Generative models

MIPT

2022

Generative and discriminative models

Discriminative models

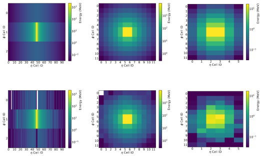
Model: $p(y|\mathrm{x})$.

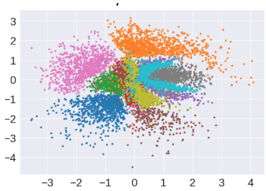
Generative models

Model: $p(y, \mathrm{x})$.

Generative models:

- Generate datasets (when generation is a goal)
- Synthetic dataset generation (for train or fine-tuning)
- Latent dataset properties obtaining

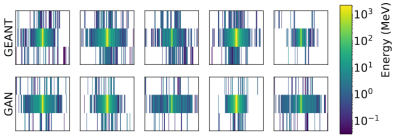


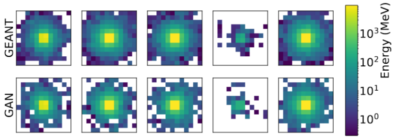


Data generation: example

Paganini et al., 2017:

- Model particle energy
- The modeling uses GAN
- Discrimination is done using GEANT software
- Result: good performance, generation is done 100-1000 times faster

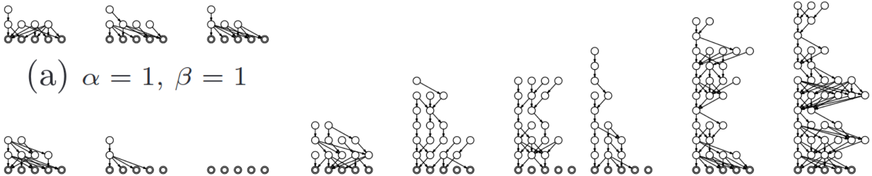




Data generation: example

Adams et al., 2010:

- The problem is to generate deep belief networks
- The model structure **Γ** is a sequence of adjacency matrices for each layer
- The generation is done using MCMC with Indian buffet process (α, β) as a prior
- α, β can be interpreted as a width and sparsity of the structure



(a) $\alpha = 1,\ \beta = 1$

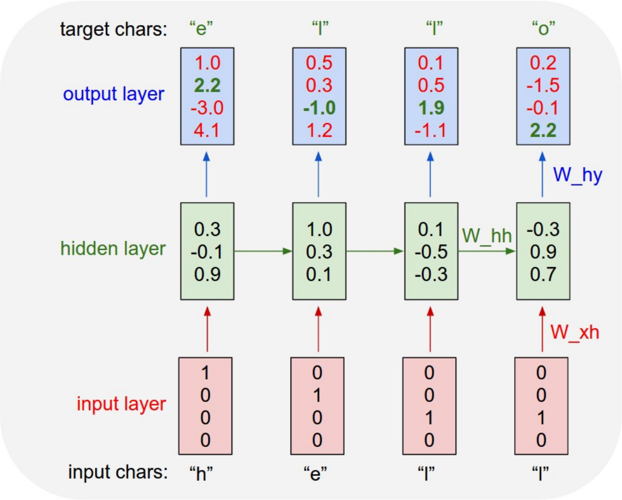
(b) $\alpha = \frac{1}{2},\ \beta = 1$ (c) $\alpha = 1,\ \beta = 2$ (d) $\alpha = \frac{3}{2},\ \beta = 1$

How to build generative models?

- **Approach 1:** assign a likelihood function ("Fully-observed likelihood"), which decomposes object likelihood into parts ("Autoregressive models").

Example: CharRNN

Karpathy, 2015



How to build generative models?

- **Approach 1:** assign a likelihood function ("Fully-observed likelihood"), which decomposes object likelihood into parts ("Autoregressive models").
  Problems:
  - hard to assign a proper likelihood function.
  - computationally intensive inference.

How to build generative models?

