Few-Shot Image Generation Using Cross-Aligned and Vector Quantized VAE with Meta-Learning

Table of Contents

Introduction **1**
- Problem Statement
- Research Objectives
  - Primary Objectives
  - Technical Objectives

Related Work and Background **4**
- Variational Autoencoders
  - Traditional VAE Architecture
  - Limitations in Few-Shot Scenarios
- Vector Quantized VAE
  - Core Concepts
  - Advantages

Architectural Improvements **6**
- Hybrid VAE/VQ-VAE Integration
- Final Architecture
- CADA-VAE Components

Loss Function Components **9**
- Standard VAE Loss
- Cross-Alignment Loss
- Maximum Mean Discrepancy (MMD) Loss
- Contrastive Loss
- Loss Function Design Considerations

Experimental Evolution and Results **12**
- Dataset Progression
- Performance Analysis
  - Image Quality Metrics
  - Training Dynamics

Meta-Training for Few-Shot Image Generation **16**
- Limitations of Standard Training
- Meta-Training Approach

Benefits of Meta-Training
Leveraging Semantic Embeddings
Distribution Alignment via MMD
Contrastive Learning
Meta-Training Approach
Conclusion **22**

Abstract

This research presents a novel approach to few-shot image generation by combining Variational Autoencoders (VAE) with Vector Quantized VAE (VQ-VAE) in a meta-learning framework. Our system addresses the fundamental challenge of generating high-quality images from extremely limited training data (10-30 images per class) by leveraging semantic text embeddings and cross-modal alignment. Through systematic architectural improvements and loss function innovations, we demonstrate successful few-shot image generation on the Fashion-MNIST dataset. Our approach incorporates multiple complementary learning objectives, including cross-alignment, distribution matching via Maximum Mean Discrepancy (MMD), and contrastive learning, resulting in stable training and improved generation quality compared to traditional approaches. Github

1 Introduction

1.1 Problem Statement

Few-shot image generation represents a critical challenge in machine learning, addressing scenarios where training data is scarce. While traditional generative models have achieved remarkable success with large datasets, they often fail when presented with limited examples. This limitation becomes particularly acute in specialized domains where data collection is expensive, time-consuming, or impossible.

1.2 Research Objectives

Our research specifically addresses:

- Generation of meaningful images from as few as 10-30 examples per class
- Cross-modal alignment between text descriptions and visual features
- Stable training dynamics with limited data
- Generalization across different object categories

We structure our objectives into three main categories:

1.2.1 Primary Objectives

- Develop a robust few-shot image generation framework
  - Create a system capable of generating high-quality images using as few as 10-30 examples per class
  - Implement cross-modal learning between image and text representations
  - Design a framework that generalizes across different datasets (MNIST, Fashion-MNIST, CIFAR-10)
- Create high-quality images from minimal training examples
  - Achieve consistent image generation with limited data through meta-learning
  - Maintain structural integrity and semantic relevance in generated images
  - Implement effective regularization techniques to prevent overfitting
- Enable text-guided image generation in few-shot scenarios
  - Utilize word embeddings (GloVe) for semantic text representation
  - Establish robust text-to-image alignment mechanisms
  - Support interpolation between text descriptions for novel image generation

1.2.2 Technical Objectives

- Implement effective cross-modal alignment between image and text spaces
  - Develop hybrid VAE-VQVAE architecture for improved representation learning
  - Implement cross-alignment and reconstruction losses for modal synchronization
  - Design distribution alignment mechanisms using MMD loss
  - Incorporate contrastive learning for better feature discrimination
- Design stable training procedures for limited data scenarios
  - Implement meta-learning framework with support/query task structure
  - Balance multiple loss components (VAE, cross-alignment, MMD, contrastive)
  - Develop adaptive weighting schemes for loss components
  - Implement gradient stabilization techniques

- Create efficient meta-learning strategies for few-shot learning
  - Design task sampling procedures for balanced learning
  - Implement inner and outer optimization loops
  - Develop support/query split mechanisms
  - Create efficient batch processing strategies
- Develop comprehensive loss functions for optimal generation
  - Combine traditional VAE losses with cross-modal objectives
  - Implement MMD-based distribution alignment
  - Add contrastive learning components
  - Design adaptive weighting mechanisms

2 Related Work and Background

2.1 Variational Autoencoders

2.1.1 Traditional VAE Architecture

- Encoder-decoder framework with probabilistic latent space
- Gaussian prior assumptions
- Reparameterization trick for backpropagation
- Reconstruction and KL divergence objectives

