A SURVEY ON UNCERTAINTY TOOLKITS FOR DEEP LEARNING

Maximilian Pintz
University of Bonn, Fraunhofer IAIS
Schloss Birlinghoven
53757 Sankt Augustin, Germany
{maximilian.alexander.pintz}@iais.fraunhofer.de

Joachim Sicking, Maximilian Poretschkin, Maram Akila
Fraunhofer IAIS
Schloss Birlinghoven
53757 Sankt Augustin, Germany
{joachim.sicking,maximilian.poretschkin,maram.akila}@iais.fraunhofer.de

ABSTRACT

The success of deep learning (DL) fostered the creation of unifying frameworks such as tensorflow or pytorch as much as it was driven by their creation in return. Having common building blocks facilitates the exchange of, e.g., models or concepts and makes developments easier replicable. Nonetheless, robust and reliable evaluation and assessment of DL models has often proven challenging. This is at odds with their increasing safety relevance, which recently culminated in the field of "trustworthy ML". We believe that, among others, further unification of evaluation and safeguarding methodologies in terms of toolkits, i.e. small and specialized framework derivatives, might positively impact problems of trustworthiness as well as reproducibility. To this end, we present the first survey on toolkits for uncertainty estimation (UE) in DL, as UE forms a cornerstone in assessing model reliability. We investigate 11 toolkits with respect to modeling and evaluation capabilities, providing an in-depth comparison for the 3 most promising ones, namely *Pyro*, *Tensorflow Probability*, and *Uncertainty Quantification 360*. While the first two provide a large degree of flexibility and seamless integration into their respective framework, the last one has the larger methodological scope.

1 INTRODUCTION

Supervised deep learning (DL) has spurred progress in various application domains including computer vision (Mahony et al., 2019; Chai et al., 2021), natural language processing (Torfi et al., 2020), and speech recognition (Alam et al., 2020) and is increasingly employed in safety-critical systems such as autonomous vehicles (Yurtsever et al., 2020) or medical diagnosis systems (Hafiz & Bhat, 2020). These systems potentially harm the environment, destroy equipment or put humans at risk (Sommerville, 2010), for instance, when vulnerable road users (in case of autonomous driving) or a severe disease (in case of a diagnostics system) are not recognized. The field of trustworthy ML (Brundage et al., 2020; Chatila et al., 2021; Liu et al., 2021) seeks to identify and mitigate such risks of ML systems to enable their responsible and safe operation. To this end, a more holistic notion of quality is proposed, that extends beyond task performance and considers aspects like fairness, interpretability and reliability. Central to the latter is the identification and handling of uncertainties, i.e. factors that affect the system but are often not or only poorly accounted for. Such uncertainties are ubiquitous in DL and stem, for instance, from data (e.g. measurement noise and incorrect annotations) or variabilities in training procedures (e.g. hyperparameters, initializations and limited training data). Quantifying their impact on a system can contribute to its safety: for instance, an autonomous vehicle, that recognizes abnormal sensor input, may switch into a conservative travel mode or may activate redundant safety systems.
The desirable adaptation of uncertainty estimation (UE) for a wide range of use cases is often hindered

by high implementation complexity. This adaptation could be accelerated by open-source toolkits, which reduce this effort by providing easy-to-use building blocks and guidance in the form of minimal working examples. Being used and reviewed by the community, toolkits may moreover strengthen technical quality of UE components and advance common practices. The latter, i.e. the use of common tools and standards, may drive further progress in the field as it facilitates easier reviews and reusing existing work.
In light of this, we survey deep uncertainty toolkits, to the best of our knowledge for the first time. The survey is structured as follows: we first give a brief overview on DL software and UE (section 2). Next, we select 11 publicly available open-source deep uncertainty toolkits from 180 considered repositories (section 3) and present a catalog of criteria for their evaluation (section 4). We then apply these criteria to the collected toolkits in an comparative analysis (section 5), describing their strengths and weaknesses. Finally, we highlight limitations and desirable future advancements of current deep uncertainty toolkits in the discussion (section 6).

2 Related work

We first describe two types of DL software, DL frameworks and specialized toolkits that are discussed throughout this survey. Next, we give an overview over surveys on DL software and sketch their methodology and focus points in the second paragraph. To provide a basis for the subsequent survey on UE toolkits, some foundational concepts of UE, especially w.r.t. their modeling and assessment, are reviewed in the third paragraph. Some simple UE capabilities are shipped as part of standard DL frameworks like tensorflow and pytorch. We outline these "default" capabilities and sketch their limitations in the last paragraph of the section.

DL frameworks and specialized toolkits There is a wide range of DL software covering different facets of DL. Two special types, namely DL frameworks and specialized toolkits, are in focus throughout this survey and described in the following.
DL frameworks provide a comprehensive, reliable (well-tested/-maintained) and highly usable (modular structure, high-level interfaces) set of base components for building and evaluating deep learning systems. They typically provide layer modules that can be put together to form custom model architectures and gradient-descent-based optimizers for model training. They are designed in respect of DL-specific software requirements including flexible data pipelines, automatic differentiation and efficient processing (GPU/multicore utilization, device distribution). Examples include tensorflow (Abadi et al., 2015), pytorch (Paszke et al., 2019) and MXNet (Chen et al., 2015).
Specialized toolkits extend the capabilities of DL frameworks. They typically provide a collection of commonly used tools within a sub-field of deep learning. Examples include spaCy (Honnibal et al., 2020) for natural language processing, GluonCV (Guo et al., 2020) for computer vision as well as the adversarial robustness toolbox (Nicolae et al., 2018). The UE toolkits analyzed in this work fall into this category of DL software.

Surveys of DL software Surveys on DL software concentrate mostly on the systematic assessment of DL frameworks (Nguyen et al., 2019; Wang et al., 2019; Landset et al., 2015; Druzhkov & Kustikova, 2016; Bahrampour et al., 2015). Typical criteria for analysis include the range of provided components, GPU/multiprocessing support, integration of big data frameworks and general software criteria such as efficiency/scalability, ease of use and framework design/extensibility. Only few studies exist regarding specialized toolkits, such as Zacharias et al. (2018) who concentrate on libraries for intelligent user interfaces or Agarwal & Das (2020) who compare toolkits for interpretable machine learning under the aspects of range of provided methods and use cases. We seek to extend the overview on software for trustworthy AI with this survey on UE toolkits.