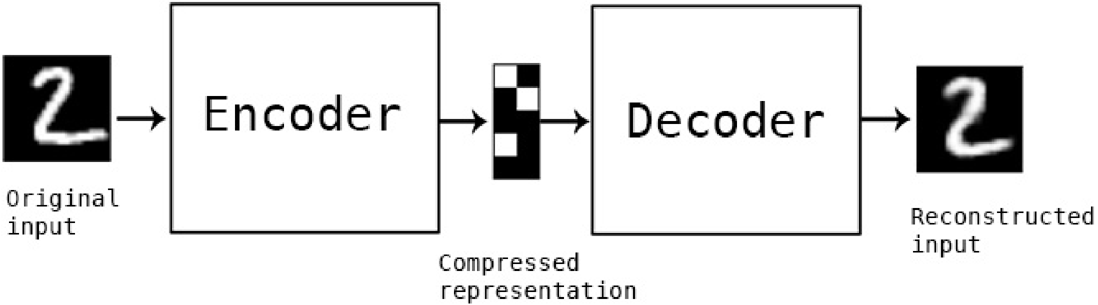
- **Approach 1:** assign a likelihood function (“Fully-observed likelihood”), which decomposes object likelihood into parts (“Autoregressive models”).
  Problems:
  - hard to assign a proper likelihood function.
  - computationally intensive inference.
- **Approach 2:** make an assumption that objects are generated by a latent variable, which is easier to analyze (“Latent variable models”).

Example: autoencoder

Autoencoder is a model of dimension reduction:

$$\mathsf{H} = \boldsymbol{\sigma}(\mathsf{W}_e \mathsf{X}),$$

$$||\boldsymbol{\sigma}(\mathsf{W}_d \mathsf{H}) - \mathsf{X}||_2^2 \to \min .$$



Autoencoder: generative model?

(Alain, Bengio 2012): consider regularized autoencoder:

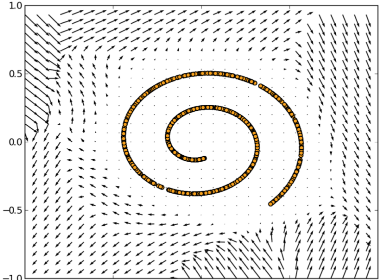
$$||\mathrm{f}(\mathrm{x}, \sigma) - \mathrm{x}||^2,$$

where σ is a noise level.

Then

$$\frac{\partial \log p(x)}{\partial x} = \frac{||\mathrm{f}(\mathrm{x}, \sigma) - \mathrm{x}||^2}{\sigma^2} + o(1) \text{ при } \sigma \to 0.$$

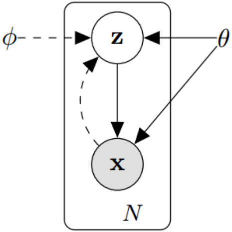
Vector field induced by reconstruction error



Variational autoencoder

Let the objects X be generated by latent variable $\mathrm{h} \sim \mathcal{N}(0, \mathrm{I})$:

$$\mathrm{x} \sim p(\mathrm{x}|\mathrm{h}, \mathrm{w}).$$



$p(\mathrm{h}|\mathrm{x}, \mathrm{w})$ is unknown.
Maximize ELBO:

$$\log p(\mathrm{x}|\mathrm{w}) \geq \mathrm{E}_{q_\phi(\mathrm{h}|\mathrm{x})} \log p(\mathrm{x}|\mathrm{h}, \mathrm{w}) - D_{\mathrm{KL}}(q_\phi(\mathrm{h}|\mathrm{x}) || p(\mathrm{h})) \to \max.$$

