Neuro-symbolic Image Manipulation through Natural Language Commands

by

Udoy Saha
23341134
Kazi Tahmid Hossain
24241176
Ashraful Alam Patwary
21301416
Md Nafis Sadique Niloy
23341106
Md Faisal
21301620

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
BRAC University
June 2025

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Udoy Saha
23341134

Kazi Tahmid Hossain
24241176

Ashraful Alam Patwary
21301416

Md Nafis Sadique Niloy
23341106

Md Faisal
21301620

Approval

The thesis titled "Neuro-symbolic Image Manipulation through Natural Language Commands" submitted by

1. Udoy Saha - 23341134
2. Kazi Tahmid Hossain - 24241176
3. Ashraful Alam Patwary - 21301416
4. Md Nafis Sadique Niloy - 23341106
5. Md Faisal - 21301620

of Spring, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on June 20, 2025.

Examining Committee:

Supervisor:
(Member)

Dr. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, Ph.D
Associate Professor
Department of Computer Science and Engineering
BRAC University

Abstract

The task of image manipulation via natural prompts is very useful across multiple AI domains. Our research proposes a novel neuro-symbolic approach to address the challenge of detailed and context-aware image manipulations. Even though traditional neural network models are proficient in handling raw image data, they usually lack interpretability and reasoning abilities. We propose a model combining symbolic concepts with neural networks, drawing inspiration from existing neuro-symbolic models NeuroSIM [1] and SIM-SG [2]. While most of these models work with relatively simpler datasets, we wish to train our model to handle more complex real world data. Our methodology involves training our model with image data along with natural language prompts. The system interprets and manipulates the images based on user descriptions and manipulation commands. Using Scene Graph generation and manipulation techniques, our proposed model aims to outperform existing models in terms of complexity and interpretability. This approach has many potential applications in improving content generation. By advancing the capabilities of neuro-symbolic AI in the context of image manipulation, we wish to set a new benchmark for accuracy and clarity in the field.

Keywords: *Neuro-symbolic AI, Image Manipulation, Natural Language Commands, Scene Graph*

Table of Contents

Chapter 1

Introduction

The field of image manipulation has taken major steps with the arrival of artificial intelligence (AI), and more specifically, in cases in which natural language commands are used to modify images. This capability is highly valuable for applications requiring precise image variations, such as content creation, data augmentation, and visual processing in AI systems. Traditional neural network models are effective in processing raw image data. However, they often struggle with complex reasoning and a lack of interpretability. This creates a gap in generating meaningful and context-aware image modifications.

On the other hand, symbolic AI, also known as rule-based AI, works on the basis of explicit knowledge representation. It represents concepts using symbols and manipulates them using logical principles. This technique enables high degrees of interpretability and reasoning since the rules and symbols may be directly understood and modified by people. It excels at tasks requiring precise logic, such as solving algebraic equations, planning, and some forms of language processing. However, its rigidity and dependence on predefined rules make it less successful when dealing with the diversity and complexity of raw image data. Combining both of these approaches develops neuro-symbolic models that combine the best of both worlds. Our research focuses on establishing a neuro-symbolic method for complex real world image modification using natural language commands. This hybrid technique combines neural networks data processing skills with symbolic AI's reasoning capabilities. We gain inspiration from SIMSG [2], as well as NeuroSIM [1], which are based on the Neuro-Symbolic Concept Learner (NSCL) [3], two neuro-symbolic image manipulation models. Our goal is to outperform these existing models by incorporating innovative methodologies and broadening the field of applicability to real images.

1.1 Motivation

The demand for precise and context-aware image manipulation tools is growing, especially in fields like AI-driven content generation and data augmentation. Traditional image manipulation models often fail to capture the significant and detailed aspects required for these tasks, limiting

their effectiveness in real-world scenarios. To address these challenges, our research focuses on using scene graphs, which is a method that represents images as a set of objects and their relationships. The scene graphs convert an image into a structured representation, with nodes representing objects and edges representing the connections or relations between them. This structured approach allows for a more controlled and detailed manipulation of images. It not only provides a foundation for manipulating images but also offers the ability to make complex manipulations with precision, making them ideal for tasks where exact control over the content and relationships within images is very crucial. By using this neuro-symbolic approach, we aim to develop a model that offers greater flexibility and accuracy than existing solutions. Our research is driven by the need to advance current image manipulation techniques and apply them to more complex and diverse datasets.

1.2 Problem Statement

Generating realistic and meaningful variations or manipulations of an image remains a significant challenge in the fields of image manipulation and computer vision. The current neural network-based models can handle raw image inputs but lack the reasoning abilities to ensure that the modifications are meaningful and contextually accurate. Conversely, symbolic AI models can do well at reasoning and in providing clear explanations, but struggle with the complexity and variability of real-world images. To fill this gap, there is a need to develop an approach that can complement the characteristics of the neural networks where it is weak and at the same time take advantage of the symbolic AI approach because of its interpretability and ability to reason. So our research proposes a neuro-symbolic approach that combines the interpretability of symbolic AI with the data processing strengths of the neural networks.

Our research also addresses several problems in the field of image manipulation and generation. One of the key challenges lies in effectively converting the scene graphs into actionable data for image manipulation, which represents the objects and their relationships in an image. Additionally, the existing model architectures require so many improvements. So we need to develop a model that is both faster and more precise than the existing models, like Neuro-SIM and SIM-SG.

1.3 Research Objective

To tackle the mentioned research problems, our research will focus on the following objectives:

- **Developing a Novel Neuro-Symbolic Model:**

    1. Developing a new model that builds on the principles of the Neuro-Symbolic Concept Learner (NSCL) and NeuroSIM, and

also improving it by incorporating advanced scene graph manipulation using the ideas from SIM-SG.

2. Making sure that the model can handle complex scenes with multiple objects and connections, and is able to solve multi-step reasoning tasks.

- **Enhancing Model Interpretability and Generalizability:**

1. Ensuring the model's manipulations that are performed are easy to understand and noticeable, providing clear insights into the changes made to the source image.

2. Testing the robustness and reliability of the model while working with new, unseen images and instructions in a practical application.

By achieving these objectives, our research aims to advance the field of image manipulation and variation generation. We also hope to set a new benchmark for the accuracy and interpretability in image manipulation and variation generation.

Chapter 2

Literature Review

2.1 Background

In this section, we will be providing an overview of the foundational concepts and technologies that are crucial to this research.

2.1.1 Neuro-Symbolic Approaches

We review the paper by Garcez et al. [4] to understand the concept of Neuro-Symbolic AI, where they talk about the integration of neural networks and symbolic reasoning and have referred to Neuro-Symbolic AI as the '3rd Wave' of Artificial Intelligence. They do note that while deep learning has been great or has achieved great success in fields like computer vision and NLP, it still has problems with explainability and robustness. To address these problems, they propose a hybrid approach. They combine the adaptive nature of neural networks with the structured knowledge of symbolic reasoning. They introduce Logic Tensor Networks as a key technology for improving interpretability and learning efficiency by fusing first-order logic into neural networks. They also point out other advancements that are crucial for pushing neuro-symbolic AI further. Like variable grounding, symbol manipulation, and combinatorial reasoning. This hybrid method is especially promising for applications requiring transparency and complex reasoning, with some of the most notable being neuro-symbolic image manipulation, which is highly relevant for the purposes of our research.

The authors in [5] discuss the integration of deep learning and common sense knowledge through neuro-symbolic methods to enhance scene representation as well as visual reasoning. It shows the process of scene graph generation, which identifies and analyzes objects, visual relationships, and attributes in images to create symbolic scene representations. The paper shows the benefits of using common-sense knowledge to improve the accuracy and reasoning abilities of these models through heterogeneous knowledge graphs. Various approaches to neuro-symbolic integration, from loose to tight coupling of neural and symbolic modules, are reviewed by the authors of this paper. The paper categorizes methods based on deep learning structures, sources of common sense knowledge, and the degree of neuro-

symbolic usage. It also addresses visual reasoning tasks, datasets, evaluation metrics and key challenges. The authors offer a complete overview of the field and encourage further advancements in neuro-symbolic scene representation and visual reasoning.

