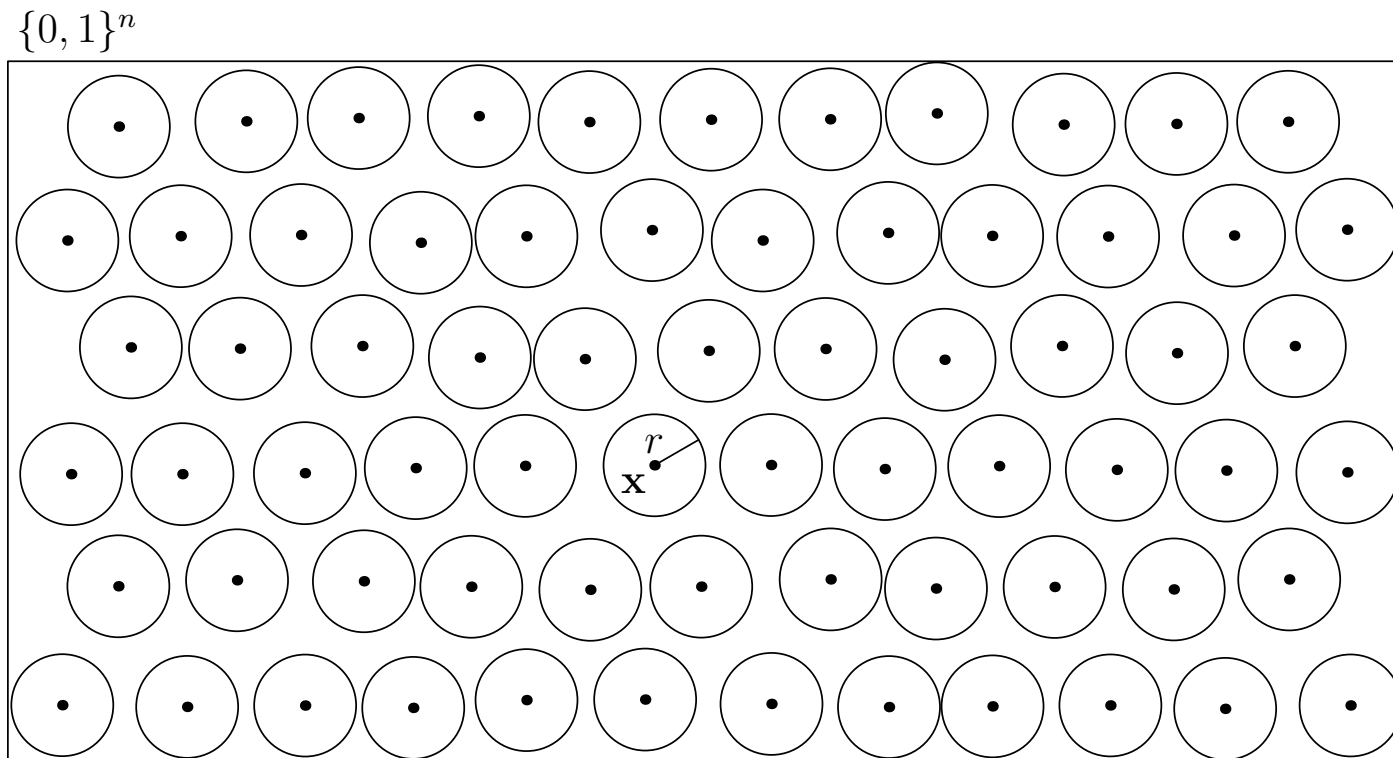


Advanced Machine Learning

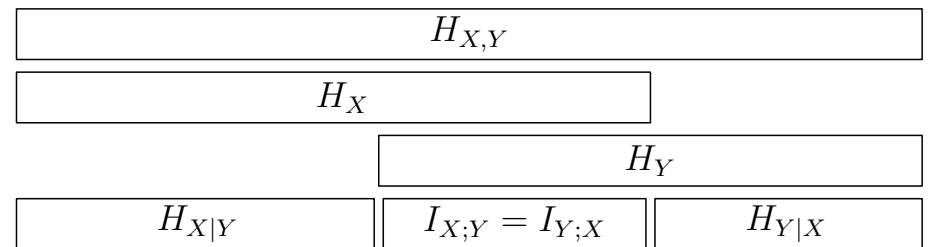
Information Theory



Information, KL-divergence, Minimum Description Length

Outline

1. **Information Theory**
2. KL-Divergence
3. Minimum Description Length
4. Variational Auto-Encoders



Communicating Via a Noisy Channel

- Information theory considers communicating down a (noisy) channel

$$X \sim \mathbb{P}(X) \xrightarrow{\text{noisy channel}} Y \sim \mathbb{P}(Y | X)$$

- We send a message X (with probability $\mathbb{P}(X)$) and receive a message Y with probability $\mathbb{P}(Y | X)$
- The uncertainty of the message sent, given we received a message y is

$$H_{X|Y=y} = - \sum_{x \in \mathcal{X}} \mathbb{P}(X = x | Y = y) \log(\mathbb{P}(X = x | Y = y))$$

- The expected uncertainty in the message sent is

$$H_{X|Y} = \sum_{y \in \mathcal{Y}} \mathbb{P}(Y = y) H_{X|Y=y} = - \sum_{x,y} \mathbb{P}(X = x, Y = y) \log(\mathbb{P}(X = x | Y = y))$$

Communicating Via a Noisy Channel

- Information theory considers communicating down a (noisy) channel

$$X \sim \mathbb{P}(X) \xrightarrow{\text{noisy channel}} Y \sim \mathbb{P}(Y | X)$$

- We send a message X (with probability $\mathbb{P}(X)$) and receive a message Y with probability $\mathbb{P}(Y | X)$
- The uncertainty of the message sent, given we received a message y is

$$H_{X|Y=y} = - \sum_{x \in \mathcal{X}} \mathbb{P}(X = x | Y = y) \log(\mathbb{P}(X = x | Y = y))$$

- The expected uncertainty in the message sent is

$$H_{X|Y} = \sum_{y \in \mathcal{Y}} \mathbb{P}(Y = y) H_{X|Y=y} = - \sum_{x,y} \mathbb{P}(X = x, Y = y) \log(\mathbb{P}(X = x | Y = y))$$

Communicating Via a Noisy Channel

- Information theory considers communicating down a (noisy) channel

$$X \sim \mathbb{P}(X) \xrightarrow{\text{noisy channel}} Y \sim \mathbb{P}(Y | X)$$

- We send a message X (with probability $\mathbb{P}(X)$) and receive a message Y with probability $\mathbb{P}(Y | X)$
- The uncertainty of the message sent, given we received a message y is

$$H_{X|Y=y} = - \sum_{x \in \mathcal{X}} \mathbb{P}(X = x | Y = y) \log(\mathbb{P}(X = x | Y = y))$$

- The expected uncertainty in the message sent is

$$H_{X|Y} = \sum_{y \in \mathcal{Y}} \mathbb{P}(Y = y) H_{X|Y=y} = - \sum_{x,y} \mathbb{P}(X = x, Y = y) \log(\mathbb{P}(X = x | Y = y))$$

Joint Entropy

- We can define the **joint entropy**

$$H_{X,Y} = - \sum_{x,y} P_{X,Y}(x,y) \log(P_{X,Y}(x,y))$$

- If the message we receive is independent of the message that is sent then $H_{X,Y} = H_X + H_Y$ (we saw this in the last lecture)
- $H_{X,Y} \neq H_X + H_Y$ if X and Y are correlated
- Since $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X) = \mathbb{P}(X|Y)\mathbb{P}(Y)$ it follows

$$H_{X,Y} = H_X + H_{Y|X} = H_Y + H_{X|Y}$$

- Or $H_X - H_{X|Y} = H_Y - H_{Y|X}$

Joint Entropy

- We can define the **joint entropy**

$$H_{X,Y} = - \sum_{x,y} P_{X,Y}(x,y) \log(P_{X,Y}(x,y))$$

- If the message we receive is independent of the message that is sent then $H_{X,Y} = H_X + H_Y$ (we saw this in the last lecture)
- $H_{X,Y} \neq H_X + H_Y$ if X and Y are correlated
- Since $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X) = \mathbb{P}(X|Y)\mathbb{P}(Y)$ it follows

$$H_{X,Y} = H_X + H_{Y|X} = H_Y + H_{X|Y}$$

- Or $H_X - H_{X|Y} = H_Y - H_{Y|X}$

Joint Entropy

- We can define the **joint entropy**

$$H_{X,Y} = - \sum_{x,y} P_{X,Y}(x,y) \log(P_{X,Y}(x,y))$$

- If the message we receive is independent of the message that is sent then $H_{X,Y} = H_X + H_Y$ (we saw this in the last lecture)
- $H_{X,Y} \neq H_X + H_Y$ if X and Y are correlated
- Since $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X) = \mathbb{P}(X|Y)\mathbb{P}(Y)$ it follows

$$H_{X,Y} = H_X + H_{Y|X} = H_Y + H_{X|Y}$$

- Or $H_X - H_{X|Y} = H_Y - H_{Y|X}$

Joint Entropy

- We can define the **joint entropy**

$$H_{X,Y} = - \sum_{x,y} P_{X,Y}(x,y) \log(P_{X,Y}(x,y))$$

- If the message we receive is independent of the message that is sent then $H_{X,Y} = H_X + H_Y$ (we saw this in the last lecture)
- $H_{X,Y} \neq H_X + H_Y$ if X and Y are correlated
- Since $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X) = \mathbb{P}(X|Y)\mathbb{P}(Y)$ it follows

$$H_{X,Y} = H_X + H_{Y|X} = H_Y + H_{X|Y}$$

- Or $H_X - H_{X|Y} = H_Y - H_{Y|X}$

Joint Entropy

- We can define the **joint entropy**

$$H_{X,Y} = - \sum_{x,y} P_{X,Y}(x,y) \log(P_{X,Y}(x,y))$$

- If the message we receive is independent of the message that is sent then $H_{X,Y} = H_X + H_Y$ (we saw this in the last lecture)
- $H_{X,Y} \neq H_X + H_Y$ if X and Y are correlated
- Since $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X) = \mathbb{P}(X|Y)\mathbb{P}(Y)$ it follows

$$H_{X,Y} = H_X + H_{Y|X} = H_Y + H_{X|Y}$$

- Or $H_X - H_{X|Y} = H_Y - H_{Y|X}$

Mutual Information

- The amount of uncertainty about the message being sent, X , before receiving the message is $H_X = -\mathbb{E}_X[\log \mathbb{P}(X)]$
- Shannon define the *mutual information* to be the expected loss in uncertainty when we receive a message

$$I_{X;Y} = H_X - H_{X|Y}$$

- Since $H_X - H_{X|Y} = H_Y - H_{Y|X}$ it follows

$$I_{X;Y} = I_{Y;X}$$

Mutual Information

- The amount of uncertainty about the message being sent, X , before receiving the message is $H_X = -\mathbb{E}_X[\log \mathbb{P}(X)]$
- Shannon define the *mutual information* to be the expected loss in uncertainty when we receive a message

$$I_{X;Y} = H_X - H_{X|Y}$$

- Since $H_X - H_{X|Y} = H_Y - H_{Y|X}$ it follows

$$I_{X;Y} = I_{Y;X}$$

Mutual Information

- The amount of uncertainty about the message being sent, X , before receiving the message is $H_X = -\mathbb{E}_X[\log \mathbb{P}(X)]$
- Shannon define the *mutual information* to be the expected loss in uncertainty when we receive a message

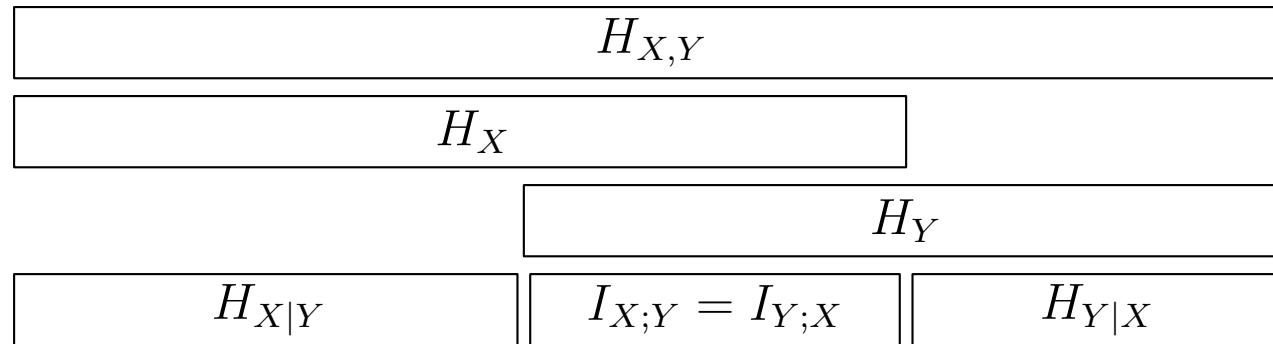
$$I_{X;Y} = H_X - H_{X|Y}$$

- Since $H_X - H_{X|Y} = H_Y - H_{Y|X}$ it follows

$$I_{X;Y} = I_{Y;X}$$

Channel Capacity

- We can summarise these relationships diagrammatically



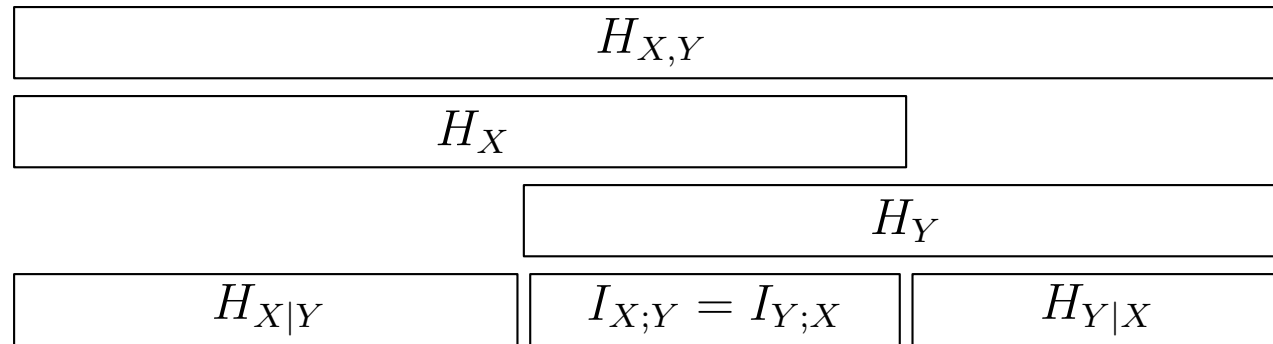
- Shannon defined the *capacity* of a noisy channel as

$$C = \max_{\mathbb{P}(X)} I_{X;Y}$$

- That is, you choose the probability distribution of the message to maximise the information gain

Channel Capacity

- We can summarise these relationships diagrammatically



- Shannon defined the *capacity* of a noisy channel as

$$C = \max_{\mathbb{P}(X)} I_{X;Y}$$

- That is, you choose the probability distribution of the message to maximise the information gain

Independent Noise

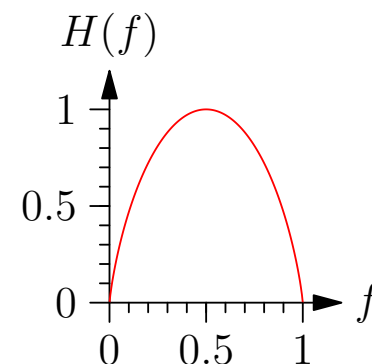
- The simplest model of a noisy channel is a binary channel where each symbol is corrupted independently with a probability f

$$\mathbb{P}(X = 1|Y = 0) = \mathbb{P}(X = 0|Y = 1) = f$$

- An elementary calculations shows that

$$H_{X_i|Y_i} = -(1 - f)\log(1 - f) - f\log(f) = H(f)$$

- For a message of length n , $H_{X|Y} = nH(f)$



Independent Noise

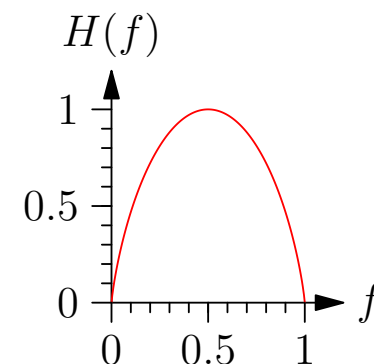
- The simplest model of a noisy channel is a binary channel where each symbol is corrupted independently with a probability f

$$\mathbb{P}(X = 1|Y = 0) = \mathbb{P}(X = 0|Y = 1) = f$$

- An elementary calculations shows that

$$H_{X_i|Y_i} = -(1 - f) \log(1 - f) - f \log(f) = H(f)$$

- For a message of length n , $H_{X|Y} = nH(f)$



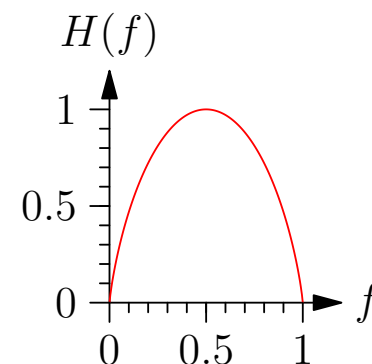
Independent Noise

- The simplest model of a noisy channel is a binary channel where each symbol is corrupted independently with a probability f

$$\mathbb{P}(X = 1|Y = 0) = \mathbb{P}(X = 0|Y = 1) = f$$

- An elementary calculations shows that

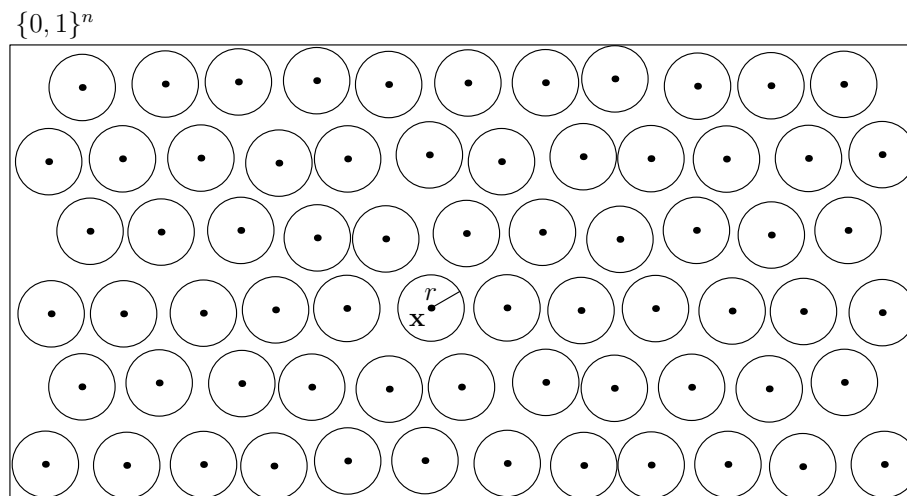
$$H_{X_i|Y_i} = -(1 - f) \log(1 - f) - f \log(f) = H(f)$$



- For a message of length n , $H_{X|Y} = nH(f)$

Error Correcting Codes

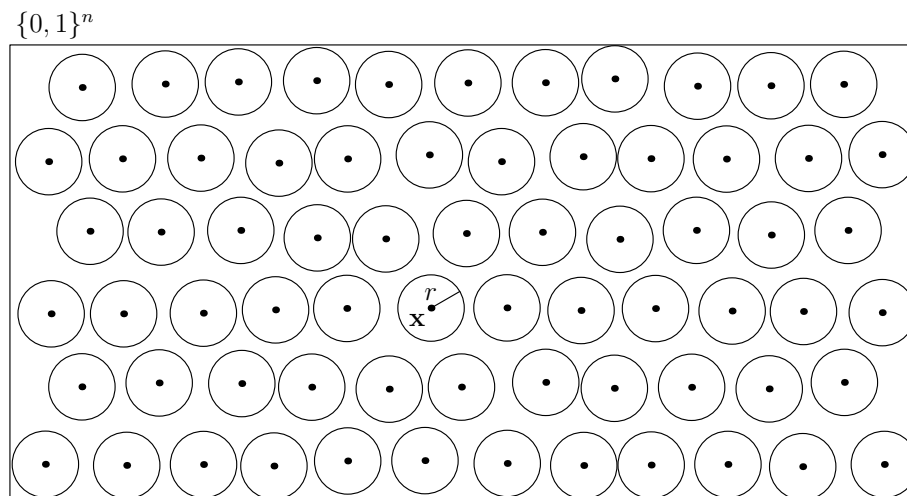
- To reduce the chance of misinterpreting a message we need to build an error correcting code
- We can do this dividing the space of binary messages into a set of Hamming balls



- A Hamming ball $B(x, r)$ is the set of strings that differ from n -dimensional binary string, x , by at most r digits

Error Correcting Codes

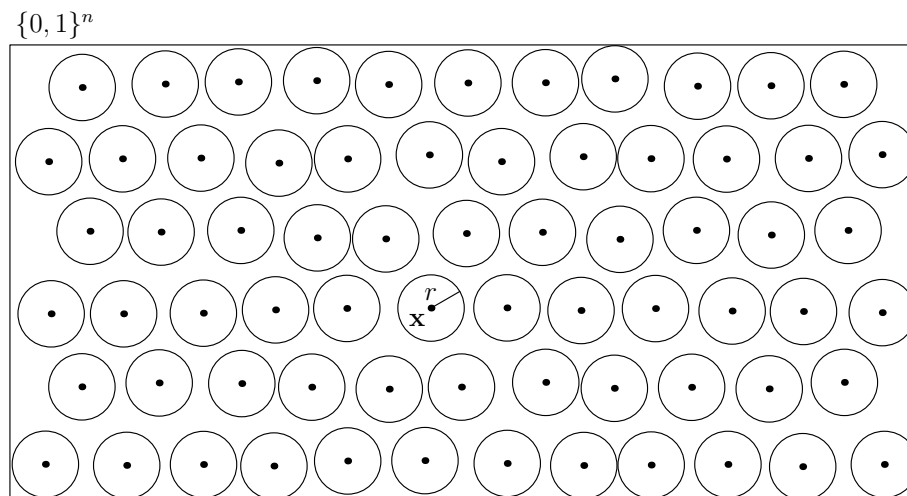
- To reduce the chance of misinterpreting a message we need to build an error correcting code
- We can do this dividing the space of binary messages into a set of Hamming balls



