UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

UNIVERSITY HEIDELBERG

SEMINAR REPORT

EXPLAINABLE MACHINE LEARNING

What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?

Author:
Sebastian GRUBER

Supervisor:
PD. Dr. Ullrich KÖTHE

31st July 2018

Contents

1 Introduction

Being able to capture what a model does not know, has become increasingly important for many applications of machine learning. Knowing if your model is under- or overconfident can help reasoning about it and the dataset.
Deep learning algorithms are now able to learn powerful representations, mapping complex data structures to an array of outputs. Making sure that these mappings are however correct and will not falsely be assumed to be, is very important. Since todays deep learning algorithms are usually unable to quantify their uncertainty, fatal predictions can be made. A striking case in which this lead to disastrous consequences occurred in May 2016, where the first fatality from an assisted driving system was caused. A white trailer was confused with the bright sky in the background, resulting in the system not engaging in emergency braking. In this case, being able to asses uncertainty to the observation could have lead to a better and safer decision by the model. In order to achieve state-of-the-art performance, often deep learning is used, which usually cannot represent or learn uncertainty. For both regression and classification settings, which roughly cover most vision applications, uncertainty can be captured with Bayesian deep learning.

2 About Bayesian Neural Networks

In Bayesian statistics, evidence about the true state of the world is conveyed in degrees of belief. Combining Bayesian statistics and deep learning handily comes with a measure of uncertainty for the networks predictions. Bayesian deep learning replaces the deterministic weights of a model with distributions, while keeping the bias parameter, that normal neural networks have. Instead of optimizing the model weights directly, one has to average over all possible weights (referred to as marginalization).
So far Bayesian deep learning models were not popular because of the much greater amount of parameters to optimize. However, with increasing interest in being able to comprehend complex models and computing an uncertainty measure alongside the model's predictions, it has become more popular and new techniques are being developed. Some of the challenges will become apparent in chapter 3.2, when epistemic uncertainty is discussed, which in the paper requires expensive Monte Carlo sampling to obtain.

3 Types of Uncertainties

Uncertainty refers to having limited knowledge in a situation where it is not possible to precisely grasp an existing state, more than one possible outcome or a prediction about a future state. Furthermore, uncertainty also includes ambiguity, uncertainty related to human concepts and definitions which are not objective facts.
There are two main types of uncertainty and it is important to understand which type is required in which situation, as well as understanding why both types are necessary to comprehensively predict uncertainty. These two types of uncertainties are called aleatoric and epistemic uncertainty. First, they are grasped as different concepts based on related work to this paper and later are included into one model.

3.1 Aleatoric Uncertainty

Aleatoric uncertainty captures uncertainty which cannot be explained with the data. For example, aleatoric uncertainty refers to occlusions, lack of visual features or over-exposure (see Figure 1). Only in theory it can be explained away if all necessary explanatory variables are known with indefinite precision. This makes it very important for real-time applications like the earlier mentioned assisted driving system. In general, it is most impactful in large data situations, where epistemic uncertainty usually plays a minor role.

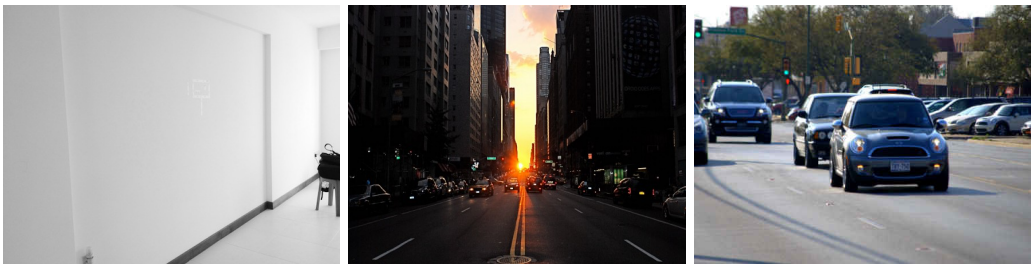


Figure 1: Examples for aleatoric uncertainty in imagery: A white wall lacking visual features; A scene at sunset with over- & underexposure; A car being partially covered (occlusion).

