A Simple Direct Uncertainty Quantification Technique Based on Machine Learning Regression

Katherine E. Brown and Douglas A. Talbert
Department of Computer Science
Tennessee Technological University
1 William L. Jones Drive
Cookeville, TN 38505

Abstract
Epistemic uncertainty quantification provides useful insight into both deep and shallow neural networks’ understanding of the relationships between their training distributions and unseen instances and can serve as an estimate of classification confidence. Bayesian-based approaches have been shown to quantify this relationship better than softmax probabilities. Those approaches to uncertainty quantification, however, require multiple Monte-Carlo samplings of a neural network, augmenting the neural network to learn distributions for its weights, or utilizing an ensemble of neural networks. Current research has yielded several techniques that estimate uncertainty deterministically but tend to struggle in independent evaluation using real-world data and tasks compared to stochastic, or variational, techniques such as dropout (Postels et al. 2021). This work presents a simple (but effective) regression-based technique for modeling the epistemic uncertainty produced by dropout (Gal and Ghahramani 2016).

Motivation
There are several motivating factors as to why we want to estimate uncertainty produced by Bayesian dropout. First, distilling Bayesian dropout into a single forward-pass helps minimize the computational overhead of performing the multiple inferences required to measure uncertainty while still utilizing this extensively verified technique. The computational overhead of “variational inference” techniques such as Bayesian dropout has motivated research in Direct Uncertainty Measures (DUM) that calculates uncertainty values deterministically with a single inference of an algorithm. Postels et al. notes that recent advances in DUM techniques are less robust in real-world out-of-distribution detection compared to traditional variational inferences techniques such as dropout (Postels et al. 2021). Current DUM techniques do not estimate an already validated uncertainty quantification technique but generate their own uncertainty value (Jain et al. 2021; Van Amersfoort et al. 2020). In situations where Bayesian dropout may be more effective as an uncertainty quantification technique, it is better suited to use an explanation from a distillation of Bayesian uncertainty rather than using an unverified technique altogether.

Background

Uncertainty Quantification
Uncertainty quantification is a growing field of study that provides information related to the reliability of neural networks in safety-critical tasks (Begoli, Bhattacharya, and Kusnezov 2019) and can provide insight into the robustness of the model given its training distribution or the data input to the model for inference (Gal and Ghahramani 2016; Kendall and Gal 2017). In deep learning, two types of uncertainty are considered, epistemic and aleatoric. Aleatoric uncertainty is beyond the scope of this work.
Epistemic uncertainty is a measure of how confident the model is inferring for a data point given the model’s training
data (Gal and Ghahramani 2016). This encompasses suitability of the training data at representing the decision space and the suitability of the model at determining classification boundaries within the training data (i.e., is the model complex enough?).

A common and accessible technique to measure epistemic uncertainty is Bayesian dropout (Gal and Ghahramani 2016). Dropout is a regularization technique that prevents overfitting in a deep neural network by zero-ing out, or removing, weights. When a neural network is sampled several times, this creates an ensemble effect by which epistemic uncertainty can be measured. Bayesian dropout has been applied in various critical fields (Leibig et al. 2017; Brown and Talbert 2019; Michelmore, Kwiatkowska, and Gal 2018).

Uncertainty quantification has been used to improve machine learning performance when paired with a human expert (Leibig et al. 2017; Brown and Talbert 2019). This task uses an uncertainty quantification technique to identify predictions that are most likely to be wrong. These predictions are referred to as uncertainty rejection calibration. Brown and Talbert (Brown and Talbert 2019) demonstrated that Bayesian dropout identifies likely incorrect predictions better than softmax probabilities.

**Direct Uncertainty Measures**

One shortcoming of Bayesian dropout (and other “variational inference” techniques) is the computational overhead required. To ascertain uncertainty, multiple inferences are typically required leading to a high computational cost for inferring uncertainty. This has led to rising studies of Direct Uncertainty Methods (DUM). DUM techniques attempt to ascertain uncertainty using less computational complexity, typically through a single inference of a proxy algorithm.

