Part II Project Proposal - Denoising Diffusion Probabilistic Models for Image Inpainting

Pranav Talluri

October 2022

Contents

1 Introduction

Denoising Diffusion Probabilistic Models (DDPMs) are a new class of latent variable models, introduced in Sohl-Dickstein et al. 2015 that are inspired by Non-equilibrium Statistical Physics. These models have been shown to deliver very promising results for certain generative applications compared to Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAEs) and Normalising Flows. Subsequent literature improves on various aspects of the original models. A recent paper (Saharia et al. 2022) introduces a framework for conditional image synthesis using DDPMs and shows that they can produce state-of-the-art samples for certain applications. Image synthesis can be conditioned on various parameters. DALL-E and other similar image generators are conditioned on a text prompt. It is possible to condition on a class label to generate images, for example, that contain cars. In this project, I will consider image inpainting, in which a model realistically fills in user-selected masked regions of an image. For image inpainting, synthesis is conditioned on the unmasked region of the image.

2 Background

2.1 Image Synthesis using Deep Generative Models

Deep generative models automatically learn complex data distributions from training data. The models can then be used to generate new samples that could have plausibly been drawn from those distributions. These can be applied to synthesising images by learning from image training data. In recent years, such generation of images has hugely grown in popularity as the samples produced have become more realistic.

2.2 Diffusion Probabilistic Models

Probabilistic models will ideally be flexible enough to describe complex data but still tractable, (feasible to compute). However, these two objectives are usually conflicting. For example, a Gaussian distribution is tractable but not very flexible. On the other hand, we can define probabilistic models in terms of any non-negative function $\phi(x)$, yielding a distribution $p(x) = \frac{\phi(x)}{Z}$, where Z is a normalisation constant. However, computing the normalisation constant is generally intractable and the model typically requires an expensive Monte Carlo process to evaluate, train or draw samples from (Sohl-Dickstein et al. 2015).

DDPMs overcome this trade-off. They offer flexibility in modelling target distributions by using a Markov chain trained using variational inference to gradually convert a simple Gaussian into the target. They also offer exact sampling as, rather than using the Markov chain to approximate the target distribution, the endpoint of the chain is taken exactly as the target.

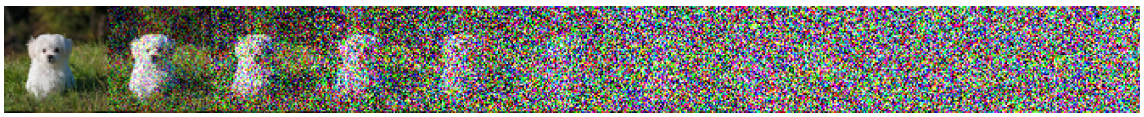

Figure 1: Illustration of a diffusion process (Sohl-Dickstein et al. 2015).

They are latent variable models with latents of the same dimensionality as the original data. DDPMs have a forward and reverse process. Figure 1 shows the forward process (left to right). Unlike other latent variable models, the forward process is fixed. It is a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule. The forward process variances can be learnt by reparameterisation or held as constant hyper-parameters. The reverse process is defined as a Markov chain with learned Gaussian transitions that reverse the addition of noise. It is possible for the reverse process to be learnt as, when the variances are small, both the forward and reverse processes have the same functional form. Training is performed by optimising the variational bound on negative log likelihood. This has been shown to be equivalent to maximising the Evidence Lower Bound (ELBO).

Since the work of Sohl-Dickstein et al. 2015, various papers have improved on DDPMs. Ho et al. 2020 identifies how reparameterising can lead to an equivalence with denoising score matching with Langevin dynamics. Nichol et al. 2021 identifies modifications that can allow DDPMs to achieve competitive log-likelihoods and also shows that learning variances of the reverse diffusion process

allows sampling with an order of magnitude fewer forward passes, witg a negligible difference in sample quality.

2.3 Image Inpainting

Image inpainting is a form of conditional image generation where, given an image with a user-selected masked portion, the region is filled in realistically. This has real-world applications such as removing undesired objects from an image.

Inpainting has existed with varying quality for over a decade. Early methods worked well on textured regions but struggled with generating semantically consistent structure. A notable early consumer-facing example is Adobe's Content-Aware Fill, in which a masked region of an image is filled based approximate nearest-neighbour matches using the PatchMatch algorithm. Another more recent example is Google's Magic Eraser, where the user can draw a mask on an image on their Pixel phone. The computation is performed on-device and within seconds.

