One-shot generative modelling

Сергей Бартунов

ФКН НИУ ВШЭ

16 сентября 2016 г.
Generative models

Any distribution over object of interest $x$ is a generative model:

$$p(x|\theta).$$
Generative models

Any distribution over object of interest $x$ is a *generative model*:

$$p(x|\theta).$$

Objects can be generated by independent sampling from the distribution:

$$\hat{x}_1, \hat{x}_2, \ldots \sim p(x|\theta).$$

Example: bag of words language model

$\hat{x} = \{x_1, x_2, \ldots, x_N\}$ is a text of length $N$

$\theta$ is just word frequencies
Generative models

Any distribution over object of interest $\mathbf{x}$ is a *generative model*:

$$ p(\mathbf{x}|\theta). $$

Objects can be generated by independent sampling from the distribution:

$$ \hat{x}_1, \hat{x}_2, \ldots \sim p(\mathbf{x}|\theta). $$

Example: bag of words language model

- $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ is a text of length $N$
- $p(\mathbf{x}|\theta) = \prod_{i=1}^{N} p(x_i|\theta)$ – bag of words model
- $\theta$ is just word frequencies
Latent-variable models

Assumption: object $x$ can be summarized or explained by latent variable $z$:

$$p(x|\theta) = \int p(z|\theta)p(x|z, \theta)dz.$$
Latent-variable models

Assumption: object \( x \) can be summarized or explained by latent variable \( z \):

\[
p(x|\theta) = \int p(z|\theta)p(x|z, \theta)dz.
\]

Example: Latent Dirichlet Allocation

- \( x = \{x_1, x_2, \ldots, x_N\} \) is a text of length \( N \)
- \( z = \{\gamma, t_1, t_2, \ldots, t_N\} \) – latent variable
  - \( p(\gamma|\theta) = \text{Dirichlet}(\alpha_1, \ldots, \alpha_K) \) – document profile
  - \( p(t_i = k|\gamma) = \gamma_k \) – word-topic assignment
- \( p(x|z, \theta) = \prod_{i=1}^{N} p(x_i|\phi_{t_i}) \) – word \( x_i \) is explained by the assigned topic \( t_i \)
- \( \theta = \{\alpha, \phi\} \) – model parameters
Learning in latent-variable models

Given training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ find the best parameters $\theta$: 

$$
\sum_{i=1}^{N} \log p(\mathbf{x}_i | \theta) \rightarrow \max_{\theta}.
$$
Learning in latent-variable models

Given training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ find the best parameters $\theta$:

$$
\sum_{i=1}^{N} \log p(\mathbf{x}_i | \theta) \rightarrow \max_{\theta}.
$$

Exact learning is hard, we resort to variational learning.

- For each $i$ introduce $q(z_i | \lambda_i) \approx p(z_i | \mathbf{x}_i, \theta)$
- Using these approximations derive a variational lower bound:

$$
\mathcal{L}(\theta, \lambda) = \sum_{i=1}^{N} \mathbb{E}_{q(z_i | \lambda_i)} \left[ \log p(\mathbf{x}_i, z_i | \theta) - \log q(z_i | \lambda_i) \right]
$$

- Optimize the lower bound with respect to $\theta$ and $\lambda = \{\lambda_i\}_{i=1}^{N}$. 
Deep latent-variable models

Design decision: $p(x|z, \theta)$ is modelled by a neural network.
Deep latent-variable models

Design decision: $p(x|z, \theta)$ is modelled by a neural network.

Example: neural language model

- $x = \{x_1, x_2, \ldots, x_N\}$ is a text of length $N$
- $p(z) = \mathcal{N}(0, 1) – d$-dimensional std normal
- Hidden-state dynamics:
  - $h_1 = \text{init}(z)$
  - $h_i = \text{state}(h_{i-1}, x_{i-1}, z), \quad i > 1$
- $p(x|z, \theta) = \prod_{i=1}^{N} p(x_i|x_{<i}, z, \theta)$
- $p(x_i = w|x_{<i}, z, \theta) = \frac{\exp(f(w, h_i))}{\sum_{v} \exp(f(v, h_i))}$
- $\theta$ controls $\text{init}$, $\text{state}$ and $f$. 
Variational autoencoders

Design decision: let not only the generative model, but also the approximate posterior $q(z_i|x, \phi) = \mathcal{N}(\mu(x_i, \phi), \sigma(x_i, \phi))$ be modelled by a neural network.
Variational autoencoders

Design decision: let not only the generative model, but also the approximate posterior $q(z_i|x, \phi) = \mathcal{N}(\mu(x_i, \phi), \sigma(x_i, \phi))$ be modelled by a neural network.
Learning in variational autoencoders

Variational lower bound:

\[ \mathcal{L}(\theta, \phi) = \sum_{i=1}^{N} \mathbb{E}_{q(z_i|x_i, \phi)} \left[ \log p(x_i, z_i|\theta) - \log q(z_i|x_i, \phi) \right]. \]
Learning in variational autoencoders

Variational lower bound:

\[ \mathcal{L}(\theta, \phi) = \sum_{i=1}^{N} \mathbb{E}_{q(z_i|x_i, \phi)} \left[ \log p(x_i, z_i|\theta) - \log q(z_i|x_i, \phi) \right]. \]

Reparameterization trick:

\[ \epsilon_i \sim \mathcal{N}(0, 1) \Rightarrow g(\epsilon_i, \phi) = \sigma(x_i, \phi)\epsilon + \mu(x_i, \phi) \sim \mathcal{N}(\mu(x_i, \phi), \sigma(x_i, \phi)). \]
Learning in variational autoencoders

Variational lower bound:

\[
\mathcal{L}(\theta, \phi) = \sum_{i=1}^{N} \mathbb{E}_{q(z_i|x_i, \phi)} \left[ \log p(x_i, z_i|\theta) - \log q(z_i|x_i, \phi) \right].
\]

Reparametrization trick:

\[
\epsilon_i \sim \mathcal{N}(0, 1) \Rightarrow g(\epsilon_i, \phi) = \sigma(x_i, \phi)\epsilon + \mu(x_i, \phi) \sim \mathcal{N}(\mu(x_i, \phi), \sigma(x_i, \phi)).
\]

Monte-carlo estimate:

\[
\hat{\mathcal{L}}(\theta, \phi) = N \left[ \log p(x_i, g(\epsilon_i, \phi)|\theta) - \log q(g(\epsilon_i, \phi)|x_i, \phi) \right],
\]

\[
i \sim \text{Uniform}(1, \ldots, N),
\]

\[
\epsilon \sim \mathcal{N}(0, 1).
\]
Modelling exchangeable data

Can we benefit from relaxation of i.i.d. assumption?

\[ p(x_1, x_2, \ldots, x_N) \neq \prod_{i=1}^{N} p(x_i). \]
Modelling exchangeable data

Can we benefit from relaxation of i.i.d. assumption?

\[ p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \neq \prod_{i=1}^{N} p(\mathbf{x}_i). \]

- By de Finetti’s theorem, there exists a *global* latent variable \( \alpha \) making data conditionally independent:

\[
p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \int p(\alpha) \prod_{i=1}^{N} p(\mathbf{x}_i | \alpha) d\alpha.
\]
Modelling exchangeable data

Can we benefit from relaxation of i.i.d. assumption?

\[ p(x_1, x_2, \ldots, x_N) \neq \prod_{i=1}^{N} p(x_i). \]

▶ By de Finetti’s theorem, there exists a *global* latent variable \( \alpha \) making data conditionally independent:

\[ p(x_1, x_2, \ldots, x_N) = \int p(\alpha) \prod_{i=1}^{N} p(x_i | \alpha) d\alpha. \]

▶ We may consider the following conditional dependence:

\[ p(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} p(x_i | x_{\leq i}). \]
Sequential generative model (Rezende et al, 2016)

A modification of the DRAW (Gregor et al, 2015), can explicitly condition on another image $x'$.

