Composing Generic and Specialized Diffusion Models for Greater Specialization

A Thesis Presented

by

Zhou Yu

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Engineering Artificial Intelligence

Department of Electrical and Computer Engineering

Stony Brook University

May 2025

Copyright by
Zhou Yu
2025

Stony Brook University

The Graduate School

Zhou Yu

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis.

Dr. Jorge Mendez Mendez – thesis Advisor
Department of Electrical and Computer Engineering
jorge.mendezmendez@stonybrook.edu

Dr. Murali Subbarao – Second Reader
Department of Electrical and Computer Engineering
murali.subbarao@stonybrook.edu

This thesis is accepted by the Graduate School

Celia Marshik

Dean of the Graduate School

Abstract of the thesis

Composing Generic and Specialized Diffusion Models for Greater Specialization

by

Zhou Yu

Master of Science

in

Engineering Artificial Intelligence

Stony Brook University

2025

This work seeks to improve the performance of diffusion models in specialized domains by leveraging two datasets from distinct resources. One diffusion model is trained on a *General Dataset*, which consists of a large number of data points under varied conditions, while the other is trained on a *Specialized Dataset*—a much smaller collection that captures specific conditions of interest.

Our approach leverages the compositional capabilities of diffusion models to draw samples that have high likelihood under both the generic and the specialized models, combining robust and broad distribution from the general dataset with the fine-grained, domain-specific details of the specialized dataset, thus improving performance in the target area.

We evaluate our framework across three distinct domains. To assess the capacity of our approach to specialize while effectively using broader knowledge in varied conditions, we vary the relation between the General and Specialized Datasets. Our findings reveal that we are able to achieve high performance under both the general and specialized domain by using our proposed method, jointly sampling.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Diffusion models have gained significant attention in recent years due to their ability to generate high-quality samples from a learned probability distribution by leveraging a structured noising and denoising process. Introduced by Ho et al. [3], these models learn by training on data generated gradually add Gaussian noise to real data over several time steps, transforming the data into a nearly random distribution. The model is then trained to reverse this process by progressively denoising the data through a learned iterative refinement process. By doing so, diffusion models learn to approximate the true data distribution, enabling them to generate realistic and diverse samples that closely resemble real-world data.

1.1 Problem Statement

However, the performance of diffusion models is highly dependent on the availability of a sufficiently large and diverse dataset. These models require extensive data to effectively learn complex patterns and distributions, ensuring that the generated samples maintain high fidelity and diversity. When only a very small dataset is available, the diffusion model struggles to generate high-quality samples due to the lack of sufficient feature representations. In such cases, the model fails to capture the full complexity of the data, leading to overfitting, poor generalization, and unrealistic outputs. This

limitation is particularly problematic in domains where data collection is expensive, time consuming, or restricted by privacy concerns, such as user specialization.

To illustrate our idea, consider a diffusion model that has learned the probability distributions of plate locations across the U.S. households. Our goal will be to leverage this, along with a small amount of data from an individual user's house, to learn a distribution over where plates are likely to be in this user's home. To address the challenges of limited data, we propose a novel dual-model sampling approach that leverages two diffusion models trained on different datasets: one on a larger, more general dataset, like plate locations in many households, and the other on a smaller, domain-specific dataset, such as the plate locations in one user's home. The key idea behind this strategy is to combine the broad, diverse distributions learned from the general dataset with the specialized but lower-quality distributions captured from the limited domain-specific dataset. By doing so, our approach aims to enhance the quality and relevance of generated samples, even in scenarios where the available data is scarce.

For this approach to be effective, the two datasets must share some level of connectivity, ensuring that knowledge from the general dataset can be meaningfully transferred to the specialized dataset. During the sampling process, our method directly combines the outputs from both the general and the specialized models through joint sampling. Specifically, the generated samples are required to have high likelihood under both learned distributions, ensuring that they are both high-quality and specialized. The general model contributes by capturing broad structures and producing high-quality samples, while the specialized model refines these samples to align closely with domain-specific characteristics.

This dual-model strategy introduces several advantages. First, it mitigates the issue of data scarcity by augmenting the learning process with broader knowledge from the general dataset. Second, it reduces the risk of overfitting to small datasets, as the model does not rely solely on limited information. Third, it improves the diversity of

generated samples while maintaining domain relevance, making it particularly useful in user-specialization applications. Ultimately, this approach improves the robustness and adaptability of diffusion models, enabling them to perform well even in challenging data-limited scenarios.

1.2 Structure of the Thesis

The remaining parts of this thesis are structured as follows:

Chapter 2 provides a comprehensive review of existing literature and related work on diffusion model specialization. This includes an overview of traditional machine learning approaches that address data scarcity, as well as recent deep learning techniques designed to enhance generative modeling performance in low-data regimes. Additionally, we explore prior research on multi-model and hybrid learning approaches, drawing connections to our proposed dual-model strategy.

Chapter 3 details the methodology used in our approach. We formally introduce the problem setup and present the mathematical foundations underpinning our dual-model diffusion framework. This section describes the model architecture, including the design and training process of both the general and specialized diffusion models, as well as the mechanism for adaptive sampling from both sources. We also outline the evaluation criteria used to assess the quality of generated samples, covering metrics such as the Fréchet Inception Distance (FID), structural similarity (SSIM), and other domain-specific measures. Additionally, implementation details are provided, including dataset preprocessing, training hyperparameters, and computational requirements.

Chapter 4 presents the experimental results obtained from extensive testing of our proposed approach. We compare the performance of our dual-model sampling method against baseline models trained solely on small datasets, as well as against alternative techniques for data-efficient generative modeling. Quantitative results are

accompanied by qualitative visualizations to illustrate the improvements in sample fidelity, diversity, and domain relevance. Furthermore, we conduct ablation studies to analyze the impact of different design choices and evaluate the sensitivity of our method to variations in dataset size and connectivity between general and specialized domains.

Chapter 5 concludes the thesis by summarizing the key findings and contributions of this work. We discuss the practical implications of our approach, emphasizing its advantages in real-world applications. Additionally, we acknowledge the limitations of our method. Finally, we outline potential future research directions.

Chapter 2

Related Work

2.1 Diffusion Models

Diffusion models, introduced by Ho et al. [3], have emerged as an effective alternative to traditional generative models. These models operate by gradually adding noise to training data and subsequently learning to reverse this process, generating high-fidelity samples. Their iterative nature allows them to model complex data structures while providing robust control over the synthesis process.

This popular model has soon been implemented to tackle many real-world problems. Like for super-resolution tasks, Saharia et al. [19] adapts DDPMs to a conditional generation setting. The proposed model named SR3 uses the diffusion framework to transform a low-resolution image into a high-resolution one through iterative denoising with external conditioning, achieving state-of-the-art visual quality. Also, Grad-TTS, a model found by Popov et al. [16], introduces a diffusion-based text-to-speech model that generates high-quality mel-spectrograms by gradually denoising latent noise aligned with textual input. It enables flexible and controllable speech synthesis, demonstrating the effectiveness of diffusion models in the text-to-speech domain. Moreover, Ho et al. [4] extends image-based diffusion models to the video domain, enabling temporally coherent and high-resolution video generation. By introducing a novel conditional sampling technique and joint training on image and video data, the model achieves state-of-the-art results in text-conditioned video synthesis,

video prediction, and unconditional video generation

After the idea of DDPMs became popular, numerous important improvements and extensions have been made to diffusion models. For example, by enhancing the training objective and introducing a principled guidance method, Nichol and Dhariwal [14] proposed the improved DDPM, which is able to generate diverse and high-quality images efficiently — all without adversarial training, improving both sample quality and sampling speed. To address the computational cost of regular DDPMs, Rombach et al. [17] proposed a model named LDM, which inherits the core diffusion process from DDPM but moves the entire modeling process onto a lower-dimensional latent space learned by a pretrained autoencoder. This greatly improves efficiency while maintaining, or even improving, generation quality. By adding cross-attention, LDMs also become general-purpose conditional generators. Additionally, Kingma et al. [5] present a variational framework for diffusion models that achieves state-of-the-art likelihoods with faster optimization and efficient noise schedule learning.