Figure 2: Left: Image with mask. Centre: Sample from Palette. Right: Original image. (Saharia et al. 2022)

One way of performing image inpainting is using deep generative models to learn the conditional distribution of output images (complete images) given the input (masked image). The models should be able to capture multi-modal distributions in the high-dimensional space of images. GANs are used widely, but often require auxiliary objectives (e.g. on structure) and struggle with sample diversity (Saharia et al. 2022). Instead, inpainting can be implemented using DDPMs conditioned on the unmasked regions of the image. Specifically, the conditional diffusion models have the form $p(y|x)$ where x is a masked image and y is filled in. An example is shown in Figure 2.

3 Work to be Undertaken

3.1 Base

3.1.1 Task 1

I will implement DDPMs for unconditional image generation according to Ho et al. 2020. This initial implementation will be on smaller images than the paper uses (likely 32×32). This acts as a base for conditional image generation.
Starting Point: I plan on making this implementation from scratch. The paper does provide an official code implementation which I may refer to for comparison of results.

3.1.2 Task 2

I will extend **Task 1** with conditional image generation to perform image inpainting according to Saharia et al. 2022. Here, I will increase the image resolution used to match the literature.
Starting Point: There is no official implementation of this work. The sample outputs and image masks from the paper are available for comparison.

3.2 Extensions

3.2.1 Task 3

I will incorporate the improvements detailed in various literature into **Task 2** (including Nichol et al. 2021) and compare my results to Saharia et al. 2022.
Starting Point: Saharia et al. 2022 releases an official code implementation, but not for inpainting. This will be my own implementation from scratch.

3.2.2 Task 4

I will create a web-application that uses semantic segmentation and the DDPM developed in **Task 3** to allow a user to upload an image and mask semantic regions so the DDPM can fill it in.
Starting Point: There are many existing implementations of semantic segmentation, so I will focus on incorporating this into the work flow and engineering a web app rather than re-implementing it.

3.3 Challenges and Risks

Challenge One of the main concerns is the amount of time taken to train the models.

Mitigation I will use smaller images than the 2020 paper for unconditional image generation. I will be using CSD3 to speed up training. If this is still too slow, I will bring forward improvements from the 2021 paper that enable faster training times to the start of the timeline. I could also use smaller images throughout the project.

Risk Due to the training time of the models, corruption of the model can be very detrimental to the project timeline.

Mitigation Constant backups of every model and records of the results associated with each model. I will use a version control system (Git).

Risk Failure to obtain access to CSD3.

Mitigation There are other computing clusters available, which I will try to use instead. I will use smaller images throughout rather than trying to match the resolutions used in the literature.

4 Success Criteria

4.1 Evaluation Methodology

In the past few years, there have been an plethora of methods used to evaluate the generation of image samples by neural networks. I will use methods according to the literature I am basing my implementations on so that I am able to compare results.

For unconditional sample generation I plan on using Fréchet Inception Distance (FID), Inception Scores (IS) and Precision and Recall (Measured using features detected by Inception-V3). I will use the CIFAR and MNIST datasets.

For image inpainting, I will use FID, IS, Perceptual Distance (PD) and Classification Accuracy (CA). To measure sample diversity for image inpainting, I will produce multiple samples and then measure similarity between them using SSIM and LPIPS scores. I will use subsets of the ImageNet and Places2 datasets as described in Saharia et al. 2022.

4.2 Base Success Criteria

4.2.1 Task 1

1. Successfully implement DDPMs for unconditional image generation
2. Measure performance of model using FID, IS, Precision and Recall

4.2.2 Task 2

1. Successfully implement DDPMs for image generation conditioned on a masked image for image inpainting
2. Measure performance of model using FID, IS, CA and PD

4.3 Extension Success Criteria

4.3.1 Task 3

1. Successfully incorporate changes from literature to improve performance of conditional DDPM for image inpainting
2. Successfully incorporate changes from literature to improve training time of DDPM
3. Measure performance of model using FID, IS, CA and PD

4.3.2 Task 4

1. Successfully implement a semantic segmentation network that works with the chosen image resolution
2. Successfully implement pipeline made of segmentation network and DDPM
3. Successfully implement a web-app allowing users to use the pipeline to remove subjects from images and realistically replace them

5 Work Packages and Milestones

The table below outlines my planned timetable for the project. **Tx** refers to **Task x** from Section **3**.