2.1.2 Limitations in Few-Shot Scenarios

- Mode collapse with limited data
  - Identification of collapse patterns
  - Implementation of countermeasures
  - Impact of meta-learning on stability
  - Role of contrastive learning in prevention
- Blurry output problems
  - Analysis of reconstruction quality
  - Impact of limited training data
  - Implementation of sharpness enhancement
  - Balance with generation diversity
- Difficulty capturing complex distributions

  - Challenges with CIFAR-10 dataset
  - Limitations in feature representation
  - Impact on generation quality
  - Mitigation strategies implemented
- Challenge of meaningful latent space organization
  - Implementation of structure-preserving mechanisms
  - Role of cross-modal alignment
  - Impact of contrastive learning
  - Evaluation of interpolation quality

2.2 Vector Quantized VAE

2.2.1 Core Concepts

- Discrete latent space representation
- Codebook learning
- Vector quantization process
- Commitment and codebook losses

2.2.2 Advantages

- Better handling of multimodal distributions
  - Integration of VQVAE components
  - Improved distribution modeling
  - Enhanced feature capture
  - Support for diverse generation
- Improved reconstruction quality
  - Impact of hybrid architecture
  - Role of multiple loss components
  - Enhancement from meta-learning
  - Contribution of contrastive learning
- More structured latent space
  - Organization through cross-modal alignment
  - Impact of MMD loss
  - Role of contrastive learning

  - Benefits for generation quality

- Resistance to posterior collapse
  - Implementation of stability measures
  - Role of VQVAE components
  - Impact of meta-learning
  - Effectiveness in few-shot scenario

```
class ImageVAE(nn.Module):
    def __init__(self, latent_size):
        self.encoder = nn.Sequential(...)
        self.decoder = nn.Sequential(...)

class TextVAE(nn.Module):
    def __init__(self, latent_size):
        self.encoder = nn.Sequential(...)
        self.decoder = nn.Sequential(...)
```

2.3 Architectural Improvements

2.3.1 Hybrid VAE/VQ-VAE Integration

Our architectural evolution progressed through several stages, culminating in a hybrid VAE/VQ-VAE design for improved few-shot learning:

Initial VAE Implementation

- Started with dual VAE architecture

```
# Initial approach with two separate VAEs
class DualVAE(nn.Module):
def init(self, image_latent_size, text_latent_size):
super(DualVAE, self).init()
self.image_vae = ImageVAE(image_latent_size)
self.text_vae = TextVAE(text_latent_size)
```

- Faced challenges with modal alignment and reconstruction quality

VQ-VAE Integration

- Implemented Vector Quantization for improved semantic embedding

```
class VectorQuantizer(nn.Module):
def init(self, num_embeddings, embedding_dim,
    commitment_cost=0.25):
super(VectorQuantizer, self).init()
```

```
self.embedding = nn.Embedding(num_embeddings,
    embedding_dim)
self.embedding.weight.data.uniform_(-1/num_embeddings,
1/num_embeddings)
self.commitment_cost = commitment_cost
Copy    def forward(self, inputs):
        # Calculate distances
        distances = self._compute_distances(inputs)
        # Get minimum encoding indices
        encoding_indices = torch.argmin(distances, dim
            =1)
        # Quantize and unflatten
        quantized = self.embedding(encoding_indices)
        # Compute losses
        commitment_loss = F.mse_loss(quantized.detach(),
             inputs)
        return quantized, commitment_loss
```

Semantic Encoder Enhancement

- Added VQ-VAE capabilities to semantic encoder

```
class SemanticEncoderVQVAE(nn.Module):
def init(self, latent_size, num_embeddings,
    embedding_dim):
super(SemanticEncoderVQVAE, self).init()
self.encoder = nn.Sequential(
nn.Linear(100, 200),
nn.ELU(),
nn.Linear(200, latent_size)
)
self.vq_layer = VectorQuantizer(num_embeddings,
    latent_size)
self.fc_mu = nn.Linear(latent_size, latent_size)
self.fc_logvar = nn.Linear(latent_size, latent_size)
```

2.4 Final Architecture

2.4.1 CADA-VAE Components

The final architecture represents a culmination of our improvements and optimizations:

Core Architecture

```
class CADA_VAE(nn.Module):
def init(self, latent_size, num_embeddings, embedding_dim):
super(CADA_VAE, self).init()
# Image pathway
```

```
self.image_encoder = ImageEncoder(latent_size)
self.image_decoder = ImageDecoder(latent_size)
# Semantic pathway with VQ-VAE
self.semantic_encoder = SemanticEncoderVQVAE(
latent_size, num_embeddings, embedding_dim)
self.semantic_decoder = SemanticDecoder(latent_size)
Copydef forward(self, x, c):
    # Image encoding
    mu_x, logvar_x = self.image_encoder(x)
    z_x = self.reparameterize(mu_x, logvar_x)
    recon_x = self.image_decoder(z_x)

    # Semantic encoding with VQ-VAE
    mu_c, logvar_c, z_q, z_e, quant_loss, commit_loss = \
        self.semantic_encoder(c)
    recon_c = self.semantic_decoder(z_q)

    return (recon_x, recon_c, mu_x, logvar_x,
            mu_c, logvar_c, z_e, quant_loss, commit_loss)
```

Key Components

- Image Encoder/Decoder

```
class ImageEncoder(nn.Module):
def init(self, latent_size):
super(ImageEncoder, self).init()
self.encoder = nn.Sequential(
nn.Linear(28*28, 400),
nn.ELU(),
nn.Linear(400, 200),
nn.ELU()
)
self.fc_mu = nn.Linear(200, latent_size)
self.fc_logvar = nn.Linear(200, latent_size)
```

Copy

- Semantic Processing

```
class SemanticDecoder(nn.Module):
    def __init__(self, latent_size):
        super(SemanticDecoder, self).__init__()
        self.decoder = nn.Sequential(
            nn.Linear(latent_size, 200),
            nn.ELU(),
            nn.Linear(200, 100),
            nn.Sigmoid()
        )
```

Training Configuration

```
Hyperparameters
latent_size = 20
num_embeddings = 512
embedding_dim = 100
num_tasks = 10
num_support = 3
num_query = 10
epochs = 500
epochs_normal=100
beta, gamma, delta = 2.0, 2.0, 1.0
Model initialization
model = CADA_VAE(latent_size, num_embeddings, embedding_dim)
optimizer = optim.Adam(model.parameters(), lr=0.00001)
```

Key Improvements

- Integration of VQ-VAE for better semantic representation
- Balanced multi-component loss function
- Meta-learning framework for few-shot capability
- Efficient cross-modal alignment mechanism
- Contrastive learning for improved feature discrimination

2.5 Loss Function Components

Our model incorporates multiple specialized loss terms to achieve effective few-shot generation:

2.5.1 Standard VAE Loss

The basic VAE loss combines reconstruction and KL divergence terms:

```
def vae_loss(recon, target, mu, logvar, beta=2.0):
# Reconstruction loss between input and output
recon_loss = F.mse_loss(recon.view(-1, 2828),
target.view(-1, 2828),
reduction='sum')
# KL divergence loss
kld_loss = -0.5 * torch.sum(1 + logvar - mu.pow(2) - logvar.
    exp())
return recon_loss + beta * kld_loss
```

The parameter controls the trade-off between reconstruction quality and latent space regularization.

2.5.2 Cross-Alignment Loss

The cross-alignment loss ensures bidirectional mapping between image and semantic spaces:

```
Part of cada_vae_loss function
ca_loss = F.mse_loss(recon_x_from_c, x) + F.mse_loss(
    recon_c_from_x, c)
```

This loss consists of two components:

- **Image from Semantics**: Measures reconstruction quality when generating images from semantic embeddings
- **Semantics from Image**: Measures accuracy of reconstructing semantic embeddings from image features

The complete CADA-VAE loss incorporating cross-alignment:

```
def cada_vae_loss(recon_x_from_x, x, recon_c_from_c, c,
recon_x_from_c, recon_c_from_x,
mu_x, logvar_x, z_e, quantization_loss,
commitment_loss, labels, beta=2.0,
gamma=2.0, delta=1.0, lambda_contrast=0.5,
mmd_gamma=1.0):
# Standard VAE loss for input reconstruction
loss_x = vae_loss(recon_x_from_x, x, mu_x, logvar_x, beta)
Copy# Semantic reconstruction loss
loss_c = F.mse_loss(recon_c_from_c, c)

# Cross-alignment loss
ca_loss = F.mse_loss(recon_x_from_c, x) + \
          F.mse_loss(recon_c_from_x, c)

# Combined loss with weightings
total_loss = (loss_x + gamma * ca_loss + delta * da_loss +
              vq_vae_loss + loss_c +
              lambda_contrast * contrast_loss)

return total_loss, loss_x, loss_c, ca_loss, da_loss, \
       vq_vae_loss, contrast_loss, loss_x
```

