UNCERTAINTY QUANTIFICATION IN MACHINE LEARNING MISCELLANEOUS RESULTS

MIGUEL A. LERMA

Abstract. Miscellaneous definitions and results useful in uncertainty quantification for machine learning.

Variance and Entropy

Law of Total Variance. If X and Y are random variables on the same probability space, and the variance of Y is finite, then

$$\mathbb{V}[Y] = \mathbb{E}[\mathbb{V}[Y \mid X]] + \mathbb{V}[\mathbb{E}[Y \mid X]]\,,$$

where $\mathbb{E}$ denotes expectation and $\mathbb{V}$ means variance.

Proof: Recall that by the law of total expectation: $\mathbb{E}[f(Y)] = \mathbb{E}[\mathbb{E}[f(Y) \mid X]]$.

So, we have

$$\mathbb{V}(Y) = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2\,.$$

On the other hand, by the law of total expectation (with $f(Y) = Y^2$):

$$\mathbb{E}[Y^2] = \mathbb{E}[\mathbb{E}[Y^2 \mid X]] = \mathbb{E}[\mathbb{V}[Y \mid X] + \mathbb{E}[Y \mid X]^2] = \mathbb{E}[\mathbb{V}[Y \mid X]] + \mathbb{E}[\mathbb{E}[Y \mid X]^2]\,,$$

hence

$$\begin{aligned}\mathbb{V}(Y) &= \overbrace{\mathbb{E}[\mathbb{V}[Y \mid X]] + \mathbb{E}[\mathbb{E}[Y \mid X]^2]}^{\mathbb{E}[Y^2]} - \mathbb{E}[Y]^2 \\ &= \mathbb{E}[\mathbb{V}[Y \mid X]] + \underbrace{\mathbb{E}[\mathbb{E}[Y \mid X]^2] - \mathbb{E}[\mathbb{E}[Y \mid X]]^2}_{\mathbb{V}[\mathbb{E}[Y|X]]} \\ &= \mathbb{E}[\mathbb{V}[Y \mid X]] + \mathbb{V}[\mathbb{E}[Y \mid X]]\,,\end{aligned}$$

where we have used again the law of total expectation $\mathbb{E}[Y] = \mathbb{E}[\mathbb{E}[Y \mid X]]$ in the second step. □

Date: September 16, 2023.

Entropy, Cross-Entropy.

Entropy:

$$H(P) = \mathbb{E}_P(\log(P))\,.$$

where $\mathbb{E}$ denotes expectation.

Cross-Entropy:

$$H(P,Q) = \mathbb{E}_P(\log(Q))\,.$$

For discrete distributions:

$$H(P) = -\sum_{x\in X} P(x)\log(P(x))\,,$$
$$H(P,Q) = -\sum_{x\in X} P(x)\log(Q(x))\,.$$

For continuous distributions:

$$H(P) = -\int_{-\infty}^{\infty} P(x)\log(P(x))\,dx\,,$$
$$H(P,Q) = -\int_{-\infty}^{\infty} P(x)\log(Q(x))\,dx\,.$$

Kullback–Leibler Divergence. Also called *relative entropy.*

For discrete distributions:

$$D_{\mathrm{KL}}(P \mid\mid Q) = \sum_{x\in X} P(x)\log\left(\frac{P(x)}{Q(x)}\right)\,.$$

For continuous distributions:

$$D_{\mathrm{KL}}(P \mid\mid Q) = \int_{-\infty}^{\infty} P(x)\log\left(\frac{P(x)}{Q(x)}\right)\,dx\,.$$

Relation to entropy and cross-entropy:

$$D_{\mathrm{KL}}(P \mid\mid Q) = \mathbb{E}_P(\log(P) - \log(Q)) = H(P,Q) - H(P)\,.$$

KL-divergence of normal distributions:

$$D_{\mathrm{KL}}(N(\mu_1,\sigma_1 \mid\mid N(\mu_2,\sigma_2)) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{(\mu_1-\mu_2)^2+\sigma_1^2-\sigma_2^2}{2\sigma_2^2}\,.$$

Section based on the following papers:

- Welling et al. (2011) Bayesian Learning via Stochastic Gradient Langevin Dynamics.

- Chakraborty et al. (2021), Augmenting saliency maps with uncertainty.

- Hu et al. (2019), Supervised Uncertainty Quantification for Segmentation with Multiple Annotations.

Some additional material has been added for clarity.

Bayesian Models. Consider a supervised learning problem such as image classification. We assume that both the data and the model parameters have uncertainty, so rather than taking fix values they will be treated as probabilistic distributions. The model predictions will be probabilistic too, with the following distribution:

$$p(y_{test}|x_{test},\mathcal{D}) = \int_\theta p(y_{test}|x_{test},\theta)p(\theta|\mathcal{D})\,d\theta = \mathbb{E}_{p(\theta|\mathcal{D})}[p(y_{test}|x_{test},\theta)]\,,$$

where x_{test} is the test image, y_{test} is the class label of the test image, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ is the set of training data containing labeled images, and θ are the parameters of the model (weights and biases of the neural network). Here $p(y_{test}|x_{test},\theta)$ means the probability that the model with parameters θ yields output y_i for sample x_i. In order to simplify notation we may represent the pair (x_i, y_i) as just x_i, $\mathcal{D} = \{x_i\}_{i=1}^N$, and write $p(y_i|x_i,\mathcal{D})$ and $p(y_i|x_i,\theta)$ as $p(x_i|\mathcal{D})$ and $p(x_i|\theta)$ respectively. If we want to refer to the probability of an arbitrary output y^* given an input x^* without (x^*, y^*) being an element of the dataset $\mathcal{D}$, then we will revert to the original notation $p(y^*|x^*,\mathcal{D})$, $p(y^*|x^*,\theta)$.

The posterior $p(\theta|\mathcal{D})$ can be seen as the result of updating the *prior distribution* $p(\theta)$ according to Bayes rule using the *likelihood* $p(\mathcal{D}|\theta)$ and the *evidence* $p(\mathcal{D})$:

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}\,,$$

where $p(\theta)$ is a prior on θ, and $p(\mathcal{D}|\theta) = \Pi_{i=1}^N p(x_i|\theta)$. A problem is that finding $p(D)$ requires to integrate respect to all network parametrizations:

$$p(\mathcal{D}) = \int_\theta p(\mathcal{D}|\theta)p(\theta)\,d\theta\,,$$

which is intractable. So $p(\theta|\mathcal{D})$ is typically approximated either by sampling, or by using a variational method. In the next two subsecions we address each of these two approaches.

Sampling Method: Stochastic Gradient Langevin Dynamics. A technique to obtain a Maximum A-Posteriori point (MAP) estimate of the model parameters consists of using Stochastic Gradient Descent (SGD). A typical numerical approach for approximate Bayesian learning is Markov Chain Monte Carlo (MCMC) sampling, where the evidence is treated as constant and the posterior distribution of the parameters is estimated by sampling from the unnormalized distribution $p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$. In particular, the goal is to maximize a Maximum A-Posteriori point (MAP) estimate of the model parameters by maximizing the following function:

$$\log(p(\mathcal{D}|\theta)p(\theta)) = \sum_{i=1}^{N} \log(p(x_i|\theta)p(\theta)) = \log p(\theta) + \sum_{i=1}^{N} \log p(x_i|\theta)\,.$$

Stochastic Gradient Descent leads to the following parameter update step:[1]

$$\theta_{k+1} = \theta_k + \frac{\epsilon_k}{2}\Big(\nabla_\theta \log p(\theta_k) + \frac{N}{n}\sum_{i=1}^{n} \nabla_\theta \log p(x_{k,i}|\theta_k)\Big)\,,$$

where N is the total number of training samples, n is the minibacth size, and ϵ_k is the step-size of the gradient descent. Robbins and Monro showed that a way to ensure convergence to a local maximizer is to require the steps to verify the conditions: $\sum_{i=1}^{\infty} \epsilon_k = \infty$ and $\sum_{i=1}^{\infty} \epsilon_k^2 < \infty$, e.g. $\epsilon_k = a/k$ for some positive constant a. Intuitively, the first constraint ensures that the parameters will reach the high probability regions no matter how far away they were initialized to, while the second ensures that the parameters will converge to the mode instead of just bouncing around it.

However, this reduces the expectation to be computed over a single MAP estimate of the parameters, which does not allow for quantification of uncertainty. An alternative is to use Stochastic Gradient Langevin Dynamics (SGLD) to sample from the posterior. In this approach, while taking the gradient steps associated with the traditional stochastic gradient optimization for maximizing the posterior probability of the parameters, Gaussian noise is induced into the parameter updates so that the parameters do not collapse into a maximum a-posteriori (MAP) solution. This leads to the following parameter update:

$$\theta_{k+1} = \theta_k + \frac{\epsilon_k}{2}\Big(\nabla_\theta \log p(\theta_k) + \frac{N}{n}\sum_{i=1}^{n} \nabla_\theta \log p(x_{k,i}|\theta_k)\Big) + \eta_k\,,$$

where N is the total number of training samples, n is the minibacth size, $\eta_k \sim \mathcal{N}(0, \epsilon_k)$ is the injected Langevin noise, and ϵ_k is the step-size of the gradient descent, verifying the conditions $\sum_{i=1}^{\infty} \epsilon_k = \infty$ and $\sum_{i=1}^{\infty} \epsilon_k^2 < \infty$.[2] In the initial phase, the stochastic gradient noise dominates the added Langevin noise and SGLD imitates standard

[1]In order for this algorithm to be practical the prior probability $p(\theta)$ must be differentiable with respect to θ, so the gradients $\nabla_\theta \log p(\theta)$ can be computed.

