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

A paper by Alex Kendall and Yarin Gal (University of Cambridge)

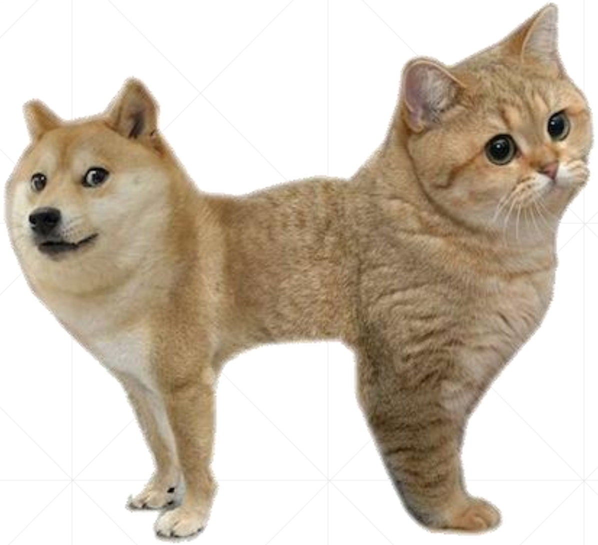
Presented by Sebastian Gruber

# List of Contents

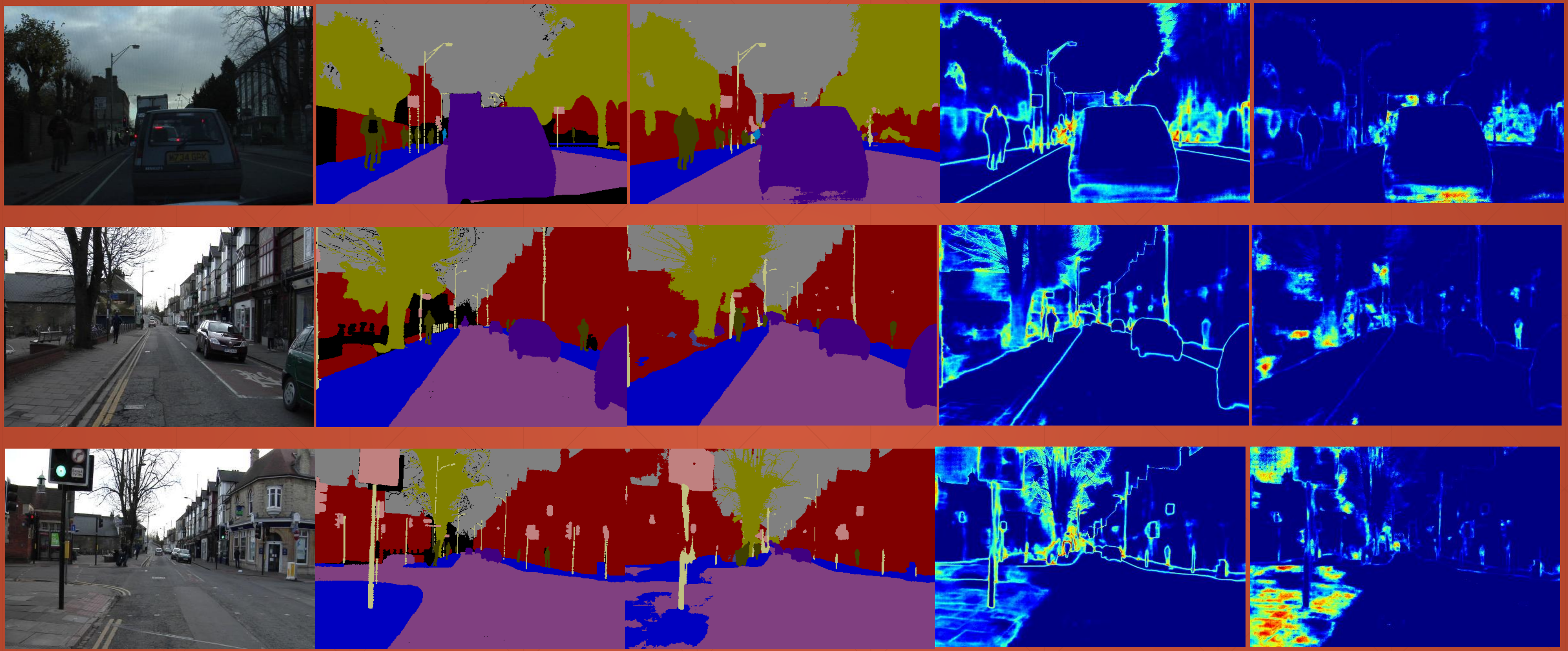
- Bayesian Neural Networks
- Epistemic uncertainy
- Aleatoric uncertainty
- Combining uncertainties into one model
- Evaluation