2.2 Diffusion Model for Data-Limited Scenarios

Despite their success, diffusion models require large-scale data to generalize effectively, and their performance tends to degrade when trained on limited or highly specialized datasets. In scenarios where data collection is costly or constrained, this data dependence poses a critical limitation. To address this, recent research has focused on improving the data efficiency of diffusion models. For instance, Giannone et al. [2] proposed Few-Shot Diffusion Models (FSDM), an extension of conditional DDPMs that incorporates patch-based set conditioning via a Vision Transformer. FSDM enables generalization to unseen classes from just a few examples, outperforming traditional DDPM baselines in low-data settings. On the other hand, Zhu et al. [21] introduced DomainStudio, a framework for adapting pre-trained diffusion models to new target

domains using limited data, while preserving both sample diversity and generation quality.

This thesis proposes a joint sampling approach, which tackles the data efficiency problem from a different perspective and can be integrated into existing diffusion frameworks to further enhance their adaptability and performance in low-data regimes.

2.3 Modular and Compositional Learning

Another promising approach to enhance specialized generative modeling is modular and compositional learning, where knowledge is structured into independent components that can be reused, composed, or adapted dynamically. Research in modular neural networks has shown that decoupling different aspects of learning can lead to better generalization between tasks [15].

Mendez et al. [12] introduced the neural composition for reinforcement learning, allowing a system to dynamically select and combine knowledge modules for decision making. Building on the idea of retaining useful knowledge over time, Schwarz et al. [20] proposed Progress & Compress, a method that consolidates past knowledge while ensuring new tasks are learned efficiently. Furthermore, Rusu et al. [18] demonstrated the benefits of Progressive Neural Networks (PNNs), which stack new network layers for each task while preserving past knowledge through lateral connections.

The methods proposed in this thesis are influenced by the work of Liu et al. [8]. This paper proposes a method called Composable Diffusion where multiple pre-trained diffusion models, each representing a concept, can be combined during inference to generate images that jointly express multiple concepts. This is made possible by interpreting diffusion models as Energy-Based Models (EBMs) — since the denoising function in diffusion models behaves similarly to the gradient of an energy function.

These approaches and ideas highlight the importance of architectural flexibility, reinforcing the idea that generative models can benefit from modular frameworks to balance generalization and specialization. Our dual-diffusion model framework aligns with this paradigm, as it leverages a general model for broader knowledge while refining outputs using a specialized model trained on domain-specific data.

2.4 Continual Learning and Knowledge Transfer

A key challenge in adapting diffusion models to small datasets is the preservation and transfer of knowledge from a broader dataset while ensuring specialization. Continual learning (CL) provides a promising avenue to address this challenge, as it enables models to retain and adapt knowledge across different tasks.

Several works have explored different strategies for continual learning, including: Regularization-based methods, such as Elastic Weight Consolidation (EWC) [6], restrict changes to important parameters to prevent catastrophic forgetting. Replay-based methods, including Gradient Episodic Memory (GEM) [9], which store small subsets of past experiences to guide future learning. And parameter isolation approaches, such as Progressive Neural Networks (PNN) [18], which allocate separate network components for new tasks to prevent interference.

Moreover, Chaudhry et al. [1] explored tiny episodic memories in continual learning, demonstrating that even small amounts of stored past data can significantly improve performance in low-data regimes. Similarly, Mendez-Mendez et al. [13] studied embodied lifelong learning, where models continuously adapt to new tasks while retaining past knowledge. These findings suggest that efficient memory utilization and adaptive knowledge integration are crucial for improving performance in small-data scenarios.

Mendez and Eaton [10, 11] also explored compositional structures in continual learning, demonstrating how modular learning approaches can enhance knowledge

transfer and task adaptation. In addition, Liu et al. [7] examined how continual learning can be leveraged for privacy-preserving unlearning, a concept that aligns with the need for adaptable, memory-efficient generative models.

Our proposed framework could be a first step in a continual learning loop, where it is possible to built upon the proposed model to incorporate knowledge from the specialized model into the general model again, then repeating the procedure for more future tasks. But there are no implementation of such ideas in this thesis.

2.5 Summary

Existing research on diffusion models, continual learning, modular architectures, and data-efficient generative modeling provides a strong foundation to tackle the challenge of small-data generative synthesis. However, current approaches struggle to balance broad generalization with domain specialization, often requiring large-scale fine-tuning or computationally expensive retraining strategies.

Our work introduces a novel dual-model diffusion framework that effectively integrates general knowledge from a broad dataset while refining details with a specialized diffusion model. This approach:

- Enhances sample diversity by leveraging large-scale pre-trained knowledge.
- Improves domain relevance by refining outputs using a specialized dataset.

By bridging the gap between broad diverse and fine-grained domain-specific distributions, our method advances the field of efficient generative modeling, enabling diffusion models to perform well even in data-constrained scenarios.

Chapter 3

Theory and Model Structure

3.1 Background

Our model is largely based on the diffusion model structure proposed by Ho et al. [3], and this section (3.1) is a summary of the original DDPM. Diffusion models are a class of generative models that learn to produce realistic data by reversing a gradual noising process. The underlying idea is adding noise to a data sample through a forward stochastic process and then learning the reverse process to recover the original data distribution as best as possible. Because of the randomness in the noise, the learned reverse process can generate diverse outputs that retain many key features of the original data. This enables diffusion models to capture complex, high-dimensional data distributions with impressive fidelity.

3.1.1 Forward Diffusion Process

Let x_0 denote a clean data sample without any noise added (e.g., an image, audio signal, or structured vector). The forward process gradually adds Gaussian noise to this sample over a sequence of T time steps, producing a sequence of increasingly noisy versions $\{x_1, x_2, \ldots, x_T\}$.

At each step $t \in \{1, 2, \ldots, T\}$, a small amount of Gaussian noise is added according to:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t \mathbf{I}\right),$$

where $\mathcal{N}(x; \mu, \Sigma)$ denotes that x is drawn from a Gaussian distribution with mean μ and covariance Σ; $\beta_t \in (0, 1)$ is a small scalar variance controlling the noise level at step t, $\mathbf{I}$ is the identity matrix, $\sqrt{1-\beta_t}$ scales the sample from the previous timestep, and $\beta_t \mathbf{I}$ is the scale of the noise added to the scaled sample.

Rather than simulating this process step-by-step, we can directly sample a noisy version x_t at any arbitrary timestep t using the closed-form equation:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}),$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$ is the accumulated signal retention factor, ϵ is standard Gaussian noise, $\sqrt{\bar{\alpha}_t}$ preserves a scaled version of the original data, $\sqrt{1-\bar{\alpha}_t}$ controls the magnitude of noise added. As $t \to T$, the signal x_t converges to pure Gaussian noise, i.e., $x_T \sim \mathcal{N}(0, \mathbf{I})$.

3.1.2 Reverse Denoising Process

The core objective of the diffusion model is to learn a reverse process that can convert a noisy sample x_t back to the clean data sample x_0. This reverse process is modeled as a parameterized conditional distribution:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathbf{I}\right),$$

where $\mu_\theta(x_t, t)$ is a neural network that predicts the mean of the reverse distribution, $\sigma_t^2 \mathbf{I}$ is a fixed variance, θ are the learnable parameters of the model.

