Deep Learning Is Not Good Enough, We Need Bayesian Deep Learning for Safe AI

Bayesian Deep Learning, Computer Vision, Uncertainty

Alex Kendall
Computer Vision & Robotics Researcher

Follow

Understanding what a model does not know is a critical part of many machine learning systems. Unfortunately, today's deep learning algorithms are usually unable to understand their uncertainty. These models are often taken blindly and assumed to be accurate, which is not always the case. For example, in two recent situations this has had disastrous consequences.

1. In May 2016 we tragically experienced the first fatality from an assisted driving system. According to the manufacturer's blog, "Neither Autopilot nor the driver noticed the white side of the tractor trailer against a brightly lit sky, so the brake was not applied."

2. In July 2015, an image classification system erroneously identified two African American humans as gorillas, raising concerns of racial discrimination. See the news report here.

And I'm sure there are many more interesting cases too! If both these algorithms were able to assign a high level of uncertainty to their erroneous predictions, then each system may have been able to make better decisions and likely avoid disaster.

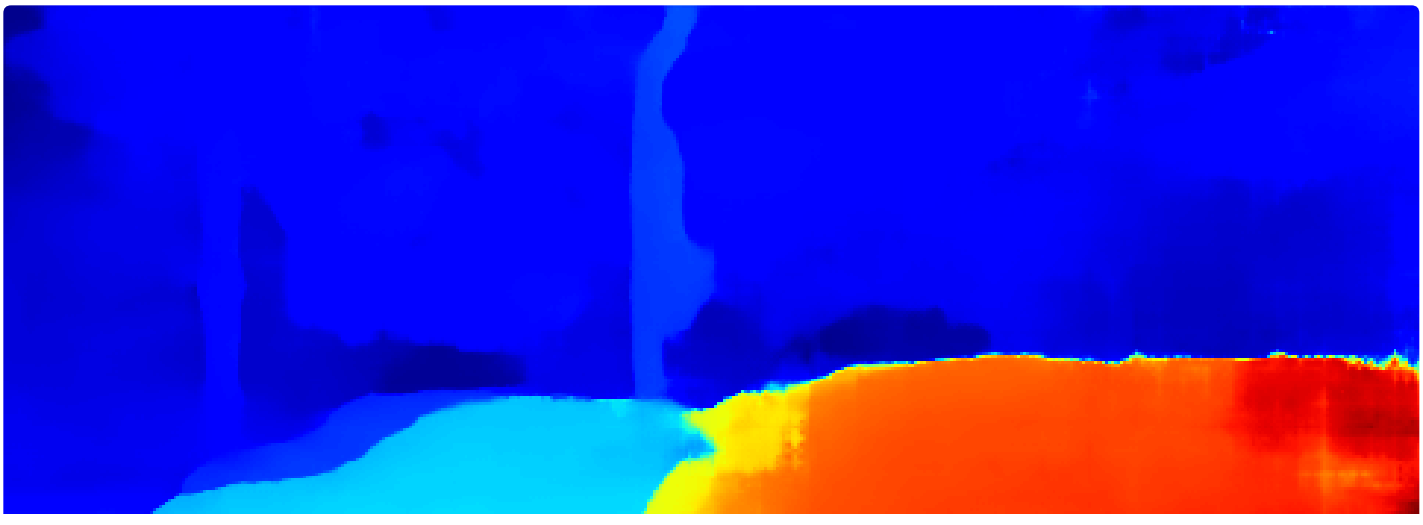
It is clear to me that understanding uncertainty is important. So why doesn't everyone do it? The main issue is that traditional machine learning approaches to understanding uncertainty, such as Gaussian processes, do not scale to high dimensional inputs like images and videos. To effectively understand this data, we need deep learning. But deep learning struggles to model uncertainty.

