Vector Quantized-Variational Auto Encoders(VQ-VAEs)

*Final Project Report

Anish Antony
(SM21MTECH11003)
IIT HYDERABAD

Rahul Bingumalla
(SM21MTECH12008)
IIT HYDERABAD

Polamraju Aakarsh
(ES18BTECH11027)
IIT HYDERABAD

***Abstract*—An Auto Encoder is a Neural Network that learns to reproduce its input. where we learn the point estimates of the parameter. One of the weaknesses of GMMs is that we have to choose k, the number of clusters, and if we choose wrong our model doesn't perform well. A Variational Auto Encoder(VAE) is a type of neural network that learns to reproduce its input and also to alter, or explore variations on data we already have, and not just in a random way either, but in desired, specific direction. Variational inference is Bayesian extension of the EM algorithm where we learn the distribution of parameters rather learning point estimates of the parameter. we can generate new images just by sampling from the distribution**

I. Introduction

An auto encoder is an unsupervised learning approach that predicts its own input using a neural network. It's a strategy for locating latent spaces in data that are sophisticated non-linear functions. The main issue with autoencoders in terms of generation is that the latent space into which they transform their inputs and where their encoded vectors reside may not be continuous or allow for easy interpolation. Autoencoders can only reconstruct their inputs and cannot produce fresh examples that are realistic.

The problem of non-regularized latent space in autoencoders is addressed by the variational autoencoder, which offers generative capacity to the whole space. Latent vectors are produced by the AE encoder. Rather than output vectors in latent space, the VAE encoder produces parameters from a pre-defined distribution in latent space for each input. The VAE then applies a restriction to this latent distribution, forcing it to be a normal one.The latent space is regularised as a result of this constraint. Variational autoencoders are particularly fascinating since the samples they generate can be extraordinarily good. Generative models have been around for a long, but their quality hasn't always been great.

The Vector Quantised Variational AutoEncoder is an attempt to improve on VAEs by replacing the continuous latent vector space with a discrete latent vector space. Language is discrete by nature, and speech is also usually represented as a series of symbols. Language can frequently express images and videos succinctly. Hence most forms of data we see can be expressed effectively through discrete representations. Discrete representations are also well suited to complicated thinking, planning, and predictive learning. The VQ-VAE attempts to leverage this fact and improve upon the performance of the VAE.

II. Literature Review

A. VAE

The VAE is a probabilistic latent variable model that relates an observed variable vector x to a low-dimensional latent variable vector z by a conditional distribution. The VAE models the probability of a data point x by

$$p_\theta(x) = \int p_\theta(x|z)p_\lambda(z)dz \tag{1}$$

where $p_\lambda(z)$ is a prior of the latent variable vector, and $p_\theta(x|z)$is the conditional distribution of x given z, which is modeled by neural networks with parameter . For example, if x is binary, this distribution is modeled by a Bernoulli distribution $B(x|\mu_\theta(z))$, where μ(z) is neural networks with parameter θ and input z. These neural networks are called the decoder.

The log marginal likelihood ln $p_\theta(x)$ is bounded below by the evidence lower bound (ELBO), which is derived from Jensen's inequality, as follows:

$$\begin{aligned} lnp_\theta(x) &= ln\,\mathbb{E}_{q\ \phi}(z|x)\left[\frac{p_\theta(x|z)p_\lambda(z)}{q_\phi(z|x)}\right] \\ &\geq \mathbb{E}_{q\ \phi}(z|x)\left[\frac{p_\theta(x|z)p_\lambda(z)}{q_\phi(z|x)}\right] \\ &\equiv \mathcal{L}(x;\theta,\phi) \end{aligned} \tag{2}$$

where $\mathbb{E}[\cdot]$ represents the expectation, and $q_\phi(z|x)$ is the posterior of z given x, which are modeled by neural networks with parameter ϕ. $q_\phi(z|x)$ is usually modeled by a Gaussian distribution $\mathcal{N}(z|\mu_\phi(x),\sigma_2(x))$, where $\mu_\phi(x)$ are neural networks with parameter and input x. These neural networks are called the encoder. The ELBO (Eq. (2)) can be also written as

$$\mathcal{L}(x;\theta,\phi) = D_{KL}(q_\phi(z|x)||p_\lambda(z)) + E_{q\phi}(z|x)[lnp_\theta(x|z)] \tag{3}$$

where $D_{KL}(P||Q)$ is the Kullback Leibler (KL) divergence between P and Q. The second expectation term in Eq.(3) is called the reconstruction term, which is also known as the negative reconstruction error. The parameters of the encoder and decoder neural networks are optimized by maximizing the following expectation of the lower bound of the log marginal likelihood:

$$\max_{\theta,\phi} \int p_D(x)\mathcal{L}(x;\theta,\phi)dx \tag{4}$$

where $p_D(x)$ is the data distribution

it shows that minimizing reconstruction loss and KL loss is actually maximising the Bayesian posterior.

