Regularization Properties of Dropout

Anna Shalova [1] Mark Peletier [1] André Schlichting [2]

[1]Eindhoven University of Technology

[2]Institut for Analysis and Numerics, WWU Münster

The Mathematics of Machine Learning,
Pisa, January 2023

Example: Student-teacher setup [1]

Teacher: a small (e.g. $m = 3$) shallow ReLU neural network:

$$f_t(\tilde{w}, x) = \frac{1}{m} \sum_{i=1}^{m} (\tilde{w}_i^T x)_+.$$

Student: a wide ($n \gg m$) shallow ReLU neural network:

$$f(w, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (w_i^T x)_+.$$

Goal: train student network to reconstruct the teacher network

$$\min_w \mathcal{L}(w)$$

where

$$\mathcal{L}(w) = \|f_t(\tilde{w}, x) - f(w, x)\|^2_{L_2(\mu_X)}$$

Example: Student-teacher setup [1]

Let $x, w_i \in \mathbb{R}^2$ and use (sub)gradient descent to find a (local) minima

$$w(k+1) = w(k) - \alpha \nabla_w \mathcal{L}(w(k))$$

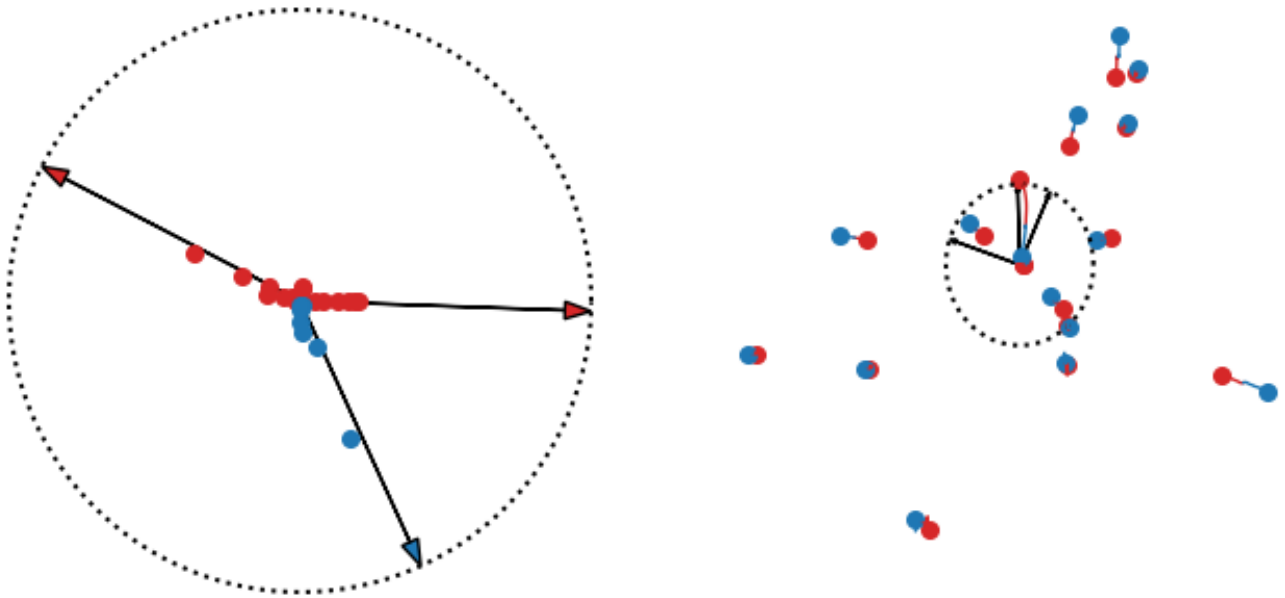


Figure: Dots are weights $\{w_i\}$ of the student network after convergence for $\beta(n) = 1$ (left) and $\beta(n) = \sqrt{n}$ (right). Arrows are the weights of the teacher network $\{\tilde{w}_i\}$

Bernoulli Dropout [2, 3]

Consider a shallow neural network with dropout:

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

Bernoulli Dropout [2, 3]

Consider a shallow neural network with dropout:

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

and the loss:

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

Bernoulli Dropout [2, 3]

Consider a shallow neural network with dropout:

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

and the loss:

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

where

$$\eta_i(k) = \begin{cases} -1, \text{ w.p. } p \\ \frac{p}{1-p} \text{ w.p. } 1-p \end{cases}$$

Bernoulli Dropout [2, 3]

Consider a shallow neural network with dropout:

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

and the loss:

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

where

$$\eta_i(k) = \begin{cases} -1, \text{ w.p. } p \\ \frac{p}{1-p} \text{ w.p. } 1-p \end{cases}$$

$$w(k+1) = w(k) - \alpha \nabla_w \mathcal{L}_d(w(k), \eta(k))$$

Bernoulli Dropout [2, 3]

Consider a shallow neural network with dropout:

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

and the loss:

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

where

$$\eta_i(k) = \begin{cases} -1, & \text{w.p. } p \\ \frac{p}{1-p} & \text{w.p. } 1-p \end{cases}$$

$$w(k+1) = w(k) - \alpha \nabla_w \mathcal{L}_d(w(k), \eta(k))$$

Note:

$$\mathbb{E}_\eta f_d(w, \eta, x) = f(w, x)$$

Bernoulli Dropout: Regularization Effect

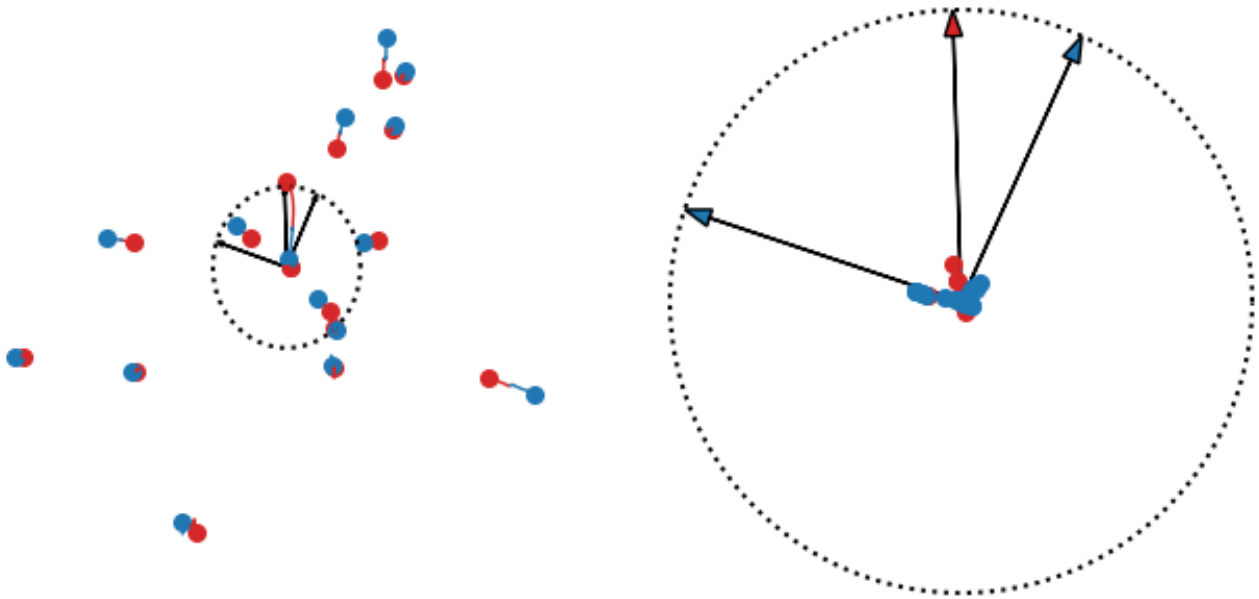


Figure: Dots are weights $\{w_i\}$ of the student network after convergence for $\beta(n) = \sqrt{n}$ without dropout (left) and with dropout (right). Arrows are the weights of the teacher network $\{\tilde{w}_i\}$

Ornstein–Uhlenbeck Dropout

For the same shallow neural network

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

Ornstein–Uhlenbeck Dropout

For the same shallow neural network

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

we introduce a continuous analog of the dropout:

Bernoulli DO:

$$w^{k+1} = w^k - \alpha \nabla_w \mathcal{L}_d(w^k, \eta^k)$$

Ornstein–Uhlenbeck DO:

$$\dot{w} = -\alpha \nabla_w \mathcal{L}_d(w, \eta_t)$$

Ornstein–Uhlenbeck Dropout

For the same shallow neural network

$$f_d(w, \eta, x) = \frac{\beta(n)}{n} \sum_{i=1}^{n} (1 + \eta_i)(w_i^T x)_+,$$

$$\mathcal{L}_d(w, \eta) = \|f_t(\tilde{w}, x) - f_d(w, \eta, x)\|^2_{L_2(\mu_X)}$$

we introduce a continuous analog of the dropout:

Bernoulli DO:

$$\eta_i^k = \begin{cases} -1, & \text{w.p. } p \\ \frac{p}{1-p}, & \text{w.p. } 1-p \end{cases}$$

$$w^{k+1} = w^k - \alpha \nabla_w \mathcal{L}_d(w^k, \eta^k)$$

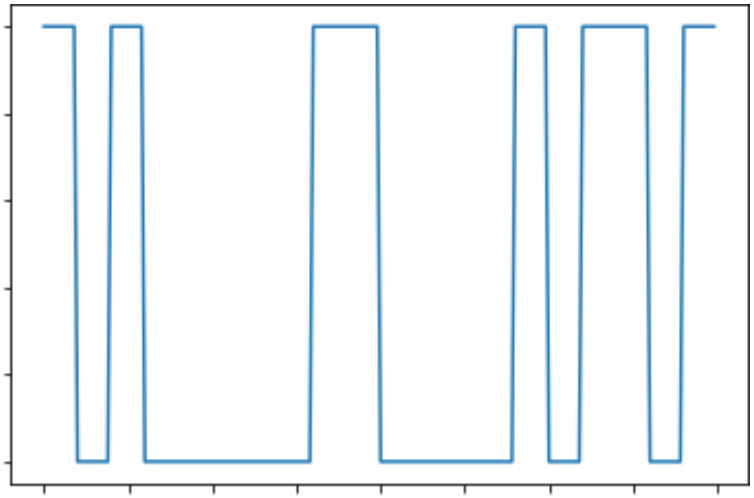
Ornstein–Uhlenbeck DO:

$$d\eta_t = -\gamma \eta_t dt + \sqrt{2\gamma\sigma^2} db_t$$

$$\dot{w} = -\alpha \nabla_w \mathcal{L}_d(w, \eta_t)$$

Ornstein–Uhlenbeck Dropout

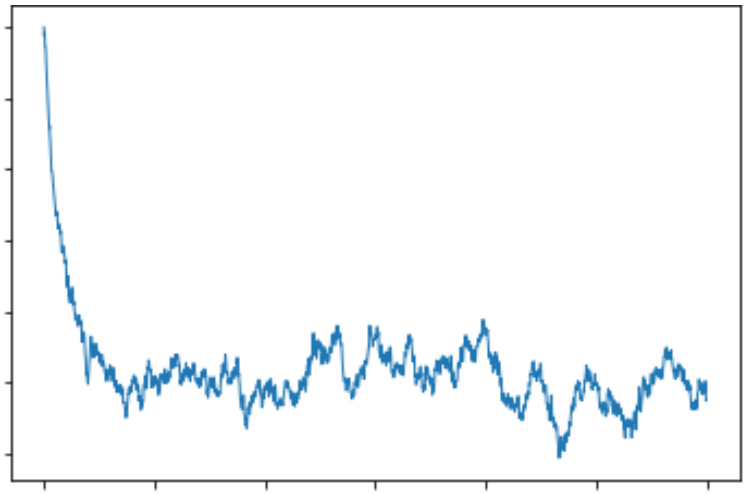
Bernoulli DO:



$$\mathbb{E}\eta^k = 0$$

$$cov(\eta^k, \eta^l) = \frac{p}{p-1}\delta_{kl}$$

Ornstein–Uhlenbeck DO:



$$\mathbb{E}\eta_t = \eta_0 e^{-\gamma t}$$

$$cov(\eta_t, \eta_s) = \sigma^2 e^{-\gamma|t-s|}$$

Our Approach (a teaser)

- We assume that minimizers of $\mathcal{L}(w)$ form a nice manifold
- We use Katzenberger's results [4] to study the training dynamic in the small noise limit $\sigma \to 0$
- We focus on a special case of uncorrelated noise $\lim_{\gamma\to\infty} \lim_{\sigma\to 0}$

References

Lenaic Chizat, Edouard Oyallon, and Francis Bach.
On lazy training in differentiable programming.
Advances in Neural Information Processing Systems, 32, 2019.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov.
Improving neural networks by preventing co-adaptation of feature detectors.
arXiv preprint arXiv:1207.0580, 2012.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.
Dropout: a simple way to prevent neural networks from overfitting.
The journal of machine learning research, 15(1):1929–1958, 2014.

Gary Shon Katzenberger.
Solutions of a stochastic differential equation forced onto a manifold by a large drift.
The University of Wisconsin-Madison, 1990.