Lecture 9:

Variational Autoencoders & Lossy Neural Compression

Robert Bamler • Summer Term of 2023

These slides are part of the course *"Data Compression With and Without Deep Probabilistic Models"* taught at University of Tübingen. More course materials—including video recordings, lecture notes, and problem sets with solutions—are publicly available at `https://robamler.github.io/teaching/compress23/`.

Recall: Variational Inference

- **Idea:**
  - approximate the (inaccessible) true posterior $P(\mathbf{Z} \mid \mathbf{X}=\mathbf{x})$ with a *variational distribution* $Q_\phi(\mathbf{Z})$.
  - Find the best approximation $\phi^* := \arg\max_\phi \text{ELBO}(\phi, \mathbf{x})$.
- **Evidence Lower Bound:** $\boxed{\text{ELBO}(\phi, \mathbf{x}) = \mathbb{E}_{Q_\phi(\mathbf{Z})}\big[\log P(\mathbf{Z}, \mathbf{X}=\mathbf{x}) - \log Q_\phi(\mathbf{Z})\big]}$
  - negative expected net bit rate of bits-back coding: $\text{ELBO}(\phi, \mathbf{x}) = -\mathbb{E}_\mathbf{s}\big[R_\phi^{\text{net}}(\mathbf{x} \mid \mathbf{s})\big]$
  - bound on the evidence: $\text{ELBO}(\phi, \mathbf{x}) = \log P(\mathbf{X}=\mathbf{x}) - D_{\text{KL}}\big(Q_\phi(\mathbf{Z}) \,\|\, P(\mathbf{Z} \mid \mathbf{X}=\mathbf{x})\big) \leq \log P(\mathbf{X}=\mathbf{x})$
  - regularized maximum likelihood: $\text{ELBO}(\phi, \mathbf{x}) = \mathbb{E}_{Q_\phi(\mathbf{Z})}\big[\log P(\mathbf{X}=\mathbf{x} \mid \mathbf{Z})\big] - D_{\text{KL}}\big(Q_\phi(\mathbf{Z}) \,\|\, P(\mathbf{Z})\big)$
  - ~~today:~~ (actually, next week ☺) rate/distortion-tradeoff: $\boxed{\text{ELBO}_\beta(\phi, \mathbf{x}) = \mathbb{E}_{Q_\phi(\mathbf{Z})}\big[\log P(\mathbf{X}=\mathbf{x} \mid \mathbf{Z})\big] - \beta\, D_{\text{KL}}\big(Q_\phi(\mathbf{Z}) \,\|\, P(\mathbf{Z})\big)}$
- **Problems:**
  - What's the generative model $P(\mathbf{Z}, \mathbf{X})$? $\longrightarrow$ variational expectation maximization
  - Expensive "$\arg\max_\phi$" for *each message* $\mathbf{x}$ in both encoder & decoder. $\longrightarrow$ amortized inference

Part 1: Learning the Generative Model

- **Goal:** learn optimal parameters θ^* of the *generative* model $P_\theta(\mathbf{Z}, \mathbf{X}) = P_\theta(\mathbf{Z})\, P_\theta(\mathbf{X} \mid \mathbf{Z})$.
  - Thus, the ELBO now depends on θ, i.e., $\text{ELBO}(\theta, \phi, \mathbf{x}) = Q_\phi(\mathbf{Z})\big[\log P_\theta(\mathbf{Z}, \mathbf{X}=\mathbf{x}) - \log Q_\phi(\mathbf{Z})\big]$
  - **Example:** data $\mathbf{X} = (X_i)_i$ are binarized images, i.e., each X_i is a pixel value $\in \{0, 1\}$.
    $\rightarrow$ Prior is fixed: $P(\mathbf{Z}=\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, I)$ (standard normal distribution)
    $\rightarrow$ Likelihood is parameterized by a (deconvolutional) neural network g_θ:
    $P_\theta(\mathbf{X} \mid \mathbf{Z}) = \prod_i P_\theta(X_i \mid \mathbf{Z})$ with $P_\theta(X_i=1 \mid \mathbf{Z}=\mathbf{z}) = \sigma(g_{\theta,i}(\mathbf{z}))$
    } see class "DecoderModel" in jupyter notebook.
- **Distinguish:**
  - *global* parameters θ^* ("model parameters"):
    $\rightarrow$ specify the *generative model* $P_{\theta^*}(\mathbf{Z}, \mathbf{X})$
    $\rightarrow$ same for all data points $\mathbf{x} \Longrightarrow$ known to both sender & receiver
  - *local* parameters ϕ^* ("variational parameters"):
    $\rightarrow$ specify an approximation $Q_{\phi^*}(\mathbf{Z})$ to the posterior $P_{\theta^*}(\mathbf{Z} \mid \mathbf{X}=\mathbf{x})$ for a *specific data point* $\mathbf{x}$
    $\rightarrow$ different for each data point $\mathbf{x} \Longrightarrow$ not available to the receiver until it has decoded $\mathbf{x}$

Variational Expectation Maximization

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

training algorithm:

initialize $\vartheta \leftarrow$ random
repeat until convergence:
- draw $\underline{x} \sim$ training set
- initialize $\phi \leftarrow$ prior or random
- repeat until convergence:
  - update $\phi \leftarrow \phi + \rho_1 \nabla_\phi \mathrm{ELBO}(\vartheta, \phi, \underline{x})$
- update $\vartheta \leftarrow \vartheta + \rho_2 \nabla_\vartheta \mathrm{ELBO}(\vartheta, \phi, \underline{x})$

expensive inner loop for each training step on ϑ

1. **In order to develop a new compression method:**
   - learn optimal parameters θ^* of the generative model $P_\theta(\mathbf{Z}, \mathbf{X})$:

     $\phi^*(\vartheta, \underline{x}) = \arg\max_\phi \mathrm{ELBO}(\vartheta, \phi, \underline{x})$

     $$\vartheta^* \leftarrow \arg\max_\vartheta \mathbb{E}_{\underline{x} \sim \text{training set}}\left[\mathrm{ELBO}(\vartheta, \phi^*(\vartheta, \underline{x}), \underline{x})\right]$$
     $$= \arg\max_\vartheta \mathbb{E}_{\underline{x} \sim \text{training set}}\left[\max_\phi \mathrm{ELBO}(\vartheta, \phi, \underline{x})\right]$$
2. **Share the learned generative model $P_{\theta^*}(\mathbf{Z}, \mathbf{X})$ between sender & receiver.**
3. **In deployment:** encode / decode a given data point $\mathbf{x}$
   - Use entropy model $Q_{\phi^*}(\mathbf{Z})$.

     ϕ^* depends on $\underline{x}$, so both encoder & decoder need to find it by running (expensive) SGD:

     $$\phi^* \leftarrow \arg\max_\phi \mathrm{ELBO}(\vartheta^*, \phi, \underline{x})$$

Part 2: Learning How to Do Inference (Fast)

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

- **Problems:**
  1. Learning the generative model requires an expensive inner loop *for every training step*.
  2. Expensive optimization over ϕ for *each message* $\mathbf{x}$ *we want compress / decompress*.
- **Solution:** *amortized* variational inference
  - learn a mapping f from $\mathbf{x}$ to variational parameters
    such that setting $\phi \leftarrow f(\mathbf{x})$ approximately maximizes $\mathrm{ELBO}(\theta^*, \phi, \mathbf{x})$ for a given $\mathbf{x}$.
  - **Notation:** inference network $f_\phi(\mathbf{x})$; variational distribution $Q_\phi(\mathbf{Z} \,|\, \mathbf{X} = \mathbf{x})$ ← in the notation we've used so far, this would be $Q_{f_\phi(\underline{x})}(\underline{Z})$
  - **Example:** Gaussian mean field variational distribution:
    → inference network $f_\phi(\mathbf{x}) = (\boldsymbol{\mu}_\phi(\mathbf{x}), \log \boldsymbol{\sigma}^2_\phi(\mathbf{x}))$ outputs means and (log) variances
    → these parameterize a variational distribution $Q_\phi(\mathbf{Z} \,|\, \mathbf{x}) = \mathcal{N}\big(\boldsymbol{\mu}_\phi(\mathbf{x}), \operatorname{diag}\big(\sigma^2_{\phi,1}(\mathbf{x}), \ldots, \sigma^2_{\phi,k}(\mathbf{x})\big)\big)$

training algorithm now:
- initialize $\vartheta, \phi \leftarrow$ random
- repeat until convergence:
  - draw $\underline{x} \sim$ training set
  - update $(\vartheta, \phi) \leftarrow (\vartheta, \phi) + \rho \nabla_{\vartheta,\phi} \mathrm{ELBO}(\vartheta, \phi, \underline{x})$

ELBO for amortized inference:

$$\mathrm{ELBO}(\vartheta, \phi, \underline{x}) = \mathbb{E}_{Q_\phi(\underline{Z} | \underline{X} = \underline{x})}\left[\log P(\underline{Z}, \underline{X} = \underline{x}) - \log Q_\phi(\underline{Z} | \underline{X} = \underline{x})\right]$$

⇒ no expensive inner loop; ϑ and ϕ are learned concurrently

Variational Autoencoders (VAEs)

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Combine variational expectation maximization with amortized variational inference. That's all.

- **Lossless compression with variational autoencoders:**
  - use bits-back trick → Problem 9.1
- **Lossy compression with variational autoencoders:**
  - **Example:** data $\mathbf{X} = (X_i)_i$ are color images, i.e., each X_i is a continuous RGB value $\in [0, 1]$.
    → Prior may be learned, e.g.: $P_\theta(\mathbf{Z} = \mathbf{z}) = \mathcal{N}\big(\mathbf{z}; 0, \operatorname{diag}(\sigma_1^2, \ldots, \sigma^2_{\texttt{num\_channels}})^{\otimes \texttt{spatial\_dim}}\big)$
    → Likelihood is parameterized by a (deconvolutional) neural network g_θ:
    $P_\theta(\mathbf{X} \,|\, \mathbf{Z}) = \prod_i P_\theta(X_i \,|\, \mathbf{Z})$ with density function $p_\theta(x_i \,|\, \mathbf{Z} = \mathbf{z}) = \mathcal{N}(x_i; g_{\theta,i}(\mathbf{z}), \frac{1}{\beta} I)$
  - **Idea:** just use $g_{\theta,i}(\mathbf{z})$ as the reconstruction of an image.
    (Don't bother using the likelihood $P_\theta(\mathbf{X} \,|\, \mathbf{Z} = \mathbf{z})$ to encode the true image.)
  - Likelihood no longer has a probabilistic meaning. But $-\log P_\theta(\mathbf{X} \,|\, \mathbf{Z} = \mathbf{z})$ is a *distortion* metric.
    $\Longrightarrow$ ELBO becomes a *rate-distortion* trade-off
  - *next week*