Instead of predicting $\mu_\theta(x_t, t)$ directly, many modern approaches (like DDPM) train the model to predict the noise ϵ that was added in the forward process, and reconstruct

the clean sample from that. The relationship is:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{1-\beta_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right),$$

where $\epsilon_\theta(x_t, t)$ is the neural network's prediction of the noise at step t, $\bar{\alpha}_t$ is the cumulative product of noise schedules as before. This formulation ensures the model learns to denoise by estimating and removing noise from x_t.

3.1.3 Sampling

Once trained, the model can generate new samples by starting from Gaussian noise $x_T \sim \mathcal{N}(0, \mathbf{I})$ and iteratively applying the reverse process for T steps:

$$x_{t-1} = \frac{1}{\sqrt{1-\beta_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t \, z, \quad z \sim \mathcal{N}(0, \mathbf{I}).$$

Here, x_t is the noisy input at step t, $\epsilon_\theta(x_t, t)$ is the predicted noise, $z \sim \mathcal{N}(0, \mathbf{I})$ is additional Gaussian noise added for stochasticity (except when $t = 1$, where it's omitted), and σ_t is the standard deviation of the reverse noise schedule.

Through this iterative denoising, the model gradually transforms pure noise into a structured, high-fidelity sample that closely resembles data from the original distribution.

3.2 Proposed Approach

3.2.1 Problem Setting

In many real-world applications, data scarcity poses a significant challenge to training robust models. This is particularly true in specialized domains where only a limited amount of domain-specific data D_s is available.

Meanwhile, there exists a larger, more diverse and general data D_g that captures the

broad variations and features of the data.

However, these two datasets are not entirely disjoint; they share a degree of connectivity that allows general knowledge to be transferred to the specialized domain. The central problem is how to leverage the abundant information from D_g to enhance the performance of models trained on D_s without overfitting or losing domain-specific details.

3.2.2 Conditional Structure

In our setting, the data follows a conditional structure, where the objective is to model the conditional distribution $p(y \mid c)$, with $y \in \mathbb{R}^{d_y}$ representing the target variable and $c \in \mathbb{R}^{d_c}$ representing the conditioning context. We extend the standard diffusion framework to accommodate this structure by modifying both the forward and reverse processes accordingly.

Let $x_0 = [y, c] \in \mathbb{R}^{d_y + d_c}$ denote the concatenated clean sample comprising the target and condition. During the forward process, Gaussian noise is added only to the target variable y, while the conditioning variable c remains fixed. Specifically, for a given timestep t, the noisy input $x_t = [y_t, c]$ is generated as:

$$y_t = \sqrt{\bar{\alpha}_t}\, y + \sqrt{1 - \bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}),$$

$$x_t = [y_t, c],$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}(1 - \beta_i)$ is the accumulated noise schedule.

The reverse process is modeled to predict the noise ϵ added to y, conditioned on both the noisy sample $x_t = [y_t, c]$ and the timestep t. A neural network $\epsilon_\theta(x_t, t)$ is trained to minimize the noise prediction loss:

$$\mathcal{L}_t(\theta) = \mathbb{E}_{x_0, \epsilon, t}\left[\|\epsilon - \epsilon_\theta([y_t, c], t)\|^2\right].$$

Our implementation consists of a feedforward neural network defined as follows:

- The input is the concatenation of the noised target y_t, the conditioning variable c, and the scalar timestep t, i.e., $[y_t, c, t] \in \mathbb{R}^{d_y+d_c+1}$.
- This input is passed through an initial linear layer:

$$h_0 = \text{ReLU}(\mathbf{W}_{\text{in}}[y_t, c, t] + b_{\text{in}}),$$

- Followed by a sequence of hidden diffusion blocks:

$$h_{i+1} = \text{ReLU}(\mathbf{W}_i h_i + b_i), \quad \text{for } i = 0, \ldots, L-1,$$

- And finally projected to the noise prediction:

$$\hat{\epsilon} = \mathbf{W}_{\text{out}} h_L + b_{\text{out}}, \quad \hat{\epsilon} \in \mathbb{R}^{d_y}.$$

This design enables the model to denoise the target variable in a manner conditioned on both time and context, facilitating generation from the learned conditional distribution $p(y \mid c)$ through iterative denoising steps starting from $y_T \sim \mathcal{N}(0, \mathbf{I})$.

3.2.3 Training Procedure

We train the conditional diffusion model to predict the noise added in the forward diffusion process using a supervised learning approach. The training objective is to minimize the discrepancy between the true noise ϵ and the model's prediction $\hat{\epsilon}_\theta(x_t, t)$ at various noise levels.

Optimization Setup. We optimize the model parameters θ using the Adam optimizer with a learning rate of 10^{-3}, and apply a learning rate decay via a step scheduler with decay factor $\gamma = 0.9$ every 10 epochs.

Training Steps. For each epoch $e = 1, \ldots, N$, and for each mini-batch $\{x_0^{(i)}\}_{i=1}^{B}$ of batch size B, the following steps are performed:

1. **Sample random timesteps:** draw a timestep $t^{(i)} \sim \text{Uniform}(\{0, \ldots, T-1\})$ for each data sample, where T is the total number of diffusion steps.

2. **Forward noising:** Apply the forward diffusion process to each sample:

$$y_t^{(i)} = \sqrt{\bar{\alpha}_{t^{(i)}}}\, y^{(i)} + \sqrt{1 - \bar{\alpha}_{t^{(i)}}}\, \epsilon^{(i)}, \quad \epsilon^{(i)} \sim \mathcal{N}(0, \mathbf{I}),$$

   where only the target component $y^{(i)}$ is noised, and the context $c^{(i)}$ is concatenated unchanged. The resulting input to the model is $x_t^{(i)} = [y_t^{(i)}, c^{(i)}]$.

3. **Noise prediction:** The model predicts the noise:

$$\hat{\epsilon}_\theta(x_t^{(i)}, t^{(i)}) = \text{model}(x_t^{(i)}, t^{(i)}).$$

4. **Loss computation:** We compute the training loss using a mean squared error (MSE) objective:

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{i=1}^{B} \left\| \hat{\epsilon}_\theta(x_t^{(i)}, t^{(i)}) - \epsilon^{(i)} \right\|^2.$$

5. **Parameter update:** Backpropagate the loss and update model parameters using Adam:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta).$$

The average loss for each epoch is stored in a list for later analysis or plotting. The model is trained for N epochs.

Output. The function returns a list of average training losses over epochs, providing a learning curve for convergence monitoring.

Algorithm 1 Training Procedure for Dual-Diffusion Framework

Initialize parameters $\theta_{general}$, $\theta_{specialized}$ for General and Specialized Models.

Train the General Model:
for each epoch **do**
    Sample noise $\epsilon_g \sim \mathcal{N}(0, \mathbf{I})$
    Sample time step $t_g \in [0,\ T-1]$
    for each batch x_g **do**
        forward noise process: $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon_g$
        loss for General Model: $\mathcal{L}_g = \|\epsilon_g - \epsilon_g(x_t, t)\|^2$
        update parameters: $\theta_{general} \leftarrow \theta_{general} - \eta \cdot \nabla_\theta \mathcal{L}_g(\theta)$
    end for
end for

Train the Specialized Model:
for each epoch **do**
    Sample noise $\epsilon_s \sim \mathcal{N}(0, \mathbf{I})$
    Sample time step $t_s \in [0,\ T-1]$
    for each batch x_s **do**
        forward noise process: $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon_s$
        loss for Specialized Model: $\mathcal{L}_s = \|\epsilon_s - \epsilon_s(x_t, t)\|^2$
        update parameters: $\theta_{specialized} \leftarrow \theta_{specialized} - \eta \cdot \nabla_\theta \mathcal{L}_s(\theta)$
    end for
end for

3.2.4 Joint Sampling

To address the problem of domain adaptation under data scarcity, we propose a dual-diffusion framework. In particular, the proposed method trains two diffusion models fully independently on different datasets without sharing any parameters or computations.

