Tutorial: Deep Latent NLP (bit.do/lvnlp)

Introduction

Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Deep Latent-Variable Models of Natural Language

Yoon Kim, Sam Wiseman, Alexander Rush

harvardnlp

Tutorial 2018

https://github.com/harvardnlp/DeepLatentNLP

1. Introduction
   - Goals
   - Background
2. Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies

1 Introduction
Goals
Background

2 Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics

6 Case Studies

Goal of Latent-Variable Modeling

Probabilistic models provide a declarative language for specifying prior knowledge and structural relationships in the context of unknown variables.

Makes it easy to specify:

- Known interactions in the data
- Uncertainty about unknown factors
- Constraints on model properties

Goal of Latent-Variable Modeling

Probabilistic models provide a declarative language for specifying prior knowledge and structural relationships in the context of unknown variables.

Makes it easy to specify:

- Known interactions in the data
- Uncertainty about unknown factors
- Constraints on model properties

Latent-Variable Modeling in NLP

Long and rich history of latent-variable models of natural language.

Major successes include, among many others:

- Statistical alignment for translation
- Document clustering and topic modeling
- Unsupervised part-of-speech tagging and parsing

Goals of Deep Learning

Toolbox of methods for learning rich, non-linear data representations through numerical optimization.

Makes it easy to fit:

- Highly-flexible predictive models
- Transferable feature representations
- Structurally-aligned network architectures

Introduction
Goals
Background
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Goals of Deep Learning

Toolbox of methods for learning rich, non-linear data representations through numerical optimization.

Makes it easy to fit:

- Highly-flexible predictive models
- Transferable feature representations
- Structurally-aligned network architectures

Deep Learning in NLP

Current dominant paradigm for NLP.

Major successes include, among many others:

- Text classification
- Neural machine translation
- NLU Tasks (QA, NLI, etc)

Tutorial: Deep Latent-Variable Models for NLP

How should a contemporary ML/NLP researcher reason about latent-variables?

What unique challenges come from modeling text with latent variables?

What techniques have been explored and shown to be effective in recent papers?

We explore these through the lens of *variational inference*.

Tutorial: Deep Latent-Variable Models for NLP

How should a contemporary ML/NLP researcher reason about latent-variables?

What unique challenges come from modeling text with latent variables?

What techniques have been explored and shown to be effective in recent papers?

We explore these through the lens of *variational inference*.

Tutorial Take-Aways

1. A collection of deep latent-variable models for NLP
2. An understanding of a variational objective
3. A toolkit of algorithms for optimization
4. A formal guide to advanced techniques
5. A survey of example applications
6. Code samples and techniques for practical use

Tutorial Non-Objectives

Not covered (for time, not relevance):

- Many classical latent-variable approaches.
- Undirected graphical models such as MRFs
- Non-likelihood based models such as GANs
- Sampling-based inference such as MCMC.
- Details of deep learning architectures.

1. Introduction
   - Goals
   - Background
2. Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies

What are deep networks?

Deep networks are parameterized non-linear functions; They transform input $\mathbf{z}$ into features $\boldsymbol{h}$ using parameters .

Important examples: The multilayer perceptron,

$$h = \text{MLP}(z; \ ) = V \ (Wz + b) + a \qquad = fV; W; a; bg;$$

The recurrent neural network, which maps a sequence of inputs $z_{1:T}$ into a sequence of features $h_{1:T}$,

$$h_t = \text{RNN}(h_t \ _1; z_t; \ ) = \ (Uz_t + Vh_t \ _1 + b) \qquad = fV; U; bg$$

.

What are deep networks?

Deep networks are parameterized non-linear functions; They transform input $\mathbf{z}$ into features $\boldsymbol{h}$ using parameters .

Important examples: The multilayer perceptron,

$$\boldsymbol{h} = \mathrm{MLP}(\mathrm{z}; \quad) = V \quad (W \mathrm{z} + b) + a \qquad = f V; W; a; bg;$$

The recurrent neural network, which maps a sequence of inputs $\mathbf{z}_{1:T}$ into a sequence of features $\boldsymbol{h}_{1:T}$,

$$h_t = \mathrm{RNN}(h_{t \ \ 1}; \mathrm{z}_t; \quad) = \quad (U \mathrm{z}_t + V h_{t \ \ 1} + b) \qquad = f V; U; bg$$

.

Introduction
Goals
Background
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

What are latent variable models?

Latent variable models give us a joint distribution

$$p(x; z; \quad):$$

- x is our observed data
- z is a collection of latent variables
- are the deterministic parameters of the model, such as the neural network parameters

Data consists of N i.i.d samples,

$$p(x^{(1:N)}; z^{(1:N)}; \quad) = \prod_{n=1}^{N} p(x^{(n)} j z^{(n)}; \quad)p(z^{(n)}; \quad):$$

What are latent variable models?

Latent variable models give us a joint distribution

$$p(x; z;\ ):$$

- x is our observed data
- z is a collection of latent variables
- are the deterministic parameters of the model, such as the neural network parameters

Data consists of N i.i.d samples,

$$p(x^{(1:N)}; z^{(1:N)};\ ) = \prod_{n=1}^{N} p(x^{(n)} j z^{(n)};\ )p(z^{(n)};\ ):$$

What are latent variable models?

Latent variable models give us a joint distribution

$$p(x; z; \quad):$$

- x is our observed data
- z is a collection of latent variables
- are the deterministic parameters of the model, such as the neural network parameters

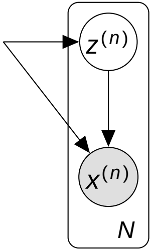
Data consists of N i.i.d samples,

$$p(x^{(1:N)}; z^{(1:N)}; \quad) = \prod_{n=1}^{N} p(x^{(n)} j z^{(n)}; \quad) p(z^{(n)}; \quad):$$

Probabilistic Graphical Models

A directed PGM shows the conditional independence structure.

By chain rule, latent variable model over observations can be represented as,



$$p(x^{(1:N)}, z^{(1:N)}; \theta) = \prod_{n=1}^{N} p(x^{(n)} \mid z^{(n)}; \theta)\, p(z^{(n)}; \theta)$$

Specific models may factor further.

Posterior Inference

For models $p(x; z; \;)$, we'll be interested in the *posterior* over latent variables z:

$$p(z \mid x; \;) = \frac{p(x; z; \;)}{p(x; \;)}.$$

Why?

- z will often represent interesting information about our data (e.g., the cluster $x^{(n)}$ lives in, how similar $x^{(n)}$ and $x^{(n+1)}$ are).
- Learning the parameters of the model often requires calculating posteriors as a subroutine.
- Intuition: if I know likely $z^{(n)}$ for $x^{(n)}$, I can learn by maximizing $p(x^{(n)} \mid z^{(n)}; \;)$.

Introduction
Goals
Background
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Posterior Inference

For models $p(x; z; \quad)$, we'll be interested in the *posterior* over latent variables z:

$$p(z \, j \, x; \quad) = \frac{p(x; z; \quad)}{p(x; \quad)}:$$

Why?

- z will often represent interesting information about our data (e.g., the cluster $x^{(n)}$ lives in, how similar $x^{(n)}$ and $x^{(n+1)}$ are).
- Learning the parameters $\quad$ of the model often requires calculating posteriors as a subroutine.
- Intuition: if I know likely $z^{(n)}$ for $x^{(n)}$, I can learn by maximizing $p(x^{(n)} \, j \, z^{(n)}; \quad)$.

Problem Statement: Two Views

Deep Models & LV Models are naturally complementary:

Rich function approximators with modular parts.

Declarative methods for specifying model constraints.

Deep Models & LV Models are frustratingly incompatible:

Deep networks make posterior inference intractable.

Latent variable objectives complicate backpropagation.

Problem Statement: Two Views

Deep Models & LV Models are naturally complementary:

Rich function approximators with modular parts.

Declarative methods for specifying model constraints.

Deep Models & LV Models are frustratingly incompatible:

Deep networks make posterior inference intractable.

Latent variable objectives complicate backpropagation.

1. Introduction

2. Models
   - Discrete Models
   - Continuous Models
   - Structured Models

3. Variational Objective

4. Inference Strategies

5. Advanced Topics

6. Case Studies

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

A Language Model

Our goal is to model a sentence, $x_1 \dots x_T$.

Context: RNN language models are remarkable at this task,

$$x_{1:T} \sim \text{RNNLM}(x_{1:T};\theta).$$

Defined as,

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t \mid x_{<t}) = \prod_{t=1}^{T} \text{softmax}(\mathbf{W} h_t)_{x_t}$$

$$\text{where} \quad h_t = \text{RNN}(h_{t-1}, x_{t-1};\theta)$$

A Language Model

Our goal is to model a sentence, $x_1 \ldots x_T$.

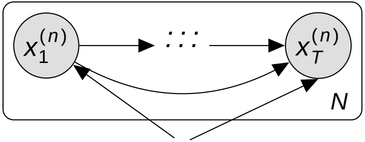
Context: RNN language models are remarkable at this task,

$$x_{1:T} \quad \mathrm{RNNLM}(x_{1:T}; \ ).$$

Defined as,

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t \mid x_{<t}) = \prod_{t=1}^{T} \mathrm{softmax}(\mathbf{W} h_t)_{x_t}$$

$$\text{where} \quad h_t = \mathrm{RNN}(h_{t-1}, \mathbf{x}_{t-1}; \ )$$



A Collection of Model Archetypes

Focus: semi-supervised or unsupervised learning, i.e. don't just learn the probabilities, but the process. Range of choices in selecting z

1. Discrete LVs z (*Clustering*)
2. Continuous LVs z (*Dimensionality reduction*)
3. Structured LVs z (*Structured learning*)

A Collection of Model Archetypes

Focus: semi-supervised or unsupervised learning, i.e. don't just learn the probabilities, but the process. Range of choices in selecting z

1. Discrete LVs z (*Clustering*)
2. Continuous LVs z (*Dimensionality reduction*)
3. Structured LVs z (*Structured learning*)

1. Introduction
2. Models
   - Discrete Models
   - Continuous Models
   - Structured Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies

Model 1: Discrete Clustering

Inference Process:

In an old house in Paris that was covered with vines lived twelve little girls in two straight lines. → Cluster 23

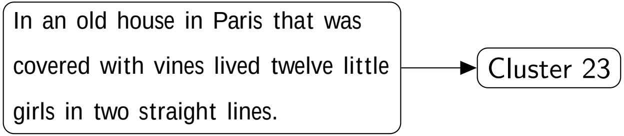
Discrete latent variable models induce a clustering over sentences $x^{(n)}$.

Example uses:

- Document/sentence clustering [Willett 1988; Aggarwal and Zhai 2012].
- Mixture of expert text generation models [Jacobs et al. 1991; Garmash and Monz 2016; Lee et al. 2016]

Model 1: Discrete Clustering

Inference Process:



Discrete latent variable models induce a clustering over sentences $x^{(n)}$.

Example uses:

- Document/sentence clustering [Willett 1988; Aggarwal and Zhai 2012].
- Mixture of expert text generation models [Jacobs et al. 1991; Garmash and Monz 2016; Lee et al. 2016]

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Model 1: Discrete - Mixture of Categoricals

Generative process:

1. Draw cluster $z \in \{1, \ldots, K\}$ from a categorical with param .
2. Draw word T words x_t from a categorical with word distribution ${}_z$.

Parameters: $= \{ \in {}^{K\ 1}, K\ \ V$ stochastic matrix $\}$

Gives rise to the "Naive Bayes" distribution:

$$p(x, z;) = p(z;)\ \ p(x \mid z;) = {}_z \prod_{t=1}^{T} \mathrm{Cat}(x_t;)$$

$$= {}_z \prod_{t=1}^{T} {}_{z, x_t}$$

Introduction

Models

Discrete Models

Continuous Models

Structured Models

Variational Objective

Inference Strategies

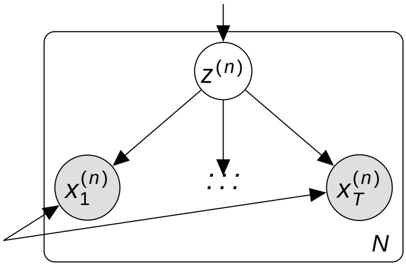
Advanced Topics

Case Studies

Conclusion

References

Model 1: Graphical Model View



$$\prod_{n=1}^{N} p(x^{(n)}, z^{(n)}; \quad ; \quad) = \prod_{n=1}^{N} p(z^{(n)}; \quad) \quad p(x^{(n)} \,|\, z^{(n)}; \quad)$$

$$= \prod_{n=1}^{N} \quad {}_{z^{(n)}} \quad \prod_{t=1}^{T} \quad {}_{z^{(n)}, x_t^{(n)}}$$

Introduction

Models

Discrete Models

Continuous Models

Structured Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

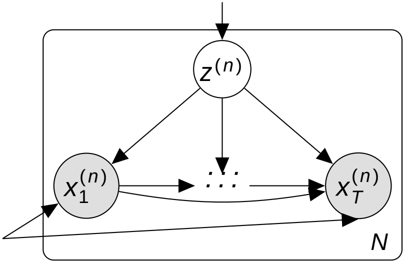
References

Deep Model 1: Discrete - Mixture of RNNs

Generative process:

1. Draw cluster $z \in \{1, \ldots, K\}$ from a categorical.
2. Draw words $x_{1:T}$ from RNNLM with parameters ${}_z$.

$$p(x, z; \ ) = {}_z \quad \text{RNNLM}(x_{1:T}; \ {}_z)$$



Introduction

Models

Discrete Models

Continuous Models

Structured Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Difference Between Models

Dependence structure:

- Mixture of Categoricals: x_t independent of other x_j given z.
- Mixture of RNNs: x_t fully dependent.

Interesting question: how will this affect the learned latent space?

Number of parameters:

- Mixture of Categoricals: $K \quad V$.
- Mixture of RNNs: $K \quad d^2 + V \quad d$ with RNN with d hidden dims.

Difference Between Models

Dependence structure:

- Mixture of Categoricals: x_t independent of other x_j given z.
- Mixture of RNNs: x_t fully dependent.

Interesting question: how will this affect the learned latent space?

Number of parameters:

- Mixture of Categoricals: $K \quad V$.
- Mixture of RNNs: $K \quad d^2 + V \quad d$ with RNN with d hidden dims.

Posterior Inference

For both discrete models, can apply Bayes' rule:

$$p(z \mid x; \;) = \frac{p(z) \quad p(x \mid z)}{p(x)}$$

$$= \frac{p(z) \quad p(x \mid z)}{\sum_{k=1}^{K} p(z=k) \quad p(x \mid z=k)}$$

For mixture of categoricals, posterior uses word counts under each $\;_k$.

For mixture of RNNs, posterior requires running RNN over x for each k.

Posterior Inference

For both discrete models, can apply Bayes' rule:

$$p(z \mid x; \theta) = \frac{p(z) \times p(x \mid z)}{p(x)} = \frac{p(z) \times p(x \mid z)}{\sum_{k=1}^{K} p(z=k) \times p(x \mid z=k)}$$

For mixture of categoricals, posterior uses word counts under each μ_k.

For mixture of RNNs, posterior requires running RNN over x for each k.

Posterior Inference

For both discrete models, can apply Bayes' rule:

$$
\begin{aligned}
p(z \mid x; \quad) &= \frac{p(z) \quad p(x \mid z)}{p(x)} \\
&= \frac{p(z) \quad p(x \mid z)}{\sum_{k=1}^{K} p(z = k) \quad p(x \mid z = k)}
\end{aligned}
$$

For mixture of categoricals, posterior uses word counts under each $\ _k$.

For mixture of RNNs, posterior requires running RNN over x for each k.

