MS-CoCo and MNIST Image Generation with Conditional VAE

Chengkun Yang, Yixuan Yang, Qinmeng Yu, Kechao Lu

Abstract

This report presents our work for the ECE 685D [Introduction to Deep Learning] final project, focusing on text-to-image generation using Conditional Variational Autoencoders (CVAEs) with CLIP embeddings. The project is carried out in two stages. First, a CVAE model is trained on the FashionMNIST dataset to generate images from short text labels. In the second stage, the model is extended to work with the COCO dataset, enabling the generation of images from longer, more descriptive text inputs. By utilizing CLIP embeddings as a condition, our approach captures the semantic relationships between text and images, facilitating coherent image generation. The results demonstrate the model's capability to handle varying text complexities and datasets, highlighting the potential of combining CVAEs with CLIP for text-to-image tasks.

1 Introduction

Text-to-image generation represents a fascinating intersection of natural language processing and computer vision.Goodfellow et al. (2016) It enables machines to translate textual descriptions into visual content. This capability has widespread applications, from creative design to assistive technologies,Ko et al. (2023) where generating meaningful visuals from text can enhance accessibility and automation. However, building effective text-to-image models requires addressing the semantic gap between language and images, a challenge that continues to drive innovation in deep learning.Gartner & Romanov (2024)

Conditional Variational Autoencoders (CVAEs) and Contrastive Language–Image Pre-training (CLIP) have emerged as valuable tools for tackling this challenge. CVAEs provide a probabilistic framework for generating images conditioned on specific inputs, making them well-suited for tasks that require control and variability.Ivanovic et al. (2020) CLIP, on the other hand, bridges language and vision by learning a shared embedding space, enabling the encoding of semantic relationships without explicit paired text-image datasets.Udandarao (2022) Together, these methods offer a foundation for exploring text-to-image generation with flexibility and efficiency.

In this project, we aimed to understand and apply these concepts by integrating CVAEs with CLIP embeddings to generate text-to-image images. We explored how CLIP embeddings encode semantic features and how CVAEs can leverage this information to generate coherent and contextually appropriate images. This project not only provided hands-on experience with these techniques but also deepened our understanding of their limitations and potential in real-world scenarios.

The following sections detail our implementation and findings, focusing on the interaction between CVAEs and CLIP embeddings and their combined role in bridging text and image synthesis.

2 Background Information

2.1 Autoencoder

Autoencoders are a type of neural network designed for unsupervised learning tasks, primarily used to learn a compressed representation of input data.Baldi (2012) The network consists of two main components:

2.1.1 ENCODER

The encoder compresses the input data into a latent representation of lower dimensionality. This process involves a series of transformations that extract essential features, while discarding less relevant details.Pintelas et al. (2021) Mathematically, the encoder can be represented as a function $g_\phi(x)$, parameterized by ϕ, which maps the input x to the latent space z.

2.1.2 DECODER

The decoder reconstructs the original input data from the latent representation. The goal is to minimize the reconstruction error, ensuring that the output is as close as possible to the input.Shen et al. (2018) The decoder is represented as a function $f_\theta(z)$, parameterized by θ, which maps the latent representation z back to the original data space.

The autoencoder is trained to optimize a reconstruction loss, such as mean squared error (MSE), to ensure the fidelity of the reconstructed output.

2.2 VARIATIONAL AUTOENCODERS (VAES)

Variational Autoencoders (VAEs) extend the concept of autoencoders by introducing a probabilistic framework. Instead of encoding the input as a single point in the latent space, VAEs encode it as a probability distribution. This allows the model to generate new data by sampling from the learned distribution, making VAEs particularly suited for generative tasks.Bond-Taylor et al. (2021)

2.2.1 KEY FEATURES OF VAES

- Probabilistic Latent Space: The encoder learns to map the input x to a distribution $q_\phi(z \mid x)$, typically parameterized as a Gaussian with mean $\mu(x)$ and variance $\sigma^2(x)$.
- Reparameterization Trick: To enable backpropagation through the stochastic sampling process, the model applies the reparameterization trick, representing z as $z = \mu + \sigma \cdot \epsilon$, where ϵ is a random variable sampled from a standard normal distribution.

