Uncertainty and Evaluation in Computer Vision

Neill DF Campbell

BMVA Summer School, 17 July 2024

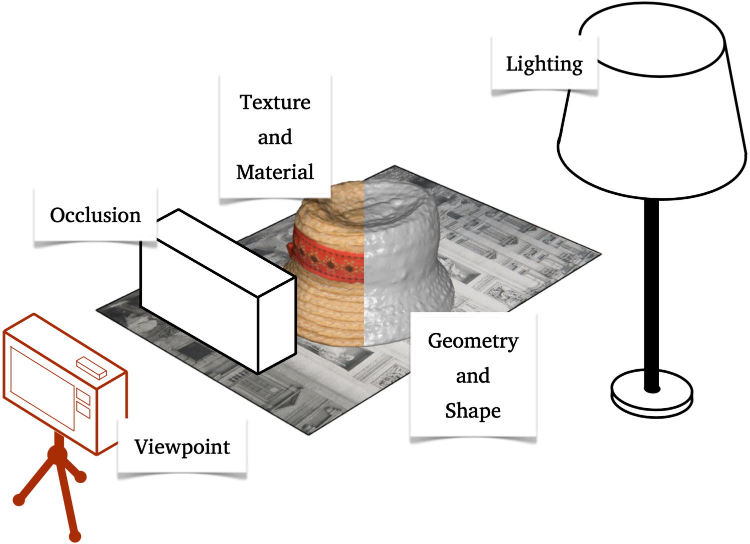
Department of Computer Science, University of Bath

(Slide input credits: Simon Prince, Mike Tipping, David MacKay)

What do you hope to take away from this session?

Why should we care about Uncertainty in Computer Vision?

Why do we need to care about Uncertainty in Computer Vision?



Motivation

Example: Uncertainty in NERF

Stochastic Neural Radiance Fields: Quantifying Uncertainty in Implicit 3D Representations

Jianxiong Shen, Adria Ruiz, Antonio Agudo, Francesc Moreno-Noguer
Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
jianxiong.shen@upc.edu

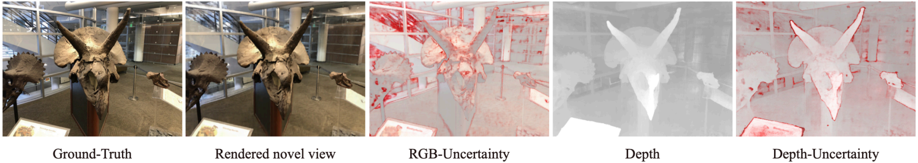
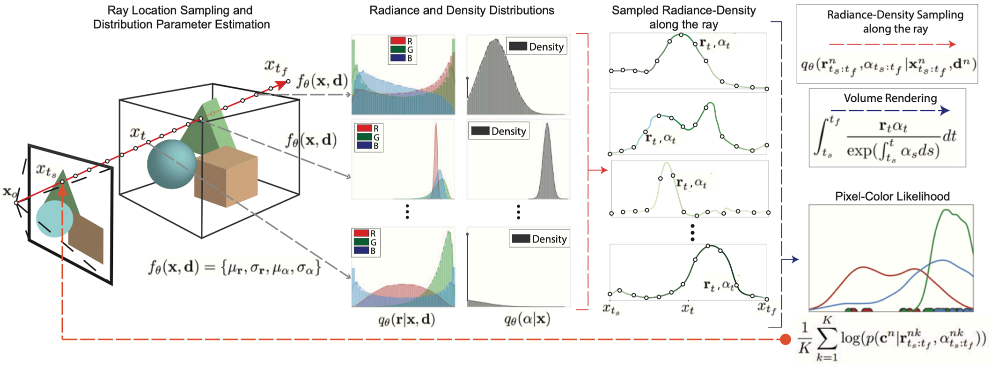
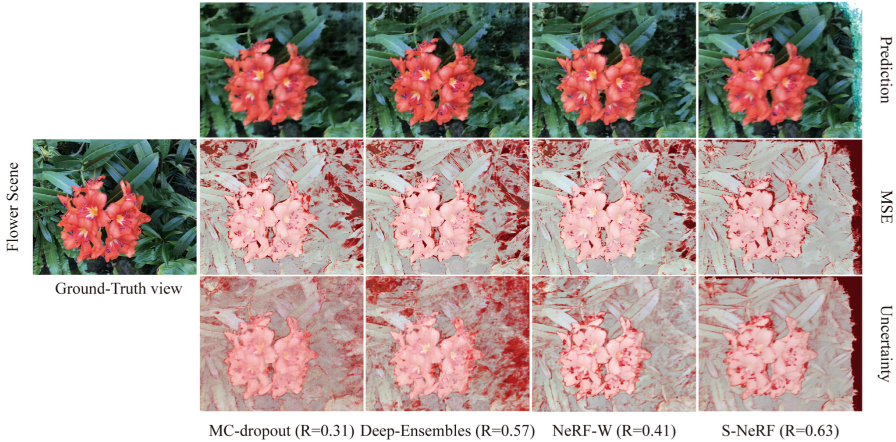


Figure 1. **Illustration of the results obtained by Stochastic Neural Radiance Fields (S-NeRF).** Our method is a probabilistic generalization of the original NeRF, which is able to not only address tasks such as novel-view generation (Rendered novel view) or depth-map estimation (Depth), but also quantify the uncertainty (red color) associated with the model outputs. This is specially important in domains such as robotics, where this information is necessary to evaluate the risk associated with decisions based on the model estimations.

Example: Uncertainty in NERF



Example: Uncertainty in NERF



On the uncertainty of self-supervised monocular depth estimation

Matteo Poggi Filippo Aleotti Fabio Tosi Stefano Mattoccia
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
{m.poggi, filippo.aleotti2, fabio.tosi5, stefano.mattoccia }@unibo.it

Abstract

Self-supervised paradigms for monocular depth estimation are very appealing since they do not require ground truth annotations at all. Despite the astonishing results yielded by such methodologies, learning to reason about the uncertainty of the estimated depth maps is of paramount importance for practical applications, yet uncharted in the literature. Purposely, we explore for the first time how to estimate the uncertainty for this task and how this affects depth accuracy, proposing a novel peculiar technique specifically designed for self-supervised approaches. On the standard KITTI dataset, we exhaustively assess the performance of each method with different self-supervised paradigms. Such evaluation highlights that our proposal i) always improves depth accuracy significantly and ii) yields state-of-the-art results concerning uncertainty estimation when training on sequences and competitive results uniquely deploying stereo pairs.

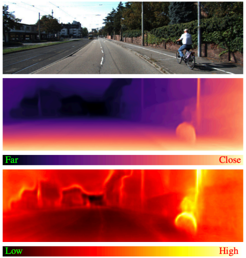
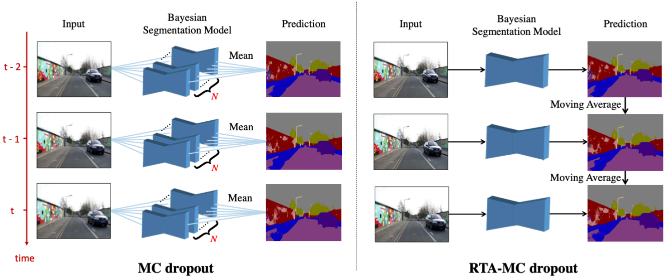


Figure 1. **How much can we trust self-supervised monocular depth estimation?** From a single input image (top) we estimate depth (middle) and uncertainty (bottom) maps. Best with colors.

Example: Uncertainty in Semantic Segmentation

Efficient Uncertainty Estimation for Semantic Segmentation in Videos

Po-Yu Huang[1], Wan-Ting Hsu[1], Chun-Yueh Chiu[1], Ting-Fan Wu[2], Min Sun[1]



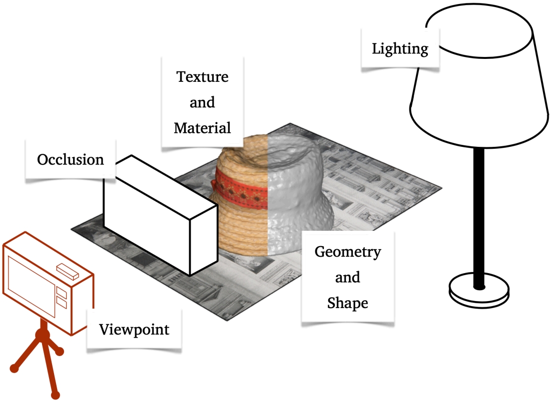
A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges

Moloud Abdar*, Farhad Pourpanah, *Member, IEEE*, Sadiq Hussain, Dana Rezazadegan, Li Liu, *Senior Member, IEEE,* Mohammad Ghavamzadeh, Paul Fieguth, *Senior Member, IEEE,* Xiaochun Cao, *Senior Member, IEEE,* Abbas Khosravi, *Senior Member, IEEE,* U Rajendra Acharya, *Senior Member, IEEE,* Vladimir Makarenkov and Saeid Nahavandi, *Fellow, IEEE*

Abstract—Uncertainty quantification (UQ) plays a pivotal role in the reduction of uncertainties during both optimization and decision making, applied to solve a variety of real-world applications in science and engineering. Bayesian approximation and ensemble learning techniques are two of the most widely-used UQ methods in the literature. In this regard, researchers have proposed different UQ methods and examined their performance in a variety of applications such as computer vision (e.g., self-driving cars and object detection), image processing (e.g., image restoration), medical image analysis (e.g., medical image classification and segmentation), natural language processing (e.g., text classification, social media texts and recidivism risk-scoring), bioinformatics, etc. This study reviews recent advances in UQ methods used in deep learning, investigates the application of these methods in reinforcement learning, and highlight the fundamental research challenges and directions associated with the UQ field.

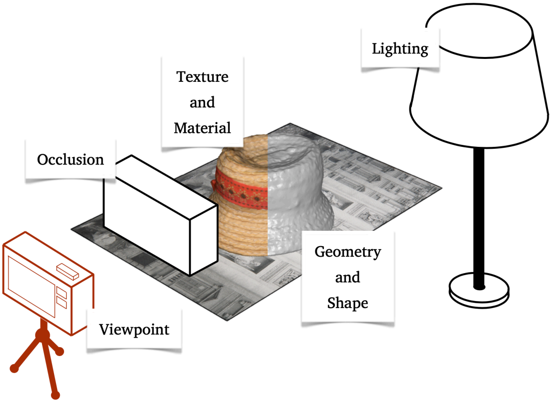
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task



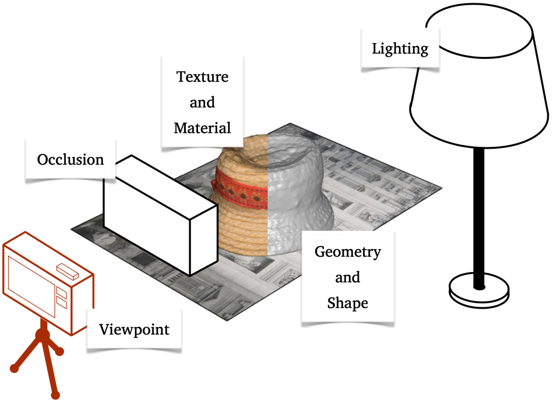
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models



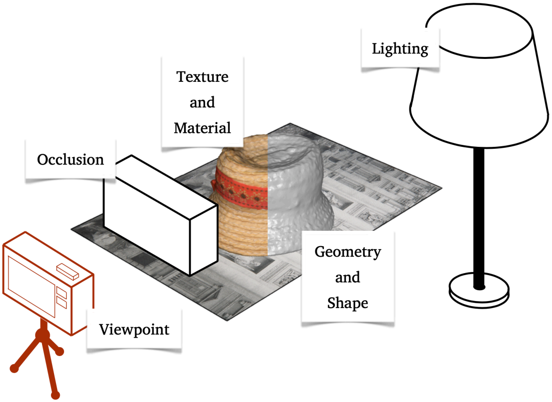
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making



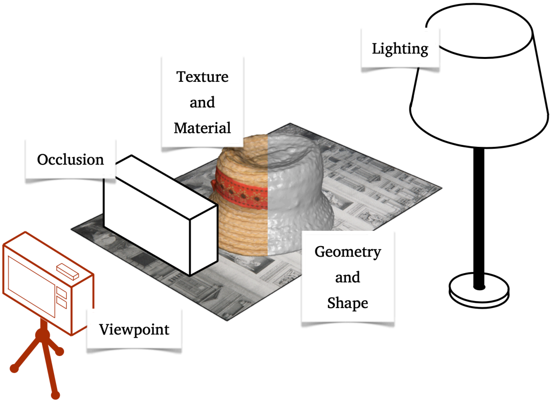
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
  - Safe



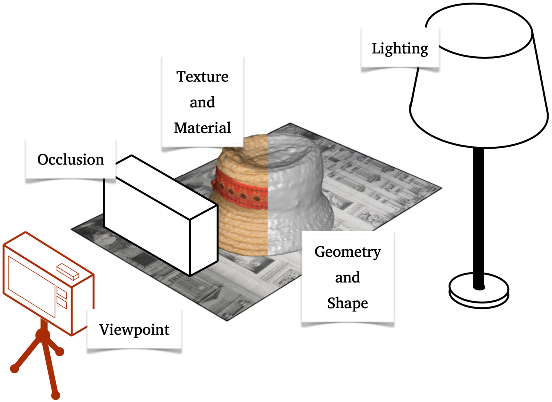
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
  - Safe
  - Robust



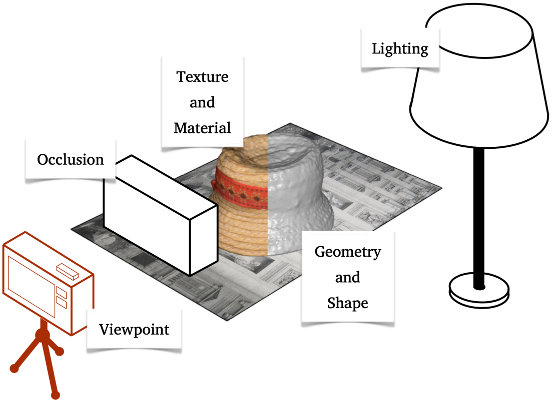
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
  - Safe
  - Robust
  - Transparent



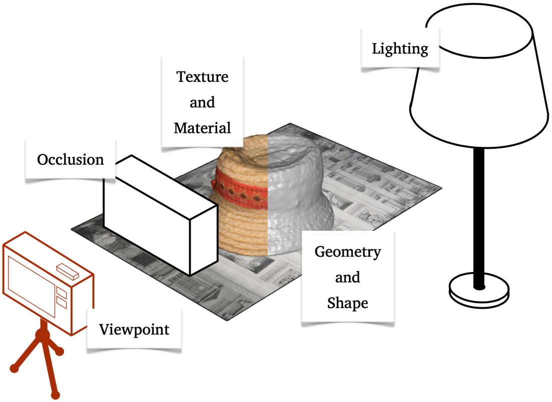
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
    - Safe
    - Robust
    - Transparent
- Improved performance



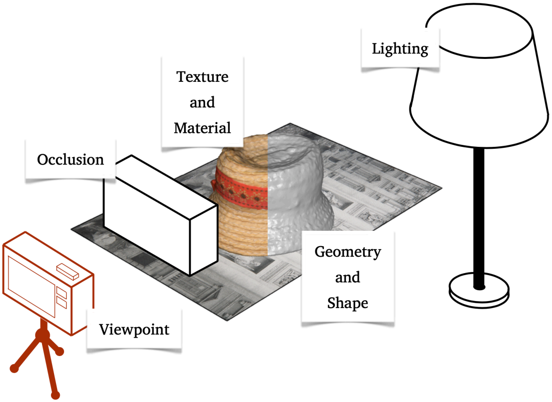
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
  - Safe
  - Robust
  - Transparent
- Improved performance
  - Data efficiency



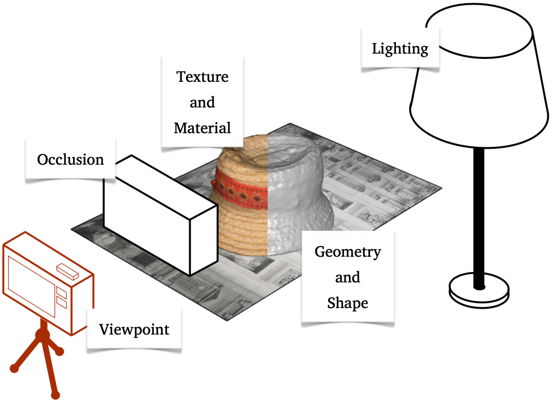
Why do we need to care about Uncertainty in Computer Vision?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
    - Safe
    - Robust
    - Transparent
- Improved performance
    - Data efficiency
    - Self-supervision

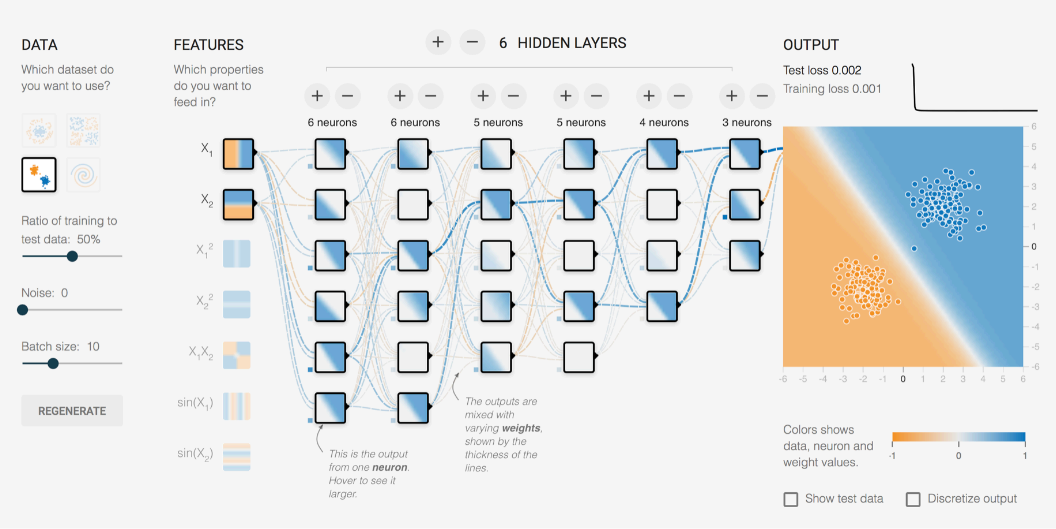


Why do we need to care about Uncertainty in Computer Vision?

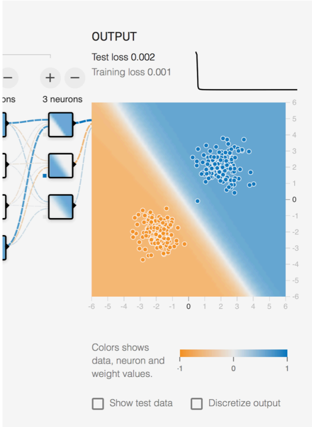
- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
    - Safe
    - Robust
    - Transparent
- Improved performance
    - Data efficiency
    - Self-supervision
- Evaluation and Model Selection!!



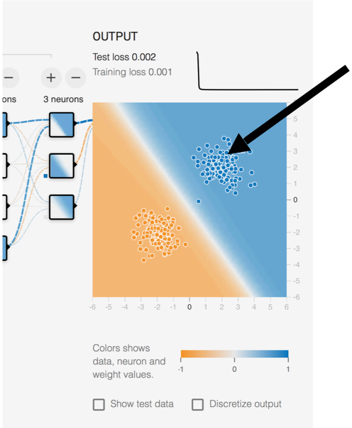
Dangers of avoiding probabilistic approaches...



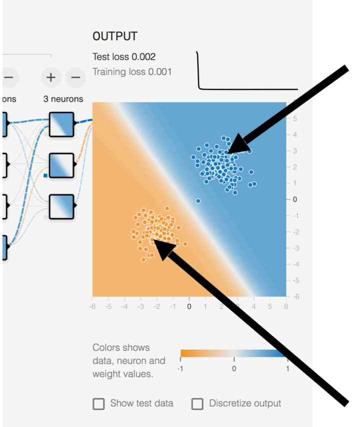
Dangers of avoiding probabilistic approaches...

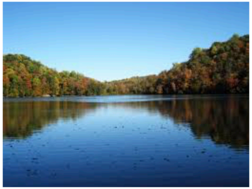


Dangers of avoiding probabilistic approaches...

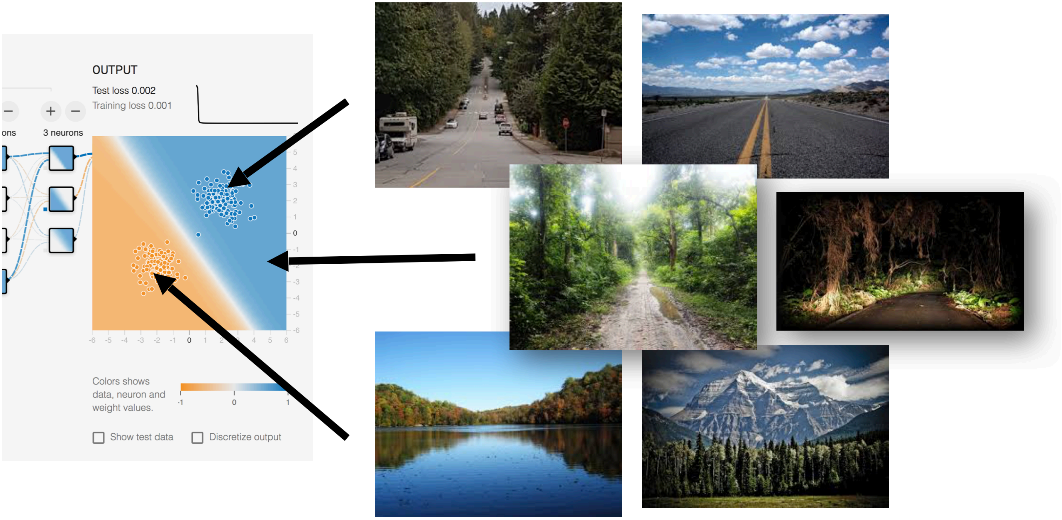


Dangers of avoiding probabilistic approaches...





