

PROBABILISTIC MACHINE LEARNING

LECTURE 18

USES FOR UNCERTAINTY IN DEEP LEARNING

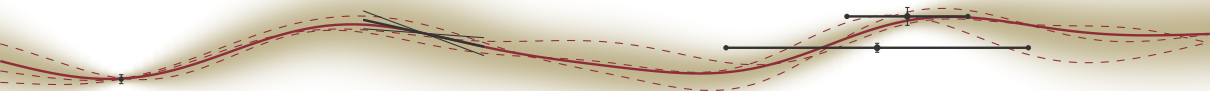
Philipp Hennig

3 July 2023

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN



FACULTY OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
CHAIR FOR THE METHODS OF MACHINE LEARNING



Recap: Deep networks *are* GPs

Linearized networks and Laplace approximations: From deep learning to GPs, in four easy steps

1. Realise that the loss is a **negative log-posterior**

$$\mathcal{L}(\boldsymbol{\theta}) = \left(\underbrace{\frac{1}{N} \sum_{i=1}^N \ell(y_i; \overbrace{f(x_i, \boldsymbol{\theta})}^{\text{deep net}})}_{\text{empirical risk}} + \underbrace{r(\boldsymbol{\theta})}_{\text{regularizer}} \right) = - \sum_{i=1}^N \log p(y | \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) = - \log p(\boldsymbol{\theta} | y) + \text{const.}$$

2. Train the deep net *as usual* to find $\boldsymbol{\theta}_* = \arg \max_{\boldsymbol{\theta} \in \mathbb{R}^D} p(\boldsymbol{\theta} | y)$
3. At $\boldsymbol{\theta}_*$, compute a **Laplace approximation** of the log-posterior, with $\Psi := -\nabla \nabla^\top \log p(\boldsymbol{\theta}_* | y)$

$$\log p(\boldsymbol{\theta} | y) + \text{const.} = \mathcal{L}(\boldsymbol{\theta}) \approx \mathcal{L}(\boldsymbol{\theta}_*) + 1/2 (\boldsymbol{\theta} - \boldsymbol{\theta}_*)^\top \Psi (\boldsymbol{\theta} - \boldsymbol{\theta}_*) = \log \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}_*, -\Psi^{-1})$$
4. Linearize $f(x, \boldsymbol{\theta})$ around $\boldsymbol{\theta}_*$, with $[J(x)]_{ij} = \frac{\partial f_i(x, \boldsymbol{\theta}_*)}{\partial \theta_j}$ as $f(x, \boldsymbol{\theta}) \approx f(x, \boldsymbol{\theta}_*) + J(x, \boldsymbol{\theta}_*)(\boldsymbol{\theta} - \boldsymbol{\theta}_*)$

$$\text{thus } p(f(\bullet) | \mathcal{D}) = \int p(f | w) dp(w) \approx \mathcal{GP}(f(\bullet); f(\bullet, \boldsymbol{\theta}_*), -J(\bullet)\Psi^{-1}J(\circ)) \quad \text{with}$$

$$\mathbb{E}(f(\bullet)) = f(\bullet, \boldsymbol{\theta}_*) \quad \text{the trained net as the mean function}$$

$$\text{cov}(f(\bullet), f(\circ)) = -J(\bullet)\Psi^{-1}J(\circ)^\top \quad \text{the Laplace tangent kernel as the covariance function}$$

Computing the exact Hessian Ψ is $\mathcal{O}(ND^2)$, and inverting it is $\mathcal{O}(D^3)$.

Ideas for Approximations:

- ▶ Sub-sample the dataset ($\mathcal{O}(MD^2)$ with $M \ll N$)
- ▶ structural approximations to the Hessian:
 - ▶ diagonal approximation: $\mathcal{O}(D)$ (inverse $\mathcal{O}(D)$)
 - ▶ last-layer approximation: $\mathcal{O}(D_L^2)$ (inverse $\mathcal{O}(D_L^3)$)
 - ▶ Kronecker factorized approximate curvature (KFAC): $\Psi \approx \text{diag}([\Lambda_\ell \otimes \Omega_\ell]_{\ell=1,\dots,L})$
with $\Lambda_\ell \in \mathbb{R}^{\text{in}_\ell \times \text{in}_\ell}$, $\Omega_\ell \in \mathbb{R}^{\text{out}_{\ell-1} \times \text{out}_{\ell-1}}$ and thus inverse $\mathcal{O}\left(\sum_\ell \text{in}_\ell^3 + \text{out}_{\ell-1}^3\right)$
 - ▶ Generalized Gauss-Newton (homework this week): $\Psi \approx \alpha I + GG^\top$ with $G \in \mathbb{R}^{D \times M}$
 - ▶ approximate eigenvalue decompositions using the Lanczos algorithm (cf. Lecture 13)

Uncertainty in Deep Learning

- ▶ fixes (asymptotic and local) overconfidence
- ▶ yields the functionality for continual learning
- ▶ many other applications not discussed here

Laplace approximations turn deep networks into GPs, inheriting all functionality of GPs

Please cite this course, as

```
@techreport{Tuebingen_ProbML23,
  title =
    {Probabilistic Machine Learning},
  author = {Hennig, Philipp},
  series = {Lecture Notes
            in Machine Learning},
  year = {2023},
  institution = {Tübingen AI Center}}
```