# Why do We Need Uncertainty?

**Uncertainty is the state of having limited knowledge where it is impossible to exactly describe the existing state, a future outcome, or more than one possible outcome.**



- Mappings  by machine learning systems are often taken blindly and assumed to be accurate, which is not always the case.
  - ➢ Understanding if your model is under-confident or falsely over-confident can help you reason about your model and your dataset
- In two recent examples this has had disastrous consequences:
  - In May 2016 there was the first fatality from an assisted driving system (Kalman filters)
  - Recently an image classification system erroneously identified two African Americans as gorillas



Input Image
Ground Truth
Segmantic Segmentation
Aleatoric Uncertainty
Epistemic Uncertainty

# Why Bayesian Deep Learning?

- "Conventional" deep learning does not allow for uncertainty representation in regression settings and classification models often give normalized score vectors, which do not necessarily capture model uncertainty.
  - For both settings uncertainty can be captured with Bayesian
- Data is limited
- We're worried about overfitting
- We have reason to believe that some facts are more likely than others, but that information is not contained in the data we model on
- We're interested in precisely knowing how likely certain facts are, as opposed to just picking the most likely fact

# Bayesian NN‘s

- Bayesian statistics is a theory in the field of statistics in which the evidence about the true state of the world is expressed in terms of degrees of belief.

- The combination of Bayesian statistics and deep learning in practice means including uncertainty in your deep learning model predictions

- Standard NN training via optimization is (from a probabilistic perspective) equivalent to maximum likelihood estimation (MLE) for the weights

- The correct (i.e., theoretically justifiable) thing to do is calculate a posterior predictive distribution

$$p(\theta \mid \mathbf{X}, \alpha) = \frac{p(\mathbf{X} \mid \theta) p(\theta \mid \alpha)}{p(\mathbf{X} \mid \alpha)} \propto p(\mathbf{X} \mid \theta) p(\theta \mid \alpha)$$

(The posterior predictive distribution is the distribution of a new data point, marginalized over the posterior)

# Types of Uncertainties

- In Bayesian modeling, there are two main types of uncertainty one can model
  - Epistemic uncertainty
    - accounts for uncertainty in the model parameters – uncertainty which captures our ignorance about which model generated our collected data
  - Aleatoric uncertainty
    - captures noise inherent in the observations
    - Homoscedastic uncertainty
      - uncertainty which stays constant for different inputs
    - Heteroscedastic uncertainty
      - depends on the inputs to the model

# Aleatoric Uncertainty

E.g.:

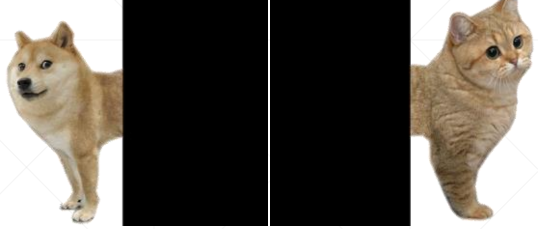
Occlusions

Lack of visual features

Under/over exposure

# An Outline

- Aleatoric and epistemic uncertainty are different and, as such, they are calculated differently.
  - Existing approaches to Bayesian deep learning capture either epistemic uncertainty alone, or aleatoric uncertainty alone
- These uncertainties are formalized as probability distributions over either the model parameters, or model outputs, respectively.
- Epistemic uncertainty is modeled by placing a prior distribution over a model’s weights, and then trying to capture how much these weights vary given some data.
- Aleatoric uncertainty on the other hand is modeled by placing a distribution over the output of the model.