Dangers of avoiding probabilistic approaches...

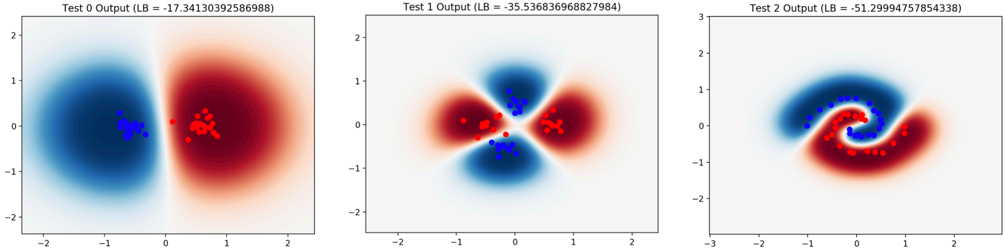


Dangers of avoiding probabilistic approaches...

What if we use a probabilistic approach?

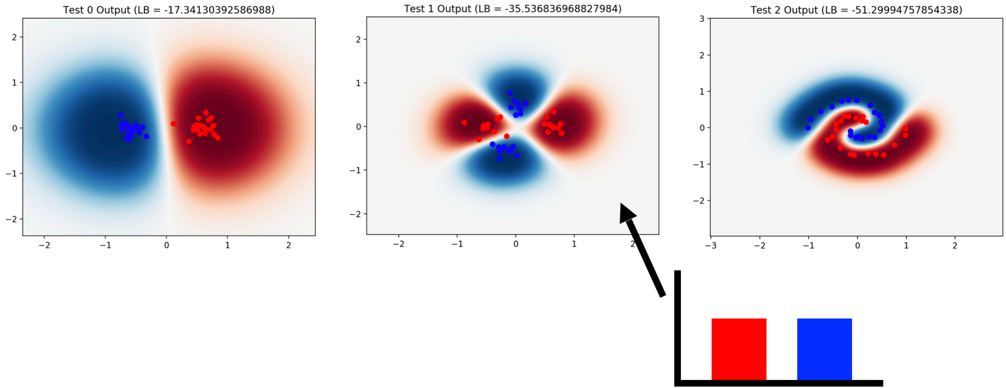
Dangers of avoiding probabilistic approaches...

What if we use a probabilistic approach?



Dangers of avoiding probabilistic approaches...

What if we use a probabilistic approach?



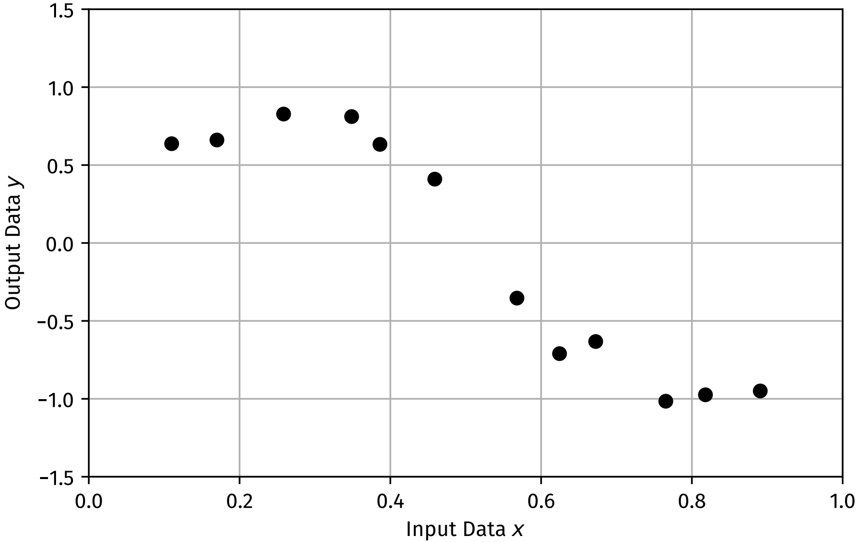
We need to consider the properties of our Computer Vision and Machine Learning models/approaches..

No Free Lunch

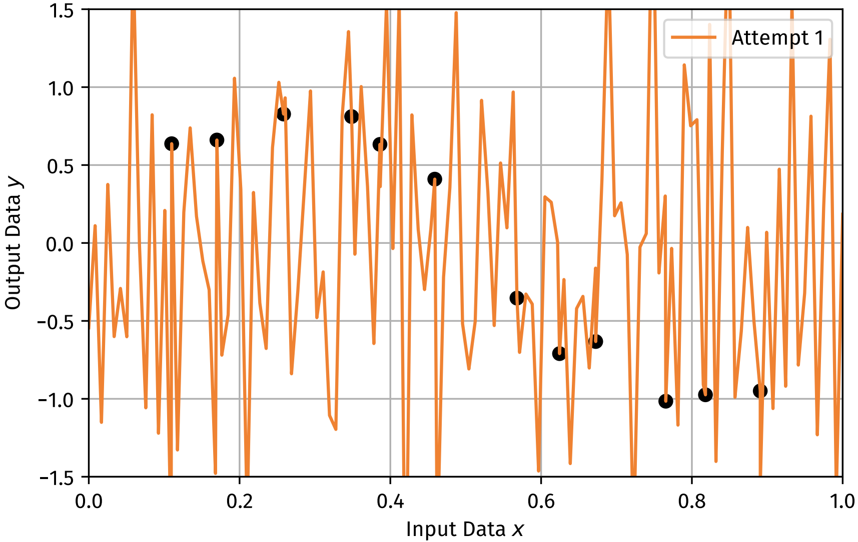
Overview...

What happens between the dots?

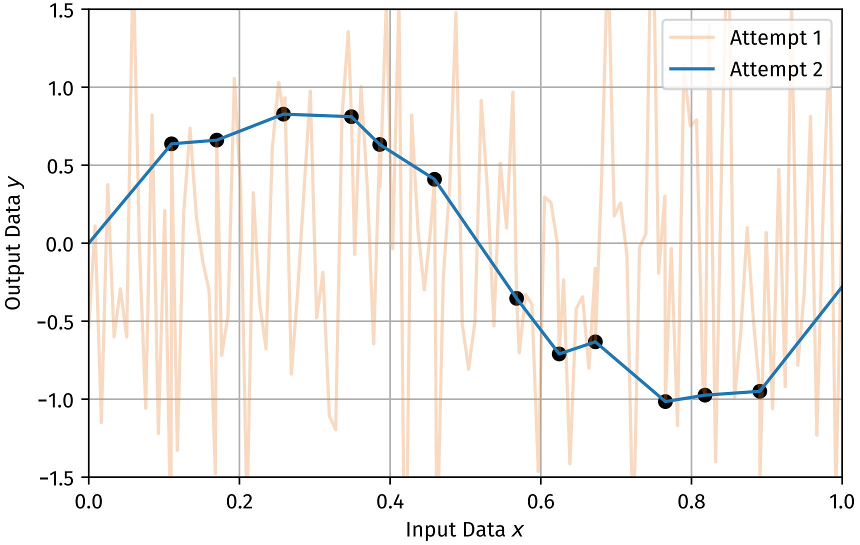
What happens between the dots?



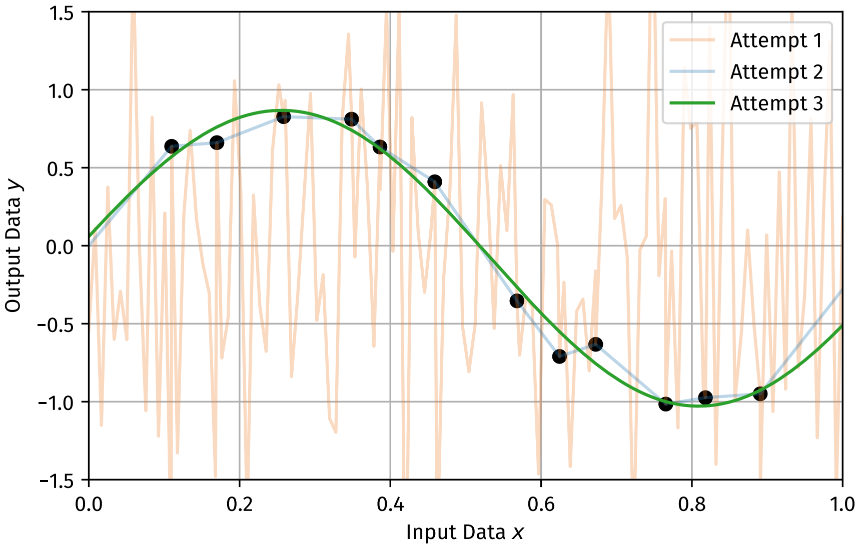
What happens between the dots?



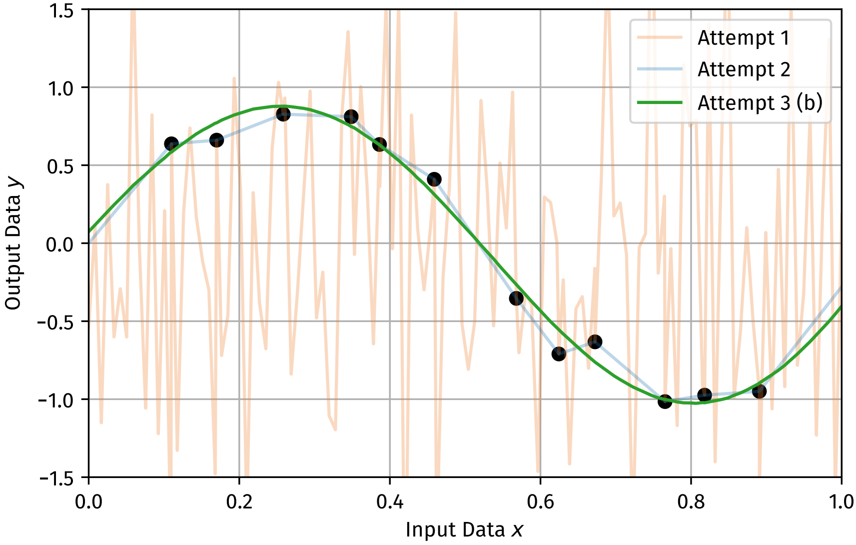
What happens between the dots?



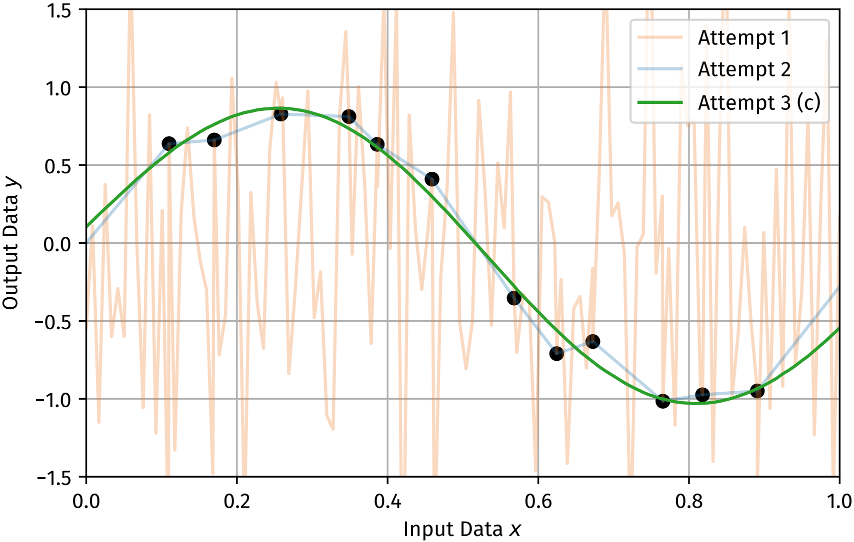
What happens between the dots?



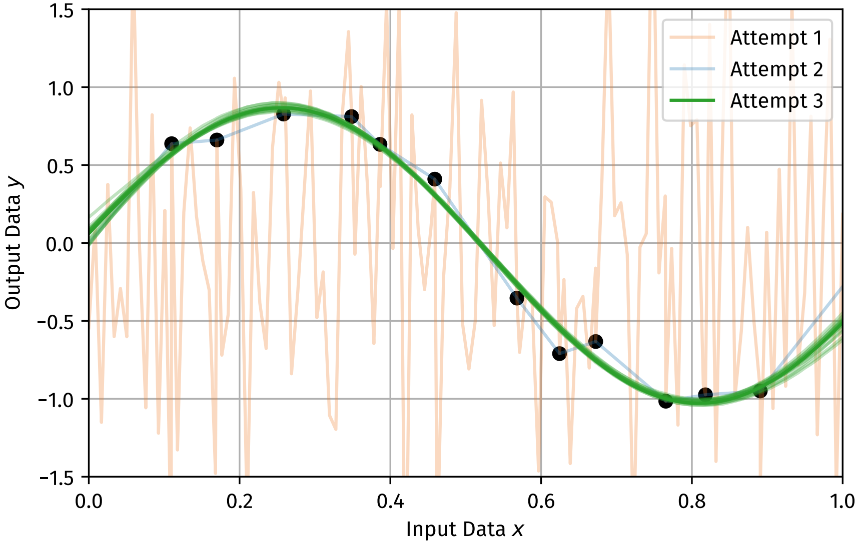
What happens between the dots?



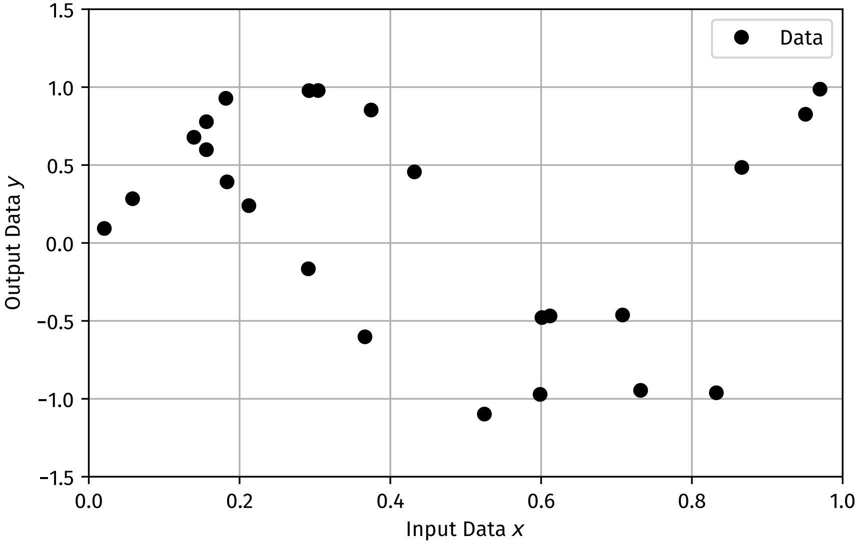
What happens between the dots?



What happens between the dots?

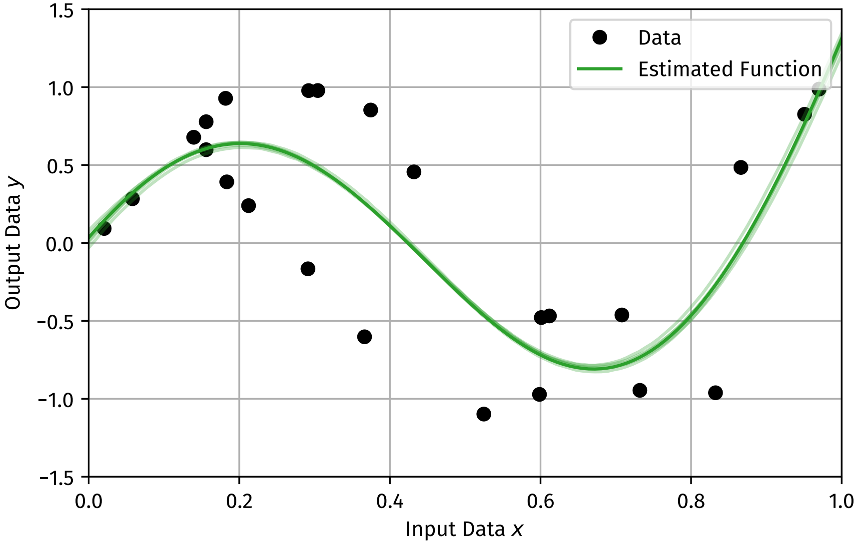


Ambiguity..



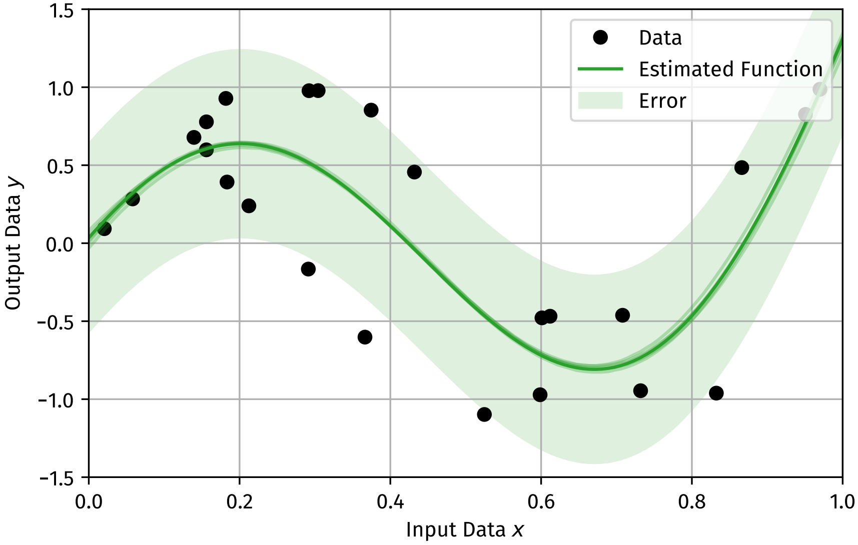
Data
Output Data y
Input Data x
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0

Ambiguity..



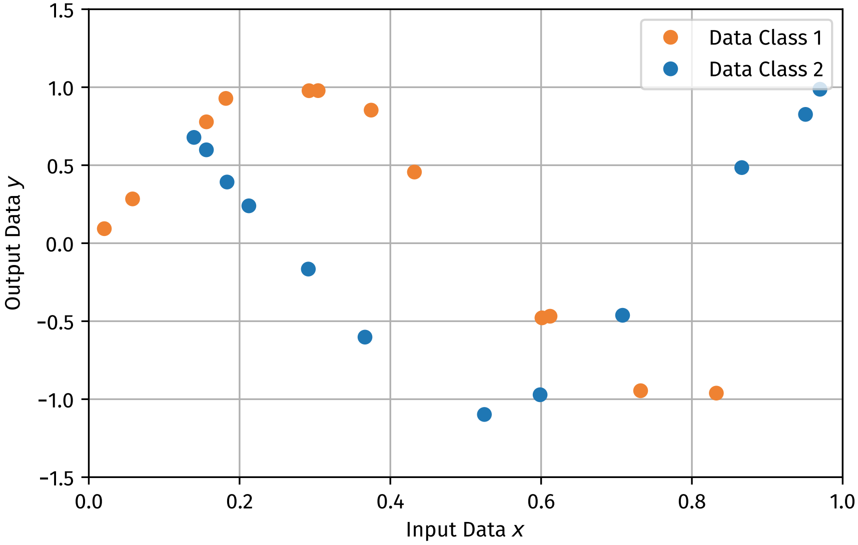
Data
Estimated Function
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Output Data y
0.0
0.2
0.4
0.6
0.8
1.0
Input Data x

Ambiguity..

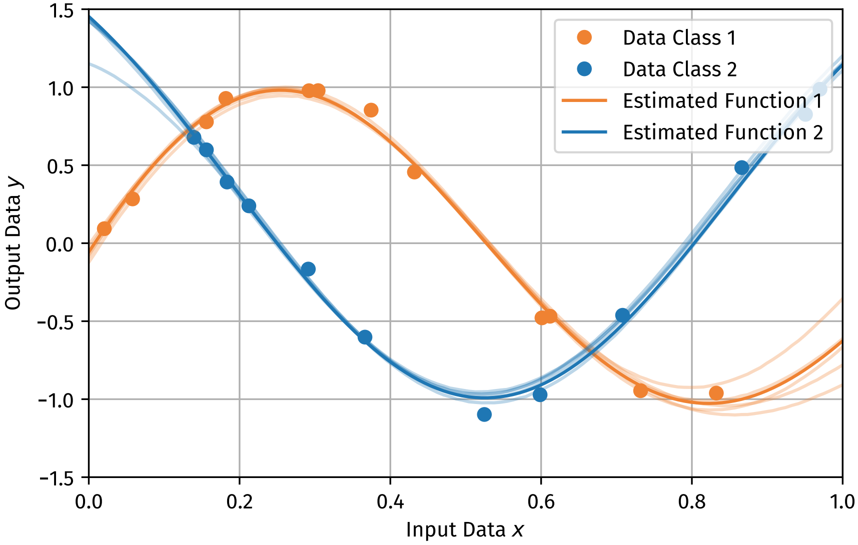


Data
Estimated Function
Error
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
Output Data y
0.0
0.2
0.4
0.6
0.8
1.0
Input Data x

Ambiguity..

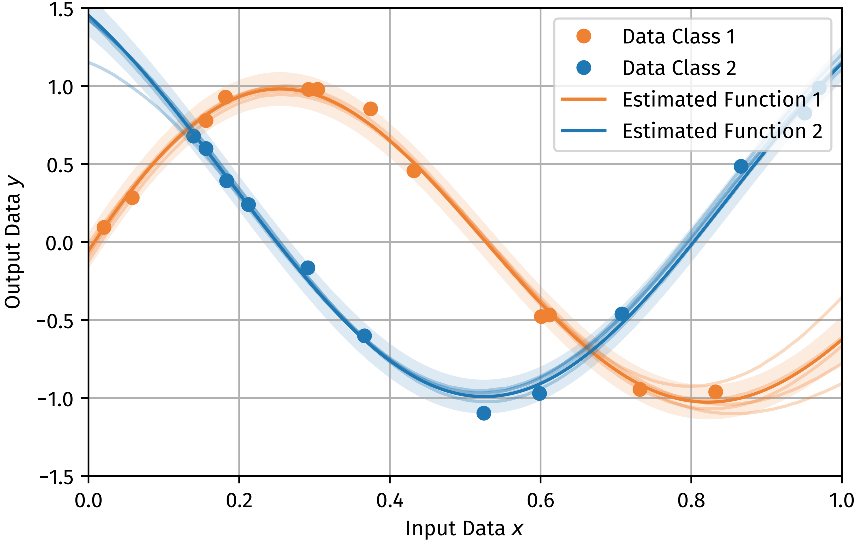

Data Class 1
Data Class 2
Input Data x
Output Data y
0.0
0.2
0.4
0.6
0.8
1.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5

Ambiguity..