Elements of uncertainty estimation In the following, we summarize central aspects of uncertainty estimation for the analysis of this survey. Further details to these aspects are provided in a methodological "deep dive" in the appendix (section 6.1).
DL systems are affected by several types of uncertainties (Hüllermeier & Waegeman, 2021). A distinction can be made between reducible *epistemic uncertainty* that arises from a lack of knowledge about the perfect model for solving a task, which includes lack of training data or approximation errors, and irreducible *aleatoric uncertainty*, which includes inherent randomness in the data-generating process (e.g. sensor noise or label ambiguity).
UE methods seek to quantify such uncertainties. In this survey we categorize them along two inde-

pendent "axes": their *integration depth* into a given model and the methodological framework they are based on. Along the former axis, we differentiate between (i) *intrinsic methods* that integrate the uncertainty estimation directly into the architecture or training procedure (e.g. dropout layers or ensemble training), (ii) *post-hoc methods* that equip standard deep learning models with probability estimates and (iii) *recalibration methods* that seek to improve existing uncertainty estimates (Guo et al., 2017; Kuleshov et al., 2018; Navrátil et al., 2021).
The "axis" of *methodological approaches* comprises the following. (i) Parametric likelihood methods consider networks that directly output distributional parameters instead of point estimates and are trained via likelihood optimization (Nix & Weigend, 1994; Amini et al., 2020; Sensoy et al., 2018). (ii) Bayesian neural networks (BNN) extend these approaches by incorporating (epistemic) uncertainty on network parameters. Examples include variational-inference based BNNs (VI-BNN, Graves (2011); Blundell et al. (2015); Rezende & Mohamed (2015)), their variants based on dropout-sampling (Gal & Ghahramani, 2016; Kendall & Gal, 2017)), Markov-chain Monte Carlo sampling (MCMC, Neal et al. (2011); Welling & Teh (2011); Li et al. (2016)) or assumed density filtering (ADF-BNN, Hernández-Lobato & Adams (2015); Gast & Roth (2018)). A related Bayesian approach are Gaussian processes (GP, Rasmussen & Williams (2006)) and their scalable variants (Hensman et al., 2013) that incorporate uncertainty in function space. (iii) Frequentist approaches directly leverage a combination of different models for UE, e.g. deep ensembles (Lakshminarayanan et al., 2017; Wen et al., 2019; Durasov et al., 2021) or the jackknife method (Giordano et al., 2019; Alaa & Van Der Schaar, 2020; Kim et al., 2020).
Several *assessment techniques* have been developed that enable the benchmarking of uncertainty methods and the evaluation of their quality. Most commonly, these include (i) *(proper) scoring rules* (Gneiting & Raftery, 2007) that measure the fit of a predicted distribution to a ground-truth value (e.g. negative log-likelihood (NLL) or brier score), (ii) *calibration testing* (Guo et al., 2017; Kuleshov et al., 2018; Navrátil et al., 2021) that assesses alignment with a validation dataset (e.g. reliability diagrams, excepted calibration error (ECE)), (iii) *qualitative assessments* (e.g. inspecting predicted distributions or the average predicted variance across a dataset) and (iv) the *performance on auxiliary tasks*, e.g. uncertainty-based separation of in-distribution and out-of-distribution datapoints as in Ovadia et al. (2019).

Uncertainty estimation in standard DL frameworks Common DL frameworks natively provide basic building blocks for uncertainty estimation. TensorFlow/Keras[1], MXNet or Pytorch[2] provide dropout functions that are originally intended for regularization (Srivastava et al., 2014) but can also be used for building dropout-based uncertainty models. Pytorch exhibits the `distributions` package, which implements backpropagation-compatible probability distributions using the reparametrization trick (Kingma et al., 2015). All mentioned frameworks support basic assessment techniques, e.g. mean squared error or histograms. Despite these implementations, building a fully featured uncertainty model still typically requires significant implementation effort. For example, implementing a VI-BNN requires custom layer modules that impose a probability distribution on network parameters and the optimization of a custom objective function. We instead focus on toolkits that build on these existing functionalities and provide a higher level of abstraction for reducing implementation effort.

3 SELECTION OF DEEP UNCERTAINTY TOOLKITS

The first step of our systematic survey is to compile a collection of open-source toolkits on deep uncertainty estimation, which is the goal of this section. For this, we first describe our procedure for compiling a pre-selection of potential code repositories on deep UE. Next, we collect basic criteria a code repository needs to fulfill to be considered as an uncertainty toolkit in the scope of this work. We finally apply these criteria to the pre-selection to obtain a collection of 11 deep uncertainty toolkits that we use in our comparative analysis (see section 5).

Search for code repositories related to UE in DL In this survey, we focus on open-source code repositories that are available on `github.com`. We concentrate on github as it is the largest platform to host open-source code and to our knowledge hosts most of the publicly available deep uncertainty estimation software. We use the public search function on github to find uncertainty-related code repositories. In particular, we make use of "topics", specific keywords for tagging repositories that

[1]Keras was originally a separate library but has become a part of tensorflow. It provides high-level APIs for model building, training and evaluation.

[2]The UE toolkits that we consider in our analysis build upon these three DL frameworks.

allow other users to find these repositories more easily. In addition, users can "star" repositories they are especially interested in, which is a way of bookmarking the repository. We examined the topics "uncertainty-quantification", "uncertainty-estimation" and also searched for the keywords "uncertainty" and "probabilistic" restricted to repositories of the topics "machine-learning" and "deep-learning". In each case, we ordered the search results by the total number of "stars" they received so far and examined the first 30 repositories in the resulting list, which adds up to a total of 180 repositories. Among these repositories, there are deep uncertainty estimation libraries, various implementations accompanying research papers as well as software without a clear focus on deep learning, e.g. general libraries on sensitivity analysis for numerical models.

Criteria constituting an uncertainty toolkit As the focus of our work is on toolkits in supervised deep learning, we further filter the 180 repositories. We count a software library as a *deep uncertainty toolkit* if it contains multiple methods for at least one of the following:

- building and training a deep learning model that (inherently) supports UE (intrinsic methods),
- extending a given deep learning model with uncertainty estimates (post-hoc methods),
- improving already existing uncertainty estimates (recalibration) or
- assessing uncertainty estimates in terms of metrics or visualizations.

Additionally, we require a toolkit to provide methods that are generally applicable to a broad range of scenarios, in particular to different datasets or network architectures. Sufficient documentation must be provided and either an application programming interface (API) or an (interactive) graphical user interface (GUI) application (in case of a standalone program). The toolkit should have a clear focus on deep learning as evident from the documentation and should be integrated with common DL frameworks.

Selected toolkits Out of the 180 repositories, 11 satisfy the above outlined criteria constituting uncertainty toolkits. They are listed in Table 1 in the appendix. Among these, we find toolkits dedicated purely to deep uncertainty estimation (dedicated UE) as well as libraries with a broader scope that also provide UE capabilities. The latter category includes *GluonTS* (GTS, Alexandrov et al. (2020)), a probabilistic time series library, and numerous deep probabilistic programming libraries (PPLs) that aim at specifying general Bayesian networks and performing inference for such models (van de Meent et al., 2018). Namely these are *Tensorflow Probability* (TFP, Dillon et al. (2017)), *Pyro* (Bingham et al., 2019), *Edward2* (ED2, Tran et al. (2018)), *ZhuSuan* (ZS, Shi et al. (2017)) and *MXFusion* (MXF, Meissner et al. (2019)).
Among the dedicated UE toolkits is, for instance, *Uncertainty Quantification 360* (UQ360, Ghosh et al. (2021)), which provides several uncertainty estimation tools and is part of a larger set of toolkits developed by IBM Research, each targeting individual dimensions of AI trustworthiness, including "AI Fairness 360" (Bellamy et al., 2018) and "AI Explainability 360" (Arya et al., 2020). We also find several libraries with a more narrow range of functionality: *Uncertainty Toolbox* (UT, Chung et al. (2021)) focuses on the recalibration and the assessment of uncertainty estimates in standard regression tasks. *Uncertainty Wizard* (UW, Weiss & Tonella (2021)) is a package for extending keras models with dropout and ensemble-based uncertainty estimation capabilities. *Bayesian Torch* (BT, Krishnan et al. (2022)) provides pytorch layer modules for building variational Bayesian neural networks and *Keras-ADF* (KADF, Maces & Contributors (2019)) provides keras layer modules for assumed density filtering. All of these toolkits are released under permissive free software licenses (namely Apache-2.0, MIT or BSD3) that come along with only minimal restrictions and allow, amongst others, their commercial usage.