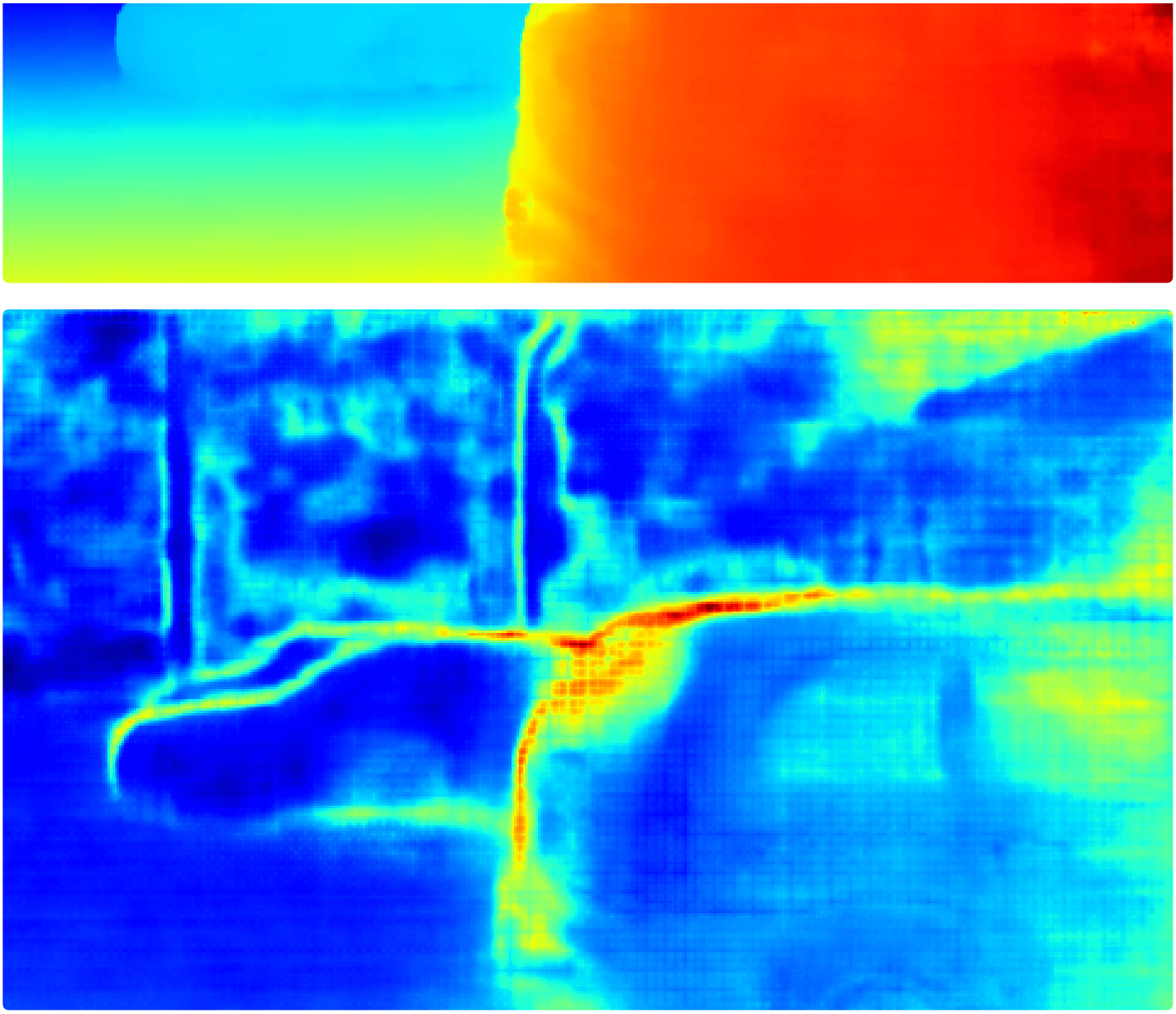
In this post I'm going to introduce a resurging field known as Bayesian deep learning (BDL), which provides a deep learning framework which can also model uncertainty. BDL can achieve state-of-the-art results, while also understanding uncertainty. I'm going to explain the different types of uncertainty

and show how to model them. Finally, I'll discuss a recent result which shows how to use uncertainty to weight losses for multi-task deep learning. The material for this blog post is mostly taken from my two recent papers:

- *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* Alex Kendall and Yarin Gal, 2017. (.pdf)
- *Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics.* Alex Kendall, Yarin Gal and Roberto Cipolla, 2017. (.pdf)

And, as always, more technical details can be found there!





An example of why it is really important to understand uncertainty for depth estimation. The first image is an example input into a Bayesian neural network which estimates depth, as shown by the second image. The third image shows the estimated uncertainty. You can see the model predicts the wrong depth on difficult surfaces, such as the red car's reflective and transparent windows. Thankfully, the Bayesian deep learning model is also aware it is wrong and exhibits increased uncertainty.

Types of uncertainty

The first question I'd like to address is what is uncertainty? There are actually different types of uncertainty and we need to understand which types are required for different applications. I'm going to discuss the two most important types – epistemic and aleatoric uncertainty.