Aleatoric uncertainty can further be divided into homoscedastic and heteroscedastic uncertainty. As the name implies, homoscedastic uncertainty describes uncertainty which is not dependent on the input data, it is a quantity which stays consistent for all data points. Heteroscedastic uncertainty on the other hand depends

on the input data and can be predicted as a model output.
In the case of most computer vision settings, heteroscedastic uncertainty is more important, therefore it will be in the focus of this report.
For non Bayesian networks, the inherent noise parameter is often included in the model's weight decay, but it can be learned if made data dependent:

$$\mathcal{L}NN(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2\sigma(\mathbf{x}_i)^2}||\mathbf{y}_i\mathbf{f}(\mathbf{x}_i)||^2 + \frac{1}{2}\log\sigma(\mathbf{x}_i)^2 \tag{1}$$

In this case, σ captures heteroscedastic uncertainty for each pixel, segment or image, depending on the task. Also weight decay parameterized by λ may be added to this loss function.

3.1.1 Aleatoric Uncertainty as Learned Loss Attenuation

Having the network predict uncertainty also comes with another effect. It is able to temper with the loss by the means of σ^2, which is dependent on the data. In consequence, the network will learn to adapt to noisy data, inputs for which the network predicts high uncertainty will be attenuated in the loss function. Therefore, erroneous labels will also have a smaller effect on the loss. This process acts in the same way as a intelligent robust regression function would.
The variance σ^2 appears twice in the loss function in order to achieve a certain balance. The $\log\sigma^2$ term prevents the model from assigning high uncertainty to all points, effectively ignoring the data. The σ^{-2} term on the other hand causes a high loss for a small σ. The model may ignore data, but in turn is penalized for that.
The paper points out that this is in fact a consequence of the probabilistic interpretation of the model and not an ad-hoc construction.

3.2 Epistemic Uncertainty

Epistemic uncertainty is also commonly refered to as model uncertainty. It measures what the model does not know due to a lack of training. It can mostly be explained away given enough training data and is, in practice, for large data settings second to aleatoric uncertainty. Epistemic uncertainty is therefore especially important for situations with little training data and safety-critical applications, since it is necessary to understand and acknowledge samples different from the training data.
In order to capture epistemic uncertainty we use a Bayesian neural network with a prior distribution put over its weights (e.g. a Gaussian prior) $\mathbf{W} \sim \mathcal{N}(0, I)$. Now that the network's weight parameters are distributions, instead of optimizing the network weights directly we average over all possible weights (marginaliza-

