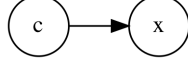


1 InfoGAN

InfoGAN extends the GAN objective to include a new term which encourages high mutual information between generated data and a subset of latent codes [1]. Let (\mathbf{c}, \mathbf{z}) be latent variable, where \mathbf{c} are latent codes capturing semantic features of the data distribution and \mathbf{z} are source of incompressible noise.

1.1 Probabilistic Interpretation



A simpler view of method presented in the paper is to consider the above generative model. The joint density can be factorized as follows

$$p_{\mathbf{c}, \mathbf{x}} = p_{\mathbf{c}}(c)p_{\mathbf{x}|\mathbf{c}}(x|c) = \prod_{l=1}^L p_{c_l}(c_l)p_{\mathbf{x}|\mathbf{c}}(x|c)$$

The paper implicitly model $p_{\mathbf{x}|\mathbf{c}}$ by using a combination of 1) a deterministic generator $G : \mathcal{C} \times \mathcal{Z} \rightarrow \mathcal{X}$ and 2) a stochastic noise sampler $\mathbf{z} \sim p_{\mathbf{z}}$. In particular, $f : \mathcal{C} \rightarrow \mathcal{X}; c \mapsto G(c, z)$ for some $z \sim p_{\mathbf{z}}$ is trained to sample from $p_{\mathbf{x}|\mathbf{c}}(\cdot|c)$ using the adversarial loss [2].

1.2 Variational Maximization of Mutual Information

The paper is motivated to construct latent code in such a way such that when given a sample, we would be quite certain what the latent codes are. In other words, we are interested in the following optimization problem

$$\min_G H(\mathbf{c}|\mathbf{x}) \quad \text{where} \quad \mathbf{x} = G(\mathbf{c}, \mathbf{z}) \quad (1)$$

If we know the parametric family of distribution \mathbf{c} is in, this is equivalent to maximizing mutual information between latent codes and generated sample. Given $H(\mathbf{c}|\mathbf{x}) = H(\mathbf{c}) - I(\mathbf{c}; \mathbf{x})$, we can rewrite (2) as

$$\max_G I(\mathbf{c}; \mathbf{x}) = \mathbb{E}_{\mathbf{c}, \mathbf{x}} \left[\log \frac{p_{\mathbf{c}, \mathbf{x}}(c, x)}{p_{\mathbf{c}}(c)p_{\mathbf{x}}(x)} \right]$$

which is intractable, since we do not know the implicit likelihood $p_{\mathbf{x}|\mathbf{c}}$ nor the posterior $p_{\mathbf{c}|\mathbf{x}}$. Instead we approximate $p_{\mathbf{c}|\mathbf{x}}$ with using $q_{\mathbf{c}|\mathbf{x}}$, parameterize by a neural network, and derive a lower bound for the objective [3, 4],

$$\begin{aligned} I(\mathbf{c}; \mathbf{x}) &= H(\mathbf{c}) - H(\mathbf{c}|\mathbf{x}) \\ &= \sum_x p_{\mathbf{x}}(x) \sum_c p_{\mathbf{c}|\mathbf{x}}(c|x) \log p_{\mathbf{c}|\mathbf{x}}(c|x) + H(\mathbf{c}) \\ &= \sum_x p_{\mathbf{x}}(x) \sum_c p_{\mathbf{c}|\mathbf{x}}(c|x) \log \frac{p_{\mathbf{c}|\mathbf{x}}(c|x)}{q_{\mathbf{c}|\mathbf{x}}(c|x)} + \sum_x p_{\mathbf{x}}(x) \sum_c p_{\mathbf{c}|\mathbf{x}}(c|x) \log q_{\mathbf{c}|\mathbf{x}}(c|x) + H(\mathbf{c}) \\ &= \mathbb{E}_{\mathbf{x}} [KL(p_{\mathbf{c}|\mathbf{x}}(c|x) || q_{\mathbf{c}|\mathbf{x}}(c|x))] + \mathbb{E}_{\mathbf{c}, \mathbf{x}} [\log q_{\mathbf{c}|\mathbf{x}}(c|x)] + H(\mathbf{c}) \\ &\geq \mathbb{E}_{\mathbf{c}, \mathbf{x}} [\log q_{\mathbf{c}|\mathbf{x}}(c|x)] + H(\mathbf{c}) \quad (KL \geq 0) \\ &= \mathbb{E}_{\mathbf{c}, \mathbf{z}} [\log q_{\mathbf{c}|\mathbf{x}}(c|G(c, z))] + H(\mathbf{c}) \quad (\text{LOTUS}) \end{aligned}$$

1.3 Gradient Estimator

This lower bound can be optimized using stochastic gradient via Monte Carlo estimation,

$$\begin{aligned}\nabla_{\theta} \{ \mathbb{E}_{\mathbf{c},z} [\log q_{\mathbf{c}|\mathbf{x}}(c|G(c,z))] + H(\mathbf{c}) \} &= \mathbb{E}_{\mathbf{c},z} [\nabla_{\theta} \log q_{\mathbf{c}|\mathbf{x}}(c|G(c,z))] \\ &\approx \sum_{i=1}^N \nabla_{\theta} \log q_{\mathbf{c}|\mathbf{x}}(c^{(i)}|G(c^{(i)}, z^{(i)})) \\ &\quad \text{where } c^{(i)} \sim p_{\mathbf{c}} \quad z^{(i)} \sim p_z \quad i = 1, \dots, N\end{aligned}$$

We could also interpret the idea of randomizing the generator using a noise sampler as performing the reparameterization trick [5]. We avoid taking gradient of expectation with respect to $p_{\mathbf{x}|\mathbf{c}}$; Instead, we take sample from a known distribution $z \sim p_z$ and then compute the desired sample $x = G(c, z)$ via a deterministic function.

1.4 Optimization

Note $q_{\mathbf{c}|\mathbf{x}}$ is parameterized by a neural network Q , which shares all convolutional layer with D except for the last fully connected layer. We assume $q_{\mathbf{c}|\mathbf{x}}$ to be factored, i.e. $q_{\mathbf{c}|\mathbf{x}} = \prod_i q_{c_i|\mathbf{x}}$. The paper only considers the case where each c_i is parameteric. For each i , $Q(c_i|x)$ outputs the parameters for c_i , e.g. apply softmax nonlinearity to output probabilities for c_i that follows a categorical distribution and outputs mean and variance for c_i that follows a simple Gaussian distribution. Following convention in section (2), we parameterize G, D, Q with $\theta_G, \theta_D, \theta_Q$ respectively. Then

$$\begin{aligned}\mathcal{L}_{GAN}(\theta_D, \theta_G) &= \mathbb{E}_{\mathbf{x}} [-\log D(x; \theta_D)] + \mathbb{E}_{\mathbf{c},z} [-\log (1 - D(G(c, z; \theta_G); \theta_D))] \\ \mathcal{L}_I(\theta_D, \theta_Q) &= \mathbb{E}_{\mathbf{c},z} [\log (Q(G(c, z; \theta_G); \theta_Q))]\end{aligned}$$

We are interested in the following optimization problem

$$\begin{aligned}\min_{\theta_G, \theta_Q} \max_{\theta_D} \mathcal{L}_{GAN}(\theta_D, \theta_G) + \mathcal{L}_I(\theta_D, \theta_Q) \\ \min_{\theta_G, \theta_Q} \max_{\theta_D} \mathbb{E}_{\mathbf{x}} [\log D(x; \theta_D)] + \mathbb{E}_{\mathbf{x}'} [\log (1 - D(x'; \theta_D)) - \lambda \log Q(x'; \theta_Q)] \\ \text{where } x' = G(c, z; \theta_G)\end{aligned}$$

