1 Notations

Let $f(x;\theta)$ a function of x with parameter θ. Let $y=f(x;\theta)$ an observable, thus the observed value obeys a Gaussian distribution. Let D denotes a set of observations, $D:=\{(x_i,y_i,\sigma_i)\colon i=1,...,N_D\}$, wherein x_i is the ith input, y_i its observed value, and σ_i the observational error of y_i. We may employ mini-batch technique, thus denote $D_m:=\{(x_i,y_i,\sigma_i)\colon i=1,...,N_m\}\subset D$ as a mini-batch, with batch-size $N_m\leqslant N_D$. We use $\mathbb{E}_{f(\theta)}[g(\theta)]$ represent the expectation of function g of a random variable obeys the p.d.f. f. Φ is for Gaussian p.d.f..

2 Neural Network for Posterior (nn4post)

2.1 The Model

Suppose we have some prior on θ, $p(\theta)$, we gain the un-normalized posterior $p(D|\theta)\,p(\theta)$. With D arbitrarily given, this un-normalized posterior is a function of θ, denoted by $p(\theta;D)$[1].[2]

We we are going to do is fit this $p(\theta;D)$ by ANN for any given D. To do so, we have to assume that $\mathrm{supp}\{p(\theta;D)\}=\mathbb{R}^d$ for some $d\in\mathbb{N}^+$ (i.e. has no compact support) but decrease exponentially fast as $\|\theta\|\to+\infty$. With this assumption, $\ln p(\theta;D)$ is well-defined. For ANN, we propose using Gaussian function as the activation-function. Thus, we have the fitting function

$$q(\theta;a,\mu,\zeta):=\sum_{i=1}^{N_c}c_i(a)\left\{\prod_{\alpha=1}^{d}\Phi(\theta_\alpha-\mu_{i\alpha},\sigma(\zeta_{i\alpha}))\right\},\tag{1}$$

where

$$c_i(a) = \frac{\exp(a_i)}{\sum_{j=1}^{N_c}\exp(a_j)}=\mathrm{softmax}(i;a);\tag{2}$$

$$\sigma(\zeta_{i\alpha}) = \ln(1+\exp(\zeta_{i\alpha})),\tag{3}$$

and $a_i,\mu_{i\alpha},\zeta_{i\alpha}\in\mathbb{R}$ for $\forall i,\forall\alpha$, and

$$\Phi(x;\mu,\sigma):=\sqrt{\frac{1}{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)\tag{4}$$

being the Gaussian p.d.f.. The introduction of ζ is for numerical consideration, see below.

2.1.1 Numerical Consideration

If, in q, we regard w, μ, and σ as independent variables, then the only singularity appears at $\sigma=0$. Indeed, σ appears in Φ (as well as the derivatives of Φ) as denominator only, while others as numerators. However, once doing numerical iterations with a finite step-length of σ, the probability of reaching or even crossing 0 point cannot be surely absent. This is how we may encounter this singularity in practice.

Introducing the ζ is our trick of avoiding this singularity. Precisely, using a singular map that pushes the singularity to infinity solves the singularity. In this case, using softplus(.) that pushes $\sigma=0$ to $\zeta\to-\infty$, so that, with finite steps of iteration, singularity (at $-\infty$) cannot be reached.

This trick (i.e. pushing a singularity to infinity) is the same as in avoiding the horizon-singularity of Schwarzschild solution of black hole.

1. This is why we use " ; " instead of " , ", indicating that D has been (arbitrarily) given and fixed.
2. The normalized posterior $p(\theta|D)=p(D|\theta)\,p(\theta)/p(D)=p(\theta;D)/p(D)$, by Bayes's rule.

2.2 Interpretation

2.2.1 As a Mixture Distribution

$q(\theta;\, a,\, \mu,\, \zeta)$ has a probabilistic interpretation. $\prod_{j=1}^{d} \Phi(\theta_j - \mu_{ij},\, \sigma(\zeta_{ij}))$ corresponds to multi-dimensional Gaussian distribution (denote $\mathcal{N}$), with all dimensions independent with each other. The $\{c_i(a)\}$ is a categorical distribution, randomly choosing the Gaussian distributions. Thus $q(\theta; a, \mu, \zeta)$ is a composition: categorical $\to$ Gaussian. This is the *mixture distribution*.