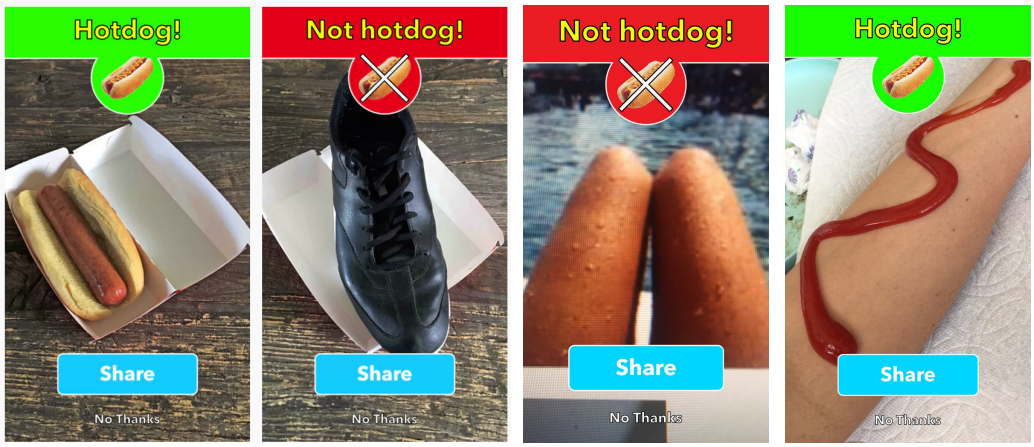


Figure 2: An interesting example for epistemic uncertainty is an app called *Not Hotdog*. It is simply supposed to tell if an image contains a hotdog or not. The model itself performs decently, however when presented with objects being covered by ketchup it gets fooled quickly. This is most likely due to the fact that it was never trained on "not-hotdog" images. A Bayesian deep learning model would have predicted a high epistemic uncertainty for the leg with ketchup.

tion).
The output of the BNN is defined as $\mathbf{f}^{\mathbf{W}}(\mathbf{x})$, the model likelihood as $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x}))$. Bayesian inference, a method of statistical interference, is used to compute the new posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$ based on more evidence (data points), which captures plausible model parameters given a dataset $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}, \mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_N\}$. However the marginal probability $p(\mathbf{Y}|\mathbf{X})$, required to calculate the posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y}) = p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})/p(\mathbf{Y}|\mathbf{X})$, cannot be evaluated analytically.
The solution given in the paper is fitting the posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$ with a simple distribution $q^*_\theta(\mathbf{W})$ (parameterized by θ). Therefore it is no longer necessary to average over all weights in the BNN, but instead perform an optimization task where we seek to optimize the parameters of this simple distribution.
In practice, often dropout variational interference is performed to approximate interference in complex models. The model is trained with dropout before every layer and at the time of testing also utilizing dropout to sample from the approximate posterior. This dropout can be interpreted as a variational Bayesian approximation. Formally, this is equivalent to finding a simple distribution $q^*_\theta(\mathbf{W})$ with approximate variational interference, which minimizes the Kullback-Leiber divergence to the true posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$.

The minimization objective is given by ([1]):

$$\mathcal{L}(\theta, p) = -\frac{1}{N}\sum_{i=1}^{N} \log p(\mathbf{y}_i|\mathbf{f}^{\widehat{\mathbf{W}}_i}(\mathbf{x}_i)) + \frac{1-p}{2N}||\theta||^2 \tag{2}$$

(with N: data points; p: dropout probability; $\widehat{\mathbf{W}}_i \sim q_\theta^*(\mathbf{W})$: samples; θ: parameters of the simple distribution to be optimized)

For dropout, θ refers to the weight matrices.
Regarding the tasks of regression and classification, this loss can be simplified and the corresponding predictive variance can easily be calculated.

3.2.1 Classification Setting

For classification, the model output is squashed through a softmax function $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) = \text{Softmax}(\mathbf{f}^{\mathbf{W}}(\mathbf{x}))$. Afterwards the resulting probability vector is sampled. Therefore the likelihood in the loss function can then be approximated with Monte Carlo integration:

$$p(y = c|\mathbf{x}, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{T}\sum_{t=1}^{T} \text{Softmax}(\mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})) \tag{3}$$

(with T: number of sampled masked model weights; $\widehat{\mathbf{W}}_t \sim q_\theta^*(\mathbf{W})$, with the dropout distribution $q_\theta(\mathbf{W})$)

The uncertainty can then be obtained by calculating the entropy of the probability vector:

$$H(\mathbf{p}) = -\sum_{c=1}^{C} p_c \log p_c \tag{4}$$

3.2.2 Regression Setting

For a regression setting the likelihood is often modeled as a Gaussian with its mean as the model output and a scalar observation noise σ, which captures the noise in the output: $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) = \mathcal{N}(\mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma^2)$. The log likelihood in the loss function for a Gaussian likelihood can then be approximated as:

$$-\log p(\mathbf{y}_i|\mathbf{f}^{\widehat{\mathbf{W}}_i}(\mathbf{x}_i)) \propto \frac{1}{2\sigma^2}||\mathbf{y}_i - \mathbf{f}^{\widehat{\mathbf{W}}_i}(\mathbf{x}_i)||^2 + \frac{1}{2}\log \sigma^2 \tag{5}$$