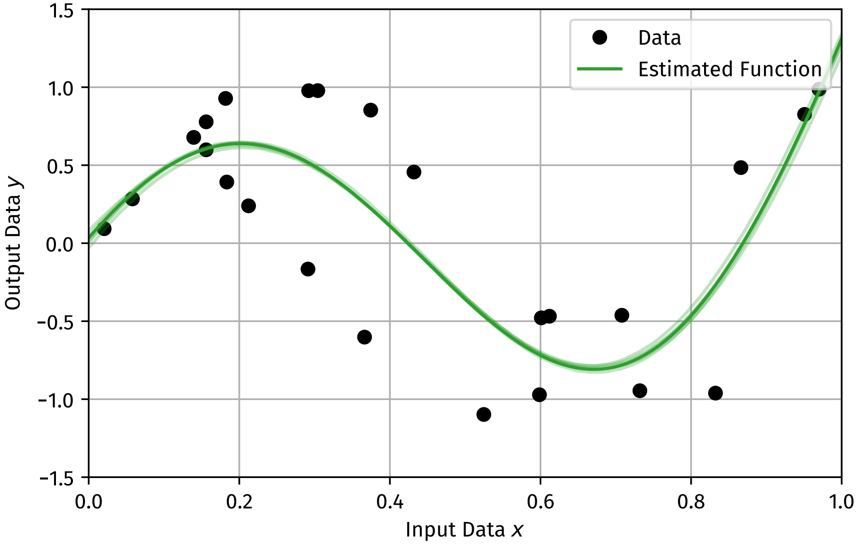
Data Class 1
Data Class 2
Estimated Function 1
Estimated Function 2
Output Data y
Input Data x
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0

Ambiguity..

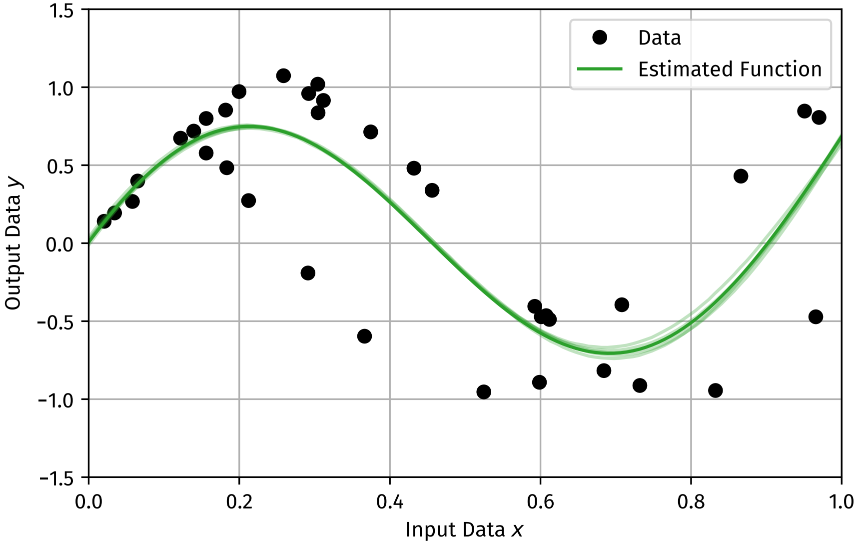

Data Class 1
Data Class 2
Estimated Function 1
Estimated Function 2
Output Data y
Input Data x
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0

Average vs Worst Case..

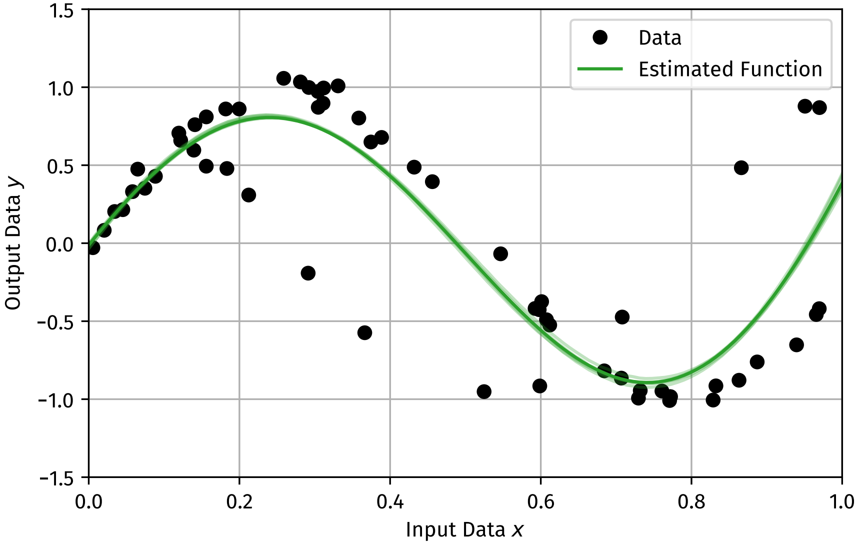
Average vs Worst Case..



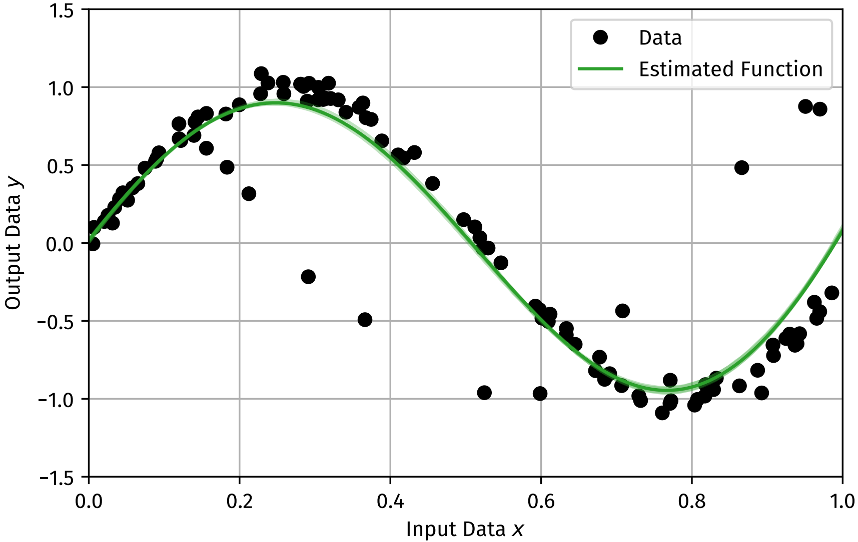
Average vs Worst Case..



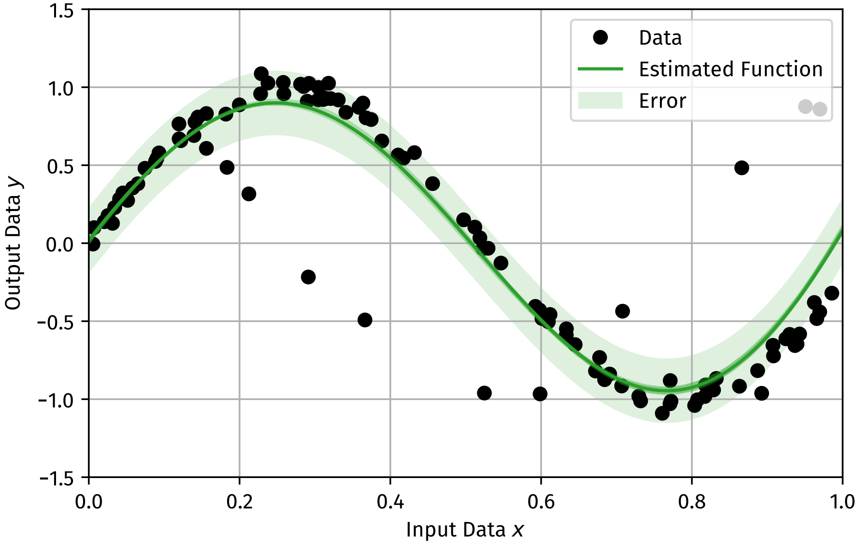
Average vs Worst Case..



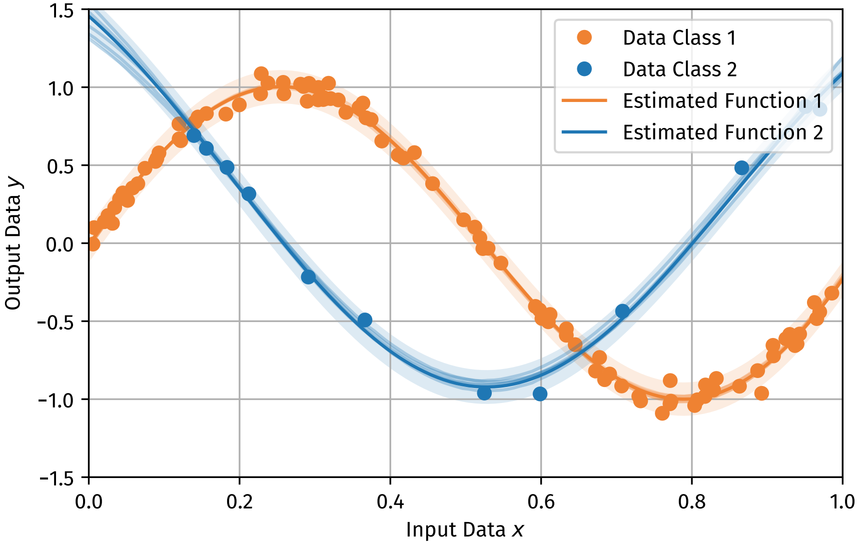
Average vs Worst Case..



Average vs Worst Case: Failure to model..



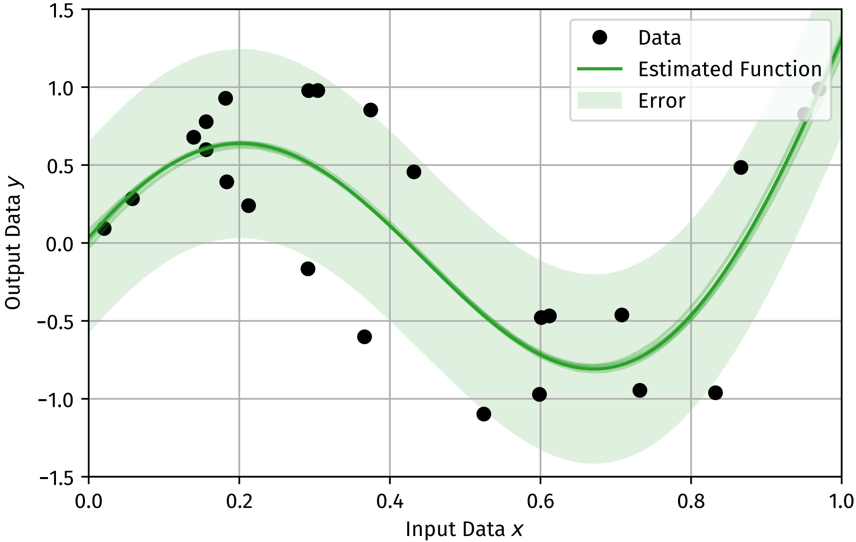
Average vs Worst Case: Explicitly accounting for imbalance..



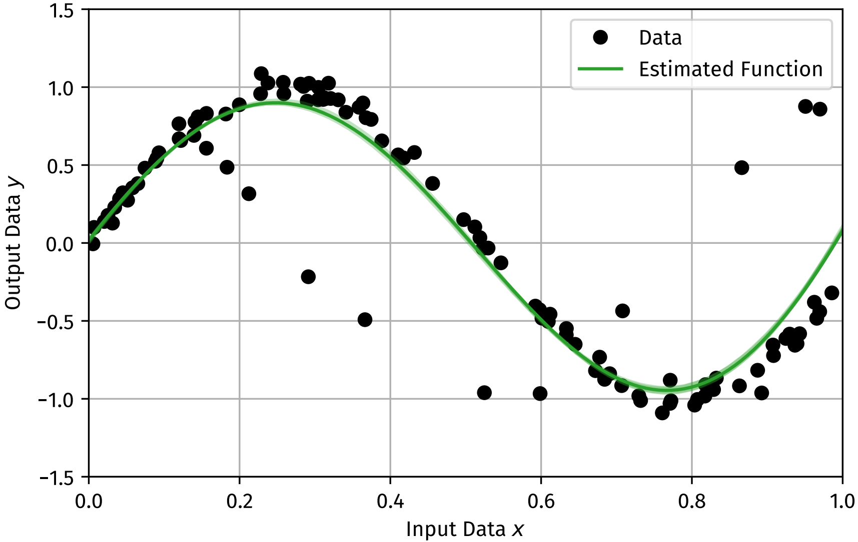
Story of Machine Learning

(in three slides!)

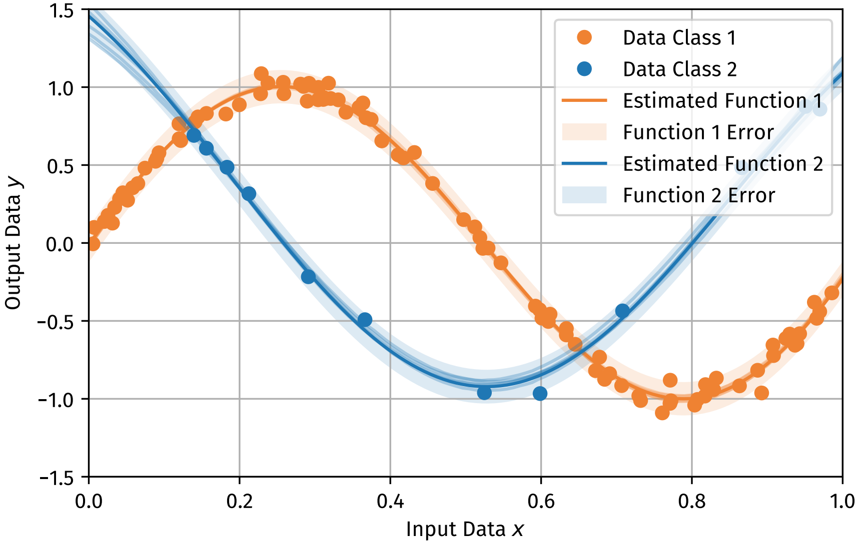
Story of ML: Past...



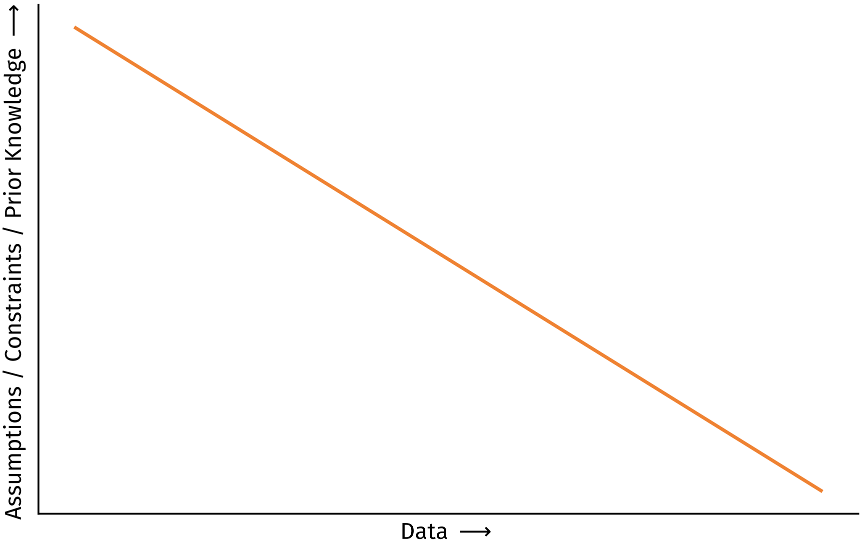
Story of ML: Present...



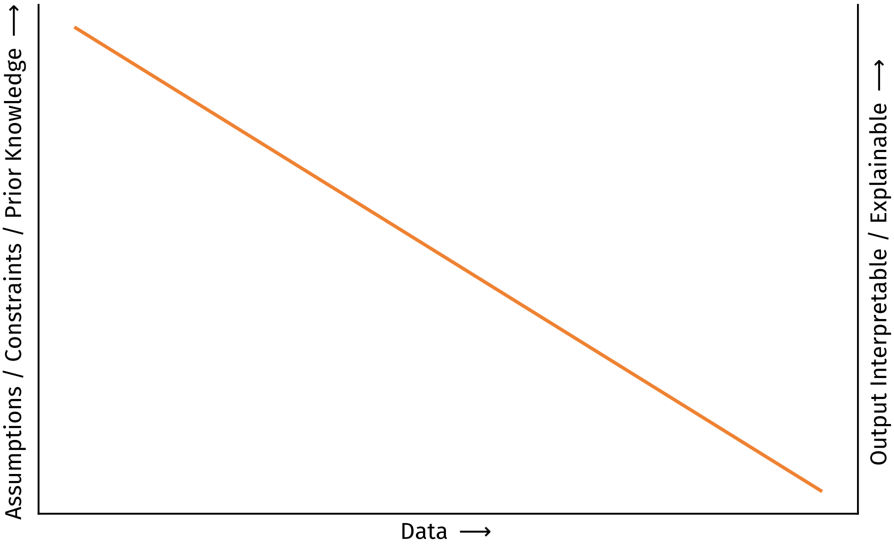
Story of ML: Future...



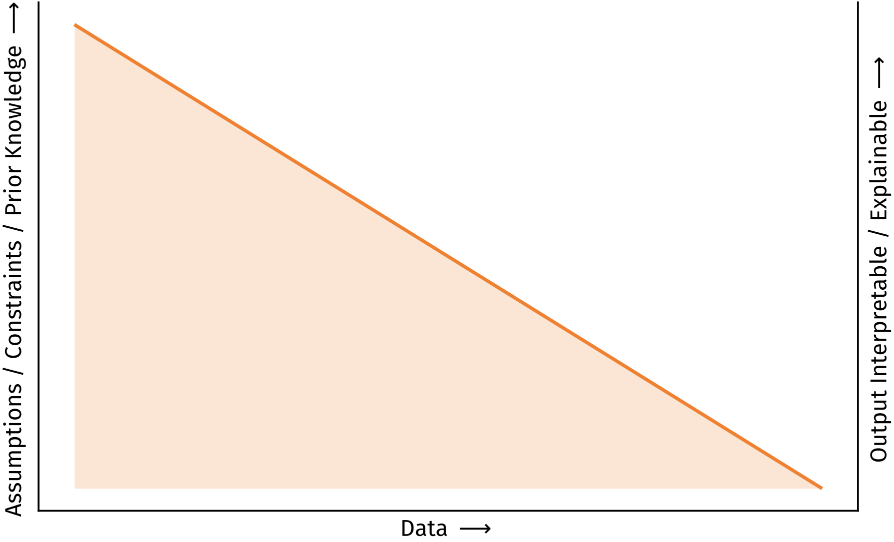
No free lunch



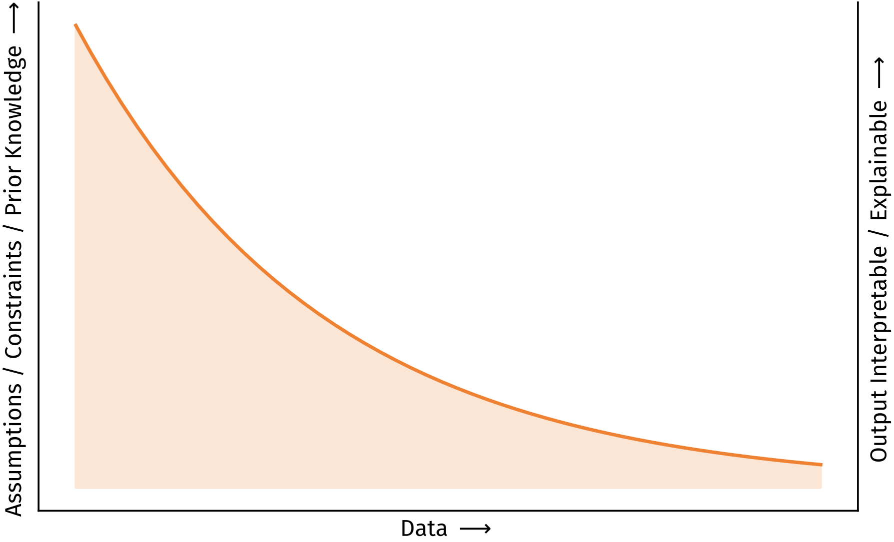
No free lunch



No free lunch



No free lunch (more realistic)



Why are we going to look at Bayesian methods?

"The Theory of probability is simply common sense reduced to calculus"

Pierre-Simon Laplace, 1749-1827

- We use Machine Learning to deal with the unknown
- Bayesian probability is the application of logic in the face of uncertainty

Why are we going to look at Bayesian methods?