A comprehensive survey of current DUM techniques is given in (Postels et al. 2021); however, we briefly summarize selected techniques. First, Van Amersfoort et al. used radial basis function networks to identify when data cannot be easily clustered among known classes (Van Amersfoort et al. 2020). Jain et al. models epistemic uncertainty by modeling predictive error. Postels et al. notes that such techniques struggle with potentially erroneous datapoints in real-world scenarios with a significant drop in performance compared to variational inference techniques including dropout.

**Methodology**

In this section, we present our technique for approximating dropout-based epistemic uncertainty using machine learning regression.

**Dropout Uncertainty Estimation**

Let $f$ be a machine learning regression algorithm, either multi-class regressor or multiple single-class regressors. To infer epistemic uncertainty using $f$, we use the original input features into the classification neural network and the output probability vector of predictions from the classification neural network. Thus, for an $m$-class classification problem, if $\vec{x} \in \mathbb{R}^n$ is the input vector of $n$ features, $\vec{p} \in [0, 1]^m$ the vector of output probabilities from classification network, then the vector of uncertainty estimates $\vec{u} \in \mathbb{R}^m$ is calculated as $\vec{u} = f(\vec{x}, \vec{p})$. We include $\vec{p}$ since uncertainty estimates should be conditioned on both the input data and the classification probabilities produced by the classification neural network.

Successfully training the uncertainty regression model requires augmenting the classification neural network’s training data. When training the classification neural network, we use a classification training set to determine model weights through backpropagation, a classification validation set for model selection, and a test set. The regression-based uncertainty estimator is trained on the regression training set and evaluated using the test set. The regression training set combines both the classification training set and the classification validation set along with their classification probability vectors. These are then labeled with their dropout uncertainty. This enables the regression model to learn from examples outside of the classification training set’s data distribution. The regression test set uses the same data as the classification test set.

**Experimental Methodology**

**Datasets and Classification Models**

We use three datasets in our experiments. Two datasets (Adult Income and MNIST) are classic benchmarking datasets for machine learning algorithms, and the third dataset is real-world critical-care dataset. We detail each dataset below as well as the classification models trained for their respective classification tasks.

**Adult Income** The first dataset is a modified version of the Adult Income dataset from the University of California at Irvine (UCI) Machine Learning Repository (Dua and Graff 2017). This dataset reflects a binary classification task to predict whether a person has an income greater than or less than $50,000. Features in this dataset include population demographics including age, gender, race, and marital status as well as occupation information including level of education, occupation type, work class (e.g., government, private, self-employed), and hours per week. Categorical features are represented using one-hot encoding. Continuous features are normalized using z-values. The hidden layer architecture for the classification task consists of a single hidden layer with a width of 128 neurons and is trained for 50 epochs.

**MNIST** The second dataset is the MNIST dataset (Deng 2012), which contains images of size $28 \times 28$ that depict handwritten numerical digits between 0 and 9. The multi-class classification task is to determine which digit is depicted per image. The features used by the classification neural network are the pixels of the $28 \times 28$ image that are converted into a one-dimensional vector of length 784. The pixel values are normalized by dividing each by 255 (the maximum pixel value). The hidden layer architecture for the classification task consists of 2 hidden layers of width 75 neurons each and is trained for 100 epochs.
The final dataset is a subset of a trauma registry from a Level 1 Trauma Center that uses 32 features to determine if a patient has an Injury Severity Score (ISS) of at least 15. An ISS ≥ 15 indicates the patient is severely injured and should be triaged as such (Sasser and others 2012). Features include physical parameters (e.g., systolic/diastolic blood pressures, heart rate, Glasgow Coma Scale score), anatomical criteria, mechanism of injury, age, and multiple computed injury scores (e.g., Revised Trauma Score and the Air Medical Prehospital Transport Score). Categorical features are represented using one-hot encoding. Continuous features are normalized using z-values. The hidden layer architecture for the classification task consists of two hidden layers with 75 neurons each and is trained for 100 epochs.

For each classification task, we train a fully-connected neural network to learn a predictive model. Each dataset is split into 10 cross-validation folds with 10% of the training data in each fold used as a validation set during training of the neural network. Hidden layers use the Rectified Linear Unit (ReLU) as their activation functions, and each output layer uses the softmax function as its activation function. Dropout is applied before each hidden layer to calculate epistemic uncertainty (Gal and Ghahramani 2016). The ADAM optimization algorithm (Kingma and Ba 2014) is used for the MNIST dataset while the RMSprop optimization function (Hinton, Srivastava, and Swersky 2012) is used for the Adult Income and Trauma Triage Registry datasets. We use the Keras API (Chollet and others 2015) with Tensorflow (Abadi and others 2015) to implement all neural networks in this work.