The uncertainty in this case can be captured with the predictive variance:

$$\text{Var}(\mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})^T \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x}_t) - E(\mathbf{y})^T E(\mathbf{y}) \tag{6}$$

(with $E(\mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})$ being an approximation of the predicitve mean)

The predictive variance measures the models uncertainty about its own predicitons. In theory it will go to $\text{Var}(\mathbf{y}) \approx 0$ when all draws $\widehat{\mathbf{W}}_t$ take a constant value, this would mean there is zero parameter uncertainty.

4 Combining Aleatoric and Epistemic Uncertainty in One Model

Now the previously explained types of uncertainties will be combined into a single model. Not only is this the main objective of the paper, but also, as mentioned before, it is crucial to be able to analyze them separately. The approach of the authors makes it possible to study the effects of aleatoric uncertainty alone, epistemic uncertainty alone, or modeling both uncertainties together.
For this we again make use of a Bayesian neural network (prior distribution over the weights).
The posterior is approximated with dropout sampling and model weights are drawn from the approximate posterior $\widehat{\mathbf{W}} \sim q(\mathbf{W})$. This allows to obtain a model output, but now consisting of a predictive mean and variance:

$$[\hat{\mathbf{y}}, \hat{\sigma}^2] = \mathbf{f}^{\widehat{\mathbf{W}}}(\mathbf{x}) \tag{7}$$

$\mathbf{f}^{\widehat{\mathbf{W}}}$ is a Bayesian convolutional neural network with model weights $\widehat{\mathbf{W}}$. With its head split (input $\mathbf{x}$ is transformed in two ways), a single network can be used to obtain both $\hat{\mathbf{y}}$ as well as $\hat{\sigma}^2$.
The minimization objective is induced by fixing a Gaussian likelihood to model aleatoric uncertainty:

$$\mathcal{L}_{BNN}(\theta) = \frac{1}{D}\sum_{i} \frac{1}{2}\hat{\sigma}_i^{-2}||\mathbf{y}_i - \hat{\mathbf{y}}_i||^2 + \frac{1}{2}\log\hat{\sigma}_i^2 \tag{8}$$

D is the number of output pixels $\mathbf{y}_i$ corresponding to input image $\mathbf{x}$. D might for example be set to 1 for regression tasks, or equal to the number of pixels for dense prediction tasks, where a unary for each input pixel is predicted. $\hat{\sigma}_i^2$ is the predicted variance for pixel i by the BNN. Furthermore weight decay can be added to the loss function, which was also done during evaluation in the paper.

On a sidenote, in practice the network predicts the log variance, $s_i := \log \hat{\sigma}_i^2$. This is simply more numerically stable and avoids possible division by zero.

The loss combines the two approaches from aleatoric and epistemic uncertainty modeling. Evaluating the epistemic uncertainty over the parameters with a stochastic sample through the model and aleatoric uncertainty as the $sigma$ regularization. The second regularization term strikes the balance mentioned earlier, preventing the model from predicting infinite uncertainty for the whole dataset. It is important to realize that no lables are needed to learn uncertainty, the variance σ^2 is implicitly learned, while the distributions over the parameters are a consequence of the statistical approach (BNN).
In conclusion, the predictive uncertainty in this combined model with outputs $\hat{\mathbf{y}}_t, \hat{\sigma}_t^2 = \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})$ for randomly masked weights $\widehat{\mathbf{W}}_t \sim q(\mathbf{W})$, can be approximated using:

$$\text{Var}(\mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{y}}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{y}}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T} \hat{\sigma}_t^2 \tag{9}$$

