Reliable training and estimation of variance networks

Martin Jørgensen [*,1] marjor@dtu.dk, Nicki S. Detlefsen[*,1] nsde@dtu.dk and Søren Hauberg[1] sohau@dtu.dk

[*]Equal contribution [1]Technical University of Denmark

Objective

Predictive *uncertainty* has not experienced the same success as mean prediction, and typically most neural network based models are overconfident. We propose methods and techniques that train *variance networks* knowing that they are different from standard regression networks.

Methods

1. *The locality sampler:* Estimating variance based on few observations have high variance itself. We propose a two-stage sampling scheme to replace standard mini-batching

$$\sum_{i=1}^{N}\left\{-\frac{1}{2}\log(\sigma^2(\mathbf{x}_i))-\frac{(y_i-\mu(\mathbf{x}_i))^2}{2\sigma^2(\mathbf{x}_i)}\right\}\approx\sum_{\mathbf{x}_j\in\mathcal{O}}\frac{1}{\pi_j}\left\{-\frac{1}{2}\log(\sigma^2(\mathbf{x}_j))-\frac{(y_j-\mu(\mathbf{x}_j))^2}{2\sigma^2(\mathbf{x}_j)}\right\}$$

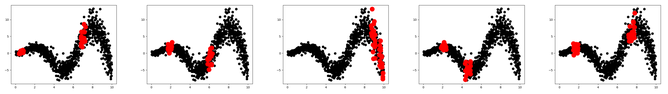

Figure: 5 batches from our proposed locality sampler

2. *Extrapolation:* One of vanilla neural networks caveats is poor extrapolation, in the sense that predictions away from data are arbitrary. Let $\{\mathbf{c}_i\}_{i=1}^{L}$ be points in $\mathbb{R}^D$ that represent the training data, akin to inducing points in sparse GPs. Then define $\delta(\mathbf{x}_0)=\min_i\|\mathbf{c}_i-\mathbf{x}_0\|$ and

$$\hat{\sigma}^2(\mathbf{x}_0)=\big(1-\nu(\delta(\mathbf{x}_0))\big)\hat{\sigma}^2_\theta+\eta\nu(\delta(\mathbf{x}_0)),$$

where $\nu:[0,\infty)\mapsto[0,1]$ is a surjective increasing function.

3. *Mean-variance split:* Ensure that the gradients are always pointing towards the MLE, by sequentially optimizing $p_\theta(y|\theta_\mu)$ and $p_\theta(y|\theta_{\sigma^2})$ separately.
4. *Student-t*-likelihood is known to be more robust in presence of small sample data – we put a Inverse-Gamma prior on σ^2, and instead train parameter functions $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$ to gain more robust estimates.

Ablation study

We individually test each contribution, and find that they increase performance individually and in combination.

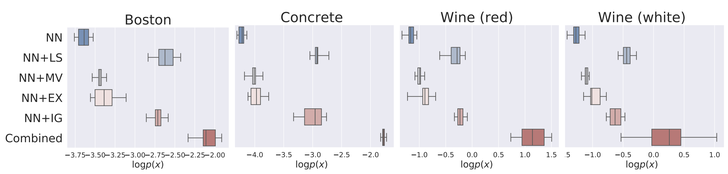

Figure: Small ablation study on four UCI benchmark datasets.

Acknowledgements

This work was supported by a research grant (15334) from VILLUM FONDEN. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement nº 757360).

Generative Models

Variational autoencoders are among state-of-the-art models for generative models. However, these models suffer from poor uncertainty estimation when the decoder is modeled as $p_\theta(\boldsymbol{x}|\boldsymbol{z})=\mathcal{N}\big(\boldsymbol{x}|\mu_\theta(\boldsymbol{z}),\sigma^2_\theta(\boldsymbol{z})\big)$. In the literature, we find that variance networks are avoided by using a Bernoulli distribution, modeled using a constant and often not included in generated samples. In our work we model the decoder variance using our *Combined* method.

Toy data

We consider generative modeling of toy data. We have mapped the two half moons into four dimensions, using mappings that MLP's struggle to learn, thus successful modeling requires the variance function to be meaningful.

