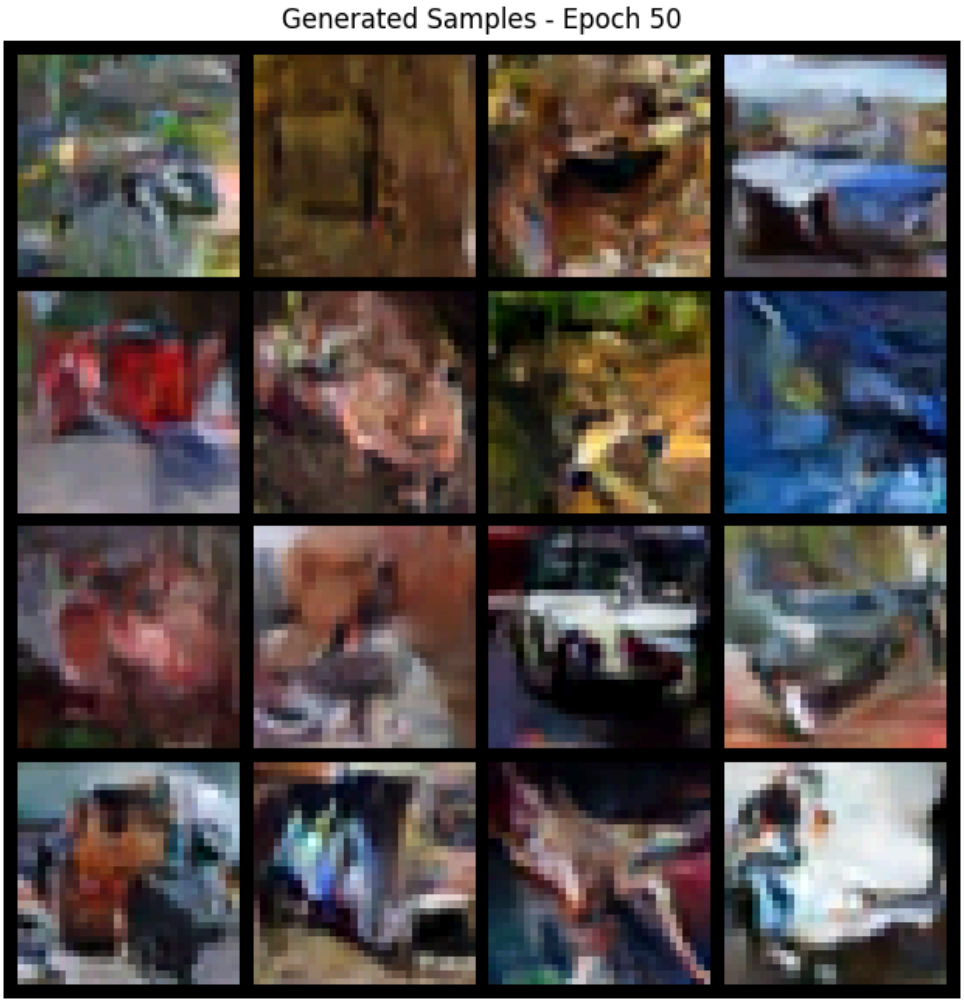
Analysis Report

Creating Images with Diffusion Models

Submitter: Varit Kobutra
Submitted for:
Prof. Patricia McManus
ITAI 2376 Deep Learning
Houston Community College
April 2025



Dataset Chosen: CIFAR-10 (Advanced - Bonus +5%)
Bonus Features Implemented: CLIP Evaluation (+10%)
Link to Google Colab Notebook:
https://colab.research.google.com/drive/1hADd8HMJT5Bzv7-5ADuuS4B-lEJJX88I

Table of Contents

Introduction

This report details the implementation, training, and evaluation of a Denoising Diffusion Probabilistic Model (DDPM) designed to generate images corresponding to the CIFAR-10 dataset classes. The project aimed to provide hands-on experience with the theory, architecture, and training dynamics of modern diffusion-based generative models.

Methodology

A U-Net neural network architecture incorporating time and class conditioning was implemented. The model was trained using the DDPM framework, involving a forward process of gradual noise addition and a learned reverse process to remove noise. Training utilized Mean Squared Error (MSE) loss between the model's predicted noise and the actual noise added at each timestep.

Dataset Choice & Adaptation

Recognizing that the provided base notebook appeared oriented towards simpler datasets like MNIST, this project undertook the more challenging CIFAR-10 dataset (32x32 color images) to gain deeper insights and leverage the bonus opportunity. This required adapting and developing a new notebook structure suitable for this more complex task.