2.2.2 As a Generalization

This model can also be interpreted as a direct generalization of mean-field variational inference. Indeed, let $N_c = 1$, this model reduces to mean-field variational inference. Remark that mean-field variational inference is a mature algorithm and has been successfully established on many practical applications.

2.2.3 As a Neural Network

2.3 Marginalization

This model can be marginalized easily. This then benefits the transfering of the model components. Precisely, for any dimension-index β given, we can marginalize all other dimensions directly, leaving

$$q(\theta_\beta; a, \mu, \zeta) = \prod_{\forall \gamma \neq \beta} \int \mathrm{d}\theta_\gamma \sum_{i=1}^{N_c} c_i(a) \left\{ \prod_{\alpha=1}^{d} \Phi(\theta_\alpha;\, \mu_{i\alpha}, \sigma(\zeta_{i\alpha})) \right\} \tag{5}$$

$$= \sum_{i=1}^{N_c} c_i(a)\, \Phi(\theta_\beta;\, \mu_{i\beta}, \sigma(\zeta_{i\beta})), \tag{6}$$

where employed the normalization of Φ.

2.4 Loss-Function

We employ "evidence of lower bound" (ELBO)[3]. It is ensured to have a unique global maximum, at which $p(\theta; D) = q(\theta; a, \mu, \zeta)$.

$$\mathrm{ELBO}(a, \mu, \zeta) := \mathbb{E}_{\theta \sim q(\theta; a, \mu, \zeta)}[\ln p(\theta; D) - \ln q(\theta; a, \mu, \zeta)].$$

Since there's no compact support for both $p(\theta; D)$ and $q(\theta; a, \mu, \zeta)$, ELBO is well-defined. The loss-function (or say loss-function, performance, etc) of the fitting is then defined as $\mathcal{L} = -\mathrm{ELBO}$, i.e.

$$\mathcal{L}(a, \mu, \zeta) = -\mathbb{E}_{\theta \sim q(\theta; a, \mu, \zeta)}[\ln p(\theta; D) - \ln q(\theta; a, \mu, \zeta)],$$

or, recall $\mathbb{H}[q] := -\mathbb{E}_q[\ln q]$ for any distribution q,

$$\mathcal{L}(a, \mu, \zeta) = -\mathbb{E}_{\theta \sim q(\theta; a, \mu, \zeta)}[\ln p(\theta; D)] - \mathbb{H}[q(\theta; a, \mu, \zeta)].$$

3 Optimization

3.1 ADVI

Automatic differentation variational inference (ADVI)[4] has the advantage that the variance of its Monte Carlo integral is orderly smaller than that of black box variational inference (i.e. optimization directly using ELBO without further reparameterization).

3. The relation between ELBO and KL-divergence is that $\mathrm{ELBO} = -\mathrm{KL}(q\|p) + \mathrm{Const}$.
4. See, Kucukelbir, et al, 2016.

3.1.1 Derivation

Precisely, recall $\mathbb{E}$ for mean value, Φ for Gaussian p.d.f., $\sigma(.)$ for softplus function, $c(.)$ for softmax function, and

$$q(\theta; a, \mu, \zeta) = \sum_{i=1}^{N_c} c_i(a)\, \Phi(\theta; \mu_i, \sigma(\zeta_i)), \tag{7}$$

we have, for any function f,

$$\begin{aligned} \mathbb{E}_{q(\theta;a,\mu,\zeta)}[f(\theta)] &= \int \mathrm{d}\theta \sum_{i=1}^{N_c} c_i(a)\, \Phi(\theta; \mu_i, \sigma(\zeta_i))\, f(\theta) && (8)\\ &= \sum_{i=1}^{N_c} c_i(a) \int \mathrm{d}\theta\, \Phi(\theta; \mu_i, \sigma(\zeta_i))\, f(\theta) && (9)\\ &= \sum_{i=1}^{N_c} c_i(a)\, \mathbb{E}_{\Phi(\theta;\mu_i,\sigma(\zeta_i))}[f(\theta)]. && (10) \end{aligned}$$

With this general relation, we get

$$\begin{aligned} \mathcal{L}(a, \mu, \zeta) &= -\{\mathbb{E}_{q(\theta;a,\mu,\zeta)}[\ln p(\theta; D)] - \mathbb{E}_{q(\theta;a,\mu,\zeta)}[\ln q(\theta; a, \mu, \zeta)]\} && (11)\\ &= -\sum_{i}^{N_c} c_i(a)\, \mathbb{E}_{\Phi_i(\theta;\mu_i,\sigma(\zeta_i))}[\ln p(\theta; D) - \ln q(\theta; a, \mu, \zeta)] && (12) \end{aligned}$$

