Neural Networks as Sparse Gaussian Processes for Sequential Learning

Aidan Scannell[⋆ 1 2] Riccardo Mereu[⋆ 1] Paul Chang[1] Ella Tamir[1] Joni Pajarinen[1] Arno Solin[1]

[1]Aalto University [2]Finnish Center for Artificial Intelligence

TL;DR

- Neural networks (NNs) have limitations: *estimating uncertainty*, *incorporating new data*, and *avoiding catastrophic forgetting*.
- Our method, Sparse Function-space Representation (SFR):
  1. converts NN to sparse Gaussian process (GP) via dual parameters,
  2. has good uncertainty estimates,
  3. can incorporate new data without retraining,
  4. can maintain a functional representation for continual learning,
  5. can be used for uncertainty-guided exploration in model-based RL.

Method

1. Train Neural Network

(a) Inputs in a *supervised setting* for NNs $f_{\mathbf{w}} : \mathbb{R}^D \to \mathbb{R}^C$:
- $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, a data set w/ input $\mathbf{x}_i \in \mathbb{R}^D$ and output $\mathbf{y}_i \in \mathbb{R}^C$;
- $\mathbf{w} \in \mathbb{R}^P$, the initial weights of the neural network.

(b) **Train:** minimize the empirical (regularized) risk:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathcal{L}(\mathcal{D}, \mathbf{w}) = \arg\min_{\mathbf{w}} \textstyle\sum_{i=1}^N \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i) + \delta\mathcal{R}(\mathbf{w}). \quad (1)$$

(c) **Output:** $\mathbf{w}^*$, the Maximum A-Posteriori (MAP) weights of the NN.

2. Convert Neural Network to Gaussian Process

(a) Linearise NN using Laplace-GGN approx at MAP weights $\mathbf{w}^*$,

$$f_{\mathbf{w}^*}(\mathbf{x}) \approx \mathcal{J}_{\mathbf{w}_*}(\mathbf{x})\,\mathbf{w} \quad \text{where} \quad \mathcal{J}_{\mathbf{w}}(\mathbf{x}) \coloneqq [\nabla_{\mathbf{w}} f_{\mathbf{w}}(\mathbf{x})]^\top \in \mathbb{R}^{C\times P}.$$

(b) Interpret linear model as GP,

$$\mu(\mathbf{x}) = 0 \quad \text{and} \quad \kappa(\mathbf{x}, \mathbf{x}') = \frac{1}{\delta}\mathcal{J}_{\mathbf{w}^*}(\mathbf{x})\,\mathcal{J}_{\mathbf{w}^*}^\top(\mathbf{x}'),$$

(c) Formulate GP predictive posterior via dual parameterization,

$$\mathbb{E}_{p(f_i\,|\,\mathbf{y})}[f_i] = \mathbf{k}_{\mathbf{x}i}^\top \boldsymbol{\alpha} \quad \text{and} \quad (2)$$
$$\mathrm{Var}_{p(f_i\,|\,\mathbf{y})}[f_i] = k_{ii} - \mathbf{k}_{\mathbf{x}i}^\top(\mathbf{K}_{\mathbf{xx}} + \mathrm{diag}(\boldsymbol{\beta})^{-1})^{-1}\mathbf{k}_{\mathbf{x}i} \quad (3)$$

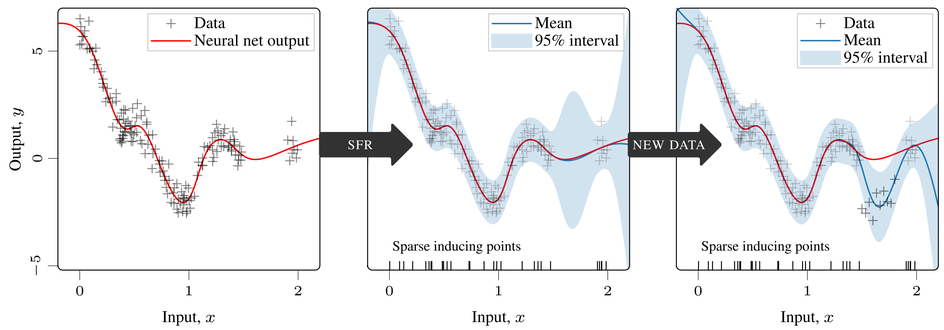
3. Sparsify the Gaussian Process

(a) Eqs. 2/3 use all data $\mathcal{O}(N^3)$ so sparsify using inducing points,

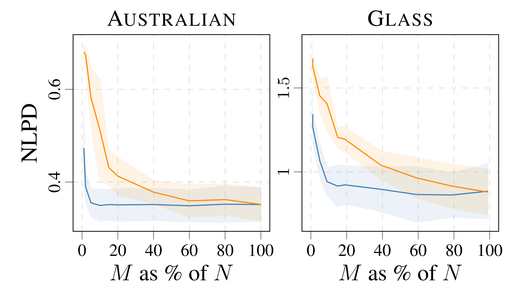
$$\mathbb{E}_{p(f_i\,|\,\mathbf{y})}[f_i] \approx \mathbf{k}_{\mathbf{z}i}^\top \mathbf{K}_{\mathbf{zz}}^{-1} \boldsymbol{\alpha}_{\mathbf{u}} \quad \text{and} \quad (4)$$
$$\mathrm{Var}_{p(f_i\,|\,\mathbf{y})}[f_i] \approx k_{ii} - \mathbf{k}_{\mathbf{z}i}^\top[\mathbf{K}_{\mathbf{zz}}^{-1} - (\mathbf{K}_{\mathbf{zz}} + \boldsymbol{B}_{\mathbf{u}})^{-1}]\mathbf{k}_{\mathbf{z}i} \quad (5)$$