2.3 LOSS FUNCTION: EVIDENCE LOWER BOUND (ELBO)

The VAE loss function, known as the Evidence Lower Bound (ELBO), consists of two components:Ding (2022)

- Reconstruction Loss: Measures how well the decoder reconstructs the input from the latent representation. This term ensures that the model retains the critical features of the input data.
- Regularization Term: Uses Kullback-Leibler (KL) divergence to ensure that the learned latent distribution $q_\phi(z \mid x)$ is close to a prior distribution $p(z)$, usually a standard normal distribution. This regularization encourages the latent space to be well-structured and continuous.

The overall Evidence Lower Bound (ELBO) is given by:

$$\text{ELBO} = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x \mid z)\right] - \text{KL}\left(q_\phi(z \mid x) \parallel p(z)\right),$$

where the first term is the reconstruction loss, and the second term is the KL divergence. The model is trained to maximize ELBO (or equivalently, minimize its negative).

This framework allows VAEs to generate new data points by sampling the latent space, making them a powerful tool for applications such as image generation and anomaly detection.

2.4 CONDITIONAL VAE

Variational Autoencoders (VAEs) inherently lack fine-grained control over the generation process. In the standard VAE, the latent representation z is sampled from a prior distribution without incorporating additional contextual information, limiting the model's generative capabilities.Bond-Taylor et al. (2021)

Conditional Variational Autoencoders (CVAEs), introduced by Sohn et al, address this limitation by introducing an observable condition c that guides the generative process.Sohn et al. (2015) While a standard VAE learns $p(x|z)$, a CVAE extends this to learn $p(x|z,c)$, conditioning both encoder and decoder distributions on auxiliary information.

This modification enables more directed and semantically controlled generative processes, allowing the model to generate samples explicitly guided by the conditional variable.Bond-Taylor et al. (2021) The key innovation lies in augmenting the probabilistic framework to incorporate additional contextual information, providing greater flexibility in generative modeling.

3 Our Model

3.1 Long-Text Model

This project implements a **Conditional Variational Autoencoder (CVAE)** to generate images conditioned on textual descriptions. The model consists of an encoder, decoder, and reparameterization module. The encoder processes input images and combines them with textual features extracted from captions using the pre-trained CLIP model. These text features are integrated with image data by concatenation along the channel dimension. The encoder outputs the latent distribution parameters, mean (`mu`) and log variance (`logvar`), which are sampled via the reparameterization trick to create latent variables (`z`). The decoder takes these latent variables and conditions as input, reconstructing the images through transposed convolutional layers. The output images are normalized using a sigmoid activation to ensure pixel values are between 0 and 1.

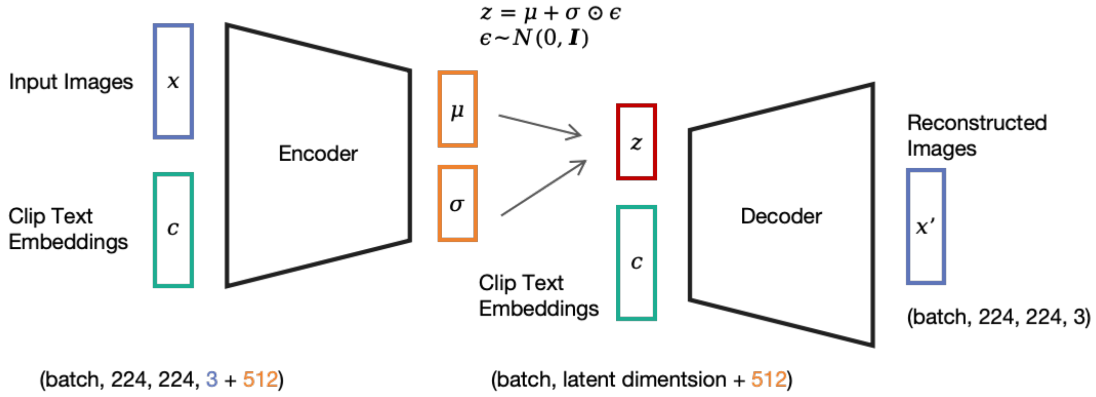

$z = \mu + \sigma \odot \epsilon$
$\epsilon \sim N(0, \boldsymbol{I})$
Input Images
x
Clip Text Embeddings
c
Encoder
μ
σ
z
Clip Text Embeddings
c
Decoder
Reconstructed Images
x'
(batch, 224, 224, 3)
(batch, 224, 224, 3 + 512)
(batch, latent dimentsion + 512)

