Deep Classifiers with Label Noise Modeling and Distance Awareness

Vincent Fortuin[*]
ETH Zürich
`fortuin@inf.ethz.ch`

Mark Collier, Florian Wenzel, James Allingham, Jeremiah Liu, Dustin Tran, Balaji Lakshminarayanan, Jesse Berent, Rodolphe Jenatton, Effrosyni Kokiopoulou
Google Research

1 Introduction

While deep learning has led to impressive advances in predictive accuracy, models often still suffer from overconfidence and ill-calibrated uncertainties (Ovadia et al., 2019). This is particularly problematic in safety-critical applications (e.g., healthcare, autonomous driving), where uncertainty estimation is crucial to ensure reliability and robustness (Filos et al., 2019; Dusenberry et al., 2020).

Predictive uncertainties generally come in two flavors: *model uncertainty* (also known as epistemic) and *data uncertainty* (also known as aleatoric) (Murphy, 2012). Model uncertainty measures how confident the model should be based on what it knows about the world, that is, how much it can know about certain test data given the training data it has seen. Data uncertainty measures the uncertainty intrinsic in the data itself, for example due to fundamental noise in the labelling process. Good model uncertainty estimates allow for out-of-distribution (OOD) detection, that is, for recognizing data examples that are substantially different from the training data. On the other hand, good data uncertainty estimates allow for in-distribution calibration, that is, knowing which training (or testing) data examples the model should be more or less confident about.

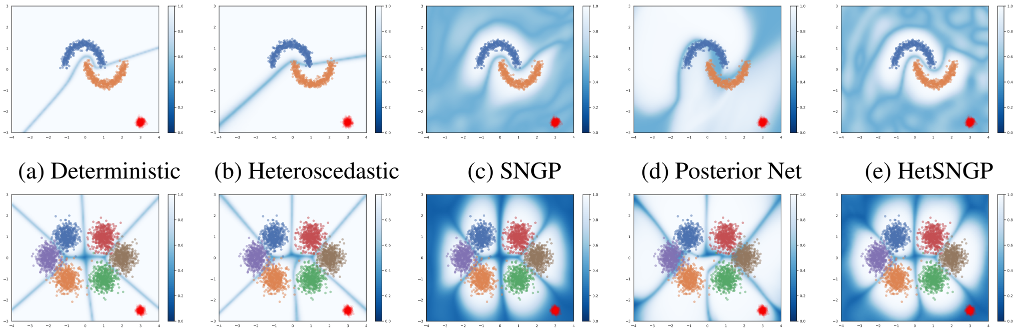
Many recently proposed models for uncertainty estimation excel at one or the other of these uncertainty types. For instance, the spectral-normalized Gaussian process (SNGP) (Liu et al., 2020) uses a latent Gaussian process to achieve distance-aware model uncertainties and thus affords excellent OOD detection. Conversely, the heteroscedastic classification method (Collier et al., 2020, 2021) offers excellent calibration and improved accuracy thanks to modeling input- and class-dependent label noise in the training data. However, there have not been many attempts to combine the complementary benefits of these two types of uncertainty modeling (e.g., Depeweg et al., 2018). In this work, we propose the heteroscedastic SNGP (HetSNGP) model, which allows for joint modeling of model and data uncertainties based on a hierarchy of two latent random variables. We show that HetSNGP gives good in-distribution and OOD accuracy and calibration, yielding a model with uncertainties suitable for deployment in critical applications.

Our main contributions are:

- We propose a new model, the heteroscedastic spectral-normalized Gaussian process (HetSNGP), which provides both distance-aware model and data uncertainties.
- We describe an efficient approximate inference scheme that allows training HetSNGP with a computational budget that is comparable to standard neural network training.
- We show empirically on different benchmark datasets that HetSNGP offers a favorable combination of model and data uncertainty. It generally preserves the SNGP's OOD per-

[*] Work done during an internship at Google Research.

Bayesian Deep Learning workshop, NeurIPS 2021.



(a) Deterministic (b) Heteroscedastic (c) SNGP (d) Posterior Net (e) HetSNGP

(f) Deterministic (g) Heteroscedastic (h) SNGP (i) Posterior Net (j) HetSNGP

Figure 1: Estimated uncertainties on synthetic datasets. The red cluster of points are unseen OOD points, while the other colors represent different classes of training data. The background color shows the normalized predictive uncertainty for the different methods. The deterministic and heteroscedastic methods are overconfident on the OOD data, while the other methods offer distance-aware uncertainties. The uncertainties of the Posterior Network do not grow quickly enough away from the data, such that only the SNGP and proposed HetSNGP provide effective OOD detection.

formance and the heteroscedastic in-distribution performance and even outperforms these baselines on some datasets, where OOD and heteroscedastic uncertainties are helpful.

2 Method

2.1 Setup and Notation

Let us consider a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ of input-output pairs, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, \dots, K\}$, that is, a classification problem with K classes. The data examples are assumed to be sampled *i.i.d.* from some true data-generating distribution as $(\boldsymbol{x}_i, y_i) \sim p^*(\boldsymbol{x}, y)$.

2.2 Generative process

We provide some background on model and data uncertainty in Appendix A. Relevant related work is described in Appendix B. To jointly model these two different types of uncertainty, we propose a hierarchical model of two latent random variables, which we denote by $\boldsymbol{f}$ and $\boldsymbol{u}$. $\boldsymbol{f}$ is a latent function value associated to the input $\boldsymbol{x}$ (as in the Gaussian process literature) and is designed to capture the model uncertainty, while $\boldsymbol{u}$ is a latent vector of logits (or *utilities*) that capture the data uncertainty, similar to the setup in Collier et al. (2020), which was inspired by the econometrics literature (Train, 2009). Similarly to Liu et al. (2020), we place a latent Gaussian process (GP) prior over $\boldsymbol{f}$, as $p(\boldsymbol{f}) = \mathcal{GP}(0, k_\theta(\cdot, \cdot))$, where k_θ is a parameterized kernel function with parameters θ. Evaluating this kernel on all pairwise combinations of datapoints in $\boldsymbol{x}$ yields the kernel matrix $\boldsymbol{K}_\theta(\boldsymbol{x}, \boldsymbol{x})$. We then define $\boldsymbol{u}$ as a (heteroscedastically) noisy observation of $\boldsymbol{f}$.

Stacking the variables across the whole dataset gives us the matrices $\boldsymbol{F}, \boldsymbol{U} \in \mathbb{R}^{N \times K}$. We refer to their respective rows as $\boldsymbol{f}_i, \boldsymbol{u}_i \in \mathbb{R}^K$ and their columns as $\boldsymbol{f}_c, \boldsymbol{u}_c \in \mathbb{R}^N$. The columns are independent under the GP prior, but the rows are not. Conversely, the columns are correlated in the heteroscedastic noise model, while the rows are independent. A hierarchical model using both uncertainties therefore leads to logits that are correlated across both data points and classes.

The full generative process is

$$\boldsymbol{f}_c \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_\theta(\boldsymbol{x}, \boldsymbol{x})) \tag{1}$$

$$\boldsymbol{u}_i \sim \mathcal{N}(\boldsymbol{f}_i, \boldsymbol{\Sigma}(\boldsymbol{x}_i; \varphi)) \tag{2}$$

$$p(y_i = c \,|\, \boldsymbol{u}_i) = \mathbb{1}\left[c = \arg\max_k \, u_{ik}\right] \tag{3}$$

The predictive can then be computed my marginalizing, that is $p(y \,|\, \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{u}}[p(y \,|\, \boldsymbol{u})] = \int p(y \,|\, \boldsymbol{u})\, p(\boldsymbol{u} \,|\, \boldsymbol{x})\, d\boldsymbol{u}$. Intuitively, $\boldsymbol{f}$ captures the model uncertainty, that is, the uncertainty about the functional mapping between $\boldsymbol{x}$ and y on the level of the latents. It uses the covariance between data points to achieve this distance-awareness, namely it uses the kernel function to assess the similarity between data points, yielding an uncertainty estimate that grows away from the data. On the other hand, $\boldsymbol{u}$ captures the data uncertainty, by explicitly modelling the per-class uncertainty on the level of the logits. φ can learn to encode correlations in the noise between different classes (e.g., the dog breeds in Imagenet). It does not itself capture the model uncertainty, but inherits it through its hierarchical dependence on $\boldsymbol{f}$, such that the resulting probability $p(y \,|\, \boldsymbol{u})$ ultimately jointly captures both types of uncertainty.

In practice, we usually learn the kernel using deep kernel learning (Wilson et al., 2016), that is, we define it as the RBF kernel $k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j) = k_{RBF}(\boldsymbol{h}_i, \boldsymbol{h}_j) = \exp(\|\boldsymbol{h}_i - \boldsymbol{h}_j\|_2^2/\lambda)$ with length scale λ and $\boldsymbol{h}_i = h(\boldsymbol{x}_i; \theta) \in \mathbb{R}^m$ with $h(\cdot; \theta)$ being a neural network model (e.g., a ResNet) parameterized by θ. This kernel is then shared between classes. Moreover, following Liu et al. (2020), we typically encourage bi-Lipschitzness of h using spectral normalization (Behrmann et al., 2019), which then leads to an approximate preservation of distances between the input space and latent space, thus allowing for distance-aware uncertainty modeling in the latent space. Additionally, $\boldsymbol{\Sigma}(\cdot; \varphi)$ is usually also a neural network parameterized by φ. To ease notation, we will typically drop these parameters in the following. Note that the prior over $\boldsymbol{f}_c$ (and thus also $\boldsymbol{u}_c$) has zero mean, thus leading to a uniform output distribution away from the data. Note that this is also reminiscent of multi-task GPs (Williams et al., 2007), where separate kernels are used to model covariances between data points and tasks and are then combined into a Kronecker structure. Note that the inference in this model is generally intractable and requires a number of computational approximations. These are described in detail in Appendix C.

3 Experiments

We conducted experiments on synthetic data and standard image classification benchmarks. As baselines we compare against a standard deterministic model (He et al., 2016; Dosovitskiy et al., 2020), the heteroscedastic method (Collier et al., 2021) and SNGP (Liu et al., 2020). We also compare against the Posterior Network model (Charpentier et al., 2020), which also offers distance-aware uncertainties, but it only applies to problems with few classes. Implementation details are deferred to Appendix C.2.

Synthetic data. Following the SNGP paper (Liu et al., 2020), to assess the distance-awareness property of HetSNGP in a visually verifiable way, we performed experiments on two-dimensional synthetic datasets; (i) two moons and (ii) a Gaussian mixture. We see in Fig. 1 that neither the deterministic or heteroscedastic models offer distance-aware uncertainties and are therefore overconfident on the OOD data (red points). The Posterior network does offer distance-aware uncertainties, which on these datasets grow slower away from the training data than the SNGP methods. However, we will see later that scaling Posterior network to datasets with many classes is too expensive to be practical. Only the SNGP and our proposed HetSNGP are highly uncertain on the OOD points, thus allowing for effective OOD detection. This OOD detection performance also transfers to the CIFAR-100 dataset (see Appendix D.1). Moreover, we performed experiments on the noisy concentric circles from Berthon et al. (2021), where the three circular classes have the same mean, but different amounts of label noise. We see in Fig. 2 (in Appendix D) that the heteroscedastic model and our proposed method are able to capture this label noise and thus achieve a better performance, while the deterministic baseline and the SNGP are not. Based on these two synthetic experiments, it becomes apparent that our proposed HetSNGP successfully combines the desirable OOD uncertainties of the SNGP with the heteroscedastic uncertainties on these simple datasets.

Imagenet. A large-scale dataset with natural label noise and established OOD benchmarks is the Imagenet dataset (Deng et al., 2009; Beyer et al., 2020). The heteroscedastic method has been shown to improve in-distribution performance on Imagenet (Collier et al., 2021). We see in Table 1 that HetSNGP outperforms the SNGP in terms of accuracy and likelihood on the in-distribution Imagenet validation set and performs almost on par with the heteroscedastic model. The slight disadvantage compared to the heteroscedastic model suggests a small trade-off due to the restricted parameterization of the output layer and application of spectral normalization. However, the true

Table 1: Results on Imagenet. We used Imagenet-C as near-OOD and Imagenet-A as far-OOD. We report the mean and standard error over 10 runs. Bold numbers are within one standard error of the best model. HetSNGP performs best in terms of accuracy on Imagenet-C and Imagenet-A.

Method	↑ID Acc	↓ID NLL	↓ID ECE	↑ImC Acc	↓ImC NLL	↓ImC ECE	↑ImA Acc	↓ImA NLL	↓ImA ECE
Det.	0.759 ± 0.000	0.952 ± 0.001	0.033 ± 0.000	0.419 ± 0.001	3.078 ± 0.007	0.096 ± 0.002	0.006 ± 0.000	8.098 ± 0.018	0.421 ± 0.001
Het.	**0.771** ± 0.000	**0.912** ± 0.001	0.033 ± 0.000	0.424 ± 0.002	3.200 ± 0.014	0.111 ± 0.001	0.010 ± 0.000	7.941 ± 0.014	0.436 ± 0.001
SNGP	0.757 ± 0.000	0.947 ± 0.001	**0.014** ± 0.000	0.420 ± 0.001	**2.970** ± 0.007	**0.046** ± 0.001	0.007 ± 0.000	7.184 ± 0.009	**0.356** ± 0.000
HetSNGP	0.769 ± 0.001	0.927 ± 0.002	0.033 ± 0.000	**0.428** ± 0.001	2.997 ± 0.009	0.085 ± 0.001	**0.016** ± 0.001	**7.113** ± 0.018	0.401 ± 0.001

Table 2: Results on Imagenet-21k. The reported values are means and standard errors over 5 runs. Bold numbers are within one standard error of the best performing model. We use standard Imagenet, Imagenet-C, Imagenet-A, Imagenet-R, and Imagenet-V2 as OOD datasets. HetSNGP outperforms the baselines on all OOD datasets.

Method	↑ID prec@1	↑Im Acc	↑ImC Acc	↑ImA Acc	↑ImR Acc	↑ImV2 Acc
Det.	0.471 ± 0.000	0.800 ± 0.000	0.603 ± 0.000	0.149 ± 0.000	0.311 ± 0.000	0.694 ± 0.000
Het.	**0.480** ± 0.001	0.796 ± 0.002	0.590 ± 0.001	0.132 ± 0.004	0.300 ± 0.006	0.687 ± 0.000
SNGP	0.468 ± 0.001	0.799 ± 0.001	0.602 ± 0.000	0.165 ± 0.003	0.328 ± 0.005	0.696 ± 0.003
HetSNGP	0.477 ± 0.001	**0.806** ± 0.001	**0.613** ± 0.003	**0.172** ± 0.007	**0.336** ± 0.002	**0.705** ± 0.001

benefits of our model become apparent when looking at the Imagenet OOD datasets (Table 1, right side). Here, we still have the noisy label properties from the original Imagenet dataset, such that heteroscedastic uncertainties are useful, but we are also outside of the training distribution, such that distance-aware model uncertainties become crucial. On Imagenet-C and Imagenet-A, we see that our proposed model makes good use of both of these types of uncertainties and thus manages to outperform all the baselines in terms of accuracy. Additional OOD results on Imagenet-R and Imagenet-V2 are shown in Table 4 in Appendix D. Notably, when ensembling our proposed model, it also outperforms other ensemble baselines on Imagenet-C and even on in-distrubtion accuracy (see Appendix D.2).

Imagenet-21k. We introduce a new large-scale OOD benchmark based on Imagenet-21k. Imagenet-21k is a larger version of the standard Imagenet dataset used above (Deng et al., 2009). It has over 12.8 million training images and 21,843 classes. Each image can have multiple labels, whereas for standard Imagenet a single label is given per image. We train a ViT-B/16 (Dosovitskiy et al., 2020) vision transformer model on the Imagenet-21k dataset. See Appendix C.2 for training details. Despite now being in a setting where the model is trained on an order of magnitude more data and greater than 21× more classes, we can use the standard Imagenet OOD datasets. This assesses the scalability of our method. We also hope this new benchmark will prove useful for future OOD research. We see in Table 2 that the parameter-efficient heteroscedastic method has the best in-distribution precision@1. However, its generalization performance to the OOD benchmarks is the weakest of all the methods. On the other hand, our HetSNGP method recovers almost the full in-distribution performance of the heteroscedastic method, significantly outperforming the deterministic and SNGP methods. Notably, it also clearly outperforms all other methods on the OOD datasets.

4 Conclusion

We have proposed a new model, the HetSNGP, that jointly captures distance-aware model uncertainties and heteroscedastic data uncertainties. HetSNGP allows for a favorable combination of these two complementary types of uncertainty and allows for out-of-distribution detection and generalization as well as in-distribution performance and calibration on different benchmark datasets.

References

Ben Adlam, Jasper Snoek, and Samuel L Smith. Cold posteriors and aleatoric uncertainty. *arXiv preprint arXiv:2008.00029*, 2020.

Laurence Aitchison. A statistical theory of cold posteriors in deep neural networks. In *International Conference on Learning Representations*, 2020.

Javier Antorán, James Urquhart Allingham, and José Miguel Hernández-Lobato. Depth uncertainty in neural networks. *arXiv preprint arXiv:2006.08437*, 2020.

Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582. PMLR, 2019.

Antonin Berthon, Bo Han, Gang Niu, Tongliang Liu, and Masashi Sugiyama. Confidence scores make instance-dependent label-noise learning possible. In *International Conference on Machine Learning*, pp. 825–836. PMLR, 2021.

Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *arXiv preprint arXiv:2006.07159*, 2020.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pp. 1613–1622. PMLR, 2015.

Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. *arXiv preprint arXiv:2006.09239*, 2020.

Kamil Ciosek, Vincent Fortuin, Ryota Tomioka, Katja Hofmann, and Richard Turner. Conservative uncertainty estimation by fitting prior networks. In *International Conference on Learning Representations*, 2019.

Mark Collier, Basil Mustafa, Efi Kokiopoulou, Rodolphe Jenatton, and Jesse Berent. A simple probabilistic method for deep classification under input-dependent label noise. *arXiv preprint arXiv:2003.06778*, 2020.

Mark Collier, Basil Mustafa, Efi Kokiopoulou, Rodolphe Jenatton, and Jesse Berent. Correlated input-dependent label noise in large-scale image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1551–1560, 2021.

Francesco D'Angelo and Vincent Fortuin. Repulsive deep ensembles are bayesian. *arXiv preprint arXiv:2106.11642*, 2021.

Francesco D'Angelo, Vincent Fortuin, and Florian Wenzel. On stein variational neural network ensembles. *arXiv preprint arXiv:2106.10760*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1184–1193. PMLR, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Michael W Dusenberry, Dustin Tran, Edward Choi, Jonas Kemp, Jeremy Nixon, Ghassen Jerfel, Katherine Heller, and Andrew M Dai. Analyzing the role of model uncertainty for electronic health records. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, pp. 204–213, 2020.

Angelos Filos, Sebastian Farquhar, Aidan N Gomez, Tim GJ Rudner, Zachary Kenton, Lewis Smith, Milad Alizadeh, Arnoud De Kroon, and Yarin Gal. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *arXiv preprint arXiv:1912.10481*, 2019.

Vincent Fortuin. Priors in bayesian deep learning: A review. *arXiv preprint arXiv:2105.06868*, 2021.

Vincent Fortuin, Adrià Garriga-Alonso, Mark van der Wilk, and Laurence Aitchison. Bnnpriors: A library for bayesian neural network inference with different prior distributions. *Software Impacts*, 9:100079, 2021a.

Vincent Fortuin, Adrià Garriga-Alonso, Florian Wenzel, Gunnar Rätsch, Richard Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. *arXiv preprint arXiv:2102.06571*, 2021b.

Adrià Garriga-Alonso and Vincent Fortuin. Exact langevin dynamics with stochastic gradients. *arXiv preprint arXiv:2102.01691*, 2021.

Alex Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

Danijar Hafner, Dustin Tran, Timothy Lillicrap, Alex Irpan, and James Davidson. Noise contrastive priors for functional uncertainty. In *Uncertainty in Artificial Intelligence*, pp. 905–914. PMLR, 2020.

Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew M Dai, and Dustin Tran. Training independent subnetworks for robust prediction. *arXiv preprint arXiv:2010.06610*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13, 1993.

Marius Hobbhahn, Agustinus Kristiadi, and Philipp Hennig. Fast predictive uncertainty for classification with bayesian deep networks. *arXiv preprint arXiv:2003.01227*, 2020.

Alexander Immer, Matthias Bauer, Vincent Fortuin, Gunnar Rätsch, and Mohammad Emtiyaz Khan. Scalable marginal likelihood estimation for model selection in deep learning. *arXiv preprint arXiv:2104.04975*, 2021a.

Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of bayesian neural nets via local linearization. In *International Conference on Artificial Intelligence and Statistics*, pp. 703–711. PMLR, 2021b.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. *Learning multiple layers of features from tiny images*. PhD thesis, University of Toronto, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.

Jeremiah Zhe Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax-Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *arXiv preprint arXiv:2006.10108*, 2020.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *arXiv preprint arXiv:1802.10501*, 2018.

Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. *arXiv preprint arXiv:1905.00076*, 2019.

Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based gaussian processes for large-scale calibrated classification. *arXiv preprint arXiv:1805.10915*, 2018.

Miguel Monteiro, Loïc Le Folgoc, Daniel Coelho de Castro, Nick Pawlowski, Bernardo Marques, Konstantinos Kamnitsas, Mark van der Wilk, and Ben Glocker. Stochastic segmentation networks: Modelling spatially correlated aleatoric uncertainty. *arXiv preprint arXiv:2006.06015*, 2020.

Kevin P Murphy. Machine learning: A probabilistic perspective, 2012.

Zachary Nado, Neil Band, Mark Collier, Josip Djolonga, Michael Dusenberry, Sebastian Farquhar, Angelos Filos, Marton Havasi, Rodolphe Jenatton, Ghassen Jerfel, Jeremiah Liu, Zelda Mariet, Jeremy Nixon, Shreyas Padhy, Jie Ren, Tim Rudner, Yeming Wen, Florian Wenzel, Kevin Murphy, D. Sculley, Balaji Lakshminarayanan, Jasper Snoek, Yarin Gal, and Dustin Tran. Uncertainty Baselines: Benchmarks for uncertainty & robustness in deep learning. *arXiv preprint arXiv:2106.04015*, 2021.

Radford M Neal. Bayesian learning via stochastic dynamics. In *Advances in neural information processing systems*, pp. 475–482, 1993.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*, 2019.

Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *International Conference on Machine Learning*, pp. 4075–4084. PMLR, 2018.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, pp. 5. Citeseer, 2007.

Carl Edward Rasmussen and Christopher K Williams. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *arXiv preprint arXiv:1806.01768*, 2018.

Andrew Stirn and David A Knowles. Variational variance: Simple, reliable, calibrated heteroscedastic noise variance parameterization. *arXiv preprint arXiv:2006.04910*, 2020.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *arXiv preprint arXiv:1811.00908*, 2018.

Kenneth E Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

Linh Tran, Bastiaan S Veeling, Kevin Roth, Jakub Swiatkowski, Joshua V Dillon, Jasper Snoek, Stephan Mandt, Tim Salimans, Sebastian Nowozin, and Rodolphe Jenatton. Hydra: Preserving ensemble diversity for model distillation. *arXiv preprint arXiv:2001.04694*, 2020.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Swikatkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? In *International Conference on Machine Learning*, 2020.

Chris Williams, Edwin V Bonilla, and Kian M Chai. Multi-task gaussian process prediction. *Advances in neural information processing systems*, pp. 153–160, 2007.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pp. 370–378. PMLR, 2016.

Wanqian Yang, Lars Lorch, Moritz A Graule, Srivatsan Srinivasan, Anirudh Suresh, Jiayu Yao, Melanie F Pradier, and Finale Doshi-Velez. Output-constrained bayesian neural networks. *arXiv preprint arXiv:1905.06287*, 2019.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

A Background

A.1 Model uncertainty and spectral-normalized Gaussian process

Model uncertainty (or epistemic uncertainty) captures all the uncertainty about whether a given model is correctly specified for a certain task, given the training data. Before any training data has been observed, this uncertainty depends only on the prior knowledge about the task, which can for instance be encoded into distributions about the model parameters or into the architecture of the model (i.e., the model class) (Fortuin, 2021). After training data has been observed, one should expect the model uncertainty to decrease within the support of the training data distribution, that is, on points that are close to the training data in the input space. We will generally call these points *in-distribution* (ID). Conversely, on points that are far away from the training points and thus *out-of-distribution* (OOD), we should not expect the model uncertainty to decrease, since the training data points are not informative enough to make any assertions about the correctness of the given model on these points. In this sense, the optimal model uncertainty should be *distance-aware*, that is, it should grow away from the training data (Liu et al., 2020).

The spectral-normalized Gaussian process (SNGP) (Liu et al., 2020) provides such distance-aware model uncertainties by specifying a Gaussian process prior (Rasmussen & Williams, 2006) over the latent data representations in the penultimate layer of the neural network. Distance-awareness is ensured by using spectral normalization on the hidden layers (Behrmann et al., 2019), which encourages bi-Lipschitzness of the mapping from data to latent space, constraining the model from collapsing any input feature dimensions in the latent representations and approximately preserving the distances between data points in the latent space (Liu et al., 2020). Note that this approach only partially captures the model uncertainty, namely in form of the uncertainty over the latents. It does not however capture the uncertainty over the model parameters, such as for instance Bayesian neural networks (MacKay, 1992; Neal, 1993) or ensemble methods (Lakshminarayanan et al., 2016).

A.2 Data uncertainty and the heteroscedastic method

As opposed to the model uncertainty described above, data uncertainty is intrinsic in the data and thus irreducible with increasing amounts of data. In the case of continuous data (e.g., regression problems), data uncertainty often comes in the form of random noise on the measurements. For discrete data (e.g., classification problems), it usually comes as label noise, that is, a certain number of data points will be assigned the wrong label in the training data. This label noise can be class- and input-dependent (Beyer et al., 2020). For instance, the Imagenet dataset (Deng et al., 2009), contains 100 different classes of dog breeds, which are often harder to tell apart for human labelers than the other classes in the dataset. Modeling this type of data uncertainty can improve the calibration and robustness of predictive models (Collier et al., 2020).

A model that does explicitly handle the input- and class-dependent label noise is the heteroscedastic method (Collier et al., 2021). The heteroscedastic method models input- and class-dependent noise by introducing a latent multivariate Gaussian distribution on the softmax logits of a standard neural network classifier. The covariance matrix of this latent distribution is a function of the input (heteroscedastic) and models inter-class correlations in the logit noise.

B Related Work

Uncertainty estimation. There has been a lot of research on uncertainty estimation in recent years (Ovadia et al., 2019), including the early observation that one needs data and model uncertainties for successful use in real-world applications (Kendall & Gal, 2017). There have also been attempts to directly optimize specialized loss functions to improve model uncertainty (Tagasovska & Lopez-Paz, 2018; Pearce et al., 2018). However, if one wants to directly implement prior knowledge regarding the OOD behavior of the model, one usually needs access to OOD samples during training (Yang et al., 2019; Hafner et al., 2020). Another popular idea is to use a hierarchical output distribution, for instance a Dirichlet distribution (Milios et al., 2018; Sensoy et al., 2018; Malinin & Gales, 2018; Malinin et al., 2019; Hobbhahn et al., 2020), such that the model uncertainty can be encoded in the Dirichlet and the data uncertainty in its Categorical distribution samples. This idea was also used in our Posterior Network baseline (Charpentier et al., 2020). Similarly to our idea of learning separate variances, it has also been proposed to treat the output variance variationally and to specify a hierarchical prior over it (Stirn & Knowles, 2020).

Bayesian neural networks and ensembles. The gold-standard for capturing model uncertainty over the parameters are Bayesian neural networks (Neal, 1993; MacKay, 1992), but they are computationally quite expensive, require a careful choice of prior distribution (Fortuin et al., 2021b; Fortuin, 2021), and require specialized approximate inference schemes, such as Laplace approximation (Immer et al., 2021b,a), variational inference (Hinton & Van Camp, 1993; Graves, 2011; Blundell et al., 2015), or Markov Chain Monte Carlo (Welling & Teh, 2011; Garriga-Alonso & Fortuin, 2021; Fortuin et al., 2021a). A more tractable model class are deep ensemble methods (Lakshminarayanan et al., 2016; Ciosek et al., 2019; D'Angelo et al., 2021; D'Angelo & Fortuin, 2021), although they are computationally still expensive. There are however some ideas to make them less expensive by distilling their uncertainties into simpler models (Malinin et al., 2019; Tran et al., 2020; Havasi et al., 2020; Antorán et al., 2020).

SNGP and heteroscedastic method. The models most relevant to our approach are the SNGP (Liu et al., 2020) and the heteroscedastic method (Collier et al., 2020, 2021). SNGP yields distance-aware model uncertainties, which grow away from the training data. The heteroscedastic method models input- and class-dependent label noise data uncertainty inside the training distribution. Heteroscedastic data uncertainty modelling can also improve segmentation models (Monteiro et al., 2020) and has been linked to the cold posterior effect in Bayesian neural networks (Wenzel et al., 2020; Adlam et al., 2020; Aitchison, 2020). Prior work has primarily focused on modelling data and distance-aware model uncertainty separately. But safety-critical practical applications require both types of uncertainty, HetSNGP is a novel method which fulfills this requirement.

C Implementation

C.1 Computational approximations

The generative process above requires computing two integrals (i.e., over $\boldsymbol{u}$ and $\boldsymbol{f}$) and inferring an exact GP posterior, making it computationally intractable. We make several approximations to ensure tractability.

Low-rank approximation. The heteroscedastic covariance matrix $\boldsymbol{\Sigma}(\boldsymbol{x}_i;\varphi)$ is a $K \times K$ matrix which is a function of the input. Parameterizing this full matrix is costly in terms of computation, parameter count, and memory. Therefore, following Collier et al. (2021), we make a low-rank approximation $\boldsymbol{\Sigma}(\boldsymbol{x}_i;\varphi) = \boldsymbol{V}(\boldsymbol{x}_i)\boldsymbol{V}(\boldsymbol{x}_i)^\top + \boldsymbol{d}^2(\boldsymbol{x}_i)$. Here, $\boldsymbol{V}(\boldsymbol{x}_i)$ is a $K \times R$ matrix with $R << K$ and $\boldsymbol{d}^2(\boldsymbol{x}_i)$ is a K dimensional vector added to the diagonal of $\boldsymbol{V}(\boldsymbol{x}_i)\boldsymbol{V}(\boldsymbol{x}_i)^\top$.

Also following Collier et al. (2021), we introduce a parameter-efficient version of our method to enable scaling to problems with many classes, e.g., Imagenet21k which has 21,843 classes. In the standard version of our method $\boldsymbol{V}(\boldsymbol{x}_i)$ is parameterized as an affine transformation of the shared representation $\boldsymbol{h}_i$, in this way HetSNGP can be added as a single output layer on top of a base network. For the parameter-efficient HetSNGP we parameterize $\boldsymbol{V}(\boldsymbol{x}_i) = \boldsymbol{v}(\boldsymbol{x}_i)\mathbf{1}_R^\top \odot \boldsymbol{V}$ where $\boldsymbol{v}(\boldsymbol{x}_i)$ is a vector of dimension R, $\mathbf{1}_R$ is a vector of ones of dimension R, $\boldsymbol{V}$ is a $K \times R$ matrix of free parameters, and $\odot$ denotes element-wise multiplication.

Random Fourier feature approximation. Computing the exact GP posterior requires $\mathcal{O}(N^3)$ operations (because ones needs to invert an $N \times N$ kernel matrix), which can be prohibitive for large datasets. Following Liu et al. (2020), we thus use a random Fourier feature (RFF) approximation (Rahimi et al., 2007), leading to a low-rank approximation of the kernel matrix as $\boldsymbol{K}_\theta(\boldsymbol{x},\boldsymbol{x}) = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ ($\boldsymbol{\Phi} \in \mathbb{R}^{N \times m}$), with random features $\boldsymbol{\Phi}_i = \sqrt{\frac{2}{m}}\cos(\boldsymbol{W}\boldsymbol{h}_i + \boldsymbol{b})$ where $\boldsymbol{W}$ is a fixed weight matrix with entries sampled from $\mathcal{N}(0,1)$ and $\boldsymbol{b}$ is a fixed bias vector with entries sampled from $\mathcal{U}(0,2\pi)$. This approximates the infinite-dimensional reproducing kernel Hilbert space (RKHS) of the RBF kernel with a subspace spanned by m randomly sampled basis functions. This reduces the GP inference complexity to $\mathcal{O}(Nm^2)$, where m is the dimensionality of the latent space. We can then write the model as a linear model in this feature space, namely

$$\begin{aligned} \boldsymbol{u}_i = \boldsymbol{f}_i + \boldsymbol{d}(\boldsymbol{x}_i) \odot \boldsymbol{\epsilon}_K + \boldsymbol{V}(\boldsymbol{x}_i)\boldsymbol{\epsilon}_R \quad \text{with} \quad \boldsymbol{f}_c = \boldsymbol{\Phi}\boldsymbol{\beta}_c \quad \text{and} \\ p(\boldsymbol{\beta}_c) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_m);\ p(\boldsymbol{\epsilon}_K) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_K);\ p(\boldsymbol{\epsilon}_R) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_R) \end{aligned} \tag{4}$$

Here, $\boldsymbol{\Phi}\boldsymbol{\beta}_c$ is a linear regressor in the space of the random Fourier features and the other two terms of $\boldsymbol{u}_i$ are the low-rank approximation to the heteroscedastic uncertainties (as described above).