4 Evaluation criteria for uncertainty toolkits

In this section, we detail our criteria that we use for analyzing the uncertainty estimation toolkits in section 5. We divide both the criteria and the analysis in a "core" part, which we evaluate for all toolkits, and an extended more in-depth analysis for a selected subset of the most versatile tools, for which we also consider "additional" quality criteria. The "core" part focuses on the range of uncertainty modeling and evaluation techniques a framework supports as well as the neural architectures and data types/structures it is compatible with. The "additional" criteria moreover take code quality into account, both in terms of integration with standard DL frameworks as well as their level of documentation, modularity and testing.

Core criteria

Range of supported uncertainty methods Each uncertainty estimation method comes with its own set of strengths and weaknesses w.r.t. estimation quality, computational/storage costs, practicability (ease of implementation, training and evaluation), flexibility or theoretical soundness. This implies that depending on the exact application scenario, some methods may be more suited than others. Thus, we consider the range of functionalities for building or improving uncertainty models or extending standard (deterministic) models a relevant (core) evaluation criterion.

Range of supported evaluation techniques A crucial part in research and development of DL models is the assessment of their prediction quality and the comparison against other models. To enable such an assessment for uncertainty models, a toolkit should provide dedicated metrics for measuring the quality of uncertainty estimates, such as proper scoring rules, calibration or auxiliary scores. As no metric is universally suited for every application scenario, a toolkit should cover a wide range of different uncertainty metrics, potentially aided by visualizations (e.g. calibration plots or confidence bands).

Range of supported architectures and data structures A toolkit that only supports the multi-layer perceptron architecture hardly finds use in computer vision tasks due to its lack of support for convolutional layers or image inputs. This illustrates that to be applicable to different use cases, a toolkit should support a wide range of network architectures, optimizers and data structures, which we consider a relevant criterion for our evaluation.

Additional criteria

Integration with DL frameworks Deep learning software in general comes with a special set of requirements that include flexible data pipelines, efficient optimization (automatic differentiation, GPU/multicore utilization), reproducibility (logging of hyperparameters, seeds and configurations) and modular model building. DL frameworks like tensorflow or pytorch are (i) geared to these desiderata, (ii) well-maintained and exhibit a large community of developers and users and (iii) can be seen as de-facto standards when developing deep learning software. In order to profit from their properties, an uncertainty toolkit should actively employ tools from DL frameworks and exhibit high interoperability with them, e.g. by providing layer modules that can be inserted into existing framework-native models.

Software quality We also evaluate the toolkits in terms of general software quality criteria including (i) usability and documentation quality, (ii) modularity and integrability and (iii) maintenance and testing. A highly usable toolkit is, among others, delivered with a simple installation procedure, a code architecture/API that is easy to understand and to work with and is well-documented. A highly modular toolkit provides flexible building blocks, ideally at different abstraction levels from high-level (being easy-to-use, but less flexible) to low-level (highest level of customization, but higher implementation effort and a higher degree of technical understanding required) that can be combined with each other. Important aspects of code maintenance include actively supporting code contributions, heeding coding style conventions (e.g. via code linting) and enforcing code testing (e.g. continuous integration and reporting code coverage).

5 Comparative analysis of the selected uncertainty toolkits

In the following, we first analyze all uncertainty toolkits from section 3 w.r.t. the core criteria of supported uncertainty methods, evaluation techniques and architectures/data structures (cf. section 4). Based on this analysis, we select the 3 most relevant toolkits, namely UQ360, Pyro and TFP, for an in-depth analysis in subsection 5.2. This will, additionally, include the extended set of evaluation criteria from section 4. We structure the analysis in both cases along the criteria (and not along the toolkits) to better contrast the toolkits against one another.

5.1 Analysis with respect to core criteria

We now briefly compare all 11 uncertainty toolkits selected in section 3 w.r.t. the core criteria. A concise summary of these findings per toolkit can be found in the upper part of Table 2 in the appendix.

Range of supported uncertainty methods UQ360 provides the widest range of methods, covering intrinsic, post-hoc and recalibration methods. All other toolkits cover only intrinsic methods, except

for UT, which has a narrow focus on basic recalibration methods. The deep PPLs provide intrinsic Bayesian methods, mainly MCMC, VI-BNNs or Gaussian processes. In contrast to comparable methods from BT or UQ360, they typically cover a broader range of prior, likelihood and variational posterior distributions. The time series library GTS only provides parametric likelihood methods, but supports a large range of distributions (especially compared to UQ360 that only supports Gaussian likelihoods). UW, BT and KADF are dedicated UE libraries with a narrow focus on the intrinsic methods dropout/ensembling, VI-BNN and ADF-BNN, respectively.

Range of supported evaluation techniques UQ360 and UT have the most comprehensive set of assessment tools ranging from scoring rules over calibration scores and plotting functions, followed by TFP, which lacks plotting functions. GTS provides scoring rules and a function for plotting forecasted time series with confidence bands. In comparison, Pyro, ED2 and ZS have more narrow capabilities and focus on the standard evaluation metric in Bayesian inference, namely computing log-probabilities of the predictive distribution. Some toolkits (UW, BT, KADF and MXF) do not contain any assessment capabilities.

Range of supported architectures and data structures Deep PPLs are constructed to enable Bayesian inference for a broad range of models with different architectures. UW, BT and KDF provide support for classification and regression models. The former provides a keras-based model interface for this purpose, while BT and KDF provide drop-in replacements for dense and convolutional layers. BT additionally covers recurrent models by including probabilistic long short-term memory layers (Hochreiter & Schmidhuber, 1997). In contrast, GTS and UT focus on specific domains such as time series modelling (GTS) or 1D regression (UT).

Overall, we find TFP, Pyro and UQ360 to provide the most comprehensive catalog of models, assessment techniques and supported architectures.

5.2 Detailed analysis of TFP, Pyro and UQ360

We now extend our analysis for the three most promising toolkits, namely UQ360, Pyro and TFP. In particular, we provide more details w.r.t. the core criteria and consider the two "additional" criteria, integration with DL frameworks and software quality. For a tabular summary of the results, see the bottom part of Table 2 in the appendix.