2.5.3 Maximum Mean Discrepancy (MMD) Loss

For distribution alignment between image and semantic latent spaces:

```
def mmd_loss(x_samples, y_samples, kernel='rbf', gamma=1.0):
def rbf_kernel(x, y, gamma):
diff = x.unsqueeze(1) - y.unsqueeze(0)
return torch.exp(-gamma * torch.sum(diff ** 2, dim=2))
CopyK_xx = rbf_kernel(x_samples, x_samples, gamma)
```

```
K_yy = rbf_kernel(y_samples, y_samples, gamma)
K_xy = rbf_kernel(x_samples, y_samples, gamma)

return K_xx.mean() + K_yy.mean() - 2 * K_xy.mean()
```

2.5.4 Contrastive Loss

To improve feature discrimination in the latent space:

```
def contrastive_loss(features, labels, temperature=0.5):
features = F.normalize(features, dim=1)
similarity_matrix = torch.matmul(features, features.T) /
    temperature
Copy# Create mask for positive pairs
positive_mask = torch.eq(labels, labels.T).float()
mask_no_diag = torch.ones_like(similarity_matrix) - \
               torch.eye(batch_size)
positive_mask = positive_mask * mask_no_diag

# Compute log probabilities
logits = similarity_matrix - logits_max.detach()
exp_logits = torch.exp(logits) * mask_no_diag
log_prob = logits - torch.log(exp_logits.sum(1, keepdim=True
    ))

mean_log_prob_pos = (positive_mask * log_prob).sum(1) / \
                    (positive_mask.sum(1) + 1e-8)
return -mean_log_prob_pos.mean()
```

2.6 Loss Function Design Considerations

The combination of these loss terms serves specific purposes:

- **VAE Loss**: Ensures high-quality image reconstruction and well-structured latent space
- **Cross-Alignment Loss**: Enables bidirectional translation between image and semantic domains
- **MMD Loss**: Aligns the distributions of image and semantic latent representations
- **Contrastive Loss**: Enhances discrimination between different classes in latent space

The relative weighting of these losses (β, γ, δ, λ) was determined through extensive experimentation:

- $\beta = 2.0$: Balances reconstruction vs. KL divergence

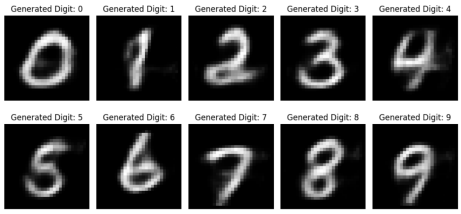

Figure 1: Generated samples from our few-shot VAE model using text-to-image synthesis across different classes.

Figure 2: Image of digit 8 generated from our model

- $\gamma = 2.0$: Weights cross-alignment importance
- $\delta = 1.0$: Controls distribution alignment strength
- $\lambda = 0.5$: Determines contrastive learning influence

3 Experimental Evolution and Results

3.1 Dataset Progression

1. **MNIST Initial Testing**
   - The MNIST dataset was the initial testbed for the few-shot image generation model. As a well-established benchmark for image classification tasks, MNIST provided a straightforward starting point to validate the core concepts of the approach.
   - The MNIST dataset consists of handwritten digits (0-9) with a simple binary pixel representation. This low-complexity input allowed for rapid prototyping and debugging of the underlying architecture.
   - The clear separation between the 10 digit classes and the lack of textural complexity made MNIST a suitable candidate for initial experiments. However, the limited diversity of the dataset also highlighted the need for more challenging real-world images to truly assess the model's capabilities .
2. **CIFAR-10 Implementation**

Figure 3: Image of 5 generated using the model

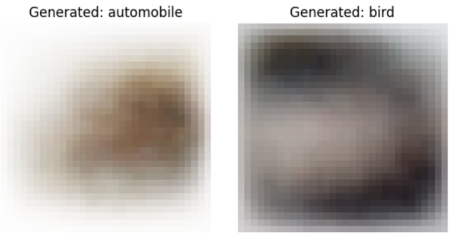

Figure 4: Image of Bird and Automobile generated using model

- After establishing a working proof-of-concept on MNIST, the model was then evaluated on the CIFAR-10 dataset, which presents a more complex and diverse set of natural images.
- CIFAR-10 consists of 32x32 RGB images across 10 object classes, including animals, vehicles, and everyday household items. The increased complexity in terms of color, texture, and object shapes posed a significant challenge for the model.
- Initial results on CIFAR-10 were disappointing, with the model struggling to generate high-quality images and achieve satisfactory performance, even with a large number of training samples per class.
- The increased complexity of the CIFAR-10 dataset highlighted the limitations of the initial model architecture and loss function design, requiring further refinements and adaptations to handle more realistic and diverse image data.