We use Keras and Tensorflow (Chollet and others 2015; Abadi and others 2015) to implement the regression neural networks. For each dataset, we consider a network with hidden layers of width 512, 255, 128, 64 neurons. We used mean squared error as the regression loss function and the ADAM algorithm as the optimization function (Kingma and Ba 2014). Each network is trained for 50 epochs.

To evaluate uncertainty estimator effectiveness, we consider the following experiments. First, we evaluate our uncertainty estimator using following regression metrics: mean absolute error of the estimated uncertainty compared to the “true” dropout uncertainty and the $R^2$ coefficient of the estimated uncertainty compared to the “true” dropout uncertainty. Mean absolute error is defined as the sum of the absolute value of the difference between the true and predicted values divided by the total number of data points (De Myttenaere et al. 2016). The $R^2$ coefficient measures how predictable the dependent variable (e.g., predicted uncertainty) given the independent variables (e.g., input features and predicted probability variables) (Di Bucchianico 2008). These measures demonstrate the effectiveness of the regression-based estimation of Bayesian dropout estimation uncertainty.

To measure the efficacy of our estimation technique as an uncertainty measure, we compare the rejection classification results using both true dropout uncertainty and estimated uncertainty. Rejection classification refers to the shift in accuracy (or other desired classification metric) as the most “uncertain” predictions are removed from the calculation of the classification metric.

We perform these experiments using both multiple, single output regression models (“multi-single”) that estimate uncertainty per class as well as a single multi-output regression model that estimates uncertainty for all classes in a single model (“multi-output”).

### Results

#### Uncertainty Estimator Effectiveness

Table 1 presents traditional regression metrics for estimating uncertainty on unseen evaluation data. Average $R^2$ is the average value of the $R^2$ correlation coefficient between true uncertainty and predicted uncertainty generated from our technique. We also report mean absolute error between true uncertainty and predicted uncertainty generated from our technique.

| Dataset      | Uncertainty Output Type | Average $R^2$ | Mean Absolute Error |
|--------------|-------------------------|---------------|---------------------|
| Adult Income | Multi-Single            | 0.964         | 0.012               |
| Adult Income | Multi-Output            | 0.966         | 0.011               |
| Trauma Triage| Multi-Single            | 0.961         | 0.011               |
| Trauma Triage| Multi-Output            | 0.963         | 0.010               |
| MNIST        | Multi-Single            | 0.730         | 0.022               |
| MNIST        | Multi-Output            | 0.665         | 0.022               |

Figure 1 presents the uncertainty calibration results for each dataset, estimated uncertainty, and dropout uncertainty. These plots show the change in accuracy as the most uncertain predictions are removed from accuracy calculations.

![Figure 1: Uncertainty calibration of proposed technique compared to dropout uncertainty](image-url)
Discussion

Regression Metrics

Table 1 presents the $R^2$ correlation coefficient for our inferred uncertainty compared to the “true” dropout uncertainty. For the Adult Income dataset, we find we are able to achieve an $R^2$ value of at least 0.964. For the Trauma Triage dataset, we achieve an $R^2$ value of at least 0.961. This implies for Adult Income and Trauma Triage, regression techniques have a close goodness-of-fit to dropout uncertainty. Moreover, mean absolute errors for these techniques are less that 0.02 for all regression techniques for both datasets.

For MNIST (see Table 1), the regression estimators achieve average $R^2$ values of at least 0.665 for a single, multi-class network and an $R^2$ value of 0.730 when uncertainty is estimated using one neural network per class.

We suspect that this difference in performance between the tabular datasets (Adult Income and Trauma Triage) and MNIST is a result of the datasets’ dimensionality. MNIST is an image dataset with 784 input pixels. Since 10 classes can be predicted from MNIST data, this results in 794 input features into the epistemic uncertainty inference models. Future work includes investigating the effect of a larger feature space on performance of uncertainty regression and possible techniques to mitigate the loss in performance.

