Meta-Learning
— an idiosyncratic tutorial

Yee Whye Teh

Statistics @ Oxford

DeepMind

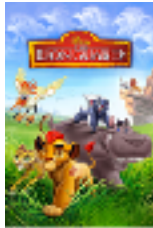http://csml.stats.ox.ac.uk/people/teh/

Small Data Problems

- Few Shot Learning

	terrier	beagle	labrador	cat	poodle
Training					

	?	?	?	?	?
Test					

Small Data Problems

- Recommender Systems

Yee Whye Teh

Small Data Problems

- Robotics

Yee Whye Teh

Credit: Sadeghi et al CVPR 2018
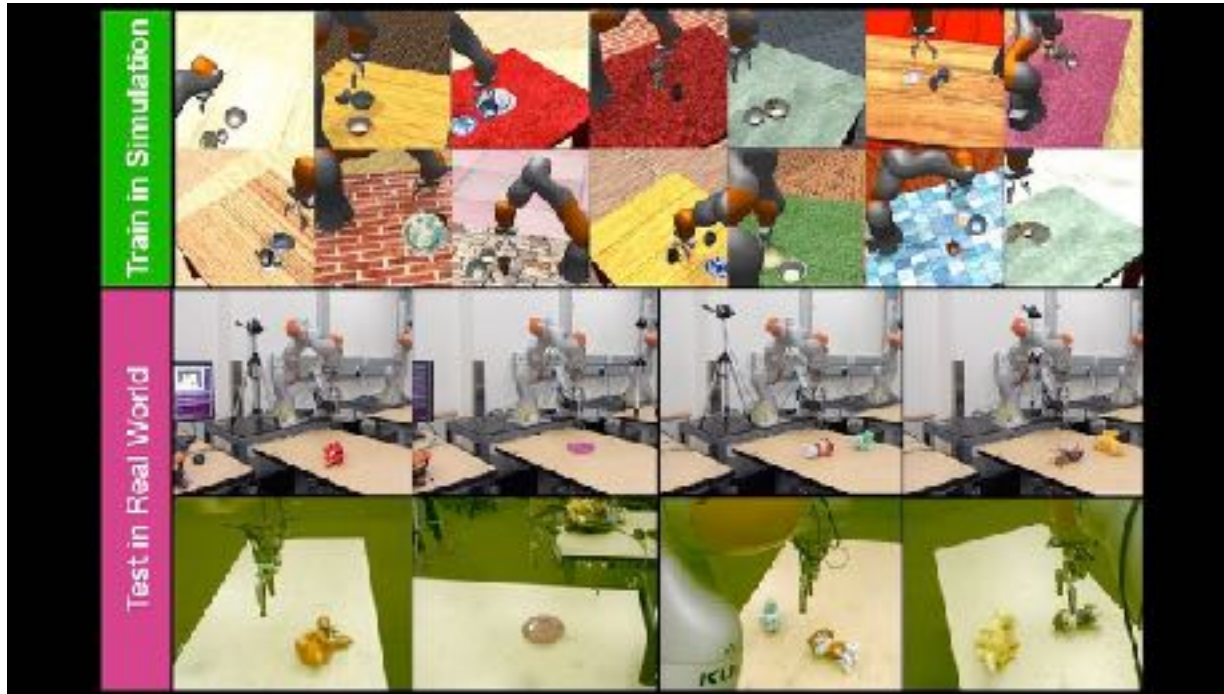
Small Data Problems

- Artificial General Intelligence

Yee Whye Teh

Machine Learning with Big Data

Credit: Yann LeCun, MNIST
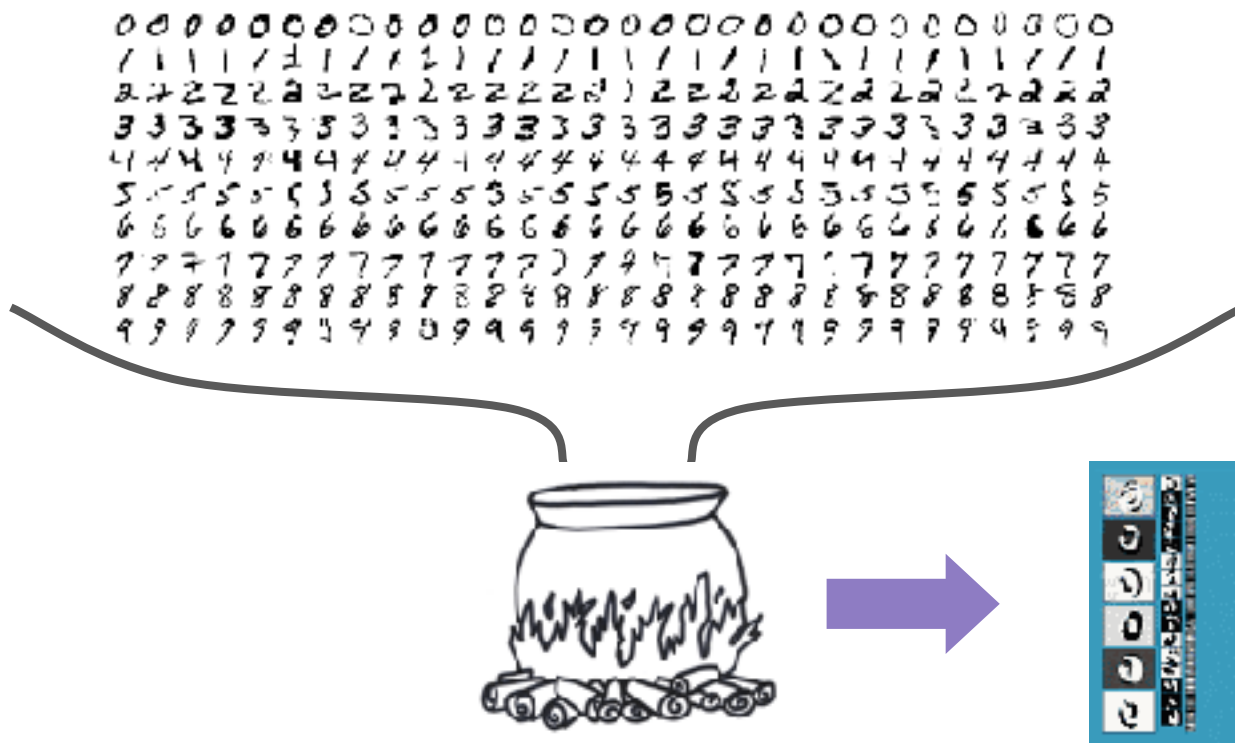
Machine Learning with Small Data

Inductive biases
- Preprocessing
- Feature engineering
- Data augmentation
- Model architectures
- Regularisation

Yee Whye Teh

Credit: Yann LeCun, MNIST

Meta-Learning, Learning-to-Learn

Inductive biases

Yee Whye Teh

Meta-Learning: an idiosyncratic tutorial

- Optimisation perspective on meta-learning
  - Optimisation-based meta-learning
  - Black-box meta-learning

- Probabilistic perspective on meta-learning
  - Stochastic processes
  - Neural processes
  - Uncertainty in meta-learning

- Probabilistic symmetries and neural architectures

- Note: no meta reinforcement learning (meta-RL)
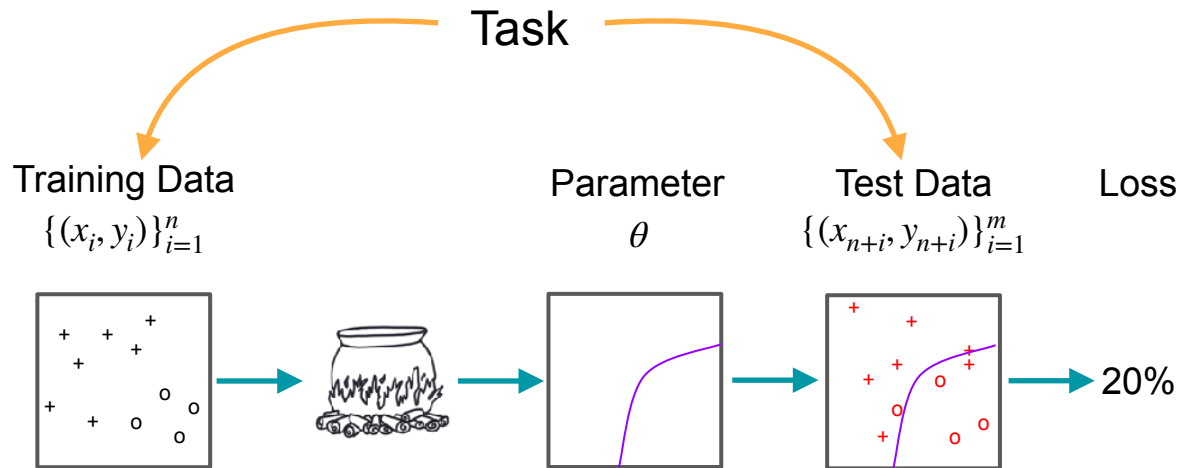
Yee Whye Teh

Meta-Learning: an idiosyncratic tutorial

- **Optimisation perspective on meta-learning**
  - Optimisation-based meta-learning
  - Black-box meta-learning

- Probabilistic perspective on meta-learning
  - Stochastic processes
  - Neural processes
  - Uncertainty in meta-learning

