

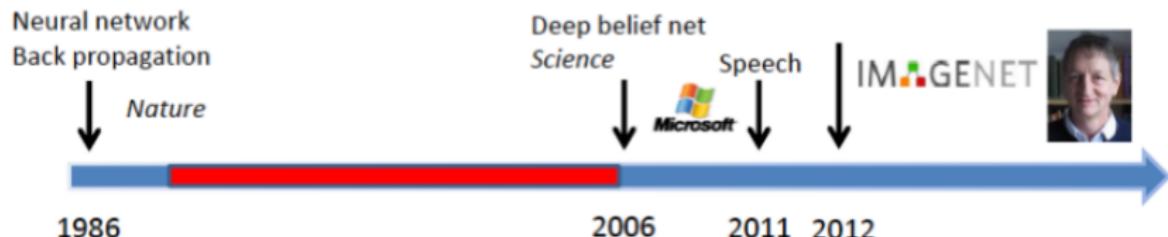
Sorbonne Université, Computer Science Master Données, Apprentissage et Connaissances (DAC) Bayesian Deep Learning

Nicolas Thome
Sorbonne Université
ISIR Lab - Machine Learning Team (MLIA)



Deep Learning Success since 2010

- 90's / 2000's: difficult to train large deep models on existing databases



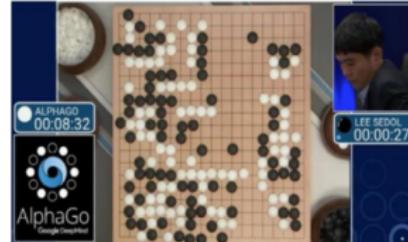
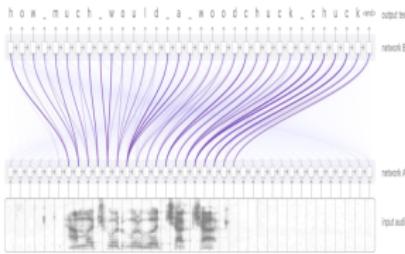
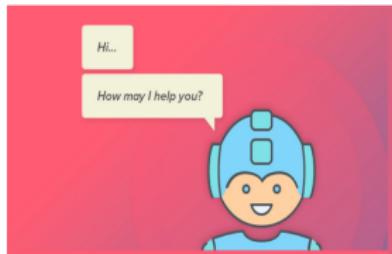
- **ILSVRC'12: the deep revolution**
⇒ outstanding success of ConvNets [Krizhevsky et al., 2012]



Rank	Name	Error rate	Description
1	U. Toronto	0.15315	Deep learning
2	U. Tokyo	0.26172	Hand-crafted features and learning models.
3	U. Oxford	0.26979	
4	Xerox/INRIA	0.27058	Bottleneck.

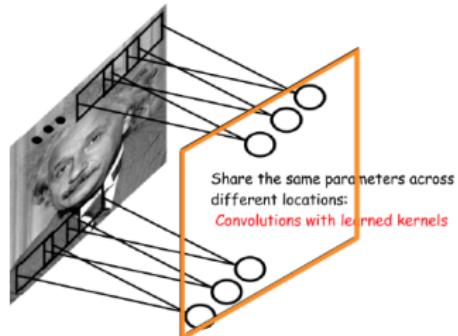
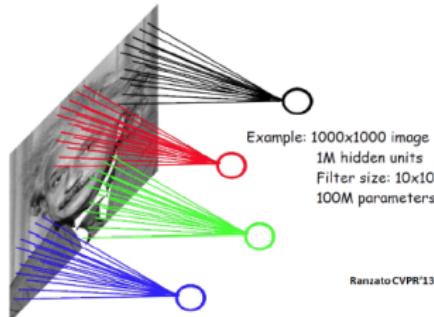
Deep Learning everywhere since 2012

- Image classification, speech recognition
- chatbots, translation,
- Games, robotics

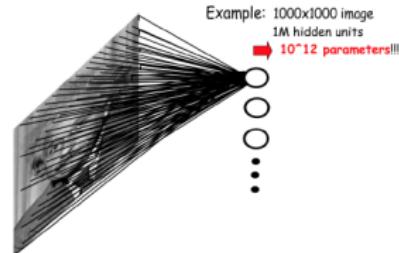
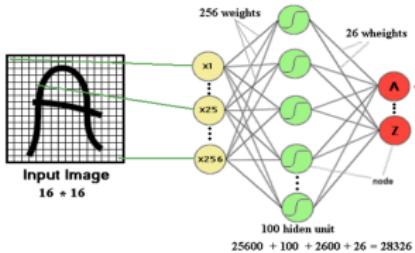


Convolutional Neural Networks (ConvNets)

- ConvNets: sparse connectivity + shared weights



- Local feature extraction (\neq FCN)
- Overcome parameter explosion for FCN on images

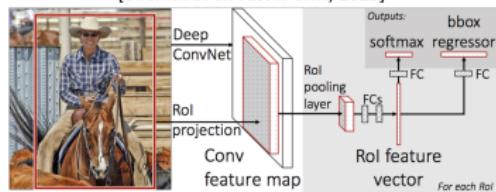


Deep Learning in Computer Vision

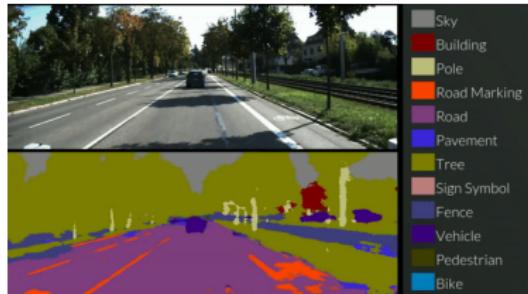
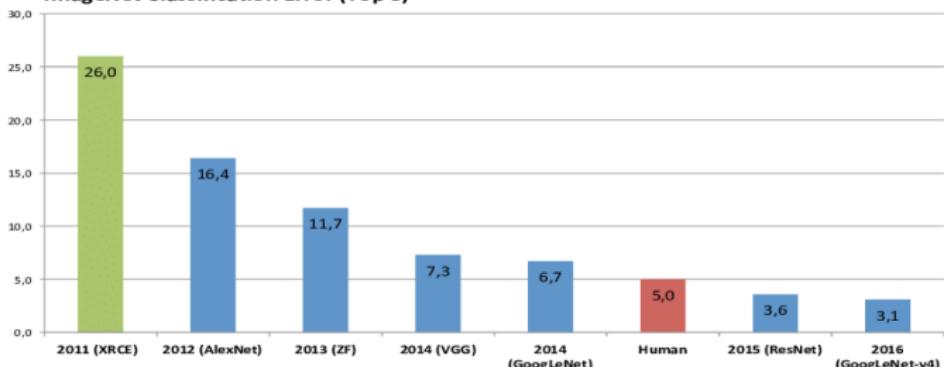
[Krizhevsky, 2012]



[Girshick et al. Fast R-CNN, 2015]



ImageNet Classification Error (Top 5)

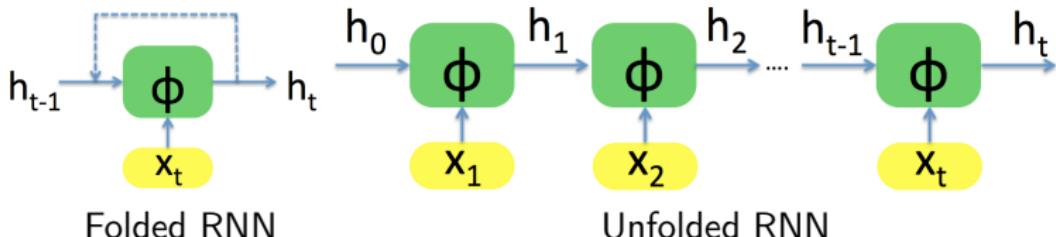


[Kendall et al. SegNet, 2015]