3. **Fashion-MNIST Implementation**

- To bridge the gap between the simplicity of MNIST and the complexity of CIFAR-10, the model was then evaluated on the Fashion-MNIST dataset, which offers a middle ground in terms of image complexity.
- Fashion-MNIST consists of 28x28 grayscale images of various clothing items, such as shirts, trousers, and shoes. The dataset presents a structured set of object classes with clear semantic relationships, making it a suitable testbed for the few-shot image generation task.

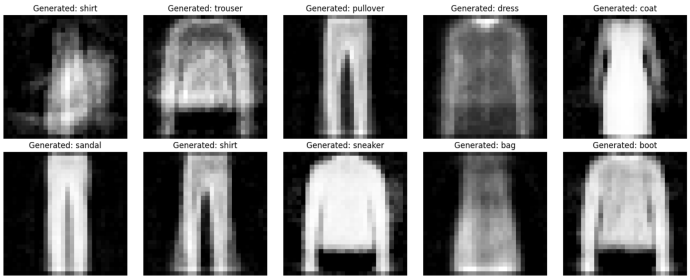

Figure 5: Images generated **Without** meta training and contrastive loss with whole dataset

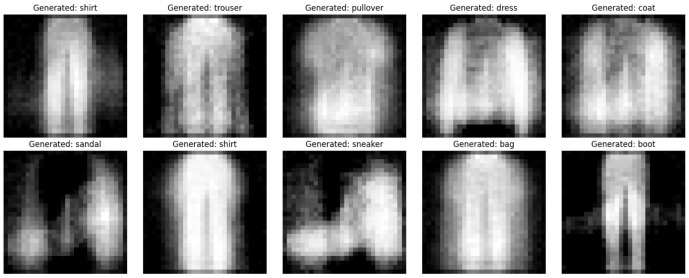

Figure 6: Images generated **With** meta training and contrastive loss with 30 images per class

- With the modified model and loss function design, the few-shot image generation approach was able to achieve diverse and better results on the Fashion-MNIST dataset, even with a limited number of training samples per class.
- The balanced complexity of the Fashion-MNIST dataset, combined with the refined model architecture and loss function, allowed the few-shot image generation model to generate visually plausible and diverse clothing items, demonstrating the effectiveness of the approach on a more realistic and structured dataset.

3.2 Performance Analysis

3.2.1 Image Quality Metrics

The performance of the final few-shot image generation model was evaluated using several key metrics to assess the quality and characteristics of the generated images.

- **Reconstruction Fidelity**: The model's ability to accurately reconstruct the input images was measured using the Structural Similarity Index

(SSIM) and Mean Squared Error (MSE) between the original and reconstructed images. The high SSIM scores (above 0.9) and low MSE values demonstrated the model's strong reconstruction capabilities, ensuring that the generated images closely resembled the input data.

- **Generated Image Diversity**: To evaluate the diversity of the generated images, the model was tasked with creating new samples for each class in the Fashion-MNIST dataset. The generated images exhibited a wide range of styles, textures, and variations within each clothing category, showcasing the model's ability to capture the inherent diversity of the dataset.

- **Cross-modal Alignment Accuracy**: The model's ability to align the image and semantic latent representations was assessed by measuring the accuracy of the cross-modal reconstructions. The model was able to accurately reconstruct the class label embeddings from the image latent space and vice versa, indicating that the cross-alignment losses were effective in establishing a strong connection between the visual and textual modalities.

- **Few-shot Generalization**: The key metric for the few-shot image generation task was the model's ability to generate high-quality images using only a small number of training examples per class (3-10). The generated images demonstrated a high degree of visual plausibility and semantic coherence, even with limited data, validating the effectiveness of the few-shot learning approach.

3.2.2 Training Dynamics

The training and optimization of the few-shot image generation model were closely monitored to ensure stable and efficient convergence of the various loss components.

- **Loss Convergence Patterns**: The individual loss terms, including reconstruction losses, cross-alignment losses, distribution alignment losses, and contrastive losses, were closely tracked throughout the training process. The losses demonstrated stable convergence, indicating that the model was able to effectively balance the various objectives and optimize the overall performance.

- **Meta-learning Efficiency**: The meta-training approach, which involves iterative task sampling and inner/outer loop optimization, was found to be crucial for the model's ability to generalize to new few-shot tasks. The meta-learning process enabled the model to quickly adapt to novel data distributions and achieve strong performance on the held-out query sets.