Epistemic uncertainty

Epistemic uncertainty captures our ignorance about which model generated our collected data. This uncertainty can be explained away given enough data, and is often referred to as *model uncertainty*. Epistemic uncertainty is really important to model for:

- Safety-critical applications, because epistemic uncertainty is required to understand examples which are different from training data,
- Small datasets where the training data is sparse.

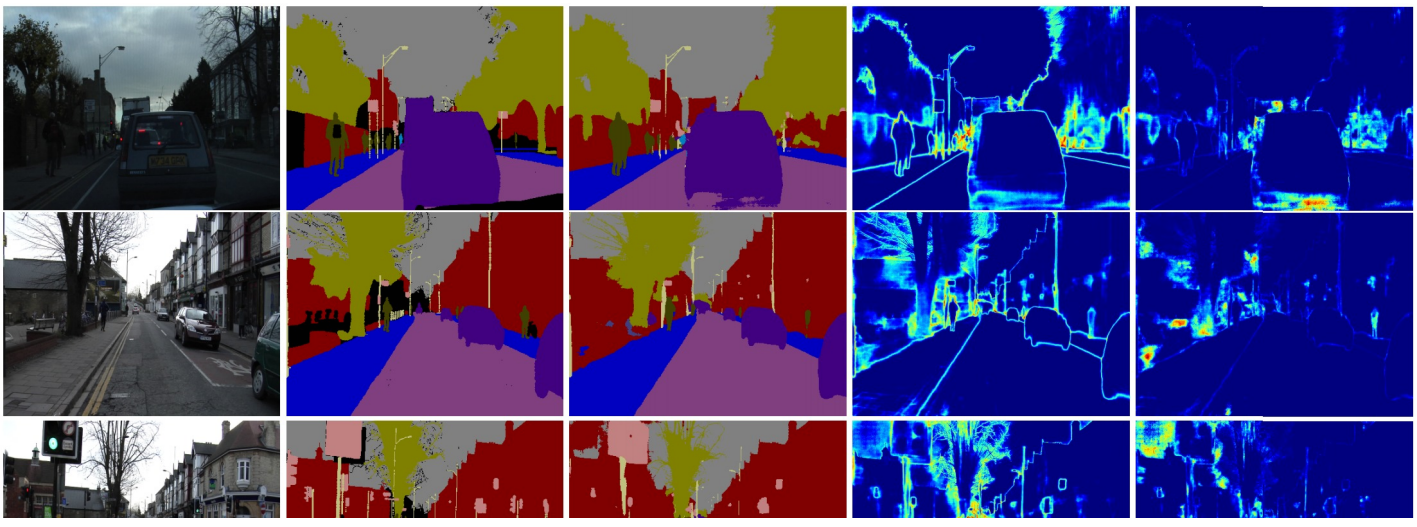
Aleatoric uncertainty

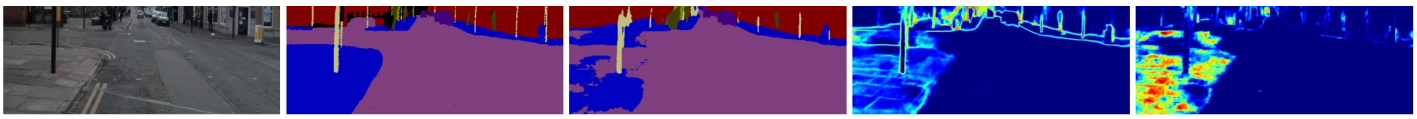
Aleatoric uncertainty captures our uncertainty with respect to information which our data cannot explain. For example, aleatoric uncertainty in images can be attributed to occlusions (because cameras can't see through objects) or lack of visual features or over-exposed regions of an image, etc. It can be explained away with the ability to observe all explanatory variables with increasing precision. Aleatoric uncertainty is very important to model for:

- Large data situations, where epistemic uncertainty is mostly explained away,
- Real-time applications, because we can form aleatoric models as a deterministic function of the input data, without expensive Monte Carlo sampling.

We can actually divide aleatoric into two further sub-categories:

- Data-dependant or Heteroscedastic uncertainty is aleatoric uncertainty which depends on the input data and is predicted as a model output.
- Task-dependant or Homoscedastic uncertainty is aleatoric uncertainty which is not dependant on the input data. It is not a model output, rather it is a quantity which stays constant for all input data and varies between different tasks. It can therefore be described as task-dependant uncertainty. Later in the post I'm going to show how this is really useful for multi-task learning.