- Probabilistic symmetries and neural architectures
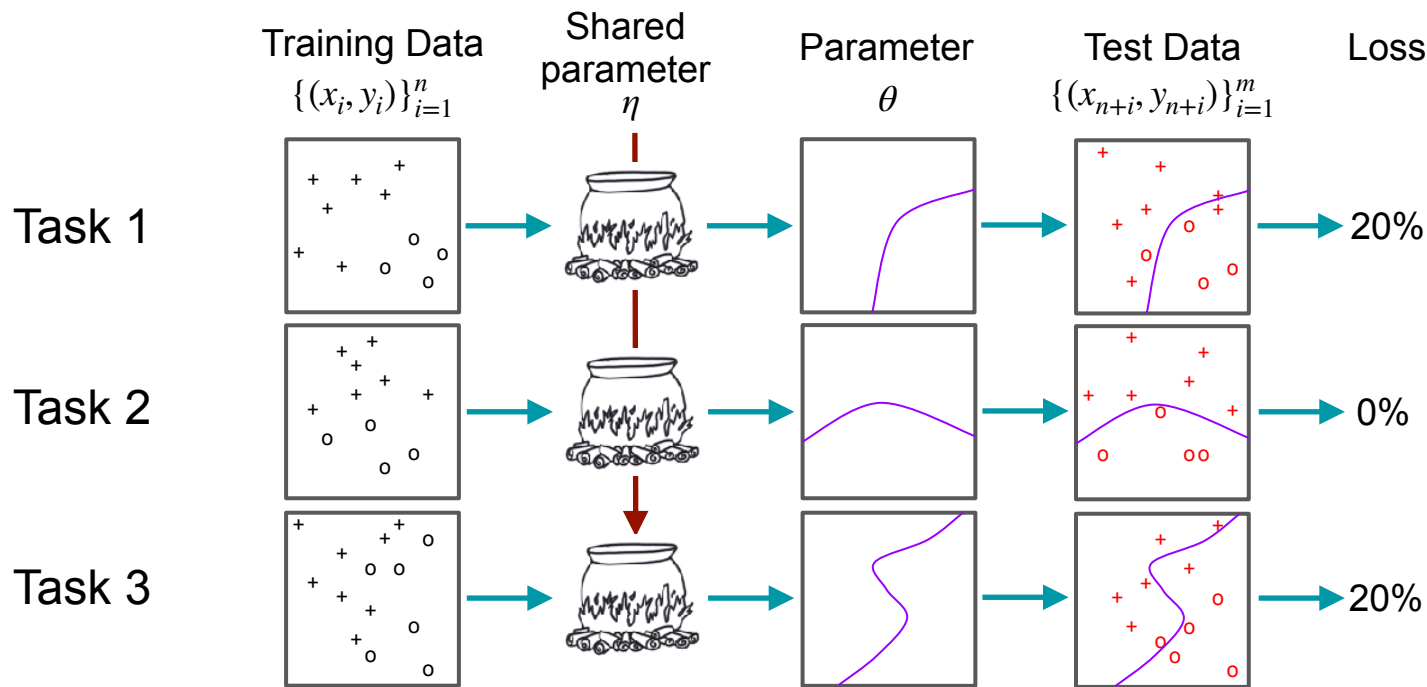
- Note: no meta reinforcement learning (meta-RL)

Yee Whye Teh

Single-Task Learning

Task

Training Data
$\{(x_i, y_i)\}_{i=1}^n$

Parameter
θ

Test Data
$\{(x_{n+i}, y_{n+i})\}_{i=1}^m$

Loss

20%

Empirical Risk
Minimisation

$$\arg\min_\theta \sum_{i=1}^n \text{Loss}(f_{\eta,\theta}(x_i), y_i) + \text{Regulariser}(\theta)$$

Yee Whye Teh

Multi-Task Learning

$$\arg\min_{\eta,\{\theta_j\}} \sum_{j=1}^{T} \sum_{i=1}^{n} \text{Loss}(f_{\eta,\theta_j}(x_{ji}), y_{ji}) + \text{Regulariser}(\theta_j)$$

Yee Whye Teh

Meta-Learning and Learning-to-Learn

- Is it possible to learn to generalise from training to test data?
- Learn:

$$\theta_j = \arg\min_{\theta_j} \sum_{i=1}^{n} \mathsf{Loss}(f_{\eta,\theta_j}(x_{ji}), y_{ji}) =: \mathsf{Learner}(\eta, \mathsf{TrainData}_j)$$

- Test performance:

$$\sum_{i=n+1}^{n+m} \mathsf{Loss}(f_{\eta,\theta_j}(x_{ji}), y_{ji}) =: \mathscr{L}(\eta, \theta_j, \mathsf{TestData}_j)$$
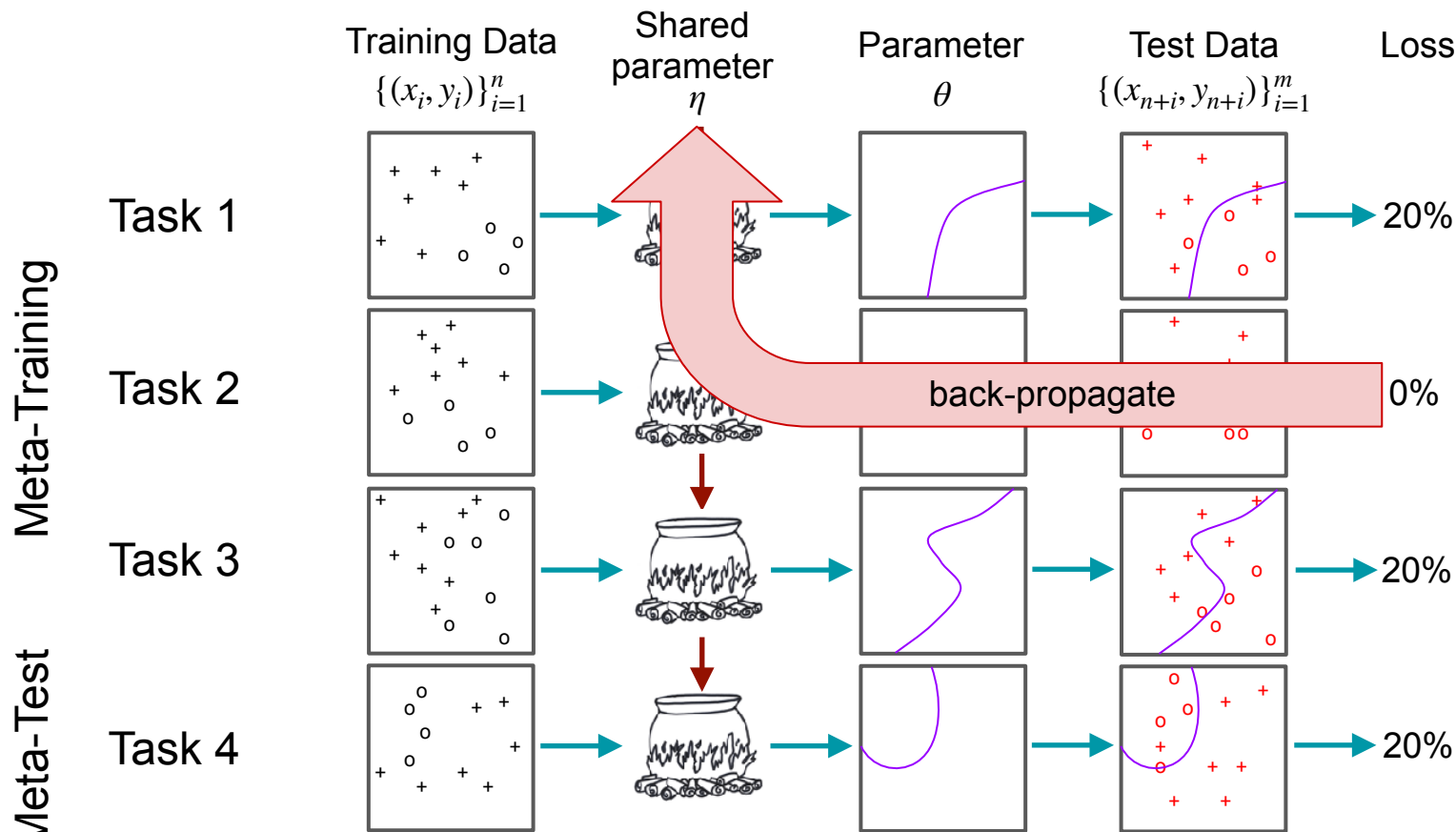
- Optimise shared parameters for test performance:

$$\arg\min_{\eta} \sum_{j=1}^{T} \mathscr{L}(\eta, \theta_j, \mathsf{TestData}_j) = \arg\min_{\eta} \sum_{j=1}^{T} \sum_{i=n+1}^{n+m} \mathsf{Loss}(f_{\eta,\mathsf{Learner}(\eta,\mathsf{TrainData}_j)}(x_{ji}), y_{ji})$$

- "Our training procedure is based on a simple machine learning principle: test and train conditions must match" — [Vinyals et al NeurIPS 2016]

Yee Whye Teh

Meta-Learning

Yee Whye Teh

Meta-Learning

- For each iteration:
  - Pick (minibatch of) training task j:
  - Base learner:

    $$\theta_j = \text{Learner}(\eta, \text{TrainData}_j) = \arg\min_{\theta_j} \sum_{i=1}^{n} L(f_{\eta,\theta_j}(x_{ji}), y_{ji})$$

