YEDITEPE UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

GENERATION OF IMAGE SERIES FROM A SEQUENCE OF RELATED SHORT SENTENCES

A Thesis Submitted

by

Mehmet Ali Özer

In Partial Fulfillment

of the Requirements for the Degree of

Master of Science

in

Data Science

Supervisor

Assoc. Prof. Dr. Dionysis Goularas

Istanbul - 2024

GENERATION OF IMAGE SERIES FROM A SEQUENCE OF RELATED SHORT SENTENCES

by

Mehmet Ali Özer

Approved by:

Assoc. Prof. Dr. Dionysis Goularas
(Yeditepe University) ……
(Thesis Supervisor)

Dr. Narjes Nikzad
(TH Köln, Cologne University of Applied Sciences) ……
(Thesis Co-Supervisor)

Prof. Dr. Semih Bilgen
(Okan University) ……

Prof. Dr. Emin Erkan Korkmaz
(Yeditepe University) ……

DATE OF APPROVAL: …. / …. / 20….

DECLARATION OF ORIGINALITY

I hereby declare that this thesis is my own work and that all information in this thesis has been obtained and presented following academic rules and ethical conduct. I have fully cited and referenced all material and results as required by these rules and conduct, and this thesis study does not contain any plagiarism. The necessary permissions have been obtained if any material used in the thesis requires copyright. No material from this thesis has been used to award another degree.

To the best of my knowledge and belief, it contains no material previously published or written by another person nor material accepted for the award of any other degree except where due acknowledgment has been made in the text.

I accept all kinds of legal liability that may arise in cases contrary to these situations.

Mehmet Ali Özer

……

ABSTRACT

GENERATION OF IMAGE SERIES FROM A SEQUENCE OF RELATED SHORT SENTENCES

The aim of this study is to generate a series of coherent images from a sequence of related short sentences, effectively illustrating a short story. While generating an image from text is a well-researched yet open problem, this work extends the challenge by incorporating information from previous images to ensure narrative continuity and avoid unrelated scenes. The Pororo-SV Dataset is re-annotated using Chain of Thought (CoT) and few-shot prompting techniques by Gemini. Additionally, the Dynamic Storytelling Pororo-DS Dataset is introduced, prepared specifically for this work to capture key dynamics such as emotional changes, goal achievements, and social interactions, providing a basis for measuring coherence. The approach leverages a pre-trained Latent Diffusion Model (LDM), adapted through Low-Rank Adaptation (LoRA), to generate high-fidelity images from text prompts. A Gated Recurrent Unit (GRU) network-based Coherence Classifier ensures the narrative consistency of the generated sequences by selecting subsequent scenes based on the initial one. Evaluation metrics include the Structural Similarity Index (SSIM), Fréchet Inception Distance (FID), and custom character representation accuracy measures, such as Character Presence Accuracy (CPA), Duplication Character Rate (DCR), and Character Exact Match Accuracy (CEMA). The results demonstrate that the model achieves competitive SSIM and FID scores, with improved character presence and coherence, effectively selecting generated images that describe a short story. The Coherence Model using CLIP encodings for text and images outperformed BERT and ResNet models, indicating that CLIP's superior semantic alignment and unified representation effectively improve learning coherence. The experiments showed that GRU with CLIP encodings performed best, highlighting the importance of advanced sequence modeling and integrated multi-modal representations for coherence learning. This study illustrates the effectiveness of combining domain adaptation techniques with coherence classification to enhance text-to-image generation for the generation of image series.

ÖZET

İLGİLİ KISA CÜMLELER DİZİSİNDEN GÖRÜNTÜ SERİSİNİN ÜRETİLMESİ

Bu çalışmanın amacı, bir dizi ilgili kısa cümleden tutarlı bir görüntü serisi oluşturarak kısa bir hikayeyi etkili bir şekilde resmetmektir. Metin tabanlı görüntü oluşturma, iyi araştırılmış ancak hala açık bir problemdir. Bu çalışma, anlatı bütünlüğünü sağlamak ve alakasız sahneleri önlemek için önceki görüntü ve metinlerden bilgi aktarımı yaparak bu problemi ele almaktadır. Pororo-SV Veri Seti, Düşünce Zinciri (CoT) ve az örnekli istem teknikleri (Few-Shot) kullanılarak Gemini tarafından yeniden etiketlenmiştir. Ayrıca, duygusal değişiklikler, hedef başarıları ve sosyal etkileşimler gibi temel dinamikleri yakalamak amacıyla Dinamik Hikaye Anlatımı Pororo-DS Veri Seti tanıtılmıştır ve tutarlılığı ölçmek için bir temel sağlamaktadır. Yaklaşım, düşük dereceli adaptasyon (LoRA) yoluyla uyarlanan önceden eğitilmiş bir stabil difüzyon modeli (LDM) kullanarak metinlerden yüksek kaliteli görüntüler üretir. GRU tabanlı Tutarlılık Sınıflandırıcısı, başlangıç sahnesine dayalı olarak sonraki sahneleri seçerek dizilerin anlatı tutarlılığını sağlar. Değerlendirme ölçütleri arasında yapısal benzerlik indeksi (SSIM), Fréchet Inception mesafesi (FID) ve bu çalışma için özelleştirilmiş karakter temsil doğruluk ölçümleri olan Karakter Varlık Doğruluğu (CPA), Tekrarlanan Karakter Oranı (DCR) ve Karakter Tam Eşleşme Doğruluğu (CEMA) bulunur. Sonuçlar, modelin rekabetçi SSIM ve FID skorlarına ulaştığını, karakter varlığı ve tutarlılığında iyileşme sağladığını ve görüntülerin kısa bir hikayeyi etkili bir şekilde tanımladığını göstermektedir. Metin ve görüntüler için CLIP kodlamalarını kullanan Tutarlılık Modeli, BERT ve ResNet kodlamalarından daha iyi performans göstererek, CLIP'in üstün anlamsal hizalama ve birleşik temsil yeteneklerinin tutarlılığı iyileştirdiğini göstermektedir. Deneyler, CLIP kodlamalarıyla birlikte GRU'nun en iyi performansı sunduğunu ortaya koymuş ve gelişmiş dizi modelleme ile entegre çoklu modal temsilin tutarlılık öğrenimindeki önemini vurgulamıştır. Bu çalışma, metinden görüntü serisi oluşturma sürecinde, alan adaptasyon tekniklerinin tutarlılık sınıflandırması ile birleştirilmesinin etkinliğini göstermektedir.

DEDICATION

to my family...

ACKNOWLEDGEMENTS

During my first year of the master's program as a Research Assistant under the guidance of Dr. Engin Kandıran and Prof. Dr. Avadis Simon Hacınlıyan, I have received invaluable support and encouragement, for which I am deeply grateful. Their mentorship has been pivotal in shaping my academic journey.

Special thanks to Dr. Narjes Nikzad for being a part of this work and guiding me as a Co-Supervisor. Her expert guidance and unwavering support have been essential to my research development.

I would like to extend my sincere thanks to the greatest mathematician, Aydın Gerek, for his support and for training me in the field of Artificial Intelligence. His expertise and dedication have greatly influenced my work and development. Serdar Helli and Burhan Arat, whose passion and friendship have been important, I would also like to thank them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AR-LDM	Auto-Regressive Latent Diffusion Model
ASE	Aligned Sentence Encoder
ASN	Average Sample Number
AWE	Attentional Word Encoder
BCE	Binary Cross Entropy
BERT	Bidirectional Encoder Representations from Transformers
BLIP	Bootstrapped Language-Image Pretraining
CLIP	Contrastive Language-Image Pretraining
CoT	Chain of Thought
CNN	Convolutional Neural Network
DDPM	Denoising Diffusion Probabilistic Model
FID	Fréchet Inception Distance
FSD	Fréchet Story Distance
GAN	Generative Adversarial Network
GRU	Gated Recurrent Unit
LDM	Latent Diffusion Model
LLM	Large Language Model
LoRA	Low-Rank Adaptation
MSE	Mean Squared Error
$\mathcal{N}$	Normal Distribution
NLP	Natural Language Processing
Pororo-DS	Pororo Dynamic Story Dataset
Pororo-SV	Pororo Story Visualization Dataset
ResNet	Residual Network
RNN	Recurrent Neural Network
SC	Story Continuation
SSIM	Structural Similarity Index
SV	Story Visualization
β_t	Variance of the Noise at Step t

μ_θ	Mean Function in DDPM
Σ_θ	Covariance Matrix in DDPM
x_t	Data at Time t

1. INTRODUCTION

The advancements in Generative Artificial Intelligence (GAI) have led to major improvements in generating content. These improvements are seen in different media types, such as text, image, and audio. Text-guided generation has performed very well when models use more than one type of media, known as multimodal settings. This success creates new possibilities for developing advanced applications and tackling challenges in the field. Specifically, there are exciting opportunities in areas like narrative story visualization (SV) and story continuation (SC), where these technologies can bring stories to life in new ways. Storytelling, enhanced by visual and multimedia elements, provides significant advantages in engaging audiences and conveying complex narratives more effectively. Visual supports, such as images and animations, can make stories more vivid and memorable, helping to illustrate critical events and emotional nuances that text alone might not fully capture. This integration of visual elements into storytelling not only enriches the narrative experience but also aids in comprehension and retention, making it a powerful tool in education, entertainment, and beyond.

1.1. PROBLEM STATEMENT

SV requires creating consistent scenes from the textual descriptions of each scene and constructing cohesive visual narratives from the flow of the story. This statement is exemplified by mediums such as picture books and comic books, where each illustration or panel functions as a discrete narrative unit within a larger, cohesive narrative framework.

This research addresses several pivotal questions that emerge from the problem statement. First, the capability of image generation systems to learn and replicate specific domain characteristics, ensuring alignment with textual descriptions of scenes, is explored, particularly in contexts with established visual and thematic elements, such as specific cartoon TV series. Moreover, the investigation focuses on enhancing the image generation pipeline by integrating narrative contextual cues, utilizing preceding text and visual content to retain temporal narrative flow and ensure the coherence and continuity of the visual narrative. Consequently, this thesis examines the following research questions:

- How effectively can image generation systems learn and replicate specific domain characteristics to ensure alignment with textual descriptions of scenes, especially in contexts with established visual and thematic elements, such as specific cartoon TV series?
- In what ways can preceding text and visual content be leveraged within the image generation pipeline to maintain temporal narrative flow and coherence?

1.1.1. Contribution of The Thesis

The contribution of this thesis lies in the development and enhancement of text-to-image generation techniques for narrative story visualization and continuation while maintaining coherence, including comparisons with different encoding models, while creating synthetic data to support these advancements.

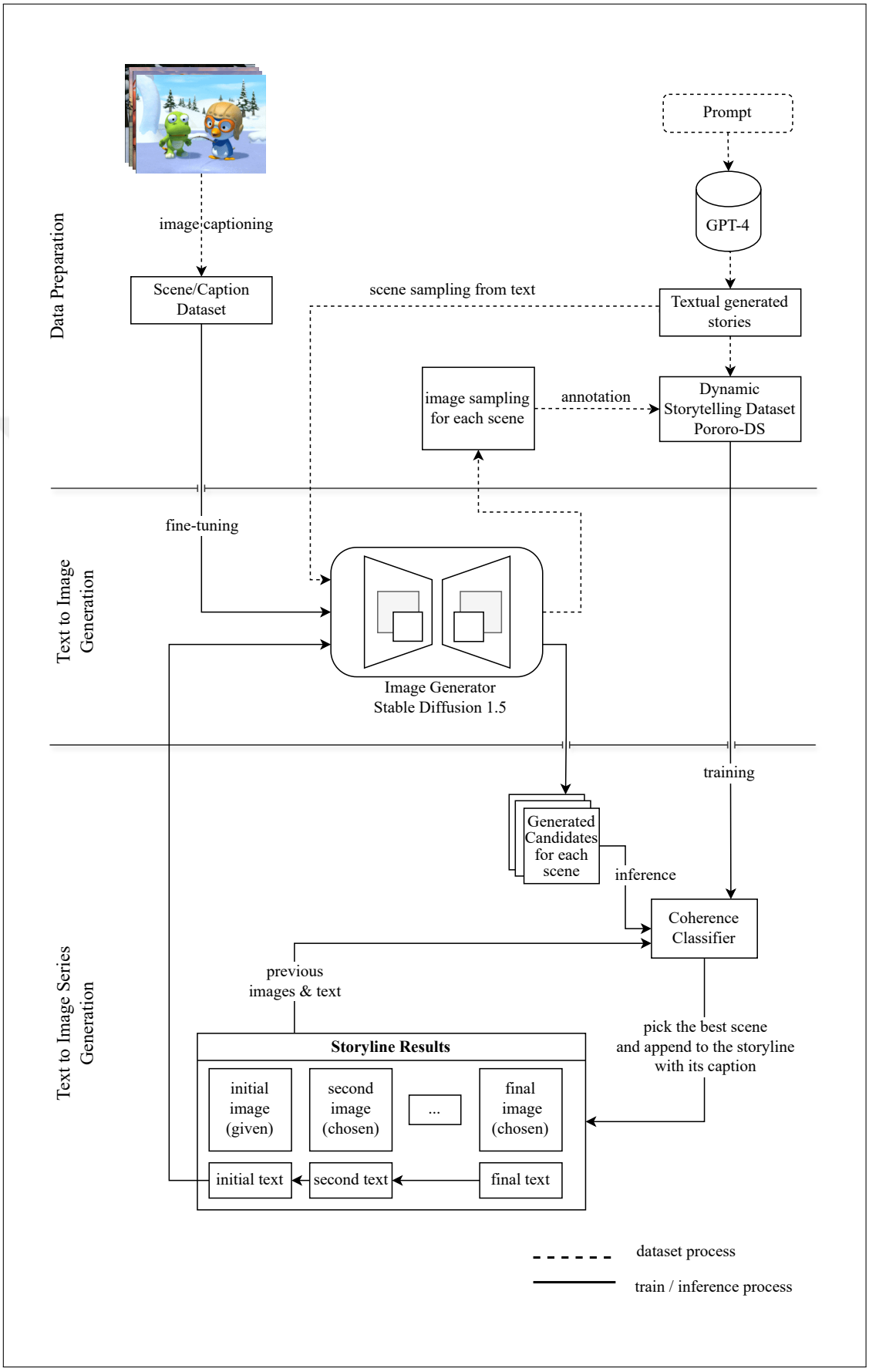

Figure 1.1. Overview of Framework from Dataset Preparation to Story Continuation

2. RELATED WORKS

In this chapter, the details of story visualization will be examined, focusing on three primary approaches: Retrieval-Based, Generative Adversarial Networks (GANs), and Diffusion-Based Models. The first part will explore Retrieval-Based Story Visualization, highlighting how discrete, narratively connected scenes are generated by leveraging annotated datasets and advanced context modeling techniques. The second part will cover the use of GANs in story visualization, discussing significant contributions, such as StoryGAN and its advancements in generating coherent visual sequences. The final part will delve into Diffusion-Based Models, explaining how advanced neural network architectures and latent diffusion processes are employed to convert textual narratives into cohesive visual stories. Through the analysis of these foundational approaches, a comprehensive understanding of the advancements in transforming text into engaging visual stories will be provided.

2.1. RETRIEVAL BASED STORY VISUALIZATION

The field of text-to-image generation has seen significant expansion and diversification in recent research efforts. Within this domain, the specific areas of SV and SC have emerged with distinct focuses. Unlike video generation tasks that create continuous sequences of motion within a consistent background, the SV approach generates a series of discrete yet narratively connected scenes. The ultimate goal of SV is to bridge the gap between textual narratives and visual representations, creating a seamless and engaging storytelling experience that captures the essence and progression of the narrative. Early works on creating storyboards and ordering sequences of consistent images can be traced back to efforts leveraging datasets with annotated captions. One such approach is the retrieval-based system [1], focusing on extracting images that correspond to individual sentences within a coherent sequence. This methodology features an end-to-end architecture, integrating context modeling through a two-level Gated Recurrent Unit (GRU) [2] network. This approach was effectively applied to the Visual Storytelling (VIST) [3] dataset. [4,5] also following the story-to-image retriever, text-based image searching approach respectively.

