Benchmarking Uncertainty Estimation Methods for Deep Learning With Safety-Related Metrics

Maximilian Henne, Adrian Schwaiger, Karsten Roscher, Gereon Weiss
Fraunhofer IKS
Hansastr. 32
80686 Munich, Germany
{firstname.lastname}@iks.fraunhofer.de

CEUR-WS.org/Vol-2560/paper35.pdf

Abstract

Deep neural networks generally perform very well on giving accurate predictions, but they often lack in recognizing when these predictions may be wrong. This absence of awareness regarding the reliability of given outputs is a big obstacle in deploying such models in safety-critical applications. There are certain approaches that try to address this problem by designing the models to give more reliable values for their uncertainty. However, even though the performance of these models are compared to each other in various ways, there is no thorough evaluation comparing them in a safety-critical context using metrics that are designed to describe trade-offs between performance and safe system behavior. In this paper we attempt to fill this gap by evaluating and comparing several state-of-the-art methods for estimating uncertainty for image classifcation with respect to safety-related requirements and metrics that are suitable to describe the models performance in safety-critical domains. We show the relationship of remaining error for predictions with high confidence and its impact on the performance for three common datasets. In particular, Deep Ensembles and Learned Confidence show high potential to significantly reduce the remaining error with only moderate performance penalties.

Introduction

The success of *Deep Neural Networks (DNN)* for many applications is based on their high capability on predicting correct results on a high number of input instances in many different fields, ranging from computer vision to natural language processing. For some tasks, like recommendation systems in retail, this strong performance is sufficient as the low occurrence of wrong results has no effect on the safety of a person. On the other hand, there are applications, for example self driving cars utilizing deep learning methods, where even a single misprediction can have life-critical consequences for humans. Obviously, it is important to increase the performance as much as possible in order to minimize the number of dangerous mistakes. Nevertheless, solely improving the models based on metrics like accuracy is not enough as it does not take the severity of different types of wrong results into account. For example, a pedestrian classified as a non-vulnerable static object with high confidence can lead to a dramatic outcome, while the same wrong prediction with low confidence can trigger a safety-fallback mechanism to avoid a collision. Unfortunately, the default way of getting confidences by just taking the softmax score and interpreting it as a type of certainty about the correctness of an output often results in overconfident estimates (Guo et al. 2017). Nonetheless, there are several different approaches to acquire a more reliable value for the confidence of a DNN than just the softmax scores, e.g. Bayesian Neural Networks (Gal and Ghahramani 2016), Deep Ensembles (Lakshminarayanan, Pritzel, and Blundell 2017), learning confidence as another output parameter (DeVries and Taylor 2018), or using a Dirichlet distribution to quantify the predictive uncertainty (Sensoy, Kaplan, and Kandemir 2018). These approaches have been compared to each other mostly in a sense of how well calibrated their predictions are. This solely measures how well the confidence estimates match the average accuracy or how well they perform on out-of-distribution examples. While these comparisons are highly valuable, they do not consider other types of errors and their severity from a safety perspective. Our main goal is to determine the suitability of the individual methods for specific safety-critical applications. For this, our main contribution in this paper is a benchmark of these methods on relevant metrics and characteristics for safety-critical applications.

AI for Autonomous Systems

A prominent use case in which the confidence of the AI is crucial are autonomous systems. An exemplary safety system for an automated driving system's perception chain (Weiss et al. 2018) is described, which shows the need for well-performing uncertainty estimations. While DNNs have proven to be capable of achieving impressive performance for perception tasks, for the usage in safety-critical systems, reliability of the perception is an inevitable precondition and due to the complexity of DNNs cannot be guaranteed inherently. To counteract this, a common practice is that a potentially unsafe AI can be embedded in a safety-critical system by encapsulating the AI in a so-called safety-envelope (or safety bag) that continuously monitors it. In case of a de-

Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

tected fault, the AI is isolated and a verified safety path is used as fallback solution. Figure 1 gives an overview of the perception pipeline for such a system.

