Variational Inference: Mean Field, Normalizing Flows and beyond.

Maxim Panov
based on the joint work with A. Thin, N. Kotelevskii, A. Durmus and E. Moulines

HSE-Yandex Autumn School 2020

27.11.2020

Skoltech

Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}(\hat{f}(\mathbf{x}) - f(\mathbf{x}))^2.$$

Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

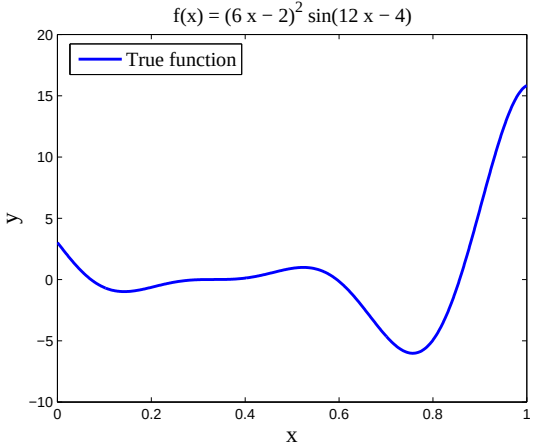
- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}\big(\hat{f}(\mathbf{x}) - f(\mathbf{x})\big)^2.$$



Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

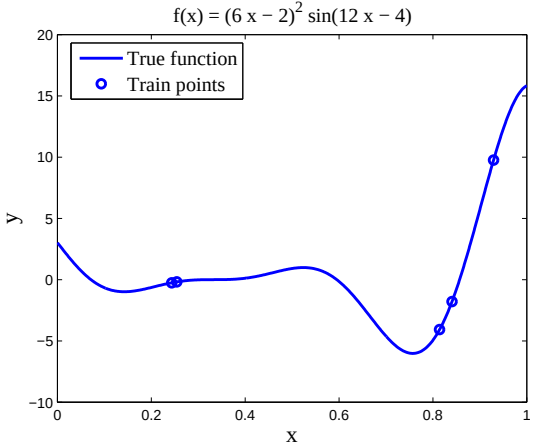
- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}\big(\hat{f}(\mathbf{x}) - f(\mathbf{x})\big)^2.$$



Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

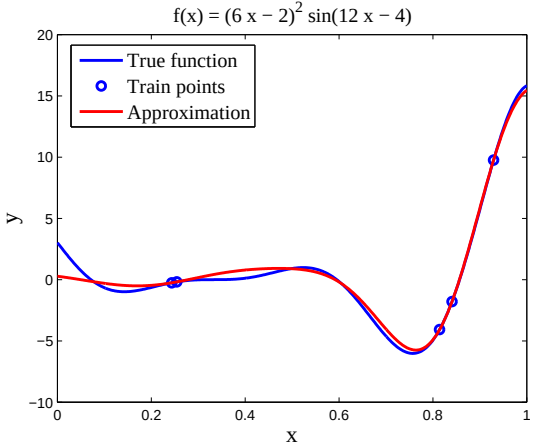
- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}\big(\hat{f}(\mathbf{x}) - f(\mathbf{x})\big)^2.$$



Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

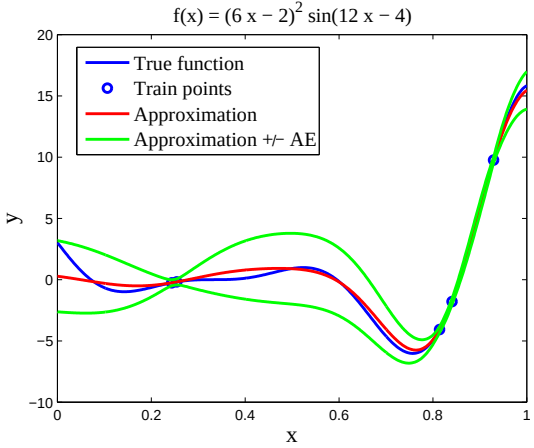
- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}(\hat{f}(\mathbf{x}) - f(\mathbf{x}))^2.$$



Regression and Uncertainty Estimation

Goal: Provide the measure of uncertainty $\hat{\sigma}(\mathbf{x})$ of ML model prediction $\hat{f}(\mathbf{x})$ at a given point.