$\rightarrow p(z) = \prod_{t=1}^{T} p(z_t)$

$\rightarrow z_t \sim \mathcal{N}(z_t|0, 1)$

$\rightarrow v_t = f_v(h_{t-1}, x'; \theta_v)$

$\rightarrow h_t = f_h(h_{t-1}, z_t, v_t; \theta_h)$

$\rightarrow c_t = f_c(c_{t-1}, h_t; \theta_c)$

$\rightarrow x \sim p(x|f_o(c_t; \theta_o))$
One-shot generalization
Neural statistical (Edwards & Storkey, 2016)

Latent-variable model implementing de Finetti’s theorem:

\[
p(x_1, \ldots x_N | \theta) = \int p(c | \theta) \prod_{i=1}^{N} \int (p(z_i | c; \theta)p(x_i | z_i, c; \theta)) \, dz_i \, dc
\]
Latent-variable model implementing de Finetti’s theorem:

\[
p(x_1, \ldots, x_N|\theta) = \int p(c|\theta) \prod_{i=1}^{N} \int (p(z_i|c, \theta)p(x_i|z_i, c, \theta)) \, dz_i \, dc
\]

Recognition model:

\[
q(z_1, \ldots, z_N, c|x_1, \ldots, x_N; \phi) = q(c|x_1, \ldots, x_N; \phi) \prod_{i=1}^{N} q(z_i|x_i, c; \phi)
\]
Neural statistical (Edwards & Storkey, 2016)

Latent-variable model implementing de Finetti’s theorem:

\[
p(x_1, \ldots, x_N | \theta) = \int p(c | \theta) \prod_{i=1}^{N} \int (p(z_i | c; \theta) p(x_i | z_i, c; \theta)) \, dz_i \, dc
\]

Recognition model:

\[
q(z_1, \ldots, z_N, c | x_1, \ldots, x_N; \phi) = q(c | x_1, \ldots, x_N; \phi) \prod_{i=1}^{N} q(z_i | x_i, c; \phi)
\]

- Context variable \( c \) represents *global* statistics of the dataset (how the character looks like).
- Local variable \( z \) controls variations applied to the global image of a character.
Training protocol

- Assume there is a distribution $p(D)$ on datasets $D = \{x_1, x_2, \ldots, x_N\}$. 
Training protocol

- Assume there is a distribution $p(D)$ on datasets $D = \{x_1, x_2, \ldots, x_N\}$.
- Maximize the expected likelihood of a dataset:

$$E_{D \sim p(D)} \log p(D|\theta) \rightarrow \max_\theta$$
Training protocol

- Assume there is a distribution $p(D)$ on datasets $D = \{x_1, x_2, \ldots, x_N\}$.
- Maximize the expected likelihood of a dataset:

$$\mathbb{E}_{D \sim p(D)} \log p(D|\theta) \rightarrow \max_{\theta}$$

- Since it's intractable, use variational inference

$$\mathbb{E}_{D \sim p(D)} \log p(D|\theta) \geq \mathbb{E}_{D \sim p(D)} \left[ \log p(\alpha|\theta) - \log q(\alpha|D; \phi) + \sum_{i=1}^{N} \log p(x_i, z_i|\alpha; \theta) - \log q(z_i|x_i, \alpha; \phi) \right]$$

$p(D)$ is constrained in a way that all objects belong to the same class.
Training protocol

- Assume there is a distribution $p(D)$ on datasets $D = \{x_1, x_2, \ldots, x_N\}$.
- Maximize the expected likelihood of a dataset:
  \[
  \mathbb{E}_{D \sim p(D)} \log p(D|\theta) \rightarrow \max_{\theta}
  \]

- Since it's intractable, use variational inference
  \[
  \mathbb{E}_{D \sim p(D)} \log p(D|\theta) \geq \mathbb{E}_{D \sim p(D)} \left[ \log p(\alpha|\theta) - \log q(\alpha|D; \phi) + \sum_{i=1}^{N} \log p(x_i, z_i|\alpha; \theta) - \log q(z_i|x_i, \alpha; \phi) \right]
  \]

- $p(D)$ is constrained in a way that all objects belong to the same class
One-shot generalization

\[ p(x|x_1, \ldots, x_N; \theta) = \int p(c|x_1, \ldots, x_N; \theta)p(x|c; \theta)dc \]

\[ \approx \int q(c|x_1, \ldots, x_N; \phi)p(x|c; \theta)dc \]
One-shot generalization

\[ p(x|x_1, \ldots, x_N; \theta) = \int p(c|x_1, \ldots, x_N; \theta)p(x|c; \theta)dc \]
\[ \approx \int q(c|x_1, \ldots, x_N; \phi)p(x|c; \theta)dc \]

