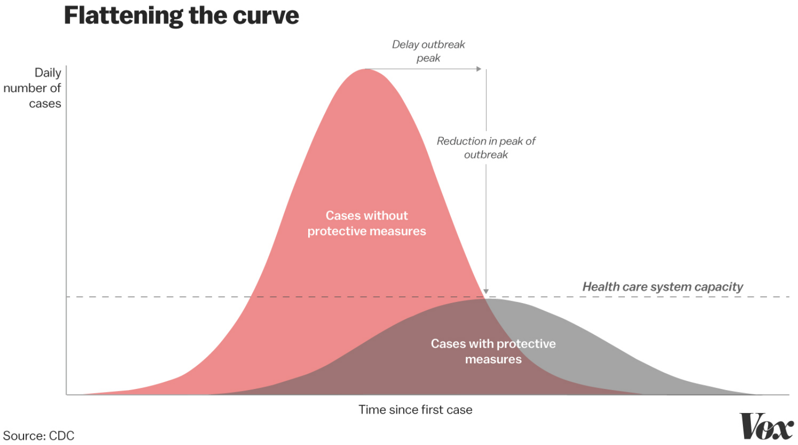
CS 533: Natural Language Processing

Autoencoders and VAEs

Karl Stratos

Rutgers University

Aside: Protective Measures are Meaningful



Logistics

- Set up 1-1 meeting for proposal feedback (March 25-27)
- Proposal and A4 due March 24
- Exam: discussion

Agenda

- EM: loose ends (hard EM)
- Autoencoders and VAEs
- VAE training techniques

Recap: Latent-Variable Generative Models (LVGMs)

- Observed data comes from the population distribution $\mathbf{pop}_X$
- LVGM: Model defining a joint distribution over X and Z

$$p_{XZ}(x, z) = p_Z(z) \times p_{X|Z}(x|z)$$

- Learning: Estimate p_{XZ} by maximizing log-likelihood of data $x^{(1)} \dots x^{(N)} \sim \mathbf{pop}_X$

$$\max_{p_{XZ}} \sum_{i=1}^{N} \log \underbrace{\sum_{z \in \mathcal{Z}} p_{XZ}(x^{(i)}, z)}_{p_X(x^{(i)})}$$

EM: Coordinate Ascent on ELBO

Input: data $x^{(1)} \dots x^{(N)} \sim \mathbf{pop}_X$, definition of p_{XZ}
Output: local optimum of

$$\max_{p_{XZ}} \sum_{i=1}^{N} \log \sum_{z \in \mathcal{Z}} p_{XZ}(x^{(i)}, z)$$

1. Initialize p_{XZ} (e.g., random distribution).
2. Repeat until convergence:

$$q_{Z|X}(z|x^{(i)}) \leftarrow \frac{p_{XZ}(x^{(i)}, z)}{\sum_{z' \in \mathcal{Z}} p_{XZ}(x^{(i)}, z')} \quad \forall z \in \mathcal{Z},\ i = 1 \dots N$$

$$p_{XZ} \leftarrow \arg\max_{\bar{p}_{XZ}} \sum_{i=1}^{N} \sum_{z \in \mathcal{Z}} q_{Z|X}(z|x^{(i)}) \log p_{XZ}(x^{(i)}, z)$$

3. Return p_{XZ}

Hard EM: Coordinate Ascent on a Different Objective

Input: data $x^{(1)} \dots x^{(N)} \sim \mathbf{pop}_X$, definition of p_{XZ}
Output: local optimum of

$$\max_{p_{XZ},\,(z_1 \dots z_N) \in \mathcal{Z}^N} \sum_{i=1}^{N} \log p_{XZ}(x^{(i)}, z_i)$$

1. Initialize p_{XZ} (e.g., random distribution).
2. Repeat until convergence:

$$(z_1 \dots z_N) \leftarrow \operatorname*{arg\,max}_{(\bar{z}_1 \dots \bar{z}_N) \in \mathcal{Z}^N} \sum_{i=1}^{N} \log p_{XZ}(x^{(i)}, \bar{z}_i)$$

$$p_{XZ} \leftarrow \operatorname*{arg\,max}_{\bar{p}_{XZ}} \sum_{i=1}^{N} \log p_{XZ}(x^{(i)}, z_i)$$

