Deep Continuous Latent Variable Models

Wilker Aziz
ILLC @ UvA

Outline

Generative Model with NN Likelihood

Goal
Define a model $p(x, z|\theta) = p(x|z, \theta)p(z)$ where the likelihood $p(x|z, \theta)$ is given by a neural network.

We fix $p(z)$ for simplicity.

It's not difficult to have $p(z)$ depend on θ.

Similarly, it's not difficult to introduce some predictors. For example, $p(y, z|x, \theta) = p(y|z, x, \theta)p(z|x, \theta)$. Once we learn all about the basic building block on the slide we will comment on such extensions.

Let's talk about language models

A language model (LM) is **a distribution over the sample space of strings** in a language.

Ideally, a language model is a *tractable* distribution. That is,

- it assigns tractable-to-compute probability $p(x|\theta)$ to an observation $x = \langle x_1, \ldots, x_n \rangle$;
- we would know how to sample random sequences from the LM.

If the language has finite vocabulary, we may choose to model observations as

$$X_i|\theta, x_{<i} \sim \mathrm{Cat}(f(x_{<i}; \theta))$$

and note this would satisfy both desiderata.

Let Σ stand for the vocabulary of a language of interest. Then the sample space of a random sequence X is a set $\mathcal{X} \subseteq \Sigma^*$. Each step of a sequence is a random variable X_i that takes on values in Σ.

If an LM factorises *autoregressively* (i.e., from left-to-right without Markov assumptions)

$$p(x|\theta) = \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)$$

and its conditional distributions $X_i|\theta, x_{<i}$ are known (both pmf and cdf) then we can always assess the likelihood of an observation and we can always sample random sequences (i.e., via *ancestral sampling*).

We use $x_{<i}$ to denote a prefix sequence, typically empty for $i = 1$. Though some prefer to think that $x_{<i}$ contains a *beginning-of-sentence symbol* at a fictitious 0th position (this position does not count towards the length of the sequence).

Can you see why the statistical model we propose requires the vocabulary of the language to be finite?

A rather general LM

Built upon an exact factorisation of the joint probability

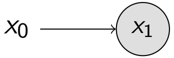
x_0

We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0 \rangle$.

A rather general LM

Built upon an exact factorisation of the joint probability

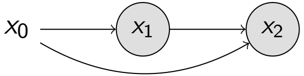


We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.

A rather general LM

Built upon an exact factorisation of the joint probability

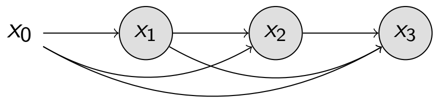


We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.
- From which we can sample the second word x_2. We repeat that process obtaining $X_3|\langle x_0, x_1, x_2\rangle$.

A rather general LM

Built upon an exact factorisation of the joint probability

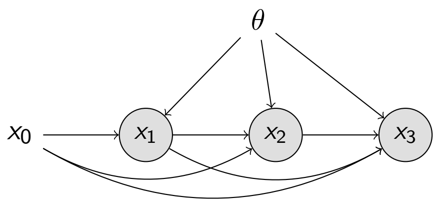


We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.
- From which we can sample the second word x_2. We repeat that process obtaining $X_3|\langle x_0, x_1, x_2\rangle$.
- From which we sample x_3. Let's suppose this is some end-of-sentence token, whose presence triggers the end of the generation process.

A rather general LM

Built upon an exact factorisation of the joint probability



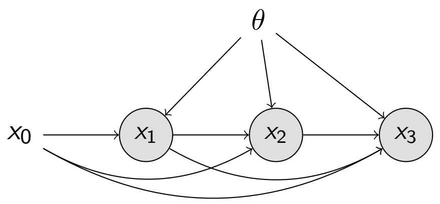
$$p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$$

We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.
- From which we can sample the second word x_2. We repeat that process obtaining $X_3|\langle x_0, x_1, x_2\rangle$.
- From which we sample x_3. Let's suppose this is some end-of-sentence token, whose presence triggers the end of the generation process.
- This model assigns probability $p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$ to the draw.

A rather general LM

Built upon an exact factorisation of the joint probability



$$p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$$

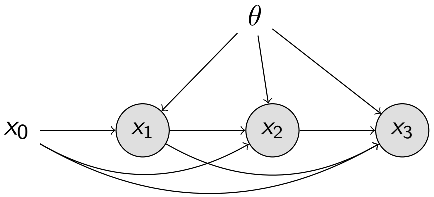
$$p(x|\theta) = \prod_{i=1}^{|x|} \mathrm{Cat}(x_i|f(x_{<i}; \theta))$$

We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.
- From which we can sample the second word x_2. We repeat that process obtaining $X_3|\langle x_0, x_1, x_2\rangle$.
- From which we sample x_3. Let's suppose this is some end-of-sentence token, whose presence triggers the end of the generation process.
- This model assigns probability $p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$ to the draw.
- And if we model with finite vocabulary, each cpd is a Categorical distribution whose parameter we can predict with a neural network (e.g., an LSTM, a Transformer).

A rather general LM

Built upon an exact factorisation of the joint probability



$$p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$$

$$p(x|\theta) = \prod_{i=1}^{|x|} \text{Cat}(x_i|f(x_{<i};\theta))$$

We can estimate θ to maximise the log-likelihood of a dataset of observations.

We can construct an LM that is extremely flexible (as a distribution). It generates data as follows:

- Let's say we start from a deterministic beginning-of-sentence symbol x_0, which we condition on to get a distribution $X_1|\langle x_0\rangle$.
- From which we can sample the first word x_1. We extend the observed prefix by this word and obtain another conditional distribution, namely, $X_2|\langle x_0, x_1\rangle$.
- From which we can sample the second word x_2. We repeat that process obtaining $X_3|\langle x_0, x_1, x_2\rangle$.
- From which we sample x_3. Let's suppose this is some end-of-sentence token, whose presence triggers the end of the generation process.
- This model assigns probability $p(x|\theta) = \prod_{i=1}^{|x|} p(x_i|x_{<i}, \theta)$ to the draw.
- And if we model with finite vocabulary, each cpd is a Categorical distribution whose parameter we can predict with a neural network (e.g., an LSTM, a Transformer).

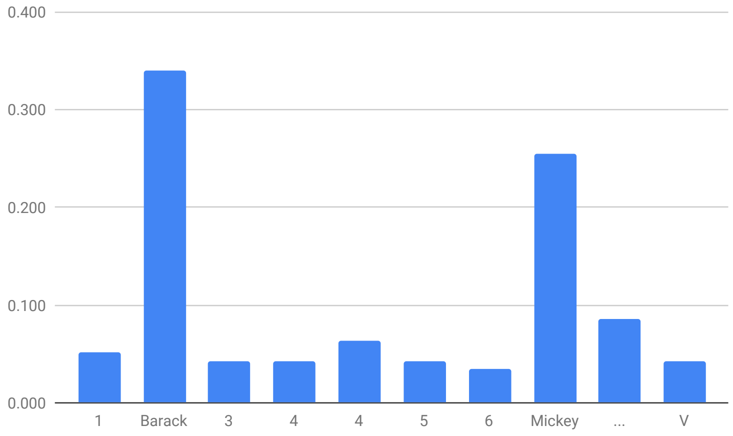
Are all sentences born equal?

The typical LM, illustrated previously, is also known as an *autoregressive* model. It factorises the probability of a sequence one element at a time without making Markov assumptions (i.e., with no conditional independence assumptions).

Every sentence x drawn from this LM conditions on the exact same information (either nothing or just a beginning-of-sentence symbol).

There's no explicit mechanism to structure the probability space in any particular way. That is, there is no partitioning of the sample space into groups of outcomes.

Are there two Donalds?



$$f(x_{<i} = \text{Perhaps Donald met with}; \theta)$$

Generating from this LM will generate sentences about the politician and about the Disney character roughly as often. And indeed our dataset contained roughly the same number of sentences about Donal Trump and Donald Fauntleroy Duck, with a slight win for the real-world Donald.

How can we get the model to disentangle two Donalds?

- One answer might be: *give me more context!* Indeed there are people going that way. Some famous NN LMs condition on ever longer excerpts of text called *prompts*.
- But I gave you a prompt. It reads *Perhaps Donald met with*. Betting that prompts will grow more and more specific to the point that the conditional $X_i|\theta, X_{<i} = \text{prompt}$ will become deterministic is betting on overfitting, or betting on the memory of your model. Remember, we should expect variance.

Think about this: conditional autoregressive models power applications such as image captioning, machine translation, and summarisation. The prompt in these models is the input predictor (image, source sentence, collection of documents). Would you say that for a given input, there is only one output (caption, translation, summary) that is reasonable?

Conditioning

1. Augment the distribution with unobserved factors z
2. Generate x conditioned on z

We can partition the probability space as to disentangle these two confusable celebrities.

We do that by positing more structure (such as a hierarchy of stochastic steps)

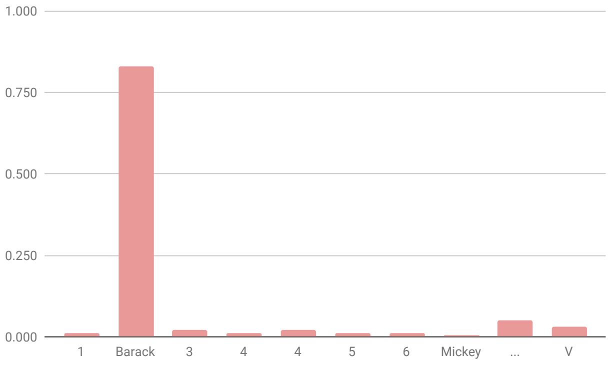
$$Z \sim \mathcal{N}(0, I_D)$$
$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

where for example I introduce D Gaussian-distributed latent factors.

Conditioning

1. Augment the distribution with unobserved factors z
2. Generate x conditioned on z

some factors are all about politics



$$f(z, x_{<i} = \text{Perhaps Donald met with}; \theta)$$

We can partition the probability space as to disentangle these two confusable celebrities.

We do that by positing more structure (such as a hierarchy of stochastic steps)

$$Z \sim \mathcal{N}(0, I_D)$$
$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

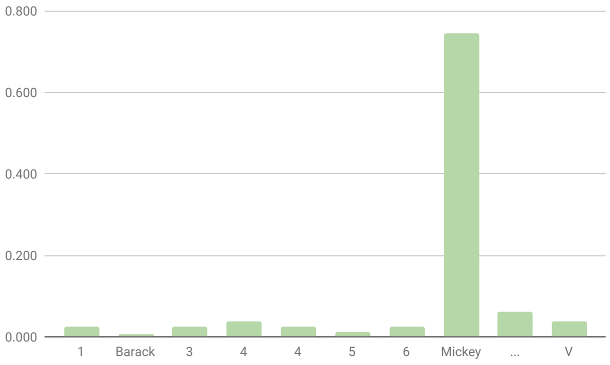
where for example I introduce D Gaussian-distributed latent factors.

- Some factors are all about politics. That is, the conditional $X|Z = z, \theta$ assigns high probability to sentences about politics when they are generated from z in a certain subset of $\mathbb{R}^D$. If $D = 2$, perhaps politics maps from the bottom-left quadrant.