2.1.2 Scene Graph Generation and Manipulation

The authors in [6] offer an extensive review of scene graph generation by underlining its importance in semantic representation and scene understanding. They thoroughly analyze 138 works and categorize image-based scene graph generation methods based on feature representation as well as refinement strategies. Their survey shows the development of visual relationship detection and the generation of scene graphs. The methods for scene graph generation include both simple models and complex and context-aware approaches. The paper mentions challenges like the need for large annotated datasets and also provides ways to solve these issues. The authors also focus on the creation of models that can generalize across many different domains. They suggest several future research directions. These include integrating scene graph generation with other computer vision tasks and implementing unsupervised learning methods for this task. The authors in [5] present different methods for scene graph generation (SGG). They focus on creating structured representations of scenes by identifying and categorizing objects and their attributes, as well as relationships between objects. This begins with detecting objects and classifying their attributes. This forms the base for understanding the visual elements in the scene. Contextual analysis and multimodal feature learning are used for analyzing the context of object appearances and predicting visual relationships using techniques such as Visual Translation Embedding Network and Iterative Message Passing. Then this information is used to construct and refine scene graphs using methods such as Scene Graph Convolutional Networks (SceneGCN) and Scene Graph Refinement Networks (SGR). Advanced methods use neuro-symbolic integration and common sense knowledge to increase the reasoning abilities of SGG models. Using heterogeneous knowledge graphs provides additional context that helps scene understanding. Specific tasks benefit from advanced techniques such as the Graphhopper method for multi-hop reasoning. The success of these methods is confirmed through evaluating and benchmarking against standard datasets and metrics. Together, these methods allow for accurate and detailed, contextually rich scene representations. The authors of NeuroSIM [1] introduced a novel "change" module in their manipulation network. This module is designed to change the attributes of objects in the scene graph that was generated from the input image. Dedicated neural networks are used for each attribute. These networks take the object embedding and the new attribute embedding and produce an updated object embedding that matches the change dictated by the functional program that was generated from the natural language prompt. For example, changing an object's color requires the network to receive the current object embedding and the embedding of the new color. Then, it gives an updated object embedding

with the applied new color. The module keeps the scene graph consistent and applies the new attribute values correctly without changing anything else. The change networks are trained with various loss functions to create accurate transformations. When an instruction is received, the system identifies the object in the scene graph and applies the appropriate neural network to update the object's embedding. Then it maps this change to create a consistent and accurate scene graph.

2.1.3 Generative Adversarial Networks (GAN)

Iglesias et al. [7] detail the methodology of Generative Adversarial Networks (GANs) in their survey paper. GANs basically consist of a generator and a discriminator. The generator generates data samples from random noise, ideally indistinguishable from real data, while the discriminator will then assess the generated samples to tell the real ones from the fakes. Both networks are trained simultaneously, where the generator minimizes the discriminator's ability to detect fake data and the discriminator maximizes its accuracy. The generator's loss is calculated based on the discriminator's probability of generated samples being real, and the discriminator's loss is based on its classification of both real and generated samples. The process is trained by optimizing the discriminator on real and fake data; tweaking the generator to produce output that becomes ever more realistic serves as a cycle that continues until the output is similar to that of real data.

2.1.4 Conditional Image generation

Conditional image generation has made significant progress thanks to deep generative models like Generative Adversarial Networks (GANs) [7] and Variational Autoencoders (VAEs) [8]. These models, in their basic sense, generate images based on defined parameters of input; hence, they offer more control over the resultant content compared to their unconditional variants, which generate images based on random noise.

The main point of conditional image generation is to generate images by modeling their distribution under some previous input. In general, such inputs or *conditions*, take many different forms depending on the task at hand. For example, conditional models are used for tasks like super-resolution, where a low-resolution image is transformed into a high-resolution one, or inpainting, where missing parts of an image are filled in. These models can also denoise images by removing noise from corrupted inputs.

2.1.5 Image manipulation

Unconditional image synthesis remains challenging, especially for complex scenes. In contrast, image manipulation focuses on editing specific parts of an image, leading to higher-quality results. Traditionally, this has been applied to object-centric tasks, such as face editing using attributes, or

manual tools like paintbrushes. Image composition, which combines objects into new scenes, often faces the difficulty of separating appearance from geometry. Generative models are frequently used for tasks like inpainting, object removal, and interactive editing via semantic maps [9]. Our approach will introduce a semi-automatic method that uses a single model to handle various edits, leveraging scene graphs to modify objects and their relationships. This approach will allow for high-level semantic edits without the need for paired data, addressing both object deformations and relational changes [10][11].

2.1.6 Image Generation From Scene Graph

Johnson et al. [11] introduce a method for generating images from scene graphs. They focus on its ability to reason about objects and their relationships. The authors highlight the limitations of previous methods, such as StackGAN which generates images from text descriptions but doesn't have the structured reasoning given by scene graphs. Their model uses graph convolutional networks to process input scene graphs and predict object layouts. They use a Cascaded Refinement Network (CRN) to transform these layouts into detailed images. The authors show that their method outperforms StackGAN in generating images that are more meaningful and contain a higher number of objects. The paper also includes various evaluations to measure the success of the proposed method. A user study that compared images generated by this method and StackGAN showed that most users liked images produced by the new method more.. The paper also shows the importance of different components of the model by making use of ablation studies. Their work is evaluated through experiments on datasets like Visual Genome [12] and COCO-Stuff [13]. The authors demonstrate that their approach shows an advancement in image generation by creating more complex and accurate visual scenes using structured scene graphs.

2.1.7 Graph Neural Networks (GNN)

Graph Neural Networks (GNNs) have gained popularity as an effective type of model for perceptual reasoning tasks based on relations over structured input, such as scene graph generation and manipulation. For the purpose of image understanding, GNNs make perfect sense as images can be viewed as scene graphs which structures wherein nodes are objects and edges are relationships (e.g., spatial, semantic, or contextual). A message passing scheme among nodes is permitted through GNNs, so that the model can learn how objects are affecting each other in an image. For example, the identity of, or placement of, an object, might be inferred based on its neighbors (e.g., a "cup" is usually "on" a "table"). This context is important when considering how to manipulate or reconstruct images. By applying multiple layers of graph convolution, the network refines object and relation embeddings iteratively, making the feature representation richer and more informative.

In semantic image manipulation systems such as SIM-SG, GNNs are employed to encode the layout information of the image, which is later passed to the rendering stage. The embedding for each object is updated not only by the appearance of the object but also in consideration of its role in the relational context of the scene graph. This leads to a more coherent and semantically-aware image synthesis. This graph-based model contributes to the performance gain, especially for multi-object or relation-rich edits.[2]

2.2 Related Work

This section focuses on recent works related to neuro-symbolic image manipulation, scene graph generation, and manipulation.

2.2.1 Generative Adversarial Networks (GAN) based models

Based on the Generative Adversarial Networks, the authors in [14] introduce a conditional recurrent GAN model for the GeNeVA task. This model has a Teller that provides drawing instructions to a Drawer through a sequential process. The generator creates images based on these instructions and the discriminator evaluates the quality of the generated images The generator outputs new images based on conditions derived from instructions and previously generated images. The discriminator uses self-attention and spectral normalization to differentiate real and generated images. It incorporates various loss functions for stability. Both generator and discriminator parameters are updated in every step by using normalization techniques for better training. This approach creates a framework for generating and refining images based on sequential instructions.

Similarly, Zhang et al. [15] presents TIM-GAN, which is also a GAN-based approach for manipulating images using text instructions. This model uses text as neural operators to locally modify image features. The work starts with extracting features from both the reference image and the instruction. The text instruction is encoded into two parts. The spatial part indicates where to edit and the operational part describes how to edit. A spatial mask is generated to identify the relevant region of the image, and a text-adaptive network creates transformations for the image feature according to the instruction. Then the image feature is modified by combining the original and transformed features through a gating mechanism. Finally, the modified image feature is used by a GAN generator to produce the manipulated image. This method allows for accurate image manipulation which is validated by extensive experiments.