Figure 1: CVAE for Text-to-Image Generation with Clip text embeddings

Text features are extracted via CLIP's `encode_text` method, mapping captions into a high-dimensional semantic space. Training aims to minimize a combination of Binary Cross-Entropy (BCE) reconstruction loss and KL Divergence loss, which regularizes the latent space to approximate a standard normal distribution. Figure 1 illustrates our CVAE architecture for text-to-image generation using CLIP text embeddings. The model takes input images (x) and corresponding **CLIP text embeddings** (c), combining them as input to an encoder that outputs a latent representation characterized by a mean (μ) and standard deviation (σ). A latent variable z is sampled from this distribution using the reparameterization trick:

$$z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

The decoder then reconstructs the original image (x') using both the latent variable z and the text embeddings c. This process enables the generation of images conditioned on both visual input and semantic information provided by the text embeddings. The output is reconstructed images with dimensions matching the input images.

The following tables show our model structure.

Layer	Input Dimensions	Output Dimensions	Operation
Input	$3 \times 224 \times 224$ + condition	$3 +$ condition_dim $\times 224 \times 224$	Concatenation
Conv1	$(3 +$ condition_dim$) \times 224 \times 224$	$32 \times 112 \times 112$	Conv2D + LeakyReLU
Conv2	$32 \times 112 \times 112$	$64 \times 56 \times 56$	Conv2D + LeakyReLU
Conv3	$64 \times 56 \times 56$	$128 \times 28 \times 28$	Conv2D + LeakyReLU
Conv4	$128 \times 28 \times 28$	$256 \times 14 \times 14$	Conv2D + LeakyReLU
Conv5	$256 \times 14 \times 14$	$512 \times 7 \times 7$	Conv2D + LeakyReLU
Flatten	$512 \times 7 \times 7$	$512 \times 7 \times 7$	Flatten
Fully Connected (mu)	$512 \times 7 \times 7$	latent_dim	Linear
Fully Connected (logvar)	$512 \times 7 \times 7$	latent_dim	Linear

Table 1: Encoder structure of the CVAE model.

Encoder Structure

Note: the Conv2D structure in encoder is (kernel=3, stride=2, padding=1)

Decoder Structure

Layer	Input Dimensions	Output Dimensions	Operation
Input	latent_dim + condition_dim	1024	Concatenation
Fully Connected 1	latent_dim + condition_dim	1024	Linear + LeakyReLU
Fully Connected 2	1024	$512 \times 7 \times 7$	Linear + LeakyReLU
Reshape	$512 \times 7 \times 7$	$512 \times 7 \times 7$	Reshape
Deconv1	$512 \times 7 \times 7$	$256 \times 14 \times 14$	ConvTranspose2D + LeakyReLU
Deconv2	$256 \times 14 \times 14$	$128 \times 28 \times 28$	ConvTranspose2D + LeakyReLU
Deconv3	$128 \times 28 \times 28$	$64 \times 56 \times 56$	ConvTranspose2D + LeakyReLU
Deconv4	$64 \times 56 \times 56$	$32 \times 112 \times 112$	ConvTranspose2D + LeakyReLU
Deconv5	$32 \times 112 \times 112$	$3 \times 224 \times 224$	ConvTranspose2D + Sigmoid

Table 2: Decoder structure of the CVAE model.

Note: the Conv2D structure in the decoder is (kernel = 3, stride = 2, padding = 1)

