Deep Learning

Lecture 11

Fernando Perez-Cruz
based on Thomas Hofmann lectures

Swiss Data Science Center
ETH Zurich and EPFL – datascience.ch

December 10, 2018

Overview

1. Implicit Models
2. VAE Models
3. Deep Gaussian Models
4. Generative Adverserial Models

Section 1

Implicit Models

Implicit Models

- Statistical models via: generating stochastic mechanism or simulation process
  - aka: 'implicit' models
- Deep implicit models
  - latent code $\mathbb{R}^d \ni \mathbf{z} \sim \pi(\mathbf{z})$, e.g. $\pi(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - parameterized mechanism $F_\theta : \mathbb{R}^d \to \mathbb{R}^m$
  - induced distribution $\mathbb{R}^m \ni \mathbf{x} \sim p_\theta(\mathbf{x})$
  - sampling is easy: random vector + forward propagation

Noise Contrastive Estimation

- Bootstrap generative models via classification problems
- Reduce density estimation to binary classification (Gutmann, Hyvärinen, 2012)
- Define joint probability (p_n: 'contrastive' distribution)

$$\tilde{p}(\mathbf{x}, y=1) = \tfrac{1}{2}p(\mathbf{x}), \quad \tilde{p}(\mathbf{x}, y=0) = \tfrac{1}{2}p_n(\mathbf{x})\,.$$

- Probabilistic classifier (induced by $\bar{p}_\theta$)

$$q_\theta = \frac{\alpha \bar{p}_\theta}{\alpha \bar{p}_\theta + p_n}, \quad \alpha > 0$$

  - Bayes optimal if $\alpha \bar{p}_\theta = p$
  - minimize logistic loss with regard to θ and $\alpha > 0$

Noise Contrastive Estimation: Theory

- NCE has a theory behind it ...
- Question #1: Is this estimator for θ consistent? Yes!
  - Gutmann, Hyvärinen, 2012, Theorem 2: as long as p_n is dominating p
- Question #2: Is the estimator statistically efficient? Sometimes, but generally 'no'.
  - Gutmann, Hyvärinen, 2012, Theorem 3: much worse than Cramer-Rao bound if p_n very different from p
- NCE is not the final answer

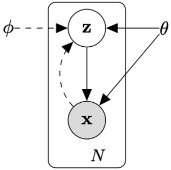
Section 2

VAE Models

Variational Auto Encoder

(Kingma and Welling., 2014)

- Use a neural network for generative modeling.
- Posterior non-tractable, approximate by another NN.



- The marginal log-likelihood:

$$p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

where

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) = D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)})$$

- The second term is the variational lower bound:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right]$$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right]$$

Variational Lower Bound derivatives

- We want to differentiate it with respect to θ and ϕ.
- Naïve Monte Carlo Estimate:

$$\nabla_{\phi}\mathbb{E}_{q_{\phi}(\mathbf{z})}\left[f(\mathbf{z})\right] = \mathbb{E}_{q_{\phi}(\mathbf{z})}\left[f(\mathbf{z})\nabla_{q_{\phi}(\mathbf{z})}\log q_{\phi}(\mathbf{z})\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z})\nabla_{q_{\phi}(\mathbf{z}^{(l)})}\log q_{\phi}(\mathbf{z}^{(l)})$$

  with

$$\mathbf{z}^{(l)} \sim q_{\phi}(\mathbf{z}|\mathbf{x}^{(i)})$$

  yields high variance estimators.
- Re-parametrization trick, substitute

$$\widetilde{\mathbf{z}} \sim q_{\phi}(\mathbf{z}|\mathbf{x})$$

  by

$$\widetilde{\mathbf{z}} = g_{\phi}(\boldsymbol{\epsilon}, \mathbf{x}) \quad \text{with} \quad \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$$

General Stochastic Gradient Variational Bayes

- We can now form the following Monte Carlo Estimates:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[f(\mathbf{z})\right] = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[f(g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}^{(i)}))\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(l)}, \mathbf{x}^{(i)}))$$

with

$$\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$$

- Leading to the General Stochastic Gradient Variational Bayes:

$$\widetilde{\mathcal{L}}^{A}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \frac{1}{L}\sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})$$

with

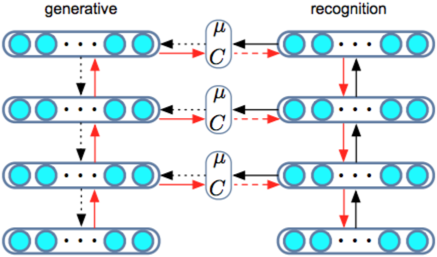
$$\mathbf{z}^{(i,l)} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)}) \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$$