"The Theory of probability is simply common sense reduced to calculus"

Pierre-Simon Laplace, 1749-1827

- We use Machine Learning to deal with the unknown
- Bayesian probability is the application of logic in the face of uncertainty
- Vision applications usually care about Inverse Probability

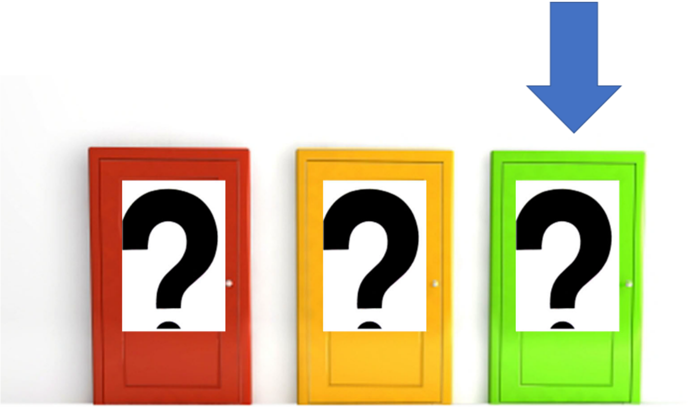
Whirlwind Introduction to Inverse Probabilities

Overview...

Monty Hall..

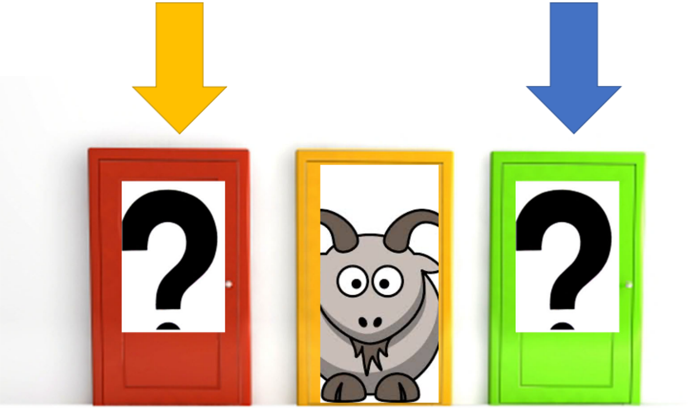
Monty Hall..

Monty Hall..

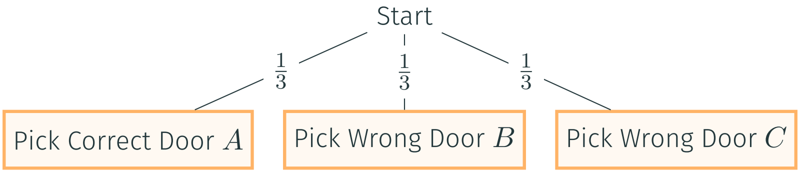


Monty Hall..

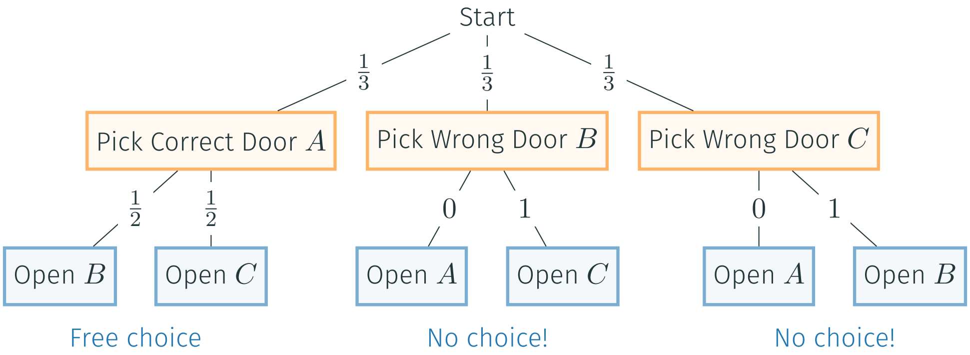
Monty Hall..



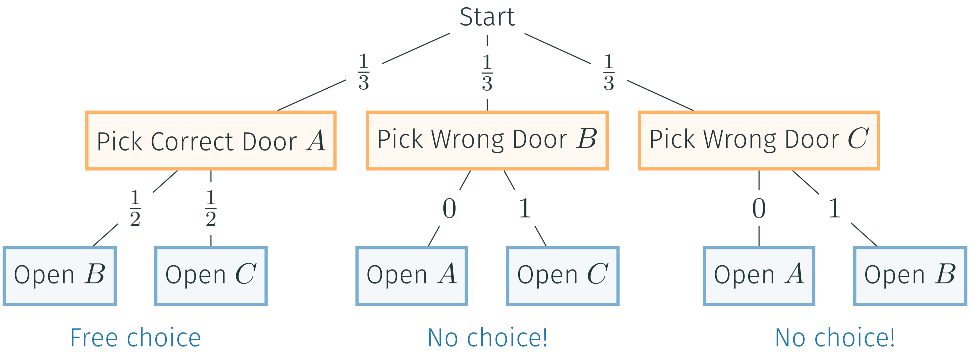
Monty Hall: Think of a tree diagram...



Monty Hall: Think of a tree diagram...



Monty Hall: Think of a tree diagram...



$\boxed{\frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}}$ $\boxed{\frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}}$ $\boxed{\frac{1}{3} \cdot \frac{0}{1} = \frac{0}{6}}$ $\boxed{\frac{1}{3} \cdot \frac{1}{1} = \frac{2}{6}}$ $\boxed{\frac{1}{3} \cdot \frac{0}{1} = \frac{0}{6}}$ $\boxed{\frac{1}{3} \cdot \frac{1}{2} = \frac{2}{6}}$

Stay = $\frac{1}{6} + \frac{1}{6} + \frac{0}{6} + \frac{0}{6} = \frac{1}{3}$ Switch = $\frac{2}{6} + \frac{2}{6} = \frac{2}{3}$

Monty Hall: How would we generate data (or simulate)?

Monty Hall: How would we generate data (or simulate)?

```
door_with_car = pick_random({1, 2, 3})
door_with_goat = {1, 2, 3} - door_with_car

door_picked = pick_random({1, 2, 3})

if door_picked == door_with_car:
    door_to_open = pick_random(door_with_goat)
else:
    door_to_open = door_with_goat - door_picked
```

Monty Hall: How would we generate data (or simulate)?

```
door_with_car = pick_random({1, 2, 3})           # 1/3 equal chance
door_with_goat = {1, 2, 3} - door_with_car

door_picked = pick_random({1, 2, 3})             # 1/3 equal chance

if door_picked == door_with_car:
    door_to_open = pick_random(door_with_goat)   # 1 times in 3
else:
    door_to_open = door_with_goat - door_picked  # 2 times in 3
```

Consider Modelling as a Generative Process

The Rules of Probability

- Notation $p(a) = p(a = \mathcal{A})$ where a is a particular outcome chosen from the set of all possible outcomes $\mathcal{A}$

$p(A \mid B)$ means "probability of A being the case given that B occurs"

- Probabilities in the range $0 \to 1$
- $0 =$ impossible
- $1 =$ certain
- Sum over all possible outcomes must be 1

The Rules of Probability

The Sum Rule (Marginalisation)

$$p(A = a) = \sum_{b \in \mathbb{B}} p(A = a, B = b)$$

- If continuous, rather than discrete, use densities and

$$p(A = a) = \int_{\mathbb{R}} p(A = a, B = b)\, db$$

The Product Rule

$$p(A = a, B = b) = p(A = a \mid B = b)\, p(B = b) = p(B = b \mid A = a)\, p(A = a)$$

- Bayes' rule follows from these rules..
- Only consistent approach for probability as "degree of plausibility" (Cox)

Bayes' Rule

- From the product rule

$$p(A = a \mid B = b) = \frac{p(B = b \mid A = a)\, p(A = a)}{p(B = b)}$$

- We can also condition on other information $\mathcal{H}$

$$p(a \mid b, \mathcal{H}) = \frac{p(b \mid a, \mathcal{H})\, p(a \mid \mathcal{H})}{p(b \mid \mathcal{H})}$$

- We give the parts of equation specific terms

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

Bayes' Rule

$$\text{Posterior Probability (after)} = \frac{\text{Likelihood (of event)} \times \text{Prior Probability (before)}}{\text{Evidence}}$$

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{p(\text{observations})}_{\text{Evidence}}}$$

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \underbrace{\frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt})\, p(\text{guilt})}}_{\text{Evidence}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \underbrace{\frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt})\, p(\text{guilt})}}_{\text{Evidence}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt})\, p(\text{guilt})}_{\text{Evidence}}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.
- Consider that the jury has to return a verdict *"beyond all reasonable doubt"*..

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

$$\underbrace{p(f \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid f)}^{\text{Likelihood}} \times \overbrace{p(f)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \qquad p(\mathcal{D}) = \sum_f p(\mathcal{D} \mid f)\, p(f)$$

Data $\mathcal{D} = \{X, Y\}$, pairs of inputs $\{x_n\}$ and outputs $\{y_n\}$, and functions f

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

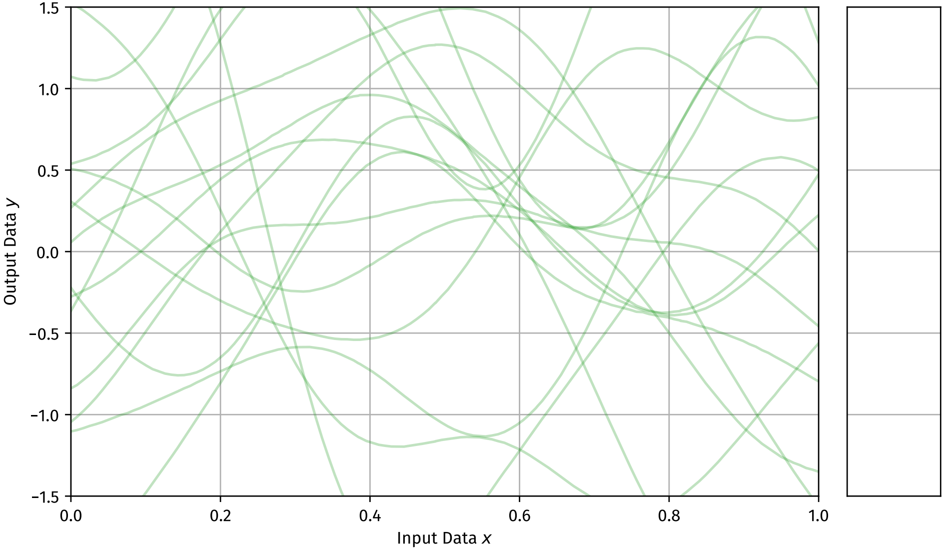
$$\underbrace{p(f \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid f)}^{\text{Likelihood}} \times \overbrace{p(f)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \qquad p(\mathcal{D}) = \sum_f p(\mathcal{D} \mid f)\, p(f)$$

Data $\mathcal{D} = \{X, Y\}$, pairs of inputs $\{x_n\}$ and outputs $\{y_n\}$, and functions f

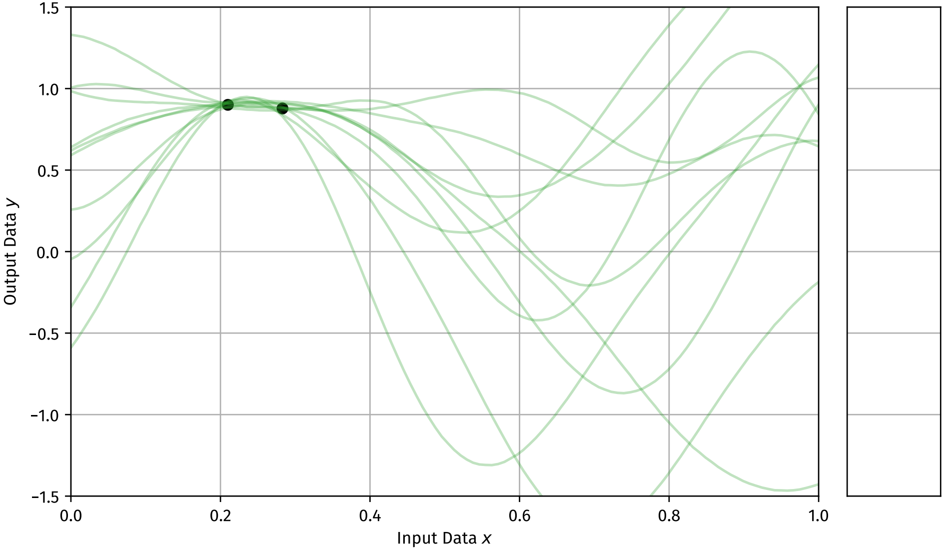
Average over functions to predict unknown output y^* for a **new input** x^*:

$$p(y^* \mid x^*, \mathcal{D}) = \sum_f p(y^* \mid x^*, f)\, p(f \mid \mathcal{D})$$

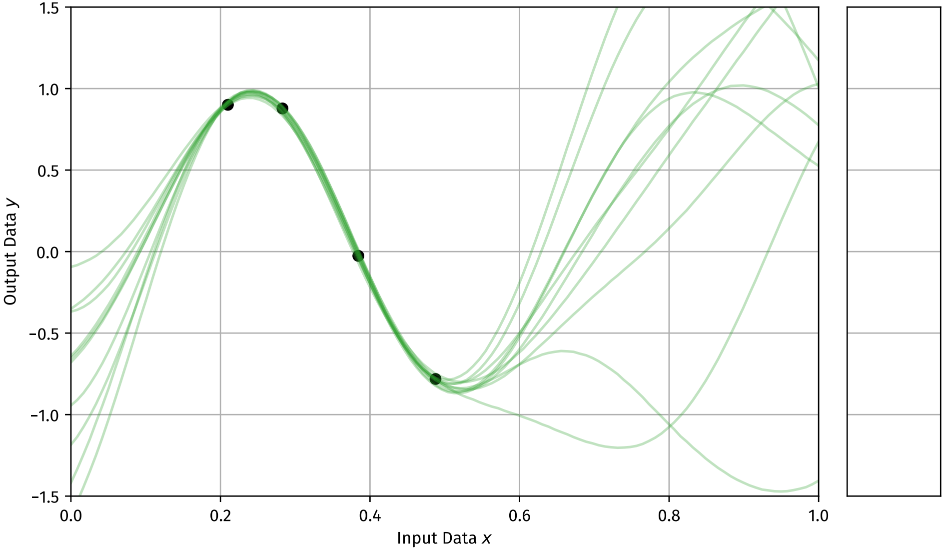
Prior over functions...



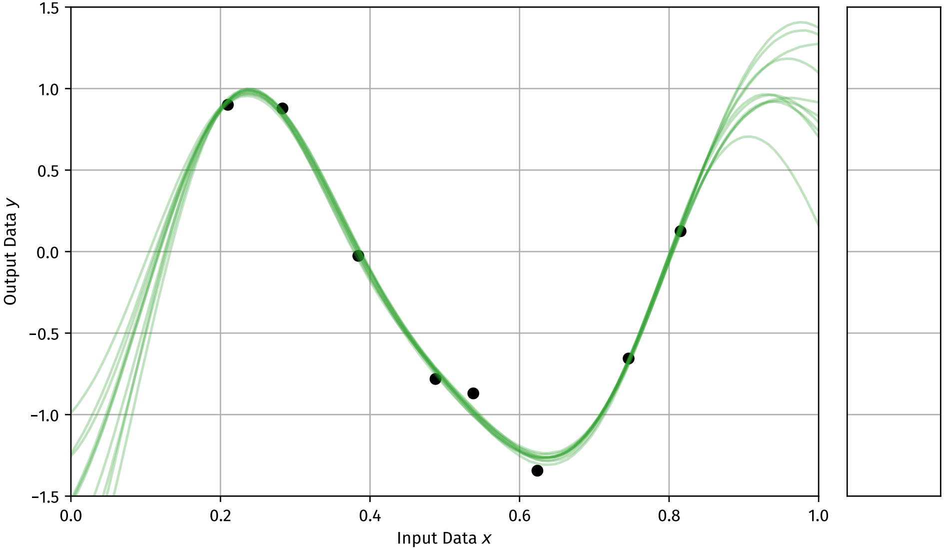
Combine prior with data...



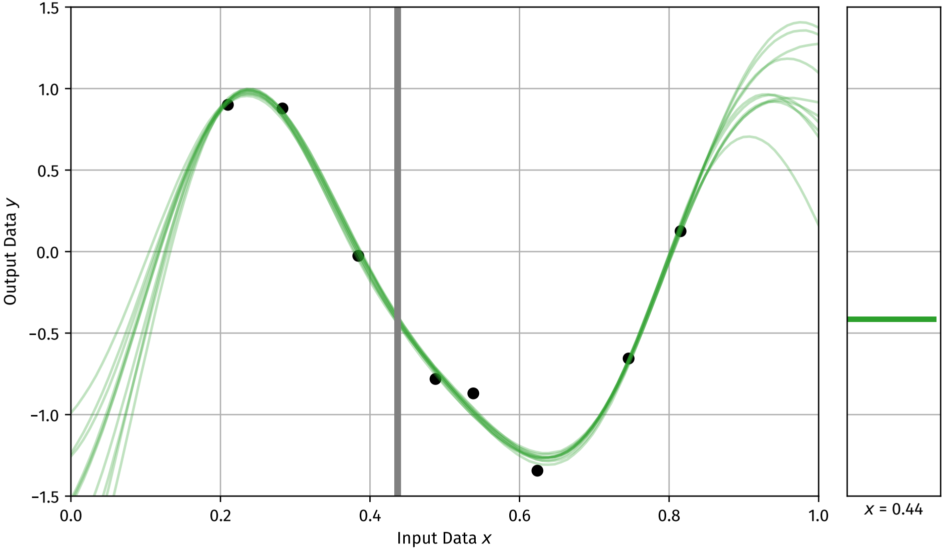
Combine prior with data...



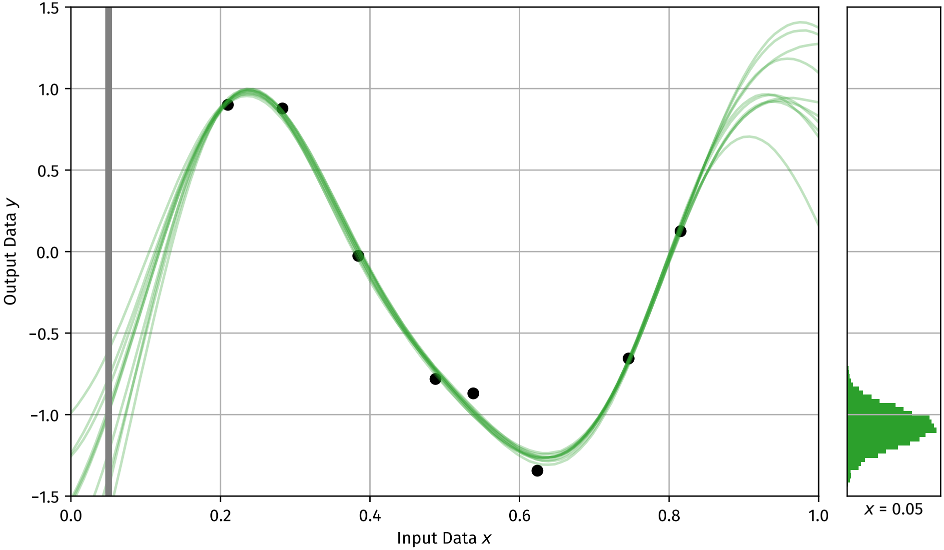
Combine prior with data...



Average over functions to predict...



Averaging over functions gives us (Epistemic) Uncertainty!



Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid \text{w})}^{\text{Likelihood}} \times \overbrace{p(\text{w})}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}}$$

Data $\mathcal{D} = X, Y$, pairs of inputs x_n and outputs y_n

Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid \text{w})}^{\text{Likelihood}} \times \overbrace{p(\text{w})}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \qquad p(\mathcal{D}) = \sum_{w} p(\mathcal{D} \mid \text{w})\, p(\text{w})$$

Data $\mathcal{D} = X, Y$, pairs of inputs x_n and outputs y_n

Prediction of output y^* for a new input x^*:

$$p(y^* \mid x^*, \mathcal{D}) = \sum_{w} p(y^* \mid x^*, w)\, p(w \mid \mathcal{D})$$

New thought process; no longer “Find the best parameters”, now “Find all the parameters that agree with the data”..

Model Selection

Overview...

Model selection (“Which model should I use?”)

- How much data do we need?

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say?

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say? The odds!

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say? The odds!

Science (and Computer Vision or Machine Learning) cannot prove things to be true via data

Science (and Computer Vision or Machine Learning) cannot prove things to be true via data

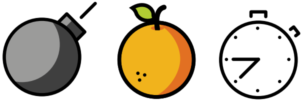
we can only demonstrate that things are inconsistent with data

Model selection illustration: Gravity!

Stable Diffusion: *"Drop cannonball and orange off the leaning tower of Pisa."*

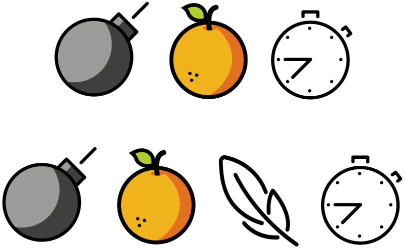
Model selection illustration: Gravity!

Stable Diffusion: *"Drop cannonball and orange off the leaning tower of Pisa."*



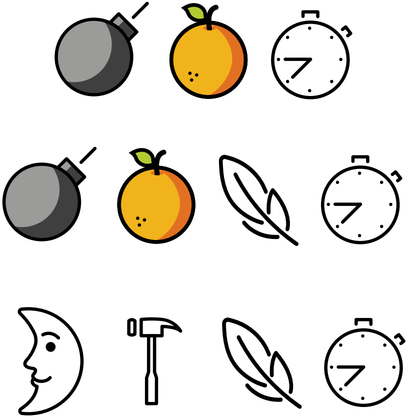
Model selection illustration: Gravity!

Stable Diffusion: *"Drop cannonball and orange off the leaning tower of Pisa."*



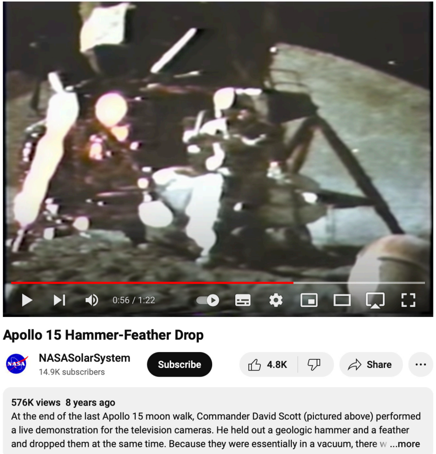
Model selection illustration: Gravity!

