COMP 64101
Reasoning and Learning under Uncertainty

Omar Rivasplata

Department of Computer Science, University of Manchester
Manchester Centre for AI Fundamentals

Lecture 4 - Part 2

Topics: Bayesian Neural Networks

I know it's [Tuesday]. It's a good day for math!

–Inspired by Max Mintz, UPenn.

- Posteriors for BNNs (§ 17.3)

Nota bene: Section numbers refer to Murphy (2023).

17.3 Posteriors for BNNs

- Most important thing to learn:

$$p(\mathbf{w}|\mathcal{D}, \boldsymbol{\theta}) = \frac{p(\mathcal{D}|\mathbf{w}, \boldsymbol{\theta})p(\mathbf{w}|\boldsymbol{\theta})}{p(\mathcal{D}|\boldsymbol{\theta})}$$

- Let's make it easy to remember:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

- Where 'evidence' is the normalization:

$$p(\mathcal{D}|\boldsymbol{\theta}) = \int p(\mathcal{D}|\mathbf{w}, \boldsymbol{\theta})p(\mathbf{w}|\boldsymbol{\theta})d\mathbf{w}$$

17.3 Posteriors for BNNs - case of supervised learning

- Dataset $\mathcal{D}$ is a list of pairs $(\mathbf{x}, y)$, with input features $\mathbf{x}$ and labels y.

- Have discussed the posterior distribution over weights $p(\mathbf{w}|\mathcal{D})$.
  - Sometimes this is written $p(\mathbf{w}|\mathcal{D}, \boldsymbol{\theta})$ to show the hyperparameters $\boldsymbol{\theta}$.

- Interest is on a posterior predictive distribution $p(y|\mathbf{x}, \mathcal{D})$.
  - Sometimes this is written $p(y|\mathbf{x}, \mathcal{D}, \boldsymbol{\theta})$ to show the hyperparameters $\boldsymbol{\theta}$.

17.3.1 Monte Carlo dropout

- **Dropout:** usually during training, turned off after training.

- **Monte Carlo Dropout:** do random sampling after training.

- Drop out hidden units according to a Bernoulli(p) distribution.
- Repeat this N times to create N neural nets, then define

$$p(y|\mathbf{x}, \mathcal{D}) \approx \frac{1}{N} \sum_{n=1}^{N} p(y|\mathbf{x}, \mathbf{W}^{(n)})$$

- Each $\mathbf{W}^{(n)}$ is a MAP estimate, with dropped-out connections.

17.3.5 Last layer methods

- Only “be Bayesian” about the weights in the final layer.
- Use MAP estimates for all the parameters of all other layers.

- Details in Section numbers refer to Murphy (2023), p. 652.

17.3.8 Methods based on the SGD trajectory

- **SWA:** stochastic weight averaging.
  - Average the weights along trajectories: $\bar{\boldsymbol{\theta}} = \frac{1}{S} \sum_{s=1}^{S} \boldsymbol{\theta}_s$
  - Predict according to $p(y|\mathbf{x}, \bar{\boldsymbol{\theta}})$
- **Snapshot ensembles, fast geometric ensembles:**
  - Average predictions along trajectories: $p(y|\mathbf{x}, \mathcal{D}) \approx \frac{1}{S} \sum_{s=1}^{S} p(y|\mathbf{x}, \boldsymbol{\theta}_s)$
- **SWAG:** stochastic weight averaging with Gaussian posterior.

17.3.9 Deep ensembles

- Create a number of models $\mathbf{w}_1, \ldots, \mathbf{w}_M$.
  (e.g. sampling from a distribution over models.)
  (e.g. training multiple models for a given task.)

- Then combine them by simple averaging or weighted averaging.
  (For classification, can use corresponding majority vote rules.)

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{w}_m|\mathcal{D})p(\boldsymbol{\theta}|\mathbf{w}_m, \mathcal{D})$$

- Use weighted combinations for predictions:

$$p(y|\mathbf{x}, \mathcal{D}) = \frac{1}{M}\sum_{m=1}^{M} \alpha_m(\mathbf{x})p(y|\mathbf{x}, \mathbf{w}_m)$$

Deep ensembles vs mixtures of experts and stacking

- This is 17.3.9.5

- **Mixtures:** coefficients $\alpha_m(\mathbf{x}) \geq 0$ that add up to 1.

- **Stacking:** coefficients $\alpha_m(\mathbf{x}) \geq 0$ not required to add up to 1.

17.3.10 Approximating the posterior predictive distribution

- Have approximated the parameter posterior: $q(\boldsymbol{\theta}|\mathcal{D}) \approx p(\boldsymbol{\theta}|\mathcal{D})$.

- Can use it to approximate the posterior predictive distribution:

$$p(y|\mathbf{x}, \mathcal{D}) \approx \int q(\boldsymbol{\theta}|\mathcal{D}) p(y|\mathbf{x}, \boldsymbol{\theta}) d\boldsymbol{\theta}$$

- Can often approximate this integral using Monte Carlo:

$$p(y|\mathbf{x}, \mathcal{D}) \approx \frac{1}{S} \sum_{s=1}^{S} p(y|f(\mathbf{x}, \boldsymbol{\theta}_s))$$

with $\boldsymbol{\theta}_s \sim q(\boldsymbol{\theta}|\mathcal{D})$ and some suitably chosen function f.

References

Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023. URL `http://probml.github.io/book2`.