3. Return p_{XZ}

K-Means: Special Case of Hard EM

- $x \in \mathbb{R}^d$, $z \in \{1 \ldots K\}$

$$p_{XZ}(x, z) = \frac{1}{K} \times \mathcal{N}(x; \mu_z, I_d)$$

- Model parameters to learn: $\mu_1 \ldots \mu_K \in \mathbb{R}^d$
- Negative log joint probability as a function of parameters

$$-\log p_{XZ}(x, z) \equiv ||x - \mu_z||^2$$

- Observed $x^{(1)} \ldots x^{(N)} \in \mathbb{R}^d$, latents $z_1 \ldots z_N \in \{1 \ldots K\}$

$$z_i \leftarrow \operatorname*{arg\,min}_{z \in \{1 \ldots K\}} \left|\left| x^{(i)} - \mu_z \right|\right|^2$$

$$\mu_k \leftarrow \operatorname*{arg\,min}_{\mu \in \{1 \ldots K\}} \sum_{i=1}^{N} \left|\left| x^{(i)} - \mu_{z_i} \right|\right|^2 = \frac{1}{\textbf{count}(z = k)} \sum_{i=1:\ z_i = k}^{N} x^{(i)}$$

Setting

- Neural autoencoding: observed X, latent Z
- Running example
  - X: sentence
  - Z: m-dimensional real-valued vector
- We need to define
  - $q_{Z|X}$: **encoder** that transforms a sentence into a distribution over $\mathbb{R}^m$
  - $p_{X|Z}$: **decoder** that transforms a vector $z \in \mathbb{R}^m$ into a distribution over sentences
  - p_Z: **prior** that defines a distribution over $\mathbb{R}^m$
- Distributions parameterized by neural networks

Example Encoder: LSTM + Gaussian

- **Input.** Sentence $x \in V^T$
- **Parameters.** Word embeddings $E \in \mathbb{R}^{|V| \times d}$, LSTMCell $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$, feedforward $\mathrm{FF}_1 : \mathbb{R}^d \to \mathbb{R}^{2m}$
- **Forward.**

$$\begin{aligned}
h_1, c_1 &\leftarrow \mathrm{LSTMCell}(E_{x_1}, (0_d, 0_d)) \\
h_2, c_2 &\leftarrow \mathrm{LSTMCell}(E_{x_2}, (h_1, c_1)) \\
&\vdots \\
h_T, c_T &\leftarrow \mathrm{LSTMCell}(E_{x_T}, (h_{T-1}, c_{T-1})) \\
\begin{bmatrix} \mu(x) \\ \sigma^2(x) \end{bmatrix} &\leftarrow \mathrm{FF}_1(h_T)
\end{aligned}$$

- Distribution over $\mathbb{R}^m$ conditioned on x

$$q_{Z|X}(\cdot|x) = \mathcal{N}(\mu(x), \mathrm{diag}(\sigma^2(x)))$$

Example Decoder: Conditional Language Model

- **Input.** Vector $z \in \mathbb{R}^m$
- **Parameters.** Word embeddings $E \in \mathbb{R}^{|V| \times d}$ (often tied with encoder), LSTMCell $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$, feedforward $\mathrm{FF}_2 : \mathbb{R}^m \to \mathbb{R}^d \times \mathbb{R}^d$
- **Forward.** Given sentence $y \in V^L$ compute its probability conditioned on z by

$$
\begin{aligned}
h_1, c_1 &\leftarrow \mathrm{LSTMCell}(E_{y_1}, \mathrm{FF}_2(z)) \\
h_2, c_2 &\leftarrow \mathrm{LSTMCell}(E_{y_2}, (h_1, c_1)) \\
&\vdots \\
h_L, c_L &\leftarrow \mathrm{LSTMCell}(E_{y_L}, (h_{L-1}, c_{L-1}))
\end{aligned}
$$

$$
p_{X|Z}(y|z) = \prod_{l=1}^{L} \underbrace{\mathsf{softmax}_{y_l}(E h_{l-1})}_{p(y_l|z, y_{<l})}
$$

Example Prior: Isotropic Gaussian

- Simplest: fixed standard normal $p_Z = \mathcal{N}(0_m, I_m)$.
  - **Parameters.** None

- Can also make it more expressive, for instance a mixture of K diagonal Gaussians