Then, for $\forall i = 1, ..., N_c$, $\forall \alpha = 1, ..., N_d$, let

$$\eta_\alpha := \frac{\theta_\alpha - \mu_{i\alpha}}{\sigma(\zeta_{i\alpha})}, \tag{13}$$

we have

$$\theta_\alpha = \sigma(\zeta_{i\alpha})\, \eta_\alpha + \mu_{i\alpha} \tag{14}$$

(or $\theta = \sigma(\zeta_i)\, \eta + \mu_i$ if hide the α index). So, for any i-component, we transform

$$\begin{aligned} \theta &\rightarrow \sigma(\zeta_i)\, \eta + \mu_i; && (15)\\ \mathbb{E}_{\Phi(\theta;\mu_i,\sigma(\zeta_i))}[f(\theta)] &\rightarrow \mathbb{E}_{\Phi(\eta;0,1)}[f(\sigma(\zeta_i)\, \eta + \mu_i)], && (16) \end{aligned}$$

where function f is arbitrary, thus holds for both $\ln p(.; D)$ and $\ln q(.; a, \mu, \zeta)$.

With this setting, the derivatives to μ and to ζ are completely independent of $\mathbb{E}[.]$. And now, the loss function becomes

$$\mathcal{L}(a, \mu, \zeta) = -\sum_{i}^{N_c} c_i(a)\, \mathbb{E}_{\Phi(\eta;0,1)}[\ln p(\sigma(\zeta_i)\, \eta + \mu_i; D) - \ln q(\sigma(\zeta_i)\, \eta + \mu_i; a, \mu, \zeta)].$$

3.2 Redefination of $\partial\mathcal{L}/\partial a$

3.2.1 Gauge Fixing

Let Δt the learning-rate. Then the updation of a_i at one iteration by gradient decent method is

$$\Delta a_i = -\frac{\partial \mathcal{L}}{\partial a_i}(a, \mu, \zeta)\, \Delta t.$$

Notice that redefining the $\partial\mathcal{L}/\partial a$ by

$$\frac{\partial\mathcal{L}}{\partial a_i}(a,\mu,\zeta)\to\frac{\partial\mathcal{L}}{\partial a_i}(a,\mu,\zeta)+C,$$

where C can be any constant, leaves the updation of $c_i(a)$ invariant, since it makes

$$\Delta a_i\to-\frac{\partial\mathcal{L}}{\partial a_i}(a,\mu,\zeta)\,\Delta t-C\,\Delta t,$$

thus

$$c_i(a+\Delta a)\to\frac{\exp\left(a_i+\Delta a_i-C\,\Delta t\right)}{\sum_j\exp\left(a_j+\Delta a_j-C\,\Delta t\right)}=\frac{\exp\left(a_i+\Delta a_i\right)}{\sum_j\exp\left(a_j+\Delta a_j\right)}=c_i(a+\Delta a).$$

This C thus provides an additional dof.[5] We can tune the value of C so that the updation of a_i is numerically stable. Indeed, let C be the average of $\{\partial\mathcal{L}/\partial a_i\colon i=1,...,N_c\}$, we find a pretty stability of a as well as a pretty accuracy of c in the iteration process of optimization, as the experiment on Gaussian mixture model shows.

3.2.2 Re-scaling of a

There shall be an additional hyper-parameter for the re-scaling of a. The re-scaling factor, constant r, redefines

$$c_i(a):=\mathrm{softmax}(i,r\,a).$$

Tuning this additional hyper-parameter may improve the optimization.

3.3 Training Strategy

Generally we hope that the gradient diminishes when and only when the optimization converges. However, even far from convergence, a tiny c_i will diminish all the derivatives in the i-component, e.g. derivatives of a_i, $\mu_{i\alpha}$, $\zeta_{i\alpha}$, since all these derivatives are proportional to c_i.

A proper strategy is setting the re-scaling factor of a, the r, small enough (e.g. vanishing) until other variables vary little, and then reset the r to any value eagered. This can avoid the problem that a pre-mature state of training may give a tiny c_i, which then slows down the training of all variables in the i-component.