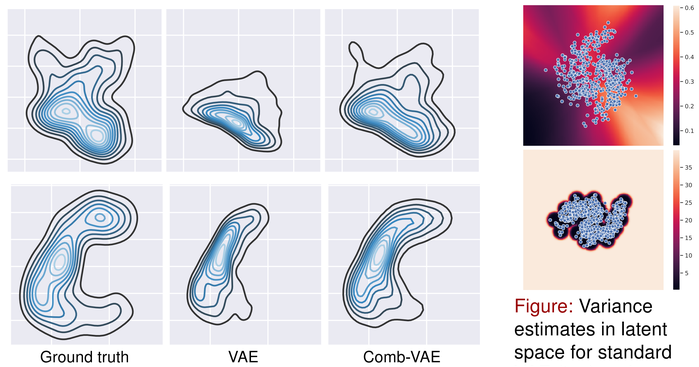

Figure: The ground truth and generated distributions. *Top*: x_1 vs. x_2. *Bottom*: x_2 vs x_3.

Figure: Variance estimates in latent space for standard VAE (top) and our Comb-VAE (bottom).

Image generation

We test our proposed method against standard VAE in image generation on MNIST, FashionMNIST, CIFAR and SVHN. We qualitatively get better ELBO and log-likelihood. Quantitatively we visually get a more expressive decoder variance.

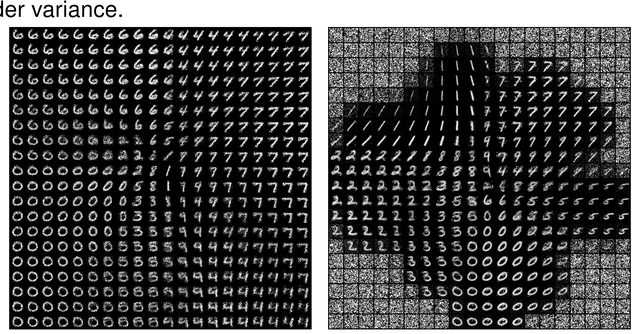

Figure: *Left:* Standard variance estimation are near constant and extrapolated badly. *Right*: Using the improved variance estimator, we are able to get more expressive variance estimated where we have data, and better extrapolation.

		MNIST	FashionMNIST	CIFAR10	SVHN
ELBO	VAE	2053.01 ± 1.60	1506.31 ± 2.71	1980.84 ± 3.32	3696.35 ± 2.94
	Comb-VAE	**2152.31 ± 3.32**	**1621.29 ± 7.23**	**2057.32 ± 8.13**	3701.41 ± 5.84
$\log p(x)$	VAE	1914.77 ± 2.15	1481.38 ± 3.68	1809.43 ± 10.32	3606.28 ± 2.75
	Comb-VAE	**2018.37 ± 4.35**	**1567.23 ± 4.82**	**1891.39 ± 20.21**	3614.39 ± 7.91

Table: Generative modeling of 4 image datasets.

Regression

We consider the problem $y=\boldsymbol{x}\cdot\sin(\boldsymbol{x})+0.3\cdot\epsilon_1+0.3\cdot\boldsymbol{x}\cdot\epsilon_2$, with $\epsilon_1,\epsilon_2\sim\mathcal{N}(0,1)$, and compare us to Gaussian Processes (GP), Neural Networks baseline (NN), Bayesian Neural Networks (BNN), Monte Carlo Dropout (MC-Dropout) and Deep Ensembles (Ens-NN).

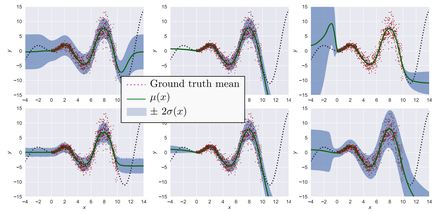

Figure: From top left to bottom right: GP, NN, BNN, MC-Dropout, Ens-NN, Combined.

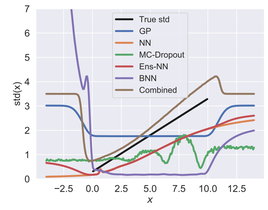

Figure: Standard deviation estimates as a function of $\boldsymbol{x}$.

