on the graph structure.

These graph convolutions are executed through an operation denoted as τ_e, which primarily operates on the edges of the graph:

$$(\alpha_{ij}^{(t+1)}, \rho_{ij}^{(t+1)}, \beta_{ij}^{(t+1)}) = \tau_e \left(\nu_i^{(t)}, \rho_{ij}^{(t)}, \nu_j^{(t)} \right), \tag{12.1}$$

where we define $\nu_i^{(0)}$ as o_i, and t denotes the layer of the SGN, and τ_e is realized as a multi-layer perceptron (MLP). Given that nodes may appear in multiple edges, the updated node feature $\nu_i^{(t+1)}$ is obtained by averaging the outcomes of the edge-wise transformation. Subsequently, the averaged result undergoes another projection using τ_n.

$$\nu_i^{(t+1)} = \tau_n \left(\frac{1}{N_i} \left(\sum_{j|(i,j)\in\mathcal{R}} \alpha_{ij}^{(t+1)} + \sum_{k|(k,i)\in\mathcal{R}} \beta_{ki}^{(t+1)} \right) \right) \tag{12.2}$$

The number of edges that start or end in node i is denoted by N_i. After T graph convolutional layers, the final layer generates a latent representation for each node, i.e., per object. This output object representation includes predicted bounding box coordinates $\hat{x}_i \in \mathbb{R}^4$, a spatial binary mask $\hat{m}_i \in \mathbb{R}^{M \times M}$, and a node feature vector $\psi_i \in \mathbb{R}^s$.

The prediction of coordinates for each object serves as a form of reconstruction, since the object locations are known and already encoded in the input o_i. This reconstruction is necessary when modifying the graph, for example, when adding a new node.

The predicted object representation is then assembled into the spatial configuration of an image, forming the scene layout. This layout will be used to generate the modified image based on the user's changes to the scene graph.

Image Generation

Scene Layout The subsequent component of our method is responsible for converting the graph-structured representations predicted by the SGN into a 2D spatial arrangement of features, which can be subsequently decoded into an image.

To achieve this, we utilize the bounding box coordinates $\hat{x}_i$ predicted by the SGN to project the corresponding masks $\hat{m}_i$ into the appropriate regions of a 2D representation that matches the resolution of the input image. By concatenating the original visual feature $z_{I,i}$ with the node features ψ_i, we obtain the final node feature.

The projected mask regions are then filled with their respective features, while the remaining areas are padded with zeros. This process is repeated for all objects, resulting in $|\mathcal{O}|$ tensors of dimensions $(n+s) \times H \times W$. These tensors are subsequently aggregated through summation to form a single layout representing the scene, which contains enough information to reconstruct the image.

We then fill the projected mask region with the respective features, while the remaining area is padded with zeros. This process is repeated for all objects, resulting in $|\mathcal{O}|$ tensors of dimensions $(n + s) \times H \times W$, which are aggregated through summation into a single layout for the image. The output of this component is an intermediate representation of the scene, which is rich enough to reconstruct an image.

Image Decoding The final stage of our pipeline involves synthesizing the target image based on the information from the source image I and the layout prediction. To accomplish this task, we employ two different decoder architectures: cascaded refinement networks (CRN) [25] (similar to [104]), as well as SPADE [174], originally designed for image synthesis from a semantic segmentation map.

To condition the image synthesis process, we concatenate the predicted layout with the extracted low-level features from the source image. Before extracting these features, specific regions of I are masked, a mechanism that is further explained in section 12.3.2. These masked regions are filled with Gaussian noise, introducing stochasticity to the generator for more diverse output.

Training

Full supervision training would necessitate quadruplet annotations $(I, \mathcal{G}, \tilde{\mathcal{G}}, \tilde{I})$, where an image I is paired with its original scene graph $\mathcal{G}$, a modified graph $\tilde{\mathcal{G}}$, and the resulting modified image $\tilde{I}$. However, obtaining ground truth data for $(\tilde{I}, \tilde{\mathcal{G}})$ is challenging. To address this, we train our model using only the $(I, \mathcal{G})$ pairs through a reconstruction-based approach. We generate annotation quadruplets $(\tilde{I}, \tilde{\mathcal{G}}, \mathcal{G}, I)$, where $(I, \mathcal{G})$ serves as the *target* supervision and $(\tilde{I}, \tilde{\mathcal{G}})$ is simulated using a random masking procedure applied to object instances.

When training, we randomly mask an object's visual features $z_{I,i}$ with probability p_{z_I} and independently mask the bounding box x_i with probability p_x. When information is "hidden," the corresponding image regions are occluded before feature extraction. This unsupervised approach enables learning without the need for paired modified images, making the training process more feasible and efficient.

The masking mechanism employed effectively transforms the image editing task into a reconstruction proxy task. During runtime, a user can directly edit the nodes or edges of the scene graph. When an edit is made, the regions in the image that are subject to modification are occluded. The network, having learned to reconstruct the image from the scene graph, then generates a plausible modified image based on the edited graph.

For instance, consider the example of a person riding a horse (Figure 12.1). Suppose the user desires to change the interaction between these entities, modifying the predicate from `riding` to `beside`. As the spatial arrangement is expected to alter, we discard the localization x_i of these entities in the original image. Instead, their new positions $\hat{x}_i$ will be estimated based on the layout of the rest of the scene (e.g., grass, trees). To facilitate this change, the system automatically masks the original image regions corresponding to the target objects. However, to preserve the *visual identities* of the horse and rider throughout the modification, their visual feature encodings $z_{I,i}$ must remain unchanged. This ensures that the appearance of the horse and rider is consistent even after the interaction has been altered.

To train the model effectively, we employ a combination of loss terms. First, the bounding box prediction is trained using the L_1-norm, represented as $\mathcal{L}_b = \|x_i - \hat{x}_i\|_1^1$, with a weighting term λ_b.

For the image synthesis task, adversarial training is utilized with two discriminators. The local discriminator D_{obj} works on each reconstructed region ensuring that the generated patches appear realistic. Additionally, we apply an auxiliary classifier loss [170] to enable D_{obj} to correctly classify the generated objects into their respective real labels. On the other hand, the global discriminator D_{global} ensures overall consistency throughout the entire image.

To preserve the image content in regions that remain unchanged, we apply an image reconstruction loss term $\mathcal{L}_r = |I - \tilde{I}|_1$. This term enforces that regions not subject to modification maintain their original appearance.

The total image generation loss is then a combination of all these individual loss terms:

$$\begin{aligned}\mathcal{L}_{\text{synthesis}} = \mathcal{L}_r + \lambda_g \min_G \max_D \mathcal{L}_{\text{GAN,global}} \\ + \lambda_o \min_G \max_D \mathcal{L}_{\text{GAN,obj}} + \lambda_a \mathcal{L}_{\text{aux,obj}},\end{aligned} \tag{12.3}$$

where λ_g, λ_o, λ_a are weighting terms.

$$\mathcal{L}_{\text{GAN}} = \mathop{\mathbb{E}}_{q \sim p_{\text{real}}} \log D(q) + \mathop{\mathbb{E}}_{q \sim p_{\text{fake}}} \log(1 - D(q)), \tag{12.4}$$

We denote the ground truth distribution of the image or its objects by p_{real} and the distribution of the generated or edited ones by p_{fake}. The discriminator input q is sampled from either p_{real} or p_{fake}. We use SPADE with a perceptual loss $\lambda_p \mathcal{L}_p$ and a GAN feature loss $\lambda_f \mathcal{L}_f$ as in [174]. We also use a multi-scale discriminator D_{global}. See the Appendix for more details on the architectures, hyper-parameters and training.

12.3.3 Disentangled Semantic Image Manipulation

We aim to learn a representation that disentangles the appearance and pose of the objects in the latent space for semantic image manipulation, which allows us to preserve specific attributes' features. In this section, we explain our disentangled graph model and our variational disentanglement approach. Figure 12.4 illustrates our method.

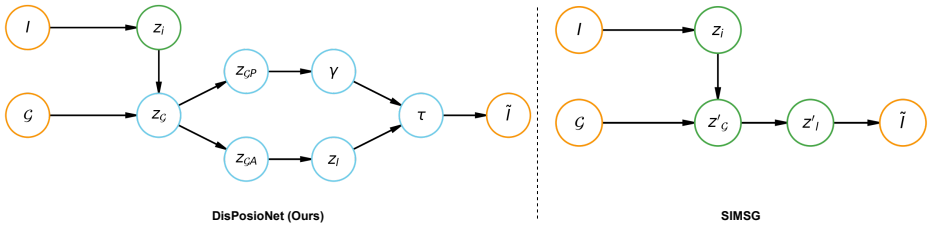


Fig. 12.5. A graphical model of DisPositioNet compared to SIMSG. Both models take an image I and a scene graph $\mathcal{G}$ as inputs. The blue color highlights the differences between the two methods. DispositioNet disentangles the graph representation and the latent embeddings, while SIMSG directly generates the image from the latent embedding z_l.

We use two encoder networks, E_A and E_P, to disentangle the pose and appearance features of the per-object features $z_{\mathcal{G}}$. The appearance features $z_{\mathcal{G}A}$ are used to predict the object bounding boxes and pseudo-segmentation maps with two networks. The pose features $z_{\mathcal{G}P}$ are used to predict a set of transformation parameters γ with a pose decoder network Q_P. These parameters define a transformation function τ that modifies the object features in the scene layout z_l. The scene layout z_l is obtained by projecting the appearance features $z_{\mathcal{G}A}$ of each object onto the image space, using the predicted bounding boxes and segmentation maps. The object features are cropped from z_l using the bounding boxes x and transformed by τ. The image decoder network Q_A takes the transformed layout $\tau(z_l)$ and produces the reconstructed image $\tilde{I}$.

Graphical Model Figure 12.5 shows the graphical model of DisPositioNet and SIMSG for comparison.

Disentangled Graph Neural Network The SIMSG formulation has a major drawback: the object features from the SGN are not separated. To enhance the pose and appearance disentanglement, we propose a disentangled graph neural network, called *DSGN*. Our proposed DSGN is inspired by [155], but it also incorporates the edge features (here predicates) in the disentangled feature extraction, as they are essential for our task. Our network handles triplets of the form $\mathcal{G}_i = (s_i, r_i, o_i)$. The DSGN uses disentangled convolutional layers and neighborhood routing mechanism [197] to project the features into different subspaces. The neighborhood routing mechanism identifies the latent factor that causes the edge between a node and its neighbors. It assigns the neighbor to a different channel to extract the features for that factor. The DSGN takes the triplets $\mathcal{G}_i = (s_i, r_i, o_i)$ as input, where $o_i, s_i \in \mathcal{O}$ and $r_i \in \mathcal{R}$. Every layer e in the DSGN is a function $f_e(.)$ that applies the edge features to the nodes in the graph and their neighbours:

$$(\alpha_{ij}^{(t+1)}, \rho_{ij}^{(t+1)}, \beta_{ij}^{(t+1)}) = f_e\left(\nu_i^{(t)}, \rho_{ij}^{(t)}, \nu_j^{(t)}\right), \tag{12.5}$$

where $\nu_i^{(0)} = o_i$, and t is the layer index of the DSGN, with T layers in total. The node features $\nu_i^{(t)}$ are first split into k factors by a Sparse Input Layer [57]. Then, each factor is processed by a separate neighbourhood routing layer. The final object features $\nu_i^{(T)}$ are obtained by concatenating the object features from all k factors.

Disentangled Variational Embedding We use two encoder and two decoder networks to force our model to disentangle the pose and appearance features in the latent embedding, following [218]. The object features $z_{\mathcal{G}}$ are fed to the variational encoders E_A, E_P, which have two subnetworks that estimate the mean μ and variance σ of the data. They both produce the latent representation z, using the reparameterization trick $z = \mu + \sigma\epsilon$, where ϵ is a random noise vector.

E_A extracts the appearance features, while E_P extracts the pose information. A simple MLP network Q_P predicts the transformation γ for each object, which is applied to the object patches cropped from the scene layout z_l. The purpose of predicting and applying the transformation γ by Q_P is to isolate the pose information in the pose branch and make the model learn only the appearance features in E_A. The image is then reconstructed by conditioning the image decoder Q_A on the scene layout z_l. In our experiments, we use the the SPADE [174] generator as our image decoder.

The affine transformation function τ for the input z is defined as follows:

$$\tau_\gamma(z) = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix} \begin{bmatrix} z_a \\ z_b \\ 1 \end{bmatrix} \tag{12.6}$$

which is then derived as:

$$\tau_{\gamma,\text{affine}}(z) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} 1 & m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_{z_a} & 0 \\ 0 & \delta_{z_b} \end{bmatrix} + \begin{bmatrix} t_{z_a} \\ t_{z_b} \end{bmatrix} \tag{12.7}$$

The MLP Q_P models the parameters γ for each object, which define the following transformations: rotation by angle α, shear by factor m, scaling by factors δ_{z_a} and δ_{z_b}, and translation by distances t_{z_a} and t_{z_b}.

Training The objective functions utilized for training the DisPositioNet model are a combination of original losses used in SIMSG [42] and variational terms. The generative adversarial loss is defined as:

$$\mathcal{L}_{\text{GAN}} = \mathop{\mathbb{E}}_{q\sim p_{\text{data}}} \log D(q) + \mathop{\mathbb{E}}_{q\sim p_{\text{g}}} \log(1 - D(q)), \tag{12.8}$$

The discriminator network D takes inputs q from either the ground truth data distribution p_{data} or the fake data distribution p_g, which consists of generated images or object patches. The model uses two discriminators: a global image discriminator D_{image} and an object discriminator D_{obj} that operates on cropped object patches to enhance their quality and realism. The model also predicts the bounding box coordinates x_i for each object and minimizes the L1 loss $\mathcal{L}_{bbox} = \lambda_b \|x_i - \hat{x}_i\|_1^1$ with respect to the ground truth coordinates $\hat{x}_i$. The generative objective is:

$$\begin{aligned} \mathcal{L}_{\text{generative}} = \lambda_g \min_G \max_D \mathcal{L}_{\text{GAN,image}} + \lambda_o \min_G \max_D \mathcal{L}_{\text{GAN,obj}} \\ + \lambda_a \mathcal{L}_{\text{aux,obj}} + \mathcal{L}_{rec} + \lambda_p \mathcal{L}_p + \lambda_f \mathcal{L}_f, \end{aligned} \tag{12.9}$$

The model uses constant weights λ_g, λ_o, λ_a to balance the different loss terms. The object classifier loss $\mathcal{L}_{\text{aux,obj}}$ [170] is an auxiliary term to encourage object-level discrimination. The perceptual loss $\mathcal{L}_p$ and the GAN feature loss $\mathcal{L}_f$ are adopted from the SPADE generator [174] to capture high-level semantic and style information. The image reconstruction loss $L_{rec} = \|I - \tilde{I}\|_1$ measures the pixel-wise difference between the input and output images.

The model also aims to disentangle the latent features by maximizing the evidence lower bound (ELBO):

$$\begin{aligned} \mathcal{L}_{var} = \mathbb{E}_{q_A,q_P}[log(p(I|z_{\mathcal{G}A}, z_{\mathcal{G}A}))] - D_{\mathbb{KL}}(q_P(z_{\mathcal{G}P}|I)||p(z_{\mathcal{G}P})) \\ - \mathbb{E}_{q_P}[D_{\mathbb{KL}}(q_A(z_{\mathcal{G}A}|I)||p(z_{\mathcal{G}A}))]. \end{aligned} \tag{12.10}$$

The final objective is then defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{var} + \mathcal{L}_{\text{generative}} + \mathcal{L}_{bbox} \tag{12.11}$$

12.4 Experiments and Results

This section presents our experimental setup, results, and discussion. We first describe the datasets, hyperparameters and the metrics used in our framework. Then we analyze the effects of our model components and compare our methods with previous work both quantitatively and qualitatively. We also conduct a user study to evaluate our methods. Finally, we discuss the implications and the limitations of our work.

12.4.1 Experimental Setup

We test our method on three datasets, CLEVR [102] Visual Genome [121], and Microsoft COCO [145], with different purposes. CLEVR is a synthetic dataset that provides ground truth image pairs for editing, which enables quantitative evaluation of our method. Visual Genome (VG) and COCO are real datasets that pose more challenges and diversity for image editing. Since VG and COCO do not have ground truth image pairs, we use an image inpainting proxy task and compare our method with a baseline based on sg2im [104]. We use standard image reconstruction metrics: the structural similarity index (SSIM), mean absolute error (MAE) and perceptual error (LPIPS) [281]. To measure the image generation quality and diversity, we use the inception score (IS) [201] and the FID [79] metric.

