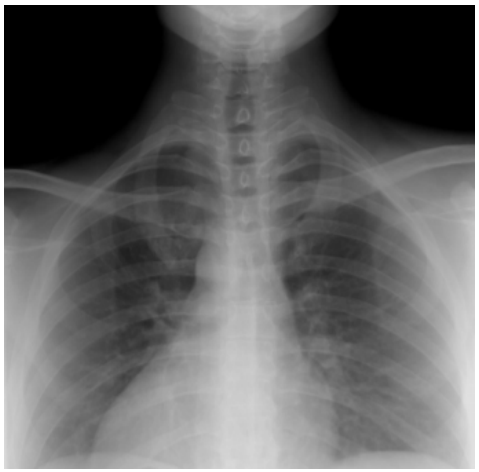
Functional Space Variational Inference for Uncertainty Estimation in Computer Aided Diagnosis

Pranav Poduval, IIT Bombay

Medical Imaging and Deep Learning (MIDL) 2020, Canada
Co-authors: Hrushikesh Loya, Amit Sethi

Motivation for Bayesian Inference



Deep learning is starting to show promise in multiple domains e.g. radiology, cancer detection etc. But we still have to solve a number of issues -

- Does it make sense to pass on obscure values like 0.1 positive chance to Doctors in order to take medical decisions?
- What is to be done, when the model see's something it has never seen before?

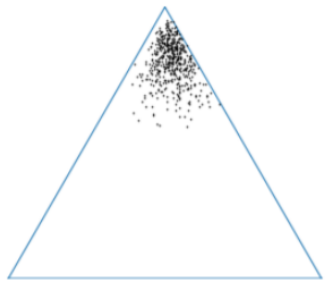
Bayesian Inference is the tool used to **"know what we don't know"**

Bayesian Inference in Neural Networks

The object of interest when we are given an input data point (x^*, y^*) is $q(y^*|x^*)$ which is obtained by marginalizing out the parameter θ i.e. The output distribution for test point (x^*, y^*) is give as -

$$q(y^*|x^*, D) = \int q(y^*|x^*, \theta)p(\theta|D)d\theta \quad (1)$$

Now since exact integration computation is impossible in case of Neural Networks we often use Monte-Carlo sampling to approximate the same -



$$q(y^*|x^*, D) \approx \frac{1}{N}\sum_{i=1}^{N} q(y^*|x^*, \theta^i), \theta^i \sim p(\theta|D) \quad (2)$$

So as seen from the figure we can view them as ensembles of similar networks with N different parameters.

Variational Inference

Exact Bayesian Inference involves computing the true posterior $p(\theta|D)$ according to Bayes rule after defining a prior $p(\theta)$ on the weight space

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}, \textbf{ where } p(D) = \int p(D|\theta)p(\theta)d\theta \tag{3}$$

Since intractable we use a trick by defining a surrogate posterior $q_\phi(\theta)$, (which could be Gaussian and in this case $\phi = (\mu, \Sigma)$) and try to bring this surrogate posterior as close to the true posterior as possible. Therefore we define ELBO loss as -

$$\mathrm{KL}[q_\phi(\theta)||p(\theta|D)] = -\mathbb{E}_{\theta\sim q(\theta)}[\log(p(D|\theta))] + \mathrm{KL}[q_\phi(\theta)||p(\theta)] + \mathbf{C} \tag{4}$$

The first term can be viewed as **Expected Cross Entropy** in case of classification or **Expected MSE** in case of regression, and the the second term as a **Regularizer**.

Priors and what meaning do they have?

For classification among K classes, deep neural networks represent a function $f_\theta : X \to \mathbf{p} \in [0,1]^K$, where X represents the input, and $\mathbf{p}$ represents a probability mass function such that $\sum_{i=1}^{K} p_i = 1$. For making predictions we assume the output distribution is - $p(Y|X,\theta) = \text{Cat}(Y|\mathbf{p})$, where $\mathbf{p} = f_\theta(X)$ i.e. the softmax output of the Neural Network.