(a) Input Image (b) Ground Truth (c) Semantic Segmentation (d) Aleatoric Uncertainty (e) Epistemic Uncertainty

Illustrating the difference between aleatoric and epistemic uncertainty for semantic segmentation. You can notice that aleatoric uncertainty captures object boundaries where labels are noisy. The bottom row shows a failure case of the segmentation model, when the model is unfamiliar with the footpath, and the corresponding increased epistemic uncertainty.

Next, I'm going to show how to form models to capture this uncertainty using Bayesian deep learning.

Bayesian deep learning

Bayesian deep learning is a field at the intersection between deep learning and Bayesian probability theory. It offers principled uncertainty estimates from deep learning architectures. These deep architectures can model complex tasks by leveraging the hierarchical representation power of deep learning, while also being able to infer complex multi-modal posterior distributions. Bayesian deep learning models typically form uncertainty estimates by either placing distributions over model weights, or by learning a direct mapping to probabilistic outputs. In this section I'm going to briefly discuss how we can model both epistemic and aleatoric uncertainty using Bayesian deep learning models.

Firstly, we can model Heteroscedastic aleatoric uncertainty just by changing our loss functions. Because this uncertainty is a function of the input data, we can learn to predict it using a deterministic mapping from inputs to model outputs. For regression tasks, we typically train with something like a Euclidean/L2 loss: $Loss = ||y - \hat{y}||_2$. To learn a Heteroscedastic uncertainty model, we simply can replace the loss function with the following:

$$Loss = \frac{||y - \hat{y}||_2}{2\sigma^2} + \frac{1}{2}\log\sigma^2$$

where the model predicts a mean $\hat{y}$ and variance σ^2. As you can see from this equation, if the model predicts something very wrong, then it will be encouraged to attenuate the residual term, by increasing uncertainty σ^2. However, the $\log\sigma^2$ prevents the uncertainty term growing infinitely large. This can be thought of as learned loss attenuation.

Homoscedastic aleatoric uncertainty can be modelled in a similar way, however the uncertainty parameter will no longer be a model output, but a free parameter we optimise.

On the other hand, epistemic uncertainty is much harder to model. This requires us to model distributions over models and their parameters which is much harder to achieve at scale. A popular technique to model this is Monte Carlo dropout sampling which places a Bernoulli distribution over the network's weights.

In practice, this means we can train a model with dropout. Then, at test time, rather than performing model averaging, we can stochastically sample from the network with different random dropout masks. The statistics of this distribution of outputs will reflect the model's epistemic uncertainty.

In the previous section, I explained the properties that define aleatoric and epistemic uncertainty. One of the exciting results in our paper was that we could show that this formulation gives results which satisfy these properties. Here's a quick summary of some results of a monocular depth regression model on two datasets:

Training Data	Testing Data	Aleatoric Variance	Epistemic Variance
Trained on dataset #1	Tested on dataset #1	0.485	2.78
Trained on 25% dataset #1	Tested on dataset #1	0.506	7.73
Trained on dataset #1	Tested on dataset #2	0.461	4.87
Trained on 25% dataset #1	Tested on dataset #2	0.388	15.0

These results show that when we train on less data, or test on data which is significantly different from the training set, then our epistemic uncertainty increases drastically. However, our aleatoric uncertainty remains relatively constant – which it should – because it is tested on the same problem with the same sensor.