Conditioning

1. Augment the distribution with unobserved factors z
2. Generate x conditioned on z

other factors are all about Disney cartoons



$$f(z, x_{<i} = \text{Perhaps Donald met with}; \theta)$$

We can partition the probability space as to disentangle these two confusable celebrities.

We do that by positing more structure (such as a hierarchy of stochastic steps)

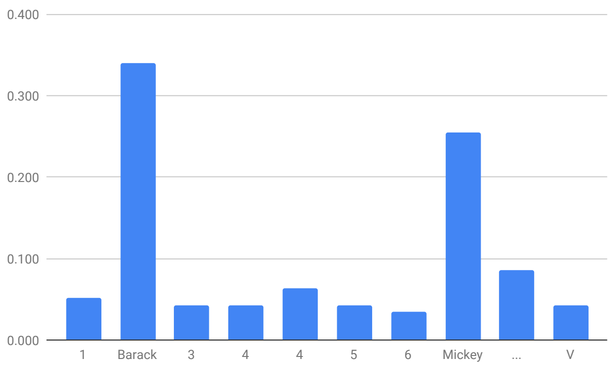
$$Z \sim \mathcal{N}(0, I_D)$$
$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

where for example I introduce D Gaussian-distributed latent factors.

- Some factors are all about politics. That is, the conditional $X|Z = z, \theta$ assigns high probability to sentences about politics when they are generated from z in a certain subset of $\mathbb{R}^D$. If $D = 2$, perhaps politics maps from the bottom-left quadrant.
- Some factors are all about Disney cartoons. That is, $X|Z = z, \theta$ assigns high probability to sentences about Disney cartoons when they are generated from z in another subset of $\mathbb{R}^D$. If $D = 2$, perhaps cartoons map from the top-right quadrant.

Conditioning

1. Augment the distribution with unobserved factors z
2. Generate x conditioned on z



Marginally entangled, but conditionally apart
(think of it as if the model had created imaginary prompts in $\mathbb{R}^D$)

We can partition the probability space as to disentangle these two confusable celebrities.

We do that by positing more structure (such as a hierarchy of stochastic steps)

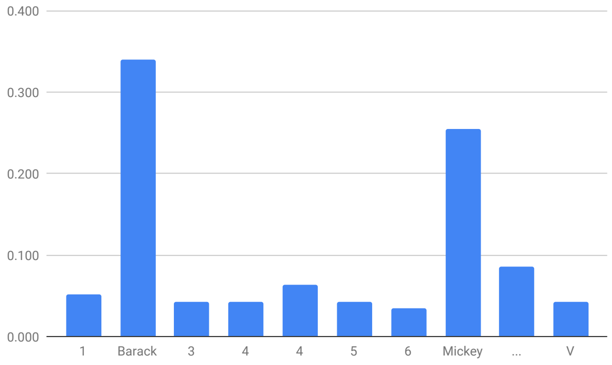
$$Z \sim \mathcal{N}(0, I_D)$$
$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

where for example I introduce D Gaussian-distributed latent factors.

- Some factors are all about politics. That is, the conditional $X|Z = z, \theta$ assigns high probability to sentences about politics when they are generated from z in a certain subset of $\mathbb{R}^D$. If $D = 2$, perhaps politics maps from the bottom-left quadrant.
- Some factors are all about Disney cartoons. That is, $X|Z = z, \theta$ assigns high probability to sentences about Disney cartoons when they are generated from z in another subset of $\mathbb{R}^D$. If $D = 2$, perhaps cartoons map from the top-right quadrant.
- Marginally, we recover the exact distribution we expected: politicians and Disney characters are about as likely to follow.

Conditioning

1. Augment the distribution with unobserved factors z
2. Generate x conditioned on z



Marginally entangled, but conditionally apart
(think of it as if the model had created imaginary prompts in $\mathbb{R}^D$)

We can partition the probability space as to disentangle these two confusable celebrities.

We do that by positing more structure (such as a hierarchy of stochastic steps)

$$Z \sim \mathcal{N}(0, I_D)$$
$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

where for example I introduce D Gaussian-distributed latent factors.

- Some factors are all about politics. That is, the conditional $X|Z = z, \theta$ assigns high probability to sentences about politics when they are generated from z in a certain subset of $\mathbb{R}^D$. If $D = 2$, perhaps politics maps from the bottom-left quadrant.
- Some factors are all about Disney cartoons. That is, $X|Z = z, \theta$ assigns high probability to sentences about Disney cartoons when they are generated from z in another subset of $\mathbb{R}^D$. If $D = 2$, perhaps cartoons map from the top-right quadrant.
- Marginally, we recover the exact distribution we expected: politicians and Disney characters are about as likely to follow.

Latent variable LMs

With latent variables we can model the data as draws from a complex marginal, which mixes (simpler) conditionals from different points in space

$$p(x|\theta) = \int p(x, z|\theta)\mathrm{d}z = \int p(z)p(x|z, \theta)\mathrm{d}z$$
$$= \int p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)\mathrm{d}z$$

Good training can lead to considerable amount of structure in the posterior

$$p(z|x, \theta) = \frac{p(z)p(x|z, \theta)}{p(x|\theta)}$$

A joint distribution $p(x, z|\theta) = p(z)p(x|z, \theta)$ describes a stochastic mapping from latent space to data space. The conditional $X|\theta, z$ can exploit statistical dependencies (think intuitively as correlations) in data space, and thus map certain patterns in data space to certain patterns in latent space. For example, certain lexical correlations we usually think of as topical could be more pronounced in data space whenever we sample from a specific subset of $\mathbb{R}^D$.

If this structure exists, it exists *in the joint distribution*. Then, the *posterior distribution* is our way to appreciate such structure. It is the mechanism to inspect what kind of patterns the model exploits. These patterns are data-driven and they need not be self-evident. Sometimes inspection can suggest that our latent variables capture topical or syntactic patterns, for example, but properly controlling for that is a different story, one that we can only begin to discuss after we learn how to model with latent variables.

Remember: the *true* posterior is nothing but a consequence of the joint distribution. In other words, we do not predict true posteriors independently, rather, we predict joint distributions and infer posteriors once we are given some observations.

Summary

Goal Define a model $p(x, z|\theta) = p(x|z, \theta)p(z)$ where the likelihood $p(x|z, \theta)$ is given by a neural network and Z is a continuous rv.

Motivations

- Inductive bias (e.g., a hierarchy of steps that promotes certain patterns to be captured or that is amenable to inspection).
- Expressiveness: for a choice of family $X|\theta, z$ the marginal of $X, Z|\theta$ is typically more expressive than the conditional $X|\theta, z$.
- Controllable generation: generating from $X|\theta, Z = z$ for z sampled from $Z|\theta, X = x$ generates data that are related to x (at least in latent space, but ideally also in data space).

Problem Intractability of the marginal likelihood $p(x|\theta) = \int p(z)p(x|z, \theta)\mathrm{d}z$

We fix $p(z)$ for simplicity, but we will revisit this decision towards the end.

There are other reasons for modelling with continuous latent variables, these are some that come to mind. Can you think of other reasons?

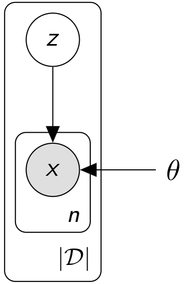
A language model is only one example, there are many more. Can you think of some?

Intractable marginals have impact on how we estimate parameters for the model, but potentially also on other practical aspects. Take the latent-variable LM as an example:

- we can sample from the marginal via ancestral sampling: sample z, condition on it and predict the distribution $X|\theta, z$, sample x.
- but we cannot assess the marginal likelihood $p(x|\theta)$ of a given sample

Outline

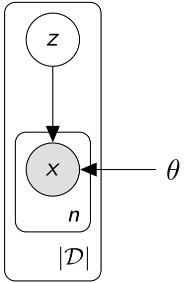
A Latent Variable Document Model



Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

We will take a unigram document model as an example model.

A Latent Variable Document Model



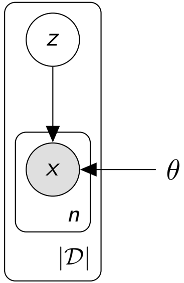
Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.

A Latent Variable Document Model



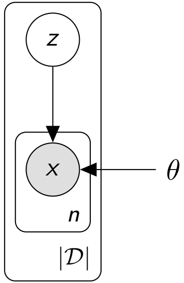
Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \text{Cat}(f(z; \theta))$

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.
- The likelihood is Categorical and fully factorised. We could have used an autoregressive likelihood (an LM), but we'll leave that as exercise.

A Latent Variable Document Model



Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

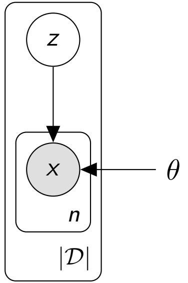
- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z; \theta))$

Designing the generative network

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.
- The likelihood is Categorical and fully factorised. We could have used an autoregressive likelihood (an LM), but we'll leave that as exercise.
- Any parameterisation will do as long as we predict a valid Categorical parameter
  - e.g., a single-hidden layer FFNN with softmax output

A Latent Variable Document Model



Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

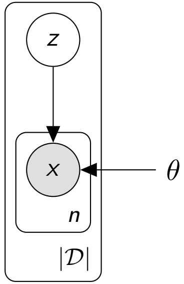
Designing the generative network

$$h = \tanh(W_1 z + b_1)$$

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.
- The likelihood is Categorical and fully factorised. We could have used an autoregressive likelihood (an LM), but we'll leave that as exercise.
- Any parameterisation will do as long as we predict a valid Categorical parameter
  - e.g., a single-hidden layer FFNN with softmax output

A Latent Variable Document Model



Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

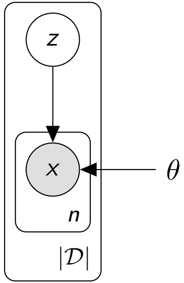
Designing the generative network

$$h = \tanh(W_1 z + b_1)$$
$$f(z, \theta) = \mathrm{softmax}(W_2 h + b_2)$$

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.
- The likelihood is Categorical and fully factorised. We could have used an autoregressive likelihood (an LM), but we'll leave that as exercise.
- Any parameterisation will do as long as we predict a valid Categorical parameter
  - e.g., a single-hidden layer FFNN with softmax output

A Latent Variable Document Model



Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

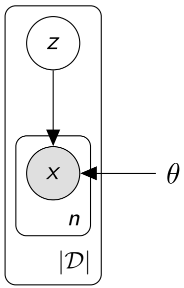
Designing the generative network

$$\begin{aligned} h &= \tanh(W_1 z + b_1) \\ f(z,\theta) &= \mathrm{softmax}(W_2 h + b_2) \\ \theta &= \{W_1, b_1, W_2, b_2\} \end{aligned}$$

We will take a unigram document model as an example model.

- The prior over D-dimensional document embeddings is a standard Gaussian. We denote the prior by $p(z|\alpha)$.
- The likelihood is Categorical and fully factorised. We could have used an autoregressive likelihood (an LM), but we'll leave that as exercise.
- Any parameterisation will do as long as we predict a valid Categorical parameter
  - e.g., a single-hidden layer FFNN with softmax output

Document Model - Likelihood



Likelihood

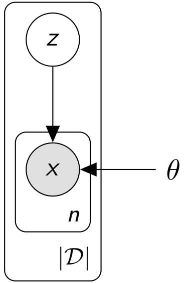
Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

$$p(x|z,\theta) = \prod_{i=1}^{n} p(x_i|z,\theta)$$

The likelihood we prescribe is clearly tractable. That is, for a given x, z, we can assess $p(x|z,\theta)$ without trouble.