2.2.2 Neuro-Symbolic Approach

In recent years, several advancements have been made in the domain of neuro-symbolic models and scene graph-based image manipulation. Key research efforts have focused on improving the interpretability and generalizability of these models, but limitations remain, especially when applied to real-world datasets. Neuro-Symbolic Concept Learner (NS-CL) [3] introduced a framework that integrates visual perception, language, and symbolic reasoning without requiring explicit supervision. NS-CL employs a neural-based perception module to extract object-level representations from images using a pre-trained Mask R-CNN and ResNet-34. These representations are then utilized by a visually-grounded semantic parser to translate natural language into executable symbolic programs. NS-CL achieves high accuracy ($\approx 99\%$) in identifying visual attributes like color, shape, and size on the CLEVR[16] dataset while using less than 10% of the data. It also demonstrates strong generalization to more complex scenes than those seen during training. The key strengths of NS-CL are its data efficiency, high performance on compositional generalization tasks, and transparency through symbolic execution, which makes its reasoning process interpretable. Despite its effectiveness in controlled settings, NS-CL is less suited for real-world scenarios. It lacks scalability for complex image manipulation tasks that involve intricate textures and dynamic scenes, limiting its application beyond visual question answering (VQA). Also, NS-CL's focus on synthetic datasets like CLEVR restricts its use in more complex, real-world image manipulation tasks. It also lacks the ability to directly modify images based on language prompts, which is crucial for interactive applications.

Later Neuro-SIM [1] extends NS-CL framework by introducing a weakly-supervised neuro-symbolic framework for image manipulation using multi-hop natural language instructions. Neuro-SIM parses multi-hop language instructions into symbolic programs, which guide the manipulation of multi-object scenes through a scene graph. This enables the system to perform tasks like adding, removing, and changing objects, without the need for paired image manipulations during training. So, by leveraging weak supervision through VQA annotations, it avoids the need for extensive labeled training data.

Neuro-SIM achieves competitive performance with other models using minimal supervision. It handles complex tasks such as adding, removing, and changing objects in multi-object scenes and generalizes well to unseen scenarios. Neuro-SIM combines flexibility, interpretability, and scalability, making it suitable for real-world applications. Its use of weak supervision reduces the cost of training and makes it adaptable to various tasks. Despite its improved scalability, Neuro-SIM still relies heavily on scene graphs, which can limit the fidelity of image synthesis in scenarios with intricate textures or highly detailed regions. The heavy reliance on scene graphs continues to pose challenges in generating high-quality images for complex

visual settings. Enhancing the image generation process while maintaining interpretability remains an open challenge.

Separately, SIM-SG (Semantic Image Manipulation using Scene Graphs) [2] enables image manipulation based on semantic content without direct supervision, utilizing scene graphs to represent objects and their relationships. The process begins with generating a scene graph from the input image, where objects are represented as nodes and relationships as edges, with each object described by its category, bounding box, and visual features. A state-of-the-art model, such as F-Net, is used to predict these graphs, capturing both semantic relationships and spatial or visual details, ensuring that object identity and appearance are preserved, even when modified. Users can alter object categories, positions, or relationships, which are processed by the Spatio-Semantic Scene Graph Network (SGN). This network uses graph convolutions to allow information to flow between objects, transforming node and edge features through multi-layer perceptrons (MLPs) to produce latent representations for each object, including its bounding box and visual attributes. These representations are then projected into a 2D spatial layout, where objects are positioned, and their regions are filled with features. The layout is aggregated into a single scene, and image synthesis is performed using architectures like Cascaded Refinement Network (CRN) or SPADE, which generate the final image. The synthesis is conditioned on the original image, incorporating low-level features and adding noise to occluded areas, allowing for flexible and detailed edits such as adding, removing, or modifying objects and relationships. So, SIM-SG performs well on tasks like object removal, replacement, and attribute modification, particularly in structured environments such as CLEVR and Visual Genome. Its key advantage lies in its flexibility, as it allows high-level semantic edits by manipulating scene graphs without needing paired training data for supervision.

However, SIM-SG struggles with image quality in complex real-world settings, particularly with highly detailed areas. The system is also predominantly object-centric, limiting its applicability in more abstract or dynamic scenes. While it is effective in structured environments, it faces scalability challenges in handling complex, real-world images where scene graphs may not fully capture the necessary visual details.

2.2.3 Diffusion Based Model

InstructPix2Pix is a diffusion-based image manipulation model that directly takes natural language instructions and images as input to generate corresponding edits. The model is trained on an extended version of the LAION-Aesthetics V2 6.5+ [17] dataset, using a process that combines large language models (LLMs) and text-to-image generation models. GPT-3 [18] generates edit instructions from captions and corresponding post-edit captions, the post-edit caption is passed to a text-to-image model based on stable diffusion [19] to generate an edited image. By creating an ex-

tended dataset of original images, edit instructions, and generated images, InstructPix2Pix learns to perform natural language-guided edits on real-world images. InstructPix2Pix shows strong performance in generating high-quality image edits directly from natural language inputs, thanks to its use of large-scale datasets and powerful generative models like stable diffusion. Despite its strengths, the model requires significant computational resources and large datasets to train effectively, which could limit its accessibility and adaptability in lower-resource settings. So, while InstructPix2Pix excels at generating high-quality edits, it lacks the interpretability and modularity provided by neuro-symbolic approaches like NS-CL, SIM-SG, and Neuro-SIM. Additionally, it does not focus on complex reasoning or multi-hop interactions between objects in a scene, which are essential for certain image manipulation tasks. The diffusion model [20] used here is a conditional diffusion model. It consists of a forward process and a backward process. In the forward process, Gaussian noise is added multiple times to the original image, and in the backward process, a denoising neural network is trained to remove noise from the image. For conditional diffusion, conditional information is added during the process so that the denoising network can generate images according to the condition.

Our Model aims to address the above mentioned model's gaps by integrating neuro-symbolic reasoning with advanced scene graph manipulation techniques for more precise, real-world image manipulations. Our approach enhances the granularity of scene graph representation to handle complex textures and detailed object attributes, ensuring high-fidelity image synthesis. We expand the scene graph capabilities to capture more abstract relationships and dynamic scene changes, improving scalability and real-world applicability. By focusing on natural language prompts for manipulation, our model also fills the gap left by the mentioned models in handling detailed, high-quality real-world images, providing greater flexibility and precision.

Chapter 3

Methodology

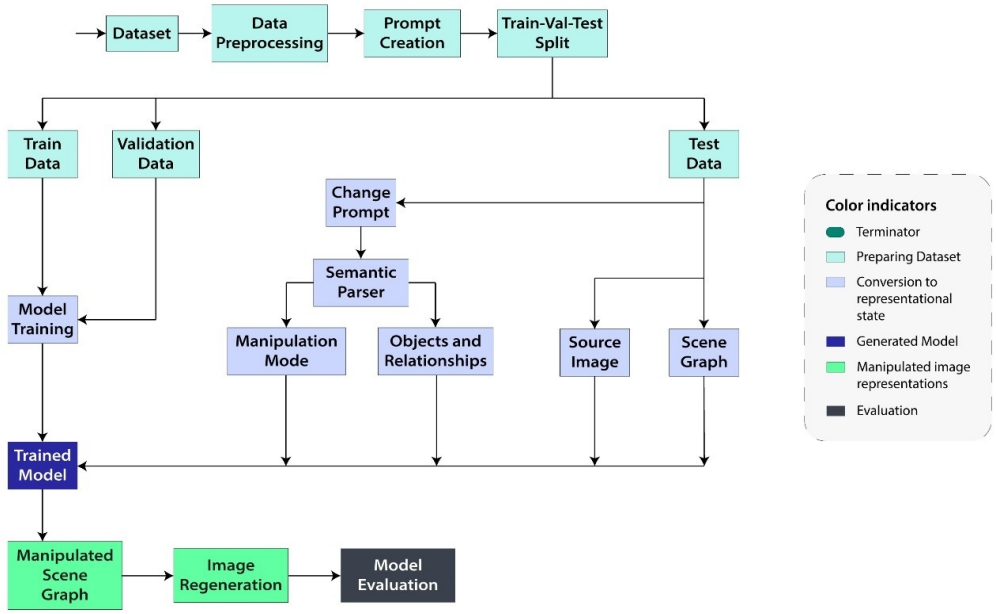

Figure 3.1: Top Level Overview

3.1 Data Preparation

The dataset might contain images with negligible resolution or very high resolution. Also, there might be unnecessary object annotations that don't need to be trained in order to achieve training efficiency. Therefore, the data needs to be processed before training.

