

7.5 Noise Robustness

Section 7.4 has motivated the use of noise applied to the inputs as a dataset augmentation strategy. For some models, the addition of noise with infinitesimal variance at the input of the model is equivalent to imposing a penalty on the norm of the weights (Bishop, 1995a,b). In the general case, it is important to remember that noise injection can be much more powerful than simply shrinking the parameters, especially when the noise is added to the hidden units. Noise applied to the hidden units is such an important topic that it merit its own separate discussion; the dropout algorithm described in section 7.12 is the main development of that approach.

Another way that noise has been used in the service of regularizing models is by adding it to the weights. This technique has been used primarily in the context of recurrent neural networks (Jim et al., 1996; Graves, 2011). This can be interpreted as a stochastic implementation of Bayesian inference over the weights. The Bayesian treatment of learning would consider the model weights to be uncertain and representable via a probability distribution that reflects this uncertainty. Adding noise to the weights is a practical, stochastic way to reflect this uncertainty.

Noise applied to the weights can also be interpreted as equivalent (under some assumptions) to a more traditional form of regularization, encouraging stability of the function to be learned. Consider the regression setting, where we wish to train a function $\hat{y}(\mathbf{x})$ that maps a set of features \mathbf{x} to a scalar using the least-squares cost function between the model predictions $\hat{y}(\mathbf{x})$ and the true values y :

$$J = \mathbb{E}_{p(\mathbf{x}, y)} [(\hat{y}(\mathbf{x}) - y)^2] . \quad (7.30)$$

The training set consists of m labeled examples $\{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(m)}, y^{(m)})\}$.

We now assume that with each input presentation we also include a random perturbation $\epsilon_{\mathbf{W}} \sim \mathcal{N}(\epsilon; \mathbf{0}, \eta \mathbf{I})$ of the network weights. Let us imagine that we have a standard l -layer MLP. We denote the perturbed model as $\hat{y}_{\epsilon_{\mathbf{W}}}(\mathbf{x})$. Despite the injection of noise, we are still interested in minimizing the squared error of the output of the network. The objective function thus becomes:

$$\tilde{J}_{\mathbf{W}} = \mathbb{E}_{p(\mathbf{x}, y, \epsilon_{\mathbf{W}})} [(\hat{y}_{\epsilon_{\mathbf{W}}}(\mathbf{x}) - y)^2] \quad (7.31)$$

$$= \mathbb{E}_{p(\mathbf{x}, y, \epsilon_{\mathbf{W}})} [\hat{y}_{\epsilon_{\mathbf{W}}}^2(\mathbf{x}) - 2y\hat{y}_{\epsilon_{\mathbf{W}}}(\mathbf{x}) + y^2] . \quad (7.32)$$

For small η , the minimization of J with added weight noise (with covariance $\eta \mathbf{I}$) is equivalent to minimization of J with an additional regularization term: