Revisiting Active Sets for Gaussian Process Decoders

Supplementary Material

Pablo Moreno-Muñoz* **Cilie W. Feldager*** **Søren Hauberg**
Section for Cognitive Systems
Technical University of Denmark (DTU)
{pabmo,cife,sohau}@dtu.dk

In this appendix, we provide additional details about stochastic active sets (SAS) as an approximation of the log-marginal likelihood for GP decoders, widely known as the *Gaussian process latent variable model* (GP-LVM). We remark that SAS revisits *active sets*, a sparse approximation predominantly used before the seminal work of Snelson and Ghahramani (2006), in combination with stochastic optimization. The code for experiments is also included, and details on the data and initial setup of hyperparameters are included at the end of this appendix.

A Detailed derivation of Stochastic Active Sets

The construction of SAS approximations for the log-marginal likelihood $\log p(\boldsymbol{x}|\boldsymbol{z})$, builds on the connection between the *evidence* and *cross-validation* (CV) (Fong and Holmes, 2020). The equivalence between *leave-R-out* CV and the log-marginal likelihood is established by the use of predictive posterior scores, such that

$$\mathcal{S}_{\text{CV}}(\boldsymbol{x}|R) = \frac{1}{\mathcal{C}}\sum_{p=1}^{\mathcal{C}}\frac{1}{R}\sum_{n\in\mathcal{R}_p}\log p(\boldsymbol{x}_n|\boldsymbol{x}_{\mathcal{A}_p},\boldsymbol{z}) = \frac{1}{R}\mathbb{E}_{\mathcal{A}_p}\left[\sum_{n\in\mathcal{R}_p}\log p(\boldsymbol{x}_n|\boldsymbol{x}_{\mathcal{A}_p},\boldsymbol{z})\right], \tag{1}$$

where $\mathcal{A}_p$ denotes the *active set* indices of the training data, such that $\mathcal{A}_p \subset \{1,2,\dots,N\}$ and $\mathcal{R}_p = \{1,2,\dots,N\}\setminus\mathcal{A}_p$ are the remaining hold-out samples. The subscript $p\in\mathcal{C}$ denotes the permutation and we average over all $\mathcal{C} = \binom{N}{R}$ possible hold-out sets. We use use R to indicate the size of the hold-out set $\mathcal{R}_p$ and let $A = |\mathcal{A}_p| = N - R$. In particular, one might obtain the log-marginal likelihood in a cumulative manner by summing the scores $\mathcal{S}_{\text{CV}}(\boldsymbol{x}|R)$ in Eq. (1) over all possible lengths of R,

$$\log p(\boldsymbol{x}|\boldsymbol{z}) = \sum_{r=1}^{N}\mathcal{S}_{\text{CV}}(\boldsymbol{x}|r), \tag{2}$$

which is the main result presented in Fong and Holmes (2020). Notice that Eq. (2) has a similar computational cost as the exact calculus of $\log p(\boldsymbol{x}|\boldsymbol{z})$ for GP decoders, since for small values of r, i.e. $r = 1, 2, 3\dots$, we need to invert large covariance matrices $\boldsymbol{K}_{\mathcal{A}\mathcal{A}}$, where $A \to N$. Here, we drop the permutation subscript p in $\mathcal{A}$ to avoid cluttered notation. Alternatively, we use the property that Eq. (2) can be factorised as

$$\log p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R) + \mathcal{S}_{\text{PCV}}(\boldsymbol{x}|R), \tag{3}$$

where $\mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R)$ is the *cumulative* CV score and $\mathcal{S}_{\text{PCV}}(\boldsymbol{x}|R)$ is defined as the *preparatory* CV. Additionally, Eq. (3) holds for every size of the hold-out data $R\in[1,2,\cdots,N]$. This factorisation is of interest for us due to

$$\mathcal{S}_{\text{PCV}}(\boldsymbol{x}|R) = \sum_{r=R+1}^{N}\mathcal{S}_{\text{CV}}(\boldsymbol{x}|r) = \frac{1}{\mathcal{C}}\sum_{p=1}^{\mathcal{C}}\log p(\boldsymbol{x}_{\mathcal{A}_p}|\boldsymbol{z}_{\mathcal{A}_p}), \tag{4}$$