Priors and what meaning do they have?

Clearly there exists a map $\theta \rightarrow f(.)$, meaning a prior on θ implicitly defines a prior measure on the space of f, denoted as $p(f)$. We therefore skip steps and directly define a uniform prior on the K-dimensional unit simplex for the functional space, such that

$$p(f) = \text{Dir}(\mathbf{p}|\langle 1, \ldots, 1 \rangle) \tag{5}$$

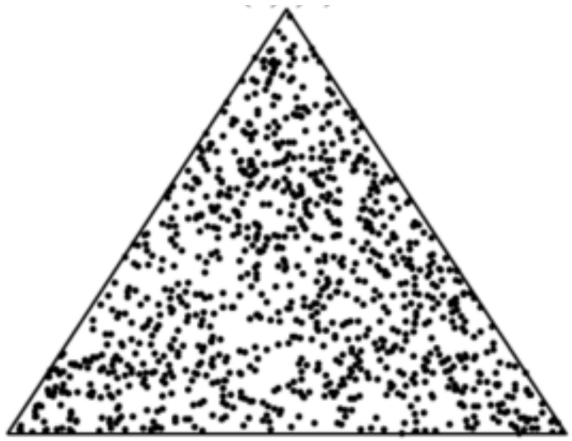


Figure: Ideal Prior for making OOD samples uncertain

A completely uncertain prior. This indicates regardless of the input we are always uncertain of the output.

Functional Space Variational Inference

For analytical tractability we assume the marginal posterior is also a Dirichlet distribution. In other words, unlike for a standard neural network where $\mathbf{p} = f_\theta(x)$ is the point estimate output, in our case $\text{Dir}(\mathbf{p}|\alpha) = q_\theta(f(x))$ is the marginal functional distribution. This is similar to how a Gaussian process has a multivariate Gaussian as its marginal distribution.

Figure: Fig (a) (left) A case where the Functional VI model is very confident of it belonging to all three classes whereas Fig (b) (right) Is the case where a regular Bayesian NN model (e.g.Dropout, Ensemble etc.) is confident of it belonging to a particular class

Functional Space Variational Inference

So in our model given the training data $D = (X^D, y^D)$ and the test points (x^*, y^*) we have:

$$p(y^*|x^*, D) = \int p(y^*|\mathbf{p})\, p(\mathbf{p}|x^*, D)\, d\mathbf{p} \tag{6}$$

As usual $p(y^*|\mathbf{p}) = \text{Cat}(y^*|\mathbf{p})$, but the difference lies in the fact the neural network estimates a Dirichlet distribution $p(\mathbf{p}|x^*, D) = \text{Dir}(\mathbf{p}|\alpha)$.
For standard neural network where $\mathbf{p} = f_\theta(x)$ is the point estimate output, in our case $\text{Dir}(\mathbf{p}|\alpha) = q_\theta(f(x))$ is the marginal functional distribution.
The true functional posterior $p(f|D)$ is intractable, but it can be approximated by minimizing the functional evidence lower bound (fELBO):

$$\mathcal{L}(q) = -\mathbb{E}_{q(f)}[\log p(y^D|f(X^D))] + \text{KL}[q(f)||p(f)] \tag{7}$$

Functional Space Variational Inference

The second term in Equation 7 is the functional KL divergence, which is hard to estimate. Therefore, we shift to a more familiar metric, the KL divergence between the marginal distributions of function values at finite sets of points $\mathbf{x}_{1:n}$:

$$\mathcal{L}_2 = \mathrm{KL}(q(f)||p(f)) = \sup_{\mathbf{x}_{1:n}} \mathrm{KL}\left[q(f(\mathbf{x}_{1:n})||p(f(\mathbf{x}_{1:n})\right] \tag{8}$$

A more relaxed way of sampling these "measure points" $\mathbf{x}_{1:n}$, is to assume $\mathbf{x}_{1:k} \sim X^D$ (training distribution) and $\mathbf{x}_{k+1:n} \sim c$ where c is a distribution having the same support as the training distribution, which could be OOD samples, that can be forced to be more uncertain.