3.1.1 Dataset Description

The Dataset we are using for our research is the Visual Genome [12] dataset, which is one of the most comprehensive resources for visual scene understanding, containing 108,249 real-world images densely annotated with objects, attributes, and relationships. Each image includes an average of 21 objects, 18 attributes, and 18 relationships between objects, allowing for a

rich and detailed representation of scene structure. The dataset emphasizes not only object detection but also the cognitive understanding of scenes by capturing how objects interact with one another. It includes over 4.1 million object instances (33,877 object categories), each localized with bounding boxes, and 1.7 million attribute annotations (68,111 attribute categories) that describe the visual characteristics of these objects. Additionally, the dataset contains 1.5 million annotated relationships (42,374 relationship categories) between objects, focusing on spatial and functional connections such as "man sitting on bench" or "dog chasing ball." Visual Genome also features region-based descriptions, with each image having an average of 42 localized descriptions to provide a more granular understanding of different parts of the scene. These region graphs are combined into comprehensive scene graphs that represent the overall image structure. Moreover, the dataset includes over 1.7 million question-answer pairs designed to support tasks such as Visual Question Answering (VQA) and cognitive reasoning.

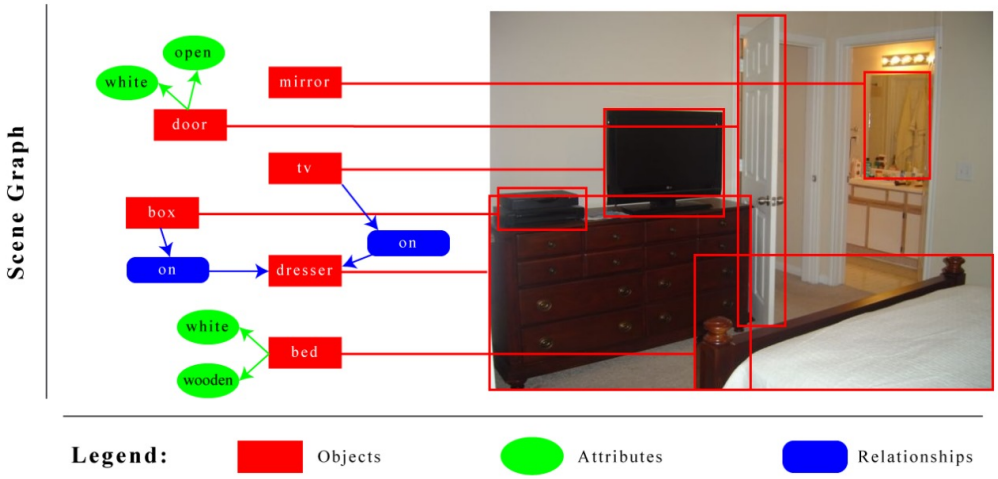

Figure 3.2: A representation of the Visual Genome dataset. Each image contains region descriptions that describe a localized portion of the image. Each region is converted to a region graph representation of objects, attributes, and pairwise relationships. Each of these region graphs is combined to form a scene graph with all the objects grounded to the image.

3.1.2 Data Preprocessing

We started by loading the image information from the image metadata, which provided a foundation for the dataset. During this initial step, we removed a total of 335 images from the training split due to them being too small. Similarly, we filtered out 45 images from the test split and 46 images from the validation split for the same reason.

Next, we loaded object and relationship aliases. This helped us standardize object and relationship names across the dataset. Then, we read the object data and the object vocabulary. This process involved analyzing 86,128 training images, which resulted in the discovery of 179 object categories

that had at least 2,000 instances. This ensures a robust representation of common objects in our dataset. We also loaded attributes and constructed an attribute vocabulary from the same set of training images, resulting in 80 attribute categories with over 2000 instances. We continued by loading relationships and identifying 46 relationship types that had at least 500 training instances. This is important because it provides a rich set of common relationships to model interactions between objects in images.

We removed 997,213 objects that were smaller than a minimum size of 32 pixels during the filtering process. We were left with 910,259 valid object instances after this.

As we encoded objects and relationships, we skipped some images for each split of the dataset. For the training split, we noted issues such as 16,402 images with too few relationships and 6,794 with too few objects, alongside some images having too many objects (187) or relationships (180). Similar statistics were gathered for the validation and test splits, ensuring a comprehensive understanding of the dataset's structure and potential issues.

After the filtering process, we were left with 62565 images for the training split, 5096 images for the testing split and 5062 images for the validation split.

We wrote the processed data into HDF5 output files for each dataset split. Each file contained datasets for image IDs, object IDs, object names, bounding boxes for objects, counts of objects and relationships per image, and various other metrics necessary for model training and evaluation.

Finally, we saved the vocabulary we created during the preprocessing. This vocabulary includes all the selected object names, relationship predicates, and attribute categories.

Table 3.1: train, test and validation split count

Split	**Count**
Train	62565
Test	5096
Val	5062

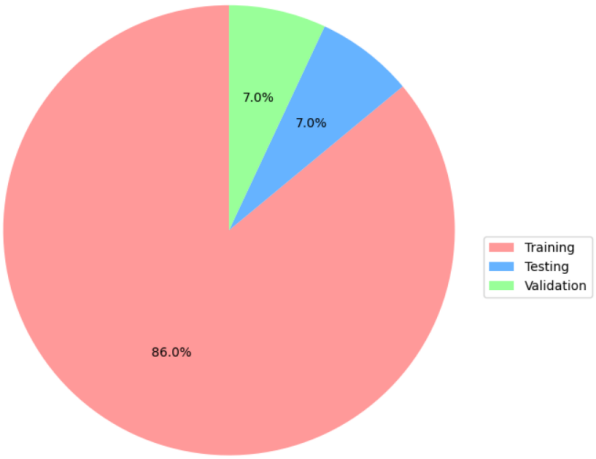

Figure 3.3: A pie chart of the dataset.

3.1.3 Natural Language Prompt Creation

By comparison with the original SIM-SG framework that relied on manual modification to scene graphs through direct structural manipulation, this work offers a more natural and scalable solution, which is image modification using natural language commands. This change requires modification commands that were dynamically interpretable and translatable into scene graph updates.

With this aim, we launched and carried out a crowdsourced annotation campaign, where participants were tasked with the responsibility to create contextually accurate natural language prompts that capture important changes to the scene described within each image. These prompts cover three categories of operations: removal, repositioning, and replacement of objects. Given the absence of a stable and coherent vocabulary in the original Visual Genome dataset, we carried out a thorough preprocessing step which included the removal of noisy, overly broad, and vague entities based on defined heuristics, leading to a more expressive and compact vocabulary better suited to subsequent reasoning and generation tasks.

Each image is associated with two to three unique natural language prompts, with a total of more than 120,000 high-quality annotations. These prompts were not only created by human annotators but also passed through moderation and review processes to ensure contextual validity. This effort yields a large novel set of human-generated natural language instructions for scene-level image manipulation, providing a strong foundation for testing our system on its capacity to modify scene graphs and generate images with natural language instructions.