Stable Diffusion: *"Drop cannonball and orange off the leaning tower of Pisa."*



Model selection illustration: Gravity!

Stable Diffusion: “*Drop cannonball and orange off the leaning tower of Pisa.*”



Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid \mathrm{w})}^{\text{Likelihood}} \times \overbrace{p(\mathrm{w})}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M} = m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathrm{w}, \mathcal{M} = m)}^{\text{Likelihood under model}} \times \overbrace{p(\mathrm{w}, \mathcal{M} = m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M} = m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M}=m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathrm{w}, \mathcal{M}=m)}^{\text{Likelihood under model}} \times \overbrace{p(\mathrm{w}, \mathcal{M}=m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M}=m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

$$\underbrace{p(\mathcal{M}=m \mid \mathcal{D})}_{\text{Posterior for model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathcal{M}=m)}^{\text{Evidence for model}} \times \overbrace{p(\mathcal{M}=m)}^{\text{Prior for model}}}{\underbrace{p(\mathcal{D})}_{\text{Data}}}$$

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M} = m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathrm{w}, \mathcal{M} = m)}^{\text{Likelihood under model}} \times \overbrace{p(\mathrm{w}, \mathcal{M} = m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M} = m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

$$\underbrace{p(\mathcal{M} = m \mid \mathcal{D})}_{\text{Posterior for model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathcal{M} = m)}^{\text{Evidence for model}} \times \overbrace{p(\mathcal{M} = m)}^{\text{Prior for model}}}{\underbrace{p(\mathcal{D})}_{\text{Data}}}$$

If prior over models is equal, we compare via the **Evidence for the Model**: $p(\mathcal{D} \mid \mathcal{M} = m)$

Model selection example

Fitting polynomial models to data under Gaussian noise, $\varepsilon_n \sim \mathcal{N}(0, \sigma^2)$:

$$\text{Model 1}: y_n = a_0 + a_1 x_n + \varepsilon_n$$

$$\text{Model 2}: y_n = a_0 + a_1 x_n + a_2 x^2 + \varepsilon_n$$

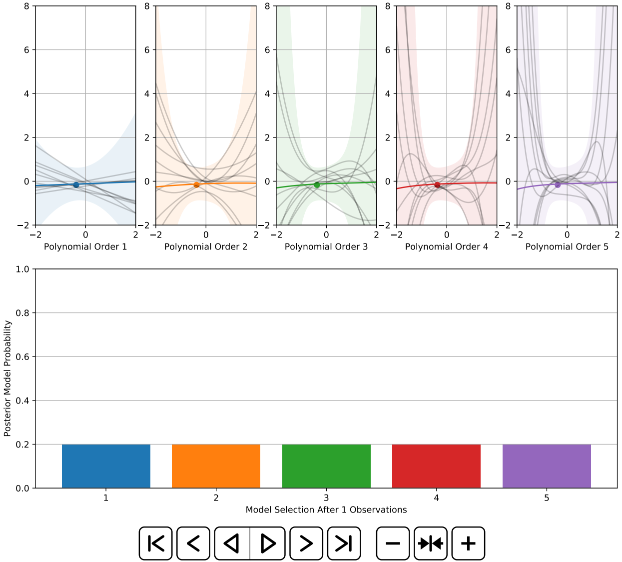
$$\text{Model 3}: y_n = a_0 + a_1 x_n + a_2 x^2 + a_3 x^3 + \varepsilon_n$$

$$\text{Model 4}: y_n = a_0 + a_1 x_n + a_2 x^2 + a_3 x^3 + a_4 x^4 + \varepsilon_n$$

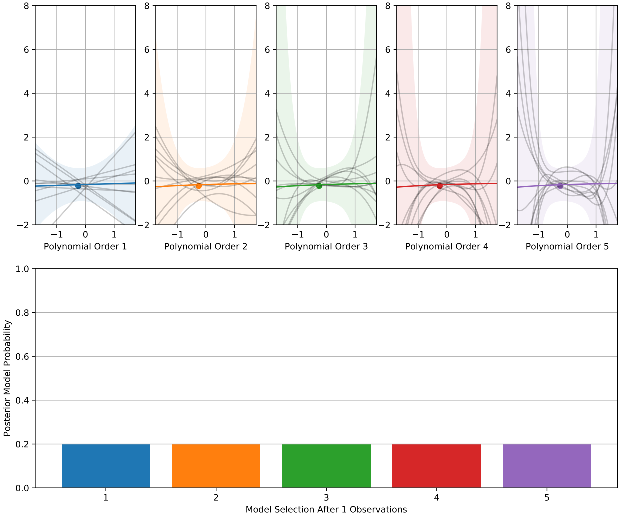
$$\text{Model 5}: y_n = a_0 + a_1 x_n + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + \varepsilon_n$$

Parameters $w_m = [a_0, \ldots, a_m]$ for model m, where $m \in [1, \ldots, 5]$.

Model selection example



Model selection example (more noise)



Evaluation

Overview...

- Motivation
- No Free Lunch
- Whirlwind Introduction to Inverse Probabilities
- Model Selection
- Evaluation
- Bayesian Machine Learning: Simple Example
- Why don't we do Model Selection in Vision?
- Illustrative Examples of Uncertainty in Vision
- Illustration: Structured Uncertainty Prediction Networks (SUPN)
- Conclusions

Evaluation in Computer Vision

We have made a cool new model with great advances..

- How do we evaluate empirical performance?
- What do we care about?

Evaluation in Computer Vision

We have made a cool new model with great advances..

- How do we evaluate empirical performance?
- What do we care about?

We probably want:

- Fair comparisons (comparing methods)
- Useful comparisons (what can we learn)
- Understand limitations (how confident should we be)

Illustrative example: Which method is best?

Model name	RMSE Mean $\downarrow$
MD2 Boot+Log	3.850
MD2 Boot+Self	3.795
Diagonal	4.000
SUPN Boot+Log	4.071
SUPN Boot+Self	4.091

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850
MD2 Boot+Self	3.795
Diagonal	4.000
SUPN Boot+Log	4.071
SUPN Boot+Self	4.091

- We are showing the Root of the Mean Squared Error
- We want lower scores so the MD2 model is the best?

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850 (1.370)
MD2 Boot+Self	3.795 (1.397)
Diagonal	4.000 (1.457)
SUPN Boot+Log	4.071 (1.489)
SUPN Boot+Self	4.091 (1.442)

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850 (1.370)
MD2 Boot+Self	3.795 (1.397)
Diagonal	4.000 (1.457)
SUPN Boot+Log	4.071 (1.489)
SUPN Boot+Self	4.091 (1.442)

- We now have standard errors (i.e. the standard deviation)
- Does this change things?

Illustrative example: Which method is best?

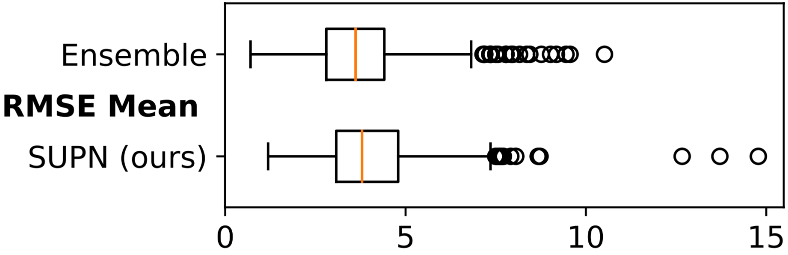


Figure 2. Box plot illustrating the strong distribution overlap between the original ensemble and the trained SUPN model for Boot+Log RMSE mean.

Illustrative example: Which method is best?

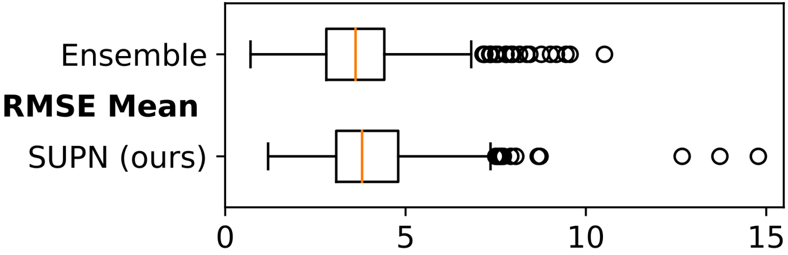


Figure 2. Box plot illustrating the strong distribution overlap between the original ensemble and the trained SUPN model for Boot+Log RMSE mean.

- Looking at the distributions reveals very similar performance
- The SUPN mean is skewed by only 3 outliers!

Important slide acknowledgements!

Important slide acknowledgements!

Illustrations taken from the excellent new text book from Simon Prince:

Understanding Deep Learning, Simon J.D. Prince, MIT Press

Final draft available on the website:

`https://udlbook.github.io/udlbook/`

Important slide acknowledgements!

Illustrations taken from the excellent new text book from Simon Prince:

Understanding Deep Learning, Simon J.D. Prince, MIT Press

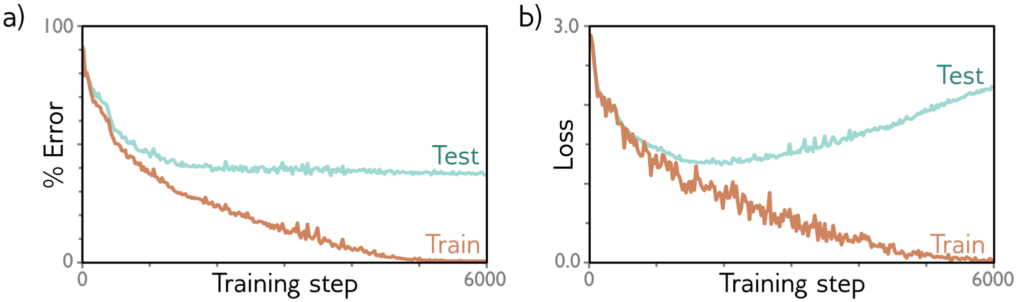
Final draft available on the website:

`https://udlbook.github.io/udlbook/`

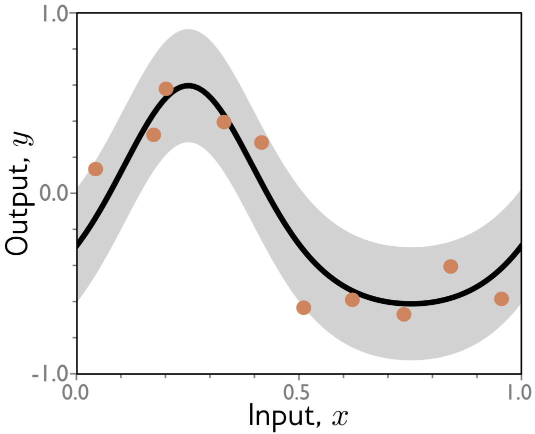
I strongly recommend you take a look!

Evaluation: Training vs Test

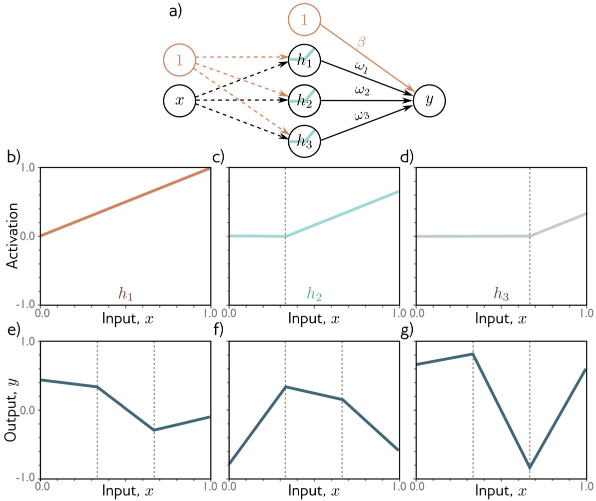
Evaluation: Training vs Test



Evaluation: Regression example



Evaluation: Toy regression model



Evaluation: Noise, Bias and Variance

$$y_n = f(x_n;\phi) + \varepsilon_n, \qquad n = 1 \dots N, \qquad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n;\phi) + \varepsilon_n, \qquad n = 1\ldots N, \qquad \varepsilon_n \sim \mathcal{N}(0,\sigma^2)$$

$$\begin{aligned} y(x) &= f(x;\phi) + \varepsilon \\ \mu(x) &= \mathbb{E}_y[\,y(x)\,] = \int y(x)\,p(y \mid x)\,dy \end{aligned}$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n;\phi) + \varepsilon_n, \qquad n = 1 \ldots N, \qquad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

$$y(x) = f(x;\phi) + \varepsilon$$
$$\mu(x) = \mathbb{E}_y[\,y(x)\,] = \int y(x)\, p(y \mid x)\, dy$$

Now consider a least squares (L2) loss function:

$$\begin{aligned}
\mathcal{L}(x,\phi) &= \big(f(x;\phi) - y(x)\big)^2 \\
&= \big((f(x;\phi) - \mu(x)) + (\mu(x) - y(x))\big)^2 \\
&= \big(f(x;\phi) - \mu(x)\big)^2 + 2\big(f(x;\phi) - \mu(x)\big)\,\big(\mu(x) - y(x)\big) + \big(\mu(x) - y(x)\big)^2
\end{aligned}$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n;\phi) + \varepsilon_n, \qquad n = 1 \dots N, \qquad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

$$y(x) = f(x;\phi) + \varepsilon$$
$$\mu(x) = \mathbb{E}_y[\,y(x)\,] = \int y(x)\, p(y \mid x)\, dy$$

Now consider a least squares (L2) loss function:

$$\begin{aligned}
\mathcal{L}(x,\phi) &= \big(f(x;\phi) - y(x)\big)^2 \\
&= \big(\big(f(x;\phi) - \mu(x)\big) + \big(\mu(x) - y(x)\big)\big)^2 \\
&= \big(f(x;\phi) - \mu(x)\big)^2 + 2\big(f(x;\phi) - \mu(x)\big)\,\big(\mu(x) - y(x)\big) + \big(\mu(x) - y(x)\big)^2
\end{aligned}$$

$$\Rightarrow \mathbb{E}_y[\,\mathcal{L}(x,\phi)\,] = \big(f(x;\phi) - \mu(x)\big)^2 + \sigma^2$$

Evaluation: Noise, Bias and Variance

$$\mathbb{E}_y[\,\mathcal{L}(x,\phi)\,] = \big(f(x;\phi) - \mu(x)\big)^2 + \underbrace{\sigma^2}_{\text{noise}}$$

We have partitioned the expected loss into two terms, the second is some irreducible noise that comes with the observations (e.g. sensor noise).

Evaluation: Noise, Bias and Variance

$$\mathbb{E}_y[\,\mathcal{L}(x,\phi)\,] = \big(f(x;\phi) - \mu(x)\big)^2 + \underbrace{\sigma^2}_{\text{noise}}$$

We have partitioned the expected loss into two terms, the second is some irreducible noise that comes with the observations (e.g. sensor noise).

So far we have ignored the fact that we actually estimate parameters from a **sampled dataset** $\mathcal{D} = \{x_n, y_n\}_{n=1}^N$

$$f_\mu(x) := \mathbb{E}_\mathcal{D}[\,f(x;\phi(\mathcal{D}))\,]$$

$$\Rightarrow \big(f(x;\phi) - \mu(x)\big)^2 = \big((f(x;\phi_\mathcal{D}) - f_\mu(x)) + (f_\mu(x) - \mu(x))\big)^2$$

$$\Rightarrow \mathbb{E}_\mathcal{D}\Big[\big(f(x;\phi) - \mu(x)\big)^2\Big] = \mathbb{E}_\mathcal{D}\Big[\big(f(x;\phi_\mathcal{D}) - f_\mu(x)\big)^2\Big] + \big(f_\mu(x) - \mu(x)\big)^2$$

Evaluation: Noise, Bias and Variance

Therefore, if we take expectations over datasets, our expected loss comprises three terms:

$$\mathbb{E}_{\mathcal{D}}[\,\mathbb{E}_y[\,\mathcal{L}(x,\phi)\,]\,] = \underbrace{\mathbb{E}_{\mathcal{D}}\Big[\,\big(f(x;\phi_{\mathcal{D}}) - f_\mu(x)\big)^2\,\Big]}_{\text{variance}} + \underbrace{\big(f_\mu(x) - \mu(x)\big)^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$

Note: more complex for models other than least squares...

Evaluation: Noise, Bias and Variance

Therefore, if we take expectations over datasets, our expected loss comprises three terms:

$$\mathbb{E}_{\mathcal{D}}\big[\,\mathbb{E}_y\big[\,\mathcal{L}(x,\phi)\,\big]\,\big] = \underbrace{\mathbb{E}_{\mathcal{D}}\Big[\,\big(f(x;\phi_{\mathcal{D}}) - f_\mu(x)\big)^2\,\Big]}_{\text{variance}} + \underbrace{\big(f_\mu(x) - \mu(x)\big)^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$

Note: more complex for models other than least squares...

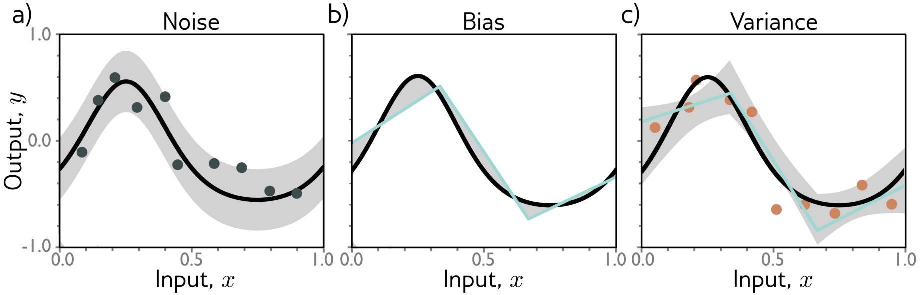
Noise Error in measurements (e.g. sensor noise, missing data, data mislabelled, ...)

Bias Systematic deviation from the true mean of the function (e.g. due to limitations of our model, ...)

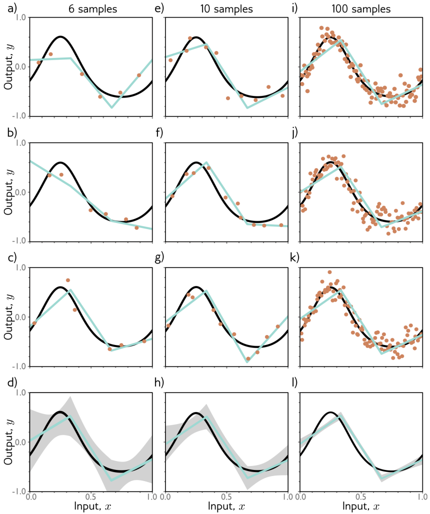
Variance Uncertainty in the fitting of our model due to limitations of the dataset (e.g. too few samples, dataset doesn't span the distribution, ...)

Evaluation: Noise, Bias and Variance

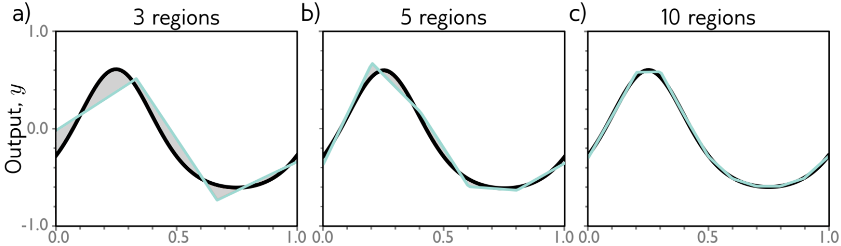
$$\mathbb{E}_{\mathcal{D}}\big[\mathbb{E}_y[\mathcal{L}(x,\phi)]\big] = \underbrace{\mathbb{E}_{\mathcal{D}}\Big[\big(f(x;\phi_{\mathcal{D}}) - \overbrace{f_{\mu}(x)}^{\text{best possible model from infinite data}}\big)^2\Big]}_{\text{variance}} + \underbrace{\big(f_{\mu}(x) - \overbrace{\mu(x)}^{\text{true function}}\big)^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$



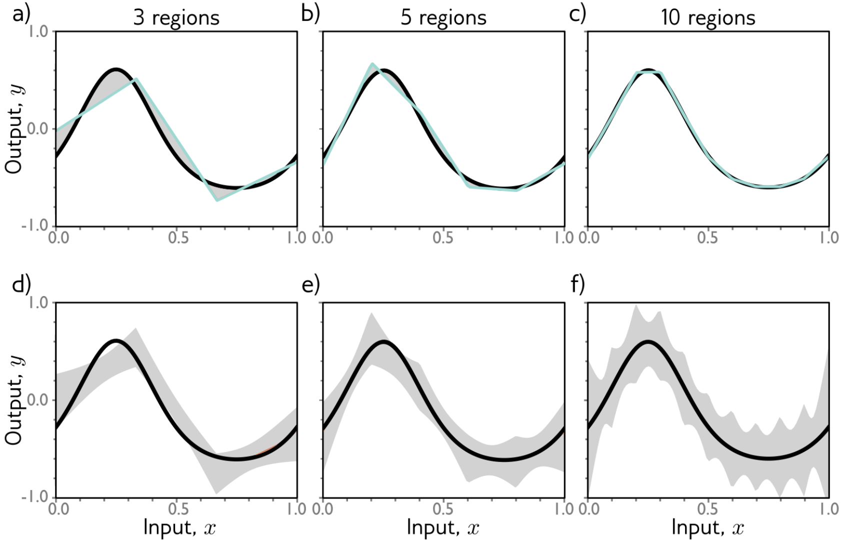
Evaluation: Variance reduction



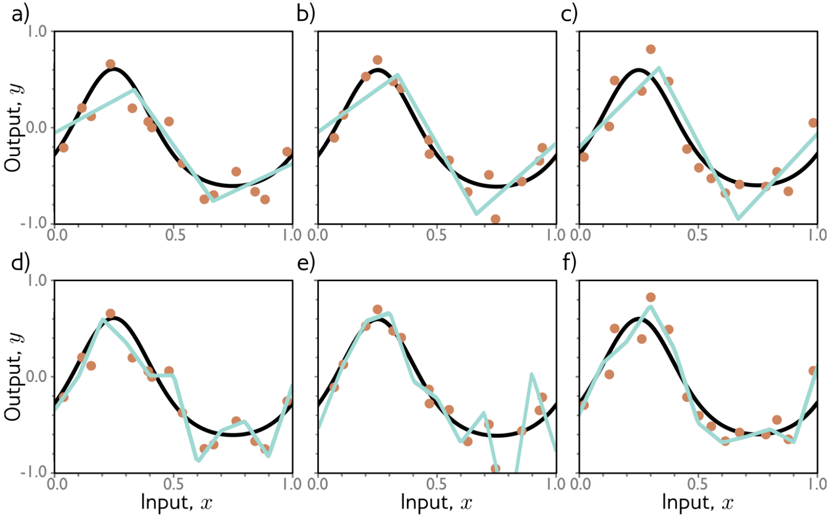
Evaluation: Bias reduction



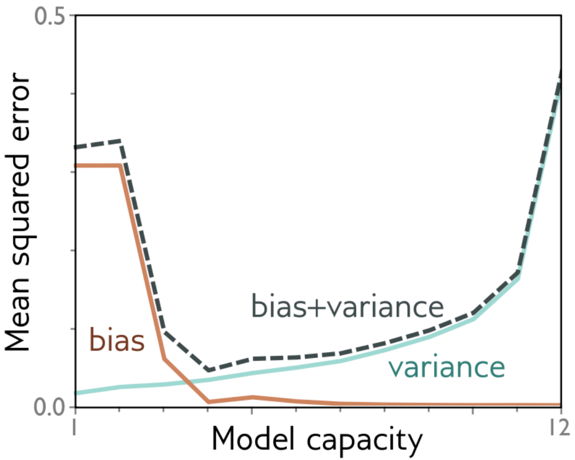
Evaluation: Bias reduction



Evaluation: Overfitting



Evaluation: Bias-Variance tradeoff



Evaluation: Statistical Significance..