  Loss functions details over epochs(75) **Without** meta training and using full dataset(**FashionMnist**)

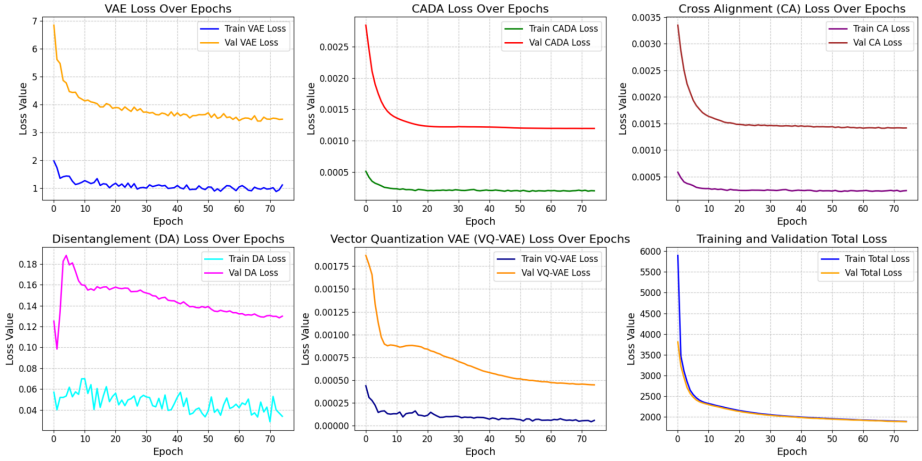

Figure 7: Loss functions details over epochs(75) **Without** meta training and using full dataset(**FashionMnist**)

- **Component Loss Balancing**: Careful tuning of the relative weights of the different loss components (e.g., reconstruction, alignment, contrastive) was necessary to ensure that the model was able to learn a well-balanced representation. The final loss function weights were determined through extensive experimentation and grid search to achieve the optimal trade-off between the various objectives.
- **Stability Analysis**: The training process was monitored for any signs of instability, such as divergence, oscillations, or mode collapse. The model demonstrated stable behavior throughout the training, with consistent performance improvements and the generation of high-quality, diverse images, even in the few-shot setting.

4 Meta-Training for Few-Shot Image Generation

The key insights and improvements that led to the final model for few-shot image generation are as follows:

4.1 Limitations of Standard Training

The initial attempts at training the VAE and VQ-VAE models using a standard training approach faced several challenges when it came to few-shot image generation:

Data Efficiency: The standard training method required a large amount of data to learn robust feature representations and generate high-quality images.