- A general-purpose model, referred to as the General Model, trained on large and diverse data D_g to capture the general distribution that covers a breadth situation (eg. all locations of plates inside cabinets in households within New York).
- A Domain-specific model, referred to as the Specialized Model, trained on small but focused data D_s to capture the specific details of the target environment (eg. locations of plates inside cabinets in a individual household setting).

With the two models trained separately, we can perform joint sampling from a distribution that aligns well with both the general and specialized domains.Each model independently learns to estimate the noise component ϵ at any given timestep t of the diffusion process, and thereby contributes to the overall denoising estimate during training and sampling.

This joint sampling procedure is inspired by compositional diffusion methods by Liu et al. [8], which aggregate denoising score functions from multiple models to guide the reverse diffusion process.

Sampling: The joint sampling algorithm follows Algorithm 2 of DDPM [3], where at each timestep t, the joint noise prediction $\hat{\epsilon}$ is computed by summing the outputs of two networks:

$$\hat{\epsilon}(x_t, t) = \epsilon_{general}(x_t, t) + \epsilon_{specialized}(x_t, t),$$

where $x_t = [y_t, c] \in \mathbb{R}^{d_y}$. The sample is then updated as:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \cdot \hat{\epsilon}(x_t, t) \right) + \sigma_t z, \quad z \sim \mathcal{N}(0, \mathbf{I}),$$

where $\alpha_t = 1 - \beta_t$ is the noise scaling factor at time step t, and the noise z is only added if $t > 1$.

Output: The function returns the final denoised sample $x_0 \in \mathbb{R}^{nsamples \times d_y}$ and a list of intermediate states from the reverse trajectory $\{x_T, x_{T-1}, \ldots, x_0\}$.

Algorithm 2 Sampling Procedure for Dual-Diffusion Framework

1: Initialize $x_T \sim \mathcal{N}(0, \mathbf{I})$
2: **for** each time step t from T to 1 **do**
3: Obtain $\epsilon_{general}(x_t, t)$ from the General model
4: Obtain $\epsilon_{specialized}(x_t, t)$ from the Specialized model
5: Combine: $\hat{\epsilon}(x_t, t) = \epsilon_{general}(x_t, t) + \epsilon_{specialized}(x_t, t)$
6: Reverse process sampling:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \cdot \hat{\epsilon}(x_t, t) \right) + \sigma_t z, \quad z \sim \mathcal{N}(0, \mathbf{I}),$$

7: **end for**
8: Return x_0

Our key idea is to train the general and specialized models separately, while jointly sampling from both to generate data with high-quality characteristics of both the general distribution captured by the general model and the specialized distribution learned by the specialized model.

3.3 Summary

The dual-model diffusion framework integrates two denoising models—General and Specialized—trained on different datasets, and adaptively fuses their outputs throughout both training and sampling. This structure enables effective domain adaptation

and robust sample generation, even when the target domain suffers from limited data availability.

Chapter 4

Experiment and Results

4.1 Synthetic Data

In this section, we will simulate two real-world scenarios, and the goal is to use a model to learn and derive meaningful distributions from two datasets: the General dataset, which represents a broad, general collection of data points, and the Specialized dataset, standing for the small, specialized group of data points.
The motivation for this scenario is to equip a home assistant robot with understanding and adapting to household environments in order to identify specific data patterns. To achieve this, the robot is first trained on the general dataset, allowing it to gain a general understanding of typical data distributions.
Following this general training, the robot must adapt to a specific environment using a smaller and more specialized House dataset. Due to its limited size, this dataset alone is insufficient for training a robust model from scratch. Therefore, the robot must leverage the knowledge acquired from the general dataset to effectively adapt and perform within individual specialized conditions.

4.1.1 Data Description

Cabinet Distribution

Both the general and specialized datasets consist of one pair of 3-dimensional data points, where the first point represents the location of a cabinet shaped as a regular cube and the second representing the exact plate location within that cabinet.

Given a cabinet location (x, y, z), the the General distribution of plates is defined to be uniform between $(x-1, y-1, z)$ and $(x+1, y+1, z+1)$, forming a full-size cuboid. The specialized domain is between $(x-1, y-1, z)$ and $(x, y, z+0.5)$.

In this domain, the conditioning variable c is the x, y, z location of the cabinet, and the goal is to estimate the distribution of plate locations given the cabinet location. The general dataset contains 55398 pairs of data points $(c, plate)$ and the specialized dataset contains 128 pairs. The model takes the first data point as input location for the cabinet and the second as the location of plates conditioning on the first data point. Figure 4.1 gives a visualization of this data distribution.

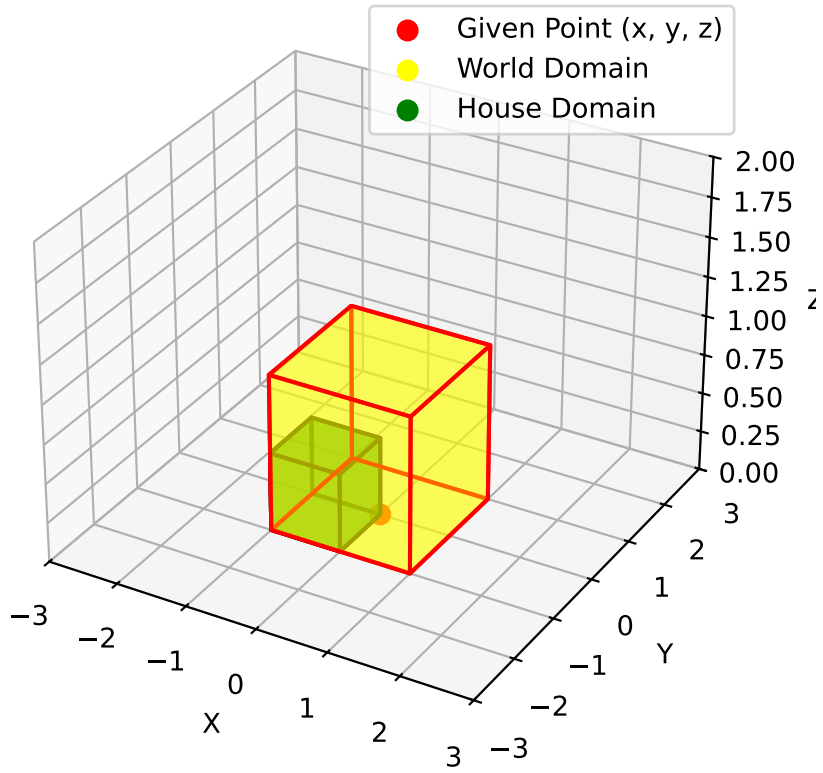


FIGURE 4.1: **Cabinet Conditions**. The red dot represents one sampled cabinet location (x,y,z) = (0,0,0). The yellow box represents the possible locations for plates inside the cabinet in the general dataset, while the green box represents the possible plate locations in the specialized dataset.

Round Table Distribution

The second data distribution shares a similar setup. Here, the model is required to model the distribution of the location of objects placed on a round table. For both datasets, the location of the round table is sampled uniformly from $(-4, -4, 0)$ to $(6, 6, 3)$, and the target object lies within a 2-D circular region located 10 units above the given point. This circle represents the surface of the table and the 10 unit is the fixed height of the table alone the z-axis.

Given a cabinet location (x, y, z), the general domain is represented as a circle with a 3-unit radius that is 10 units above the point, and the specialized domain is a circle with radius 1 at the same height. Moreover, The general dataset contains 55398 pairs

of data points $(table, object)$ and the specialized dataset contains 128 pairs. Figure 4.2 provides a straightforward visualization of this distribution.