Figure 3.4: Exmaple prompts

3.2 Model Architecture

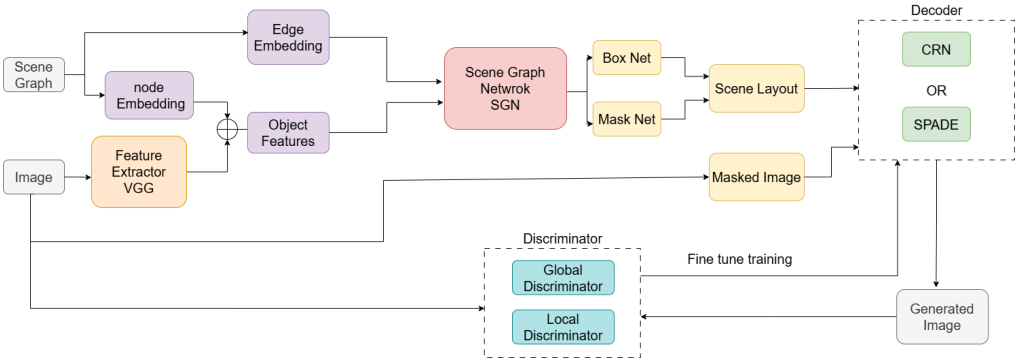

Figure 3.5: Training model architecture

3.2.1 VGG16: Feature Extractor

We used VGG16 for high-level feature extraction. VGG16 consists of five successive convolutional blocks, each adapted to capture increasingly higher-level visual features sequentially and down-sampling to decrease spatial resolution. Each of these blocks contains multiple 3×3 kernels and 1 pixel padding convolutional layers in order to keep the spatial size the same, followed by ReLU activation. Every block is then followed by a 2×2 max pooling layer with a stride of 2. The output channels of the five blocks are 64, 128, 256, 512, and 512. The first two blocks consist of two convolutional layers each, while the other three blocks consist of three convolutional layers each.

The last convolutional feature map is obtained after passing through the five blocks. This feature is passed to three fully connected layers and finally a softmax output layer. In our case, we drop the softmax layer and add a 128-dimensional fully connected layer. This modified VGG16 provides us with high-level visual features that we can pass as input to our SGN.

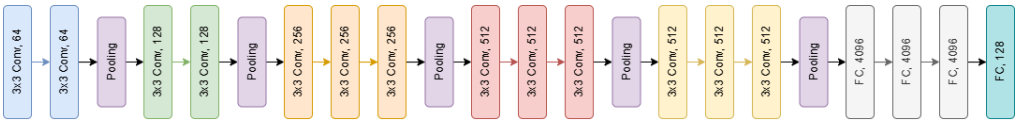

Figure 3.6: Architecture of VGG16

3.2.2 SGN Architecture

The Spatio-Semantic Scene Graph Network (SGN) is designed to process scene graphs and produce structured representations that can be used to synthesize images. It consists of five graph convolutional layers. Each object in the scene graph is represented by combining three components:

- A 128-dimensional embedding that encodes the object's semantic category.
- The object's bounding box, defined by its top, left, bottom, and right coordinates.
- A 128-dimensional visual feature vector, extracted from the corresponding cropped region of the image using a VGG-16 network, followed by a fully connected layer.

Relationships between objects (predicates) are also encoded as 128-dimensional embeddings.

To encourage robustness and simulate real-world edits during training, the model randomly removes (or *masks*) parts of the object information. Specifically, the visual features are hidden 25% of the time, and the bounding box coordinates are hidden 35% of the time. When either is masked, the corresponding region in the image is also occluded.

The SGN processes this information using graph convolutions. Each layer includes two components: one that updates edge features and one that updates node features. Both are implemented using two-layer neural networks with 512 hidden units and an output size of 128.

In the final layer, for each object, the SGN predicts:

- A 128-dimensional feature vector capturing the updated representation of the object.
- A binary spatial mask with a resolution of 16 by 16 pixels.
- The predicted bounding box coordinates, generated using a two-layer network with 128 hidden units.

These outputs are combined into a spatial layout that serves as an intermediate representation for image generation.

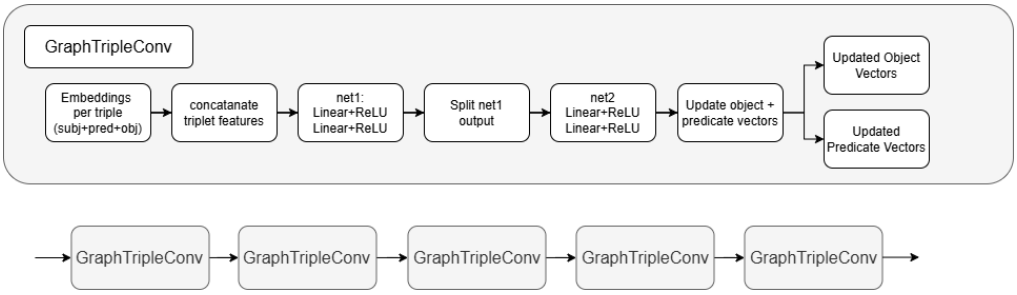

Figure 3.7: Architecture of SGN

3.2.3 Image Decoder Architectures

We employ two alternative image decoder backbones: the Cascaded Refinement Network (CRN) and SPADE.

CRN

CRN consists of five cascaded refinement modules. Each module includes two convolutional layers with 3×3 kernels, followed by batch normalization and Leaky ReLU activations. The output channel sizes of the modules are 1024, 512, 256, 128, and 64, respectively. The output of each module is concatenated with a progressively downsampled version of the initial input, which itself is the concatenation of the predicted layout and masked low-level image features. This architecture produces images at a resolution of 64×64.

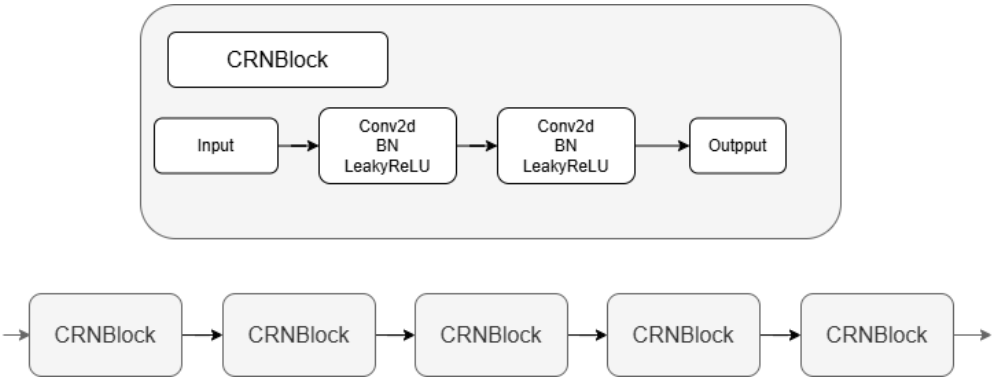

Figure 3.8: Architecture of CRN

SPADE

SPADE employs four residual blocks, with output channels set to 1024, 256, 128, and 64 across layers. In each block, the predicted layout modulates activations via spatially-adaptive normalization layers (SPADE), while the masked image features are concatenated to the intermediate activations. This architecture provides fine-grained spatial control over the generated content.