3.2 Short-Text Model

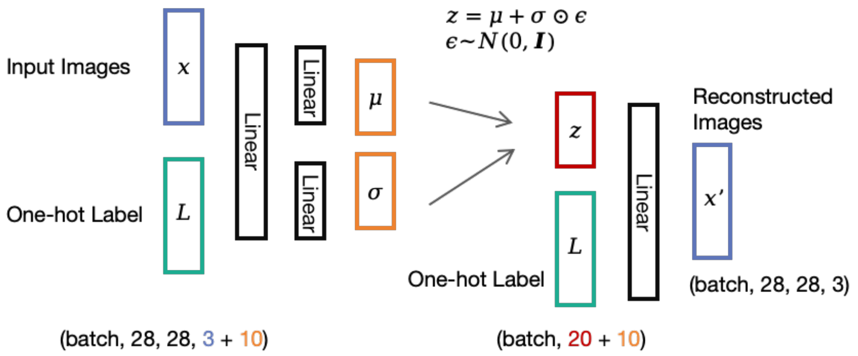


Figure 2: CVAE for Text-to-Image Generation with One-hot Label

Figure 2 depicts our CVAE architecture for generating MNIST images conditioned on short text labels encoded as one-hot vectors. The CVAE comprises an Encoder, a Decoder, and a latent space. The Encoder concatenates the input image and one-hot label, processes them through a fully connected layer with ReLU activation, and outputs the latent mean (μ) and log-variance ($\log(\sigma^2)$) of a Gaussian distribution. Using the reparameterization trick, the sampling of latent vector (z) is the similar to the sampling mentioned in the long-text model. The Decoder concatenates z with the one-hot label and reconstructs the image by passing the combined input through fully connected layers with ReLU activation, followed by a sigmoid function to normalize pixel intensities. This process ensures the latent space and reconstructed images are conditioned on the label, enabling the generation of digit-specific MNIST images by sampling latent vectors and decoding them with the desired labels. The architecture effectively integrates both image and label information for conditional generation.

4 Training Model

4.1 Dataset

The **MSCOCO 2017 Dataset** is a subset of the Common Objects in Context (COCO) dataset designed for image captioning. It contains 123,287 images (118,287 for training and 5,000 for validation), each paired with five human-annotated captions. Featuring diverse everyday scenes, the dataset is ideal for tasks such as image captioning and understanding vision language. The images are stored as `.jpg` files, with captions provided in JSON format linking the image IDs to the descriptive text. COCO also includes bounding boxes and object categories, supporting multitask learning across 80 classes. We built the training data set by cropping and scaling the aligned images to 224 x 224 pixels. **This data set trains the CVAE to handle more complex and long input text**. The **Fashion-MNIST Dataset** is a large-scale data set designed as a drop-in replacement for the original MNIST dataset, which allows for benchmarking machine learning algorithms on more complex visual patterns. It consists of 70,000 grayscale images of size 28 $\times$ 28 pixels, split into 60,000 training images and 10,000 test images.**This data set trains the CVAE to generate multiple images from simple text inputs**. Due to computational constraints, our model is trained on a 10% subset of the original COCO training dataset.

4.2 Long-Text Training

The long-text CVAE is trained by integrating the features of the image and the text, where the input image and the condition of the text (encoded using CLIP) are combined and passed through the encoder. In one approach, the text condition is concatenated with the image at the input stage; in another, it is added to the tensor after passing through the first few convolutional layers. The encoder computes the latent distribution parameters, mean (μ) and log variance $(\log \sigma^2)$, which are then sampled using the reparameterization trick to obtain latent variables (z). The decoder reconstructs the image using z and the condition, outputting normalized pixel values through a sigmoid activation. The model is trained for **20 epochs** on the **MS-COCO 2017** dataset, using a **batch size of 64**, a **latent dimension of 128**, and a **learning rate of 0.0001** with the **Adam optimizer**. Training minimizes a combined loss of **Binary Cross-Entropy (BCE)** for reconstruction accuracy and **KL Divergence** to regularize the latent space. Despite experimenting with deeper encoder networks and attention layers, these changes showed no significant improvement over the base architecture.

4.3 Short-Text Training

The CVAE model, initialized with an input dimension of 28 $\times$ 28, a label dimension of 10, and a **latent space dimension of 20**, is trained using the Adam optimizer with a **learning rate of** 1×10^{-3}. During training, images are flattened, normalized, and combined with the one-hot labels before being passed to the Encoder, which outputs the latent mean (μ) and log-variance $(\log(\sigma^2))$. Using the reparameterization trick, a latent vector (z) is sampled and passed, along with the labels, to the Decoder, which reconstructs the input images. The loss function combines **BCE** (reconstruction loss) and **KL divergence** (latent space regularization), which is minimized through backpropagation. Over **50 epochs**, the model iteratively updates its parameters to conditionally generate high-quality images based on the labels, with the average training loss monitored after each epoch to ensure convergence.

5 Results

5.1 Long-Text Generated Images

We trained the model saved the parameters after 50, 100, 150, 200, 250, and 300 epochs. By comparing the results, we observed that the quality of images generated from long-text conditions peaked at 250 epochs and gradually declined when training reached 300 epochs.