  - Test performance:

    $$\mathscr{L}(\eta, \theta_j, \text{TestData}_j) = \sum_{i=n+1}^{n+m} \text{Loss}(f_{\eta,\theta_j}(x_{ji}), y_{ji})$$
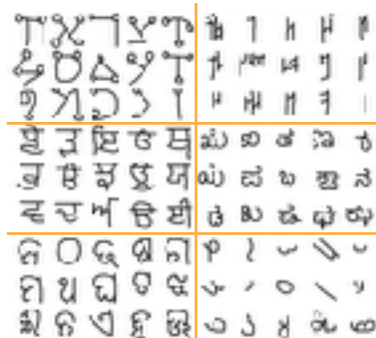
  - Meta-learner: optimise shared parameters for test performance:

    $$\eta \leftarrow \eta - \epsilon \frac{d}{d\eta} \sum_{i=n+1}^{n+m} \text{Loss}(f_{\eta,\text{Learner}(\eta,\text{TrainData}_j)}(x_{ji}), y_{ji})$$
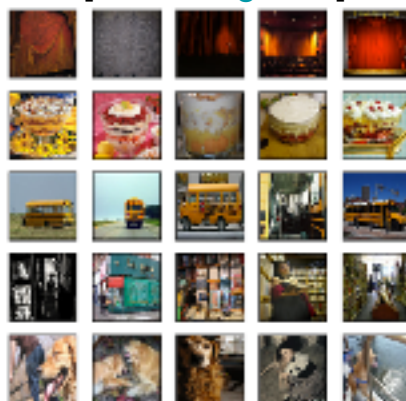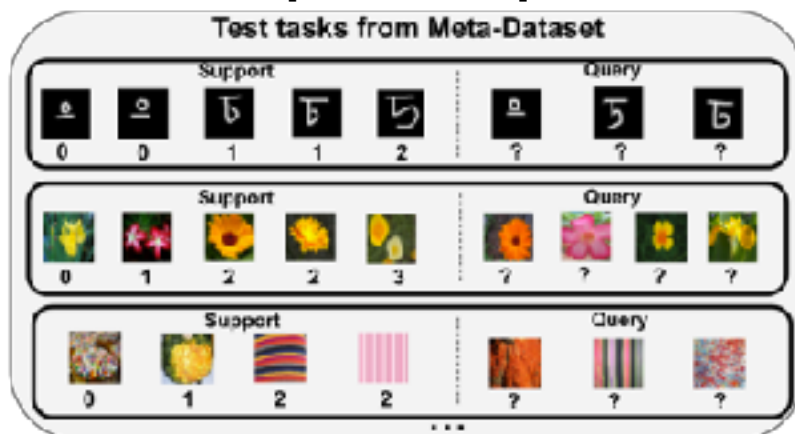
Yee Whye Teh

Image Datasets

[mini-ImageNet]

[Omniglot]

[Meta Dataset]

Test tasks from Meta-Dataset

Support					Query		
0	0	1	1	2	?	?	?

Support					Query		
0	1	2	2	3	?	?	?

Support				Query		
0	1	2	2	?	?	?

...

Yee Whye Teh

| Dataset (other names) |
|---|
| ilsvrc_2012 (ImageNet, ILSVRC) [instructions] |
| omniglot [instructions] |
| aircraft (FGVC-Aircraft) [instructions] |
| cu_birds (Birds, CUB-200-2011) [instructions] |
| dtd (Describable Textures, DTD) [instructions] |
| quickdraw (Quick, Draw!) [instructions] |
| fungi (FGVCx Fungi) [instructions] |
| vgg_flower (VGG Flower) [instructions] |
| traffic_sign (Traffic Signs, German Traffic Sign Recognition Benchmark, GTSRB) [instructions] |
| mscoco (Common Objects in Context, COCO) [instructions] |
| Total (All datasets) |

Different Kinds of Meta-Learning Methods

- Different choices of **Learner** and function class f lead to different methods.
  - Optimisation-based
  - black-box
  - metric-based
  - memory-based
  - Bayesian…
- Different choices correspond to different **inductive biases**.
  - Psychological theories
  - Symmetries, invariances and equivariances
  - Computational and learnability constraints
  - Intuitions and experimental findings

Yee Whye Teh

Optimisation-based Meta-Learning
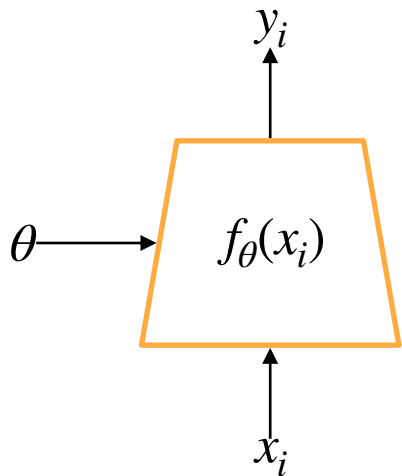
- Optimisation-based base learner.

$$\theta_0$$
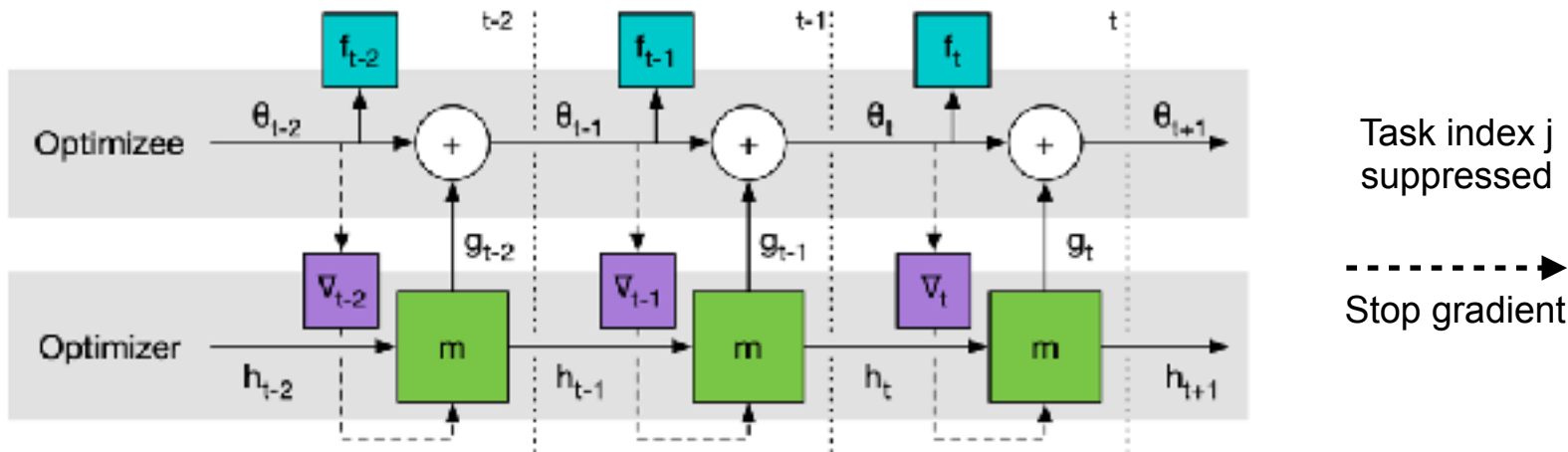$$\theta_1 \leftarrow \theta_0 + \mathsf{Update}_\eta(\nabla_{\theta_0} L(f_{\theta_0}(X_1), Y_1))$$
$$\theta_2 \leftarrow \theta_1 + \mathsf{Update}_\eta(\nabla_{\theta_1} L(f_{\theta_1}(X_2), Y_2))$$
$$\vdots$$

- Meta-learn meta-parameters θ_0, η.

Yee Whye Teh

Learning to Learn by Gradient Descent by Gradient Descent & LSTM Meta-Learner

Task index j suppressed

- - - - - - - →
Stop gradient

$$f_t = f_{\theta_t}(X_t)$$

$$\nabla_t = \frac{d}{d\theta_t} L(f_t, Y_t)$$

X_t : training inputs in iter t
Y_t : training targets in iter t

ADAM
$$\mu_t = \beta_1 \mu_{t-1} + (1 - \beta_1) \nabla_t$$
$$V_t = \beta_2 V_{t-1} + (1 - \beta_2) \nabla_t^2$$
$$g_t = -\frac{\epsilon}{\sqrt{V_t} + \delta} \mu_t$$

$$\theta_{t+1} = \theta_t + g_t$$

Yee Whye Teh

[Andrychowicz NeurIPS 2016, Ravi & Larochelle ICLR 2017]

Model-Agnostic Meta-Learning (MAML)

— meta-learning
---- learning/adaptation
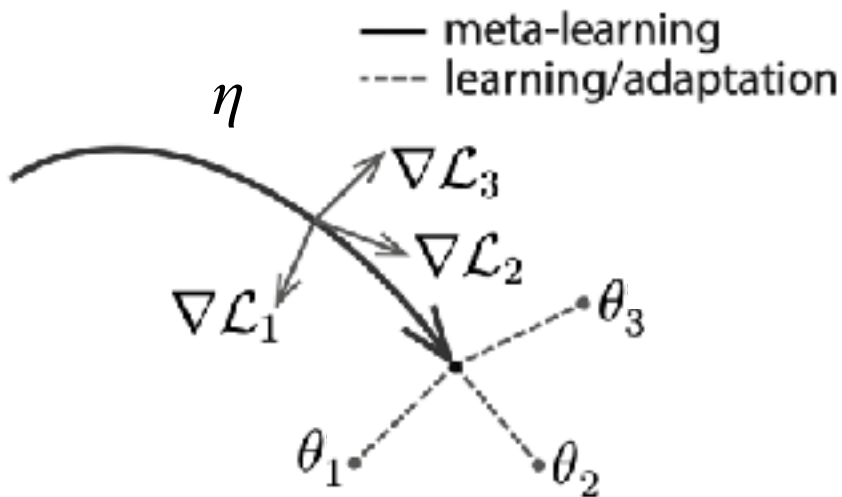
Single (or few) step gradient descent learner:

$$\theta_j = \eta - \epsilon \, \nabla_\eta L(f_\eta(X_j), Y_j)$$

Meta-learn shared initialisation η:

$$\eta \leftarrow \eta - \epsilon_0 \frac{d}{d\eta} \sum_j L(f_{\theta_j}(X_{jt}), Y_{jt})$$

Compute gradient through learner gradient step.

Yee Whye Teh

Model-Agnostic Meta-Learning (MAML)

$$\theta_j = \eta - \epsilon \nabla_\eta L(f_\eta(X_j), Y_j)$$

$$\frac{d}{d\eta} L(f_{\theta_j}(X_{jt}), Y_{jt}) = \nabla_{\theta_j} L(f_{\theta_j}(X_{jt}), Y_{jt}) \frac{d\theta_j}{d\eta}$$

$$= \nabla_{\theta_j} L(f_{\theta_j}(X_{jt}), Y_{jt}) \Big(I - \epsilon \nabla_\eta^2 L(f_\eta(X_{jt}), Y_{jt}) \Big)$$

- Difficulty is in computing $\nabla_{\theta_j} L(f_{\theta_j}(X_{jt}), Y_{jt}) \nabla_\eta^2 L(f_\eta(X_{jt}), Y_{jt})$.