2.2. GENERATIVE ADVERSARIAL NETWORKS FOR STORY VISUALIZATION

StoryGAN [6], was introduced as a pioneering effort in SV, exploiting the text-to-image synthesis capability of GANs [7,8]. The model comprises a Generator, which includes a Story Encoder, Context Encoder, and Image Generator, along with two task-specific discriminators. The Image Discriminator ensures the relevance of text-to-image conversion, while the Story Discriminator preserves the coherence of the image sequences over time. Additionally, the research introduced the Pororo-SV dataset, an augmentation of the Pororo-QA Cartoon dataset [9], initially used for visual question-answering tasks in videos. This development marked a significant milestone in the field of SV, illustrating the potential of GANs in generating narrative-consistent visual sequences. Following this, PororoGAN [10] introduced the use of an Attentional Word Encoder (AWE) to enhance fine-grained, word-level information throughout the image generation process, and an Aligned Sentence Encoder (ASE) to better align story content. This approach aimed to fill the gap in matching image and text pairs by learning a multimodal encoding representation, as opposed to relying on a universal sentence encoder [11]. In 2020, Character-Preserving Coherent Story Visualization (CP-CSV) [12] architecture was introduced, which enhanced the story and context encoders and incorporated an auxiliary segmentation task by adding an extra figure-ground discriminator. This improvement aimed to better distinguish between figure-ground elements and preserve character features within the images. Additionally, they proposed a novel metric, the Frechet Story Distance (FSD), as an alternative to the Frechet Video Distance (FVD) [13], designed to measure the score of short and discrete image sequences. Dual-Copy StoryGAN (DUCO-STORYGAN) [14], to solve the global consistency issue which is related to maintaining background and character visual properties between frames, the story encoder leveraged by Memory Augmented Recurrent Transformer (MART) as Context Encoder with MART based transformers [15]. In addition, they integrate a video captioning module, called video re-description, to utilize dual learning and adjust the semantic alignment between story and image frames. In [16], constituency parse trees encoding, which is composed by Memory-Augmented Recurrent Tree Transformer (MARTT) [15] as input to a structure-aware encoder is used. In order to provide object and characters information and succeed in semantic alignment in generated scenes they adopt the intra-story contrastive loss. With Story DALL-E [17], Maharana et al. introduce the SC task, which is an extension of the SV problem. They

provided the first frame to narrate a story using a series of text captions. This method is based on retrofitting, which involves combining a pre-trained model with additional modules. Instead of generating a single image, the procedure can generate images sequentially to provide the flow of a narrative. It employs a copying mechanism that allows the model to copy from a previous or specified frame in order to maintain visual consistency. Because they proposed a new task, SC, they also provided the StoryGANc model as a baseline.

2.3. DIFFUSION BASED MODELS FOR STORY VISUALIZATION

Auto-Regressive Latent Diffusion Models (AR-LDMs) [18] use advanced neural network architectures to convert textual narratives into cohesive visual stories. The models use Contrastive Language-Image Pre-Training (CLIP) and Bootstrapping Language-Image Pre-training (BLIP) encoders, with CLIP encoding textual descriptions and ensuring that images are relevant to the given text and BLIP integrating multimodal data—combining prior images and their associated captions to maintain narrative consistency across the sequence of generated images. Crucially, the AR-LDMs use a U-Net [19] architecture, which is critical in the diffusion process. The U-Net refines and denoises latent representations gradually, effectively reversing the initial noise introduction to produce detailed and contextually accurate images. This architecture not only improves the visual quality of the generated images but also ensures that they fit seamlessly into the evolving storyline. Story-LDM [20] is introduced, which uses latent diffusion models (LDM) to generate stories. This framework includes a novel memory-attention mechanism that captures and applies the contextual relevance between previously generated parts of the story and the current frame, thereby improving character and background consistency throughout the narrative. Second, the study expands the MUGEN, FlintstonesSV, and PororoSV datasets with more complex scenarios and referential text to address challenges with co-reference resolution and visual narrative consistency.

3. BACKGROUND

This chapter provides a comprehensive overview of the foundational concepts and methodologies relevant to this research. It begins with DDPMs, which represent a significant advancement in generative modeling, allowing for the production of high-quality images through a reverse diffusion process. Following this, CLIP will be explored, a technique designed to learn visual concepts from natural language descriptions, enabling various vision tasks without direct supervision. Additionally, BERT and ResNet will be examined as prominent architectures for natural language processing and image recognition. Next, LoRA, a method for efficiently adapting deep learning models to specialized tasks, will be discussed. Lastly, RNNs and GRUs will be covered as architectures designed for processing sequential data. Each section delves into these technologies' fundamental principles, mathematical formulations, and practical applications, providing a solid foundation for understanding their role and implementation in this research.

3.1. DENOISING DIFFUSION PROBABILISTIC MODELS

DDPMs [21] are a significant advancement in the field of generative models, providing a novel method for producing high-quality images. This methodology is based on non-equilibrium thermodynamics theory, and it uses a stochastic process called diffusion, which is gradually reversed to generate data samples [22]. The essential idea is to model data generation as a reverse Markov process that begins with noise. By gradually removing noise using learned steps, DDPMs eventually reconstruct samples that resemble the training data. The forward diffusion process is that;

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{3.1}$$

where the $x_{t-1}$and x_t represent the data in over time between $t-1$ and t. β_t is the variance of the noise added at step t, which is a predefined schedule of increasing values, typically ranging from a small positive value close to 0 to a value near 1. $\mathcal{N}$ indicates a normal distribution, I denotes the identity matrix, which ensures that the added noise is isotropic so

that Gaussian noise added has the same variance in every dimension of the data space. This uniformity is critical for the model to learn a generalized capability to reverses she noise in any direction during the reverse diffusion process.

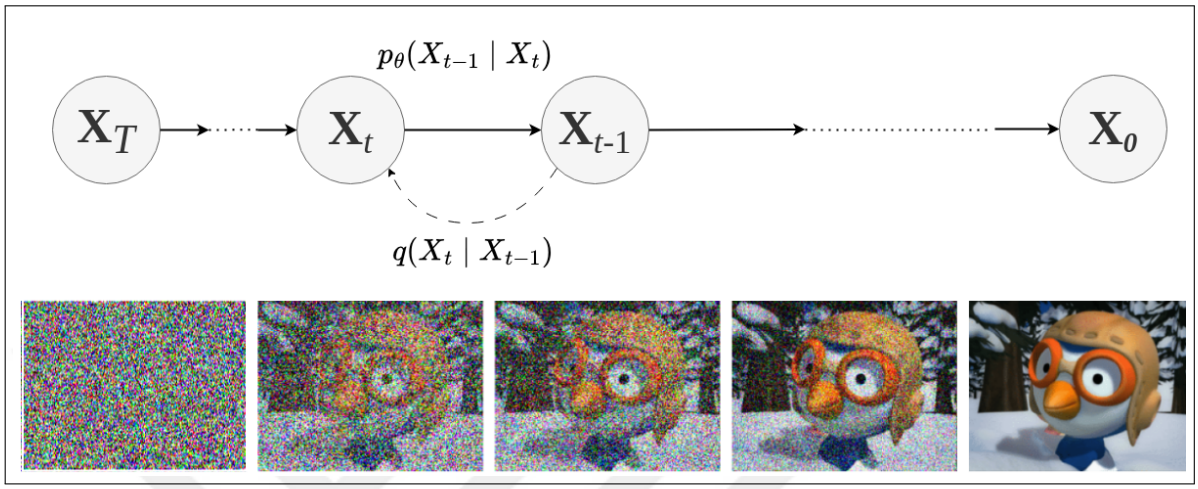

Figure 3.1. Forward and Backward Diffusion Processes

The reverse diffusion process, which can be modeled as a sequence of conditional distributions, reverse the Gaussian noise added from the forward process 3.1, which can be written as below;

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{3.2}$$

$\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are the mean and covariance matrix predicted by the neural network, parameterized by θ, for step t.

The primary training objective function is to minimize the expectation of the squared Euclidean distance between the noise ϵ added to the data during the forward diffusion process and the noise $\epsilon_\theta(x_t, t)$ as predicted by the neural network at a given timestep t.

$$\theta^* = \arg\min_\theta L_{\text{simple}}(\theta) = \arg\min_\theta \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right] \tag{3.3}$$

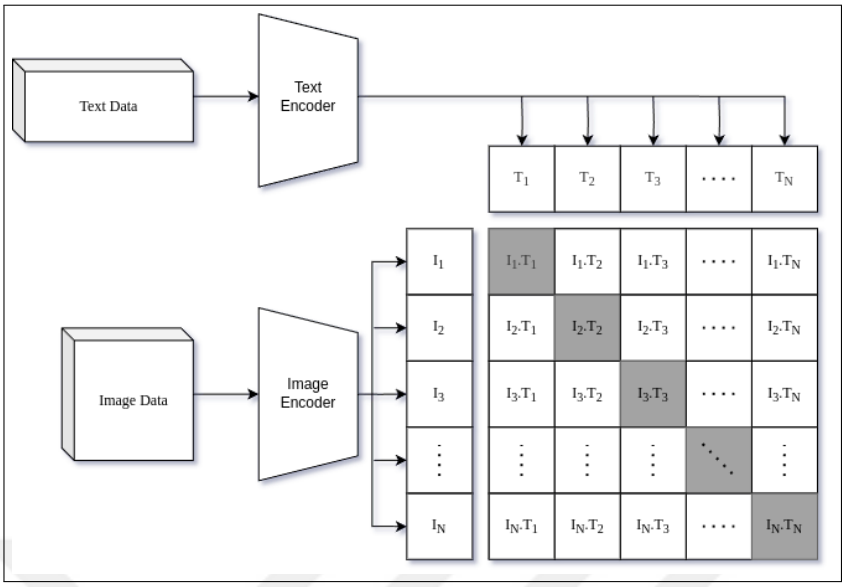

Figure 3.2. The architecture of CLIP. Text data and image data are separately encoded using the Text Encoder and Image Encoder, respectively.

3.2. CONTRASTIVE LANGUAGE-IMAGE PRETRAINING

CLIP [23] is designed to learn visual concepts from natural language descriptions, enabling the model to perform a variety of vision tasks without direct supervision on those tasks. The primary methodology involves jointly training an image encoder and a text encoder. The image encoder processes visual inputs, and the text encoder processes descriptions of those visuals. Training involves using a contrastive loss function, which aligns the image and text representations in a shared embedding space. The objective is pairing (I_i, T_i) over all the other possible pairings (I_i, T_j) where $j \neq i$ in 3.4.

$$L = -\sum_{i=1}^{N} \log \frac{\exp(\text{sim}(I_i, T_i))}{\sum_{j=1}^{N} \exp(\text{sim}(I_i, T_j))} \tag{3.4}$$

(I, T) are image-text pairs, and L is the total loss for a given batch of N to minimize. $\text{sim}(I_i, T_i)$ function computes the similarity between the embedding of image I_i and the embedding of its true pair text T_i. Typically, this similarity is measured using cosine similarity 3.5

$$\text{cosine\_similarity}(I_i, T_i) = \frac{I_i \cdot T_i}{\|I_i\| \|T_i\|} \tag{3.5}$$

Thus, CLIP objective function can be written explicitly;

$$L = -\sum_{i=1}^{N} \log \left(\frac{\exp\left(\frac{I_i \cdot T_i}{\|I_i\| \|T_i\|}\right)}{\sum_{j=1}^{N} \exp\left(\frac{I_i \cdot T_j}{\|I_i\| \|T_j\|}\right)} \right) \tag{3.6}$$

After training, CLIP can be applied to new tasks without any task-specific training, a method known as zero-shot learning [24]. This is achieved by constructing text prompts that describe the classes of the new task (e.g., "a photo of a class label"). At inference time, the text encoder generates embeddings for these prompts, and the image encoder processes the input image.

$$\hat{y} = \arg \max_{k \in \text{Classes}} \text{cosine\_similarity}(I, T_k) \tag{3.7}$$

The prediction is made by selecting the class whose text embedding has the highest cosine similarity with the image embedding.

3.3. BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS

BERT, Bidirectional Encoder Representations from Transformers [25], marks a significant advancement in Natural Language Processing (NLP). It utilizes only the encoder part of the transformer architecture, consisting of multiple layers of transformer blocks (12 layers for base and 24 for large). Each transformer block includes a self-attention mechanism and a feed-forward neural network. The most innovative aspect of this model is its bidirectional training method. Unlike traditional models that process text in a single direction, BERT simultaneously considers both the left and right contexts. This bidirectional approach enables the model to capture a richer and more nuanced context for each word within a sentence. The self-attention mechanism allows each word to attend to every other word in the phrase, computing attention scores based on the surrounding words. By generating contextual embeddings for each word, BERT's representation is influenced by its context. This approach captures subtle nuances and multiple interpretations of words in different situations. As a result, the model serves as an excellent text encoder, embedding rich

contextual information for words. Its comprehensive representation enhances performance in tasks such as named entity recognition, question answering, and language inference, demonstrating its effectiveness in text encoding.

3.4. RESIDUAL NETWORK

ResNet (Residual Network) [26] is a sophisticated convolutional neural network (CNN) architecture renowned for its exceptional depth and ability to effectively address the vanishing gradient problem, a common issue in training deep neural networks. ResNet incorporates residual learning frameworks, wherein identity shortcut connections, or skip connections, bypass one or more layers. These connections allow the network to learn residual functions with reference to the layer inputs, facilitating more efficient gradient flow during backpropagation. This architectural innovation permits the training of much deeper networks, such as ResNet-50, ResNet-101, and ResNet-152, without degradation in performance.

As an image encoder, ResNet excels in extracting robust and hierarchical feature representations from images. By discarding the final fully connected layers and utilizing the output from an intermediate layer, often the global average pooling layer, ResNet transforms input images into fixed-size feature vectors. These vectors encapsulate rich semantic information about the images, making them ideal for various downstream tasks, such as image classification, object detection, and image retrieval. Leveraging pre-trained ResNet models and fine-tuning them on specific datasets can significantly enhance their performance and adaptability for specialized applications. This involves modifying the architecture by removing the final fully connected layer, which is originally designed for classification tasks with a specific number of output classes. Instead, we focus on the layers preceding this final classification layer to obtain feature encodings. The typical approach involves loading a pre-trained ResNet model, such as ResNet-50, ResNet-101, or ResNet-152, and programmatically removing the final layer to access the high-level feature representation of the image.

3.5. LOW-RANK ADAPTATION

Adapting deep learning models to specialized tasks without significant retraining is crucial. [27,28] demonstrates that overparametrized models inherently possess a low intrinsic rank. Building on this, [29] introduced LoRa, a leading Parameter efficient fine-tuning (PEFT) method. This approach suggests that weight changes during model adaptation maintain a low "intrinsic rank." While the outputs of the pre-trained model remain frozen, LoRA learns low-rank decomposition matrices that represent the necessary changes for the desired task during fine-tuning.

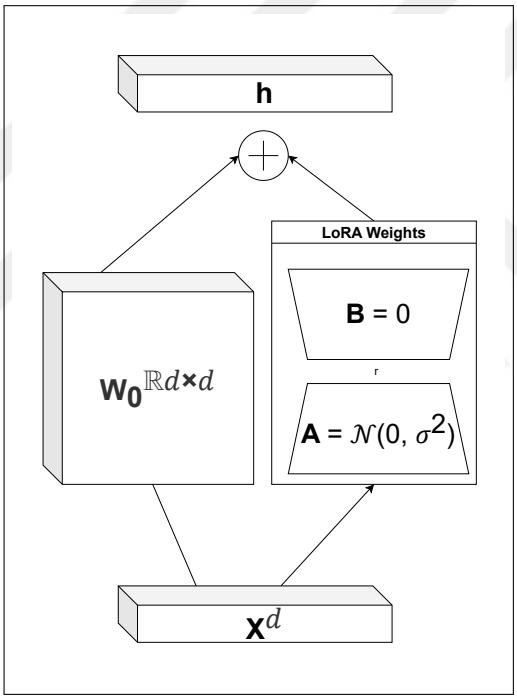

Figure 3.3. Low Rank Reparametrization

Advantages of LoRa may be listed as below;

- A trained model can be employed to generate various LoRA modules for distinct applications. By freezing the common model and just changing the matrices A and B, the task-switching can be efficiently managed, which also benefits in reducing storage requirements.
- The training process becomes more efficient in terms of memory usage and time because it only retains the smaller, low-rank matrices that are injected, and it doesn't

require computing gradients or keeping the optimizer states for most of the pre-trained model's parameters.

- Even with fewer parameters, LoRA maintains the quality and inference speed of the original model.
- The orthogonality property of LoRa makes it possible to combine it with other PEFT methods, allowing for enhanced functionality and performance improvements [29].

Since the method is not tied to any particular training objective, it can be applied to any parametrized model. The method was originally explained for language models, $LM_\Phi(y \mid x)$, parameterized by Φ, the pre-training objective is to maximize the language modeling objective which is given previous words to predict next one. 3.8.

$$\max_{\Phi} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(P_\Phi(y_t|x, y_{<t})\right) \tag{3.8}$$

Objective function with LoRa weights can be written as below;

$$\max_{\Theta} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(p_{\Phi_0+\Delta\Phi(\Theta)}(y_t|x, y_{<t})\right) \tag{3.9}$$

3.6. RECURRENT NEURAL NETWORKS AND GATED RECURRENT UNITS

Processing sequential data such as speech, text, or time series is fundamental for numerous applications in artificial intelligence. However, the architecture of basic neural networks like Multilayer Perceptrons (MLPs) lacks a direct mechanism to handle these types of sequences. In contrast, RNNs and GRUs [30,2] are two advanced neural network architectures designed specifically for this purpose. These networks maintain a hidden state that acts as a form of memory, allowing them to process each element of the input sequence by continually updating the hidden state based on both the current input and the information retained from previous inputs.