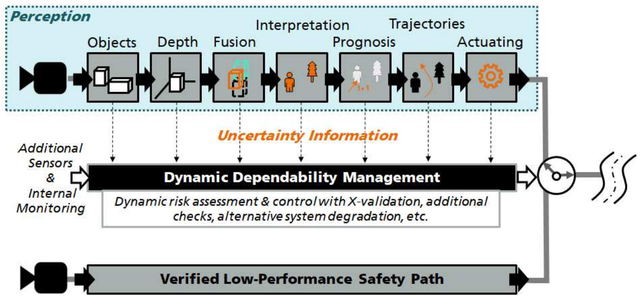

Perception
Objects
Depth
Fusion
Interpretation
Prognosis
Trajectories
Actuating
Uncertainty Information
Additional Sensors & Internal Monitoring
Dynamic Dependability Management
Dynamic risk assessment & control with X-validation, additional checks, alternative system degradation, etc.
Verified Low-Performance Safety Path

Figure 1: Concept overview for utilizing uncertainty information of modular perception stages for dynamic dependability management.

In different stages of the perception, AI is applied to solve the respective tasks like object interpretation. Additionally is depicted, how the uncertainty is used as a self-reporting and diagnosis mechanism, which is in turn used by the surrounding safety systems to take runtime decisions about which information to trust and which potential mitigation actions to take. However, a safety supervisor system like the one presented is only possible, if the self-reporting mechanisms of DNNs, especially the uncertainty quantification, are sufficiently reliable and high-performing from a safety point of view.

Related Work

Embedding AI in Safety-Critical Systems A considerable amount of work has been done to analyze and define the various aspects that help making AI safe. For example in the whitepaper "Safety First for Automated Driving" (Wood et al. 2019), to which 11 companies working on automated driving systems have contributed, a lot of potential risks and viable countermeasures for DNNs employed in autonomous systems are covered. Among them are data quality, both for gathering and labeling, self-reporting, plausibilisation, external observation of DNNs, and the surrounding systems, e.g. safety envelopes. A more specific framework for managing perceptual uncertainties, especially for autonomous driving, has been proposed by (Czarnecki and Salay 2018). The authors describe various factors that influence the uncertainty of DNNs and cover the whole process, from data acquisition to the deployment of the trained models. Lastly, practical experience reports, e.g. (Frtunikj 2019), show the encountered difficulties when developing DNN-based perception for automated driving systems and can give first hints about the viability of different safety measures.

Verification of DNNs Automated verification of DNNs and the resulting formal guarantees for them would greatly benefit the applicability of such models in safety-critical systems. However, due to their highly non-linear and non-convex nature, classic verification tools like satisfiability modulo theory or linear programming solvers are not suited for neural networks (Katz et al. 2017). That being said, there are advancements in this area, e.g. Reluplex (Katz et al. 2017) or the approach presented by (Ehlers 2017), which aims to generate linear approximations of the behavior of neural networks, in order to then solve these with conventional verification tools. These methods, however, have some limitations and are, as of now, only suited for very shallow and narrow networks. Especially in the domain of machine perception, where large architectures are dominant and the state space is enormous, a formal verification of DNNs is currently intractable.

Out-of-Distribution Detection Similar to uncertainty estimation, *out-of-distribution (OOD)* detection can be used to minimize the misclassifications made by DNNs. The goal here is to detect inputs that are not included in the data distribution the network has been trained on, as generalizability to these can not be assumed. Various approaches for this task exist, e.g. the monitoring of activation patterns at runtime (Cheng, Nührenberg, and Yasuoka 2018) or the addition of a discriminator based on adversarial autoencoders (Pidhorskyi, Almohsen, and Doretto 2018). OOD detection certainly is promising when it comes to increasing the safety of systems operating in open world domains, but on their own are not sufficient, as they don't cover the in-distribution inputs.

Explainable AI DNNs are black boxes and therefore most of the time it is not entirely clear on what features their predictions are based. To counteract this, there are efforts to make the networks explainable, which belong to the research area of *explainable AI (XAI)*. Among these approaches are visualizations of the most relevant features or input regions for a prediction, also known as saliency maps (Hong et al. 2015). Another direction of XAI is the semantic disentanglement of the learned features into high-level, human-comprehensible features which are then used to form the network's opinion (Zhang et al. 2018). Although these approaches don't directly affect the predictive quality of DNNs, they can be used to, e.g., find suitable architectures, validate the safety at design time and provide a tool to analyze errors that occur after the models have been deployed.

Predictive Uncertainty Quantification of DNNs

In the following, the approaches used for the benchmark and discussion later in the paper are introduced.