Section 3

Deep Gaussian Models

Deep Latent Gaussian Models

- two coupled networks:
  top-down (generative) and
  bottom-up (recognition)
- forward pass:
  deterministic recognition,
  sampled generative
- backward pass:
  deterministic, but:
  stochastic backpropagation
- cf. Rezende, Mohamed & Wierstra, 2014



Deep Latent Gaussian Model

Generalize factor analysis idea with depth (Rezende et al., 2014).

Noise variables

$$\mathbf{z}^l \stackrel{iid}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad l = 1, \ldots, L$$

Hidden activities (top-down: $\mathbf{h}^L \to \mathbf{h}^1$)

$$\mathbf{h}^L = \mathbf{W}^L \mathbf{z}^L, \quad \mathbf{h}^l = \underbrace{F^l(\mathbf{h}^{l+1})}_{\text{deterministic}} + \underbrace{\mathbf{W}^l \mathbf{z}^l}_{\text{stochastic}}$$

Hidden layer (conditional) distribution [vs. factor analysis]

$$\mathbf{h}|\mathbf{h}^+ \sim \mathcal{N}\left(F(\mathbf{h}^+), \mathbf{W}\mathbf{W}^\top\right) \quad \left[\text{vs. } \mathbf{x}|\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{W}\mathbf{W}^T + \boldsymbol{\Sigma})\right]$$

ELBO: Evidence Lower BOund

Restrict q to (possibly) simpler family

Variational distributions, e.g.:

$$q(\mathbf{z};\mathbf{x}) = \prod_{l=1}^{L} q(\mathbf{z}^l;\mathbf{x}), \quad \mathbf{z}^l \sim \mathcal{N}(\boldsymbol{\mu}^l(\mathbf{x}), \boldsymbol{\Sigma}^l(\mathbf{x}))$$

We need to learn functions $\mathbf{x} \mapsto \boldsymbol{\mu}^l(\mathbf{x})$ (similarly for the covariance)

Use another DNN: inference (or recognition) network

Inference Network

Recognition model

$$\mathbf{x} \overset{\vartheta}{\mapsto} (\boldsymbol{\mu}^l, \boldsymbol{\Sigma}^l)_{l=1}^L \mapsto q \sim \mathcal{N}(\dots)$$

- mapping can be realized by independent (deep) network
- parametric form with parameters ϑ: generalization across $\mathbf{x}$, aka amortized inference
- KL-divergence can be though of as regularization
- practicalities: constrained precision matrix, e.g.

$$\boldsymbol{\Sigma}^{-1} = \underbrace{\mathbf{D}}_{\text{diagonal}} + \underbrace{\mathbf{u}\mathbf{u}^\top}_{\text{rank } 1}$$

Generative Model Optimization

First assume that for given $\mathbf{x}$, $q(\mathbf{z}|\mathbf{x})$ is fixed.

How can we optimize θ?

Sample noise variables $(\mathbf{z}^1, \ldots, \mathbf{z}^L) \sim q(\mathbf{z}|\mathbf{x})$.

Perform backpropagation and SGD step for θ.

- If q is simple, sampling is efficient (no MCMC necessary).
- Unbiased estimate of gradient, small variance overhead.
- May even be beneficial (training w/ noise).

Stochastic Backpropagation

Optimizing over q involves gradients of expectations!

How does a change of the q-distributions change q-averages?