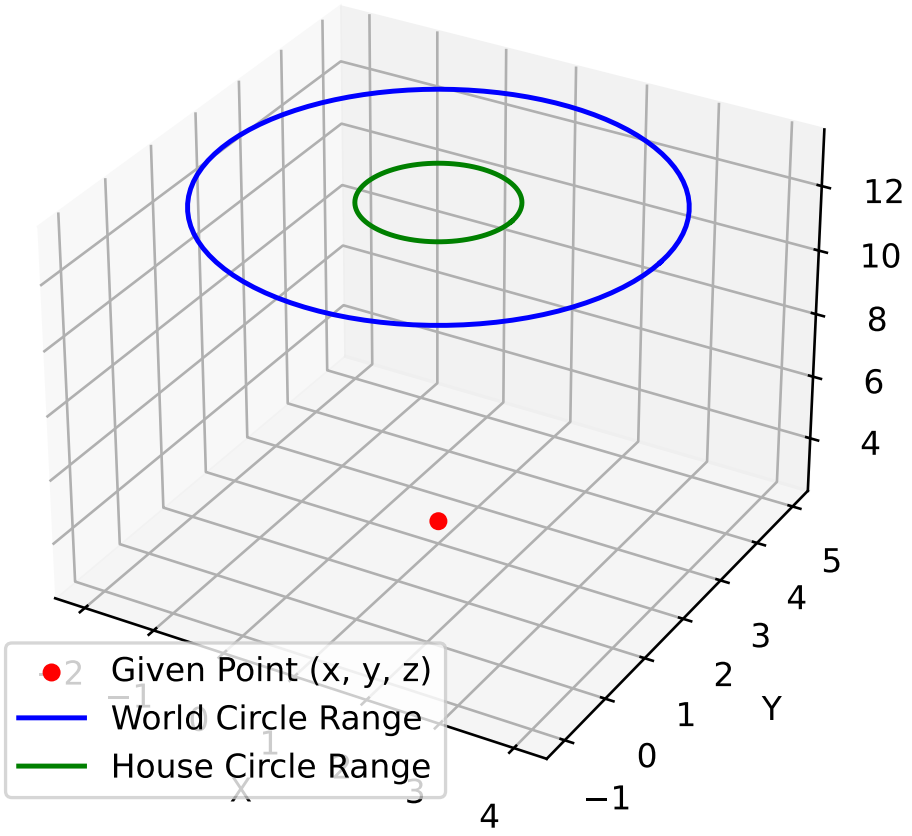


FIGURE 4.2: **Round Table Conditions.** The red dot represents one sampled round table location (x,y,z) = (1,2,3). The blue circle represents the possible locations for plates on the table in the general dataset, while the green circle shows the possible object locations in the specialized dataset.

4.1.2 Experiment Design

To evaluate the effectiveness and generalization ability of our proposed method, we conduct a comparative study using five distinct model configurations. Each model is designed to represent a unique strategy for leveraging data from both the general domain and the specialized domain. The models under comparison are as follows:

1. **general Model:** Trained solely based on the general knowledge. This model captures broad knowledge and patterns that are not specific to any single domain.

2. **specialized Model:** Trained exclusively on relatively little data from the target domain. This model reflects domain-specific knowledge and performance.

3. **Finetuned Model:** This model utilized the parameters learned from the general Model and is subsequently finetuned using the specialized data. We carefully control the number of finetuning epochs to ensure a balance between the general knowledge from the general data and the domain-specific characteristics that will be learned from the specialized data. This approach simulates a scenario where a pretrained model is adapted to a new domain with limited data access.

4. **Joint Model:** In this setup, samples from both the general and specialized models are used jointly during inference or sampling, as described in Chapter 3. This model seeks to combine the strengths of both domains, leveraging the diversity and robustness from the general Model to achieve a better performance in the specialized domain.

5. **Finetuned Joint Model:** This approach combines the general Model and the Finetuned Model the same way it was used in the Joint Model. By sampling from both, we aim to benefit from the robustness of the general Model, the domain-awareness of the specialized Model, and the subtle adaptations learned during finetuning. This hybrid strategy seeks to exploit the advantages of both finetuning and composition.

Each model is evaluated on the same set of 100,000 tasks, each generating 10 samples for every test point. With these one million samples, we report key performance metrics to assess their accuracy, adaptability, and generalization capabilities across domains. This experimental design provides insights into the trade-offs and benefits of various domain adaptation and fusion strategies.

4.1.3 Evaluations Metrics

The performance of different models are evaluated in three matrices:

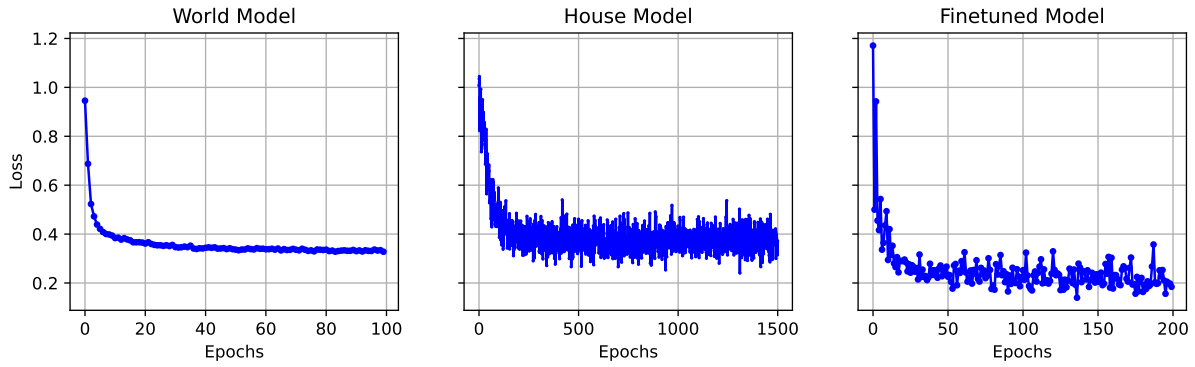
Average Accuracy a dataset containing 1000 randomly given condition points is created and checked to see if they are within the range. Given each test condition, the models will generate 10 random samples and calculate how many points are within the range, then average accuracy is calculated by taking the mean of all 1000 test' accuracies.

Average Distance Using the same test data, the distance between all sampled points outside the range and the given test point (cabinet/roundtable locations) is calculated and averaged to obtain more detailed information from the errors.

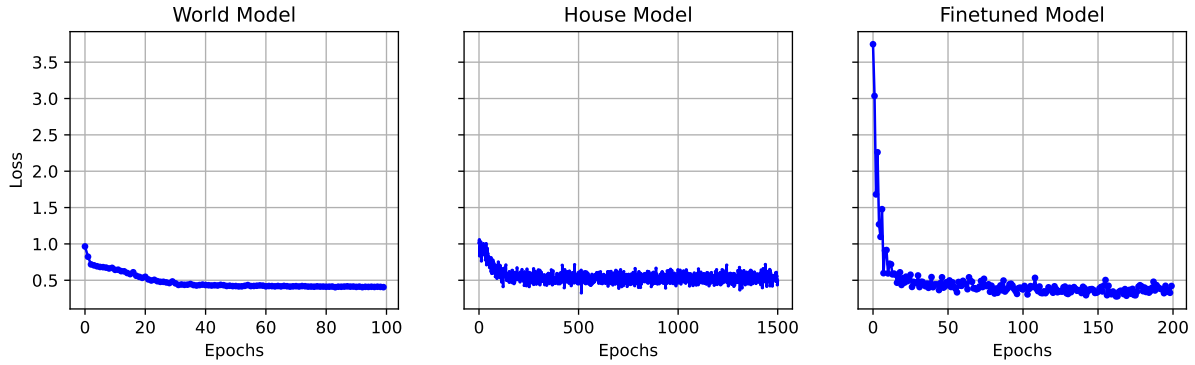
Visualization graph We pick one individual (x,y,z) location for the cabinet or table and draw multiple samples from each model and draw a scatter plot of all the samples. This metric has a lot of uncertainty if choosing different test points so it is only used for visualization purposes.