Datasets

CLEVR [102]. To be able to analyze paired image editing setting in a supervised manner, we create 21,310 image pairs that show the same scene with a specific modification, such as swapping positions, adding, removing or changing the attributes of the objects. We use 80% of the image pairs for training, 10% for validation and 10% for testing. The images have a size of $128 \times 128 \times 3$ and include n random objects ($3 \leq n \leq 7$) with random colors and shapes. The dataset does not have graph annotations, so we use the relative positions `in front of`, `behind`, `left of`, `right of` of different object pairs as predicates. The dataset also provides annotated information of scene graphs, bounding boxes, object classes and object attributes.

Visual Genome (VG) [121]. We follow the splits of [104] and use 80%, 10% and 10% of the VG v1.4 dataset for training, validation and testing, respectively. After applying the pre-processing of [104], the dataset has 178 object categories and 45 relationship types. The processed dataset consists of 62,565 training, 5,506 validation, and 5,088 testing images with graph annotations. We test our models with ground truth (GT) scene graphs on all the testing images. For the experiments with predicted (P) scene graphs, some images are filtered out (*e.g.* no objects are detected), so we test on 3874 images from the testing set. We notice that some relationships are duplicated in the dataset, which does not affect the image generation task, but causes confusion when editing only one of the duplicate edges (when using GT graphs). Therefore, we remove such duplicates when one of them is edited.

Microsoft COCO [145] We follow a similar preprocessing to [104] for the COCO dataset. The dataset includes 40,000 train and 5000 validation images with bounding boxes and segmentation masks for 171 categories. We use the ground truth bounding box and object

category information to construct geometric scene graphs. The six relationship categories consists of (*left of, right of, above, below, inside, and surrounding*).

Baselines

Conditional SG2Im Baseline (Cond-SG2Im). We adapt the SG2IM model of [104] as a baseline. Their method generates images from scene graphs without using a source image, so we condition their image synthesis network on the input image by concatenating it with the layout component (instead of noise as in the original work). To make it fair to our approach, we mask the image regions that correspond to the target objects before concatenation.

Interactive Scene Generation (ISG) [4] We compare our models againt the interactive scene generation (ISG) model from Ashual et al. Since the ISG model is not originally designed for image editing, we evaluate it in the fully generative setting where the whole image region is conditioned on a scene graph.

Modification types.

We introduce a new task of image editing using scene graphs, and we define several modification modes, depending on the user's interaction with the graph. In the testing phase, we support four modification modes: relationship change, object replacement, object removal, and object addition. The model takes the source image and the desired modification on the graph as input. It masks specific features based on the modification mode and generates the target image by the decoder. For example, for relationship changes, it keeps the object features but masks the bounding box features. For object replacement, it drops the object features but keeps the bounding box features. We explain each modification mode in detail below:

Object removal A node and its connected edges are deleted from the graph. The source image region that belongs to the object is masked.

Object replacement A node is changed to a different semantic category. We do not delete the whole node; however, we set the visual encoding $z_{I,i}$ of the original object to zeros, as it does not match the new object. The location of the original entity is used to place the new object, while the size comes from the bounding box predicted by the SGN, to fit the new category.

Relationship change This operation usually involves changing the positions of entities. The goal is to preserve the subject and object but alter their interaction, *e.g.* `<sitting>` to `<standing>`. Both the original and new appearance image regions are occluded, to allow background in-painting and target object generation. The visual encodings $z_{I,i}$ are used to condition the SGN and retain the visual identities of objects on re-appearance.

Object addition: A new node is inserted in the graph, with masked x_i and $z_{I,i}$.

Object Addition The node location is predicted by the model from the target scene graph. For the visual features, we query objects from the same category for the object to be added.

Evaluation Metrics

We use common GAN metrics such as Inception Score (IS) [201], Frechet Inception Distance (FID) [79], structural similarity metric (SSIM) [251], Perceptual Similarity (LPIPS) [281] and the Mean Absolute Error (MAE) to evaluate the quality of our generated images. We also compute the MAE and SSIM for the Region of Interest (RoI) where the modification or reconstruction occurs.

Implementation Details

We train all models on 64×64 images with a batch size of 32. We use a VGG-16 pretrained on ImageNet for visual feature extraction. The learning rate for all models is $2e-4$, and the disentangling factor k in the Disentangled SGN is set to 16. We train all models for $300k$ iterations on VG and COCO. The networks E_P, E_A, and D_P are MLPs with two FC layers with 64 filters, 1 BN layer, and a LeakyReLU activation function. The decoder and discriminator networks in SIMSG and DisPositioNet have the same architecture. We choose the hyperparameter values empirically or based on previous works. We provide the details of network architectures in the appendix.

12.4.2 Results

In this section, we present the results on both synthetic and real data.

Synthetic Data

We use the CLEVR framework [102] to create a dataset (see the Appendix for details) of image and scene graph editing pairs $(I, \mathcal{G}, \tilde{\mathcal{G}}, \tilde{I})$, to test our method with exact ground truth.

We train our model *without using image pairs* and compare it with a fully-supervised setting. In the fully-supervised setting, the model receives the complete source image and target graph and is trained by minimizing the $\mathcal{L}_1$ loss to the ground truth target image instead of using the proposed masking scheme.

Table 12.1 shows the mean SSIM, MAE, LPIPS and FID on CLEVR for the manipulation task (replacement, removal, relationship change and addition). Our method outperforms or matches the fully-supervised setting on the reconstruction metrics, which demonstrates the ability of generating meaningful changes. The FID results indicate that additional supervision for pairs, if available, would improve the visual quality. Figure 12.6 presents qualitative results of our model on CLEVR. At test time, we make changes to the scene graph in four different modes: changing relationships (a), removing an object (b), adding an object (d) or changing its identity (c). We mark the modification with a bounding box around the selected object.

Ablation study on CLEVR We report additional results on CLEVR for the image reconstruction and manipulation tasks in Table 12.2 and Table 12.4. We find that our method with a SPADE decoder performs better than the other models in the reconstruction setting. For the manipulation modes, our method excels for relationship changes, while the performance for other changes is comparable with the baseline.

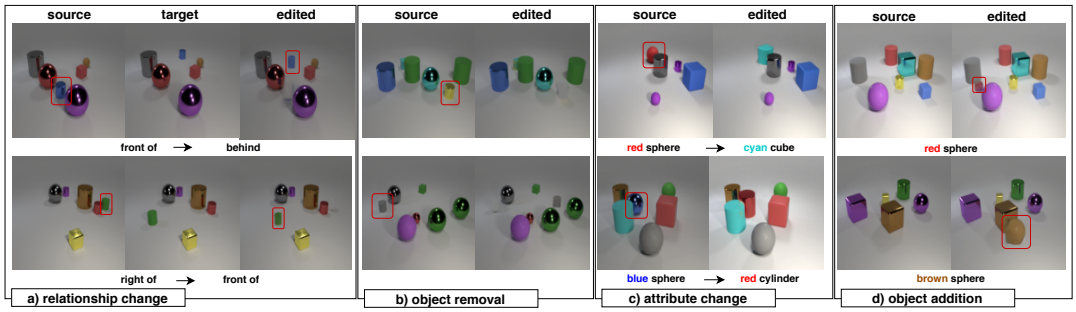


Fig. 12.6. **Image manipulation on CLEVR** We demonstrate different changes in the scene such as changing the relationship between two objects, removing a node or changing a node (corresponding to attribute changing).

Method	All pixels				RoI only	
	MAE ↓	SSIM ↑	LPIPS ↓	FID ↓	MAE ↓	SSIM ↑
Fully Supervised (CRN)	6.75	**97.07**	0.035	**3.35**	9.34	93.49
SIMSG (CRN)	7.83	96.16	0.036	6.32	10.09	93.54
SIMSG (SPADE)	**5.47**	96.51	0.035	4.73	**7.22**	**94.98**

Tab. 12.1. **Image manipulation on CLEVR.** We compare SIMSG to a fully-supervised baseline.

Real Images

We test our method on Visual Genome [121] and COCO [145] to demonstrate its performance on realistic images.

Feature Encoding We measure the role of the visual feature $z_{I,i}$ in encoding visual appearance. For a given image and its graph, we condition the SGN on all the object locations x_i and visual features (w/ $z_{I,i}$). However, we mask the region of the conditioning image that belongs to a candidate node. The task can be seen as conditional in-painting. We test our approach in two scenarios; using ground truth graphs (GT) and graphs predicted from the input images (P). We evaluate on all objects in the test set and report the results in Table 12.5, measuring the reconstruction error a) over all pixels and b) in the target area only (RoI). We compare with the same model without using visual features (w/o $z_{I,i}$) but only the object category to condition the SGN. As expected, in all cases, including the missing region's visual features improves the reconstruction metrics (MAE, SSIM, LPIPS). However, inception score and FID remain similar, as these metrics do not consider similarity between direct corresponding pairs of generated and ground truth images. From Table 12.5 we can observe that while both decoders perform similarly in reconstruction metrics (CRN is slightly better), SPADE excels for the FID and inception score, indicating higher visual quality.

Quantitative Results We use the reconstruction quality to evaluate our methods on a real-world dataset, since there is no paired source and target data for image manipulation. The input images are partially masked, and the model has to reconstruct the masked regions using the scene graph. We report the results of our experiments in Table 12.5 and Table 12.6. In the generative mode, the entire region of the image is masked, and the model generates the image only from the scene graph to test its image generation performance. We test our method in a

Method		Decoder	All pixels				RoI only	
			MAE ↓	SSIM ↑	LPIPS ↓	FID ↓	MAE ↓	SSIM ↑
Image Resolution			64×64					
Fully-supervised		CRN	6.74	97.07	0.035	5.34	9.34	93.49
SIMSG (GT)	w/o $z_{I,i}$	CRN	7.96	97.92	0.016	4.52	14.36	81.75
SIMSG (GT)	w/ $z_{I,i}$	CRN	6.15	98.50	0.008	3.73	10.47	88.53
SIMSG (GT)	w/o $z_{I,i}$	SPADE	4.25	98.79	0.009	3.75	9.67	87.13
SIMSG (GT)	w/ $z_{I,i}$	SPADE	2.73	99.35	0.002	3.42	5.42	94.16
Image Resolution			128×128					
Fully-supervised		CRN	9.83	97.36	0.061	4.42	12.38	91.94
SIMSG (GT)	w/o $z_{I,i}$	CRN	14.82	96.85	0.041	8.09	20.59	74.71
SIMSG (GT)	w/ $z_{I,i}$	CRN	14.47	96.93	0.038	8.36	19.56	75.25
SIMSG (GT)	w/o $z_{I,i}$	SPADE	9.26	98.27	0.029	3.21	15.74	79.81
SIMSG (GT)	w/ $z_{I,i}$	SPADE	5.39	99.18	0.007	1.17	8.32	89.84

Tab. 12.2. **Image reconstruction on CLEVR.** The results are reported using ground truth scene graphs (GT).

fully generative setting, where we occlude the whole image and only use the encoded features $z_{I,i}$ for each object. We compare with the state of the art in interactive scene generation (ISG) [4], tested in the same setting. Table 12.5 shows similar reconstruction errors for the generative task, while we outperform [4] when a source image is available. This supports our choice of manipulating an existing image directly, rather than merging different node features, as parts of the image need to be kept. Inception score and FID mostly depend on the decoder architecture, where SPADE performs better than Pix2pixHD and CRN.

The models use either ground truth graphs or predicted ones by a scene graph to image model [139] as input. The results indicate that SIMSG and DispositioNet models achieve the best performance in semantic image manipulation in all metrics and scenarios. We show that DispositioNet, which disentangles shape and pose, performs better than SIMSG. We also present the results of our user study on comparing SIMSG and DispositioNet for different manipulation modes in Table 12.7.

Qualitative Results Figure 12.7 shows qualitative examples of SIMSG compared to the baseline. Both SIMSG and the Cond-SG2Im baseline can generate realistic object categories and shapes. However, SIMSG can preserve visual features from the original image in the output. This property is especially useful when we want to move objects in the image without altering their identity.

We show some qualitative results of SIMSG and DisPositioNet on VG dataset in Figure 12.8. DisPositioNet learns better feature representations and generates more realistic results compared to the baselines and SIMSG.

Image Modification We present visual results in three different settings in Figure 12.9,—,object removal, replacement and relationship changes. The user makes all image modifications at test time, by editing nodes or edges in the graph. We demonstrate diverse replacements (a), from small objects to background elements. The new entity fits the image context, *e.g.* the ocean (second row) does not cover the person, which we would expect in standard image inpainting. A more difficult scenario is to change the interaction between two objects, which

Tab. 12.3. **Image manipulation on CLEVR.** The results are reported for different modifications categories on 64 × 64 images using GT graphs.

Method	Decoder	All pixels			RoI only	
		MAE ↓	SSIM ↑	LPIPS ↓	MAE ↓	SSIM ↑
Image Resolution		64 × 64				
Change Mode		Addition				
Fully-supervised	CRN	6.57	98.60	0.013	7.68	97.72
SIMSG w/ $z_{I,i}$	CRN	7.88	96.93	0.027	9.79	95.10
SIMSG w/ $z_{I,i}$	SPADE	4.96	97.45	0.026	6.13	96.86
Change Mode		Removal				
Fully-supervised	CRN	4.52	98.60	0.006	5.53	97.17
SIMSG w/ $z_{I,i}$	CRN	5.67	97.13	0.026	7.02	96.41
SIMSG w/ $z_{I,i}$	SPADE	3.45	97.32	0.022	3.88	98.09
Change Mode		Replacement				
Fully-supervised	CRN	6.64	97.76	0.015	7.33	97.11
SIMSG w/ $z_{I,i}$	CRN	8.24	96.96	0.025	9.29	96.02
SIMSG w/ $z_{I,i}$	SPADE	5.88	97.43	0.023	6.56	97.48
Change Mode		Relationship changing				
Fully-supervised	CRN	9.76	93.91	0.111	17.51	83.24
SIMSG w/ $z_{I,i}$	CRN	10.09	93.50	0.0678	14.91	86.17
SIMSG w/ $z_{I,i}$	SPADE	8.11	93.75	0.069	13.01	86.99

usually involves changing positions. Figure 12.9 (b) shows that the model can distinguish between semantic concepts, such as `sitting` vs. `standing` and `riding` vs. `next to`. The objects are repositioned accordingly to the change in relationship type. In the case of object removal (c), the method performs well for backgrounds with uniform texture, but can also handle more complex structures, such as the background in the first example. Interestingly, when the building on the rightmost example is removed, the remaining sign is improvised standing in the bush.

Relationship Changes Figure 12.10 shows how our method handles relationship changes in more detail. We examine how the bounding box placement and the image generation of an object are affected by changing one of its relationships. We contrast the results between auto-encoding mode and modification mode. The bounding box coordinates are masked in both modes so that the model can determine where to place the target object based on the relationships. In auto-encoding mode, the predicted boxes (red) match the original relationship, while in the modified setup, the predicted boxes follow the changed relationship, *e.g.* in auto mode, the person stays on the horse, while in modification mode the box moves next to the horse.

Comparison to ISG [4] Figure 12.11 shows qualitative samples of SIMSG and a comparison with [4] for the auto-encoding (a) and object removal task (b). We modify [4] for object removal by deleting a node and its connected edges from the input graph (same as in ours), while the visual features of the remaining nodes (from our source image) are used to reconstruct the rest of the image. We obtain similar results for the auto-encoding, even though our method is not specifically trained for the fully-generative task. For object removal, SIMSG

Tab. 12.4. **Image manipulation on CLEVR.** The results are reported for different categories of modifications on 128×128 images using GT graphs.

Method	Decoder	All pixels			RoI only	
		MAE ↓	SSIM ↑	LPIPS ↓	MAE ↓	SSIM ↑
Image Resolution		128×128				
Change Mode		Addition				
Fully-supervised	CRN	9.72	97.57	0.031	10.61	94.09
SIMSG w/ $z_{I,i}$	CRN	13.77	96.44	0.048	13.21	91.05
SIMSG w/ $z_{I,i}$	SPADE	7.79	97.89	0.040	7.57	96.18
Change Mode		Removal				
Fully-supervised	CRN	6.15	98.72	0.014	7.27	95.58
SIMSG w/ $z_{I,i}$	CRN	11.75	97.21	0.052	11.55	92.34
SIMSG w/ $z_{I,i}$	SPADE	4.48	98.54	0.042	4.60	97.68
Change Mode		Replacement				
Fully-supervised	CRN	10.49	97.57	0.035	11.23	95.09
SIMSG w/ $z_{I,i}$	CRN	16.38	96.14	0.052	14.74	91.98
SIMSG w/ $z_{I,i}$	SPADE	10.25	97.51	0.041	9.98	96.14
Change Mode		Relationship changing				
Fully-supervised	CRN	13.91	95.26	0.169	21.49	82.46
SIMSG w/ $z_{I,i}$	CRN	16.61	94.60	0.128	19.21	85.24
SIMSG w/ $z_{I,i}$	SPADE	11.62	95.76	0.125	14.01	89.15

Tab. 12.5. **Image reconstruction on Visual Genome.** We compare the results of our method to previous works using ground truth (GT) and predicted scene graphs. In the experiments denoted by (Generative), the whole input image is masked. N/A: Not Applicable.

