# SODA: Bottleneck Diffusion Models for Representation Learning

**Shivam Sharma (210983, sshivam21@iitk.ac.in)**

## 1 Model Objective

The model has a twofold objective-

- To get high quality, low dimensional embeddings of the images which can be used for downstream tasks like classification.
- To get a diffusion denoiser, conditioned on these embeddings for controlled image generation.

The paper also explores other objectives such as Novel View generation, by futher conditioning the denoiser of the view information, but I've focused on these two.

Project Page - `https://soda-diffusion.github.io/`(official implementation is not out yet)

## 2 Dataset and Training Environment

Reproducing the paper results on a larger scale dataset like **ImageNet** was not possible due to the limited GPU availablity, Instead I've used the **CIFAR10** dataset, which has about 60000, $32 \times 32$ images, classified into one of 10 classes.

All of the training and inference tests were performed on **gpu01.cc.iitk.ac.in** which has two **Nvidia A40, 48GB**, although I use only one GPU for training. With the current batch size, atleast 20GB of GPU memory is required for training.

## 3 Implementation Details

### 3.1 Encoder

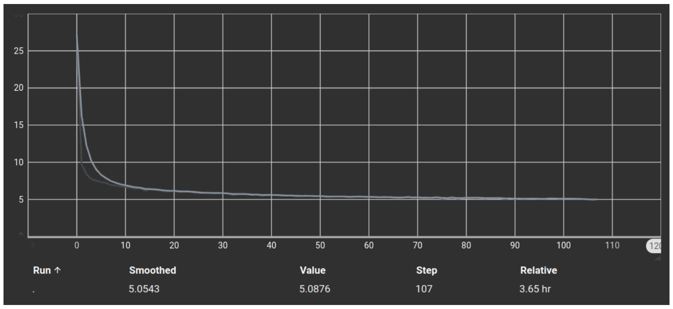
Pytorch's implementation of **resnet18** is used as the encoder, with the size of output embeddings set to 128. Also, the size of the first conv layer is changed to $3 \times 3$, due to the small image size.

### 3.2 Denoiser

The denoiser is a UNet architecture, with each block containing the **AdaGN** normalization layer, which is conditioned on the embeddings. The denoiser takes an image as input and outputs another image of the same dimension.

### 3.3 Training

We iterate over the dataset in a batch size of 256 and generate a random timestep for each image. The images are perturbed upto the timestep(using the direct formula) and are passed through the denoiser. The MSE loss between the predicted noise and the actual noise is minimized, using the AdamW optimizer with weight decay(L2) and betas as given in the paper. It takes atleast 50 epochs for the model to show some results. Note the the class labels are not used here.



| Run ↑ | Smoothed | Value | Step | Relative |
|---|---|---|---|---|
| . | 5.0543 | 5.0876 | 107 | 3.65 hr |

Figure 1: MSE loss in a batch over epochs

# 4 Results

## 4.1 Linear Probe Test

A way of testing the quality of the embeddings is to train a very small classifier network over the embeddings and check the accuracy over a test set. In this case the model consists of a single linear layer to output the logits. The model achieves **56 %** accuracy of the true class being the most probable class and **85 %** accuracy of the truth class being in the top 3 most probable classes.

## 4.2 Image Resconstruction

The following iamge illustrates the generation capabilities of the model, the first two rows are reconstrued by the model, conditioned on only the embeddings. The next two are originals.

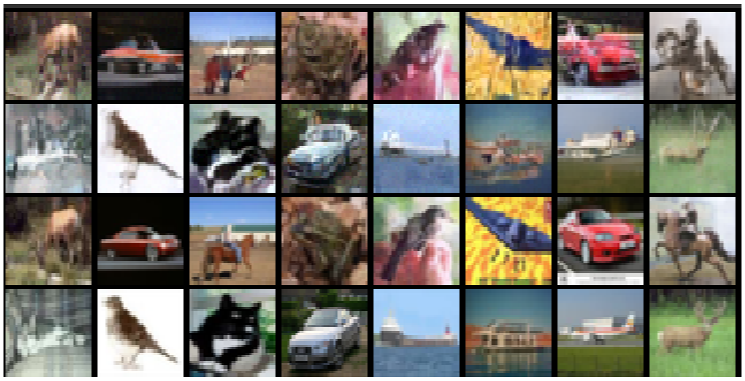


Figure 2: Image Generation