*Equal contribution.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

where the r.h.s. term is equivalent to $\mathcal{S}_{\text{PCV}}(\boldsymbol{x}|R) = \mathbb{E}_{\mathcal{A}_p}[\log p(\boldsymbol{x}_{\mathcal{A}_p}|\boldsymbol{z}_{\mathcal{A}_p})]$. We remark that the computational cost of Eq. (4) is *cheaper* than $\mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R)$ when the choice of R is sufficiently large. Additionally, the *cumulative* CV is defined as

$$\mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R) = \sum_{r=1}^{R} \mathcal{S}_{\text{CV}}(\boldsymbol{x}|r) = \sum_{r=1}^{R} \frac{1}{\mathcal{C}_r} \sum_{p=1}^{\mathcal{C}_r} \frac{1}{r} \sum_{n \in \mathcal{R}_p} \log p(\boldsymbol{x}_n|\boldsymbol{x}_{\mathcal{A}_p}, \boldsymbol{z}), \tag{5}$$

where $\mathcal{C}_r = \binom{N}{r}$ are all the possible hold-out set for every value of r considered. We remark the convenience of Eq. (5) for stochastic optimization as it includes predictive posterior probabilities $p(\boldsymbol{x}_n|\boldsymbol{x}_{\mathcal{A}_p}, \boldsymbol{z})$ which emerge from the factorization of hold-R-out CV. This expression can be also rewritten as

$$\mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R) = \sum_{r=1}^{R} \frac{1}{r} \mathbb{E}_{\mathcal{A}_p} \left[\sum_{n \in \mathcal{R}_p} \log p(\boldsymbol{x}_n|\boldsymbol{x}_{\mathcal{A}_p}, \boldsymbol{z}) \right]. \tag{6}$$

The SAS approximation stochastically estimates both $\mathcal{S}_{\text{CCV}}(\boldsymbol{x}|R)$ and $\mathcal{S}_{\text{PCV}}(\boldsymbol{x}|R)$, with particular attention to the CCV score, which for large values of R induces the largest computational cost.

B Experiments, Algorithms and Metrics

The code for the experiments is written in Python 3.7 and uses the Pytorch syntax for the automatic differentiation of the GP models. It can be found in the repository `https://github.com/pmorenoz/SASGP`, where we also use the library Pyro for some baselines. In this section, we provide a detailed description of the experiments and the data used, the initialization of both latent variables $\boldsymbol{z}$, the parameters of the *amortization* network and hyperparameters $\boldsymbol{\theta}$. The training algorithms are provided in the main manuscript for both the *deterministic* and Bayesian approaches to the GP decoder. The performance metrics included in the main manuscript, e.g. the negative log-predictive density (NLPD), the root mean square error (RMSE) and the mean absolute error (MAE).

B.1 Detailed description and initialization

All the models have matching encoding network architecture: three linear, fully connected layers with ReLU activation functions. The first two layers have sizes of 512 and 256 hidden units and the network encodes to two dimensions. The variances of the latent variables $\boldsymbol{z}$ were encoded in a different way. In the VAE model, the variance was obtained by inputting the latent means to a soft-plus layer and the Bayesian SAS-GP and the Bayesian GP-LVM had separate network (with similar architecture) encoding the variance. The decoder in the VAE was built by a linear, fully connected layer with 400 hidden units, a softplus function and a sigmoid mapping. The GP-LVM baselines are implemented in Pyro (Bingham et al., 2019). Our implementation of the variational autoencoder is based on the official Pyro tutorial for VAEs. Importantly, the VAE and the SAS-GP implementations use standard *data loaders* whereas the Pyro code must keep all the data in memory. This has limited the scaling possibilities of experiments with baselines.

In most of our experiments, we use the *vanilla* RBF kernel, where we initially set the *amplitude* hyperparameter to $\sigma_a^2 = 0.5$, the *lengthscale* to $\ell = 0.1$ and the likelihood noise variance to $\sigma_n^2 = 0.5$. The initial location of latent variables $\boldsymbol{z}$ is subject to the initialization of the amortization networks, which is set up in a standard manner using the Pytorch `nn` module. Learning rates are set in the range $[10^{-4}, 10^{-2}]$ and the maximum number of epochs considered is 300.

B.2 Datasets

Our experiments make use of three well-known datasets: MNIST (LeCun et al., 1998), FMNIST (Xiao et al., 2017) and CIFAR10 (Krizhevsky, 2009). All of them are downloaded from the `torchvision` repository included in the Pytorch library (`https://pytorch.org/vision/stable/datasets.html`). These particular datasets are not subject to use constraints or they include licenses which allow their use for research purposes.

