Fortuna: A Library for Uncertainty Quantification in Deep Learning

Gianluca Detommaso[1] DETOMMA@AMAZON.DE
Alberto Gasparin[2] ALBGAS@AMAZON.DE
Michele Donini[1] DONINI@AMAZON.DE
Matthias Seeger[1] MATTHIS@AMAZON.DE
Andrew Gordon Wilson[3] WILSMMAN@AMAZON.COM
Cedric Archambeau[1] CEDRICA@AMAZON.DE

[1] *AWS, Berlin, Germany*
[2] *Amazon, Berlin, Germany*
[3] *AWS & New York University*

Abstract

We present Fortuna, an open-source library for uncertainty quantification in deep learning. Fortuna supports a range of calibration techniques, such as conformal prediction that can be applied to any trained neural network to generate reliable uncertainty estimates, and scalable Bayesian inference methods that can be applied to deep neural networks trained from scratch for improved uncertainty quantification and accuracy. By providing a coherent framework for advanced uncertainty quantification methods, Fortuna simplifies the process of benchmarking and helps practitioners build robust AI systems.

1. Introduction

Virtually every application of machine learning ultimately involves decision making under uncertainty. Predictive uncertainty lets us evaluate the trustworthiness of model predictions, prompts human intervention, or determines whether a model can be safely deployed in the real-world. Proper uncertainty estimation is crucial for ensuring the reliability and safety of machine learning applications.

Unfortunately, deep neural networks are often overconfident. In classification, overconfidence means that the estimated probability of the predicted class is significantly higher than the actual proportion of correctly classified input data points (Guo et al., 2017). Overconfidence is problematic because it impacts decisions and ensuing actions. For example, a doctor may have requested an additional test if she were to know that a diagnosis made by an AI was less confident, and a self-driving car may have asked a human driver to take over if it was unsure about the existence of an obstacle in front of the car. Hence, calibrated uncertainty estimates are vital for assessing the reliability of machine learning systems, triggering human intervention, or judging whether a model can be safely deployed.

There are many techniques for estimating and calibrating uncertainty estimates, including temperature scaling (Guo et al., 2017), conformal prediction (Vovk et al., 2005) and Bayesian inference (Gelman et al., 2013). While there are existing open-source implementa-

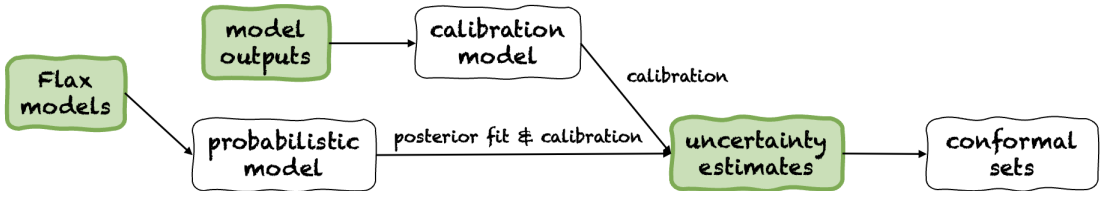

Figure 1: Fortuna provides three usage modes, each starting from one of the colored panels.

tions of methods for uncertainty quantification (Nado et al., 2021; Chung et al., 2021; Ghosh et al., 2021; Phan et al., 2019; Bingham et al., 2019; Tran et al., 2019; Ritter and Karaletsos, 2022), they tend to provide general-purpose probabilistic programming languages, without support for scalable state-of-the-art methods, or individual implementations that do not support a broad range of methods in a unified interface. While modern techniques are highly scalable and practical, the lack of a unified framework has hindered the adoption of uncertainty quantification in practice, a gap that we address by the release of Fortuna.

Fortuna is an open-source library for uncertainty quantification that brings together state-of-the-art scalable methods from the literature and provides them to users through a standardized and easy-to-use interface.[1] Our goal is to make it simpler for practitioners to deploy a variety of advanced uncertainty quantification techniques in regression and classification tasks. Fortuna allows users to calibrate uncertainty estimates for trained deep neural networks using techniques such as temperature scaling and conformal prediction. Fortuna also provides scalable Bayesian inference methods for the training of neural networks from scratch, which can improve both calibration and accuracy. Fortuna is written in JAX (Bradbury et al., 2018), a fast growing NumPy-like framework that allows for native and efficient computation of gradients essential for large-scale Bayesian inference, and adopts Flax (Heek et al., 2020), which has been integrated in many other AI frameworks, including Hugging Face (Wolf et al., 2020).