- If you have some statistics training you may be familiar with the concept of statistical significance
- At high-level, this concerns the overlapping of (error) distributions and whether you could distinguish reliably between two distributions

Evaluation: Statistical Significance..

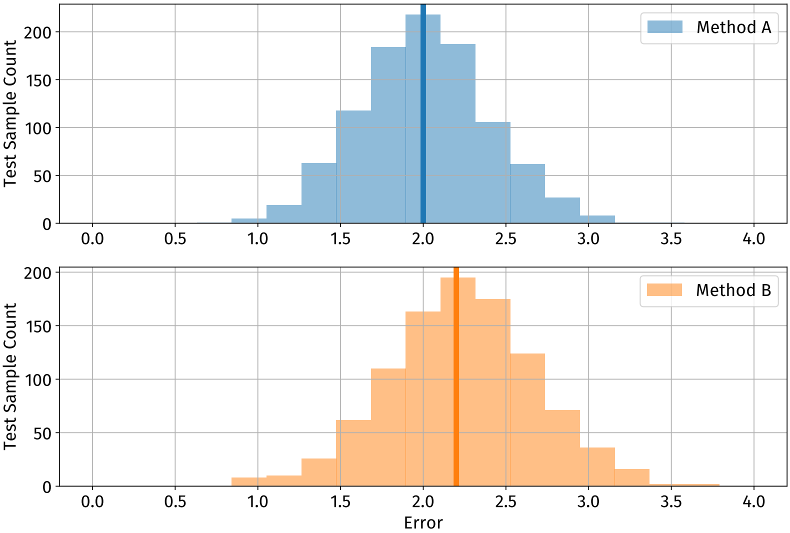
- If you have some statistics training you may be familiar with the concept of statistical significance
- At high-level, this concerns the overlapping of (error) distributions and whether you could distinguish reliably between two distributions
- It is possible to conduct formal tests...
- *My Advice:* try to avoid this as it is prone to all kinds of subtle decisions and arguments. Better to show the raw data in a useful form and people can perform their own assessment...

Evaluation: Example with statistical tests..

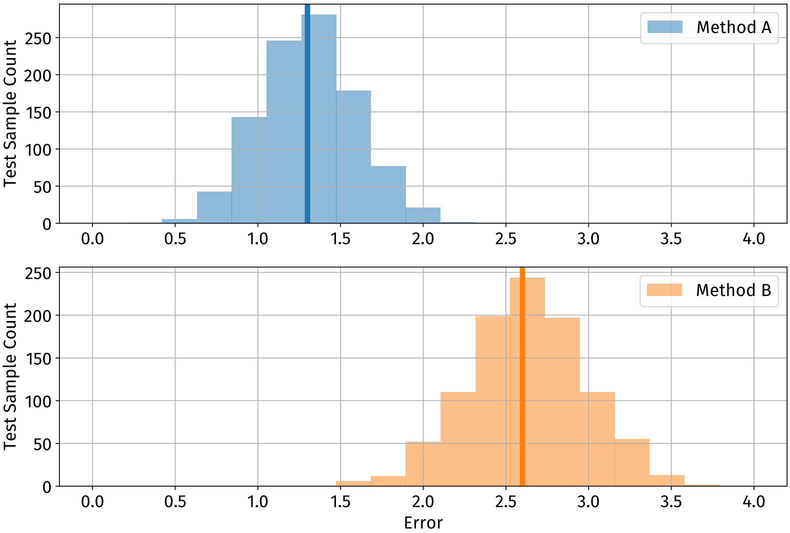
Table 1. Mean gap performance for various test functions; higher is better. The upper table shows the results after 50 objective function evaluations and the lower table after 100 evaluations. Due to computational cost, Warped GP results are only reported for 50 evaluations. Methods not significantly different from the best performing method with respect by a two-sided paired Wilcoxon signed-rank test at a 5% significance level over 20 repetitions are shown in bold (Malkomes & Garnett, 2018). For results in terms of regret, see the supplement.

Benchmark	Evals	Dim	Properties	GP	Warped GP	Homosced GP	Heterosced GP	LGP
Hartmann	50	6	boring	**0.959**	0.537	0.881	**0.973**	**0.937**
Griewank	50	2	oscillatory	**0.914**	0.493	0.752	**0.913**	**0.897**
Shubert	50	2	oscillatory	0.378	0.158	**0.378**	**0.480**	**0.593**
Ackley $[-10, 30]^d$	50	2	complicated, oscillatory	**0.924**	0.274	**0.892**	**0.912**	**0.927**
Cross In Tray	50	2	complicated, oscillatory	**0.954**	0.385	0.929	**0.977**	**0.945**
Holder table	50	2	complicated, oscillatory	0.939	0.896	0.900	0.931	**0.993**
Corrupted Holder Table	50	2	complicated, oscillatory	0.741	0.798	0.826	0.729	**0.896**

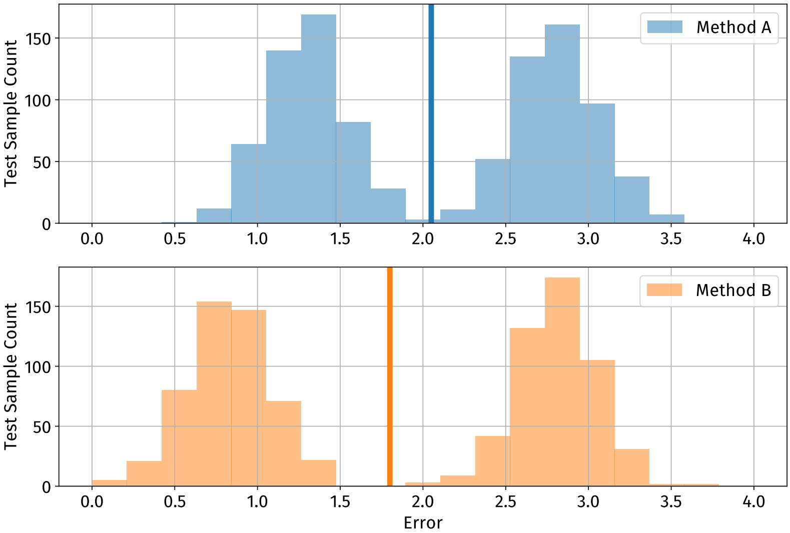
Evaluation: Look at test error distributions (e.g. histograms..)



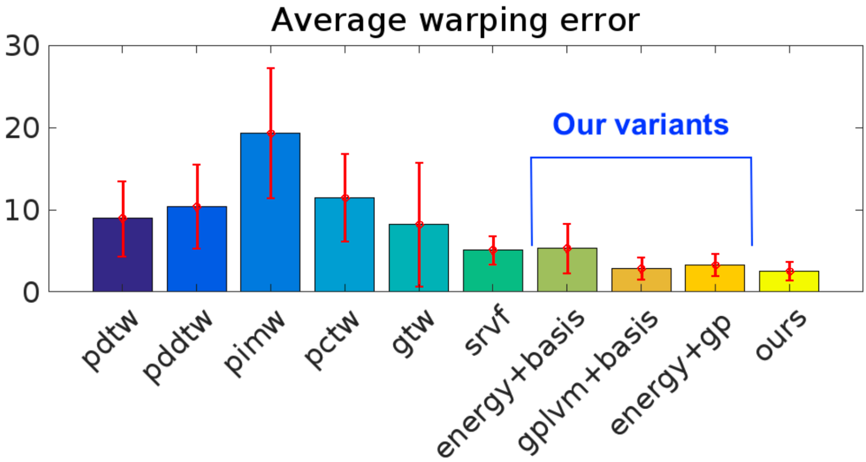
Evaluation: Look at test error distributions (e.g. histograms..)



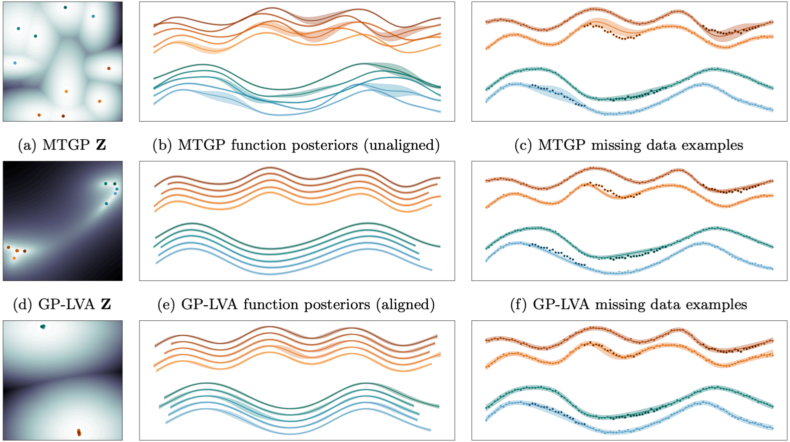
Evaluation: Look at test error distributions (e.g. histograms..)



Evaluation: Alignments example with error bars..



Evaluation: Alignments example with error bars..

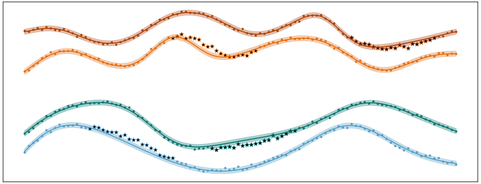


(a) MTGP $\mathbf{Z}$ (b) MTGP function posteriors (unaligned) (c) MTGP missing data examples

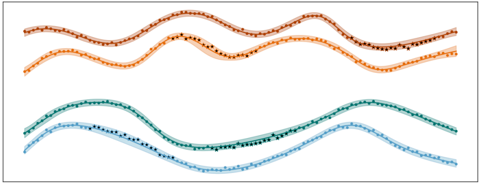
(d) GP-LVA $\mathbf{Z}$ (e) GP-LVA function posteriors (aligned) (f) GP-LVA missing data examples

(g) AMTGP $\mathbf{Z}$ (h) AMTGP function posteriors (aligned) (i) AMTGP missing data examples

Evaluation: Alignments example with error bars..

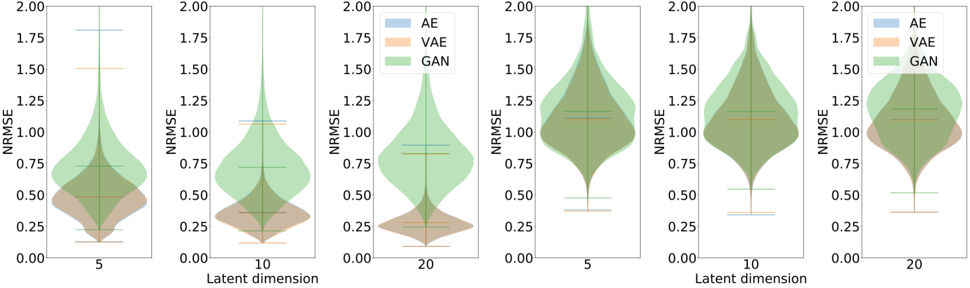


(f) GP-LVA missing data examples



(i) AMTGP missing data examples

Evaluation: Example with histograms (e.g. violin plots)..



(a) MNIST dataset (b) Shapes dataset

Evaluation: Histograms with illustrations!

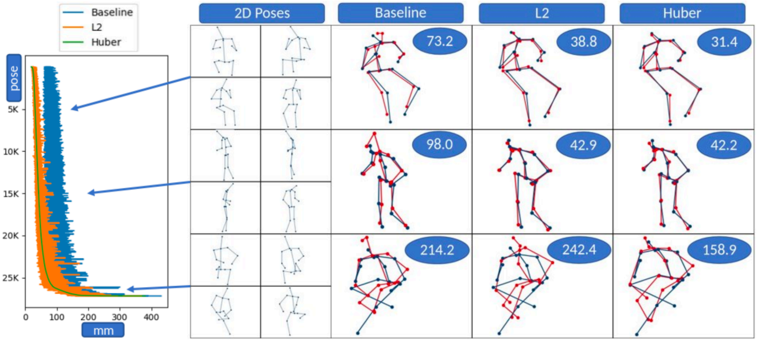


Figure 1: Error distribution on Human36M using our multi-camera model. Results are included for the untrained (Baseline) network and for the learnt shapes for the energy functions using both the ℓ_2 and the Huber loss. For visual clarity, we sort the datapoints by order of increasing error for the Huber case (i.e. our most effective approach). All models perform well on typical input instances, and fail in a limited number of cases. These outliers, however, have a noticeable, negative impact on the average error. We also provide reconstruction examples from the low, medium and tail part of the distribution.

Bayesian Machine Learning: Simple Example

Overview...

Bayesian Machine Learning: Simple Example

Switch to demo notebook..

Why don’t we do Model Selection in Vision?

Overview...

Why don't we do Model Selection in Vision?

- History?
- Ablation Studies?
- Philosphical / Paradigm?

Why don’t we do Model Selection in Vision?

- History?
- Ablation Studies?
- Philosphical / Paradigm?

- It’s difficult to do!
- Integrals are often difficult or considered expensive

Why don’t we do Model Selection in Vision?

- History?
- Ablation Studies?
- Philosphical / Paradigm?

- It’s difficult to do!
- Integrals are often difficult or considered expensive

- Sometimes this is true, sometimes you can be clever/approximate

Why don't we do Model Selection in Vision?

- History?
- Ablation Studies?
- Philosphical / Paradigm?

- It's difficult to do!
- Integrals are often difficult or considered expensive

- Sometimes this is true, sometimes you can be clever/approximate

In era of **empirical** computer vision, how you evaluate is really **important**!

Illustrative Examples of Uncertainty in Vision

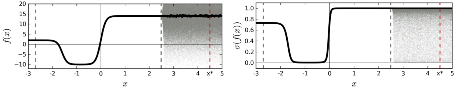
Overview...

Illustrative (practical) examples of uncertainty in vision

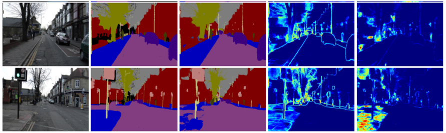
- Bayesian Deep Learning (BDL)
- Ensemble (deep) Approaches
- Structured Approximations

Uncertainty in vision: Bayesian Deep Learning

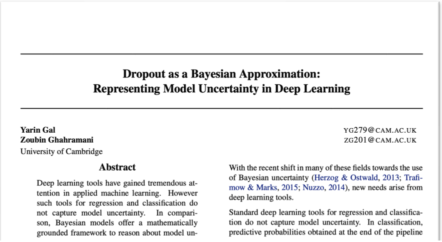
- Bayesian Deep Learning..



(a) Arbitrary function $f(\mathbf{x})$ as a function of data $\mathbf{x}$ (softmax *input*) (b) $\sigma(f(\mathbf{x}))$ as a function of data $\mathbf{x}$ (softmax *output*)



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



Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning

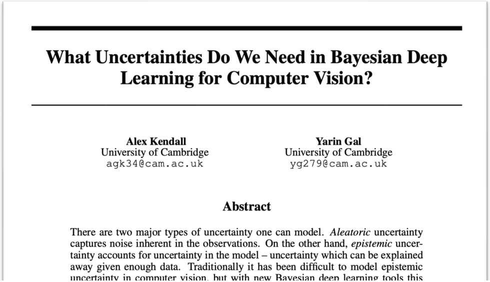
Yarin Gal YG279@CAM.AC.UK
Zoubin Ghahramani ZG201@CAM.AC.UK
University of Cambridge

Abstract

Deep learning tools have gained tremendous attention in applied machine learning. However such tools for regression and classification do not capture model uncertainty. In comparison, Bayesian models offer a mathematically grounded framework to reason about model un-

With the recent shift in many of these fields towards the use of Bayesian uncertainty (Herzog & Ostwald, 2013; Trafimow & Marks, 2015; Nuzzo, 2014), new needs arise from deep learning tools.

Standard deep learning tools for regression and classification do not capture model uncertainty. In classification, predictive probabilities obtained at the end of the pipeline



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

Alex Kendall
University of Cambridge
agk34@cam.ac.uk

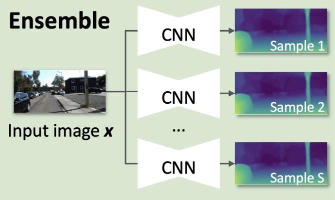
Yarin Gal
University of Cambridge
yg279@cam.ac.uk

Abstract

There are two major types of uncertainty one can model. *Aleatoric* uncertainty captures noise inherent in the observations. On the other hand, *epistemic* uncertainty accounts for uncertainty in the model – uncertainty which can be explained away given enough data. Traditionally it has been difficult to model epistemic uncertainty in computer vision, but with new Bayesian deep learning tools this

Uncertainty in vision: Deep ensembles

- Deep Ensembles..



Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles

Balaji Lakshminarayanan **Alexander Pritzel** **Charles Blundell**
DeepMind
{balajiln,apritzel,cblundell}@google.com

Abstract

On the uncertainty of self-supervised monocular depth estimation

Matteo Poggi Filippo Aleotti Fabio Tosi Stefano Mattoccia
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
{m.poggi, filippo.aleotti2, fabio.tosi5, stefano.mattoccia }@unibo.it

Abstract

Self-supervised paradigms for monocular depth estimation are very appealing since they do not require ground truth annotations at all. Despite the astonishing results yielded by such methodologies, learning to reason about the uncertainty of the estimated depth maps is of paramount importance for practical applications, yet uncharted in the literature. Purposely, we explore for the first time how to estimate the uncertainty for this task and how this affects depth accuracy, proposing a novel peculiar technique specifically designed for self-supervised approaches. On the standard KITTI dataset, we exhaustively assess the performance of each method with different self-supervised paradigms. Such evaluation highlights that our proposal i) always improves depth accuracy significantly and ii) yields state-of-the-art results concerning uncertainty estimation when training on sequences and competitive results uniquely deploying stereo pairs.

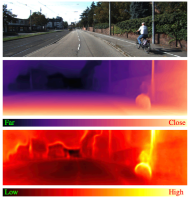


Figure 1. **How much can we trust self-supervised monocular depth estimation?** From a single input image (top) we estimate depth (middle) and uncertainty (bottom) maps. Best with colors.

Uncertainty in vision: Deep ensembles

- Deep Ensembles..

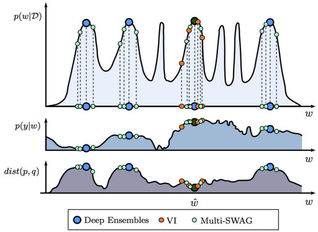
A Simple Baseline for Bayesian Uncertainty in Deep Learning

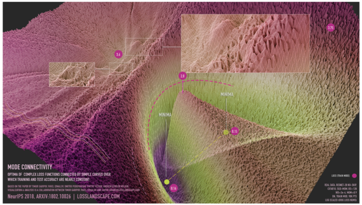
Wesley J. Maddox[*1] **Timur Garipov**[*2] **Pavel Izmailov**[*1]
Dmitry Vetrov[2,3] **Andrew Gordon Wilson**[1]

[1] New York University
[2] Samsung AI Center Moscow
[3] Samsung-HSE Laboratory, National Research University Higher School of Economics

Abstract

We propose SWA-Gaussian (SWAG), a simple, scalable, and general purpose approach for uncertainty representation and calibration in deep learning. Stochastic Weight Averaging (SWA), which computes the first moment of stochastic gradient descent (SGD) iterates with a modified learning rate schedule, has recently been shown to improve generalization in deep learning. With SWAG, we fit a Gaussian using the SWA solution as the first moment and a low rank plus diagonal covariance also derived from the SGD iterates, forming an approximate posterior distribution over neural network weights; we then sample from this Gaussian distribution to

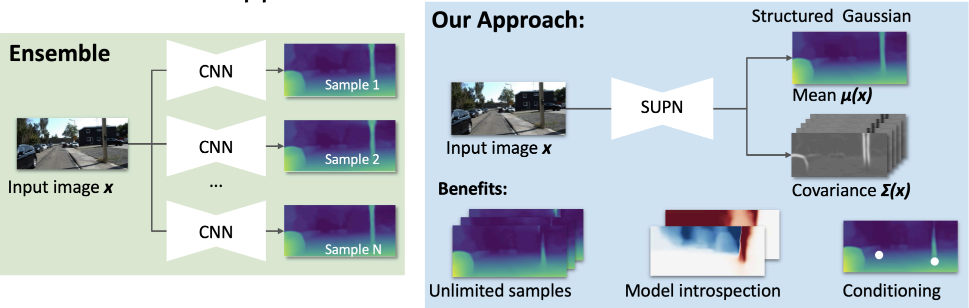




Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs
T. Garipov, P. Izmailov, D. Podoprikhin, D. Vetrov, A.G. Wilson
NeurIPS 2018

Uncertainty in vision: Structured approximation

- Structured approximation..



