St Petersburg University

Generative AI Competitions and Open-Source Solution Analysis

St Petersburg University

RMIT GenAI and Cyber Security Hackathon

- Description
- Data analysis
- Project Tasks (challenge1-3)

RMIT GenAI and Cyber Security Hackathon------Description

Welcome to our hackathon! Participants will engage in three interconnected challenges, each designed to build upon the skills and knowledge developed in previous tasks. Together, these challenges aim to foster innovation in AI deployment, security, and adversarial testing (Private challenges).

Challenge 1: Participants will deploy an OpenAI-like API on AWS EC2, simulating a real-world environment where AI services are accessible to users. This challenge emphasizes setting up infrastructure, managing cloud resources, and ensuring the AI system is operational. Successful deployment will pave the way for securely integrating AI with other services.
https://www.kaggle.com/code/aisuko/foundational-level-challenge-1

Challenge 2: Implementing Encryption and Decryption in Python. Once the AI system is deployed, security becomes a top priority. In this challenge, participants will implement Python-based encryption and decryption to protect communications between the system and users, addressing the threat of man-in-the-middle (MITM) attacks. This challenge highlights the importance of securing data in transit, ensuring that messages are protected from interception and tampering.
https://www.kaggle.com/code/aisuko/foundational-level-challenge-2

Challenge 3: Fine-tuning an AI Model for Binary Classification of Web Traffic. With a secure foundation established, the next step is to fine-tune an AI model for binary classification of web traffic, detecting anomalies that could indicate security risks. This challenge focuses on training a model capable of identifying normal versus malicious traffic, directly tying into the need for real-time monitoring and defense mechanisms in AI-driven applications. https://www.kaggle.com/code/aisuko/advanced-level-challenge-1

RMIT GenAI and Cyber Security Hackathon------Data analysis

ID	Text	Label	Vulnerability Type	Timestamp
1	Malicious script execution attempt detected on /admin panel	1	Insider Threat	2024-09-21 14:00:00
2	Unauthorized login attempt detected from 203.0.113.5 to server 198.51.100.1	1	DDoS	2024-09-21 14:05:00
3	Failed SSH login attempt from IP 45.33.32.156 with multiple retries	1	Packet Flooding	2024-09-21 14:10:00
4	Suspicious login detected at an unusual hour from 192.168.1.101	0	None	2024-09-21 14:15:00
5	Unauthorized login attempt detected from 203.0.113.5 to server 198.51.100.1	1	Port Scanning	2024-09-21 14:20:00
6	Suspicious packet flooding detected from IP 172.217.15.110 to internal services	1	Port Scanning	2024-09-21 14:25:00
7	Worm detected spreading through internal subnet 172.16.0.0/24	1	Data Exfiltration	2024-09-21 14:30:00
8	Detected ARP spoofing attempt on internal network from 192.168.10.2	1	Reconnaissance	2024-09-21 14:35:00
9	Exploit attempt detected using outdated library version on web service	0	None	2024-09-21 14:40:00
10	Standard GET request on port 80 from 172.16.0.5 to 172.16.0.7	1	Reconnaissance	2024-09-21 14:45:00
11	Unusually high bandwidth usage detected between internal servers	1	Ransomware	2024-09-21 14:50:00
12	Incoming request on port 8080, multiple failed login attempts detected	1	Brute Force	2024-09-21 14:55:00

This dataset records different types of security events, and each record contains the following information:

- ID: numeric (integer type), used to identify a unique record.
- Text: String type (text field) containing a description of the event.
- Label: numeric (usually integer type, 0 or 1), used for classification or labeling.
- Vulnerability Type: String type (categorized field) listing the various security event types.
- Timestamp: datetime type (usually a string type, indicating a specific time).

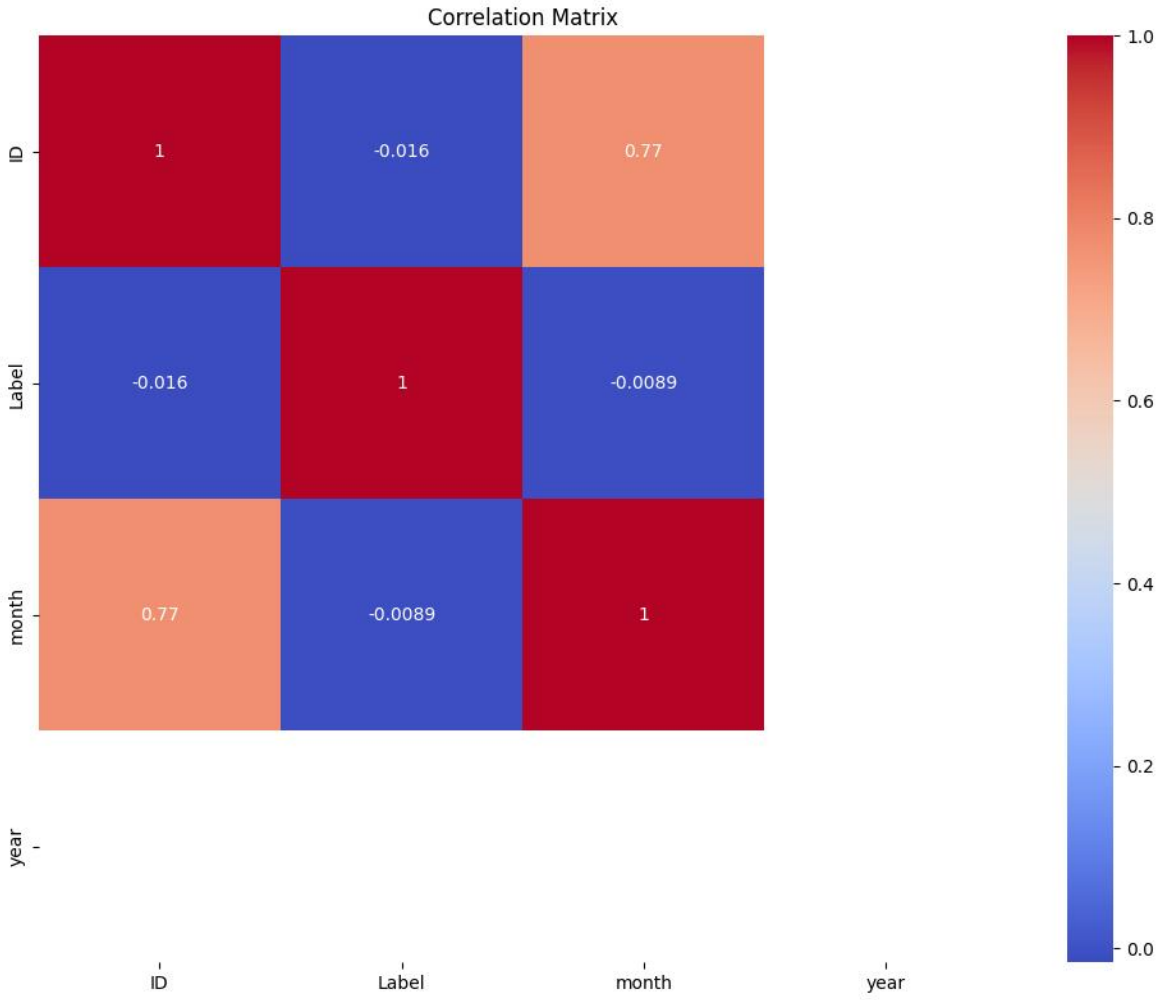
RMIT GenAI and Cyber Security Hackathon------Data analysis

Analysis Results

Analysis Aspect	Details
Missing Values	ID: 0, Text: 0, Label: 0, Vulnerability Type: 523, Timestamp: 0
Dataset Info	Total Records: 10,000, Columns: 5 (ID, Text, Label, Vulnerability Type, Timestamp)
Vulnerability Type Distribution	- SSL Vulnerability: 558 - Privilege Escalation: 558 - Reconnaissance: 558 ...
Label Distribution	Label 1 (High Risk): 9,477 Label 0 (Low Risk): 523

- Missing values: There are no missing values in the ID, Text, Label, Timestamp columns and 523 missing values in the Vulnerability Type column.
- Dataset Info: The dataset contains 10,000 rows and 5 columns of data. Column names include ID, Text, Label, Vulnerability Type, Timestamp. data types include integer (int64) and string (object).
- Vulnerability Type Distribution: The number of records for each vulnerability type is more or less evenly distributed; SSL Vulnerability, Privilege Escalation, Reconnaissance, etc. all have over 500 records.
- Label Distribution: Most of the labels were 1, indicating a high risk event, and only 523 data were labeled 0 (low risk).