Document Model - Likelihood



Likelihood

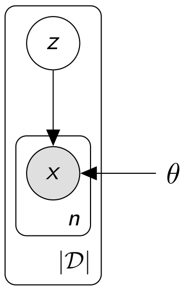
Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

$$p(x|z,\theta) = \prod_{i=1}^{n} p(x_i|z,\theta) = \prod_{i=1}^{n} \mathrm{Cat}(x_i|\underbrace{f(z;\theta)}_{=\pi})$$

The likelihood we prescribe is clearly tractable. That is, for a given x, z, we can assess $p(x|z,\theta)$ without trouble.

Document Model - Likelihood



Likelihood

Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

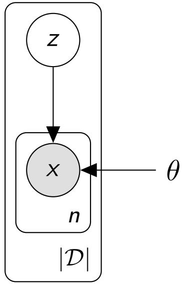
- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

$$p(x|z,\theta) = \prod_{i=1}^{n} p(x_i|z,\theta) = \prod_{i=1}^{n} \mathrm{Cat}(x_i|\underbrace{f(z;\theta)}_{=\pi})$$

$$= \prod_{i=1}^{n} \pi_{x_i}$$

The likelihood we prescribe is clearly tractable. That is, for a given x, z, we can assess $p(x|z,\theta)$ without trouble.

Document Model - Marginal Likelihood



Marginal

Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

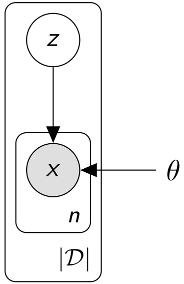
$$p(x|\theta) = \int p(z) \prod_{i=1}^{n} p(x_i|z,\theta)\,\mathrm{d}z$$

The marginal distribution is intractable.

If the model was constrained to a specific length n, then the marginal would be a distribution over strings of length n, and therefore a Gibbs distribution would fit the bill. However, we cannot assess its parameter, since it takes marginalising Z out. Interestingly, whereas X_i are independent given z, that is, they are independent in the likelihood, they are all dependent of one another in the marginal likelihood. In this case, the latent variable model leads to a marginal distribution that is more structured (i.e., captures correlations) than the likelihood (which is fully factorised).

In other words, this model is not really a unigram document model. While the likelihood (that is, given z) is indeed a distribution over independent unigrams, the marginal likelihood is a distribution over sets of unigrams.

Document Model - Marginal Likelihood



Marginal

Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

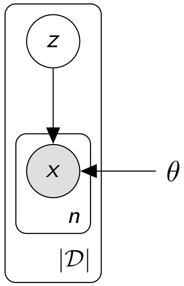
$$
\begin{aligned}
p(x|\theta) &= \int p(z) \prod_{i=1}^{n} p(x_i|z,\theta)\,\mathrm{d}z \\
&= \int \mathcal{N}(z|0, I) \prod_{i=1}^{n} \mathrm{Cat}(x_i|f(z;\theta))\,\mathrm{d}z
\end{aligned}
$$

The marginal distribution is intractable.

If the model was constrained to a specific length n, then the marginal would be a distribution over strings of length n, and therefore a Gibbs distribution would fit the bill. However, we cannot assess its parameter, since it takes marginalising Z out. Interestingly, whereas X_i are independent given z, that is, they are independent in the likelihood, they are all dependent of one another in the marginal likelihood. In this case, the latent variable model leads to a marginal distribution that is more structured (i.e., captures correlations) than the likelihood (which is fully factorised).

In other words, this model is not really a unigram document model. While the likelihood (that is, given z) is indeed a distribution over independent unigrams, the marginal likelihood is a distribution over sets of unigrams.

Document Model - Posterior



Posterior

Generative story of a document $x = \langle x_1, \ldots, x_n \rangle$

- Draw a document embedding $Z \sim \mathcal{N}(0, I_D)$
- Paramaterise a Categorical distribution and draw n words $X_i|\theta, z \sim \mathrm{Cat}(f(z;\theta))$

$$p(z|x,\theta) = \frac{p(x,z|\theta)}{p(x|\theta)}$$

As a consequence of having an intractable marginal, we have an intractable posterior. Moreover, in this case, we have no clue what the posterior family is.

Since the prior $\mathcal{N}(0, I_D)$ gives support to the whole of $\mathbb{R}^D$, and the likelihood $X|z$ is strictly positive for any given z, we know that the posterior must be a distribution over the whole of $\mathbb{R}^D$. Except for trivially uninteresting models where $Z \perp X|\theta$, we also know that Z_d are all dependent on one another in the posterior. But that is really all we know.

Variational Inference

We can lowerbound an intractable marginal

$$\begin{aligned}\log p(x|\theta) &\geq \overbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x,z|\theta)\right] + \mathbb{H}\left(q(z|x,\lambda)\right)}^{\mathrm{ELBO}_x(\lambda,\theta)} \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta) + \log p(z)\right] + \mathbb{H}\left(q(z|x,\lambda)\right) \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)\end{aligned}$$

We have already developed a technique to deal with intractable marginals, namely, variational inference.

- We shall introduce an approximate posterior $q(z|x,\lambda)$ which is independently parameterised and tractable (we know how to sample from it and we can assess the density of samples). This approximation can be used to obtain a lowerbound on the evidence (ELBO).

Variational Inference

We can lowerbound an intractable marginal

$$\begin{aligned}\log p(x|\theta) &\geq \overbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x,z|\theta)\right] + \mathbb{H}\left(q(z|x,\lambda)\right)}^{\text{ELBO}_x(\lambda,\theta)} \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta) + \log p(z)\right] + \mathbb{H}\left(q(z|x,\lambda)\right) \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)\end{aligned}$$

And estimate parameters that maximise the bound

$$\underset{\theta,\lambda}{\arg\max}\ \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$$

We have already developed a technique to deal with intractable marginals, namely, variational inference.

- We shall introduce an approximate posterior $q(z|x,\lambda)$ which is independently parameterised and tractable (we know how to sample from it and we can assess the density of samples). This approximation can be used to obtain a lowerbound on the evidence (ELBO).
- We then optimise the ELBO with respect to our choice of $q(z|x,\lambda)$, in a certain tractable parametric family, and $p(x,z|\theta)$, also in a certain parametric family.

Variational Inference

We can lowerbound an intractable marginal

$$\begin{aligned}\log p(x|\theta) &\geq \overbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x,z|\theta)\right] + \mathbb{H}\left(q(z|x,\lambda)\right)}^{\text{ELBO}_x(\lambda,\theta)} \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta) + \log p(z)\right] + \mathbb{H}\left(q(z|x,\lambda)\right) \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)\end{aligned}$$

And estimate parameters that maximise the bound

$$\arg\max_{\theta,\lambda}\ \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$$

As we get to choose $q(z|x,\lambda)$, we can pick it such that

- MC estimation of $\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right]$ is possible
- and perhaps $\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$ is known in closed form true for exponential families

We have already developed a technique to deal with intractable marginals, namely, variational inference.

- We shall introduce an approximate posterior $q(z|x,\lambda)$ which is independently parameterised and tractable (we know how to sample from it and we can assess the density of samples). This approximation can be used to obtain a lowerbound on the evidence (ELBO).
- We then optimise the ELBO with respect to our choice of $q(z|x,\lambda)$, in a certain tractable parametric family, and $p(x,z|\theta)$, also in a certain parametric family.
- We then approximate expectations via sampling from the tractable approximate posterior.

Variational Inference

We can lowerbound an intractable marginal

$$\begin{aligned}\log p(x|\theta) &\geq \overbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x,z|\theta)\right] + \mathbb{H}\left(q(z|x,\lambda)\right)}^{\mathrm{ELBO}_x(\lambda,\theta)} \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta) + \log p(z)\right] + \mathbb{H}\left(q(z|x,\lambda)\right) \\ &= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)\end{aligned}$$

And estimate parameters that maximise the bound

$$\arg\max_{\theta,\lambda}\ \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$$

As we get to choose $q(z|x,\lambda)$, we can pick it such that

- MC estimation of $\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right]$ is possible
- and perhaps $\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$ is known in closed form true for exponential families

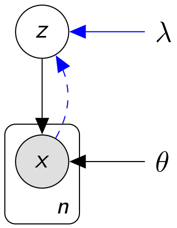
We have already developed a technique to deal with intractable marginals, namely, variational inference.

- We shall introduce an approximate posterior $q(z|x,\lambda)$ which is independently parameterised and tractable (we know how to sample from it and we can assess the density of samples). This approximation can be used to obtain a lowerbound on the evidence (ELBO).
- We then optimise the ELBO with respect to our choice of $q(z|x,\lambda)$, in a certain tractable parametric family, and $p(x,z|\theta)$, also in a certain parametric family.
- We then approximate expectations via sampling from the tractable approximate posterior.

Document Model - Approximate Posterior

Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$



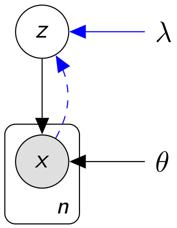
VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

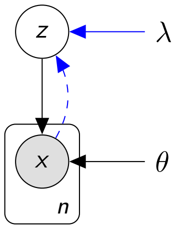
The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \text{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\text{supp}(q(z|x,\lambda)) \subseteq \text{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

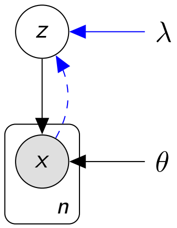
The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

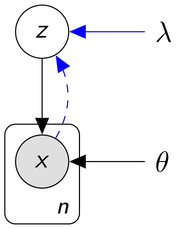
The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \operatorname{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

$$\mu(x;\lambda) = M_2 h + c_2$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\operatorname{supp}(q(z|x,\lambda)) \subseteq \operatorname{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

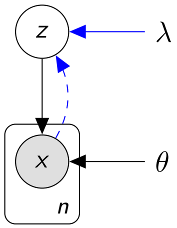
The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$
$$h = \tanh(M_1 s + c_1)$$

$$\mu(x;\lambda) = M_2 h + c_2$$
$$\sigma(x;\lambda) = \mathrm{softplus}(M_3 h + c_3)$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

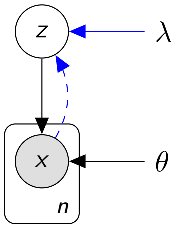
The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

Document Model - Approximate Posterior



Inference model

- $Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

$$\mu(x;\lambda) = M_2 h + c_2$$

$$\sigma(x;\lambda) = \mathrm{softplus}(M_3 h + c_3)$$

$$\lambda = \{E, M_1^3, c_1^3\}$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z|\alpha))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$.

For an arbitrary choice of likelihood $X|\theta, z$ and prior over Z, we have no clue what the true posterior family really is. This is unlike the mixture model, where the posterior was a Categorical distribution (whose parameter was tractable to compute), and unlike the latent binary factor model, where the posterior was a Gibbs distribution (whose parameter was intractable).

A Gaussian is just convenient (as we shall see) and, beyond respecting the support constraint, it's not really motivated by the choice of prior

Any parameterisation will do, as long as we predict valid Gaussian parameters. For example: embed words, average them, and predict locations and scales using a shared hidden layer, they each take an affine transformation, but the scales are softplus-activated for strict-positivity.