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b_h) \tag{3.10}$$

Although this model seems good at processing sequences, it actually causes the gradient vanishing problem on long input sequences during training [31].

GRUs solve this problem through a clever architecture that incorporates gating mechanisms learning in long sequences, which help to control and manage the flow of information. These gates decide what information should be passed along to the output and what should continue to propagate through the network as memory.

Update Gate, given in Equation 3.11, helps the model determine how much of the past information autoregressively needs to be passed along to the future. This gate acts like a blend between the memory content and the new information.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{3.11}$$

Conversely, the Reset Gate 3.12 controls the amount of past information to be forgotten, playing a key role in the network's ability to discard irrelevant information from earlier in the sequence.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{3.12}$$

Here, x_t represents the input at time t, W_z and W_r are the weights for the update and reset gates, respectively, and h_{t-1} is the output from the previous hidden layer. The output of the candidate hidden layer at time t, denoted $\tilde{h}_t$, retains historical information, influenced by the reset gate as shown;

$$\tilde{h}_t = \tanh(W \cdot [r_t \odot h_{t-1}, x_t]) \tag{3.13}$$

The final output of the hidden layer is computed by balancing the information from the previous hidden state and the candidate hidden state, controlled by z_t:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \tag{3.14}$$

When targeting complex sequence learning, stacking multiple GRUs layers enhances their capabilities by allowing each layer to process the hidden states from the previous layer. This method boosts the network's ability to handle intricate features and capture detailed temporal dependencies, increasing both the accuracy and depth of learning. Such a configuration is particularly effective for challenging tasks requiring advanced sequence learning, enabling the GRUs to operate at higher abstraction levels and better model complex patterns in sequential data.

4. METHODOLOGY

This chapter outlines the research methods and analytical procedures implemented in this study. The initial section covers the steps taken to sample and prepare the data necessary for this research, including a detailed explanation of the dataset, the extension of the dataset pipeline, and the specific procedures followed. The subsequent section focuses on the image generation process, describing the models used to convert textual scenarios into scene images. Following the image generation process, this section elaborates on the training of the coherence function, which is crucial for maintaining story visualization continuity. It details how narrative coherence is achieved through visualizations and the methods used to ensure that the generated images are consistent with the ongoing storyline. The final section of this chapter examines the evaluation metrics utilized in this study. It defines the criteria for evaluating the effectiveness of the image generation process and the narrative coherence, providing the foundation for the analysis presented in the subsequent chapters.

4.1. DATASET PREPARATION

The Pororo-SV Dataset [6], derived from a popular cartoon TV series named Pororo the Little Penguin,is annotated for research in video understanding and narrative comprehension. This series for preschool children centered around the adventures of Pororo, a curious and playful little penguin, and his friends in the snowy village of Porong Porong Forest. The show features educational themes, humor, and fun, teaching lessons on friendship, cooperation, and problem-solving. Key locations include the characters' distinct homes, a playground, and the village square.

In this study, the scene captions were re-annotated to enhance descriptions and reduce caption error rates. This re-annotated process yields a total of 12,287 scene image-caption pairs to be used as training data. The occurrence of the six main characters listed above in the cartoon series within these scene captions is depicted in 4.3

The main characters and their overall description listed as in Figure 4.1

Figure 4.1. Main characters of *Pororo the Little Penguin*.

4.1.1. Text to Scene Dataset

The dataset frames were individually captioned using the multimodal large language model (MLLM) through the Gemini [32]. This process leveraged the few-shot learning capabilities [33] of LLMs, combination with CoT [34] technique, with the prompt function being formalized as; $P(s) = T$, which generates a structured text prompt T for scene image s.

The components of T can be defined as:

1. **Instruction (I):** Provides general guidelines on how to approach the captioning task.
2. **Character Identification (CI):** Offers detailed descriptions and images to help identify characters in the scenes.
3. **Scene Description (SD):** Directs how to detail the scene's elements, such as the background, objects, and character actions.
4. **Few-Shot Examples (FSE):** Combines the elements of CoT and example captions to illustrate how to compose detailed scene captions.

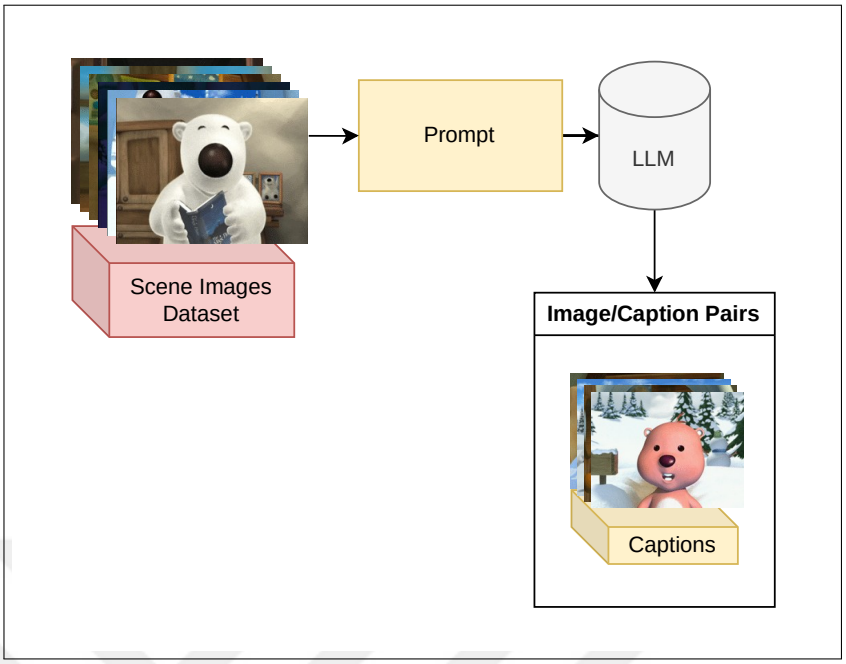

Figure 4.2. Scene Captioning Process

Given an array of images $I = [i_1, i_2, ..., i_n]$ used for character identification and few-shot examples, the total prompt T can be constructed as follows:

$$T = I + CI(i_1, i_2, ..., i_m) + SD + FSE(i_{m+1}, ..., i_n) + "caption:" \tag{4.1}$$

- $CI(i_1, i_2, ..., i_m)$ represents the character identification text constructed using images i_1 to i_m
- $FSE(i_{m+1}, ..., i_n)$ represents the few-shot examples, combining the narrative flow and caption examples, using the remaining images i_{m+1} to i_n. FSE providing direct examples of how to execute this task.
- The final "caption: " string is added to signal the start of the caption generation task for the image s)

The prompt used in this work can be found in the Appendix A, specifically in Figures A.3, A.4

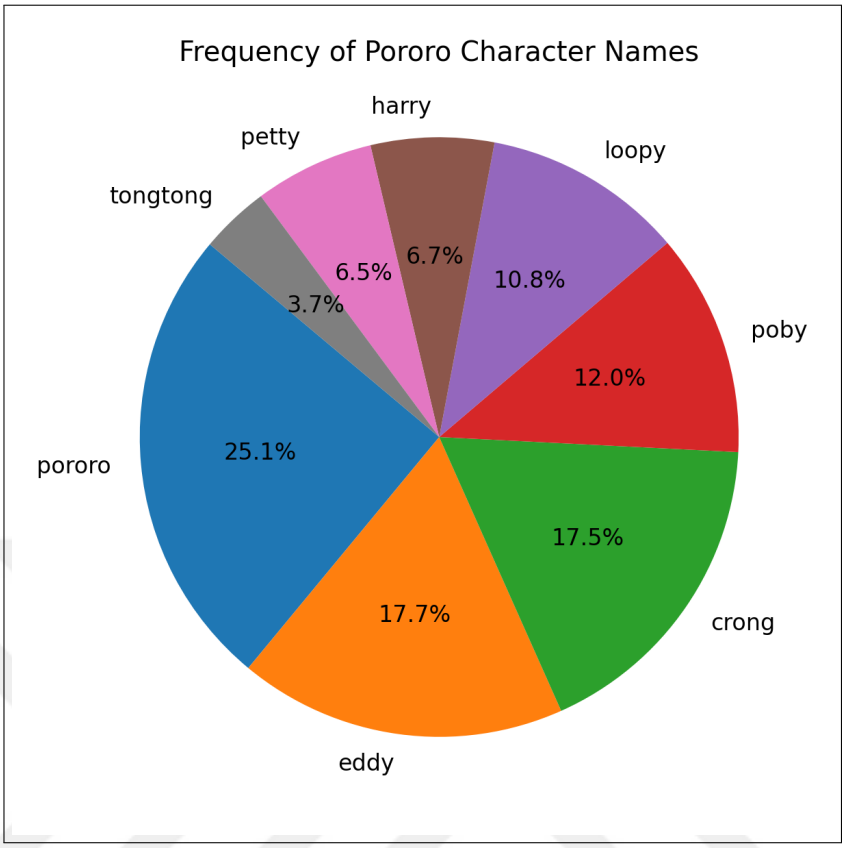

Figure 4.3. Frequency of Character Names in Captions

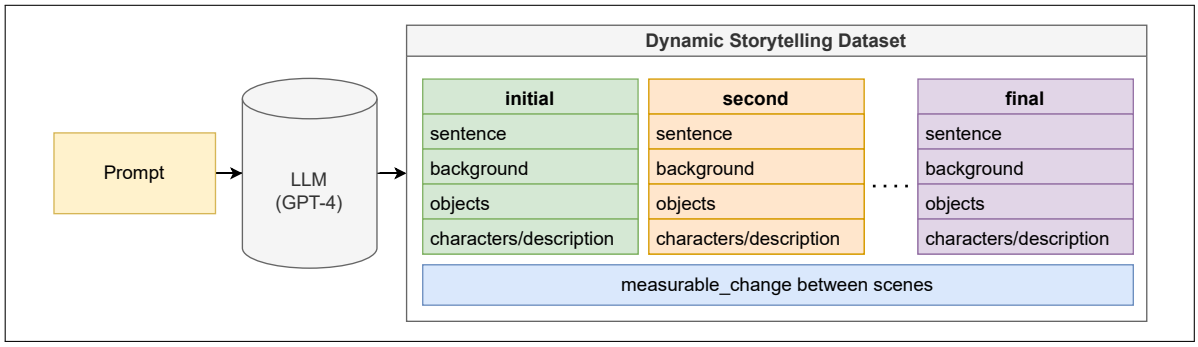

Figure 4.4. Story Sampling from LLM for Global Consistency

4.1.2. Dynamic Storytelling Dataset

In this study, we identified that the problem proposed with the Pororo-SV dataset is too complex and the original dataset has some labeling issues. For instance, in some scenes, the characters do not appear as described, likely due to misalignment of frames, leading to annotation corruption. Additionally, while some scenes have multiple sentences, others have only one, despite the problem being defined as each scene having a single sentence. Moreover, the image descriptions often lack detail, failing to adequately describe the scene. For example, in Figure 5.8 frame #5 only mentioned by caption *"Crong sighs."*, with no explanation about the background, objects, or character status. These issues make it challenging to assert that the model learns a coherent function between scenes. To address this, we extended the Pororo-SV dataset to create the Dynamic Storytelling version, the Pororo-DS, reflecting dynamic changes and progression in story elements. We crafted story examples that highlight transitions between scenes through selected "measurable change types," capturing the storyline from beginning to end with initial and final texts, backgrounds, key objects, and characters. The prompt used to create Pororo-DS dataset on this purpose with utilizing GPT-4 [35] can be found in A, in A.1 and A.2.

This approach aims to demonstrate a learnable coherence function between scenes, ultimately facilitating the sampling of new story continuations. To evaluate long sequence training results, this dataset was further extended with second and third scenes and their text captions as shown in 4.9.

The defined change types in this work are in three category:

- **Emotional Change:** This type traces the evolution of characters' emotions, starting from negative feelings like fear, sadness, or frustration, and culminating in positive outcomes such as joy, comfort, or confidence.
- **Goal Achievement:** It reflects the journey of characters attaining or transforming their goals or aspirations, often through acquiring new skills.
- **Social Interaction:** This involves the introduction of new characters who assist or interact with the initial characters to complete tasks or communicate important

information, while preserving the significance of the primary characters.

This Pororo-DS dataset will be completed with sampling of initial and final scenes from the fine-tuned LDM and annotating the positive and negative samples. The dataset was extended to measure four-scene length by using previously generated stories. GPT-4 was then asked to insert the second and third texts, including background, objects, and characters. In Figure 4.9, examples of four sequential scenes are provided.

Algorithm 4.1. Textual example of a story data entry in the Pororo-DS dataset, showcasing the transition from an initial scene to a final scene, along with detailed descriptions of background, objects, characters, and the measurable change in the storyline.

```
story_title: "Crong's Icy Adventure"
initial_text: "Crong is grumpy because of the cold, icy morning."
final_text: "Crong is thrilled, sliding joyfully on the frozen pond."
initial_background: "Snowy land, icy morning"
final_background: "Snowy land, frozen pond, midday"
initial_objects: "Ice, snow"
final_objects: "Ice skates, frozen pond"
initial_characters:
  - name: "Crong"
    description: "Grumpy, cold"
final_characters:
  - name: "Crong"
    description: "Thrilled, playful"
measurable_change: "Crong's grumpiness turns to thrill as he enjoys ice skating"
change_type: "emotional_change"
```

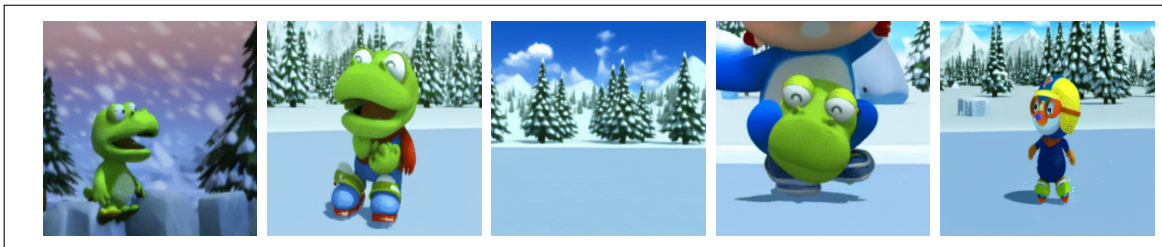

Figure 4.5. Visual Examples of Initial and Final Scenes with Labels for Dynamic Storytelling. The positive final scene maintains narrative coherence, while negative final scenes show errors such as missing character, incomplete character, and wrong character.

The synthetic dataset manually labeled and the chosen scenes (positives) and rejected scenes (negatives) distribution listed in Figures 4.10, 4.11, 4.12, 4.13.