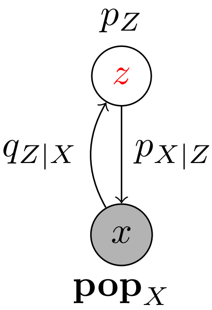
$$p_Z = \sum_{k=1}^{K} \mathrm{softmax}_k(\gamma) \times \mathcal{N}(\mu_k, \mathsf{diag}(\sigma_k^2))$$

  - **Parameters.** $\gamma \in \mathbb{R}^m$ and $\mu_k, \sigma_k^2 \in \mathbb{R}^m$ for $k = 1 \ldots K$
  - Multimodal instead of unimodal

Summary

- Sentence X, d-dimensional vector Z
- Learnable parameters
  - Word embeddings E shared by encoder and decoder
  - LSTM and feedforward parameters in $q_{Z|X}$
  - LSTM and feedforward parameters in $p_{X|Z}$
  - (Optional) Parameters in the prior p_Z
- We will now consider learning all these parameters together in the **autoencoding** framework

Autoencoders (AEs)



$q_{Z|X}$: encoder

$p_{X|Z}$: decoder

p_Z : prior

Objective.

$$\max_{p_Z,\, p_{X|Z},\, q_{Z|X}} \underbrace{\mathop{\mathbf{E}}_{\substack{x \sim \mathbf{pop}_X \\ z \sim q_{Z|X}(\cdot|x)}} \left[\log p_{X|Z}(x|z)\right]}_{\text{reconstruction}} + \underbrace{R(\mathbf{pop}_X, p_Z, p_{X|Z}, q_{Z|X})}_{\text{regularization}}$$

Naive Autoencoders

Objective

$$\max_{p_{X|Z},\,\mathrm{LSTM}} \mathop{\mathbf{E}}_{x \sim \mathbf{pop}_X} \left[\log p_{X|Z}(x | \mathrm{LSTM}(x)) \right]$$

- Deterministic encoding: equivalent to learning a point-mass encoder

$$q_{Z|X}(\mathrm{LSTM}(x)|x) = 1$$

- No regularization (hence no role for prior)

Denoising Autoencoders

Objective

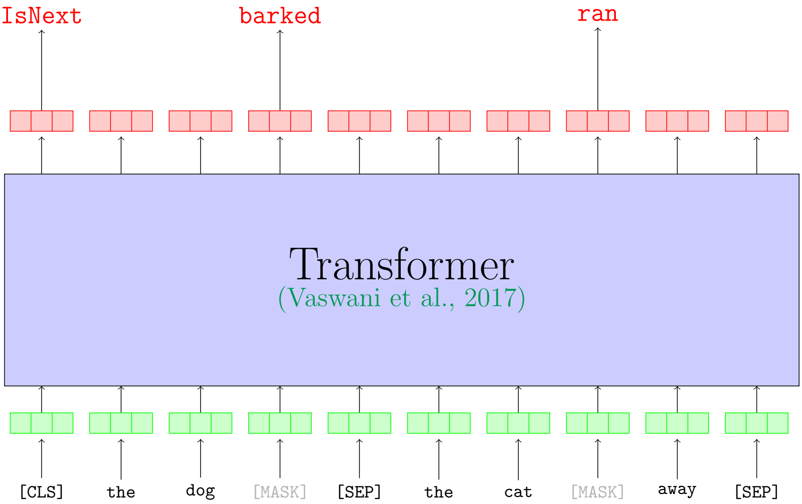
$$\max_{p_{X|Z},\, \text{LSTM}} \mathop{\mathbf{E}}_{\substack{x \sim \mathbf{pop}_X \\ \epsilon \sim p_{\mathcal{E}}}} \left[\log p_{X|Z}(x|\text{LSTM}(x+\epsilon)) \right]$$

- Noise introduced at input, reconstruct original input
- Equivalent to learning encoder

$$q_{Z|X}(\text{LSTM}(x+\epsilon)|x) = p_{\mathcal{E}}(\epsilon)$$

- Still no regularization, so no prior
- Example: masked language modeling

BERT as Denoising AE (Devlin et al., 2019)



Variational Autoencoders (VAEs)