Some machine learning models along with

- approximation

$$\hat{f}(\mathbf{x}) \simeq f(\mathbf{x})$$

can provide

- uncertainty estimation

$$\hat{\sigma}^2(\mathbf{x}) \simeq \mathbb{E}(\hat{f}(\mathbf{x}) - f(\mathbf{x}))^2.$$

Use cases:

- Possibility of rejection to predict:
  - out-of-distribution data detection
  - adversarial examples detection
- Active learning
- Bayesian optimization

Machine Learning Models and Uncertainty Estimation

- General approaches:
  - Analytic statistical approaches (variance estimates and confidence intervals based on CLT);
  - Bootstrap.
- Bayesian inference
- Model-specific approaches:
  - Gaussian processes for regression and classification;
  - Neural networks with variance-predicting subnetwork;
  - Decision trees variance estimation at leaves.

Bayesian Approach Machine Learning

Consider a probabilistic model (for example, neural network):

$$p(y \mid \mathbf{x}, \mathbf{w}),$$

where

- $\mathbf{x}$ is a neural network input;
- $\mathbf{w}$ is a vector of model parameters (i.e., neural network weights).

In Bayesian approach, $\mathbf{w}$ is assumed to be a random variable with some prior distribution:

$$\mathbf{w} \sim p.$$

Bayesian Model Averaging

Let us be given the dataset $D = \{\mathbf{x}_i, y_i\}_{i=1}^N$.
We can compute a posterior distribution:

$$p(\mathbf{w} \mid D) = \frac{p(D \mid \mathbf{w})p(\mathbf{w})}{\int p(D \mid \mathbf{w})p(\mathbf{w})d\mathbf{w}}.$$

The standard approach starts from considering the posterior predictive distribution

$$p(y \mid \mathbf{x}, D) = \int p(y \mid \mathbf{x}, \mathbf{w})\ p(\mathbf{w} \mid D)\ d\mathbf{w} = \mathbb{E}_{p(\mathbf{w}|D)}\ p(y \mid \mathbf{x}, \mathbf{w}).$$

If we can sample from posterior, then we can naturally perform Bayesian model averaging

$$\mathbb{E}_{p(\mathbf{w}|D)}\ p(y \mid \mathbf{x}, \mathbf{w}) \approx \bar{p}_T(y \mid \mathbf{x}, D) = \frac{1}{T}\sum_{t=1}^{T} p(y \mid \mathbf{x}, \mathbf{w}_t),$$

where $\mathbf{w}_t \sim p(\mathbf{w} \mid D),\ t = 1, \ldots, T$.

Why Bayesian?

In practice, the access to posterior $p(\mathbf{w} \mid D)$ gives you very reach information, i.e. not only average, but

- variance of $p(y \mid \mathbf{x}, \mathbf{w})$,
- quantiles.

However, sampling from $p(\mathbf{w} \mid D)$ might be very complicated.

That's why essentially non-Bayesian model averaging is often performed, i.e.

$$\bar{p}_T(y \mid \mathbf{x}, D) = \frac{1}{T} \sum_{t=1}^{T} p(y \mid \mathbf{x}, \mathbf{w}_t),$$

where parameters $\mathbf{w}_t$ are obtained via

- training of model from different random initialization $\Rightarrow$ ensembling;
- usage of dropout on inference stage $\Rightarrow$ Monte-Carlo dropout.

Bayesian Inference problem

In Bayesian problems we are interested in posterior distribution of latent variables:

$$p(z \mid x) = \frac{p(x \mid z)\ p(z)}{p(x)},$$

where x – observed data, z – latent (unobserved) variables.

Posterior allows **to reason about the uncertainties** in latent variables.

The following distributions are involved:

- $p(z \mid x)$ – posterior (our updated knowledge about z after we have observed data x);
- $p(z)$ – prior (our knowledge about z before we have observed data x);
- $p(x \mid z)$ – likelihood (probability of data x given latent variables z);
- $p(x)$ – normalizing constant for $p(z \mid x)$ to be a proper distribution.

Bayesian Inference problem

Posterior distribution:

$$p(z \mid x) = \frac{p(x \mid z)\ p(z)}{p(x)}.$$

The problem with exact posterior computation comes from the denominator:

$$p(x) = \int_z p(x, z) dz$$

as

- generally, the complexity of this integral computation grows exponentially with dimensionality;
- an exception is the case of conjugate pairs of prior and likelihood.

That is why in practice we use approximate Bayesian inference:

- MCMC (Markov Chain Monte Carlo);
- Variational Inference.

Markov Chain Monte Carlo

Markov chain Monte Carlo (MCMC) – a family of generic methods, which have theoretical guarantees under some mild conditions of receiving **exact samples** from the posterior distribution.

Idea:

- design a Markov chain $(z_k)_{k \in N}$, whose stationary distribution is

$$p(z \mid x) \propto p(x \mid z)\ p(z)$$

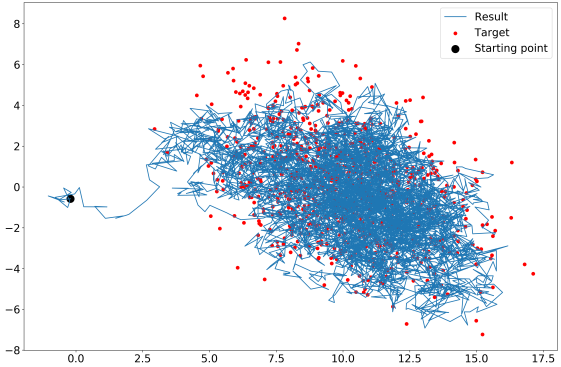
  known up to a normalization constant;
- it means, that starting from a sample from a prior $p(z)$, distribution of z_K converges to the target $p(z \mid x)$ as K goes to infinity.

Markov Chain Monte Carlo

Metropolis Hastings (MH) algorithms is an option:

- Draw a proposal z' from some transition density $\pi(z' \mid z, x)$.
- Accept / Reject the proposal with probability

$$\alpha(z, z') = 1 \wedge \frac{p(z' \mid x)\ \pi(z \mid z', x)}{p(z \mid x)\ \pi(z' \mid z, x)}.$$



Markov Chain Monte Carlo

Many recent advances for efficient MCMC methods, using Langevin dynamics, Hamiltonian Monte Carlo, ...

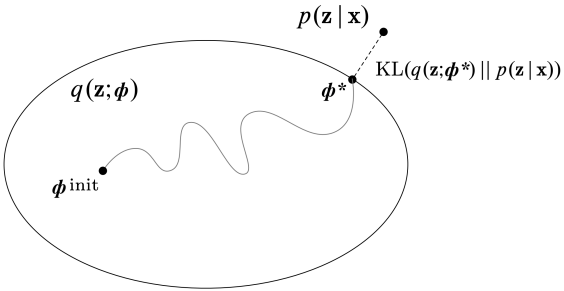
Advantages:

1. Generic (does not require introduction of a family of distributions).
2. Theoretical guarantees for fairly general cases.

Disadvantages:

- The rate of convergence could be really slow.
- You never know in advance how long to run a chain to receive decent samples from it.

Variational Inference



1. VI turns inference into optimization.
2. Introduce a variational family of distributions over the latent variables:

$$\mathcal{Q} = \{q_\phi(z), \quad \phi \in \Phi\}.$$

3. Fit the variational parameters ϕ to become close (usually in KL) to the exact posterior.

Source: David Blei, Rajesh Ranganath, Shakir Mohamed: Variational Inference:Foundations and Modern Methods. NIPS 2016, December 5, 2016.