Softmax

Using the softmax output of a neural network and interpreting it as a probability distribution or uncertainty is arguably the simplest way of approaching this problem. However, it has been shown in previous research that especially for deeper neural networks, these values tend to be overconfident and poorly calibrated (Guo et al. 2017). Moreover, cross-entropy loss, which is usually used for classification tasks, can be interpreted as a maximum likelihood estimation. Therefore, it is not suited for the estimation of a predictive distribution's variance, which informs about the predictive uncertainty (Sensoy, Kaplan, and Kandemir 2018). For safety-critical domains in particular, these characteristics are very undesirable. Nevertheless, we still use the softmax as a baseline to compare the other uncertainty estimation methods with.

Monte-Carlo Dropout

Arguably the most known method for the quantification of uncertainty in DNNs is Monte-Carlo dropout (Gal and Ghahramani 2016). They argue that when dropout is applied at training and test time, it can be used to perform a variational approximation of a Bayesian neural network that has Bernoulli distributions as prior. In consequence, when randomly sampling over multiple forward passes with random dropout masks and averaging the obtained softmax distributions, the mean and variance of the predictive distribution can be approximated. As dropout is already part of, or can be integrated into, many modern DNN architectures, it is quite straightforward to use Monte-Carlo dropout for the estimation of their predictive uncertainty. It should, however, be noted that several forward passes are required to achieve a proper approximation. In embedded real-time systems this can be a hindrance.

Deep Ensembles

Deep Ensembles is another sampling-based approach for the estimation of the predictive uncertainty of DNNs (Lakshminarayanan, Pritzel, and Blundell 2017). As with other ensembling methods, multiple models having the same basic architecture are trained. Their softmax outputs are then averaged to obtain a predictive mean and variance. Although bootstrapping or bagging are often employed in other ensemble learning methods, the authors argue that DNNs generally perform better with more data and their training takes a significant amount of time, which is why they train each ensemble member in parallel and on the entire dataset. To further smooth the predictive distribution of Deep Ensembles and make them more robust to adversarial attacks, an augmentation of the training data with adversarial examples has been proposed (Lakshminarayanan, Pritzel, and Blundell 2017). Even though Deep Ensembles have no Bayesian grounding, empirically they often outperform Monte-Carlo dropout, even requiring significantly less samples, as e.g. (Beluch et al. 2018) and (Lakshminarayanan, Pritzel, and Blundell 2017) have shown. (Beluch et al. 2018) examined why Deep Ensembles generally perform better and suggest that it is mainly due to an increased model capacity, as Deep Ensembles require no dropout at inference time, and due to different weight initializations, which cause each network to converge to a different local minimum.

Learned Confidence Estimates

A further method to estimate the uncertainty of neural networks is to learn the confidence values by incentivising the neural network to produce confidence estimates which correctly reflect the ability of the model to produce correct predictions for given inputs in exchange for a reduction in loss as shown in (DeVries and Taylor 2018). They proposed a method to give neural networks the ability to ask for hints during training. This is done by adding a confidence estimation branch to any neural network in parallel with the class prediction branch where both branches receive the same input. Equation 1 formalizes the outputs of the network, whereby p is the softmax distribution and c is the confidence in the softmax output.

$$p, c = f(x, \theta) \quad p_i, c \in [0, 1] \quad \sum_{i=1}^{M} p_i = 1. \tag{1}$$

The confidence branch outputs a single scalar between 0 and 1 which is parameterized as a sigmoid. That value represents the confidence of the neural network to correctly produce the target output to a given input. A low confidence value indicates that the network can not be trusted about the given prediction. During training, the network is given hints by adjusting the softmax prediction scores for the classification, interpolating between the target and the original predictions probability distribution, where the extent of interpolation is determined by the networks confidence, as shown in equation 2.

$$p'_i = cp_i + (1 - c)y_i. \tag{2}$$

The overall loss is composed of a task- and a confidence loss. The task loss is computed by applying a standard classification loss function (e.g. negative log-likelihood) on the modified prediction outputs. The confidence loss is a binary cross-entropy loss where the target value is always 1. This confidence loss is added to the task loss with a certain weighting factor in order to prevent the network from minimizing the loss by always choosing to have 0 confidence and receiving the entire ground truth.