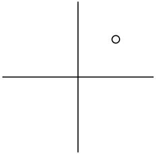
Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

1. Introduction
2. Models
   - Discrete Models
   - Continuous Models
   - Structured Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies

Model 2: Continuous / Dimensionality Reduction

Inference Process:

In an old house in Paris that was covered with vines lived twelve little girls in two straight lines. →



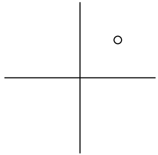
Find a lower-dimensional, well-behaved continuous representation of a sentence. Latent variables in $\mathbb{R}^d$ make distance/similarity easy. Examples:

- Recent work in text generation assumes a latent vector per sentence [Bowman et al. 2016; Yang et al. 2017; Hu et al. 2017].
- Certain sentence embeddings (e.g., Skip-Thought vectors [Kiros et al. 2015]) can be interpreted in this way.

Model 2: Continuous / Dimensionality Reduction

Inference Process:

In an old house in Paris that was covered with vines lived twelve little girls in two straight lines.



Find a lower-dimensional, well-behaved continuous representation of a sentence. Latent variables in $\mathbb{R}^d$ make distance/similarity easy. Examples:

- Recent work in text generation assumes a latent vector per sentence [Bowman et al. 2016; Yang et al. 2017; Hu et al. 2017].
- Certain sentence embeddings (e.g., Skip-Thought vectors [Kiros et al. 2015]) can be interpreted in this way.

Model 2: Continuous "Mixture"

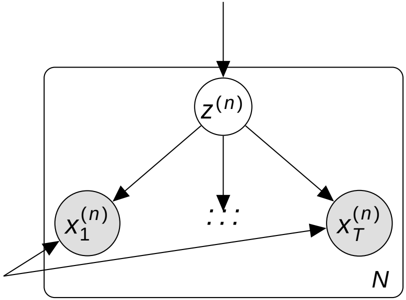
Generative Process:

1. Draw continuous latent variable **z** from Normal with param .
2. For each t, draw word x_t from categorical with param $\mathrm{softmax}(W\mathbf{z})$.

Parameters: $= \{ \;\in \mathbb{R}^d\}, \;= \{W \in \mathbb{R}^{V\;d}\}$

Intuition: is a global distribution, **z** captures local word distribution of the sentence.

Graphical Model View



Gives rise to the joint distribution:

$$\prod_{n=1}^{N} p(x^{(n)}; z^{(n)}; \quad) = \prod_{n=1}^{N} p(z^{(n)}; \quad) \quad p(x^{(n)} | z^{(n)}; \quad)$$

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Deep Model 2: Continuous "Mixture" of RNNs

Generative Process:

1. Draw latent variable $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
2. Draw each token x_t from a conditional RNNLM.

RNN is also conditioned on latent $\mathbf{z}$,

$$
\begin{aligned}
p(x, \mathbf{z}; \theta, \mu, I) &= p(\mathbf{z}; \mu, I) \times p(x \mid \mathbf{z}; \theta) \\
&= \mathcal{N}(\mathbf{z}; \mu, I) \times \mathrm{CRNNLM}(x_{1:T}; \theta, \mathbf{z})
\end{aligned}
$$

where

$$
\mathrm{CRNNLM}(x_{1:T}; \theta, \mathbf{z}) = \prod_{t=1}^{T} \mathrm{softmax}(\mathbf{W} h_t)_{x_t}
$$

$$
h_t = \mathrm{RNN}(h_{t-1}, [\mathbf{x}_{t-1}; \mathbf{z}]; \theta)
$$

Introduction

Models

Discrete Models

Continuous Models

Structured Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Deep Model 2: Continuous "Mixture" of RNNs

Generative Process:

1. Draw latent variable $\mathbf{z} \quad \mathcal{N}(\; ; I)$.
2. Draw each token x_t from a conditional RNNLM.

RNN is also conditioned on latent $\mathbf{z}$,

$$p(x; \mathbf{z}; \; ; \; ; I) = p(\mathbf{z}; \; ; I) \quad p(x \mid \mathbf{z}; \;)$$

$$= \mathcal{N}(\mathbf{z}; \; ; I) \quad \mathrm{CRNNLM}(x_{1:T}; \; ; \mathbf{z})$$

where

$$\mathrm{CRNNLM}(x_{1:T}; \; ; \mathbf{z}) = \prod_{t=1}^{T} \mathrm{softmax}(W h_t)_{x_t}$$

$$h_t = \mathrm{RNN}(h_{t \; 1}; [\mathbf{x}_{t \; 1}; \mathbf{z}]; \;)$$

Deep Model 2: Continuous "Mixture" of RNNs

Generative Process:

1. Draw latent variable $\mathbf{z} \quad N(\;; I)$.
2. Draw each token x_t from a conditional RNNLM.

RNN is also conditioned on latent $\mathbf{z}$,

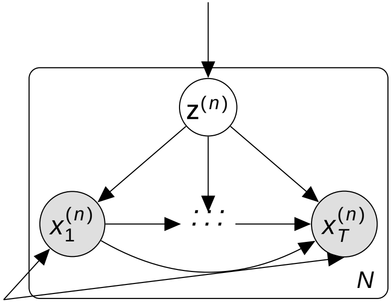
$$\begin{aligned} p(x; \mathrm{z};\ ;\ ; I) &= p(\mathrm{z};\ ; I) \quad p(x \,j\, \mathrm{z};\ ) \\ &= N(\mathrm{z};\ ; I) \quad \mathrm{CRNNLM}(x_{1:T};\ ; \mathrm{z}) \end{aligned}$$

where

$$\mathrm{CRNNLM}(x_{1:T};\ ; \mathrm{z}) = \prod_{t=1}^{T} \mathrm{softmax}(W h_t)_{x_t}$$

$$h_t = \mathrm{RNN}(h_{t\ 1}; [\mathrm{x}_{t\ 1}; \mathrm{z}];\ )$$

Graphical Model View



Posterior Inference

For continuous models, Bayes' rule is harder to compute,

$$p(z \mid x; \theta) = \frac{p(z; \theta) \times p(x \mid z; \theta)}{\int_z p(z; \theta) \times p(x \mid z; \theta)\, dz}$$

- Shallow and deep Model 2 variants mirror Model 1 variants exactly, but with continuous z.
- Integral intractable (in general) for both shallow and deep variants.

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Posterior Inference

For continuous models, Bayes' rule is harder to compute,

$$p(z \mid x; \theta) = \frac{p(z; \theta) \times p(x \mid z; \theta)}{\int_z p(z; \theta) \times p(x \mid z; \theta)\, dz}$$

- Shallow and deep Model 2 variants mirror Model 1 variants exactly, but with continuous z.
- Integral intractable (in general) for both shallow and deep variants.

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

1 Introduction

2 Models
   Discrete Models
   Continuous Models
   Structured Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics

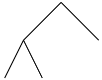
6 Case Studies

Introduction
Models
Discrete Models
Continuous Models
Structured Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Model 3: Structure Learning

Inference Process:

In an old house in Paris that was covered with vines lived twelve little girls in two straight lines.



Structured latent variable models are used to infer unannotated structure:

- Unsupervised POS tagging [Brown et al. 1992; Merialdo 1994; Smith and Eisner 2005]
- Unsupervised dependency parsing [Klein and Manning 2004; Headden III et al. 2009]

Or when structure is useful for *interpreting* our data:

- Segmentation of documents into topical passages [Hearst 1997]
- Alignment [Vogel et al. 1996]

Introduction

Models

Discrete Models

Continuous Models

Structured Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Model 3: Structured - Hidden Markov Model

Generative Process:

1. For each t, draw $z_t \in \{1, \ldots, K\}$ from a categorical with param $_{z_{t-1}}$.
2. Draw observed token x_t from categorical with param $_{z_t}$.

Parameters: $= \{K \times K$ stochastic matrix $, K \times V$ stochastic matrix $\}$

Gives rise to the joint distribution:

$$p(x, z; \ ) = \prod_{t=1} p(z_t \mid z_{t-1}; \ _{z_{t-1}}) \prod_{t=1} p(x_t \mid z_t; \ _{z_t})$$

$$= \prod_{t=1} \ _{z_{t-1}, z_t} \prod_{t=1} \ _{z_t, x_t}$$

Introduction

Models
- Discrete Models
- Continuous Models
- **Structured Models**

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Model 3: Structured - Hidden Markov Model

Generative Process:

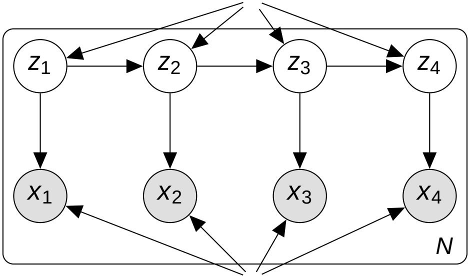
1. For each t, draw $z_t \in \{1, \ldots, K\}$ from a categorical with param $\;_{z_{t-1}}$.
2. Draw observed token x_t from categorical with param $\;_{z_t}$.

Parameters: $\; = \{ \; K \;\; K$ stochastic matrix $\;, K \;\; V$ stochastic matrix $\; \}$

Gives rise to the joint distribution:

$$
\begin{aligned}
p(x, z; \;) &= \prod_{t=1}^{T} p(z_t \mid z_{t-1}; \;_{z_{t-1}}) \prod_{t=1}^{T} p(x_t \mid z_t; \;_{z_t}) \\
&= \prod_{t=1}^{T} \;_{z_{t-1}, z_t} \prod_{t=1}^{T} \;_{z_t, x_t}
\end{aligned}
$$

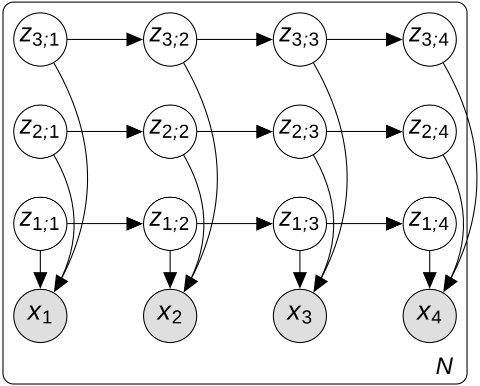
Graphical Model View



$$p(x, z; \ ) = \prod_{t=1}^{T} p(z_t \mid z_{t-1}; \ _{z_{t-1}}) \prod_{t=1}^{T} p(x_t \mid z_t; \ _{z_t})$$

$$= \prod_{t=1}^{T} \ _{z_{t-1}, z_t} \prod_{t=1}^{T} \ _{z_t, x_t}$$

Further Extension: Factorial HMM



$$p(x, z; \theta) = \prod_{l=1}^{L} \prod_{t=1}^{T} p(z_{l,t} \mid z_{l,t-1}) \prod_{t=1}^{T} p(x_t \mid z_{1:L,t})$$

Deep Model 3: Deep HMM

Parameterize transition and emission distributions with neural networks (c.f., Tran et al. [2016])

Model transition distribution as

$$p(z_t \, j \, z_{t \quad 1}) = \text{softmax}(\text{MLP}(z_{t \quad 1}; \quad))$$

Model emission distribution as

$$p(x_t \, j \, z_t) = \text{softmax}(\text{MLP}(z_t; \quad))$$

Note: $K \quad K$ transition parameters for standard HMM vs. $O(K \quad d + d^2)$ for deep version.

Deep Model 3: Deep HMM

Parameterize transition and emission distributions with neural networks (c.f., Tran et al. [2016])

Model transition distribution as

$$p(z_t \mid z_{t-1}) = \text{softmax}(\text{MLP}(z_{t-1}; \theta))$$

Model emission distribution as

$$p(x_t \mid z_t) = \text{softmax}(\text{MLP}(z_t; \theta))$$

Note: $K \times K$ transition parameters for standard HMM vs. $O(K \times d + d^2)$ for deep version.

Introduction

Models

Discrete Models
Continuous Models
Structured Models

Variational Objective

Inference Strategies

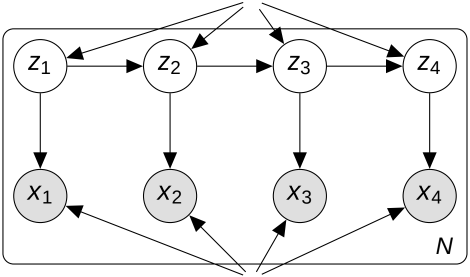
Advanced Topics

Case Studies

Conclusion

References

Graphical Model View



$$p(x, z; \quad) = \prod_{t=1}^{T} p(z_t \mid z_{t-1}; \quad_{z_{t-1}}) \prod_{t=1}^{T} p(x_t \mid z_t; \quad_{z_t})$$

$$= \prod_{t=1}^{T} \quad_{z_{t-1}, z_t} \prod_{t=1}^{T} \quad_{z_t, x_t}$$

Posterior Inference

For structured models, Bayes' rule may tractable,

$$p(z \mid x;\ ) = \frac{p(z;\ )\quad p(x \mid z;\ )}{\sum_{z'} p(z';\ )\quad p(x \mid z';\ )}$$

- Unlike previous models, z contains interdependent "parts."
- For *both* shallow and deep Model 3 variants, it's possible to calculate $p(x;\ )$ exactly, with a dynamic program.
- For some structured models, like Factorial HMM, the dynamic program may still be intractable.

Posterior Inference

For structured models, Bayes' rule may tractable,

$$p(z \mid x; \ ) = \frac{p(z; \ ) \quad p(x \mid z; \ )}{\sum_{z'} p(z'; \ ) \quad p(x \mid z'; \ )}$$

- Unlike previous models, z contains interdependent "parts."
- For *both* shallow and deep Model 3 variants, it's possible to calculate $p(x; \ )$ exactly, with a dynamic program.
- For some structured models, like Factorial HMM, the dynamic program may still be intractable.

1 Introduction

2 Models

3 Variational Objective

Maximum Likelihood

ELBO

4 Inference Strategies

5 Advanced Topics

6 Case Studies

1. Introduction
2. Models
3. Variational Objective
   - Maximum Likelihood
   - ELBO
4. Inference Strategies
5. Advanced Topics
6. Case Studies

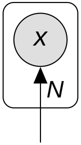
Learning with Maximum Likelihood

Objective: Find model parameters that maximize the likelihood of the data,

$$= \arg\max \sum_{n=1}^{N} \log p(x^{(n)}; \quad)$$

Learning Deep Models

$$L(\ ) = \sum_{n=1}^{N} \log p(x^{(n)};\ \ )$$



Dominant framework is gradient-based optimization:

$$^{(i)} = \ ^{(i\ \ 1)} + \ \ r\ \ L(\ )$$

$r\ \ L(\ )$ calculated with backpropagation.

Tactics: mini-batch based training, adaptive learning rates [Duchi et al. 2011; Kingma and Ba 2015].