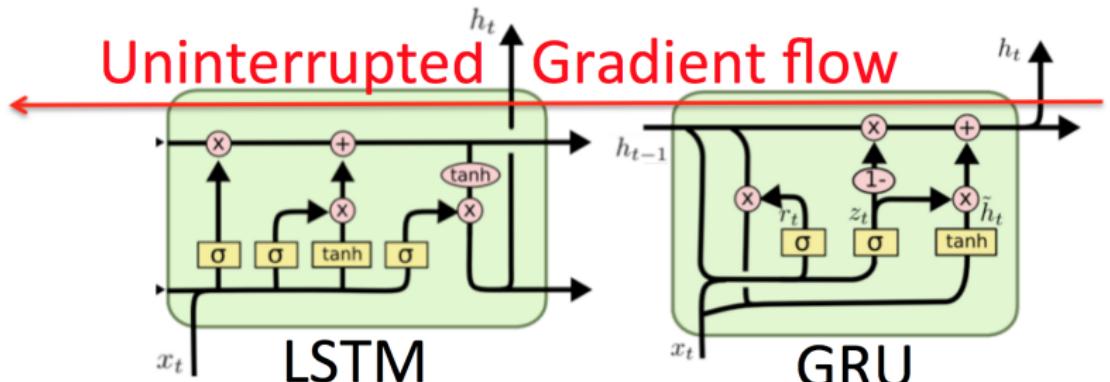
Brought significant improvements in multiple vision tasks

Recurrent Neural Networks (RNNs)

- **RNN Cell:** $h_t = \phi(x_t, h_{t-1}) = f(Ux_t + Wh_{t-1} + b_h)$ [Elman, 1990]
 - ▶ h_t : network memory up to time $t \Rightarrow$ Sequence processing

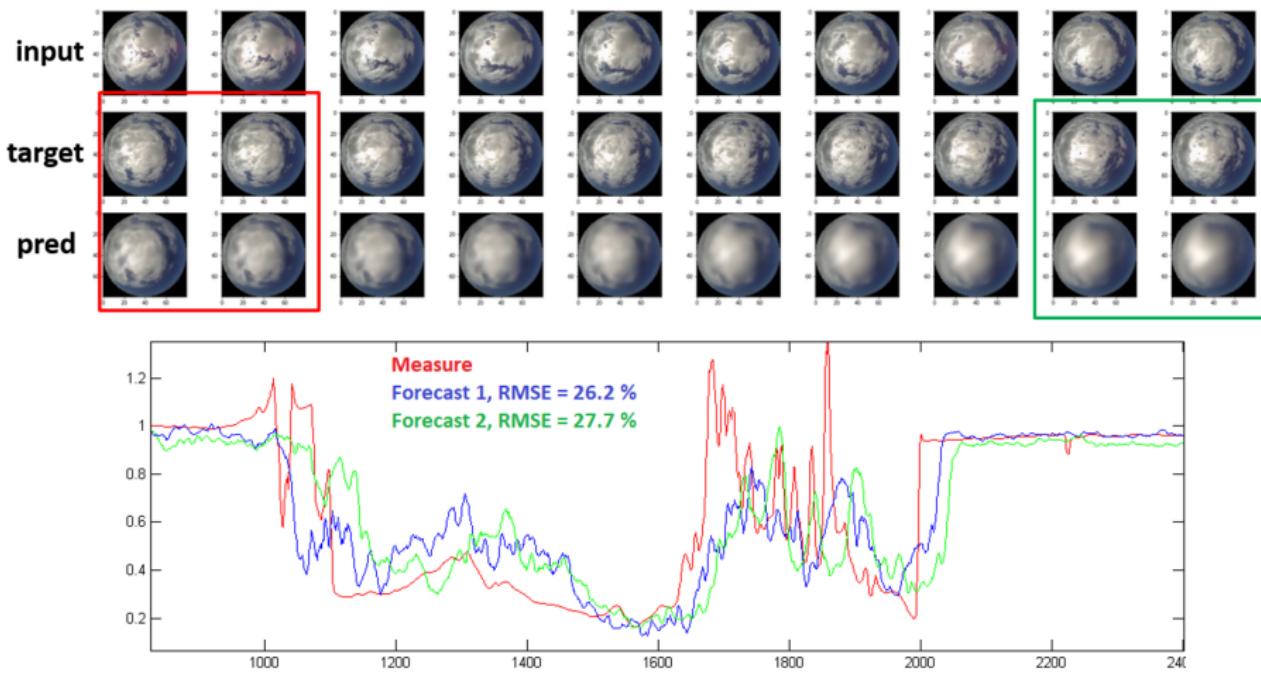


- **Specific architectures for vanishing gradients:**
LSTM [Hochreiter and Schmidhuber, 1997], GRU [Cho et al., 2014]



Deep Learning for Sequence Processing

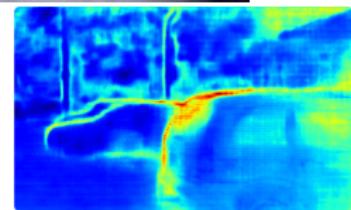
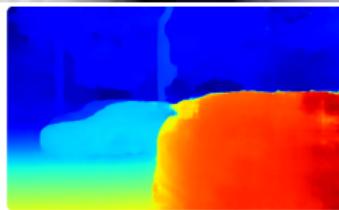
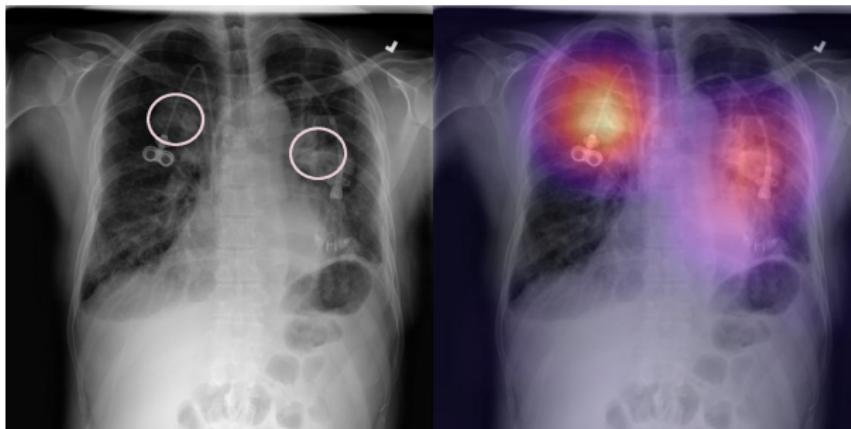
- RNNs SOTA for many sequential decision making tasks: speech, translation, text/music generation, times series, etc
 - Ex: video forecasting, *i.e.* prediction future frames



Robustness in Deep Learning

Deep Learning: huge gain in average performance, e.g. precision for classification

- Need for **performance certification in safety-critical applications: robustness**
 - ▶ Healthcare, autonomous steering, nuclear, defense, etc



Robustness in Deep Learning

Performance certification

- **Formal understanding of deep learning:**

- ▶ Understanding generalization performances with over-parameterized networks
- ▶ Optimization, landscape loss function

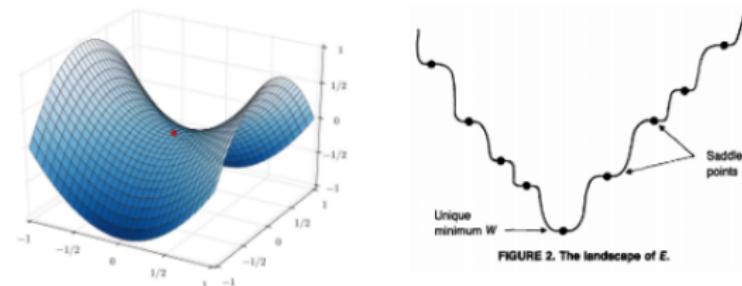
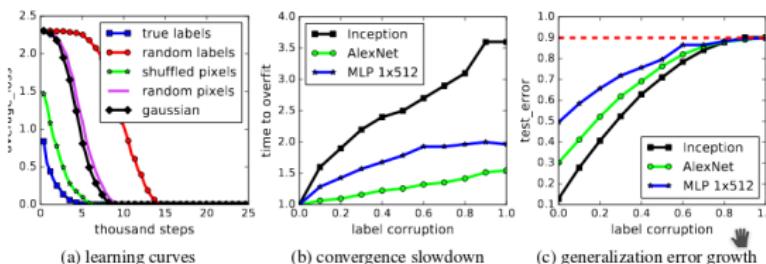