- A Hamming ball $B(\mathbf{x}, r)$ is the set of strings that differ from n -dimensional binary string, \mathbf{x} , by at most r digits

Error Correcting Codes

- To reduce the chance of misinterpreting a message we need to build an error correcting code
- We can do this dividing the space of binary messages into a set of Hamming balls



- A Hamming ball $B(x, r)$ is the set of strings that differ from n -dimensional binary string, x , by at most r digits

Volume of Coding Space

- The expected number of errors in a string of length n given an error rate of f is nf
- For sufficiently large n we would expect all errors are smaller than $(f + \epsilon)n$ (for $\epsilon > 0$)
- If we make the radius of the Hamming ball $r = (f + \epsilon)n$ ($\epsilon > 0$) then we would expect no error for sufficiently large n
- An upper bound on the number of code words we can send in a string of length n is

$$\frac{2^n}{|B(\mathbf{x}_i, r)|} = c\sqrt{n}2^{I_{X;Y}}$$

Volume of Coding Space

- The expected number of errors in a string of length n given an error rate of f is nf
- For sufficiently large n we would expect all errors are smaller than $(f + \epsilon)n$ (for $\epsilon > 0$)
- If we make the radius of the Hamming ball $r = (f + \epsilon)n$ ($\epsilon > 0$) then we would expect no error for sufficiently large n
- An upper bound on the number of code words we can send in a string of length n is

$$\frac{2^n}{|B(\mathbf{x}_i, r)|} = c\sqrt{n}2^{I_{X;Y}}$$

Volume of Coding Space

- The expected number of errors in a string of length n given an error rate of f is nf
- For sufficiently large n we would expect all errors are smaller than $(f + \epsilon)n$ (for $\epsilon > 0$)
- If we make the radius of the Hamming ball $r = (f + \epsilon)n$ ($\epsilon > 0$) then we would expect no error for sufficiently large n
- An upper bound on the number of code words we can send in a string of length n is

$$\frac{2^n}{|B(\mathbf{x}_i, r)|} = c\sqrt{n}2^{I_{X;Y}}$$

Volume of Coding Space

- The expected number of errors in a string of length n given an error rate of f is nf
- For sufficiently large n we would expect all errors are smaller than $(f + \epsilon)n$ (for $\epsilon > 0$)
- If we make the radius of the Hamming ball $r = (f + \epsilon)n$ ($\epsilon > 0$) then we would expect no error for sufficiently large n
- An upper bound on the number of code words we can send in a string of length n is

$$\frac{2^n}{|B(\mathbf{x}_i, r)|} = c\sqrt{n}2^{I_{X;Y}}$$

Lower Bounds

- Shannon also showed that choosing $2^{I_{X;Y}}$ random strings of length n the Hamming distance between balls would be at least f with high probability
- This means that we can send information at rate of $I_{X;Y}$
- The maximum rate is given by the channel capacity $\max I_{X;Y}$
- If $f = 0.1$ then $C = I_{X;Y} = 0.469$ bits so we need codes of just over twice as long to communicate accurately over a noisy channel with a 10% corruption rate
- Unfortunately, we can't efficiently decode random code positions, so although we know Shannon's bound is achievable we don't have practical codes that do this

Lower Bounds

- Shannon also showed that choosing $2^{I_{X;Y}}$ random strings of length n the Hamming distance between balls would be at least f with high probability
- This means that we can send information at rate of $I_{X;Y}$
- The maximum rate is given by the channel capacity $\max I_{X;Y}$
- If $f = 0.1$ then $C = I_{X;Y} = 0.469$ bits so we need codes of just over twice as long to communicate accurately over a noisy channel with a 10% corruption rate
- Unfortunately, we can't efficiently decode random code positions, so although we know Shannon's bound is achievable we don't have practical codes that do this

Lower Bounds

- Shannon also showed that choosing $2^{I_{X;Y}}$ random strings of length n the Hamming distance between balls would be at least f with high probability
- This means that we can send information at rate of $I_{X;Y}$
- The maximum rate is given by the channel capacity $\max I_{X;Y}$
- If $f = 0.1$ then $C = I_{X;Y} = 0.469$ bits so we need codes of just over twice as long to communicate accurately over a noisy channel with a 10% corruption rate
- Unfortunately, we can't efficiently decode random code positions, so although we know Shannon's bound is achievable we don't have practical codes that do this

Lower Bounds

- Shannon also showed that choosing $2^{I_{X;Y}}$ random strings of length n the Hamming distance between balls would be at least f with high probability
- This means that we can send information at rate of $I_{X;Y}$
- The maximum rate is given by the channel capacity $\max I_{X;Y}$
- If $f = 0.1$ then $C = I_{X;Y} = 0.469$ bits so we need codes of just over twice as long to communicate accurately over a noisy channel with a 10% corruption rate
- Unfortunately, we can't efficiently decode random code positions, so although we know Shannon's bound is achievable we don't have practical codes that do this

Lower Bounds

- Shannon also showed that choosing $2^{I_{X;Y}}$ random strings of length n the Hamming distance between balls would be at least f with high probability
- This means that we can send information at rate of $I_{X;Y}$
- The maximum rate is given by the channel capacity $\max I_{X;Y}$
- If $f = 0.1$ then $C = I_{X;Y} = 0.469$ bits so we need codes of just over twice as long to communicate accurately over a noisy channel with a 10% corruption rate
- Unfortunately, we can't efficiently decode random code positions, so although we know Shannon's bound is achievable we don't have practical codes that do this

Using Mutual Information

- Mutual information is used quite often in machine learning
 - ★ Wikipedia mentions 14 applications
- Suppose we want to align two sets of images through some non-linear transformations
- One way of doing this is to choose the non-linear transformations that maximise the mutual information (or normalised mutual information) between the two sets of images

Using Mutual Information

- Mutual information is used quite often in machine learning
 - ★ Wikipedia mentions 14 applications
- Suppose we want to align two sets of images through some non-linear transformations
- One way of doing this is to choose the non-linear transformations that maximise the mutual information (or normalised mutual information) between the two sets of images

Using Mutual Information

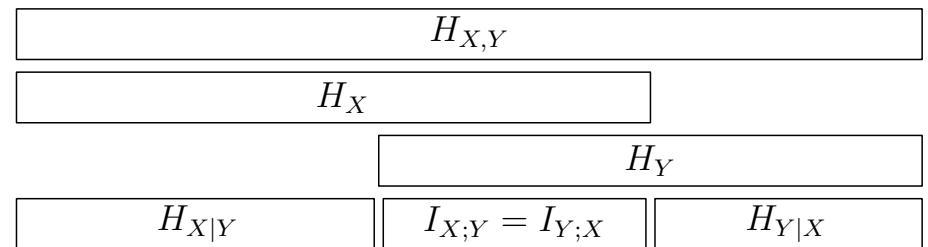
- Mutual information is used quite often in machine learning
 - ★ Wikipedia mentions 14 applications
- Suppose we want to align two sets of images through some non-linear transformations
- One way of doing this is to choose the non-linear transformations that maximise the mutual information (or normalised mutual information) between the two sets of images

Using Mutual Information

- Mutual information is used quite often in machine learning
 - ★ Wikipedia mentions 14 applications
- Suppose we want to align two sets of images through some non-linear transformations
- One way of doing this is to choose the non-linear transformations that maximise the mutual information (or normalised mutual information) between the two sets of images

Outline

1. Information Theory
2. **KL-Divergence**
3. Minimum Description Length
4. Variational Auto-Encoders



KL-Divergence

- We have met the Kullback-Leibler divergence

$$\begin{aligned}\text{KL}(p\|q) &= \mathbb{E}_{X \sim p(X)} \left[\log \left(\frac{p(X)}{q(X)} \right) \right] \\ &= -\mathbb{E}_{X \sim p(X)} [\log(q(X))] - H_X\end{aligned}$$

- Recall $-\log(q(X = x))$ is the length of code need to send a message x with a probability $q(X = x)$
- Thus $-\mathbb{E}_{X \sim p(X)} [\log(q(X))]$ is the expected length of message needed to code $X \sim p(X)$ using the optimal code for the distribution $q(X)$ that than $p(X)$
- $\text{KL}(p\|q)$ is also known as the **relative entropy** and measures the expected extra length in coding $X \sim p(X)$ if we use the wrong distribution $q(X)$

KL-Divergence

- We have met the Kullback-Leibler divergence

$$\begin{aligned}\text{KL}(p\|q) &= \mathbb{E}_{X \sim p(X)} \left[\log \left(\frac{p(X)}{q(X)} \right) \right] \\ &= -\mathbb{E}_{X \sim p(X)} [\log(q(X))] - H_X\end{aligned}$$

- Recall $-\log(q(X = x))$ is the length of code need to send a message x with a probability $q(X = x)$
- Thus $-\mathbb{E}_{X \sim p(X)} [\log(q(X))]$ is the expected length of message needed to code $X \sim p(X)$ using the optimal code for the distribution $q(X)$ that than $p(X)$
- $\text{KL}(p\|q)$ is also known as the **relative entropy** and measures the expected extra length in coding $X \sim p(X)$ if we use the wrong distribution $q(X)$

KL-Divergence

- We have met the Kullback-Leibler divergence

$$\begin{aligned}\text{KL}(p\|q) &= \mathbb{E}_{X \sim p(X)} \left[\log \left(\frac{p(X)}{q(X)} \right) \right] \\ &= -\mathbb{E}_{X \sim p(X)} [\log(q(X))] - H_X\end{aligned}$$

- Recall $-\log(q(X = x))$ is the length of code need to send a message x with a probability $q(X = x)$
- Thus $-\mathbb{E}_{X \sim p(X)} [\log(q(X))]$ is the expected length of message needed to code $X \sim p(X)$ using the optimal code for the distribution $q(X)$ that than $p(X)$
- $\text{KL}(p\|q)$ is also known as the **relative entropy** and measures the expected extra length in coding $X \sim p(X)$ if we use the wrong distribution $q(X)$

KL-Divergence

- We have met the Kullback-Leibler divergence

$$\begin{aligned}\text{KL}(p\|q) &= \mathbb{E}_{X \sim p(X)} \left[\log \left(\frac{p(X)}{q(X)} \right) \right] \\ &= -\mathbb{E}_{X \sim p(X)} [\log(q(X))] - H_X\end{aligned}$$

- Recall $-\log(q(X = x))$ is the length of code need to send a message x with a probability $q(X = x)$
- Thus $-\mathbb{E}_{X \sim p(X)} [\log(q(X))]$ is the expected length of message needed to code $X \sim p(X)$ using the optimal code for the distribution $q(X)$ that than $p(X)$
- $\text{KL}(p\|q)$ is also known as the **relative entropy** and measures the expected extra length in coding $X \sim p(X)$ if we use the wrong distribution $q(X)$

Variational Approximation

- Recall we use MCMC in Bayesian inference because the posterior distribution is too complicated to write down in closed form
- In the variational approximation we approximate the posterior distribution by a simpler (typically factored distribution), e.g.

$$f(\boldsymbol{\theta} \mid \mathcal{D}) \approx g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) = \prod_i g(\theta_i \mid \phi_i)$$

- The standard method for solving this is to maximise the **variational free energy**

$$\Phi(\boldsymbol{\phi}) = - \int g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) \log \left(\frac{g(\boldsymbol{\theta} \mid \boldsymbol{\phi})}{f(\boldsymbol{\theta}, \mathcal{D})} \right) d\boldsymbol{\theta}$$

Variational Approximation

- Recall we use MCMC in Bayesian inference because the posterior distribution is too complicated to write down in closed form
- In the variational approximation we approximate the posterior distribution by a simpler (typically factored distribution), e.g.

$$f(\boldsymbol{\theta} \mid \mathcal{D}) \approx g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) = \prod_i g(\theta_i \mid \phi_i)$$

- The standard method for solving this is to maximise the **variational free energy**

$$\Phi(\boldsymbol{\phi}) = - \int g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) \log \left(\frac{g(\boldsymbol{\theta} \mid \boldsymbol{\phi})}{f(\boldsymbol{\theta}, \mathcal{D})} \right) d\boldsymbol{\theta}$$

Variational Approximation

- Recall we use MCMC in Bayesian inference because the posterior distribution is too complicated to write down in closed form
- In the variational approximation we approximate the posterior distribution by a simpler (typically factored distribution), e.g.

$$f(\boldsymbol{\theta} \mid \mathcal{D}) \approx g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) = \prod_i g(\theta_i \mid \phi_i)$$

- The standard method for solving this is to maximise the **variational free energy**

$$\Phi(\boldsymbol{\phi}) = - \int g(\boldsymbol{\theta} \mid \boldsymbol{\phi}) \log \left(\frac{g(\boldsymbol{\theta} \mid \boldsymbol{\phi})}{f(\boldsymbol{\theta}, \mathcal{D})} \right) d\boldsymbol{\theta}$$

Evidence Lower Bound (ELBO)

- We can re-write the variational free energy as

$$\begin{aligned}\Phi(\phi) &= - \int g(\boldsymbol{\theta} \mid \phi) \log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{(f(\boldsymbol{\theta}, \mathcal{D})/f(\mathcal{D})) f(\mathcal{D})} \right) d\boldsymbol{\theta} \\ &= - \int g(\boldsymbol{\theta} \mid \phi) \left(\log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{f(\boldsymbol{\theta} \mid \mathcal{D})} \right) - \log(f(\mathcal{D})) \right) d\boldsymbol{\theta} \\ &= -\text{KL}(g(\boldsymbol{\theta} \mid \phi) \parallel f(\boldsymbol{\theta} \mid \mathcal{D})) + \log(f(\mathcal{D}))\end{aligned}$$

- If we maximise $\Phi(\phi)$, we end up minimising the KL divergence between g and f so that $g \approx f$ and $\Phi(\phi) \approx \log(f(\mathcal{D}))$
- That is, we choose the parameters of our simple factorised distribution so that it is close to the true posterior

Evidence Lower Bound (ELBO)

- We can re-write the variational free energy as

$$\begin{aligned}\Phi(\phi) &= - \int g(\boldsymbol{\theta} \mid \phi) \log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{(f(\boldsymbol{\theta}, \mathcal{D})/f(\mathcal{D})) f(\mathcal{D})} \right) d\boldsymbol{\theta} \\ &= - \int g(\boldsymbol{\theta} \mid \phi) \left(\log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{f(\boldsymbol{\theta} \mid \mathcal{D})} \right) - \log(f(\mathcal{D})) \right) d\boldsymbol{\theta} \\ &= -\text{KL}(g(\boldsymbol{\theta} \mid \phi) \parallel f(\boldsymbol{\theta} \mid \mathcal{D})) + \log(f(\mathcal{D}))\end{aligned}$$

- If we maximise $\Phi(\phi)$, we end up minimising the KL divergence between g and f so that $g \approx f$ and $\Phi(\phi) \approx \log(f(\mathcal{D}))$
- That is, we choose the parameters of our simple factorised distribution so that it is close to the true posterior

Evidence Lower Bound (ELBO)

- We can re-write the variational free energy as

$$\begin{aligned}\Phi(\phi) &= - \int g(\boldsymbol{\theta} \mid \phi) \log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{(f(\boldsymbol{\theta}, \mathcal{D})/f(\mathcal{D})) f(\mathcal{D})} \right) d\boldsymbol{\theta} \\ &= - \int g(\boldsymbol{\theta} \mid \phi) \left(\log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{f(\boldsymbol{\theta} \mid \mathcal{D})} \right) - \log(f(\mathcal{D})) \right) d\boldsymbol{\theta} \\ &= -\text{KL}(g(\boldsymbol{\theta} \mid \phi) \parallel f(\boldsymbol{\theta} \mid \mathcal{D})) + \log(f(\mathcal{D}))\end{aligned}$$

- If we maximise $\Phi(\phi)$, we end up minimising the KL divergence between g and f so that $g \approx f$ and $\Phi(\phi) \approx \log(f(\mathcal{D}))$
- That is, we choose the parameters of our simple factorised distribution so that it is close to the true posterior

Evidence Lower Bound (ELBO)

- We can re-write the variational free energy as

$$\begin{aligned}\Phi(\phi) &= - \int g(\boldsymbol{\theta} \mid \phi) \log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{(f(\boldsymbol{\theta}, \mathcal{D})/f(\mathcal{D})) f(\mathcal{D})} \right) d\boldsymbol{\theta} \\ &= - \int g(\boldsymbol{\theta} \mid \phi) \left(\log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{f(\boldsymbol{\theta} \mid \mathcal{D})} \right) - \log(f(\mathcal{D})) \right) d\boldsymbol{\theta} \\ &= -\text{KL}(g(\boldsymbol{\theta} \mid \phi) \parallel f(\boldsymbol{\theta} \mid \mathcal{D})) + \log(f(\mathcal{D}))\end{aligned}$$

- If we maximise $\Phi(\phi)$, we end up minimising the KL divergence between g and f so that $g \approx f$ and $\Phi(\phi) \approx \log(f(\mathcal{D}))$
- That is, we choose the parameters of our simple factorised distribution so that it is close to the true posterior

Evidence Lower Bound (ELBO)

- We can re-write the variational free energy as

$$\begin{aligned}\Phi(\phi) &= - \int g(\boldsymbol{\theta} \mid \phi) \log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{(f(\boldsymbol{\theta}, \mathcal{D})/f(\mathcal{D})) f(\mathcal{D})} \right) d\boldsymbol{\theta} \\ &= - \int g(\boldsymbol{\theta} \mid \phi) \left(\log \left(\frac{g(\boldsymbol{\theta} \mid \phi)}{f(\boldsymbol{\theta} \mid \mathcal{D})} \right) - \log(f(\mathcal{D})) \right) d\boldsymbol{\theta} \\ &= -\text{KL}(g(\boldsymbol{\theta} \mid \phi) \parallel f(\boldsymbol{\theta} \mid \mathcal{D})) + \log(f(\mathcal{D}))\end{aligned}$$

- If we maximise $\Phi(\phi)$, we end up minimising the KL divergence between g and f so that $g \approx f$ and $\Phi(\phi) \approx \log(f(\mathcal{D}))$
- That is, we choose the parameters of our simple factorised distribution so that it is close to the true posterior

Put Another Way

- We can rewrite the variational free energy as $\Phi(\phi) = L_q(\phi) + H_q(\phi)$ where

$$L_q(\phi) = \int g(\boldsymbol{\theta} \mid \phi) (\log(f(\mathcal{D} \mid \boldsymbol{\theta})) + \log(f(\boldsymbol{\theta}))) d\phi$$

acts like an expected posterior term that is maximised when the data is well modelled (we put the probability density, $g(\boldsymbol{\theta} \mid \phi)$ where the $f(\boldsymbol{\theta}, \mathcal{D})$ is large)

- The second term is an entropy

$$H_q(\phi) = - \int g(\boldsymbol{\theta} \mid \phi) \log(g(\boldsymbol{\theta} \mid \phi)) d\phi$$

That is, we maximise the uncertainty of the distribution $g(\boldsymbol{\theta} \mid \phi)$

Put Another Way

- We can rewrite the variational free energy as $\Phi(\phi) = L_q(\phi) + H_q(\phi)$ where

$$L_q(\phi) = \int g(\boldsymbol{\theta} \mid \phi) (\log(f(\mathcal{D} \mid \boldsymbol{\theta})) + \log(f(\boldsymbol{\theta}))) d\phi$$

acts like an expected posterior term that is maximised when the data is well modelled (we put the probability density, $g(\boldsymbol{\theta} \mid \phi)$ where the $f(\boldsymbol{\theta}, \mathcal{D})$ is large)

- The second term is an entropy

$$H_q(\phi) = - \int g(\boldsymbol{\theta} \mid \phi) \log(g(\boldsymbol{\theta} \mid \phi)) d\phi$$

That is, we maximise the uncertainty of the distribution $g(\boldsymbol{\theta} \mid \phi)$

Using Variational Methods

- Variational methods can be much faster than MCMC (although they tend to involve some iterations to minimise the variation free energy)
- They can produce very good approximations, although this is not guaranteed (depends on the problem)
- They can be extended (e.g. by minimising $\text{KL}(g||f)$ rather than $\text{KL}(f||g)$)—this is known as *belief propagation*)
- MCMC is less elegant, but is a controlled approximation (we get better results by increasing the number of iterations)
- MCMC is slower, but on modern computers this isn't usually a problem

Using Variational Methods

- Variational methods can be much faster than MCMC (although they tend to involve some iterations to minimise the variation free energy)
- They can produce very good approximations, although this is not guaranteed (depends on the problem)
- They can be extended (e.g. by minimising $\text{KL}(g||f)$ rather than $\text{KL}(f||g)$)—this is known as *belief propagation*)
- MCMC is less elegant, but is a controlled approximation (we get better results by increasing the number of iterations)
- MCMC is slower, but on modern computers this isn't usually a problem

Using Variational Methods

- Variational methods can be much faster than MCMC (although they tend to involve some iterations to minimise the variation free energy)
- They can produce very good approximations, although this is not guaranteed (depends on the problem)
- They can be extended (e.g. by minimising $\text{KL}(g||f)$ rather than $\text{KL}(f||g)$)—this is known as *belief propagation*
- MCMC is less elegant, but is a controlled approximation (we get better results by increasing the number of iterations)
- MCMC is slower, but on modern computers this isn't usually a problem

Using Variational Methods

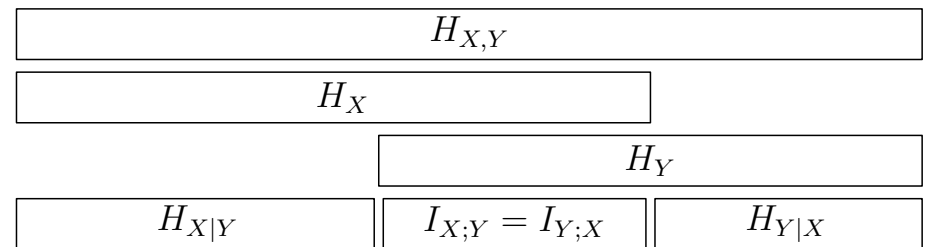
- Variational methods can be much faster than MCMC (although they tend to involve some iterations to minimise the variation free energy)
- They can produce very good approximations, although this is not guaranteed (depends on the problem)
- They can be extended (e.g. by minimising $\text{KL}(g\|f)$ rather than $\text{KL}(f\|g)$)—this is known as *belief propagation*)
- MCMC is less elegant, but is a controlled approximation (we get better results by increasing the number of iterations)
- MCMC is slower, but on modern computers this isn't usually a problem

Using Variational Methods

- Variational methods can be much faster than MCMC (although they tend to involve some iterations to minimise the variation free energy)
- They can produce very good approximations, although this is not guaranteed (depends on the problem)
- They can be extended (e.g. by minimising $\text{KL}(g||f)$ rather than $\text{KL}(f||g)$)—this is known as *belief propagation*)
- MCMC is less elegant, but is a controlled approximation (we get better results by increasing the number of iterations)
- MCMC is slower, but on modern computers this isn't usually a problem

Outline

1. Information Theory
2. KL-Divergence
3. **Minimum Description Length**
4. Variational Auto-Encoders



Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection—most of ML isn't Bayesian
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection—most of ML isn't Bayesian
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection—most of ML isn't Bayesian
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection—most of ML isn't Bayesian
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Compression and Model Selection

- Outside of the Bayesian framework it is difficult to do model selection—most of ML isn't Bayesian
- When is it better to accept a more complex model for a better fit and when are we just over-fitting?
- Usually we answer this using a validation set, but this is not always possible
- One principled approach is to use the model that allows us to maximally compress the data
- If we are compressing the data then we are capturing features of the data

Alice and Bob

- Suppose Alice has data $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \dots, m\}$ while Bob has only the feature vectors $\{\mathbf{x}_i \mid i = 1, 2, \dots, m\}$
- Alice wants to communicate y_i to Bob as efficiently as possible
- We suppose Alice & Bob have available a model $\hat{f}(\mathbf{x}|\boldsymbol{\theta})$
- Rather than sending the complete list $\{y_i \mid i = 1, 2, \dots, m\}$ Alice can send Bob the parameter $\boldsymbol{\theta}$ and the errors

$$\delta_i = y_i - \hat{f}(\mathbf{x}_i|\boldsymbol{\theta})$$

- Assuming the δ_i 's have a distribution p_δ then the cost of communicating an error to accuracy Δ is $-\log(p_\delta(\delta_i) \times \Delta)$

Alice and Bob

- Suppose Alice has data $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \dots, m\}$ while Bob has only the feature vectors $\{\mathbf{x}_i \mid i = 1, 2, \dots, m\}$
- Alice wants to communicate y_i to Bob as efficiently as possible
- We suppose Alice & Bob have available a model $\hat{f}(\mathbf{x}|\boldsymbol{\theta})$
- Rather than sending the complete list $\{y_i \mid i = 1, 2, \dots, m\}$ Alice can send Bob the parameter $\boldsymbol{\theta}$ and the errors

$$\delta_i = y_i - \hat{f}(\mathbf{x}_i|\boldsymbol{\theta})$$

- Assuming the δ_i 's have a distribution p_δ then the cost of communicating an error to accuracy Δ is $-\log(p_\delta(\delta_i) \times \Delta)$

Alice and Bob

- Suppose Alice has data $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \dots, m\}$ while Bob has only the feature vectors $\{\mathbf{x}_i \mid i = 1, 2, \dots, m\}$
- Alice wants to communicate y_i to Bob as efficiently as possible
- We suppose Alice & Bob have available a model $\hat{f}(\mathbf{x}|\boldsymbol{\theta})$
- Rather than sending the complete list $\{y_i \mid i = 1, 2, \dots, m\}$ Alice can send Bob the parameter $\boldsymbol{\theta}$ and the errors

$$\delta_i = y_i - \hat{f}(\mathbf{x}_i|\boldsymbol{\theta})$$

- Assuming the δ_i 's have a distribution p_δ then the cost of communicating an error to accuracy Δ is $-\log(p_\delta(\delta_i) \times \Delta)$

Alice and Bob

- Suppose Alice has data $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \dots, m\}$ while Bob has only the feature vectors $\{\mathbf{x}_i \mid i = 1, 2, \dots, m\}$
- Alice wants to communicate y_i to Bob as efficiently as possible
- We suppose Alice & Bob have available a model $\hat{f}(\mathbf{x}|\boldsymbol{\theta})$
- Rather than sending the complete list $\{y_i \mid i = 1, 2, \dots, m\}$ Alice can send Bob the parameter $\boldsymbol{\theta}$ and the errors

$$\delta_i = y_i - \hat{f}(\mathbf{x}_i|\boldsymbol{\theta})$$

- Assuming the δ_i 's have a distribution p_δ then the cost of communicating an error to accuracy Δ is $-\log(p_\delta(\delta_i) \times \Delta)$

Alice and Bob

- Suppose Alice has data $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \dots, m\}$ while Bob has only the feature vectors $\{\mathbf{x}_i \mid i = 1, 2, \dots, m\}$
- Alice wants to communicate y_i to Bob as efficiently as possible
- We suppose Alice & Bob have available a model $\hat{f}(\mathbf{x}|\boldsymbol{\theta})$
- Rather than sending the complete list $\{y_i \mid i = 1, 2, \dots, m\}$ Alice can send Bob the parameter $\boldsymbol{\theta}$ and the errors

$$\delta_i = y_i - \hat{f}(\mathbf{x}_i|\boldsymbol{\theta})$$

- Assuming the δ_i 's have a distribution p_δ then the cost of communicating an error to accuracy Δ is $-\log(p_\delta(\delta_i) \times \Delta)$

Description Length

- The **description length** for $\{y_i \mid i = 1, 2, \dots, m\}$ is then the cost of transmitting θ plus the cost of transmitting the errors

$$L = \sum_{k=1}^n \ell(\theta_k) - \sum_{i=1}^m \left(\log \left(p_{\delta} \left(y_i - \hat{f}(\mathbf{x}_i | \theta) \right) \right) + \log(\Delta) \right)$$

where $\ell(\theta_k)$ is the number of bits need to communicate θ_k (we get to choose the accuracy if is worth encoding the parameters)

- To select between models we choose the model with the **minimum description length**
- Note that the accuracy Δ will lead to the same cost, $-m \log(\Delta)$, for all models so doesn't affect which model is selected

Description Length

- The **description length** for $\{y_i \mid i = 1, 2, \dots, m\}$ is then the cost of transmitting θ plus the cost of transmitting the errors

$$L = \sum_{k=1}^n \ell(\theta_k) - \sum_{i=1}^m \left(\log \left(p_{\delta} \left(y_i - \hat{f}(\mathbf{x}_i | \theta) \right) \right) + \log(\Delta) \right)$$

where $\ell(\theta_k)$ is the number of bits need to communicate θ_k (we get to choose the accuracy if is worth encoding the parameters)

- To select between models we choose the model with the **minimum description length**
- Note that the accuracy Δ will lead to the same cost, $-m \log(\Delta)$, for all models so doesn't affect which model is selected

Description Length

- The **description length** for $\{y_i \mid i = 1, 2, \dots, m\}$ is then the cost of transmitting θ plus the cost of transmitting the errors

$$L = \sum_{k=1}^n \ell(\theta_k) - \sum_{i=1}^m \left(\log \left(p_{\delta} \left(y_i - \hat{f}(\mathbf{x}_i | \theta) \right) \right) + \log(\Delta) \right)$$

where $\ell(\theta_k)$ is the number of bits need to communicate θ_k (we get to choose the accuracy if is worth encoding the parameters)

- To select between models we choose the model with the **minimum description length**
- Note that the accuracy Δ will lead to the same cost, $-m \log(\Delta)$, for all models so doesn't affect which model is selected

Minimum Description Length (MDL) Method

- The minimum description length method can be a powerful way of choosing between models
- Often it is the only principled method available
- It allows you to trade model accuracy against model complexity
- It can be fiddly as we need to determine the accuracy to which we should store the parameters of our model
- This isn't something we usually think about, but often we can get very good models even when we truncate the parameters to low precision

Minimum Description Length (MDL) Method

- The minimum description length method can be a powerful way of choosing between models
- Often it is the only principled method available
- It allows you to trade model accuracy against model complexity
- It can be fiddly as we need to determine the accuracy to which we should store the parameters of our model
- This isn't something we usually think about, but often we can get very good models even when we truncate the parameters to low precision

Minimum Description Length (MDL) Method

- The minimum description length method can be a powerful way of choosing between models
- Often it is the only principled method available
- It allows you to trade model accuracy against model complexity
- It can be fiddly as we need to determine the accuracy to which we should store the parameters of our model
- This isn't something we usually think about, but often we can get very good models even when we truncate the parameters to low precision

Minimum Description Length (MDL) Method

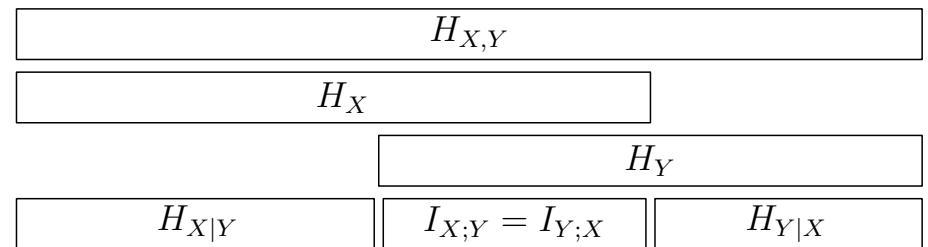
- The minimum description length method can be a powerful way of choosing between models
- Often it is the only principled method available
- It allows you to trade model accuracy against model complexity
- It can be fiddly as we need to determine the accuracy to which we should store the parameters of our model
- This isn't something we usually think about, but often we can get very good models even when we truncate the parameters to low precision

Minimum Description Length (MDL) Method

- The minimum description length method can be a powerful way of choosing between models
- Often it is the only principled method available
- It allows you to trade model accuracy against model complexity
- It can be fiddly as we need to determine the accuracy to which we should store the parameters of our model
- This isn't something we usually think about, but often we can get very good models even when we truncate the parameters to low precision