Evidence Lower Bound (ELBO)

Typically, in VI methods we are interested in minimizing Kullback-Leibler divergence between variational distribution and the target:

$$\mathrm{KL}(q_\phi(z) \| p(z \mid x)) = \int q_\phi(z) \log \frac{q_\phi(z)}{p(z \mid x)}.$$

But in practice it is convenient to consider the equivalent task, which is called ELBO:

$$\mathcal{L}(x; \phi) = \int q_\phi(z) \log \frac{p(z \mid x)}{q_\phi(z)} dz$$

and can be interpreted as a lower bound for the log-likelihood:

$$\log p(x) \geq \mathcal{L}(x; \phi).$$

Evidence Lower Bound (ELBO)

- A key observation here is that to compute both these formulas we still need to have $p(z \mid x)$.
- And as we discussed, it is complicated because of the normalizing constant $p(x)$.
- It can be shown, that for optimization **we do not need** $p(z \mid x)$ **to be normalized**:

$$\mathcal{L}(x; \phi) = \int q_\phi(z) \log \frac{p(z \mid x)}{q_\phi(z)} dz = \int q_\phi(z) \log \frac{p(x, z)}{p(x)\ q_\phi(z)} dz = \int q_\phi(z) \log \frac{p(x, z)}{q_\phi(z)} dz - C.$$

as $p(z, x) = p(x \mid z)p(z)$ and **the constant** $C = p(x)$ **does not depend on** ϕ.

- The integral is usually computed using MC-estimate.

Variational Inference

Advantages:

1. Compared to MCMC, scales faster to high dimensions.
2. Allows us to **leverage complexity and accuracy**, selecting a family of distribution.
3. Allows efficient mini-batch optimization, which scales to massive data.

Disadvantages:

- By construction it **introduces bias**, since we are considering only a family of parametric distributions.
- It means, we in principle cannot obtain exact approximation of the posterior.

Mean Field Gaussian Variational Inference

The most straightforward choice of the Variational family is fully factorized Gaussian.

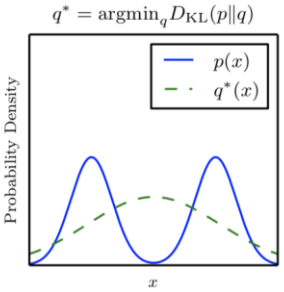
In this case,

$$q_\phi = \mathcal{N}(\mu_\phi, diag\{\sigma_\phi^2\}),$$

where each μ_ϕ and σ_ϕ^2 are vectors of size $\dim(z)$.

Main advantage: there are only $2\dim(z)$ learnable parameters.

The **disadvantage** is in its expressiveness:



Normalizing Flows

- Recently, new deterministic parametric models, **Normalizing Flows**, were suggested to transform one probability density to another.
- Formally, **Normalizing Flow:**
  - invertible parametrized transformation T_ϕ;
  - both T_ϕ and T_ϕ^{-1} are differentiable.
- Moreover, determinant J_{T_ϕ} of Jacobian of T_ϕ should be easy to compute.
- Given a sample from a simple density $u \sim g(u)$, we deterministically transform it as

$$z = T_\phi(u).$$

- Using *change of variables* formula we can compute the resulting density:

$$q_\phi(z) = g(u)|J_T(u)|^{-1}.$$

Normalizing Flows

In case of K flows, resulting density is expressed as follows:

$$\log q_\phi^K(z) = \log g(u_0) - \sum_{i=0}^{K} \log |J_{T_i}(u_i)|.$$

This log-density is then used in the ELBO, using Monte Carlo estimation of the integral:

$$\mathcal{L}(x; \phi) = \int q_\phi^K(z) \log \frac{p(x, z)}{q_\phi(z)} dz,$$

where

$$\log \frac{p(x, z)}{q_\phi(z)} = \log p(x, z) - \log g(u_0) + \sum_{i=0}^{K} \log |J_{T_i}(u_i)|.$$

Normalizing Flows

- We can compose these transformations, and use them in variational inference framework, minimizing KL between resulting distribution $q_\phi^K(z)$ and target.
- Even simple transformation stacked on top of each other are capable to transform a simple density to a complex one:

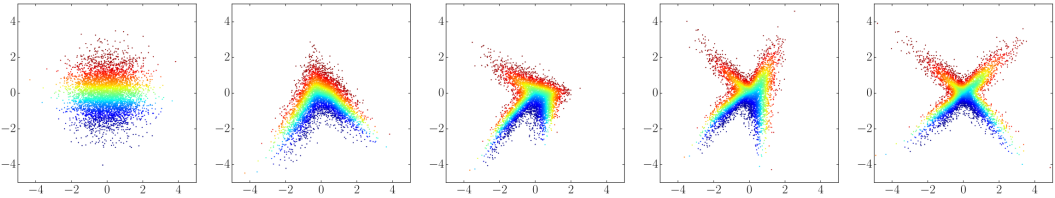


Figure 1: Example of a 4-step flow transforming samples from a standard-normal base density to a cross-shaped target density.