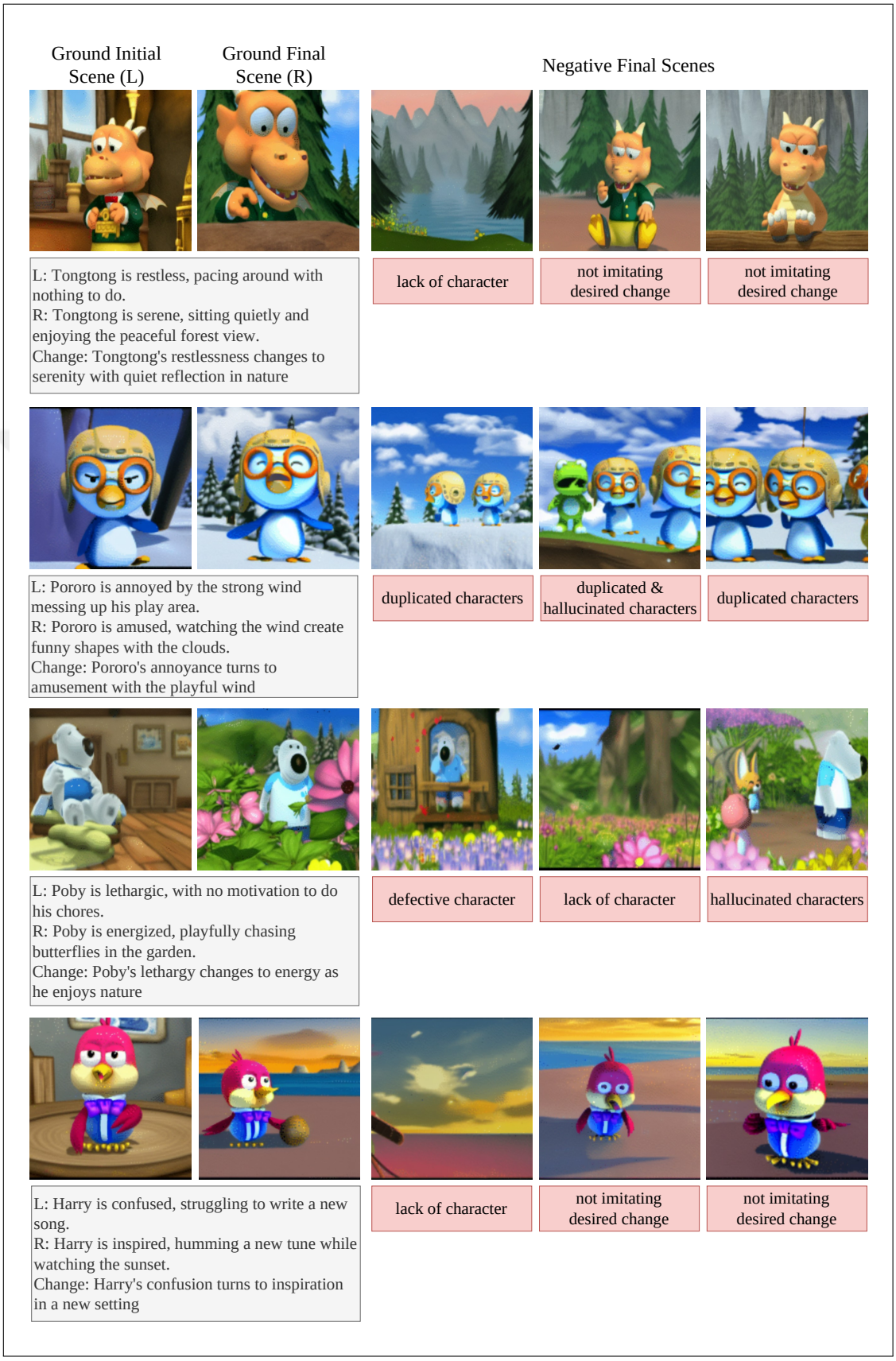

Figure 4.6. Examples of Dynamic Storytelling Emotional Change with Initial and Final Scenes, Captions, and Negative Labels

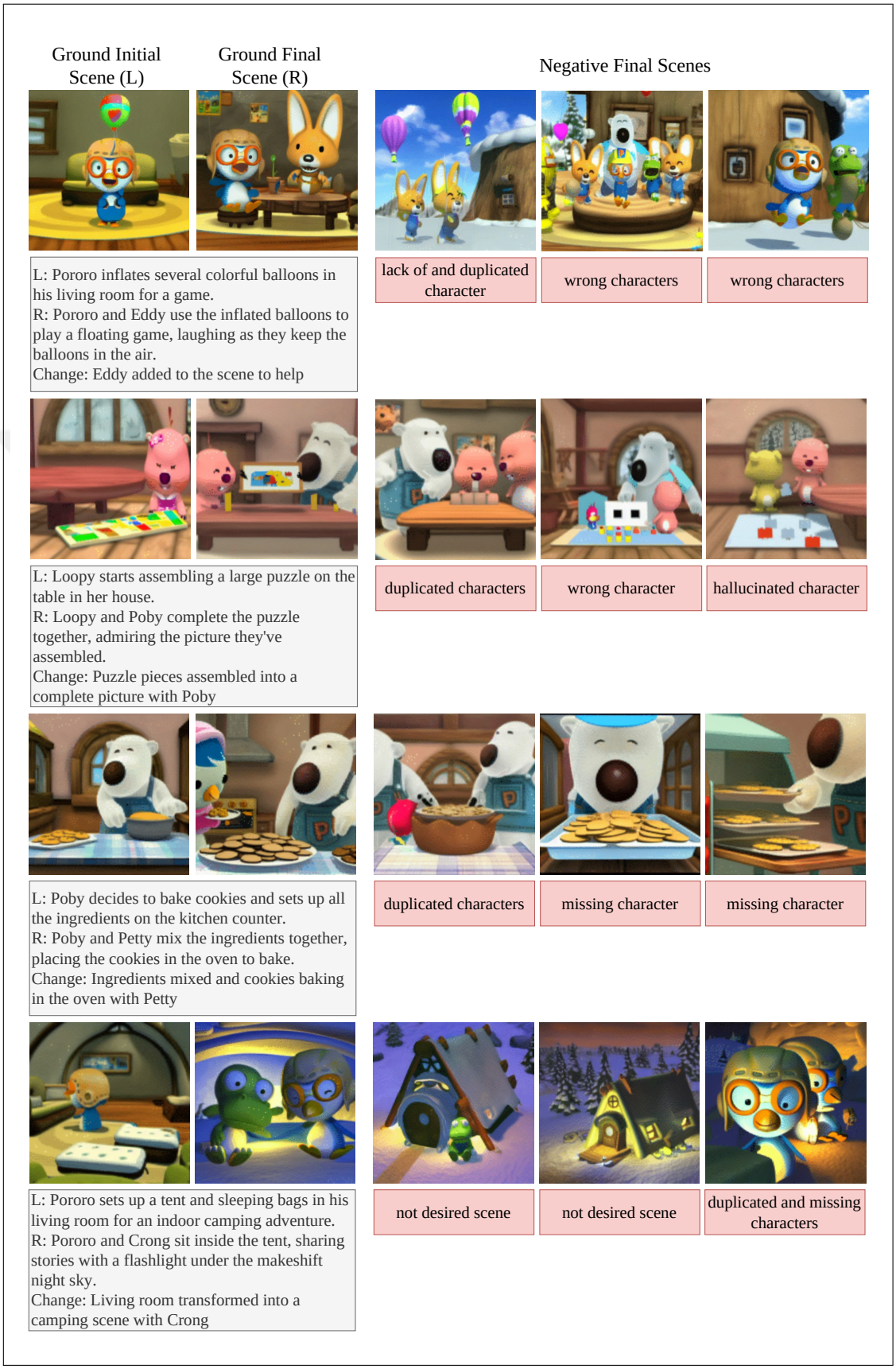

Figure 4.7. Examples of Dynamic Storytelling Social Interaction with Initial and Final Scenes, Captions, and Negative Labels

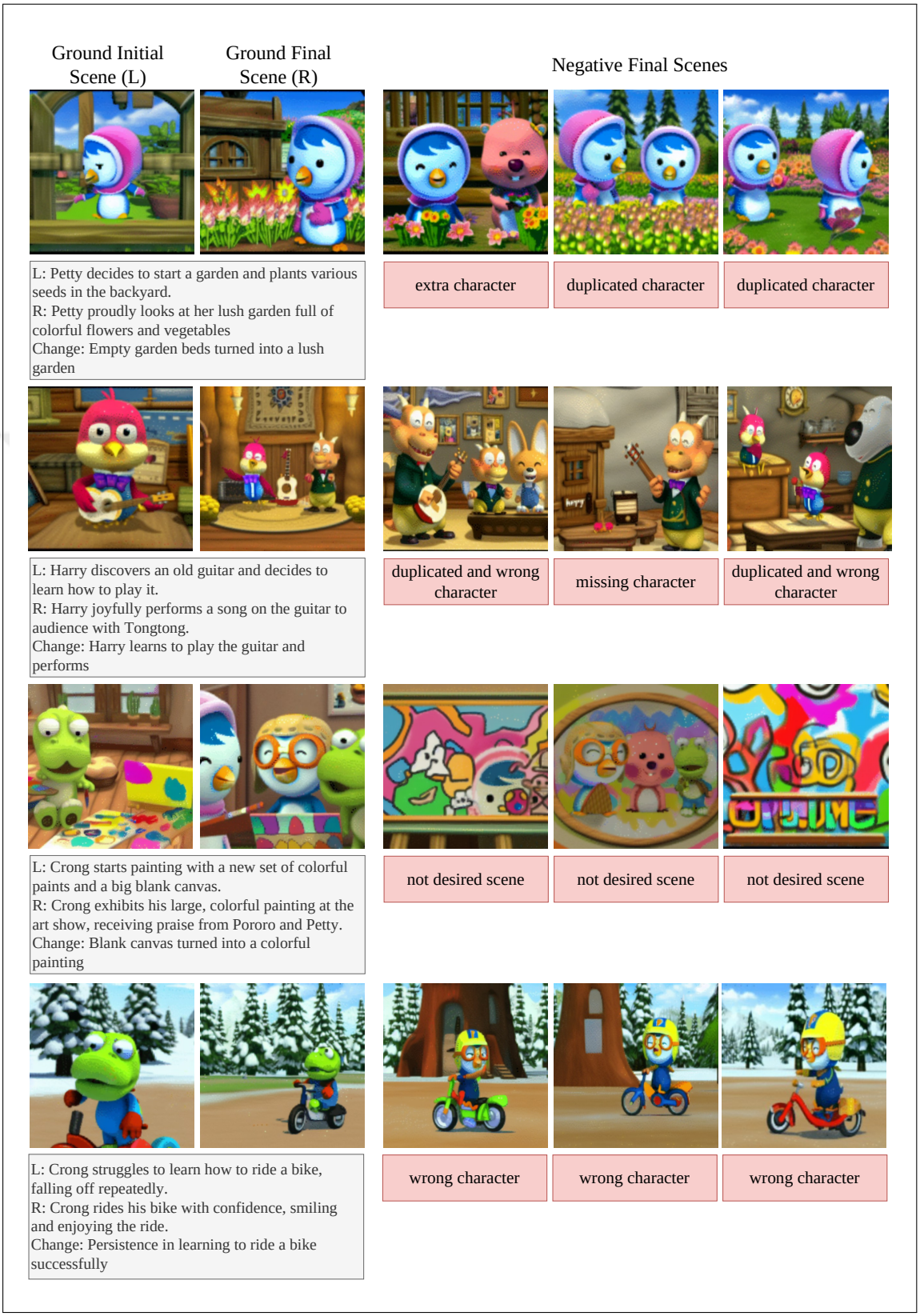

Figure 4.8. Examples of Dynamic Storytelling Goal Achievement with Initial and Final Scenes, Captions, and Negative Labels

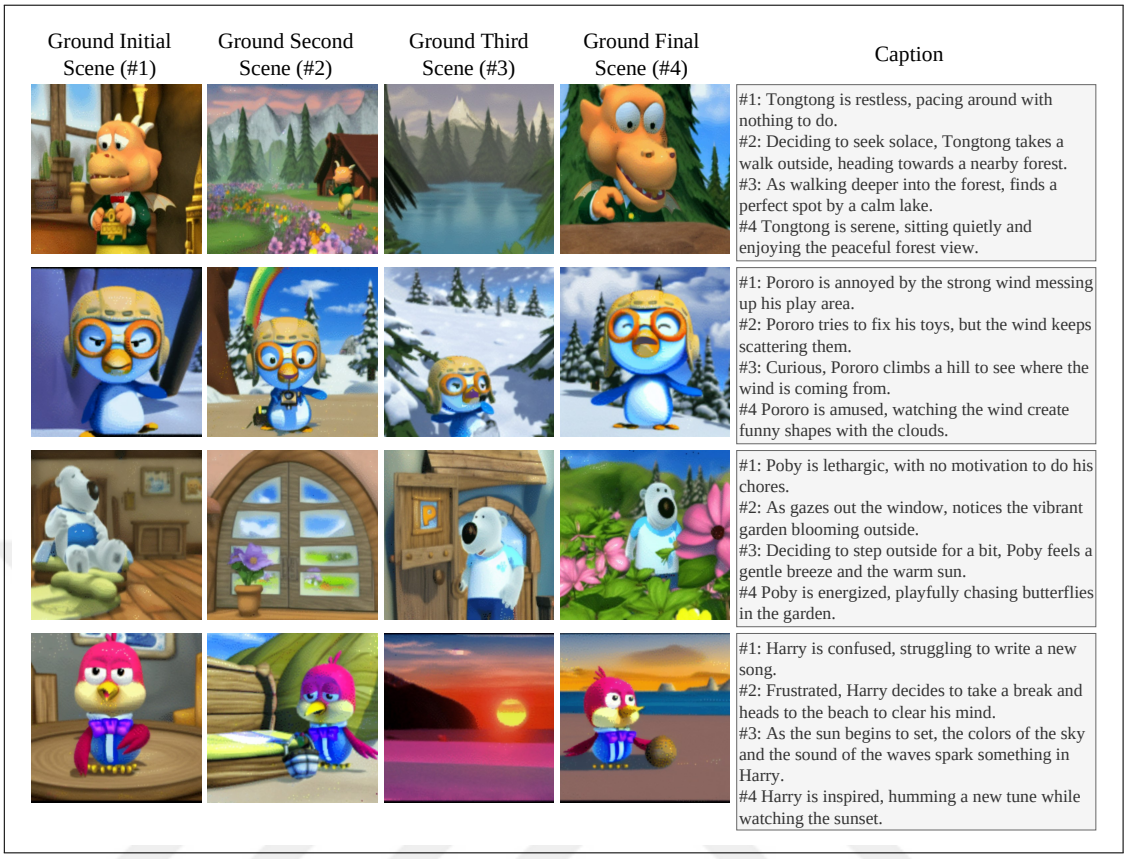

Figure 4.9. Examples of Dynamic Storytelling with Four Sequential Scenes and Captions

For training and improved generalization, a combination of chosen and rejected scenes was used. Since the initial image is always positive, the task requires selecting the appropriate class for the given last image, which can be either positive or negative. For example, the emotional_change dataset contains 562 initial chosen images with 560 different ways for chosen stories, and 562 initial images with 829 different ways for rejected stories. For the long version, there are 260 x 428 x 429 different ways for chosen stories and 260 x 428 x 360 different ways for rejected stories in sequence 3. The same calculations are applied for sequence 4.

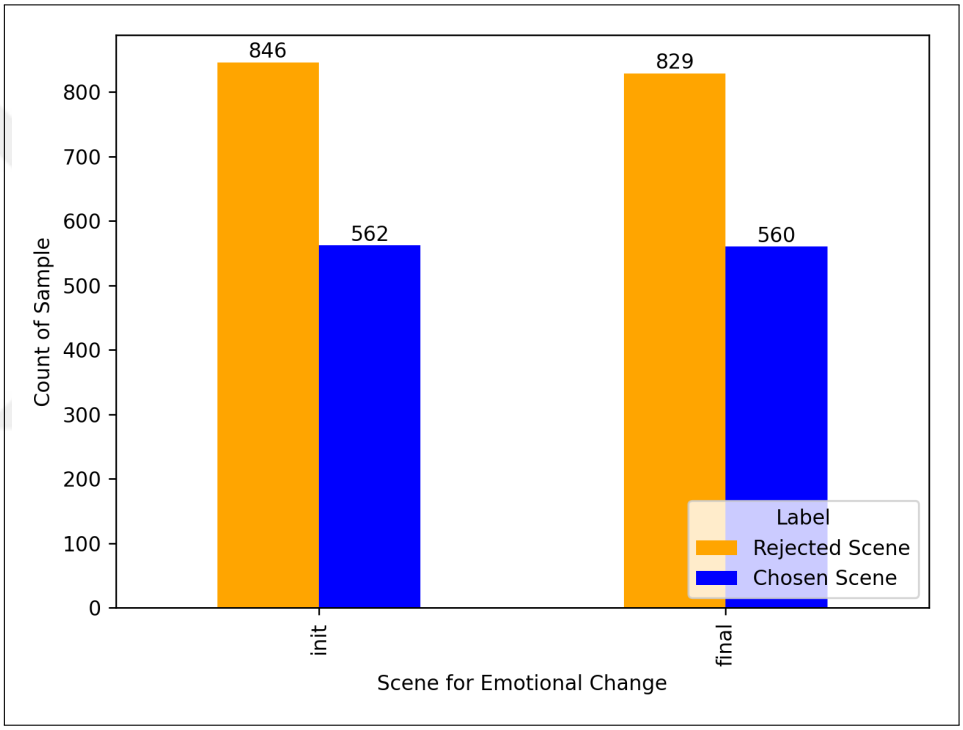

Figure 4.10. Distribution of Scene Labels for Emotional Change

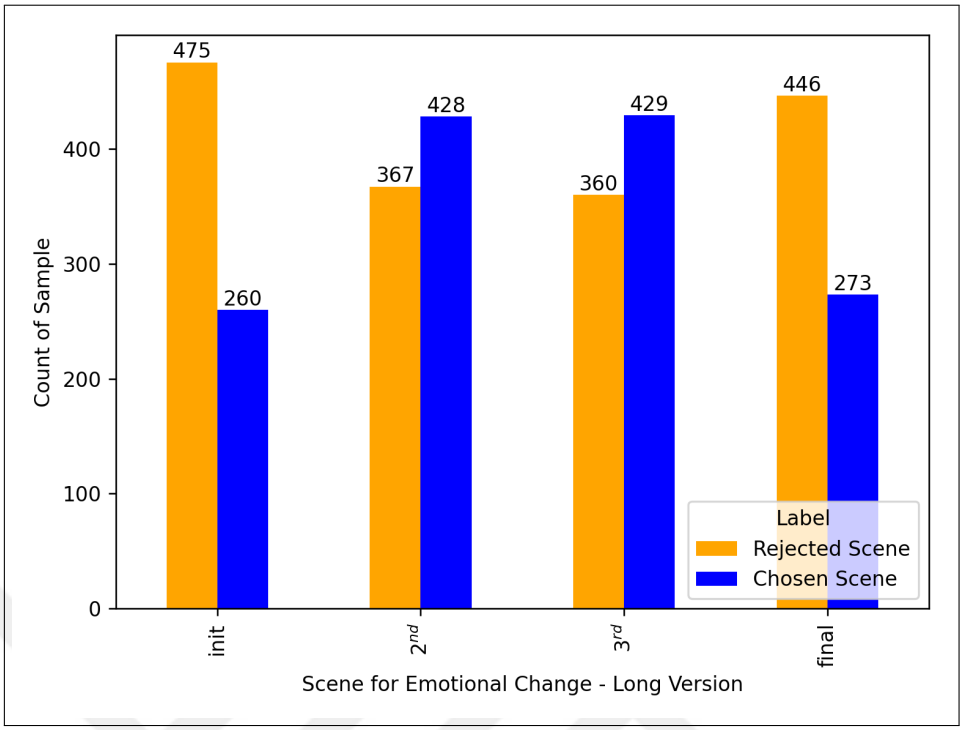

Figure 4.11. Distribution of Scene Labels for Emotional Change - Long Version

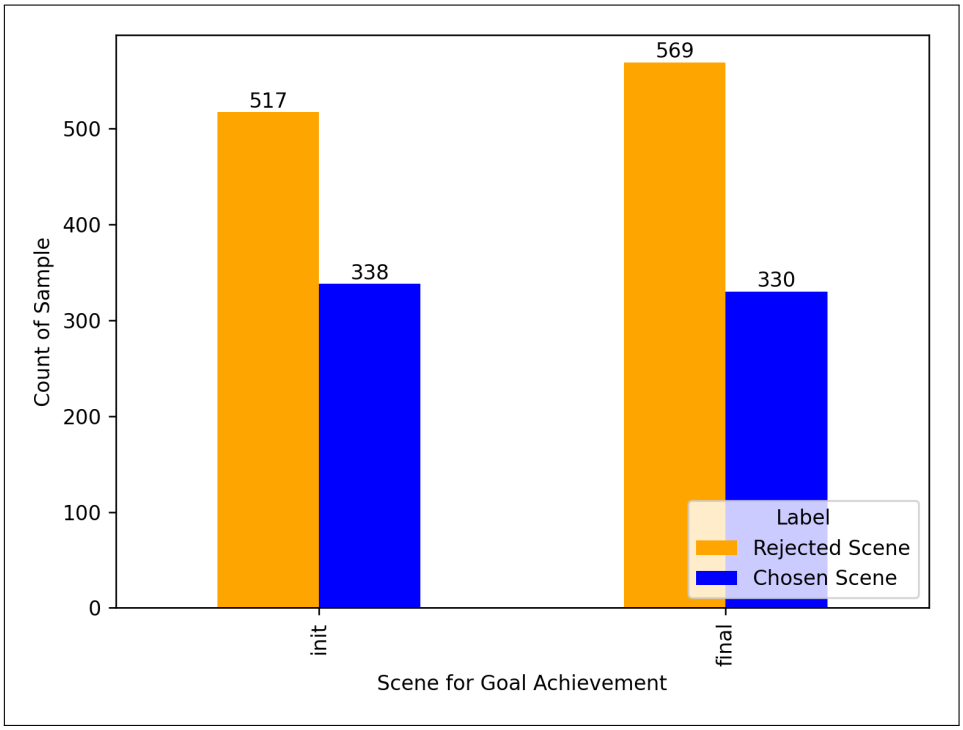

Figure 4.12. Distribution of Scene Labels for Goal Achievement

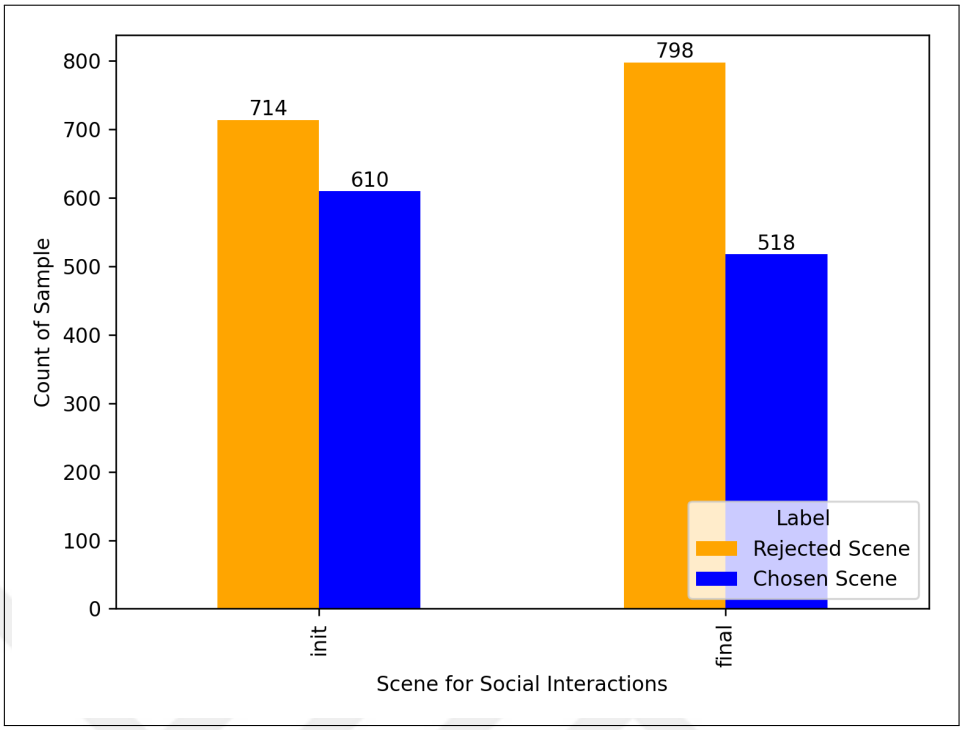

Figure 4.13. Distribution of Scene Labels for Social Interactions

4.2. TEXT TO IMAGE GENERATION

To adapt a pre-trained LDM for our specific image-caption pairing dataset, LoRa was employed as a domain adaptation strategy. The underlying principle was to fine-tune the pre-trained model with our dataset while preserving the general knowledge encoded in the model. Stable diffusion v1.5 [36] model is leveraged as a foundational LDM due to its robustness and proven capability in generating high-fidelity images. The model's parameters were kept frozen during the adaptation process to retain its pre-trained capabilities. LoRA is a parameter-efficient fine-tuning approach that introduces trainable rank-decomposed weight matrices into specific layers of the model. For the selected domain adaptation in this work the Pororo Cartoon TV-Series, LoRA is applied to the query (q), key (k), value (v), and output (w) layers of the Transformer within the LDM. This process involved the addition of trainable weights A and B while the pre-trained weights remained static. This allows for learning domain-specific features without large-scale changes to the model's structure.

The prepared dataset comprised pairs of images and captions meticulously curated for this study. During fine-tuning, only the LoRA weights were updated using the backpropagation

algorithm. The objective was to minimize the mean squared error (MSE) loss that measures the discrepancy between the predicted and true noise in the diffusion process. The DDPM sampler was utilized in the iterative generation of images. The DDPM operates by gradual denoising a sample through a series of steps, where each step predicts the noise that needs to be removed from the image representation. The MSE loss function was pivotal for training the LoRA weights. It was employed to quantify the prediction accuracy of the noise that the DDPM needs to reverse during the diffusion process. The loss computation is based on the expected noise at each diffusion step, guiding the model to produce more accurate reconstructions.

4.3. TEXT TO IMAGE SERIES GENERATION

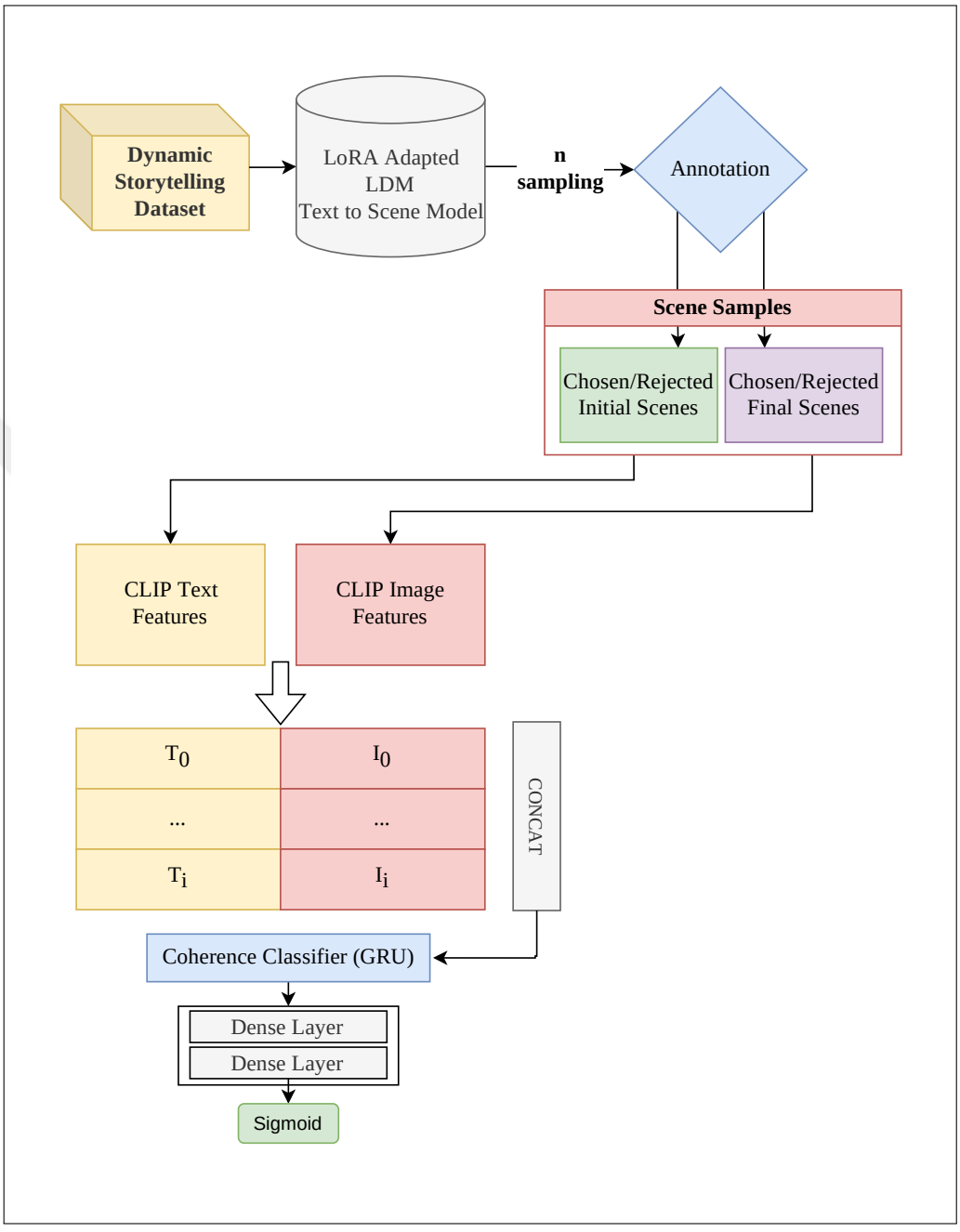

Figure 4.14. Pipeline from Annotation to Coherence Classification Model

Following the adaptation of the LDM with LoRA, a binary classification model was established to assess the coherence of image series for a specific change type c. This classifier plays a pivotal role in the generation process, determining the acceptance or rejection of scene samples. A Dynamic Storytelling Dataset, Pororo-DS, consisting of image-text pairs, was used. The dataset was annotated for coherence to reflect whether the narrative in the text

aligns with the corresponding image for each change type c. Feature extraction utilized the same CLIP model [37] , as implemented in the stable diffusion framework. This ensured consistency in the encoding representations across the textual and visual modalities. For each image-text pair, extractions of CLIP Text Features and CLIP Image Features were conducted separately, facilitating the learning of nuanced representations of coherence within the same embedding space. Scene samples were generated from the LoRA-adapted LDM Text to Scene Model by sampling n times for each change type c. These samples underwent an annotation process where they were labeled as coherent (chosen) or incoherent (rejected), resulting in two sets of scene samples: initial and final scenes. A Gated Recurrent Unit (GRU) network was employed as the Coherence Classifier. This binary classifier discerns between coherent and incoherent image-text pairings, trained on the extracted CLIP features using the annotated samples as ground truth. The classifier was trained using a binary cross-entropy loss function, optimized via an AdamW [38] optimizer.

Upon completion of training, the Coherence Classifier was integrated into the image generation process. The classifier functioned as a gating mechanism during the image sequence generation, allowing only coherent sequences to proceed for each change type c, thus ensuring the generation of coherent image sequences aligned with the textual narrative.

Algorithm 4.2. Inference Algorithm of Sequencial Image Generation with Coherence Classifier

Require: $\mathbf{I}_1$, $\mathbf{T}_1$, N, L
Initialize $InitialConcat$ with (I_1, T_1)
for $i = 2$ to L **do**
    Initialize $\mathbf{Images}_i$ as an empty list
    for $j = 1$ to N **do**
        $I_i^j \Leftarrow G(T_i)$ ▷ Diffusion Inference generating I_i^j
        Append I_i^j to $\mathbf{Images}_i$
    end for
    for $k = 1$ to $i - 1$ **do**
        $InitialConcat \Leftarrow$ Concatenate$(InitialConcat, I_k, T_k)$
    end for
    for $j = 1$ to N **do**
        $CandidateIT \Leftarrow$ Concatenate(I_i^j, T_i) ▷ Concatenate each candidate image with its corresponding text
    end for
    $I_i \Leftarrow C(InitialConcat, CandidateIT)$ ▷ Choose the best image from $\mathbf{Images}_i$
end for

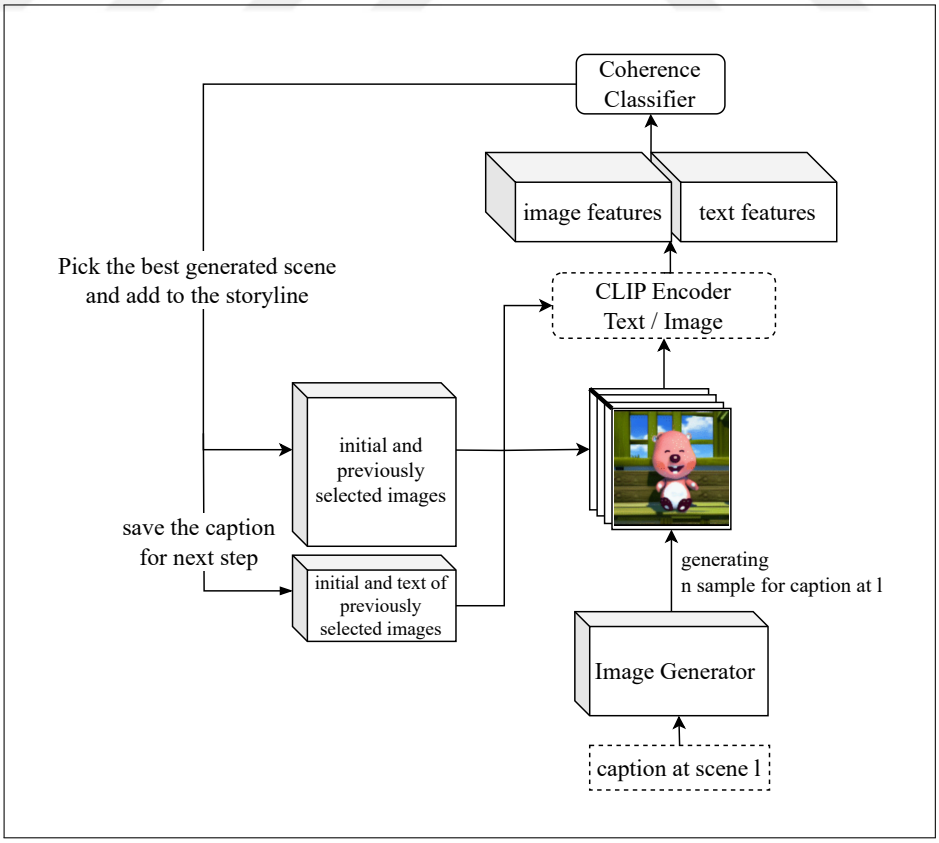

Figure 4.15. Inference Diagram of Sequential Image Generation

4.4. IMPLEMENTATION DETAILS

In this thesis, Stable Diffusion v1.5 is fine-tuned with a low-rank approach for enhancing text-to-image generation capabilities. The model undergoes training for 700 epochs with a batch size of 1, using a seed of 42. It processes images at a resolution of 512×512 pixels with a center crop technique. The learning rate is set to 1×10^{-4}, with a low-rank factor $r = 256$, and LoRA alpha set at 512. A neural network architecture called the Conditional Coherence Classifier is created to use PyTorch to combine and sort multimodal data, namely text and image features. The model's job is to figure out if an image could reasonably follow a certain scene, a task defined as binary classification. Image and text data are first processed through a GRU network, capturing the temporal dynamics essential for understanding the complex interactions between modalities. For image and text encoding, a CLIP encoder is utilized. To test the effectiveness of CLIP, the image encoder is replaced with ResNet-50, and BERT is used as the text encoder in another setting. Specifically, we use the BERT-base-uncased model, which produces encodings of 768 dimensions, and modify the ResNet-50 by removing its final layer and replacing it with a linear projection layer to match the 768-dimensional output of BERT. This is followed by several fully connected layers employing ReLU activation functions and a dropout rate of 0.05 to prevent overfitting and introduce necessary non-linearities. The architecture concludes with a sigmoid activation layer that outputs the probability of binary classification. Training employs the AdamW optimizer, with a learning rate of 2×10^{-4} and a weight decay of 0.01, alongside Binary Cross-Entropy Loss, to quantify prediction accuracy. The model is trained over 300 epochs on a CUDA-enabled GPU, specifically a Google Colab V100 GPU, enhancing performance and efficiency to meet the computational demands of processing large and complex datasets. This setup was carefully thought out to maximize deep learning's abilities to handle and make predictions based on multimodal data. It is an important part of this thesis's investigation of using advanced machine-learning techniques in the real world.