B.3 Error metrics

Having defined the test dataset as $\boldsymbol{x}_* = \{\boldsymbol{x}_n\}_{n=1}^{N_*}$, we use the following error metrics to test the performance of the SAS approximation for GP decoders

$$\text{RMSE}(\boldsymbol{x}_*) = \sqrt{\frac{1}{N_*}\sum_{n=1}^{N_*}(\boldsymbol{x}_n - \boldsymbol{\mu}_n^*)^2}, \tag{7}$$

$$\text{MAE}(\boldsymbol{x}_*) = \frac{1}{N_*}\sum_{n=1}^{N_*}|\boldsymbol{x}_n - \boldsymbol{\mu}_n^*|\,, \tag{8}$$

$$\text{NLPD}(\boldsymbol{x}_*) = \frac{1}{2}\log(2\pi) + \frac{1}{2N_*}\sum_{n=1}^{N_*}\left[\log \boldsymbol{v}_n^* + \frac{(\boldsymbol{x}_n - \boldsymbol{\mu}_n^*)^2}{\boldsymbol{v}_n^*}\right], \tag{9}$$

where $\boldsymbol{\mu}_n^*$ and $\boldsymbol{v}_n^*$ are the predictive mean and variance per nth test sample, respectively.

B.4 Additional experiment with latent spaces of dimension larger than two

We re-computed the experiments used for Table 2 using latent spaces of dimension *three* and *four*. The main outcome from these experiments is that training is as stable as in the former cases with two dimensions in the latent space. In general, we observed a similar performance as in the rest of experiments included in the main manuscript. So we remark that there is no limitation in our framework to accept $\dim(\mathcal{Z}) > 2$.

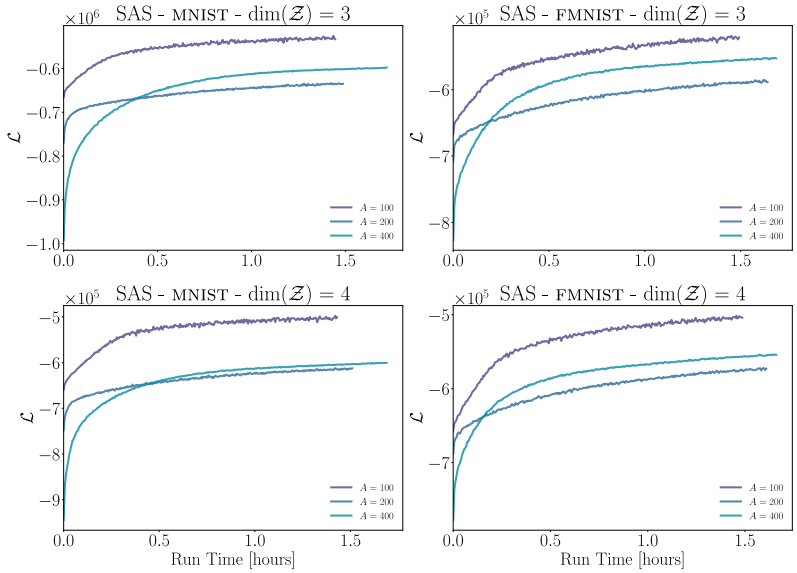

Figure 1: Training curves for different active set sizes A and dimensionalities of the latent spaces. Each row contains the results for MNIST and FMNIST, respectively.

B.5 Ablation study

We did additional experiments as an ablation study based on Eq. 6. In particular, we ran the SAS model for $A = \{100, 200, 400\}$ using the MNIST and FMNIST datasets. The first ablation experiment shown in Figure 2 corresponds to using only the *second term* $\log p(\boldsymbol{x}_\mathcal{A}|\boldsymbol{z}_\mathcal{A})$ in Eq. 6. We can observe that the performance is not as good as in the results illustrated in the main manuscript. Alternatively, we also included another ablation experiment using the *first term* of Eq. 6, which is shown in Figure 3. In this last case, the performance is not good and the structure in the latent space is only provided by the amortization net.