# Epistemic Uncertainty in Bayesian Deep Learning

**In practice, Monte Carlo dropout sampling means including dropout in your model and running your model multiple times with dropout turned on at test time to create a distribution of outcomes. You can then calculate the predictive entropy (the average amount of information contained in the predictive distribution).**

- To capture epistemic uncertainty in a neural network (NN) we put a prior distribution over its weights, for example a Gaussian prior distribution: : $\mathbf{W} \sim N(0, I)$.
- Instead of optimizing the network weights directly we want to average over all possible weights
- Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}, \mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, Bayesian inference is used to compute the posterior over the weights $p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$.
- The model likelihood is defined by $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x}))$.
  - For regression $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) = N(\mathbf{f}^{\mathbf{W}}(\mathbf{x}), \sigma^2)$, with an observation noise scalar $\sigma$.
  - For classification squash the model output through a softmax function, and sample from the resulting probability vector: $p(\mathbf{y}|\mathbf{f}^{\mathbf{W}}(\mathbf{x})) = \mathrm{Softmax}(\mathbf{f}^{\mathbf{W}}(\mathbf{x}))$.

➢ 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

# Epistemic Uncertainty in Bayesian Deep Learning

- The posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$ is fitted with a simple distribution , parameterized by θ
- Dropout variational inference is a practical approach for approximate inference in large and complex models. The minimalization objective is given by:

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

- For Regression: $-\log p(\mathbf{y}_i|\mathbf{f}^{\widehat{\mathbf{W}}_i}(\mathbf{x}_i)) \propto \frac{1}{2\sigma^2}||\mathbf{y}_i - \mathbf{f}^{\widehat{\mathbf{W}}_i}(\mathbf{x}_i)||^2 + \frac{1}{2}\log\sigma^2$
- For Classification: Softmax over output $p(y=c|\mathbf{x},\mathbf{X},\mathbf{Y}) \approx \frac{1}{T}\sum_{t=1}^{T}$ Softmax($\mathbf{f}^{\mathbf{W}_{c_t}}(\mathbf{x})$)

# Heteroscedastic Aleatoric Uncertainty

- Aleatoric uncertainty is a function of the input data. Therefore, a deep learning model can learn to predict aleatoric uncertainty by using a modified loss function
  - ➢Teaching the model to predict aleatoric variance is an example of unsupervised learning because the model doesn't have variance labels to learn from
- In non-Bayesian neural networks, this observation noise parameter is often fixed as part of the model's weight decay, and ignored. However, when made data-dependent, it can be learned as a function of the data:

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

(with added weight decay parameterized by $\lambda$)

# Combining Aleatoric and Epistemic Uncertainty

- Predict [$\mathbf{y}$,$\sigma^2$] = $\mathbf{f}^{\mathbf{W}_c}$(x) (single network)

$$\mathcal{L}_{BNN}(\theta) = \frac{1}{D}\sum_i \frac{1}{2}\hat{\sigma}_i^{-2}||\mathbf{y}_i - \hat{\mathbf{y}}_i||^2 + \frac{1}{2}\log \hat{\sigma}_i^2 \qquad \mathrm{Var}(\mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{y}}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{y}}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t^2$$

→ In practice learn $s_i := \log \sigma_i^2$

- This loss consists of two components
  - the residual regression obtained with a stochastic sample through the model
  - an uncertainty regularization term

# Heteroscedastic Uncertainty as Learned Loss Attenuation

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

- It allows the network to adapt the residual's weighting, and even allows the network to learn to attenuate the effect from erroneous labels.
- The model is discouraged from predicting high uncertainty for all points – in effect ignoring the data – through the log term.
- The model can learn to ignore the data – but is penalized for that.
- The model is also discouraged from predicting very low uncertainty for points with high residual error, as it will exaggerate the contribution of the residual and will penalize the model.

