Variational optimization for the minimization of functions on the binary domain

Antti Ajanki

February 14, 2018

We consider the task of minimizing a non-linear scalar function $f(\boldsymbol{x})$, where the input is an N dimensional binary vector, $\boldsymbol{x} \in \{0,1\}^N$.

1 Variational optimization

The minimum of a function is always less than or equal to its expected value over any distribution $p(x|\theta)$:

$$\min f(x) \leq E\left[f(x)\right]_{p(x|\theta)}$$

The bound can be made tight if $p(x|\theta)$ is so flexible that all the probability mass can be concentrated on the true minimum $\text{argmin} f(x)$.

The idea of variational optimization [1] is to consider the upper bound $U(\theta) = E\left[f(x)\right]_{p(x|\theta)}$ as a function of θ and minimize that. This converts the task of minimizing a function $f(x)$ of binary variables to minimization of a function $U(\theta)$ of a continuous variable. Any method for continuous optimization can be applied for find a local minimum of $U(\theta)$.

2 Stochastic gradient descent

Because $\boldsymbol{x}$ is a binary vector, it is natural to choose to take the expectation over a separable Bernoulli distribution:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \prod_i p_i(x_i|\theta_i) = \prod_i \theta_i^{x_i}(1-\theta_i)^{1-x_i}$$

Let's use the stochastic gradient descent to find the local minimum of the upper bound $U(\boldsymbol{\theta})$. First we need the partial derivates:

$$\begin{aligned}
\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} E\left[f(\boldsymbol{x})\right]_{p(\boldsymbol{x}|\boldsymbol{\theta})} \\
&= \int f(\boldsymbol{x}) \frac{\partial}{\partial \theta_j} p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} \\
&= \int f(\boldsymbol{x}) \frac{\partial}{\partial \theta_j} \theta_j^{x_j} (1-\theta_j)^{1-x_j} \prod_{i \neq j} p_i(x_i|\theta_i) d\boldsymbol{x} \\
&= \int f(\boldsymbol{x}) \left(\theta_j^{x_j} (x_j - 1)(1-\theta_j)^{1-x_j-1} + x_j \theta_j^{x_j-1} (1-\theta_j)^{1-x_j} \right) \prod_{i \neq j} p_i(x_i|\theta_i) d\boldsymbol{x} \\
&= \int f(\boldsymbol{x}) \theta_j^{x_j} (1-\theta_j)^{1-x_j} \left(\frac{x_j - 1}{1-\theta_j} + \frac{x_j}{\theta_j} \right) \prod_{i \neq j} p_i(x_i|\theta_i) d\boldsymbol{x} \\
&= \int f(\boldsymbol{x}) \left(\frac{x_j - 1}{1-\theta_j} + \frac{x_j}{\theta_j} \right) \prod_{i} p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} \\
&= E\left[f(\boldsymbol{x}) \left(\frac{x_j - 1}{1-\theta_j} + \frac{x_j}{\theta_j} \right) \right]_{p(\boldsymbol{x}|\boldsymbol{\theta})}
\end{aligned}$$

David Barber proposed approximating the last expectation by sampling [2]:

$$\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_j} \approx \frac{1}{K} \sum_{k=1}^{K} f\left(\boldsymbol{x}^{(k)}\right) \left(\frac{x_j^{(k)} - 1}{1-\theta_j} + \frac{x_j^{(k)}}{\theta_j} \right),$$

where $\boldsymbol{x}^{(1)}$ through $\boldsymbol{x}^{(K)}$ are samples from $p(\boldsymbol{x}|\boldsymbol{\theta})$.

Now that we have a way to approximate the gradient $\nabla U(\boldsymbol{\theta})$, we can apply the stochastic gradient descent to iteratively search for the minimum. The $\boldsymbol{\theta}$ is updated by taking small steps in the direction of the negative gradient:

$$\boldsymbol{\theta}^{\text{new}} = \boldsymbol{\theta} - \frac{\eta}{K} \sum_{k=1}^{K} f\left(\boldsymbol{x}^{(k)}\right) \left(\frac{x_j^{(k)} - 1}{1-\theta_j} + \frac{x_j^{(k)}}{\theta_j} \right),$$

where η is the learning rate. Next, new $\boldsymbol{x}$ samples are drawn from $p(\boldsymbol{x}|\boldsymbol{\theta}^{\text{new}})$ and the iteration is repeated until convergence.

References

[1] Joe Staines, David Barber: *Variational Optimization*, `https://arxiv.org/abs/1212.4507v2`, 2012.

[2] David Barber: *Evolutionary Optimization as a Variational Method*, `https://davidbarber.github.io/blog/2017/04/03/variational-optimisation/`, Apr 3, 2017.