[2]These conditions may be relaxed by using a sequence $\epsilon_k \to \epsilon_0 > 0$. In this case the SGLD algorithm must be complemented with an accept-reject step similar to the Metropolis–Hastings algorithm.

stochastic gradient optimization. In the later phase, also called as Langevin phase, the injected Langevin noise dominates the stochastic gradient noise and SGLD imitates Langevin dynamics. This process generates approximate samples from the posterior distribution of θ after the Langevin phase has been reached, when the variances from the injected Gaussian noise and stochastic gradient computation get balanced.

Variational Method: Variational Inference. Another popular method is *variational inference*, consisting of approximating $p(\theta|\mathcal{D})$ with some $q_\lambda(\theta)$ from a tractable family of distributions, where λ is called the *variational parameters*—e.g., q_λ could be a Gaussian $\mathcal{N}(\mu, \sigma^2)$ with parameters $\lambda = (\mu, \sigma)$. The approximation is typically fitted by minimizing the reverse KL-divergence

$$D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta|\mathcal{D})) = \mathbb{E}_{q_\lambda(\theta)}(\log(q_\lambda(\theta)) - \log(p(\theta|\mathcal{D}))) \,,$$

which measures how close $q_\lambda(\theta)$ is to the true posterior $p(\theta|\mathcal{D})$.

This is still intractable because it contains the posterior term, but it can be rearranged as follows. First note that after using Bayes we get:

$$\begin{aligned} D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta|\mathcal{D})) &= \mathbb{E}_{q_\lambda(\theta)}(\log(q_\lambda(\theta)) - \log(p(\mathcal{D}|\theta)) - \log(p(\theta)) + \log p(\mathcal{D})) \\ &= D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta)) - \mathbb{E}_{q_\lambda(\theta)}(\log(p(\mathcal{D}|\theta))) + \log(p(\mathcal{D})) \end{aligned}$$

where $p(\theta)$ is a prior on θ, and $p(\mathcal{D}|\theta) = \Pi_{i=1}^N p(y_i|x_i, \theta)$. Since the term $\log(p(\mathcal{D}))$ does not depend on θ, minimizing $D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta|\mathcal{D}))$ is the same as maximizing the *evidence lower-bound* (ELBO) $= -\mathcal{L}_\lambda(\mathcal{D}|\theta)$, where:

$$\boxed{\mathcal{L}_\lambda(\mathcal{D}|\theta) = \underbrace{-\mathbb{E}_{q_\lambda(\theta)}[\log(p(\mathcal{D}|\theta))]}_{\text{likelihood cost}} + \underbrace{D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta))}_{\text{complexity cost}}} \,,$$

that is:

$$\boxed{\min_\lambda D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta|\mathcal{D})) = \max_\lambda -\mathcal{L}_\lambda(\mathcal{D}|\theta)} \,.$$

The name *evidence lower-bound*, or ELBO, comes from the fact that

$$\log(\underbrace{p(\mathcal{D})}_{\text{evidence}}) = -\mathcal{L}_\lambda(\mathcal{D}|\theta) + \underbrace{D_{\mathrm{KL}}(q_\lambda(\theta) \mid\mid p(\theta|\mathcal{D}))}_{\geq 0} \geq -\underbrace{\mathcal{L}_\lambda(\mathcal{D}|\theta)}_{\text{ELBO}}$$

At training time SGD can be used with $\mathcal{L}_\lambda(\mathcal{D}|\theta)$ as loss function. During the forward pass θ can be sampled with probability $p(\theta|\mathcal{D})$ using MC, and for backpropagation the gradients of the loss function with respect to $\lambda = (\mu, \sigma)$ are used (reparametrization trick).

Aleatoric and Epistemic Uncertainty. The *predictive uncertainty* is the posterior of the predictive distribution. It can be decomposed into two parts. By the law of total variance, we can write predictive variances as a sum of these two independent components:

$$\underbrace{\mathbb{V}_{p(y|\theta,x)}[y]}_{\text{predictive uncertainty}} = \underbrace{\mathbb{E}_{q_\lambda(\theta)}[\mathbb{V}_{p(y|\theta,x)}[y]]}_{\text{aleatoric uncertainty}} + \underbrace{\mathbb{V}_{q_\lambda(\theta)}[\mathbb{E}_{p(y|\theta,x)}[y]]}_{\text{epistemic uncertainty}},$$

The aleatoric term measures the average of the output variance $\mathbb{V}_{p(y|\theta,x)}[y]$, and depends on the data. The epistemic term measures fluctuations in the mean prediction. These fluctuations exist because of uncertainty in the approximate posterior $q_\lambda(\theta)$, which depends on the model.

Dropout Approach to Uncertainty Estimation

(***** This section is under construction *****)