Stochastic backpropagation

$$\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad f\text{: smooth and integrable, then}$$

$$\nabla_{\boldsymbol{\mu}} \mathbf{E}[f(\mathbf{z})] = \mathbf{E}\left[\nabla_{\mathbf{z}} f(\mathbf{z})\right], \quad \nabla_{\boldsymbol{\Sigma}} \mathbf{E}[f(\mathbf{z})] = \frac{1}{2} \mathbf{E}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z})\right],$$

- due to Bonnet, 1964, Price, 1958; Kingma & Welling, 2014
- proof (Bonnet's theorem): integration by parts

$$\nabla_{\boldsymbol{\mu}} \mathbf{E} f(\mathbf{z}) = -\int \nabla_{\mathbf{z}} \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) f(\mathbf{z}) d\mathbf{z} = \int \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \nabla_{\mathbf{z}} f(\mathbf{z}) d\mathbf{z}$$

- simple Monte Carlo integration (!) of RHS integral

Section 4

Generative Adverserial Models

From Optimal Discrimination to Generation

- Proposed by Goodfellow et al., 2014
- Classification problem: distinguish between data & model

$$\tilde{p}_\theta(\mathbf{x}, y = 1) = \tfrac{1}{2} p(\mathbf{x}), \quad \tilde{p}_\theta(\mathbf{x}, y = 0) = \tfrac{1}{2} p_\theta(\mathbf{x})\,.$$

- Bayes optimal classifier: posterior

$$q_\theta = p/(p + p_\theta)\,.$$

- Train generator via minimizing the logistic likelihood

$$\theta \xrightarrow{\min} \ell^*(\theta) := \mathbf{E}_{\tilde{p}_\theta}\left[y \ln q_\theta(\mathbf{x}) + (1 - y) \ln(1 - q_\theta(\mathbf{x}))\right]$$

  - generator's goal: generate samples that are indistinguishable from real data, even for the best possible classifier

Minimzing Jensen-Shannon Divergence (1 of 2)

0. Given: distribution p over $(\mathbf{x}, y) \in \mathbb{R}^m \times \{0, 1\}$ with $p(y) = \frac{1}{2}$.

1. Entropy = minimizer of cross entropy.

$$\min_q \{H(p, q) := -p \ln q - (1-p) \ln(1-q)\} = H(p)$$

2. Hence: maximum of logistic log-likelihood at a fixed $\mathbf{x}$

$$\max_q \{p(y=1|\mathbf{x}) \ln q + p(y=0|\mathbf{x}) \ln(1-q)\} = -H(y; \mathbf{x})$$

3. and in expectation

$$\begin{aligned} \ell^*(\theta) &= -H(y|\mathbf{x}) = H(\mathbf{x}) - H(\mathbf{x}, y) \\ &= H(\mathbf{x}) - H(y) - H(\mathbf{x}|y) \end{aligned}$$

Minimzing Jensen-Shannon Divergence (2 of 2)

4. Note that: $H(y) = \ln 2$ and $H(\mathbf{x}|y) = \frac{1}{2}H(\mathbf{x};0) + \frac{1}{2}H(\mathbf{x};1)$.

$$\ell^*(\theta) = H\left(\frac{1}{2}p + \frac{1}{2}p_\theta\right) - \frac{1}{2}H(p) + \frac{1}{2}H(p_\theta) - \ln 2$$

5. Identify w/ Jensen-Shannon divergence

$$\mathsf{JS}(p, p_\theta) = \frac{1}{2}\mathsf{KL}(p, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta) + \frac{1}{2}\mathsf{KL}(p_\theta, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta)$$

6. Where we use simple identities

$$\mathsf{KL}(p, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta) = H(p, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta) - H(p)$$

$$H(p, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta) + H(p_\theta, \tfrac{1}{2}p + \tfrac{1}{2}p_\theta) = 2H(\tfrac{1}{2}p + \tfrac{1}{2}p_\theta)$$

Hence: $\ell^*(\theta) = \mathsf{JS}(p, p_\theta) - \ln 2$

From Real Discriminaton to Generation

- Optimal classifier is in general inaccessible
- Instead: define a classification model

$$q_\phi : \mathbf{x} \mapsto [0;1], \quad \phi \in \Phi$$

- Define objective via bound

$$\ell^*(\theta) \geq \sup_{\phi \in \Phi} \ell(\theta, \phi)$$

$$\ell(\theta, \phi) := \mathbf{E}_{\tilde{p}_\theta} \left[y \ln q_\phi(\mathbf{x}) + (1-y) \ln(1 - q_\phi(\mathbf{x})) \right]$$

  - find best classifier within restricted family
  - typically: $\Phi =$ weight space of DNN
  - training objective for generator is defined implicitly over $\sup$

Optimizing GANs

- Saddle-point problem

$$\theta^* := \operatorname*{argmin}_{\theta \in \Theta} \{ \sup_{\phi \in \Phi} \ell(\theta, \phi) \}$$