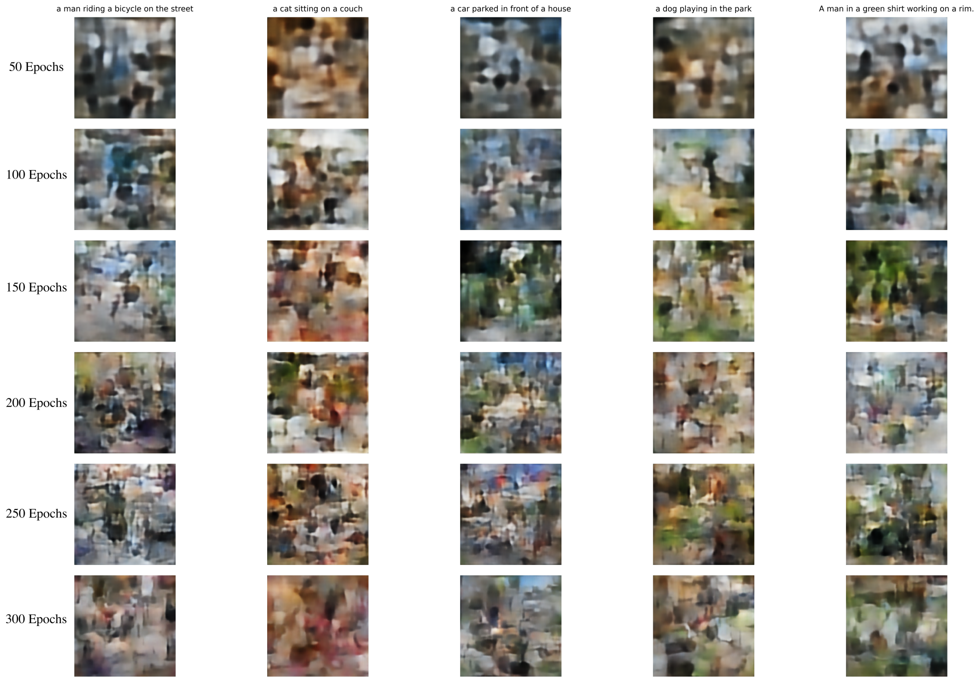

a man riding a bicycle on the street
a cat sitting on a couch
a car parked in front of a house
a dog playing in the park
A man in a green shirt working on a rim.
50 Epochs
100 Epochs
150 Epochs
200 Epochs
250 Epochs
300 Epochs

Figure 3: Quality of the generated images increased and then decreased after 250 epochs

5.1.1 Quantitative Evaluation of Text-to-Image Results

For quantitatively analyzing our generation model performance, we use two metrics:

1. **CLIP Score:** Measure the semantic similarity between the generated images and the text prompts, capturing how well the model can translate text into corresponding visual outputs.Zhengwentai (2023)
2. **Fréchet Inception Distance (FID):** Evaluate the realism of the generated images by comparing their statistics to those of real images.Heusel et al. (2017)

	Epoch=50	**Epoch=100**	**Epoch=150**	**Epoch=200**	**Epoch=250**	**Epoch=300**
Clip Score	0.1951	**0.2023**	0.2001	0.1992	0.1999	0.2002
FID	326.70	321.86	315.62	314.98	313.64	**295.40**

Table 3: Performance metrics (CLIP Score and FID) over different epochs.

Table 3 presents the CLIP Score and FID values over different training epochs, ranging from 50 to 300. The CLIP Score nearly does not change indicating generated images are not well aligned with text input with epochs increasing. Concurrently, the FID metric decreases from 326.70 to 295.40, suggesting the generated images become more realistic over the course of training.