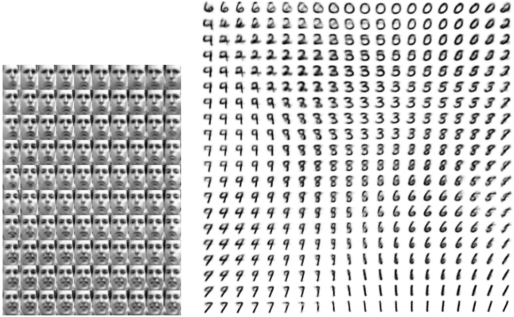
Distributions $q_\phi(\mathrm{h}|\mathrm{x})$ и $p(\mathrm{x}|\mathrm{h}, \mathrm{w})$ are modeled by neural networks:

$$q_\phi(\mathrm{h}|\mathrm{x}) \sim \mathcal{N}(\boldsymbol{\mu}_\phi(\mathrm{x}), \boldsymbol{\sigma}^2_\phi(\mathrm{x})),$$

$$p(\mathrm{x}|\mathrm{h}, \mathrm{w}) \sim \mathcal{N}(\boldsymbol{\mu}_w(\mathrm{h}), \boldsymbol{\sigma}^2_w(\mathrm{h})),$$

where $\boldsymbol{\mu}, \boldsymbol{\sigma}$ are neural network's outputs.

Variational autoencoder: generation process



(a) Learned Frey Face manifold (b) Learned MNIST manifold

How to build generative models?

- **Approach 1:** assign a likelihood function ("Fully-observed likelihood"), which decomposes object likelihood into parts ("Autoregressive models").
  Problems:
  - hard to assign a proper likelihood function.
  - computationally intensive inference.
- **Approach 2:** make an assumption that objects are generated by a latent variable, which is easier to analyze ("Latent variable models").
  Problems:
  - $p(\mathrm{x})$ is intractible
- Problem of both methods: high likelihhod and high sampling quality can be not independent (Theis et al., 2015).
- Given a noisy mixutre:

$$p_w(x) = 0.01 p_{\text{data}}(x) + 0.99 p_{\text{noise}}(x), \log p_w(x) \geq \log p_{\text{data}}(x) - \log 100$$

- For another direction: overfitting

How to build generative models?

- **Approach 1:** assign a likelihood function ("Fully-observed likelihood"), which decomposes object likelihood into parts ("Autoregressive models").
  Problems:
  - hard to assign a proper likelihood function.
  - computationally intensive inference.
- **Approach 2:** make an assumption that objects are generated by a latent variable, which is easier to analyze ("Latent variable models").
- **Approach 3:** do not use likelihood and work straightforwardly with generative process (from likelihood modeling to statistical testing).

Generative-adversarial models (Goodfellow et al., 2014)

Main idea: train two models, generator G and discriminator D:

$$\min_{\mathrm{W}_G} \max_{\mathrm{w}_D} \mathsf{E}_{\mathrm{x} \in \mathfrak{D}} \log p(\mathrm{x} | \mathrm{w}_D, D) + \mathsf{E}_{\mathrm{x} \in p_G} \log(1 - p(\mathrm{x} | \mathrm{w}_D, D)).$$

The algorithm is iterative

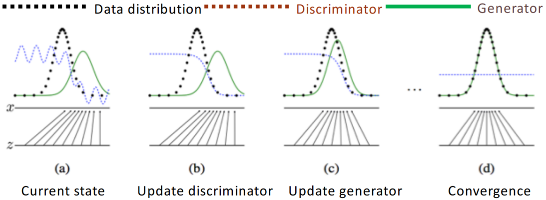
- $\mathsf{E}_{\mathrm{x} \in \mathfrak{D}} \log p(\mathrm{x} | \mathrm{w}_D, D) \to \max_{\mathrm{w}_D}$
- $\mathsf{E}_{\mathrm{x} \in p_G} \log(1 - p(\mathrm{x} | \mathrm{w}_D, D)) \to \min_{\mathrm{w}_G}$
- Alternative: $\mathsf{E}_{\mathrm{x} \in p_G} \log p(\mathrm{x} | \mathrm{w}_D, D) \to \max_{\mathrm{w}_G}$

GAN: optimality

When a discriminator is in global optimum, the generator minimizes JS:

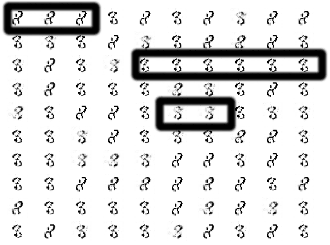
$$-\log(4) + KL\left(p(\mathrm{x}|\frac{p(\mathrm{x}) + p_G(\mathrm{x})}{2})\right) + KL\left(p_G\mathrm{x}|\frac{p(\mathrm{x}) + p_G(\mathrm{x})}{2})\right) \to \min_{\mathrm{w}_G}.$$

Consequent: the optimal generator distribution: $p_G = p(\mathrm{x})$.



Optimization details for GAN

- Generator optimization can be made in two regimes: $\mathsf{E}_{\mathsf{x}\in p_G} \log(1 - p(\mathsf{x}|\mathsf{w}_D, D)) \to \min_{\mathsf{w}_G}$ or $\mathsf{E}_{\mathsf{x}\in p_G} \log p(\mathsf{x}|\mathsf{w}_D, D) \to \max_{\mathsf{w}_G}$: the optima coincide, but for the first regime the gradient is more smooth.
- Generator can converge to a local optimum and generate only similar objects (mode collapse).



https://machinelearningmastery.com/practical-guide-to-gan-failure-modes/

Dataset shift

Dataset shift is an event when distribuition $p(X, y)$ significantly differ for the training and test/inference phases.

- Covariate shift — difference in $p(X)$
- Prior probability shift — difference in $p(y)$
- Concept shift — difference in $p(y|X)$

Dataset shift

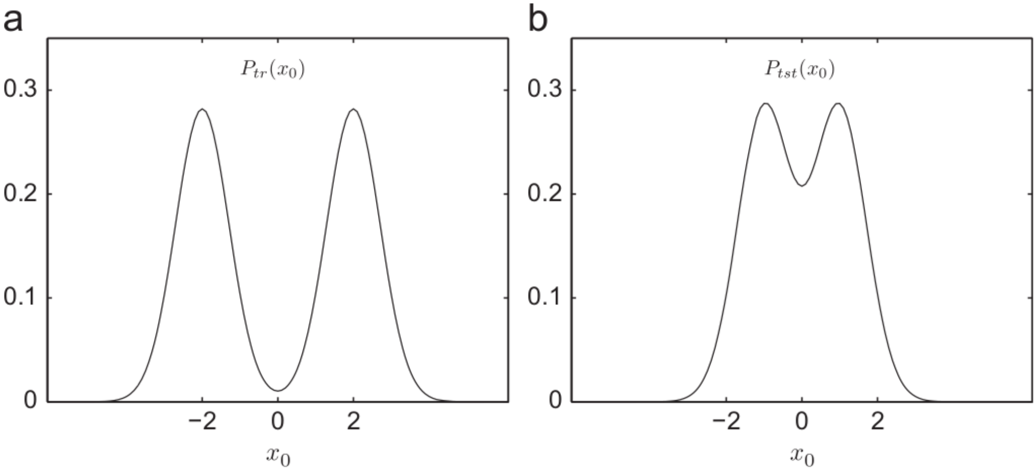
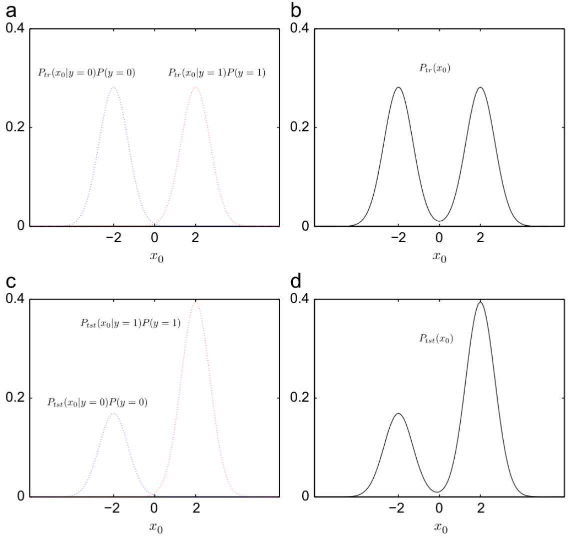