Both losses can then be computed following equation 3.

$$L_{task} = -\sum_{i=1}^{M} log(p'_i)y_i, \quad L_{conf} = -log(c). \tag{3}$$

Finally, both the task and the confidence loss are combined to get the total loss.

$$L_{all} = L_{task} + \lambda L_{conf}. \tag{4}$$

Now it shows, that for cases where the confidence c approaches 1, the predictive distribution will not receive any hints from the ground truth (see equation 2) and thus is equal to the standard predictive distribution. As the confidence loss will be zero, the overall loss is identical to a normal classification network with a standard task loss (see equation 3). On the other hand, if c is approaching 0, the predictive distribution will be equal to the ground truth and hence, the task loss would approach zero. Contrary, this would result in a very high confidence loss. The λ parameter is balancing how much it costs the network to ask for hints.

Evidential Deep Learning

Another sampling-free approach for uncertainty quantification is Evidential Deep Learning (Sensoy, Kaplan, and Kandemir 2018). The authors take ideas from the Dempster-Shafer Theory of Evidence and its formalization, Subjective Logic. There, in the case of classification tasks, a Dirichlet Distribution is used to quantify the belief masses and overall predictive uncertainty. The parameters of such a Dirichlet distribution can be learned by replacing the softmax activation of the output layer with a different activation function, e.g. ReLU or softplus. To fit the Dirichlet distribution to data, the authors propose three methods for the loss computation. The empirically best performing has the following form

$$L_i(\theta) = \sum_{j=1}^{K} (y_{ij} - \hat{p}_{ij})^2 + \frac{\hat{p}_{ij}(1 - \hat{p}_{ij})}{S_i + 1} \tag{5}$$

where i is the current data sample, y is the ground truth, p is the prediction, K are the individual class labels and, in consequence, the parameters of the learned Dirichlet distribution, and $S = \sum_{i=1}^{K}(e_i + 1)$, where e_i is the evidence for a class label i. This loss function generates more evidence for correct classifications, while simultaneously removing evidence that could lead to misclassifications. Overall, the authors have shown in their experiments that Evidential Deep Learning can perform on par with other uncertainty methods and can be also help with the detection of OOD samples, while requiring only a single forward pass.

Evaluation

Our benchmark for uncertainty estimation is applied to the common task of image classification, where a label from a defined set of classes is assigned to each individual input image. We compare the following methods as described in the previous section: Softmax, Monte-Carlo Dropout (MCDO), Deep Ensemble (DE), Learned Confidence (LC) and Evidential Deep Learning (EDL).

Experimental Setup

In order to compare the uncertainty estimation methods we trained a simple 6-layer CNN (SimpleCNN) and a deeper VGG16 (Simonyan and Zisserman 2014) network. In both networks, batch normalization is used after each convolutional layer and dropout is applied after each max pooling layer with a rate of 0.5. For the LC and EDL methods, the last prediction layers and the loss function had to be changed adapted as described in the previous section.

We compare the performance on three different datasets for image classification: MNIST, CIFAR-10, and German Traffic Sign Recogntion Benchmark (GTSRB) (Stallkamp et al. 2011) which are all publicly available. The task for the MNIST dataset is to classify handwritten digits into 10 classes. For CIFAR-10 images have to be classified into 10 different classes, e.g. automobile, truck or dog. The objective of the GTSRB is to classify images of german traffic signs into one of 43 classes. The first two datasets are arguably simple standard datasets to test new machine learning approaches, whereas the latter one is highly relevant for the field of autonomous driving systems. For CIFAR-10, data augmentation (rotation, flip, color, crop) was applied in order to increase performance. As optimizer, Adam was used with the learning rate set to $3 \cdot 10^{-4}$ and the momentum parameters set to the tensorflow keras default values. We stopped the training early if the validation loss did not change for a few epochs. Unfortunately, EDL did not perform reasonably well on the GTSRB dataset with both selected models. Therefore, the corresponding results are excluded in the following discussion.

	Certain	Uncertain
Correct	CC	UC
Incorrect	CI	UI

Table 1: Classification result interpretation with uncertainty

Evaluation Metrics

Without uncertainty, predicted labels are either correct, if the highest softmax output matches the respective target label, or incorrect otherwise. However, if we consider the confidence as well, each of the results can also either be certain, if the confidence is above a defined threshold, or uncertain otherwise. Table 1 summarizes the possible outcomes.