- Vector-Hessian products can be automatically computed in linear time.

Yee Whye Teh

[Pearlmutter NECO 1994]

Optimisation-based Meta-Learning

- Two level optimisation problem
  - Flexible and applicable to wide range of neural architectures
  - Positive inductive bias
- Challenges
  - sensitive to neural architectures
  - can be expensive and unstable
- Difficulties of back-propagating through many base learner iterations.
  - If base learner optimisation can be analytically solved, gradients can be computed exactly [Harrison et al WAFC 2018, Lee et al CVPR 2019].
  - Alternatively, use implicit function theorem [Rajeswaran et al NeurIPS 2019].

Yee Whye Teh

Black-box Meta-Learning

- So far, the base learner is an **optimisation-based** learner.
- Not all learning algorithms are optimisation-based.

- For example, in k-nearest neighbour, learner simply "memorises" training data, and matches test inputs with memory to make predictions.

- **Black-box meta-learning**: Implement the base learner using a differentiable programming framework.

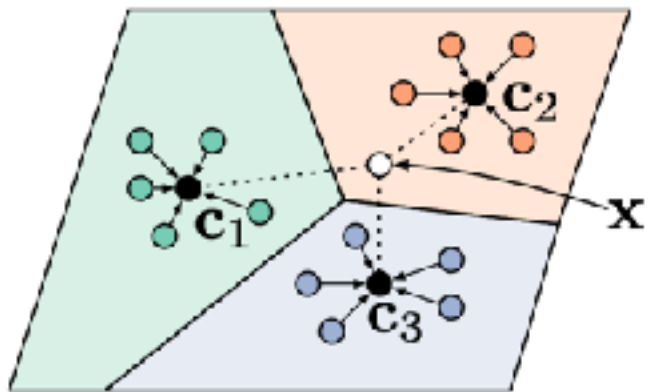Yee Whye Teh
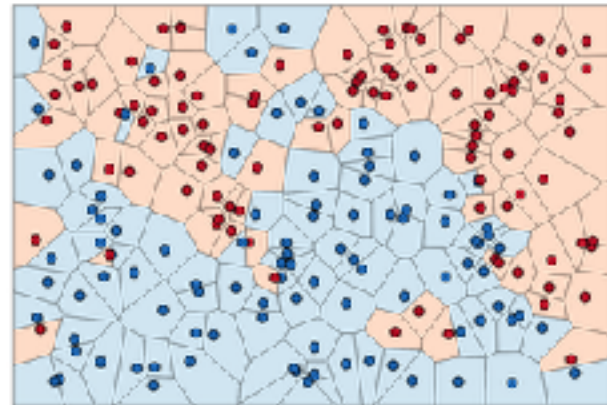
Matching Networks
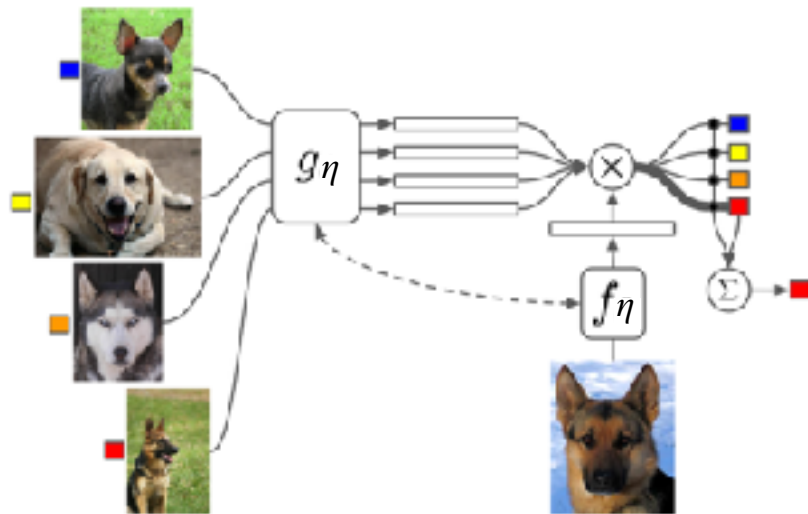
- Embed each train input $x_i \mapsto g_\eta(x_i)$.
- Embed test input $x_i \mapsto f_\eta(x_i)$

- "Softened" 1-NN classifier:

$$p(\,\cdot\,|x_{n+j}, \mathsf{TrainData}) = \sum_{i=1}^{n} \frac{e^{g_\eta(x_i)^\top f_\eta(x_{n+j})}}{\sum_{l=1}^{n} e^{g_\eta(x_l)^\top f_\eta(x_{n+j})}} y_i$$

- Memory-based meta-learning
- Metric-based meta-learning

Yee Whye Teh

[Vinyals et al NeurIPS 2016]

Prototypical Networks

- Embed each input $x_i \mapsto f_\eta(x_i)$.
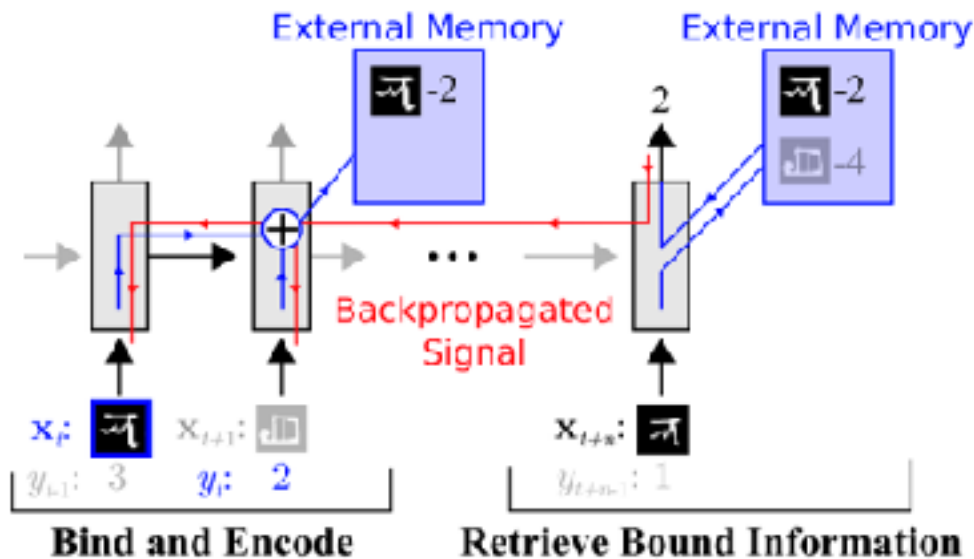
- Form "prototypes":
$$c_k = \frac{\sum_{i:y_i=k} f_\eta(x_i)}{\sum_{i:y_i=k} 1}$$

- Predict:
$$p(y_{n_j} = k \,|\, x_{n+j}, \mathsf{TrainData}) = \frac{e^{-\|f_\eta(x_{n+j})-c_k\|^2/\sigma^2}}{\sum_{l=1}^{K} e^{-\|f_\eta(x_{n+j})-c_l\|^2/\sigma^2}}$$

Yee Whye Teh

[Snell et al NeurIPS 2017]

Memory-Augmented Neural Networks (MANNs)

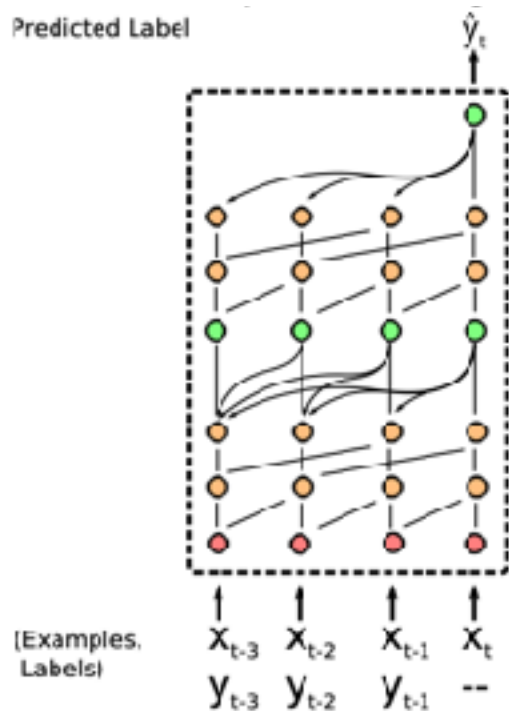Each task: sequence of input/output pairs $(x_1, y_1), (x_2, y_2), \ldots$

Base learner directly predicts

$$p_\eta(y_t \mid (x_i, y_i)_{i=1}^{t-1}, x_t)$$

Neural Turing machine with external memory.

Yee Whye Teh

[Santoro et al ICML 2016]
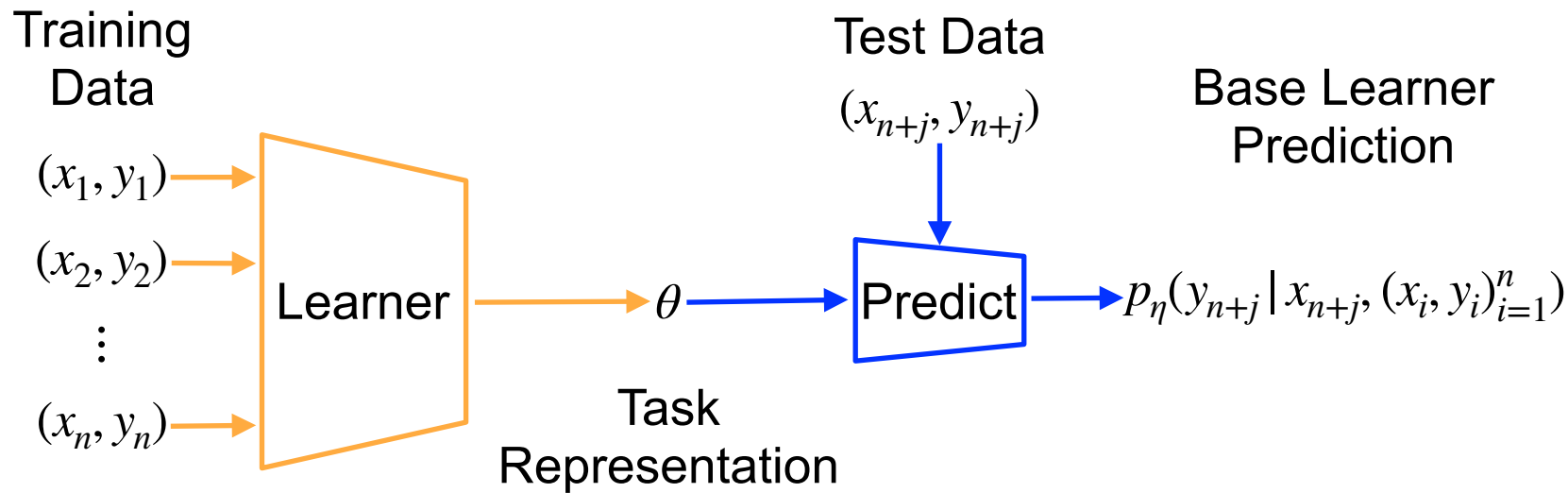
Simple Neural AttentIve Learner (SNAIL)

$$p_\eta(y_t \mid (x_i, y_i)_{i=1}^{t-1}, x_t)$$

Simple:
- treats base learner problem as sequential prediction
- Use (causal) convolution and attention layers.