Evaluation

Model performance was assessed through qualitative visual inspection of generated image samples throughout the training process and quantitatively via Contrastive Language-Image Pre-training (CLIP) scores to measure the semantic alignment between generated images and text descriptions of the target classes.

Structure

This report outlines the implementation steps, details the training process and results, presents the CLIP evaluation findings, addresses the assignment's assessment questions, and concludes with a discussion of limitations and future work.

Implementation Details

Dataset Setup and Preparation

- The CIFAR-10 dataset was loaded using *torchvision.datasets.CIFAR10*.
- Preprocessing included converting images to PyTorch tensors, normalizing pixel values to the *[−1,1]* range required by the diffusion model, and applying random horizontal flips during training for data augmentation.
- A 90%/10% train-validation split was created using torch.utils.data.random_split, with a fixed random seed ensuring reproducibility.
- *DataLoader* instances were configured for efficient batching (Batch Size: 128), shuffling of training data, and parallel data loading using multiple workers (num_workers=2). Initial checks confirmed successful data loading and transformation.

Model Architecture Components

- Time embeddings were generated using *SinusoidalPositionEmbeddings* based on the Transformer architecture.
- A core residual *Block* was implemented featuring two convolutional layers, *GroupNorm* for normalization, *SiLU* activation functions, and integration of time and class embeddings via a learned MLP projection.
- *DownBlock* and *UpBlock* modules were created to implement the contracting and expanding paths of the U-Net, respectively, handling max-pooling for downsampling, bilinear upsampling, and concatenation of skip connection features.
- The final *UNet* architecture was assembled, connecting the initial convolution, downsampling blocks, bottleneck layers, upsampling blocks (correctly handling skip connection channel dimensions), and the final output convolution. Class conditioning was implemented by adding learned class embeddings to the time embeddings before the MLP projection. Debugging ensured correct tensor flow and channel dimensions throughout the network.

Diffusion Process Implementation

- Diffusion hyperparameters were set *(TIMESTEPS = 1000, BETA_START = 0.0001, BETA_END = 0.02)*, and a linear noise schedule was used to precompute variance (β_t), and cumulative alpha (α_t) values.
- The forward process *q_sample* function was implemented to directly compute noisy images χ_t from clean images χ_0 using the closed-form equation

  $\chi_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon.$

- The p_losses function calculated the training objective by sampling timesteps t, generating x_t via *q_sample*, obtaining the model's noise prediction $\epsilon_0(x_t, t, class)$, and computing the MSE loss against the true noise ϵ.
- A *training_step* function streamlined the process of sampling timesteps and calculating the batch loss.

Training and Evaluation Setup

- The reverse diffusion sampling step (p_sample) and the full iterative sampling loop (p_sample_loop) were implemented to generate images by starting from noise x_T and progressively denoising using the trained model.
- Training was configured with the AdamW optimizer (lr=1e-4), a target of 50 epochs, gradient clipping (norm=1.0), and paths for saving the model state and generated samples.
- The main training loop iterated through epochs, performing training steps (loss calculation, backpropagation, optimizer update) and validation steps. Samples were generated and saved every 5 epochs.
- Post-training cells were added for robust evaluation: loading the saved model state, visualizing final samples, generating samples conditioned on specific classes, visualizing sample quality progression over epochs, and re-plotting loss curves.