4.1.4 Results and Analysis

We apply the same structure for all diffusion models using identical number of blocks, hidden layers and number of nodes each layer. But the training epochs are distinct: for the general Model, we do not have enough resources to train too long since it contains a lot of data, so we used 100 epochs with each epoch size equal to 2048; for the specialized Model, we used a batch size of 64 and 1500 epochs to fit the small amount of specialized data as accurately as possible; for the Finetuned Model, we used the same batch size as the specialized model but only 200 epochs to avoid overwriting (forgetting) knowledge from the [general/world] data completely. The training losses are shown below in Figure 4.3.



(A) Cabinet Data



(B) Round Table Data

FIGURE 4.3: **Training Losses.** All models are trained until convergence, but models trained on smaller amounts of data have noisier training curves, as expected.

According to the training loss, all models are able to train to convergence using data from both distributions. Then let's jump in to the test data. Here are the tables recording all the testing numbers from the experiment.

TABLE 4.1: **Model Evaluation Results of Cabinet Data**

Model	**Avg. Accuracy**	**Avg. Out-of-Range Distance**
World Model	0.0835	0.6656
House Model	0.0932	1.0521
Finetuned Model	0.4874	0.4139
Joint Model	0.3464	0.4138
Finetuned Joint Model	**0.6516**	**0.3852**

TABLE 4.2: Both the general and the specialized models performed poorly, yet finetuning and joint sampling both individually improved performance. The best performance across both metrics was obtained by combining the strengths of finetuning and joint sampling.

TABLE 4.3: **Model Evaluation Results of Round Table Data**

Model	**Avg. Accuracy**	**Avg. Out-of-Range Distance**
World Model	0.0904	10.6076
House Model	0.0386	17.7673
Finetuned Model	0.3624	10.2100
Joint Model	**0.6653**	**8.0266**
Finetune Joint Model	0.2482	10.2289

TABLE 4.4: Both the general and the specialized models performed badly, but finetuning and joint sampling both individually improved performance. The best performance across both metrics was the joint model, obtained by joint sampling alone without finetuning.

Not only there are one million samples tested, the results has been run for three times each distribution. For the Cabinet dataset (represented in table 4.1), the Finetuned Joint Model achieved the best performance, with the highest accuracy and the lowest out-of-range distance, indicating effective adaptation through combined training and fine-tuning. The Finetuned Model also performed well, while the general and specialized models showed limited capability when trained independently.
In contrast, for the Round Table dataset (represented in table 4.2), the Joint Model performed best. This suggests that the evidence is insufficient to determine when fine-tuning will help because it does help in the first but not the second domain, but joint sampling improves results across both cases.

4.1.5 Overlapping Examination

One limitation identified in prior experiments is the fixed relationship between the specialized and general datasets: we have so far assumed that the support of the specialized distribution lies completely within the support of the general distribution. However, this assumption does not reflect real-world scenarios. In practice, the relationship between domain-specific data and general data is often more complex: there may be significant overlap or shared structures, but the supports are rarely in a strict subset relationship. This oversimplification may restrict the model's ability to generalize to realistic domain adaptation settings. We next empirically evaluate the impact of changing this relationship between the distributions' supports. We have designed 6 different situations:

1. **No Overlapping and far away** There is absolutely no similarity between the general and specialized data. Their domains are 10 units apart,

2. **No Overlapping but close** There is no overlapping between the two support sets but they are close to each other, separated by 3 units.

3. **Little Overlapping** There is less than 50% overlap, meaning the dataset is a little overlapped. Their domains have crossed a little.

4. **Significant Overlapping** There is less than 50% overlap, the majority of the two datasets are within the same range. Their domains intersect largely.

5. **Subset but right shifted** The specialized data is the subset of the general data but not strict, meaning it maybe at the corner or edge of the general distribution.

6. **Strict Subset** The specialized data is a strict subset of the general data, with identical shape of domains.

We used Round table distribution to perform this experiment and no parameters have been changed except the specialized dataset distribution. Table 4.3 contains the results.

TABLE 4.5: **Model Performance Across Different Levels of Dataset Overlapping**

Overlap Setting	Model	Avg. Accuracy	Mean Distance
No Overlap (Far)	World	**0.0826**	**3.3536**
	House	0.0000	22.6468
	Joint	0.0000	5.1017
No Overlap (Close)	World	**0.0817**	**3.3738**
	House	0.0000	18.8574
	Joint	0.0057	3.0952
Partial Overlap (<50%)	World	**0.0835**	**3.3511**
	House	0.0002	16.9943
	Joint	0.0630	2.3889
Strong Overlap (>50%)	World	0.0817	3.3354
	House	0.0001	20.4825
	Joint	**0.0880**	**2.2146**
Subset (Shifted)	World	0.0803	3.4022
	House	0.0031	17.6120
	Joint	**0.2779**	**1.5768**
Subset (100% Overlap)	World	0.0784	3.3525
	House	0.0140	16.7189
	Joint	**0.5876**	**1.2642**

TABLE 4.6: Both the general and the specialized models performed badly, but finetuning and joint sampling both individually improved performance. The best performance across both metrics was the joint model, obtained by joint sampling alone without finetuning.

Overlap Experiment Analysis. Results in the above table reveal a clear trend: as the overlap between the specialized and general datasets increases, the performance of the Joint model improves significantly. In the case of full overlap, the Joint model achieves its highest accuracy and lowest mean distance. When the datasets are disjoint, especially with distant distributions, the Joint model fails to improve over the general model. Notably, even a partial overlap allows the Joint model to outperform both general and specialized models, demonstrating the importance of shared distributional structure for effective joint training. The specialized model consistently performs poorly unless there is full or near-full overlap, which is not a surprise for its lack of data.

4.2 Simulated 2D Robotic Data

4.2.1 Data Description

The dataset used in this section originates from the simulated environment developed by Mendez et al. [13]. For our experiments, we selected a subset of the simulated data to perform our experiment. Specifically, we focused on those collected from a simulated robot performing a series of *navigate-to* actions. It contains more complicated patterns comparing to others so we can have a better view of the model's learning abilities. This setup allows us to model and evaluate the system's performance in controlled yet diverse spatial scenarios.

The dataset contains navigate_to actions for various target object types (e.g., navigate_to_table, navigate_to_book, navigate_to_ball, ect.). For each object, the dataset contains target (x,y) coordinates that would enable the robot to reach the target object with its arm, given 10 features describing the current state of the robot and the target object (e.g., positions, orientations). For each object type, the robot executed multiple tasks, each of which required executing multiple navigate_to actions. In consequence,

the dataset contains multiple data points corresponding to each task, as indicated by a task index. To measure the ability of the learned models to generalize to unseen tasks, we use the task index to split the data into training and test data. The general dataset consists of 9 different navigation tasks (navigate to Basket, Bin, Book, Box, Cup, Cupboard, Shelf, Stick, and Tray) with the total size of 1918048 data points. Figure 4.4 shows the distibuiton of this dataset.

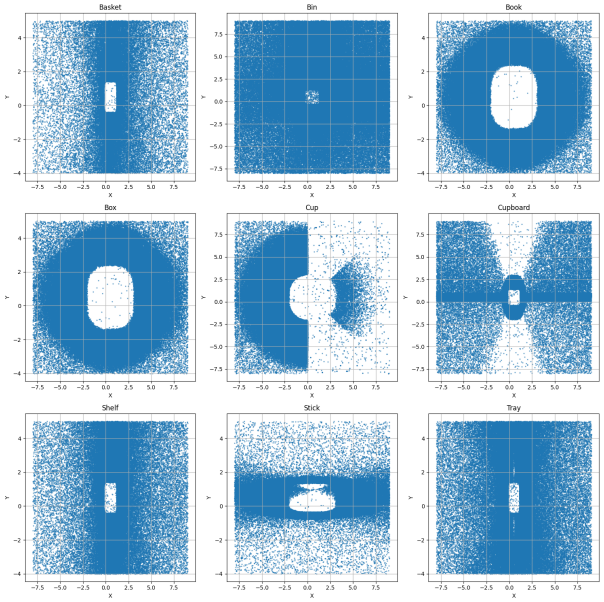


FIGURE 4.4: **Robot 2D General Dataset Distribution** This is the distribution of 9 different tasks within the training set for general model. The data plotted here are all original and haven't been normalized.