Yee Whye Teh

[Mishra et al ICLR 2018]

Base Learner Architecture and Task Representation

Training
Data

Test Data

(x_{n+j}, y_{n+j})

Base Learner
Prediction

(x_1, y_1)

(x_2, y_2)

$\vdots$

(x_n, y_n)

Learner

θ

Predict

$p_\eta(y_{n+j} \mid x_{n+j}, (x_i, y_i)_{i=1}^n)$

Task
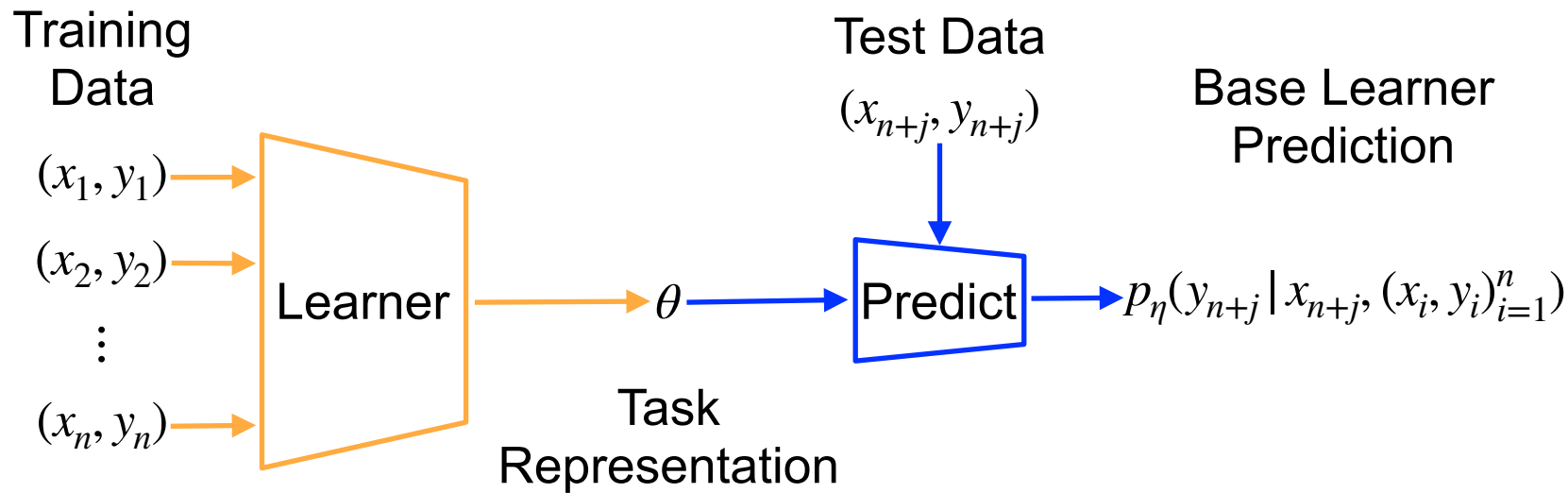Representation

Yee Whye Teh

Black-box Meta-Learning

- Learn a differentiable function to map from training data to test predictions.
  - Simple: reduces meta-learning back to supervised learning.
  - Broad and flexible framework.

- Challenges:
  - Harder to learn as no inductive bias for base learner to optimise on training data.
  - Less able to generalise out of meta-training distribution.

Yee Whye Teh

Base Learner Architecture and Task Representation

Training
Data

(x_1, y_1)

(x_2, y_2)

$\vdots$

(x_n, y_n)

Learner

θ

Task
Representation

Test Data

(x_{n+j}, y_{n+j})

Predict

Base Learner
Prediction

$p_\eta(y_{n+j} | x_{n+j}, (x_i, y_i)_{i=1}^n)$
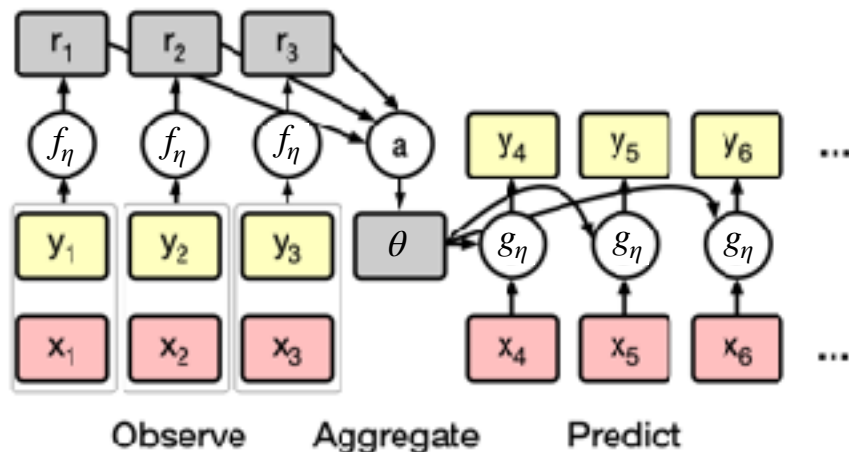
Permutation Invariance

- Assumed that data items within each task are iid.
- Learner function should be invariant to permutations of the training data:

$$\text{Learner}(\eta, \{(x_1, y_1), \ldots, (x_n, y_n)\}) = \text{Learner}(\eta, \{(x_{\pi(1)}, y_{\pi(1)}), \ldots, (x_{\pi(n)}, y_{\pi(n)})\})$$

- MAML, Prototypical Nets (and simpler version of Matching Nets) are.
- LSTM meta-learner, MANN, SNAIL (and a version of Matching Nets) are not permutation invariant.

- Permutation invariance is an inductive bias.
- How to design neural architectures for permutation invariance? [more later]

Yee Whye Teh

Conditional Neural Processes

Observe   Aggregate   Predict

- Embed input/output pairs
$$(x_i, y_i) \mapsto f_\eta(x_i, y_i)$$

- Aggregate embeddings
$$\theta = \frac{1}{n} \sum_{i=1}^{n} f_\eta(x_i, y_i)$$

- Permutation invariant.
- Interpret θ as a task representation.

Yee Whye Teh

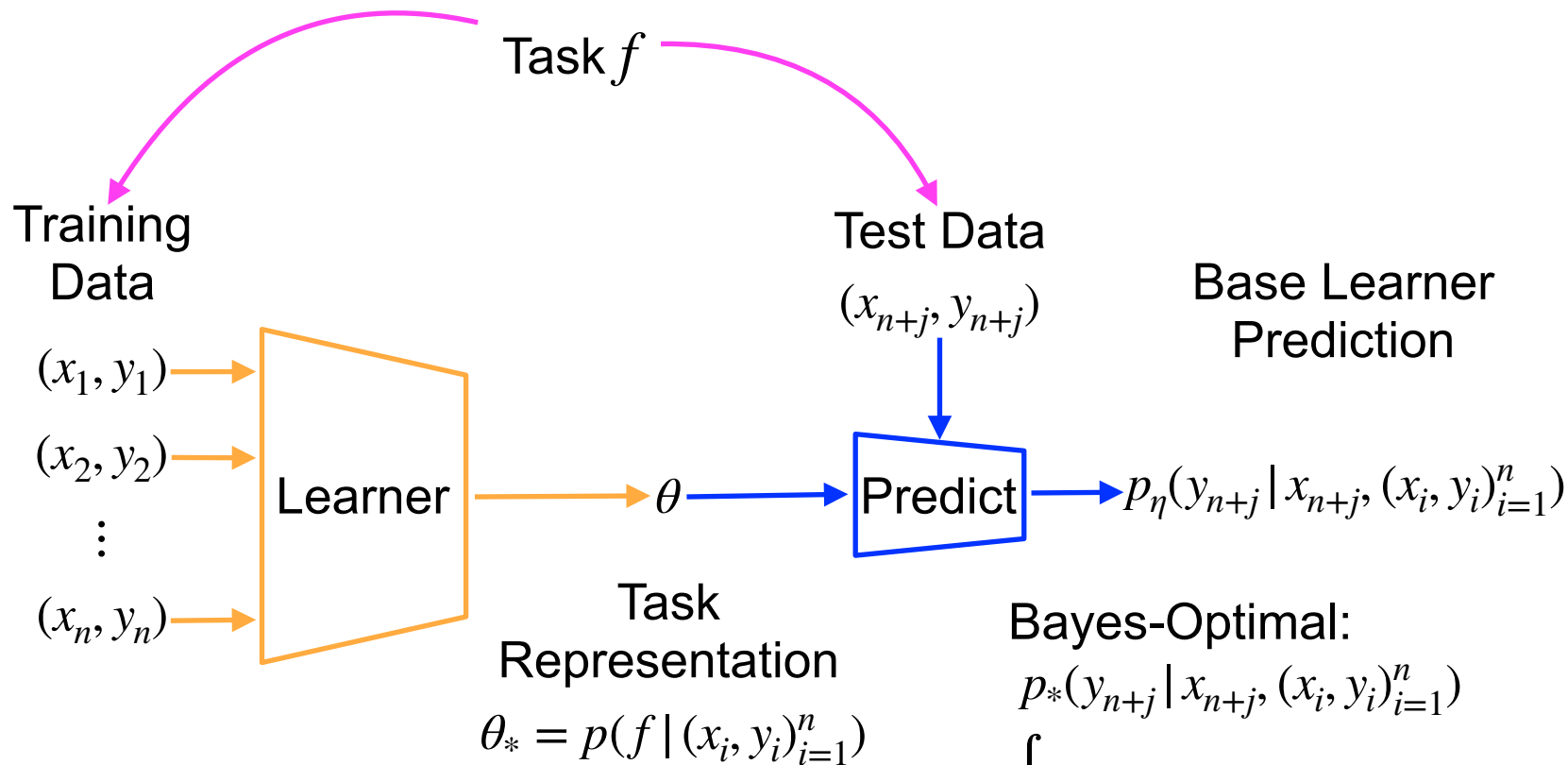[Garnelo et al ICML 2018, Garnelo et al ICML DeepGen Workshop 2018]