(with $\{\hat{\mathbf{y}}_t, \hat{\sigma}_t^2\}_{t=1}^T$ a set of T samples)

5 Experiments

The authors evaluated their methods with semantic segmentation and pixel-wise depth regression tasks. They used the (popular) datasets CamVid, Make3D and NYUv2 Depth. CamVid is a road scene segmentation dataset with 600 images in day and dusk settings and 10 classes. NYUv2 is a indoor segmentation dataset consisting of 40 different classes and about 1500 images. Make3D and NYUv2 Depth on the other hand are depth regression datasets. Make3D consists of 550 images of various scenery images, NYUv2 Depth is the same dataset as for the segmentation task, so indoor images with depth labels for each pixel.

Figure 3: NYUv2 40-Class segmentation results. From left to right: input image, ground truth, segmentation, aleatoric and epistemic uncertainty.

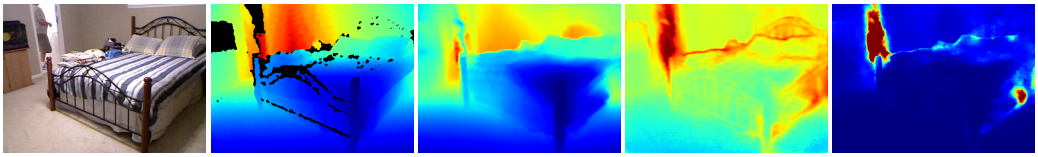


Figure 4: NYUv2 Depth regression results. From left to right: input image, ground truth, depth regression, aleatoric and epistemic uncertainty.

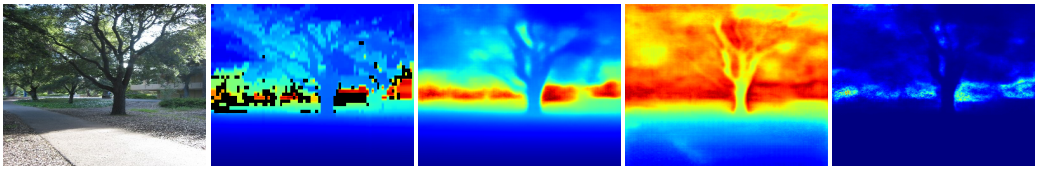


Figure 5: Make3D depth regression results. From left to right: input image, ground truth, depth prediction, aleatoric an epistemic uncertainty.

For the evaluation the authors used their own implementation of DenseNet, training with RMS-Prop, a constant learning rate of 0.001 and a weight decay of 10^{-4}. Epistemic uncertainty is modeled utilizing Monte Carlo dropout with $p = 0.2$ after each convolutional layer. Also, instead of using a Gaussian prior, a Laplacian prior used, since according to the authors L1 regularization outperforms L2 in regression tasks. These however are just minor adjustments to get improved results,

the unmodified approach should have worked in the same way.

5.1 Model Performance

For the segmentation tasks, Intersection over Union (IoU) was used as a measurement of accuracy.

CamVid	IoU
SegNet	46.4
FCN-8	57.0
DeepLab-LFOV	61.6
Bayesian SegNet	63.1
Dilation8	65.3
Dilation8 + FSO	66.1
DenseNet	66.9
This work:	
DenseNet (Our Implementation)	67.1
+ Aleatoric Uncertainty	67.4
+ Epistemic Uncertainty	67.2
+ Aleatoric & Epistemic	**67.5**

(a) CamVid dataset (road scenes).

NYUv2 40-class	IoU
SegNet	23.6
FCN-8	31.6
Bayesian SegNet	32.4
Eigen & Fergus	34.1
This work:	
DeepLabLargeFOV	36.5
+ Aleatoric Uncertainty	37.1
+ Epistemic Uncertainty	36.7
+ Aleatoric & Epistemic	**37.3**

(b) NYUv2 40-class dataset (indoor scenes).