# Evaluation / Experiments

- Performed on CamVid, Make3D, and NYUv2 Depth
  - Laplace prior instead of Gaussian (for L1)
- The Goal: Real-Time Application
  - The model based on DenseNet can process a 640x480 resolution image in 150ms on a NVIDIA Titan X GPU.
    - ➢ epistemic models require expensive Monte Carlo dropout sampling
  - Using ResNet instead of DenseNet for economical reasons

# Evaluation

| CamVid | IoU |
|---|---|
| SegNet [28] | 46.4 |
| FCN-8 [29] | 57.0 |
| DeepLab-LFOV [24] | 61.6 |
| Bayesian SegNet [22] | 63.1 |
| Dilation8 [30] | 65.3 |
| Dilation8 + FSO [31] | 66.1 |
| DenseNet [20] | 66.9 |
| *This work:* | |
| DenseNet (Our Implementation) | 67.1 |
| + Aleatoric Uncertainty | 67.4 |
| + Epistemic Uncertainty | 67.2 |
| + Aleatoric & Epistemic | **67.5** |

(a) CamVid dataset for road scene segmentation.

| NYUv2 40-class | Accuracy | IoU |
|---|---|---|
| SegNet [28] | 66.1 | 23.6 |
| FCN-8 [29] | 61.8 | 31.6 |
| Bayesian SegNet [22] | 68.0 | 32.4 |
| Eigen and Fergus [32] | 65.6 | 34.1 |
| *This work:* | | |
| DeepLabLargeFOV | 70.1 | 36.5 |
| + Aleatoric Uncertainty | 70.4 | 37.1 |
| + Epistemic Uncertainty | 70.2 | 36.7 |
| + Aleatoric & Epistemic | **70.6** | **37.3** |

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

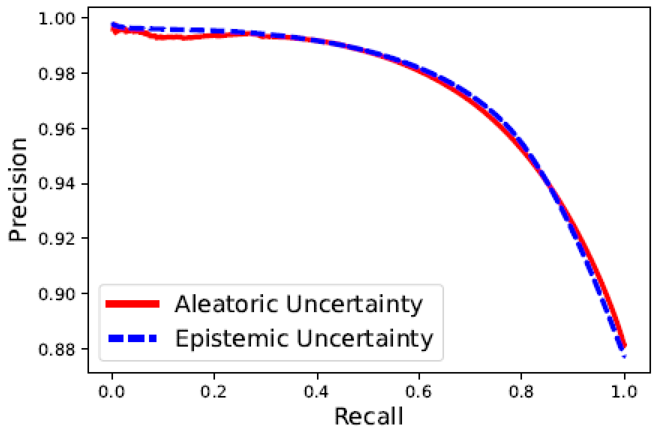
| Make3D | rel | rms | $\log_{10}$ |
|---|---|---|---|
| Karsch et al. [33] | 0.355 | 9.20 | 0.127 |
| Liu et al. [34] | 0.335 | 9.49 | 0.137 |
| Li et al. [35] | 0.278 | 7.19 | 0.092 |
| Laina et al. [26] | 0.176 | 4.46 | 0.072 |
| *This work:* | | | |
| DenseNet Baseline | 0.167 | 3.92 | 0.064 |
| + 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 [25].