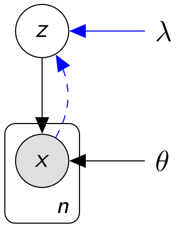
Document Model - ELBO

Generative model

- $Z|\alpha \sim \mathcal{N}(0, I_D)$
- $X_i|\theta, z \sim \text{Cat}(\underbrace{f(z;\theta)}_{=\pi})$ for $i = 1, \ldots, n$

Inference model

- $Z|\lambda, x \sim \mathcal{N}(\underbrace{\mu(x;\lambda)}_{=u}, \text{diag}(\underbrace{\sigma^2(x;\lambda)}_{=s^2}))$



ELBO optimisation

$$\arg\max_{\lambda,\theta} \mathbb{E}_{\mathcal{N}(u,s^2)}\left[\sum_{i=1}^n \log \pi_{x_i}\right] - \underbrace{\text{KL}\left(\mathcal{N}(u, s^2) \mid\mid \mathcal{N}(0, I)\right)}_{\frac{-1}{2}\sum_{d=1}^D \left(1+\log\left(s_d^2\right)-u_d^2-s_d^2\right)}$$

We then seek a choice of λ and θ that optimises a lowerbound on the log-evidence, the ELBO.

Note how the KL divergence from the prior to the approximate posterior is known in closed-form. That is generally the case when the prior and the approximate posterior are in the same exponential family.

Parameter Estimation

We need gradients for parameter updates

$$\boldsymbol{\nabla}_{\lambda,\theta}\,\mathrm{ELBO}_x(\lambda,\theta)$$

And if we cannot get exact gradients, an unbiased gradient estimator

$$\boldsymbol{\nabla}_{\lambda,\theta}\,\mathrm{ELBO}_x(\lambda,\theta) = \mathbb{E}[\boldsymbol{\nabla}_{\lambda,\theta} S_x(\lambda,\theta,Z)]$$

is just as good, as we can use MC to estimate the gradient.

As usual, we count on gradient-based optimisation. Thus we will have to look into how to estimate gradients.

$\boldsymbol{\nabla}_{\lambda,\theta} S_x(\lambda,\theta,Z)$ is called a gradient estimator.

$S_x(\lambda,\theta,Z)$ is called a *stochastic surrogate* objective, the expected value of its gradient $\boldsymbol{\nabla}_{\lambda,\theta}$ is the gradient of the objective we seek to minimise.

Some stochastic surrogates have a remarkable resemblance to the objective function we seek to optimise. For example, $\log p(x|z,\theta)$, for a z sampled from $Z|\lambda,x$, is a stochastic surrogate for the estimation of $\frac{\partial}{\partial\theta}\,\mathrm{ELBO}_x(\lambda,\theta)$. Do you see that?

But don't get too attached to that resemblance. For example, the score function estimator uses a surrogate that does not resemble the ELBO as much, even though it is used to estimate $\frac{\partial}{\partial\lambda}\,\mathrm{ELBO}_x(\lambda,\theta)$.

Updating the generative model

$$\frac{\partial}{\partial \theta}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathsf{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)}^{\text{constant wrt } \theta}\right)$$

Updating the generative model is actually rather simple

- The second term is constant in this case, and poses no challenge. Even if it depended on θ, that is, if the prior depended on θ, as long as we can evaluate the KL term, autodiff would differentiate it for us. The first term seems less obvious, after all, we cannot solve the expected value in closed-form (it would take a sum over $z \in \mathcal{Z}$, and avoiding this sum is the whole point).

Updating the generative model

$$\frac{\partial}{\partial\theta}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)}^{\text{constant wrt } \theta}\right)$$

$$= \underbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\frac{\partial}{\partial\theta}\log p(x|z,\theta)\right]}_{\text{expected gradient :)}}$$

Updating the generative model is actually rather simple

- The second term is constant in this case, and poses no challenge. Even if it depended on θ, that is, if the prior depended on θ, as long as we can evaluate the KL term, autodiff would differentiate it for us. The first term seems less obvious, after all, we cannot solve the expected value in closed-form (it would take a sum over $z \in \mathcal{Z}$, and avoiding this sum is the whole point).
- But note that the distribution we take expectations with respect to is the inference model $q(z|x,\lambda)$, which does not depend on θ. As derivatives are linear, we compute an expected derivative instead of differentiating an expected value.

Updating the generative model

$$\frac{\partial}{\partial \theta}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)}^{\text{constant wrt } \theta}\right)$$

$$= \underbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\frac{\partial}{\partial \theta}\log p(x|z,\theta)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S}\frac{\partial}{\partial \theta}\log p(x|z^{(s)},\theta) \quad \text{where } z^{(s)} \sim q(z|x,\lambda)$$

Updating the generative model is actually rather simple

- The second term is constant in this case, and poses no challenge. Even if it depended on θ, that is, if the prior depended on θ, as long as we can evaluate the KL term, autodiff would differentiate it for us. The first term seems less obvious, after all, we cannot solve the expected value in closed-form (it would take a sum over $z \in \mathcal{Z}$, and avoiding this sum is the whole point).
- But note that the distribution we take expectations with respect to is the inference model $q(z|x,\lambda)$, which does not depend on θ. As derivatives are linear, we compute an expected derivative instead of differentiating an expected value.
- Expected values are great for we know how to estimate them without bias. More often than not we use a single sample per observation.

Updating the generative model

$$\frac{\partial}{\partial\theta}\left(\mathbb{E}_{q(z|x,\lambda)}[\log p(x|z,\theta)] - \overbrace{\mathrm{KL}\left(q(z|x,\lambda)\ ||\ p(z)\right)}^{\text{constant wrt } \theta}\right)$$

$$= \underbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\frac{\partial}{\partial\theta}\log p(x|z,\theta)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S}\frac{\partial}{\partial\theta}\log p(x|z^{(s)},\theta) \quad \text{where } z^{(s)} \sim q(z|x,\lambda)$$

Monte Carlo (MC) estimation gives us a gradient estimate with a computation that does not depend on the size of $\mathcal{Z}$.

Updating the generative model is actually rather simple

- The second term is constant in this case, and poses no challenge. Even if it depended on θ, that is, if the prior depended on θ, as long as we can evaluate the KL term, autodiff would differentiate it for us. The first term seems less obvious, after all, we cannot solve the expected value in closed-form (it would take a sum over $z \in \mathcal{Z}$, and avoiding this sum is the whole point).
- But note that the distribution we take expectations with respect to is the inference model $q(z|x,\lambda)$, which does not depend on θ. As derivatives are linear, we compute an expected derivative instead of differentiating an expected value.
- Expected values are great for we know how to estimate them without bias. More often than not we use a single sample per observation.

Updating the inference model

$$\frac{\partial}{\partial \lambda}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathrm{KL}\left(q(z|x,\lambda)\ ||\ p(z)\right)}^{\text{analytical}}\right)$$

Updating the inference model is not as simple

- The KL term is tractable to assess, thus autodiff will handle it, and we don't need to worry about the exact form of the gradient.

Updating the inference model

$$\frac{\partial}{\partial \lambda}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathsf{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)}^{\text{analytical}}\right)$$

$$= \frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \underbrace{\frac{\partial}{\partial \lambda}\mathsf{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)}_{\text{analytical computation}}$$

Updating the inference model is not as simple

- The KL term is tractable to assess, thus autodiff will handle it, and we don't need to worry about the exact form of the gradient.
- The first term requires an intractable sum over $z \in \mathcal{Z}$ which we mean to avoid. Unfortunately this time we cannot simply 'push' the derivative inside as the expectation is taken w.r.t. $q(z|x,\lambda)$, which clearly depends on λ.

Updating the inference model

$$\frac{\partial}{\partial\lambda}\left(\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \overbrace{\mathsf{KL}\left(q(z|x,\lambda) \,||\, p(z)\right)}^{\text{analytical}}\right)$$

$$= \frac{\partial}{\partial\lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \underbrace{\frac{\partial}{\partial\lambda}\mathsf{KL}\left(q(z|x,\lambda) \,||\, p(z)\right)}_{\text{analytical computation}}$$

The first term again requires approximation by sampling, but the measure of integration depends on the parameter λ.

Updating the inference model is not as simple

- The KL term is tractable to assess, thus autodiff will handle it, and we don't need to worry about the exact form of the gradient.
- The first term requires an intractable sum over $z \in \mathcal{Z}$ which we mean to avoid. Unfortunately this time we cannot simply 'push' the derivative inside as the expectation is taken w.r.t. $q(z|x,\lambda)$, which clearly depends on λ.

Score Function Estimator

For discrete LVMs, we developed the score function estimator. Can we do the same here?

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right]$$
$$= \int \frac{\partial}{\partial\lambda}(q(z|x,\lambda)) \log p(x|z,\theta)\mathrm{d}z$$

It turns out we've already seen this form of gradient when we derived the general form of $\nabla_\theta \log p(x|\theta)$ for discrete LVMs. Fortunately, the identities we used still hold for continuous rvs. Technically we are being a little sneaky: there are a few conditions for differentiation under the integral sign (the mathematically inclined may want to check *Leibniz integral rule*), luckily our application satisfies those.

Score Function Estimator

For discrete LVMs, we developed the score function estimator. Can we do the same here?

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right]$$
$$= \int \frac{\partial}{\partial\lambda}(q(z|x,\lambda)) \log p(x|z,\theta)\mathrm{d}z$$
$$= \int q(z|x,\lambda)\frac{\partial}{\partial\lambda}(\log q(z|x,\lambda)) \log p(x|z,\theta)\mathrm{d}z$$

It turns out we've already seen this form of gradient when we derived the general form of $\nabla_\theta \log p(x|\theta)$ for discrete LVMs. Fortunately, the identities we used still hold for continuous rvs. Technically we are being a little sneaky: there are a few conditions for differentiation under the integral sign (the mathematically inclined may want to check *Leibniz integral rule*), luckily our application satisfies those.

- We can use the log identity for derivatives (i.e., $f' = f(\log f)'$) to re-express the sum as an expectation with respect to $q(z|x,\lambda)$.

Score Function Estimator

For discrete LVMs, we developed the score function estimator. Can we do the same here?

$$\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right]$$
$$= \int \frac{\partial}{\partial\lambda}(q(z|x,\lambda)) \log p(x|z,\theta)\mathrm{d}z$$
$$= \int q(z|x,\lambda)\frac{\partial}{\partial\lambda}(\log q(z|x,\lambda)) \log p(x|z,\theta)\mathrm{d}z$$
$$= \underbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\frac{\partial}{\partial\lambda}\log q(z|x,\lambda)\right]}_{\text{expected gradient :)}}$$

It turns out we've already seen this form of gradient when we derived the general form of $\nabla_\theta \log p(x|\theta)$ for discrete LVMs. Fortunately, the identities we used still hold for continuous rvs. Technically we are being a little sneaky: there are a few conditions for differentiation under the integral sign (the mathematically inclined may want to check *Leibniz integral rule*), luckily our application satisfies those.

- We can use the log identity for derivatives (i.e., $f' = f(\log f)'$) to re-express the sum as an expectation with respect to $q(z|x,\lambda)$.
- This estimator is known as the *score function estimator*.