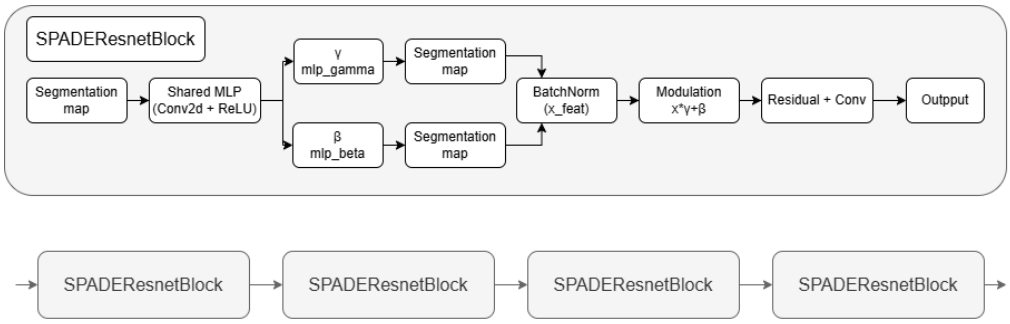

Figure 3.9: Architecture of SPADE

3.2.4 Discriminator

AcCropDiscriminator

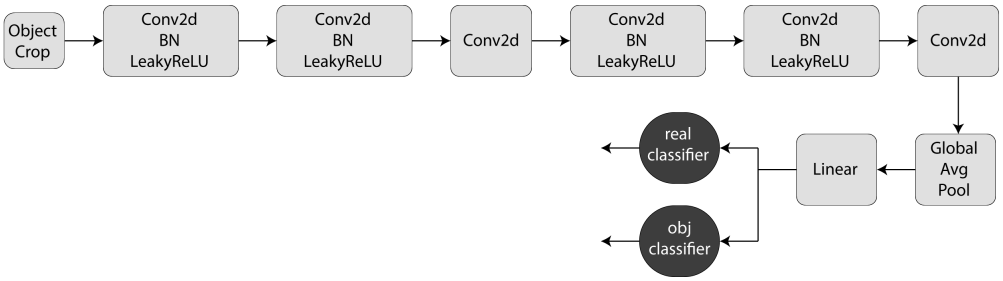

Figure 3.10: Architecture of AcCropDiscriminator

The `AcCropDiscriminator` takes image of dimension (3, H, w) and it consists of three convolutional layer. Batch normalization is applied in each layer and `LeakyReLU` is used as an activation function for the layers. The convolutional layers progressively downsamples the input image from (3, H, W) to (256, H/8, W/8) using 4×4 kernels and a stride = 2. Parallel CNN pathway repeats this CNN layers but uses `GlobalAvgPool` to reduce dimensions to 256, then to Linear (256 to 1024) to generate high dimensional embedding. This embedding is then passed to two types of classifier, `real_classifier` and `obj_classifier`. `real_classifier` has only one output (1024 to 1), indicating real or fake discrimination and `obj_classifier` has 179 outputs (1024 to 179) to identify 179 types of objects.

PatchDiscriminator

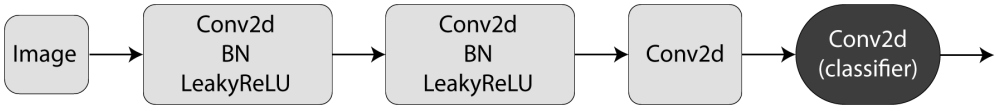

Figure 3.11: Architecture of PatchDiscriminator

The `PatchDiscriminator` takes image of dimension (3, H, W) and it consists of three layer CNN. For each layer, 4×4 kernel is used of stride = 2 for downsampling. Then `BatchNorm2d` and `LeakyReLu` is used. The convolutional layers progressively downsamples the input image from (3, H, W) to (256, H/8, W/8) feature map. Finally, to 1×1 classifier (256 to 1) to indicate real or fake discrimination.

3.2.5 Semantic Parser

The semantic parser contains three core modules: an operation classifier, a candidate extractor, and a fuzzy vocabulary matcher. The operation classifier uses a zero-shot classification pipeline BART Large MNLI in order to

identify the most likely operation indicated by the prompt among remove, replace, or reposition. The input prompt is tokenized with BERT Base Uncased tokenizer.

In order to map these candidates to pre-selected scene graph vocabulary, fuzzy matching is done using Sentence-BERT. Both the input tokens and the vocabulary entries are embedded, and cosine similarity is used to select the closest matches above a certain threshold. The output is a semantic representation in a structure that contains the mode of operation, matched objects, and relations. This design allows interpretable and flexible manipulation of scene graph components based on natural language inputs.

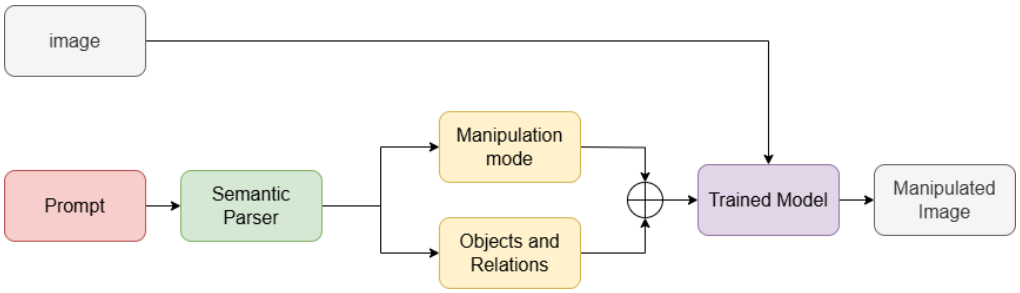

Figure 3.12: Semantic Parsing

Chapter 4

Implementation and Result Analysis

In this section, we present the training strategy of the model. Additionally, we run and evaluate an existing pretrained model and show the performance the model shows.

4.1 Model training

Training the proposed model in a fully supervised setting would require annotated quadruplets (I, G, G', I'), where:

- I is the original image,
- G is its scene graph,
- G' is the modified scene graph,
- I' is the corresponding modified image.

Since such data is difficult to acquire, we instead train the model using only pairs (I, G), and reformulate the problem as one of image and graph reconstruction. We synthetically generate pseudo-quadruplets $(\tilde{I}, \tilde{G}, G, I)$ by randomly masking information in the available data.

Object visual features and bounding box coordinates are masked with a specific probability. The corresponding image regions are also occluded before feature extraction. This essentially turns the problem into a kind of task similar to inpainting, making it a Semi-Supervised task.

If the scene graph is semantically edited based on the corresponding natural language prompts (like changing a predicate from "on' to "near"), we discard the original spatial coordinates of the affected nodes. The new positions are predicted by the model based on the rest of the scene. However, to preserve the visual identity of the object (like a rider), we don't change their visual embeddings.

Loss Functions

Bounding Box Regression Loss. We penalize incorrect bounding box predictions using an L1 loss:

$$L_b = \lambda_b \left\| x_i - \hat{x}_i \right\|_1 \tag{4.1}$$

Photometric Reconstruction Loss. To preserve content in image regions that are not edited, we apply an L1 loss over image pixels:

$$L_r = \lambda_r \left| I - I' \right\|_1 \tag{4.2}$$

Adversarial Loss. We employ two discriminators: a global discriminator D_{global} and a local object-level discriminator D_{obj}. The standard GAN loss is defined as:

$$L_{\text{GAN}} = \mathbb{E}_{q \sim p_{\text{real}}}[\log D(q)] + \mathbb{E}_{q \sim p_{\text{fake}}}[\log(1 - D(q))] \tag{4.3}$$

Here, q represents an image or region sampled from either the real data distribution p_{real} or the model-generated distribution p_{fake}.

Total Synthesis Loss. The overall synthesis loss combines all the components:

$$L_{\text{synthesis}} = L_b + L_r + \lambda_g \min_G \max_D L_{\text{GAN,global}} + \lambda_o \min_G \max_D L_{\text{GAN,obj}} \tag{4.4}$$

Here λ_g, λ_o are weighting factors.
These help improve semantic consistency and visual quality by aligning intermediate feature maps with those of real images.

Loss factor	Weight (CRN)	Weight (SPADE)
λ_g	0.01	1
λ_o	0.01	0.1
λ_b	10	50
λ_r	0.1	0.1

Table 4.1: Loss factor weights used for CRN and SPADE models.

All models are optimized using the Adam optimizer with a base learning rate of 10^{-4}. The batch size is set to 32 for 64×64 image resolution. During training, all object instances in a batch are processed simultaneously through the SGN, visual feature encoder, and discriminators.

4.2 Result Analysis

We evaluated our model into two parts, Reconstruction and Manipulation.

4.2.1 Reconstruction

We assessed the quality of reconstructed images using three standard metrics: Mean Absolute Error (MAE), Structural Similarity Index Measure (SSIM), Fréchet Inception Distance (FID). Each metric was computed both globally over the entire image and locally over the Region of Interest (RoI).

