Notes From Papers I've Read

Conditional Image Generation with PixelCNN Decoders

This paper introduces a model which can **generate images** from a label. Unlike a GAN, it can give a probability of how likely the image is to be accurate. This model can be conditioned on prior information.

This model is based on the PixelRNN model. There is also a PixelCNN model, which is faster to train because conv nets are better at parallelization, but it is less effective. Because the image datasets are so massive, we want to parallelize as much as possible while maintaining the RNN's performance. We use a gated variant of PixelCNN, *gated PixelCNN*.

We then introduce a conditional variant of gated PixelCNN, *Conditional PixelCNN*.

PixelCNNs model the joint probability distribution of pixels over an image $\mathbf{x}$ as

$$p(x) = \prod_{i=1}^{n^2} p(x_i | x_1, \cdots, x_{i-1}),$$

where x_i is a single pixel

As you can see, each pixel only depends on the previously-generated pixels, built on a raster-scan left-to-right, up-to-down order.

Neural Discrete Representation Learning

I do a sentence-by-sentence break-down of section 3.1.

> We define a latent embedding space $e \in \mathbb{R}^{D \times K}$ where K is the size of the discrete latent space (i.e., a K-way categorical), and D is the dimensionality of each latent embedding vector e_i.

A *latent embedding space* is a subset of a vector space in which latent representations (e.g. word vectors) are found, or *embedded*. Here, the entire latent space is a matrix e of size $D \times K$, and the matrix is broken into K vectors, each having D components. Thus, each embedded vector is a vector in e, so there are only a finite number of different embeddings, K.

> Note that there are K embedding vectors $e_i \in \mathbb{R}^D, i = 1, 2, \ldots, K$.

Each vector in the matrix e is a D-dimensional real vector, and it need not be one-hot. An example matrix e is

$$\begin{bmatrix} 1 & 1.2 & 3 \\ 3.2 & 23 & -1 \\ 1.32 & 0 & 1.352 \end{bmatrix}.$$

As shown in Figure 1, the model takes an input x, that is passed through an encoder producing output $z_e(x)$.

The output, $z_e(x)$, is a D-dimensional vector over the reals. An example is

$$z_e(x) = \begin{bmatrix} 2 \\ 4.2 \\ -9 \end{bmatrix}.$$

The discrete latent variables z are then calculated by a nearest neighbour look-up using the shared embedding space e as shown in equation 1. Equation 1 is:

$$q(z = k|x) = \begin{cases} 1 & k = \underset{j}{\operatorname{argmin}} ||z_e(x) - e_j|| \\ 0 & \text{else} \end{cases}.$$

Know that z is an integer in $\{1, 2, \ldots, K\}$, and that the z^{th} column in e is the one closest to $z_e(x)$.

The input to the decoder is the corresponding embedding vector e_k as given in equation 2.

Equation 2 is:

$$z_q(x) = e_k$$

where $k = \underset{j}{\operatorname{argmin}} ||z_e - e_j||$.

One can see this forward computation pipeline as a regular autoencoder with a particular non-linearity that maps the latents to 1-of-K embedding vectors.

A normal autoencoder has an information bottleneck. This particular model is like an autoencoder whose information bottleneck maps an arbitrary input to one of K vectors.

The complete set of parameters for the model is the union of the parameters of the encoder, decoder, and the embedding space e.

Trivial.

For sake of simplicity we use a single random variable z to represent the discrete latent variables in this section. However, for speech, images, and videos, we actually extract a 1D, 2D and 3D latent feature space, respectively.

This is just too complicated for me to understand.

The posterior categorical distribution $q(z|x)$ probabilities are defined as one-hot by equation 1.

The prior is the raw data, $p(x|z)$, and the encoder gives a sample from the posterior distribution, $q(z|x)$. The posterior is a categorical distribution because it selects one of K vectors, so it puts each x into a category defined by z. The distribution is one-hot because the value at e_k is 1 and the value for the other e_j is 0.

> We view this model as a VAE in which we can bound $\log p(x)$ with the ELBO.

A variational autoencoder is an autoencoder whose latent space is a multivariate Gaussian distribution. The ELBO sets a lower bound on the quality of the reconstruction $p(x)$, so we maximize it to improve the reconstruction.

> Our proposal distribution $q(z = k|x)$ is deterministic.

The distribution $q(z = k|x)$ is deterministic because it is generated by a VAE encoder.

> By defining a simple uniform prior over z, we obtain a KL divergence constant and equal to $\log K$.

The KL divergence $D_{\mathrm{KL}}(P||Q)$ is defined by

$$D_{\mathrm{KL}}(P||Q) = -\sum_{x \in \mathcal{X}} P(x) \log \frac{Q(x)}{P(x)}.$$

We see that, since $P(k) = 1$ and $P(j) = 0$ for $j \neq k$, the KL divegence becomes