Score Function Estimator

For discrete LVMs, we developed the score function estimator. Can we do the same here?

$$
\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \int \frac{\partial}{\partial\lambda}(q(z|x,\lambda))\log p(x|z,\theta)\mathrm{d}z \\
&= \int q(z|x,\lambda)\frac{\partial}{\partial\lambda}(\log q(z|x,\lambda))\log p(x|z,\theta)\mathrm{d}z \\
&= \underbrace{\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\frac{\partial}{\partial\lambda}\log q(z|x,\lambda)\right]}_{\text{expected gradient :)}}
\end{aligned}
$$

We turned the derivative of an expectation into the expected value of a derivative!

It turns out we've already seen this form of gradient when we derived the general form of $\nabla_\theta \log p(x|\theta)$ for discrete LVMs. Fortunately, the identities we used still hold for continuous rvs. Technically we are being a little sneaky: there are a few conditions for differentiation under the integral sign (the mathematically inclined may want to check *Leibniz integral rule*), luckily our application satisfies those.

- We can use the log identity for derivatives (i.e., $f' = f(\log f)'$) to re-express the sum as an expectation with respect to $q(z|x,\lambda)$.
- This estimator is known as the *score function estimator*.

SFE and its variance

We can now build an MC estimator

$$
\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] \\
&= \mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\frac{\partial}{\partial\lambda}\log q(z|x,\lambda)\right] \\
&\overset{\text{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S}\log p(x|z^{(s)},\theta)\frac{\partial}{\partial\lambda}\log q(z^{(s)}|x,\lambda) \\
&\text{where } z^{(s)} \sim q(z|x,\lambda)
\end{aligned}
$$

Unfortunately, **this one has high variance**. But, as it turns out, **we can do a lot better**!

And, as always, expected gradients can be estimated free of bias via MC.

The high variance of the score function estimator can be intuitively justified by the fact that the learning signal (i.e., the part of the estimator that interacts directly with the observed data) cannot influence the direction of the gradient, rather only its magnitude.

By the way, this is the complete stochastic surrogate objective (for z sampled from $Z|\lambda, x$:

$$
\log p(x|z,\theta) - \overbrace{\mathrm{KL}\left(q(z|x,\lambda)\,||\,p(z)\right)}^{\text{analytical}} - \underbrace{\log p(x|z,\cancel{\theta})}_{\text{'detached'}}\log q(z|x,\lambda)
$$

Can you see that $\nabla_{\lambda,\theta}$ gives us the correct partials?

Outline

Inference Network Gradient

We need to re-express the gradient as an expected value, but the measure of integration depends on λ

$$\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right]$$

What if we could re-express $q(z|x,\lambda)$ in terms of some other distribution that does not depend on λ?

Something like

1. Sample u from a fixed noise source
2. Apply a differentiable transformation $\mathcal{T}^{-1}(u)$ and get $Z|\lambda, x$
   $\mathcal{T}^{-1}$ can depend on any other quantity already available (e.g., λ, x)

Let's think about this proposal. Say we have a univariate random variable Z, it could be $Z \sim \mathcal{N}(\psi_\mu, \psi_{\sigma^2})$ or $Z \sim \text{Gamma}(\psi_\alpha, \psi_\beta)$ amongst many other options, say the parameters are predicted by some NN: e.g., $\psi = g(x;\lambda)$. We are looking for something like this:

$$U \sim \mathcal{U}(0,1)$$
$$\mathcal{T}^{-1}(U) \sim Z|\psi$$

This would make the path from the parameters λ to a sample z deterministic and differentiable given some uniform draw $u \in [0,1]$. If we find such 'magical' transformation that absorbs parameters of a distribution, then $\frac{\partial}{\partial\lambda}\mathbb{E}_{q(z|x,\lambda)}[\log p(x|z,\theta)] = \frac{\partial}{\partial\lambda}\mathbb{E}_{\mathcal{U}(0,1)}[\log p(x|z = \mathcal{T}^{-1}(u|x,\lambda),\theta)]$ and suddenly $\frac{\partial}{\partial\lambda}$ could be 'pushed' inside as it happened for $\frac{\partial}{\partial\theta}$.

Do you know of any such transformation for univariate rvs?

Asking differently, do you know what transformation $\mathcal{T}$ of Z has the property that $\mathcal{T}(Z) \sim \mathcal{U}(0,1)$?

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0, 1)$$

and conversely, for $U \sim \mathcal{U}(0, 1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0,1)$$

and conversely, for $U \sim \mathcal{U}(0,1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

$$\frac{\partial}{\partial \psi}\mathbb{E}_{Z|\psi}[\ell(z)] = \frac{\partial}{\partial \psi}\mathbb{E}_{\mathcal{U}(0,1)}[\ell(\underbrace{F^{-1}(u|\psi)}_{=z})]$$

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

- We can re-parameterise expectations. This is known as the *law of the unconscious statistician*. There's a lot of careful maths going on behind it, if you are curious, check the appendix.

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0,1)$$

and conversely, for $U \sim \mathcal{U}(0,1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

$$\begin{aligned}\frac{\partial}{\partial\psi}\mathbb{E}_{Z|\psi}[\ell(z)] &= \frac{\partial}{\partial\psi}\mathbb{E}_{\mathcal{U}(0,1)}[\ell(\underbrace{F^{-1}(u|\psi)}_{=z})] \\ &= \mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial\psi}\ell(F^{-1}(u|\psi))\right]\end{aligned}$$

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

- We can re-parameterise expectations. This is known as the *law of the unconscious statistician*. There's a lot of careful maths going on behind it, if you are curious, check the appendix.
- Which allows us to re-express the gradient as an expected value.

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0,1)$$

and conversely, for $U \sim \mathcal{U}(0,1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

$$\begin{aligned}\frac{\partial}{\partial\psi}\mathbb{E}_{Z|\psi}[\ell(z)] &= \frac{\partial}{\partial\psi}\mathbb{E}_{\mathcal{U}(0,1)}[\ell(\underbrace{F^{-1}(u|\psi)}_{=z})] \\ &= \mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial\psi}\ell(F^{-1}(u|\psi))\right]\end{aligned}$$

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

- We can re-parameterise expectations. This is known as the *law of the unconscious statistician*. There's a lot of careful maths going on behind it, if you are curious, check the appendix.
- Which allows us to re-express the gradient as an expected value.
- Chain rule requires the derivative of the sample z w.r.t. ψ: $\mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial z}\ell(z)\frac{\partial}{\partial\psi}F^{-1}(u|\psi)\right]$

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0,1)$$

and conversely, for $U \sim \mathcal{U}(0,1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

$$\frac{\partial}{\partial\psi}\mathbb{E}_{Z|\psi}[\ell(z)] = \frac{\partial}{\partial\psi}\mathbb{E}_{\mathcal{U}(0,1)}[\ell(\underbrace{F^{-1}(u|\psi)}_{=z})]$$

$$= \mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial\psi}\ell(F^{-1}(u|\psi))\right]$$

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

- We can re-parameterise expectations. This is known as the *law of the unconscious statistician*. There's a lot of careful maths going on behind it, if you are curious, check the appendix.
- Which allows us to re-express the gradient as an expected value.
- Chain rule requires the derivative of the sample z w.r.t. ψ: $\mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial z}\ell(z)\frac{\partial}{\partial\psi}F^{-1}(u|\psi)\right]$

This looks great. However, it's very easy to find examples of univariate continuous rvs for which the cdf and/or its inverse are unknown. Thus we cannot always count on this method. Moreover, we are often interested in multivariate variables, which would require some form of special treatment.

Reparameterised gradients: Inverse cdf

The cdf $F(z|\psi)$ of a univariate rv $Z|\psi$ is a transformation that *by definition* meets our goals. That is, no matter the distribution of $Z|\psi$,

$$F(Z|\psi) \sim \mathcal{U}(0,1)$$

and conversely, for $U \sim \mathcal{U}(0,1)$

$$F^{-1}(U|\psi) \sim Z|\psi$$

Moreover, the cdf is *by definition* differentiable w.r.t. $z \in \mathcal{Z}$, and often also differentiable w.r.t. the parameters ψ of the pdf of the rv. Then,

$$\frac{\partial}{\partial \psi}\mathbb{E}_{Z|\psi}[\ell(z)] = \frac{\partial}{\partial \psi}\mathbb{E}_{\mathcal{U}(0,1)}[\ell(\underbrace{F^{-1}(u|\psi)}_{=z})]$$
$$= \mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial \psi}\ell(F^{-1}(u|\psi))\right]$$

The cumulative distribution function (cdf) of a continuous rv is a monotonically increasing function, and therefore invertible.

Its inverse $F^{-1}(u|\psi)$ is known as the *quantile function*.

- We can re-parameterise expectations. This is known as the *law of the unconscious statistician*. There's a lot of careful maths going on behind it, if you are curious, check the appendix.
- Which allows us to re-express the gradient as an expected value.
- Chain rule requires the derivative of the sample z w.r.t. ψ: $\mathbb{E}_{\mathcal{U}(0,1)}\left[\frac{\partial}{\partial z}\ell(z)\frac{\partial}{\partial \psi}F^{-1}(u|\psi)\right]$

This looks great. However, it's very easy to find examples of univariate continuous rvs for which the cdf and/or its inverse are unknown. Thus we cannot always count on this method. Moreover, we are often interested in multivariate variables, which would require some form of special treatment.

Reparameterised gradients: Location-scale families

A location-scale family is a group of two-parameters (known as location and scale) continuous distribution such that any member $Z|\mu, \sigma$ of the family can be mapped to and from the *standard* member $\phi(\epsilon)$ via a standardisation procedure:

$$\frac{Z - \mu}{\sigma} \sim \phi(\epsilon)$$
$$\mu + \epsilon\sigma \sim Z|\mu, \sigma$$

Location-scale families include multivariate distributions. Then, for $Z|\mathbf{u}, \mathbf{C}$

$$\mathbf{C}^{-1}(Z - \mathbf{u}) \sim \phi(\epsilon)$$
$$\mathbf{u} + \mathbf{C}\epsilon \sim Z|\mathbf{u}, \mathbf{C}$$

The cdf is not the only way to absorb parameters. Every location scale family has a standard member and every member of the family can be mapped to the standard member via an affine transformation.

Examples: *Gaussian* (as we use in our running example), Gumbel, Laplace, Logistic, Cauchy, Uniform, Student's t. Also their multivariate versions.

Gaussian rvs and the reparameterisation trick

Recall we made a mean field Gaussian assumption

$$Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$. As we saw earlier, this is unlikely to be the case in the true posterior.

Gaussian rvs and the reparameterisation trick

Recall we made a mean field Gaussian assumption

$$Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$$

Then we have

$$\mathcal{T}(z, \lambda; x) = \frac{z - \mu(x;\lambda)}{\sigma(x;\lambda)} = \epsilon \sim \mathcal{N}(0, \mathrm{I})$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$. As we saw earlier, this is unlikely to be the case in the true posterior.

Picking a Gaussian approximation is just convenient for Gaussians can be expressed in terms of the fixed standard Gaussian and this leads to a convenient gradient estimator.