Meta-Learning: an idiosyncratic tutorial

- Optimisation perspective on meta-learning
  - Optimisation-based meta-learning
  - Black-box meta-learning

- **Probabilistic perspective on meta-learning**
  - Stochastic processes
  - Neural processes
  - Uncertainty in meta-learning

- Probabilistic symmetries and neural architectures

- Note: no meta reinforcement learning (meta-RL)
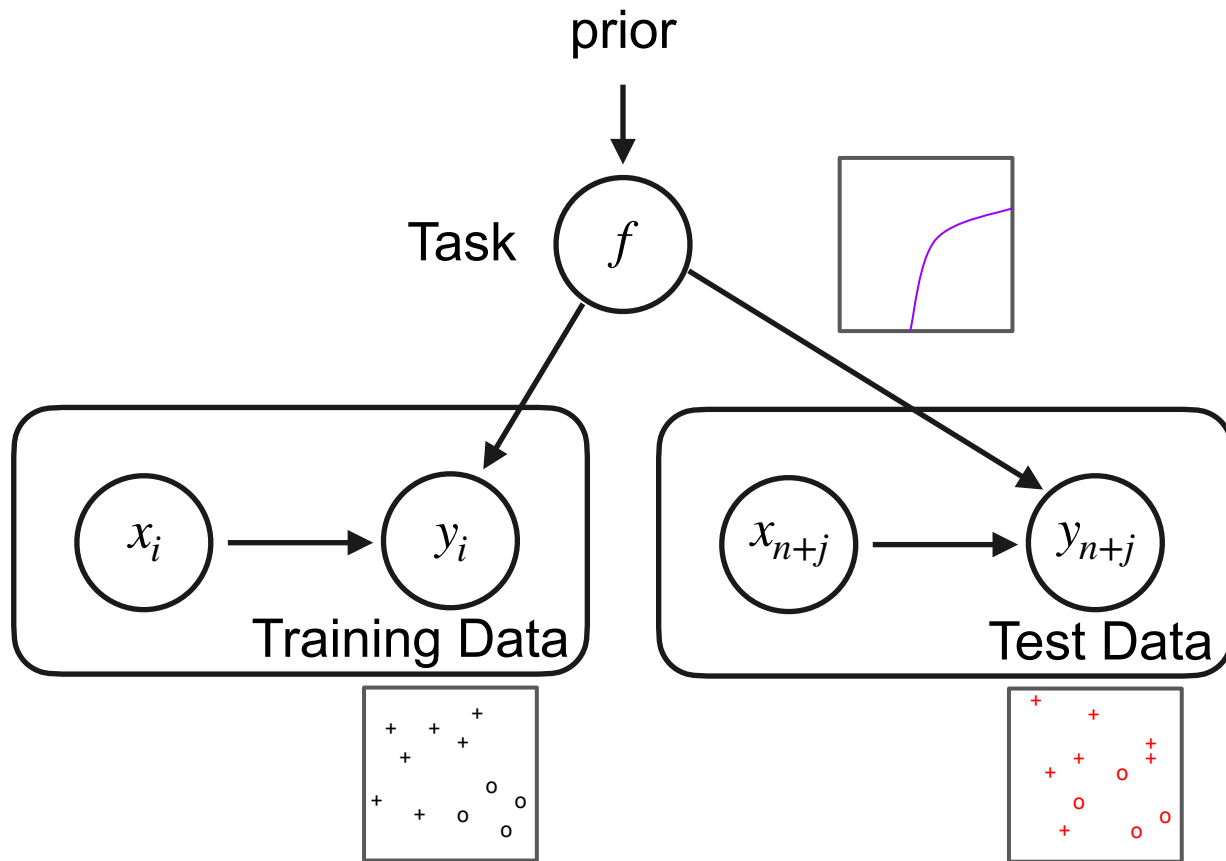
Yee Whye Teh

Probabilistic Perspective on Meta-Learning

Task f

Training Data

(x_1, y_1)
(x_2, y_2)
$\vdots$
(x_n, y_n)

Learner

θ

Task Representation
$\theta_* = p(f \mid (x_i, y_i)_{i=1}^n)$

Test Data
(x_{n+j}, y_{n+j})

Predict

Base Learner Prediction

$p_\eta(y_{n+j} \mid x_{n+j}, (x_i, y_i)_{i=1}^n)$

Bayes-Optimal:
$p_*(y_{n+j} \mid x_{n+j}, (x_i, y_i)_{i=1}^n)$
$= \int p(y_{n+j} \mid x_{n+j}, f) p(f \mid (x_i, y_i)_{i=1}^n) df$

Yee Whye Teh

Generative Process

prior

Task f

Training Data

$x_i \rightarrow y_i$

Test Data

$x_{n+j} \rightarrow y_{n+j}$
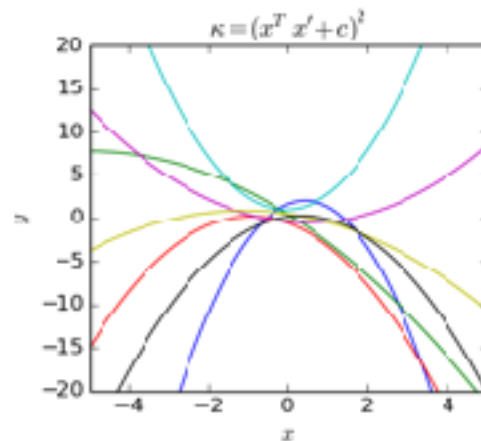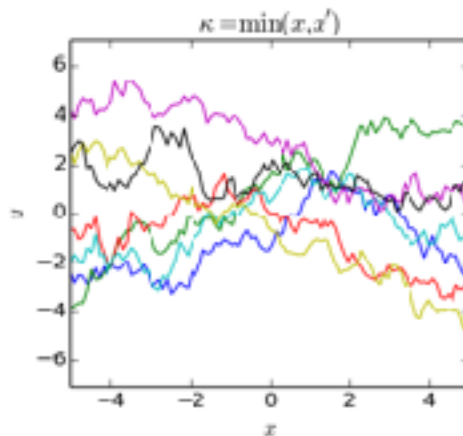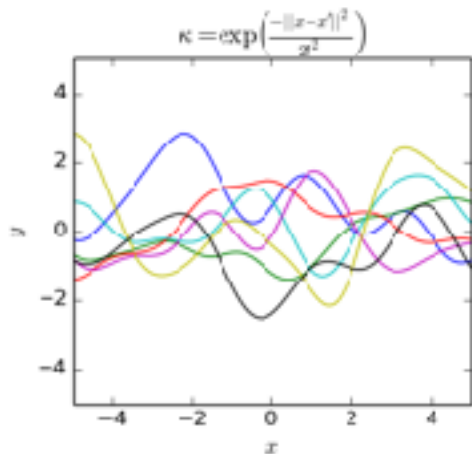
Specifying Stochastic Processes

- Gaussian processes are typically described via marginal distributions:

$$\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_t) \end{pmatrix} \sim \mathcal{N} \left(\begin{pmatrix} \mu(x_1) \\ \mu(x_2) \\ \vdots \\ \mu(x_t) \end{pmatrix}, \begin{pmatrix} K(x_1, x_1) & K(x_1, x_2) & \cdots & K(x_1, x_t) \\ K(x_2, x_1) & K(x_2, x_2) & \cdots & K(x_2, x_t) \\ \vdots & & \ddots & \vdots \\ K(x_t, x_1) & K(x_t, x_2) & \cdots & K(x_t, x_t) \end{pmatrix} \right)$$

Yee Whye Teh

[Rasmussen & Williams 2006]

Specifying a Stochastic Process

- A stochastic process is a joint distribution over an infinite collection of random variables $(f(x))_{x \in \mathcal{X}}$.

- Kolmogorov Extension Theorem:
  - Constructs a stochastic process by specifying its finite dimensional marginal distributions.