Source: Papamakarios et al., 2019, Normalizing flows for probabilistic modeling and inference.

Examples of normalizing flows

We briefly consider examples of the most known normalizing flows. Some of them are simple, yet very expressive.

1. **Planar and Radial Flows**. The pioneering work on normalizing flows were planar and radial flows were introduced. Simple form transitions, parametrized by learnable parameters.
2. **RealNVP**. The idea is to keep some part of coordinates unchanged, while transform the rest using an affine transformation. Parameters of affine transformation are usually arbitrarily complex neural networks.
3. **IAF**. Invertible Autoregressive Flow (IAF) also performs affine transformation, but in contrast to RealNVP, it changes all coordinates, without keeping unchanged. To compute determinant of Jacobian effectively, we introduce a restriction, that neural networks which perform Affine transformation should be autoregressive.

D. Rezende et al., Variational Inference with Normalizing Flows

L. Dinh et al., Density estimation using Real NVP

D. Kingma et al., Improved Variational Inference with Inverse Autoregressive Flow

MetFlow: Metropolized Flows

- We already know, that **Variational Inference methods suffer from bias**.
- Even if variational family is very expressive, there is still no guarantees it contains target distribution.
- **MCMC methods do not have the issue. But they could be impractically slow.**
- **MetFlow** proposes a possible **way to enhance Variational Inference**
  - with Metropolis-Hastings algorithm
  - using expressive **Normalizing Flows as proposals.**

Source: Thin, **Kotelevskii**, Denain, Grinsztajn, Durmus, **Panov**, Moulines, MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference

Why is it difficult to make stochastic transitions within VI framework?

$$\mathcal{L}(\phi; x) = \int q_\phi^K(z) \log \frac{p(x, z)}{q_\phi^K(z)} dz.$$

1. Deterministic normalizing flows:

$$q_\phi^K(z) = q_\phi^0(z) \prod_{i=1}^{K} |J_{T_\phi^i}(z_i)|^{-1},$$

where $J_{T_\phi^i}(z_i)$ is a determinant of Jacobian of T_ϕ^i;

2. General case:

$$q_\phi^K(z) = q_\phi(z_K) = \int q_\phi(z_0, \ldots, z_{K-1}, z_K) dz_{0 \ldots K-1}.$$

So it requires integration over all intermediate variables to be used in the ELBO.

MetFlow: Notations

- $T_{\phi,u_{k+1}}(z_k)$ – proposal mapping;
- $(u_k)_{k\in N*}$ – sequence of i.i.d. random variables, called innovation noise with density h;
- ϕ – parameters used in proposal;
- $\alpha_{\phi,u}(z)$ – acceptance function, associated with current point z and proposal point $T_{\phi,u}(z)$.
- $M_{\phi,h}(z_k, A) = \int h(u) Q_{\phi,u}(z_k, A) du$ – integrated kernel;
- $Q_{\phi,\mathbf{u}}(z, A) = \alpha_{\phi,\mathbf{u}}(z)\delta_{T_{\phi,\mathbf{u}}(z)}(A) + \{1 - \alpha_{\phi,\mathbf{u}}(z)\}\delta_z(A)$ – conditional kernel, associated with u.

MetFlow: New Variational Family

Denote ξ_ϕ^0 our initial distribution.

Variational family after applying K such Markov kernels:

$$Q = \{\xi_\phi^K = \xi_\phi^0 M_{\phi,h_1} M_{\phi,h_2} \dots M_{\phi,h_K} : \phi \in \Phi\}.$$

We want to minimize KL divergence: $KL(\xi_\phi^K \| \pi)$.

Next key assumption: $T_{\phi,u}$ is C^1 diffeomorphism, and $J_{T_{\phi,u}}$ denotes determinant of Jacobian of the transformation.