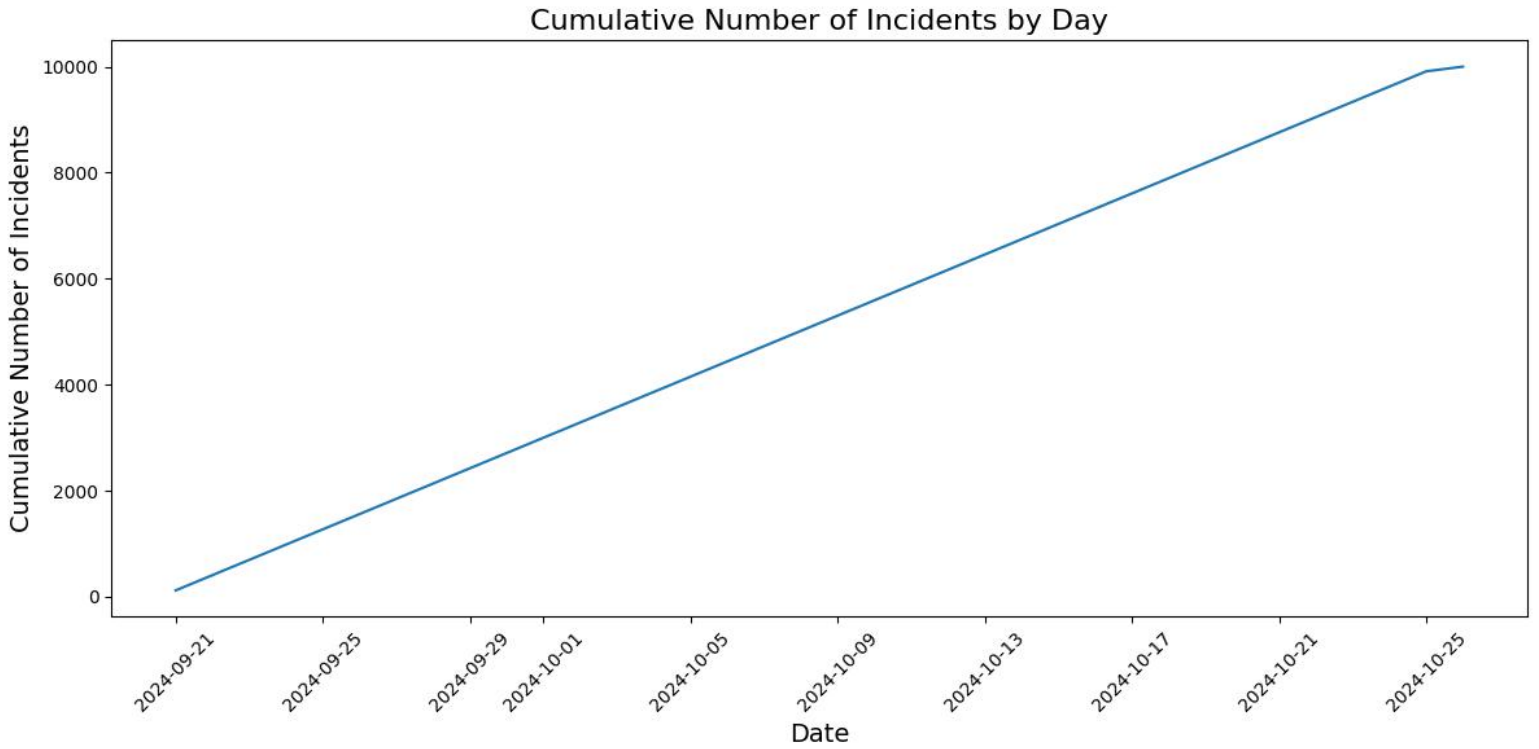
RMIT GenAI and Cyber Security Hackathon------Data analysis



This figure shows the correlation between numerical variables. It can be seen in this heat map:

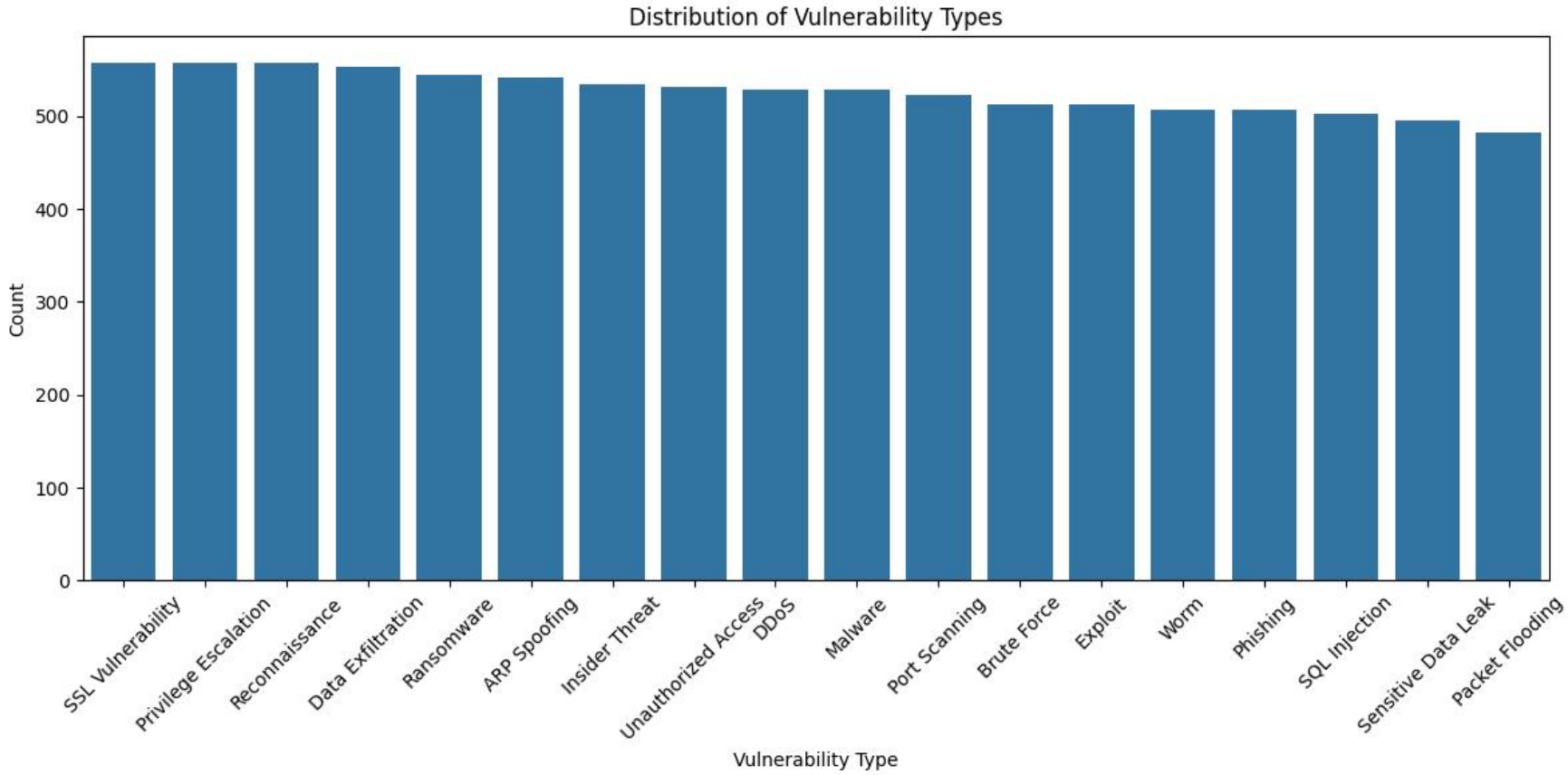
- There is a strong positive correlation between ID and month (0.77), which may indicate that events are recorded in relation to the month.
- There is almost no correlation (close to 0) between Label and the other variables (e.g. month, year, etc.), which suggests that there is no strong correlation between Label (whether or not it is high-risk) and the variables in these time dimensions.

RMIT GenAI and Cyber Security Hackathon------Data analysis



This graph shows the cumulative number of security incidents per day from September 21 to October 26, increasing steadily from about 0 to nearly 10,000 The growth curve is linear, reflecting a steady increase.

RMIT GenAI and Cyber Security Hackathon------Data analysis



The graph shows the frequency of occurrence of different types of vulnerabilities. Using the bar chart, we can see that the distribution of vulnerability types in the dataset is roughly even, with a similar number of vulnerability records for each type.

RMIT GenAI and Cyber Security Hackathon------challenge1

Objective: Deploy an OpenAI-like API on AWS EC2, simulating a real-world environment where users can access AI services. This challenge emphasizes setting up infrastructure, managing cloud resources, and ensuring the AI system operates correctly, paving the way for secure integration of AI with other services. Here is Solution Steps:

1. **Setting up the AWS Environment (using boto3):**
   - Use the boto3 library to interact with AWS EC2 services.
   - Initialize AWS credentials securely via Kaggle Secrets.
   - Set the AWS region (e.g., ap-southeast-2 for Sydney).
2. **Creating the User Data Script:**
   - Pass a User Data script when launching the EC2 instance, which runs automatically during initialization.
   - The script is used to install necessary software, set up environments, and configure the server, such as configuring Voyager.