Learning Structured Gaussians to Approximate Deep Ensembles

Ivor J.A. Simpson
University of Sussex, UK
i.simpson@sussex.ac.uk

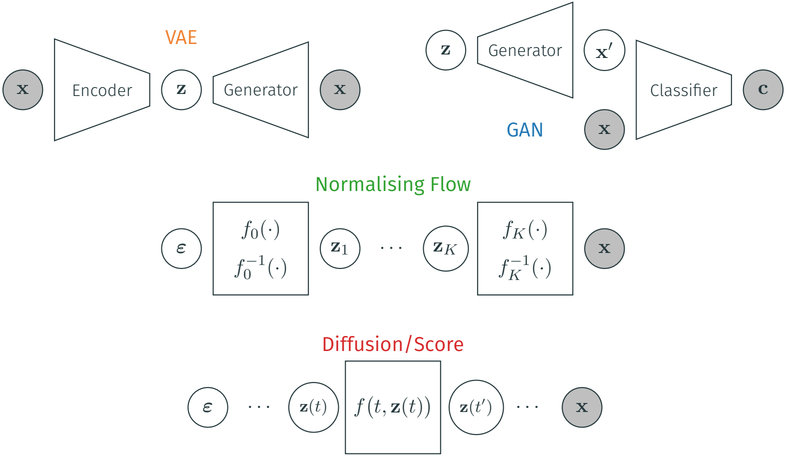
Sara Vicente
Niantic, UK
svicente@nianticlabs.com

Neill D.F. Campbell
University of Bath, UK
n.campbell@bath.ac.uk

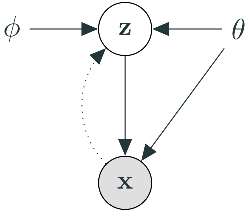
Illustration: Structured Uncertainty Prediction Networks (SUPN)

Overview...

Generative model zoo



Unreasonable expectations of generative models?

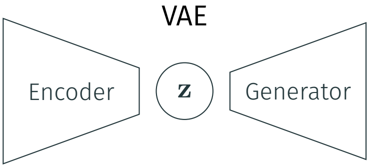


e.g. VAE with:

$$\mathbf{z} \in \mathbb{R}^M,$$
$$\mathbf{x} \in [0, 1]^{3 \times N \times N}$$

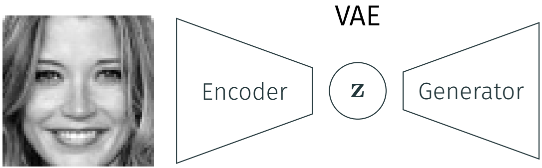
Figure 1: How many degrees of freedom are there in the image?

“VAEs produce overly smooth output”



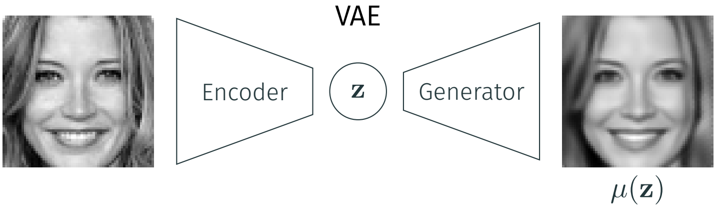
[Dorta et al. 2018]

“VAEs produce overly smooth output”



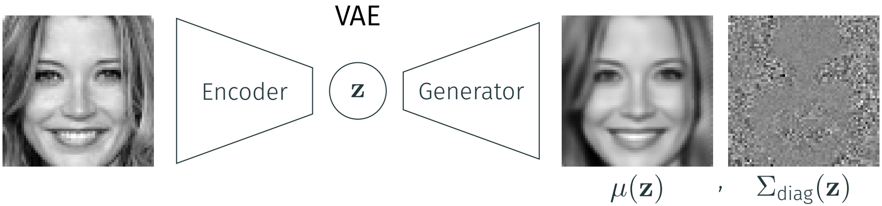
[Dorta et al. 2018]

“VAEs produce overly smooth output”



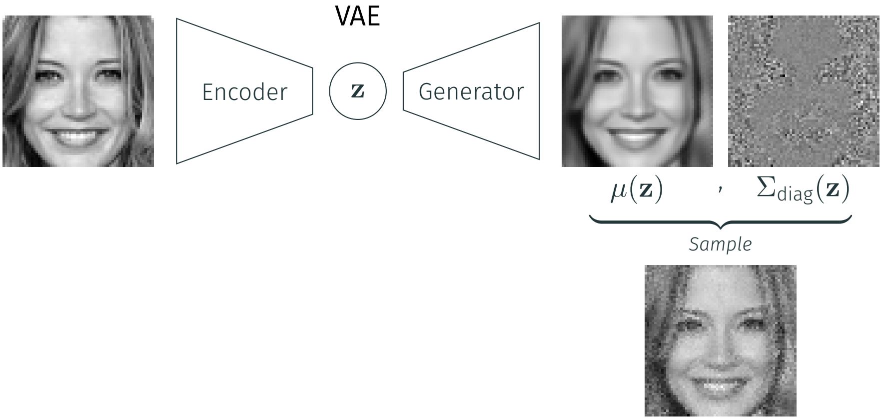
[Dorta et al. 2018]

"VAEs produce overly smooth output"



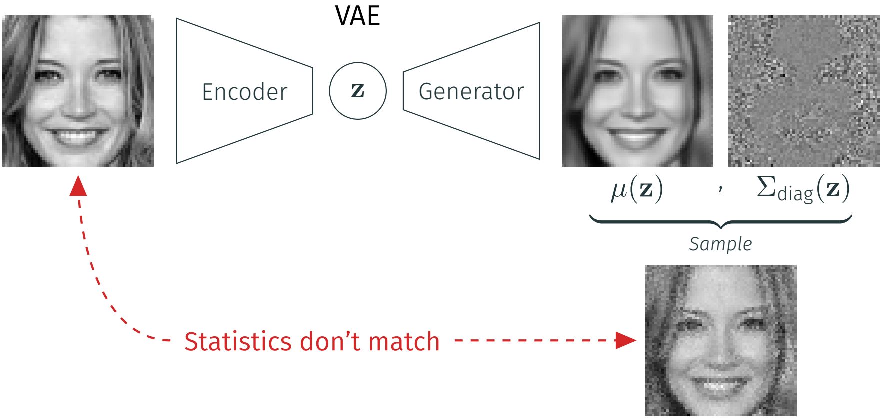
[Dorta et al. 2018]

"VAEs produce overly smooth output"



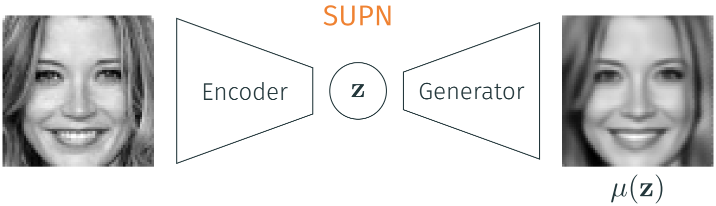
[Dorta et al. 2018]

“VAEs produce overly smooth output”



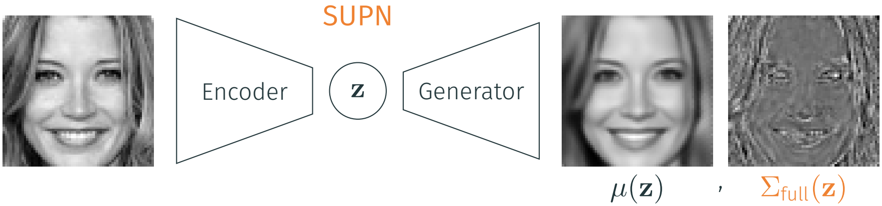
[Dorta et al. 2018]

“VAEs produce overly smooth output”



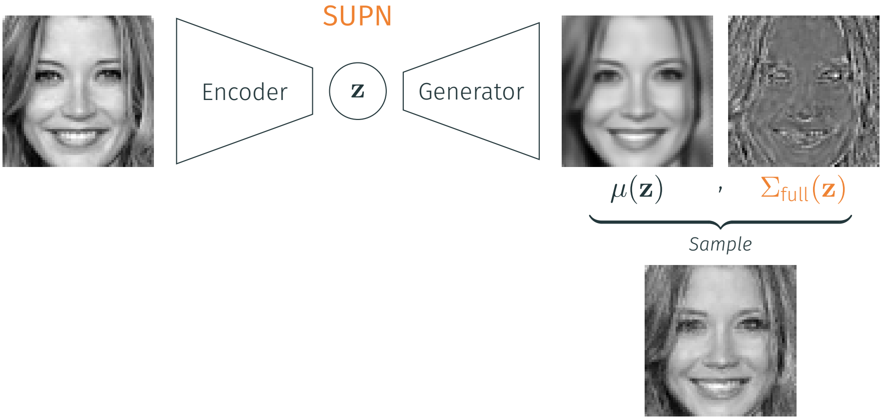
[Dorta et al. 2018]

“VAEs produce overly smooth output”



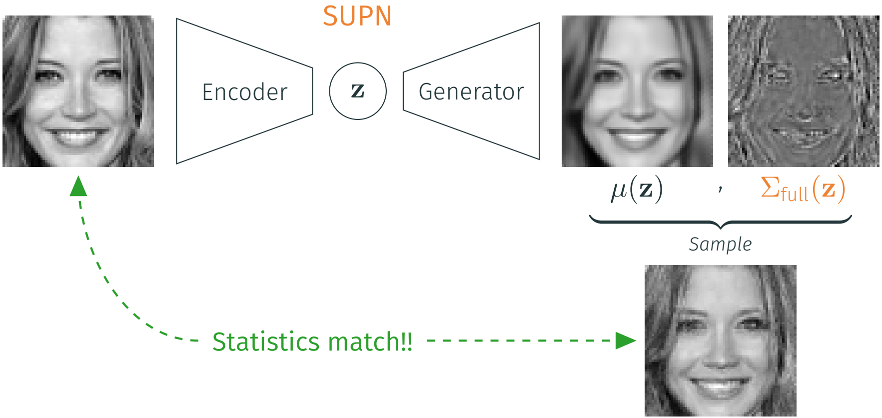
[Dorta et al. 2018]

“VAEs produce overly smooth output”



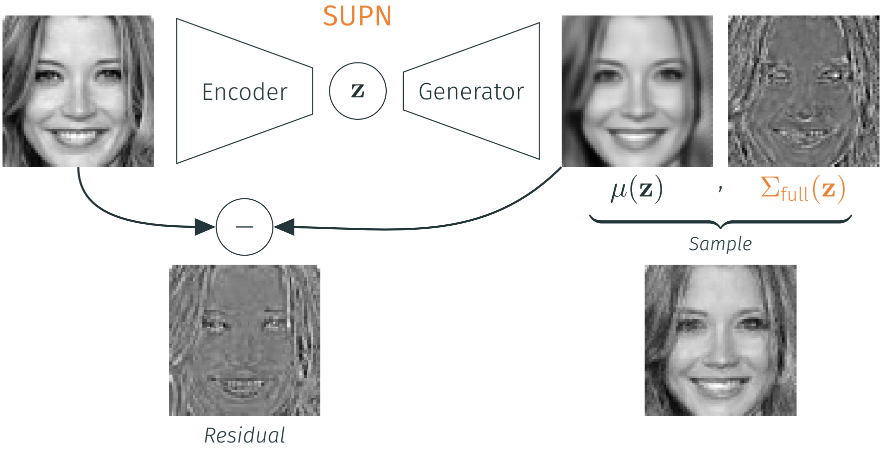
[Dorta et al. 2018]

"VAEs produce overly smooth output"



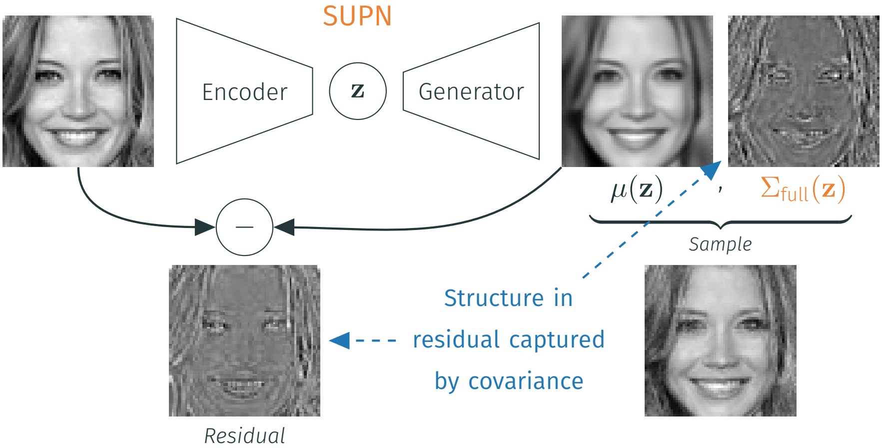
[Dorta et al. 2018]

"VAEs produce overly smooth output"



[Dorta et al. 2018]

"VAEs produce overly smooth output"



[Dorta et al. 2018]

Problem! Dense covariance $\mathcal{O}(N^2)$...

- **Problem**: $\Sigma_{\text{full}}(\mathbf{z})$ is quadratic in the number of pixels

Problem! Dense covariance $\mathcal{O}(N^2)$...

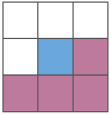
- **Problem**: $\Sigma_{\text{full}}(\mathbf{z})$ is quadratic in the number of pixels
- **Solution**: Sparse parameterisation of the Cholesky factor of the precision

$$\Sigma(\mathbf{z}) := \left[\Lambda(\mathbf{z})\right]^{-1} := \left[L_\Lambda(\mathbf{z})\, L_\Lambda^\top(\mathbf{z})\right]^{-1}$$

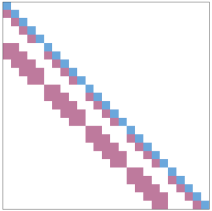
Problem! Dense covariance $\mathcal{O}(N^2)$...

- **Problem**: $\Sigma_{\text{full}}(\mathbf{z})$ is quadratic in the number of pixels
- **Solution**: Sparse parameterisation of the Cholesky factor of the precision

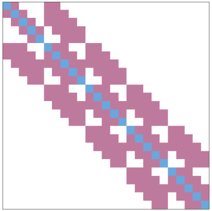
$$\Sigma(\mathbf{z}) := [\Lambda(\mathbf{z})]^{-1} := [L_\Lambda(\mathbf{z})\, L_\Lambda^\top(\mathbf{z})]^{-1}$$



Neighbourhood in image domain



Sparsity in the precision Cholesky matrix L_Λ



Sparsity in the precision matrix $\Lambda(\mathbf{z}) := \Sigma^{-1}(\mathbf{z})$

Efficient implementation

- Sparse parameterisation of the Cholesky factor of the precision

$$\Sigma(\mathbf{z}) := [\Lambda(\mathbf{z})]^{-1} := [L_\Lambda(\mathbf{z})\, L_\Lambda^\top(\mathbf{z})]^{-1}$$

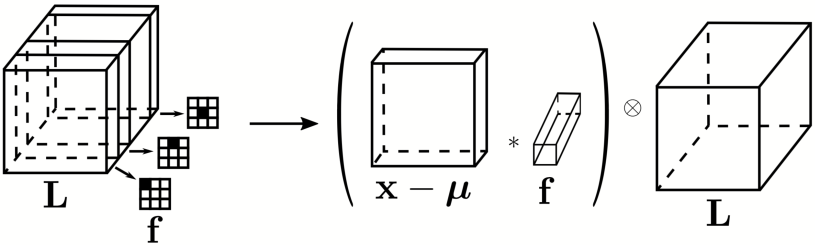


Figure 2: Implementation through convolutional structure: matrix-vector product in $\mathcal{O}(N)$

Examples of samples

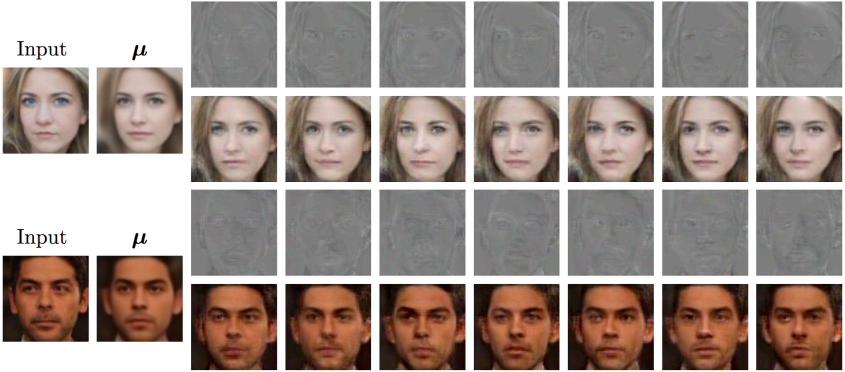


Figure 3: Variation in samples from the model on test data

Introspection of the captured covariance structure

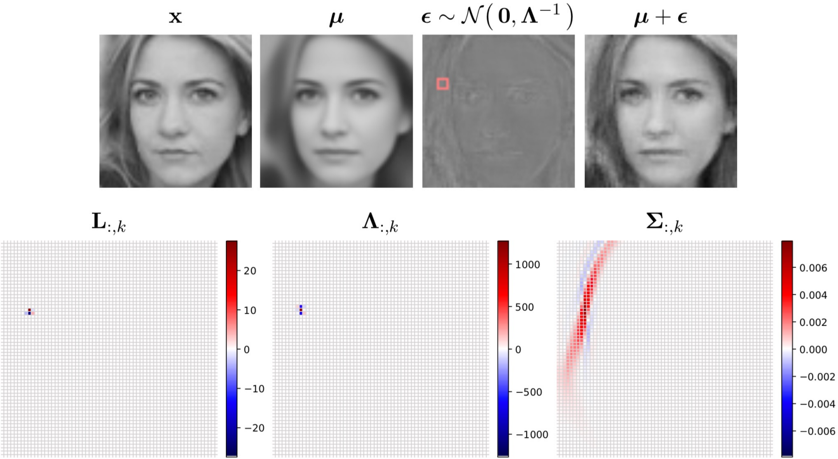


Figure 4: Visualisation of the learned correlations

Links to established concepts...

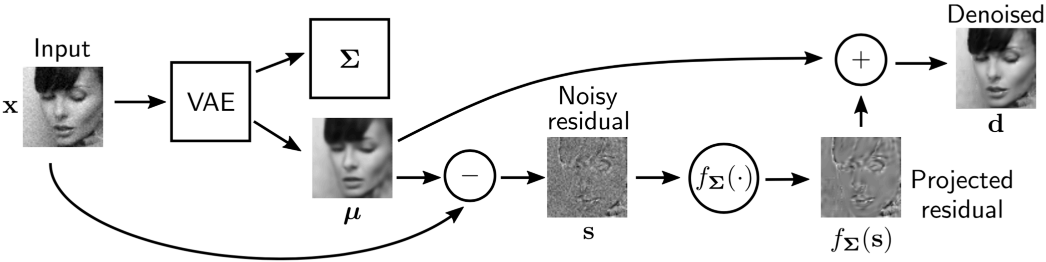
- Links to Conditional Random Field (CRF) models
    - a Gaussian CRF - e.g. "Regression Tree Fields" [Jancsary et al. 2012]
- Links to adaptive local regularisation models
    - e.g. locally adaptive TV or Laplacian based methods
- Links to Wavelet approaches
    - considering hierarchical extensions or combining fixed basis functions

Links to established concepts…

- Links to Conditional Random Field (CRF) models
    - a Gaussian CRF - e.g. "Regression Tree Fields" [Jancsary et al. 2012]
- Links to adaptive local regularisation models
    - e.g. locally adaptive TV or Laplacian based methods
- Links to Wavelet approaches
    - considering hierarchical extensions or combining fixed basis functions

- Things to be careful about
    - priors on sparse precision (consider Cholesky structure)
    - need to bound terms
    - *lots to say about these things…*

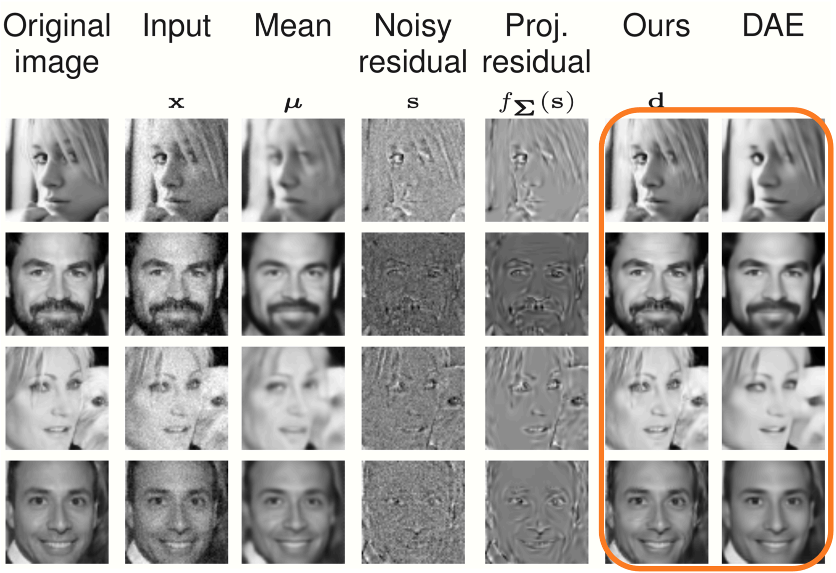
Testing with denoising...