Laplace approximation. When using a Gaussian likelihood (i.e., in a regression setting), the GP posterior inference can be performed in closed form. However, for classification problems this is not possible, because the Categorical likelihood used is not conjugate to the Gaussian prior. We thus need to resort to approximate posterior inference. Again following Liu et al. (2020), we perform a Laplace approximation (Rasmussen & Williams, 2006) to the RFF-GP posterior, which yields the closed-form approximate posterior for $\boldsymbol{\beta}_c$

$$p(\boldsymbol{\beta}_c \,|\, \mathcal{D}) = \mathcal{N}(\hat{\boldsymbol{\beta}}_c, \widehat{\boldsymbol{\Sigma}}_c) \quad \text{with} \quad \widehat{\boldsymbol{\Sigma}}_c^{-1} = \boldsymbol{I}_m + \sum_{i=1}^{N} p_{i,c}(1 - p_{i,c})\boldsymbol{\Phi}_i\boldsymbol{\Phi}_i^\top \tag{5}$$

where $p_{i,c}$ is shorthand for the softmax output $p(y_i = c \,|\, \hat{\boldsymbol{u}}_i)$ as defined in Eq. (3), where $\hat{u}_{i,c} = \boldsymbol{\Phi}_i^\top \hat{\boldsymbol{\beta}}_c$. The derivation of this is deferred to Appendix E. Here, $\widehat{\boldsymbol{\Sigma}}_c^{-1}$ can be cheaply computed over minibatches of data by virtue of being a sum over data points. Moreover, $\hat{\boldsymbol{\beta}}_c$ is the MAP solution, which can be obtained using gradient descent on the unnormalized log posterior, $-\log p(\boldsymbol{\beta} \,|\, \mathcal{D}) \propto -\log p(\mathcal{D} \,|\, \boldsymbol{\beta}) - \|\boldsymbol{\beta}\|^2$, where the squared norm regularizer stems from the standard Gaussian prior on $\boldsymbol{\beta}$. Using our likelihood, the training objective takes the form of a ridge-regularized cross-entropy loss. We also use this objective to train the other trainable parameters θ and φ of our model.

Monte Carlo approximation. Finally, we are ultimately interested in $\mathbb{E}_{\boldsymbol{u}}[p(y \,|\, \boldsymbol{u})] = \int p(y \,|\, \boldsymbol{u})\, p(\boldsymbol{u} \,|\, \boldsymbol{x})\, d\boldsymbol{u}$, which again would require solving a high-dimensional integral and is thus computationally intractable. Following Collier et al. (2020), we approximate it using Monte Carlo samples. Moreover, we approximate the argmax in Eq. (3) with a softmax. To allow for a more controlled trade-off between bias and variance in this approximation, we introduce an additional temperature parameter τ into this softmax, leading to

$$\begin{aligned} & p(y_i = c \,|\, \boldsymbol{x}_i) = \frac{1}{S}\sum_{s=1}^{S} p(y_i = c \,|\, \boldsymbol{u}_i^s) = \frac{1}{S}\sum_{s=1}^{S} \frac{\exp(u_{i,c}^s/\tau)}{\sum_{k=1}^{K}\exp(u_{i,k}^s/\tau)} \\ & \text{with} \quad u_{i,c}^s = \boldsymbol{\Phi}_i^\top \boldsymbol{\beta}_c^s + \boldsymbol{d}(\boldsymbol{x}_i) \odot \boldsymbol{\epsilon}_K^s + \boldsymbol{V}(\boldsymbol{x}_i)\boldsymbol{\epsilon}_R^s \quad \text{and} \quad \boldsymbol{\beta}_c^s \sim p(\boldsymbol{\beta}_c \,|\, \mathcal{D}) \quad \text{and} \\ & \boldsymbol{\epsilon}_K^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_K) \quad \text{and} \quad \boldsymbol{\epsilon}_R^s = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_R) \end{aligned} \tag{6}$$

It should be noted that we only use Eq. (6) at test time, while during training we replace $\boldsymbol{\beta}_c^s$ with its mean $\hat{\boldsymbol{\beta}}_c$ instead of sampling it (see Appendix C). Note also that, while a larger number S of Monte Carlo samples increases the computational cost, these samples can generally be computed in parallel on modern hardware, such that the wallclock runtime stays roughly the same. The exact algorithms used in our implementation are shown in Appendix C.

Algorithm 1 HetSNGP training

```
Require: dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$
  Initialize $\theta, \varphi, \boldsymbol{W}, \boldsymbol{b}, \widehat{\boldsymbol{\Sigma}}$
  for train_step $= 1$ to max_step do
    Take minibatch $(\boldsymbol{X}_i, \boldsymbol{y}_i)$ from $\mathcal{D}$
    for $s = 1$ to $S$ do
      $\epsilon_K^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_K)$, $\epsilon_R^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_R)$
      $u_{i,c}^s = \boldsymbol{\Phi}_i^\top \hat{\boldsymbol{\beta}}_c + \boldsymbol{d}(\boldsymbol{x}_i) \odot \boldsymbol{\epsilon}_K^s + \boldsymbol{V}(\boldsymbol{x}_i)\boldsymbol{\epsilon}_R^s$
    end for
    $\mathcal{L} = -\frac{1}{S} \sum_{s=1}^S \log p(\boldsymbol{X}_i, \boldsymbol{y}_i \,|\, \boldsymbol{u}^s) - \|\hat{\boldsymbol{\beta}}\|^2$

    Update $\{\theta, \varphi, \hat{\boldsymbol{\beta}}\}$ via SGD on $\mathcal{L}$
    if final_epoch then
      Compute $\{\widehat{\boldsymbol{\Sigma}}_c^{-1}\}_{c=1}^K$ as per Eq. (5)
    end if
  end for
```

Algorithm 2 HetSNGP prediction

```
Require: test example $\boldsymbol{x}^*$
  $\boldsymbol{\Phi}^* = \sqrt{\frac{2}{m}} \cos(\boldsymbol{W}\boldsymbol{h}(\boldsymbol{x}^*) + \boldsymbol{b})$
  for $s = 1$ to $S$ do
    $\boldsymbol{\beta}_c^s \sim p(\boldsymbol{\beta}_c \,|\, \mathcal{D})$
    $\epsilon_K^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_K)$, $\epsilon_R^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_R)$
    $u_{*,c}^s = \boldsymbol{\Phi}^{*\top} \boldsymbol{\beta}_c^s + \boldsymbol{d}(\boldsymbol{x}^*) \odot \boldsymbol{\epsilon}_K^s + \boldsymbol{V}(\boldsymbol{x}^*)\boldsymbol{\epsilon}_R^s$
  end for
  $p(y^* = c \,|\, \boldsymbol{x}^*) = \frac{1}{S} \sum_{s=1}^S \frac{\exp(u_{*,c}^s/\tau)}{\sum_{k=1}^K \exp(u_{*,k}^s/\tau)}$
  Predict $y^* = \arg\max_c p(y^* = c \,|\, \boldsymbol{x}^*)$
```

The training of our proposed model is described in Algorithm 1 and the prediction in Algorithm 2.