  - Family of finite dimensional joint distributions $\rho_{x_{1:n}}$, one for each $n \in \mathbb{N}$ and finite sequence $x_{1:n} \in \mathcal{X}$. We will want these to form the marginals:

$$\rho_{x_{1:n}}(y_{1:n}) = p(f(x_1) = y_1, \ldots, f(x_n) = y_n))$$

Yee Whye Teh

Exchangeability and Consistency

- Exchangeability: for each n, $x_{1:n}$, and permutation π of $\{1,\ldots,n\}$

$$\rho_{x_1,\ldots,x_n}(y_1,\ldots,y_n) = \rho_{x_{\pi(1)},\ldots,x_{\pi(n)}}(y_{\pi(1)},\ldots,y_{\pi(n)})$$

- Consistency: for each n, m, $x_{1:n+m}$

$$\rho_{x_1,\ldots,x_n}(y_1,\ldots,y_n) = \int \rho_{x_{1:n+m}}(y_{1:n+m})dy_{n+1:n+m}$$

Bayesian nonparametrics

- Gaussian processes: regression, classification [Neal 1994, Rasmussen & Williams 2006]
- Dirichlet processes: infinite mixture models, clustering [Neal JCGS 1999, Rasmussen NeurIPS 2000]
- Hierarchical Dirichlet processes: topic models and HMMs [Teh et al JASA 2006]
- Pitman-Yor and Poisson-Kingman processes: power laws and language models [Teh ACL 2006], species discovery problems [Favaro et al Biometrics 2015]
- Sparse networks models and power-law structures [Caron & Fox JRSSB 2017]

Yee Whye Teh

[Hjort et al 2011, Ghosal & van der Vaart 2017, Jordan & Teh 20XX]

Specifying Stochastic Processes

- Gaussian processes are typically described via marginal distributions:

$$\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_t) \end{pmatrix} \sim \mathcal{N} \left(\begin{pmatrix} \mu(x_1) \\ \mu(x_2) \\ \vdots \\ \mu(x_t) \end{pmatrix}, \begin{pmatrix} K(x_1, x_1) & K(x_1, x_2) & \cdots & K(x_1, x_t) \\ K(x_2, x_1) & K(x_2, x_2) & \cdots & K(x_2, x_t) \\ \vdots & & \ddots & \vdots \\ K(x_t, x_1) & K(x_t, x_2) & \cdots & K(x_t, x_t) \end{pmatrix} \right)$$

Yee Whye Teh
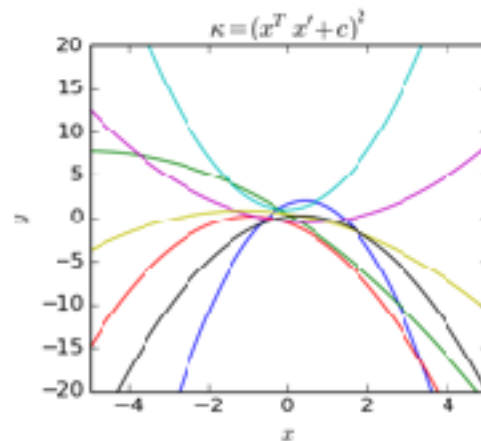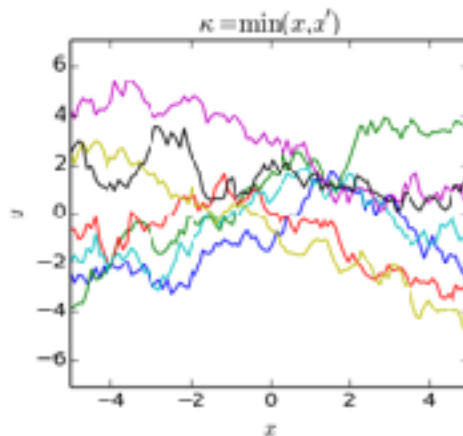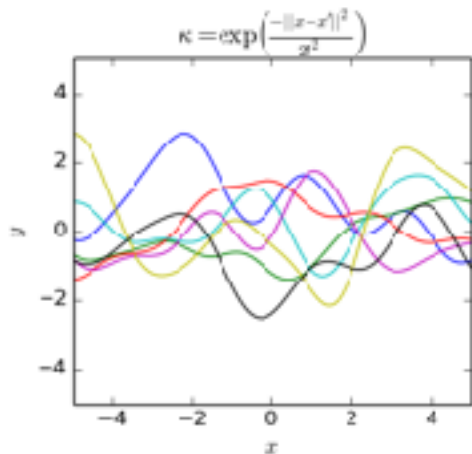
Specifying Stochastic Processes

- Gaussian processes can equivalently be described via its conditional distributions:

$$f(x_{t+1})|f(x_1) = y_1, \ldots, f(x_t) = y_t$$

$$\sim \mathcal{N}\left(\mu(x_{t+1}) + K_{t+1,1:t}K_{1:t,1:t}^{-1}y_{1:t}, \; K_{t+1,t+1} - K_{t+1,1:t}K_{1:t,1:t}^{-1}K_{1:t,t+1}\right)$$
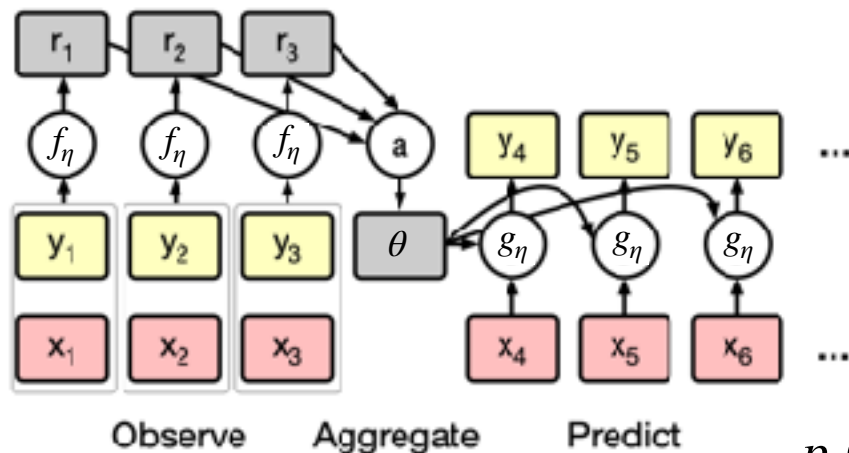
- In general, stochastic processes can also be described using a consistent family of conditional distributions:

$$p(f(x_{n+j}) = y_{n+j}|f(x_1) = y_1, \ldots, f(x_n) = y_n)$$

for training dataset $\{(x_i, y_i)\}_{i=1}^{n}$ and test data (x_{n+j}, y_{n+j}).

Yee Whye Teh

Conditional Neural Processes

Observe   Aggregate   Predict

- Embed input/output pairs

$$(x_i, y_i) \mapsto f_\eta(x_i, y_i)$$

- Aggregate embeddings

$$\theta = \frac{1}{n} \sum_{i=1}^{n} f_\eta(x_i, y_i)$$

- Predict

$$p_\eta(y_{n+j} | x_{n+j}, (x_i, y_i)_{i=1}^{n}) = \mathcal{N}(y_{n+j}; g_\eta(\theta, x_{n+j}))$$

- Meta-learning learns the stochastic process!

Yee Whye Teh

[Garnelo et al ICML 2018]

Conditional Neural Processes

Task = Function on 1D space.

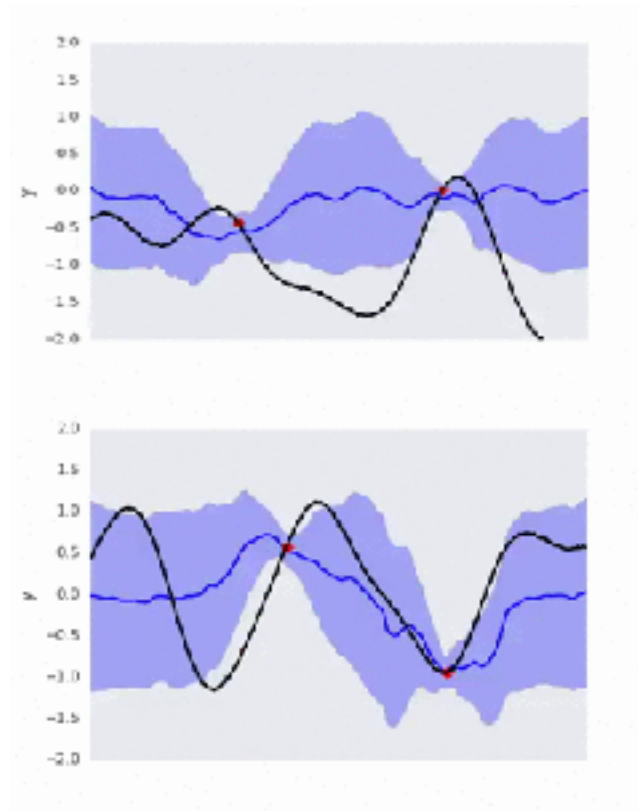Given training points, use neural processes to predict mean and std of function values at other locations.

Yee Whye Teh

[Garnelo et al ICML 2018]

Image Completion and Super-resolution

Task = Image = Function on 2D space.