Gaussian rvs and the reparameterisation trick

Recall we made a mean field Gaussian assumption

$$Z|\lambda, x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$$

Then we have

$$\mathcal{T}(z, \lambda; x) = \frac{z - \mu(x;\lambda)}{\sigma(x;\lambda)} = \epsilon \sim \mathcal{N}(0, \mathrm{I})$$

and conversely, for $\epsilon \sim \mathcal{N}(0, I)$, we have:

$$\mathcal{T}^{-1}(\epsilon, \lambda; x) = \mu(x;\lambda) + \sigma(x;\lambda) \odot \epsilon = z \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$$

VI (due to KL) imposes a support constraint on $Z|\lambda, x$: we need $\mathrm{supp}(q(z|x,\lambda)) \subseteq \mathrm{supp}(p(z))$, thus for a prior over $\mathbb{R}^D$, a Gaussian approximation is a valid choice.

The choice on the slide is a product of D independent Gaussians (see the diagonal covariance matrix). This is a **mean field** assumption! That is, in the posterior approximation we assume that $Z_d \perp Z_{d'}|x$ for $d \neq d'$. As we saw earlier, this is unlikely to be the case in the true posterior.

Picking a Gaussian approximation is just convenient for Gaussians can be expressed in terms of the fixed standard Gaussian and this leads to a convenient gradient estimator.

Updating the inference model

$$= \frac{\partial}{\partial \lambda} \int q(z|x, \lambda) \log p(x|z, \theta) \mathrm{d}z$$

So we are back to the board trying to obtain a nice gradient estimate for the inference model.

Updating the inference model

$$
\begin{aligned}
&= \frac{\partial}{\partial \lambda} \int q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z \\
&= \frac{\partial}{\partial \lambda} \int \phi(\epsilon) \log \left(p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z},\theta) \right) \mathrm{d}\epsilon
\end{aligned}
$$

So we are back to the board trying to obtain a nice gradient estimate for the inference model.

- This time around, we will assume this *differentiable and invertible reparameterisation* exists (we know it exists for location-scale families, for example) and use the law of the unconscious statistician instead of score function estimation.

Updating the inference model

$$
\begin{aligned}
&= \frac{\partial}{\partial \lambda} \int q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z \\
&= \frac{\partial}{\partial \lambda} \int \phi(\epsilon) \log \left(p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z}, \theta) \right) \mathrm{d}\epsilon \\
&= \int \phi(\epsilon) \frac{\partial}{\partial \lambda} \left[\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z}, \theta) \right] \mathrm{d}\epsilon
\end{aligned}
$$

So we are back to the board trying to obtain a nice gradient estimate for the inference model.

- This time around, we will assume this *differentiable and invertible reparameterisation* exists (we know it exists for location-scale families, for example) and use the law of the unconscious statistician instead of score function estimation.
- Which allows us to re-express the gradient as an expected value.

Updating the inference model

$$
\begin{aligned}
&= \frac{\partial}{\partial \lambda} \int q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z \\
&= \frac{\partial}{\partial \lambda} \int \phi(\epsilon) \log \left(p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z}, \theta) \right) \mathrm{d}\epsilon \\
&= \int \phi(\epsilon) \frac{\partial}{\partial \lambda} \left[\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z}, \theta) \right] \mathrm{d}\epsilon
\end{aligned}
$$

So we are back to the board trying to obtain a nice gradient estimate for the inference model.

- This time around, we will assume this *differentiable and invertible reparameterisation* exists (we know it exists for location-scale families, for example) and use the law of the unconscious statistician instead of score function estimation.
- Which allows us to re-express the gradient as an expected value.

Updating the inference model

$$
\mathbb{E}_{\phi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z},\theta)\right]
$$

$$
\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z},\theta)
$$

where $\epsilon_i \sim \phi(\epsilon)$

As usual, we estimate expectations via Monte Carlo (MC). But see how this time around we sample from a fixed noise source $\phi(\epsilon)$.

Updating the inference model

$$\mathbb{E}_{\phi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z},\theta)\right]$$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z},\theta)$$

where $\epsilon_i \sim \phi(\epsilon)$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\underbrace{\frac{\partial}{\partial z}\log p(x|z,\theta)\times\frac{\partial}{\partial\lambda}\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z}}_{\text{chain rule}}$$

As usual, we estimate expectations via Monte Carlo (MC). But see how this time around we sample from a fixed noise source $\phi(\epsilon)$.

- Usually, you leave all calls to chain rule to your automatic differentiation algorithm, but sometimes it's informative to look into what it does. Here we see that chain rule will differentiate the actual sample. We are effectively differentiating through the sampling procedure! It's a remarkable feat.

Updating the inference model

$$\mathbb{E}_{\phi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z},\theta)\right]$$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z},\theta)$$

where $\epsilon_i \sim \phi(\epsilon)$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\underbrace{\frac{\partial}{\partial z}\log p(x|z,\theta)\times\frac{\partial}{\partial\lambda}\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z}}_{\text{chain rule}}$$

As usual, we estimate expectations via Monte Carlo (MC). But see how this time around we sample from a fixed noise source $\phi(\epsilon)$.

- Usually, you leave all calls to chain rule to your automatic differentiation algorithm, but sometimes it's informative to look into what it does. Here we see that chain rule will differentiate the actual sample. We are effectively differentiating through the sampling procedure! It's a remarkable feat.
- This technique goes by a few names:
  - *reparameterisation trick* (Kingma and Welling, 2014)
  - stochastic backpropagation (Rezende et al., 2014)
  - reparameterised gradient (Titsias and Lázaro-Gredilla, 2014)

  It's the key technical development in the *variational auto-encoder* (VAE; Kingma and Welling, 2014).

Updating the inference model

$$\mathbb{E}_{\phi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon,\lambda)}^{=z},\theta)\right]$$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z},\theta)$$

where $\epsilon_i \sim \phi(\epsilon)$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\underbrace{\frac{\partial}{\partial z}\log p(x|z,\theta)\times\frac{\partial}{\partial\lambda}\overbrace{\mathcal{T}^{-1}(\epsilon_i,\lambda)}^{=z}}_{\text{chain rule}}$$

As usual, we estimate expectations via Monte Carlo (MC). But see how this time around we sample from a fixed noise source $\phi(\epsilon)$.

- Usually, you leave all calls to chain rule to your automatic differentiation algorithm, but sometimes it's informative to look into what it does. Here we see that chain rule will differentiate the actual sample. We are effectively differentiating through the sampling procedure! It's a remarkable feat.
- This technique goes by a few names:
  - *reparameterisation trick* (Kingma and Welling, 2014)
  - stochastic backpropagation (Rezende et al., 2014)
  - reparameterised gradient (Titsias and Lázaro-Gredilla, 2014)

  It's the key technical development in the *variational auto-encoder* (VAE; Kingma and Welling, 2014).

Derivatives of mean field Gaussian reparameterisation

For our mean field Gaussian approximation we have

$$\mathcal{T}^{-1}(\epsilon, \lambda) = \mu(x; \lambda) + \sigma(x; \lambda) \odot \epsilon$$

We get two gradient paths!

- one is deterministic
  $\frac{\partial \mathcal{T}^{-1}(\epsilon,\lambda)}{\partial \mu(x;\lambda)} = \frac{\partial}{\partial \mu(x;\lambda)}[\mu(x; \lambda) + \sigma(x; \lambda) \odot \epsilon] = 1$
- the other is stochastic
  $\frac{\partial \mathcal{T}^{-1}(\epsilon,\lambda)}{\partial \sigma(x;\lambda)} = \frac{\partial}{\partial \sigma(x;\lambda)}[\mu(x; \lambda) + \sigma(x; \lambda) \odot \epsilon] = \epsilon$

Let us again check what chain rule does for us.

This is the case for every location-scale family, and it generalises as expected to the multivariate case (full-rank covariance matrix).

Can you think about any difficulties in predicting a full-rank covariance matrix with an inference network?

Gaussian KL

Let's get back to the ELBO

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathrm{KL}\left(q(z|x,\lambda) \;||\; p(z)\right)$$

and handle the KL term.

For a standard Gaussian prior and a mean field Gaussian posterior approximation

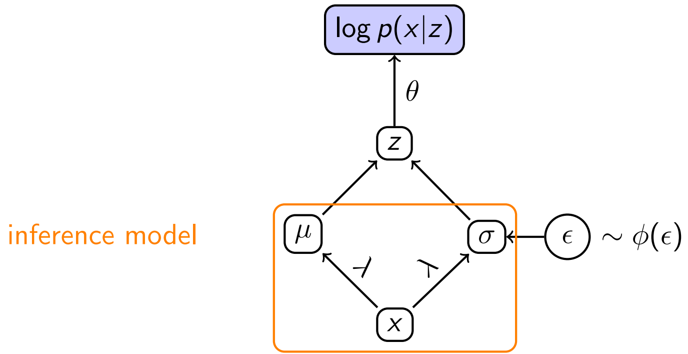
$$-\mathrm{KL}\left(q(z|x,\lambda) \;||\; p(z)\right) = \frac{1}{2}\sum_{d=1}^{D}\left(1 + \log\left(\sigma_d^2\right) - \mu_d^2 - \sigma_d^2\right)$$

KL between two members of the same exponential family is usually known. Sometimes it may involve terms that can only be approximated numerically, or whose derivatives need numerical approximation, but as a general rule our chances are better if we match exponential families.

When KL is not known, we can always see it as an expected value and use reparameterised gradients

$$\begin{aligned}\frac{\partial}{\partial\lambda}\mathrm{KL}\left(q(z|x,\lambda) \;||\; p(z)\right) &= \frac{\partial}{\partial\lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log\frac{q(z|x,\lambda)}{p(z)}\right] \\ &= \mathbb{E}_{\phi(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log\frac{q(z=\mathcal{T}^{-1}(\epsilon,\lambda)|x,\lambda)}{p(z=\mathcal{T}^{-1}(\epsilon,\lambda))}\right]\end{aligned}$$

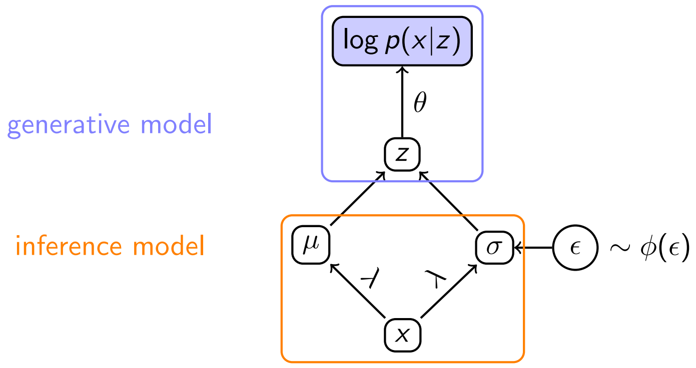
Computation Graph



Let's put everything together in a computation graph

- we map an observation x to the parameters μ and σ of our inference model, this uses an NN with parameters λ; with those we can parameterise an affine transformation that maps samples from a standard location-scale family to samples from the approximate posterior;

