Variational Autoencoders
We understand AE. What is Variational AE?

- Basically a AE, but a generative model
  - The encoder parameterizes a distribution and not just a point estimate
    - Hence, probabilistic non-linear dimensionality reduction
  - Decoder takes a latent variable, then generates an output

- Main hypothesis
  - There is a latent variable $z$ which can be used to generate our data
  - This variable explains the main sources of variation in our data
  - We can incorporate a prior for this $z$ as a regularization objective

- What is variational? we will cover this soon.
Hidden representation = Latent variable?
Hidden representation = Latent variable.

- Latent variables
- Hidden representation - a representation that captures the lower dimensional signals that **explain** the data best
  - Data can be explained by a couple of factors of variation
    - E.g. in images - cats/dogs vs book/pen → what if we had a hidden neuron that encoded “has a leg”
    - Or in MNIST → a hidden neuron that encodes “has a loop”
  - Main idea in PCA - project it to a space where only a couple of dimensions explain most of the data
Recap of Bayes theorem

Prior $P(z)$: How likely is a certain value of the latent variable $z$?

Likelihood $P(X|z)$: Given a certain value of $z$, how likely is a data point? (e.g. image)

Posterior $P(z|X)$: Given a datapoint, what is the probability of a latent variable?

Bayes theorem: $P(z|X) = \frac{P(X|z)P(z)}{P(X)}$

https://blog.evjang.com/2016/08/variational-bayes.html
Bayes

Recap of Bayes theorem

Prior $P(z)$: Let’s choose this to be a simple distribution - standard Gaussian.

Likelihood $P(X|z)$: Let’s call this the decoder part of our network. We can easily parameterize this.

Posterior $P(z|X)$: We try to do this in the encoder part of our network.

https://blog.evjang.com/2016/08/variational-bayes.html
Bayes

Bayes theorem:

$$P(z|X) = \frac{P(X|z)P(z)}{P(X)}$$

$$P(z|X) = \frac{P(X|z)P(z)}{\int_{z} P(X|z)P(z) \, dz}$$

All values of $z \rightarrow$ All real values!

- How is it even possible to compute this integral/sum? It’s not, it's intractable
Variational Bayes

We want another distribution to approximate the posterior - which is hard to compute. $Q(Z)$ to approximate $P(Z|X)$

How can we know this distribution is a good approximation?

KL-divergence: $\text{KL}(Q(Z) \| P(Z|X)) = \sum_{z \in Z} Q(z|x) \log \frac{Q(z|x)}{P(z|x)}$
What is KL divergence

A way to measure dissimilarity between two probability distributions, based on information theory.

Consider entropy: 

\[ H(x) = - \sum_x P(x) \log P(x) \]

KL divergence - many, many interpretations

- Number of bits of information lost if we use distribution \( q \) to represent \( p \)
- Negative log likelihood that samples generated by distribution \( p \) has been generated by \( q \)

Interesting reading [here](#).

**Note:** KL divergence is always positive!
KL-divergence of variational distribution $Q$

$$KL(Q_{\phi}(Z)\|P(Z|X)) = \sum_{z \in Z} q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p(z|x)}$$

Forward KL

This decomposes through a lot of math (which you can look up here) to

$$\log P(X) - \left( \mathbb{E}_Q \left[ \log P(x|z) \right] - KL(Q(Z|X)||P(Z)) \right)$$

- Log Likelihood of data (we want to maximize this!)
  - maximizing $P(x) \Rightarrow$ having some parameterization of the probability distribution our data that best explains our data
- This is called the Variational Lower Bound $\mathcal{L}$
Maximizing the Variational Lower Bound?

Simple rearranging from before, we get

\[ P(x) = \mathcal{L} + KL(Q||P) \]

KL-divergence is always \( > 0 \) - its like an error term.

Hence \( \mathcal{L} \) is the **lower bound**. So... the more you increase \( \mathcal{L} \), the better you describe your data, because \( P(x) \) increases.