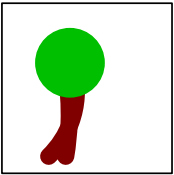
Outline

1. Information Theory
2. KL-Divergence
3. Minimum Description Length
4. **Variational Auto-Encoders**

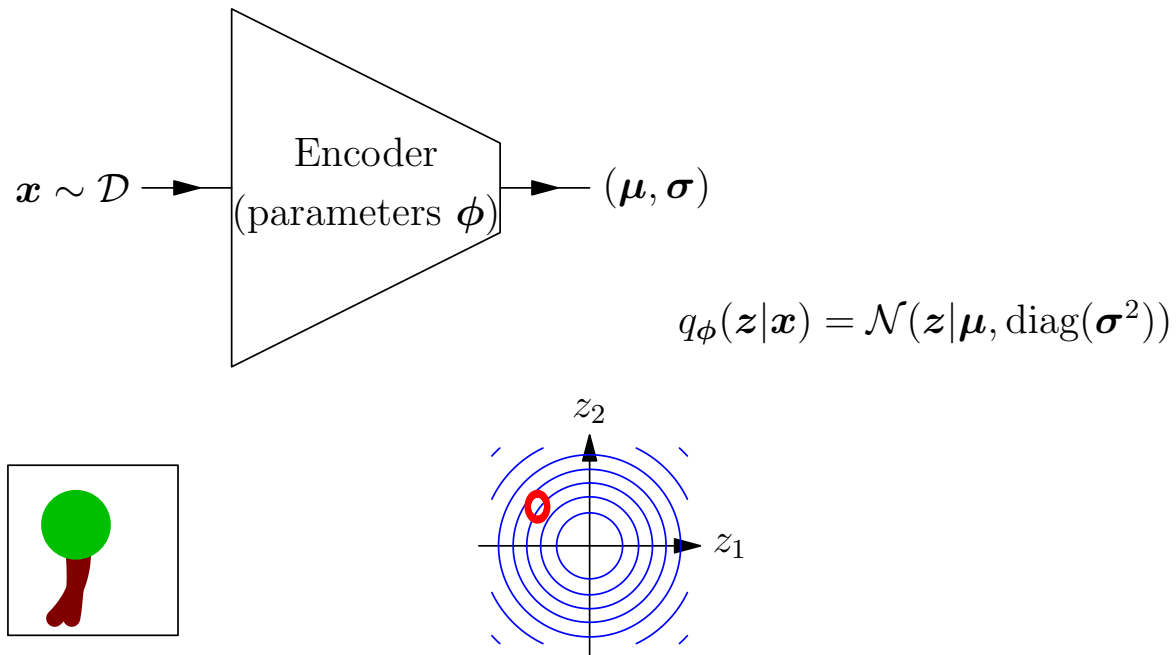


Variational Auto-Encoders VAE

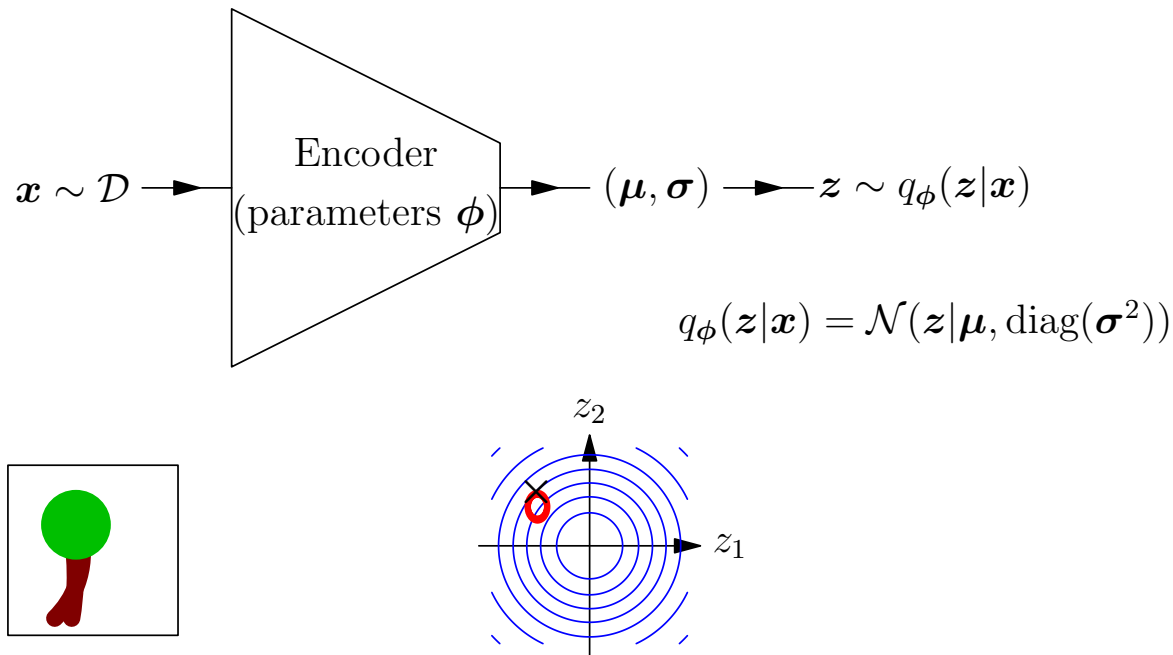
$$x \sim \mathcal{D}$$



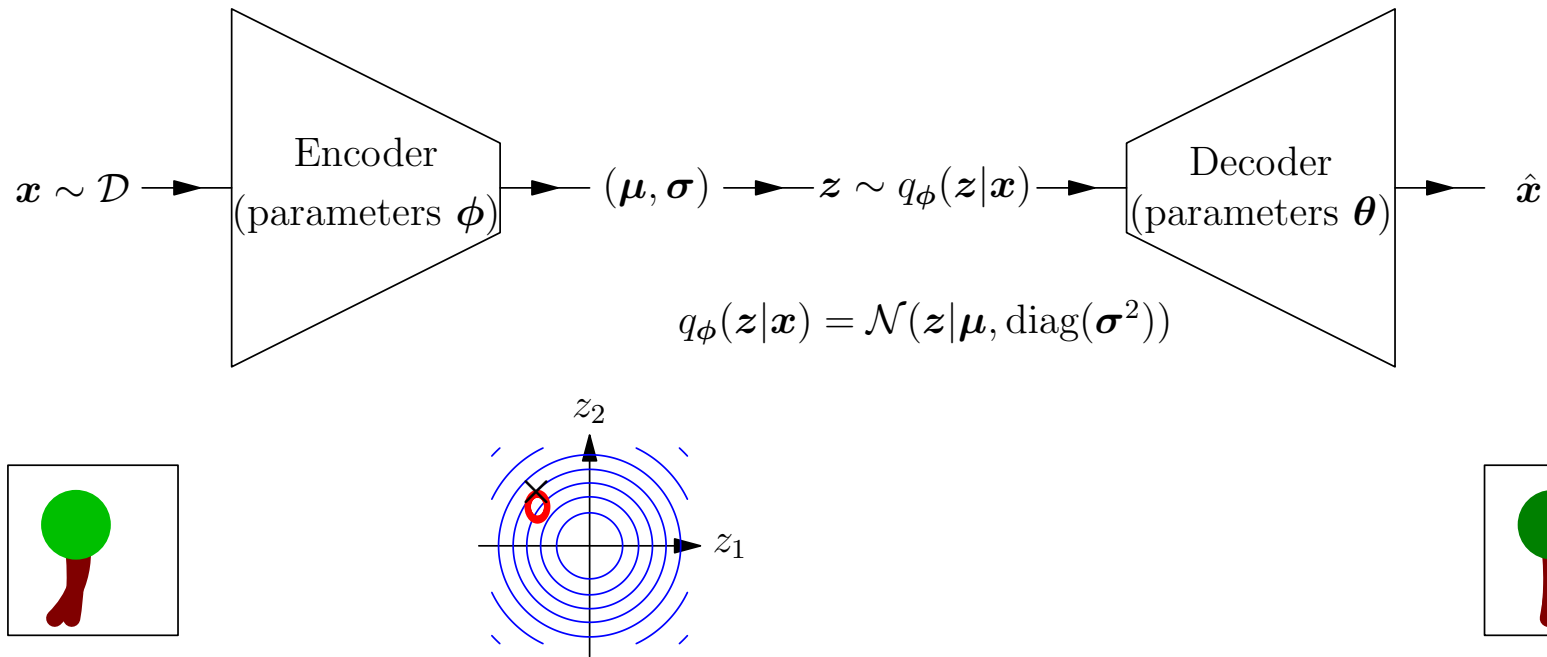
Variational Auto-Encoders VAE



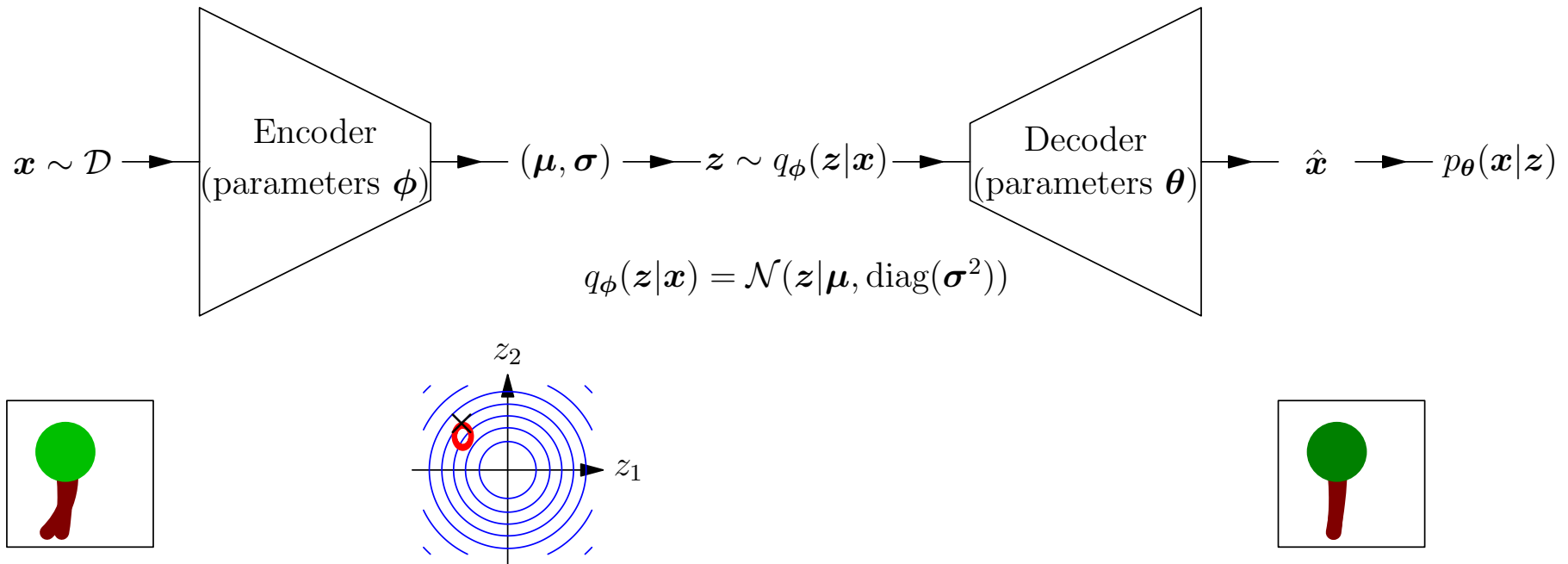
Variational Auto-Encoders VAE