Range of supported uncertainty methods As discussed before, UQ360 provides a wide range of methods at different levels of integration depth. Specifically, its intrinsic methods comprise pure aleatoric uncertainty estimation via Gaussian likelihoods as well as joint modeling approaches for aleatoric and epistemic uncertainty via VI-BNNs or deep ensembles. Post-hoc methods include the infinitesimal jackknife (Giordano et al., 2019) and surrogate model approaches (Chen et al., 2019). UQ360 moreover is the only toolkit with recalibration methods other than standard Platt scaling or isotonic regression and additionally includes e.g. auxiliary interval predictors (Thiagarajan et al., 2020) and UCC rescaling (Navrátil et al., 2021).
The PPLs TFP and Pyro cover intrinsic methods such as parametric likelihood (aleatoric uncertainty), VI-BNNs (aleatoric + epistemic) and (variational) Gaussian processes (functional uncertainty). They also provide non-parametric posterior sampling methods such as Hamiltonian Monte Carlo (Neal et al., 2011) or general Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970), which may be useful for researchers due to its high estimation accuracy, but is typically infeasible to apply in large-scale models. A more scalable variant, SGLD (Welling & Teh, 2011; Li et al., 2016), is additionally provided by TFP. By design, the PPLs support a much broader range of prior, likelihood and variational posterior distributions compared to the corresponding modules of UQ360. For example, it is possible to employ normalizing flows (Rezende & Mohamed, 2015) for more flexible priors or posteriors.

Range of supported evaluation techniques UQ360 offers a wide range of assessment techniques. It covers standard metrics, namely the Gaussian NLL for regression, the classification ECE and the Brier score as well as less frequently used scores such as the area under the risk-rejection-rate curve (Franc & Prusa, 2019) or the uncertainty calibration curve (UCC, Navrátil et al. (2021)). Apart from these metrics, it has helpful "tools" for assessment such as a function that decomposes samples of class probabilities into aleatoric and epistemic uncertainty or various plotting functions (e.g. calibration curves and prediction intervals).
In comparison, TFP has more narrow assessment capabilities when it comes to comparing uncertainty estimates with ground truth labels and provides the above mentioned standard metrics for

this purpose. Instead of scalar values, a TFP model predicts distribution objects. These provide functions for computing standard statistics such as log-probabilities, moments, quantiles, correlation, cumulative distribution functions (cdf's) and KL divergences to other distributions. These statistics can be computed for a wide range of distributions besides the Gaussian, including e.g. multivariate distributions and mixture models.
Pyro offers a module for approximate sampling from the predictive distribution of a given Bayesian model, which serves as a basis for further assessment. For this, it provides general statistical utilities (e.g. for computing quantiles, autocorrelation or prediction intervals) and the CRPS (Gneiting & Raftery, 2007) as a scoring rule.

Range of supported architectures and data structures UQ360's method catalog covers regression and classification scenarios (mostly in 1D). However, many model classes (e.g. VI-BNN or deep ensembles) have pre-written training procedures or even architectures, which limits the ability to extend a given deep learning model with uncertainty estimates. Inputs are typically required to be of (2D) tabular form, limiting the ability to deal with sequential data or other data types. In contrast, TFP is a collection of smaller building blocks and provides layer and optimizer modules that can be used in addition to or as drop-in replacements for the standard tensorflow modules. It provides, for instance, probabilistic drop-in replacement modules for dense or convolutional layers to turn given NNs into BNNs. For pytorch, Pyro provides comparable capacities for turning NNs into BNNs, essentially by subclassing the respective NN classes and replacing the pytorch parameters with stochastic modules. The highly modular structure of both PPLs allows to build models with a wide variety of different architectures and inputs, for instance, for multidimensional regression and for sequential data.

Integration with deep learning frameworks As Tensorflow Probability (TFP) is a part of the tensorflow ecosystem, its modules are designed to interface seamlessly with other modules of tensorflow/keras. In particular, it integrates with the keras model API, which allows the developer to build and train probabilistic models using keras' native `compile` and `fit` functions. Pyro utilizes layer, optimizer, data and distribution functions from its base framework pytorch. When building BNNs in Pyro, one can use standard pytorch modules (e.g. layers and activation functions) in addition to the newly provided stochastic modules. Moreover, models and inference procedures can be encapsulated as pytorch modules, which enables the creation of TorchScript programs for deployment.
Most model classes of UQ360 are based on pytorch. These are easy-to-use, but not as flexible and modular as TFP's building blocks for keras models. Changing network architectures, input pipelines or the training procedure often requires modifications to the code, instead of simply passing different modules (e.g. optimizers and dataloaders) to the model class.

Software quality All 3 toolkits have a simple installation procedure and are listed on the Python package index. The toolkits and all their dependencies can be installed via a single pip command, except for TFP, which requires installing a compatible version of tensorflow beforehand. Pyro additionally provides, besides pip, a docker image for installation. There is a sensible choice of dependencies across the toolkits, using a base DL framework plus standard libraries like numpy, scipy, matplotlib or pandas. All toolkits are actively maintained, welcome to post issues and pull requests and provide an explicit documentation for contributing (via a readme). TFP and Pyro follow coding style guides (e.g. PEP8 [3]) to ensure that newly added code fits to coding conventions and, moreover, employ continuous integration for code testing.
We next discuss interface structures of the toolkits. UQ360 has a simple scikit-learn-esque API (i.e. model classes providing `fit` and `predict` functions taking numpy arrays as input) that is equally intuitive to understand. TFP employs the keras model API, which is highly flexible and also easy to understand. Pyro's API puts more emphasis on the Bayesian modelling perspective of uncertainty estimation. The user needs to define two separate classes (or functions), where one defines the model with its prior and likelihood distributions and the other (the so-called guide) defines the variational posterior distribution. As Pyro uses an API structure different from the widely known APIs (for instance of such as sklearn or keras), it might be more difficult for users to quickly get started with it. On the other hand, the possibility to automatically generate standard posterior distributions (e.g. mean-field VI) or to create constrained network parameters (e.g. positive scale parameters for distributions) improves API usability. In comparison to TFP, Pyro misses direct drop-in replacements for standard layers, which results in a higher implementation effort for customizations.
The code of UQ360 is simple and highly readable, but could more frequently employ input checks

[3] PEP8 are coding style guidelines used by the official python distribution.

(e.g. for types or shapes) and proper exception handling. The code bases of Pyro and TFP are designed to be flexible and to satisfy compatibility requirements (e.g. implementing functions of superclasses) of their respective base library (pytorch or tensorflow) and have (partly as a result from this) a more complex and covert structure. It is of high overall quality, which manifests in appropriate input checking and exception handling and in a clean and consistent coding style.