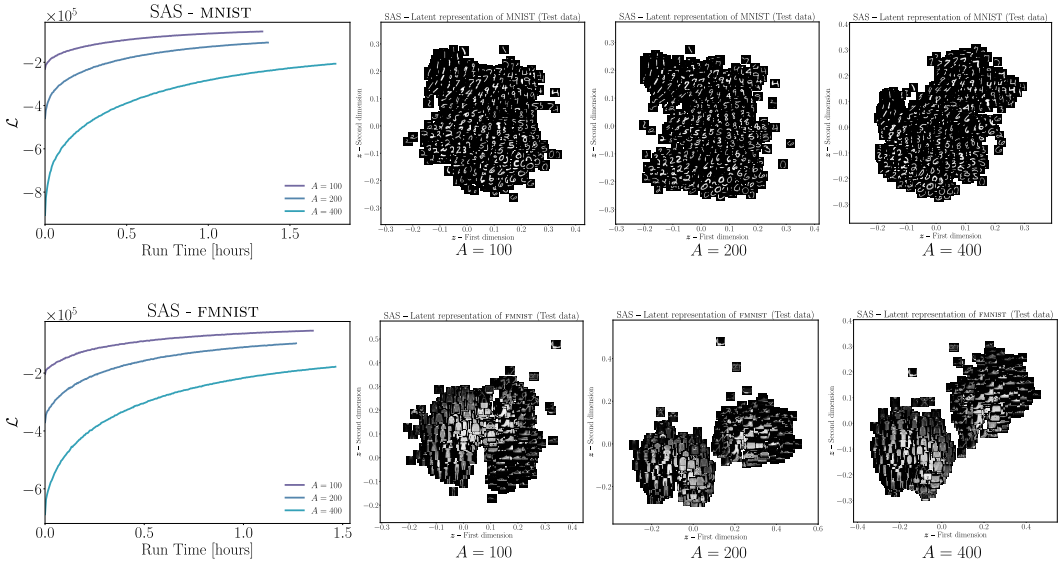

Figure 2: Ablation study for Eq. 6. The approximation of the log-marginal likelihood is only computed with the second term (full covariance). Curves are computed for $A = \{100, 200, 400\}$ and rows indicate the dataset MNIST or FMNIST.

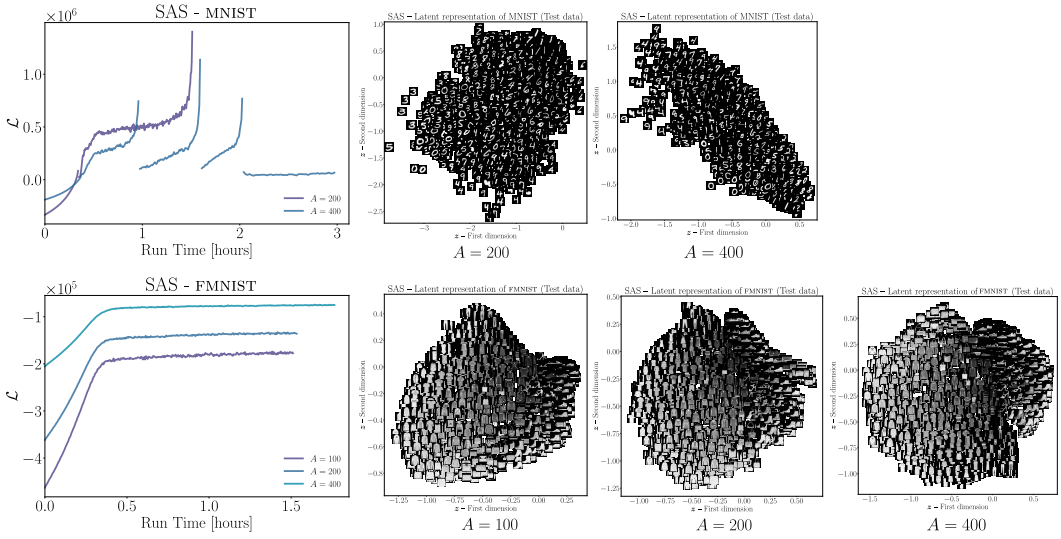

Figure 3: Ablation study for Eq. 6. The approximation of the log-marginal likelihood is only computed with the firm term (factorisation). Curves are computed for $A = \{100, 200, 400\}$ and rows indicate the dataset MNIST or FMNIST.

References

E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. A. Szerlip, P. Horsfall, and N. D. Goodman. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research (JMLR)*, 20:28:1–28:6, 2019.

E. Fong and C. C. Holmes. On the marginal likelihood and cross-validation. *Biometrika*, 107(2):489–496, 2020.

A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1257–1264, 2006.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.