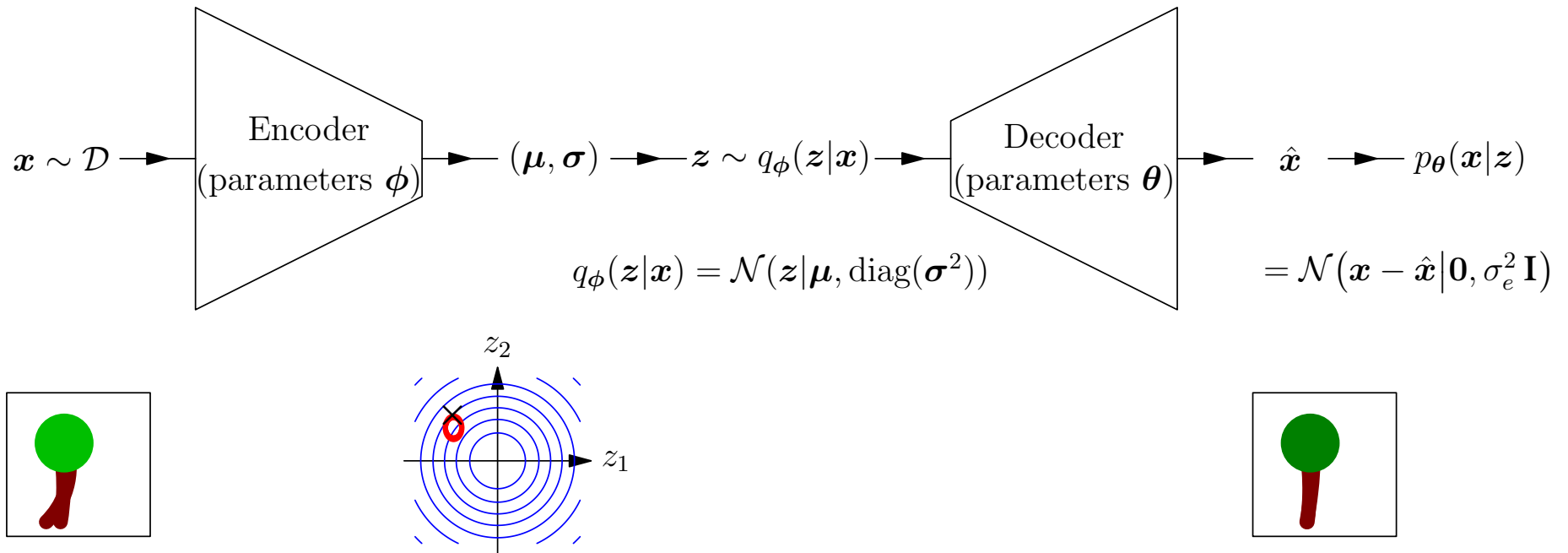
Variational Auto-Encoders VAE



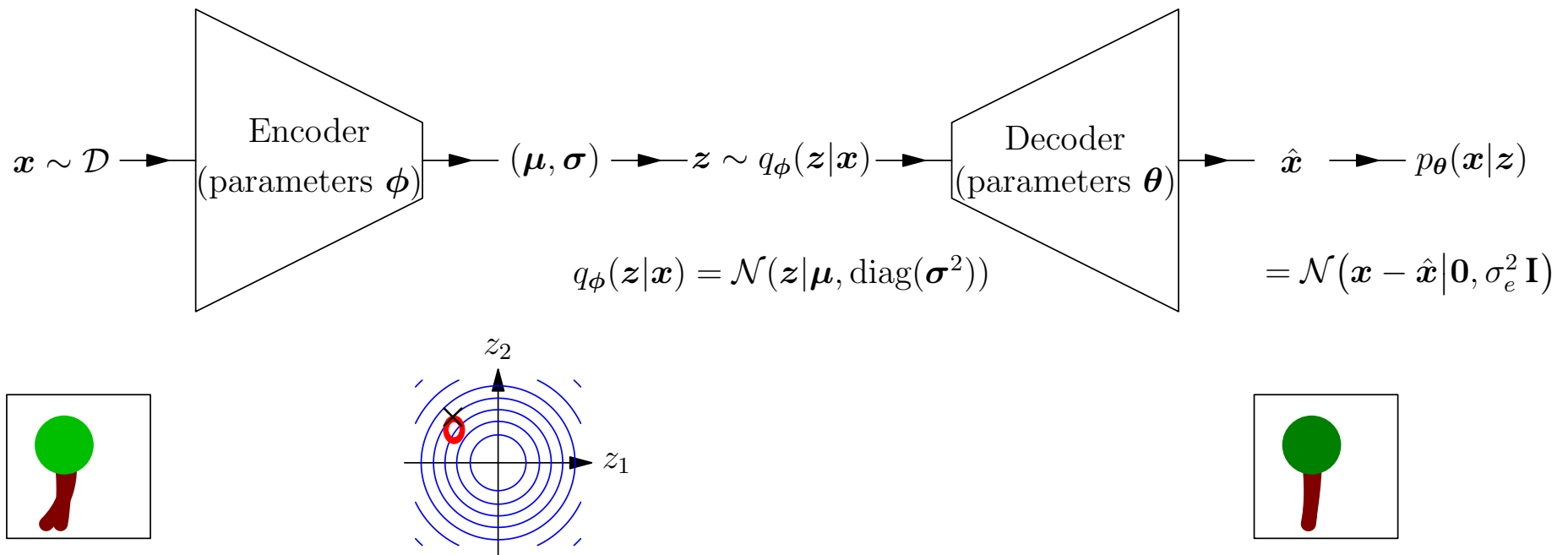
Variational Auto-Encoders VAE



Variational Auto-Encoders VAE

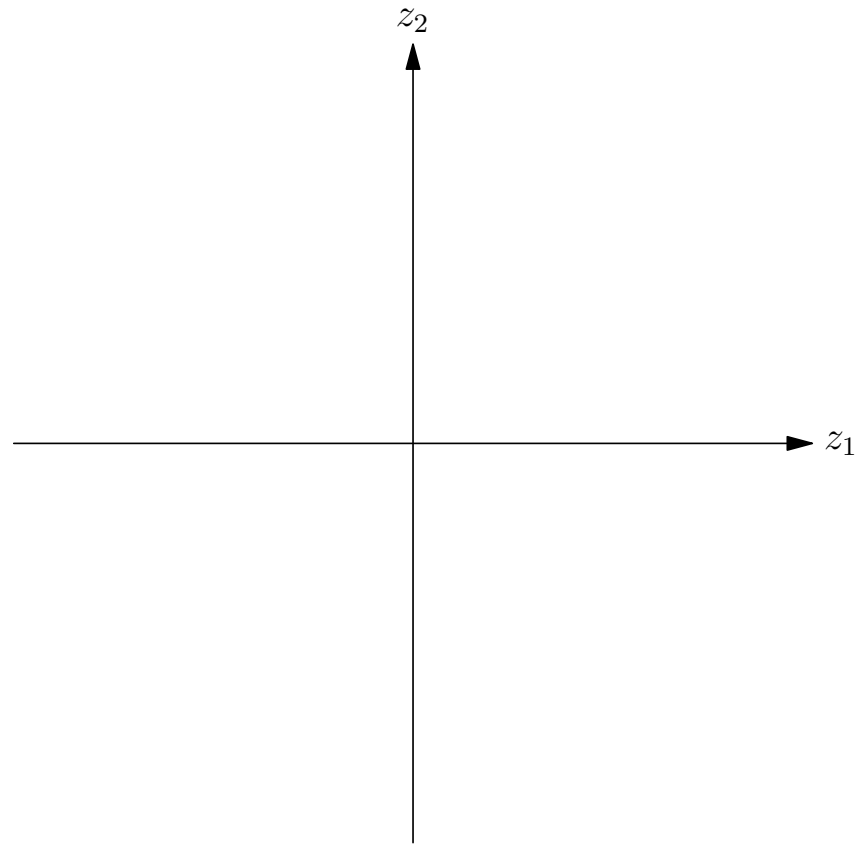


Variational Auto-Encoders VAE

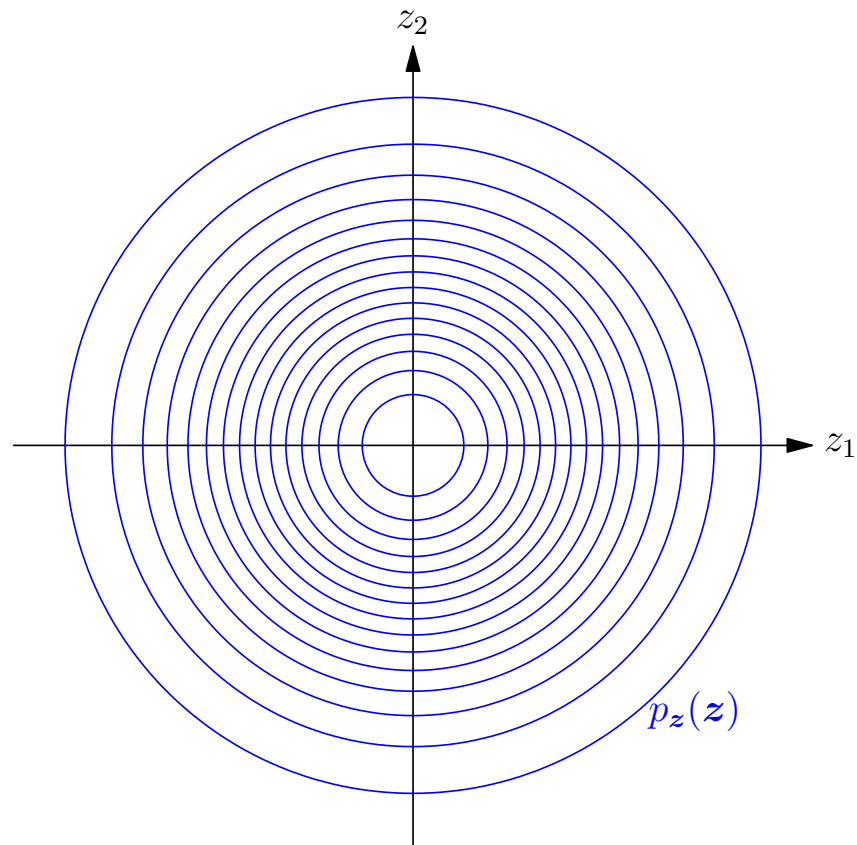


$$\mathcal{L} = \mathbb{E}_{x \sim \mathcal{D}} [\text{KL}(q_\theta(z|x) || \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_\theta(x|z(x)))]$$