A system which relies on these estimates is expected to be in functional mode if the predictions are *certain* and in fall-back/mitigation mode if the prediction is *uncertain*. However, the most critical result regarding safety are the predictions where the model is certain about its prediction but incorrect (CI). We call the ratio of the number of certain but incorrect samples to all samples the Remaining Error Rate (RER). For minimizing the overall risk, it needs to be as low as possible. Nonetheless, if a model would always give a low confidence as output, the system would constantly remain in fall-back mode and will be unable to provide the intended functionality. Therefore, the ratio of the number of certain and correct samples to all samples - we call it the Remaining Accuracy Rate (RAR) - needs to be as high as possible to stay in performance mode for most of the time.

Results and Discussion

All results shown are based on a separate test set that was used for neither training nor hyper-parameter selection.

Model Selection All tests were performed with both models where the plain classification accuracy showed only minor differences between the two. Therefore, we stick to the SimpleCNN for most of the discussion since it has drastically fewer parameters and therefore is faster to train and apply and also is closer to the size of networks used in embedded real-time systems. Furthermore, we observed that MCDO showed almost no variance in the sampled output with VGG16 for all three datasets. This remained true even if we introduced further dropout layers into the architecture. We speculate, that given the moderate difficulty and small input image size of the three datasets, the large VGG16 model learns a lot of redundant paths that lead to the same output even if parts of the network are dropped.

In contrast to that we consistently observed, that the LC method is working considerably better with VGG16 for its confidence predictions, which is why we selected it for this particular approach. Given our observation, we assume that increased depth of the network improves the model's capability to learn better confidence values. However, further research has to be done in the future, to allow for a better comparison between the uncertainty quantification methods and their underlying architectures, especially with respect to the applicability in embedded systems.

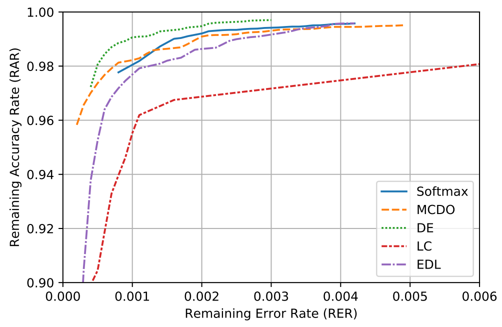
Number of Samples For the sampling based methods we investigated the influence of the number of samples on the accuracy and uncertainty estimation performance. For DE 5 samples already provided good results, with a slight improvement if 7 samples were considered. Adding more samples did not improve the results any further. MCDO on the other hand benefited much more from a higher sample count, where 10-20 samples already provided good results and further minor improvements could be observed using 100 samples.

In the following discussion we used 7 samples for DE and 100 samples for MCDO to highlight the best possible performance for each approach.

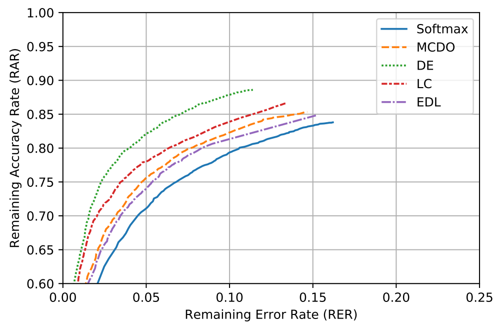
Remaining Error and Accuracy The remaining error and accuracy directly depend on the uncertainty estimation method and the threshold value that separates the certain from the uncertain outputs. However, requirements on error or accuracy rate are highly application dependent and so is the selection of a suitable threshold. Therefore, Figure 2 shows the remaining accuracy against the remaining error for each method plotted for all thresholds $t \in [0, 1)$ which are sampled with steps of 0.005 for the visualization.

On all three datasets, DE provide the best results in terms of remaining error and accuracy at the cost of keeping multiple model instances and sampling during inference time. With MNIST, depending on the selected threshold, the RER can be reduced from 0.3% to less than 0.1% with only a slight reduction in RAR. On this dataset, MCDO can reduce the RER even further at the cost of a loss of almost 4% of accuracy. Note that the results for MNIST should be taken with a grain of salt as the overall performance of every model is already very good and the differences often lie in a handful of wrong predictions which can also be attributed to the convergence to a different local minimum of each model.