4.5. EVALUATION METRICS

In this section, various evaluation metrics used to assess the performance and quality of generated images and classification models will be discussed. These metrics are crucial for understanding how well the models perform regarding image similarity, binary classification, realism and diversity of generated images, character representation accuracy, and overall classification performance. Each subsection will delve into specific metrics, providing detailed explanations and mathematical formulations to understand their importance and comprehensive application in model evaluation.

4.5.1. Structural Similarity Index

The Structural Similarity Index (SSIM) [39] is a metric used to measure the similarity between two images. SSIM is based on the perception-based model that considers image degradation as a perceived change in structural information.

The SSIM index is calculated using various image windows. The measure between two windows x and y of common size $N \times N$ is:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4.2}$$

where:

- μ_x and μ_y are the average of x and y,
- σ_x^2 and σ_y^2 are the variance of x and y,
- σ_{xy} is the covariance of x and y,
- $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ are two variables to stabilize the division with weak denominator,
- L is the dynamic range of the pixel-values (typically this is $2^{\#\text{bits per pixel}} - 1$),
- $k_1 = 0.01$ and $k_2 = 0.03$ by default.

4.5.2. Binary Cross Entropy Loss

In training a neural network for a binary classification task, the Binary Cross Entropy (BCE) loss function is crucial for updating the model's weights. The BCE loss function measures the difference between the predicted probabilities and the actual binary class labels, penalizing incorrect predictions more heavily. This loss function helps the model learn by adjusting its weights to minimize the discrepancy between its predictions and the actual outcomes, thereby improving its accuracy over time. BCE loss function is defined as follows:

$$BCE(y, p) = -[y \log(p) + (1 - y) \log(1 - p)] \tag{4.3}$$

4.5.3. FID Score

Fréchet Inception Distance (FID) score [40] measures how similar generated images are to real images based on their feature representations. The assessment evaluates the degree of realism and diversity exhibited by the generated images.

$$d^2\left((\mathbf{m}, \Sigma), (\mathbf{m}_w, \Sigma_w)\right) = \|\mathbf{m} - \mathbf{m}_w\|_2^2 + \mathrm{Tr}\left(\Sigma + \Sigma_w - 2\left(\Sigma\Sigma_w\right)^{1/2}\right). \tag{4.4}$$

$\mathbf{m}$ and Σ refer to the mean and covariance of the features from the real images, while $\mathbf{m}_w$ and Σ_w correspond to those from the generated images. The equation first computes the squared distance between the means of these two groups, which assesses differences in their overall characteristics. It then measures the variation around the means by comparing the covariances. A lower FID score, resulting from this calculation, indicates that the generated images more closely resemble real ones, suggesting higher quality in the generated visuals.

InceptionV3 [41], a convolutional neural network pre-trained on the vast ImageNet [42] dataset, is integral to calculating the FID score. It serves as a feature extractor, turning images into feature vectors that encapsulate key visual elements like textures and shapes. When evaluating the quality of images generated by models like GANs, InceptionV3 processes both real and generated images to obtain their deep-layer feature representations. These features are then used to compute the FID score by calculating the mean and covariance of

the feature vectors and measuring the Fréchet distance between these statistical distributions. A lower FID score suggests a closer resemblance between the generated images and real ones, indicating higher image quality and better model performance.

4.5.4. Character Representation Accuracy

Yolo8 [43] has been fine-tuned as an object detection model to classify the existence of characters in generated scenes. The accuracy metrics have been enhanced to target the appearance of characters with more precision. The **Character Presence Accuracy (CPA)** assesses how well the characters mentioned in the caption text are actually present in the scene. The **Duplication Character Rate (DCR)** measures the frequency of character duplication within a scene and the presence of characters not mentioned in the caption. Finally, the **Character Exact Match Accuracy (CEMA)** evaluates whether the scene contains only the characters mentioned in the caption, ensuring that there are no additional or missing characters, thereby confirming an exact match. The corresponding metric functions can be found in the Appendix B.

4.5.5. Classification Metrics

In evaluating classification model performance, **Accuracy**, **F1 Score**, **Recall**, and **Precision**. **Accuracy** is calculated by the equation:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{4.5}$$

The **F1 Score** provides a balance between **Precision** and **Recall**, computed as:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.6}$$

Recall is defined by the formula:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{4.7}$$

Lastly, **Precision** is measured as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{4.8}$$

Recall and precision alone can be misleading, so the F1 Score is crucial. Precision measures positive prediction accuracy but does not miss positive cases, which can lead to missing significant cases. Recall also measures the model's ability to identify all positive cases but ignores accuracy, potentially including many false positives. The F1 Score is useful in situations where both the thoroughness of capturing positives and the accuracy of the predictions are equally important because it balances both the precision of the predictions and the ability of the model to identify all relevant cases.

5. RESULTS

The SD1.5-LoRa model has a maximum SSIM score value of 0.894238, a minimum value of 0.086078, and a standard deviation of 0.087. These values were calculated based on a sample size of 12202.

Table 5.1. Comparison of SSIM Scores Across Different Models for the Pororo-SV Dataset

Models	**SSIM Score**
StoryGAN	0.481
PororoGAN	0.509
Improved-StoryGan	0.521
SD1.5-LoRa	**0.548**

Table 5.2. Character Representation Accuracy Scores

Metric	**Chosen Scenes**	**Rejected Scenes**
Character Presence Accuracy (CPA)	0.8977	0.6224
Duplication Character Rate (DCR)	0.0774	0.2793
Character Exact Match Accuracy (CEMA)	0.8219	0.4554

The Figures 5.1 and 5.2 demonstrate that landscapes, being easier to reconstruct, generally achieve high scores, while scenes featuring characters receive intermediate scores. The model appears to encounter difficulties with dark scenes, indicating that these are more challenging to accurately reconstruct.

Table 5.3 compares FID scores across various models using the Pororo-SV Dataset, highlighting SD1.5-LoRa as used in this work with an FID score of 21.45. Notably, AR-LDM stands out as the most successful model, achieving the lowest FID score of 17.40. As this model is fully pre-trained using the Pororo-SV dataset without any low-rank adaptations, such a low FID score is expected.

The training and evaluation scores of the coherence model across various story dynamics and combined categories are presented in Table 5.4 5.5 respectively. The evaluation results show that the model performs best in social interaction stories with an accuracy of 84.09%.

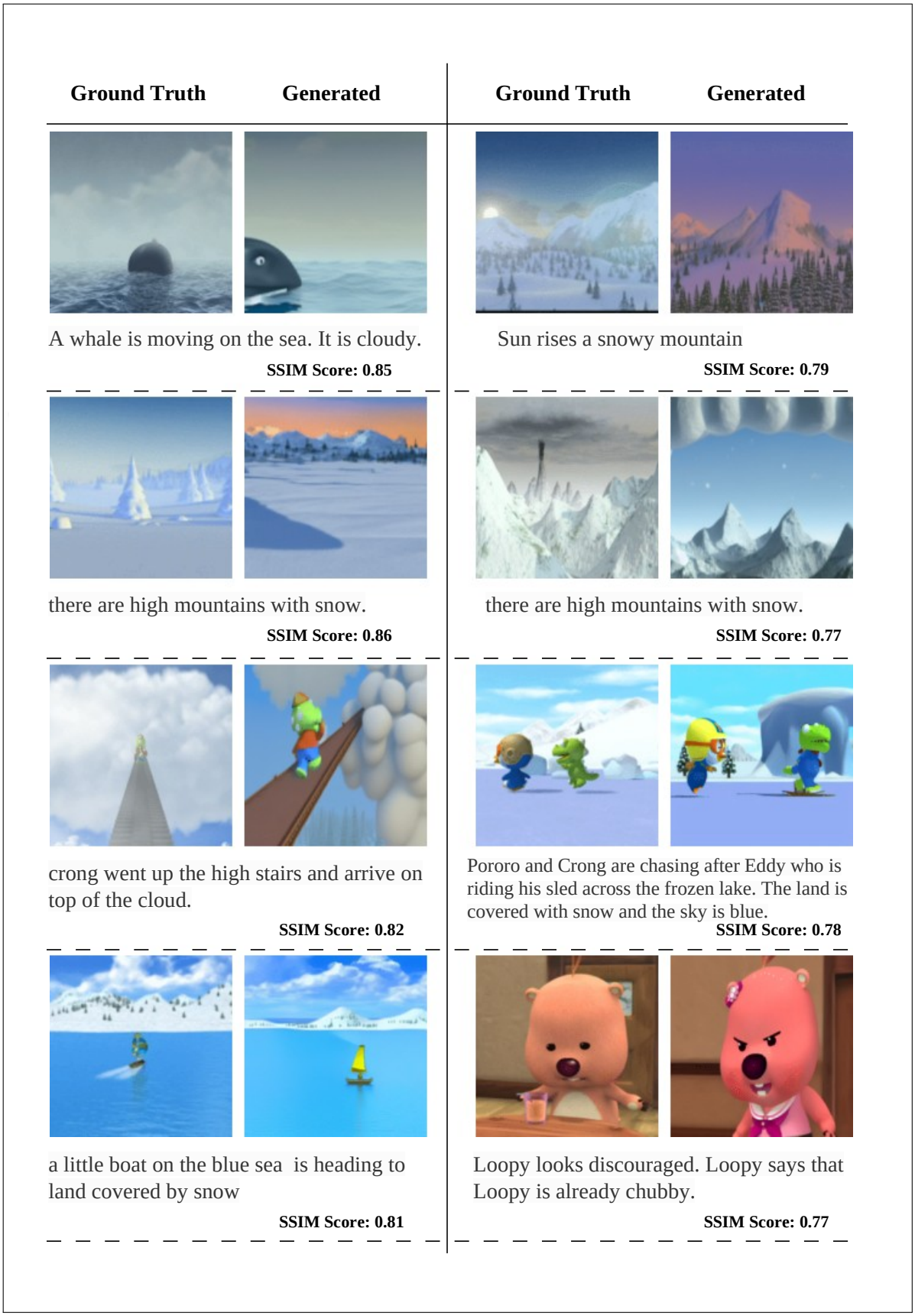

Figure 5.1. Generated Samples with High SSIM Scores Compared to Ground Truths

Figure 5.2. Generated Samples: Middle SSIM Scores on the Left, Low SSIM Scores on the Right

Table 5.3. Comparison of FID Scores Across Different Models for the Pororo-SV Dataset

Models	**FID Score**
StoryGANc (BERT)	72.98
StoryGANc (CLIP)	74.63
CP-CSV	67.76
StoryDALL·E (prompt tuning)	61.23
StoryDALL·E (finetuning)	25.90
AR-LDM	**17.40**
SD1.5-LoRa	21.45

Table 5.4. Training Scores of the Coherence Model Across Different Story Dynamics and Combined Dataset

Pororo-DS Dataset Name	**Accuracy**	**Precision**	**Recall**	**F1**
Emotional Change	0.8367	0.8448	0.875	0.8596
Goal Achievement	0.8876	0.881	0.881	0.881
Social Interaction	0.9886	0.975	0.9024	0.9873
All Stories	0.8945	0.9028	0.8966	0.8997

Table 5.5. Evaluation Scores of the Coherence Model Across Various Story Dynamics and Combined Dataset

Pororo-DS Dataset Name	**Accuracy**	**Precision**	**Recall**	**F1**
Emotional Change	0.6327	0.6818	0.8654	0.7627
Goal Achievement	0.7303	0.8286	0.58	0.6824
Social Interaction	0.8409	0.8261	0.7755	0.8
All Stories	0.7164	0.6211	0.7937	0.6969

The precision of 82.61%, a recall of 77.55%, and an F1 score of 80%. In comparison, goal achievement and emotional change stories have lower performance, with goal achievement showing an accuracy of 73.03% and emotional change with an accuracy of 63.27%. For combined story categories, the model achieves an accuracy of 71.64%.

The training scores indicate that the model exhibits high performance during training, especially in social interaction stories, with an accuracy of 98.86% and an F1 score of 98.73%. Emotional change and goal achievement stories also show strong results, with accuracies of 83.67% and 88.76%, respectively. The combined story categories achieve a training accuracy of 89.45%. These results suggest that while the model performs well during training, there is a notable drop in performance during evaluation, highlighting potential areas for further improvement and fine-tuning.

Table 5.6. Evaluation Scores of the Coherence Model Using Different Representation Encoding

Dataset+Encoder Pairs	**Accuracy**	**Precision**	**Recall**	**F1**
All Stories+CLIP	**0.7084**	**0.7346**	**0.6556**	**0.6903**
All Stories+BERT&ResNet	0.4994	0.0982	0.1745	0.1155

The study explored the effectiveness of different model architectures for coherence learning using sequential image and story text data, comparing GRU architectures and CLIP encodings versus BERT and ResNet encodings. The GRU model demonstrated superior performance, showing rapid and steady improvement in accuracy and F1 score over training and evaluation epochs, suggesting better handling of sequential dependencies and long-term coherence. Models using CLIP encodings for both text and images outperformed those using BERT and ResNet, due to CLIP's ability to capture semantic alignment and provide a unified representation. The BERT and ResNet model showed the least improvement, possibly due to less effective cross-modal coherence. The experiments indicated that GRU with CLIP encodings offered the best performance, highlighting the importance of advanced sequence modeling techniques and integrated multi-modal representations for coherence learning in this context.