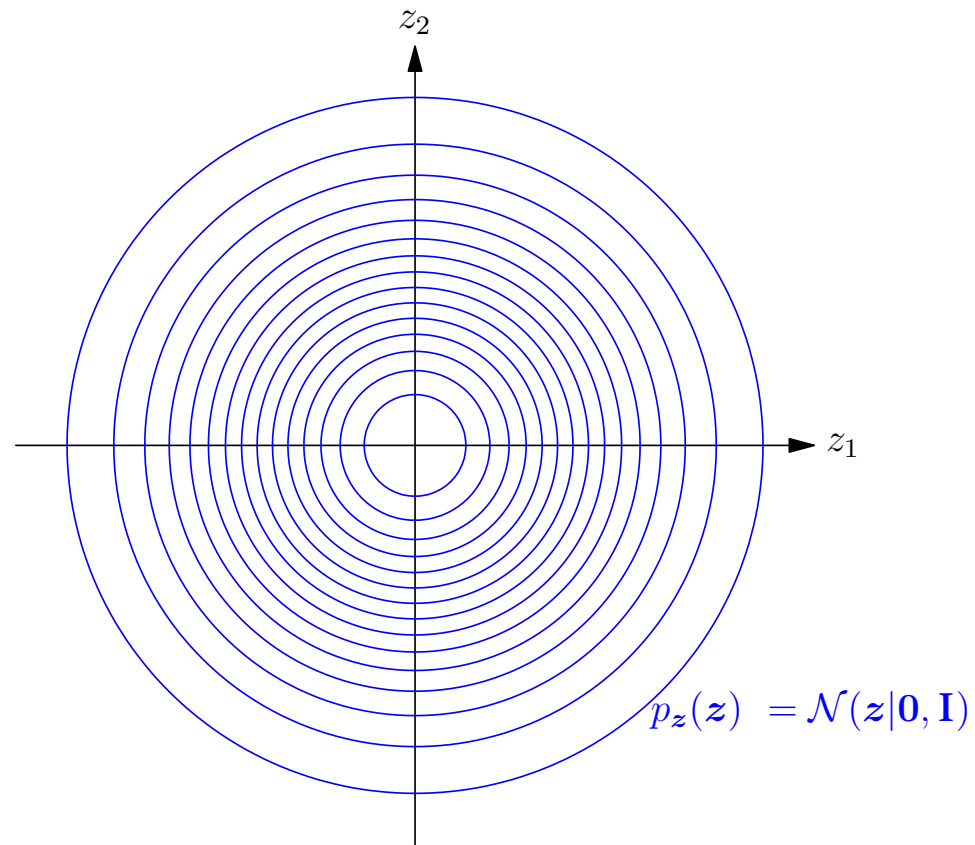
Latent Space



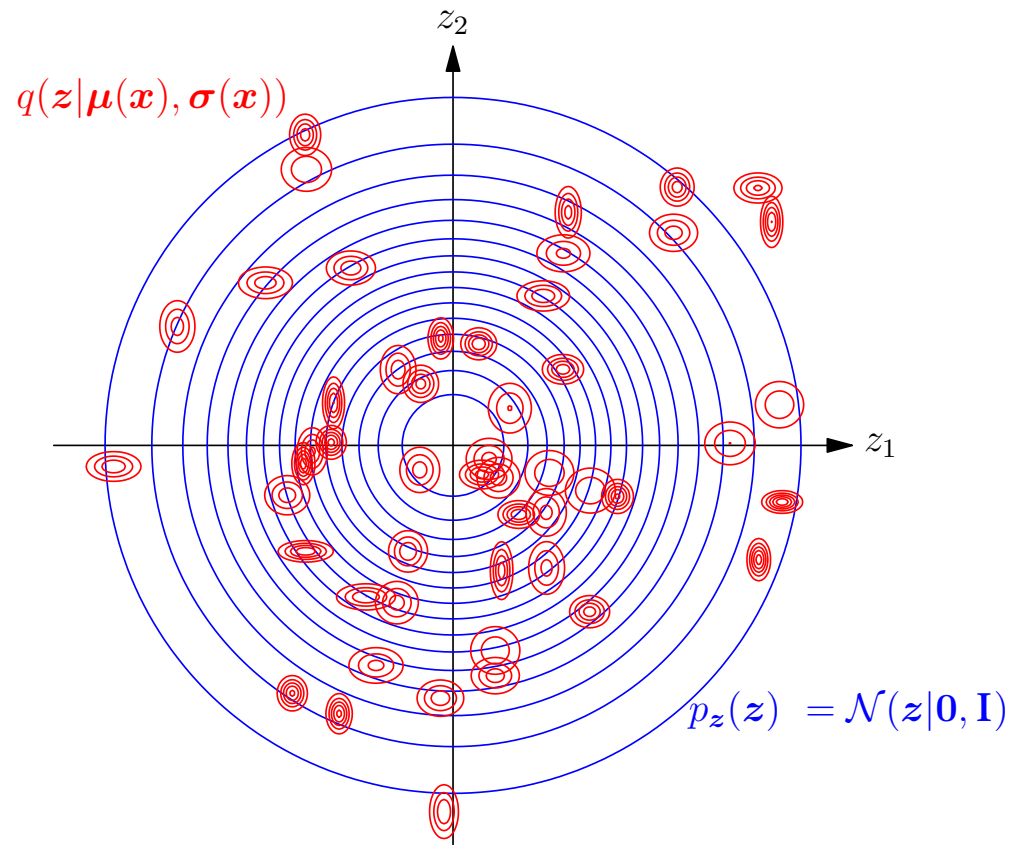
Latent Space



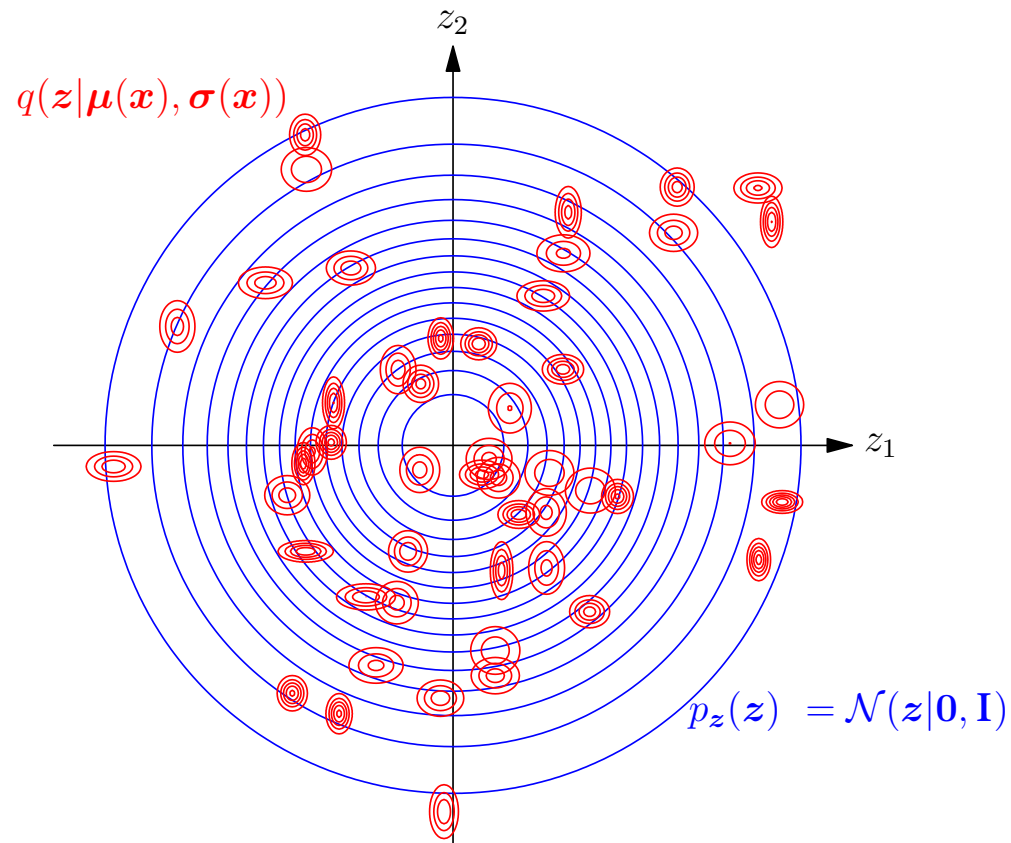
Latent Space



Latent Space



Latent Space



$$\text{KL}(q_\theta(z|x) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$$

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(\mathbf{z}|\mathbf{x})$ which she communicates to Bob
- She also communicates the errors $\delta = \mathbf{x} - \bar{\mathbf{x}}$
- Bob uses the decoder to decode $q(\mathbf{z}|\mathbf{x})$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(\mathbf{z}|\mathbf{x})$ which she communicates to Bob
- She also communicates the errors $\delta = \mathbf{x} - \bar{\mathbf{x}}$
- Bob uses the decoder to decode $q(\mathbf{z}|\mathbf{x})$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(\mathbf{z}|\mathbf{x})$ which she communicates to Bob
- She also communicates the errors $\delta = \mathbf{x} - \bar{\mathbf{x}}$
- Bob uses the decoder to decode $q(\mathbf{z}|\mathbf{x})$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(\mathbf{z}|\mathbf{x})$ which she communicates to Bob
- She also communicates the errors $\delta = \mathbf{x} - \bar{\mathbf{x}}$
- Bob uses the decoder to decode $q(\mathbf{z}|\mathbf{x})$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(z|x)$ which she communicates to Bob
- She also communicates the errors $\delta = x - \bar{x}$
- Bob uses the decoder to decode $q(z|x)$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(\mathbf{z}|\mathbf{x})$ which she communicates to Bob
- She also communicates the errors $\delta = \mathbf{x} - \bar{\mathbf{x}}$
- Bob uses the decoder to decode $q(\mathbf{z}|\mathbf{x})$ and δ to repair the images

Understanding the Loss Function

- The original paper derived the loss function as a variational approximation to maximising some posterior
- This is difficult to understand (at least, for me)
- It has a very natural explanation in terms of minimum description length
- Alice wants to communicate the images to Bob
- Alice uses the encoder to derive a (latent) code $q(z|x)$ which she communicates to Bob
- She also communicates the errors $\delta = x - \bar{x}$
- Bob uses the decoder to decode $q(z|x)$ and δ to repair the images

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
 - ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function
 - What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
- ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function
- What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
- ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function
- What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
- ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function
- What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
- ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function **equivalent to MDL**
- What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Description Length

- The loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) - \log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))]$$

can be interpreted as

- ★ The cost of communicating the code $\text{KL}(q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$
- ★ Plus the cost to send the repair $\log(p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}(\mathbf{x})))$
- We minimise the loss function equivalent to MDL
- What is really clever is that we can choose the accuracy of the code we send $q_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ to minimise the over-all cost

Conclusions

- Information theory has regularly been used in machine learning
- It requires some understanding and care to do it properly
- The KL-divergence (or relative entropy) is often used to make two probability distribution more alike
- The minimum description length is a powerful principle for model selection
- Variational Auto-Encoders have a very natural interpretation in terms of minimising a description length

Conclusions

- Information theory has regularly been used in machine learning
- It requires some understanding and care to do it properly
- The KL-divergence (or relative entropy) is often used to make two probability distribution more alike
- The minimum description length is a powerful principle for model selection
- Variational Auto-Encoders have a very natural interpretation in terms of minimising a description length

Conclusions

- Information theory has regularly been used in machine learning
- It requires some understanding and care to do it properly
- The KL-divergence (or relative entropy) is often used to make two probability distribution more alike
- The minimum description length is a powerful principle for model selection
- Variational Auto-Encoders have a very natural interpretation in terms of minimising a description length

Conclusions

- Information theory has regularly been used in machine learning
- It requires some understanding and care to do it properly
- The KL-divergence (or relative entropy) is often used to make two probability distribution more alike
- The minimum description length is a powerful principle for model selection
- Variational Auto-Encoders have a very natural interpretation in terms of minimising a description length

Conclusions

- Information theory has regularly been used in machine learning
- It requires some understanding and care to do it properly
- The KL-divergence (or relative entropy) is often used to make two probability distribution more alike
- The minimum description length is a powerful principle for model selection
- Variational Auto-Encoders have a very natural interpretation in terms of minimising a description length