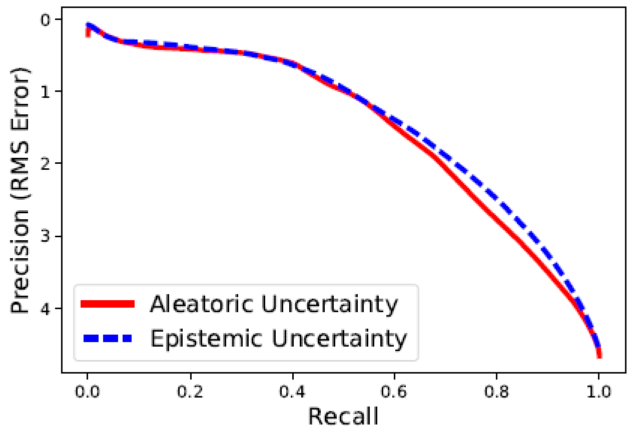
| NYU v2 Depth | rel | rms | $\log_{10}$ | $\delta_1$ | $\delta_2$ | $\delta_3$ |
|---|---|---|---|---|---|---|
| Karsch et al. [33] | 0.374 | 1.12 | 0.134 | - | - | - |
| Ladicky et al. [36] | - | - | - | 54.2% | 82.9% | 91.4% |
| Liu et al. [34] | 0.335 | 1.06 | 0.127 | - | - | - |
| Li et al. [35] | 0.232 | 0.821 | 0.094 | 62.1% | 88.6% | 96.8% |
| Eigen et al. [27] | 0.215 | 0.907 | - | 61.1% | 88.7% | 97.1% |
| Eigen and Fergus [32] | 0.158 | 0.641 | - | 76.9% | 95.0% | 98.8% |
| Laina et al. [26] | 0.127 | 0.573 | 0.055 | 81.1% | 95.3% | 98.8% |
| *This work:* | | | | | | |
| DenseNet Baseline | 0.117 | 0.517 | 0.051 | 80.2% | 95.1% | 98.8% |
| + Aleatoric Uncertainty | 0.112 | 0.508 | 0.046 | 81.6% | 95.8% | 98.8% |
| + Epistemic Uncertainty | 0.114 | 0.512 | 0.049 | 81.1% | 95.4% | 98.8% |
| + Aleatoric & Epistemic | **0.110** | **0.506** | **0.045** | **81.7%** | **95.9%** | **98.9%** |

(b) NYUv2 depth dataset [23].