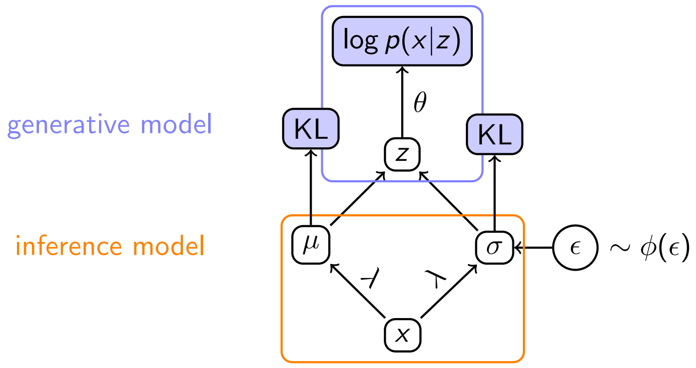
Computation Graph



Let's put everything together in a computation graph

- we map an observation x to the parameters μ and σ of our inference model, this uses an NN with parameters λ; with those we can parameterise an affine transformation that maps samples from a standard location-scale family to samples from the approximate posterior;
- besides, we have our main neural network, which maps from z to the log-probability log $p(x|z, \theta)$, this is a quantity that depends on θ;

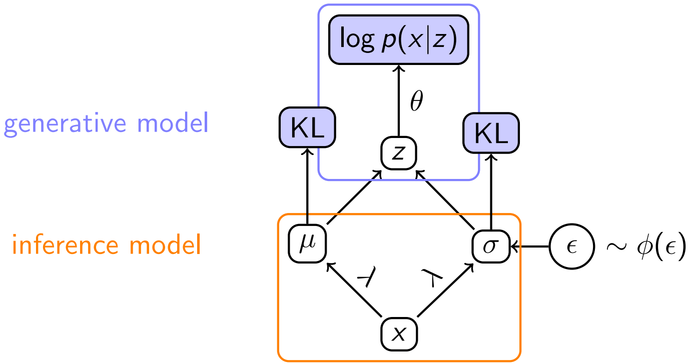
Computation Graph



Let's put everything together in a computation graph

- we map an observation x to the parameters μ and σ of our inference model, this uses an NN with parameters λ; with those we can parameterise an affine transformation that maps samples from a standard location-scale family to samples from the approximate posterior;
- besides, we have our main neural network, which maps from z to the log-probability $\log p(x|z, \theta)$, this is a quantity that depends on θ;
- with μ, σ, and the prior parameter α in this case, we can assess $\mathrm{KL}\left(q(z|x, \lambda) \,||\, p(z|\alpha)\right)$, whose gradient we need in order to update the inference model;

Computation Graph



Let's put everything together in a computation graph

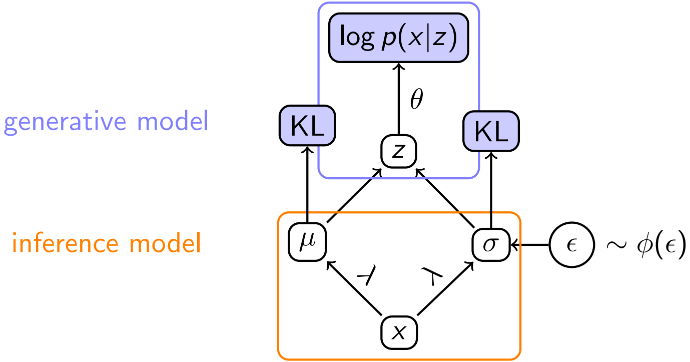
- we map an observation x to the parameters μ and σ of our inference model, this uses an NN with parameters λ; with those we can parameterise an affine transformation that maps samples from a standard location-scale family to samples from the approximate posterior;
- besides, we have our main neural network, which maps from z to the log-probability $\log p(x|z, \theta)$, this is a quantity that depends on θ;
- with μ, σ, and the prior parameter α in this case, we can assess $\mathrm{KL}\left(q(z|x, \lambda) \mid\mid p(z|\alpha)\right)$, whose gradient we need in order to update the inference model;

The surrogate objective resembles a single-sample estimate of the ELBO:

$$\log p(x|z = \mu(x; \lambda) + \epsilon \odot \sigma(x; \lambda)) - \mathrm{KL}\left(q(z|x, \lambda) \mid\mid p(z|\alpha)\right)$$

Can you verify that $\nabla_{\lambda,\theta}$ yields the correct partials?

Computation Graph



Let's put everything together in a computation graph

- we map an observation x to the parameters μ and σ of our inference model, this uses an NN with parameters λ; with those we can parameterise an affine transformation that maps samples from a standard location-scale family to samples from the approximate posterior;
- besides, we have our main neural network, which maps from z to the log-probability $\log p(x|z, \theta)$, this is a quantity that depends on θ;
- with μ, σ, and the prior parameter α in this case, we can assess $\mathrm{KL}\left(q(z|x, \lambda) \mid\mid p(z|\alpha)\right)$, whose gradient we need in order to update the inference model;

The surrogate objective resembles a single-sample estimate of the ELBO:

$$\log p(x|z = \mu(x; \lambda) + \epsilon \odot \sigma(x; \lambda)) - \mathrm{KL}\left(q(z|x, \lambda) \mid\mid p(z|\alpha)\right)$$

Can you verify that $\nabla_{\lambda,\theta}$ yields the correct partials?

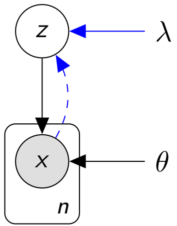
Document Model - Reparameterised ELBO

Generative Model

- Prior: $Z \sim \mathcal{N}(0, I)$
- Likelihood: $X_i|z \sim \mathrm{Cat}(f(z;\theta))$

Inference Model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \mathrm{diag}(\sigma^2(x;\lambda)))$



ELBO

$$\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[\sum_{i=1}^{n}\log \pi_{x_i}\right] - \mathrm{KL}\left(\mathcal{N}(u, s^2)\,||\,\mathcal{N}(0, I)\right)$$

where $u = \mu(x;\lambda)$, $s = \sigma(x;\lambda)$, and $\pi = f(z = u + \epsilon \odot s;\theta)$

And this concludes our example VAE!

A stochastic auto-encoder with a KL regulariser, right?

You could describe it like that. It more or less covers what you should implement, but avoid taking much more than that from it.

Let's see

- The *stochasticity* is not arbitrary, it follows from the need to estimate the log evidence.
- The fact that there's something that looks like an *auto-encoder* is accidental, it just so happens that posteriors condition on data, and likelihoods generate data.
- The *regulariser* is not a post-hoc patch to the objective, but it's an integral part of it.

The stochasticity comes from a choice of approximate posterior, and we want one which is differentiably reparameterisable, and whose support is contained in the support of the prior.

The objective is indeed a bound on the logarithm of the marginal likelihood.

The 'KL regulariser' is not optional. It fell off of the derivation of the ELBO and removing it or scaling it is a heuristic that may or may not lead to something meaningful.

For example, there are alternative variational objectives that are motivated from a view other than bounding the log evidence, those will have objectives other than the ELBO, and they are supported from the point of view of their respective theories. Sometimes, you can find theoretical support to certain strategies that manipulate that KL term, but freely manipulating it without any theoretical support for it being *'a regulariser of some stochastic auto-encoder loss'* is a void motivation.

Marginal likelihood assessments

What if we need to assess $\log p(x|\theta)$ under a deep latent variable model?

For example, that is useful in language modelling and other density estimation problems.

Marginal likelihood assessments

What if we need to assess $\log p(x|\theta)$ under a deep latent variable model?

For example, that is useful in language modelling and other density estimation problems.

Importance sampling fundamental identity

$$\log p(x|\theta) = \log \int p(x, z|\theta)\mathrm{d}z$$

- We start from the definition of the marginal and introduce an importance distribution.

Marginal likelihood assessments

What if we need to assess $\log p(x|\theta)$ under a deep latent variable model?

For example, that is useful in language modelling and other density estimation problems.

Importance sampling fundamental identity

$$\begin{aligned}\log p(x|\theta) &= \log \int p(x, z|\theta)\mathrm{d}z \\ &= \log \int \frac{w(z)}{w(z)} p(x, z|\theta)\mathrm{d}z\end{aligned}$$

- We start from the definition of the marginal and introduce an importance distribution.
- For example, we could use our already trained inference model $w(z) := q(z|x, \lambda)$.

Marginal likelihood assessments

What if we need to assess $\log p(x|\theta)$ under a deep latent variable model?

For example, that is useful in language modelling and other density estimation problems.

Importance sampling fundamental identity

$$\begin{aligned}\log p(x|\theta) &= \log \int p(x, z|\theta)\mathrm{d}z \\ &= \log \int \frac{w(z)}{w(z)} p(x, z|\theta)\mathrm{d}z \\ &= \log \mathbb{E}_{w(z)}\left[\frac{p(x, z|\theta)}{w(z)}\right]\end{aligned}$$

- We start from the definition of the marginal and introduce an importance distribution.
- For example, we could use our already trained inference model $w(z) := q(z|x, \lambda)$.
- This gives us an expectation, which we can estimate via MC using a large sample. The result is a *stochastic lowerbound*, and it gets tighter the more we sample, and it is tight in the limit.

This result is closely-related to the ELBO and can be used also for parameter estimation. See for example (Burda et al., 2016; Cremer et al., 2017).

Marginal likelihood assessments

What if we need to assess $\log p(x|\theta)$ under a deep latent variable model?

For example, that is useful in language modelling and other density estimation problems.

Importance sampling fundamental identity

$$\begin{aligned}\log p(x|\theta) &= \log \int p(x, z|\theta)\mathrm{d}z \\ &= \log \int \frac{w(z)}{w(z)} p(x, z|\theta)\mathrm{d}z \\ &= \log \mathbb{E}_{w(z)}\left[\frac{p(x, z|\theta)}{w(z)}\right]\end{aligned}$$

- We start from the definition of the marginal and introduce an importance distribution.
- For example, we could use our already trained inference model $w(z) := q(z|x, \lambda)$.
- This gives us an expectation, which we can estimate via MC using a large sample. The result is a *stochastic lowerbound*, and it gets tighter the more we sample, and it is tight in the limit.

This result is closely-related to the ELBO and can be used also for parameter estimation. See for example (Burda et al., 2016; Cremer et al., 2017).

Outline

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$p(z|x, \theta) = \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)}$$

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$\begin{aligned} p(z|x, \theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\ &= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} \end{aligned}$$

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \text{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$\begin{aligned} p(z|x,\theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\ &= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)} \end{aligned}$$

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$\begin{aligned} p(z|x, \theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\ &= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)} \\ &= p(z) \end{aligned}$$

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Posterior collapse

Suppose we choose to model with an autoregressive likelihood, e.g.,

$$X_i|\theta, z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

We point estimate θ along with λ

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$\begin{aligned} p(z|x, \theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\ &= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)} \\ &= p(z) \end{aligned}$$

- the **true posterior** *collapses* to the prior

Posterior collapse is a failure mode of *maximum likelihood estimation* that also afflict VAEs. This problem happens whether or not we employ approximate inference (yes, you read it right, it can happen also with a tractable mixture model!), and because VAEs are trained by *Frequentist* VI, they also suffer from this failure mode.

It's particularly pronounced when the likelihood is sufficiently expressive to assign high likelihood to the observed data.

Does it mean there's no point to LVMs whenever we have an expressive likelihood? Not necessarily, if you go back to our motivations for LVMs (both discrete and continuous) you will see that expressiveness is only one of them. We still have many others: inductive bias, generalisation, transparency, controllable generation, semi-supervised learning, uncertainty estimates (yet to be discussed).

To learn more about posterior collapse check (Chen et al., 2017; Alemi et al., 2018). In an language modelling context, see (Pelsmaeker and Aziz, 2020).