Method	Decoder	All pixels					RoI only	
		MAE ↓	SSIM ↑	LPIPS ↓	FID ↓	IS ↑	MAE ↓	SSIM ↑
Generative, GT Graphs								
ISG [4]	Pix2pixHD	46.44	28.10	0.32	58.73	$6.64_{\pm 0.07}$	N/A	N/A
SIMSG (Ours) [42]	CRN	**41.57**	33.9	0.34	89.55	$6.03_{\pm 0.17}$	N/A	N/A
SIMSG (Ours) [42]	SPADE	41.88	34.89	0.27	44.27	$7.86_{\pm 0.49}$	N/A	N/A
DispositioNet (Ours)	SPADE	41.62	**35.30**	**0.26**	**40.75**	$\mathbf{7.93}_{\pm 0.36}$	N/A	N/A
GT Graphs								
Cond-SG2Im [104]	CRN	14.25	84.42	0.081	13.40	$11.14_{\pm 0.80}$	29.05	52.51
SIMSG (Ours) [42] w/o $z_{I,i}$	CRN	9.83	86.52	0.073	10.62	$11.45_{\pm 0.61}$	27.16	52.01
SIMSG (Ours) [42] w/ $z_{I,i}$	CRN	**7.43**	**88.29**	0.058	11.03	$11.22_{\pm 0.52}$	**20.37**	**60.03**
SIMSG (Ours) [42] w/o $z_{I,i}$	SPADE	10.36	86.67	0.069	8.09	$12.05_{\pm 0.80}$	27.10	54.38
SIMSG (Ours) [42] w/ $z_{I,i}$	SPADE	8.61	87.55	0.050	**7.54**	$\mathbf{12.07}_{\pm 0.97}$	**21.62**	58.51
DispositioNet (Ours) w/ $z_{I,i}$	SPADE	8.41	87.56	**0.048**	7.66	$11.65_{\pm 0.58}$	21.76	58.18
Predicted Graphs								
SIMSG (Ours) [42] w/o $z_{I,i}$	CRN	9.24	87.01	0.075	18.09	$10.67_{\pm 0.43}$	29.08	48.62
SIMSG (Ours) [42] w/ $z_{I,i}$	CRN	**7.62**	**88.31**	0.063	19.49	$10.18_{\pm 0.27}$	**22.89**	**55.07**
SIMSG (Ours) [42] w/o $z_{I,i}$	SPADE	13.16	84.61	0.083	16.12	$10.45_{\pm 0.15}$	32.24	47.25
SIMSG (Ours) [42] w/ $z_{I,i}$	SPADE	13.82	83.98	0.077	16.69	$10.61_{\pm 0.37}$	28.82	49.34
DispositioNet (Ours) w/ $z_{I,i}$	SPADE	9.39	86.91	**0.052**	**14.42**	$\mathbf{10.69}_{\pm 0.33}$	25.40	51.85

performs better in general, since it is designed for direct manipulation on an image. For a fair comparison, we train [4] on Visual Genome. Since we focus on semantically rich relations, we use their publicly available code to train [4] on Visual Genome. Since Visual Genome does not have segmentation masks, we disable the mask discriminator. Therefore, we expect lower

Fig. 12.7. **Visual feature encoding.** We compare the baseline (top) and our method (center). The scene graph is the same; an object in the image is masked, but $z_{I,i}$ and x_i are not. Our latent features $z_{I,i}$ maintain appearance when the objects are hidden from the image.

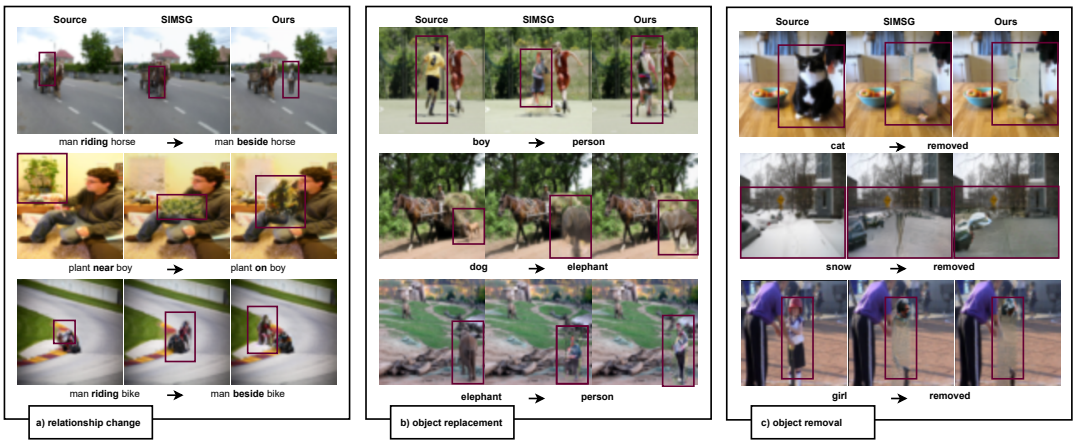


Fig. 12.8. **Qualitative comparison to SIMSG [42] on VG.** We can see that in a) the plant looks more realistic and has a more similar shape to the original object, and the same goes for b) where the boy and the elephant are replaced by person. For the object removal in c) there are some artifacts after removing the cat and snow, but our method does not have these artifacts and produces more realistic images.

quality results than in the original paper (trained on MS-COCO with mask supervision and simpler scene graphs).

Results on CoCo We show the qualitative results on the COCO [145] dataset in Figure 12.12. DispositioNet predicts the target bounding box for moving the object more accurately due to the disentanglement of pose and appearance. The modifications by DispositioNet have fewer artefacts and look more realistic than SIMSG.

User Study We conduct a user study to compare the qualitative results of DisPositioNet and SIMSG [42] for different image manipulation modes on the Visual Genome [121] dataset. The user study has 21 questions with images before and after the manipulation by the two methods. The users have to select the image that matches better with the specified change. The user study has 37 participants and the summary of their responses is given in the main paper. The order of the methods for each example in the study is randomized, to avoid bias.

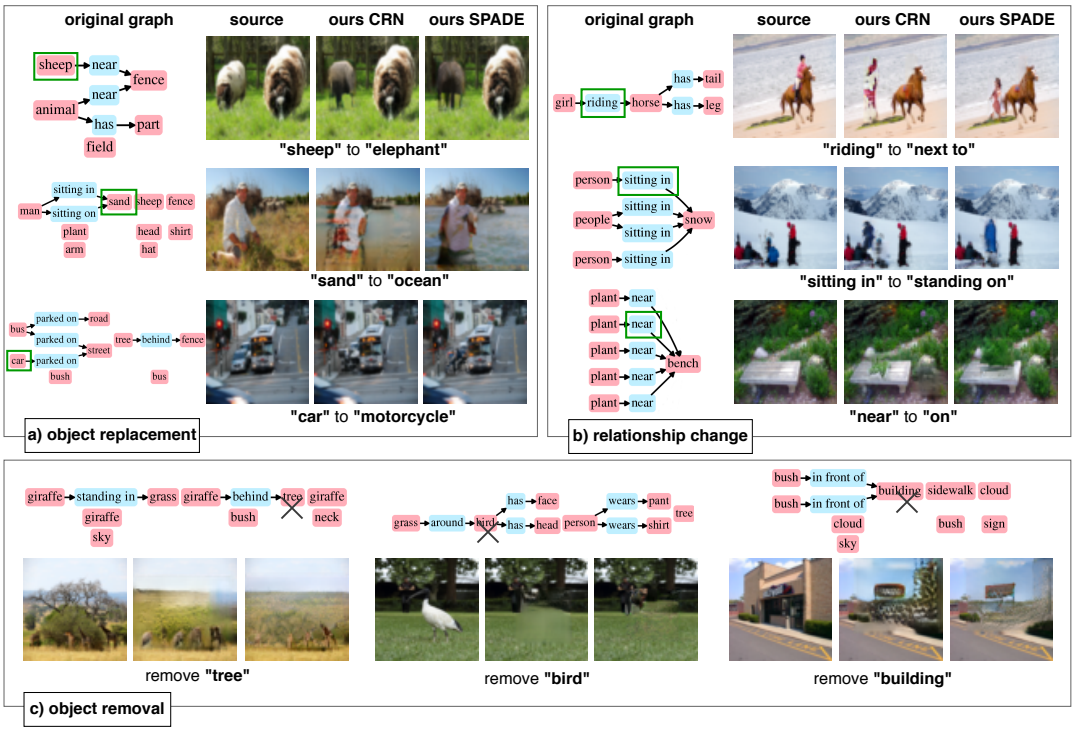


Fig. 12.9. **Image manipulation** We edit the image semantically by modifying the graph, given the source image and the GT scene graph. **a)** object replacement, **b)** relationship changes, **c)** object removal. Green box marks the edited node or edge.

The values reported in the user study table in the main text, show the percentage of images chosen as having higher quality and relevance to the specified change by the participants. The results indicate that our method outperforms SIMSG significantly in all manipulation modes.

Ablation Study

We report the results of our ablation study in Table 12.8 and Figure 12.13. In Figure 12.13 we qualitatively ablate the components of our method. For a given image, we occlude a certain object instance that we want to reconstruct. We test the method with all the possible combinations of occluding bounding boxes x_i and/or visual features $z_{I,i}$ from the augmented

Tab. 12.6. **Image reconstruction on COCO**

Method	All pixels			RoI only	
	MAE ↓	SSIM ↑	LPIPS ↓	MAE ↓	SSIM ↑
Generative					
SIMSG [42]	54.03	24.12	0.490	N/A	N/A
DispositioNet (Ours)	**51.07**	**26.53**	**0.418**	N/A	N/A
Non Generative					
SIMSG [42]	9.36	87.00	0.086	27.68	49.93
DispositioNet (Ours)	**9.24**	**88.26**	**0.057**	**27.52**	**50.35**

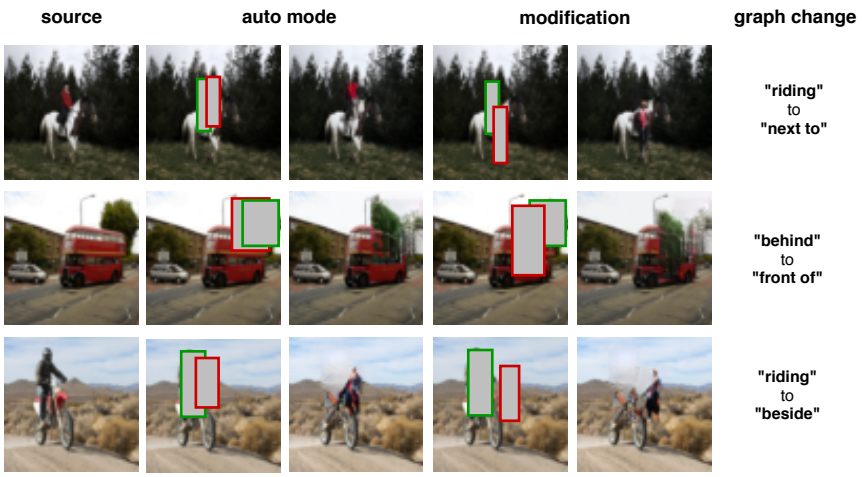


Fig. 12.10. **Detailed evaluation of re-positioning.** We mask the bounding box x_i of an object and generate a target image in two modes. We select a relationship that involves this object. In auto-mode (left), we keep the relationship unchanged. In modification mode, we change the relationship. Red: Predicted box for the original or modified relationship. Green: Ground truth bounding box for the original relationship.

Tab. 12.7. **User study on VG**

Method	Removal	Replacement	Relationship Change	Mean
SIMSG [42]	14.06	27.68	26.95	23.51
DispositioNet (Ours)	**85.94**	**72.32**	**73.04**	**76.49**

graph representation. Since we may want to in-paint the region with a different object (changing either the category or style), we also experiment with an additional setting, in which external visual features z_I are extracted from an image of the query object. Masking the box properties leads to a small shift in the location and size of the reconstructed object, while masking the object features can result in an object with a different identity than that in the original image.

In Table 12.8, we first show the model performance without the disentanglement. Then, we test the effect of disentangling the latent embeddings. Finally, we show the model performance with disentanglement in both latent embedding and the graph features. The disentanglement of both components improves most metrics.

Spatial Distribution of Predicates Figure 12.14 shows the heatmaps of the ground truth and predicted bounding box distributions for each predicate. For each triplet (*i.e.* subject - predicate - object) in the test set, we predict the subject and object bounding box coordinates $\hat{x}_i$. Then, we compute the relative distance between the object and subject centers for each triplet and group them by predicate category. The plot illustrates the spatial distribution of each predicate. We notice similar distributions, especially for the relationships that have clear spatial constraints, such as `wears`, `above`, `riding`, etc. This suggests that SIMSG can accurately localize new (predicted) objects with respect to existing objects in the scene.

Diversity Analysis DisPositioNet [50] generates more diverse images than SIMSG [42] by using a variational representation for the object features. This allows our model to sample

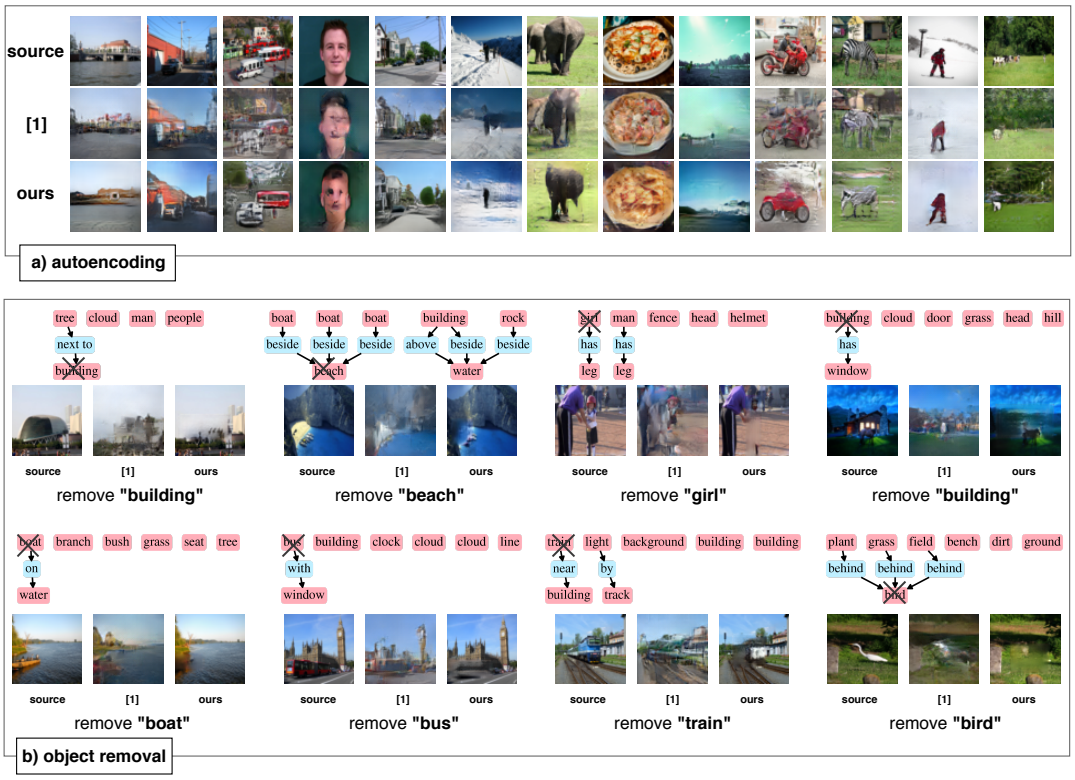


Fig. 12.11. **Qualitative results comparing SIMSG + CRN and [4] a)** Fully-generative setting **b)** Object removal

features probabilistically from the latent space. However, this also introduces some quality loss in the objects, as they may not match the realistic appearance of the scene. For instance, in Figure 12.15, our model generates sand that looks like grass, which is unrealistic. Therefore, our model is more suitable for scenarios where diversity is more important than realism.