RMIT GenAI and Cyber Security Hackathon------challenge1

3. **Initializing the EC2 Client Session:**
   - Use boto3 to initialize an AWS EC2 client session to interact with EC2 services.
   - Manage the EC2 instance lifecycle, including launching instances and configuring security groups.
4. **Creating a Security Group:**
   - A security group acts like a firewall to control inbound and outbound traffic.
   - Open specific ports (e.g., 22 for SSH, 80 for HTTP, 443 for HTTPS, and 8000 for the Voyager API) to ensure secure communication with the instance.
5. **Launching the EC2 Instance:**
   - Use Amazon Linux 2 AMI to launch an EC2 instance.
   - Set up SSH access using a key pair.
6. **Getting the Public IP Address of the EC2 Instance:**
   - After the instance is launched, retrieve its public IP address to interact with it and test the API.

RMIT GenAI and Cyber Security Hackathon------challenge1

7. Testing the API:

Once the EC2 instance is up and running, test the API to ensure everything is working correctly:

- Health Check: Verify the instance is responding correctly.
- Inference Tests: Test OpenAI and Voyager's inference APIs to ensure the deployed model is working as expected.
- Embedding and RAG Tests: Additional tests for embeddings and retrieval-augmented generation (RAG) completions.

The core objective of this challenge is to deploy an OpenAI-like API on AWS EC2 and configure it to provide secure, reliable AI inference services. Through the use of boto3 for infrastructure management, the process ensures the proper operation and security of the instance. The final goal is to securely integrate the AI system with other services, enabling automated deployment and testing.

RMIT GenAI and Cyber Security Hackathon------challenge2

In Challenge 2, the main task was to implement encryption and decryption in Python to secure communication between the system and users, protecting against Man-in-the-Middle (MITM) attacks, and ensuring data confidentiality and integrity. Here's a simplified process:

1. **Encryption and Decryption Methods:**
    - AES-CBC Mode: This mode encrypts data in fixed-size blocks (16 bytes) and requires the use of padding (PKCS7) to ensure the data size is a multiple of the block size. It uses an initialization vector (IV) for randomness in encryption.
    - AES-GCM Mode: This mode is an authenticated encryption scheme that not only encrypts the data but also provides integrity checks through an authentication tag. It uses a 12-byte IV for encryption and ensures that the ciphertext has not been tampered with.
2. **Key Generation:**
    - The PBKDF2-HMAC algorithm is used to derive an encryption key from the user-provided password and salt. This process strengthens the security of the password.

RMIT GenAI and Cyber Security Hackathon------challenge2

3. **Encryption Process:**
    - In AES-CBC Mode, a random 16-byte IV is generated, the message is encrypted, and the encrypted data is returned.
    - In AES-GCM Mode, a random 12-byte IV is generated along with an authentication tag, ensuring the integrity of the encrypted data.
4. **Decryption Process:**
    - In AES-CBC Mode, the IV is extracted, and padding is removed to recover the original message.
    - In AES-GCM Mode, the IV and authentication tag are extracted, the integrity of the data is verified, and the message is decrypted.
5. **Performance and Memory Testing:**
    - Performance tests for encryption and decryption were conducted, especially when handling large data, comparing the encryption speeds and memory usage between the modes.

RMIT GenAI and Cyber Security Hackathon------challenge2

6. **RSA Encryption:**
    - RSA encryption was implemented on the AWS instance, where the public key is used to encrypt the message, and the private key is used to decrypt it, ensuring that only users with the private key can decrypt the message.

Through this process, the challenge successfully demonstrated how to securely transmit data between the system and users, preventing data leakage and tampering, and ensuring both data confidentiality and integrity

RMIT GenAI and Cyber Security Hackathon------challenge3

Fine-tuning AI Model for Binary Classification of Web Traffic Objective: Fine-tune a BERT model to classify web traffic into two categories: "Normal Traffic" and "Vulnerability Detected." This task focuses on real-time anomaly detection and monitoring in cyber security.

1. **Environment Setup:**
   - Used Kaggle Secrets to securely access tokens for Hugging Face and WANDB for model tracking.
2. **Dataset Loading:**
   - Loaded a CSV dataset with labeled web traffic data and ensured a balanced class distribution (normal vs. malicious traffic).
3. **Data Preprocessing:**
   - Tokenized text data using BERT’s tokenizer and split the dataset into training, validation, and test sets.

RMIT GenAI and Cyber Security Hackathon------challenge3

4. **Fine-Tuning the Model:**
   - Initialized a pre-trained bert-base-uncased model for binary classification and trained it with parameters like batch size 32, learning rate 2e-5, and 5 epochs.
5. **Model Evaluation:**
   - Evaluated the model using metrics like accuracy, precision, recall, F1 score, and confusion matrix.
6. **Cross-Validation:**
   - Implemented 5-Fold Cross-Validation to ensure robust performance across different data splits.
7. **Deployment:**
   - Published the fine-tuned model on Hugging Face and created a classification pipeline for easy use.

RMIT GenAI and Cyber Security Hackathon------challenge3

8. **Tracking:**
    - Integrated TensorBoard and WANDB for real-time performance tracking.

Through this challenge, we successfully used the BERT model to perform binary classification of web traffic and demonstrated its potential applications in cybersecurity. The fine-tuned model not only accurately identifies normal and malicious traffic but also provides strong support for AI-driven security monitoring and defense mechanisms.

St Petersburg University

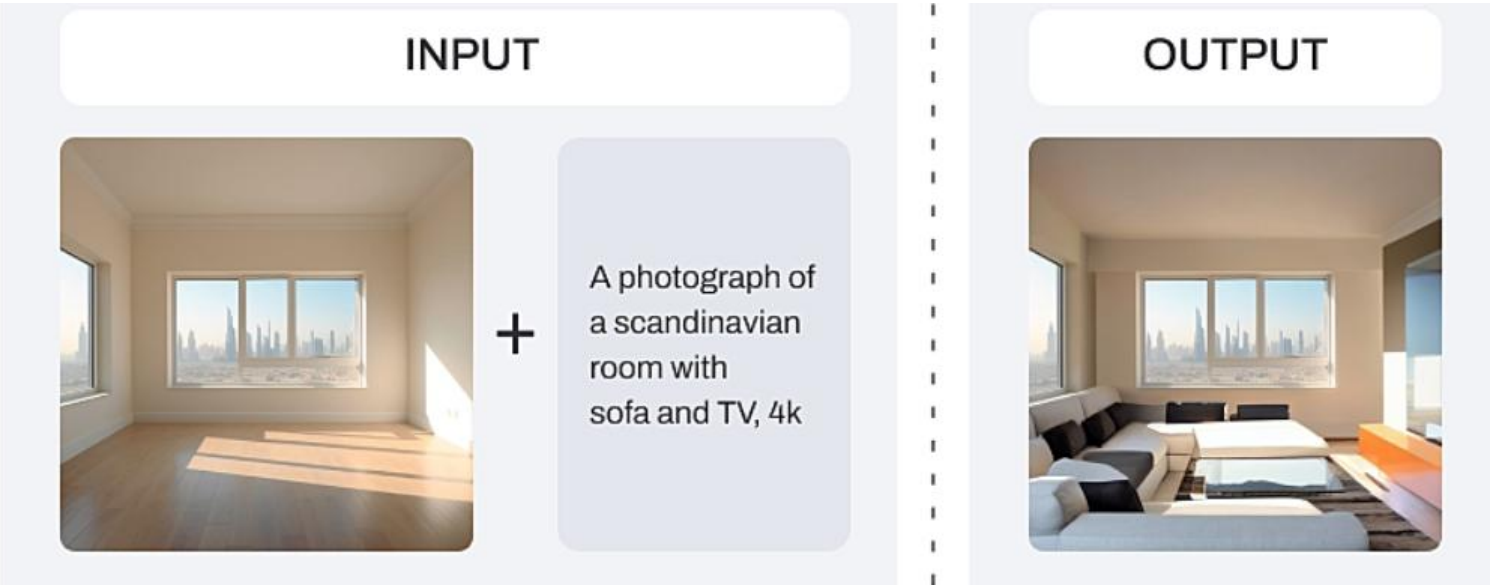
Generative Interior Design Challenge 2024