Learning Deep Latent-Variable Models: Marginalization

Likelihood requires summing out the latent variables,

$$p(x;\ ) = \sum_{z \in \mathcal{Z}} p(x, z;\ ) \quad (= \int p(x, z;\ ) dz \text{ if continuous } z)$$

In general, **hard to optimize** log-likelihood for the training set,

$$L(\ ) = \sum_{n=1}^{N} \log \sum_{z \in \mathcal{Z}} p(x^{(n)}, z;\ )$$

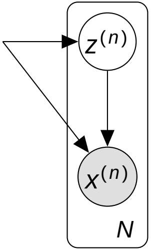
Learning Deep Latent-Variable Models: Marginalization

Likelihood requires summing out the latent variables,

$$p(x;\theta) = \sum_{z \in \mathcal{Z}} p(x, z; \theta) \quad \left(= \int p(x, z; \theta) dz \text{ if continuous } z\right)$$

In general, **hard to optimize** log-likelihood for the training set,

$$L(\theta) = \sum_{n=1}^{N} \log \sum_{z \in \mathcal{Z}} p(x^{(n)}, z; \theta)$$



1 Introduction

2 Models

3 Variational Objective

Maximum Likelihood

ELBO

4 Inference Strategies

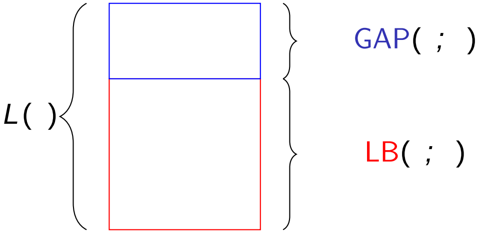
5 Advanced Topics

6 Case Studies

Introduction
Models
Variational Objective
Maximum Likelihood
ELBO
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Variational Inference

High-level: decompose objective into lower-bound and gap.



$$L(\ ) = \text{LB}(\ ;\ ) + \text{GAP}(\ ;\ ) \text{ for some}$$

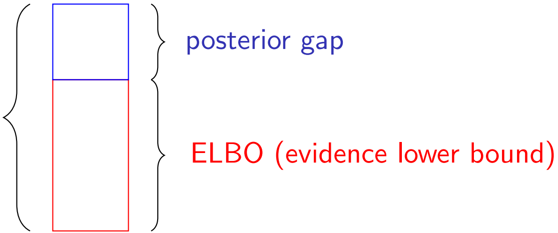
Provides framework for deriving a rich set of optimization algorithms.

Introduction
Models
Variational Objective
Maximum Likelihood
ELBO
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Marginal Likelihood: Variational Decomposition

For any[1] distribution $q(z \mid x; \ )$ over z,

$$L(\ ) = \mathbb{E}_q\left[\log \frac{p(x, z; \ )}{q(z \mid x; \ )}\right] + \mathrm{KL}[q(z \mid x; \ ) \,\|\, p(z \mid x; \ )]$$



Since KL is always non-negative, $L(\ ) \quad \mathrm{ELBO}(\ ; \ )$.

[1] Technical condition: $\mathrm{supp}(q(z)) \quad \mathrm{supp}(p(z \mid x; \ ))$

Evidence Lower Bound: Proof

$$
\begin{aligned}
\log p(x; \ ) &= \mathbb{E}_q \log p(x) \quad \textit{(Expectation over z)} \\
&= \mathbb{E}_q \log \frac{p(x, z)}{p(z \mid x)} \quad \textit{(Mult/div by p(z|x), combine numerator)} \\
&= \mathbb{E}_q \log \left(\frac{p(x, z)}{q(z \mid x)} \frac{q(z \mid x)}{p(z \mid x)} \right) \quad \textit{(Mult/div by q(z|x))} \\
&= \mathbb{E}_q \log \frac{p(x, z)}{q(z \mid x)} + \mathbb{E}_q \log \frac{q(z \mid x)}{p(z \mid x)} \quad \textit{(Split Log)} \\
&= \mathbb{E}_q \log \frac{p(x, z; \ )}{q(z \mid x; \ )} + \mathrm{KL}[q(z \mid x; \ ) \parallel p(z \mid x; \ )]
\end{aligned}
$$

Introduction

Models

Variational Objective

Maximum Likelihood

ELBO

Inference Strategies

Advanced Topics

Case Studies

Conclusion

References

Evidence Lower Bound: Proof

$$
\begin{aligned}
\log p(x; \ ) &= \mathrm{E}_q \log p(x) \quad \textit{(Expectation over z)} \\
&= \mathrm{E}_q \log \frac{p(x; z)}{p(z\, j\, x)} \quad \textit{(Mult/div by } p(zjx)\textit{, combine numerator)} \\
&= \mathrm{E}_q \log \left(\frac{p(x; z)}{q(z\, j\, x)} \frac{q(z\, j\, x)}{p(z\, j\, x)} \right) \quad \textit{(Mult/div by } q(zjx)\textit{)} \\
&= \mathrm{E}_q \log \frac{p(x; z)}{q(z\, j\, x)} + \mathrm{E}_q \log \frac{q(z\, j\, x)}{p(z\, j\, x)} \quad \textit{(Split Log)} \\
&= \mathrm{E}_q \log \frac{p(x; z; \ )}{q(z\, j\, x; \ )} + \mathrm{KL}[q(z\, j\, x; \ ) \, k \, p(z\, j\, x; \ )]
\end{aligned}
$$

Introduction
Models
Variational Objective
Maximum Likelihood
ELBO
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Evidence Lower Bound: Proof

$$
\begin{aligned}
\log p(x; \;) &= \mathbb{E}_q \log p(x) \quad \textit{(Expectation over } z\textit{)} \\
&= \mathbb{E}_q \log \frac{p(x, z)}{p(z \mid x)} \quad \textit{(Mult/div by } p(z|x)\textit{, combine numerator)} \\
&= \mathbb{E}_q \log \left(\frac{p(x, z)}{q(z \mid x)} \frac{q(z \mid x)}{p(z \mid x)} \right) \quad \textit{(Mult/div by } q(z|x)\textit{)} \\
&= \mathbb{E}_q \log \frac{p(x, z)}{q(z \mid x)} + \mathbb{E}_q \log \frac{q(z \mid x)}{p(z \mid x)} \quad \textit{(Split Log)} \\
&= \mathbb{E}_q \log \frac{p(x, z; \;)}{q(z \mid x; \;)} + \mathrm{KL}[q(z \mid x; \;) \,\|\, p(z \mid x; \;)]
\end{aligned}
$$

Evidence Lower Bound: Proof

$$
\begin{aligned}
\log p(x; \quad) &= \mathbb{E}_q \log p(x) \quad \textit{(Expectation over z)} \\
&= \mathbb{E}_q \log \frac{p(x; z)}{p(z \mid x)} \quad \textit{(Mult/div by } p(z|x)\textit{, combine numerator)} \\
&= \mathbb{E}_q \log \left(\frac{p(x; z)}{q(z \mid x)} \frac{q(z \mid x)}{p(z \mid x)} \right) \quad \textit{(Mult/div by } q(z|x)\textit{)} \\
&= \mathbb{E}_q \log \frac{p(x; z)}{q(z \mid x)} + \mathbb{E}_q \log \frac{q(z \mid x)}{p(z \mid x)} \quad \textit{(Split Log)} \\
&= \mathbb{E}_q \log \frac{p(x; z; \quad)}{q(z \mid x; \quad)} + \mathrm{KL}[q(z \mid x; \quad) \,\|\, p(z \mid x; \quad)]
\end{aligned}
$$

Introduction
Models
Variational Objective
Maximum Likelihood
ELBO
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

Evidence Lower Bound: Proof

$$
\begin{aligned}
\log p(x;\theta) &= \mathbb{E}_q \log p(x) \quad \textit{(Expectation over z)} \\
&= \mathbb{E}_q \log \frac{p(x,z)}{p(z \mid x)} \quad \textit{(Mult/div by } p(z|x)\textit{, combine numerator)} \\
&= \mathbb{E}_q \log \left(\frac{p(x,z)}{q(z \mid x)} \frac{q(z \mid x)}{p(z \mid x)} \right) \quad \textit{(Mult/div by } q(z|x)\textit{)} \\
&= \mathbb{E}_q \log \frac{p(x,z)}{q(z \mid x)} + \mathbb{E}_q \log \frac{q(z \mid x)}{p(z \mid x)} \quad \textit{(Split Log)} \\
&= \mathbb{E}_q \log \frac{p(x,z;\theta)}{q(z \mid x;\lambda)} + \mathrm{KL}[q(z \mid x;\lambda) \,\|\, p(z \mid x;\theta)]
\end{aligned}
$$

Evidence Lower Bound over Observations

$$\text{ELBO}(\theta, \lambda; x) = \mathbb{E}_{q(z)}\Big[\log \frac{p(x, z; \theta)}{q(z \mid x; \lambda)}\Big]$$

ELBO is a function of the generative model parameters, θ, and the variational parameters, λ.

$$\begin{aligned}\sum_{n=1}^{N} \log p(x^{(n)}; \theta) \geq &\sum_{n=1}^{N} \text{ELBO}(\theta, \lambda; x^{(n)}) \\ &= \sum_{n=1}^{N} \mathbb{E}_{q(z \mid x^{(n)}; \lambda)}\Big[\log \frac{p(x^{(n)}, z; \theta)}{q(z \mid x^{(n)}; \lambda)}\Big] \\ &= \text{ELBO}(\theta, \lambda; x^{(1:N)}) = \text{ELBO}(\theta, \lambda)\end{aligned}$$

Setup: Selecting Variational Family

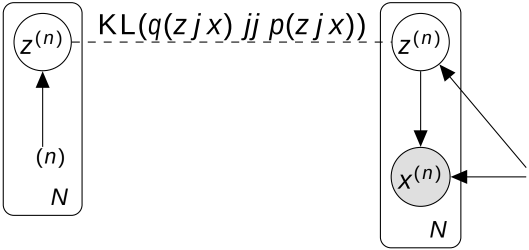
Just as with p and , we can select any form of q and that satisfies ELBO conditions.

Different choices of q will lead to different algorithms.

We will explore several forms of q:

- Posterior
- Point Estimate / MAP
- Amortized
- Mean Field (later)

Example Family : Full Posterior Form



$= [\ ^{(1)}; \ldots ; \ ^{(N)}]$ is a concatenation of local variational parameters $\ ^{(n)}$, e.g.

$$q(z^{(n)} \,j\, x^{(n)}; \ \ ) = q(z^{(n)} \,j\, x^{(n)}; \ ^{(n)}) = N(\ ^{(n)}; 1)$$

Introduction

Models

Variational Objective

Maximum Likelihood

ELBO

Inference Strategies

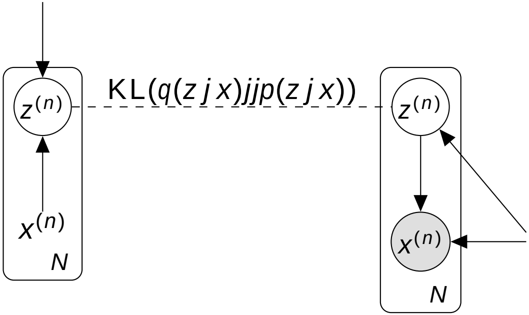
Advanced Topics

Case Studies

Conclusion

References

Example Family: Amortized Parameterization [Kingma and Welling 2014]



parameterizes a global network (encoder/inference network) that is run over $x^{(n)}$ to produce the local variational distribution, e.g.

$$q(z^{(n)} \mid x^{(n)}; \quad) = N(\quad ^{(n)}, 1), \qquad ^{(n)} = \mathrm{enc}(x^{(n)}; \quad)$$

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies

Exact Gradient

Sampling

Conjugacy

5 Advanced Topics

6 Case Studies

Maximizing the Evidence Lower Bound

Central quantity of interest: almost all methods are maximizing the ELBO

$$\arg\max_{;} \mathrm{ELBO}(\; ; \;)$$

Aggregate ELBO objective,

$$\arg\max_{;} \mathrm{ELBO}(\; ; \;) = \arg\max_{;} \sum_{n=1}^{N} \mathrm{ELBO}(\; ; \; ; x^{(n)})$$

$$= \arg\max_{;} \sum_{n=1}^{N} \mathbb{E}_q \left[\log \frac{p(x^{(n)}; z^{(n)}; \;)}{q(z^{(n)} \mid x^{(n)}; \;)} \right]$$

Maximizing the Evidence Lower Bound

Central quantity of interest: almost all methods are maximizing the ELBO

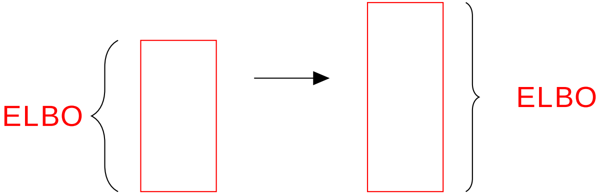
$$\arg\max_{\;;\;} \text{ELBO}(\;;\;)$$

Aggregate ELBO objective,

$$\begin{aligned}\arg\max_{\;;\;} \text{ELBO}(\;;\;) &= \arg\max_{\;;\;} \sum_{n=1}^{N} \text{ELBO}(\;;\;;x^{(n)}) \\ &= \arg\max_{\;;\;} \sum_{n=1}^{N} \mathbb{E}_q\left[\log \frac{p(x^{(n)}, z^{(n)};\;)}{q(z^{(n)} \mid x^{(n)};\;)}\right]\end{aligned}$$

Maximizing ELBO: Model Parameters

$$\arg\max \mathbb{E}_q \left[\log \frac{p(x; z; \ )}{q(z \mid x; \ )} \right] = \arg\max \mathbb{E}_q[\log p(x; z; \ )]$$



Intuition: Maximum likelihood problem under variables drawn from $q(z \mid x; \ )$.

Introduction
Models
Variational Objective
Inference Strategies
Exact Gradient
Sampling
Conjugacy
Advanced Topics
Case Studies
Conclusion
References

Model Estimation: Gradient Ascent on Model Parameters

Easy: Gradient respect to

$$
\begin{aligned}
r \;\; \mathrm{ELBO}(\;;\;\;; x) &= r \;\; \mathbb{E}_q \Big[\log p(x; z; \;\;) \Big] \\
&= \mathbb{E}_q \Big[r \;\; \log p(x; z; \;\;) \Big]
\end{aligned}
$$

Since q not dependent on , r moves inside expectation.

Estimate with samples from q. Term $\log p(x; z; \;)$ is easy to evaluate. (In practice single sample is often sufficient).

In special cases, can exactly evaluate expectation.

Model Estimation: Gradient Ascent on Model Parameters

Easy: Gradient respect to θ

$$
\begin{aligned}
\nabla_\theta \, \text{ELBO}(\theta, \lambda; x) &= \nabla_\theta \, \mathbb{E}_q \Big[\log p(x, z; \theta) \Big] \\
&= \mathbb{E}_q \Big[\nabla_\theta \log p(x, z; \theta) \Big]
\end{aligned}
$$

- Since q not dependent on θ, ∇ moves inside expectation.
- Estimate with samples from q. Term $\log p(x, z; \theta)$ is easy to evaluate. (In practice single sample is often sufficient).
- In special cases, can exactly evaluate expectation.

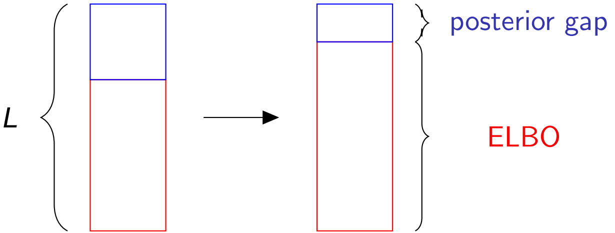
Introduction
Models
Variational Objective
Inference Strategies
Exact Gradient
Sampling
Conjugacy
Advanced Topics
Case Studies
Conclusion
References

Maximizing ELBO: Variational Distribution

$$\arg\max \text{ELBO}(\ ;\ )$$

$$= \arg\max \log p(x;\ ) \quad \text{KL}[q(z\,j\,x;\ )\,k\,p(z\,j\,x;\ )]$$

$$= \arg\min \text{KL}[q(z\,j\,x;\ )\ k\ p(z\,j\,x;\ )]$$



Intuition: q should approximate the posterior $p(zjx)$. However, may be difficult if q or p is a deep model.

Model Inference: Gradient Ascent on λ?

Hard: Gradient respect to λ

$$\nabla_{\lambda} \text{ELBO}(\theta, \lambda; x) = \nabla_{\lambda} \mathbb{E}_{q}\left[\log \frac{p(x, z; \theta)}{q(z \mid x; \lambda)}\right]$$

$$\neq \mathbb{E}_{q}\left[\nabla_{\lambda} \log \frac{p(x, z; \theta)}{q(z \mid x; \lambda)}\right]$$

Cannot naively move ∇ inside the expectation, since q depends on λ.