CLIP Evaluation

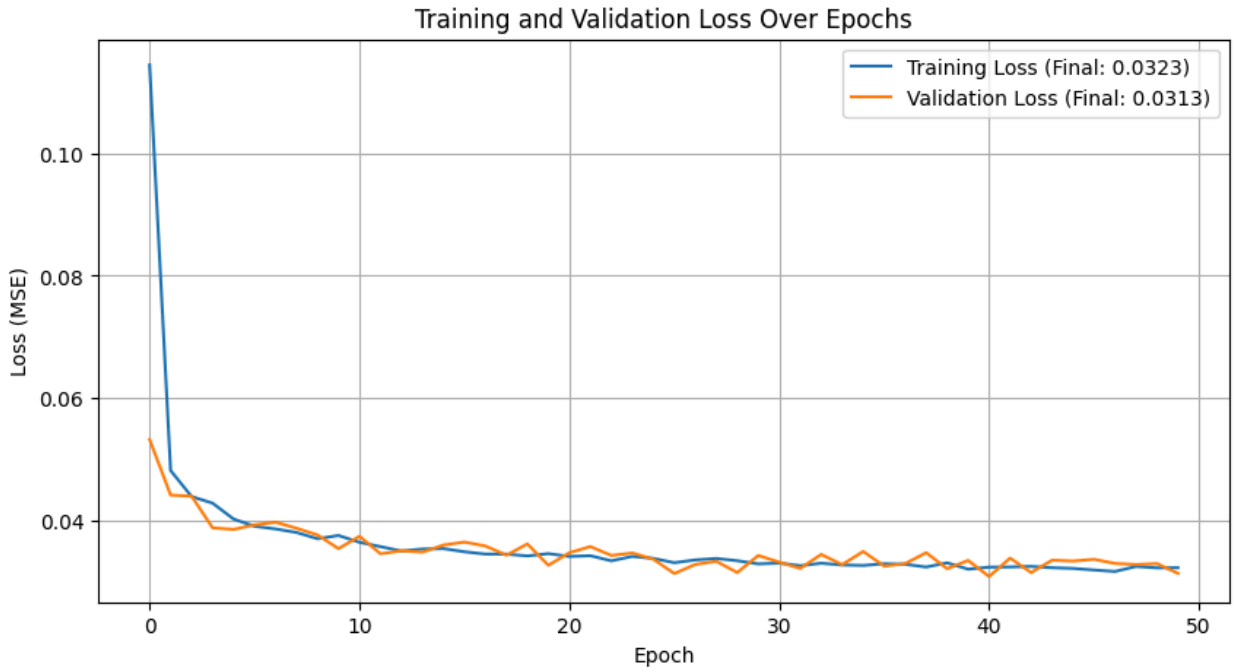
- A pre-trained *ViT-B/32* CLIP model and its associated image preprocessor were loaded.
- Text prompts ("a photo of a [class_name]") for the 10 CIFAR-10 classes were generated and encoded into normalized text feature embeddings.
- Generated image samples from the final training epoch were loaded for evaluation.
- Images were preprocessed using clip_preprocess, and normalized image feature embeddings were computed using the CLIP image encoder.
- A cosine similarity matrix between all image embeddings and all text embeddings was calculated.
- Analysis included calculating the average top-1 similarity score. Matching accuracy and intended similarity were skipped as the loaded samples lacked ground-truth label association.
- Visualizations were created showing generated images with their best-matching CLIP text label and a heatmap of the full similarity matrix.

Training and Results

Training Environment & Setup

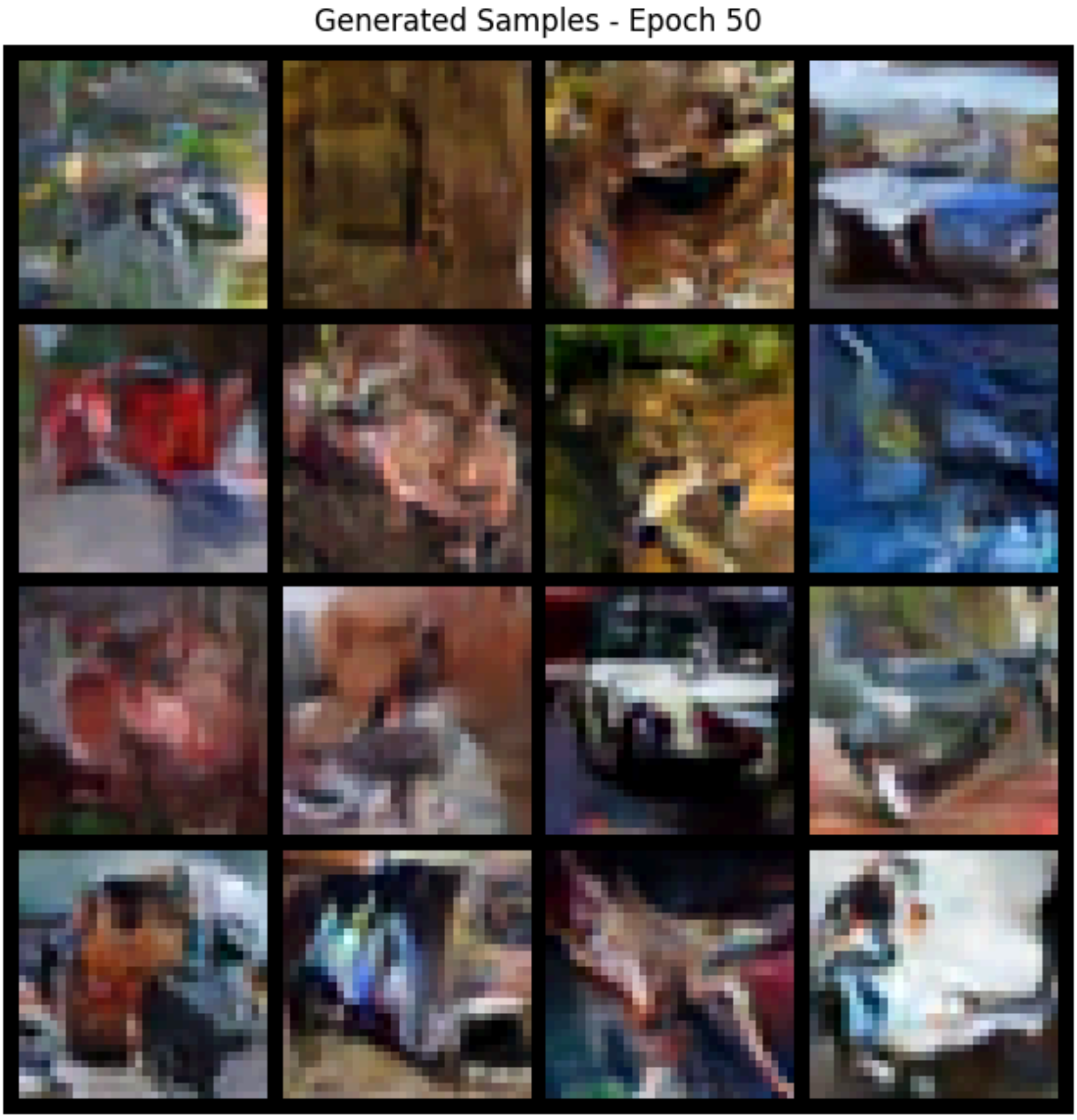
- Training was conducted using Google Colab, utilizing a T4 GPU.
- The model was trained for 50 epochs with a batch size of 128. The AdamW optimizer was used with a learning rate of 1×10^{-4}. The diffusion process used 1000 timesteps.