  - explicitly performing inner $\sup$ is impractical
  - various methods from optimization / solving games

- SGD as a heuristic (may diverge!)

$$\theta^{t+1} = \theta^t - \eta \nabla_\theta \ell(\theta^t, \phi^t)$$

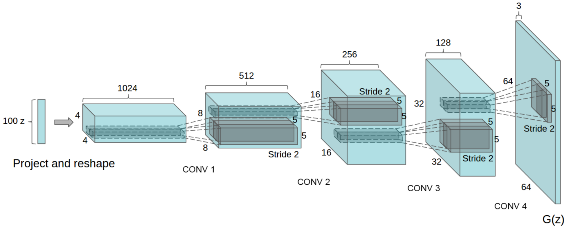
$$\phi^{t+1} = \phi + \eta \nabla_\phi \ell(\theta^{t+1}, \phi^t)$$

- Ongoing research: find better optimization methods (e.g. better gradients Goodfellow et al. 2014 unrolled GANs, Metz et al. 2016; regularization Roth et al 2017)

Example: Image Generation

From Radford et al. 2015 – DC GAN paper

Architecture

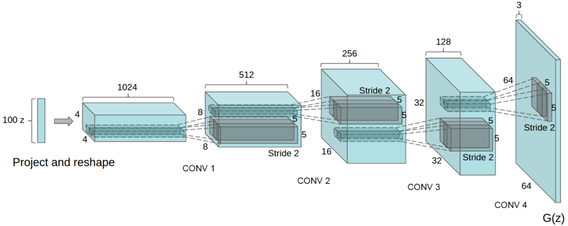


Examples of generated images (bedroom scenes)

Example: Image Generation

From Radford et al. 2015 – DC GAN paper

Architecture



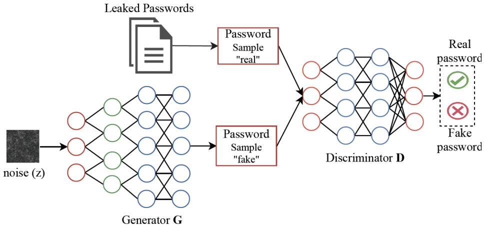
Examples of generated images (bedroom scenes)

Evaluating GANs

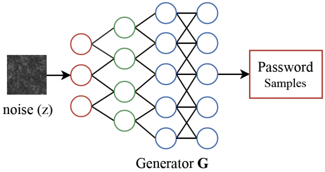
- Convergence to $p(\mathbf{x})$:
  - it is analyzed in Goodfellow et al. 2014.
  - moment matching (Liu, Bousquet and Chaudhuri 2017).
  - AdaGAN (Tolstikhin et al. 2017)
- How to measure quality of implicit models? (fundamental question)
  - out-of-sample evaluations is not available for implicit models.
  - visual inspection (inception score).
- Trade-offs:
  1. noisy samples (e.g. blurry images), but adequate representation of the variability.
  2. faithful (as in good looking) samples, but lack of representation ("mode dropping").

  which one is better?
- Evaluation that is based on each application.

Password learning and generation



Improved Wasserstein Gan



General Results

Approach	(1) Unique Passwords	(2) Matches
JTR Spyderlab	10^9	461,395 (23.32%)
Markov Model 3-gram	$4.9 \cdot 10^8$	532,961 (26.93%)
HashCat gen2	10^9	597,899 (30.22%)
HashCat Best64	$3.6 \cdot 10^8$	630,068 (31.84%)
PCFG	10^{10}	650,695 (32.89%)
FLA 10^{-10}	$7.4 \cdot 10^8$	652,585 (32.99%)
PassGAN	$2.1 \cdot 10^9$	515,079 (26.04%)
PassGAN	$3.6 \cdot 10^9$	584,466 (29.54%)
PassGAN	$4.9 \cdot 10^9$	625,245 (31.60%)
PassGAN	$6.0 \cdot 10^9$	653,978 (33.06%)
PassGAN	$7.1 \cdot 10^9$	676,439 (34.19%)

Unique Passwords	(1) PassGAN	(2) FLA	(3) PassGAN ∪ FLA	(4) PassGAN, and not from FLA	(5) FLA, and not from PassGAN
10^4	14	2	16	14	2
10^5	95	40	133	93	38
10^6	881	1,183	2,016	833	1,135
10^7	7,633	16,330	22,203	5,873	14,570
10^8	44,490	117,262	137,415	20,153	92,925
10^9	155,369				
$7 \cdot 10^9$	320,365				