with SFR dual parameters,

$$\boldsymbol{\alpha}_{\mathbf{u}} = \sum_{i=1}^N \mathbf{k}_{\mathbf{z}i}\,\hat{\alpha}_i \quad \text{and} \quad \boldsymbol{B}_{\mathbf{u}} = \sum_{i=1}^N \mathbf{k}_{\mathbf{z}i}\,\hat{\beta}_i\,\mathbf{k}_{\mathbf{z}i}^\top \quad (6)$$



(a) **Regression example on an MLP with two hidden layers.** Left: Predictions from the trained neural network. Middle: Our approach summarizes all the training data with the help of a set of inducing points. The model captures the predictive mean and uncertainty, and (right) makes it possible to incorporate new data without retraining the model.



(b) **SFR (—) requires fewer inducing points** M to converge to good NLPD than GP subset (—) on UCI classification.

Dual Parameters

- Dual parameters are expectations of derivatives,

$$\alpha_i \coloneqq \mathbb{E}_{p(\mathbf{w}\,|\,\mathbf{y})}[\nabla_f \log p(y_i\,|\,f)|_{f=f_i}] \quad \text{and} \quad (7)$$
$$\beta_i \coloneqq -\mathbb{E}_{p(\mathbf{w}\,|\,\mathbf{y})}[\nabla_{ff}^2 \log p(y_i\,|\,f_i)|_{f=f_i}] \quad (8)$$

  and we simply consider the MAP of $p(\mathbf{w}\,|\,\mathbf{y})$,
- Conditioning on new data $\mathcal{D}^{\text{new}}$ with dual parameters is easy,

$$\boldsymbol{\alpha}_{\mathbf{u}} = \boldsymbol{\alpha}_{\mathbf{u}}^{\text{old}} + \underbrace{\sum_{\mathbf{x}_i, y_i \in \mathcal{D}^{\text{new}}} \mathbf{k}_{\mathbf{z}i}\,\hat{\alpha}_i}_{\text{update}} \quad \text{and} \quad \boldsymbol{B}_{\mathbf{u}} = \boldsymbol{B}_{\mathbf{u}}^{\text{old}} + \underbrace{\sum_{\mathbf{x}_i, y_i \in \mathcal{D}^{\text{new}}} \mathbf{k}_{\mathbf{z}i}\,\hat{\beta}_i\,\mathbf{k}_{\mathbf{z}i}^\top}_{\text{update}}$$

Model-based Reinforcement Learning

Strategy Policy $\pi : \mathcal{S} \to \mathcal{A}$ based on posterior sampling:

$$\pi^{\text{PS}} = \arg\max_{\pi\in\Pi} \mathbb{E}_{\boldsymbol{\epsilon}_{0:\infty}}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right] \quad \text{s.t. } \tilde{f} \sim q_{\mathbf{u}}(f\,|\,\mathcal{D}), \quad (9)$$

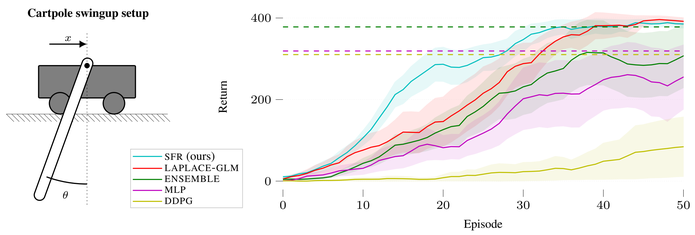


Figure 2. **Cartpole swingup with sparse reward.** Training curves showing that SFR's uncertainty estimates improve sample efficiency in RL. Our method (—) converges in fewer environment steps than the baseline model-based RL method and DDPG, the model-free baseline.

Practical Considerations

- Calculating dual parameters is costly,
  - We batch over data to keep memory occupation in check.
- Uncertainty is highly dependent on prior, e.g. activation fns,
  - Tanh works well in practice (encodes smoothness).
- We tune prior precision (δ) using Bayesian optimisation,
  - After calculating dual parameters.

Outlook

- What functionality of SFR can we leverage for model-based RL?
  - Can we update dynamics during an episode using dual parameters?
  - Use consistent function samples during planning, i.e. pathwise conditioning.
  - Are there clever ways to select inducing points?
    - We only need accurate dynamics along optimal trajectory,
    - Select using determinantal point process (DPP)?
- Can we convert any BNN to SFR, e.g. variational inference,
  - We should be able to consider any distribution over weights $p(\mathbf{w}\,|\,\mathbf{y})$.
- How good are the dual updates in practice?
  - They depend on linearisation around MAP weights,
  - And performing updates pushes us away from the MAP weights...
- Can we speed up computation of dual parameters? E.g. KFAC.

References

[1] V. Adam, P. Chang, M. E. Khan, and A. Solin, "Dual parameterization of sparse variational Gaussian processes," in *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[2] A. Immer, M. Korzepa, and M. Bauer, "Improving predictions of Bayesian neural nets via local linearization," in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

aidan.scannell@aalto.fi