Also, when is this bound **tight**? When \( Q(z) = P(z|x) \) and the KL-divergence term is 0 (or close to 0). Thus, the better your \( Q \), the better you are able to train your model.
Decomposing the variational lower bound

\[ \mathbb{E}_Q \left[ \log P(x|z) \right] - KL(Q(Z|X) \parallel P(Z)) \]

(negative) Reconstruction Error       Regularization

- **Reconstruction Error:**
  - In traditional autoencoders we use L2 loss
  - With binary data, here, it turns out to Binary Cross Entropy loss
  - What about real valued data? We can say \( P(x | z) \) is a Gaussian distribution.

- **Regularization term:**
  - You don’t want the latent variable values to be all over the place
  - Basically, without this term, this is as good as a normal AE.
  - Centered around 0 is a good place to start
    - However, this does not always work well. What if the latent structure of the data was actually a circle (or hypersphere in higher dimensions)?
    - We need to consider other priors
EM … looks familiar?
EM … looks familiar?
EM … looks familiar?

$KL(q||p)$

$\mathcal{L}(q, \theta^{\text{new}})$

$\ln p(X|\theta^{\text{new}})$
How do we actually train it?

We want to maximize the variational lower bound. 

\[ \mathbb{E}_Q \left[ \log P(x|z) \right] - KL(Q(Z)\|P(Z)) \]

Which means the negative of this is our loss function.

\( KL(Q(Z)\|P(Z)) \) is just a deterministic function (Gaussian Q, Gaussian prior for P)

\[ \mathbb{E}_Q \left[ \log P(x|z) \right] \] can be computed empirically. That is, sample a bunch of z values, calculate \( \log P(x|z) \) using those sampled z, and take an average. In the paper, the authors just take 1 sample.

But how is sampling differentiable?
Reparameterization trick

Sampling from a distribution $\mathcal{N}(\mu, \sigma^2)$ itself is not a differentiable operation.

But, due to the nice properties of Gaussian distributions, we can first sample $\epsilon \sim \mathcal{N}(0, 1)$

Then we can transform it to the $z$ that we need using this $z = g(\epsilon) = \mu + \epsilon \circ \sigma$

Then, this $\mathbb{E}_{z \sim \mathcal{N}(\mu, \sigma^2)}[f(z)]$ and $\mathbb{E}_{\epsilon \sim \mathcal{N}(0, 1)}[f(g(\epsilon))]$ are equivalent.

Addition, element-wise multiplication are all differentiable operations - and epsilon can be sampled agnostic of the actual mean and variance. Then epsilons are just ‘inputs’ to a network.

The reparameterization trick is helpful in many other places too.
VAE, the full picture

$$z = \mu + \sigma \odot \epsilon$$
$$\epsilon \sim \mathcal{N}(0, I)$$
Effect of prior

The KL Divergence term is a strong prior for latent variable values which says the latent values lie in a hypersphere.

What if this is not true? For example, what if we generated samples using latent variables that are distributed like a disc?

How do we force latent representations to be more meaningful?

An interesting approach is to *disentangle* latent variables.

Disentangling - each latent variable should capture different information from another.

Think of PCA - each projection is orthogonal to others.

**β-VAE**

- Main idea is to give an even stronger prior
  - This forces the posterior \( Q(z|x) \) to be highly factorized (diagonal covariance)
  - Diagonal covariance \( \rightarrow \) latent dimensions are uncorrelated
Main concepts to remember

- VAEs consist of two tractable distributions (should be easy to compute)
  - P(z) prior of latent variable
  - P(x|z) likelihood given latent variable
- We can train VAEs using back-prop because of reparameterization trick
- VAEs are very very similar to EM.
- We can make VAEs better by obtaining tighter definitions of the lower bound
  - Importance Weighted Autoencoders are an improvement.
- Choosing a good prior matters
  - Also, weighting prior differently gives different disentaglement results.