Tensor-Train Diffusion Models

M. Baddar , M. Eigel

April 25, 2024

1 Abstract

In this work, we explore the application of fixed, low-rank tensor-train to Denoising Probabilistic Diffusion Models. We show the parametric noise can be modeled using tensor-trains and basis functions. We will also provide details on how the model can be trained using Riemannian Optimization algorithm for fixed-rank tensor-trains. The main objective is to develop a more efficient DDPM with respect to memory and training-time.

2 Background

2.1 Denoising Diffusion Probabilisitce Models (DDPM)

DDPM models are one of the state-of-the-art generative models[**?**]. Given a random variable $\mathbf{x}_0 \in \mathbb{R}^D$ with density $q(\mathbf{x}_0)$. The task is to find a set of parameters $\boldsymbol{\theta}$ for the approximate density function $p(\mathbf{x}_0; \boldsymbol{\theta})$ such that the log-likelihood $\log(p(\mathbf{x}_0; \boldsymbol{\theta}))$ is maximized.

In DDPM, the training happens in two phases:

Forward Process In this phase, a set of intermediate latent variables are generated $\mathbf{x}_t \quad t = 0, 1, 2, .., T$, where $\mathbf{x}_t$is the input data and $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad \mathbf{0} \in \mathbb{R}^D, \mathbf{I} \in \mathbb{R}^{D \times D}$.

Given the following set of constants:

$$\begin{aligned}
&\beta_t, \quad \leq \beta_t \leq 1 \\
&\alpha_t = 1 - \beta_t \\
&\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i
\end{aligned} \tag{1}$$

The latent variable $\mathbf{x}_t$ has the following distribution:

$$
\begin{aligned}
q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right) \\
q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right) &= \prod_{t=1}^{T} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right)
\end{aligned} \tag{2}
$$

TBD : Add details for deriving the forward sampling equation: Which can be rewritten as

$$
q\left(\mathbf{x}_t \mid \mathbf{x}_0\right) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, \left(1-\bar{\alpha}_t\right)\mathbf{I}\right) \tag{3}
$$

Accordingly, in the forward phase for each time $t = 1, 2, \ldots T$, a corresponding $\mathbf{x}_t$ is generated by sampling from density in eq.(3).

Reverse Process This the process in which we try to train a model the reconstructs $\mathbf{x}_0^{'} \sim q$ f rom normal noise $\mathbf{x}_T$.

Add loss function derivation

The loss function for DDPM can be written as:

$$
L_{\text{simple}}(\theta) = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right] \tag{4}
$$

For $t = 1, 2, \ldots, T$.(See eq (14) in[**?**])

.

How the loss is calculated: For each batch of the optimization, sample t uniformly from $1, 2, \ldots T$, then compute the loss as in eq.(4).

2.2 Tensor-Trains

2.3 Tensor-Train optimization

3 Model

3.1 Architecture

3.2 Optimization

4 Experiments

4.1 Plan

1. Isolation experiments for parametric noise model

2. Use a ResNet, Unet for training such data (for validation)
3. Apply TT opt to isolated data coming from different distributions (the effect of distribution on tt opt)
4. The effect of rank and number of cores,
5. The effect of opt algorithm parameters
6. Drawing a loss landscape for the original DDPM and TT ones
7. Analyzing the convergence of each optimization method
8. Analyzing the number of computations with each method
9. Analyze the memory footprint for each

convergence analysis examples https://arxiv.org/abs/2208.05314 https://openreview.net/pdf?id=
https://www.igpm.rwth-aachen.de/Download/reports/pdf/IGPM423.pdf https://www.jstor.org/stable
https://youtu.be/4WDedaz$_TV4?si = ksqwnfZMYNF_U595https : //www.cis.upenn.edu/\ cis6100/Ri$
$Wirth - optim - Riemann.pdf$ **https://arxiv.org/abs/1712.09913**

4.2 Analysis of Gradient Descent Optimization with Neural Networks Model

4.3 Optimization with Gradient Descent for Tensor-Train Model

4.4 Optimization with Alternating Linear Scheme for Tensor-Train Model

4.5 Optimization with Riemannian Gradient Descent for Tensor-Train Model

Why ? OPTIMIZATION METHODS ON RIEMANNIAN MANIFOLDS AND THEIR APPLICATION TO SHAPE SPACE https://www.uni-muenster.de/AMM/num/wirth/fi
"Even if an embedding is known, one might hope that a Riemannian optimization method performs more efficiently since it exploits the underlying geometric structure of the manifold. For this purpose, various methods have been devised, from simple gradient descent on manifolds [25] to sophisticated trust region methods [5]."

4.6 Results