Fig. 1. Covariate shift: $P_{tst}(y|x_0) = P_{tr}(y|x_0)$ and $P_{tr}(x_0) \neq P_{tst}(x_0)$. (a) Training data and (b) test data.

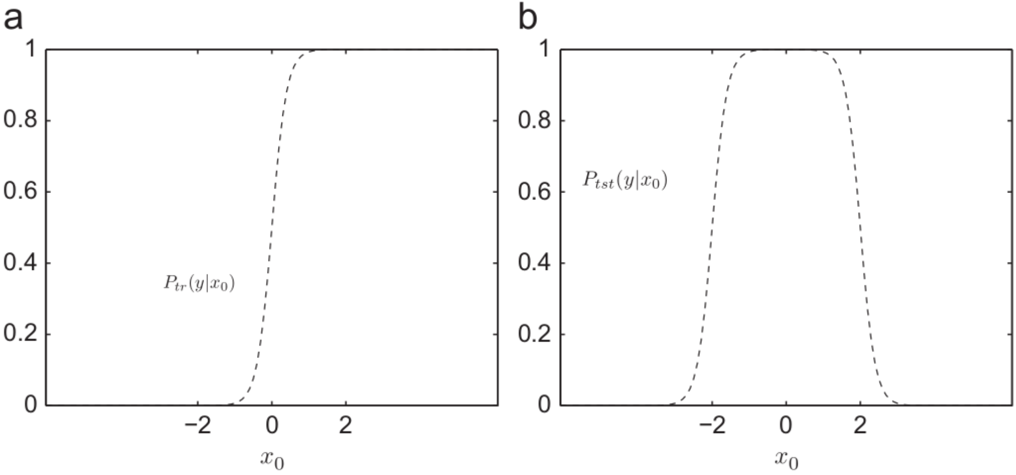
Moreno-Torres et al., 2012

Dataset shift



Moreno-Torres et al., 2012

Dataset shift



Moreno-Torres et al., 2012

Evidence vs Cross-validation

Evidece:

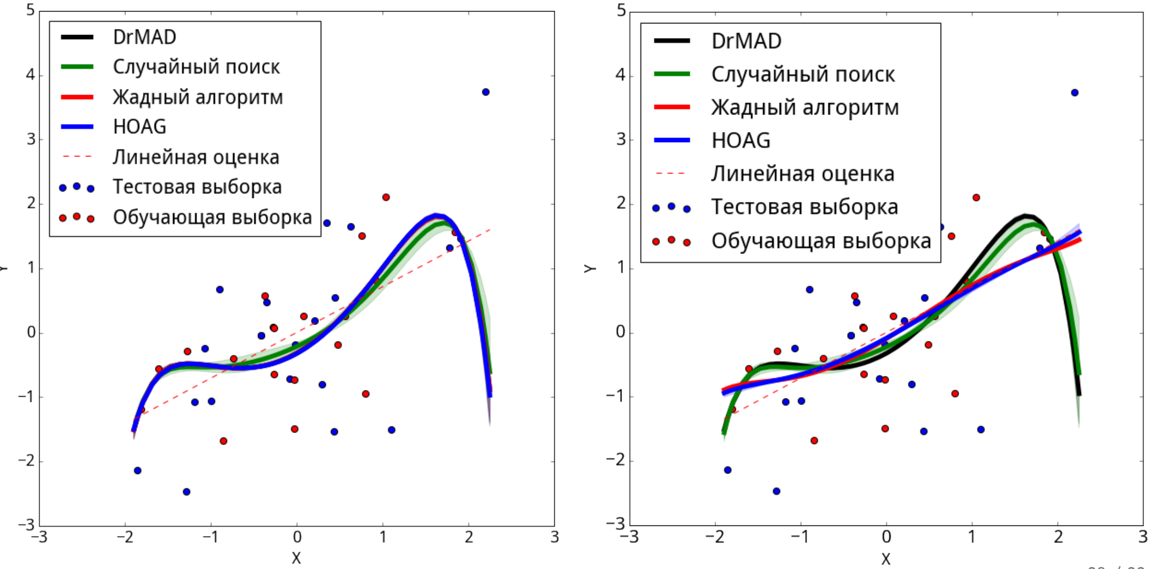
$$\log\ p(\mathrm{X}|\mathrm{f}) = \log\ p(\mathrm{x}_1|\mathrm{f}) + \log\ p(\mathrm{x}_2|\mathrm{x}_1, \mathrm{f}) + \dots + \log\ p(\mathrm{x}_n|\mathrm{x}_1, \dots, \mathrm{x}_{n-1}, \mathrm{f}).$$