MetFlow: Notation and theory behind

Lemma 1. Assume that ξ_ϕ^0 admits density m_ϕ^0.
Assume that $T_{\phi,u}$ is C^1 diffeomorphism. Then the distribution

$$\xi_\phi^1(\cdot \mid u) = \int_{\mathbb{R}^d} m_\phi^0(z_0) Q_{\phi,u}(z_0, \cdot) dz_0,$$

has density, given by:

$$m_\phi^1(z \mid u) = \alpha_{\phi,u}(T_{\phi,u}^{-1}(z)) m_\phi^0(T_{\phi,u}^{-1}(z)) J_{T_{\phi,u}^{-1}}(z) + (1 - \alpha_{\phi,u}(z)) m_\phi^0(z).$$

And distribution ξ_ϕ^1 has density given by $m_\phi^1(z) = \int m_\phi^1(z \mid u) h(u) du$.

MetFlow: Notation and theory behind

Proposition 1: Assume that ξ_ϕ^0 admits density m_ϕ^0. Assume that $T_{\phi,u}$ is C^1 diffeomorphism. Then for any $\{u_i\}_{i=1}^K \in \mathrm{U}^K$ the distribution $\xi_\phi^K(\cdot \mid u_{1\dots K}) = \xi_\phi^0 Q_{\phi,u_1} \dots Q_{\phi,u_K}$ has density m_ϕ^K given by:

$$m_\phi^K(z \mid u_{1\dots K}) = \sum_{a_{1\dots K} \in \{0,1\}^K} m_\phi^K(z, a_{1\dots K} \mid u_{1\dots K}),$$

$$m_\phi^K(z, a_{1\dots K} \mid u_{1\dots K}) = \prod_{i=1}^K \alpha_{\phi,u_i}^{a_i}(\circ_{j=i}^K T_{\phi,u_j}^{-a_j}(z))\, m_\phi^0(\circ_{j=1}^K T_{\phi,u_j}^{-a_j}(z))\, J_{\circ_{j=1}^K T_{\phi,u_j}^{-a_j}(z)}(z),$$

where $\circ_{j=i}^K T_j = T_i \circ \dots \circ T_K$.

MetFlow: Notation and theory behind

We can derive density of variational family after application of K such kernels M:

$$m_\phi^K(z, a_{1\dots K} \mid u_{1\dots K}) = \prod_{i=1}^{K} \alpha_{\phi,u_i}^{a_i}(\circ_{j=i}^{K} T_{\phi,u_j}^{-a_j}(z)) m_\phi^0(\circ_{j=1}^{K} T_{\phi,u_j}^{-a_j}(z)) J_{\circ_{j=1}^{K} T_{\phi,u_j}^{-a_j}(z)}(z),$$

where $\circ_{j=i}^{K} T_j = T_i \circ \cdots \circ T_K$.

And marginal density can be obtained by

$$m_\phi^K(z \mid u_{1\dots K}) = \sum_{a_{1\dots K} \in \{0,1\}^K} m_\phi^K(z, a_{1\dots K} \mid u_{1\dots K}),$$

$$m_\phi^K(z) = \int m_\phi^K(z \mid u_{1\dots K}) h(u_{1\dots K}) du_{1\dots K}.$$

MetFlow: A New ELBO

- **Objective:** optimize the ELBO

$$\mathcal{L}(\phi; x) = \int \log\left(\frac{p(x,z)}{m_\phi^K(z)}\right) m_\phi^K(z)\mathrm{d}z\,.$$

- **Problem**: The distribution m_ϕ^K is untractable (a mixture of 2^K *components*)!!!