And the specialized data only contains navigate to ball, and is split into a training set of 442 data points and a test set of 13398 ones. Figure 4.5 shows the specializedd data plotting.

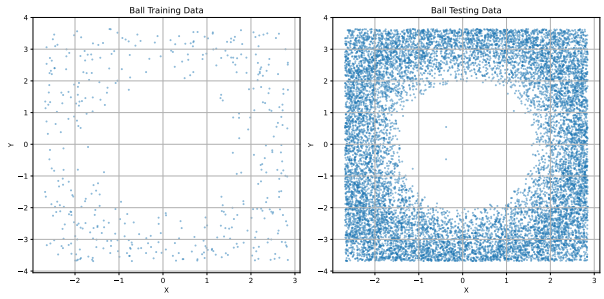


FIGURE 4.5: **Robot 2D Specialized Dataset Distribution** This is the distribution of the only task within the data set for general model. the left is the raining set and right is the test set. The data plotted here are all original and haven't been normalized.

4.2.2 Experiment Design and Evaluation

For this experiment, We employed a significantly larger model architecture compared to what was used for the synthesized dataset, as the robotic dataset contains approximately 2 million data points. This larger network is necessary to capture the increased complexity and patterns within the data.

Due to the lack of suitable numerical evaluation metrics for this type of robotic data, we opted for direct visualization as our primary evaluation approach. Since the target outputs (actions) are two-dimensional, the visualizations are both interpretable and informative.

The training results are shown in the figures below.

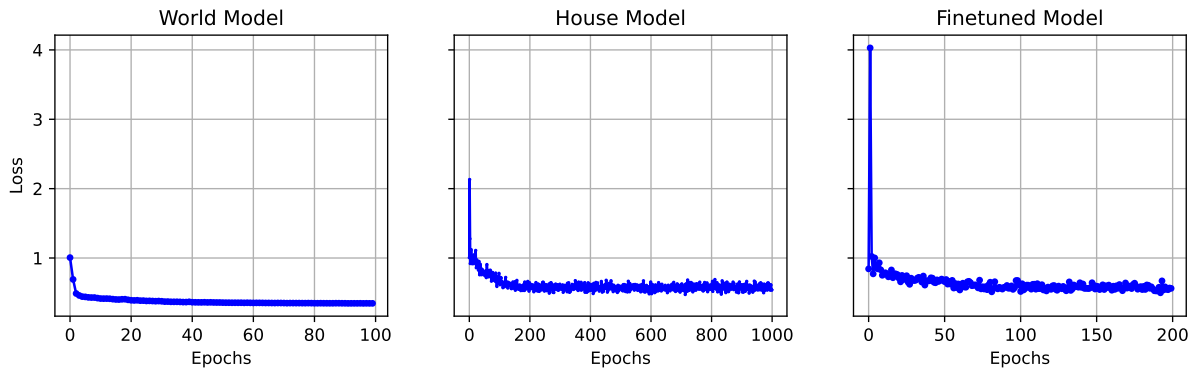


FIGURE 4.6: **Robot Data Training Loss** General, Specialized, and Finetuning Models are all trained until reaching convergence. It is shown in the graph that the General model has lower training loss than others.

The three learning curves show that the models were able to train to convergence, as desired.

4.2.3 Results and Analysis

We evaluated the same five models used in our synthesized data experiments: the general, specialized, Finetuned, joint, and Finetuned Joint Models. Figure 4.7 show the corresponding visualization results.

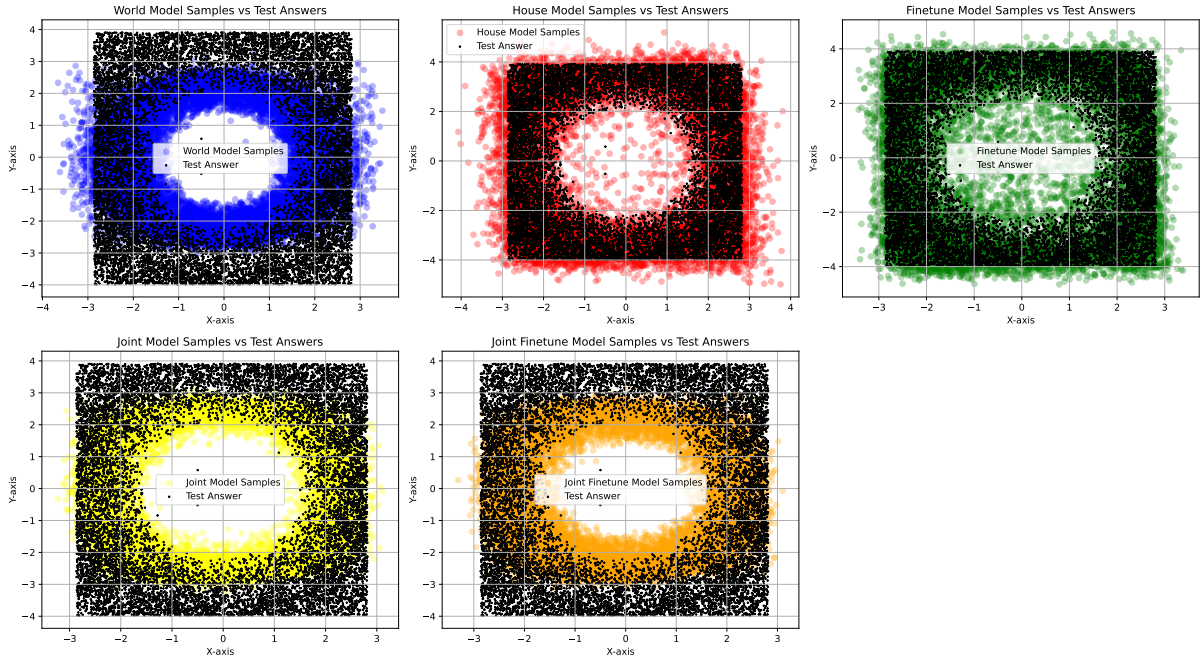


FIGURE 4.7: **Robot Data Sampling Comparison** The test set has shape [13398, 12], and every model extracts one sample for every test point. This is the sampling result from five different models: the blue one is the general (world) model, the red one is the specialized model, the green one is the world model finetuned on specialized data, the yellow one is the joint samples using general and specialized model and the orange one is the joint samples using general and finetuned model.

From the visualizations above, we observe that the specialized model has the best performance (house model), the general Model tends to sample only within a limited region of the target space while the Specialized and Finetuned models are able to capture the full test-data space. And because of the world model and joint sampling, both Joint models are unable to capture the full pattern.

This result corresponds to the previous overlapping experiment results in section 4.1.5. Since the 2D robot data used here are split according to different tasks: navigate_to_balls as the testing data and the others as the training data, the test data cannot be a subset of the training data. They do possess differences, and I have conducted a Principle Component Analysis (PCA) between the training data and the testing data to see if they vary a lot. But the result turns out they are quite similar. The results are shown in Figure 4.8.