- Description
- Dataset and Features
- Challenge

Generative Interior Design Challenge 2024------Description

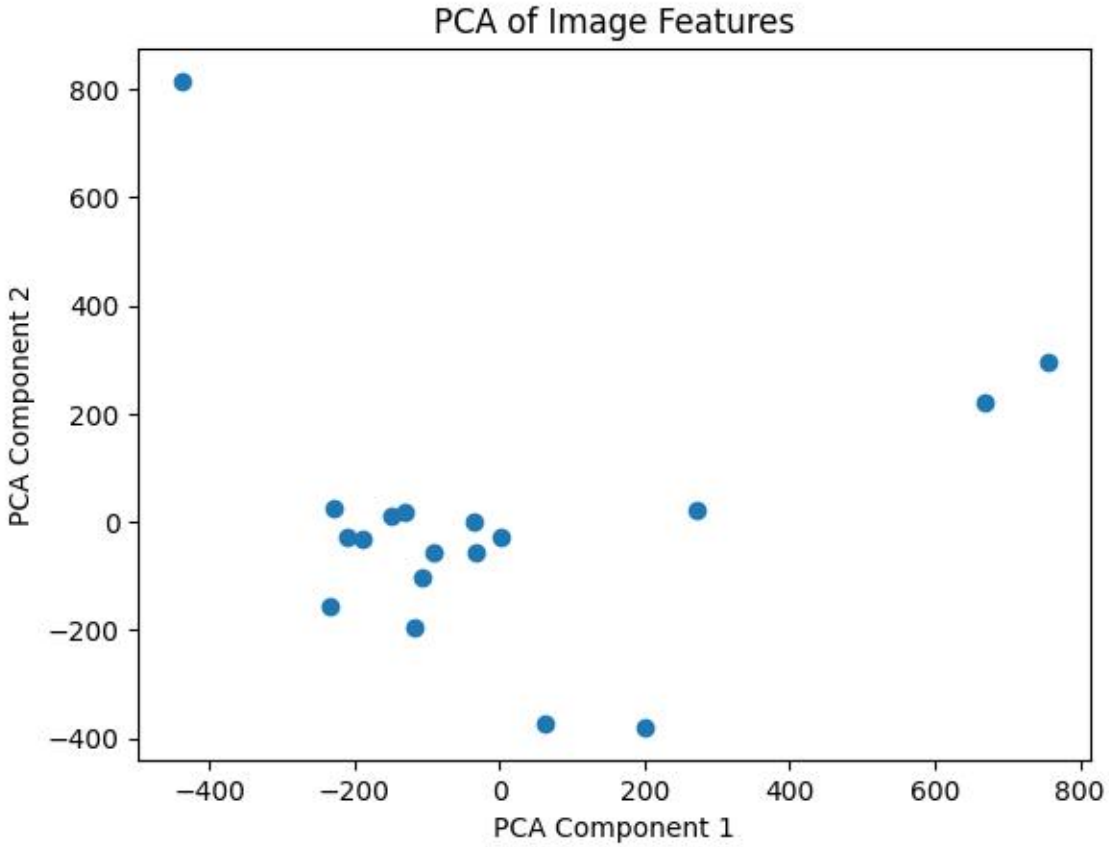
- **Objective**: Revolutionize interior design with the power of AI.
- **Challenge**: Develop an algorithm to transform a text description and an image of an empty room into a fully furnished space.

- **Technologies**:
  - Computer Vision
  - Generative AI
  - Text-to-image models
- **Goal**: Assist in visualizing potential designs for new constructions, renovations, and rental properties.