Objective

$$\max_{p_Z,\, p_{X|Z},\, q_{Z|X}} \mathop{\mathbf{E}}_{\substack{x \sim \mathbf{pop}_X \\ z \sim q_{Z|X}(\cdot|x)}} \left[\log p_{X|Z}(x|z)\right] - D_{\mathrm{KL}}(q_{Z|X} || p_Z)$$

- Great deal of flexibility in terms of how to optimize it
- Popular approach for the current setting
  - Optimize the reconstruction term by **sampling + reparameterization trick**

$$z \sim q_{Z|X}(\cdot|x) \qquad \Leftrightarrow \qquad \begin{aligned} &\epsilon \sim \mathcal{N}(0_m, I_m) \\ &z = \mu(x) + \sigma(x) \odot \epsilon \end{aligned}$$

  - Optimize the KL term in closed form

$$\begin{aligned} &D_{\mathrm{KL}}(\mathcal{N}(\mu(x), \mathrm{diag}(\sigma^2(x))) || \mathcal{N}(0_m, I_m)) \\ &= \frac{1}{2}\left(\sum_{i=1}^{m} \sigma_i^2(x) + \mu_i^2(x) - 1 - \log \sigma_i^2(x)\right) \end{aligned}$$

VAE Loss: Concrete Steps

Given a sentence $x \sim \mathbf{pop}_X$ (in general a minibatch)

1. **Encoding**. Run the encoder to calculate the Gaussian parameters $\mu(x), \sigma^2(x) \in \mathbb{R}^m$

$$\mu(x), \sigma^2(x) \leftarrow \textbf{Encoder}(x)$$

2. **KL**. Calculate the KL term

$$\kappa \leftarrow \frac{1}{2}\left(\sum_{i=1}^{m} \sigma_i^2(x) + \mu_i^2(x) - 1 - \log \sigma_i^2(x)\right)$$

3. **Reconstruction**. Estimate the reconstruction term by sampling + reparameterization trick

$$\rho \leftarrow \textbf{DecoderNLL}(x, \mu(x) + \sigma(x) \odot \epsilon) \qquad \epsilon \sim \mathcal{N}(0_m, I_m)$$

4. **Loss**. Take a gradient step (wrt. all parameters) on $\rho - \beta\kappa$ where β is some weight.

Uses of VAEs

- **Representation learning.** Run encoder on a sentence x to obtain its m-dimensional "meaning" vector
- **Controlled generation.** Run decoder on some seed vector to conditionally generate sentences
  - Can "interpolate" between two sentences x_1, x_2

$$
\begin{aligned}
z_1 &\sim q_{Z|X}(\cdot|x_1) \\
z_2 &\sim q_{Z|X}(\cdot|x_2) \\
x_\alpha &\leftarrow \textbf{Decode}(\alpha z_1 + (1-\alpha) z_2) \qquad\qquad \alpha \in [0,1]
\end{aligned}
$$

Interpolation Examples

the girl is drinking milk with the camera .

---

the girl is drinking milk with the camera .
the girl is drinking milk with her hands .
the girl is drinking water with a bucket .
the girl is using a camera .
two girls are outside with a blue umbrella .
two girls are outside with a blue umbrella .
two girls are outside with a dog .
two girls are taking a picture of a tree .
two guys are on a bench .

---

two guys are on a boat .

two boys are at a beach .

---

two boys are at a beach .
two men are looking at a man in a wheelchair .
the children are at the beach .
the children are looking at the sky .
a woman is looking at a man in a wheelchair .
a woman is looking at a man in a wheelchair .
a woman is looking at a map .
a woman is waiting for a bus to come out of the road .
a woman is waiting for a bus to come out of the city .

---

a woman is waiting for a bus .

A Surprisingly Effective Fix for Deep Latent Variable Modeling of Text (Li et al., 2019)

VAEs in Computer Vision

Random (never before seen) faces sampled from VAE decoder!