This section: Inference in practice:

1. Exact gradient
2. Sampling: score function, reparameterization
3. Conjugacy: closed-form, coordinate ascent

Introduction
Models
Variational Objective
Inference Strategies
Exact Gradient
Sampling
Conjugacy
Advanced Topics
Case Studies
Conclusion
References

Model Inference: Gradient Ascent on ?

Hard: Gradient respect to

$$\nabla \ \mathrm{ELBO}(\ ; \ ; x) = \nabla \ \mathbb{E}_q \left[\log \frac{p(x, z; \ )}{q(z \mid x; \ )} \right]$$

$$\neq \mathbb{E}_q \left[\nabla \ \log \frac{p(x, z; \ )}{q(z \mid x; \ )} \right]$$

- Cannot naively move ∇ inside the expectation, since q depends on .
- This section: Inference in practice:
  1. Exact gradient
  2. Sampling: score function, reparameterization
  3. Conjugacy: closed-form, coordinate ascent

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies
- Exact Gradient
- Sampling
- Conjugacy

5 Advanced Topics

6 Case Studies

Strategy 1: Exact Gradient

$$
\begin{aligned}
\nabla \ \mathrm{ELBO}(\theta, \phi; x) &= \nabla \ \mathbb{E}_{q(z \mid x; \phi)}\Big[\log \frac{p(x, z; \theta)}{q(z \mid x; \phi)}\Big] \\
&= \nabla \sum_{z \in \mathcal{Z}} q(z \mid x; \phi) \log \frac{p(x, z; \theta)}{q(z \mid x; \phi)}
\end{aligned}
$$

Naive enumeration: Linear in $|\mathcal{Z}|$.

Depending on structure of q and p, potentially faster with dynamic programming.

Applicable mainly to Model 1 and 3 (Discrete and Structured), or Model 2 with point estimate.

Strategy 1: Exact Gradient

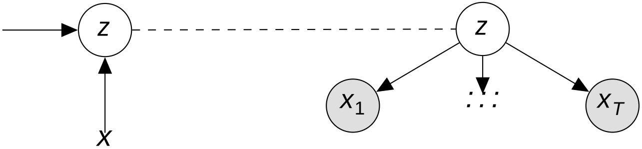
$$
\begin{aligned}
r \quad \mathrm{ELBO}(\ ; \ ; x) &= r \quad \mathrm{E}_{q(z\, j\, x; \ )} \overset{\mathrm{h}}{\ } \log \frac{p(x; z; \ )}{q(z\, j\, x; \ )} \overset{\mathrm{i}}{\ } \\
&= r \quad \sum_{z 2 Z} q(z\, j\, x; \ ) \log \frac{p(x; z; \ )}{q(z\, j\, x; \ )}
\end{aligned}
$$

Naive enumeration: Linear in jZj.

Depending on structure of q and p, potentially faster with dynamic programming.

Applicable mainly to Model 1 and 3 (Discrete and Structured), or Model 2 with point estimate.

Example: Model 1 - Naive Bayes



Let $q(z\, j\, x;\ \ ) = \text{Cat}(\ \ )$ where $\ \ = \text{enc}(x;\ \ )$

$$r \quad \text{ELBO}(\ ;\ ;x) = r \quad \mathbb{E}_{q(z\,j\,x;\ )}\Big[\log \frac{p(x;z;\ )}{q(z\,j\,x;\ )}\Big]$$

$$= r \quad \sum_{z 2 Z} q(z\,j\,x;\ ) \log \frac{p(x;z;\ )}{q(z\,j\,x;\ )}$$

$$= r \quad \sum_{z 2 Z} \ _z \log \frac{p(x;z;\ )}{\ _z}$$

Introduction

Models

Variational Objective

Inference Strategies

Exact Gradient

Sampling

Conjugacy

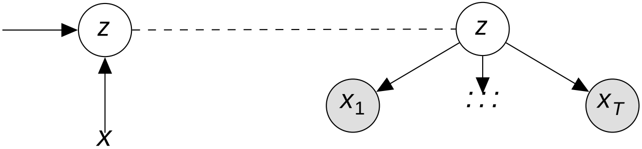
Advanced Topics

Case Studies

Conclusion

References

Example: Model 1 - Naive Bayes



Let $q(z \mid x; \ ) = \text{Cat}(\ )$ where $\ = \text{enc}(x; \ )$

$$\nabla \ \text{ELBO}(\ ; \ ; x) = \nabla \ \mathbb{E}_{q(z \mid x; \ )} \Big[\log \frac{p(x; z; \ )}{q(z \mid x; \ )} \Big]$$

$$= \nabla \ \sum_{z \in Z} q(z \mid x; \ ) \log \frac{p(x; z; \ )}{q(z \mid x; \ )}$$

$$= \nabla \ \sum_{z \in Z} \ _z \log \frac{p(x; z; \ )}{\ _z}$$

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies
- Exact Gradient
- Sampling
- Conjugacy

5 Advanced Topics

6 Case Studies

Strategy 2: Sampling

$$\nabla_{\phi} \operatorname{ELBO}(\theta, \phi; x) = \nabla_{\phi} \mathbb{E}_q\left[\log \frac{\log p(x, z; \theta)}{\log q(z \mid x; \phi)}\right]$$

$$= \nabla_{\phi} \mathbb{E}_q\left[\log p(x, z; \theta)\right] - \nabla_{\phi} \mathbb{E}_q\left[\log q(z \mid x; \phi)\right]$$

How can we approximate this gradient with sampling? Naive algorithm fails to provide non-zero gradient.

$$z^{(1)}, \ldots, z^{(J)} \sim q(z \mid x; \phi)$$

$$\nabla_{\phi} \frac{1}{J} \sum_{j=1}^{J} \left[\log p(x, z^{(j)}; \theta)\right] = 0$$

Manipulate expression so we can move ∇_{ϕ} inside $\mathbb{E}_q$ before sampling.

Introduction

Models

Variational Objective

Inference Strategies

- Exact Gradient
- **Sampling**
- Conjugacy

Advanced Topics

Case Studies

Conclusion

References

Strategy 2: Sampling

$$\nabla \ \mathrm{ELBO}(\ ,\ ; x) = \nabla \ \mathbb{E}_q \left[\log \frac{\log p(x, z;\ )}{\log q(z \mid x;\ )} \right]$$

$$= \color{blue}{\nabla \ \mathbb{E}_q \left[\log p(x, z;\ ) \right]} \quad \color{red}{\nabla \ \mathbb{E}_q \left[\log q(z \mid x;\ ) \right]}$$

How can we approximate this gradient with sampling? Naive algorithm fails to provide non-zero gradient.

$$z^{(1)}, \dots, z^{(J)} \quad q(z \mid x;\ )$$

$$\nabla \ \frac{1}{J} \sum_{j=1}^{J} \left[\log p(x, z^{(j)};\ ) \right] = 0$$

Manipulate expression so we can move ∇ inside $\mathbb{E}_q$ before sampling.

Introduction

Models

Variational Objective

Inference Strategies

- Exact Gradient
- **Sampling**
- Conjugacy

Advanced Topics

Case Studies

Conclusion

References

Strategy 2a: Sampling — Score Function Gradient Estimator

First term. Use basic identity:

$$\nabla \log q = \frac{\nabla q}{q} \Rightarrow \nabla q = q \nabla \log q$$

Policy-gradient style training [Williams 1992]

$$\nabla_{\phi} \mathbb{E}_q \Big[\log p(x, z; \theta) \Big] = \sum_z \nabla_{\phi} q(z \mid x; \phi) \log p(x, z; \theta)$$

Strategy 2a: Sampling — Score Function Gradient Estimator

First term. Use basic identity:

$$r \log q = \frac{r q}{q} \quad) \quad r q = q \, r \log q$$

Policy-gradient style training [Williams 1992]

$$r \quad \mathbb{E}_q \left[\log p(x; z; \quad) \right] = \sum_z \underbrace{r \quad q(z \mid x; \quad)}_{q \, r \, \log q} \log p(x; z; \quad)$$

Strategy 2a: Sampling — Score Function Gradient Estimator

First term. Use basic identity:

$$\nabla \log q = \frac{\nabla q}{q} \Rightarrow \nabla q = q \nabla \log q$$

Policy-gradient style training [Williams 1992]

$$\nabla_{\theta} \mathbb{E}_q \Big[\log p(x; z; \theta) \Big] = \sum_z \nabla_{\theta} q(z \mid x; \phi) \log p(x; z; \theta)$$

$$= \sum_z q(z \mid x; \phi) \nabla_{\phi} \log q(z \mid x; \phi) \log p(x; z; \theta)$$

Strategy 2a: Sampling — Score Function Gradient Estimator

First term. Use basic identity:

$$\nabla \log q = \frac{\nabla q}{q} \Rightarrow \nabla q = q \nabla \log q$$

Policy-gradient style training [Williams 1992]

$$\begin{aligned}
\nabla_\phi \mathbb{E}_q\Big[\log p(x, z; \theta)\Big] &= \sum_z \nabla_\phi q(z \mid x; \phi) \log p(x, z; \theta) \\
&= \sum_z q(z \mid x; \phi) \nabla_\phi \log q(z \mid x; \phi) \log p(x, z; \theta) \\
&= \mathbb{E}_q\Big[\log p(x, z; \theta) \nabla_\phi \log q(z \mid x; \phi)\Big]
\end{aligned}$$

Strategy 2a: Sampling — Score Function Gradient Estimator

Second term. Need additional identity:

$$\sum \nabla q = \nabla \sum q = \nabla 1 = 0$$

$$\nabla \mathbb{E}_q \left[\log q(z \mid x; \ ) \right] = \sum_z \nabla \ q(z \mid x; \ ) \log q(z \mid x; \ )$$

Strategy 2a: Sampling — Score Function Gradient Estimator

Second term. Need additional identity:

$$\sum \nabla q = \nabla \sum q = \nabla 1 = 0$$

$$\color{red}{\nabla \; \mathbb{E}_q \left[\log q(z \mid x; \;) \right]} = \sum_z \nabla \; q(z \mid x; \;) \log q(z \mid x; \;)$$

$$= \sum_z \underbrace{\nabla \; q(z \mid x; \;)}_{q \, \nabla \, \log q} \log q(z \mid x; \;) + q(z \mid x; \;) \underbrace{\nabla \; \log q(z \mid x; \;)}_{\frac{\nabla q}{q}}$$

Strategy 2a: Sampling — Score Function Gradient Estimator

Second term. Need additional identity:

$$\sum \nabla q = \nabla \sum q = \nabla 1 = 0$$

$$\nabla \mathbb{E}_q \Big[\log q(z \mid x; \theta) \Big] = \sum_z \nabla \Big(q(z \mid x; \theta) \log q(z \mid x; \theta) \Big)$$

$$= \sum_z \log q(z \mid x; \theta) q(z \mid x; \theta) \nabla \log q(z \mid x; \theta) + \sum_z \nabla q(z \mid x; \theta)$$

Strategy 2a: Sampling — Score Function Gradient Estimator

Second term. Need additional identity:

$$\sum \nabla q = \nabla \sum q = \nabla 1 = 0$$

$$\nabla_\theta \mathbb{E}_q \Big[\log q(z \mid x;\theta) \Big] = \sum_z \nabla_\theta \Big(q(z \mid x;\theta) \log q(z \mid x;\theta) \Big)$$

$$= \sum_z \log q(z \mid x;\theta)\, q(z \mid x;\theta) \nabla_\theta \log q(z \mid x;\theta) + \underbrace{\sum_z \nabla_\theta q(z \mid x;\theta)}_{= \nabla \sum q = \nabla 1 = 0}$$

Strategy 2a: Sampling — Score Function Gradient Estimator

Second term. Need additional identity:

$$\sum \nabla q = \nabla \sum q = \nabla 1 = 0$$

$$\nabla \ \mathbb{E}_q \Big[\log q(z \mid x; \ ) \Big] = \sum_z \nabla \ \ q(z \mid x; \ ) \log q(z \mid x; \ )$$

$$= \sum_z \log q(z \mid x; \ ) q(z \mid x; \ ) \nabla \ \log q(z \mid x; \ ) + \sum_z \nabla \ q(z \mid x; \ )$$

$$= \mathbb{E}_q [\log q(z \mid x; \ ) \nabla \ q(z \mid x; \ )]$$

Introduction

Models

Variational Objective

Inference Strategies

Exact Gradient

Sampling

Conjugacy

Advanced Topics

Case Studies

Conclusion

References

Strategy 2a: Sampling — Score Function Gradient Estimator

Putting these together,

$$
\begin{aligned}
\nabla\ \text{ELBO}(\ ;\ ;x) &= \nabla\ \mathbb{E}_q\left[\log \frac{p(x;z;\ )}{q(z\,|\,x;\ )}\right] \\
&= \mathbb{E}_q\left[\log \frac{p(x;z;\ )}{q(z\,|\,x;\ )}\,\nabla\ \log q(z\,|\,x;\ )\right] \\
&= \mathbb{E}_q\left[R_{\ ;\ }(z)\,\nabla\ \log q(z\,|\,x;\ )\right]
\end{aligned}
$$

Introduction
Models
Variational Objective
Inference Strategies
Exact Gradient
Sampling
Conjugacy
Advanced Topics
Case Studies
Conclusion
References

Strategy 2a: Sampling — Score Function Gradient Estimator

Estimate with samples,

$$z^{(1)}, \dots, z^{(J)} \quad q(z \mid x; \;)$$

$$\mathbb{E}_q \left[R_{;}(z) \nabla \; \log q(z \mid x; \;) \right]$$

$$\frac{1}{J} \sum_{j=1}^{J} R_{;}(z^{(j)}) \nabla \; \log q(z^{(j)} \mid x; \;)$$

Intuition: if a sample $z^{(j)}$ is has high reward $R_{;}(z^{(j)})$, increase the probability of $z^{(j)}$ by moving along the gradient $\nabla \; \log q(z^{(j)} \mid x; \;)$.

Strategy 2a: Sampling — Score Function Gradient Estimator

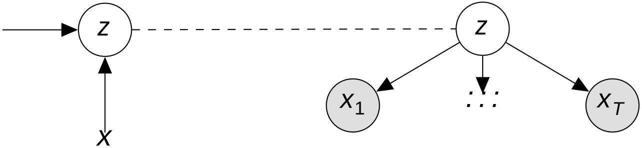
Essentially reinforcement learning with reward $R_{\ ;\ }(z)$

Score function gradient is generally applicable regardless of what distribution q takes (only need to evaluate $r \quad \log q$).

This generality comes at a cost, since the reward is "black-box": unbiased estimator, but high variance.

In practice, need variance-reducing **control variate** B. (More on this later).