Uncertainty for multi-task learning

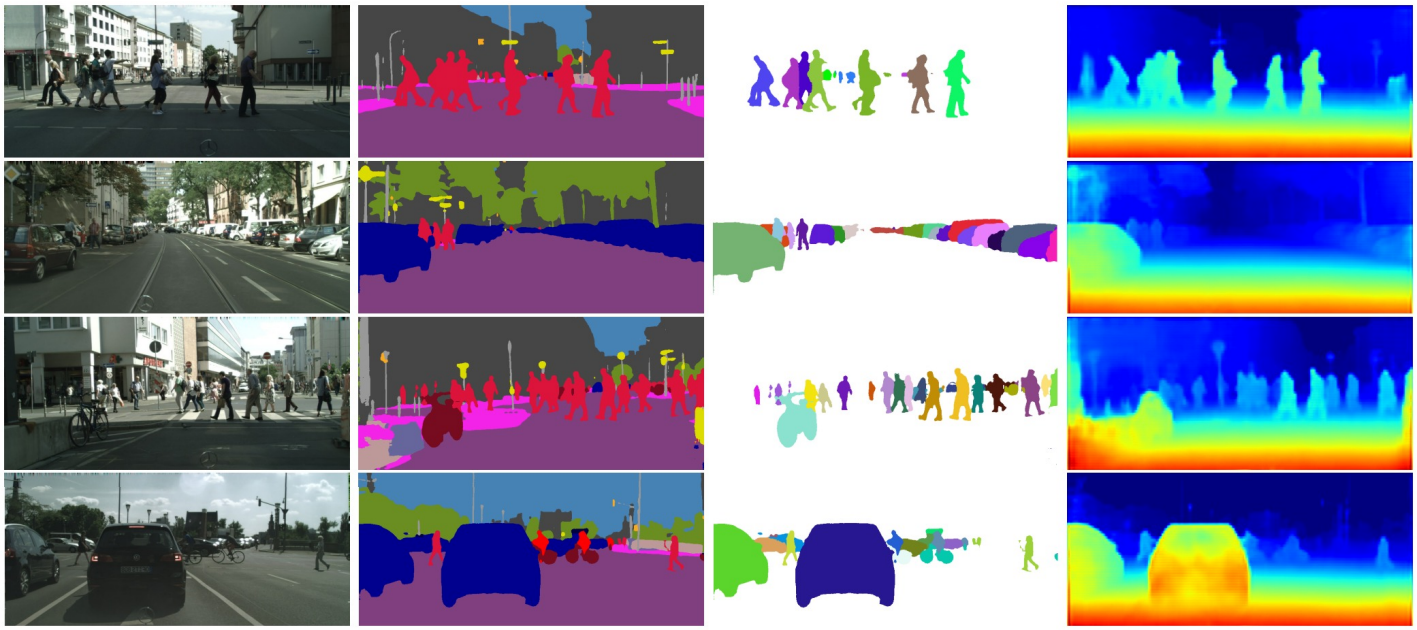
Next I'm going to discuss an interesting application of these ideas for multi-task learning.

Multi-task learning aims to improve learning efficiency and prediction accuracy by learning multiple objectives from a shared representation. It is prevalent in many areas of machine learning, from NLP to speech recognition to computer vision. Multi-task learning is of crucial importance in systems where long computation run-time is prohibitive, such as the ones used in robotics. Combining all tasks into a single model reduces computation and allows these systems to run in real-time.

Most multi-task models train on different tasks using a weighted sum of the losses. However, the performance of these models is strongly dependent on the relative weighting between each task's loss. Tuning these weights by hand is a difficult and expensive process, making multi-task learning prohibitive in practice.

In our recent paper, we propose to use homoscedastic uncertainty to weight the losses in multi-task learning models. Since homoscedastic uncertainty does not vary with input data, we can interpret it as task uncertainty. This allows us to form a principled loss to simultaneously learn various tasks.

We explore multi-task learning within the setting of visual scene understanding in computer vision. Scene understanding algorithms must understand both the geometry and semantics of the scene at the same time. This forms an interesting multi-task learning problem because scene understanding involves joint learning of various regression and classification tasks with different units and scales. Perhaps surprisingly, we show our model can learn multi-task weightings and outperform separate models trained individually on each task.



(a) Input image (b) Segmentation output (c) Instance output (d) Depth output

Multi-task learning improves the smoothness and accuracy for depth perception because it learns a representation that uses cues from other tasks, such as segmentation (and vice versa).

Some challenging research questions

Why doesn't Bayesian deep learning power all of our A.I. systems today? I think they should, but there are a few really tough research questions remaining. To conclude this blog I'm going to mention a few of them:

- Real-time epistemic uncertainty techniques are preventing epistemic uncertainty models from being deployed in real-time robotics applications. Either increasing sample efficiency, or new methods which don't rely on Monte Carlo inference would be incredibly beneficial.
- Benchmarks for Bayesian deep learning models. It is incredibly important to quantify improvement to rapidly develop models – look at what benchmarks like ImageNet have done for computer vision. We need benchmark suites to measure the calibration of uncertainty in BDL models too.

- Better inference techniques to capture multi-modal distributions. For example, see the demo Yarin set up here which shows some multi-modal data that MC dropout inference fails to model.

Tags: computer vision | deep learning | uncertainty

Categories: computer_vision

Updated: May 23, 2017

Previous | Next

LEAVE A COMMENT