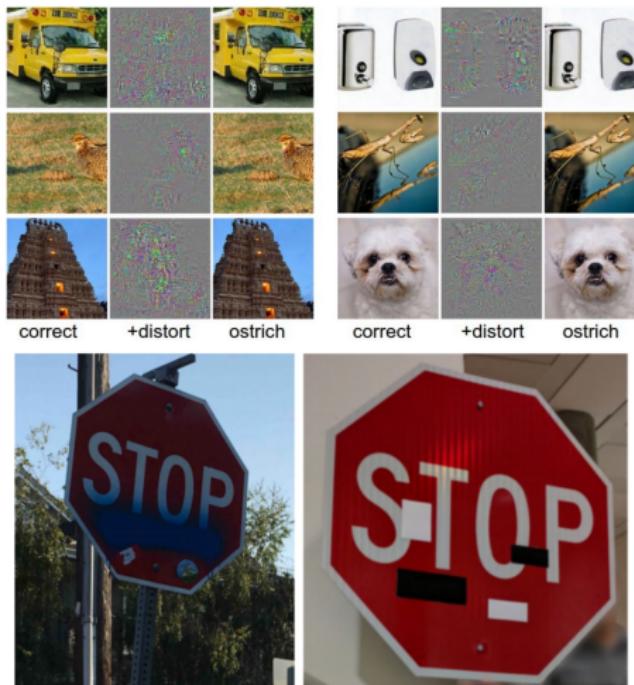
FIGURE 2. The landscape of E .



Robustness in Deep Learning

Performance certification

- Stability of the decision function, e.g. robustness to adversarial attacks

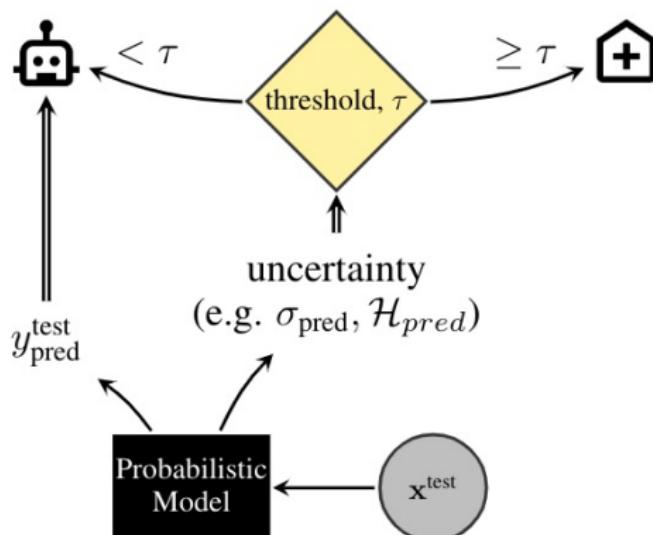


[Evtimov et al., 2017]

Robustness in Deep Learning

Performance certification

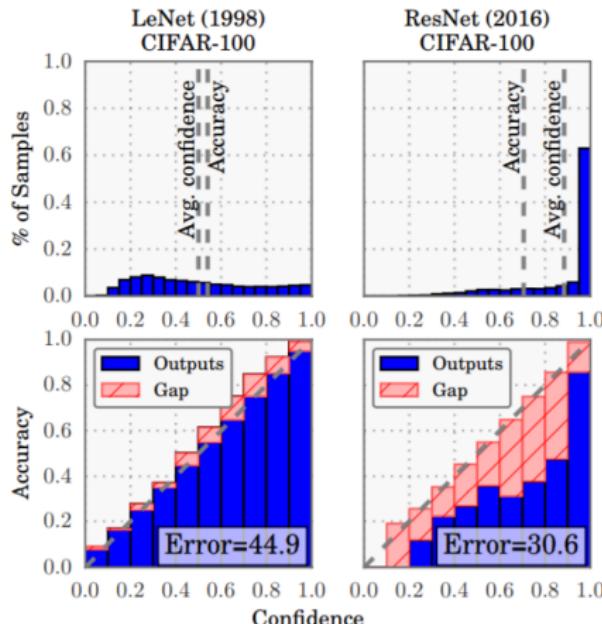
- **Reliable confidence / uncertainty estimation** of the decision process
 - ▶ "Know when you do not know"



Robustness in Deep Learning

Performance certification

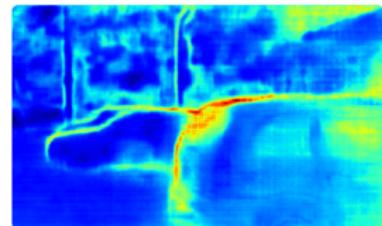
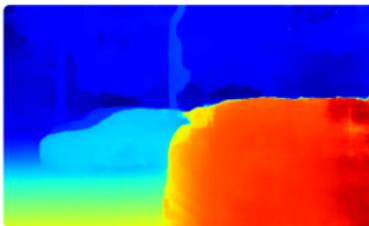
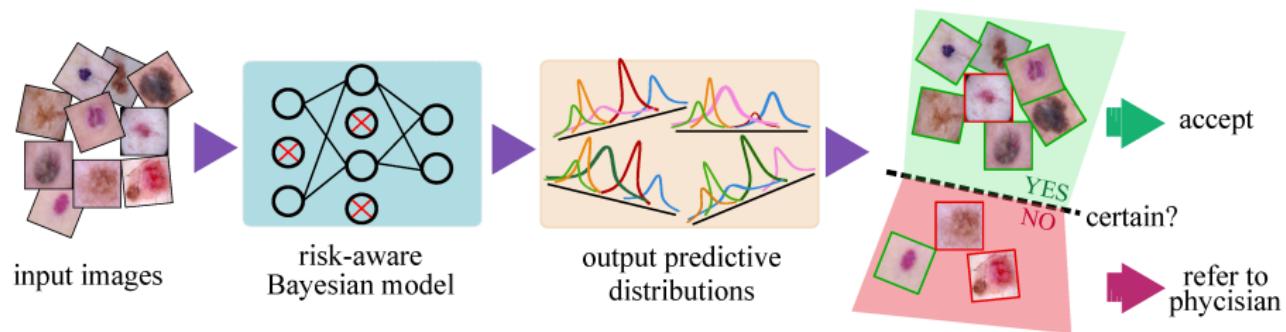
- Reliable confidence / uncertainty estimation of the decision process
 - ▶ Deep learning models overconfident, poor at this task [Guo et al., 2017]



Robustness in Deep Learning

Performance certification

- **Reliable confidence / uncertainty estimation** of the decision process
 - ▶ Crucial for deployment in healthcare / autonomous steering applications



Robustness in Deep Learning

This course: 3 weeks on robust deep learning

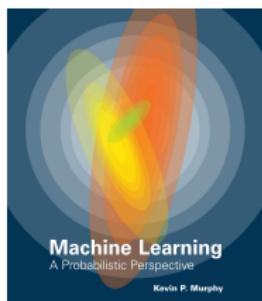
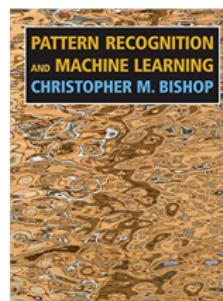
<https://rdfia.github.io/>

1. Bayesian models and uncertainty estimation
 - ▶ Bayesian linear regression and extension to non-linear feature maps
2. Bayesian deep learning
 - ▶ Approximate posterior inference, variational approximation
 - ▶ Monte Carlo dropout
3. Application of uncertainty, other robustness issues
 - ▶ failure prediction and OOD detection

- **References:**

- Pattern Recognition and Machine Learning [Bishop, 2006]
- Machine Learning: A Probabilistic Perspective [Murphy, 2012]

- **Eval:** practical session report (20% of RDFIA)



Outline

Uncertainty

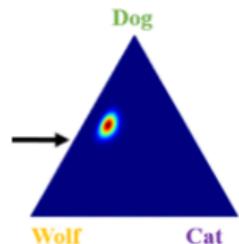
Bayesian Models

Bayesian Linear Regression

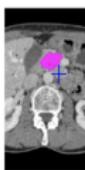
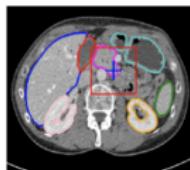
Type of uncertainties