Example: Model 1 - Naive Bayes



Let $q(z \mid x; \ ) = \text{Cat}(\ )$ where $\ = \text{enc}(x; \ )$

Sample $z^{(1)}, \dots, z^{(J)} \quad q(z \mid x; \ )$

$$\nabla \ \text{ELBO}(\ ; \ ; x) = \mathbb{E}_q \left[\log \frac{p(x, z; \ )}{q(z \mid x; \ )} \nabla \ \log q(z \mid x; \ ) \right]$$

$$\frac{1}{J} \sum_{j=1}^{J} \ _{z^{(j)}} \log \frac{p(x, z^{(j)}; \ )}{\ _{z^{(j)}}} \nabla \ \log \ _{z^{(j)}}$$

Computational complexity: $O(J)$ vs $O(|Z|)$

Tutorial: Deep Latent NLP (bit.do/lvnlp)

Introduction

Models

Variational Objective

Inference Strategies

Exact Gradient

Sampling

Conjugacy

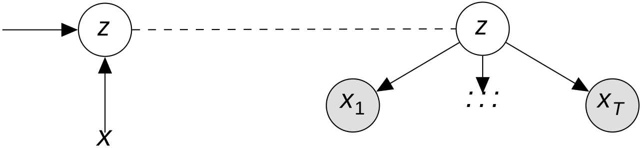
Advanced Topics

Case Studies

Conclusion

References

Example: Model 1 - Naive Bayes



Let $q(z\, j\, x; \quad) = \text{Cat}(\quad)$ where $\quad = \text{enc}(x; \quad)$

Sample $z^{(1)}; : : : ; z^{(J)} \quad q(z\, j\, x; \quad)$

$$r \quad \text{ELBO}(\ ; \ ; x) = \mathbb{E}_q \overset{h}{\ } \log \frac{p(x; z; \quad)}{q(z\, j\, x; \quad)} r \quad \log q(z\, j\, x; \quad) \overset{i}{\ }$$

$$\frac{1}{J} \sum_{j=1}^{J} \quad z^{(j)} \log \frac{p(x; z^{(j)}; \quad)}{z^{(j)}} r \quad \log \quad z^{(j)}$$

Computational complexity: $O(J)$ vs $O(jZj)$

Strategy 2b: Sampling — Reparameterization

Suppose we can sample from q by applying a deterministic, differentiable transformation g to a base noise density,

$$U \qquad z = g(\; ; \;)$$

Gradient calculation (first term):

$$r \; \mathrm{E}_{z \quad q(z j x; \;)} \Big[\log p(x; z; \;) \Big] = r \; \mathrm{E}_{\quad U} \Big[\log p(x; g(\; ; \;); \;) \Big]$$

$$= \mathrm{E}_{\quad U} \Big[r \; \log p(x; g(\; ; \;); \;) \Big]$$

$$\frac{1}{J} \sum_{j=1}^{J} r \; \log p(x; g(^{(j)}; \;); \;)$$

where

$$^{(1)}; \ldots ; \; ^{(J)} \qquad U$$

Strategy 2b: Sampling — Reparameterization

Suppose we can sample from q by applying a deterministic, differentiable transformation g to a base noise density,

$$\mathcal{U} \qquad z = g(\ ; \ )$$

Gradient calculation (first term):

$$\begin{aligned}\nabla \ \mathbb{E}_{z \ \ q(z \mid x; \ )} \left[\log p(x; z; \ ) \right] &= \nabla \ \mathbb{E} \ _{\mathcal{U}} \left[\log p(x; g(\ ; \ ); \ ) \right] \\ &= \mathbb{E} \ _{\mathcal{U}} \left[\nabla \ \log p(x; g(\ ; \ ); \ ) \right] \\ & \quad \frac{1}{J} \sum_{j=1}^{J} \nabla \ \log p(x; g(\ ^{(j)}; \ ); \ )\end{aligned}$$

where

$$^{(1)}, \ldots, \ ^{(J)} \quad \mathcal{U}$$

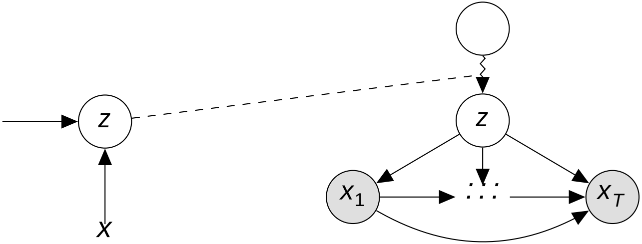
Strategy 2b: Sampling — Reparameterization

Unbiased, like the score function gradient estimator, but empirically lower variance.

In practice, single sample is often sufficient.

Cannot be used out-of-the-box for discrete z.

Strategy 2: Continuous Latent Variable RNN



Choose variational family to be an amortized diagonal Gaussian

$$q(z \mid x; \quad) = \mathcal{N}(\quad ; \quad ^2)$$

$$; \quad ^2 = \text{enc}(x; \quad)$$

Then we can sample from $q(z \mid x; \quad)$ by

$$\mathcal{N}(0; \mathbf{I}) \quad z = \quad + $$

Introduction

Models

Variational Objective

Inference Strategies

- Exact Gradient
- **Sampling**
- Conjugacy

Advanced Topics

Case Studies

Conclusion

References

Strategy 2b: Sampling — Reparameterization

(Recall $R_{\ ;\ }(z) = \log \frac{p(x;z;\ )}{q(z\,j\,x;\ )}$)

Score function:

$$r\ \ \mathrm{ELBO}(\ ;\ \ ;x) = \mathrm{E}_{z\ \ q}[R_{\ ;\ }(z)\, r\ \ \log q(z\,j\,x;\ \ )]$$

Reparameterization:

$$r\ \ \mathrm{ELBO}(\ ;\ \ ;x) = \mathrm{E}_{\ \ N(\mathbf{0};\mathbf{I})}[\,r\ \ R_{\ ;\ }(g(\ ;\ \ ;x))]$$

where $g(\ ;\ \ ;x) = \ \ + \ \ \ .$

Informally, reparameterization gradients differentiate through $R_{\ ;\ }(\ )$ and thus has “more knowledge” about the structure of the objective function.

1. Introduction
2. Models
3. Variational Objective
4. Inference Strategies
   - Exact Gradient
   - Sampling
   - Conjugacy
5. Advanced Topics
6. Case Studies

Strategy 3: Conjugacy

For certain choices for p and q, we can compute parts of

$$\arg\max \text{ELBO}(\ ;\ ; x)$$

exactly in closed-form.

Recall that

$$\arg\max \text{ELBO}(\ ;\ ; x) = \arg\min \text{KL}[q(z \mid x;\ ) \| p(z \mid x;\ )]$$

Strategy 3: Conjugacy

For certain choices for p and q, we can compute parts of

$$\arg\max \mathrm{ELBO}(\;;\;;x)$$

exactly in closed-form.

Recall that

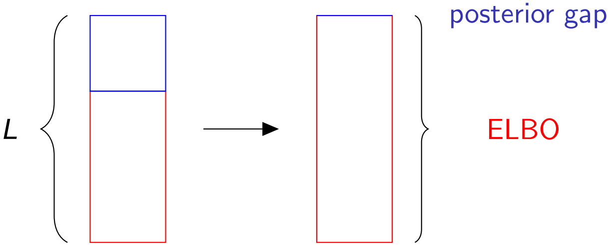
$$\arg\max \mathrm{ELBO}(\;;\;;x) = \arg\min \mathrm{KL}[q(z \mid x;\;) \| p(z \mid x;\;)]$$

Strategy 3a: Conjugacy — Tractable Posterior Inference

Suppose we can tractably calculate $p(z \mid x;\ )$. Then $\mathrm{KL}[q(z \mid x;\ ) \| p(z \mid x;\ )]$ is minimized when,

$$q(z \mid x;\ ) = p(z \mid x;\ )$$

The E-step in Expectation Maximization algorithm [Dempster et al. 1977]



Introduction

Models

Variational Objective

Inference Strategies

Exact Gradient

Sampling

Conjugacy

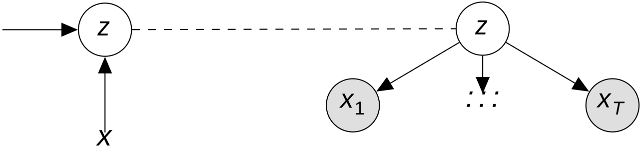
Advanced Topics

Case Studies

Conclusion

References

Example: Model 1 - Naive Bayes

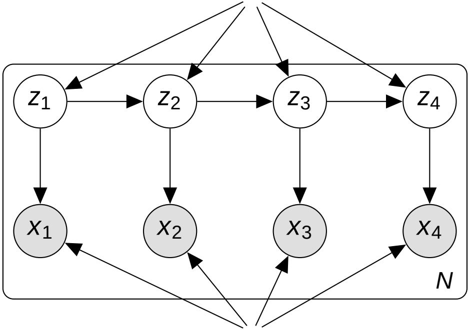


$$p(z \mid x;\ ) = \frac{p(x; z;\ )}{\sum_{z'=1}^{K} p(x; z';\ )}$$

So is given by the parameters of the categorical distribution, i.e.

$$= [p(z = 1 \mid x;\ ); \ldots; p(z = K \mid x;\ )]$$

Example: Model 3 — HMM



$$p(x; z; \quad) = p(z_0) \prod_{t=1}^{T} p(z_t \mid z_{t-1}; \quad) p(x_t \mid z_t; \quad)$$

Example: Model 3 — HMM

Run forward/backward dynamic programming to calculate posterior marginals,

$$p(z_t, z_{t+1} \mid x; \quad)$$

variational parameters $\quad \in \mathbb{R}^{TK^2}$ store edge marginals. These are enough to calculate

$$q(z; \quad) = p(z \mid x; \quad)$$

(i.e. the exact posterior) over any sequence z.

Example: Model 3 — HMM

Run forward/backward dynamic programming to calculate posterior marginals,

$$p(z_t; z_{t+1} \, j \, x; \quad)$$

variational parameters $\quad 2 \, \mathbb{R}^{T K^2}$ store edge marginals. These are enough to calculate

$$q(z; \quad) = p(z \, j \, x; \quad)$$

(i.e. the exact posterior) over any sequence z.

Connection: Gradient Ascent on Log Marginal Likelihood

Why not perform gradient ascent directly on log marginal likelihood?

$$\log p(x; \theta) = \log \sum_z p(x, z; \theta)$$

Same as optimizing ELBO with posterior inference (i.e EM). Gradients of model parameters given by (where $q(z \mid x; \lambda) = p(z \mid x; \theta)$):

$$\nabla_\theta \log p(x; \theta) = \mathbb{E}_{q(z \mid x; \lambda)}[\nabla_\theta \log p(x, z; \theta)]$$

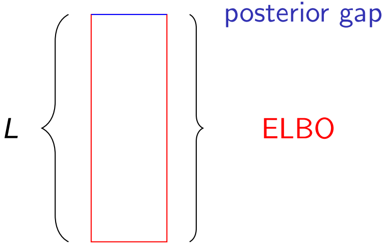
Connection: Gradient Ascent on Log Marginal Likelihood

Why not perform gradient ascent directly on log marginal likelihood?

$$\log p(x;\theta) = \log \sum_z p(x, z;\theta)$$

Same as optimizing ELBO with posterior inference (i.e EM). Gradients of model parameters given by (where $q(z \mid x;\phi) = p(z \mid x;\theta)$):

$$\nabla_\theta \log p(x;\theta) = \mathbb{E}_{q(z \mid x;\phi)}[\nabla_\theta \log p(x, z;\theta)]$$



Connection: Gradient Ascent on Log Marginal Likelihood

Practically, this means we don't have to manually perform posterior inference in the E-step. Can just calculate $\log p(x; \quad)$ and call backpropagation.

Example: in deep HMM, just implement forward algorithm to calculate $\log p(x; \quad)$ and backpropagate using autodiff. No need to implement backward algorithm. (Or vice versa).

(See Eisner [2016]: "Inside-Outside and Forward-Backward Algorithms Are Just Backprop")

Strategy 3b: Conditional Conjugacy

Let $p(z \mid x; \theta)$ be intractable, but suppose $p(x, z; \theta)$ is **conditionally conjugate**, meaning $p(z_t \mid x, z_{-t}; \theta)$ is exponential family.

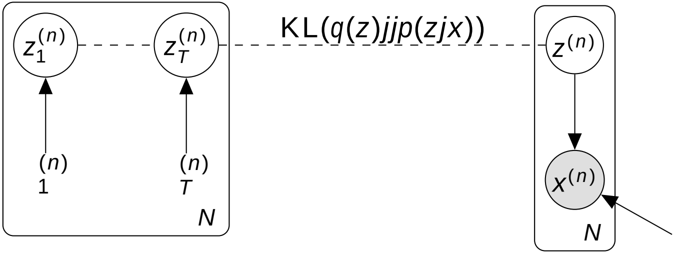
Restrict the family of distributions q so that it factorizes over z_t, i.e.

$$q(z; \lambda) = \prod_{t=1}^{T} q(z_t; \lambda_t)$$

(**mean field** family)

Further choose $q(z_t; \lambda_t)$ so that it is in the same family as $p(z_t \mid x, z_{-t}; \theta)$.

Strategy 3b: Conditional Conjugacy



$$q(z;\ ) = \prod_{t=1}^{T} q(z_t;\ {}_t)$$

Mean Field Family

Optimize ELBO via coordinate ascent, i.e. iterate for $\lambda_1, \dots, \lambda_T$

$$\arg\max_{\lambda_t} \mathrm{KL}\Big[\prod_{t=1}^{T} q(z_t; \lambda_t) \| p(z \mid x; \theta)\Big]$$

Coordinate ascent updates will take the form

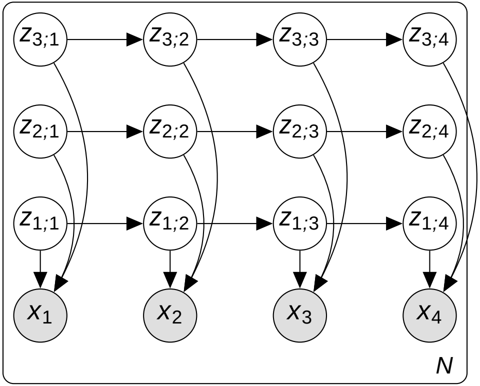
$$q(z_t; \lambda_t) \propto \exp\Big(\mathbb{E}_{q(z_{-t}; \lambda_{-t})}[\log p(x, z; \theta)]\Big)$$

where

$$\mathbb{E}_{q(z_{-t}; \lambda_{-t})}[\log p(x, z; \theta)] = \sum_{j \neq t} \prod_{j \neq t} q(z_j; \lambda_j) \log p(x, z; \theta)$$

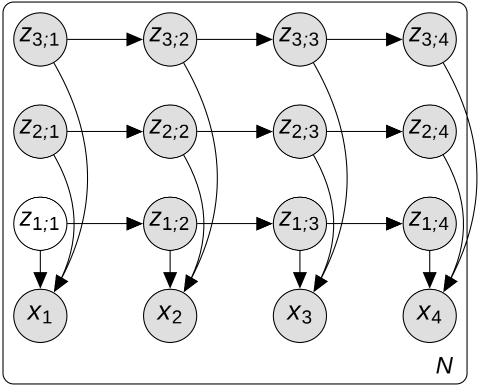
Since $p(z_t \mid x, z_{-t})$ was assumed to be in the exponential family, above updates can be derived in closed form.

Example: Model 3 — Factorial HMM



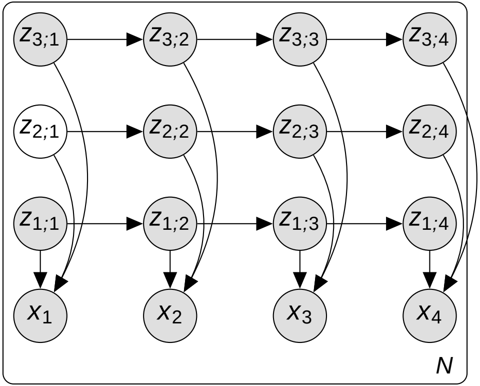
$$p(x, z; \theta) = \prod_{l=1}^{L} \prod_{t=1}^{T} p(z_{l,t} \mid z_{l,t-1}; \theta) p(x_t \mid z_{l,t}; \theta)$$

Example: Model 3 — Factorial HMM



$$q(z_{1;1}; \quad _{1;1}) \ / \ \exp \quad \mathrm{E}_{q(z \ _{(1;1)}; \quad _{(1;1)})}[\log p(x; z; \quad)]$$

Example: Model 3 — Factorial HMM



$$q(z_{2;1};\ \ _{2;1}) \ /\ \exp\ \ \mathrm{E}_{q(z\ \ _{(2;1)};\ \ \ _{(2;1)})}[\log p(x;z;\ \ )]$$

Example: Model 3 — Factorial HMM

Exact Inference:

Naive: K states, L levels $\implies$ HMM with K^L states $\implies$ $O(TK^{2L})$

Smarter: $O(TLK^{L+1})$

Mean Field:

Gaussian emissions: $O(TLK^2)$ [Ghahramani and Jordan 1996].