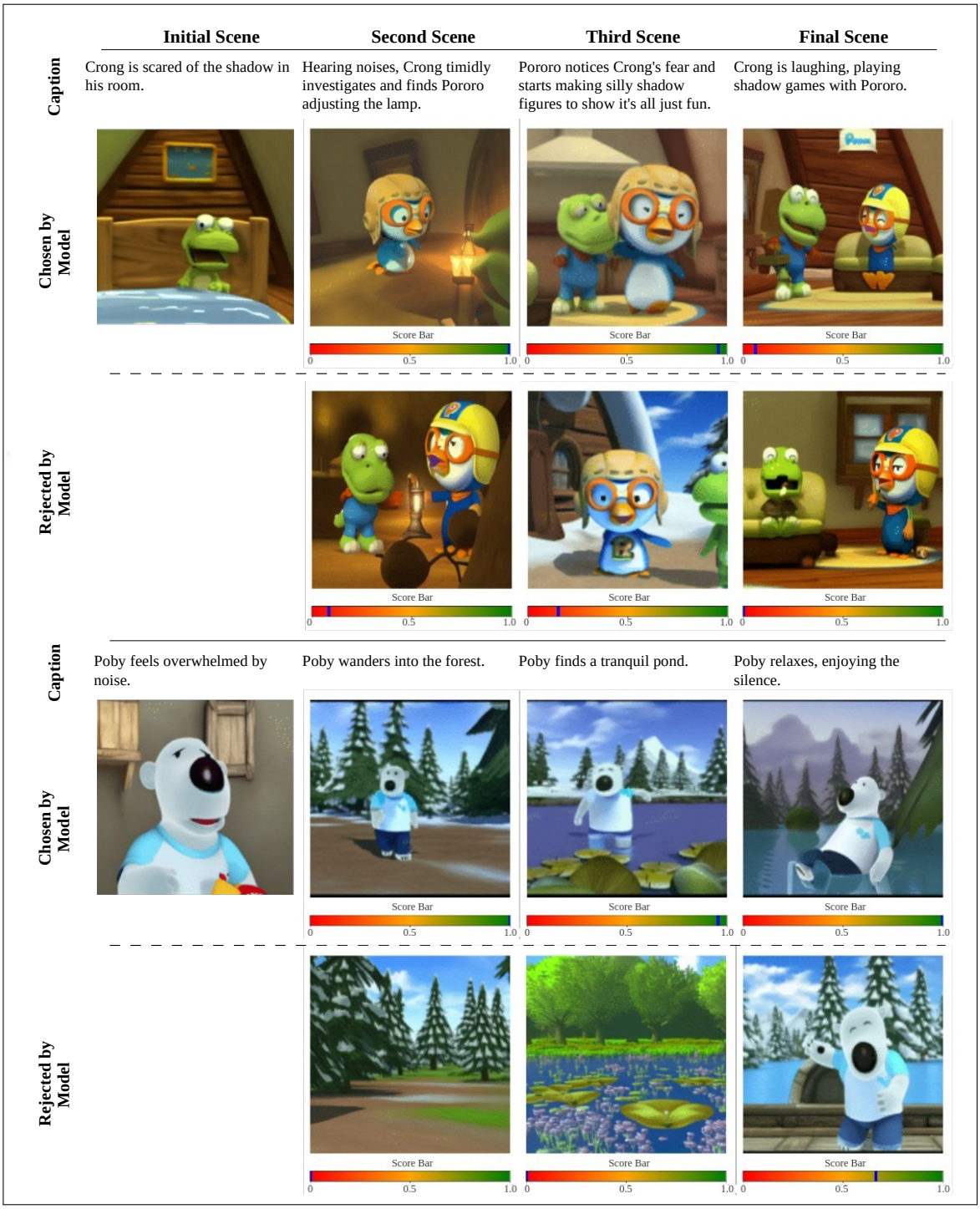

Figure 5.3. Inference Results of Coherence Classifier for Story Continuation: Four-Scene Outcomes when an Initial Scene Given Sample: 1

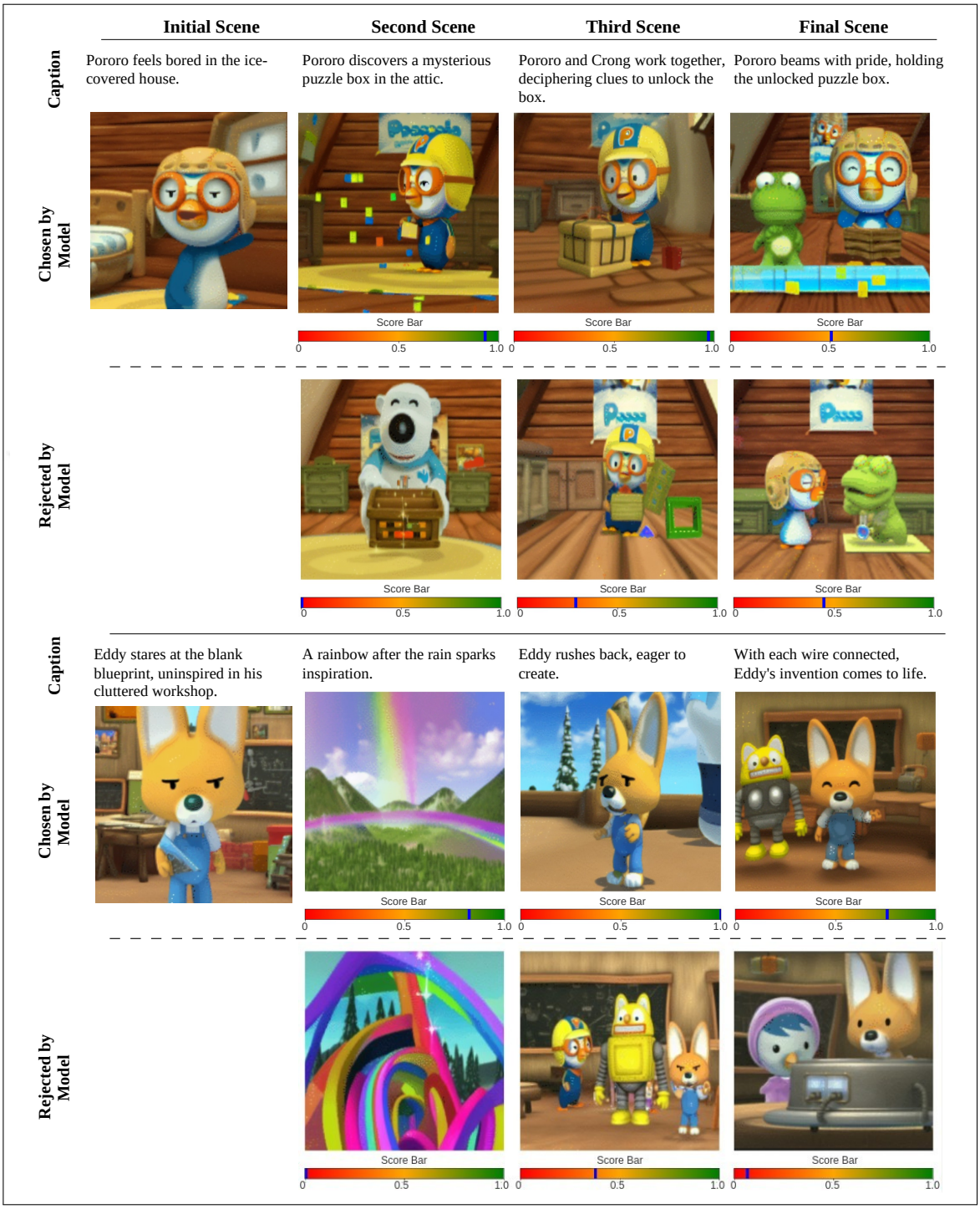

Figure 5.4. Inference Results of Coherence Classifier for Story Continuation: Four-Scene Outcomes when an Initial Scene Given Sample: 2

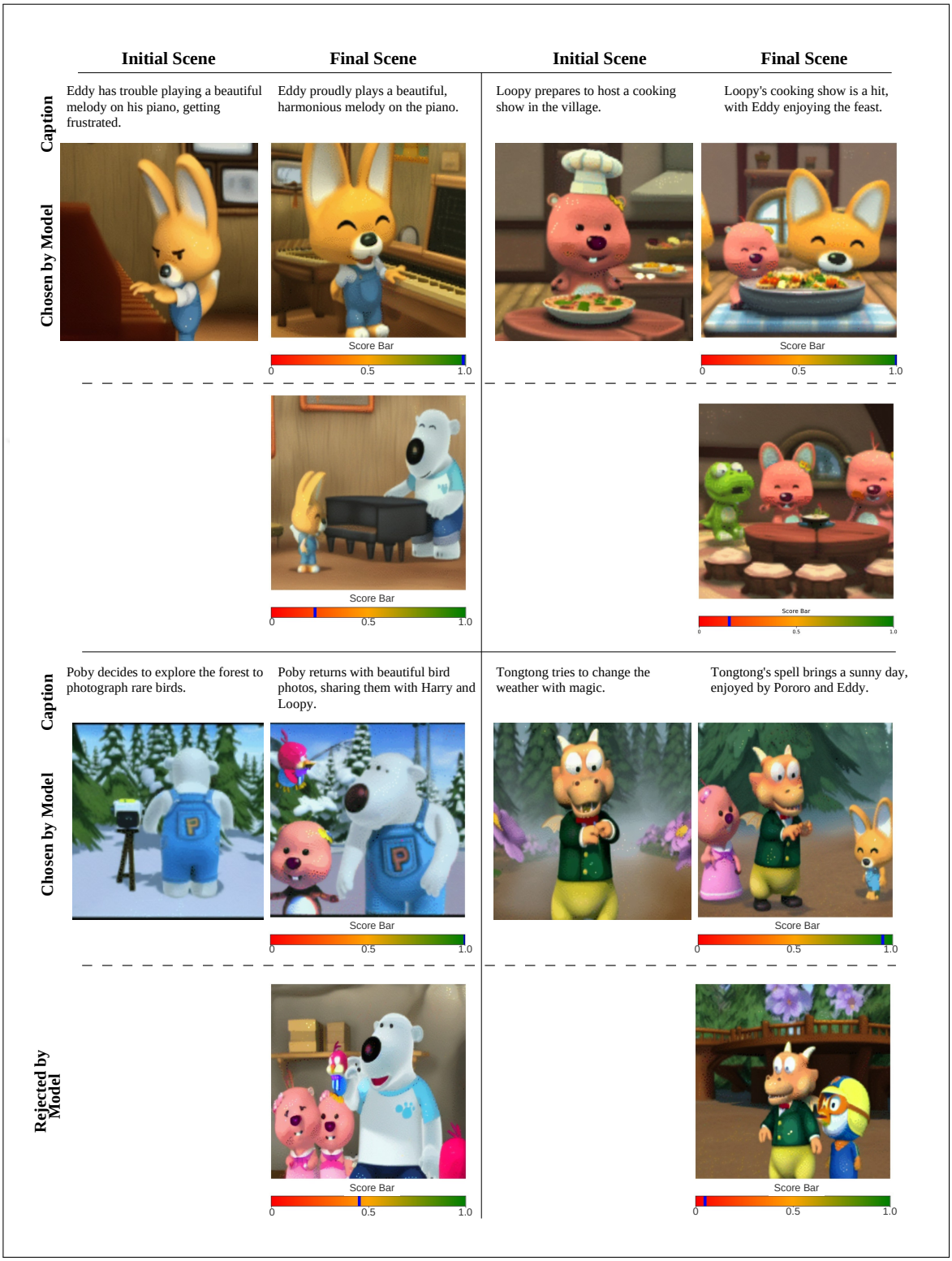

Figure 5.5. Inference Results of Coherence Classifier for Goal Achievement: Two-Scene Outcomes from an Initial Scene

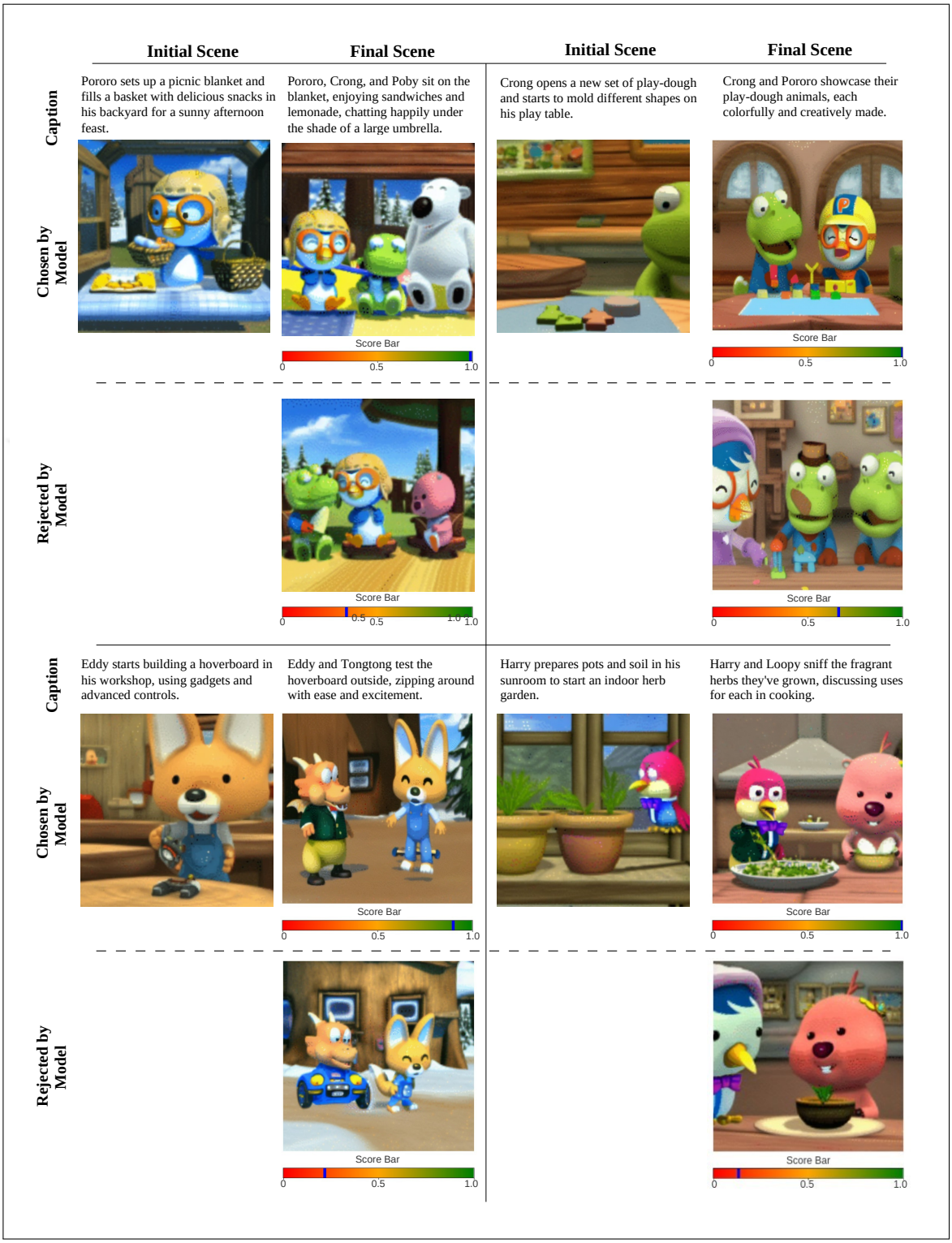

Figure 5.6. Inference Results of Coherence Classifier for Social Interactions: Two-Scene Outcomes from an Initial Scene

Figure 5.7. Comparison of Story Continuation, Example 1: Various Methods Versus Ours, with Ground Truth, Following an Initial Scene

Figure 5.8. Comparison of Story Continuation, Example 2: Various Methods Versus Ours, with Ground Truth, Following an Initial Scene

6. DISCUSSION

The process of story visualization and continuation presents a unique set of challenges, distinguishing it significantly from tasks like video generation. In video generation, the background elements and objects are usually consistent across a continuous time frame, with the primary changes being in motion and action sequences. However, story continuation involves visualizing discrete moments that may not follow a linear or predictable pattern, making the task of maintaining narrative coherence particularly complex. Unlike videos, where the continuity is inherently temporal and spatial, story visualization requires bridging significant narrative gaps between scenes, which can involve drastic changes in settings, characters, and actions. Developing a practical coherence function is one of the primary hurdles in this domain. While datasets derived from comic books and cartoons provide a basis for understanding sequential visual storytelling, they often need to catch up in capturing the nuanced reasoning and global consistency required for seamless story continuation across diverse contexts. This limitation underscores the necessity for advanced models capable of sophisticated reasoning and high computational efficiency. The complexity of ensuring narrative continuity involves visual consistency and the logical progression of the story, demanding a model that can navigate the intricacies of both visual and narrative elements.

In this work, SD1.5 LoRa demonstrates improved performance over previous models, as shown in Table 5.1. This model achieves high image quality for scene generation, closely resembling the original frames, by training only a subspace of the weights with a rank of 256. In Figure 5.1, we can see that high SSIM scores are generally for landscapes, mountainous, and forested areas. In contrast, Figure 5.2 shows that medium scores are for scenes where characters and environments are intertwined, and very low scores are for night and dark scenes.

Table 5.2 shows the character representation on the positive (chosen) and negative (rejected) scenes, providing a more detailed analysis with subcategories such as duplication and exact match. As expected, duplication is 20% higher in negative scenes compared to positive ones. CEMA and CPA are 82% and 89%, respectively, in positive scenes, indicating that the model has performed well in character selection.

In the Story Continuation comparison results, although the problem statement involves generating each scene from a single sentence, in the literature and the original Pororo-SV dataset, some scenes are generated from multiple sentences. In Figure 5.7, the model has been tested against other models at this point and has produced visuals with coherent story flow. In frame #2, the text "... Petty raises her hands ..." does not correspond to Petty raising her hand in the ground truth image. Due to many such story-scene misalignments, it is quite challenging to make comparisons with the ground truth for this problem. Similarly, in Figure 5.8, in frame #5, the caption is "Crong sighs," but in the ground truth image, Pororo is also in the scene. There is no mention of Pororo in the previous captions, making it unclear how this information appeared in the final scene.

Although this model does not provide a guarantee of consistently identifying the subsequent scene, it enhances the probability of producing a relevant scene by using a substantial sampling size, L in Algorithm 4.2. This strategy exploits the probabilistic characteristics of the model, which, with a sufficient number of trials, is highly probable to generate a logical continuation that aligns with the narrative context. Hence, while individual samples might occasionally be unrelated, the model's overall performance improves as the sampling size increases, ensuring that at least one of the generated scenes aligns with the story's progression. This method balances the inherent uncertainties in generative models and practical applications where multiple attempts can yield the desired outcome.