Aleatoric uncertainty

- *Aleator* (lat.) = dice player
- **Noise inherent in the observations**, i.e. natural randomness.
 - ▶ Inherent stochasticity, e.g. class confusion



- ▶ Ambiguous data (e.g. medical images), or sensor quality



Rain drops*



Lack of visual features



Glare

Type of uncertainties

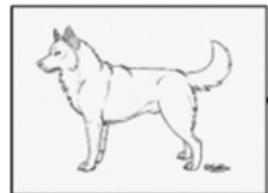
Aleatoric uncertainty

- *aleator* (lat.) = dice player
- **Noise inherent in the observations**, i.e. natural randomness.
- Cannot be reduced (need to change input/sensor), but can be estimated/learned
- Two (sub)-types of aleatoric uncertainty:
 1. **homoscedastic uncertainty**: stays constant for different input values, limited, captures 'average' uncertainty
 2. **heteroscedastic uncertainty**: depends on the input, learned from data

Type of uncertainties

Epistemic uncertainty

- *Episteme* (gr.): knowledge \Rightarrow **model uncertainty**
- Lack of knowledge about the generating process y (class) \rightarrow input (image)



- **Main feature:** detects samples far from the training distribution
- Can be explained away given enough data
 - ▶ Epistemic uncertainty \downarrow when number of data N (evidence) \uparrow
 - ▶ Epistemic uncertainty $\rightarrow 0$ when $N \rightarrow \infty$

Outline

Uncertainty

Bayesian Models

Bayesian Linear Regression

Bayesian Models

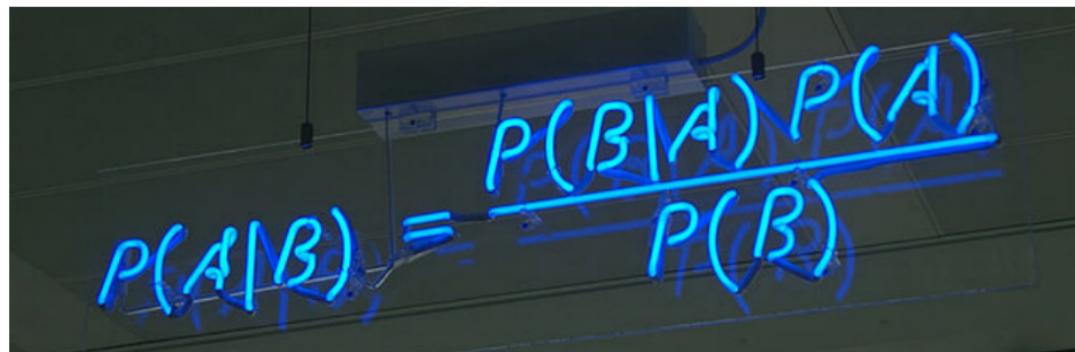
- Observed inputs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$
 - $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}^K$ (classification or regression)
 - Model with parameters \mathbf{w} : $\hat{\mathbf{y}}_i = f_{\mathbf{w}}(\mathbf{x}_i)$
- Bayes rule:** $p(\mathbf{Y}, \mathbf{w}/\mathbf{X}) = p(\mathbf{Y}/\mathbf{X}, \mathbf{w})p(\mathbf{w}) = p(\mathbf{w}/\mathbf{X}, \mathbf{Y})p(\mathbf{Y}/\mathbf{X})$

$$\Rightarrow p(\mathbf{w}/\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}/\mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{Y}/\mathbf{X})} \propto p(\mathbf{Y}/\mathbf{X}, \mathbf{w})p(\mathbf{w})$$

A photograph of a whiteboard with handwritten Bayes' theorem. The formula is written in blue marker: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. The background shows a lecture hall setting with rows of desks and chairs.

Bayesian Models

- **What we model:** $p(Y/X, w)$: likelihood and $p(w)$: prior knowledge
- **What we compute:** $p(w/X, Y) \propto p(Y/X, w)p(w)$ posterior
biases prior $p(w)$ once data observed through likelihood $p(Y/w, X)$
How to estimate optimal w ?
- **Maximum Likelihood (ML):** ignore (or assume uniform) prior
⇒ find \hat{w} s.t $p(Y/X, w)$ max
- **Maximum A Posteriori (MAP):** use prior $p(w)$
⇒ find \hat{w} s.t $p(w/X, Y)$ max


$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

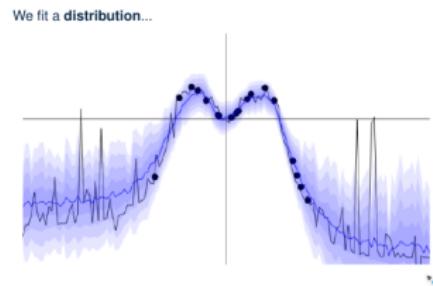
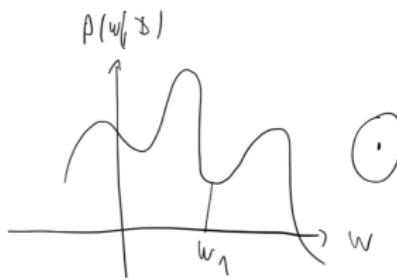
Bayesian Models & Uncertainty

- From posterior $p(w/X, Y) \Rightarrow$ compute **predictive distribution** given new input x^* ($\forall y$): $p(y/x^*, Y, X) = \int p(y, w/x^*, Y, X) dw$

$$p(y/x^*, Y, X) = \int p(y/x^*, w)p(w/X, Y) dw = \mathbb{E}_{p(w|D)}[p(y|x^*, w)]$$

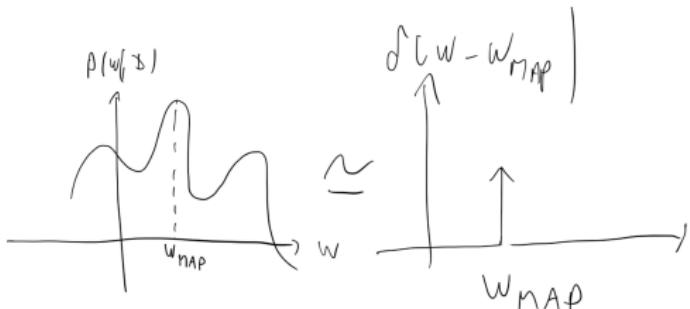
- ▶ Naturally gives a measure of uncertainty

$$p(y, w | x^*, D) \propto p(y | x^*, w) p(w | D)$$

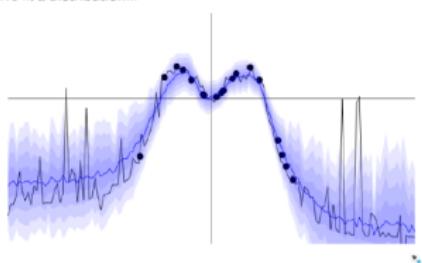


Bayesian vs deterministic models

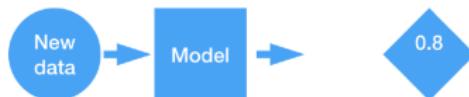
- Deterministic models:
 $\mathbf{w}_{\text{MAP}} = \arg \min_{\mathbf{w}} p(\mathbf{w} | \mathcal{D})$
- Only using \mathbf{w}_{MAP} for the posterior:
 $p(\mathbf{w} | \mathbf{X}, \mathbf{Y}) \approx \delta(\mathbf{w} - \mathbf{w}_{\text{MAP}})$
 $\Rightarrow p(y|\mathbf{x}^*, \mathcal{D}) \approx p(y|\mathbf{x}, \mathbf{w}_{\text{MAP}})$



We fit a distribution...



