# Variational Autoencoders

Backbone of generative AI

Why Variational Autoencoders?

Autoencoder principals – compress to lower space (latent space)

One idea for generating images is to just sample points in this latent space and run it through the decoder – the issue is that it doesn’t result in good or meaningful images

Why not just look at nearby points in the latent space for an encoded activation? – Its poorly structured and nearby points don’t necessary represent the same image

We would like an organised latent space where sampling a point leads to a new image

Introduction to Bayesian Statistics

Random var X (0 and 10), sampling X (aka between 0 and 10).

PDF of X p(X), expectation (average/mean).

P(x) = data distribution

P(z) = Latent distribution

P(z|x) – Gives probability that a latent vector z was generated by a particular image X

P(x|z) – given a latent Z, tells us the probability of reconstructing an image X from it

Idea – sample latent vectors from the posterior distribution P(z|x) those latents are likely to have been generated by images from our original data distribution P(x). if we can reconstruct these latent vectors back into images, we’ll effectively generate new samples from our original data distribution.

Assume latent distribution is a normal distribution

Still missing P(Z|X). As we don’t know the P(Z|X) we approximate it using a gaussian distribution which we’ll call Q(Z|X), this has paramteres mu and sigma which we need to learn, which is an optimisation process known as variational bayes.

Train a deep encoder to estimate paramters mu and sigma from the images.

Use a decoder to reconstruct images from the latent variables that are sampled from the approximate posterior

Trained via a data consistency term and a regularisation term

Data consistency measures how well it can reconstruct an image (X) from the encoded version Z (Mean square error).

KL divergence – distance between two probability distributions

Measures how close our approximate posterior is to the prior distribution p(z|x), since p(z|x) is a normal distribution, this means we optimise the ELBO we constrain the approximate posterior to take the shape of the normal distribution as well.

ELBO is the training objective of our variational autoencoder. It is designed to ensure that the generated samples come from our original data distribution P(x|z) – it can be thought of as a regularized reconstruction loss. With the first loss representing the usual L2 loss (MSE) and the second term imposing a normal distribution shape on our latent space.

How to we do this in practice?

Instead of mapping the image to a single point in the latent space, the encoder converts the input into a probability distribution that we chose to be a Gaussian. Namely, mean (mu) and sigma (std). Now instead of representing the image as a single point in the latent space, its represented as a gaussian distribution.

From this latent distribution we sample points at random and the decoder converts the different parts of the elbo loss and backpropagate it through the network.

How can we back propagate through this process? We cannot – this is where reparameterization trick comes in. Instead of sampling directly from the Gaussian distribution, we introduce a random variable (Epsilon) to handle randomness.

First we sample a random point from a normal distribution – variable epsilon, scale it by the variance of our approximated posterior distribution and shift it by its mean. As if we had sampled directly from the posterior distribution, except that by using the reparameterization trick the process becomes differentiable with respect to the mean and variance. By doing this the process becomes differentiable with respect to the mean and variance. This allows us to train the VAE end-to-end using standard gradient based optimisation techniques like ADAM.