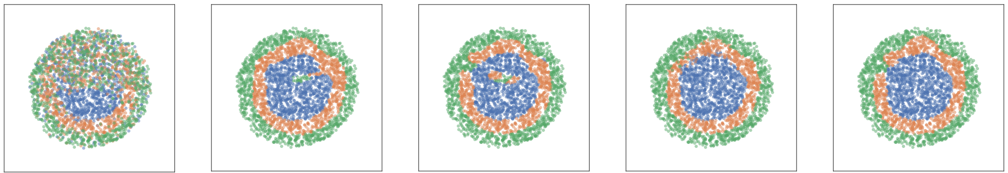
Intuition. The distance-aware uncertainties of the SNGP are modeled through $\boldsymbol{\Phi}$, which itself depends on the latent representations $\boldsymbol{h}$, which are approximately distance-preserving thanks to the bi-Lipschitzness. This means that when we have an input $\boldsymbol{x}_{\text{far}}$ that is far away from the training data, the values of the RBF kernel $\exp(\|\boldsymbol{h} - \boldsymbol{h}_{\text{far}}\|_2^2/\lambda)$ will be small, and thus the uncertainty of $\boldsymbol{f}_c$ will be large. The resulting output uncertainties will therefore be large regardless of the heteroscedastic variance $\boldsymbol{\Sigma}(\boldsymbol{x}_i)$ (since they are additive) allowing for effective OOD detection. Conversely, if we have an in-distribution input, the kernel values will be large and the model uncertainty captured in $\boldsymbol{f}_c$ will be small. In this case, the additive output uncertainty will be dominated by the heteroscedastic variance $\boldsymbol{\Sigma}(\boldsymbol{x}_i)$, allowing for effective label noise modeling in-distribution.

Challenges and limitations. As outlined above, the main challenge in this model is the inference. We chose an approximate inference scheme that is computationally efficient, utilizing a series of approximations, and the fact that we model the distance-aware and heteroscedastic variances additively. If we wanted to make the model more powerful, at the cost of increased computation, we could thus consider several avenues of improvement: (i) use inducing points or even full GP inference for the GP posterior; (ii) use a more powerful approximation than Laplace, for instance, a variational approximation; or (iii) model the variances jointly, such that we do not only have covariance between data points in $\boldsymbol{f}$ and between classes in $\boldsymbol{u}$, but have a full covariance between different classes of different data points in $\boldsymbol{u}$. All of these ideas are interesting avenues for future research.

C.2 Implementation details

To assess our proposed model's predictive performance and uncertainty estimation capabilities, we conducted experiments on synthetic two moons data (Pedregosa et al., 2011), a mixture of Gaussians, the CIFAR-100 dataset (Krizhevsky et al., 2009), and the Imagenet dataset (Deng et al., 2009). We compare against a standard deterministic ResNet model as a baseline (He et al., 2016), against the heteroscedastic method (Collier et al., 2020, 2021) and the SNGP (Liu et al., 2020) (which form the basis for our combined model) and against the recently proposed Posterior Network model (Charpentier et al., 2020), which also offers distance-aware uncertainties, similarly to the SNGP. We used the same backbone neural network architecture for all models, which was a fully-connected ResNet for the synthetic data, a WideResNet18 on CIFAR and a ResNet50 in Imagenet.

For most baselines, we used the hyperparameters from the `uncertainty_baselines` library (Nado et al., 2021). On CIFAR, we trained our HetSNGP with a learning rate of 0.1 for 300 epochs and used $R = 6$ factors for the heteroscedastic covariance, a softmax temperature of $\tau = 0.5$ and $S = 5000$ Monte Carlo samples. On Imagenet, we trained with a learning rate of 0.07 for 270 epochs and used $R = 15$ factors, a softmax temperature of $\tau = 1.25$ and $S = 5000$ Monte Carlo samples. We



Train labels

Deterministic
(acc = 0.855 ± 0.004)

SNGP
(acc = 0.855 ± 0.001)

Heteroscedastic
(acc = 0.868 ± 0.002)

HetSNGP
(acc = 0.867 ± 0.002)

Figure 2: Predicted labels on synthetic data with label noise. We see that the proposed method performs on par with the heteroscedastic method and that they both outperform the other baselines thanks to their label noise modeling capabilities.

Table 3: Results on CIFAR-100. We used Places365 as far-OOD and CIFAR-10 as near-OOD datasets. The reported values are means and standard errors over 10 runs. Bold numbers are within one standard error of the best performing model. Our model outperforms the baselines in terms of accuracy on corrupted data and performs on par with the best models on OOD detection.

Method	↑ID Acc	↓ID NLL	↑Corr Acc	↓Corr NLL	↑Near-ROC	↓Near-FPR95	↑Far-ROC	↓Far-FPR95
Det.	**0.808** ± 0.000	0.794 ± 0.002	0.455 ± 0.001	2.890 ± 0.010	**0.506** ± 0.006	0.963 ± 0.005	0.442 ± 0.060	0.935 ± 0.041
Post.Net.	0.728 ± 0.001	1.603 ± 0.014	0.444 ± 0.001	3.086 ± 0.009	0.472 ± 0.013	1.000 ± 0.000	**0.518** ± 0.016	1.000 ± 0.000
Het.	0.807 ± 0.001	0.782 ± 0.002	0.447 ± 0.001	3.130 ± 0.018	0.496 ± 0.005	**0.957** ± 0.004	0.420 ± 0.024	0.958 ± 0.010
SNGP	0.797 ± 0.001	**0.762** ± 0.002	0.466 ± 0.001	**2.339** ± 0.007	0.493 ± 0.011	0.961 ± 0.005	**0.518** ± 0.073	**0.919** ± 0.030
HetSNGP	0.799 ± 0.001	0.856 ± 0.003	**0.471** ± 0.001	2.565 ± 0.007	**0.499** ± 0.010	**0.955** ± 0.005	**0.525** ± 0.038	**0.910** ± 0.024

implemented all models in TensorFlow in Python and trained on Tensor Processing Units (TPUs) in the Google Cloud.

We train all Imagnet-21k models for 90 epochs with batch size 1024 on 8×8 TPU slices. We train using the Adam optimizer with initial learning rate of 0.001 using a linear learning rate decay schedule with termination point 0.00001 and a warm-up period of 10,000 steps. We train using the sigmoid cross-entropy loss function and L2 weight decay with multiplier 0.03. The heteroscedastic method uses a temperature of 0.4, 1,000 Monte Carlo samples and $R = 50$ for the low rank approximation. HetSNGP has the same heteroscedastic hyperparameters except the optimal temperature is 1.5. For SNGP and HetSNGP the GP covariance is approximated using the momentum scheme presented in Liu et al. (2020) with momentum parameter 0.999.

D Additional experiments

D.1 CIFAR experiment

We start by assessing our method on a real-world image dataset; we trained it on CIFAR-100 and used CIFAR-10 as a near-OOD dataset and Places365 (Zhou et al., 2017) as far-OOD. We measure the OOD detection performance in terms of area under the receiver-operator-characteristic curve (ROC) and false-positive-rate at 95% confidence (FPR95). We also evaluated the methods' generalization performance on corrupted CIFAR-100 (Hendrycks & Dietterich, 2019). In Table 3 we see that HetSNGP yields a performance between the heteroscedastic and SNGP methods in terms of in-distribution accuracy, but outperforms all baselines in accuracy on the corrupted data. Moreover, it performs on par with the best performing models on both near- and far-OOD detection.