6 DISCUSSION

Reliable models should be able to identify the boundaries at which they function properly. Finding sources of uncertainty and quantifying their impact on the model performance contributes to this aspect. While many approaches to uncertainty estimation have been developed, there are entry barriers on their use, including high technical complexity. By providing high-quality software components, toolkits for UE help to overcome such entry barriers and additionally facilitate standardized evaluation. To help the reader in selecting an appropriate toolkit, we provide the first survey on existing deep uncertainty toolkits. We defined minimum requirements for such toolkits and analyzed 11 of them with respect to range of supported uncertainty methods, evaluation techniques, architectures and data structures. The 3 most relevant ones (TFP, Pyro and UQ360) were additionally examined under the aspects of integration with DL frameworks and software quality. All analyzed toolkits provide modules to ease the development and assessment of uncertainty models. They encompass deep probabilistic programming libraries (e.g. Pyro, MXF, ZS) that focus on infusing Bayesian inference into DL models as well as toolkits dedicated to UE (e.g. UQ360, UT) that cover a broader range of methods or assessments. We plan to extend our detailed analysis to more toolkits in future iterations of this work. The survey also reveals desirable improvements to UE software and future incentives that are discussed in the following.

Extend the technical capabilities of uncertainty toolkits The considered UE toolkits are either more comprehensive (e.g. UQ360) or more modular and interoperable with DL frameworks (e.g. TFP, Pyro, UW). An UE toolkit should ideally combine both aspects. Further concrete means for improving the technical comprehensiveness of currently available toolkits include: (i) providing more post-hoc methods, which are of high interest in scenarios where rebuilding a model or retraining is associated with high costs; (ii) infusing uncertainty into application-specific network components, e.g. into non-maximum suppression or clustering procedures of computer vision models (as in Meyer et al. (2019); Harakeh et al. (2020)); (iii) providing tools for task performance benchmarking that account for uncertainties (e.g. based on hypothesis testing as in (Gorman & Bedrick, 2019)).

Interfacing with other toolkits There is a multitude of relevant tasks and problems in deep learning besides uncertainty estimation. For example, different metrics and approaches are considered to assess and ensure different aspects of AI trustworthiness, such as robustness, interpretability or fairness. To address multiple such problems simultaneously employed toolkits should provide interoperable interfaces. This might be facilitated if, as we evaluated, the toolkits integrate seamlessly with the (same) underlying DL framework. Such interoperability between toolkits can even directly impact UE. For instance, data augmentation toolkits (e.g. augLy (Papakipos & Bitton, 2022)) provide input corruptions for out-of-distribution assessments and online-learning toolkits (e.g. river (Montiel et al., 2020)) can be used to adapt uncertainty models to future observations. A future prospect is the combination of several toolkits on trustworthy AI (e.g. AI Fairness 360 (Bellamy et al., 2018), Adversarial Robustness Toolbox (Nicolae et al., 2018)) into a highly comprehensive testing framework.

Guidance on correct tool usage By providing usable high-level interfaces, toolkits reduce an entry barrier against employing (potentially highly complex) algorithms by a broad range of users. They should, additionally, support correct usage of the provided tools, e.g. by providing comprehensive guidelines. UQ360's documentation provides guidance on choosing an uncertainty method and on communicating uncertainty estimates to stakeholders, which we see as a step in the right direction. Additional means that support a more informed and effective use of UE include the attribution of uncertainty to concrete sources (i.e. how much uncertainty arises from data, hyperparameter tuning, initializations, etc.), instructions on reducing uncertainties as well as contrasting evaluation scores against one another and describing their properties. Visual and interactive interfaces can aid correct tool use further. To generally improve documentation practices, standards for documenting toolkits and their tools could be developed, compare for instance "model cards" (Mitchell et al., 2019).

Maintaining code quality Open-source seems to be a prerequisite for trustworthy implementations of evaluation standards and many of the reviewed toolkits principally place value on code quality as evident by supporting code contributions or employing automated testing tools. But, this is not always sufficient to ensure high quality and absence of critical bugs. Especially in the context of Trustworthy AI and safety-critical systems, which we see as a large application field for UE, this can become an issue. To further improve in that regard, it seems desirable to incentivize systematic code reviews, e.g. by introducing bug bounty programs for major toolkits and generally calling greater attention towards this topic (e.g. via dedicated workshops on code quality).

ACKNOWLEDGMENTS

The development of this publication was supported by the Ministry of Economic Affairs, Innovation, Digitalization and Energy of the State of North Rhine-Westphalia as part of the flagship project ZERTIFIZIERTE KI. The authors would like to thank the consortium for the successful cooperation.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Namita Agarwal and Saikat Das. Interpretable machine learning tools: A survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1528–1534. IEEE, 2020.

Ahmed Alaa and Mihaela Van Der Schaar. Discriminative jackknife: Quantifying uncertainty in deep learning via higher-order influence functions. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 165–174. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/alaa20a.html`.

M. Alam, M.D. Samad, L. Vidyaratne, A. Glandon, and K.M. Iftekharuddin. Survey on deep neural networks in speech and vision systems. *Neurocomputing*, 417:302–321, 2020. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2020.07.053. URL `https://www.sciencedirect.com/science/article/pii/S0925231220311619`.

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. GluonTS: Probabilistic and Neural Time Series Modeling in Python. *Journal of Machine Learning Research*, 21(116):1–6, 2020. URL `http://jmlr.org/papers/v21/19-820.html`.

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in neural information processing systems*, volume 33, pp. 14927–14937. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/aab085461de182608ee9f607f3f7d18f-Paper.pdf`.

Vijay Arya, Rachel K. E. Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Q. Vera Liao, Ronny Luss, Aleksandra Mojsilović, Sami Mourad, Pablo Pedemonte, Ramya Raghavendra, John Richards, Prasanna Sattigeri, Karthikeyan Shanmugam, Moninder Singh, Kush R. Varshney, Dennis Wei, and Yunfeng Zhang. Ai explainability 360: Hands-on tutorial. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, pp. 696, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450369367. doi: 10.1145/3351095.3375667. URL `https://doi.org/10.1145/3351095.3375667`.

Soheil Bahrampour, Naveen Ramakrishnan, Lukas Schott, and Mohak Shah. Comparative study of deep learning software frameworks. *arXiv: Learning*, 2015.

Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Kr. Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John T. Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *ArXiv*, abs/1810.01943, 2018.

William H. Beluch, Tim Genewein, Andreas Nurnberger, and Jan M. Kohler. The power of ensembles for active learning in image classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9368–9377, 2018. doi: 10.1109/CVPR.2018.00976.

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.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1613–1622. JMLR.org, 2015.

Miles Brundage, Shahar Avin, Jasmine Wang, Haydn Belfield, Gretchen Krueger, Gillian K. Hadfield, Heidy Khlaaf, Jingying Yang, Helen Toner, Ruth Fong, Tegan Maharaj, Pang Wei Koh, Sara Hooker, Jade Leung, Andrew Trask, Emma Bluemke, Jonathan Lebensbold, Cullen O'Keefe, Mark Koren, Theo Ryffel, J. B. Rubinovitz, Tamay Besiroglu, Federica Carugati, Jack Clark, Peter Eckersley, Sarah de Haas, Maritza L. Johnson, Ben Laurie, Alex Ingerman, Igor Krawczuk, Amanda Askell, Rosario Cammarota, Andrew J. Lohn, David Krueger, Charlotte Stix, Peter Henderson, Logan Graham, Carina E. A. Prunkl, Bianca Martin, Elizabeth Seger, Noa Zilberman, Se'an 'O h'Eigeartaigh, Frens Kroeger, Girish Sastry, Rebecca Kagan, Adrian Weller, Brian Tse, Elizabeth Barnes, Allan Dafoe, Paul Scharre, Ariel Herbert-Voss, Martijn Rasser, Shagun Sodhani, Carrick Flynn, Thomas Krendl Gilbert, Lisa Dyer, Saif Khan, Yoshua Bengio, and Markus Anderljung. Toward trustworthy AI development: Mechanisms for supporting verifiable claims. *ArXiv*, abs/2004.07213, 2020.