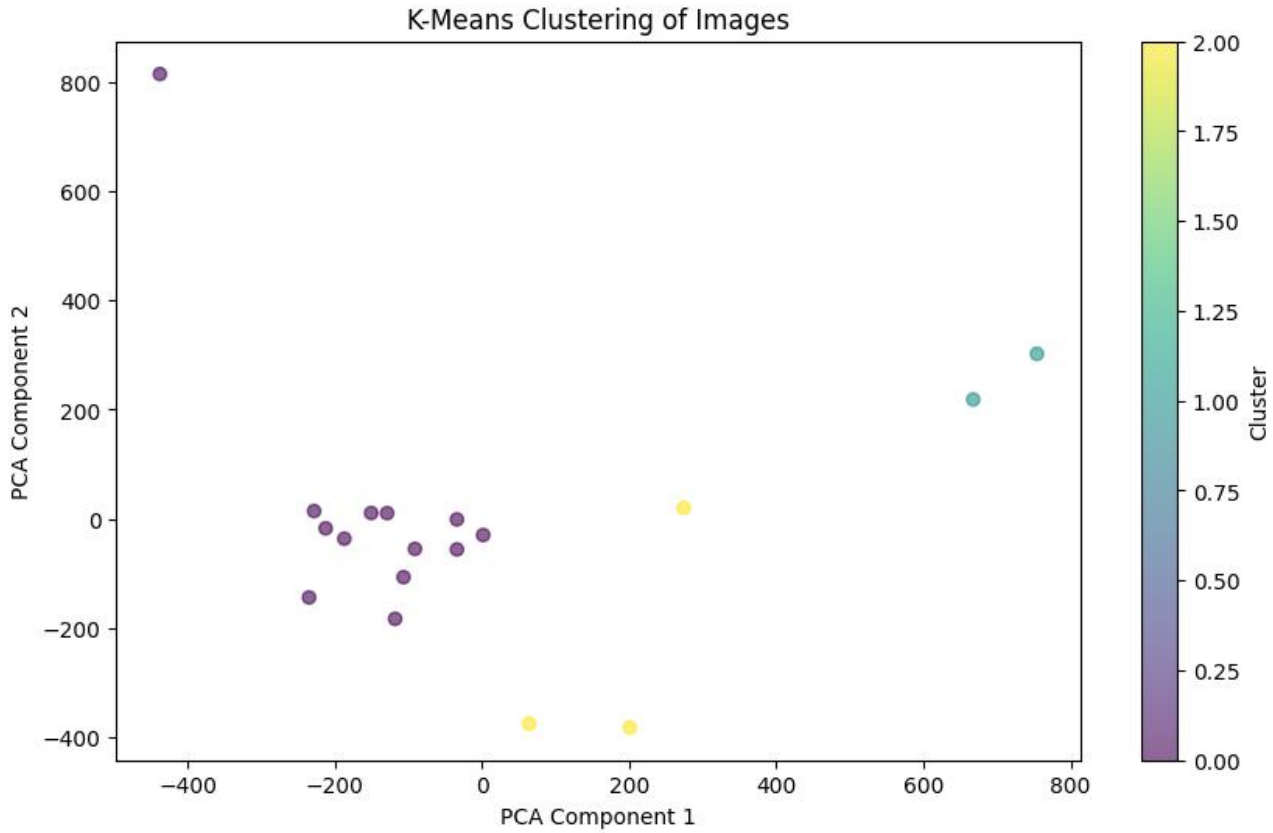


Dataset and Features

- PCA



- K-Means

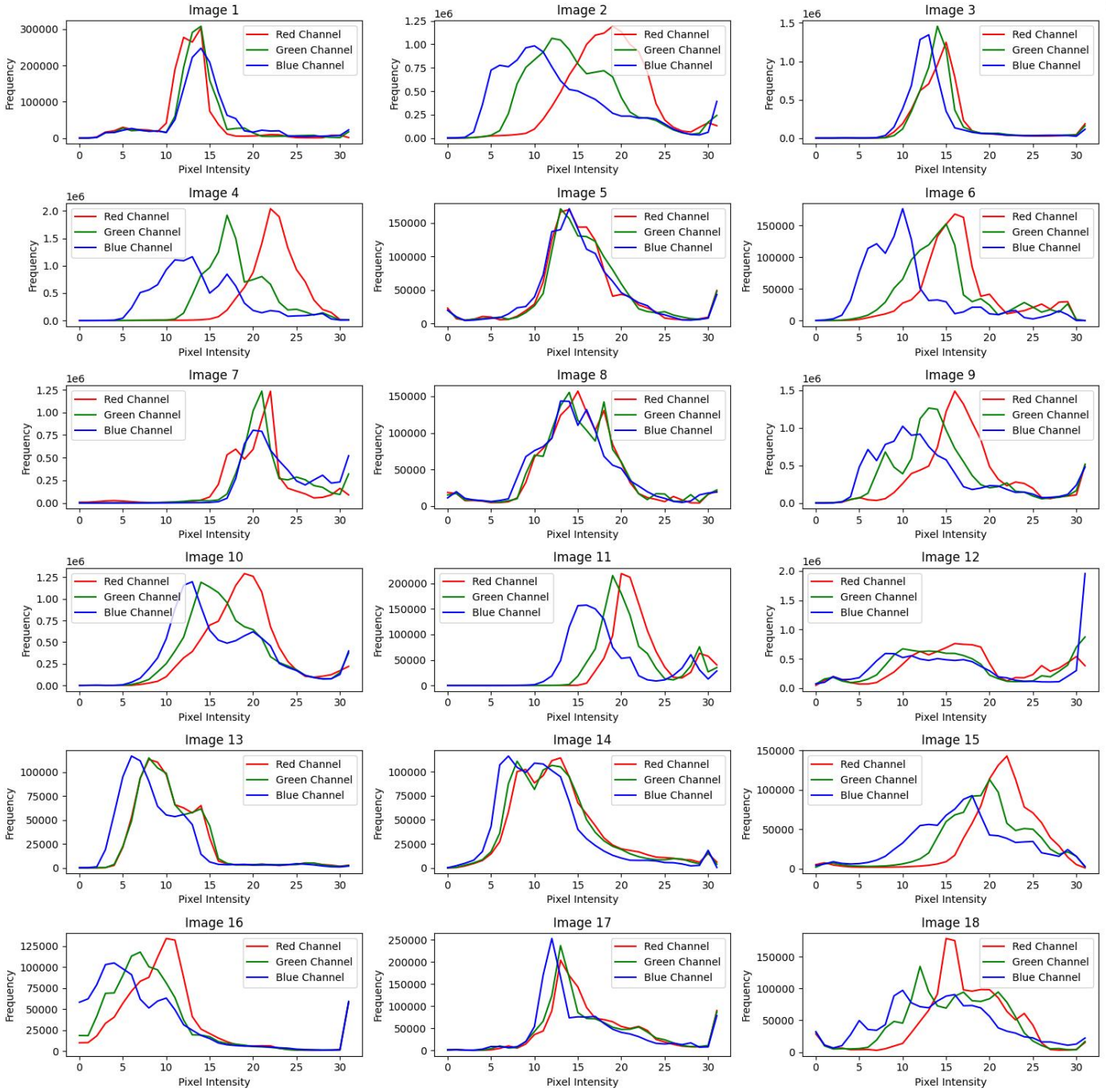


Dataset and Features

- Primary colours of the image

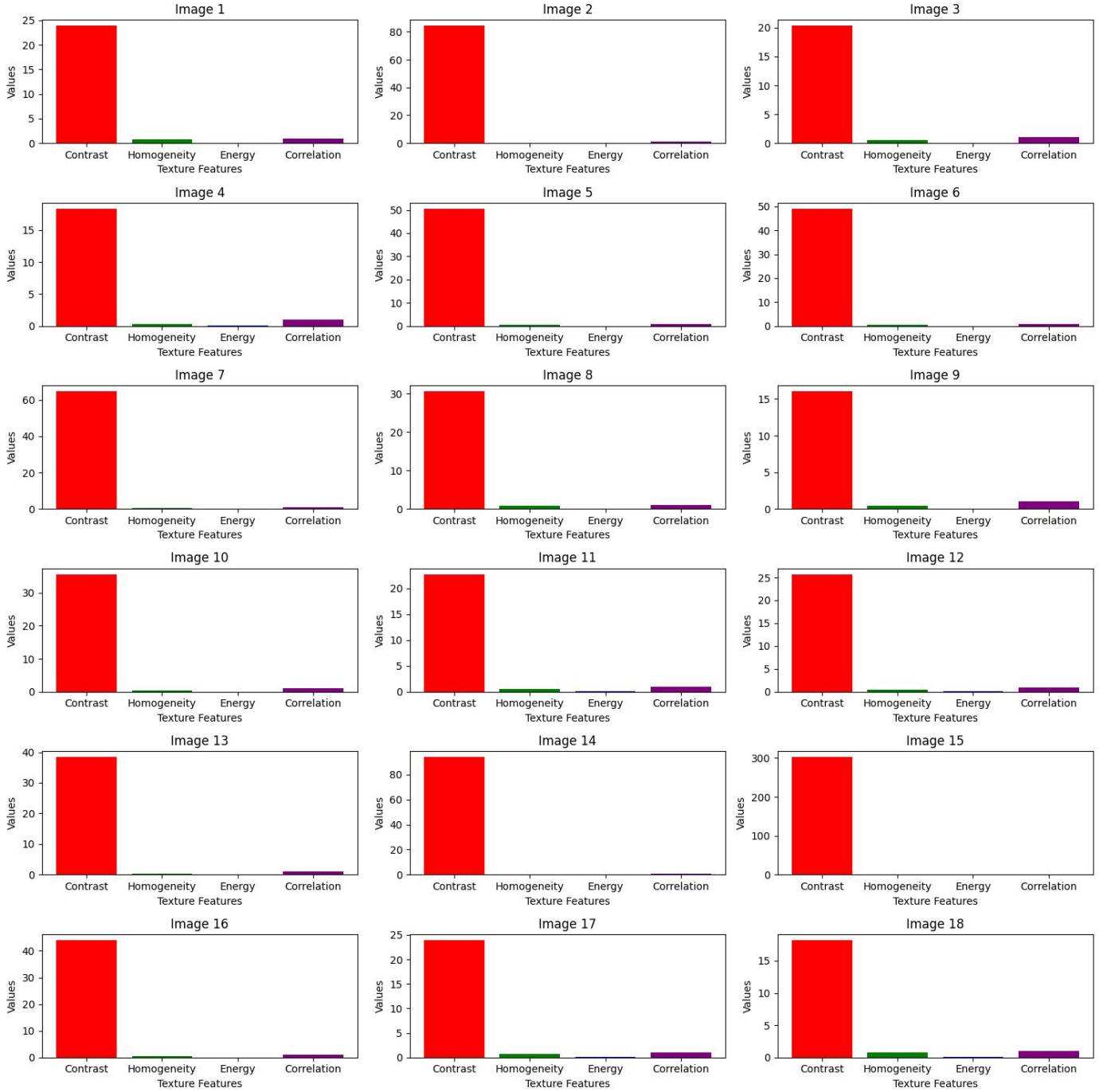
Dataset and Features

- color histogram



Dataset and Features

- Texture Features



Challenge

1. **Dataset Construction:**
    - Source: The author used the LAION5B dataset, selecting approximately 130,000 images of interior designs, covering 15 types of rooms and nearly 30 design styles.
    - Metadata Generation:
    - Description Generation: The author used the BLIP model to generate descriptive text for each image.
    - Segmentation Masks: The UperNetForSemanticSegmentation model was used to generate segmentation masks for each image.
    - Data Processing Framework: The entire data processing pipeline was built using Fondant, an open-source data preparation framework provided by ML6.

Challenge

2. **Model Training:**
   - Model Choice: The author trained a ControlNet model, utilizing the generated segmentation masks and textual descriptions as conditional inputs to generate high-quality interior design images.
   - User Control: By using segmentation masks, users could precisely control the placement and layout of objects in the room.

Challenge

3. **Application Showcase:**
    - Functionality: A community pipeline was developed that supports image-to-image (image2image) and inpainting features, allowing users to keep certain elements of the room while modifying specific parts.
    - Model Loading: Upon the first use, the model needs to be loaded into memory. To avoid simultaneous requests, a queuing mechanism similar to Gradio was implemented to ensure only one user request is processed at a time.
    - Automation: During the demo, the system automatically calculated the segmentation masks and categories for each image, simplifying the user experience and eliminating the need for users to manually create segmentation masks in external tools.

Our Selection

We chose the **Generative Interior Design Challenge 2024** mainly because:

- The competition combines computer vision with creative design, allowing us to leverage both technology and artistic expression, which we find really exciting.
- It covers areas like image generation and natural language processing, giving us the opportunity to apply various technologies to solve real-world problems.
- Compared to other competitions, this project allows us more space to be creative, as it’s not just about implementing the technology but also adding our own design ideas, which perfectly matches our expectations.

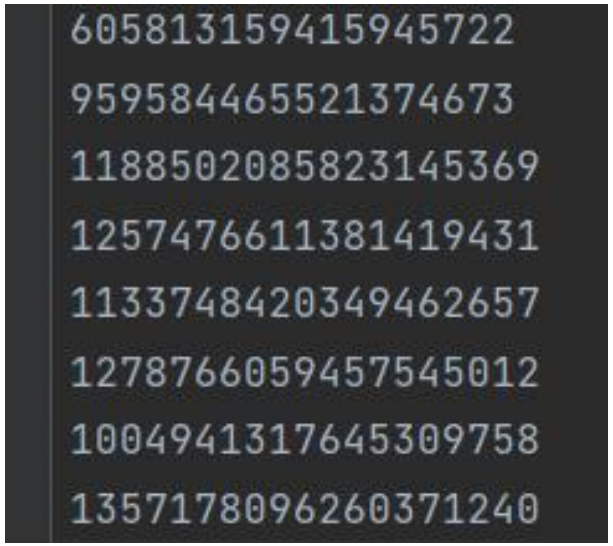
St Petersburg University

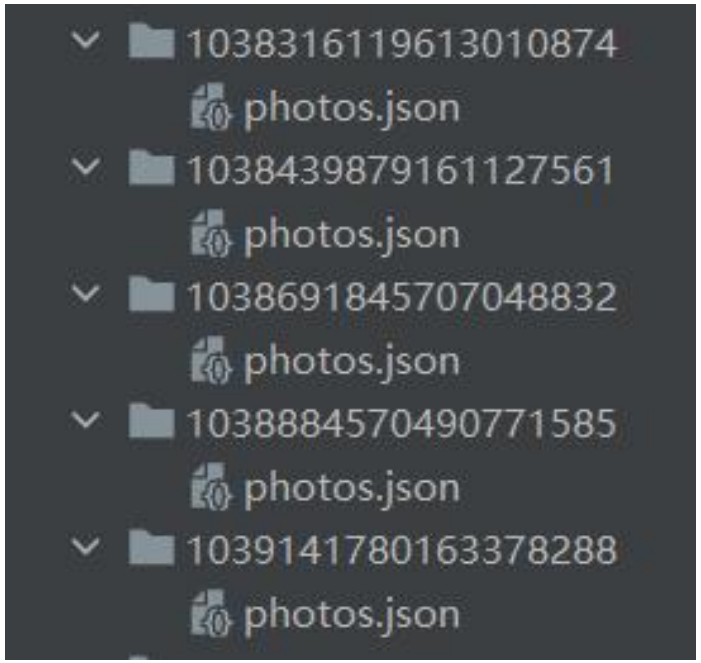
Indoor Design Project Code Implementation

Data Preparation

Using Airbnb API to Fetch Images:

- Download listings from Airbnb: Filter by city and price, and store the listing data as listings.txt.
- Download image metadata for each listing: Output the filtered listing data as image metadata (photo.json).