Categorical emission: need more variational approximations, but ultimately $O(LKVT)$ [Nepal and Yates 2013].

Example: Model 3 — Factorial HMM

Exact Inference:

Naive: K states, L levels $\implies$ HMM with K^L states $\implies$ $O(TK^{2L})$

Smarter: $O(TLK^{L+1})$

Mean Field:

Gaussian emissions: $O(TLK^2)$ [Ghahramani and Jordan 1996].

Categorical emission: need more variational approximations, but ultimately $O(LKVT)$ [Nepal and Yates 2013].

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics
- Gumbel-Softmax
- Flows
- IWAE

6 Case Studies

Advanced Topics

1. Gumbel-Softmax: Extend reparameterization to discrete variables.
2. Flows: Optimize a tighter bound by making the variational family q more flexible.
3. Importance Weighting: Optimize a tighter bound through importance sampling.

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics
- Gumbel-Softmax
- Flows
- IWAE

6 Case Studies

Challenges of Discrete Variables

Review: we can always use score function estimator

$$\nabla_{\phi} \text{ELBO}(x;\theta,\phi) = \mathbb{E}_q\left[\log \frac{p(x,z;\theta)}{q(z \mid x;\phi)} \nabla_{\phi} \log q(z \mid x;\phi)\right]$$
$$= \mathbb{E}_q\left[\left(\log \frac{p(x,z;\theta)}{q(z \mid x;\phi)} - B\right) \nabla_{\phi} \log q(z \mid x;\phi)\right]$$

$\mathbb{E}_q[B \nabla_{\phi} \log q(z \mid x;\phi)] = 0$ (since $\mathbb{E}[\nabla \log q] = \sum q \nabla \log q = \sum \nabla q = 0$)

Control variate B (not dependent on z, but can depend on x).

Estimate this quantity with another neural net [Mnih and Gregor 2014]

$$\left(B(x;\psi) - \log \frac{p(x,z;\theta)}{q(z \mid x;\phi)}\right)^2$$

Challenges of Discrete Variables

Review: we can always use score function estimator

$$\nabla_\phi \text{ ELBO}(x; \theta, \phi) = \mathbb{E}_q \Big[\log \frac{p(x, z; \theta)}{q(z \mid x; \phi)} \nabla_\phi \log q(z \mid x; \phi) \Big]$$
$$= \mathbb{E}_q \Big[\Big(\log \frac{p(x, z; \theta)}{q(z \mid x; \phi)} - B \Big) \nabla_\phi \log q(z \mid x; \phi) \Big]$$

$\mathbb{E}_q[B \nabla_\phi \log q(z \mid x; \phi)] = 0$ (since $\mathbb{E}[\nabla \log q] = \sum q \nabla \log q = \sum \nabla q = 0$)

Control variate B (not dependent on z, but can depend on x).

Estimate this quantity with another neural net [Mnih and Gregor 2014]

$$\Big(B(x; \psi) - \log \frac{p(x, z; \theta)}{q(z \mid x; \phi)} \Big)^2$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

The "Gumbel-Max" trick [Papandreou and Yuille 2011]

$$p(z_k = 1; \quad) = \frac{\ _k}{\sum_{j=1}^{K} \ _j}$$

where $z = [0; 0; \dots; 1; \dots; 0]$ is a one-hot vector.

Can sample from $p(z; \quad)$ by

1. Drawing independent Gumbel noise $\ = \ _1; \dots; \ _K$

$$\ _k = \ \log(\ \log u_k) \qquad u_k \quad U(0; 1)$$

2. Adding $\ _k$ to $\log \ _k$, finding argmax, i.e.

$$i = \arg\max_k[\log \ _k + \ _k] \qquad z_i = 1$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

The "Gumbel-Max" trick [Papandreou and Yuille 2011]

$$p(z_k = 1; \quad) = \frac{{}_k}{\sum_{j=1}^{K} {}_j}$$

where $z = [0; 0; \dots; 1; \dots; 0]$ is a one-hot vector.

Can sample from $p(z; \quad)$ by

1. Drawing independent Gumbel noise $\quad = {}_1; \dots; {}_K$

$$ {}_k = \quad \log(\quad \log u_k) \qquad u_k \quad U(0;1)$$

2. Adding ${}_k$ to $\log {}_k$, finding argmax, i.e.

$$i = \arg\max_k [\log {}_k + {}_k] \qquad z_i = 1$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

The "Gumbel-Max" trick [Papandreou and Yuille 2011]

$$p(z_k = 1; \;) = \frac{\;_k}{\sum_{j=1}^{K} \;_j}$$

where $z = [0, 0, \dots, 1, \dots, 0]$ is a one-hot vector.

Can sample from $p(z; \;)$ by

1. Drawing independent Gumbel noise $\; = \;_1, \dots, \;_K$

$$\;_k = \; \log(\; \log u_k) \qquad u_k \;\; U(0, 1)$$

2. Adding $\;_k$ to $\log \;_k$, finding argmax, i.e.

$$i = \arg\max_k [\log \;_k + \;_k] \qquad z_i = 1$$

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Gumbel-Softmax
Flows
IWAE
Case Studies
Conclusion
References

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

Reparameterization:

$$z = \arg\max_{s \in \Delta^{K-1}} (\log \quad + \quad)^{\top} s = g(\; ; \;)$$

$z = g(\; ; \;)$ is a deterministic function applied to stochastic noise.

Let's try applying this:

$$q(z_k = 1 \mid x; \;) = \frac{k}{\sum_{j=1}^{K} \; j} \qquad = \text{enc}(x; \;)$$

(Recalling $R_{\;;\;}(z) = \log \frac{p(x, z; \;)}{q(z \mid x; \;)}$),

$$\nabla \; \mathbb{E}_{q(z \mid x; \;)}[R_{\;;\;}(z)] = \nabla \; \mathbb{E} \quad_{\text{Gumbel}}[R_{\;;\;}(g(\; ; \;))]$$

$$= \mathbb{E} \quad_{\text{Gumbel}}[\nabla \; R_{\;;\;}(g(\; ; \;))]$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

Reparameterization:

$$z = \arg\max_{s \in \Delta^{K-1}} (\log \pi + \epsilon)^\top s = g(\epsilon; \theta)$$

$z = g(\epsilon; \theta)$ is a deterministic function applied to stochastic noise.

Let's try applying this:

$$q(z_k = 1 \mid x; \phi) = \frac{\pi_k}{\sum_{j=1}^{K} \pi_j} \qquad \pi = \mathrm{enc}(x; \phi)$$

(Recalling $R_{\theta,\phi}(z) = \log \frac{p(x, z; \theta)}{q(z \mid x; \phi)}$),

$$\nabla_\phi \mathbb{E}_{q(z \mid x; \phi)}[R_{\theta,\phi}(z)] = \nabla_\phi \mathbb{E}_{\epsilon \sim \mathrm{Gumbel}}[R_{\theta,\phi}(g(\epsilon; \phi))]$$

$$= \mathbb{E}_{\epsilon \sim \mathrm{Gumbel}}[\nabla_\phi R_{\theta,\phi}(g(\epsilon; \phi))]$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

Reparameterization:

$$z = \arg\max_{s \in \quad K \quad 1} (\log \quad + \quad)^{>} s = g(\; ; \quad)$$

$z = g(\; ; \quad)$ is a deterministic function applied to stochastic noise.

Let's try applying this:

$$q(z_k = 1 \mid x; \quad) = \frac{k}{\sum_{j=1}^{K} \quad j} \qquad = \text{enc}(x; \quad)$$

(Recalling $R \, ; \; (z) = \log \frac{p(x; z; \quad)}{q(z \mid x; \quad)}$),

$$r \quad \mathbb{E}_{q(z \mid x; \quad)}[R \, ; \; (z)] = r \quad \mathbb{E} \quad {}_{\text{Gumbel}}[R \, ; \; (g(\; ; \quad)]$$

$$= \mathbb{E} \quad {}_{\text{Gumbel}}[\, r \quad R \, ; \; (g(\; ; \quad))]$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

But this won't work, because zero gradients (almost everywhere)

$$z = g(\; ; \;) = \arg\max_{s2 \; K \; 1} (\log \; + \;)^{>} s \; =) \quad r \; R_{\;;}(z) = 0$$

Gumbel-Softmax trick: replace arg max with softmax

$$z = \text{softmax}\left(\frac{\log \; + \;}{\;}\right) \qquad z_k = \frac{\exp((\log \;_k + \;_k)= \;)}{\sum_{j=1}^{K} \exp((\log \;_j + \;_j)= \;)}$$

(where is a temperature term.)

$$r \; \mathbb{E}_{q(z \, j \, x; \;)}[R_{\;;}(z)] \quad \mathbb{E}_{\text{Gumbel}}\left[r \; R_{\;;}\left(\text{softmax}\left(\frac{\log \; + \;}{\;}\right)\right)\right]$$

Introduction

Models

Variational Objective

Inference Strategies

Advanced Topics

Gumbel-Softmax

Flows

IWAE

Case Studies

Conclusion

References

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

But this won't work, because zero gradients (almost everywhere)

$$z = g(\;;\;) = \arg\max_{s2 \quad K \quad 1} (\log \quad + \quad)^{>} s \quad =) \quad r \quad R_{;}(z) = 0$$

Gumbel-Softmax trick: replace $\arg\max$ with softmax

$$z = \text{softmax}\left(\frac{\log \quad + \quad}{\quad}\right) \qquad z_k = \frac{\exp((\log \;_k + \;_k)= \;)}{\text{P}_{j=1}^{K} \exp((\log \;_j + \;_j)= \;)}$$

(where is a temperature term.)

$$r \quad \mathbb{E}_{q(z\,j\,x;\;)}[R_{;}(z)] \quad \mathbb{E}_{\text{Gumbel}} \left[r \quad R_{;} \quad \text{softmax}\left(\frac{\log \quad + \quad}{\quad}\right) \right]$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

But this won't work, because zero gradients (almost everywhere)

$$z = g(\ ;\ \ ) = \arg\max_{s2\ \ K\ \ 1} (\log\ \ +\ \ )^{>} s\ \ =)\ \ r\ \ R_{\ ;}\ (z) = 0$$

Gumbel-Softmax trick: replace arg max with softmax

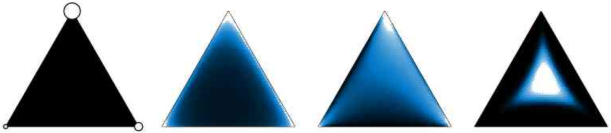
$$z = \text{softmax}\ \frac{\log\ \ +\ \ }{\ } \qquad z_k = \frac{\exp((\log\ \ _k +\ \ _k)=\ \ )}{\text{P}^{K}_{j=1} \exp((\log\ \ _j +\ \ _j)=\ \ )}$$

(where   is a temperature term.)

$$r\ \ \mathbb{E}_{q(z\,j\,x;\ \ )}[R_{\ ;}\ (z)]\ \ \ \mathbb{E}_{\ \ \text{Gumbel}} \text{h}\ r\ \ R_{\ ;}\ \ \text{softmax}\ \frac{\log\ \ +\ \ }{\ }\ \text{i}$$

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

- Approaches a discrete distribution as ! 0 (anneal during training).
- Reparameterizable by construction
- Differentiable and has non-zero gradients



(a) $\lambda = 0$ (b) $\lambda = 1/2$ (c) $\lambda = 1$ (d) $\lambda = 2$

(from Maddison et al. [2017])

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Gumbel-Softmax
Flows
IWAE
Case Studies
Conclusion
References

Gumbel-Softmax: Discrete Reparameterization [Jang et al. 2017; Maddison et al. 2017]

See Maddison et al. [2017] on whether we can use the original categorical densities $p(z), q(z)$, or need to use relaxed densities $p_{\text{GS}}(z), q_{\text{GS}}(z)$.

Requires that $p(x \mid z; \theta)$ "makes sense" for non-discrete z (e.g. attention).

Lower-variance, but biased gradient estimator. Variance $\to \infty$ as $\tau \to 0$.

1. Introduction

2. Models

3. Variational Objective

4. Inference Strategies

5. Advanced Topics
   - Gumbel-Softmax
   - Flows
   - IWAE

6. Case Studies

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Recall

$$\log p(x;\ ) = \text{ELBO}(\ ;\ ;x) - \text{KL}[q(z \mid x;\ ) \,\|\, p(z \mid x;\ )]$$

Bound is tight when variational posterior equals true posterior

$$q(z \mid x;\ ) = p(z \mid x;\ ) \implies \log p(x;\ ) = \text{ELBO}(\ ;\ ;x)$$

We want to make $q(z \mid x;\ )$ as flexible as possible: can we do better than just Gaussian?

Introduction

Models

Variational Objective

Inference Strategies

Advanced Topics

Gumbel-Softmax

Flows

IWAE

Case Studies

Conclusion

References

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Recall

$$\log p(x; \quad) = \mathrm{ELBO}(\quad ; \quad ; x) \quad \mathrm{KL}[q(z \mid x; \quad) \| p(z \mid x; \quad)]$$

Bound is tight when variational posterior equals true posterior

$$q(z \mid x; \quad) = p(z \mid x; \quad) \implies \log p(x; \quad) = \mathrm{ELBO}(\quad ; \quad ; x)$$

We want to make $q(z \mid x; \quad)$ as flexible as possible: can we do better than just Gaussian?

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Idea: transform a sample from a simple initial variational distribution,

$$z_0 \quad q(z \, j \, x; \quad) = N(\; ; \; ^2) \qquad ; \; ^2 = \text{enc}(x; \quad)$$

into a more complex one

$$z_K = f_K \qquad f_2 \quad f_1(z_0; \quad)$$

where $f_i(z_i \;\; _1; \quad)$'s are **invertible** transformations (whose parameters are absorbed by $\quad$).

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Idea: transform a sample from a simple initial variational distribution,

$$z_0 \quad q(z \, j \, x; \quad) = N(\; ; \; ^2) \qquad ; \; ^2 = \text{enc}(x; \quad)$$

into a more complex one

$$z_K = f_K \qquad f_2 \quad f_1(z_0; \quad)$$

where $f_i(z_{i \;\; 1}; \quad)$'s are **invertible** transformations (whose parameters are absorbed by $\quad$).

