Implementing Variational Autoencoders Generative Modelling using Pyro vs Pytorch

Shreyas V - 2020B2A71350G
Pranav Srikanth - 2020B5A71865G
Nandish Chokshi - 2020B1A72031G
Anish Sreenivas - 2020B3A71464G

Overview

- Aim is to show improvement in reliability and ease of use for the developer for generation of images.
- Reliability- Reduces possibility of data type mismatches because we are using inference algorithms.
- Ease of use- Only the model and the guide specification is needed to run the optimizer (the objective function does not need to be specified as in the PyTorch implementation).

Variational Auto-Encoders (VAEs)

VAEs are a type of generative model that is used to learn the distribution of a given dataset. They work by training a neural network to learn a latent representation of the data, which is then used to generate new data samples.

VAEs are trained using a two-stage process:

- Encoding: The encoder takes a data sample as input and produces a latent representation of the data.
- Decoding: The decoder takes the latent representation as input and produces a new data sample.

The latent representation is a vector of numbers that captures the essential features of the data sample. The decoder learns to generate new data samples by inverting the encoding process.

Pytorch and Pyro

- PyTorch is known for its dynamic computation graph which is known as define-by-run paradigm, which allows for more flexibility and ease of use when building complex models.
- Pyro is a probabilistic programming library that is built on top of PyTorch. It is designed to facilitate the creation and evaluation of probabilistic models, enabling users to focus on the model itself rather than the algorithmic details of inference.
- Pyro scales to large data sets with little overhead compared to hand-written code.
- Pyro is agile and maintainable as it is implemented with a small core of powerful, composable abstractions.

Loss and Gradient Function

A loss function is a mathematical function that measures how well a machine learning model is performing. It is used to quantify the error between the model's predictions and the actual labels of the training data. The goal of training a machine learning model is to minimize the loss function.

The gradient of a loss function is a vector that points in the direction of the steepest increase in the loss. It is used to update the parameters of a machine learning model during training.

Binary Cross Entropy

Binary cross entropy (BCE) is a loss function used in machine learning and deep learning to measure the difference between predicted binary outcomes and actual binary labels. It is designed to penalize the model for incorrect predictions, and is typically used in classification tasks where the target variable can only take on two values, such as 0 or 1. The BCE loss function ranges from 0 to infinity, with a lower value indicating a better fit between the predicted and actual labels. A BCE value of 0 indicates a perfect fit, while a BCE value of infinity indicates a complete mismatch.

$$\text{Loss} = -\frac{1}{\text{output size}} \sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)$$

Pyro

Guide

- A guide is essentially a parameterized family of distributions which serves as an approximate posterior distribution in variational inference.
- The guide function specifies the variational family and learns the parameters that would make this family a good approximation to the true posterior.

JitTrace_ELBO

- ELBO (Evidence Lower BOund) is the objective function used for variational inference, where you try to maximize the evidence lower bound to get as close as possible to the true data distribution.
- The JIT(just in time) compilation is useful when you have a static computation graph because it can optimize the operations at runtime, thereby reducing overhead and speeding up computation

SVI (Stochastic Variational Inference)

- The svi object in Pyro takes in a model, a guide, and an optimization algorithm, and provides methods to step through the variational inference process.
- Each step of svi will take a step towards optimizing the parameters of the guide to make it a better approximation of the true posterior.

Adam optimiser

Adam Optimizer is an optimization algorithm that is commonly used in deep learning. It is an extension of the stochastic gradient descent (SGD) algorithm, and is designed to update the weights of a neural network during training in a more efficient and effective way.

Adam Optimizer is used in a wide variety of deep learning applications, including computer vision, natural language processing, and machine translation. It is also a popular choice for training variational autoencoders (VAEs), which are a type of generative model.

MNIST Dataset

https://d2hg8soec8ck9v.cloudfront.net/datasets/mnist/

THANK YOU