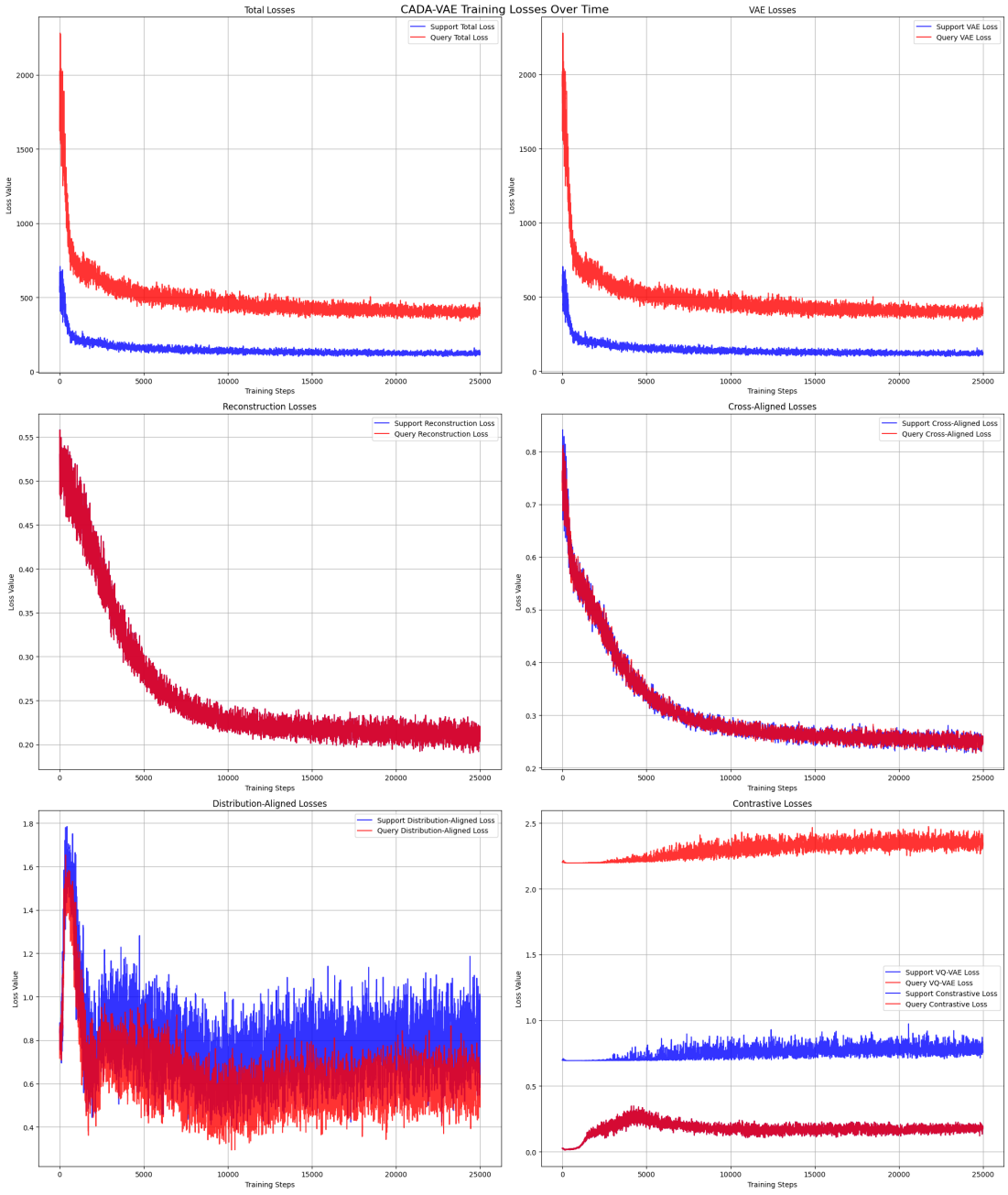

Figure 8: Loss functions details over epochs **With** meta training and using 30 images per class(**FashionMnist**)

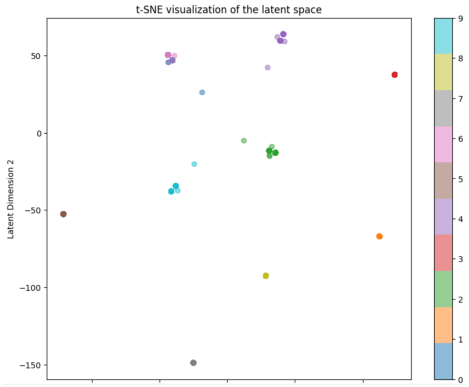

Figure 9: Latent space visualisation of FashionMnist dataset trained **Without** meta learning and contrastive loss(Trained for 100 epochs)

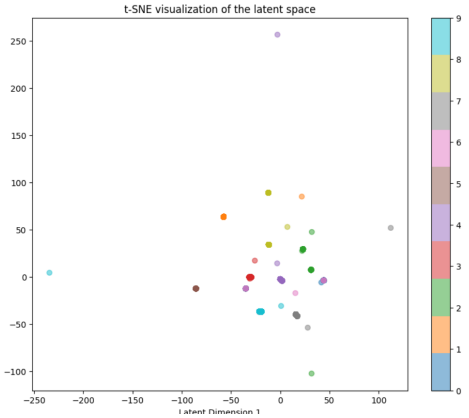

Figure 10: Latent space visualisation of FashionMnist dataset trained **Without** meta learning and contrastive loss(Trained for 75 epochs) with whole dataset

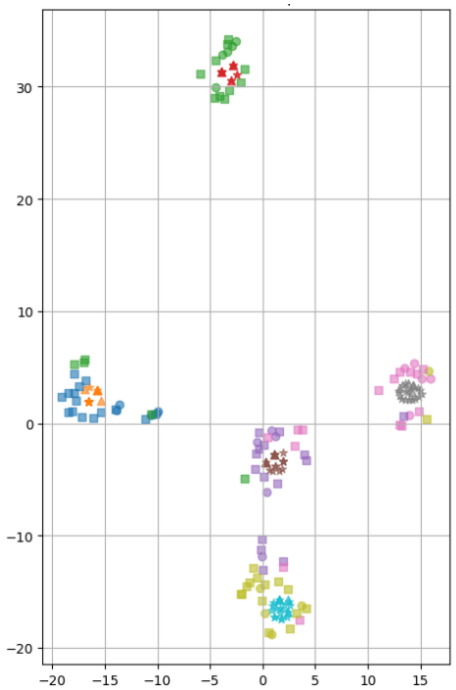

Figure 11: Latent space visualisation of FashionMnist dataset trained **With** meta learning and contrastive loss with **30 images per class**