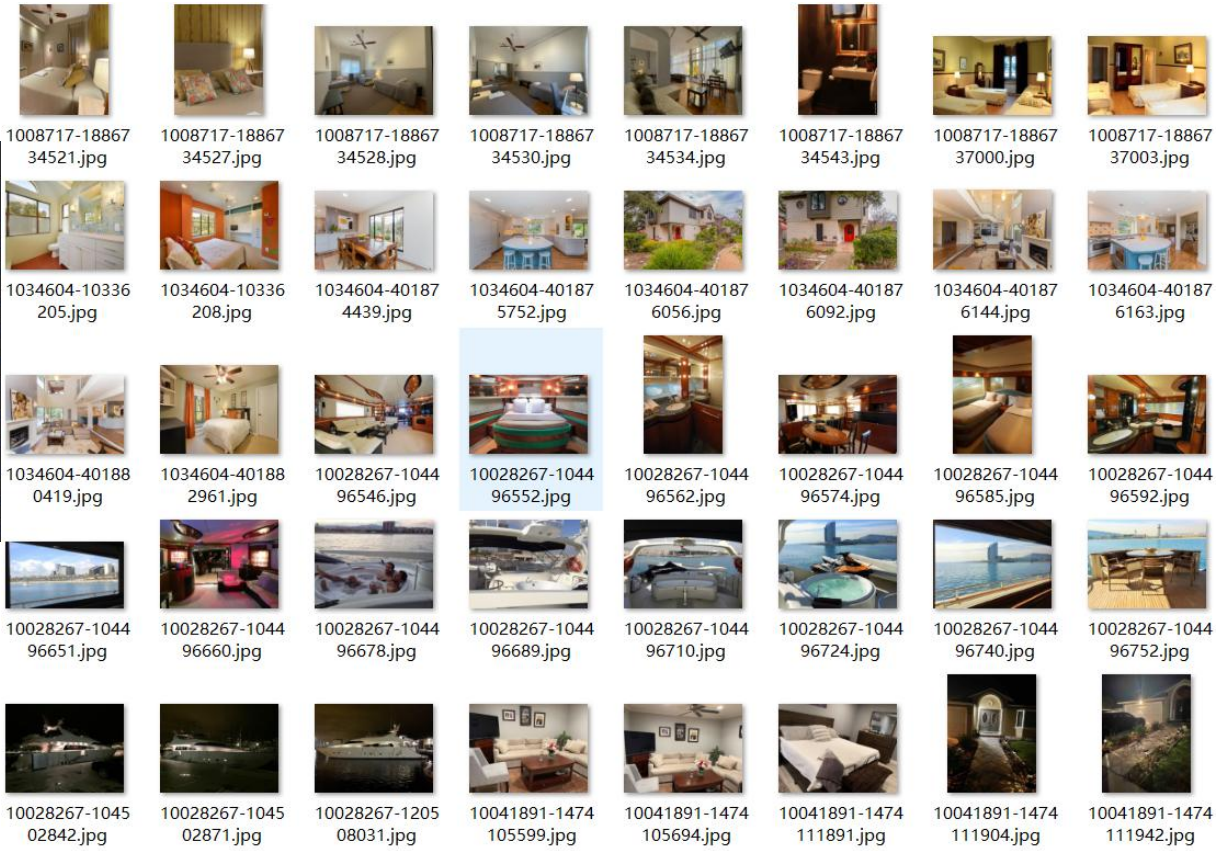




Data Preparation

- Create a single TSV file: Extract image URLs and titles from photo.json,store them in photo.csv.
- Batch download room images: Use multi-process downloading with run_downloader().

listing_id	photo_id	url	caption
1000016109686428185	1911629703	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629714	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629744	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629749	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629768	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629787	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629802	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911629810	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1854879233	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1854879249	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1854879257	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1854879274	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	
1000016109686428185	1911627082	https://a0.muscache.com/im/pictures/prohost-api/Hosting-1000016109686428185/	



Data Processing

- **Filter Out Outdoor Images:** Use a pre-trained Places365 CNN model to filter out outdoor images.
- **Filter Out Bathroom Images:** Use segmentation maps to filter out bathroom images.
- **Generate Segmentation and Depth Masks:** Generate segmentation and depth masks for each image.
- **Generate Captions for Furnished Images:** Add captions to each furnished image.

Training Data

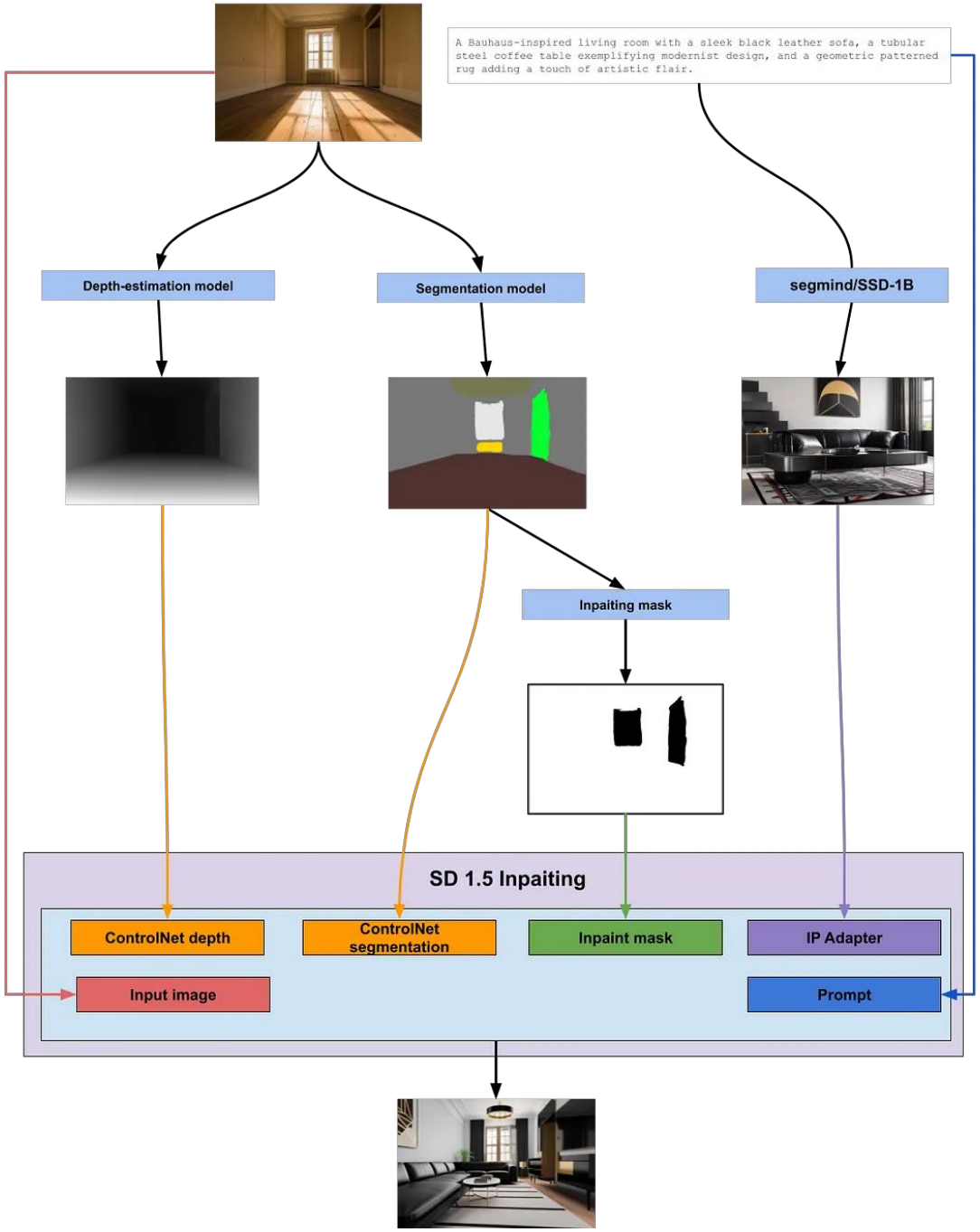
ControlNet for Segmentation: Use segmentation images to train ControlNet.

ControlNet for Depth: Use depth-estimation images to train ControlNet.

Why Train Two ControlNets?:

- Segmentation Model: Provides information about room structure (e.g., windows, walls, doors). Helps with room layout preservation but may require additional techniques.
- Depth Model: Better at room layout preservation but may not be sufficient alone. Combining both models improves results.

Training Data



St Petersburg University

Benchmark Analysis for Text-to-Image Generation

benchmark 1

Data Extrapolation for Text-to-image Generation on Small Datasets

An Innovative Approach for Enhancing T2I Models on Limited Data

https://paperswithcode.com/paper/data-extrapolation-for-text-to-image

Introduction

- Text-to-image generation (T2I) models typically require large-scale datasets to produce high-quality images.
- Many real-world applications rely on small datasets, limiting the model's ability to generate diverse and realistic images.
- Traditional data augmentation methods (e.g., cropping, flipping, mixing) fail to introduce new information, offering only marginal improvements.
- This project proposes a linear extrapolation-based data augmentation approach, leveraging internet image retrieval and outlier detection to significantly improve text-to-image generation performance on small datasets.