Generating Diverse High-Fidelity Images with VQ-VAE-2 (Razavi et al., 2019)

VAE is EM

VAE Objective

$$\mathop{\mathbf{E}}_{\substack{x \sim \mathbf{pop}_X \\ z \sim q_{Z|X}(\cdot|x)}} \left[\log p_{X|Z}(x|z) \right] - D_{\mathrm{KL}}(q_{Z|X} || p_Z) = \mathrm{ELBO}(p_{XZ}, q_{Z|X})$$

- Thus when you optimize VAE you are maximizing a lower bound on marginal log likelihood defined by your LVGM
- Taking gradient steps for decoder/encoder/prior simultaneously is alternating optimization of ELBO
- Difference with the classical EM: we no longer insist on solving the E step exactly (i.e., setting $q_{Z|X} = p_{Z|X}$)
  - Train a separate variational model $q_{Z|X}$ alongside p_{XZ}

Practical Issues

- Posterior collapse
- Quantities to monitor

VAE Objective: Cheats

$$\min_{p_{X|Z},\, q_{Z|X}} \mathop{\mathbf{E}}_{\substack{x \sim \mathbf{pop}_X \\ z \sim q_{Z|X}(\cdot|x)}} \left[-\log p_{X|Z}(x|z) \right] + D_{\mathrm{KL}}(q_{Z|X} || \mathcal{N}(0_m, I_m))$$

What's one undesirable strategy to minimize the VAE objective?

Posterior Collapse

Annihilate the KL term by setting

$$q_{Z|X}(\cdot|x) = \mathcal{N}(0_m, I_m) \qquad \forall x \in \mathcal{X}$$

which leaves us with

$$\min_{p_{X|Z}} \underset{\substack{x \sim \mathbf{pop}_X \\ z \sim \mathcal{N}(0_m, I_m)}}{\mathbf{E}} \left[-\log p_{X|Z}(x|z) \right]$$

The decoder $p_{X|Z}$ will ignore z!

Without Addressing Posterior Collapse

Posterior distribution

$$q_{Z|X}(\cdot|\texttt{The company said it expects to report net income of \$UNK-NUM million})$$
$$= q_{Z|X}(\cdot|\texttt{The two sides hadn't met since Oct. 18.})$$
$$= q_{Z|X}(\cdot|\texttt{The inquiry soon focused on the judge.})$$
$$\vdots$$
$$= q_{Z|X}(\cdot|\texttt{Whatever sentence you provide})$$
$$= \mathcal{N}(0_m, I_m)$$

Greedy decoding from $p_{X|Z}(\cdot|z)$

$z = (0.1, 0.3, \ldots, -0.7)$ → `The company said it expects to report net income of $UNK-NUM million`

$z = (-0.6, 0.2 \ldots, 0.2)$ → `The company said it expects to report net income of $UNK-NUM million`

$\vdots$

$z = (0.2, 0.1 \ldots, 0.1)$ → `The company said it expects to report net income of $UNK-NUM million`

$z = (-0.8, -0.5 \ldots, -0.5)$ → `The company said it expects to report net income of $UNK-NUM million`

Tricks to Address Posterior Collapse

- Free bits (Kingma et al., 2016): replace KL term with

$$\kappa \leftarrow \sum_{i=1}^{m} \max \left\{ \lambda, D_{\mathrm{KL}}(q_{Z_i|X} || \mathcal{N}(0,1)) \right\}$$

  $\lambda = 1 \ldots 10$

- KL annealing (Bowman et al., 2016): weight on KL gradually increasing from 0 to 1 for the first 10 epochs

$$0 \times \kappa \quad 0.001 \times \kappa \quad 0.002 \times \kappa \quad \ldots \quad 0.999 \times \kappa \quad 1 \times \kappa$$

- Current best practice (Li et al., 2019): do both with encoder pretraining
  - Pretrain without KL term
  - Reset decoder
  - Train with annealing on the free-bits KL term

Quantities to Monitor During Training

- NLL ($\neq$ -ELBO)

$$\mathop{\mathbf{E}}_{x \sim \mathbf{pop}} [\log p_X(x)] = \mathop{\mathbf{E}}_{x \sim \mathbf{pop}} \left[\log \mathop{\mathbf{E}}_{z \sim q_{Z|X}(\cdot|x)} \left[\frac{p_{XZ}(x, z)}{q_{Z|X}(z|x)} \right] \right]$$

- -ELBO
  - Reconstruction error
  - KL
- Mutual information between X and Z
- Number of active units (Burda et al., 2016)

Other VAE Models in NLP

- "Document hashing": https://arxiv.org/pdf/1908.11078.pdf
- See introduction of Pelsmaeker and Aziz (2019) for other examples: https://arxiv.org/pdf/1904.08194.pdf