Mean Absolute Error (MAE)

MAE measures the average absolute difference between corresponding pixels in the reconstructed image and the ground truth image.

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} \left| I_i - \hat{I}_i \right| \tag{4.5}$$

Where:

- I_i is the pixel intensity at position i in the ground truth image.
- $\hat{I}_i$ is the pixel intensity at position i in the reconstructed image.
- N is the total number of pixels over which the error is computed (e.g., entire image or RoI).

Lower MAE indicates better pixel-wise reconstruction accuracy. It's a perceptually insensitive metric, which means that even if two images look very similar structurally but have small shifts in pixel values, MAE can still be high. Values range from 0 to 255.

Structural Similarity Index Measure (SSIM)

SSIM quantifies the perceptual similarity between two images based on structural information. It considers luminance, contrast, and structure.

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.6}$$

Where:

- x, y are corresponding local patches from the ground truth and reconstructed images.
- μ_x, μ_y are the mean pixel intensities of patches x and y, respectively.
- σ_x^2, σ_y^2 are the variances of the patches.
- σ_{xy} is the covariance between x and y.

SSIM values can range from -1 to 1, but are typically between 0 and 1. Here, 1 indicates perfect structural similarity. SSIM is more sensitive to changes in texture, edges, and local patterns. This makes it different from MAE, which is purely pixel-level.

Fréchet Inception Distance (FID)

The Fréchet Inception Distance (FID) is a metric used to evaluate the quality of images generated by generative models, particularly GANs. It compares the distribution of feature representations of real and generated images, extracted using a pre-trained Inception v3 model.

FID assumes that these feature representations follow a multivariate Gaussian distribution and calculates the Fréchet distance between the two distributions.

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{1/2}\right) \tag{4.7}$$

where:

- μ_r, Σ_r: mean and covariance of the real image features
- μ_g, Σ_g: mean and covariance of the generated image features

A lower FID score indicates that the generated images are more similar to the real ones in terms of visual quality and diversity.

Results

Table 4.2: Evaluation metrics for reconstruction methods under different modes.

Mode	**Decoder**	**MAE**	**MAE-RoI**	**SSIM**	**SSIM-RoI**	**FID**
Generative	SPADE	41.76	39.64	0.38	0.36	46.62
	CRN	42.04	40.54	0.34	0.37	86.27
RoI w/ Features	SPADE	8.61	21.59	0.87	0.61	7.27
	CRN	8.65	21.62	0.82	0.58	8.64
RoI w/o Features	SPADE	10.32	27.10	0.89	0.54	10.34
	CRN	10.37	27.12	0.86	0.51	12.74

Figure 4.1: Reconstruction result samples from our model

The findings show that both decoders perform similarly at this task. Global MAE measures are of moderate size with means of 41.76 for SPADE and 42.04 for CRN, showing a considerable average pixel error when reconstructed from abstract scene graph representations. MAE-RoI measures are slightly smaller (39.64 for SPADE, 40.54 for CRN). This shows that object-centric regions are reconstructed to the same or slightly better degree than the background. SSIM values remain quite low for both (0.38 for SPADE and 0.34 for CRN). This result shows the difficulty of reconstructing high-frequency structure and texture from scene graph representation alone. SSIM specific to RoI is slightly greater in both cases, which indicates better structure preservation in object regions.

When we only focus on regenerating the Regions of Interest, there is a significant improvement in reconstruction quality. When visual features are provided, SPADE and CRN both witness substantial reductions in MAE (8.61 and 8.65, respectively) and substantial increases in SSIM (0.87 and 0.82). MAE-RoI also substantially improves (21.59 for SPADE, 21.62 for CRN), indicating more accurate reconstruction in object-centric regions. SSIM-RoI also increases (0.61 and 0.58), indicating a more coherent structure where object existence is most critical. Even when visual features are not provided, the condition only slightly degrades, with increased MAE and decreased SSIM on both decoders. This result is expected for both cases since we're only reconstructing the RoI while leaving the rest of the image unchanged. Significantly, SPADE has visually somewhat higher quality in general, and more noticeably in RoI-specific SSIM which shows superior use of spatial conditioning mechanisms.

Generally, the result is not as good as pixel-level reconstruction or inpainting benchmarks, which benefit from spatial continuity. But with the abstraction and high-level input of the scene graph, these outcomes demonstrate a reasonable ability of both decoders to produce recognizable reconstructions. SPADE decoder produces images slightly better in perceptual quality in general. These results highlight the challenges and trade-offs of scene graph-based image reconstruction.

4.2.2 Manipulation

We evaluate the model's semantic image editing capability through three fundamental graph-level operations: object replacement, object removal, and relationship change. These operations are performed using the prompts associated with the images through the modification of the scene graph.

Despite the model's demonstration of some capability to respond to these changes, such as generating objects in accordance with spatial context or moving entities based on updated relationships, the outcomes are not consistent. In particular, object replacement is often problematic. Some images fail to properly blend the new object into the scene, resulting in visual artifacts, incorrect scaling, or irregular blending with the scene. This

shows the model's limitations in generalizing between classes while being coherent. Though object removal and relationship modification have shown improved performance on simpler scenes, they also face the same issues in complicated visual scenes. These are indications that while the strategy has potential for manipulating images using scene graphs, it still needs to be fine-tuned to generate consistently realistic and structurally stable results, especially for tasks involving complex semantic or visual transformation.