2. Usage modes

One can make use of Fortuna starting from (1) uncertainty estimates (Section 2.1), (2) model outputs (Section 2.2), or (3) Flax models (Section 2.3). These modes are ordered in terms of decreasing convenience, but increasing flexibility and control over the pipeline. We illustrate these usage modes in Figure 1.

2.1 Starting from uncertainty estimates

Starting from uncertainty estimates is the easiest method of interacting with the library. This usage mode offers conformal prediction methods for both classification and regression tasks. These methods take uncertainty estimates in the form of a `numpy.ndarray` and return a set of predictions with a user-specified probability level. In univariate regression tasks, conformal sets can be thought of as confidence or credible intervals with a calibration guarantee. However, if the provided uncertainty estimates are inaccurate, the resulting conformal sets can be too large to be useful. For this reason, we recommend the usage modes in 2.2 and 2.3 to start conformal methods from better uncertainty estimates.

1. Fortuna documentation and GitHub repository.

At the time of writing, for classification the library supports the baseline conformal prediction by Vovk et al. (2005) and the more advanced adaptive conformal prediction by Romano et al. (2020). For regression, it offers conformalized quantile regression (Romano et al., 2019), conformal intervals from a scalar uncertainty measure (Angelopoulos et al., 2022), Jackknife+, Jackknife-minmax and CV+ (Barber et al., 2021). For time series regression, Fortuna supports EnbPI (Xu and Xie, 2021). For online settings with distribution shifts, it supports Adaptive Conformal Inference (Gibbs and Candes, 2021). The library further supports a variety of multivalid calibration methods, including Multicalibrate (Hébert-Johnson et al., 2018; Roth, 2022) and BatchMVP (Jung et al., 2022). The method of choice may depend on which uncertainty estimates the user wants to start from, which conformal guarantees they want to achieve, or whether re-training the model is a feasible option in their setting.

Example. We wish to calibrate credible intervals with coverage error given by `error`. We assume to be given credible intervals (`test_lower_bounds` and `test_upper_bounds`) corresponding to different test input variables, and prediction intervals for several validation inputs (`val_lower_bounds` and `val_upper_bounds`), along with corresponding validation targets (`val_targets`). The following code produces conformal prediction intervals as calibrated versions of the test prediction intervals.

```
from fortuna.conformal.regression import QuantileConformalRegressor
conformal_intervals = QuantileConformalRegressor().conformal_interval(
    val_lower_bounds=val_lower_bounds, val_upper_bounds=val_upper_bounds,
    test_lower_bounds=test_lower_bounds, test_upper_bounds=test_upper_bounds,
    val_targets=val_targets, error=error)
```

2.2 Starting from model outputs

This mode assumes that a model has already been trained, possibly in another framework, and that model outputs for each input data point are available to Fortuna (i.e., estimates of logits for classification or conditional expectations and variances for regression). This usage mode allows for calibration of the model outputs, estimation of the uncertainty, computation of the metrics, and generation of the conformal sets. Compared to the mode in Section 2.1, this mode offers additional control over the final uncertainty estimates, at the price of a few more lines of code. Nevertheless, if the model was trained as a point estimator, the resulting epistemic uncertainty estimates may not be reliable, and the usage mode in Section 2.3 may be even more preferable.

Models outputs are passed to an additional model written in Flax. The simplest and most popular choice is temperature scaling, which scales the outputs of the model (specifically, the logits in classification and the output uncertainty in regression) using a single parameter (Guo et al., 2017). The temperature scaling method is provided explicitly in Fortuna.

Example. Assuming we have calibration and test model outputs (`calib_outputs` and `test_outputs`) as well as calibration targets (`calib_targets`), the following code provides a minimal example of obtaining calibrated predictive entropy estimates for a classification task.

```
from fortuna.output_calib_model import OutputCalibClassifier
output_calib_model = OutputCalibClassifier()
status = output_calib_model.calibrate(calib_outputs=calib_outputs,
                                      calib_targets=calib_targets)
test_entropies = output_calib_model.predictive.entropy(outputs=test_outputs)
```

2.3 Starting from Flax models

The usage modes discussed above are agnostic to how model outputs or initial uncertainty estimates are obtained. By contrast, the most flexible mode discussed here requires the input of deep learning models written in Flax. By replacing traditional model training with scalable Bayesian inference, one can improve the quantification of predictive uncertainty and accuracy significantly (Wilson and Izmailov, 2020). Bayesian methods can represent *epistemic* uncertainty — uncertainty in the model parameters due to lack of information. Since neural networks can represent many different compelling solutions through different parameter settings, Bayesian methods can be particularly useful in deep learning. Fortuna offers various scalable Bayesian inference methods that provide uncertainty estimates, as well as improved accuracy and calibration, at the price of a training time overhead.