3.4 Approximations

Comparing to the traditional MAP approach, using multi-peak mixture model makes the $\mathbb{H}(q)$ complicated, especially in the optimization process.

3.4.1 Entropy Lower Bound

Consider any mixture distribution with p.d.f. $\sum_i^{N_c}c_i\,q_i$ where cs are the categorical probabilities and q_i the p.d.f. of the component distributions of the mixture.

$$\mathbb{H}\left[\sum_i^{N_c}c_i\,q_i\right]\geqslant\sum_i^{N_c}c_i\,\mathbb{H}[q_i].$$

5. As CL explained, the cs have less dofs as they look, since $\sum_i^{N_c}c_i=1$. This restriction can provides an additional gauge. And the new dof C fixes this gauge.

So, if define

$$\mathcal{L}'(a,\mu,\zeta) := -\sum_{i}^{N_c} c_i(a)\left\{\mathbb{E}_{\Phi(\eta;0,1)}[\ln p(\sigma(\zeta_i)\,\eta+\mu_i;D)] + \mathbb{H}[\Phi(\theta,\mu_i,\sigma(\zeta_i))]\right\}, \tag{17}$$

then we have

$$\mathcal{L}' \geqslant \mathcal{L} \geqslant \min(\mathcal{L}) > -\infty,$$

thus $\mathcal{L}'$ has an global minimum. In this way, the entropy part becomes completely analytic (and simple).

However, as experiment on Gaussian mixture model shows, using entropy lower bound cannot get the enough accuracy as using entropy does.

3.5 Stochastic Optimization

3.5.1 Difference between Bayesian and Traditional Methods

Suppose, instead of use the whole dataset, we employ mini-batch technique. Since all data are independent, if suppose that D_m is unbiased in D, then we have,

$$\ln p(D|\theta) = \sum_{D} p((x_i,y_i,\sigma_i)|\theta) \approx \frac{N_D}{N_m}\sum_{D_m} p((x_i,y_i,\sigma_i)|\theta) = \frac{N_D}{N_m}\ln p(D_m|\theta). \tag{18}$$

Then,

$$\ln p(\theta;D) = \ln p(D|\theta) + \ln p(\theta) = \frac{N_D}{N_m}\ln p(D_m|\theta) + \ln p(\theta), \tag{19}$$

thus as previous

$$\ln p(\theta;D) = \frac{N_D}{N_m}\sum_{(x_i,y_i,\sigma_i)\in D_m}\left\{-\frac{1}{2}\ln(2\pi\sigma_i^2) - \frac{1}{2}\left(\frac{y_i - f(x_i;\theta)}{\sigma_i}\right)^2\right\} + \ln p(\theta). \tag{20}$$

In this we meet one of the main differences between the Bayesian and the traditional. In the traditional method, N_D does not matters in training, being absent in the optimizer. However, in Bayesian, the number of data that are employed is encoded into Bayesian model, and has to, since the greater number of data gives more confidence. So, while using stochastic optimization in Bayesian mode, the factor N_D/N_m of likelihood has to be taken into account. We have to know how many data we actually have, thus how confident we are.

4 Problems and Solutions

4.1 Non-Synchronous Problem

(XXX see the e-mail.)

5 Deep Learning

It cannot solve the vanishing gradient problem of deep neural network, since this problem is intrinsic to the posterior of deep neural network. Indeed, the posterior has the shape like $\exp(-x^2/\sigma^2)$ with $\sigma \to 0$, where x is the variable (argument) of the posterior. It has a sharp peak, located at a tiny area, with all other region extremely flat. The problem of find this peak, or equivalently, finding its tiny area, is intrinsically intractable.

So, even for Bayesian neural network, a layer by layer abstraction along depth cannot be absent.

6 Transfer Learning

Transfer learning demands that the model can be separated so that, say, some lower level layers can be extracted out and directly transfered to another model as its lower level layers without any modification on these layers. To do so, we have to demand that the marginalization of the $q(\theta; a, \mu, \zeta)$ on some θ_is shall be easy to take. Indeed, the marginalization of our model is straight forward.

7 Why not MCMC?

Instead, the MCMC approximation to posterior cannot be marginalized easily, and even intractable. So, MCMC approximation cannot provide transfer learning as we eager. This is the most important reason that we do not prefer MCMC. Furthermore, MCMC is not greedy enough so that it converges quite slow, especially in high-dimensional parameter-space.