12.4.3 Discussion

Our proposed models, SIMSG and its extended version DispositioNet, achieves state-of-the-art performance in image manipulation using semantic graphs. DispositioNet generates images with higher quality, more meaningful results, and more diversity than SIMSG. It also reduces artifacts and preserves the original image content. We conducted a user study to compare DispositioNet with SIMSG [42]. The users were asked to rate the models based on the image quality and the consistency between the graph and the image. DispositioNet was preferred over SIMSG in 76.49% of the cases. However, both methods still faces some challenges that are common in this domain, which we discuss in this section.

Limitations and Failure Cases Our approach has some limitations in manipulating high-resolution images and reconstructing faces and complex scenes. We think that these limitations come from the difficulty in generating high-quality images from scene graphs [104] that could be due to the wild nature of images in the VG dataset and occasional errors in the scene graph annotations. We expect that using a higher quality dataset with scene graphs and semantic segmentation annotations would help to overcome this issue. Regarding the face

Tab. 12.8. Ablation Study of Disenatanglement on VG

Model	Disentanglement		All pixels			RoI only	
	Embeddings	Graph	MAE ↓	SSIM ↑	LPIPS ↓	MAE ↓	SSIM ↑
Generative							
SIMSG	–	–	41.88	34.89	0.27	N/A	N/A
DisPositioNet	✓	–	41.80	35.18	0.26	N/A	N/A
DisPositioNet	✓	✓	**41.62**	**35.30**	**0.26**	N/A	N/A
GT Graphs							
SIMSG	–	–	8.61	87.55	0.050	**21.62**	**58.51**
DisPositioNet	✓	–	8.47	87.53	0.048	21.77	58.30
DisPositioNet	✓	✓	**8.41**	**87.56**	**0.048**	21.76	58.18
Predicted Graphs							
SIMSG	–	–	13.82	83.98	0.077	28.82	49.34
DisPositioNet	✓	–	9.65	86.68	0.054	25.62	51.19
DisPositioNet	✓	✓	**9.39**	**86.91**	**0.052**	**25.40**	**51.85**

reconstruction problem, although this is an easy task with datasets of pure face images, the model does not generalize well to the faces when mixed with images in the wild. We show some failure case examples below.

In the image manipulation task we propose, we have to restrict the feature encoding to prevent the encoder from "copying" the whole RoI, which is not desired if we want to change non-rigid objects, e.g. from `sitting` to `standing`. While the model can keep general appearance information such as colors and textures, it is true that some visual properties of modified objects are not recovered. For example, the color of the green object in Figure 12.16 a) is kept but not the material.

The model does not adjust unchanged areas of the image as a result of a change in the modified parts. For instance, shadows or reflections do not follow the moved objects, if they are not nodes of the graph and explicitly marked as changing by the user, Figure 12.16 b).

Moreover, like other methods tested on Visual Genome, the quality of some close objects is limited, *e.g.* close-up of people eating, Figure 12.16 c). Also, having a node `face` on animals often gives them a human face.

12.5 Conclusion

In this work, we have introduced a novel task, semantic image manipulation using scene graphs, and have proposed an innovative approach to address the learning problem without relying on paired training data. This unique method empowers users to directly interact with the nodes and edges of the scene graph, allowing for modifications to both the content and relationships among scene entities. We have demonstrated the competitiveness of our resulting system with existing image synthesis methods, and qualitative evaluations provide

compelling evidence of its ability to support real-world image modification. Moving forward, we plan to focus on further enhancing these capabilities and exploring potential applications in interactive editing and robotics.

An additional key contribution of our work is the novel disentangling framework for image manipulation using scene graphs. Our experiments have convincingly shown that leveraging disentangled representations in the latent embedding significantly enhances image generation and manipulation quality. Notably, the variational representation for object features enables the generation of diverse images, surpassing previous approaches. Furthermore, our use of a disentangled graph neural network for scene graph feature extraction results in more meaningful and useful features for the disentangled latent embedding, leading to superior reconstruction performance. As part of our future research direction, we aim to explore the potential of diffusion models to further improve the decoder network and extend the capabilities of our system.

Fig. 12.12. Qualitative results for image manipulation on COCO

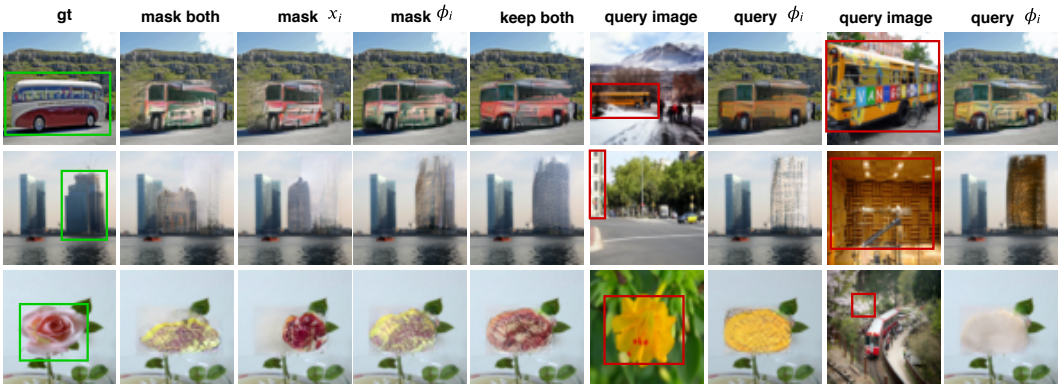


Fig. 12.13. **Ablation of the SIMSG components** We show all the different combinations in which the SIMSG operates - *i.e.* masked vs. active bounding boxes x_i and/or visual features $z_{I,i}$. When using a query image, we extract visual features of the object marked with a red bounding box and update the node of an object of the same category in the original image.

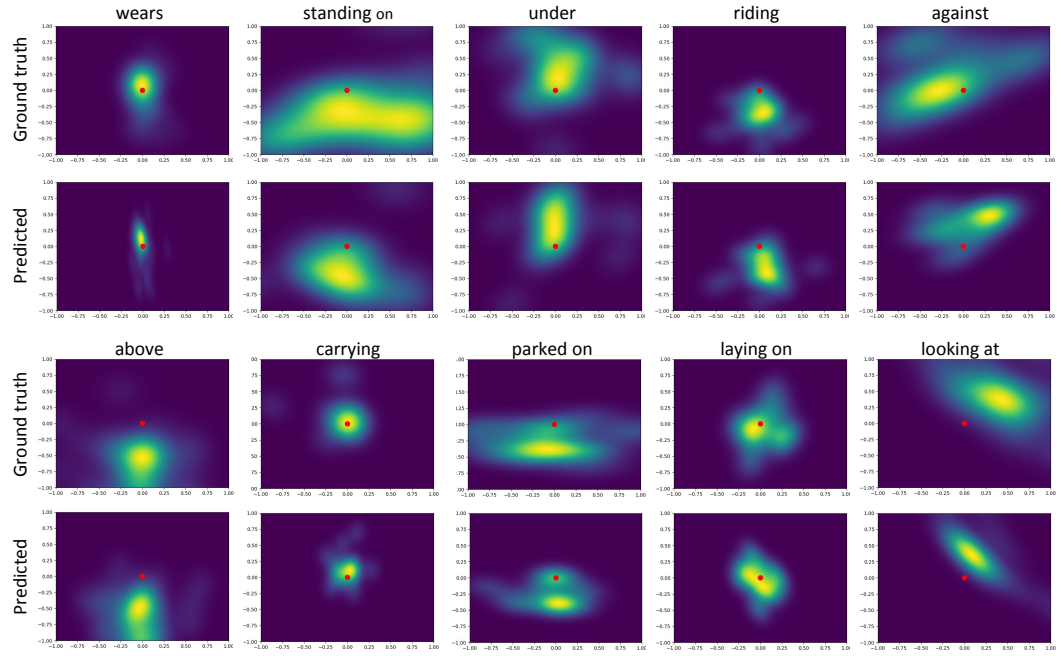


Fig. 12.14. **Heatmaps of object and subject relative positions for different predicates.** The object is located at the origin $(0, 0)$ and the subject's position is relative to the object. The heatmaps are based on the relative distances between the object and subject centers. Top: Ground truth boxes. Bottom: Our predicted boxes (without using location information from the graph and relying on synthesis).

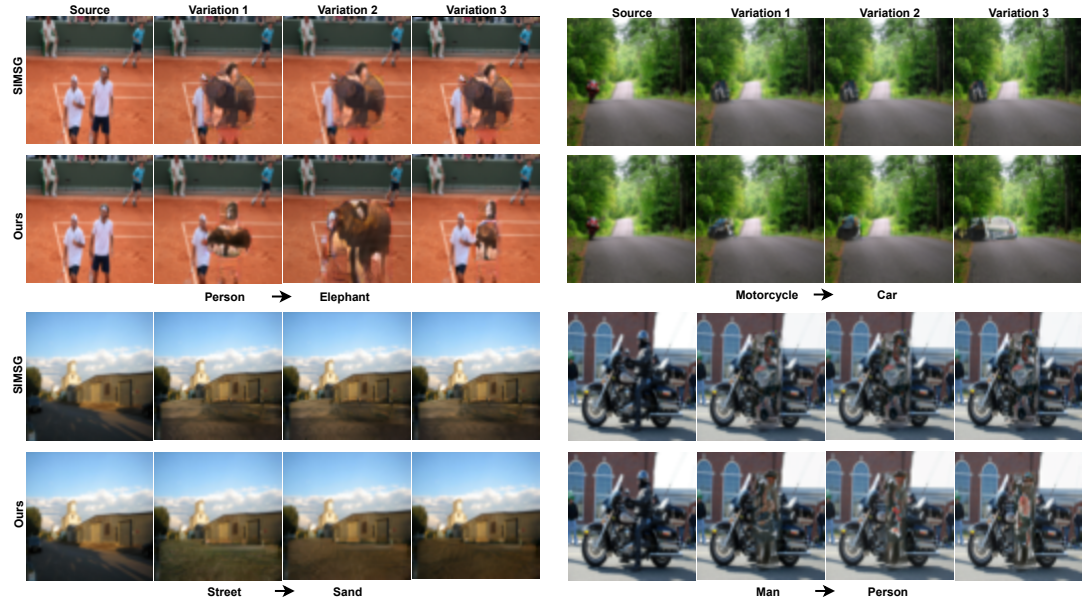


Fig. 12.15. **Diversity in image generation.** DisPositioNet leverages variational embedding to introduce diversity in the object features, which leads to diverse image generation. In contrast, SIMSG produces the same image repeatedly. As shown in the figure, DisPositioNet can generate objects with varying textures, sizes, and colors, while SIMSG cannot.

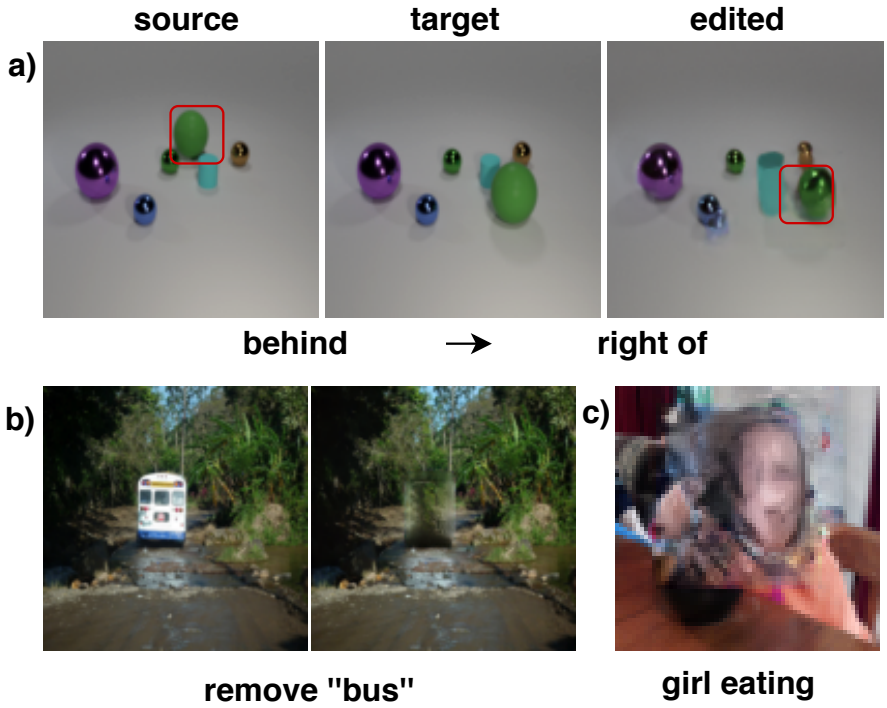


Fig. 12.16. Some Failure Cases

Part IV

Conclusion and Outlook

13 Summary of Findings

This thesis has investigated different aspects of learning to learn, a concept that aims to enhance the performance of machine learning models on new tasks and domains by learning from other models, other data, and other semantics. We have applied this paradigm to various computer vision and medical imaging applications. Our contributions have been divided into two parts: the first part has dealt with fundamental research on learning to learn from data, and the second part has dealt with learning from semantics using generative models.

In the first part, we have addressed the different challenges arising in learning to learn from the data. First, we tackle the problem of non-iid data distribution in federated, collaborative and distributed learning, where each model learns from the knowledge acquired from other nodes without directly accessing their data. We have developed IDA and FedAP, two methods that use meta-learning and hierarchical clustering to aggregate parameters and personalize models. We have also developed MetaMedSeg, a meta-learning framework for few-shot medical image segmentation that uses other organs with abundant labels to boost the performance on a target organ with scarce labels. Moreover, we have developed YNet, a novel method for image segmentation that extracts and fuses spectral and spatial features.

In the second part, we have used generative models and semantics to learn a representation of the scene. We have used scene graphs, a structured representation that describes the objects and their relationships in an image, as a versatile tool for image generation and manipulation. We have developed several methods that use scene graphs for different tasks, such as unconditional image generation, few-shot image generation, and self-supervised image manipulation. We have also developed DisPositioNet, a framework for self-supervised feature disentanglement in the semantic image manipulation task.

Our work opens up new possibilities for future research on learning to learn and semantic representation learning for computer vision and medical imaging applications. We hope that our work will inspire more researchers to investigate the potential of learning from existing knowledge sources and semantics, rather than relying solely on large-scale data and models. By doing so, we believe that we can achieve more efficient, robust, and generalizable machine learning models that can better understand and interact with the visual world. We discuss some possible directions for future work in the next chapter.

14 Future Work

In this thesis, we have explored representation learning with limited data and supervision, as well as methods for scene modeling using generative models that can capture and manipulate the appearance and structure of objects in an image. However, there are still many challenges and opportunities for future research in this area. In particular, the following directions can be explored:

Modeling scenes using dynamic scene graphs We aim to develop methods that can model the scenes with graphs that account for the dynamics of the scene caused by actions. We will investigate how to model the scene changes due to actions by agents or external forces, how to generate realistic images that match the updated scene graph after an action, and how to infer the causal effects of actions on the scene graph. This direction is relevant for both medical and non-medical domains, where actions can have significant impacts on the scene.

Modeling scenes using physics We aim to develop methods that can incorporate physics into scene modeling, which can provide a principled way of modeling scenes that is consistent with reality and generalizable across domains. We will investigate how to model the physical properties of objects, such as mass, shape, material, elasticity, friction, etc., how to model the physical interactions between objects, such as collision, contact, gravity, etc., and how to model the physical effects of actions on objects, such as deformation, displacement, rotation, etc. This direction is crucial for both medical and non-medical domains, where physics can affect the appearance and state of objects in a scene. For example, we can use physics to capture the biomechanical properties of human anatomy and physiology that can affect health conditions.

Part V

Appendix

MIGS

A

A.0.1 Architecture Details

CRN The CRN version of the decoder network is composed of five successive refinement blocks, with their respective channel counts being 1024, 512, 256, 128, and 64. Each block is built with two 3×3 convolution layers, each of which is followed by batch normalization and a leaky ReLU activation function. The output of each block is concatenated with the initial input to the CRN, after being rescaled to match the feature resolution.

SPADE The SPADE decoder is structured with five residual blocks, each with respective channel counts of 1024, 512, 256, 128, and 64. We leverage the layout to modulate the layer activations within each block, instead of using the semantic map from the original design. The global discriminator, D_{global}, is designed with two scales.

GCN Our GCN network is constructed with five layers. Each layer processes triplet embeddings of subject-predicate-object, which are derived from feeding each semantic label into an embedding layer. The construction of each layer involves three steps. First, the propagation layer (a two-layer MLP) accepts the concatenated triplet feature, resulting in an output of 128 channels. Second, the aggregation layer calculates the average of features related to a certain node. Third, the update layer applies a final processing to each node feature via another two-layer MLP. Both MLPs mentioned above incorporate a hidden layer with 512 channels. The input embeddings for the objects and predicates each consist of 128 dimensions. The final layer of the GCN delivers node features (128 channels), binary masks (16×16), and bounding box prediction by implementing a two-layer MLP with a hidden layer size of 128.