Strong generators

If your likelihood model is able to express dependencies between the output variables (e.g. an RNN), the model may simply ignore the latent code.

Note that though $X \perp Z$ (or $X_i \perp Z \mid X_{<i}$)
$\prod_{i=1}^{n} p(x_i|x_{<i}, \theta)$ *still is* an exact factorisation of $p(x|\theta)$.

We call such models *strong generators*.

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \;||\; p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

$I(X;Z|\lambda) = \int\int q(x,z|\lambda)\log\frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)}\mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

An excellent further reading here is (Alemi et al., 2018).

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$ is close to 0 for most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;

$I(X;Z|\lambda) = \int\int q(x,z|\lambda)\log\frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)}\mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

An excellent further reading here is (Alemi et al., 2018).

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\,(q(z|x,\lambda)\mid\mid p(z))]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\,(q(z|x,\lambda)\mid\mid p(z))$ is close to 0 for most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;
- greedy decoding $\arg\max_{x_i} \log p(x_i|z,x_{<i})$ from a prior sample $z \sim p(z)$ is deterministic;

$I(X;Z|\lambda) = \int\int q(x,z|\lambda)\log\frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)}\mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

An excellent further reading here is (Alemi et al., 2018).

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\,(q(z|x,\lambda) \mid\mid p(z))]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\,(q(z|x,\lambda) \mid\mid p(z))$ is close to 0 for most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;
- greedy decoding $\arg\max_{x_i} \log p(x_i|z,x_{<i})$ from a prior sample $z \sim p(z)$ is deterministic;
- this does not mean ancestral samples from $p(x|z,\theta)$ will be bad

$I(X;Z|\lambda) = \int\int q(x,z|\lambda)\log\frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)}\mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

An excellent further reading here is (Alemi et al., 2018).

KL scaling

Gradually incorporate the KL term into the objective

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \beta\,\mathrm{KL}\left(q(z|x,\lambda)\,||\,p(z)\right)$$

where β starts at 0 and goes to 1 after a number of steps.

KL scaling, a.k.a. 'KL annealing', was proposed by Bowman et al. (2016).

βVAE (Higgins et al., 2017) extends the idea.

KL scaling

Gradually incorporate the KL term into the objective

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \beta\,\mathrm{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)$$

where β starts at 0 and goes to 1 after a number of steps.

This sometimes helps reach better local optimum, but there are not guarantees. In fact, oftentimes, soon after we reach 1, the posterior collapses again.

KL scaling, a.k.a. ‘KL annealing’, was proposed by Bowman et al. (2016).

βVAE (Higgins et al., 2017) extends the idea.

Free bits

Another strategy is to promote the posterior to deviate a bit from the prior by not penalising for the first few nats of information:

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \max(r, \mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right))$$

where $r \geq 0$ is known as "free bits"

This is an attempt to promote solutions where $R \geq r$

Free bits was presented by Kingma et al. (2016). For an alternative version known as soft free bits see (Chen et al., 2017).

For a view of free bits related to constraints on the ELBO, see Pelsmaeker and Aziz (2020). Check the citations therein for more on posterior collapse.

Attention!

But note that if we scale down the KL term permanently, or allow too many free bits, then the conditional $p(x|z, \theta)$ will over-specialise to samples from the approximate posterior $q(z|x, \lambda)$. This can lead to bad generalisation and/or poor samples when generating from the prior.

Outline

Predictors

Suppose we are modelling some data points $(x, y) \in \mathcal{D}$ conditionally. We can introduce a latent variable z, just like we did in the case of discrete LVMs.

$$p(y|x, \theta) = \int p(z|x, \theta)p(y|x, z, \theta)\mathrm{d}z$$

In that case the ELBO becomes

$$\mathbb{E}_{q(z|x,y,\lambda)}\left[\log p(y|x, z, \theta)\right] - \mathrm{KL}\left(q(z|x, y, \lambda) \mid\mid p(z|x, \theta)\right)$$

The KL term now contributes to updating both λ and θ.

For the conditional $p(z|x, \theta)$ we may choose $Z|\theta, x \sim \mathcal{N}(\mu(x; \theta), \sigma^2(x; \theta))$.

Check for example variational NMT (Zhang et al., 2016).

And, of course, we can also model paired observations with a *joint model*. That is, $p(x, y|\theta) = \int p(z)p(x|z, \theta)p(y|x, z, \theta)\mathrm{d}z$. Check for example, auto-encoding variational NMT (Eikema and Aziz, 2019).

Some extensions

- Richer priors (Tomczak and Welling, 2018; Pelsmaeker and Aziz, 2020)
- Richer posteriors (Kingma et al., 2016; Huang et al., 2018; De Cao et al., 2020)
- Spherical distributions (Davidson et al., 2018; De Cao and Aziz, 2020)
- hierarchical models (Fraccaro et al., 2016; Schulz et al., 2018; Ziegler and Rush, 2019)

Applications

A non-exhaustive list of examples

- language modelling (Bowman et al., 2016; Xu and Durrett, 2018)
- word representation (Rios et al., 2018; Bražinskas et al., 2018)
- machine translation (Zhang et al., 2016; Schulz et al., 2018; Eikema and Aziz, 2019)
- syntactic parsing (Corro and Titov, 2018; Kim et al., 2019; Corro and Titov, 2019)
- semantic parsing (Lyu and Titov, 2018)
- generation of inflected wordforms (Zhou and Neubig, 2017; Ataman et al., 2020)
- interpretability (Bastings et al., 2019; Cao et al., 2020)
- question answering (Deng et al., 2018)

Variational Autoencoder

Advantages

- Backprop training
- Easy to implement
- Posterior inference possible
- One objective for both NNs
- Amortised inference

Drawbacks

- Discrete latent variables are not possible
- Optimisation may be difficult with several latent variables

References I

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A. Saurous, and Kevin Murphy. Fixing a Broken ELBO. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 159–168, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR. URL http://proceedings.mlr.press/v80/alemi18a.html.

Duygu Ataman, Wilker Aziz, and Alexandra Birch. A Latent Morphology Model for Open-Vocabulary Neural Machine Translation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJxSI1SKDH.

References II

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 2963–2977, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1284. URL https://www.aclweb.org/anthology/P19-1284.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/K16-1002. URL https://www.aclweb.org/anthology/K16-1002.

References III

Arthur Bražinskas, Serhii Havrylov, and Ivan Titov. Embedding Words as Distributions with a Bayesian Skip-gram Model. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1775–1789, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1151.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. *arXiv:1509.00519 [cs, stat]*, November 2016. URL http://arxiv.org/abs/1509.00519. arXiv: 1509.00519.

Nicola De Cao, Michael Schlichtkrull, Wilker Aziz, and Ivan Titov. How do Decisions Emerge across Layers in Neural Models? Interpretation with Differentiable Masking. In *EMNLP*, 2020.

References IV

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. In *International Conference on Machine Learning*, 2017. Journal Abbreviation: arXiv preprint arXiv:1611.02731.

Caio Corro and Ivan Titov. Differentiable Perturb-and-Parse: Semi-Supervised Parsing with a Structured Variational Autoencoder. In *ICLR*, September 2018. URL https://openreview.net/forum?id=BJlgNh0qKQ.

Caio Corro and Ivan Titov. Learning Latent Trees with Stochastic Perturbations and Differentiable Dynamic Programming. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5508–5521, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1551. URL https://www.aclweb.org/anthology/P19-1551.

References V

Chris Cremer, Quaid Morris, and David Duvenaud. Reinterpreting Importance-Weighted Autoencoders. *arXiv:1704.02916 [stat]*, August 2017. URL http://arxiv.org/abs/1704.02916. arXiv: 1704.02916.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence (UAI-18)*, 2018.

Nicola De Cao and Wilker Aziz. The Power Spherical distribution. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, June 2020. URL http://arxiv.org/abs/2006.04437. arXiv: 2006.04437.

References VI

Nicola De Cao, Wilker Aziz, and Ivan Titov. Block neural autoregressive flow. In Ryan P. Adams and Vibhav Gogate, editors, *UAI*, volume 115 of *Proceedings of machine learning research*, pages 1263–1273, Tel Aviv, Israel, July 2020. PMLR. URL http://proceedings.mlr.press/v115/de-cao20a.html. tex.pdf: http://proceedings.mlr.press/v115/de-cao20a/de-cao20a.pdf.

Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander Rush. Latent Alignment and Variational Attention. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 9712–9724. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/8179-latent-alignment-and-variational-attention.pdf.

References VII

Bryan Eikema and Wilker Aziz. Auto-Encoding Variational Neural Machine Translation. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 124–141, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4315. URL https://www.aclweb.org/anthology/W19-4315.

Marco Fraccaro, Sø ren Kaae Sø nderby, Ulrich Paquet, and Ole Winther. Sequential Neural Models with Stochastic Layers. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2199–2207. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6039-sequential-neural-models-with-stochastic-layers.pdf.

References VIII

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural Autoregressive Flows. In *International Conference on Machine Learning*, pages 2078–2087. PMLR, July 2018. URL http://proceedings.mlr.press/v80/huang18d.html. ISSN: 2640-3498.

References IX

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised Recurrent Neural Network Grammars. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1105–1117, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1114. URL https://www.aclweb.org/anthology/N19-1114.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

References X

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved Variational Inference with Inverse Autoregressive Flow. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4743–4751. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6581-improved-variational-inference-with-inverse-autoregres pdf.

Chunchuan Lyu and Ivan Titov. AMR Parsing as Graph Prediction with Latent Alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1037. URL https://www.aclweb.org/anthology/P18-1037.

References XI

Tom Pelsmaeker and Wilker Aziz. Effective estimation of deep generative language models. In *ACL*, 2020.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China, June 2014. PMLR. URL http://proceedings.mlr.press/v32/rezende14.html. Issue: 2.

References XII

Miguel Rios, Wilker Aziz, and Khalil Sima'an. Deep Generative Model for Joint Alignment and Word Representation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1011–1023, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1092. URL https://www.aclweb.org/anthology/N18-1092.

Philip Schulz, Wilker Aziz, and Trevor Cohn. A Stochastic Decoder for Neural Machine Translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1243–1252, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1115. URL https://www.aclweb.org/anthology/P18-1115.

References XIII

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly Stochastic Variational Bayes for non-Conjugate Inference. In *International Conference on Machine Learning*, pages 1971–1979. PMLR, June 2014. URL http://proceedings.mlr.press/v32/titsias14.html. ISSN: 1938-7228.

Jakub M Tomczak and Max Welling. VAE with a VampPrior. In *AISTATS*, 2018. URL https://arxiv.org/pdf/1705.07120.pdf.

Jiacheng Xu and Greg Durrett. Spherical Latent Spaces for Stable Variational Autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4503–4513, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1480. URL https://www.aclweb.org/anthology/D18-1480.

References XIV

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. Variational Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1050. URL https://www.aclweb.org/anthology/D16-1050.

Chunting Zhou and Graham Neubig. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 310–320, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1029. URL https://www.aclweb.org/anthology/P17-1029.

References XV

Zachary Ziegler and Alexander Rush. Latent Normalizing Flows for Discrete Sequences. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7673–7682, Long Beach, California, USA, June 2019. PMLR. URL http://proceedings.mlr.press/v97/ziegler19a.html.