A Summary of A General Framework for Uncertainty Estimation in Deep Learning

Sharan Yalburgi

July 17, 2019

URL: `https://arxiv.org/pdf/1907.06890.pdf`

1 Aim

Uncertainity Estimation in DNN without changing the neural network. Their framework can be applied to any existing neural network and task, without the need to change the network's architecture orloss, or to train the network.

2 Problem of interest

Steering wheel predictions. We want prediction on postition of steering wheel along with the uncertainity estimation.

3 Basic Idea

Forward propagating the uncertainities in inputs and model(weight) uncertainities.

4 Related Work

4.1 Dropout to Estimate Model Uncertainty

Monte Carlo Dropout : Gal and Ghahramani, 2016 proposed capturing model uncertainity by applying dropout at test time.

Limitations: Assumes σ constant for all inputs. Does not accomodate adverserial inputs.

4.2 Learning Model and Data Uncertainty Together

Kendall and Gal, 2017 proposed to jointly train to predict the model prediction and uncertainity. This technique can accomodate non-uniform label noise.

The model is trained under ***Heteroscedastic Loss***.

Heteroscedastic Loss is defined as :

$$L_{NN}(\theta) =) \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\sigma^2(x_i)} \|y_i - f(x_i)\|^2 + \frac{1}{2} log\sigma^2(x_i) \tag{1}$$

Here, we have input dependent noise $\sigma^2(\mathrm{x_i})$.

Limitations

- This technique requires us to change the existing architecture of the NN, into a "two-head" system.
- Heteroscedastic loss often results in performance drop.

4.3 Data Uncertainty Propagation with Assumed Density Filtering

Gast and Roth, 2018 introduced a lightweight approach to recover **data uncertainty** while maintaining the same network architecture, with minor changes to propagate both mean and variance of the input distribution.

Limitations

Disregards model uncertainity, assuming large amount of data will nullify its effect which is often not true in applications like self driving.

5 Recovering Total Uncertainity

5.1 Fusing MC Dropout and Assumed Density Filtering

The key idea is to train a regular NN and convert it into its ADF counter part. ADF can be seen as a probabilistic model p(y|z, ω)p(z|x), where p(z|x) = N(z;x,σ_n^2) is the input perturbed by white gaussian noise.

We have two kinds of uncertainity here,

- Model Uncertainity: We recover the model uncertainity by collecting stochastic samples from the ADF.
- Data Uncertainity: This would be retrieved directly from the one of the data outputs, aka data variance $\hat{\sigma}^2_{data}$.

We have a proof confirming that the total uncertainity Var(y) $\approx$ Model Uncertainity + Data Uncertainity

TODO What is Assumed Density Filtering(ADF)?

6 L

References

Gal, Yarin and Zoubin Ghahramani (2016). "Dropout as a bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*, pp. 1050–1059 (cit. on p. 1).

Gast, Jochen and Stefan Roth (2018). "Lightweight probabilistic deep networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3369–3378 (cit. on p. 2).

Kendall, Alex and Yarin Gal (2017). "What uncertainties do we need in bayesian deep learning for computer vision?" In: *Advances in neural information processing systems*, pp. 5574–5584 (cit. on p. 1).