SIMSG

B

B.0.1 Implementation details

Image → scene graph

A state-of-the-art scene graph prediction network [139] is used to acquire scene graphs for the experiments on VG. We use their publicly available implementation[1] to train the model. The data used to train the network is pre-processed following [36], resulting in a typically used subset of Visual Genome (sVG) that includes 399 object and 24 predicate categories. We then split the data as in [104] to avoid overlap in the training data for the image manipulation model. We train the model for 30 epochs with a batch size of 8 images using the default settings from [139].

Scene graph → image

SGN architecture details. The learned embeddings of the object c_i and predicate r_i both have 128 dimensions. We create the full representation of each object o_i by concatenating c_i together with the bounding box coordinates x_i (top, left, bottom, right) and the visual features (n=128) corresponding to the cropped image region defined by the bounding box. The features are extracted by a VGG-16 architecture [216] followed by a 128-dimensional fully connected layer.

During training, to hide information from the network, we randomly mask the visual features ϕ_i and/or object coordinates x_i with independent probabilities of $p_\phi = 0.25$ and $p_x = 0.35$.

The SGN consists of 5 layers. τ_e and τ_n are implemented as 2-layer MLPs with 512 hidden and 128 output units. The last layer of the SGN returns the outputs; the node features (s=128), binary masks (16×16) and bounding box coordinates by 2-layer MLP with a hidden size of 128 (which is needed to add or re-position objects).

CRN architecture details. The CRN architecture consists of 5 cascaded refinement modules, with the output number of channels being 1024, 512, 256, 128 and 64 respectively. Each module consists of two convolutions (3×3), each followed by batch normalization [89] and leaky Relu. The output of each module is concatenated with a down-sampled version of the initial input to the CRN. The initial input is the concatenation of the predicted layout and the masked image features. The generated images have a resolution of 64×64.

SPADE architecture details. The SPADE architecture used in this work contains 5 residual blocks. The output number of channels is namely 1024, 512, 256, 128 and 64. In each block,

[1] https://github.com/yikang-li/FactorizableNet

the layout is fed in the SPADE normalization layer, to modulate the layer activations, while the image counterpart is concatenated with the result. The global discriminator D_{global} contains two scales.

The object discriminator in both cases is only applied on the image areas that have changed, *i.e*. have been in-painted.

Full-image branch details. The image regions that we randomly mask during training are replaced by Gaussian noise. Image features are extracted using 32 convolutional filters (1×1), followed by batch normalization and Relu activation. Additionally, a mask is concatenated with the image features that is 1 in the regions of interest (noise) and 0 otherwise, so that the areas to be modified are easier for the network to identify.

Training settings. In all experiments presented in this paper, the models were trained with Adam optimization [118] with a base learning rate of 10^{-4}. The weighting values for different loss terms in our method are shown in Table B.1. The batch size for the images in 64×64 resolution is 32, while for 128×128 is 8. All objects in an image batch are fed at the same time in the object-level units, *i.e*. SGN, visual feature extractor and discriminator.

All models on VG were trained for 300k iterations and on CLEVR for 40k iterations. Training on an Nvidia RTX GPU, for images of size 64×64 takes about 3 days for Visual Genome and 4 hours for CLEVR.

Loss factor	Weight CRN	Weight SPADE
λ_g	0.01	1
λ_o	0.01	0.1
λ_a	0.1	0.1
λ_b	10	50
λ_f	-	10
λ_p	-	10

Tab. B.1. Loss weighting values

C DisPositioNet

The weighting hyperparameters follow the same values as SIMSG and are presented in table B.1. The optimizer used for all the experiments is Adam [118] with an initial learning rate of 0.0002. The batch size used for all trainings is set to 32. The models were trained on a single NVIDIA Titan V, which takes five days for 300k iterations.

C.1 Architecture Details

DSGN architecture

The input to the DSGN is the visual features from a VGG-16 [216] network, bounding box coordinates, node embeddings, and the edge embeddings, along with the edges. The DSGN has two subnetworks, with the same architecture shown in table C.1. The first network produces latent features by applying the edge information to the concatenated features. The second network receives these latent features and produces the final object features, which are later used by the pose and appearance encoder network. Each DSGN subnetwork reduces the input dimensionality via a Sparse input layer [57] to $k * nhidden$, where $nhidden = 14$ is the size of the hidden dimension in the neighborhood routing (NeibRouting) layer, and $k = 16$ determines the disentangling factor. The routing is performed for 12 iterations for each of the 10 NeibRouting layers. The output of these layers is then processed by a fully connected layer, 1D Batchnnorm, and an activation function.

Tab. C.1. DisenGCN Architecture

Input: node_features (node_dim), edges (edge_dim)
SparseInputLayer (node_dim, k * nhidden)
LeakyRelu
10X NeibRouting (k * nhidden)
FC (k * nhidden, out_dim)
BatchNorm1D (out_dim)
LeakyRelu
Output: $z_{\mathcal{G}}$ (out_dim)

Appearance decoder Q_A

The appearance decoder network consists of 5 SPADE Residual Blocks from the original implementation [174] with up-sampling layers in between, followed by a Conv2D, an activation function, and another Conv2D layer as shown in table C.2.

Tab. C.2. Appearance decoder architecture

Input: z_l (in_dim)
SPADEResBlock (in_dim, 1024)
Upsample (scale=2)
SPADEResBlock (1024, 512)
Upsample (scale=2)
SPADEResBlock (512, 256)
Upsample (scale=2)
SPADEResBlock (256, 128)
Upsample (scale=2)
SPADEResBlock (128, 64)
Conv2D (64, 64, kernel=(3,3), padding=1)
LeakyRelu
Conv2D (64, 3, kernel=(1,1))
Output: Image (64, 64, 3)

Encoder architectures

For both E_A, E_P, we use small convolutional networks to encode the object features. Both encoders have the same architecture and consist of two encoders, one for modeling data mean μ and one for the data variance σ. The variance encoder branch has a *SoftPlus* activation layer in the output. table C.3 shows the architecture of the encoders.

Tab. C.3. Encoder Architecture

Input: $z_{\mathcal{G}}$ (in_dim)
BatchNorm2D (in_dim)
Conv2D (in_dim, 128, kernel=(1,1), stride=1)
LeakyRelu
BatchNorm2D (128)
Conv2D (128, in_dim, kernel=(1,1), stride=1)
LeakyRelu
Output: $z_{\mathcal{G}A/P}$ (in_dim)

Pose Decoder Q_P

The pose decoder network Q_P receives the latent embeddings from the pose encoder E_P and outputs a vector with the size of 6 per object for each of the transformation parameters. As shown in table C.4, the final output has the shape of out_dim, which is equal to 6 here. Similar to the encoders, the Q_P has two subnetworks with the same architecture for modeling μ and σ. The σ branch, again has a *SoftPlus* layer in the output.

Discriminator architecture

Both global and local discriminators follow the implementation of Multiscale Discriminator from SPADE [174]. There are three discriminator networks at different scale values, with each having 3 Conv2D layers.

Tab. C.4. Pose Decoder Architecture

Input: $z_{\mathcal{G}P}$ (in_dim)
FC (in_dim, 64)
LeakyRelu
FC (64, out_dim)
LeakyRelu
Output: γ (out_dim = 6)

Additional Qualitative Results on VG [121] More qualitative results compared to [42] are provided in Figure C.1. Similar to the results on the COCO dataset, our model modifies the objects more realistically on the VG dataset.

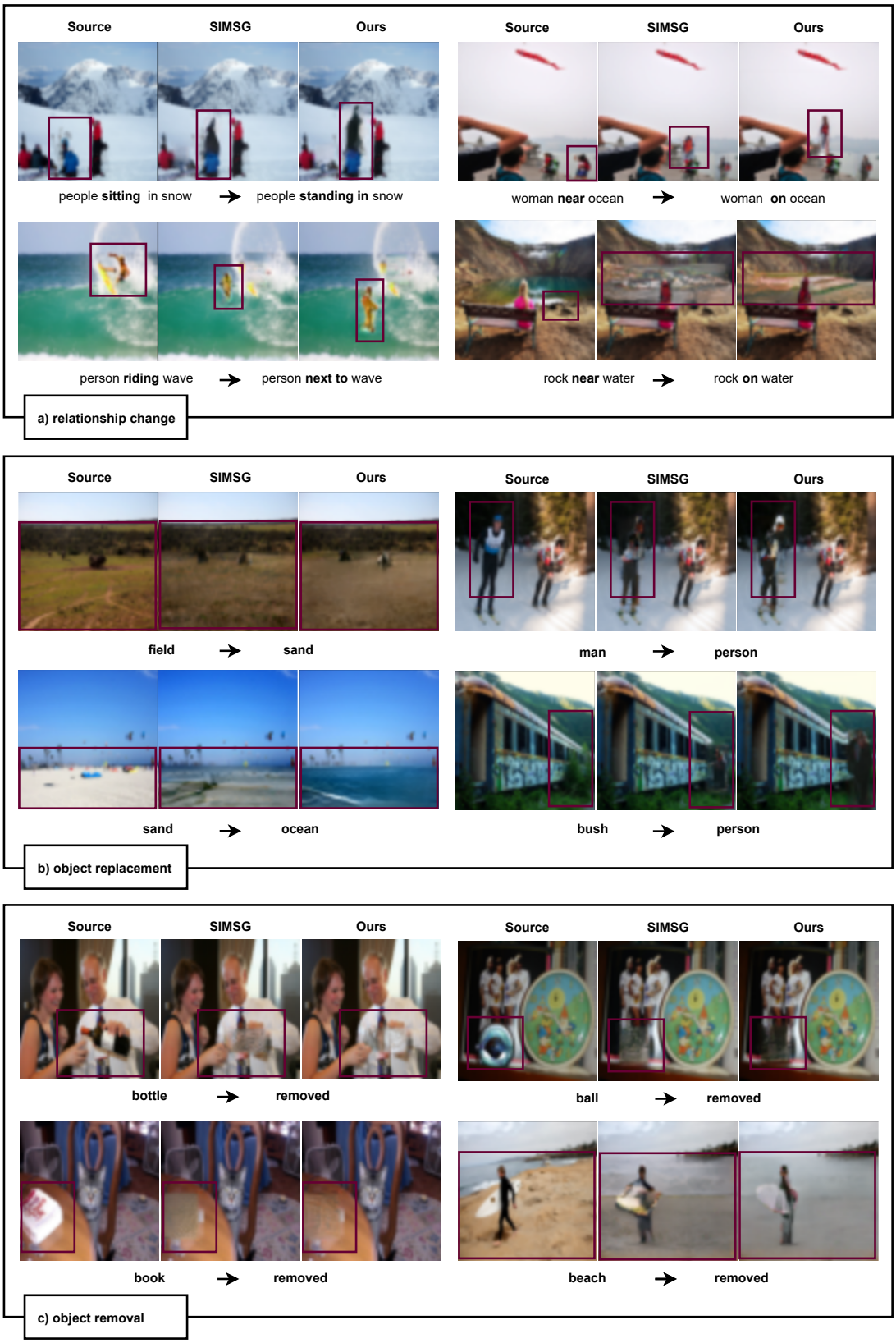


Fig. C.1. Additional qualitative results for image manipulation on VG

Bibliography

[1] E. Abbe. “Community Detection and Stochastic Block Models: Recent Developments”. In: *Journal of Machine Learning Research* 18.177 (2018), pp. 1–86 (cit. on p. 62).

[2] N. Abraham and N. M. Khan. “A novel focal tversky loss function with improved attention u-net for lesion segmentation”. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE. 2019, pp. 683–687 (cit. on p. 40).

[3] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary. “Federated Learning with Personalization Layers”. In: *arXiv:1912.00818 [cs, stat]* (Dec. 2019). arXiv: 1912.00818 (cit. on p. 18).

[4] O. Ashual and L. Wolf. “Specifying object attributes and relations in interactive scene generation”. In: *ICCV*. 2019, pp. 4561–4569 (cit. on pp. 61, 80, 98, 101–103, 112, 115–117, 121).

[5] R. Azad, A. R. Fayjie, C. Kauffmann, I. Ben Ayed, M. Pedersoli, and J. Dolz. “On the texture bias for few-shot CNN segmentation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 2674–2683 (cit. on p. 31).

[6] S. Azadi, D. Pathak, S. Ebrahimi, and T. Darrell. “Compositional gan: Learning conditional image composition”. In: *arXiv preprint arXiv:1807.07560* (2018) (cit. on p. 101).

[7] J. Bai, W. Wang, and C. Gomes. “Contrastively Disentangled Sequential Variational Autoencoder”. In: *arXiv preprint arXiv:2110.12091* (2021) (cit. on p. 103).

[8] T. Bdair, N. Navab, and S. Albarqouni. “FedPerl: Semi-supervised Peer Learning for Skin Lesion Classification”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2021, pp. 336–346 (cit. on p. 18).

[9] T. Bdair, N. Navab, and S. Albarqouni. “Semi-Supervised Federated Peer Learning for Skin Lesion Classification”. In: *Machine Learning for Biomedical Imaging* 1.April 2022 issue (2022), pp. 1–10 (cit. on p. 18).

[10] J. Beel. “Federated Meta-Learning: Democratizing Algorithm Selection Across Disciplines and Software Libraries”. In: *Science (AICS)* 210 (2018), p. 219 (cit. on p. 17).

[11] Y. Bengio, A. Courville, and P. Vincent. “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828 (cit. on pp. 7, 13).

[12] A. Bertugli, S. Vincenzi, S. Calderara, and A. Passerini. “Few-shot unsupervised continual learning through meta-examples”. In: *arXiv preprint arXiv:2009.08107* (2020) (cit. on p. 32).

[13] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. “Demystifying mmd gans”. In: *arXiv preprint arXiv:1801.01401* (2018) (cit. on p. 84).

[14] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann. “NetGAN: Generating Graphs via Random Walks”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 610–619 (cit. on pp. 60, 62).

[15] M. Boudiaf, H. Kervadec, Z. I. Masud, P. Piantanida, I. Ben Ayed, and J. Dolz. “Few-Shot Segmentation Without Meta-Learning: A Good Transductive Inference Is All You Need?” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 13979–13988 (cit. on p. 31).

[16] S. Boughorbel, J.-P. Tarel, and F. Fleuret. “Non-Mercer Kernels for SVM Object Recognition”. In: (Jan. 2004) (cit. on p. 68).

[17] C. Briggs, Z. Fan, and P. Andras. *Federated learning with hierarchical clustering of local updates to improve training on non-IID data*. May 2020 (cit. on pp. 16, 25, 28).

[18] A. Brock, J. Donahue, and K. Simonyan. “Large Scale GAN Training for High Fidelity Natural Image Synthesis”. In: *7th International Conference on Learning Representations, ICLR*. OpenReview.net, 2019 (cit. on p. 79).

[19] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. “Neural photo editing with introspective adversarial networks”. In: *arXiv preprint arXiv:1609.07093* (2016) (cit. on p. 101).

[20] J. Bronskill, J. Gordon, J. Requeima, S. Nowozin, and R. Turner. “Tasknorm: Rethinking batch normalization for meta-learning”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 1153–1164 (cit. on p. 33).

[21] T. Brown, B. Mann, N. Ryder, et al. “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901 (cit. on p. 58).

[22] N. D. Cao and T. Kipf. “MolGAN: An implicit generative model for small molecular graphs”. In: *ICML 2018 workshop on Theoretical Foundations and Applications of Deep Generative Models* (2018). arXiv: `1805.11973 [stat.ML]` (cit. on pp. 60, 62).

[23] M. Caron, H. Touvron, I. Misra, et al. “Emerging properties in self-supervised vision transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 9650–9660 (cit. on p. 14).

[24] F. Chen, Z. Dong, Z. Li, and X. He. “Federated meta-learning for recommendation”. In: *arXiv preprint arXiv:1802.07876* (2018) (cit. on pp. 16, 17).

[25] Q. Chen and V. Koltun. “Photographic image synthesis with cascaded refinement networks”. In: *ICCV*. 2017, pp. 1511–1520 (cit. on pp. 80, 84, 97, 101, 106).

[26] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 2180–2188 (cit. on p. 103).

[27] Y. Chen, X. Sun, and Y. Jin. “Communication-Efficient Federated Deep Learning With Layerwise Asynchronous Model Update and Temporally Weighted Aggregation”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2019) (cit. on p. 17).

[28] L. Chi, B. Jiang, and Y. Mu. “Fast fourier convolution”. In: *Advances in Neural Information Processing Systems* (2020) (cit. on pp. 45–47).

[29] S. J. Chiu, X. T. Li, P. Nicholas, C. A. Toth, J. A. Izatt, and S. Farsiu. “Automatic segmentation of seven retinal layers in SDOCT images congruent with expert manual segmentation”. In: *Opt. Express* (2010) (cit. on p. 45).

[30] S. J. Chiu, M. J. Allingham, P. S. Mettu, S. W. Cousins, J. A. Izatt, and S. Farsiu. “Kernel regression based segmentation of optical coherence tomography images with diabetic macular edema”. In: *Biomed. Opt. Express* (2015) (cit. on p. 50).

[31] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. 2014. arXiv: 1409.1259 [cs.CL] (cit. on p. 60).

[32] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation". In: *CVPR*. 2018, pp. 8789–8797 (cit. on p. 101).

[33] L. Clouâtre and M. Demers. "Figr: Few-shot image generation with reptile". In: *arXiv preprint arXiv:1901.02199* (2019) (cit. on pp. 81, 82).

[34] L. Corinzia and J. M. Buhmann. "Variational Federated Multi-Task Learning". In: *arXiv preprint arXiv:1906.06268* (2019) (cit. on p. 17).

[35] C. Cortes and V. Vapnik. "Support-vector networks". In: *Machine learning* 20 (1995), pp. 273–297 (cit. on p. 7).

[36] B. Dai, Y. Zhang, and D. Lin. "Detecting visual relationships with deep relational networks". In: *CVPR*. 2017, pp. 3076–3086 (cit. on pp. 102, 137).

[37] Y. Dawoud, J. Hornauer, G. Carneiro, and V. Belagiannis. "Few-Shot Microscopy Image Cell Segmentation". In: *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part V*. Vol. 12461. Springer, 2020, pp. 139–154 (cit. on pp. 31, 32, 34, 37, 39).

[38] J. Devaranjan, A. Kar, and S. Fidler. "Meta-Sim2: Learning to Generate Synthetic Datasets". In: *ECCV*. 2020 (cit. on p. 62).

[39] H. Dhamo, F. Manhardt, N. Navab, and F. Tombari. "Graph-to-3D: End-to-End Generation and Manipulation of 3D Scenes Using Scene Graphs". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021 (cit. on p. 80).

[40] H. Dhamo, F. Manhardt, N. Navab, and F. Tombari. "Graph-to-3D: End-to-End Generation and Manipulation of 3D Scenes using Scene Graphs". In: *IEEE International Conference on Computer Vision (ICCV)*. 2021 (cit. on p. 102).

[41] H. Dhamo, N. Navab, and F. Tombari. "Object-Driven Multi-Layer Scene Decomposition From a Single Image". In: *ICCV*. 2019 (cit. on p. 101).

[42] H. Dhamo, A. Farshad, I. Laina, et al. "Semantic image manipulation using scene graphs". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 5213–5222 (cit. on pp. 2, 9, 10, 59, 61, 80, 98, 99, 110, 117–121, 141).

[43] C. Doersch, A. Gupta, and A. A. Efros. "Unsupervised visual representation learning by context prediction". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430 (cit. on p. 14).

[44] W. Duan, Y. Zheng, Y. Ding, et al. "A Generative Model for OCT Retinal Layer Segmentation by Groupwise Curve Alignment". In: *IEEE Access* (2018) (cit. on p. 46).

[45] P. Erdös and A. Rényi. "On Random Graphs I". In: *Publicationes Mathematicae Debrecen* 6 (1959), p. 290 (cit. on p. 62).

[46] P. Esser, E. Sutter, and B. Ommer. "A variational u-net for conditional appearance and shape generation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8857–8866 (cit. on p. 103).

[47] A. Fallah, A. Mokhtari, and A. Ozdaglar. *Personalized Federated Learning: A Meta-Learning Approach*. 2020. arXiv: 2002.07948 [cs.LG] (cit. on p. 16).

[48] S. Fan and B. Huang. *Labeled Graph Generative Adversarial Networks*. 2019. arXiv: 1906.03220 [cs.LG] (cit. on pp. 60, 62).

[49] L. Fang, D. Cunefare, C. Wang, R. Guymer, S. Li, and S. Farsiu. “Automatic segmentation of nine retinal layer boundaries in OCT images of non-exudative AMD patients using deep learning and graph search”. In: *Biomedical Optics Express* (2017) (cit. on p. 45).

[50] A. Farshad, Y. Yeganeh, H. Dhamo, F. Tombari, and N. Navab. “DisPositioNet: Disentangled Pose and Identity in Semantic Image Manipulation”. In: *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022 (cit. on pp. 1, 9, 10, 120).

[51] A. Farshad, Y. Yeganeh, and N. Navab. “Learning to learn in medical applications: A journey through optimization”. In: *Meta-Learning with Medical Imaging and Health Informatics Applications*. Elsevier, 2023, pp. 3–25 (cit. on pp. 1, 13).

[52] A. Farshad, A. Makarevich, V. Belagiannis, and N. Navab. “MetaMedSeg: Volumetric Meta-learning for Few-Shot Organ Segmentation”. In: *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*. Springer, 2022 (cit. on pp. 1, 9).

[53] A. Farshad, S. Musatian, H. Dhamo, and N. Navab. “MIGS: Meta Image Generation from Scene Graphs”. In: *BMVC*. 2021 (cit. on pp. 1, 9, 10, 101).

[54] A. Farshad, Y. Yeganeh, Y. Chi, C. Shen, B. Ommer, and N. Navab. “SceneGenie: Scene Graph Guided Diffusion Models for Image Synthesis”. In: *arXiv preprint arXiv:2304.14573* (2023) (cit. on p. 9).

[55] A. Farshad, Y. Yeganeh, P. Gehlbach, and N. Navab. “Y-Net: A Spatiospectral Network for Retinal OCT Segmentation”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2022 (cit. on pp. 1, 9, 10).

[56] P. F. Felzenszwalb and D. P. Huttenlocher. “Efficient graph-based image segmentation”. In: *International journal of computer vision* 59 (2004), pp. 167–181 (cit. on p. 7).

[57] J. Feng and N. Simon. “Sparse-input neural networks for high-dimensional nonparametric regression and classification”. In: *arXiv preprint arXiv:1711.07592* (2017) (cit. on pp. 109, 139).

[58] S. Feng, H. Zhao, F. Shi, et al. “CPFNet: Context pyramid fusion network for medical image segmentation”. In: *IEEE transactions on medical imaging* (2020) (cit. on p. 46).

[59] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera. *Learning from imbalanced data sets*. Vol. 11. Springer, 2018 (cit. on p. 27).

[60] C. Finn, P. Abbeel, and S. Levine. “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1126–1135 (cit. on pp. 7, 13, 16, 58, 81, 83).

[61] M. Fisher, M. Savva, and P. Hanrahan. “Characterizing Structural Relationships in Scenes Using Graph Kernels”. In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. Association for Computing Machinery, 2011 (cit. on pp. 67, 70).

[62] S. Gairola, M. Hemani, A. Chopra, and B. Krishnamurthy. “SimPropNet: Improved Similarity Propagation for Few-shot Image Segmentation”. In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020 [scheduled for July 2020, Yokohama, Japan, postponed due to the Corona pandemic]*. Ed. by C. Bessiere. ijcai.org, 2020, pp. 573–579 (cit. on p. 31).

[63] S. Garg, H. Dhamo, A. Farshad, S. Musatian, N. Navab, and F. Tombari. “Unconditional scene graph generation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 16362–16371 (cit. on pp. 1, 9, 10, 60, 80, 102).

[64] S. Ghosh, G. Burachas, A. Ray, and A. Ziskind. *Generating Natural Language Explanations for Visual Question Answering using Scene Graphs and Visual Attention*. 2019. arXiv: 1902.05715 [cs.CL] (cit. on pp. 59, 61).

[65] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587 (cit. on p. 57).

[66] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016 (cit. on pp. 7, 13, 14).

[67] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. “Generative adversarial nets”. In: *NeurIPS*. 2014 (cit. on pp. 80, 97, 100).

[68] A. Grover, A. Zweig, and S. Ermon. “Graphite: Iterative Generative Modeling of Graphs”. In: *arXiv preprint arXiv:1803.10459* (2018). arXiv: 1803.10459 [stat.ML] (cit. on p. 60).

[69] R. Hadsell, S. Chopra, and Y. LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 1735–1742 (cit. on p. 14).

[70] K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969 (cit. on p. 57).

[71] Y. He, A. Carass, Y. Liu, et al. “Deep learning based topology guaranteed surface and MME segmentation of multiple sclerosis subjects from retinal OCT”. In: *Biomed. Opt. Express* (2019) (cit. on p. 46).

[72] Y. He, A. Carass, Y. Liu, et al. “Fully convolutional boundary regression for retina OCT segmentation”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2019 (cit. on p. 46).

[73] Y. He, A. Carass, B. M. Jedynak, et al. *Topology guaranteed segmentation of the human retina from OCT using convolutional neural networks*. 2018. arXiv: 1803.05120 [cs.CV] (cit. on p. 46).

[74] Y. He, A. Carass, Y. Yun, et al. “Towards Topological Correct Segmentation of Macular OCT from Cascaded FCNs”. In: *Fetal, Infant and Ophthalmic Medical Image Analysis*. 2017 (cit. on p. 46).

[75] M. Hein and O. Bousquet. *Kernels, Associated Structures and Generalizations*. Tech. rep. 127. Max Planck Institute for Biological Cybernetics, Tübingen, Germany, July 2004 (cit. on p. 68).

[76] D. Hendrycks, M. Mazeika, and T. Dietterich. “Deep Anomaly Detection with Outlier Exposure”. In: *ICLR* (2019). arXiv: 1812.04606 [cs.LG] (cit. on p. 73).

[77] R. Herzig, A. Bar, H. Xu, G. Chechik, T. Darrell, and A. Globerson. “Learning Canonical Representations for Scene Graph to Image Generation”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2020 (cit. on p. 80).

[78] R. Herzig, M. Raboh, G. Chechik, J. Berant, and A. Globerson. “Mapping images to scene graphs with permutation-invariant structured prediction”. In: *NeurIPS*. 2018 (cit. on pp. 80, 102).

[79] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *arXiv preprint arXiv:1706.08500* (2017) (cit. on pp. 71, 84, 111, 113).

[80] I. Higgins, L. Matthey, A. Pal, et al. “beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework”. In: *International Conference on Learning Representations*. 2017 (cit. on p. 103).

[81] S. Hong, D. Yang, J. Choi, and H. Lee. “Inferring semantic layout for hierarchical text-to-image synthesis”. In: *CVPR*. 2018 (cit. on pp. 97, 101).

[82] S. Hong, X. Yan, T. E. Huang, and H. Lee. “Learning hierarchical semantic image manipulation through structured representations”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 2713–2723 (cit. on pp. 97, 101).

[83] K. Hsieh, A. Phanishayee, O. Mutlu, and P. B. Gibbons. “The Non-IID Data Quagmire of Decentralized Machine Learning”. In: *arXiv preprint arXiv:1910.00189* (2019) (cit. on p. 15).

[84] S.-M. Hu, F.-L. Zhang, M. Wang, R. R. Martin, and J. Wang. “PatchNet: A patch-based image representation for interactive library-driven image editing”. In: *ACM Transactions on Graphics (TOG)* 32.6 (2013), pp. 1–12 (cit. on pp. 101, 102).

[85] T. Hu, P. Yang, C. Zhang, G. Yu, Y. Mu, and C. G. Snoek. “Attention-based multi-context guiding for few-shot semantic segmentation”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 2019, pp. 8441–8448 (cit. on p. 31).

[86] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. “Densely connected convolutional networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4700–4708 (cit. on p. 21).

[87] L. Huang, A. L. Shea, H. Qian, A. Masurkar, H. Deng, and D. Liu. “Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records”. In: *Journal of Biomedical Informatics* 99 (2019), p. 103291 (cit. on p. 18).

[88] J. Ingraham, V. K. Garg, R. Barzilay, and T. Jaakkola. “Generative Models for Graph-Based Protein Design”. In: *Advances in Neural Information Processing Systems*. 2019 (cit. on p. 62).

[89] S. Ioffe and C. Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015) (cit. on p. 137).

[90] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. “Image-to-image translation with conditional adversarial networks”. In: *CVPR*. 2017, pp. 1125–1134 (cit. on pp. 80, 101).

[91] M. Ivgi, Y. Benny, A. Ben-David, J. Berant, and L. Wolf. “Scene Graph to Image Generation with Contextualized Object Layout Refinement”. In: *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2021, pp. 2428–2432 (cit. on p. 101).

[92] S. Jadon. “COVID-19 detection from scarce chest x-ray image data using few-shot deep learning approach”. In: *Medical Imaging 2021: Imaging Informatics for Healthcare, Research, and Applications*. Vol. 11601. International Society for Optics and Photonics. 2021, p. 116010X (cit. on p. 31).

[93] S. Jae Hwang, S. N. Ravi, Z. Tao, H. J. Kim, M. D. Collins, and V. Singh. “Tensorize, factorize and regularize: Robust visual relationship learning”. In: *CVPR*. 2018, pp. 1014–1023 (cit. on p. 102).

[94] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim. “Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data”. In: *arXiv preprint arXiv:1811.11479* (2018) (cit. on p. 17).

[95] J. Ji, R. Krishna, L. Fei-Fei, and J. C. Niebles. “Action Genome: Actions As Compositions of Spatio-Temporal Scene Graphs”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020 (cit. on pp. 80, 84, 86).

[96] S. Jia, D.-J. Chen, and H.-T. Chen. “Instance-level meta normalization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4865–4873 (cit. on p. 33).

[97] L. Jiang, B. Dai, W. Wu, and C. C. Loy. “Focal frequency loss for image reconstruction and synthesis”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021 (cit. on p. 52).

[98] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan. “Improving Federated Learning Personalization via Model Agnostic Meta Learning”. In: *arXiv preprint arXiv:1909.12488* (2019) (cit. on pp. 16, 17).

[99] W. Jin, R. Barzilay, and T. Jaakkola. “Junction Tree Variational Autoencoder for Molecular Graph Generation”. In: *ICML* (2018). arXiv: `1802.04364 [cs.LG]` (cit. on p. 62).

[100] L. Jing and Y. Tian. “Self-supervised visual feature learning with deep neural networks: A survey”. In: *IEEE transactions on pattern analysis and machine intelligence* 43.11 (2020), pp. 4037–4058 (cit. on p. 13).

[101] Y. Jo and J. Park. “SC-FEGAN: Face Editing Generative Adversarial Network with User’s Sketch and Color”. In: *arXiv preprint arXiv:1902.06838* (2019) (cit. on p. 101).

[102] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C Lawrence Zitnick, and R. Girshick. “Clevr: A diagnostic dataset for compositional language and elementary visual reasoning”. In: *CVPR*. 2017 (cit. on pp. 99, 111, 113).

[103] J. Johnson, A. Karpathy, and L. Fei-Fei. “Densecap: Fully convolutional localization networks for dense captioning”. In: *CVPR*. 2016, pp. 4565–4574 (cit. on p. 97).

[104] J. Johnson, A. Gupta, and L. Fei-Fei. “Image Generation from Scene Graphs”. In: *CVPR*. 2018 (cit. on pp. 59, 61, 71, 72, 79–81, 86–88, 92, 98, 101, 102, 106, 111, 112, 117, 121, 137).

[105] J. Johnson, R. Krishna, M. Stark, et al. “Image retrieval using scene graphs”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3668–3678 (cit. on pp. 57, 59, 61, 79, 80, 82, 102).

[106] J. Johnson, A. Alahi, and L. Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *European conference on computer vision*. 2016 (cit. on p. 82).

[107] I. T. Jolliffe and J. Cadima. “Principal component analysis: a review and recent developments”. In: *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences* 374.2065 (2016), p. 20150202 (cit. on p. 7).

[108] P. Kairouz, H. B. McMahan, B. Avent, et al. “Advances and open problems in federated learning”. In: *arXiv preprint arXiv:1912.04977* (2019) (cit. on p. 18).

[109] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren. “Secure, privacy-preserving and federated machine learning in medical imaging”. In: *Nature Machine Intelligence* 2.6 (2020), pp. 305–311 (cit. on p. 18).

[110] A. Kar, A. Prakash, M.-Y. Liu, et al. “Meta-Sim: Learning to Generate Synthetic Datasets”. In: *ICCV*. 2019 (cit. on pp. 59, 62).

[111] T. Karras, S. Laine, and T. Aila. “A style-based generator architecture for generative adversarial networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4401–4410 (cit. on pp. 79, 80).

[112] T. Karras, M. Aittala, S. Laine, et al. “Alias-free generative adversarial networks”. In: *arXiv preprint arXiv:2106.12423* (2021) (cit. on p. 101).