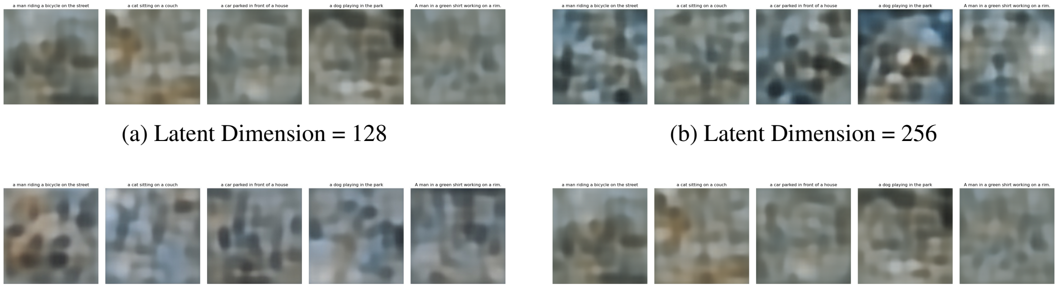
	Latent Dim=128	Latent Dim=256	Latent Dim=512	Latent Dim=1024
Clip Score	**0.2023**	0.1998	0.1937	0.1886
FID	**286.06**	315.01	306.73	305.43

Table 4: Performance metrics (CLIP Score and FID) for different latent dimensions.

Table 4 further explores the impact of latent dimension size on the model's performance. The best CLIP Score of 0.2023 is achieved with a latent dimension of 128, while the lowest FID of 305.43 is obtained with a latent dimension of 1024.

5.1.2 Influence of Latent Space Dimension on Image Generation

In order to further understand the effect of latent space, we investigated the influence of the latent space dimension on Image Generation.


a man riding a bicycle on the street
a cat sitting on a couch
a car parked in front of a house
a dog playing in the park
A man in a green shirt working on a rim.
a man riding a bicycle on the street
a cat sitting on a couch
a car parked in front of a house
a dog playing in the park
A man in a green shirt working on a rim.
a man riding a bicycle on the street
a cat sitting on a couch
a car parked in front of a house
a dog playing in the park
A man in a green shirt working on a rim.
a man riding a bicycle on the street
a cat sitting on a couch
a car parked in front of a house
a dog playing in the park
A man in a green shirt working on a rim.

(a) Latent Dimension = 128

(b) Latent Dimension = 256

(c) Latent Dimension = 512

(d) Latent Dimension = 1024

Figure 4: Image Generation Comparison with Different Latent Dimensions

The results indicate that as the latent dimension increases, image resolution improves, but the quality of segmented features deteriorates. This suggests that there is still room for improvement in the model structure. Simply increasing the size of the latent dimension does not necessarily lead to better model performance.

5.2 Short-Text Generated Images

Figure 4 demonstrates the output of a Conditional Variational Autoencoder (CVAE) trained on the Fashion MNIST dataset. The CVAE can generate images based on the provided class labels, corresponding to different clothing items. Single label is able to generate diverse images that align with the given label descriptions.

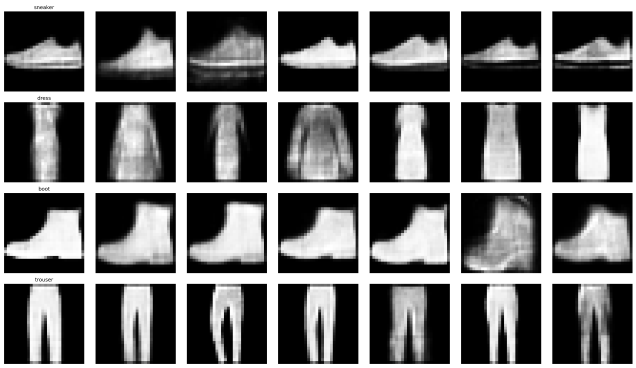


Figure 5: Short-Text Generated MNIST Images

The results show the CVAE generates visually plausible images aligned with the given labels, except for the ”Invalid” class, which the model has not been trained on, resulting in a distorted output.

5.3 Figure Reconstruction

Similarly, we used models trained for different epochs to reconstruct images. The performance improved before 250 epochs but started to decline after 250 epochs.