Table 1: **Semantic segmentation performance.** Comparison to previous approaches on segmentation for the datasets.

Make3D	rel	rms	$\log_{10}$
Karsch et al.	0.355	9.20	0.127
Liu et al.	0.335	9.49	0.137
Li et al.	0.278	7.19	0.092
Laina et al.	0.176	4.46	0.072
This work:			
DenseNet Baseline	0.167	3.92	0.064
+ Aleatoric Uncertainty	**0.149**	3.93	**0.061**
+ Epistemic Uncertainty	0.162	**3.87**	0.064
+ Aleatoric & Epistemic	**0.149**	4.08	0.063

(a) Make3D depth dataset

NYU v2 Depth	rel	rms	$\log_{10}$
Karsch et al.	0.374	1.12	0.134
Liu et al.	0.335	1.06	0.127
Li et al.	0.232	0.821	0.094
Eigen et al.	0.215	0.907	-
Eigen and Fergus	0.158	0.641	-
Laina et al.	0.127	0.573	0.055
This work:			
DenseNet Baseline	0.117	0.517	0.051
+ Aleatoric Uncertainty	0.112	0.508	0.046
+ Epistemic Uncertainty	0.114	0.512	0.049
+ Aleatoric & Epistemic	**0.110**	**0.506**	**0.045**

(b) NYUv2 depth dataset

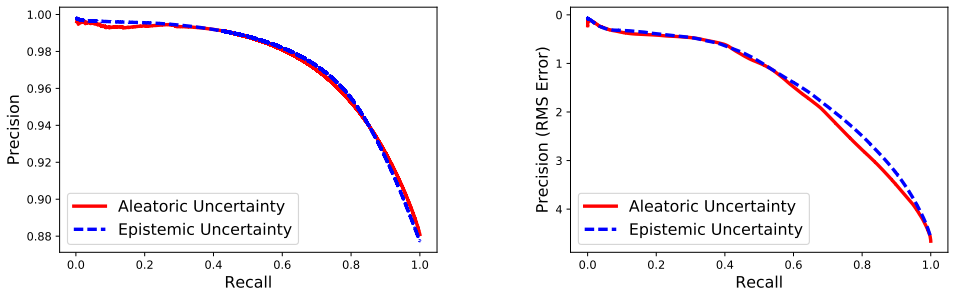
Table 2: **Monocular depth regression performance.** Comparison to previous approaches on depth regression for the datasets.

Table 1 shows, that modeling aleatoric and epistemic uncertainty improves over the baseline result. Also, aleatoric uncertainty provides a larger improvement

over the baseline than epistemic uncertainty, nevertheless modeling both at the same time increases the accuracy even further. For the NYUv2 dataset the IoU is a lot smaller, mainly because the indoor setting is challenging and the amount of classes is much greater than for CamVid. Table 2 also depicts that modeling uncertainties for regression tasks increases accuracy, however for the Make3D dataset, modeling both at the same time did not produce the best results. The authors did not comment on that, however it might be a consequence of the dataset not having labels for depths greater than 70m and in turn the learned loss attenuation might affect results more. Still, in theory, the combination of both should result in the best model.
As expected, aleatoric uncertainty is dominant for large depths, occlusion boundaries and boundaries in general, as well as reflective or badly lit surfaces. This can be seen in figures 3, 4 & 5. The qualitative results also demonstrate that epistemic uncertainty captures difficulties as a consequence of too little data. Objects which occur less frequent have higher epistemic uncertainty, for example the person in figure 4.

5.2 Further Experiments

Aside from the performance for classification and regression tasks, more interesting experiments were shown in the paper.



(a) Classification (CamVid) (b) Regression (Make3D)

Figure 6: Precision-recall plots displaying the ability of both uncertainties to account for each other in absence. Moreover they are able to capture accuracy, as with increasing uncertainty precision decreases.