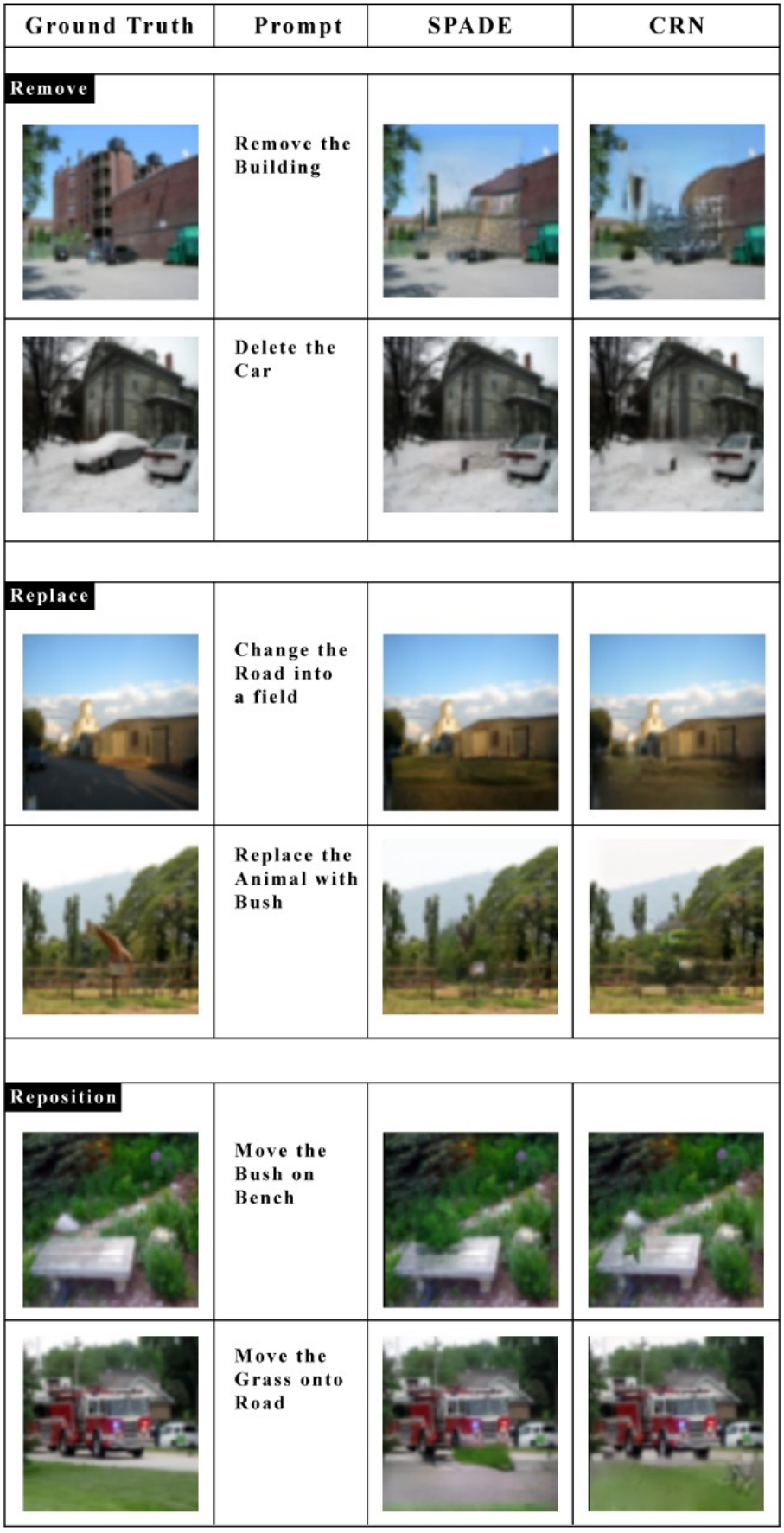

Figure 4.2: Manipulation result samples from our model

Chapter 5

Conclusion

We introduced the task of semantic image editing via scene graphs using natural language prompts and proposed an approach that enables editing image content and object relationships via graph-level operations derived from prompts that doesn't require original and target image pairings for training. We created a model that can successfully regenerate and manipulate images based on changes made to the scene graph. While our approach enables high-level image structure manipulation, qualitative and quantitative evaluation yields promising results. Overall, we have demonstrated that our system performs reasonably well on image synthesis and shown strong qualitative evidence of its effectiveness in modifying real-world images.

5.1 Research Limitation

5.1.1 Multi hop reasoning

The model can't perform multi-hop reasoning, which is a form of complex reasoning where a set of relational steps needs to be inferred (and operated) before the reasoning step. For example, the instruction to `Change the object the man standing next to the car is holding to be a black umbrella` would require the system to:

- Identify the `man` from a spatial relation (`near the car`),
- Determine what he is `holding`,
- Remove that object,
- Add a new object (`black umbrella`) in the correct position.

The model is currently specialized for single-hop or pairwise manipulation tasks. Using no dedicated symbolic planner or logic executor does not enable tracking complex dependencies across objects and their relational paths. This leads to a wrongly edited or no action if a multiple-step instruction is executed.

Also we have limited operations, `replace`, `remove`, `reposition`. Prompts without these operations can't be parsed.

5.1.2 Hardware Limitations

The model is trained on a fixed vocabulary of 61 object categories, 33 predicates, and 33 relationship types. Although this setup helps to improve the efficiency of training and keep a uniform structure of the scene graph, it has a rather rigid semantic constraint. Any input command involving words or phrases outside this predefined set, such as `rainbow`, `mirror`, `and glass table` is as a result unparseable by the PromptParser and untranslatable to the visual graph. Therefore, such commands will be ignored or leaked to the closest known entity as an incorrect command, which will cause semantics to be lost or execution to fail in manipulation.

This constraint limits the system's adaptability to open-domain prompts, making it less effective in scenarios where natural user language varies significantly or when domain-specific vocabularies are involved. In practical deployments, this rigidity could result in a poor user experience and missed manipulation opportunities.

5.2 Future Works

A key refinement is to loosen the rigid vocabulary constraints. The closed vocabulary of object classes, predicates, and relations in the present model limits the model's ability to generalize to open-domain language. In real-world scenarios, users will refer to entities or concepts outside the predefined vocabulary ("mirror" or "glass table"), leading to unrecognized or incorrect operations. Future work can explore the extension of vocabulary through external knowledge sources, dynamic grounding methods, or continuous learning mechanisms to enable the model to be more resilient to diverse and domain-specific language. These enhancements would significantly improve the system's robustness, allowing the system to be more flexible and user-friendly in practical applications.

A key avenue for future work is enabling multi-hop reasoning over the scene graph. Complex instructions that consist of chaining multiple spatial and semantic relations—e.g., describing an object through intermediate entities or operating on nested attributes—are beyond the capability of the current system. Incorporating a logic-based reasoning module can allow the model to reason over and execute these multi-step transformations more effectively. Additionally, generalizing the system to handle a broader range of operations beyond replace, remove, and reposition would allow it to be more expressive and convenient.

References

[1] H. Singh, P. Garg, M. Gupta, K. Shah, A. Goswami, S. Modi, A. K. Mondal, D. Khandelwal, D. Garg, and P. Singla, "Image manipulation via multi-hop instructions – a new dataset and weakly-supervised neuro-symbolic approach," 2023. [Online]. Available: https://arxiv.org/abs/2305.14410

[2] H. Dhamo, A. Farshad, I. Laina, N. Navab, G. D. Hager, F. Tombari, and C. Rupprecht, "Semantic image manipulation using scene graphs," 2020. [Online]. Available: https://arxiv.org/abs/2004.03677

[3] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision," 2019. [Online]. Available: https://arxiv.org/abs/1904.12584

[4] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic ai: The 3rd wave," 2020. [Online]. Available: https://arxiv.org/abs/2012.05876

[5] M. J. Khan, F. Ilievski, J. G. Breslin, and E. Curry, "A survey of neurosymbolic visual reasoning with scene graphs and common sense knowledge," *Neurosymbolic Artificial Intelligence*, vol. 1, pp. NAI–240 719, 2025. [Online]. Available: https://doi.org/10.3233/NAI-240719

[6] G. Zhu, L. Zhang, Y. Jiang, Y. Dang, H. Hou, P. Shen, M. Feng, X. Zhao, Q. Miao, S. A. A. Shah, and M. Bennamoun, "Scene graph generation: A comprehensive survey," 2022. [Online]. Available: https://arxiv.org/abs/2201.00443

[7] G. Iglesias, E. Talavera, and A. Díaz-Álvarez, "A survey on gans for computer vision: Recent research, analysis and taxonomy," *Computer Science Review*, vol. 48, p. 100553, May 2023. [Online]. Available: http://dx.doi.org/10.1016/j.cosrev.2023.100553

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

[9] 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 *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.

[10] 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," *ACM Trans. Graph.*, vol. 32, no. 6, Nov. 2013. [Online]. Available: https://doi.org/10.1145/2508363.2508381

[11] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1219–1228.

[12] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F.-F. Li, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," 2016. [Online]. Available: https://arxiv.org/abs/1602.07332

[13] H. Caesar, J. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1209–1218.

[14] A. El-Nouby, S. Sharma, H. Schulz, D. Hjelm, L. E. Asri, S. E. Kahou, Y. Bengio, and G. W. Taylor, "Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction," 2019. [Online]. Available: https://arxiv.org/abs/1811.09845

[15] T. Zhang, H.-Y. Tseng, L. Jiang, W. Yang, H. Lee, and I. Essa, "Text as neural operator:image manipulation by text instruction," in *Proceedings of the 29th ACM International Conference on Multimedia*, ser. MM '21. ACM, Oct. 2021, p. 1893–1902. [Online]. Available: http://dx.doi.org/10.1145/3474085.3475343

[16] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," 2016. [Online]. Available: https://arxiv.org/abs/1612.06890

[17] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, P. Schramowski, S. Kundurthy, K. Crowson, L. Schmidt, R. Kaczmarczyk, and J. Jitsev, "Laion-5b: An open large-scale dataset for training next generation image-text models," 2022. [Online]. Available: https://arxiv.org/abs/2210.08402

[18] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[19] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2022. [Online]. Available: https://arxiv.org/abs/2112.10752

[20] M. Chen, S. Mei, J. Fan, and M. Wang, "An overview of diffusion models: Applications, guided generation, statistical rates and optimization," 2024. [Online]. Available: https://arxiv.org/abs/2404.07771