[113] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. “Analyzing and improving the image quality of stylegan”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020 (cit. on pp. 72, 79, 80, 101).

[114] T. Karras, T. Aila, S. Laine, and J. Lehtinen. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *6th International Conference on Learning Representations, ICLR*. OpenReview.net, 2018 (cit. on pp. 79, 80).

[115] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. “Training Generative Adversarial Networks with Limited Data”. In: *Advances in Neural Information Processing Systems 33 (NeurIPS)*. 2020 (cit. on pp. 79, 80).

[116] M. Khodak, M.-F. Balcan, and A. Talwalkar. *Adaptive Gradient-Based Meta-Learning Methods*. 2019. arXiv: 1906.02717 [cs.LG] (cit. on p. 16).

[117] F. Kiaee, H. Fahimi, and H. Rabbani. “Intra-Retinal Layer Segmentation of Optical Coherence Tomography Using 3D Fully Convolutional Networks”. In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. 2018 (cit. on p. 46).

[118] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on pp. 51, 138, 139).

[119] D. P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013) (cit. on p. 100).

[120] J. Konečnỳ, B. McMahan, and D. Ramage. “Federated optimization: Distributed optimization beyond the datacenter”. In: *arXiv preprint arXiv:1511.03575* (2015) (cit. on p. 15).

[121] R. Krishna, Y. Zhu, O. Groth, et al. “Visual genome: Connecting language and vision using crowdsourced dense image annotations”. In: *IJCV* 123.1 (2017), pp. 32–73 (cit. on pp. 59, 61, 69, 80, 84, 86, 99, 111, 114, 118, 141).

[122] A. Krizhevsky, G. Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009) (cit. on p. 21).

[123] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C. Burges, L. Bottou, and K. Weinberger. Vol. 25. Curran Associates, Inc., 2012 (cit. on p. 57).

[124] J. Kugelman, D. Alonso-Caneiro, S. Read, et al. “Automatic choroidal segmentation in OCT images using supervised deep learning methods”. In: *Scientific Reports* (2019) (cit. on p. 46).

[125] J. Kugelman, D. Alonso-Caneiro, S. Read, S. Vincent, and M. Collins. “Automatic segmentation of OCT retinal boundaries using recurrent neural networks and graph search”. In: *Biomedical Optics Express* (2018) (cit. on p. 46).

[126] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, et al. “Fader networks: Manipulating images by sliding attributes”. In: *NeurIPS*. 2017, pp. 5967–5976 (cit. on p. 101).

[127] Y. LeCun, B. Boser, J. S. Denker, et al. “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation* 1.4 (1989), pp. 541–551 (cit. on p. 21).

[128] Y. LeCun, Y. Bengio, and G. Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444 (cit. on p. 7).

[129] C. Ledig, L. Theis, F. Huszár, et al. “Photo-realistic single image super-resolution using a generative adversarial network”. In: *CVPR*. 2017 (cit. on p. 101).

[130] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. “Kronecker Graphs: An Approach to Modeling Networks”. In: *J. Mach. Learn. Res.* 11 (Mar. 2010), 985–1042 (cit. on p. 62).

[131] D. Li and J. Wang. “FedMD: Heterogenous Federated Learning via Model Distillation”. In: *arXiv preprint arXiv:1910.03581* (2019) (cit. on pp. 17, 18).

[132] J. Li, P. Jin, J. Zhu, et al. “Multi-scale GCN-assisted two-stage network for joint segmentation of retinal layers and discs in peripapillary OCT images”. In: *Biomedical Optics Express* (2021) (cit. on p. 45).

[133] L. Li, K. Fan, and C. Yuan. “Cross-modal Representation Learning and Relation Reasoning for Bidirectional Adaptive Manipulation”. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. Ed. by L. D. Raedt. Main Track. International Joint Conferences on Artificial Intelligence Organization, July 2022, pp. 3222–3228 (cit. on p. 101).

[134] Q. Li, S. Li, Z. He, et al. “DeepRetina: Layer segmentation of retina in OCT images using deep learning”. In: *Translational Vision Science & Technology* (2020) (cit. on p. 46).

[135] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. “Federated learning: Challenges, methods, and future directions”. In: *IEEE Signal Processing Magazine* 37.3 (2020), pp. 50–60 (cit. on pp. 15, 18).

[136] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. “Federated optimization in heterogeneous networks”. In: *arXiv preprint arXiv:1812.06127* (2018) (cit. on p. 18).

[137] W. Li, F. Milletarì, D. Xu, et al. “Privacy-preserving Federated Brain Tumour Segmentation”. In: *International Workshop on Machine Learning in Medical Imaging*. Springer. 2019, pp. 133–141 (cit. on p. 18).

[138] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. “On the Convergence of FedAvg on Non-IID Data”. In: *arXiv:1907.02189 [cs, math, stat]* (June 2020). arXiv: 1907.02189 (cit. on pp. 18, 28).

[139] Y. Li, W. Ouyang, B. Zhou, J. Shi, C. Zhang, and X. Wang. “Factorizable net: an efficient subgraph-based framework for scene graph generation”. In: *ECCV*. 2018, pp. 335–351 (cit. on pp. 61, 98, 102, 105, 115, 137).

[140] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang. “Scene graph generation from objects, phrases and region captions”. In: *ICCV*. 2017, pp. 1261–1270 (cit. on p. 102).

[141] Y. Li, Z. Gan, Y. Shen, et al. “StoryGAN: A Sequential Conditional GAN for Story Visualization”. In: *arXiv preprint arXiv:1812.02784* (2018) (cit. on pp. 80, 97, 101).

[142] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia. *Learning Deep Generative Models of Graphs*. 2018. arXiv: 1803.03324 [cs.LG] (cit. on p. 62).

[143] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency. “Think Locally, Act Globally: Federated Learning with Local and Global Representations”. In: *arXiv preprint arXiv:2001.01523* (2020) (cit. on p. 18).

[144] R. Liao, Y. Li, Y. Song, et al. “Efficient Graph Generation with Graph Recurrent Attention Networks”. In: *NeurIPS*. 2019 (cit. on p. 62).

[145] T.-Y. Lin, M. Maire, S. Belongie, et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755 (cit. on pp. 99, 111, 114, 118).

[146] H. Ling, K. Kreis, D. Li, S. W. Kim, A. Torralba, and S. Fidler. “EditGAN: High-Precision Semantic Image Editing”. In: *arXiv preprint arXiv:2111.03186* (2021) (cit. on p. 101).

[147] M.-Y. Liu, X. Huang, A. Mallya, et al. “Few-shot unsupervised image-to-image translation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 10551–10560 (cit. on p. 81).

[148] Q. Liu, M. Allamanis, M. Brockschmidt, and A. L. Gaunt. “Constrained Graph Variational Autoencoders for Molecule Design”. In: *The Thirty-second Conference on Neural Information Processing Systems* (2018) (cit. on p. 62).

[149] W. Liu, C. Zhang, G. Lin, and F. Liu. “Crnet: Cross-reference networks for few-shot segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4165–4173 (cit. on p. 31).

[150] W. Liu, Y. Sun, and Q. Ji. “MDAN-UNet: Multi-Scale and Dual Attention Enhanced Nested U-Net Architecture for Segmentation of Optical Coherence Tomography Images”. In: *Algorithms* (2020) (cit. on p. 46).

[151] J. Long, E. Shelhamer, and T. Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440 (cit. on p. 57).

[152] D. G. Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60 (2004), pp. 91–110 (cit. on p. 7).

[153] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei. “Visual relationship detection with language priors”. In: *ECCV*. 2016 (cit. on pp. 97, 102).

[154] A. Luo, Z. Zhang, J. Wu, and J. B. Tenenbaum. “End-to-End Optimization of Scene Layout”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3754–3763 (cit. on p. 102).

[155] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu. “Disentangled Graph Convolutional Networks”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 4212–4221 (cit. on pp. 58, 103, 109).

[156] A. Madabhushi and G. Lee. “Image analysis and machine learning in digital pathology: Challenges and opportunities”. In: *Medical image analysis* 33 (2016), pp. 170–175 (cit. on p. 15).

[157] H. Maier, S. Faghihroohi, and N. Navab. “A Line to Align: Deep Dynamic Time Warping for Retinal OCT Segmentation”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2021 (cit. on pp. 46, 50).

[158] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Singh and J. Zhu. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, 2017, pp. 1273–1282 (cit. on pp. 17, 20, 21, 26, 28).

[159] G. A. Miller. “WordNet: a lexical database for English”. In: *Communications of the ACM* 38.11 (1995), pp. 39–41 (cit. on p. 58).

[160] F. Milletari, N. Navab, and S.-A. Ahmadi. “V-net: Fully convolutional neural networks for volumetric medical image segmentation”. In: *2016 fourth international conference on 3D vision (3DV)*. IEEE. 2016, pp. 565–571 (cit. on p. 31).

[161] M. Mirza and S. Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014) (cit. on pp. 100, 101).

[162] A. Mohammed, S. Yildirim, I. Farup, M. Pedersen, and Ø. Hovde. “Y-net: A deep convolutional neural network for polyp detection”. In: *arXiv preprint arXiv:1806.01907* (2018) (cit. on p. 46).

[163] A. K. Mondal, J. Dolz, and C. Desrosiers. “Few-shot 3d multi-modal medical image segmentation using generative adversarial learning”. In: *arXiv preprint arXiv:1810.12241* (2018) (cit. on p. 31).

[164] V. Nair, M. Chatterjee, N. Tavakoli, A. Namin, and C. Snoeyink. “Optimizing CNN using Fast Fourier Transformation for Object Recognition”. In: 2020 (cit. on p. 46).

[165] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi. “EdgeConnect: Structure Guided Image Inpainting using Edge Prediction”. In: *The IEEE International Conference on Computer Vision (ICCV) Workshops*. 2019 (cit. on p. 101).

[166] A. Newell and J. Deng. “Pixels to graphs by associative embedding”. In: *NeurIPS*. 2017, pp. 2171–2180 (cit. on pp. 59, 61, 80, 98, 102, 105).

[167] A. Nichol, J. Achiam, and J. Schulman. “On first-order meta-learning algorithms”. In: *arXiv preprint arXiv:1803.02999* (2018) (cit. on pp. 13, 16, 31, 34, 81–83).

[168] A. Q. Nichol and P. Dhariwal. “Improved denoising diffusion probabilistic models”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8162–8171 (cit. on pp. 80, 100).

[169] E. Ntavelis, A. Romero, I. Kastanis, L. Van Gool, and R. Timofte. “SESAME: Semantic Editing of Scenes by Adding, Manipulating or Erasing Objects”. In: *Computer Vision – ECCV 2020*. Ed. by A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm. Cham: Springer International Publishing, 2020, pp. 394–411 (cit. on pp. 97, 101).

[170] A. Odena, C. Olah, and J. Shlens. “Conditional image synthesis with auxiliary classifier gans”. In: *ICML*. 2017, pp. 2642–2651 (cit. on pp. 101, 107, 110).

[171] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. “Conditional image generation with pixelcnn decoders”. In: *NeurIPS*. 2016, pp. 4790–4798 (cit. on p. 100).

[172] J. I. Orlando, P. Seeböck, H. Bogunović, et al. “U2-net: A bayesian u-net model with epistemic uncertainty feedback for photoreceptor layer segmentation in pathological oct scans”. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE. 2019 (cit. on p. 46).

[173] C. Ouyang, C. Biffi, C. Chen, T. Kart, H. Qiu, and D. Rueckert. “Self-supervision with Superpixels: Training Few-Shot Medical Image Segmentation Without Annotation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 762–780 (cit. on p. 31).

[174] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. “Semantic image synthesis with spatially-adaptive normalization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2337–2346 (cit. on pp. 79, 80, 84, 101, 106, 108–110, 139, 140).

[175] G. Parmar, D. Li, K. Lee, and Z. Tu. “Dual contradistinctive generative autoencoder”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 823–832 (cit. on p. 103).

[176] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. “Context encoders: Feature learning by inpainting”. In: *CVPR*. 2016 (cit. on p. 101).

[177] M. Pekala, N. Joshi, T. A. Liu, N. M. Bressler, D. C. DeBuc, and P. Burlina. “Deep learning based retinal OCT segmentation”. In: *Computers in biology and medicine* (2019) (cit. on p. 46).

[178] J. Pennington, R. Socher, and C. D. Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543 (cit. on p. 58).

[179] K. Pillutla, S. M. Kakade, and Z. Harchaoui. “Robust Aggregation for Federated Learning”. In: *arXiv preprint* (2019) (cit. on p. 18).

[180] M. Qi, W. Li, Z. Yang, Y. Wang, and J. Luo. “Attentive Relational Networks for Mapping Images to Scene Graphs”. In: *arXiv preprint arXiv:1811.10696* (2018) (cit. on p. 102).

[181] A. Radford, L. Metz, and S. Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015) (cit. on p. 100).

[182] M. Rahman and N. Bouguila. “Efficient Feature Mapping in Classifying Proportional Data”. In: *IEEE Access* PP (Dec. 2020), pp. 1–1 (cit. on p. 68).

[183] M. A. Rahman and Y. Wang. “Optimizing intersection-over-union in deep neural networks for image segmentation”. In: *International symposium on visual computing*. Springer. 2016, pp. 234–244 (cit. on pp. 35, 36).

[184] K. Rakelly, E. Shelhamer, T. Darrell, A. Efros, and S. Levine. “Conditional networks for few-shot semantic segmentation”. In: (2018) (cit. on p. 14).

[185] A. Rashno, B. Nazari, D. D. Koozekanani, et al. “Fully-automated segmentation of fluid regions in exudative age-related macular degeneration subjects: Kernel graph cut in neutrosophic domain”. In: *PloS one* 12.10 (2017), e0186949 (cit. on p. 50).

[186] S. Reddi, Z. Charles, M. Zaheer, et al. “Adaptive Federated Optimization”. In: *arXiv:2003.00295 [cs, math, stat]* (Dec. 2020). arXiv: 2003.00295 (cit. on p. 18).

[187] T. G. P. Reddy, K. S. Ashritha, T. Prajwala, et al. “Retinal-Layer Segmentation Using Dilated Convolutions”. In: *Proceedings of 3rd International Conference on Computer Vision and Image Processing*. 2020 (cit. on p. 46).

[188] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. “Generative adversarial text to image synthesis”. In: *arXiv preprint arXiv:1605.05396* (2016) (cit. on pp. 80, 97, 101).

[189] S. Ren, K. He, R. Girshick, and J. Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *NeurIPS*. 2015, pp. 91–99 (cit. on pp. 97, 102).

[190] N. Rieke, J. Hancox, W. Li, et al. *The Future of Digital Health with Federated Learning*. 2021. arXiv: 2003.08119 [cs.CY] (cit. on p. 18).

[191] M. Rohrbach, S. Ebert, and B. Schiele. “Transfer learning in a transductive setting”. In: *Advances in neural information processing systems* 26 (2013), pp. 46–54 (cit. on p. 31).

[192] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695 (cit. on p. 14).

[193] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241 (cit. on pp. 31, 33, 38, 45–47, 50, 51).

[194] A. G. Roy, S. Siddiqui, S. Pölsterl, A. Farshad, N. Navab, and C. Wachinger. “Few-shot segmentation of 3D medical images”. In: *Meta-Learning with Medical Imaging and Health Informatics Applications*. Elsevier, 2023, pp. 161–183 (cit. on p. 1).

[195] A. G. Roy, S. Conjeti, S. P. K. Karri, et al. “ReLayNet: retinal layer and fluid segmentation of macular optical coherence tomography using fully convolutional networks”. In: *Biomedical optics express* (2017) (cit. on pp. 46, 50).

[196] S. J. Russell and P. Norvig. *Artificial Intelligence: a modern approach*. 3rd ed. Pearson, 2009 (cit. on p. 7).

[197] S. Sabour, N. Frosst, and G. E. Hinton. “Dynamic Routing Between Capsules”. In: *Advances in Neural Information Processing Systems* 30 (2017) (cit. on p. 109).

[198] M. A. Sadeghi and A. Farhadi. “Recognition using visual phrases”. In: *CVPR*. 2011 (cit. on p. 102).

[199] W. Sae-Lim, W. Wettayaprasit, and P. Aiyarak. “Convolutional neural networks using mobilenet for skin lesion classification”. In: *2019 16th international joint conference on computer science and software engineering (JCSSE)*. IEEE. 2019, pp. 242–247 (cit. on p. 26).

[200] M. S. M. Sajjadi, O. Bachem, M. Lučić, O. Bousquet, and S. Gelly. “Assessing Generative Models via Precision and Recall”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018 (cit. on pp. 71, 84).