Core Ideas

Data Augmentation via Linear Extrapolation:

- Applies extrapolation only to text features.
- Retrieves new images from the internet using search engines, guided by dataset text descriptions.
- Extrapolated text descriptions enhance dataset diversity and better align retrieved images with dataset features.

Outlier Detection for Data Quality Improvement:

- Cluster Detector: Uses CLIP embeddings to cluster images and remove those that deviate significantly from dataset distributions.
- Classification Detector: Trains a fine-grained classifier to verify whether retrieved images match dataset categories.

Core Ideas

NULL-Guidance for Improved Score Estimation:

- Introduces a NULL text prompt (e.g., "a picture of bird") to refine the image generation process.
- Stabilizes generated outputs and maintains consistency with expected styles.

Recurrent Affine Transformation (RAT) for Better Text-Image Alignment:

- Enhances the consistency of text embeddings across transformer layers.
- Ensures better fusion of text features into the image generation process.

Implementation Process

Collecting Similar Images:

- Uses search engine APIs to collect images based on dataset labels (e.g., "Scandinavian living room" for interior design).
- Retrieves hundreds of thousands of additional images to enhance training diversity.

Data Cleaning & Linear Extrapolation:

- Cluster Detection: Filters out irrelevant images based on distance in CLIP feature space.
- Classification Detection: Ensures new images belong to the correct category by using a trained classifier.
- Text Extrapolation: Generates new descriptions for retrieved images using local manifold learning.

Implementation Process

Training the Diffusion Model:

- Fine-tunes a latent diffusion model on the expanded dataset.
- Incorporates Recurrent Affine Transformation (RAT) to improve text-image alignment.

Generating Images:

- Uses NULL-Guidance to refine score estimation and stabilize image generation.
- Synthesizes high-quality images from Gaussian noise via diffusion.

Results & Performance

The proposed method was tested on CUB (birds), Oxford (flowers), and COCO (general objects) datasets. Evaluations were conducted using Inception Score (IS) and Frechet Inception Distance (FID):

Methods	IS(Fine-tune) ↑		IS(ImageNet) ↑		FID(Fine-tune) ↓		FID(ImageNet) ↓		
	CUB	Oxford	CUB	Oxford	CUB	Oxford	CUB	Oxford	COCO
StackGAN++	4.04	3.26	4.04	3.26	23.96	48.68	15.30	32.33	81.59
AttnGAN	4.36	-	4.36	-	-	-	23.98	-	35.49
DAE-GAN	4.42	-	-	-	-	-	15.19	-	28.12
DM-GAN	4.75	-	-	-	-	-	16.09	-	32.64
DF-GAN	5.10	3.80	4.96	3.92	17.23	18.90	14.81	22.56	21.42
RAT-GAN	5.36	4.09	5.00	3.95	13.91	16.04	10.21	18.68	14.60
GALIP	-	-	-	-	-	-	10.05	-	5.85
VQ-Diffusion	-	-	-	-	-	-	10.32	14.10	13.86
U-ViT	-	-	-	-	-	-	-	-	5.45
Ours	**6.56**	**4.35**	**6.37**	**4.11**	**7.91**	**8.58**	**6.36**	**9.52**	**5.00**

- Higher IS and Lower FID indicate better performance.
- The proposed model outperforms baselines across all datasets.

Conclusion

This study proposes an innovative data augmentation approach combining:

- Search engine data retrieval.
- Linear extrapolation.
- Outlier detection.
- NULL-Guidance.
- Recurrent Affine Transformation (RAT).

The approach significantly improves text-to-image generation performance on small datasets, producing more realistic and diverse images.

benchmark 2

GLIGEN: Open-Set Grounded Text-to-Image Generation

Enhancing Text-to-Image Models for Open-Set Scenarios

gligen/GLIGEN: Open-Set Grounded Text-to-Image Generation

Introduction

- Text-to-image generation is a popular research direction in computer vision and NLP.
- Existing methods are limited to predefined categories and struggle with the open-set problem (unseen objects/concepts).
- GLIGEN proposes an innovative approach to address this challenge by enabling generation for diverse and unseen descriptions.

Research Objective

- To enable text-to-image generation to handle unseen objects and concepts by introducing the open-set concept.
- Improve the flexibility and realism of generated images.

Datasets

- **COCO and Conceptual Captions:** Train and test basic text-to-image generation capabilities.
- **Open Images and Visual Genome:** Provide diverse object annotations for complex visual-textual relationships.
- **Additional Open-Set Data:** Validate GLIGEN's ability to generate meaningful images for unseen objects using external knowledge.

Methods Overview

- **Core Idea of Open-Set Processing:** Leverage external knowledge to generate images from new textual descriptions.
- **Multimodal Knowledge Fusion:** Integrate visual and linguistic information through shared multimodal encoding.
- **Open-Set Learning Mechanism:** Map textual and visual representations to a unified space using external resources.
- **Image Generation Process:** Align text and image features, use conditional generation, and predict unseen objects via external knowledge.

Methods Overview

- **Loss Functions and Training:** Reconstruction, multimodal alignment, and open-set consistency losses ensure realistic and text-aligned image generation.
- **Role of External Knowledge Bases:** Enhance generative capabilities through conceptual and semantic extensions.
- **Reasoning and Generation:** Dynamically adapt to input text and external knowledge for diverse and creative image generation.

GLIGEN's Contributions

- Improves controllability of text-to-image diffusion models by allowing users to specify object positions and layouts.
- Introduces a new control layer for generating invisible objects without changing original model weights.
- Outperforms SOTA on zero-shot layout-to-image tasks, demonstrating the power of fine-tuning and augmentation.

Conclusion

- GLIGEN addresses the open-set problem in text-to-image generation by leveraging external knowledge and multimodal fusion.
- Enables flexible, realistic, and creative image generation for unseen objects and concepts.

St Petersburg University

Model improvements

Method Improvements

- Adjusted T2I Model

  Competition 2nd Place: SSD(1B) Model

  Ours: GLIGEN Model

- Enhanced image generation quality

  Use some evaluation metrics to check if the images generated by our model better than the Competition 2nd Place

Comparison of LLM, SSD-1B, and GLIGEN

	LLM (Large Language Model)	SSD-1B (Stable Diffusion Model)	GLIGEN (Grounded-T2I)
Key Differences	Processes and generates text, but does not create images	Generates images from text but lacks spatial control	Generates images with spatial control using text + bounding boxes
Primary Function	Text understanding, generation, and reasoning	AI-driven creative image generation	Structured image generation with spatial constraints
advantages	Excels in text-based tasks, supports diverse NLP applications	Produces high-quality, diverse images from text prompts	Provides precise object placement, useful for controlled AI-generated design
disadvantages	Cannot generate images directly, relies on other models for visual tasks	Limited control over object placement, relies on prompt quality	Requires bounding box input, more complex setup than text-only models

Reasons for choosing GLIGEN

- Use ControlNet to control the depth, edges, etc. of the room.

  Mainly control room style, e.g. keep doors and windows in position.

  It cannot do text-to-image transformation independently.

- For text + image generation tasks, it is difficult for SSD-1B to control the placement of objects.

  So we additionally use GLIGEN to do the text-to-image transformation.

Images generated by Top2 in competition

Stable Design demo

Prompt

An elegantly appointed bedroom in the Art Deco style, featuring a grand king-size bed with geometric bedding, a luxurious velvet armchair, and a mirrored nightstand that reflects the room's opulence. Art Deco-inspired artwork adds a touch of glamour