One of those was removing pixels above a certain threshold for either aleatoric or epistemic uncertainty. Figure 6 displays the respective precision-recall curves for both classification and regression. The correlation between accuracy and uncertainty measurements is clear, since all of the curves are strictly decreasing.

Furthermore, the curves for aleatoric and epistemic uncertainty are similar. This can be attributed to the fact that if only one type of uncertainty is modeled, it compensates for the other type, which again underlines the importance of modeling both together. However it also shows that they are both able to capture similar uncertainty quantities in the absence of each other.

Train dataset	Test dataset	RMS	Aleatoric variance	Epistemic variance
Make3D / 4	Make3D	5.76	0.506	7.73
Make3D / 2	Make3D	4.62	0.521	4.38
Make3D	Make3D	3.87	0.485	2.78
Make3D / 4	NYUv2	-	0.388	15.0
Make3D	NYUv2	-	0.461	4.87

(a) Regression

Train dataset	Test dataset	IoU	Aleatoric entropy	Epistemic logit variance ($\times 10^{-3}$)
CamVid / 4	CamVid	57.2	0.106	1.96
CamVid / 2	CamVid	62.9	0.156	1.66
CamVid	CamVid	67.5	0.111	1.36
CamVid / 4	NYUv2	-	0.247	10.9
CamVid	NYUv2	-	0.264	11.8

(b) Classification

Table 3: Accuracy and epistemic & aleatoric uncertainties for differently sized training sets as well as distinct test sets. Epistemic and aleatoric uncertainty measures are the mean value of all pixels. It shows that aleatoric uncertainty stays roughly constant while epistemic uncertainty increases with less data and different test sets.

Also, the authors tried using smaller sized subsets of the datasets for training, as well as evaluating on different sets. Table 3 shows, that epistemic uncertainty increases as the training dataset gets smaller and that is very high if evaluated on different data. Moreover aleatoric uncertainty stays about the same for all of the tests. Once more this underlines the difference between the two types.

6 Conclusion

The paper introduced a Bayesian deep learning approach to model both aleatoric and epistemic uncertainty at the same time. It showed applications for both regression and classification tasks and at the time of publishing set new state of the art performances. It further substantiated that the two types of uncertainties model different quantities, while not being mutually exclusive. The claims that epistemic uncertainty is less important for large data settings and that aleatoric uncertainty measures uncertainty inherent in the input data were confirmed.
Modeling aleatoric uncertainty is important for big datasets, where epistemic uncertainty is less critical and the same applies to epistemic uncertainty for the vice versa scenario. From a safety-critical point of view and in reference to the assisted driving accident mentioned in the introduction, epistemic uncertainty is especially important to understand examples different from the training data, and do so in real time. Since this however requires expensive Monte Carlo sampling, which according to the authors took about 150ms on a NVIDIA Titan X GPU for a 640×480 image and is hard to parallelize, this topic of research is far from complete. Being able to properly asses uncertainty is necessary for machine learning to take over human tasks than can cause potential harm.

References

[1] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[2] Alex Kendall & Yarin Gal, University of Cambridge. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? *Machine learning*, 2017

[3] Deep Learning Is Not Good Enough, We Need Bayesian Deep Learning for Safe AI. https://alexgkendall.com/computer_vision/bayesian_deep_learning_for_safe_ai *Machine learning*, 2017

[4] Building a Bayesian deep learning classifier. https://github.com/kyle-dorman/bayesian-neural-network-blogpost *Machine learning*, 2017

[5] Bayes by Backprop from scratch (NN, classification). https://gluon.mxnet.io/chapter18_variational-methods-and-uncertainty/bayes-by-backprop.html *Machine learning*, 2017

[6] NHTSA. PE 16-007. Technical report, U.S. Department of Transportation, National Highway Traffic Safety Administration, Jan 2017. Tesla Crash Preliminary Evaluation Report.