- **Idea**: Define a new ELBO

$$\mathcal{L}_{aux}(\phi; x) = \sum_{a_{1\ldots K}\in\{0,1\}^K} \int h(u_{1\ldots K}) m_\phi^K(z_K, a_{1\ldots K} \mid u_{1\ldots K}) s_\phi(x, z_K, a_{1\ldots K}, u_{1\ldots K})\mathrm{d}z_K\mathrm{d}u_{1\ldots K}\,,$$

where

$$s_\phi(x, z_K, a_{1\ldots K}, u_{1\ldots K}) = \log\left(2^{-K} p(x, z_K)/m_\phi^K(z_K, a_{1\ldots K} \mid u_{1\ldots K})\right)\,.$$

A new ELBO

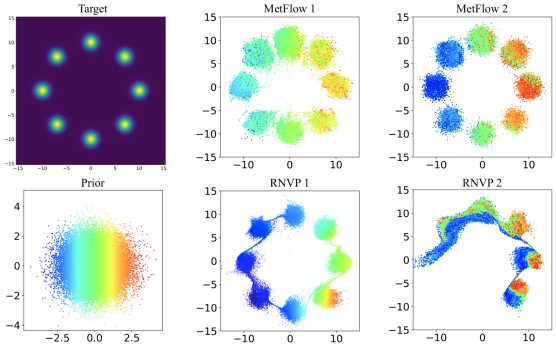
This is a proper evidence lower bound!!!

Jensen's inequality for $m_\phi^K(z_K, a_{1\dots K} \mid u_{1\dots K})$ indeed shows:

$$\sum_{a_{1\dots K}\in\{0,1\}^K} \int m_\phi^K(z_K, a_{1\dots K} \mid u_{1\dots K}) \log \left(\frac{2^{-K} p(x, z_K)}{m_\phi^K(z_K, a_{1\dots K} \mid u_{1\dots K})} \right) \mathrm{d}z_K \leq \log p(x)\,.$$

MetFlow: Results

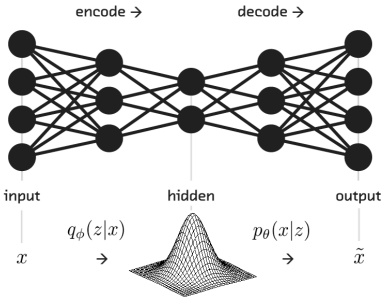
- Sampling from a mixture of 8 Gaussians.
- MetFlow consists here of 5 RealNVP flows, separated by the MH algorithm.
- As a competitor, we have 5 RealNVP flows of the same architecture.



- We see, that MH algorithm prevents “tails” between modes.

Intuition behind VAE

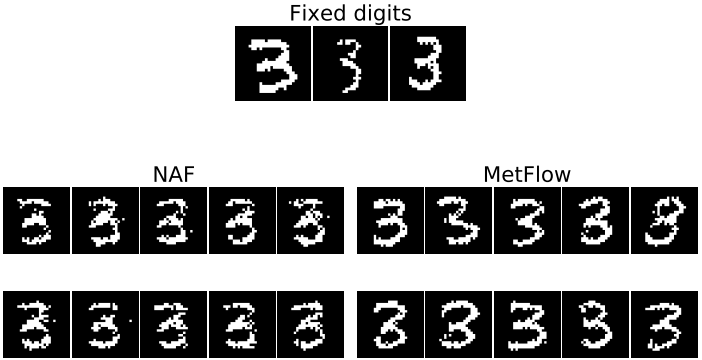
- Variational AutoEncoder (or VAE) is one of the most popular generative models nowadays.
- In contrast to GAN, in VAE we simultaneously train:
  - inference model (or encoder, which learns meaningful latent representation);
  - generation model (or decoder, which reconstructs latent variables into objects).
- Schematically, VAEs architecture could be expressed as:



Source: What The Heck Are VAE-GANs?

MetFlow to MetVAE

- Fix a generative model p_θ from VAE achieving SOTA results.
- Approximate the posterior $p_\theta(z \mid (x_i)_{i=1}^L)$ with a NAF and MetFlow with 5 RealNVP flows.



MetFlow to MetVAE

As an example, we will address collaborative filtering problem.