Leave-one-out:

$$\mathrm{LOU} = \mathrm{E}\log\ p(\mathrm{x}_n|\mathrm{x}_1, \dots, \mathrm{x}_{n-1}, \mathrm{f}).$$

Cross-validation uses mean value of the last term $p(\mathrm{x}_n|\mathrm{x}_1, \dots, \mathrm{x}_{n-1}, \mathrm{f})$ for complexity estimation.
Evidence considers **full** complexity.

Evidence vs Cross-validation: example



References

- Bishop C. M. Pattern recognition //Machine learning. – 2006. – T. 128. – №. 9.
- Paganini M., de Oliveira L., Nachman B. Accelerating science with generative adversarial networks: an application to 3D particle showers in multilayer calorimeters //Physical review letters. – 2018. – T. 120. – №. 4. – C. 042003.
- Antoran J., Miguel A. Disentangling and learning robust representations with natural clustering //2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). – IEEE, 2019. – C. 694-699.
- Adams R. P., Wallach H., Ghahramani Z. Learning the structure of deep sparse graphical models //Proceedings of the thirteenth international conference on artificial intelligence and statistics. – JMLR Workshop and Conference Proceedings, 2010. – C. 1-8.
- Alain G., Bengio Y. What regularized auto-encoders learn from the data-generating distribution //The Journal of Machine Learning Research. – 2014. – T. 15. – №. 1. – C. 3563-3593.
- Theis L., Oord A., Bethge M. A note on the evaluation of generative models //arXiv preprint arXiv:1511.01844. – 2015.
- Kingma D. P., Welling M. Auto-Encoding Variational Bayes //stat. – 2014. – T. 1050. – C. 10.
- Efron B., Tibshirani R. *An Introduction to the Bootstrap*, 1993.
- Moreno-Torres J. G. et al. A unifying view on dataset shift in classification //Pattern recognition. – 2012. – T. 45. – №. 1. – C. 521-530.
- Bakhteev O. Y., Strijov V. V. Comprehensive analysis of gradient-based hyperparameter optimization algorithms //Annals of Operations Research. – 2020. – T. 289. – №. 1. – C. 51-65.

References

- Aditya Grover et al., Deep Generative Models tutorial, 2018: goo.gl/H1prjP
- Fei-Fei Li et al., Generative Models tutorial, 2017, http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf
- Shakir Mohamed et al., UAI 2017 Tutorial, 2017, https://www.youtube.com/watch?v=JrO5fSskISY
- Andrej Karpathy: The Unreasonable Effectiveness of Recurrent Neural Networks, 2015: http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Maxim Panov: Uncertainty, Out-of-distribution detection for NNs: https://www.youtube.com/watch?v=N-p_qSLzoAI
- https://machinelearningmastery.com/practical-guide-to-gan-failure-modes/
- Cat generator: https://github.com/aleju/cat-generator