Deep NN



Point estimate

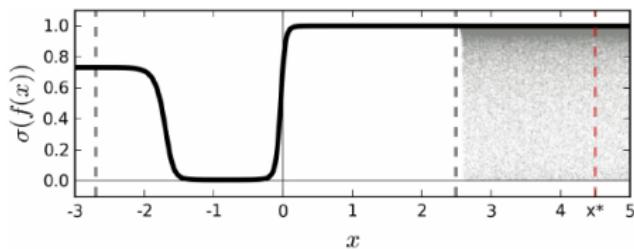
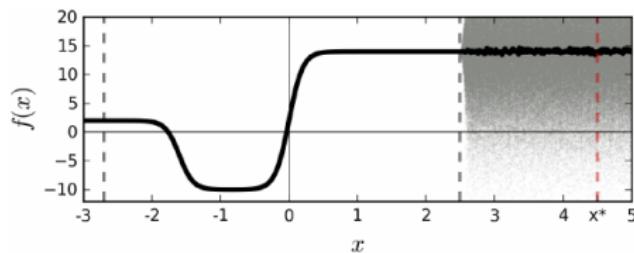
Bayesian NN



Estimate
Distribution

Point-wise vs distribution modeling

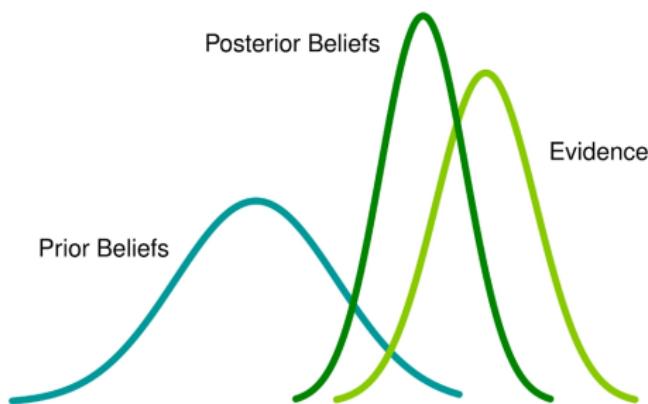
In binary classification, training a neural network to learn a function $f^w(x)$ (e.g. tanh) from a 1D dataset $X=[-3,2]$, and apply Softmax to obtain probabilities.



- Point estimate through Softmax
⇒ unjustified high confidence far from data: only model aleatoric uncertainty
- Distribution $p(y/x^*, Y, X)$ ⇒ model epistemic uncertainty, should increase far from training data

Bayesian Models & Uncertainty

- RECAP: for uncertainty estimate with Bayesian models
 1. Define prior $p(w)$ and likelihood $p(Y/X, w)$
 2. Compute posterior distribution $p(w/X, Y)$
 3. Compute predictive distribution $p(y/x^*, Y, X)$
- Easy? NO !! \Rightarrow steps 2 and 3 computationally hard in general!
 - ▶ Typically no closed form for step 2
 - ▶ High-dimensional integration for step 3



Outline

Uncertainty

Bayesian Models

Bayesian Linear Regression

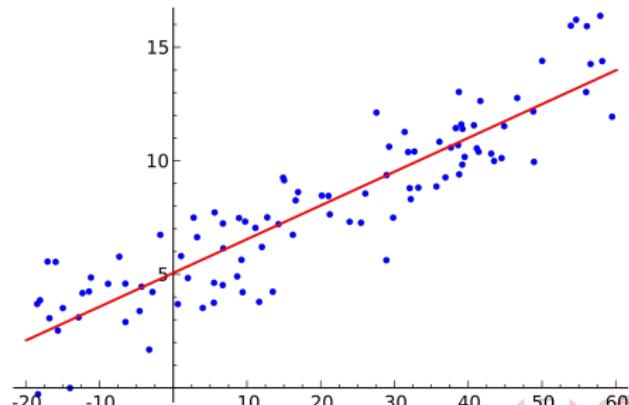
Probabilistic Linear Regression

- N training examples $(x_i, y_i)_{i \in \{1;N\}}$; $x_i \in \mathbb{R}^p$, $y_i \in \mathbb{R}^K$
- Matrix notation including bias in w: Φ of size $N \times (p + 1)$

$$\Phi = \begin{pmatrix} 1 & x_{11} & \dots & x_{1p} \\ 1 & x_{21} & \dots & x_{2p} \\ \dots & \dots & \dots & \dots \\ 1 & x_{N1} & \dots & x_{Np} \end{pmatrix}$$

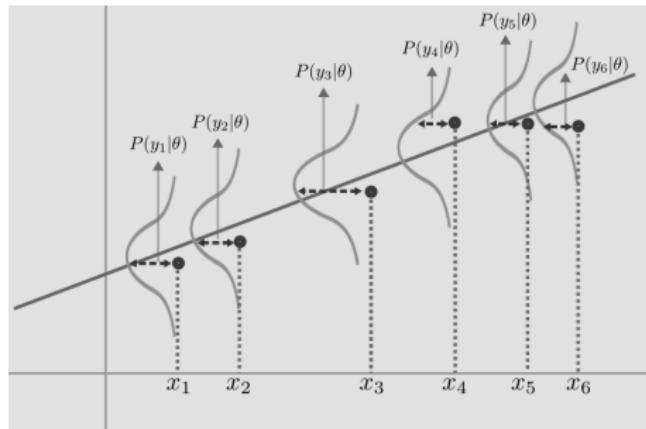
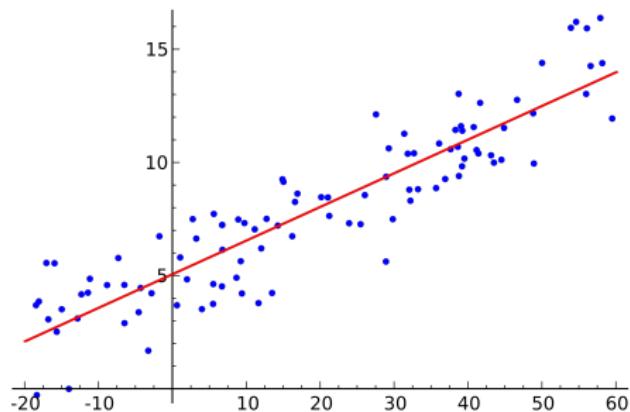
- $Y = \Phi w + \varepsilon$, $\varepsilon \in \mathbb{R}^{N \times K}$, $\varepsilon_{i,k} \sim \mathcal{N}(0, \sigma^2)$
- $Y \in \mathbb{R}^{N \times K}$, $\Phi \in \mathbb{R}^{N \times (p+1)}$, $w \in \mathbb{R}^{(p+1) \times K}$
 - $\Phi_i^T := (1 \quad x_{i1} \quad \dots \quad x_{ip})^T \in \mathbb{R}^{1 \times (p+1)}$
 - $y_i = \Phi_i^T w + \varepsilon_i$, $y_i \in \mathbb{R}^{1 \times K}$, $\varepsilon_i \in \mathbb{R}^{1 \times K}$

- Ex: scalar inputs and outputs $p = 1$, $K = 1 \Rightarrow \Phi \in \mathbb{R}^{N \times 2}$, $w \in \mathbb{R}^2$:



Probabilistic Linear Regression

- N training examples (x_i, y_i) $y_i = \Phi_i^T w + \varepsilon_i$, with $\varepsilon_i \sim \mathcal{N}(0, 1)$
- $p(y_i/x_i, w) \sim \mathcal{N}(\Phi_i^T w, \sigma^2)$ or $p(y_i - \Phi_i^T w) \sim \mathcal{N}(0, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(y_i - \Phi_i^T w)^2}{2\sigma^2}}$
 - ▶ $p(y_i/x_i, w)$: likelihood, $\sigma \sim$ aleatoric uncertainty
 - ▶ σ independent of $x \Rightarrow$ homoscedastic uncertainty



Probabilistic Linear Regression

- Trained with Maximum Likelihood (ML)

- Examples assumed to be *i.i.d* (independent and identically distributed)

$$\Rightarrow p(Y/X, w, \sigma) = \prod_{i=1}^N p(y_i/x_i, w, \sigma)$$

- MLE: $(\hat{w}, \hat{\sigma}) = \arg \max_{(w, \sigma)} p(X, Y/w, \sigma) = \arg \min_{w, \sigma} - \sum_{i=1}^N \log [p(y_i/x_i, w)]$

- MLE solution w : $\hat{w} = \arg \min_w C(w) = \arg \min_w \sum_{i=1}^N (y_i - \Phi_i^T w)^2$

\Rightarrow Ordinary least square problem (closed form):

- $C(w) = \|Y - \Phi w\|^2 = (Y - \Phi w)^T (Y - \Phi w), \nabla_w C = 2\Phi^T (Y - \Phi w)$
 - $\nabla_w C = 0 \Leftrightarrow \Phi^T \Phi w = \Phi^T Y$

$$\boxed{\hat{w} = (\Phi^T \Phi)^{-1} \Phi^T Y}$$

- ML solution σ : $\arg \min_{\sigma} [N \log(\sigma) + \frac{1}{2\sigma^2} \sum_{i=1}^N (y_i - \Phi_i^T w)^2]$

- Closed form solution: $\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^N (y_i - \Phi_i^T w)^2 \Rightarrow$ interpretation: data std

Limits of MLE

- Learning model to predict coin toss with MLE

▶ Bernoulli variable X with param p : $P(x|p) = \prod_{i=1}^N P(x_i|p) = \prod_{i=1}^N p^{x_i} (1-p)^{1-x_i}$

▶ MLE: $\ln P(x|p) = \sum_{i=1}^N [x_i \ln p + (1 - x_i) \ln(1 - p)] \Rightarrow p_{MLE} = \frac{1}{N} \sum_{i=1}^N x_i$

▶ MLE: predict $P(X|p_{MLE}) = 1$ for all futures tosses!