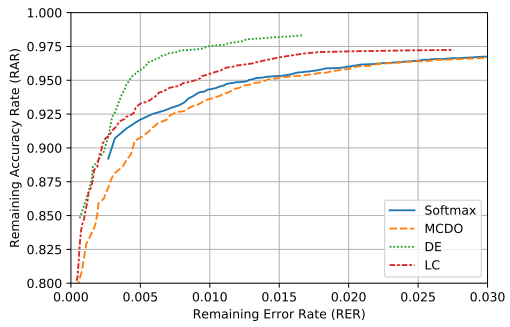
CIFAR-10 shows a much lower performance overall. Furthermore, since all curves are steeper, there is a higher penalty for a reduction in the remaining error on the remaining accuracy. However, it also covers by far the largest RER range of all three datasets. Nonetheless, the RER can be reduced from almost 12% to 2% if a reduction of the RAR to 75% percent is acceptable. On this dataset, LC achieves a strong second place and EDL performs almost as good as MCDO.



(a) MNIST



(b) CIFAR-10



(c) GTSRB

Figure 2: RAR vs. RER for each method

Finally, the performance on GTSRB shows some similarities to MNIST: the curves have a reasonable flat part on the right indicating a good trade-of between remaining error and accuracy for lower thresholds. LC, MCDO and softmax are quite close, where especially the results for softmax are surprisingly good. However, the curve for softmax ends earlier while all other methods allow to reduce the RER even further.

Confidence Calibration A further metric, which is often used to show the quality of uncertainty estimates in context to the overall performance is the network calibration. Figure 3 shows the achieved accuracy for the predictions on each confidence interval.

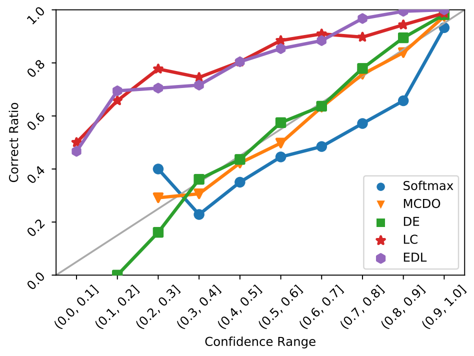


Figure 3: Network calibration, CIFAR-10

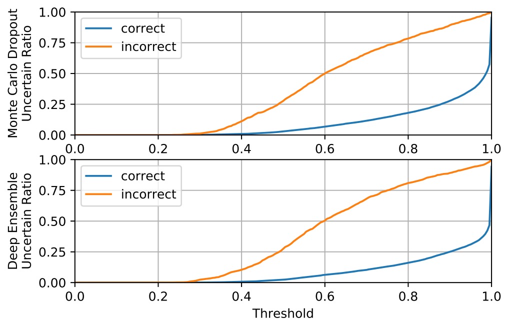
A perfect calibration would match the diagonal line from bottom left to the top right. It can be observed that Deep Ensembles give the best calibrated results as the curve is very close to the diagonal for a large part and also contains values in low confidence bins. While the softmax predictions often tend to be overconfident (confidence higher than accuracy), MCDO also produces well calibrated estimations for higher confidence scores. However, both methods do not estimate confidences of less than 0.2 at all. Contrary to that, the estimates for EDL and LC are very cautious and often result in low confidence values even though the output is correct. Nevertheless, the confidences estimated by these methods cover the entire value range which can be beneficial for safety-critical applications.

Uncertainty Estimation Quality A perfect uncertainty estimation approach would rate all incorrect predictions as uncertain and all correct predictions as certain, therefore maximizing the achievable performance while reducing the remaining error to zero.

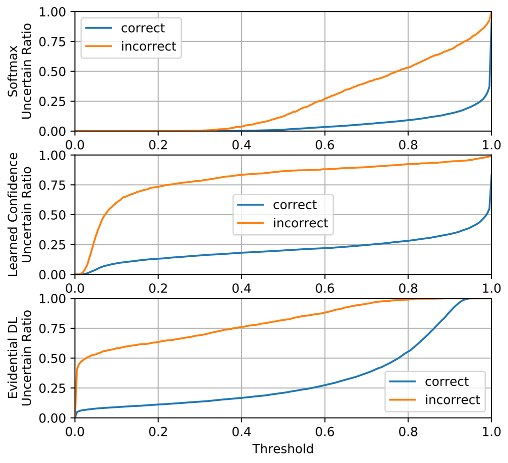
In an attempt to visualize the uncertainty estimation quality of a method, we plot the ratio of uncertain correct/incorrect to all correct/incorrect samples respectively, highlighting the ability to *shift* predictions from certain to uncertain depending on the defined threshold. Figure 4 shows the two ratios for each method.