Normal(mu(avg(s)), sigma(avg(s)))

average

statistics

s s s

...
One-shot generalization

\[ p(x|x_1, \ldots, x_N; \theta) = \int p(c|x_1, \ldots, x_N; \theta)p(x|c; \theta)dc \]

\[ \approx \int q(c|x_1, \ldots, x_N; \phi)p(x|c; \theta)dc \]
Generative Matching Network

Explicit conditioning (no global latent variable):

\[ p(x_1, x_2, \ldots, x_N|\theta) = \prod_{i=1}^{N} \int p(z_i|x_{<i}; \theta) p(x_i|z_{<i}, x_{<i}; \theta) dz_i. \]

Recognition model:

\[ q(z_1, \ldots, z_N|x_1, \ldots, x_N; \phi) = \prod_{i=1}^{N} q(z_i|x_{<i}; \phi) \]

No assumptions on the class structure of data! Inspired by (Vinyals et al, 2016).
Generative part

\[ p(x|x_1, \ldots, x_k; \theta) = \int p(z|x_1, \ldots, x_k; \theta)p(x|z, x_1, \ldots, x_k; \theta)dz \]

- \( z \sim \mathcal{N}(z|0, I) \)
- \( K(x_i, z) = \frac{\exp(\cos(f'(z), f(x_i)))}{\sum_{j=1}^{k} \exp(\cos(f'(z), f(x_j)))} \)
- \( h = \sum_{i=1}^{k} K(x_i, z)g(x_i) \)
- \( x \sim p(x|h) \)
Recognition part

\[ q(z|x, x_1, \ldots, x_k; \phi) \]

\[ K(x_i, x) = \frac{\exp(\cos(f(x), f(x_i)))}{\sum_{j=1}^{k} \exp(\cos(f(x), f(x_j)))} \]

\[ h = \sum_{i=1}^{k} K(x_i, x)g(x_i) \]

\[ z \sim \mathcal{N}(z|\mu(h), \Sigma(h)) \]
Training protocol

- Assume there is a distribution $p(D)$ on datasets $D = \{x_1, x_2, \ldots, x_N\}$.
- Maximize the expected likelihood of a dataset:
  
  $$\mathbb{E}_{D \sim p(D)} \log p(D|\theta) \rightarrow \max_{\theta}$$

- Since it’s intractable, use variational inference

  $$\mathbb{E}_{D \sim p(D)} \log p(D|\theta) \geq \mathbb{E}_{D \sim p(D)} \left[ \sum_{i=1}^{N} \log p(x_i, z_i|x_{<i}; \theta) - \log q(z_i|x_i, x_{<i}; \phi) \right]$$

- $p(D)$ is constrained in a way that all objects belong up to $C$ classes
Does it work?

<table>
<thead>
<tr>
<th>Model</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4rd</th>
<th>5th</th>
<th>10th</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMN FC</td>
<td>-101.00</td>
<td>-99.67</td>
<td>-98.06</td>
<td>-97.48</td>
<td>-95.89</td>
<td>-92.11</td>
</tr>
<tr>
<td>GMN conv</td>
<td>-94.35</td>
<td>-92.26</td>
<td>-90.39</td>
<td>-89.75</td>
<td>-88.65</td>
<td>-84.14</td>
</tr>
<tr>
<td>∼ C=3</td>
<td>-95.05</td>
<td>-93.40</td>
<td>-91.87</td>
<td>-91.06</td>
<td>-91.02</td>
<td>-87.55</td>
</tr>
<tr>
<td>IWAE K=50, FC</td>
<td>-103.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seq Gen steps=80</td>
<td>≥ -95.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Рис.: Results on test Omniglot data. Each column is $\mathbb{E}_D \log p(x_i|x_{<i})$. 
One-shot generalization
One-shot generalization
One-shot generalization
One-shot generalization
Future work

- Architecture design
- Direct parameter prediction
  - e.g. filters of convolution
- Curriculum learning
  - Gradual increasing max. number of classes $C$ during the training