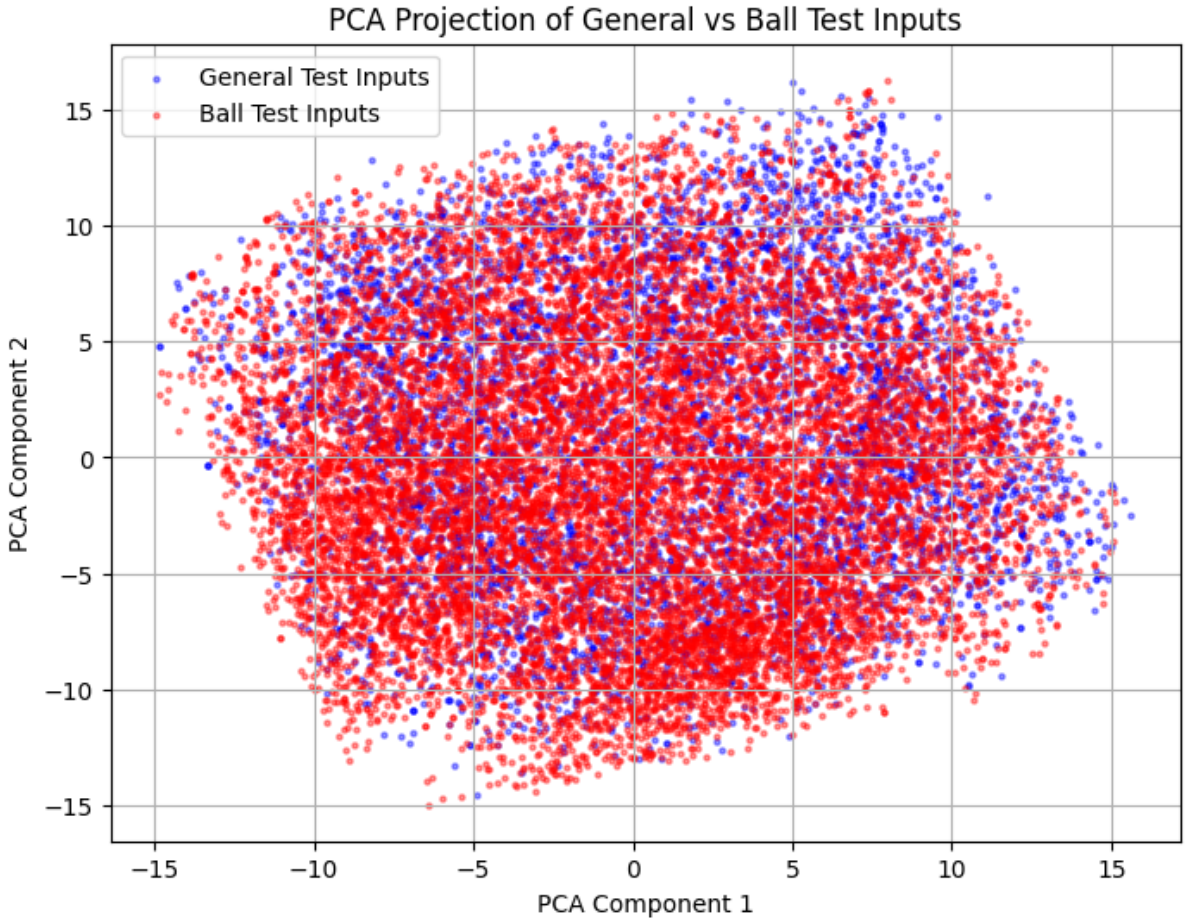


FIGURE 4.8: **Principle Component Analysis Graph** This graph represents the similarity between the robot training data and robot testing data. The result suggests they are very similar datasets since they almost overlap together.

Chapter 5

Conclusion

In this thesis, we proposed an improved sampling process for diffusion models and introduced the concept of joint sampling from two distinct models. Our experiments demonstrate that this approach can significantly improve the specialization capability of diffusion models, especially in scenarios where high-quality data access is limited in the target domain. The central motivation lies in leveraging abundant data from related or adjacent domains to support learning in data-scarce fields—an approach with meaningful implications for real-world applications.

Despite its potential, the proposed method has limitations. Currently, the joint sampling mechanism simply aggregates noise from both models without any adaptive control. Introducing a learnable parameter to modulate the noise contribution from each model could improve accuracy and efficiency. Furthermore, the effectiveness of joint sampling heavily depends on the similarity between datasets; if the domains are too dissimilar, performance may degrade.

Overall, this project offers a novel perspective on domain specialization in generative modeling and represents a promising step forward in the broader field of artificial intelligence.

Bibliography

[1] Arslan Chaudhry et al. "On tiny episodic memories in continual learning". In: *arXiv preprint arXiv:1902.10486* (2019).

[2] Giorgio Giannone, Didrik Nielsen, and Ole Winther. *Few-Shot Diffusion Models*. 2022. arXiv: 2205.15463 [cs.CV]. URL: https://arxiv.org/abs/2205.15463.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in Neural Information Processing Systems* (2020).

[4] Jonathan Ho et al. *Video Diffusion Models*. 2022. arXiv: 2204.03458 [cs.CV]. URL: https://arxiv.org/abs/2204.03458.

[5] Diederik P. Kingma et al. *Variational Diffusion Models*. 2023. arXiv: 2107.00630 [cs.LG]. URL: https://arxiv.org/abs/2107.00630.

[6] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the National Academy of Sciences* (2017).

[7] Bowen Liu, Qiang Liu, and Peter Stone. "Continual learning and private unlearning". In: *Conference on Lifelong Learning Agents* (2022).

[8] Nan Liu et al. *Compositional Visual Generation with Composable Diffusion Models*. 2023. arXiv: 2206.01714 [cs.CV]. URL: https://arxiv.org/abs/2206.01714.

[9] David Lopez-Paz and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning". In: *Advances in Neural Information Processing Systems* (2017).

[10] Juan A. Mendez and Eric Eaton. "How to reuse and compose knowledge for a lifetime of tasks: A survey on continual learning and functional composition". In: *arXiv preprint arXiv:2207.07730* (2022).

[11] Juan A. Mendez and Eric Eaton. "Lifelong learning of compositional structures". In: *arXiv preprint arXiv:2007.07732* (2020).

[12] Juan A. Mendez, Harm van Seijen, and Eric Eaton. "Modular lifelong reinforcement learning via neural composition". In: *arXiv preprint arXiv:2207.00429* (2022).

[13] Jorge Mendez-Mendez, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. *Embodied Lifelong Learning for Task and Motion Planning*. 2023. arXiv: 2307.06870 [cs.RO]. URL: https://arxiv.org/abs/2307.06870.

[14] Alex Nichol and Prafulla Dhariwal. *Improved Denoising Diffusion Probabilistic Models*. 2021. arXiv: 2102.09672 [cs.LG]. URL: https://arxiv.org/abs/2102.09672.

[15] Jonas Pfeiffer et al. *Modular Deep Learning*. 2024. arXiv: 2302.11529 [cs.LG]. URL: https://arxiv.org/abs/2302.11529.

[16] Vadim Popov et al. *Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech*. 2021. arXiv: 2105.06337 [cs.LG]. URL: https://arxiv.org/abs/2105.06337.

[17] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV]. URL: https://arxiv.org/abs/2112.10752.

[18] Andrei A. Rusu et al. "Progressive neural networks". In: *arXiv preprint arXiv:1606.04671* (2016).

[19] Chitwan Saharia et al. *Image Super-Resolution via Iterative Refinement*. 2021. arXiv: 2104.07636 [eess.IV]. URL: https://arxiv.org/abs/2104.07636.

[20] Jonathan Schwarz et al. "Progress & compress: A scalable framework for continual learning". In: *International Conference on Machine Learning* (2018).

[21] Jingyuan Zhu et al. *DomainStudio: Fine-Tuning Diffusion Models for Domain-Driven Image Generation using Limited Data*. 2024. arXiv: 2306.14153 [cs.CV]. URL: https://arxiv.org/abs/2306.14153.