Junyi Chai, Hao Zeng, Anming Li, and Eric W.T. Ngai. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Machine Learning with Applications*, 6: 100134, 2021. ISSN 2666-8270. doi: https://doi.org/10.1016/j.mlwa.2021.100134. URL https://www.sciencedirect.com/science/article/pii/S2666827021000670.

Raja Chatila, Virginia Dignum, Michael Fisher, Fosca Giannotti, Katharina Morik, Stuart Russell, and Karen Yeung. Trustworthy ai. In *Reflections on Artificial Intelligence for Humanity*, pp. 13–39. Springer, 2021.

Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015. URL http://dblp.uni-trier.de/db/journals/corr/corr1512.html#ChenLLLWWXXZZ15.

Tongfei Chen, Jirí Navrátil, Vijay Iyengar, and Karthikeyan Shanmugam. Confidence scoring using whitebox meta-models with linear classifier probes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1467–1475. PMLR, 2019.

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.

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matthew D. Hoffman, and Rif A. Saurous. Tensorflow distributions. *CoRR*, abs/1711.10604, 2017. URL http://arxiv.org/abs/1711.10604.

Paul Druzhkov and Valentina Kustikova. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26:9–15, 01 2016. doi: 10.1134/S1054661816010065.

N. Durasov, T. Bagautdinov, P. Baque, and P. Fua. Masksembles for Uncertainty Estimation. In *CVPR*, 2021.

Vojtech Franc and Daniel Prusa. On discriminative learning of prediction uncertainty. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1963–1971. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/franc19a.html.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Maria-Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1050–1059. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/gal16.html.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1183–1192. JMLR.org, 2017.

Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 3369–3378. IEEE Computer Society, 2018. doi: 10/gfx44n.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.

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.

Ryan Giordano, William T. Stephenson, Runjing Liu, Michael I. Jordan, and Tamara Broderick. A Swiss army infinitesimal jackknife. In *AISTATS*, 2019.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Kyle Gorman and Steven Bedrick. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2786–2791, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1267. URL https://aclanthology.org/P19-1267.

Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in neural information processing systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017. URL http://proceedings.mlr.press/v70/guo17a.html.

Jian Guo, He He, Tong He, Leonard Lausen, Mu Li, Haibin Lin, Xingjian Shi, Chenguang Wang, Junyuan Xie, Sheng Zha, Aston Zhang, Hang Zhang, Zhi Zhang, Zhongyue Zhang, Shuai Zheng, and Yi Zhu. Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. *Journal of Machine Learning Research*, 21(23):1–7, 2020. URL http://jmlr.org/papers/v21/19-429.html.

Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. *A survey of deep learning techniques for medical diagnosis*, pp. 161–170. Springer, 2020.

Ali Harakeh, Michael Smart, and Steven L Waslander. BayesOD: A Bayesian approach for uncertainty estimation in deep object detectors. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 87–93. IEEE, 2020.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL http://www.jstor.org/stable/2334940.

James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, pp. 282–290, Arlington, Virginia, USA, 2013. AUAI Press.

José Miguel Hernández-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1861–1869. JMLR.org, 2015.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spacy: Industrial-strength natural language processing in python, 2020.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.

Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5574–5584, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/2650d6089a6d640c5e85b2b88265dc2b-Abstract.html.

Byol Kim, Chen Xu, and Rina Barber. Predictive inference is free with the jackknife+-after-bootstrap. *Advances in Neural Information Processing Systems*, 33:4138–4149, 2020.

Diederik P. Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Proceedings of the 28th international conference on neural information processing systems - volume 2*, NIPS'15, pp. 2575–2583. MIT Press, 2015. Number of pages: 9 Place: Montreal, Canada.

Ranganath Krishnan, Pi Esposito, and Mahesh Subedar. Bayesian-Torch: Bayesian neural network layers for uncertainty estimation. https://github.com/IntelLabs/bayesian-torch, January 2022. URL https://doi.org/10.5281/zenodo.5908307.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th international conference on machine learning*, volume 80 of *Proceedings of machine learning research*, pp. 2796–2804. PMLR, 2018. URL https://proceedings.mlr.press/v80/kuleshov18a.html. tex.pdf: http://proceedings.mlr.press/v80/kuleshov18a/kuleshov18a.pdf.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6402–6413, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

Sara Landset, Taghi Khoshgoftaar, Aaron Richter, and Tawfiq Hasanin. A survey of open source tools for machine learning with big data in the hadoop ecosystem. *Journal of Big Data*, 2, 11 2015. doi: 10.1186/s40537-015-0032-1.

Chunyuan Li, Changyou Chen, David E. Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In Dale Schuurmans and Michael P. Wellman (eds.), *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pp. 1788–1794. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11835.

Haochen Liu, Yiqi Wang, Wenqi Fan, Xiaorui Liu, Yaxin Li, Shaili Jain, Yunhao Liu, Anil K Jain, and Jiliang Tang. Trustworthy AI: A computational perspective. *arXiv preprint arXiv:2107.06641*, 2021.

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.

Jan Maces and Repository Contributors. Github repository of keras-adf, 2019. URL https://github.com/jmaces/keras-adf.

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

Niall O' Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Adolfo Velasco-Hernández, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *CVC*, 2019.

Eric Meissner, Zhenwen Dai, Tom Diethe, and Repository Contributors. Github repository of MXFusion, 2019. URL https://github.com/amzn/MXFusion.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114. URL https://doi.org/10.1063/1.1699114.

Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *CVPR*, pp. 12677–12686. Computer Vision Foundation / IEEE, 2019. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2019.html#MeyerLKVW19.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pp. 220–229, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450361255. doi: 10.1145/3287560.3287596. URL https://doi.org/10.1145/3287560.3287596.

Jacob Montiel, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine, Heitor Murilo Gomes, Jesse Read, Talel Abdessalem, and Albert Bifet. River: machine learning for streaming data in python, 2020.

Jirí Navrátil, Benjamin Elder, Matthew Arnold, Soumya Shubhra Ghosh, and Prasanna Sattigeri. Uncertainty characteristics curves: A systematic assessment of prediction intervals. *ArXiv*, abs/2106.00858, 2021.

Radford M Neal et al. MCMC using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

Giang Nguyen, Stefan Dlugolinsky, Martin Bobák, Viet Tran, Álvaro López García, Ignacio Heredia, Peter Malík, and Ladislav Hluch? Machine learning and deep learning frameworks and libraries for large-scale data mining: A survey. *Artif. Intell. Rev.*, 52(1):77–124, jun 2019. ISSN 0269-2821. doi: 10.1007/s10462-018-09679-z. URL https://doi.org/10.1007/s10462-018-09679-z.

Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial robustness toolbox v1.2.0. *CoRR*, 1807.01069, 2018. URL https://arxiv.org/pdf/1807.01069.

D.A. Nix and A.S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pp. 55–60 vol.1, 1994. doi: 10/fth5jt.

Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz E Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota. Practical deep learning with Bayesian principles. *Advances in neural information processing systems*, 32, 2019.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Proceedings of the 33rd international conference on neural information processing systems*. Curran Associates Inc., 2019. Number of pages: 12 tex.articleno: 1254.

Zoe Papakipos and Joanna Bitton. Augly: Data augmentations for robustness, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006. ISBN 026218253X.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd international conference on international conference on machine learning - volume 37*, ICML'15, pp. 1530–1538. JMLR.org, 2015. Place: Lille, France Number of pages: 9.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Skdvd2xAZ.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in Neural Information Processing Systems*, 31, 2018.

Jiaxin Shi, Jianfei. Chen, Jun Zhu, Shengyang Sun, Yucen Luo, Yihong Gu, and Yuhao Zhou. ZhuSuan: A library for Bayesian deep learning. *arXiv preprint arXiv:1709.05870*, 2017.

Ian Sommerville. *Software Engineering*. Addison-Wesley, Harlow, England, 9 edition, 2010. ISBN 978-0-13-703515-1.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Jayaraman J. Thiagarajan, Bindya Venkatesh, Prasanna Sattigeri, and Peer-Timo Bremer. Building calibrated deep models via uncertainty matching with auxiliary interval predictors. In *AAAI*, 2020.

Amirsina Torfi, Rouzbeh A Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A Fox. Natural language processing advancements by deep learning: A survey. *arXiv preprint arXiv:2003.01200*, 2020.

Dustin Tran, Rajesh Ranganath, and David M Blei. The variational gaussian process. In *4th International Conference on Learning Representations, ICLR 2016*, 2016.

Dustin Tran, Matthew D. Hoffman, Dave Moore, Christopher Suter, Srinivas Vasudevan, Alexey Radul, Matthew Johnson, and Rif A. Saurous. Simple, distributed, and accelerated probabilistic programming. In *Neural Information Processing Systems*, 2018.

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An introduction to probabilistic programming. *arXiv preprint arXiv:1809.10756*, 2018.

Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 550–564, 2018.

Zhaobin Wang, Ke Liu, Jian Li, Ying Zhu, and Yaonan Zhang. Various frameworks and libraries of machine learning and deep learning: A survey. *Archives of Computational Methods in Engineering*, 02 2019. doi: 10.1007/s11831-018-09312-w.

Michael Weiss and Paolo Tonella. Uncertainty-wizard: Fast and user-friendly neural network uncertainty quantification. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, pp. 436–441. IEEE, 2021.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2019.

Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE Access*, 8:58443–58469, 2020. doi: 10.1109/ACCESS.2020.2983149.

Jan Zacharias, Michael Barz, and Daniel Sonntag. A survey on deep learning toolkits and libraries for intelligent user interfaces. *CoRR*, abs/1803.04818, 2018. URL `http://arxiv.org/abs/1803.04818`.

Shengjia Zhao, Tengyu Ma, and Stefano Ermon. Individual calibration with randomized forecasting. In *International Conference on Machine Learning*, pp. 11387–11397. PMLR, 2020.

APPENDIX

6.1 DETAILED OVERVIEW ON UNCERTAINTY MODELING AND ASSESSMENT

The following section gives an literature overview over uncertainty estimation in DL, describing sources of uncertainty, approaches and applications for uncertainty estimation and assessment. It provides further details to the paragraph on "elements of uncertainty estimation" in section 2.

Neural networks are subject to several kinds of uncertainties (Hüllermeier & Waegeman, 2021). We generally do not know whether the chosen neural network model is optimal for the given task. There are also approximation errors caused by the optimization procedure (e.g. due to (suboptimal) hyperparameter choices, random initializations, lack of data). Such uncertainties that arise from lack of knowledge about the optimal model for a given task are subsumed under the term *epistemic uncertainty*, which is theoretically reducible e.g. by selecting better models or supplying more training data. Additionally, there is irreducible *aleatoric uncertainty*, which is stochasticity in the outcome of the training procedure caused by inherently random factors, such as most importantly the data (e.g. sensor noise, mislabeled inputs or imperfect information). Besides for quantifying prediction and estimation variance, uncertainty estimation has also been explored for auxiliary tasks, such as detecting out-of-distribution inputs (Ovadia et al., 2019; Vyas et al., 2018), active learning (Gal et al., 2017; Beluch et al., 2018), detection of adversarial examples (Ritter et al., 2018; Amini et al., 2020) and continual learning (Osawa et al., 2019; Nguyen et al., 2018). The main tool for deep uncertainty estimation are probability distributions, typically over network outputs. The expected value serves as main network prediction and uncertainty is quantified in terms of variance, quantiles or entropy. Approaches for uncertainty estimation include the following (see Gawlikowski et al. (2021) for a comprehensive survey):

- In parametric likelihood (PL) methods, the network outputs distributional parameters instead of point estimates and is trained via likelihood optimization (Nix & Weigend, 1994; Amini et al., 2020; Sensoy et al., 2018). They are commonly used to estimate aleatoric uncertainty and are combined with other methods to incorporate epistemic uncertainty in addition (Kendall & Gal, 2017).
- Bayesian neural networks (BNNs) extend the parametric likelihood approach and incorporate (epistemic) uncertainty on network parameters by estimating a posterior and predictive distribution based on Bayes rule. Different posterior estimation procedures have been considered for DL, including variational inference (VI-BNNs, Graves (2011); Blundell et al. (2015); Rezende & Mohamed (2015)) with dropout sampling (Gal & Ghahramani, 2016; Kingma et al., 2015) being an important variant, expectation propagation (Hernández-Lobato & Adams, 2015; Gast & Roth, 2018), Markov-chain Monte Carlo sampling (MCMC, Neal et al. (2011); Welling & Teh (2011); Li et al. (2016)) or the Laplace approximation (MacKay, 1992; Ritter et al., 2018). Gaussian processes (GP, Rasmussen & Williams (2006)) are a related Bayesian approach that incorporate uncertainty in a more general function space, instead of the network's parameter space. Standard GPs are computationally infeasible for large-scale models. However, there are scalable variants (Hensman et al., 2013) that can even be used as a differentiable network component (Tran et al., 2016) and as a building block for UE in deep learning models (Liu et al., 2020).
- Frequentist approaches directly leverage a combination of different models for uncertainty estimation. Common approaches are based on ensembling (Lakshminarayanan et al., 2017; Wen et al., 2019; Durasov et al., 2021)) or the jackknife method (Giordano et al., 2019; Alaa & Van Der Schaar, 2020; Kim et al., 2020).