- For each user u, the model starts by sampling a D-dimensional latent representation z_u from a standard Gaussian prior.
- The latent representation z_u is transformed via a non-linear function g_θ to produce a probability distribution $\pi_\theta(z_u)$ over I items. Here we set

$$\pi_\theta(z) = \text{softmax}(g_\theta(z)).$$

- Given the total number of interactions $N_u = \sum_i x_{u,i}$, x_u is assumed to be sampled from $x_u \mid z_u, N_u \sim \text{Mult}\,(N_u, \pi_\theta(z_u))$
- The log-likelihood for user u conditioned on the latent representation is

$$\log p_\theta(x_u \mid z_u) = \sum_{i=1}^{I} x_{u,i} \log \pi_{\theta,i}(z_u).$$

MetVAE: Evaluation of the models

- Need to have access to number of items chosen by the user for the generative model.
- To assess performance, use top-K metrics.
- Complete the items selected by an user and compare it to all of the selections using

$$\text{Recall@}n = \frac{|\text{relevant items} \cap \text{recommended items}|}{|\text{recommended items}|};$$

$$\text{nDCG@}n = \frac{\text{DCG@}n}{\text{IDCG@}n},$$

where

$$\text{DCG@}n = \sum_{i=1}^{n} \text{rel}(i)/\log_2(i+1), \text{ and } \text{IDCG@}n = \sum_{i=1}^{|R_n|} 1/\log_2(i+1).$$

R_n: set of the n relevant items

rel(i): relevance function of the i-th recommended item of the list, equal to 1 if the item ranked at i is relevant, and 0 else.

MetVAE: Datasets & Competitors

- Three real world datasets: Foursquare [Yuan et al., 2013], Gowalla [Cho et al., 2011], MovieLens.
- Preprocess to binarize them to fit CF task [Liang et al., 2018].
- Competitors
  - MultiVAE [Liang et al., 2018] a VAE for CF.
  - WRMF [Hu et al., 2008] a weighted regularized matrix factorization for implicit feedback datasets.
  - BPR [Rendle et al., 2009] a Bayesian ranking method.
  - GlbAvg, a generic naive baseline (recommends the most popular items among all users).

MetVAE: Results

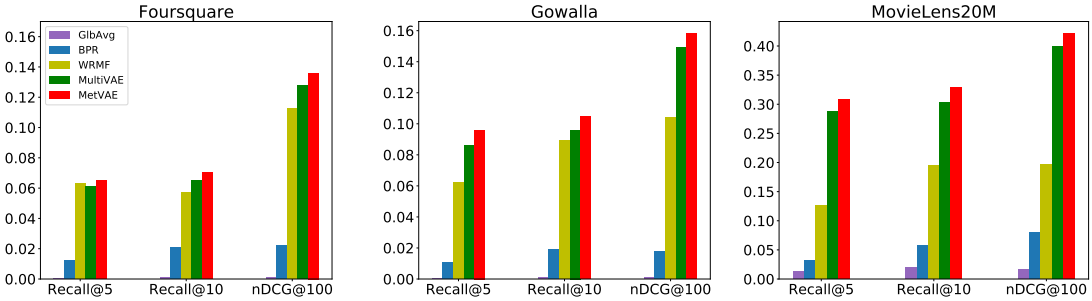


Figure: Recommendation scores in terms of Recall@5, Recall@10 and nDCG@100 of the considered methods on Foursquare, Gowalla and MovieLens datasets. `MetVAE` shows consistently better results compared to other methods.

Conclusions and Outlook

Summary:

- Bayesian inference is very promising approach, but it is extremely challenging due to computational complexity.
- Variational inference is a viable solution.
- Enriching variational family via normalizing flows leads to very expressive approximations of posterior.
- Enriching VI with MCMC-type transitions can significantly improve the quality.

Areas to grow:

- application to Bayesian neural networks;
- approximate sampling from pure BNN (MCMC, HMC, ...).

Thank you for your attention!

References

Cho, E., Myers, S. A., and Leskovec, J. (2011).

Friendship and mobility: User movement in location-based social networks.

KDD '11.

Hu, Y., Koren, Y., and Volinsky, C. (2008).

Collaborative filtering for implicit feedback datasets.

In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 263–272, USA. IEEE Computer Society.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. (2018).

Variational autoencoders for collaborative filtering.

In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 689–698, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009).

Bpr: Bayesian personalized ranking from implicit feedback.

In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 452–461, Arlington, Virginia, USA. AUAI Press.

Yuan, Q., Cong, G., Ma, Z., Sun, A., and Thalmann, N. M. (2013).

Time-aware point-of-interest recommendation.

In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 363–372, New York, NY, USA. ACM.