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Sample from final variational posterior is given by z_K. Density is given by the change of variables formula:

$$\log q_K(z_K \mid x; \theta) = \log q(z_0 \mid x; \theta) + \sum_{k=1}^{K} \log \left| \frac{\partial f_k}{\partial z_k} \right|^{-1}$$

$$= \underbrace{\log q(z_0 \mid x; \theta)}_{\text{log density of Gaussian}} - \sum_{k=1}^{K} \underbrace{\log \left| \frac{\partial f_k}{\partial z_{k-1}} \right|}_{\text{log determinant of Jacobian}}$$

Determinant calculation is $O(N^3)$ in general, but can be made faster depending on parameterization of f_k

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Can still use reparameterization to obtain gradients. Letting $F(z) = f_K \quad f_1(z)$,

$$\mathrm{ELBO}(\;;\;;x) = \nabla \; \mathbb{E}_{q_K(z_K \mid x;\;)}\left[\log \frac{p(x;z;\;)}{q_K(z_K \mid x;\;)}\right]$$

$$= \nabla \; \mathbb{E}_{q(z_0 \mid x;\;)}\left[\log \frac{p(x;F(z_0);\;)}{q(z_0 \mid x;\;)} \quad \log \frac{\partial F}{\partial z_0}\right]$$

$$= \mathbb{E}_{\;\mathcal{N}(\mathbf{0};\mathbf{I})}\left[\nabla \quad \log \frac{p(x;F(z_0);\;)}{q(z_0 \mid x;\;)} \quad \log \frac{\partial F}{\partial z_0}\right]$$

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]

Examples of $f_k(z_{k\ \ 1};\ \ \ )$

Normalizing Flows [Rezende and Mohamed 2015]

$$f_k(z_{k\ \ 1}) = z_{k\ \ 1} + u_k h(w_k^{>} z_{k\ \ 1} + b_k)$$

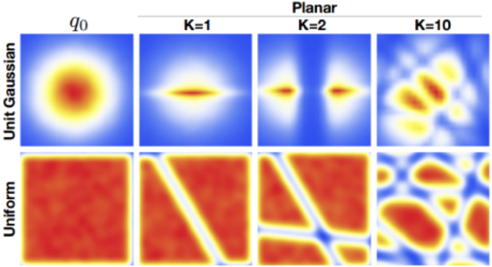
Inverse Autoregressive Flows [Kingma et al. 2016]

$$f_k(z_{k\ \ 1}) = z_{k\ \ 1}\ \ \ {}_k + \ \ {}_k$$

$$_{k;d} = \text{sigmoid}(\text{NN}(z_{k\ \ 1;<d})) \qquad _{k;d} = \text{NN}(z_{k\ \ 1;<d})$$

(In this case the Jacobian is upper triangular, so determinant is just the product of diagonals)

Flows [Rezende and Mohamed 2015; Kingma et al. 2016]



(from Rezende and Mohamed [2015])

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics
- Gumbel-Softmax
- Flows
- IWAE

6 Case Studies

Importance Weighted Autoencoder (IWAE) [Burda et al. 2015]

Flows are a way of tightening the ELBO by making the variational family more flexible.

Not the only way: can obtain a tighter lower bound on $\log p(x; \ )$ by using multiple importance samples.

Consider:

$$I_K = \frac{1}{K} \sum_{k=1}^{K} \frac{p(x, z^{(k)}; \ )}{q(z^{(k)} \mid x; \ )},$$

where $z^{(1:K)} \quad \prod_{k=1}^{K} q(z^{(k)} \mid x; \ )$.

Note that I_K is an unbiased estimator of $p(x; \ )$:

$$\mathbb{E}_{q(z^{(1:K)} \mid x; \ )} [I_K] = p(x; \ ).$$

Importance Weighted Autoencoder (IWAE) [Burda et al. 2015]

- Flows are a way of tightening the ELBO by making the variational family more flexible.
- Not the only way: can obtain a tighter lower bound on $\log p(x; \;)$ by using multiple importance samples.

Consider:

$$I_K = \frac{1}{K} \sum_{k=1}^{K} \frac{p(x, z^{(k)}; \;)}{q(z^{(k)} \mid x; \;)},$$

where $z^{(1:K)} \quad \prod_{k=1}^{K} q(z^{(k)} \mid x; \;)$.

Note that I_K is an unbiased estimator of $p(x; \;)$:

$$\mathbb{E}_{q(z^{(1:K)} \mid x; \;)} [I_K] = p(x; \;).$$

Importance Weighted Autoencoder (IWAE) [Burda et al. 2015]

Any unbiased estimator of $p(x;\theta)$ can be used to obtain a lower bound, using Jensen's inequality:

$$p(x;\theta) = \mathbb{E}_{q(z^{(1:K)}\,|\,x;\phi)}[I_K]$$

$$\Longrightarrow \log p(x;\theta) \geq \mathbb{E}_{q(z^{(1:K)}\,|\,x;\phi)}[\log I_K]$$

$$= \mathbb{E}_{q(z^{(1:K)}\,|\,x;\phi)}\left[\log \frac{1}{K}\sum_{k=1}^{K}\frac{p(x, z^{(k)};\theta)}{q(z^{(k)}\,|\,x;\phi)}\right]$$

However, can also show [Burda et al. 2015]:

$$\log p(x;\theta) \geq \mathbb{E}[\log I_K] \geq \mathbb{E}[\log I_{K-1}]$$

$$\lim_{K\to\infty} \mathbb{E}[\log I_K] = \log p(x;\theta) \text{ under mild conditions}$$

Importance Weighted Autoencoder (IWAE) [Burda et al. 2015]

$$\mathbb{E}_{q(z^{(1:K)} \mid x; \;)} \left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p(x, z^{(k)}; \;)}{q(z^{(k)} \mid x; \;)} \right]$$

- Note that with $K = 1$, we recover the ELBO.
- Can interpret $\frac{p(x, z^{(k)}; \;)}{q(z^{(k)} \mid x; \;)}$ as importance weights.
- If $q(z \mid x; \;)$ is reparameterizable, we can use the reparameterization trick to optimize $\mathbb{E}[\log I_K]$ directly.
- Otherwise, need score function gradient estimators [Mnih and Rezende 2016].

1. Introduction
2. Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies
   - Sentence VAE
   - Encoder/Decoder with Latent Variables
   - Latent Summaries and Topics

1. Introduction
2. Models
3. Variational Objective
4. Inference Strategies
5. Advanced Topics
6. Case Studies
   - Sentence VAE
   - Encoder/Decoder with Latent Variables
   - Latent Summaries and Topics

Sentence VAE Example [Bowman et al. 2016]

Generative Model (Model 2):

Draw $\mathbf{z}$ $\quad \mathcal{N}(\mathbf{0}, \mathbf{I})$

Draw $x_t \mid \mathbf{z}$ $\quad \mathrm{CRNNLM}(\;, \mathbf{z})$

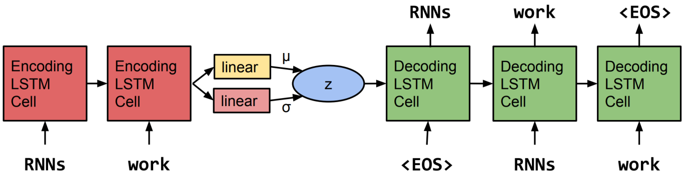
Variational Model (Amortized): Deep Diagonal Gaussians,

$$q(\mathbf{z} \mid x; \;) = \mathcal{N}(\;, \; {}^{\mathbf{2}})$$

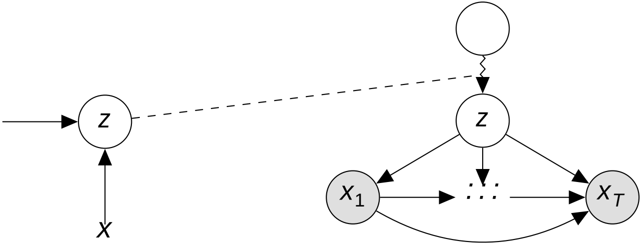
$$\tilde{h}_T = \mathrm{RNN}(x; \;)$$

$$= \mathbf{W}_1 \tilde{h}_T \qquad {}^{\mathbf{2}} = \exp(\mathbf{W}_2 \tilde{h}_T) \qquad = \{\mathbf{W}_1, \mathbf{W}_2, \;\}$$

Sentence VAE Example [Bowman et al. 2016]



(from Bowman et al. [2016])



Issue 1: Posterior Collapse

$$\mathrm{ELBO}(\ ;\ ) = \mathbb{E}_{q(z \mid x;\ )}\left[\log \frac{p(x, z;\ )}{q(z \mid x;\ )}\right]$$

$$= \underbrace{\mathbb{E}_{q(z \mid x;\ )}[\log p(x \mid z;\ )]}_{\text{Reconstruction likelihood}} \quad \underbrace{\mathrm{KL}[q(z \mid x;\ ) \| p(z)]}_{\text{Regularizer}}$$

Model	L/ELBO	Reconstruction	KL
RNN LM	-329.10	-	-
RNN VAE	-330.20	-330.19	0.01

(On Yahoo Corpus from Yang et al. [2017])

Issue 1: Posterior Collapse

x and z become independent, and $p(x; z; \ )$ reduces to a non-LV language model.

Chen et al. [2017]: If it's possible to model $p_?(x)$ without making use of z, then ELBO optimum is at:

$$p_?(x) = p(x \, j \, z; \ ) = p(x; \ ) \qquad q(z \, j \, x; \ ) = p(z)$$

$$\mathrm{KL}[q(z \, j \, x; \ ) k p(z)] = 0$$

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Sentence VAE
Encoder/Decoder with Latent Variables
Latent Summaries and Topics
Conclusion
References

Mitigating Posterior Collapse

Use less powerful likelihood models [Miao et al. 2016; Yang et al. 2017], or "word dropout" [Bowman et al. 2016].

Model	LL/ELBO	Reconstruction	KL
RNN LM	-329.1	-	-
RNN VAE	-330.2	-330.2	0.01
+ Word Drop	-334.2	-332.8	1.44
CNN VAE	-332.1	-322.1	10.0

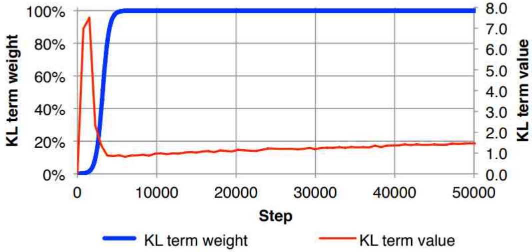
(On Yahoo Corpus from Yang et al. [2017])

Mitigating Posterior Collapse

Gradually anneal multiplier on KL term, i.e.

$$\mathrm{E}_{q(z\,|\,x;\ )}[\log p(x\,|\,z;\ )] \qquad \mathrm{KL}[q(z\,|\,x;\ )\|p(z)]$$

goes from 0 to 1 as training progresses



(from Bowman et al. [2016])

Mitigating Posterior Collapse

Other approaches:

- Use auxiliary losses (e.g. train z as part of a topic model) [Dieng et al. 2017; Wang et al. 2018]
- Use von Mises–Fisher distribution with a fixed concentration parameter [Guu et al. 2017; Xu and Durrett 2018]
- Combine stochastic/amortized variational inference [Kim et al. 2018]
- Add skip connections [Dieng et al. 2018]

In practice, often necessary to combine various methods.

Issue 2: Evaluation

ELBO always lower bounds $\log p(x; \ )$, so can calculate an upper bound on PPL efficiently.

When reporting ELBO, should also separately report,

$$\mathrm{KL}[q(z \mid x; \ ) \| p(z)]$$

to give an indication of how much the latent variable is being "used".

Issue 2: Evaluation

Also can evaluate $\log p(x;\ )$ with importance sampling

$$p(x;\ ) = \mathbb{E}_{q(z \mid x;\ )}\left[\frac{p(x \mid z;\ )p(z)}{q(z \mid x;\ )}\right]$$

$$\frac{1}{K}\sum_{k=1}^{K}\frac{p(x|z^{(k)};\ )p(z^{(k)})}{q(z^{(k)} \mid x;\ )}$$

So

$$\Rightarrow \quad \log p(x;\ ) \quad \log \frac{1}{K}\sum_{k=1}^{K}\frac{p(x|z^{(k)};\ )p(z^{(k)})}{q(z^{(k)} \mid x;\ )}$$

Evaluation

Qualitative evaluation

- Evaluate samples from prior/variational posterior.
- Interpolation in latent space.

i went to the store to buy some groceries .
i store to buy some groceries .
i were to buy any groceries .
horses are to buy any groceries .
horses are to buy any animal .
horses the favorite any animal .
horses the favorite favorite animal .
horses are my favorite animal .

(from Bowman et al. [2016])

Introduction

Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

Sentence VAE

Encoder/Decoder with Latent Variables

Latent Summaries and Topics

Conclusion

References

1 Introduction

2 Models

3 Variational Objective

4 Inference Strategies

5 Advanced Topics

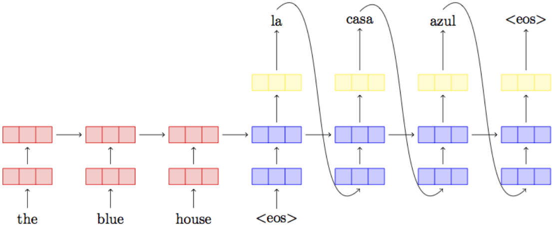
6 Case Studies

Sentence VAE

Encoder/Decoder with Latent Variables

Latent Summaries and Topics

Encoder/Decoder [Sutskever et al. 2014; Cho et al. 2014]



Given: Source information $s = s_1, \ldots, s_M$.

Generative process:

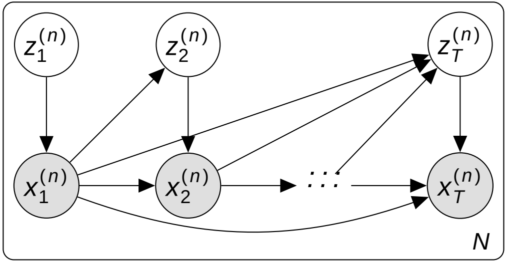
Draw $x_{1:T} \mid s \sim \mathrm{CRNNLM}(\theta, \mathrm{enc}(s))$.

Latent, Per-token Experts [Yang et al. 2018]

Generative process: For $t = 1, \dots, T$,

Draw $z_t \mid x_{<t}; \theta \sim \text{softmax}(U h_t)$.

Draw $x_t \mid z_t, x_{<t}; \theta \sim \text{softmax}(W \tanh(Q_{z_t} h_t); \theta)$



If $U \in \mathbb{R}^{K \times d}$, used K experts; increases the flexibility of per-token distribution.

Case-Study: Latent Per-token Experts [Yang et al. 2018]

Learning: z_t are independent given $x_{<t}$, so we can marginalize at each time-step (Method 3: Conjugacy).

$$\arg\max \log p(x \mid s; \ ) =$$

$$\arg\max \log \prod_{t=1}^{T} \sum_{k=1}^{K} p(z_t = k \mid s; x_{<t}; \ ) \, p(x_t \mid z_t = k; x_{<t}; s; \ ).$$

Test-time:

$$\arg\max_{x_{1:T}} \prod_{t=1}^{T} \sum_{k=1}^{K} p(z_t = k \mid s; x_{<t}; \ ) \, p(x_t \mid z_t = k; x_{<t}; s; \ ).$$

Case-Study: Latent, Per-token Experts [Yang et al. 2018]

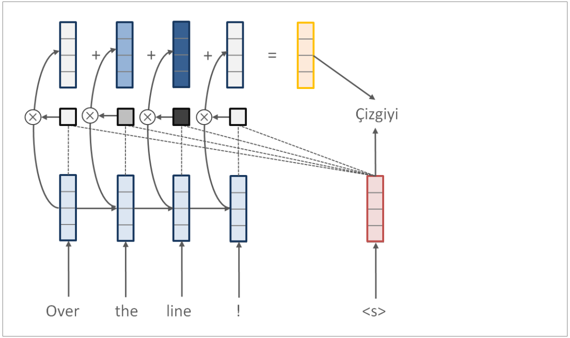
PTB language modeling results (s is constant):

Model	PPL
Merity et al. [2018]	57.30
Softmax-mixture [Yang et al. 2018]	54.44

Dialogue generation results (s is context):

Model	BLEU	
	Prec	Rec
No mixture	14.1	11.1
Softmax-mixture [Yang et al. 2018]	15.7	12.3

Attention [Bahdanau et al. 2015]



Decoding with an attention mechanism:

$$x_t \mid x_{<t}, s \sim \text{softmax}(W[h_t, \sum_{m=1}^{M} \alpha_{t,m}\, \text{enc}(s)_m]).$$

Copy Attention [Gu et al. 2016; Gulcehre et al. 2016]

Copy attention models copying words directly from s.

Generative process: For $t = 1; : : : ; T$,

- Set $_t$ to be attention weights.
- Draw $z_t \, j \, x_{<t}; s$ Bern(MLP($[h_t; \text{enc}(s)]$)).
- If $z_t = 0$
  - Draw $x_t \, j \, z_t; x_{<t}; s$ softmax($W h_t$).
- Else
  - Draw x_t 2 $f s_1; : : : ; s_M g \, j \, z_t; x_{<t}; s$ Cat($_t$).