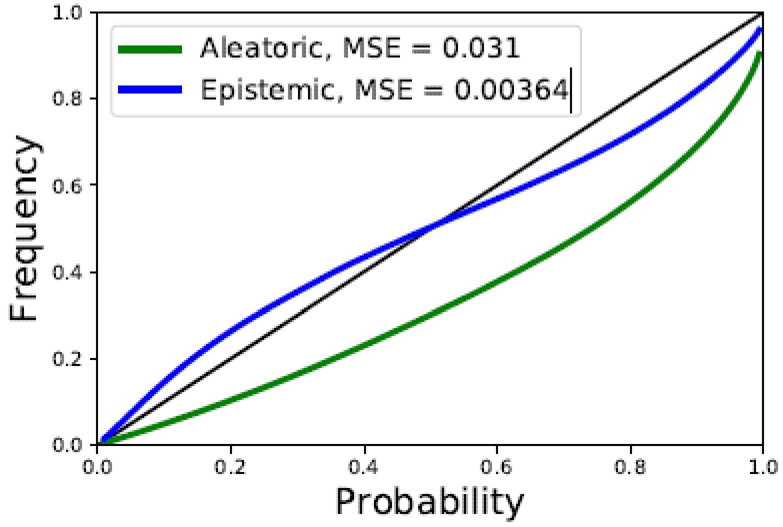
# Evaluation



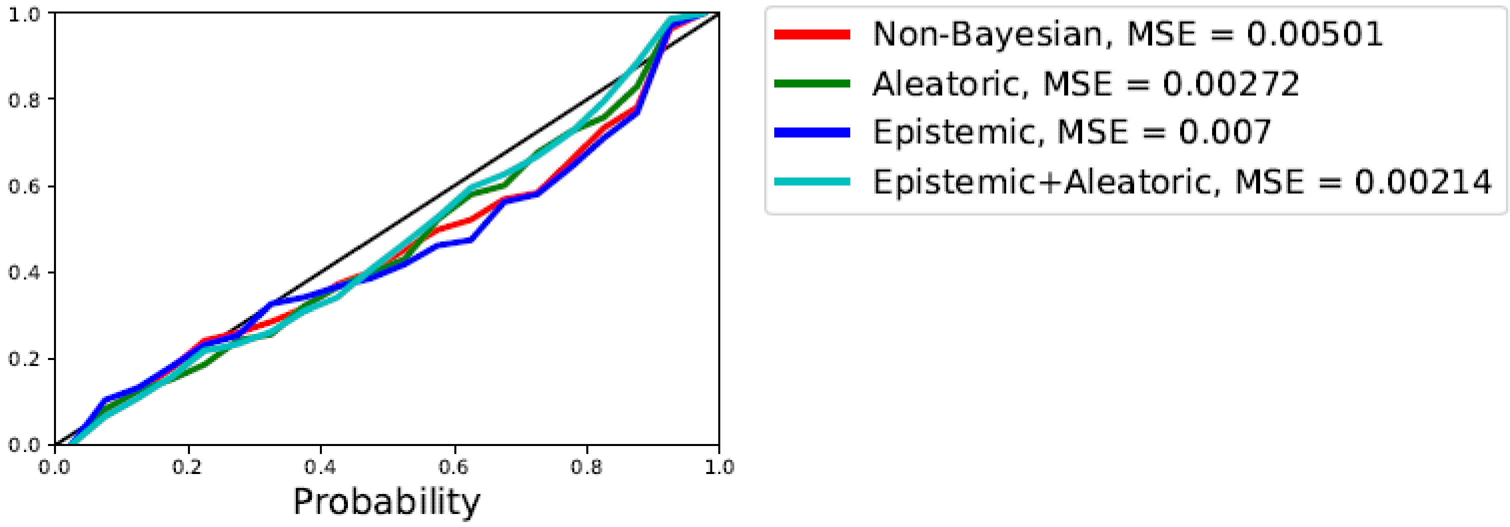

(a) Classification (CamVid)




(b) Regression (Make3D)




(a) Regression (Make3D)



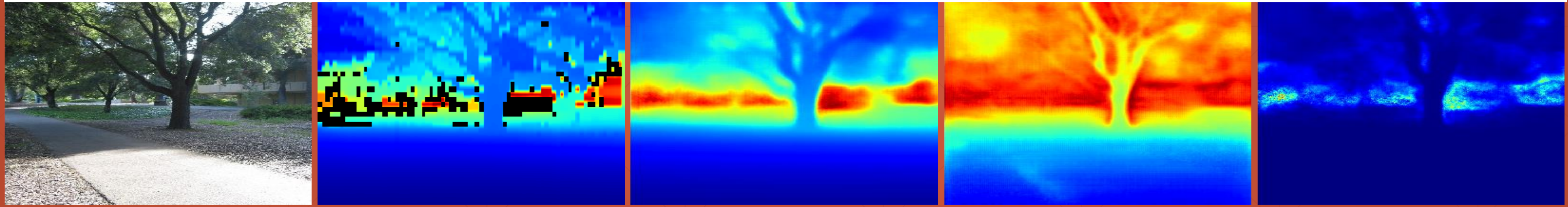

(b) Classification (CamVid)

| 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



Input Image

Ground Truth

Segmantic Segmentation/ Depth Regression

Aleatoric Uncertainty

Epistemic Uncertainty

## Takeaways

**Aleatoric Uncertainty is important for:**

- Large data situations, where epistemic uncertainty is explained away
- Real-time applications

**Epistemic uncertainty is important for:**

- Safety-critical applications
- Small datasets where the training data is sparse.

➢ However aleatoric and epistemic uncertainty models are not mutually exclusive

# Additional Sources

- https://alexgkendall.com/computer_vision/bayesian_deep_learning_for_safe_ai/
- https://github.com/kyle-dorman/bayesian-neural-network-blogpost (recommended)
- https://gluon.mxnet.io/chapter18_variational-methods-and-uncertainty/bayes-by-backprop.html
- https://en.wikipedia.org/wiki/Bayesian_inference
- https://en.wikipedia.org/wiki/Variational_Bayesian_methods
- https://de.wikipedia.org/wiki/Maximum_a_posteriori
- http://mlg.eng.cam.ac.uk/yarin/blog_3d801aa532c1ce.html

# Thank you for your attention