The sampling based approaches and softmax do not estimate low confidence values below 0.3 even for incorrect classified examples. However, once they start to label predictions as uncertain, the ratio increases much faster for incorrect outputs, which is a good sign. Overall, favorable thresholds with a maximum separation can be found at the higher end of the threshold interval, e.g. around 0.8.

On the other hand, the sampling free approaches, excluding softmax, are much more conservative in their predictions giving low confidences to an overwhelming number of the incorrect predictions. Unfortunately, they also shift some of the correctly classified outputs into the uncertain



(a) Sampling based methods



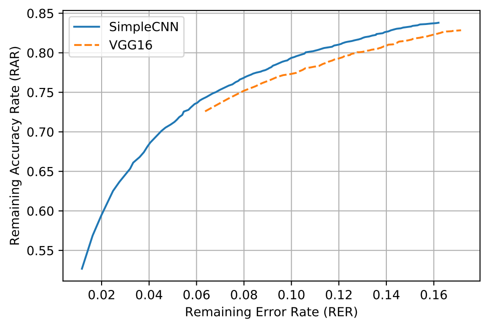
(b) Sampling free methods

Figure 4: Uncertainty ratios for different thresholds

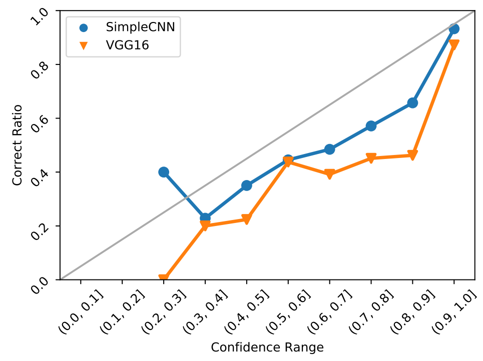
domain right from the start. However, they provide a very wide range of suitable thresholds allowing for a finer control of the trade-of between remaining performance and error. Furthermore, sampling free methods are more time efficient during inference as they only require one forward pass which makes them a strong candidate for time sensitive and safety-critical applications, for example object detection in autonomous vehicles. It is also worth mentioning that DE need much fewer forward passes while still consistently outperforming MCDO for both performance as well as safety related metrics.

Softmax Performance Even though softmax is usually outperformed by the other proposed methods, its scores were a better uncertainty estimate than we anticipated - especially on GTSRB. We attribute this to the comparatively small model size of the SimpleCNN which is in line with the observations in (Guo et al. 2017) where softmax showed to be well calibrated for small models.

Figure 5 provides further evidence for this hypothesis by comparing the results for the SimpleCNN and VGG16 both in terms of remaining accuracy and error as well as calibration. It can be observed that the softmax output from VGG16 is much less useful to reduce the RER and also significantly worse calibrated compared to the output of the smaller model.



(a) Remaining accuracy vs. error



(b) Calibration

Figure 5: Softmax uncertainty with SimpleCNN and VGG16 for CIFAR-10

Conclusion and Future Work

In this paper we described and evaluated several different methods for uncertainty estimation on metrics which were designed to give more insights on the performance with respect to safety-critical applications. We also briefly put the information gained from these uncertainty values into the context of autonomous system, where it can be used for system adaption and risk mitigation. In the results we saw that all four uncertainty estimation methods outperformed the shallow softmax predictions in almost all cases, especially for deeper networks where the softmax was highly overconfident. Furthermore, we discovered that the sampling-free approaches, especially the Learned Confidence method, are much more restrictive in their predictions giving consistently lower confidences and show a high capability to reject false predictions while still maintaining a fairly low rate of being uncertain about correct predictions. On the other hand, for lower thresholds, the sampling based methods have far lower rejection rates as they almost never give confidence values below 0.2. Nevertheless, especially Deep Ensembles show strong performance while still rejecting most of the false examples for many thresholds above that level. Overall we see strong potential by combining the two overall best performing methods which are Deep Ensembles and Learned Confidence and bringing together the good calibration and general performance of Deep Ensembles with the strong capability of rejecting false examples with a very low confidence of Learned Confidence. In the future work we will examine these promising methods and combinations of them on other tasks like out-of-distribution detection with respect to the proposed safety metrics.