Copy Attention

Learning: Can maximize the log per-token marginal [Gu et al. 2016], as with per-token experts:

$$\max_{\theta} \log p(x_1, \dots, x_T \mid s; \theta)$$

$$= \max_{\theta} \log \prod_{t=1}^{T} \sum_{z' \in \{0,1\}} p(z_t = z' \mid s, x_{<t}; \theta)\, p(x_t \mid z', x_{<t}, x; \theta).$$

Test-time:

$$\arg\max_{x_{1:T}} \prod_{t=1}^{T} \sum_{z' \in \{0,1\}} p(z_t = z' \mid s, x_{<t}; \theta)\, p(x_t \mid z', x_{<t}, s; \theta).$$

Attention as a Latent Variable [Deng et al. 2018]

Generative process: For $t = 1, \ldots, T$,

- Set $_t$ to be attention weights.
- Draw $z_t \mid x_{<t}, s$ $\quad$ $\mathrm{Cat}(\ _t)$.
- Draw $x_t \mid z_t, x_{<t}, s$ $\quad$ $\mathrm{softmax}(W[h_t; \mathrm{enc}(s_{z_t})]; \ )$.

Attention as a Latent Variable [Deng et al. 2018]

Marginal likelihood under latent attention model:

$$p(x_{1:T} \mid s; \theta) = \prod_{t=1}^{T} \sum_{m=1}^{M} \alpha_{t,m} \, \mathrm{softmax}(W[h_t, \mathrm{enc}(s_m)]; \theta)_{x_t}.$$

Standard attention likelihood:

$$p(x_{1:T} \mid s; \theta) = \prod_{t=1}^{T} \mathrm{softmax}(W[h_t, \sum_{m=1}^{M} \alpha_{t,m} \, \mathrm{enc}(s_m)]; \theta)_{x_t}.$$

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Sentence VAE
Encoder/Decoder with Latent Variables
Latent Summaries and Topics
Conclusion
References

Attention as a Latent Variable [Deng et al. 2018]

Learning Strategy #1: Maximize the log marginal via enumeration as above.

Learning Strategy #2: Maximize the ELBO with AVI:

$$\max_{\theta,\phi} \mathbb{E}_{q(z_t;\,\phi)}[\log p(x_t \mid x_{<t}, z_t, s)] - \mathrm{KL}[q(z_t;\,\phi) \| p(z_t \mid x_{<t}, s)].$$

- $q(z_t \mid x; \phi)$ approximates $p(z_t \mid x_{1:T}, s; \theta)$; implemented with a BLSTM.
- q isn't reparameterizable, so gradients obtained using REINFORCE + baseline.

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Sentence VAE
Encoder/Decoder with Latent Variables
Latent Summaries and Topics
Conclusion
References

Attention as a Latent Variable [Deng et al. 2018]

Test-time: Calculate $p(x_t \mid x_{<t}; S; \ )$ by summing out z_t.

MT Results on IWSLT-2014:

Model	PPL	BLEU
Standard Attn	7.03	32.31
Latent Attn (marginal)	6.33	33.08
Latent Attn (ELBO)	6.13	33.09

Encoder/Decoder with Structured Latent Variables

At least two EMNLP 2018 papers augment encoder/decoder text generation models with *structured* latent variables:

1. Lee et al. [2018] generate $x_{1:T}$ by iteratively refining sequences of words $z_{1:T}$.
2. Wiseman et al. [2018] generate $x_{1:T}$ conditioned on a latent template or plan $z_{1:S}$.

1. Introduction

2. Models

3. Variational Objective

4. Inference Strategies

5. Advanced Topics

6. Case Studies
   - Sentence VAE
   - Encoder/Decoder with Latent Variables
   - Latent Summaries and Topics

Introduction

Models

Variational Objective

Inference Strategies

Advanced Topics

Case Studies

- Sentence VAE
- Encoder/Decoder with Latent Variables
- **Latent Summaries and Topics**

Conclusion

References

Summary as a Latent Variable [Miao and Blunsom 2016]

Generative process for a document $x = x_1, \ldots, x_T$:

- Draw a latent summary $z_1, \ldots, z_M$ RNNLM( )
- Draw $x_1, \ldots, x_T \mid z_{1:M}$ CRNNLM($; z$)

Posterior Inference:

$$p(z_{1:M} \mid x_{1:T}; \ ) = p(\text{summary} \mid \text{document}; \ ).$$

Summary as a Latent Variable [Miao and Blunsom 2016]

Generative process for a document $x = x_1, \ldots, x_T$:

- Draw a latent summary $z_1, \ldots, z_M \quad \text{RNNLM}(\ )$
- Draw $x_1, \ldots, x_T \mid z_{1:M} \quad \text{CRNNLM}(\ , z)$

Posterior Inference:

$$p(z_{1:M} \mid x_{1:T};\ ) = p(\text{summary} \mid \text{document};\ ).$$

Summary as a Latent Variable [Miao and Blunsom 2016]

Learning: Maximize the ELBO with amortized family:

$$\max_{;} \mathbb{E}_{q(z_{1:M};\ )}[\log p(x_{1:T}\, j\, z_{1:M};\ )]\quad \mathrm{KL}[q(z_{1:M};\ )kp(z_{1:M};\ )]$$

$q(z_{1:M};\ )$ approximates $p(z_{1:M}\, j\, x_{1:T};\ )$; also implemented with encoder/decoder RNNs.

$q(z_{1:M};\ )$ not reparameterizable, so gradients use REINFORCE + baselines.

Summary as a Latent Variable [Miao and Blunsom 2016]

Semi-supervised Training: Can also use documents *without* corresponding summaries in training.

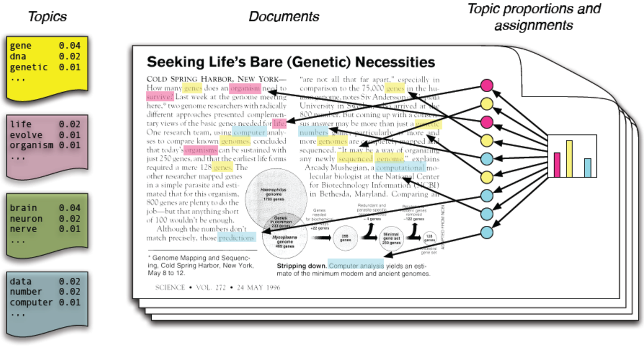
Train $q(z_{1:M};\ )\quad p(z_{1:M}\, j\, x_{1:T};\ )$ with labeled examples.

Infer summary z for an *unlabeled* document with q.

Use inferred z to improve model $p(x_{1:T}\, j\, z_{1:M};\ )$.

Allows for outperforming strictly supervised models!

Topic Models [Blei et al. 2003]



Generative process: for each document $x^{(n)} = x_1^{(n)}, \dots, x_T^{(n)}$,

Draw topic distribution $z_{top}^{(n)} \quad Dir(\quad)$

For $t = 1, \dots, T$:

Draw topic $z_t^{(n)} \quad Cat(z_{top}^{(n)})$

Draw $x_t \quad Cat(\quad_{z_t^{(n)}})$

Simple, Deep Topic Models [Miao et al. 2017]

Motivation: easy to learn deep topic models with VI if $q(\mathbf{z}_{top}^{(n)}; \quad)$ is reparameterizable.

Idea: draw $\mathbf{z}_{top}^{(n)}$ from a transformation of a Gaussian.

Draw $\mathbf{z}_0^{(n)} \quad N(\ _0; \ ^2_0)$

Set $\mathbf{z}_{top}^{(n)} = \text{softmax}(W\mathbf{z}_0^{(n)})$.

Use analogous transformation when drawing from $q(\mathbf{z}_{top}^{(n)}; \quad)$.

Simple, Deep Topic Models [Miao et al. 2017]

Learning Step #1: Marginalize out per-word latents $z_t^{(n)}$.

$$p(\{x^{(n)}\}_{n=1}^{N}; \{\mathbf{z}_{top}^{(n)}\}_{n=1}^{N};\ ) = \prod_{n=1}^{N} p(\mathbf{z}_{top}^{(n)} \mid\ ) \prod_{t=1}^{T} \sum_{k=1}^{K} z_{top;k}^{(n)} \quad {}_{k;x_t^{(n)}}$$

Learning Step #2: Use AVI to optimize resulting ELBO.

$$\max_{;} \mathbb{E}_{q(\mathbf{z}_{top}^{(n)};\ )} \left[\log p(x^{(n)} \mid \mathbf{z}_{top}^{(n)};\ ) \right] \quad \mathrm{KL}[\mathcal{N}(\mathbf{z}_0^{(n)};\ ) \| \mathcal{N}(\mathbf{z}_0^{(n)};\ {}_0;\ {}_0^2)]$$

Simple, Deep Topic Models [Miao et al. 2017]

Perplexities on held-out documents, for three datasets:

Model	MXM	20News	RCV1
OnlineLDA [Ho man et al. 2010]	342	1015	1058
AVI-LDA [Miao et al. 2017]	272	830	602

Introduction
Models
Variational Objective
Inference Strategies
Advanced Topics
Case Studies
Conclusion
References

1 Introduction
2 Models
3 Variational Objective
4 Inference Strategies
5 Advanced Topics
6 Case Studies
7 Conclusion

Deep Latent-Variable NLP: Two Views

Deep Models & LV Models are naturally complementary:

- Rich set of model choices: discrete, continuous, and structured.
- Real applications across NLP including some state-of-the-art models.

Deep Models & LV Models are frustratingly incompatible:

- Many interesting approaches to the problem: reparameterization, score-function, and more.
- Lots of area for research into improved approaches.

Deep Latent-Variable NLP: Two Views

Deep Models & LV Models are naturally complementary:

Rich set of model choices: discrete, continuous, and structured.

Real applications across NLP including some state-of-the-art models.

Deep Models & LV Models are frustratingly incompatible:

Many interesting approaches to the problem: reparameterization, score-function, and more.

Lots of area for research into improved approaches.

Implementation

Modern toolkits make it easy to implement these models.

Combine the flexibility of auto-differentiation for optimization (PyTorch) with distribution and VI libraries (Pyro).

In fact, we have implemented this entire tutorial. See website link: `http://bit.do/lvnlp`

Implementation

Modern toolkits make it easy to implement these models.

Combine the flexibility of auto-differentiation for optimization (PyTorch) with distribution and VI libraries (Pyro).

In fact, we have implemented this entire tutorial. See website link:
`http://bit.do/lvnlp`

Charu C. Aggarwal and ChengXiang Zhai. 2012. A Survey of Text Clustering Algorithms. In *Mining Text Data*, pages 77{128. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of ICLR*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993{1022.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyal, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of CoNLL*.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based N-gram Models of Natural Language. *Computational Linguistics*, 18(4):467{479.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2015. Importance Weighted Autoencoders. In *Proceedings of ICLR*.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2017. Variational Lossy Autoencoder. In *Proceedings of ICLR*.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of the Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1{38.

Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander M. Rush. 2018. Latent Alignment and Variational Attention. In *Proceedings of NIPS*.

Adji B. Dieng, Yoon Kim, Alexander M. Rush, and David M. Blei. 2018. Avoiding Latent Variable Collapse with Generative Skip Models. In *Proceedings of the ICML Workshop on Theoretical Foundations and Applications of Deep Generative Models*.

Adji B. Dieng, Chong Wang, Jianfeng Gao, and John Paisley. 2017. TopicRNN: A Recurrent Neural Network With Long-Range Semantic Dependency. In *Proceedings of ICLR*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12.

Jason Eisner. 2016. Inside-Outside and Forward-Backward Algorithms Are Just Backprop (Tutorial Paper). In *Proceedings of the Workshop on Structured Prediction for NLP*.

Ekaterina Garmash and Christof Monz. 2016. Ensemble Learning for Multi-source Neural Machine Translation. In *Proceedings of COLING*.

Zoubin Ghahramani and Michael I. Jordan. 1996. Factorial Hidden Markov Models. In *Proceedings of NIPS*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631{1640.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 140{149.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2017. Generating Sentences by Editing Prototypes. *arXiv:1709.08878*.

William P Headden III, Mark Johnson, and David McClosky. 2009. Improving Unsupervised Dependency Parsing with Richer Contexts and Smoothing. In *Proceedings of NAACL*.

Marti A Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33{64.

Matthew Ho man, Francis R Bach, and David M Blei. 2010. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856{864.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward Controlled Generation of Text. In *Proceedings of ICML*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geo rey E Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79{87.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. In *Proceedings of ICLR*.

Yoon Kim, Sam Wiseman, Andrew C. Miller, David Sontag, and Alexander M. Rush. 2018. Semi-Amortized Variational Autoencoders. In *Proceedings of ICML*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

Diederik P. Kingma, Tim Salimans, and Max Welling. 2016. Improving Variational Inference with Autoregressive Flow. *arXiv:1606.04934*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought Vectors. In *Proceedings of NIPS*.

Dan Klein and Christopher D Manning. 2004. Corpus-based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Proceedings of ACL*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Re nement. In *Proceedings of EMNLP*.

Stefan Lee, Senthil Purushwalkam Shiva Prakash, Michael Cogswell, Viresh Ranjan, David Crandall, and Dhruv Batra. 2016. Stochastic Multiple Choice Learning for Training Diverse Deep Ensembles. In *Proceedings of NIPS*.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *Proceedings of ICLR*.

Bernard Merialdo. 1994. Tagging English Text with a Probabilistic Model. *Computational Linguistics*, 20(2):155{171.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*.

Yishu Miao and Phil Blunsom. 2016. Language as a Latent Variable: Discrete Generative Models for Sentence Compression. In *Proceedings of EMNLP*.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering Discrete Latent Topics with Neural Variational Inference. In *Proceedings of ICML*.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural Variational Inference for Text Processing. In *Proceedings of ICML*.

Andriy Mnih and Danilo J. Rezende. 2016. Variational Inference for Monte Carlo Objectives. In *Proceedings of ICML*.

Andryi Mnih and Karol Gregor. 2014. Neural Variational Inference and Learning in Belief Networks. In *Proceedings of ICML*.

Anjan Nepal and Alexander Yates. 2013. Factorial Hidden Markov Models for Learning Representations of Natural Language. *arXiv:1312.6168*.

George Papandreou and Alan L. Yuille. 2011. Perturb-and-Map Random Fields: Using Discrete Optimization to Learn and Sample from Energy Models. In *Proceedings of ICCV*.

Danilo J. Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. In *Proceedings of ICML*.

Noah A. Smith and Jason Eisner. 2005. Contrastive Estimation: Training Log-Linear Models on Unlabeled Data. In *Proceedings of ACL*.

Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of NIPS*.

Ke Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. 2016. Unsupervised Neural Hidden Markov Models. In *Proceedings of the Workshop on Structured Prediction for NLP*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836{841. Association for Computational Linguistics.

Wenlin Wang, Zhe Gan, Wenqi Wang, Dinghan Shen, Jiaji Huang, Wei Ping, Sanjeev Satheesh, and Lawrence Carin. 2018. Topic Compositional Neural Language Model. In *Proceedings of AISTATS*.

Peter Willett. 1988. Recent Trends in Hierarchic Document Clustering: A Critical Review. *Information Processing & Management*, 24(5):577{597.

Ronald J. Williams. 1992. Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2018. Learning Neural Templates for Text Generation. In *Proceedings of EMNLP*.

Jiacheng Xu and Greg Durrett. 2018. Spherical Latent Spaces for Stable Variational Autoencoders. In *Proceedings of EMNLP*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2018. Breaking the Softmax Bottleneck: A High-Rank RNN Language Model. In *Proceedings of ICLR*.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. In *Proceedings of ICML*.