Model	MSE	PSNR	SSIM
DAE	0.005 ± 0.003	28.89 ± 1.69	0.90 ± 0.03
SUPN	$\mathbf{0.003 \pm 0.001}$	$\mathbf{31.38 \pm 0.92}$	$\mathbf{0.92 \pm 0.02}$

Figure 5: Denoising example using SUPN (vs a denoising autoencoder). The SUPN model has only been trained as in a generative manner (i.e. as a prior).

Testing with denoising...



SUPN as a prior for inverse problems

- Consider a hierarchical model for the inverse problem

$$p(\mathbf{x}, \mathbf{z} \,|\, \mathbf{y}) \propto p(\mathbf{y} \,|\, \mathbf{x})\, p_{\mathcal{G}}(\mathbf{x} \,|\, \mathbf{z})\, p_{\mathcal{Z}}(\mathbf{z})$$

- We will take a MAP estimate for $\mathbf{z}$ rather than marginalising `:-(`

SUPN as a prior for inverse problems

- Consider a hierarchical model for the inverse problem

$$p(\mathbf{x}, \mathbf{z} \,|\, \mathbf{y}) \propto p(\mathbf{y} \,|\, \mathbf{x})\, p_{\mathcal{G}}(\mathbf{x} \,|\, \mathbf{z})\, p_{\mathcal{Z}}(\mathbf{z})$$

- We will take a MAP estimate for $\mathbf{z}$ rather than marginalising `:-(`
- From before (with a Gaussian observation likelihood) and $p_{\mathcal{Z}}(\mathbf{z}) \sim \mathcal{N}(\mathbf{0}, I)$

$$D(\mathbf{y}, A\,\mathbf{x}) := \frac{1}{2\sigma^2}\|A\,\mathbf{x} - \mathbf{y}\|_2^2$$

$$R(\mathbf{x}) := \min_{\mathbf{z}\in\mathcal{Z}}\; \log|\Sigma_\theta(\mathbf{z})| + \frac{1}{2}\|\mathbf{x} - \mu_\theta(\mathbf{z})\|^2_{\Sigma_\theta(\mathbf{z})} + \frac{1}{2}\|\mathbf{z}\|_2^2$$

- Where the *Generator* provides $\mathcal{N}(\mathbf{x} \,|\, \mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z}))$ via a network $[\mu, L_\Lambda] = f(\mathbf{z}; \theta)$ and $\|\mathbf{a}\|^2_\Sigma := \mathbf{a}^\top\,\Sigma^{-1}\,\mathbf{a}$ denotes a Gaussian weighted norm

SUPN as a prior for inverse problems

- Consider a hierarchical model for the inverse problem

$$p(\mathbf{x}, \mathbf{z} \,|\, \mathbf{y}) \propto p(\mathbf{y} \,|\, \mathbf{x})\, p_{\mathcal{G}}(\mathbf{x} \,|\, \mathbf{z})\, p_{\mathcal{Z}}(\mathbf{z})$$

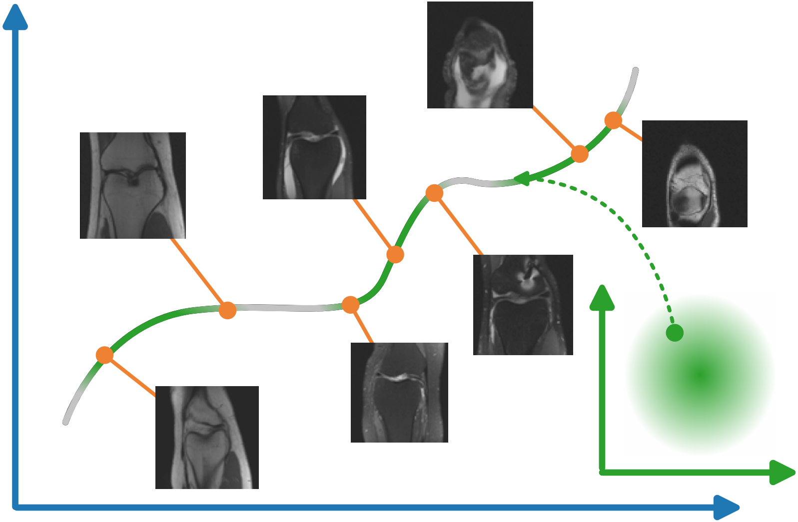
- We will take a MAP estimate for $\mathbf{z}$ rather than marginalising `:-(`
- From before (with a Gaussian observation likelihood) and $p_{\mathcal{Z}}(\mathbf{z}) \sim \mathcal{N}(\mathbf{0}, I)$

$$D(\mathbf{y}, A\,\mathbf{x}) := \frac{1}{2\sigma^2}\|A\,\mathbf{x} - \mathbf{y}\|_2^2$$

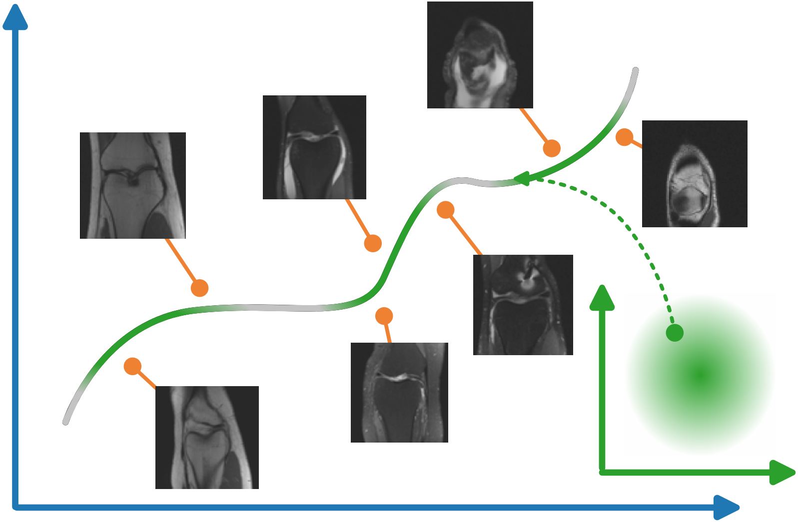
$$R(\mathbf{x}) := \min_{\mathbf{z}\in\mathcal{Z}} \; \log|\Sigma_\theta(\mathbf{z})| + \frac{1}{2}\|\mathbf{x} - \mu_\theta(\mathbf{z})\|^2_{\Sigma_\theta(\mathbf{z})} + \frac{1}{2}\|\mathbf{z}\|_2^2$$

- Where the *Generator* provides $\mathcal{N}(\mathbf{x} \,|\, \mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z}))$ via a network $[\mu, L_\Lambda] = f(\mathbf{z}; \theta)$ and $\|\mathbf{a}\|^2_\Sigma := \mathbf{a}^\top \Sigma^{-1} \mathbf{a}$ denotes a Gaussian weighted norm
- *Note: the network still outputs $\mathcal{O}(N)$ values and evaluation of $R(\mathbf{x})$ can be performed in $\mathcal{O}(N)$ time using L_Λ for the first two terms*

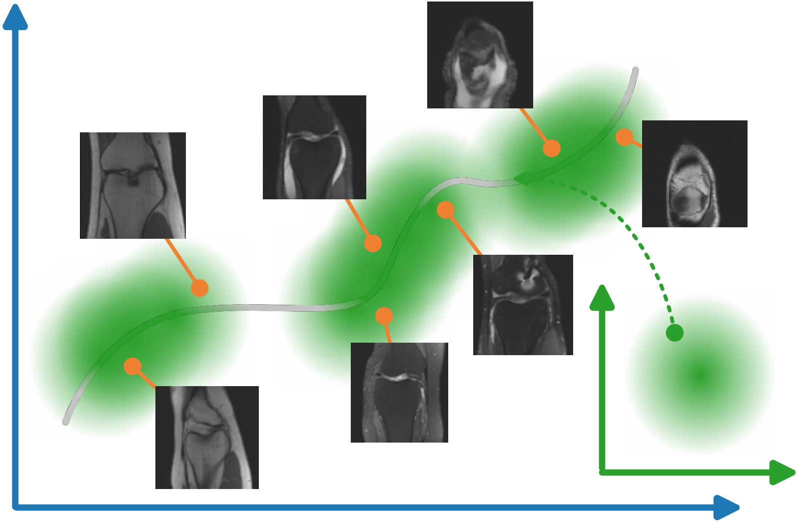
Aside: Images and manifolds



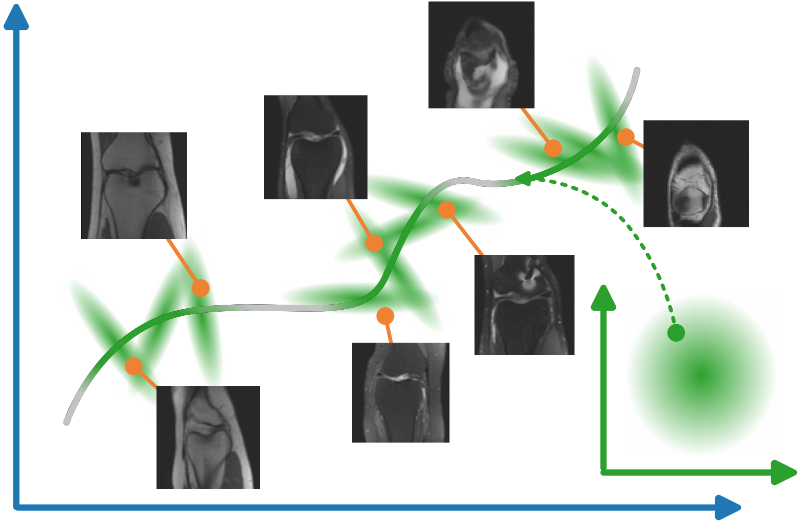
Aside: Images and manifolds



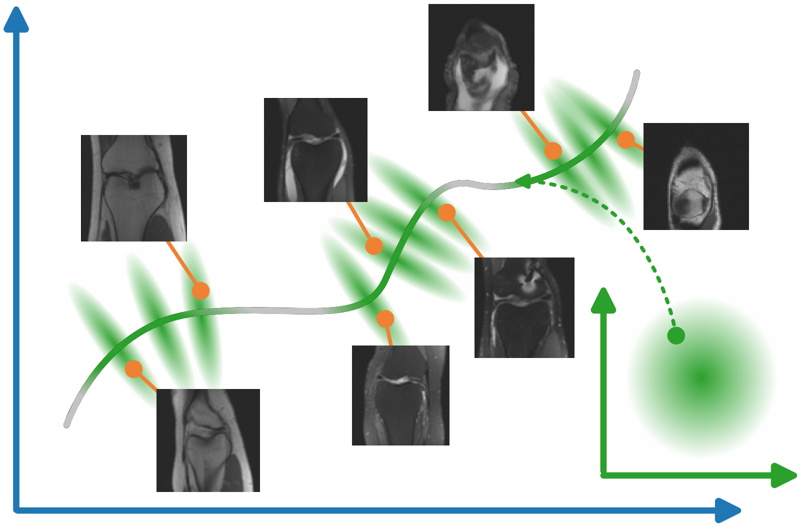
Aside: Images and manifolds



Aside: Images and manifolds



Aside: Images and manifolds



FastMRI knee covariance models...

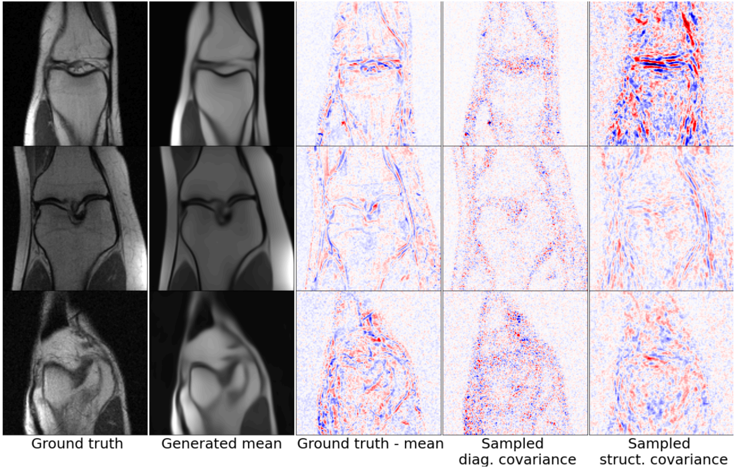


Figure 12: Samples from trained generative models with diagonal and structured covariances

Introspection: Visualisation of learned covariances...

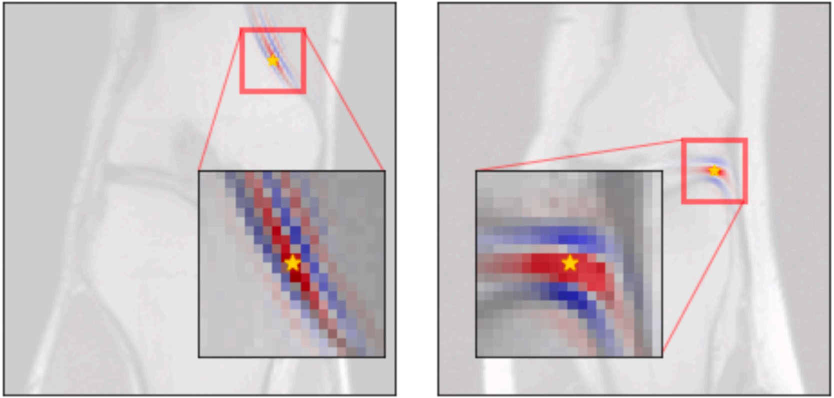


Figure 13: Visualisation of learned covariances; red indicates a high positive correlation, and blue is a strong negative correlation.

Comparison vs supervised reconstruction method

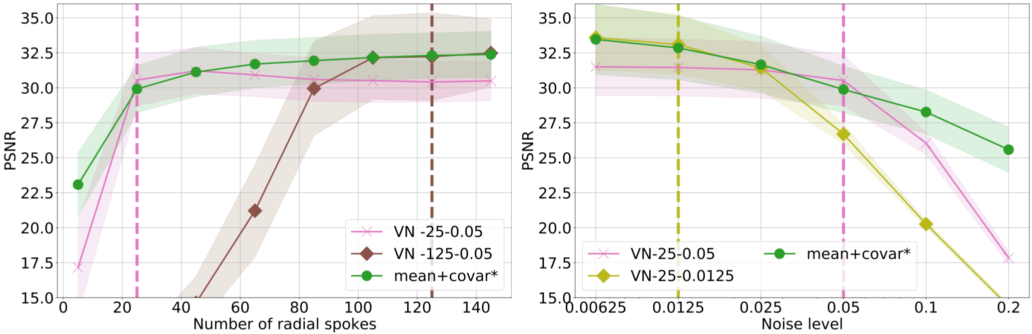


Figure 14: Comparison with the supervised variational networks [Hammernik et al. 2018]. The vertical lines depict the experimental settings the variational networks were trained on.

Example reconstruction comparison (varying number of spokes)

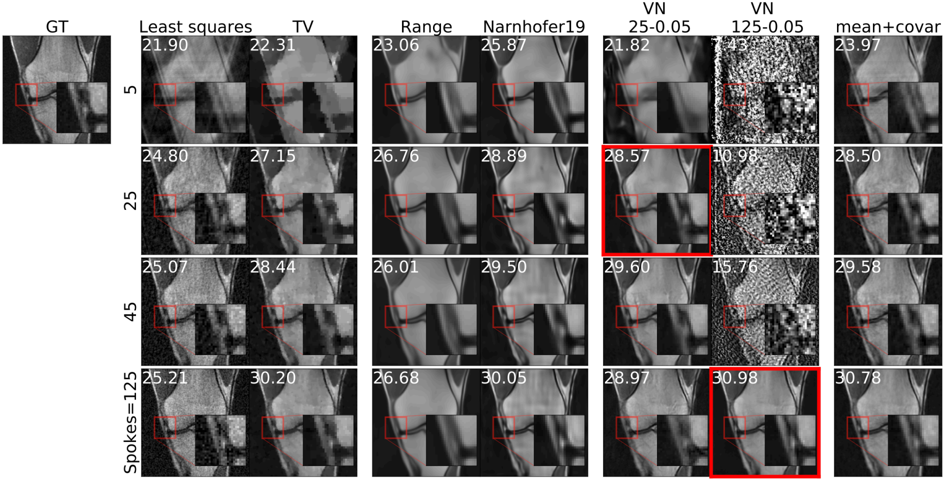
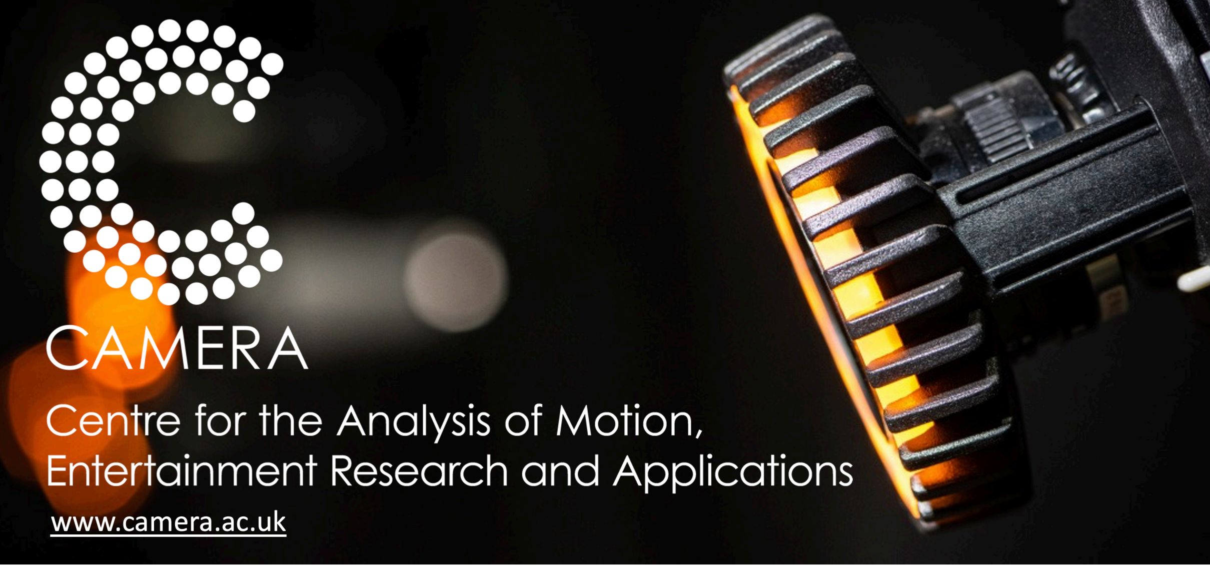


Figure 15: Varying number of spokes. The PSNR values are added in white and the red boxes indicate the settings the highlighted variational network has been trained on.

Conclusions

Conclusions

- Hopefully motivated the need for uncertainty in vision (?)
- These techniques are important!
- Make vision safe, trustworthy and robust for applications
- Rigour in experimental validation
- There is still much work do be done here!

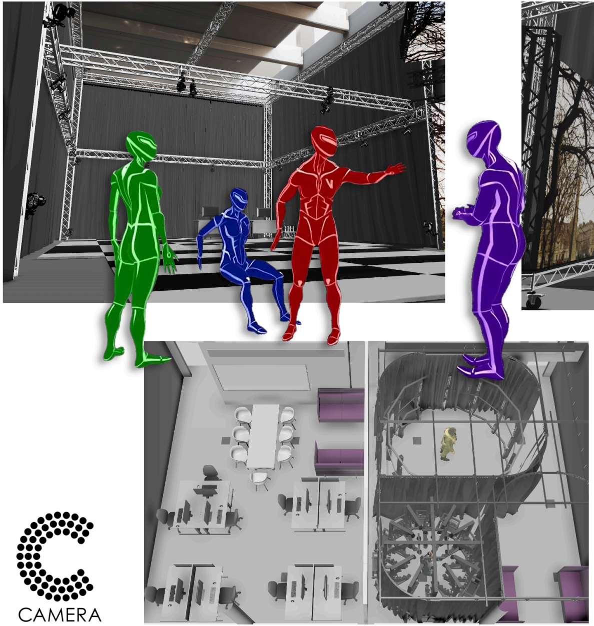


CAMERA Centre for the Analysis of Motion, Entertainment Research and Applications

MyWorld

UNIVERSITY OF BATH

UKRI Engineering and Physical Sciences Research Council



CAMERA

3rd Workshop on Uncertainty in Vision (at ECCV)

https://uncertainty-cv.github.io/2024/

About Call for Papers Accepted Papers Program

In the last decade, substantial progress has been made w.r.t. the performance of computer vision systems, a significant part of it thanks to deep learning. These advancements prompted sharp community growth and a rise in industrial investment. However, most current models lack the ability to reason about the confidence of their predictions; integrating uncertainty quantification into vision systems will help recognize failure scenarios and enable robust applications.

In addition to advances in Bayesian deep learning, providing practical approaches for vision problems, the workshop will provide a forum for discussing promising research directions, which have received less attention, as well as advancing current practices to drive future research. Examples include: the development of new metrics that reflect the real-world need for uncertainty when using vision systems with down-stream tasks; and moving beyond point-estimates to address the multi-modal ambiguities inherent in many vision tasks.

This years UNcertainty quantification for Computer Vision (UNCV) Workshop aims to raise awareness and generate discussion regarding how predictive uncertainty can, and should, be effectively incorporated into models within the vision community. The workshop will bring together experts from machine learning and computer vision to create a new generation of well-calibrated and effective methods that *know when they do not know*.

Important slide acknowledgements!

Illustrations taken from the excellent new text book from Simon Prince:

Understanding Deep Learning, Simon J.D. Prince, MIT Press

Final draft available on the website:

`https://udlbook.github.io/udlbook/`

Further Reading

- **Understanding Deep Learning**, Simon J.D. Prince
    - `https://udlbook.github.io/udlbook/`
- **Information Theory, Inference, and Learning Algorithms**, David MacKay
    - `https://www.inference.org.uk/itprnn/book.html`
- **Pattern Recognition and Machine Learning**, Christopher M. Bishop
    - Microsoft Website with PDF
- **Computer Vision: Models, Learning, and Inference**, Simon J.D. Prince
    - `http://www.computervisionmodels.com/`

That’s all folks..