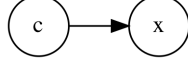# 1   InfoGAN

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 (c, z) be latent variable, where c are latent codes capturing semantic features of the data distribution and 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

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

The paper implicitly model $p_{\mathsf{x}|\mathsf{c}}$ by using a combination of 1) a deterministic generator $G : \mathcal{C} \times \mathcal{Z} \to \mathcal{X}$ and 2) a stochastic noise sampler $\mathsf{z} \sim p_\mathsf{z}$. In particular, $f : \mathcal{C} \to \mathcal{X}; c \mapsto G(c, z)$ for some $z \sim p_\mathsf{z}$ is trained to sample from $p_{\mathsf{x}|\mathsf{c}}(\cdot|c)$ 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 generated sample, we would be quite certain what the latent codes are. In other words, we want the generator to be making use of the latent code as much as possible. We are interested in the following optimization problem,

$$\min_G H(\mathsf{c}|\mathsf{x}) \qquad \text{where} \qquad \mathsf{x} = G(\mathsf{c}, \mathsf{z}) \tag{1}$$

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

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

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

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

1.3 Gradient Estimator

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

$$\nabla_\theta \left\{ \mathbb{E}_{\mathsf{c},\mathsf{z}} \left[\log q_{\mathsf{c}|\mathsf{x}}(c|G(c,z)) \right] + H(\mathsf{c}) \right\} = \mathbb{E}_{\mathsf{c},\mathsf{z}} \left[\nabla_\theta \log q_{\mathsf{c}|\mathsf{x}}(c|G(c,z)) \right]$$

$$\approx \sum_{i=1}^{N} \nabla_\theta \log q_{\mathsf{c}|\mathsf{x}}(c^{(i)}|G(c^{(i)}, z^{(i)}))$$

$$\text{where} \quad c^{(i)} \sim p_{\mathsf{c}} \quad z^{(i)} \sim p_{\mathsf{z}} \qquad i = 1, \cdots, N$$

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 $p_{\mathsf{x}|\mathsf{c}}$; Instead, we take sample from a known distribution $z \sim p_{\mathsf{z}}$ and then compute the desired sample $x = G(c, z)$ via a deterministic function.

1.4 Optimization

Note, Bernoulli distributed $p_{\mathsf{y}|\mathsf{x}}$ is approximated with the discriminator network D, parameterized by θ_D. Similarly, $q_{\mathsf{c}|\mathsf{x}}$ is approximated by a neural netowrk Q, parameterized by θ_Q. We assume $q_{\mathsf{c}|\mathsf{x}}$ to be factored, i.e. $q_{\mathsf{c}|\mathsf{x}} = \prod_i q_{\mathsf{c}_i|\mathsf{x}}$. For each i, $Q(c_i|x)$ outputs the parameters for distributions of c_i, e.g. class probabilities for categorical c_i and mean and variance for Gaussian c_i.

$$p_{\mathsf{y}|\mathsf{x}}(y|x;\theta_D) = p(x;\theta_D)^{\mathbb{1}_{y=1}}(1 - p(x;\theta_D))^{\mathbb{1}_{y=0}} \qquad\qquad p(x;\theta_D) \leftarrow D(x;\theta_D)$$

$$q_{\mathsf{c}_i|\mathsf{x}}(c_i|x;\theta_Q) = \prod_{i=1}^{K} p_k(x;\theta_Q)^{\mathbb{1}_{c_i=k}} \qquad\qquad \{p_k(x;\theta_Q)\}_{k=1}^{K} \leftarrow Q(x;\theta_Q)$$

$$q_{\mathsf{c}_i|\mathsf{x}}(c_i|x;\theta_Q) = \mathcal{N}(c_i; \mu(x;\theta_Q), \sigma^2(x;\theta_Q)) \qquad\qquad (\mu(x;\theta_Q), \sigma^2(x;\theta_Q)) \leftarrow Q(x;\theta_Q)$$

Let θ_G be parameters for the generator. Following convention in section (2), we can write

$$\mathcal{L}_{GAN}(\theta_D, \theta_G) = \mathbb{E}_{\mathsf{x}} \left[-\log p_{\mathsf{y}|\mathsf{x}}(1|x;\theta_D) \right] + \mathbb{E}_{\mathsf{c},\mathsf{z}} \left[-\log \left(1 - p_{\mathsf{y}|\mathsf{x}}(1|G(c,z;\theta_G);\theta_D)\right) \right]$$

$$\mathcal{L}_I(\theta_D, \theta_Q) = \mathbb{E}_{\mathsf{c},\mathsf{z}} \left[\log q_{\mathsf{c}|\mathsf{x}}(c|G(c,z;\theta_G);\theta_Q) \right]$$