- Using prior knowledge on p , e.g. $P(p) = 0.5$ or $P(p) = 0.3 \Rightarrow$ MAP

Bayesian Linear Regression

Include prior $p(w) \Rightarrow$ biased posterior $p(w/X, Y)$ (MAP)

- Choose **Prior**, e.g. $p(w|\alpha) = \mathcal{N}(w; 0, \alpha^{-1}I)$
- **Likelihood**, as before: $p(y_i/x_i, w) = \mathcal{N}(\Phi_i^T w, \beta^{-1})$ with $\beta = \frac{1}{2\sigma^2}$
⇒ **Compute posterior** $p(w/x_i, y_i) \propto p(y_i/x_i, w)p(w)$
 - ▶ Here, prior same form as likelihood (Gaussian), i.e. **Prior conjugate to likelihood**
⇒ **Posterior as the same form as the prior**
⇒ **Closed-form for the posterior, which is also Gaussian!**
 - ▶ With $p(w|\alpha) = \mathcal{N}(w; 0, \alpha^{-1}I)$ and $p(y_i/x_i, w) = \mathcal{N}(\Phi_i^T w, \beta^{-1})$, we can show that:

$$p(w/X, Y) = \mathcal{N}(w|\mu, \Sigma)$$

$$\Sigma^{-1} = \alpha I + \beta \Phi^T \Phi$$

$$\mu = \beta \Sigma \Phi^T Y$$

- Closed form solution for MAP (μ , median ↔ mode)
- $\alpha \rightarrow 0 \Rightarrow$ recover ML ; $N = 0 \Rightarrow$ recover prior

Bayesian Linear Regression

Come from general results [Bishop, 2006]

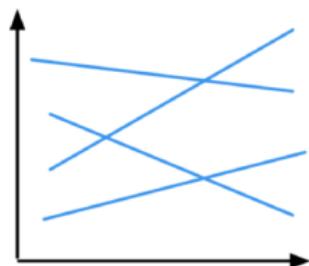
- Assume that:
 - ▶ $p(x) = \mathcal{N}(x|\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$
 - ▶ $p(y|x) = \mathcal{N}(y|Ax + b, \boldsymbol{\Sigma}_y)$
- Then: $p(x|y) = \mathcal{N}(x|\boldsymbol{\mu}_{x|y}, \boldsymbol{\Sigma}_{x|y})$, with:

$$\boldsymbol{\Sigma}_{x|y}^{-1} = \boldsymbol{\Sigma}_x^{-1} + \mathbf{A}^T \boldsymbol{\Sigma}_y^{-1} \mathbf{A}$$

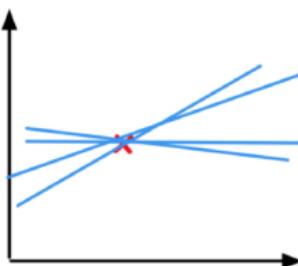
$$\boldsymbol{\mu}_{x|y} = \boldsymbol{\Sigma}_{x|y} [\mathbf{A}^T \boldsymbol{\Sigma}_y^{-1} (y - b) + \boldsymbol{\Sigma}_x^{-1} \boldsymbol{\mu}_x]$$

Bayesian Linear Regression: Posterior Sampling

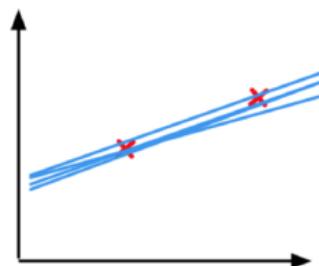
- $p(w|X, Y) = \mathcal{N}(w; \mu, \Sigma) \Rightarrow$ we can sample from posterior
 - ▶ No observation: $p(w|X, Y) = p(w)$
 - ▶ $N \geq 1$: prior biased by data likelihood
 - ▶ Ex: $p(w|x_0, y_0) \propto p(w)p(y_0|x_0, w)$
 - ▶ $p(w|(x_0, y_0), (x_1, y_1)) \propto p(w|x_0, y_0)p(y_1|x_1, w) \propto p(w)p(y_0|x_0, w)p(y_1|x_1, w)$
 -



no observations

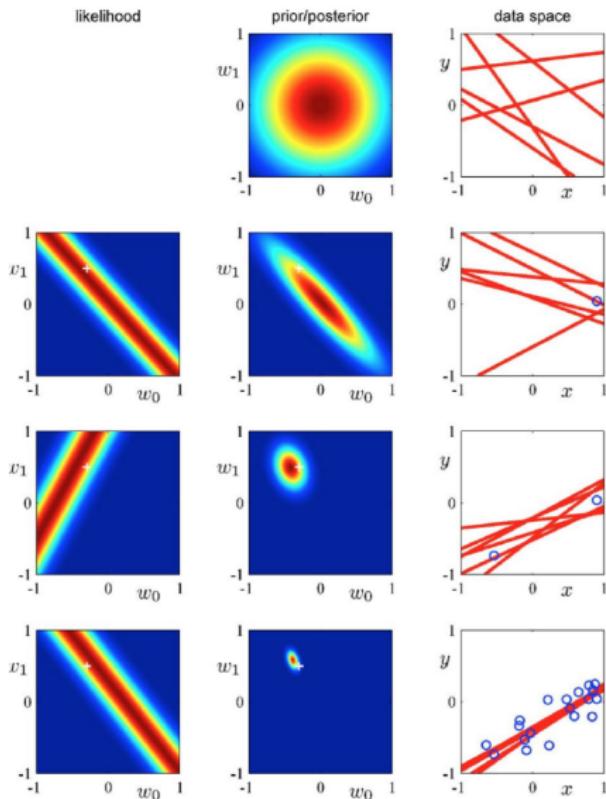


one observation



two observations

Bayesian Linear Regression: Posterior Sampling



0 data points are observed.

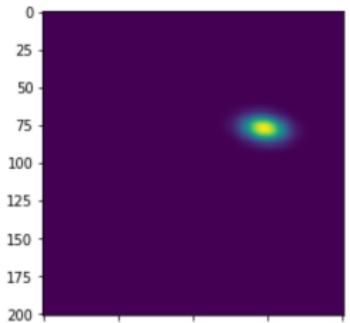
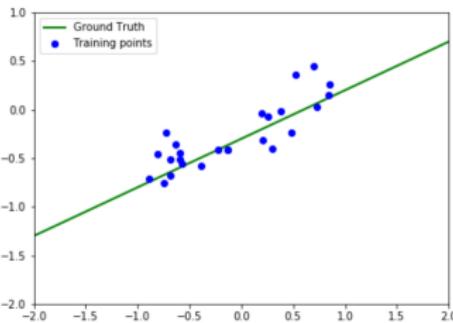
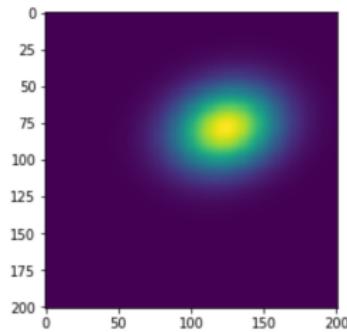
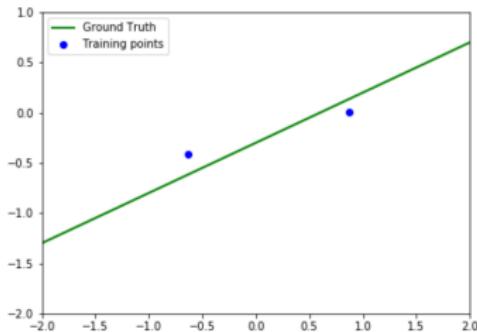
1 data point is observed.