[201] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. “Improved techniques for training gans”. In: *Advances in neural information processing systems* 29 (2016), pp. 2234–2242 (cit. on pp. 71, 111, 113).

[202] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. “Mobilenetv2: Inverted residuals and linear bottlenecks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520 (cit. on p. 26).

[203] F. Sattler, K.-R. Müller, and W. Samek. “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints”. In: *IEEE transactions on neural networks and learning systems* 32.8 (2020), pp. 3710–3722 (cit. on p. 18).

[204] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek. “Robust and communication-efficient federated learning from non-iid data”. In: *IEEE transactions on neural networks and learning systems* (2019) (cit. on pp. 18, 28).

[205] J. Schmidhuber. “Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook”. PhD thesis. Technische Universität München, 1987 (cit. on p. 7).

[206] J. M. Schmitt, S. Xiang, and K. M. Yung. “Speckle in optical coherence tomography”. In: *Journal of biomedical optics* 4.1 (1999), pp. 95–105 (cit. on p. 46).

[207] D. Seita. “BDD100K: A Large-scale Diverse Driving Video Database”. In: *The Berkeley Artificial Intelligence Research Blog. Version* 511 (2018), p. 41 (cit. on pp. 79, 80, 84, 85).

[208] A. Setlur, O. Li, and V. Smith. “Is Support Set Diversity Necessary for Meta-Learning?” In: *arXiv preprint arXiv:2011.14048* (2020) (cit. on p. 32).

[209] M. J. Sheller, B. Edwards, G. A. Reina, et al. “Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data”. In: *Scientific reports* 10.1 (2020), pp. 1–12 (cit. on p. 18).

[210] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas. “Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation”. In: *International MICCAI Brainlesion Workshop*. Springer. 2018, pp. 92–104 (cit. on pp. 15, 18).

[211] R. R. Shetty, M. Fritz, and B. Schiele. “Adversarial scene editing: Automatic object removal from weak supervision”. In: *NeurIPS*. 2018, pp. 7717–7727 (cit. on p. 101).

[212] J. Shi and J. Malik. “Normalized cuts and image segmentation”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905 (cit. on p. 7).

[213] Z. Shu, M. Sahasrabudhe, R. A. Guler, D. Samaras, N. Paragios, and I. Kokkinos. “Deforming autoencoders: Unsupervised disentangling of shape and appearance”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 650–665 (cit. on p. 103).

[214] M. Siam, B. N. Oreshkin, and M. Jagersand. “Amp: Adaptive masked proxies for few-shot segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5249–5258 (cit. on p. 31).

[215] M. Simonovsky and N. Komodakis. “GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders”. In: *ICANN* (2018). arXiv: 1802.03480 [cs.LG] (cit. on pp. 60, 62).

[216] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on pp. 21, 46, 104, 137, 139).

[217] A. L. Simpson, M. Antonelli, S. Bakas, et al. “A large annotated medical image dataset for the development and evaluation of segmentation algorithms”. In: *arXiv preprint arXiv:1902.09063* (2019) (cit. on p. 38).

[218] N. Skafte and S. Hauberg. “Explicit Disentanglement of Appearance and Perspective in Generative Models”. In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 1018–1028 (cit. on pp. 103, 109).

[219] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar. “Federated multi-task learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4424–4434 (cit. on pp. 17, 18).

[220] J. Snell, K. Swersky, and R. Zemel. “Prototypical networks for few-shot learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4077–4087 (cit. on p. 31).

[221] S. Su, L. Gao, J. Zhu, J. Shao, and J. Song. “Fully Functional Image Manipulation Using Scene Graphs in A Bounding-Box Free Way”. In: *Proceedings of the 29th ACM International Conference on Multimedia*. 2021, pp. 1784–1792 (cit. on p. 102).

[222] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M Jorge Cardoso. “Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations”. In: *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2017 (cit. on pp. 40, 49).

[223] M. Suhail, A. Mittal, B. Siddiquie, et al. “Energy-Based Learning for Scene Graph Generation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 13936–13945 (cit. on p. 102).

[224] W. Sun and T. Wu. “Image Synthesis From Reconfigurable Layout and Style”. In: *ICCV*. 2019 (cit. on pp. 80, 101–103).

[225] R. Suvorov, E. Logacheva, A. Mashikhin, et al. “Resolution-robust Large Mask Inpainting with Fourier Convolutions”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022 (cit. on p. 46).

[226] T. Sylvain, P. Zhang, Y. Bengio, R. D. Hjelm, and S. Sharma. “Object-Centric Image Generation from Layouts”. In: *AAAI*. 2021 (cit. on p. 80).

[227] S. A. Taghanaki, Y. Zheng, S. K. Zhou, et al. “Combo loss: Handling input and output imbalance in multi-organ segmentation”. In: *Computerized Medical Imaging and Graphics* (2019) (cit. on p. 49).

[228] K. Tang. *A Scene Graph Generation Codebase in PyTorch*. https://github.com/KaihuaTang/Scene-Graph-Benchmark.pytorch. 2020 (cit. on p. 69).

[229] K. Tang, Y. Niu, J. Huang, J. Shi, and H. Zhang. "Unbiased scene graph generation from biased training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3716–3725 (cit. on pp. 69, 102).

[230] N. J. Tatro, S. C. Schonsheck, and R. Lai. "Unsupervised Geometric Disentanglement for Surfaces via CFAN-VAE". In: *arXiv preprint arXiv:2005.11622* (2020) (cit. on p. 103).

[231] L. Theis, A. van den Oord, and M. Bethge. *A note on the evaluation of generative models*. 2016. arXiv: 1511.01844 [stat.ML] (cit. on p. 70).

[232] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 2012 (cit. on p. 13).

[233] P. Tian, Z. Wu, L. Qi, L. Wang, Y. Shi, and Y. Gao. "Differentiable meta-learning model for few-shot semantic segmentation". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 12087–12094 (cit. on p. 31).

[234] Z. Tian, H. Zhao, M. Shu, Z. Yang, R. Li, and J. Jia. "Prior guided feature enrichment network for few-shot segmentation". In: *IEEE Annals of the History of Computing* (2020), pp. 1–1 (cit. on p. 31).

[235] A. Tran, J. Weiss, S. Albarqouni, S. Faghi Roohi, and N. Navab. "Retinal layer segmentation reformulated as OCT language processing". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2020 (cit. on pp. 46, 50).

[236] P. Tschandl, C. Rosendahl, and H. Kittler. "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions". In: *Scientific data* 5 (2018), p. 180161 (cit. on pp. 21, 26, 27).

[237] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool. "Scan: Learning to classify images without labels". In: *European Conference on Computer Vision*. Springer. 2020, pp. 268–285 (cit. on pp. 83, 86).

[238] A. Van Oord, N. Kalchbrenner, and K. Kavukcuoglu. "Pixel Recurrent Neural Networks". In: *ICML*. 2016, pp. 1747–1756 (cit. on p. 100).

[239] J. Vanschoren. "Meta-learning: A survey". In: *arXiv preprint arXiv:1810.03548* (2018) (cit. on p. 16).

[240] R. Vilalta and Y. Drissi. "A perspective view and survey of meta-learning". In: *Artificial intelligence review* 18 (2002), pp. 77–95 (cit. on p. 7).

[241] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 1096–1103 (cit. on p. 14).

[242] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. "Matching networks for one shot learning". In: *Advances in neural information processing systems* 29 (2016) (cit. on p. 13).

[243] G. Virgili, F. Menchini, G. Casazza, et al. "Optical coherence tomography (OCT) for detection of macular oedema in patients with diabetic retinopathy". In: *Cochrane Database of Systematic Reviews* (2015) (cit. on p. 45).

[244] J. Wald, H. Dhamo, N. Navab, and F. Tombari. "Learning 3D Semantic Scene Graphs from 3D Indoor Reconstructions". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 (cit. on pp. 61, 80, 102).

[245] H. Wang, X. Zhang, Y. Hu, Y. Yang, X. Cao, and X. Zhen. "Few-shot semantic segmentation with democratic attention networks". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*. Springer. 2020, pp. 730–746 (cit. on p. 31).

[246] K. Wang, Y. Lin, B. Weissmann, M. Savva, A. X. Chang, and D. Ritchie. “PlanIT: planning and instantiating indoor scenes with relation graph and spatial prior networks”. In: *ACM Trans. Graph.* 38 (2019), 132:1–132:15 (cit. on pp. 59, 61, 62).

[247] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng. “Panet: Few-shot image semantic segmentation with prototype alignment”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9197–9206 (cit. on p. 31).

[248] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage. “Federated Evaluation of On-device Personalization”. In: *arXiv:1910.10252 [cs, stat]* (Oct. 2019). arXiv: 1910.10252 (cit. on p. 18).

[249] Q. Wang and K. Chen. “Multi-label zero-shot human action recognition via joint latent ranking embedding”. In: *Neural networks* 122 (2020), pp. 1–23 (cit. on p. 14).

[250] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. “High-resolution image synthesis and semantic manipulation with conditional gans”. In: *CVPR*. 2018, pp. 8798–8807 (cit. on pp. 80, 97, 101, 102).

[251] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612 (cit. on p. 113).

[252] S. Wasserman and P. Pattison. “Logit Models and Logistic Regressions for Social Networks: I. An Introduction to Markov Graphs and p*”. In: *Psychometrika* 61 (Sept. 1996), pp. 401–425 (cit. on p. 62).

[253] H. Wei and P. Peng. “The Segmentation of Retinal Layer and Fluid in SD-OCT Images Using Mutex Dice Loss Based Fully Convolutional Networks”. In: *IEEE Access* (2020) (cit. on p. 46).

[254] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari. “Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 7515–7525 (cit. on pp. 61, 102).

[255] L Wynants, R. Riley, D Timmerman, and B Van Calster. “Random-effects meta-analysis of the clinical utility of tests and prediction models”. In: *Statistics in medicine* 37.12 (2018), pp. 2034–2052 (cit. on p. 27).

[256] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. “Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.9 (2018), pp. 2251–2265 (cit. on p. 14).

[257] H. Xiao, K. Rasul, and R. Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017) (cit. on p. 21).

[258] X. Xing, T. Han, R. Gao, S.-C. Zhu, and Y. N. Wu. “Unsupervised disentangling of appearance and geometry by deformable generator network”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10354–10363 (cit. on p. 103).

[259] D. Xu, Y. Zhu, C. Choy, and L. Fei-Fei. “Scene Graph Generation by Iterative Message Passing”. In: *CVPR*. 2017 (cit. on pp. 59, 61, 69, 80, 102, 105).

[260] J. Xu and F. Wang. “Federated Learning for Healthcare Informatics”. In: *arXiv preprint arXiv:1911.06270* (2019) (cit. on p. 18).

[261] X. Yan, J. Yang, K. Sohn, and H. Lee. “Attribute2image: Conditional image generation from visual attributes”. In: *ECCV*. 2016, pp. 776–791 (cit. on p. 101).

[262] B. Yang, C. Liu, B. Li, J. Jiao, and Q. Ye. “Prototype Mixture Models for Few-Shot Semantic Segmentation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 763–778 (cit. on p. 31).

[263] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh. “Graph r-cnn for scene graph generation”. In: *ECCV*. 2018, pp. 670–685 (cit. on pp. 61, 102).

[264] Y. Yang, Z. Feng, M. Song, and X. Wang. “Factorizable Graph Convolutional Networks”. In: *Advances in Neural Information Processing Systems* 33 (2020) (cit. on p. 103).

[265] Z. Yang, Z. Qin, J. Yu, and Y. Hu. *Scene Graph Reasoning with Prior Visual Relationship for Visual Question Answering*. 2019. arXiv: 1812.09681 [cs.MM] (cit. on p. 61).

[266] S. Yao, T. M. Hsu, J.-Y. Zhu, et al. “3D-aware scene manipulation via inverse graphics”. In: *NeurIPS*. 2018 (cit. on p. 102).

[267] Y. Yeganeh, A. Farshad, P. Weinberger, S.-A. Ahmadi, E. Adeli, and N. Navab. “DIAMANT: Dual Image-Attention Map Encoders For Medical Image Segmentation”. In: *arXiv preprint arXiv:2304.14571* (2023) (cit. on p. 9).

[268] Y. Yeganeh, A. Farshad, J. Boschmann, R. Gaus, M. Frantzen, and N. Navab. “FedAP: Adaptive Personalization in Federated Learning for Non-IID Data”. In: *Distributed, Collaborative and Federated Learning (DeCaF)*. Springer, 2022 (cit. on pp. 1, 9, 15).

[269] Y. Yeganeh, A. Farshad, N. Navab, and S. Albarqouni. “Inverse distance aggregation for federated learning with non-iid data”. In: *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*. Springer, 2020, pp. 150–159 (cit. on pp. 2, 9, 15, 32).

[270] Y. Yeganeh, A. Farshad, and N. Navab. “Shape-Aware Masking for Inpainting in Medical Imaging”. In: *arXiv preprint arXiv:2207.05787* (2022) (cit. on p. 1).

[271] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do. “Semantic image inpainting with deep generative models”. In: *CVPR*. 2017, pp. 5485–5493 (cit. on p. 101).

[272] G. Yin, L. Sheng, B. Liu, et al. “Zoom-net: Mining deep feature interactions for visual relationship recognition”. In: *ECCV*. 2018, pp. 322–338 (cit. on p. 102).

[273] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec. “GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models”. In: *ICML* (2018). arXiv: 1802.08773 [cs.LG] (cit. on pp. 60–62, 69).

[274] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. “Free-Form Image Inpainting With Gated Convolution”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 (cit. on p. 101).

[275] L. Yue, D. Tian, W. Chen, X. Han, and M. Yin. “Deep learning for heterogeneous medical data analysis”. en. In: *World Wide Web* 23.5 (Sept. 2020), pp. 2715–2737 (cit. on p. 18).

[276] M. D. Zeiler and R. Fergus. “Visualizing and understanding convolutional networks”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*. Springer. 2014, pp. 818–833 (cit. on p. 7).

[277] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi. “Neural motifs: Scene graph parsing with global context”. In: *CVPR*. 2018, pp. 5831–5840 (cit. on pp. 61, 102, 105).

[278] B. Zeno, I. Kalinovskiy, and Y. Matveev. “IP-GAN: learning identity and pose disentanglement in generative adversarial networks”. In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 535–547 (cit. on p. 103).

[279] F. Zhan, H. Zhu, and S. Lu. “Spatial Fusion GAN for Image Synthesis”. In: *arXiv preprint arXiv:1812.05840* (2018) (cit. on p. 101).

[280] H. Zhang, T. Xu, H. Li, et al. “Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks”. In: *ICCV*. 2017, pp. 5907–5915 (cit. on pp. 80, 97, 101).

[281] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *CVPR*. 2018 (cit. on pp. 111, 113).

[282] R. Zhang, T. Che, Z. Ghahramani, Y. Bengio, and Y. Song. “MetaGAN: An Adversarial Approach to Few-Shot Learning.” In: *NeurIPS* 2 (2018), p. 8 (cit. on p. 81).

[283] Y. Zhang, A. Khakzar, Y. Li, A. Farshad, S. T. Kim, and N. Navab. “Fine-grained neural network explanation by identifying input features with predictive information”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 20040–20051 (cit. on p. 2).

[284] Z. Zhang, Y. Xie, and L. Yang. “Photographic text-to-image synthesis with a hierarchically-nested adversarial network”. In: *CVPR*. 2018 (cit. on pp. 80, 97, 101).

[285] B. Zhao, L. Meng, W. Yin, and L. Sigal. “Image generation from layout”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8584–8593 (cit. on pp. 80, 97, 101, 102).

[286] B. Zhao, B. Chang, Z. Jie, and L. Sigal. “Modular generative adversarial networks”. In: *ECCV*. 2018 (cit. on p. 101).

[287] Y. Zhao, B. L. Price, S. Cohen, and D. Gurari. “Guided Image Inpainting: Replacing an Image Region by Pulling Content From Another Image”. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 1514–1523 (cit. on p. 101).

[288] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. “Federated learning with non-iid data”. In: *arXiv preprint arXiv:1806.00582* (2018) (cit. on pp. 18, 28).

[289] Y. Zhou, Z. While, and E. Kalogerakis. “SceneGraphNet: Neural Message Passing for 3D Indoor Scene Augmentation”. In: *IEEE Conference on Computer Vision (ICCV)*. 2019 (cit. on p. 61).

[290] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. “Generative visual manipulation on the natural image manifold”. In: *ECCV*. 2016, pp. 597–613 (cit. on p. 101).

[291] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. “Unpaired image-to-image translation using cycle-consistent adversarial networks”. In: *ICCV*. 2017 (cit. on pp. 80, 101).

List of Figures

List of Tables