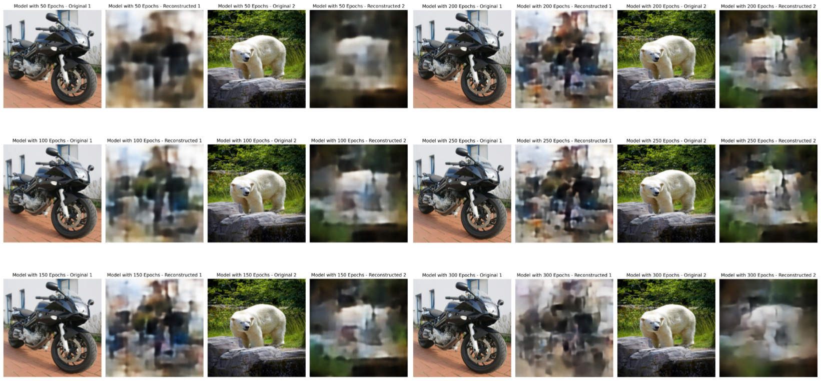

Model with 50 Epochs - Original 1
Model with 50 Epochs - Reconstructed 1
Model with 50 Epochs - Original 2
Model with 50 Epochs - Reconstructed 2
Model with 200 Epochs - Original 1
Model with 200 Epochs - Reconstructed 1
Model with 200 Epochs - Original 2
Model with 200 Epochs - Reconstructed 2
Model with 100 Epochs - Original 1
Model with 100 Epochs - Reconstructed 1
Model with 100 Epochs - Original 2
Model with 100 Epochs - Reconstructed 2
Model with 250 Epochs - Original 1
Model with 250 Epochs - Reconstructed 1
Model with 250 Epochs - Original 2
Model with 250 Epochs - Reconstructed 2
Model with 150 Epochs - Original 1
Model with 150 Epochs - Reconstructed 1
Model with 150 Epochs - Original 2
Model with 150 Epochs - Reconstructed 2
Model with 300 Epochs - Original 1
Model with 300 Epochs - Reconstructed 1
Model with 300 Epochs - Original 2
Model with 300 Epochs - Reconstructed 2

Figure 6: Quality of the reconstructed images increased and then decreased after 250 epochs

6 Summary

Text-to-image generation is a significant area in deep learning that combines natural language understanding with image synthesis. This project explored the use of Conditional Variational Autoencoders (CVAEs) and Contrastive Language–Image Pre-training (CLIP) embeddings for this task. Autoencoders compress and reconstruct data using an encoder-decoder framework, while VAEs extend this concept by introducing probabilistic latent spaces, enabling generative capabilities. By leveraging CLIP embeddings, which capture semantic relationships between text and images, our project integrated these techniques to perform text-to-image generation. This approach provided an opportunity to understand how CVAEs and CLIP work together to translate textual descriptions into visual content. Although the project was conducted as a course exercise, it highlighted the foundational principles and challenges of building models that connect language and vision, contributing to our understanding of these technologies. This project was conducted with assistance from ChatGPT.

Author Contributions

Short-Text MNIST Generation: Chengkun Yang, Qinmeng Yu

Long-Text ms-CoCo Generation: Chengkun Yang, Yixuan Yang, Qinmeng Yu, Kechao Lu

Report: Chengkun Yang, Yixuan Yang, Qinmeng Yu, Kechao Lu

REFERENCES

P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pp. 37–49. JMLR Workshop and Conference Proceedings, June 2012.

S. Bond-Taylor, A. Leach, Y. Long, and C.G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7327–7347, 2021. doi: 10.1109/TPAMI.2021.3087591.

M. Ding. The road from mle to em to vae: A brief tutorial. *AI Open*, 3:29–34, 2022. doi: 10.1016/j.aiopen.2022.01.003.

J. Gartner and M. Romanov. The advantages of ai text to image generation. *International Journal of Art, Design, and Metaverse*, 2(1):1–8, 2024.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

B. Ivanovic, K. Leung, E. Schmerling, and M. Pavone. Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *IEEE Robotics and Automation Letters*, 6(2):295–302, 2020.

H.K. Ko, G. Park, H. Jeon, J. Jo, J. Kim, and J. Seo. Large-scale text-to-image generation models for visual artists' creative works. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, pp. 919–933, March 2023.

E. Pintelas, I.E. Livieris, and P.E. Pintelas. A convolutional autoencoder topology for classification in high-dimensional noisy image datasets. *Sensors*, 21(22):7731, 2021. doi: 10.3390/s21227731.

X. Shen, H. Su, S. Niu, and V. Demberg. Improving variational encoder-decoders in dialogue generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, April 2018.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

V. Udandarao. Understanding and fixing the modality gap in vision-language models. Master's thesis, University of Cambridge, 2022.

SUN Zhengwentai. clip-score: CLIP Score for PyTorch. `https://github.com/taited/clip-score`, March 2023. Version 0.1.1.