Acknowledgments

This work was partially supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy through the Center for Analytics – Data – Applications (ADA-Center) within the framework of ”BAYERN DIGITAL II” and within the Intel Collaborative Research Institute Safe Automated Vehicles.

References

Beluch, W. H.; Genewein, T.; Nürnberger, A.; and Köhler, J. M. 2018. The power of ensembles for active learning in image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June.

Cheng, C.-H.; Nührenberg, G.; and Yasuoka, H. 2018. Runtime monitoring neuron activation patterns.

Czarnecki, K., and Salay, R. 2018. Towards a framework to manage perceptual uncertainty for safe automated driving. In Gallina, B.; Skavhaug, A.; Schoitsch, E.; and Bitsch, F., eds., *Computer Safety, Reliability, and Security*, 439–445. Springer International Publishing.

DeVries, T., and Taylor, G. W. 2018. Learning confidence for out-of-distribution detection in neural networks.

Ehlers, R. 2017. Formal verification of piece-wise linear feed-forward neural networks.

Frtunikj, J. 2019. Practical experience report: Engineering safe deep neural networks for automated driving systems. In Romanovsky, A.; Troubitsyna, E.; and Bitsch, F., eds., *Computer Safety, Reliability, and Security*, volume 11698 of *Lecture Notes in Computer Science*. Springer International Publishing. 235–244.

Gal, Y., and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Balcan, M. F., and Weinberger, K. Q., eds., *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, 1050–1059. PMLR. 20–22 Jun.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. ICML 2017.

Hong, S.; You, T.; Kwak, S.; and Han, B. 2015. Online tracking by learning discriminative saliency map with convolutional neural network. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, 597–606. JMLR.org.

Katz, G.; Barrett, C.; Dill, D.; Julian, K.; and Kochenderfer, M. 2017. Reluplex: An efficient smt solver for verifying deep neural networks. This is the extended version of a paper with the same title that appeared at CAV 2017.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc. 6402–6413.

Pidhorskyi, S.; Almohsen, R.; and Doretto, G. 2018. Generative probabilistic novelty detection with adversarial autoencoders. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc. 6822–6833.

Sensoy, M.; Kaplan, L.; and Kandemir, M. 2018. Evidential deep learning to quantify classification uncertainty.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition.

Stallkamp, J.; Schlipsing, M.; Salmen, J.; and Igel, C. 2011. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IEEE International Joint Conference on Neural Networks*, 1453–1460.

Weiss, G.; Schleiss, P.; Schneider, D.; and Trapp, M. 2018. Towards integrating undependable self-adaptive systems in safety-critical environments. In *13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*.

Wood, M.; Robbel, P.; Maass, M.; Tebbens, R. D.; Mejis, M.; Harb, M.; Reach, J.; Robinson, K.; Wittmann, D.; Srivastava, T.; Bouzouraa, M. E.; Liu, S.; Wang, Y.; Knobel, C.; Boymanns, D.; Lhning, M.; Dehlink, B.; Kaule, D.; Krger, R.; Frtunikj, J.; Raisch, F.; Gruber, M.; Steck, J.; Meja-Hernandez, J.; Syguda, S.; Blher, P.; Klonecki, K.; Schnarz, P.; Wiltschko, T.; Pukallus, S.; Sedlaczek, K.; Garbacik, N.; Szmera, D.; Li, D.; Timmons, A.; Bellotti, M.; O'Brien, M.; Schöllhorn, M.; Dannebaum, U.; Weast, J.; Tatourian, A.; Dornieden, B.; Schnetter, P.; Themann, P.; Weidner, T.; and Schlicht, P. 2019. Safety first for automated driving. Technical report.

Zhang, Q.; Cao, R.; Shi, F.; Wu, Y. N.; and Zhu, S.-C. 2018. Interpreting cnn knowledge via an explanatory graph.