Images generated by our group

Stable Design demo

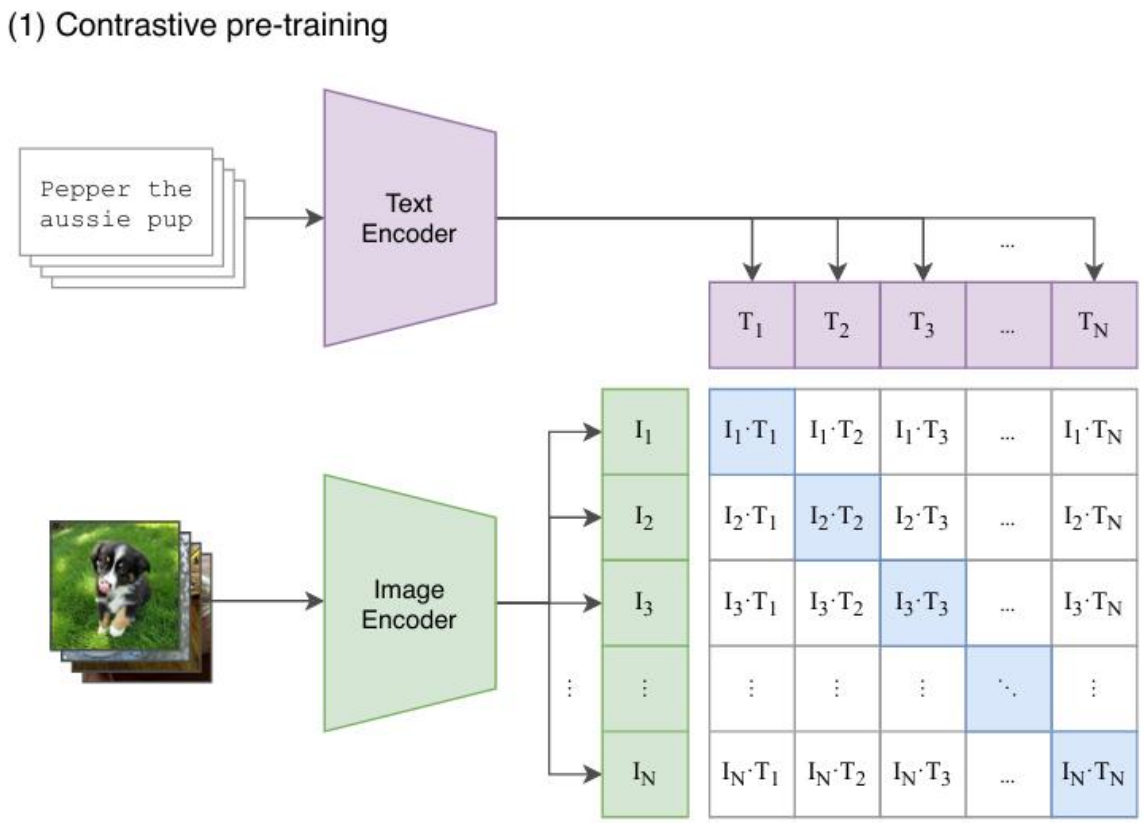
Prompt

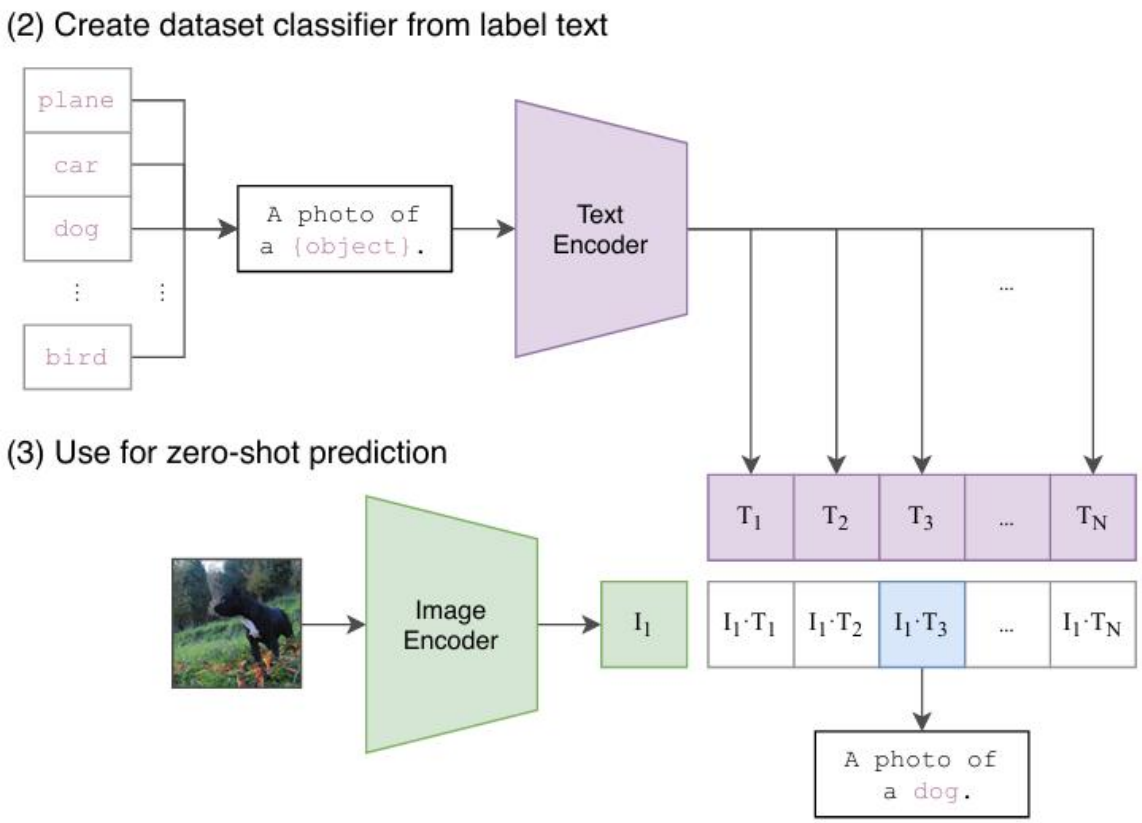
An elegantly appointed bedroom in the Art Deco style, featuring a grand king-size bed with geometric bedding, a luxurious velvet armchair, and a mirrored nightstand that reflects the room's opulence. Art Deco-inspired artwork adds a touch of glamour

Some metrics to compare images generated by different models

1.CLIP (Contrastive Language-Image Pretraining) training and inference process

CLIP is a model proposed by OpenAI that measures the similarity between text and images.





Calculate the CLIP Score

1. Input Text → Compute the text feature vector.
2. Output Image → Compute the image feature vector.
3. Compute the cosine similarity between the text and image vectors to get CLIP Score.
4. Score Range: The value ranges from 0 to 1, with higher scores indicating better alignment between the text and image.

Example 1

Image generated by Top2

Clip: 0.3083

Prompt

An elegantly appointed bedroom in the Art Deco style, featuring a grand king-size bed with geometric bedding, a luxurious velvet armchair, and a mirrored nightstand that reflects the room's opulence. Art Deco-inspired artwork adds a touch of glamour

Image generated by our group

Clip: 0.3168

Example 2

Prompt

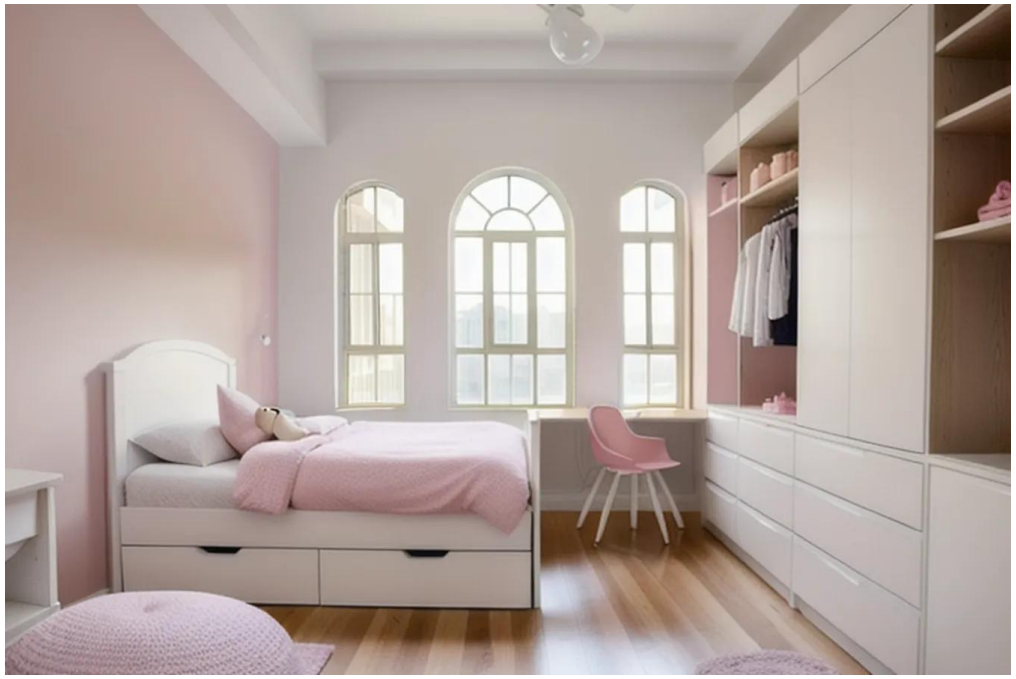
A simple, practical, and cozy children's bedroom for everyday use. The walls are painted in light tones, and the furniture features clean, minimalist lines. There is a pink children's bed, a large wardrobe with ample storage space, a child-sized desk, and a comfortable chair. The floor is finished in a light color to maintain a bright and airy atmosphere.

Image generated by Top2

Clip: 0.3448

Image generated by our group

Clip: 0.3487



Example 3

Prompt

A cozy dining room that captures the essence of rustic charm with a solid wooden farmhouse table at its core, surrounded by an eclectic mix of mismatched chairs. An antique sideboard serves as a statement piece, and the ambiance is warmly lit by a series of quaint Edison bulbs dangling from the ceiling

Image generated by Top2

Clip: 0.2869

Image generated by our group

Clip: 0.3025

St Petersburg University

Thank you for the attention!