High probability passwords

Password	Occurrence in Training Data	Frequency in Training Data	GAN Estimated Frequency
123456	232,844	0.98%	100,971,288 (1.0%)
12345	63,135	0.27%	21,614,548 (0.22%)
123456789	61,531	0.26%	22,208,040 (0.22%)
password	47,507	0.20%	85,889 (8.6e-4%)
iloveyou	40,037	0.17%	10,056,700 (0.10%)
princess	26,669	0.11%	190,796 (0.0019%)
1234567	17,399	0.073%	7,545,708 (0.075%)
rockyou	16,765	0.071%	55,515 (5.5e-4%)
12345678	16,536	0.070%	5,070,673 (0.051%)
abc123	13,243	0.056%	6,545 (6.5e-5%)
nicole	12,992	0.055%	206,277 (0.0021%)
daniel	12,337	0.052%	3,304,567 (0.033%)
babygirl	12,130	0.051%	13,076 (1.3e-4%)
monkey	11,726	0.050%	116,602 (0.0012%)
lovely	11,533	0.049%	1,026,362 (0.010%)
jessica	11,262	0.048%	220,849 (0.0022%)
654321	11,181	0.047%	19,912 (1.9e-4%)
michael	11,174	0.047%	517 (5.2e-6%)
ashley	10,741	0.045%	116,858 (0.0012%)
qwerty	10,730	0.045%	135,124 (0.0013%)
iloveu	10,587	0.045%	4,839,368 (0.048%)
111111	10,529	0.044%	101,903 (0.0010%)
000000	10,412	0.044%	108,300 (0.0011%)
michelle	10,210	0.043%	739,220 (0.0073%)
tigger	9,381	0.040%	658,360 (0.0066%)
sunshine	9,252	0.039%	3,628 (3.6e-5%)
chocolate	9,012	0.038%	12 (1.2e-7%)
password1	8,916	0.038%	6,427 (6.4e-5%)
soccer	8,752	0.037%	25 (2.5e-7%)
anthony	8,752	0.036%	not generated

High probability passwords

Password	Rank in Training Set	Frequency in Training Set	Probability assigned by FLA
123456	1	0.9833%	2.81E-3
12345	2	0.2666%	1.06E-3
123457	3,224	0.0016%	2.87E-4
1234566	5,769	0.0010%	1.85E-4
1234565	9,692	0.0006%	1.11E-4
1234567	7	0.0735%	1.00E-4
12345669	848,078	0.0000%	9.84E-5
123458	7,359	0.0008%	9.54E-5
12345679	7,818	0.0007%	9.07E-5
123459	8,155	0.0007%	7.33E-5
lover	457	0.0079%	6.73E-5
love	384	0.0089%	6.09E-5
223456	69,163	0.0001%	5.14E-5
22345	118,098	0.0001%	4.61E-5
1234564	293,340	0.0000%	3.81E-5

FLA

Password	Rank in Training Set	Frequency in Training Set	Probability in PassGAN's Output
123456	1	0.9833%	1.01E-2
123456789	3	0.25985%	2.2E-3
12345	2	0.26662%	2.16E-3
iloveyou	5	0.16908%	1.01E-3
1234567	7	0.07348%	7.6E-4
angel	33	0.03558%	6.4E-4
12345678	9	0.06983%	5.1E-4
iloveu	21	0.04471%	4.9E-4
angela	109	0.01921%	3.4E-4
daniel	12	0.0521%	3.3E-4
sweety	90	0.02171%	2.6E-4
angels	57	0.02787%	2.5E-4
maria	210	0.01342%	1.6E-4
loveyou	52	0.0287%	1.5E-4
andrew	55	0.02815%	1.3E-4

GAN

Conclusion about PassGANs

- PassGAN is comparable to current Passwords guessing procedures.
- PassGANs provides better density estimate (not good anyway).
- New structures are needed to improve in the mid-range estimates.

- Passwords is a good benchmark for text generation GANs:
  - It is a relevant problem.
  - It is easier than natural language.
  - It has an *on-the-job* performance measure: number of guess passwords.
  - It can help with density estimation evaluation, as the choices are discrete and repeatable.