We are interested in the following optimization problem

$$\min_{\theta_G, \theta_Q} \max_{\theta_D} \mathcal{L}_{GAN}(\theta_D, \theta_G) + \mathcal{L}_I(\theta_D, \theta_Q)$$

$$\min_{\theta_G, \theta_Q} \max_{\theta_D} \mathbb{E}_{\mathsf{x}} \left[\log p_{\mathsf{y}|\mathsf{x}}(1;x;\theta_D) \right] + \mathbb{E}_{\mathsf{x}'} \left[\log(1 - p_{\mathsf{y}|\mathsf{x}}(1|x';\theta_D)) - \lambda \log q_{\mathsf{c}|\mathsf{x}}(c|x';\theta_Q) \right]$$

$$\text{where} \qquad x' = G(c, z; \theta_G)$$

Similar to equation (3), we can optimizing in an alternating fashion

$$\min_{\theta_D} \mathbb{E}_{\mathsf{x}} \left[-\log p_{\mathsf{y}|\mathsf{x}}(1|x;\theta_D) \right] + \mathbb{E}_{\mathsf{x}'} \left[-\log \left(1 - p_{\mathsf{y}|\mathsf{x}}(1|x';\theta_D)\right) \right]$$

$$\min_{\theta_G} \mathbb{E}_{\mathsf{c},\mathsf{z}} \left[\log \left(1 - p_{\mathsf{y}|\mathsf{x}}(1|G(c,z;\theta_G)) - \lambda \log q_{\mathsf{c}|\mathsf{x}}(c|G(c,z;\theta_G))\right) \right]$$

$$\min_{\theta_Q} \lambda \mathbb{E}_{\mathsf{x}'} \left[-\log q_{\mathsf{c}|\mathsf{x}}(c|x';\theta_Q) \right]$$

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 $\mathcal{X}_d = \{x_1, \cdots, x_N\}$ be the training dataset and $\mathcal{X}_g = \{x_1', \cdots, x_N'\}$ be the generated dataset. Let 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 $\mathsf{y}_i \sim \mathrm{Bern}(p)$. We want to build a *max a posterior* classifier to classify y given u,

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

Equivalently, we can arrive at an equivalent decision rule based on (log) density ratio. Let $\mathsf{y} \sim \mathrm{Bern}(1/2)$ and use Bayes rule,

$$p_{\mathsf{y}|\mathsf{u}}(1|u) = \frac{p_{\mathsf{u}|\mathsf{y}}(u|1)p_{\mathsf{y}}(1)}{p_{\mathsf{u}|\mathsf{y}}(u|1)p_{\mathsf{y}}(1) + p_{\mathsf{u}|\mathsf{y}}(u|0)p_{\mathsf{y}}(0)} = \sigma\left(\log \frac{p_{\mathsf{u}|\mathsf{y}}(u|1)}{p_{\mathsf{u}|\mathsf{y}}(u|0)}\right)$$

$$p_{\mathsf{y}|\mathsf{u}}(0|u) = 1 - p_{\mathsf{y}|\mathsf{u}}(1|u)$$

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

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

2.2 NCE

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

$$p_{\mathsf{u}|\mathsf{y}}(u|1) = p_m(u; \phi) \qquad\qquad \text{(model distribution)}$$
$$p_{\mathsf{u}|\mathsf{y}}(u|0) = p_n(u) \qquad\qquad \text{(fixed noise distribution)}$$

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 $D : \mathcal{U} \to [0,1]$

$$p_{\mathsf{y}|\mathsf{u}}(1|u) = D(u;\phi)$$
$$p_{\mathsf{y}|\mathsf{u}}(1|u) = 1 - D(u;\phi)$$

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 $D : \mathcal{Z} \to \mathcal{X}$, which takes a sample from a latent distribution $z \sim p_{\mathsf{z}}$ to generate samples for an implicit model data distribution $p_m(\cdot)$. GAN's loss can be derived from (2)

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

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

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

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

$$\min_\phi \mathbb{E}_{p_d}\left[-\log D(x;\phi)\right] + \mathbb{E}_{p_g}\left[-\log\left(1 - D(x;\phi)\right)\right]$$
$$\min_\theta \mathbb{E}_{p_z}\left[\log\left(1 - D(G(z;\theta))\right)\right] \qquad (3)$$

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. 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. 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. 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. URL: http://arxiv.org/abs/1610.03483 (visited on 01/27/2020).