Loss Analysis

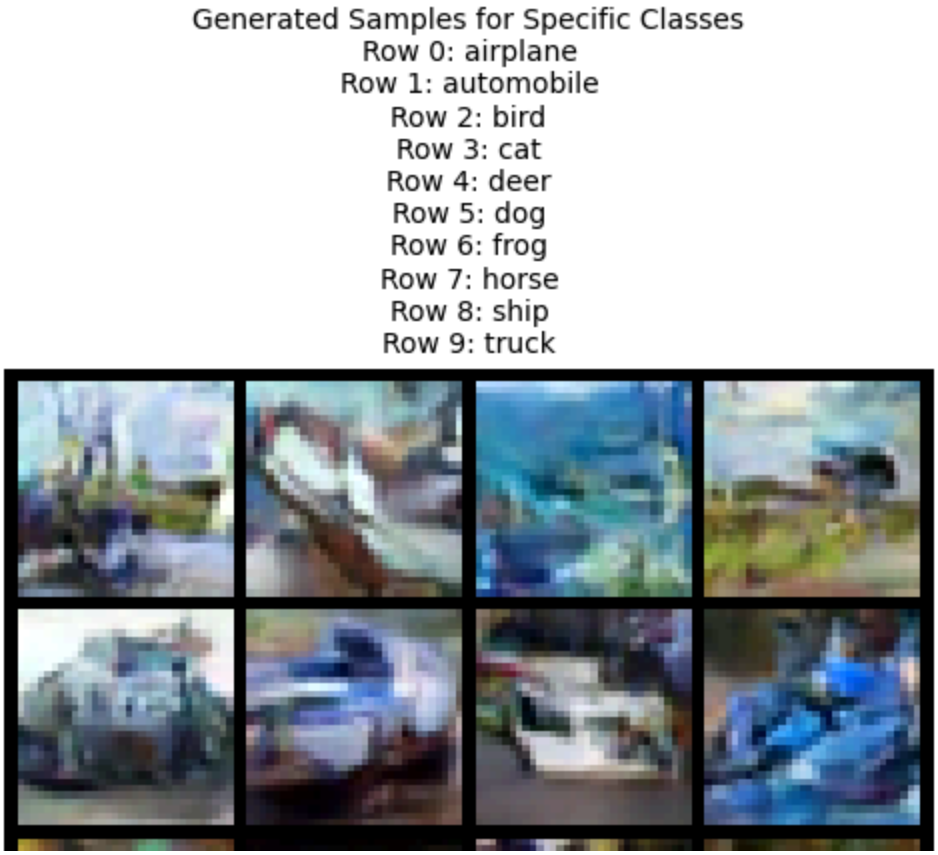


The training and validation loss curves demonstrate successful learning. A sharp decrease in loss occurred within the first few epochs (Train Loss Epoch 1: 0.1144 -> Epoch 2: 0.0482). Subsequently, both training and validation losses continued to decrease steadily but more gradually throughout the 50 epochs, reaching final values of approximately 0.0323 (training) and 0.0313 (validation). There were no significant signs of overfitting within this training duration, as the validation loss generally tracked the training loss downwards.