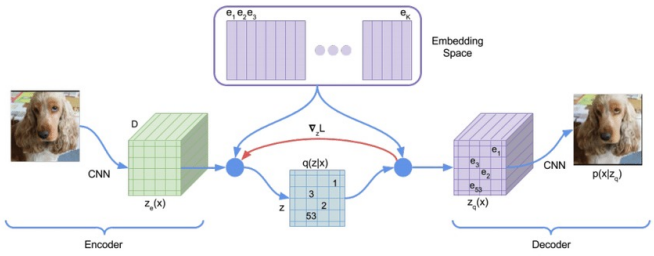
The encoder's output representation and the decoder's input are in the same D-dimensional space, and the gradients provide important information on how the encoder should adjust its output to reduce the reconstruction loss. Only the first loss term is optimised by the decoder, the first and final loss terms are optimised by the encoder, and the intermediate loss term is optimised by the embeddings. Because we assume a uniform prior for z, the KL component in the ELBO is constant with respect to the encoder parameters and may thus be ignored for training.

B. Vector Quantization

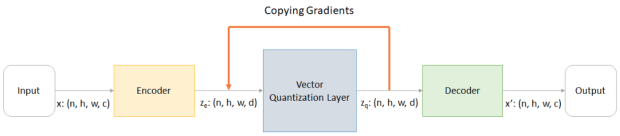
Vector quantization (VQ) is an efficient coding technique to quantize signal vectors.it is a classical quantization technique from signal processing that allows the modeling of probability density functions by the distribution of prototype vectors. In VQ, the input image data is first decomposed into k-dimensional input vectors.

III. VQ-VAE

A Vector Quantised Variational autoencoder follows the same basic concept as the variational auto-encoders(VAE). It uses discrete latent embeddings as each dimension of the latent vector is a discrete integer instead of the continuous normal distribution used by VAE while encoding the inputs. These integral values are used to index a dictionary of embeddings which are then passed on to the decoder.



VQ-VAE consists of an encoder, ze, an embedding(or a codeBook) and a decoder, zq. When an image is passed as input, it is converted into latent vectors using the encoder network. The embedding space consists of many latent vectors, which are compared to that of the input one. The distances are calculated and the most similar(least distance) latent vector(in the embedding space) to the input's latent vector is selected. The selected one is fed into the decoder network which reconstructs the image.



The above figure shows various top level components in the architecture along with dimensions at each step, where n : batch size h: image height w: image width c: number of channels in the input image d: number of channels in the hidden state

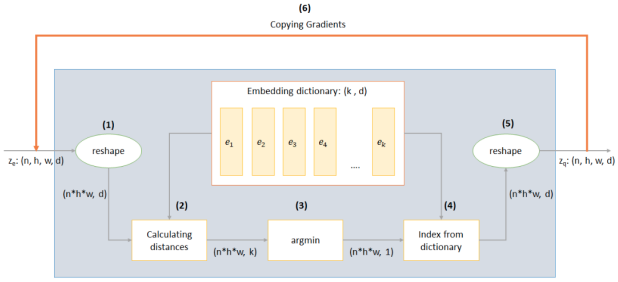
- Encoder takes in images x: (n, h, w, c) and give outputs z_e: (n, h, w, d)
- Vector Quantization layer takes z_e and select embeddings from a dictionary based on distance and outputs z_q
- Decoder consumes z_q and outputs x' trying to recreate input x

A. Loss Function

The total loss is actually composed of three components:

1) Reconstruction loss: which optimizes the decoder and encoder
2) Codebook loss: due to the fact that gradients bypass the embedding, we used dictionary learning algorithm which uses an l2 error to move the embedding vectors e_i towards the encoder output
3) Commitment loss: since the volume of the embedding space is dimensionless, it can grow arbirtarily if the embeddings e_i do not train as fast as the encoder parameters, and thus we add a commitment loss to make sure that the encoder commits to an embedding

B. Vector Quantization layer



We implement vector quantization by using a Vector Quantization layer, the working of which can be explained in six steps:

- Reshape: all dimensions except the last one are combined into one so that we have n*h*w vectors each of dimensionality,d.