Uncertainty methods can be further categorized into (i) intrinsic methods, that integrate the uncertainty estimation directly into the architecture or training procedure (e.g. dropout layers, ensemble training), (ii) post-hoc methods, that extend standard deep learning models (e.g. Laplace approximation, infinitesimal jackknife, surrogate models (Chen et al., 2019)) and (iii) recalibration methods, that seek to improve existing uncertainty estimates (Guo et al., 2017; Kuleshov et al., 2018; Navrátil et al., 2021).

We now discuss several techniques for assessing the quality of uncertainty estimates.

- (Proper) Scoring rules are point-wise metrics that measure for the fit of a predicted distribution to a ground-truth value and are typically evaluated on held-out validation datasets. Common scoring rules include the brier score, negative log-likelihood (NLL), continuous ranked probability score (CRPS) or the interval score (see Gneiting & Raftery (2007)).

- Confidence calibration measures the alignment of confidence estimates with a validation dataset (e.g. a 90% confidence interval should contain 90% correct labels). Calibration can be visually assessed via reliability diagrams or curves (Guo et al., 2017; Navrátil et al., 2021) that plot confidence levels against a correctness metric (e.g. classification accuracy). Quantitative metrics typically consider the area under the aforementioned curves, most notably the expected calibration error (ECE) (Guo et al., 2017; Kuleshov et al., 2018). Adversarial group calibration (Zhao et al., 2020) is a more strict variant of calibration assessment that considers alignment of confidence estimates with every subset of the dataset, instead of only the whole dataset.
- Uncertainty estimates can also be qualitatively assessed, e.g. by visually inspecting predicted distributions, confidence intervals against ground-truth values. Considering the average uncertainty across a dataset (also called sharpness) is crucial in calibration assessments as argued by Kuleshov et al. (2018).
- Auxiliary scores measure the performance of uncertainty estimates in auxiliary tasks such as out-of-distribution detection. Standard evaluation methods for binary threshold classifiers (e.g. separating true from false detections) include precision-recall and receiver operating curves for visual assessment or computing average precision or AUCROC scores. Another approach is to compare histograms of uncertainty estimates (e.g. variance or predictive entropy) on inputs from each class (e.g. in-distribution or out-of-distribution), as done in Ovadia et al. (2019).

6.2 Comparative analysis of the selected uncertainty toolkits

This section provides 2 Tables with additional information about the toolkits. Table 1 lists all toolkits with their license, intended purpose and the exact version that has been considered in this survey. Table 2 provides a concise summary of the results of our analysis from section 5.

Table 1: General information on the examined uncertainty estimation toolkits. All toolkits use python as programming language. We provide the github commit number in cases where no version number is given.

UE toolkit	**Developer**	**License**	**Version/ commit**	**Base libraries/ frameworks**	**Toolkit type**
TFP	Dillon et al. (2017) (Google Brain)	Apache-2.0	`0.15.0`	Tensorflow/ Keras	PPL
Pyro	Bingham et al. (2019) (Uber AI Labs)	Apache-2.0	`1.8.0`	Pytorch, JAX	PPL
MXF	Meissner et al. (2019) (Amazon Web Services)	Apache-2.0	`0.3.1`	MXNet	PPL
ZS	Shi et al. (2017)	MIT	`4386b2a`	Tensorflow	PPL
ED2	Tran et al. (2018) (Google Brain)	Apache-2.0	`f420d83`	Tensorflow/ Keras	PPL
GTS	Alexandrov et al. (2020) (Amazon Web Services)	Apache-2.0	`0.9.0`	MXNet, Pytorch	Time series
UQ360	Ghosh et al. (2021) (IBM Research)	Apache-2.0	`2378bfa`	Pytorch	Dedicated UE
UT	Chung et al. (2021)	MIT	`v0.1.0`	Scipy, Scikit-learn	Dedicated UE
UW	Weiss & Tonella (2021)	MIT	`v0.2.0`	Tensorflow/ Keras	Dedicated UE
BT	Krishnan et al. (2022) (Intel Labs)	BSD3	`99876f3`	Pytorch	Dedicated UE
KADF	Maces & Contributors (2019)	MIT	`19.1.0`	Tensorflow/ Keras	Dedicated UE

Table 2: Analysis summary of the uncertainty toolkits selected in section 3. The top table considers all toolkits with respect to our core criteria (cf. section 4). The bottom table provides additional criteria and considers the three best-performing toolkits with respect to the core criteria. "Standard statistics" refers to general (sample) measures such as variance, quantiles or correlation.

UE toolkit	**Range of supported uncertainty methods**	**Range of supported evaluation techniques**	**Range of supported architectures and data structures**
TFP	• intrinsic (PL, VI-BNN, GP, MCMC) • high range of supported distributions	• broad range of standard statistics • scoring rules (NLL, Brier) • classification calibration (ECE)	• regression, classification, sequential
Pyro	• intrinsic (PL, VI-BNN, GP, MCMC) • high range of supported distributions	• standard statistics • scoring rules (NLL, CRPS)	• regression, classification, sequential
ED2	• intrinsic (PL, VI-BNN, GP, MCMC) • high range of supported distributions	• narrow range of scoring rules (NLL)	• regression, classification, sequential
ZS	• intrinsic (PL, VI-BNN, MCMC) • high range of supported distributions	• narrow range of standard statistics • narrow range of scoring rules (NLL)	• regression, classification, sequential
MXF	• intrinsic (PL, VI-BNN, GP)	[none]	• regression, classification
GTS	• intrinsic (PL)	• scoring rules • plotting function for conf. intervals	• sequential
UQ360	• intrinsic (PL, VI-BNN, ensembling) • post-hoc (jackknife-based, surrogate models) • recalibration	• scoring rules • calibration assessment (ECE) • plotting functions	• regression, classification • tabular inputs • low flexibility in intrinsic methods
UT	• basic recalibration	• broad range of scoring rules • calibration assessment (ECE, AGC) • plotting functions	• 1D-regression
UW	• intrinsic (ensembling, dropout)	[none]	• regression, classification • custom uncertainty quantifiers
BT	• intrinsic (VI-BNN)	[none]	• regression, classification, recurrent
KADF	• intrinsic (VI-ADF)	[none]	• regression, classification

Abbreviations: PL: parametric likelihood, VI-BNN: variational inference-based Bayesian neural networks, GP: Gaussian processes, VI-ADF: assumed density filtering Bayesian neural network (a variant of expectation propagation), MCMC: Markov chain Monte Carlo-based posterior sampling methods, AGC: adversarial group calibration (Zhao et al., 2020)

UE toolkit	**Integration with DL frameworks**	**Software quality**
TFP	• part of tensorflow ecosystem • high integration with keras model API	• high code quality • continuous testing • well-documented
Pyro	• pytorch-based implementation • models/inference procedures can be encapsulated as torch modules	• high code quality • continuous testing • well-documented
UQ360	• pytorch-based implementation • custom pytorch models can be passed to some modules	• simple, easy-to-use sklearn-esque API • well-documented, provides user guidance on uncertainty methods/metrics