Generated Sample Analysis



Visual inspection of the samples generated after 50 epochs reveals that the model has learned to produce image-like outputs that are distinct from random noise, but they do not yet clearly resemble objects from the CIFAR-10 classes. The images exhibit abstract shapes, color distributions characteristic of the dataset (e.g., blues for sky/water, greens/browns for landscapes/animals), but lack fine details, coherent object structures, and photorealism.



When generating samples conditioned on specific class labels, the resulting images remain abstract. While subtle differences in color palettes or textures might exist between rows corresponding to different requested classes (e.g., potentially more blue/white for 'airplane' or 'ship'), distinct, recognizable objects corresponding to the labels are not formed after 50 epochs. The model's ability to strongly adhere to class conditions is not yet evident.

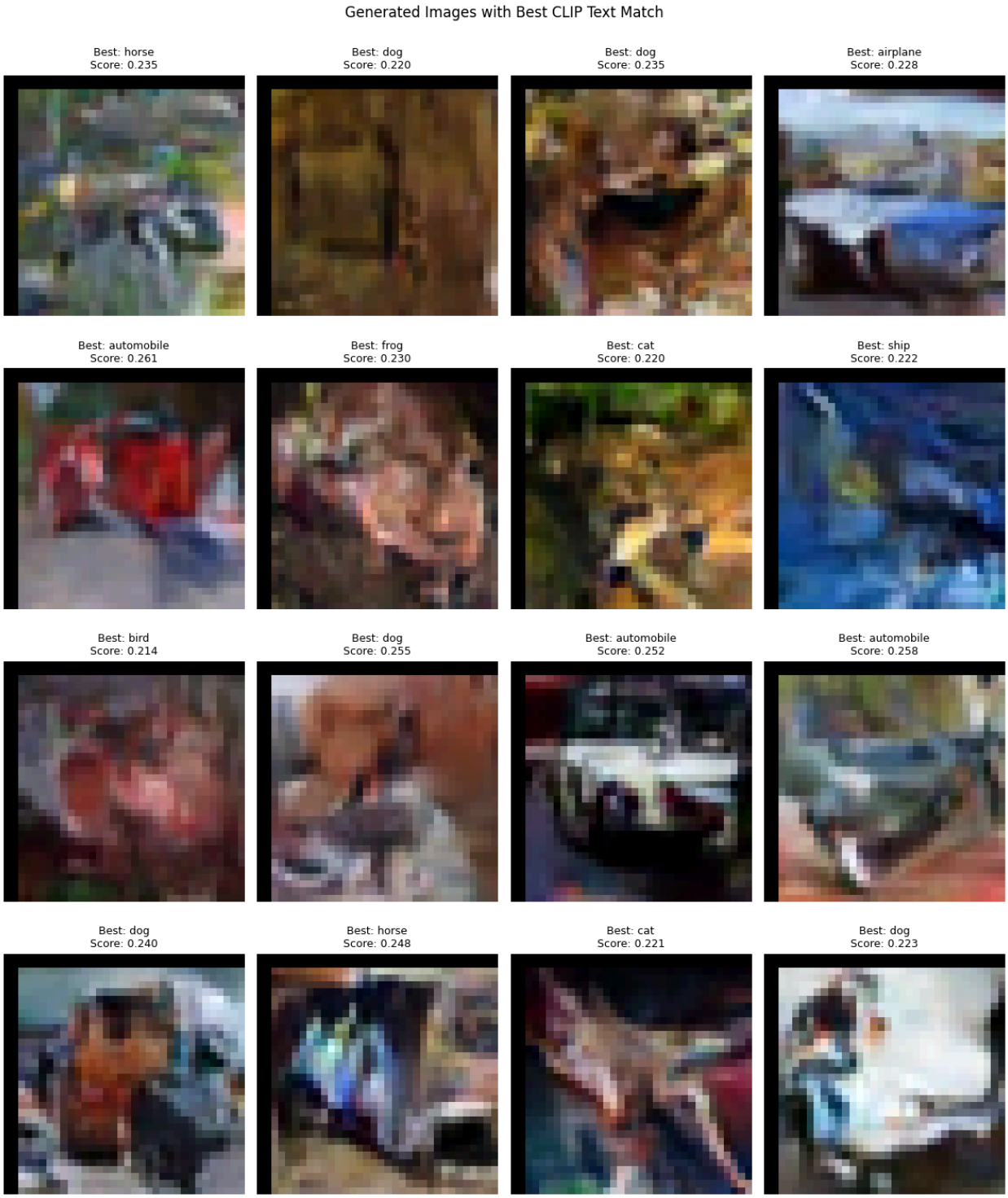
A clear improvement in generation quality was observed over the 50 epochs. Samples from early epochs (e.g., Epoch 5) appear very close to random noise. By Epochs 15-25, more defined color patches and blurry structures emerge. Towards the end of training (Epochs 40-50), the samples exhibit more complex, albeit still abstract, patterns and shapes, indicating the model progressively learned to reverse the diffusion process and capture some underlying structure of the data distribution. Recognizable objects, however, did not fully materialize within this training period.