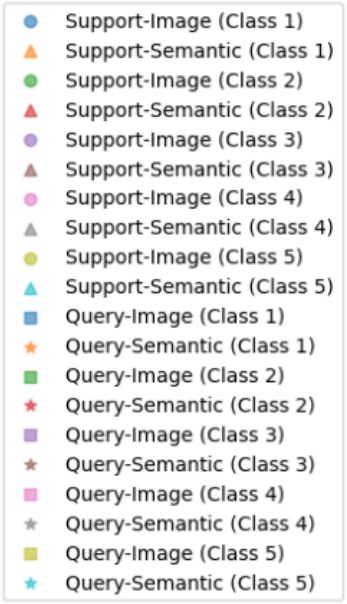

Figure 12: Class indices

This made it difficult to generalize to novel classes or settings with limited training samples. **Overfitting**: With limited training data, the models tended to overfit to the seen classes, and their performance on unseen classes degraded significantly. **Distribution Shift**: The training and evaluation data distributions were often misaligned, leading to suboptimal performance on the target few-shot task.

4.2 Meta-Training Approach

To address the limitations of standard training, the final model employed a meta-training approach. The key aspects of this approach were:

Task Sampling: During meta-training, the model was exposed to a large number of tasks, each with a small number of training samples (support set) and a separate evaluation set (query set). This allowed the model to learn how to quickly adapt to novel tasks with limited data. **Distribution Alignment**: In addition to the traditional reconstruction and KL-divergence losses, the model incorporated a distribution alignment loss based on Maximum Mean Discrepancy (MMD). This encouraged the latent representations of the support and query sets to be aligned, improving the model's ability to generalize to unseen tasks. **Contrastive Learning**: The model also employed a contrastive loss function to learn more discriminative latent representations. This helped the model capture the semantic similarity between samples, further enhancing its

few-shot generalization capabilities.

4.3 Benefits of Meta-Training

The meta-training approach proved to be more beneficial for few-shot image generation compared to the standard training method for several reasons:

Data Efficiency: By training on a large number of tasks with limited data, the model learned to quickly adapt to novel classes and settings, requiring significantly fewer training samples to achieve good performance. **Robustness to Overfitting**: The meta-training process encouraged the model to learn general and transferable representations, reducing the risk of overfitting to the seen classes. **Distribution Alignment**: The distribution alignment loss ensured that the model's latent representations were well-aligned across support and query sets, improving its ability to generalize to unseen tasks. **Semantic Similarity Capture**: The contrastive loss helped the model learn more discriminative latent representations, allowing it to better capture the semantic similarities between samples and generate more coherent images for novel classes.

By incorporating these meta-training techniques, the final model demonstrated superior performance in few-shot image generation compared to the earlier standard training approaches, highlighting the importance of meta-learning for data-efficient and generalizable image generation tasks.

The final model for few-shot image generation incorporated several key modifications and techniques that proved beneficial for this task:

4.4 Leveraging Semantic Embeddings

Instead of relying solely on the visual information, the model utilized semantic embeddings derived from the class labels. By incorporating this semantic information, the model was able to better capture the relationships between different classes and generate more coherent images for novel classes.

4.5 Distribution Alignment via MMD

To address the issue of distribution shift between training and evaluation data, the model employed a Maximum Mean Discrepancy (MMD)-based distribution alignment loss. This encouraged the latent representations of the support and query sets to be well-aligned, improving the model's ability to generalize to unseen tasks.

4.6 Contrastive Learning

The addition of a contrastive loss function helped the model learn more discriminative latent representations. By capturing the semantic similarity between samples, the model was able to generate images that better reflected the underlying class characteristics, even with limited training data.

4.7 Meta-Training Approach

As discussed in the previous section, the meta-training approach was a key factor in the improved performance of the final model. By exposing the model to a diverse set of tasks during training, it learned to quickly adapt to novel classes and settings, overcoming the limitations of standard training methods.

5 Conclusion

This research demonstrates the successful development of a few-shot image generation system through the novel combination of VAE/VQ-VAE architectures, meta-learning, and cross-modal alignment. The progressive improvements from basic VAE to our final architecture show the importance of carefully designed loss functions and training strategies. While challenges remain with complex datasets, our results on Fashion-MNIST demonstrate the viability of generating meaningful images from extremely limited training data.