At the time of writing, Fortuna supports Maximum-A-Posteriori (MAP) (Bassett and Deride, 2019), Automatic Differentiation Variational Inference (ADVI) (Kucukelbir et al., 2017), Deep Ensembles (Lakshminarayanan et al., 2017), Laplace approximation with diagonal Generalized Gauss-Newton (GNN) Hessian approximation (Daxberger et al., 2021; Schraudolph, 2002), SWAG (Maddox et al., 2019), Stochastic Gradient Hamiltonian Monte Carlo (Chen et al., 2014), Cyclical Stochastic Gradient Langevin Dynamics (Zhang et al., 2022) and Spectral-normalized Neural Gaussian Process (Liu et al., 2020).

Example. If we have a Flax deep learning classifier (`model`) that maps inputs to `output_dim` logits, as well as training, validation, and calibration TensorFlow data loaders (`train_data_loader` , `val_data_loader` , `test_data_loader`), the following code provides a minimal example for obtaining calibrated probability estimates.

```
from fortuna.data import DataLoader
train_data_loader = DataLoader.from_tensorflow_data_loader(train_data_loader)
calib_data_loader = DataLoader.from_tensorflow_data_loader(val_data_loader)
test_data_loader = DataLoader.from_tensorflow_data_loader(test_data_loader)
test_inputs_loader = test_data_loader.to_inputs_loader()

from fortuna.prob_model import ProbClassifier
prob_model = ProbClassifier(model=model)
status = prob_model.train(train_data_loader=train_data_loader,
                          calib_data_loader=calib_data_loader)
test_means = prob_model.predictive.mean(inputs_loader=test_inputs_loader)
```

3. Conclusion

We introduced Fortuna, a library for uncertainty quantification in deep learning. Fortuna supports state-of-the-art methods in a coherent interface. To get started with Fortuna, you can consult the GitHub repository, the documentation, and the AWS blog post.

References

Anastasios N Angelopoulos, Amit Pal Kohli, Stephen Bates, Michael Jordan, Jitendra Malik, Thayer Alshaabi, Srigokul Upadhyayula, and Yaniv Romano. Image-to-image regression with distribution-free uncertainty quantification and applications in imaging. In *International Conference on Machine Learning*, pages 717–730. PMLR, 2022.

Rina Foygel Barber, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1):486–507, 2021.

Robert Bassett and Julio Deride. Maximum a posteriori estimators as a limit of bayes estimators. *Mathematical Programming*, 174(1):129–144, 2019.

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul A. Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep universal probabilistic programming. *J. Mach. Learn. Res.*, 20:28:1–28:6, 2019. URL http://jmlr.org/papers/v20/18-403.html.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103, 2021.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

Soumya Ghosh, Q. Vera Liao, Karthikeyan Natesan Ramamurthy, Jiri Navratil, Prasanna Sattigeri, Kush R. Varshney, and Yunfeng Zhang. Uncertainty quantification 360: A holistic toolkit for quantifying and communicating the uncertainty of ai, 2021.

Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020. URL http://github.com/google/flax.

Christopher Jung, Georgy Noarov, Ramya Ramalingam, and Aaron Roth. Batch multivalid conformal prediction. *arXiv preprint arXiv:2209.15145*, 2022.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *Journal of machine learning research*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Jeremiah 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. *Advances in Neural Information Processing Systems*, 33: 7498–7512, 2020.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

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.

Du Phan, Neeraj Pradhan, and Martin Jankowiak. Composable effects for flexible and accelerated probabilistic programming in numpyro. *arXiv preprint arXiv:1912.11554*, 2019.

Hippolyt Ritter and Theofanis Karaletsos. Tyxe: Pyro-based bayesian neural nets for pytorch. *Proceedings of Machine Learning and Systems*, 4:398–413, 2022.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

Yaniv Romano, Matteo Sesia, and Emmanuel Candes. Classification with valid and adaptive coverage. *Advances in Neural Information Processing Systems*, 33:3581–3591, 2020.

Aaron Roth. Uncertain: Modern topics in uncertainty estimation, 2022.

Nicol N Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural computation*, 14(7):1723–1738, 2002.

Dustin Tran, Mike Dusenberry, Mark Van Der Wilk, and Danijar Hafner. Bayesian layers: A module for neural network uncertainty. *Advances in neural information processing systems*, 32, 2019.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Chen Xu and Yao Xie. Conformal prediction interval for dynamic time-series. In *International Conference on Machine Learning*, pages 11559–11569. PMLR, 2021.

Ruqi Zhang, Andrew Gordon Wilson, and Christopher De Sa. Low-precision stochastic gradient langevin dynamics. In *International Conference on Machine Learning*, pages 26624–26644. PMLR, 2022.