Similar to equation (3), we can write

$$\begin{aligned}\min_{\theta_D} \mathbb{E}_{\mathbf{x}} [-\log D(x; \theta_D)] + \mathbb{E}_{\mathbf{x}'} [-\log (1 - D(x; \theta_D))] \\ \min_{\theta_G} \mathbb{E}_{\mathbf{c},z} [\log (1 - D(G(c, z; \theta_G)) - \lambda \log Q(G(c, z; \theta_G))] \\ \min_{\theta_Q} \lambda \mathbb{E}_{\mathbf{x}'} [-\log Q(x'; \theta_Q)]\end{aligned}$$

2 Clarification on GAN's loss

Formulation of GAN loss bears assemblance to the idea of *learning by comparison* in the noise contrastive estimation (NCE) paper [6]. It turns out the connection between hypothesis testing and learning implicit generative models is quite extensively studied [7]. Here is a reproduction of a subset of ideas in these two papers, in addition to a brief comparison between NCE and GAN.

2.1 Learning by Comparison

The goal of both GAN and NCE is to approximate the true data distribution $p_d(\cdot)$ with a parameterized model $p_m(\cdot)$, where learning is driven by classification of which data distribution the sample come from. We formulate this idea below. Let $\mathcal{X}_d = \{x_1, \dots, x_N\}$ be the training dataset and $\mathcal{X}_g = \{x'_1, \dots, x'_N\}$ be the generated dataset. Let \mathbf{u} be a random variable that takes value on $\mathcal{U} = \mathcal{X}_d \cup \mathcal{X}_g$. We can assign all data points in \mathcal{U} binary class labels $\mathcal{Y} = \{y_i \mid y_i = \mathbb{1}_{u_i \in \mathcal{X}_d}\}$, i.e. assign value of 1 to real data point and 0 to generated data point. We can think of each label following a Bernoulli distribution $y_i \sim \text{Bern}(p)$. We want to build a *max a posteriori* classifier to classify \mathbf{y} given \mathbf{u} ,

$$\hat{y}(\mathbf{u}) = \arg \max_{y \in \{0,1\}} p_{\mathbf{y}|\mathbf{u}}(y|\mathbf{u})$$

Equivalently, we can arrive at an equivalent decision rule based on (log) density ratio and notice its similarity to logistic regression. Let $\mathbf{y} \sim \text{Bern}(1/2)$ and use Bayes rule,

$$\begin{aligned} p_{\mathbf{y}|\mathbf{u}}(1|\mathbf{u}) &= \frac{p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|1)p_{\mathbf{y}}(1)}{p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|1)p_{\mathbf{y}}(1) + p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|0)p_{\mathbf{y}}(0)} = \sigma \left(\log \frac{p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|1)}{p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|0)} \right) \\ p_{\mathbf{y}|\mathbf{u}}(0|\mathbf{u}) &= 1 - p_{\mathbf{y}|\mathbf{u}}(1|\mathbf{u}) \end{aligned}$$

Let ϕ be parameters for our classifier $\hat{y}(\cdot)$. We want our data $\{(u_i, y_i)\}_{i=1}^{2N}$ to be likely under the result of classification. This is equivalent to maximizing log likelihood of parameters ϕ

$$\begin{aligned} \ell(\phi) &= \frac{1}{2N} \log \prod_{i=1}^{2N} p_{\mathbf{y}|\mathbf{u}}(y_i|\mathbf{u}_i; \phi) \\ &= \frac{1}{2N} \left(\sum_{i=1}^{2N} y \log p_{\mathbf{y}|\mathbf{u}}(1|\mathbf{u}_i; \phi) + (1 - y) \log p_{\mathbf{y}|\mathbf{u}}(0|\mathbf{u}_i; \phi) \right) \\ &= \frac{1}{2N} \sum_{i=1}^N (\log p_{\mathbf{y}|\mathbf{u}}(1|x_i; \phi) + \log (1 - p_{\mathbf{y}|\mathbf{u}}(1|x'_i; \phi))) \\ &= \frac{1}{2N} (\mathbb{E}_{p_d} [\log p_{\mathbf{y}|\mathbf{u}}(1|x; \phi)] + \mathbb{E}_{p_g} [\log (1 - p_{\mathbf{y}|\mathbf{u}}(1|x; \phi))]) \end{aligned} \quad (2)$$

2.2 NCE

In NCE, we model the class conditional likelihood with parametric distributions,

$$\begin{aligned} p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|1) &= p_m(\mathbf{u}; \phi) && \text{(model distribution)} \\ p_{\mathbf{u}|\mathbf{y}}(\mathbf{u}|0) &= p_n(\mathbf{u}) && \text{(fixed noise distribution)} \end{aligned}$$

In essence, we estimate parameters for the data distribution by learning the parameters for the classifier, ϕ by maximizing the objective function (2).