7. CONCLUSIONS

In this thesis, text-to-image generation techniques for narrative story visualization and continuation coherence were developed and enhanced while creating synthetic coherent data to support these advancements. By re-annotating and extending the Pororo-SV dataset to include dynamic storytelling elements, a robust foundation for training models that capture narrative transitions and character developments was provided, and utilizing LoRa for the domain chosen for the Pororo cartoon series, a pre-trained LDM was fine-tuned, resulting in the SD1.5-LoRa model. This model demonstrated significant improvements in generating high-fidelity images that closely resemble the original frames, as evidenced by superior SSIM and FID scores. Moreover, we successfully tackled the challenges of generating coherent visual sequences from textual descriptions. A Coherence Classification Model based on CLIP encodings and a GRU network was developed to ensure temporal and thematic coherence in generated image series. This classifier effectively filtered out incoherent sequences, maintaining the narrative flow and enhancing the storytelling experience. Our success in addressing these challenges demonstrates the potential of GAI in creating immersive and interactive storytelling experiences and reinforces the effectiveness of our approach. Additionally, we compared different encoding models, demonstrating that models using CLIP encodings for text and images outperformed those using BERT and ResNet encodings in terms of coherence. The evaluation metrics, including SSIM, Binary Cross Entropy Loss, FID score, and character representation accuracy, comprehensively assessed the model's performance. The results demonstrated that the methods could produce coherent and visually appealing narratives.

7.1. FUTURE WORK

The development of narrative datasets and the annotation of preferences within these collections open new avenues for research. Future work could investigate the integration of alignment methods within the diffusion process of LDMs. Such advancements would aim to refine the LDM's ability to autoregressively generate images that are not only visually compelling but also narratively consistent. By aligning the generative process more closely

with the narrative structure, researchers can work towards models that better understand and replicate the storytelling dynamics. This approach would mark a significant step forward in the field, bridging the gap between the current capabilities of image generation models and the nuanced requirements of story visualization and continuation.

Furthermore, additional research could focus on enhancing the coherence of generated sequences beyond individual frames to encompass entire video segments. By enabling models to understand events, semantics, and other relevant features within a video, it becomes possible to merge coherent video parts to create complete, cohesive movies. This would add a new dimension to storytelling, as advanced multimodal models could generate single images and seamless, narrative-driven video content. Additionally, exploring techniques to dynamically adjust the narrative flow based on user feedback or specific thematic requirements could further personalize and enhance the storytelling experience.

REFERENCES

1. Ravi H, Wang L, Muniz C, Sigal L, Metaxas D, Kapadia M. Show Me a Story: Towards Coherent Neural Story Illustration. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018:.

2. Cho K, Van Merriënboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:14091259*. 2014.

3. Huang TK, Ferraro F, Mostafazadeh N, Misra I, Agrawal A, Devlin J, Girshick R, He X, Kohli P, Batra D, Zitnick CL, Parikh D, Vanderwende L, Galley M, Mitchell M. Visual Storytelling. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics. 2016:1233-9.

4. Chen S, Liu B, Fu J, Song R, Jin Q, Lin P, Qi X, Wang C, Zhou J. Neural Storyboard Artist: Visualizing Stories with Coherent Image Sequences. *Proceedings of the 27th ACM International Conference on Multimedia*. MM '19. New York, NY, USA: Association for Computing Machinery. 2019:2236–2244.

5. Guo Y, Chen J, Zhang H, Jiang YG. Visual Relations Augmented Cross-modal Retrieval. *Proceedings of the 2020 International Conference on Multimedia Retrieval*. ICMR '20. New York, NY, USA: Association for Computing Machinery. 2020:9–15.

6. Li Y, Gan Z, Shen Y, Liu J, Cheng Y, Wu Y, Carin L, Carlson D, Gao J. StoryGAN: A Sequential Conditional GAN for Story Visualization. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019:6322-31.

7. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative Adversarial Networks. *Commun ACM*. 2020 Oct;63(11):139–144.

8. Reed S, Akata Z, Yan X, Logeswaran L, Schiele B, Lee H. Generative Adversarial Text to Image Synthesis. *Proceedings of The 33rd International Conference on Machine Learning*. vol. 48 of Proceedings of Machine Learning Research. New York, New York, USA: PMLR. 2016:1060-9.

9. Kim KM, Heo MO, Choi S, Zhang BT. DeepStory: Video Story QA by Deep Embedded Memory Networks. *ArXiv*. 2017;abs/1707.00836.

10. Zeng G, Li Z, Zhang Y. Pororogan: An improved story visualization model on pororo-sv dataset. *Proceedings of the 2019 3rd International Conference on Computer Science and Artificial Intelligence*. 2019:155-9.

11. Cer D, Yang Y, Kong SY, Hua N, Limtiaco N, St. John R, Constant N, Guajardo-Cespedes M, Yuan S, Tar C, Strope B, Kurzweil R. Universal sentence encoder for English. *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations*. 2018:169-74.

12. Song YZ, Rui Tam Z, Chen HJ, Lu HH, Shuai HH. Character-Preserving Coherent Story Visualization. *Computer Vision – ECCV 2020*. Cham: Springer International Publishing. 2020:18-33.

13. Unterthiner T, van Steenkiste S, Kurach K, Marinier R, Michalski M, Gelly S. FVD: A new Metric for Video Generation. *DGS@ICLR*. 2019:.

14. Maharana A, Hannan D, Bansal M. Improving generation and evaluation of visual stories via semantic consistency. *arXiv preprint arXiv:210510026*. 2021.

15. Lei J, Wang L, Shen Y, Yu D, Berg TL, Bansal M. Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning. *arXiv preprint arXiv:200505402*. 2020.

16. Maharana A, Bansal M. Integrating Visuospatial, Linguistic, and Commonsense Structure into Story Visualization. *ArXiv*. 2021;abs/2110.10834.

17. Maharana A, Hannan D, Bansal M. StoryDALL-E: Adapting Pretrained Text-to-Image Transformers for Story Continuation. *Computer Vision – ECCV 2022*. Cham: Springer Nature Switzerland. 2022:70-87.

18. Pan X, Qin P, Li Y, Xue H, Chen W. Synthesizing Coherent Story With Auto-Regressive Latent Diffusion Models. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2024:2920-30.

19. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer. 2015:234-41.

20. Rahman T, Lee HY, Ren J, Tulyakov S, Mahajan S, Sigal L. Make-a-Story: Visual Memory Conditioned Consistent Story Generation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023:2493-502.

21. Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*. 2020;33:6840-51.

22. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. Deep unsupervised learning using nonequilibrium thermodynamics. *International conference on machine learning*. PMLR. 2015:2256-65.

23. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, Krueger G, Sutskever I. Learning transferable visual models from natural language supervision. *International conference on machine learning*. PMLR. 2021:8748-63.

24. Yu X, Aloimonos Y. Attribute-based transfer learning for object categorization with zero/one training example. *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part V 11*. Springer. 2010:127-40.

25. Devlin J, Chang MW, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:181004805*. 2018.

26. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016:770-8.

27. Li C, Farkhoor H, Liu R, Yosinski J. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:180408838*. 2018.

28. Aghajanyan A, Zettlemoyer L, Gupta S. Intrinsic dimensionality explains the

effectiveness of language model fine-tuning. *arXiv preprint arXiv:201213255*. 2020.

29. Hu EJ, Shen Y, Wallis P, Allen-Zhu Z, Li Y, Wang S, Wang L, Chen W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:210609685*. 2021.

30. Rumelhart DE, Hinton GE, Williams RJ. Learning Internal Representations by Error Propagation. *Readings in Cognitive Science*. Morgan Kaufmann. 1988:399-421.

31. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*. 1994;5(2):157-66.

32. Gemini Team, Anil R, Borgeaud S, Wu Y, Alayrac JB, Yu J, Soricut R, Schalkwyk J, Dai AM, Hauth A, others. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:231211805*. 2023.

33. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D. Language models are few-shot learners. *Advances in neural information processing systems*. 2020;33:1877-901.

34. Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E, Le Q, Zhou D. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*. 2022;35:24824-37.

35. Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, Almeida D, Altenschmidt J, Altman S, Anadkat S, others. Gpt-4 technical report. *arXiv preprint arXiv:230308774*. 2023.

36. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B. High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022:10684-95.

37. openai/clip-vit-large-patch14 · Hugging Face [cited 2024 29 April]. Available from: https://huggingface.co/openai/clip-vit-large-patch14.

38. Loshchilov I, Hutter F. Decoupled weight decay regularization. *arXiv preprint arXiv:171105101*. 2017.

39. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*. 2004;13(4):600-12.

40. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*. 2017;30.

41. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016:2818-26.

42. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009:248-55.

43. Reis D, Kupec J, Hong J, Daoudi A. Real-time flying object detection with YOLOv8. *arXiv preprint arXiv:230509972*. 2023.

APPENDIX A: UTILIZED PROMPTS FOR DATASET PROCESSES IN LLMS

```
Create a series of short, interconnected stories for children featuring the
characters from the Pororo TV series engaging in various activities.

Each story should have a clear, simple cause-and-effect relationship.
Include basic events and objects that are easy for children to understand
and relate to.
Start each story with the characters discovering or trying something new,
and end with the outcome or resolution of their actions.
Backgrounds can be indoors or outdoors. Use a wide range of objects.
Choose characters by their names (Pororo, Crong, Eddy, Poby, Loopy, Petty,
Harry, Tongtong, Car).

Do not use phrases like "their/his/her friends."
For backgrounds, use descriptive keywords such as "house," "snowy land,"
"forest," or "sky," instead of specific locations like "Pororo's house" or
"Eddy's house."
Do not refer to the characters as friends. Choose character names randomly
to create a coherent storyline.

For each story, provide the following details in a structured JSON format:

- story_title: The title of the story described
- initial_text: A sentence describing the starting situation.
- final_text: A sentence describing the concluding situation, ensuring it's
a direct consequence of the initial action.
initial_background: Keywords describing the setting at the start of the
story.
final_background: Keywords describing the setting at the end of the story,
noting any changes from the initial setting.
initial_objects: Keywords listing significant objects involved in the
initial situation.
final_objects: Keywords listing significant objects present in the final
situation, including any new objects introduced.
initial_characters: Names and brief descriptions of the characters involved
in the initial situation.
final_characters: Names and brief descriptions of the characters present in
the final situation, noting any new characters introduced.
Focus on creating content that is engaging, cause and effect, situation
involving a pictorial change in difference and Initial and final text are
plain, non-complex texts in which only one situation is described, fosters
imagination among young readers. Each story should encapsulate a moral or
learning moment in a subtle manner.
measurable_change: succinctly describes the specific, observable
transformation or outcome from the story's start to its conclusion,
emphasizing the narrative's cause-and-effect progression.
change_type:
CHOOSE THE ACTIONS WHICH ARE MEASURABLE BETWEEN TWO PHOTOS, DONT USE VERY
ABSTRACT THINGS, YOU MAY USE CONCRETE OBJECTS TO MENTION IN THEIR ACTION
ETC.!
```

Figure A.1. Prompt for Creating the Pororo-DS Dataset: Part 1

````
EXAMPLE OUTPUT
//START EXAMPLE
```json
    {
        "story_title": "Pororo's Snowman Building",
        "initial_text": "Pororo gathers snow to build a snowman in the snowy
land.",
        "final_text": "Pororo stands next to a tall snowman, complete with a
carrot nose and coal eyes.",
        "initial_background": "Snowy land, daytime",
        "final_background": "Snowy land, daytime",
        "initial_objects": "Snow, carrot, coal",
        "final_objects": "Snowman, carrot nose, coal eyes",
        "initial_characters": [
            {"name": "Pororo", "description": "Creative, building"}
        ],
        "final_characters": [
            {"name": "Pororo", "description": "Proud, admiring"}
        ],
        "measurable_change": "Snow piles into a snowman",
        "change_type": "physical_transformation"
    },
    {
            "story_title": "Poby's Bookshelf Building",
            "initial_text": "Poby measures wood to build a new bookshelf in
his house.",
            "final_text": "Poby fills the newly built bookshelf with books,
organized neatly.",
            "initial_background": "House, morning",
            "final_background": "House, afternoon",
            "initial_objects": "Wood, measuring tape",
            "final_objects": "Bookshelf, books",
            "initial_characters": [
                {"name": "Poby", "description": "Handy, constructing"}
            ],
            "final_characters": [
                {"name": "Poby", "description": "Satisfied, organizing"}
            ],
            "measurable_change": "Wood pieces become a bookshelf",
            "change_type": "goal_achievement"
    },
    {
     ... // other stories
    }
```
//END EXAMPLE
YOU MUST USE THE WORDS PROVIDED IN THESE LISTS!
WORD_LIST_1:
"house, snow, looks, says, covered, looking, something, sky, head ...
````

Figure A.2. Prompt for Creating the Pororo-DS Dataset: Part 2

The cartoon tv-series Pororo has the following characters I will introduce them for you:

The character's names from left to right: Eddy, Loopy, Crong, Pororo, Poby is behind them the bear.

The main characters also listed in this image, the names written under the character pictures, the photos above listed as Pororo, Loopy, Eddy, Harry, Poby respectively and the photos below listed as Tongtong, Crong, Rody, Petty respectively. Be careful not to confuse Petty, Loopy, and Harry between them; Harry is a bird, Loopy is a pink beaver. Rody seems like a robotic cat, Tongtong is an orange dragon. Eddy is a fox, Poby is a white polar bear, Pororo and Petty are penguins. Petty is a female penguin. Be careful, the character name on the caption has to be precise!

The characters respectively: Eddy, Petty, Poby, Harry (which is flying), Poby, Crong

Loopy on the left, Pororo, Poby white bear in the center behind sitting in the snow, Crong, Eddy

Figure A.3. Prompt for Captioning the Pororo-SV Dataset: Part 1

Instruction: Given the above photos, recall the characters and write caption text for the given image as shown in the examples below for the characters in the scene. The caption may include which character, the action or behavior or situation, background, object, and environment, etc.

Caption: Poby holding two boxes

Caption: Eddy chooses a book from the library at home and looks at its cover.

Caption: Crong hides behind the bed and peeks her head out slightly.

... some other few-shot examples ...

[image given for annotation] Caption:

Figure A.4. Prompt for Captioning the Pororo-SV Dataset: Part 2

APPENDIX B: CHARACTER REPRESENTATION METRICS

Algorithm B.1. Character Presence Accuracy (CPA)

```python
def character_presence_accuracy(
    pred_array: list[int],
    char_name_count: dict[str, int],
    char_list: list[str]
) -> float:
    """
    Character Presence Accuracy (CPA)
    Calculate the character presence accuracy in the predictions.

    Args:
    pred_array (list[int]): List of predicted character IDs.
    char_name_count (dict[str, int]): Dictionary with character
    names as keys and their intended counts as values.
    char_list (list[str]): List of character names corresponding
    to the IDs in pred_array.

    Returns:
    float: The character accuracy, which is the proportion of
    correctly predicted characters.
    """
    pred_char_names = [char_list[id] for id in pred_array]
    pred_counts = {}
    for char in set(pred_char_names):
        pred_counts[char] = pred_char_names.count(char)
    matched_count = sum(
        min(pred_counts.get(char, 0), count)
        for char, count in char_name_count.items()
    )
    total_pred_chars = len(pred_array)
    if total_pred_chars > 0:
        accuracy = matched_count / total_pred_chars
    else:
        accuracy = 0
    return accuracy
```

Algorithm B.2. Duplication Character Rate (DCR)

```python
def duplication_character_rate(
    pred_array: list[int],
    char_count_dict: dict[str, int],
    char_list: list[str]
) -> float:
    """
    Duplication Character Rate (DCR)
    Calculate the duplication ratio in the predictions.

    Args:
    pred_array (list[int]): List of predicted character IDs.
    char_count_dict (dict[str, int]): Dictionary with character
    names as keys and their intended counts as values.
    char_list (list[str]): List of character names corresponding to
    the IDs in pred_array.

    Returns:
    float: The duplication ratio, which is the proportion of
    over-predicted or extra characters.
    """
    pred_char_names = [char_list[id] for id in pred_array]
    pred_counts = {}
    for char in set(pred_char_names):
        pred_counts[char] = pred_char_names.count(char)
    excess_count = 0

    for char, count in pred_counts.items():
        intended_count = char_count_dict.get(char, 0)
        if count > intended_count:
            excess_count += count - intended_count

    for char in pred_counts:
        if char not in char_count_dict:
            excess_count += pred_counts[char]

    total_pred_chars = len(pred_array)
    if total_pred_chars > 0:
        duplication_ratio = excess_count / total_pred_chars
    else:
        duplication_ratio = 0

    return duplication_ratio
```

Algorithm B.3. Character Exact Match Accuracy (CEM)

```python
def character_exact_match_accuracy(
    pred_array: list[int],
    char_count_dict: dict[str, int],
    char_list: list[str]
) -> float:
    """
    Character Exact Match Accuracy (CEMA)
    Check if the predicted characters and their counts
    exactly match the intended ones.

    Args:
    pred_array (list[int]): List of predicted character IDs.
    char_count_dict (dict[str, int]): Dictionary with
    character names as keys and their intended counts as values.
    char_list (list[str]): List of character names corresponding to
    the IDs in pred_array.

    Returns:
    float: 1.0 if the predictions exactly match the intended counts,
    0.0 otherwise.
    """
    pred_char_names = [char_list[id] for id in pred_array]
    pred_counts = {}
    for char in set(pred_char_names):
        pred_counts[char] = pred_char_names.count(char)

    return 1.0 if pred_counts == char_count_dict else 0.0
```