CLIP Evaluation Analysis

Quantitative Results

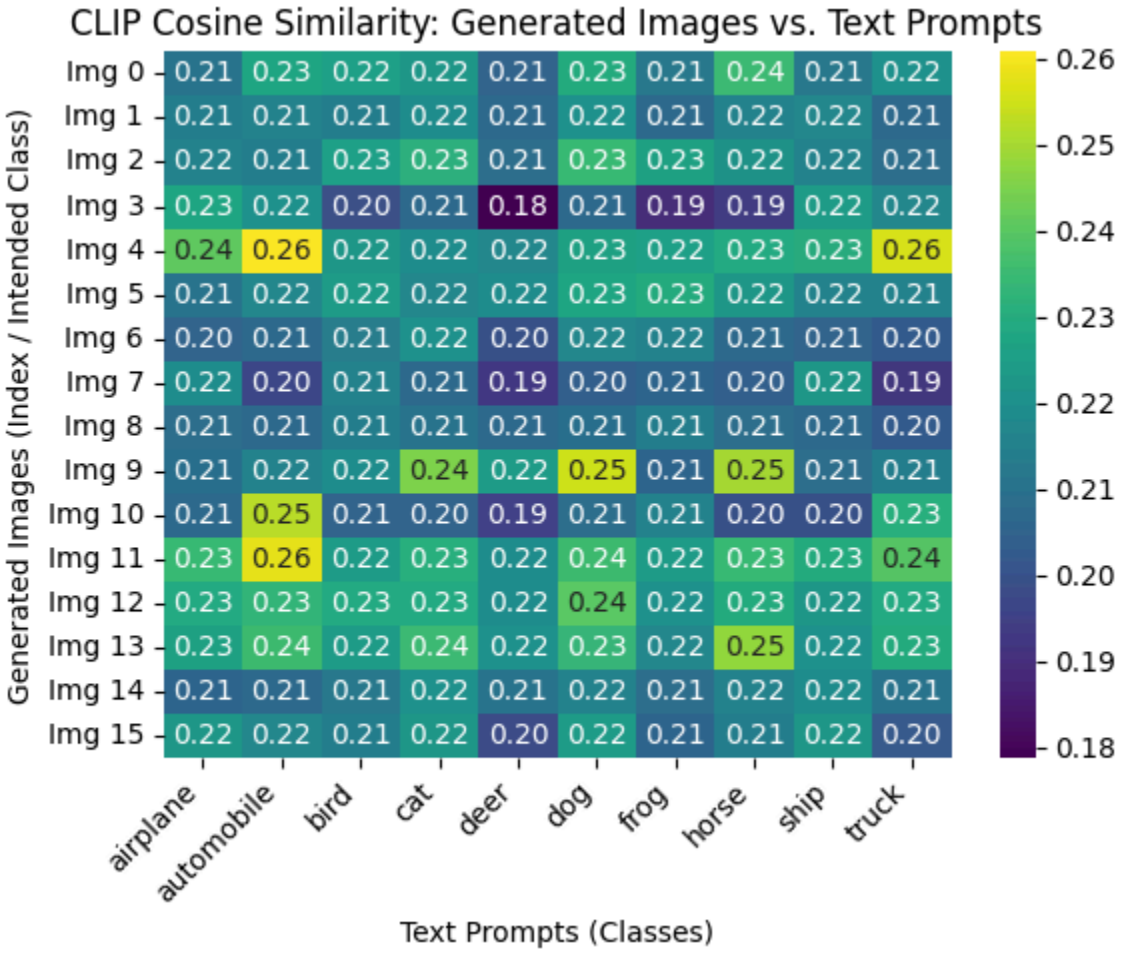
- The CLIP evaluation was performed on the sample grid generated at Epoch 50.
- **Average Top-1 Similarity Score: 0.2351**. This score represents the average cosine similarity between each generated image embedding and its best-matching text prompt embedding among the 10 CIFAR-10 classes. A score of 0.2351 is significantly higher than random chance (which would be around 0.1 for 10 classes) but still relatively low, indicating that while the images possess some semantic features detectable by CLIP, they do not strongly align with any single, specific class description. This quantitatively supports the visual observation that the images are abstract and not clearly classifiable.
- **Matching Accuracy & Intended Similarity**: These metrics could not be calculated as the evaluated samples were loaded from the saved grid file, which did not preserve the intended class label for each generated image. Future evaluations should generate samples specifically for this purpose to enable these accuracy calculations.