- Calculating distances: for each of the n*h*w vectors we calculate distance from each of k vectors of the embedding dictionary to obtain a matrix of shape (n*h*w, k)
- Argmin: for each of the n*h*w vectors we find the the index of closest of the k vectors from dictionary
- Index from dictionary: index the closest vector from the dictionary for each of n*h*w vectors
- Reshape: convert back to shape (n, h, w, d)
- Copying gradients: If you followed up till now you'd realize that it's not possible to train this architecture through back propagation as the gradient won't flow through argmin. Hence we try to approximate by copying the gradients from z_q back to z_e. In this way we're not actually minimizing the loss function but are still able to pass some information back for training.

IV. Architecture

We start by defining a latent embedding space of dimension [K, D] where K are the number of embeddings and D is the dimensionality of each latent embeddng vector, i.e. $e_i \in \mathbb{R}^D$. The model is comprised of an encoder and a decoder. The encoder will map the input to a sequence of discrete latent variables, whereas the decoder will try to reconstruct the input from these latent sequences.

The model will take in batches of RGB images and pass it through a ConvNet encoder producing some output, where we make sure the channels are the same as the dimensionality of the latent embedding vectors. To calculate the discrete latent variable we find the nearest embedding vector and output it's index. The input to the decoder is the embedding vector corresponding to the index which is passed through the decoder to produce the reconstructed image.

V. Code Link

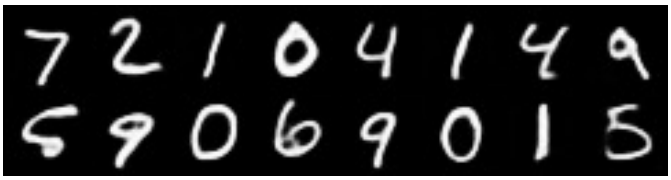
https://colab.research.google.com/drive/1THlweFu021Jo-WgALiqWJqxOoDuSXltk?usp=sharing

VI. Results

Hyperparameters:

hidden size=40
k=512
batch size=64
num. of epochs=10
learning rate=2e-4
beta=1.0

The generated images after the first and last EPOCH is shown below



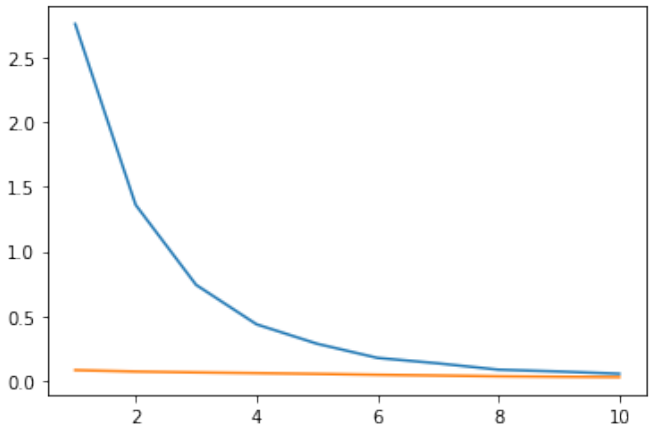


Fig: Training loss Vs Number Of Epochs

VII. Summary

Latent spaces are compressed representations of data that frequently emphasise the most important and semantically interesting features of the data. Many downstream algorithms can benefit from a learned latent representation. Autoencoders are a technique for locating latent spaces, which are complex non-linear data functions. Variational autoencoders augment the autoencoder latent space with a prior. Thus we can smoothly interpolate the data distribution through the latents. Variational autoencoders (VAEs) are autoencoders that tackle the problem of the latent space irregularity by making the encoder return a distribution over the latent space instead of a single point and by adding in the loss function a regularisation term over that returned distribution in order to ensure a better organisation of the latent space

The fundamental difference between a VAE and a VQ-VAE is that VAE learns a continuous latent representation, whereas VQ-VAE learns a discrete latent representation.

VQ-VAE extends the standard autoencoder by adding a discrete codebook component to the network. The codebook is basically a list of vectors associated with a corresponding index. It is used to quantize the bottleneck of the autoencoder; the output of the encoder network is compared to all the vectors in the codebook, and the codebook vector closest in euclidean distance is fed to the decoder.The decoder is then tasked with reconstructing the input from this quantized vector as in the standard autoencoder formulation that VAEs enforce a pre-defined prior on the latent space p(z), and the

encoder is tasked with approximating the posterior distribution of the latents p(z|x). Lastly, the decoder approximates the reconstruction from the latent space p(x|z).

REFERENCES

[1] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu, "Neural Discrete Representation Learning,"arXiv:1711.00937v2 [cs.LG] 30 May 2018.

[2] Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, Satoshi Yagi, "Variational Autoencoder with Implicit Optimal Priors," The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19).