Stage	Weeks	Dates	Work to Complete
Package 1	1 and 2	17/10 – 30/10	– **T1** Make detailed notes on DDPMs – **T1** Experiment with the network architectures used in the literature – **Write-up** Complete Introduction
Package 2	3 and 4	31/10 – 13/11	– **T1** Implement DDPMs for unconditional image generation (at smaller resolutions) – **Write-up** Start **T1** Implementation
Package 3	5 and 6	14/11 – 27/11	– **T1** Fine-tune code and parameters – **T1** Compare results with Ho et al. 2020 – **Write-up** Complete **T1** Implementation
Milestone 1			– Completed **T1** – Completed Introduction for **Write-up** – Completed **T1** Implementation in **Write-up**
Package 4	7 and 8	28/11 – 11/12	– **T2** Make detailed notes on conditional DDPMs for image inpainting – **T2** Start implementing conditional DDPMs for inpainting – **Write-up** Start **T2** Implementation – **Write-up** Start Preparation
Break	9 and 10	12/12 – 25/12	– Holiday and catch-up
Package 5	11 and 12	26/12 – 8/1	– **T2** Finish implementing conditional DDPMs for inpainting – **T2** Start fine tuning code and parameters – **Write-up** Continue **T2** Implementation – **Write-up** Continue Preparation
Package 6	13 and 14	9/1 – 22/1	– **T2** Finish fine tuning code and parameters – **T2** Compare results with Saharia et al. 2022 – **Write-up** Complete **T2** Implementation – **Write-up** Continue Preparation
Milestone 2			– Completed **T2** – Completed **T2** Implementation in **Write-up**

Package 7	15 and 16	23/1 – 5/2	– **T3** Make notes on the improvements presented in various literature, including Nichol et al. 2021 – **T3** Start incorporating the improvements into the work from **T2** – **Write-up** Start **T3** Implementation – **Write-up** Continue Preparation – **Write-up** Progress Report for 3/2
Package 8	17 and 18	6/2 – 19/2	– **T3** Finish incorporating the improvements into **T2** – **Write-up** Continue **T3** Implementation – **Write-up** Complete Preparation
Package 9	19 and 20	20/2 – 5/3	– **T3** Fine tune code and parameters – **T3** Compare with results from Saharia et al. 2022 – **Write-up** Complete **T3** Implementation – **Write-up** Start Evaluation
Milestone 3			– Completed **T3** – Completed **T3** Implementation in **Write-up** – Completed Preparation
Package 10	21 and 22	6/3 – 19/3	– **T4** Research and implement segmentation network for image size used by DDPM – **T4** Connect segmentation network with DDPM – **Write-up** Start **T4** Implementation – **Write-up** Continue Evaluation
Package 11	23 and 24	20/3 – 2/4	– **T4** Build server that runs the models – **Write-up** Continue Evaluation – **Write-up** Continue **T4** Implementation
Package 12	25 and 26	3/4 – 16/4	– **T4** Build web-app that connects to the server – **Write-up** Complete **T4** Implementation – **Write-up** Complete Evaluation
Milestone 4			– Completed **T4** – Completed **T4** Implementation in **Write-up** – Completed Evaluation
Package 13	27, 28, 29	17/4 – 12/5	– **Write-up** Complete Conclusion
Milestone 5			– Submission

6 Resource Declaration

- **Personal Laptop**: I will use my personal device for research, writing implementations and writing the write-up. My computer is a Microsoft Surface Book 3 with 16GB RAM and a NVIDIA GeForce GTX 1650 Max-Q. I accept full responsibility for this machine and I have made contingency plans to protect myself against hardware and/or software failure. I have a backup machine and will use a revision control system (Git) to which I will make regular backups.
- **CSD3**: In order to train my models, I will use the free tier of CSD3.
- **Google Colab**: I may also use Google Colab for training of models on smaller images.

References

Sohl-Dickstein, Jascha et al. (2015). *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. DOI: 10.48550/ARXIV.1503.03585. URL: https://arxiv.org/abs/1503.03585.

Ho, Jonathan et al. (2020). *Denoising Diffusion Probabilistic Models*. DOI: 10.48550/ARXIV.2006.11239. URL: https://arxiv.org/abs/2006.11239.

Nichol, Alex et al. (2021). *Improved Denoising Diffusion Probabilistic Models*. DOI: 10.48550/ARXIV.2102.09672. URL: https://arxiv.org/abs/2102.09672.

Saharia, Chitwan et al. (2022). *Palette: Image-to-Image Diffusion Models*. DOI: 10.1145/3528233.3530757. URL: https://doi.org/10.1145/3528233.3530757.