2.3 GAN

In GAN, we model the posterior directly with a discriminator network $D : \mathcal{U} \rightarrow [0, 1]$

$$\begin{aligned} p_{y|u}(1|u) &= D(u; \phi) \\ p_{y|u}(0|u) &= 1 - D(u; \phi) \end{aligned}$$

We can interpret the score that the discriminator network computes as an approximation for the log likelihood ratio. In addition to classification, GAN uses a generator network $D : \mathcal{Z} \rightarrow \mathcal{X}$, which takes a sample from a latent distribution $z \sim p_z$ to generate samples for an implicit model data distribution $p_m(\cdot)$. GAN’s loss can be derived from (2)

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{p_d} [-\log D(x; \phi)] + \mathbb{E}_{p_g} [-\log (1 - D(x; \phi))] \\ &= \mathbb{E}_{p_d} [-\log D(x; \phi)] + \mathbb{E}_{p_z} [-\log (1 - D(G(z; \theta); \phi))] \end{aligned} \quad (\text{LOTUS})$$

We form a minimax game where the discriminator tries to identify counterfakes and the generator tries to generate realistic samples.

$$\min_G \max_D \mathbb{E}_{p_d} [-\log D(x; \phi)] + \mathbb{E}_{p_z} [-\log (1 - D(G(z; \theta); \phi))]$$

Note \mathcal{L} is separable with respect to ϕ, θ , so we can do alternating optimization,

$$\begin{aligned} \min_{\phi} \mathbb{E}_{p_d} [-\log D(x; \phi)] + \mathbb{E}_{p_g} [-\log (1 - D(x; \phi))] \\ \min_{\theta} \mathbb{E}_{p_z} [\log (1 - D(G(z; \theta))) \end{aligned} \quad (3)$$

References

- [1] Xi Chen et al. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”. In: *arXiv:1606.03657 [cs, stat]* (June 11, 2016). arXiv: [1606.03657](https://arxiv.org/abs/1606.03657). URL: <http://arxiv.org/abs/1606.03657> (visited on 12/04/2019).
- [2] Ian J. Goodfellow et al. “Generative Adversarial Networks”. In: *arXiv:1406.2661 [cs, stat]* (June 10, 2014). arXiv: [1406.2661](https://arxiv.org/abs/1406.2661). URL: <http://arxiv.org/abs/1406.2661> (visited on 12/12/2019).
- [3] David Barber and Felix Agakov. “The IM Algorithm: A Variational Approach to Information Maximization.” In: Jan. 1, 2003.
- [4] Ben Poole and Sherjil Ozair. “On Variational Bounds of Mutual Information”. In: (2019), p. 10.
- [5] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *arXiv:1312.6114 [cs, stat]* (May 1, 2014). arXiv: [1312.6114](https://arxiv.org/abs/1312.6114). URL: <http://arxiv.org/abs/1312.6114> (visited on 11/13/2019).
- [6] Michael Gutmann and Aapo Hyvärinen. “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Mar. 31, 2010, pp. 297–304. URL: <http://proceedings.mlr.press/v9/gutmann10a.html> (visited on 12/10/2019).
- [7] Shakir Mohamed and Balaji Lakshminarayanan. “Learning in Implicit Generative Models”. In: *arXiv:1610.03483 [cs, stat]* (Feb. 27, 2017). arXiv: [1610.03483](https://arxiv.org/abs/1610.03483). URL: <http://arxiv.org/abs/1610.03483> (visited on 01/27/2020).