2 data points are observed.

20 data points are observed.

Bayesian Linear Regression: Posterior Sampling

Practical session: compute posterior



- More data points: reducing posterior (epistemic) uncertainty
- $N \rightarrow \infty$ posterior uncertainty \Leftrightarrow aleatoric uncertainty

Bayesian Linear Regression: Predictive Distribution

$p(w|\mathcal{D}, \alpha, \beta) \Rightarrow$ compute predictive distribution by marginalizing over w:

- $p(y|x^*, \mathcal{D}, \alpha, \beta) = \int p(y|x^*, w, \beta)p(w|\mathcal{D}, \alpha, \beta)dw$
 - ▶ $p(y|x^*, w, \beta) = \mathcal{N}(y; \Phi(x^*)^T w, \beta^{-1})$: likelihood
 - ▶ $p(w|\mathcal{D}, \alpha, \beta) = \mathcal{N}(w; \mu, \Sigma)$: w posterior
 - ▶ $\Sigma^{-1} = \alpha I + \beta \Phi^T \Phi$
 - ▶ $\mu = \beta \Sigma \Phi^T Y$
- $p(y|x^*, \mathcal{D}, \alpha, \beta)$: convolution of two Gaussians \Rightarrow Gaussian
 - ▶ Mean of predictive distribution $\mu^T \Phi(x^*)$
 - ▶ Variance of predictive distribution $\sigma_{pred}^2(x^*) = \frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*)$

$$p(y|x^*, \mathcal{D}, \alpha, \beta) = \mathcal{N}(y; \mu^T \Phi(x^*), \frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*))$$

Bayesian Linear Regression: Predictive Distribution

$$p(y|x^*, \mathcal{D}, \alpha, \beta) = \mathcal{N}(y; \mu^T \Phi(x^*), \frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*))$$

- $\sigma_{pred}^2(x^*) = \frac{1}{\beta} + \Phi(x^*)^T \Sigma \Phi(x^*)$
- β is actually our noise representation (**aleatoric**)
- $\Phi(x^*)^T \Sigma \Phi(x^*)$ is uncertainty over parameters w (**epistemic**)
- $\sigma_{pred}^2(x^*)$ actually depends on N , $\sigma_{pred}^2(x^*, N)$
 - ▶ $\sigma_{pred}^2(x^*, N+1) < \sigma_{pred}^2(x^*, N)$
 - ▶ $\lim_{N \rightarrow \infty} \Phi(x^*)^T \Sigma \Phi(x^*) = 0$: epistemic uncertainty removed by adding data samples

Bayesian Linear Regression: Predictive Distribution

$$p(y|x^*, \mathcal{D}, \alpha, \beta) = \mathcal{N}(y; \mu^T \Phi(x^*), \sigma_{pred}^2(x^*))$$

- $\sigma_{pred}^2(x^*) = \beta^{-1} + \Phi(x^*)^T \Sigma \Phi(x^*)$
- $\Phi(x^*)^T \Sigma \Phi(x^*)$ is uncertainty over parameters w (**epistemic**)
- **Practical session:** in case of 1D inputs & outputs, i.e. $x_i \in \mathbb{R}$, $X \in \mathbb{R}^{N \times 1}$, $X \in \mathbb{R}^{N \times 1}$, $w \in \mathbb{R}^2$, $\Phi \in \mathbb{R}^{N \times 2}$:

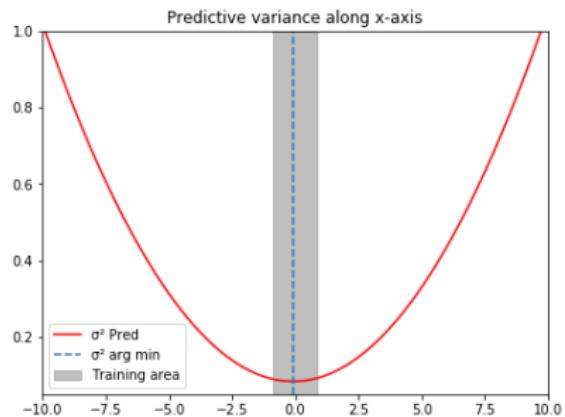
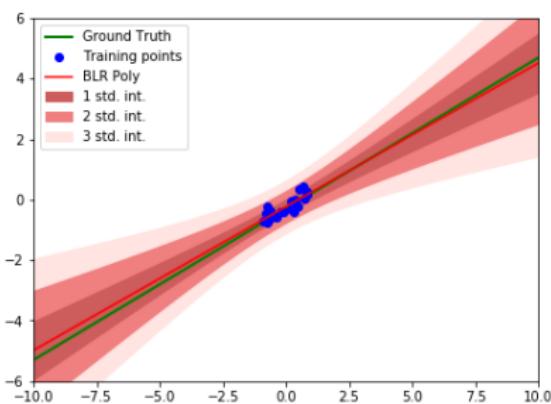
$$\Sigma^{-1} = \alpha I + \beta \Phi^T \Phi = \begin{pmatrix} \alpha + \beta N & \beta \mathbf{1}^T X \\ \beta \mathbf{1}^T X & \alpha + \beta X^T X \end{pmatrix}$$

$\Rightarrow \Phi(x^*)^T \Sigma \Phi(x^*)$ Increases when x^* far from training data

► $x_{min}^* = \frac{\sum_i x_i}{N+\alpha/\beta}$

Bayesian Linear Regression: Predictive Distribution

Practical session: predictive distribution and uncertainty



Bayesian Linear Regression

- Note on MAP estimate:

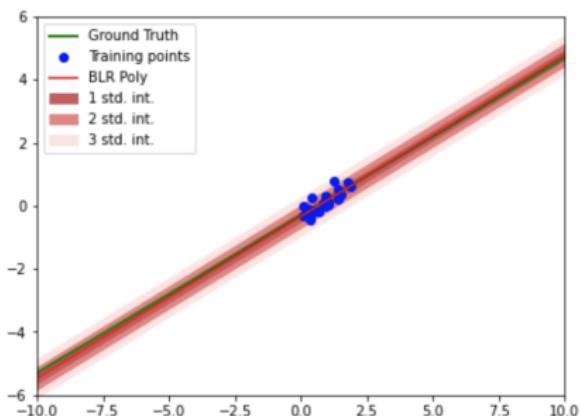
$$\begin{aligned} w_{MAP} &= \arg \min_w - \sum_{n=1}^N \log p(w|x_n, y_n, \beta, \alpha) \\ &= \arg \min_w - \sum_{n=1}^N \log p(y_n|x_n, w, \beta, \alpha) - \log p(w|\alpha) \\ &= \arg \min_w \frac{\beta}{2} \sum_{i=1}^N \|y_n - f^w(x_n)\|^2 + \frac{\alpha}{2} w^T w \end{aligned}$$

⇒ adding a Gaussian prior with precision on weights α acts like L_2 regularisation (weight decay) with $\lambda = \alpha/\beta$

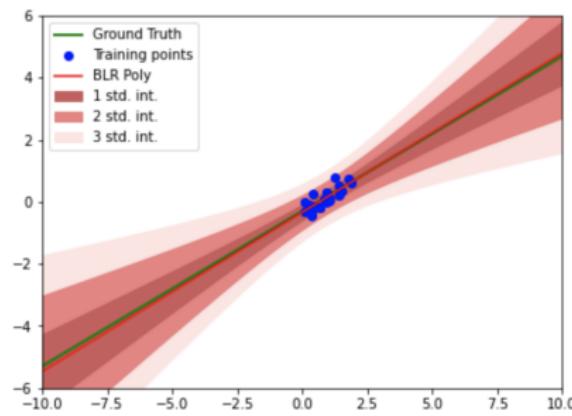
Bayesian Linear Regression: Predictive Distribution