UCI Benchmark

We outperform current methods, on 9 out of 13 dataset measured by test set log likelihood

	N	D	GP	SGP	NN	BNN	MC-Dropout	Ens-NN	Combined
Boston	506	13	$\mathbf{-1.76\pm0.3}$	-1.85 ± 0.25	-3.64 ± 0.09	-2.59 ± 0.11	-2.51 ± 0.31	-2.45 ± 0.25	-2.09 ± 0.09
Carbon	10721	7	-	3.74 ± 0.53	-2.03 ± 0.14	-1.1 ± 1.76	-1.08 ± 0.05	-0.44 ± 7.28	$\mathbf{4.35\pm0.16}$
Concrete	1030	8	-2.13 ± 0.14	-2.29 ± 0.12	-4.23 ± 0.07	-3.31 ± 0.05	-3.11 ± 0.12	-3.06 ± 0.32	$\mathbf{-1.78\pm0.04}$
Energy	768	8	-1.85 ± 0.34	-2.22 ± 0.15	-3.78 ± 0.04	-2.07 ± 0.08	-2.01 ± 0.11	$\mathbf{-1.48\pm0.31}$	-1.68 ± 0.13
Kin8nm	8192	8	-	2.01 ± 0.02	-0.08 ± 0.02	0.95 ± 0.08	0.95 ± 0.15	1.18 ± 0.03	$\mathbf{2.49\pm0.07}$
Naval	11934	16	-	-	3.47 ± 0.21	3.71 ± 0.05	3.80 ± 0.09	5.55 ± 0.05	$\mathbf{7.27\pm0.13}$
Power plant	9568	4	-	-1.9 ± 0.03	-4.26 ± 0.14	-2.89 ± 0.01	-2.89 ± 0.14	-2.77 ± 0.04	$\mathbf{-1.19\pm0.03}$
Protein	45730	9	-	-	-2.95 ± 0.09	-2.91 ± 0.00	-2.93 ± 0.14	$\mathbf{-2.80\pm0.02}$	$\mathbf{-2.83\pm0.05}$
Superconduct	21263	81	-	-4.07 ± 0.01	-4.92 ± 0.10	-3.06 ± 0.14	-2.91 ± 0.19	-3.01 ± 0.05	$\mathbf{-2.43\pm0.05}$
Wine (red)	1599	11	0.96 ± 0.18	-0.08 ± 0.01	-1.19 ± 0.11	-0.98 ± 0.01	-0.94 ± 0.01	-0.93 ± 0.09	$\mathbf{1.21\pm0.23}$
Wine (white)	4898	11	-	-0.14 ± 0.05	-1.29 ± 0.09	-1.41 ± 0.17	-1.26 ± 0.01	-0.99 ± 0.06	$\mathbf{0.40\pm0.42}$
Yacht	308	7	$\mathbf{0.16\pm1.22}$	-0.38 ± 0.32	-4.12 ± 0.17	-1.65 ± 0.05	-1.55 ± 0.12	-1.18 ± 0.21	$\mathbf{-0.07\pm0.05}$
Year	515345	90	-	-	-5.21 ± 0.87	-3.97 ± 0.34	-3.78 ± 0.01	-3.42 ± 0.02	$\mathbf{-3.01\pm0.14}$

Table: Dataset characteristics and tests set log-likelihoods for the different methods. A "-" indicates the model was infeasible to train. Bold highlights the best results.

Active Learning

We construct a active learning setting based on UCI datasets. On some dataset, we see observe faster learning than other methods. On other datasets, we are on par with existing methods.

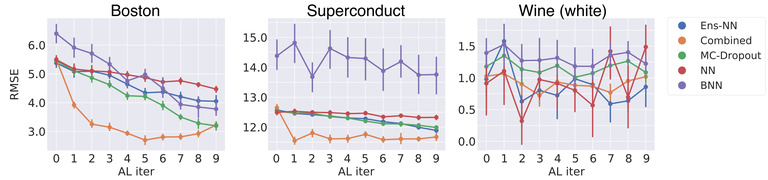

Figure: Average test set RMSE and standard errors in active learning.

Conclusions

- We propose methodologies for better variance estimation in neural nets, that can be used in combination or individually.
- We clearly outperform established ensemble methods in regression modelling and active learning.
- In generative modeling, we better capture the data distribution and get more interpretable variance estimates than baseline methods.