Qualitative Results



Visualizing the images alongside their best-matching CLIP label confirms the quantitative findings. For example, the first few images were best matched with labels like 'horse', 'dog', 'airplane', 'automobile', with scores around 0.22-0.26. Visually, these images do not strongly resemble the assigned labels, highlighting the abstract nature of the generations and the

limitations of the model after 50 epochs. The low scores reflect CLIP's assessment of this weak alignment.



The heatmap of the image-text similarity matrix shows relatively low scores across the board, without a strong diagonal pattern. This indicates that no generated image strongly matched its intended class prompt (if labels were known) more than other prompts, and generally, the similarity scores were low overall. There were no prominent off-diagonal bright spots suggesting consistent confusion between specific classes, further emphasizing the abstract and undifferentiated nature of the generated samples at this stage.

Assessment Questions

Understanding Diffusion

Forward Process: The forward diffusion process involves iteratively adding small amounts of Gaussian noise to an initial clean image x_0 over a series of discrete timesteps T. At each step t, the image x_t is generated based only on the previous image x_{t-1} by adding noise scaled by a predefined variance schedule β_t. This forms a Markov chain where $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1}$. Due to the properties of Gaussian noise, we can directly sample x_t from x_0 using the formula $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$, where $\bar{\alpha}_t$ is the product of $(1-\beta_t)$ up to step t. The visualization in Cell 16 clearly shows an image progressively degrading into near-isotropic noise as t approaches T.

Gradual Noise: Noise is added gradually (small β_t) so that the change between consecutive steps x_{t-1} and x_t is small. This makes the reverse process – predicting x_{t-1} from x_t – tractable to learn. If large noise steps were taken, the mapping would be too complex. Gradual addition ensures that significant structural information from the original image is preserved until later stages of the forward process, giving the model a smoother distribution transformation to learn in reverse.

Recognizability Point: Based on the sample progression visualization (Cell 20d), images started as noise (Epoch 5), evolved into blurry color fields (Epochs 15-25), and showed more defined abstract structures around Epochs 30-50. However, even at Epoch 50, clearly recognizable objects corresponding to CIFAR-10 classes were not consistently formed. Recognizability likely requires significantly more training beyond 50 epochs for this dataset and model configuration.

Model Architecture

U-Net Suitability: The U-Net architecture is well-suited because its encoder-decoder structure with skip connections naturally handles image-to-image tasks where input and output have the same spatial dimensions (like predicting noise of the same size as the input image). The encoder captures hierarchical features at different resolutions, while the decoder reconstructs the output, allowing the model to consider both contextual (deep features) and fine-grained (shallow features) information.

Skip Connections: Skip connections directly link feature maps from the encoder (downsampling path) to corresponding layers in the decoder (upsampling path). Their importance lies in allowing the decoder to reuse low-level feature information (like edges and textures) directly from the encoder, preventing this information from being lost in the bottleneck. This significantly

aids in reconstructing detailed outputs and improves gradient flow during training, preventing vanishing gradients in deep networks.

Class Conditioning: In this implementation (Cell 10), class conditioning works by first converting the integer class label into a dense vector embedding using an *nn.Embedding* layer. This class embedding is then element-wise added to the sinusoidal time embedding. The resulting combined embedding is processed through an MLP (*time_mlp*) and then injected into each residual *Block* of the U-Net. By adding this projected embedding to the intermediate feature maps within the U-Net, the model's noise prediction ϵ_θ becomes conditioned not only on the noisy input x_t and timestep t but also on the provided class label, guiding the generation towards images of the specified class.

Training Analysis

Loss Value Interpretation: The loss value represents the Mean Squared Error (MSE) between the random Gaussian noise (ϵ) that was actually added to create the noisy image x_t during the forward process, and the noise (ϵ_θ) predicted by the U-Net model given x_t, timestep t, and the class label. A lower loss indicates the model is becoming better at predicting the noise component for any given noise level and input, which is the core task required to reverse the diffusion process.

Image Quality Change: As documented earlier and visualized in Cell 20d, the generated image quality showed clear improvement over 50 epochs. It progressed from random noise patterns to blurry color blobs, and finally to more structured, albeit abstract, compositions with color distributions somewhat characteristic of the dataset. However, it did not reach the stage of generating recognizable objects.

Time Embedding Purpose: The time embedding informs the U-Net about the current noise level, represented by the timestep t (ranging from 0 to $T - 1$). The amount and characteristics of noise vary significantly across different timesteps. The model needs to know the specific noise level t of the input x_t to make an accurate prediction of the noise ϵ that was added. Without the time embedding, the model would not know whether to predict a small amount of noise (for low t) or a large amount (for high t), making the denoising task impossible.

CLIP Evaluation

CLIP Score Meaning: The CLIP scores quantify the semantic similarity between the generated images and text descriptions of the target classes. The low average top-1 similarity (0.2351) indicates that, according to CLIP's understanding of image-text relationships, the generated images after 50 epochs do not strongly resemble any specific CIFAR-10 class concept. This aligns with the visual assessment of the images as abstract.

Generation Difficulty Hypothesis: Based on visual complexity and potential overlap, it's hypothesized that classes with very distinct shapes and color palettes (e.g., 'airplane' often with blue sky, 'ship' often with blue water) might become distinguishable earlier in training than classes with high intra-class variation or visual similarity (e.g., 'cat' vs 'dog', different breeds/poses; 'automobile' vs 'truck'). The abstract nature of current generations makes it hard to confirm this yet.

Improving Generation with CLIP: CLIP scores could potentially improve generation via: (1) **CLIP Guidance**: During the sampling (reverse) process, the gradient of the CLIP similarity score (between the current noisy image x_t and a target text prompt) could be used to "steer" the denoising step towards images that better match the prompt. (2) **Re-ranking**: Generate multiple candidate images and use CLIP scores to select the one that best matches the desired text description. (3) **CLIP Loss (Advanced)**: Incorporate a CLIP-based loss term during training, encouraging the model to generate images that are not only reconstructed well (MSE loss) but also have high similarity to corresponding text descriptions.

Practical Applications

Real-world Applications: Diffusion models power state-of-the-art image generation (art, design assets), image editing (inpainting, outpainting, style transfer), data augmentation for training other ML models, scientific discovery (e.g., generating molecular structures), and potentially video generation.

Model Limitations: The current model, after 50 epochs, suffers from insufficient training time, resulting in low visual fidelity and abstract outputs. It requires significant computational resources (especially for higher resolution/complexity). Controllability beyond simple class conditioning is limited without further techniques. Like all generative models, it can potentially inherit and amplify biases present in the training data.

Proposed Improvements:

- **Extended Training & Hardware**: Train for significantly longer (e.g., 500+ epochs) and utilize more powerful hardware to accelerate convergence and allow the model to learn finer details.
- **Improved Noise Schedule**: Replace the linear schedule with a cosine schedule, which is often found to improve sample quality, particularly in later training stages, by managing the signal-to-noise ratio more effectively.
- **Increase Model Capacity**: Experiment with increasing the *base_dim* or adding more layers/blocks (*dim_mults*) in the U-Net architecture. A larger model might be necessary to capture the complexity of the CIFAR-10 distribution, although this would also increase training time and memory requirements.

Discussion and Future Work

This project successfully implemented the core components of a conditional Denoising Diffusion Probabilistic Model for the challenging CIFAR-10 dataset, adapting from a potentially simpler base structure. The training process over 50 epochs demonstrated successful learning, evidenced by consistently decreasing training and validation losses and a clear visual progression in generated samples from noise towards more structured, albeit abstract, forms.

The primary challenge encountered was the significant training time required for diffusion models to achieve high fidelity on complex datasets like CIFAR-10. While improvement was evident, 50 epochs proved insufficient for generating recognizable objects or achieving strong class conditioning. The CLIP evaluation quantitatively confirmed the limited semantic alignment of the generated images at this stage (Avg. Top-1 Similarity: 0.2351).

Future work should prioritize significantly extended training runs, ideally leveraging more powerful GPUs like the A100, to allow the model to converge further. Experimentation with alternative noise schedules (e.g., cosine) and potentially increasing model capacity could further enhance performance. Implementing quantitative CLIP evaluation with known intended labels (by generating samples specifically for evaluation) would provide more precise metrics like matching accuracy. Exploring techniques like classifier-free guidance could also improve sample quality and adherence to conditioning.

Conclusion

This midterm assignment provided valuable hands-on experience in implementing and training a diffusion model. A conditional U-Net architecture was successfully built and trained on the CIFAR-10 dataset, demonstrating an understanding of the underlying DDPM principles, including time and class conditioning. While the generated images after 50 epochs did not achieve photorealism, the observable progression in sample quality and the decreasing loss curves validate the correctness of the implementation and the learning process. The CLIP evaluation provided useful quantitative insights into the model's current capabilities. This project serves as a solid foundation, highlighting the potential of diffusion models while underscoring the computational requirements for achieving state-of-the-art results.