Prediction distribution \neq likelihood at $w = w_{MAP}$

- **Likelihood at $w = w_{MAP}$:** $p(y|x^*, w_{MAP}) = \mathcal{N}(y; \mu^T \Phi(x^*), \sigma^2)$
 - ▶ $\sigma^2 = Cte \quad \forall x^*$
- **Prediction distribution:** $p(y|x^*, \mathcal{D}, \alpha, \beta) = \int p(y|x^*, w, \beta)p(w|\mathcal{D}, \alpha, \beta)dw = \mathcal{N}(y; \mu^T \Phi(x^*), \sigma_{pred}^2(x^*))$
 - ▶ $\sigma_{pred}^2 = f(x^*) = \beta^{-1} + \Phi(x^*)^T \Sigma \Phi(x^*)$



$$p(y|x^*, w_{MAP})$$

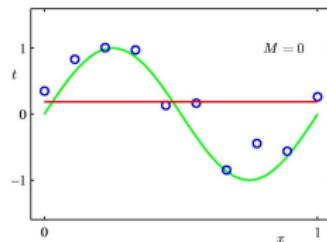


$$p(y|x^*, \mathcal{D}, \alpha, \beta)$$

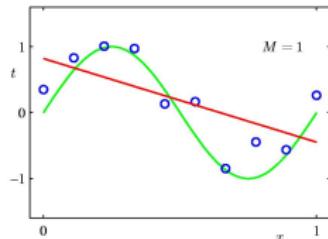
Non-Linear Regression

- Linear regression: limited in many datasets
- Non-linear extension by designing explicit non-linear feature maps Φ
 - ▶ Ex: Polynomial regression for 1D input, i.e. $x_i \in \mathbb{R}$:

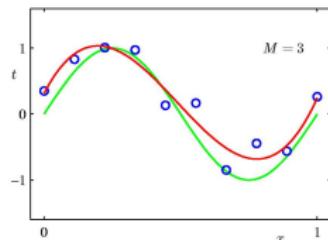
$$\Phi = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^M \\ 1 & x_2 & x_2^2 & \dots & x_2^M \\ \dots & \dots & \dots & & \dots \\ 1 & x_N & x_N^2 & \dots & x_N^M \end{pmatrix}$$



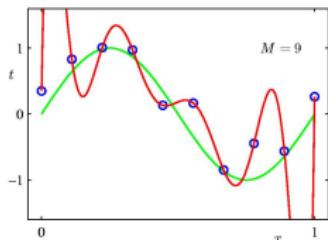
$$Y = \Phi w + \varepsilon \quad \varepsilon = (\varepsilon_1 \quad \dots \quad \varepsilon_N)^T, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$



Poor representations of $\sin(2\pi x)$



Best Fit to $\sin(2\pi x)$



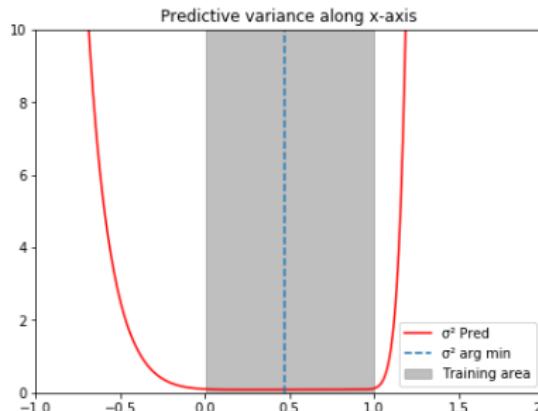
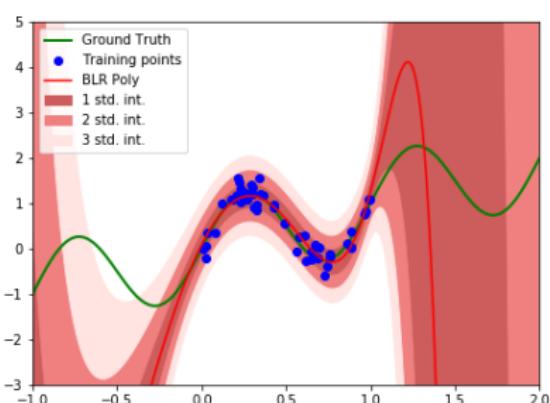
Over Fit
Poor representation of $\sin(2\pi x)$

Bayesian Polynomial Regression

- Polynomial regression for 1D input, i.e. $x_i \in \mathbb{R}$:

$$\Phi = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^M \\ 1 & x_2 & x_2^2 & \dots & x_2^M \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_N & x_N^2 & \dots & x_N^M \end{pmatrix} \quad Y = \Phi w + \varepsilon \quad \varepsilon = (\varepsilon_1 \quad \dots \quad \varepsilon_N)^T, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

- Apply Bayesian linear regression in non-linear feature space Φ
 - ▶ Same closed-form solution for posterior and predictive distribution in Φ
- Practical session: regression for $f(x) = x + \sin(2\pi x)$, $M = 10$, $\alpha = 0.05$

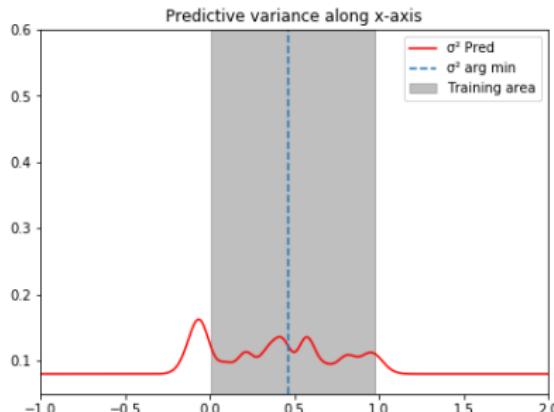
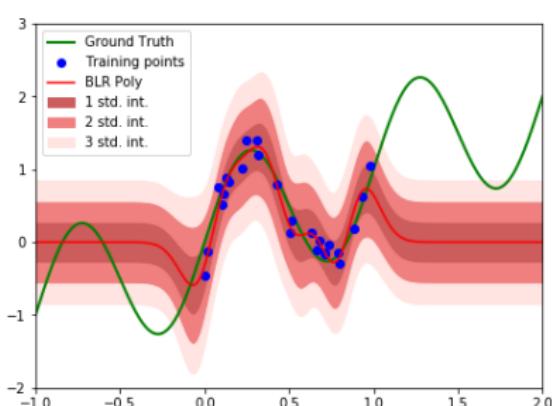


RBF Gaussian Regression

- Gaussian regression for 1D input, i.e. $x_i \in \mathbb{R}$, $\Phi_j(x_i) = \exp\left(-\frac{(x_i - \mu_j)^2}{2s^2}\right)$:

$$\Phi = \begin{pmatrix} \Phi_1(x_1) & \Phi_2(x_1) & \dots & \Phi_M(x_1) \\ \dots & \dots & \dots & \dots \\ \Phi_1(x_N) & \Phi_2(x_N) & \dots & \Phi_M(x_N) \end{pmatrix} \quad Y = \Phi w + \varepsilon, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

- Apply Bayesian linear regression in non-linear feature space Φ



- Practical session: issue with localized features, epistemic uncertainty
 $\Phi(x^*)^T \Sigma \Phi(x^*) \rightarrow 0$ far from training data (μ_j)

References |

- [Bishop, 2006] Bishop, C. M. (2006).
Pattern Recognition and Machine Learning (Information Science and Statistics).
Springer-Verlag, Berlin, Heidelberg.
- [Cho et al., 2014] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014).
Learning phrase representations using rnn encoder-decoder for statistical machine translation.
cite arxiv:1406.1078Comment: EMNLP 2014.
- [Elman, 1990] Elman, J. L. (1990).
Finding structure in time.
Cognitive Science, 14(2):179–211.
- [Guo et al., 2017] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017).
On calibration of modern neural networks.
CoRR, abs/1706.04599.
- [Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997).
Long short-term memory.
Neural Comput., 9(8):1735–1780.
- [Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).
Imagenet classification with deep convolutional neural networks.
In *Advances in neural information processing systems*, pages 1097–1105.
- [Murphy, 2012] Murphy, K. P. (2012).
Machine Learning: A Probabilistic Perspective.
The MIT Press.