Note: the KL divergence between two Dirichlet distributions can be computed in closed form.

Functional Space Variational Inference

We get a closed form solution for the first part in Equation 7 by assuming y to be a one-hot vector as follows:

$$\mathcal{L}_1 = \int \left[\prod_{i=1}^{K} -\log p(y_i|\mathbf{p}) \right] \frac{1}{B(\alpha)} \prod_{i=1}^{K} p_i^{\alpha_i - 1} d\mathbf{p} \tag{9}$$

By assuming $p(y|\mathbf{p}) = \text{Cat}(y|\mathbf{p})$ we have-

$$\mathcal{L}_1 = \int \left[\sum_{i=1}^{K} -\log p_i^{y_i} \right] \frac{1}{B(\alpha)} \prod_{i=1}^{K} p_i^{\alpha_i - 1} d\mathbf{p} = \sum_{i=1}^{K} y_i \left(F(\sum_{j=1}^{K} \alpha_j) - F(\alpha_i) \right) \tag{10}$$

Where $B(\alpha)$ is the Beta distribution and $F(.)$ is the digamma function.

Combining $\mathcal{L}_1 + \mathcal{L}_2$ we will get the same loss function as Evidential Deep Learning (NeurIPS 2018) and has a simple closed form solution.

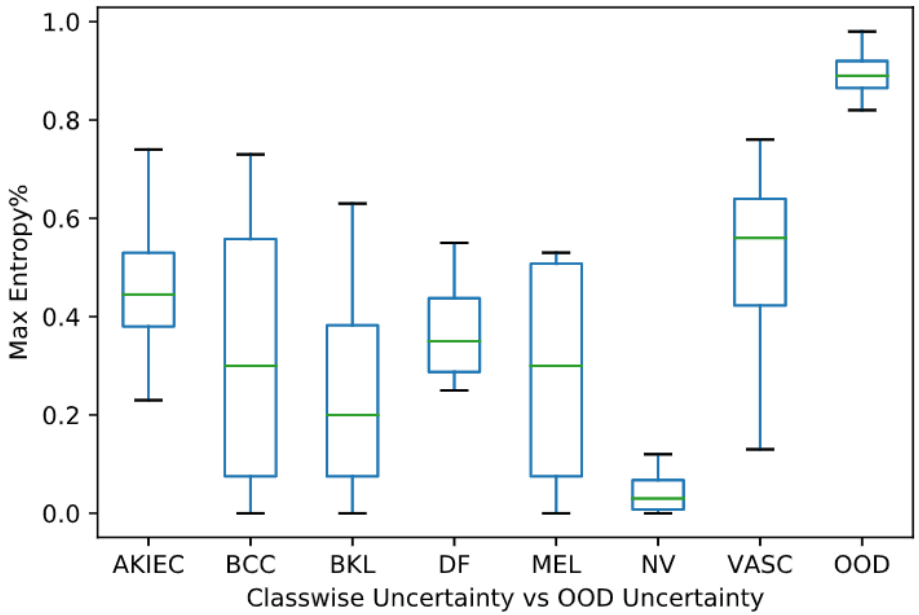
Expected Calibration Error

If we have a well calibrated weather prediction model that predicts sunny event with 80% probability for 100 days then, any deviation from 80 sunny days and 20 non-sunny days will imply a poorly calibrated model.
Important for model interpretability.

Table: Comparison of classification accuracy and ECE on HAM10000 dataset

Method	Standard NN	Dropout	Ensembles	Functional VI
Test Accuracy	84.38%	**86.32%**	85.21%	84.84%
ECE (M = 15)	7.73%	6.39%	3.12%	**1.17%**

Additional Experiment



We observe our model is very confident on Nevi (NV) class, which is expected since it make majority of the dataset. We can also see our OOD samples can be distinctly separated from the in-class samples. The OOD sample used for training and testing are from different distributions. For simplicity we used Gaussian Distribution for training OOD samples and Uniform Distribution for testing OOD samples.