Uncertainty Rejection Calibration

Our next experiment evaluates how well an uncertainty measure can identify potentially incorrect predictions. Several works in the existing literature (Gal and Ghahramani 2016; Leibig et al. 2017; Brown and Talbert 2019) demonstrate the effectiveness of dropout uncertainty quantification at identifying likely incorrect instances and the benefit of using such a technique in medicine.

Figure 1 demonstrates how accuracy changes as the most uncertain data points are removed for each dataset. For Adult Income and Trauma Triage, each regression technique results in similar changes, as compared to the “correct” dropout uncertainty, in accuracy as more uncertain data points are removed. For MNIST, the regression algorithms demonstrate similar overall behavior but with a slight (< 1 %) degradation in classification accuracy.

Conclusion

In this work, we present a technique that allows epistemic uncertainty to be estimated through regression-based machine learning algorithms. This technique requires fewer machine learning inferences to yield uncertainty estimates compared to classic, but resource-expensive, techniques such as dropout.

This work has several avenues through which it can be extended. First, we find that simpler, less-expensive machine learning techniques perform poorly on high-dimensional data such as images. Future work includes uncovering why this occurs and possible solutions. Moreover, this technique can also be extended to other measures of uncertainty such as aleatoric uncertainty, to provide a more complete picture of the uncertainty associated with a prediction. Finally, we would like to explore how XAI could be applied to epistemic uncertainty estimators to determine if it can provide insight into causes of epistemic uncertainty.

References

Abadi, M., et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Begoli, E.; Bhattacharya, T.; and Kusnezhov, D. 2019. The Need for Uncertainty Quantification in Machine-Assisted Medical Decision Making. Nature Machine Intelligence 1(1):20–23.

Brown, K. E., and Talbert, D. A. 2019. Estimating uncertainty in deep image classification. In AMIA.

Chollet, F., et al. 2015. Keras.

De Myttenaere, A.; Golden, B.; Le Grand, B.; and Rossi, F. 2016. Mean absolute percentage error for regression models. Neurocomputing 192:38–48.

Deng, L. 2012. The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine 29(6):141–142.

Di Bucchianico, A. 2008. Coefficient of determination ($r^2$). Encyclopedia of Statistics in Quality and Reliability 1.

Dua, D., and Graff, C. 2017. UCI Machine Learning Repository.

Gal, Y., and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of The 33rd International Conference on Machine Learning, 1050–1059.

Hinton, G.; Srivastava, N.; and Swersky, K. 2012. Neural Networks for Machine Learning: Overview of mini-batch gradient descent.

Jain, M.; Lahlou, S.; Nekoei, H.; Butoi, V.; Bertin, P.; Rector-Brooks, J.; Korablyov, M.; and Bengio, Y. 2021. Deup: Direct epistemic uncertainty prediction. arXiv preprint arXiv:2102.08501.

Kendall, A., and Gal, Y. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In Advances in neural information processing systems, 5574–5584.

Kingma, D. P., and Ba, J. 2014. Adam: A Method for Stochastic Optimization.

Leibig, C.; Allken, V.; Ayhan, M. S.; Berens, P.; and Wahl, S. 2017. Leveraging uncertainty information from deep neural networks for disease detection. Scientific reports 7(1):1–14.

Michelmore, R.; Kwiatkowska, M.; and Gal, Y. 2018. Evaluating uncertainty quantification in end-to-end autonomous driving control. arXiv preprint arXiv:1811.06817.

Postels, J.; Segu, M.; Sun, T.; Van Gool, L.; Yu, F.; and Tombari, F. 2021. On the practicality of deterministic epistemic uncertainty. arXiv preprint arXiv:2107.00649.

Sasser, S. M., et al. 2012. Guidelines for Field Triage of Injured Patients: Recommendations of the National Expert Panel on Field Triage, 2011. Morbidity and Mortality Weekly Report: Recommendations and Reports 61(1):1–20.

Van Amersfoort, J.; Smith, L.; Teh, Y. W.; and Gal, Y. 2020. Uncertainty estimation using a single deep deterministic neural network. In International conference on machine learning, 9690–9700. PMLR.