D.2 Ensembling experiment

Deep ensembles are popular methods for model uncertainty estimation due to their simplicity and good performance (Lakshminarayanan et al., 2016). We propose a variant of our HetSNGP method which captures a further source of model uncertainty in the form of parameter uncertainty by a deep ensemble of HetSNGP models. HetSNGP Ensemble is therefore a powerful yet simple method for capturing three major sources of uncertainty; 1) data uncertainty in the labels, 2) model uncertainty in the latent representations and 3) model uncertainty in the parameters.

Table 4: Additional Imagenet OOD results on Imagenet-R and Imagenet-V2. The reported values are means and standard errors over 10 runs. Bold numbers are within one standard error of the best performing model. Our model outperforms the baselines in terms of accuracy on Imagenet-V2.

Method	↑ImR Acc	↓ImR NLL	↓ImR ECE	↑ImV2 Acc	↓ImV2 NLL	↓ImV2 ECE
Det.	0.229 ± 0.001	5.907 ± 0.014	0.239 ± 0.001	0.638 ± 0.001	1.598 ± 0.003	0.077 ± 0.001
Het.	**0.235** ± 0.001	5.761 ± 0.010	0.251 ± 0.001	**0.648** ± 0.001	1.581 ± 0.002	0.084 ± 0.001
SNGP	0.230 ± 0.001	**5.344** ± 0.009	**0.175** ± 0.001	0.637 ± 0.001	**1.552** ± 0.001	**0.041** ± 0.001
HetSNGP	0.232 ± 0.001	5.452 ± 0.011	0.225 ± 0.002	**0.647** ± 0.001	1.564 ± 0.003	0.080 ± 0.001

Table 5: Results on Imagenet. Ensemble size=4. Imagenet-C is used as the OOD dataset. HetSNGP outperforms the baselines in terms of in-distribution accuracy and all on OOD metrics.

Method	↑ID Acc	↓ID NLL	↓ID ECE	↑ImC Acc	↓ImC NLL	↓ImC ECE
Det Ensemble	0.779	0.857	0.017	0.449	2.82	0.047
Het Ensemble	0.795	**0.790**	**0.015**	0.449	2.93	0.048
SNGP Ensemble	0.781	0.851	0.039	0.449	2.77	0.050
HetSNGP Ensemble	**0.797**	0.798	0.028	**0.458**	**2.75**	**0.044**

We compare the HetSNGP Ensemble to a determinstic ensemble as well as ensembles of heteroscedastic and SNGP models. We see in Table 5 that in the case of an ensemble of size four, HetSNGP Ensemble outperforms the baselines on the Imagenet-C OOD dataset in terms of all metrics, while also outperforming them on the standard in-distribution Imagenet dataset in terms of accuracy. Due to computational constraints we do not have error bars in this experiment. However, we do not expect them to be much larger than in Table 1.

E Laplace approximation

In this section, we will derive the Laplace posterior in Eq. (5). The derivation follows mostly from the sections 3.4 and 3.5 in Rasmussen & Williams (2006).

First note that the log posterior of $\boldsymbol{\beta}_c$ given the data is

$$\log p(\boldsymbol{\beta}_c \,|\, \boldsymbol{x}, y) = \log p(y \,|\, \boldsymbol{\beta}_c) + \log p(\boldsymbol{\beta}_c) - Z \tag{7}$$

where Z is a normalization constant that does not depend on $\boldsymbol{\beta}_c$. Following Rasmussen & Williams (2006), we will denote the unnormalized log posterior as

$$\Psi(\boldsymbol{\beta}_c) = \log p(y \,|\, \boldsymbol{\beta}_c) + \log p(\boldsymbol{\beta}_c) \tag{8}$$

Recall that the first term is the likelihood and the second term is our prior from Eq. (4).

The Laplace approximation now approximates the posterior with a local second-order expansion around the MAP solution, that is

$$p(\boldsymbol{\beta}_c \,|\, \boldsymbol{x}, y) \approx \mathcal{N}(\hat{\boldsymbol{\beta}}_c, \boldsymbol{\Lambda}^{-1}) \tag{9}$$

with the MAP solution $\hat{\boldsymbol{\beta}}_c = \arg\max_{\boldsymbol{\beta}_c} \Psi(\boldsymbol{\beta}_c)$ and the Hessian $\boldsymbol{\Lambda} = -\nabla^2 \Psi(\beta_c)|_{\boldsymbol{\beta}_c = \hat{\boldsymbol{\beta}}_c}$.

The MAP solution can be found using standard (stochastic) gradient descent, while the Hessian is given by

$$\begin{aligned}
\nabla^2 \Psi(\boldsymbol{\beta}_c) &= \nabla^2 \log p(y \,|\, \boldsymbol{\beta}_c) + \nabla^2 \log p(\beta_c) \\
&= \nabla_{\boldsymbol{\beta}} (\nabla_{\boldsymbol{u}} \log p(y \,|\, \boldsymbol{u}) \nabla_{\boldsymbol{\beta}} \boldsymbol{u}) - \boldsymbol{I}_m \\
&= \nabla_{\boldsymbol{\beta}} (\nabla_{\boldsymbol{u}} \log p(y \,|\, \boldsymbol{u}) \boldsymbol{\Phi}) - \boldsymbol{I}_m \\
&= \boldsymbol{\Phi}^\top \nabla^2_{\boldsymbol{u}} \log p(y \,|\, \boldsymbol{u}) \boldsymbol{\Phi} - \boldsymbol{I}_m \\
&= -W \boldsymbol{\Phi}^\top \boldsymbol{\Phi} - \boldsymbol{I}_m
\end{aligned}$$

where we used the chain rule and the fact that $\boldsymbol{u} = \boldsymbol{\Phi}\boldsymbol{\beta}$ and W is a diagonal matrix of point-wise second derivatives of the likelihood, that is, $W_{ii} = -\nabla^2 \log p(y_i \,|\, \boldsymbol{u}_i)$ (Rasmussen & Williams, 2006). For instance, in the case of the logistic likelihood, $W_{ii} = \boldsymbol{p}_i\,(1 - \boldsymbol{p}_i)$, where $\boldsymbol{p}_i$ is a vector of output probabilities for logits $\boldsymbol{u}_i$. To get the Hessian at the MAP, we then just need to compute this quantity for $\hat{\boldsymbol{u}} = \boldsymbol{\Phi}\hat{\boldsymbol{\beta}}$.

The approximate posterior is therefore

$$p(\boldsymbol{\beta}_c \mid \boldsymbol{x}, y) \approx \mathcal{N}(\hat{\boldsymbol{\beta}}_c, (W\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \boldsymbol{I}_m)^{-1}) \tag{10}$$

where the precision matrix can be computed over data points (recovering Eq. (5)) as

$$\boldsymbol{\Lambda} = \boldsymbol{I}_m + \sum_{i=1}^{N} \boldsymbol{p}_i(1 - \boldsymbol{p}_i)\boldsymbol{\Phi}_i\boldsymbol{\Phi}_i^\top \tag{11}$$