Bottom half prediction

Super-resolution
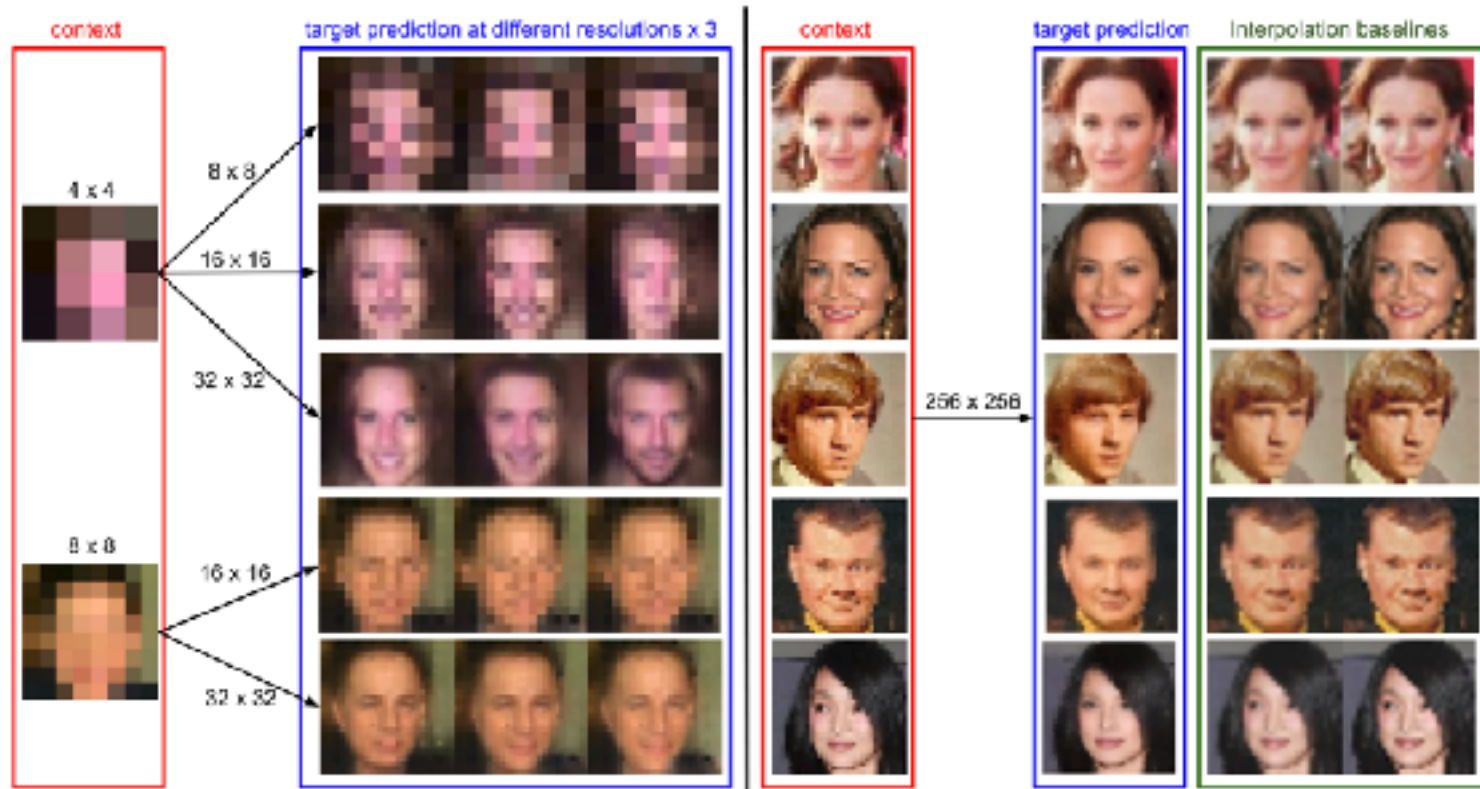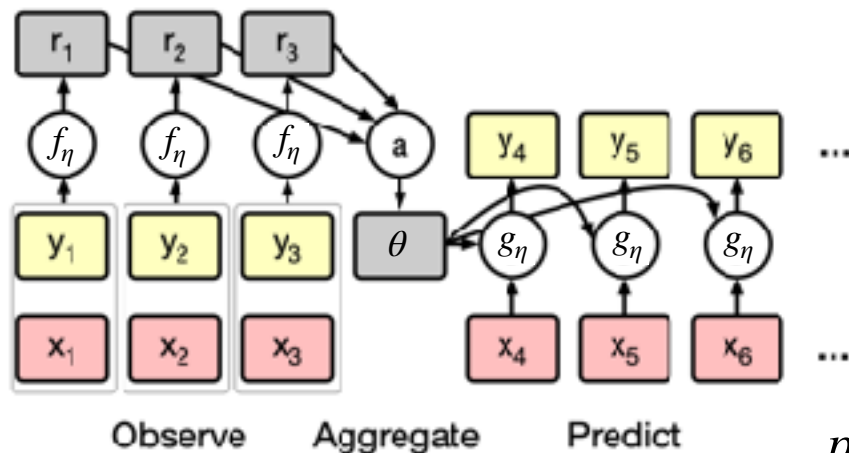
Yee Whye Teh

[Kim et al ICLR 2019]

Image Super-resolution

[Kim et al ICLR 2019]

Conditional Neural Processes

Observe   Aggregate   Predict

- Embed input/output pairs

$$(x_i, y_i) \mapsto f_\eta(x_i, y_i)$$

- Aggregate embeddings

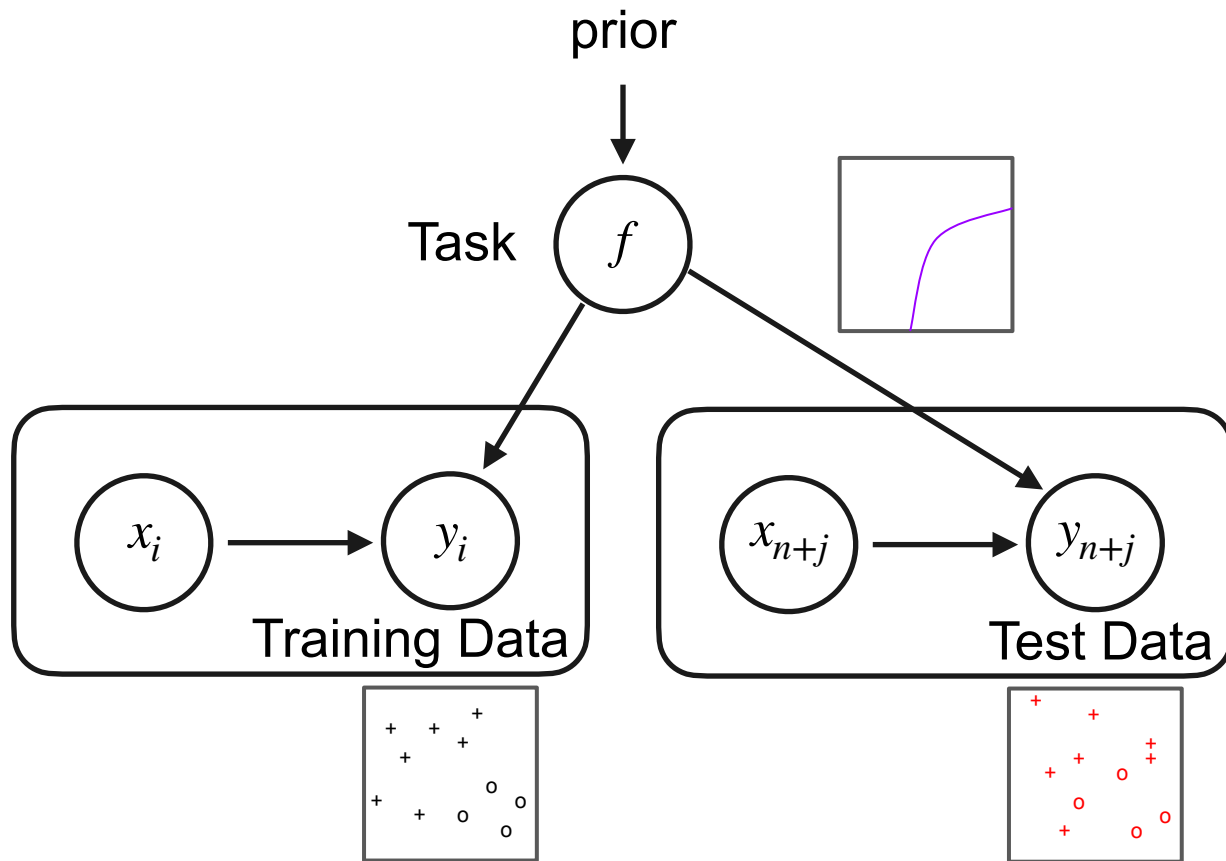$$\theta = \frac{1}{n} \sum_{i=1}^{n} f_\eta(x_i, y_i)$$

- Predict

$$p_\eta(y_{n+j} | x_{n+j}, (x_i, y_i)_{i=1}^{n}) = \mathcal{N}(y_{n+j}; g_\eta(\theta, x_{n+j}))$$

- But: architecture cannot model dependence among test outputs!

[Garnelo et al ICML 2018]

Generative Process

Yee Whye Teh

Latent Variable Model

- Generative **model**:

$$p_\eta(z, y_{1:n+m} \,|\, x_{1:n+m}) = p_\eta(z) \prod_{i=1}^{n+m} p_\eta(y_i \,|\, z, x_i)$$

$$p_\eta(z) = \mathcal{N}(z; 0, I)$$

$$p_\eta(y_i \,|\, z, x_i) = \mathcal{N}(y_i; g_\eta(z, x_i))$$

- Variational learning objective:

$$\log p(y_{1:n+m} \,|\, x_{1:n+m})$$

$$\geq \mathbb{E}_{q(z|x_{1:n+m}, y_{1:n+m})} \left[\sum_{i=1}^{n+m} \log p_\eta(y_i \,|\, z, x_i) + \log p(z) - \log q(z \,|\, x_{1:n+m}, y_{1:n+m}) \right]$$

$$q(z \,|\, x_{1:t}, y_{1:t}) = \mathcal{N}\left(z; s_\eta\left(\frac{1}{t} \sum_{i=1}^{t} f_\eta(x_i, y_i) \right) \right)$$

Yee Whye Teh

[Garnelo et al ICML DeepGen Workshop 2018]

Alternative Learning Objective

- "our training procedure is based on a simple machine learning principle: test and train conditions must match" — [Vinyals et al NeurIPS 2016]
- Alternative objective:

$$\log p(y_{n+:n+m} \,|\, x_{1:n}, y_{1:n}, x_{n+1:n+m})$$

$$\geq \mathbb{E}_{q(z|x_{1:n+m}, y_{1:n+m})} \left[\sum_{i=1}^{n+m} \log p_{\eta}(y_i \,|\, z, x_i) + \log p(z \,|\, x_{1:n}, y_{1:n}) - \log q(z \,|\, x_{1:n+m}, y_{1:n+m}) \right]$$

$$\approx \mathbb{E}_{q(z|x_{1:n+m}, y_{1:n+m})} \left[\sum_{i=1}^{n+m} \log p_{\eta}(y_i \,|\, z, x_i) + \log q(z \,|\, x_{1:n}, y_{1:n}) - \log q(z \,|\, x_{1:n+m}, y_{1:n+m}) \right]$$

$$q(z \,|\, x_{1:t}, y_{1:t}) = \mathcal{N}\left(z; s_{\eta}\left(\frac{1}{t} \sum_{i=1}^{t} f_{\eta}(x_i, y_i) \right) \right)$$

Yee Whye Teh

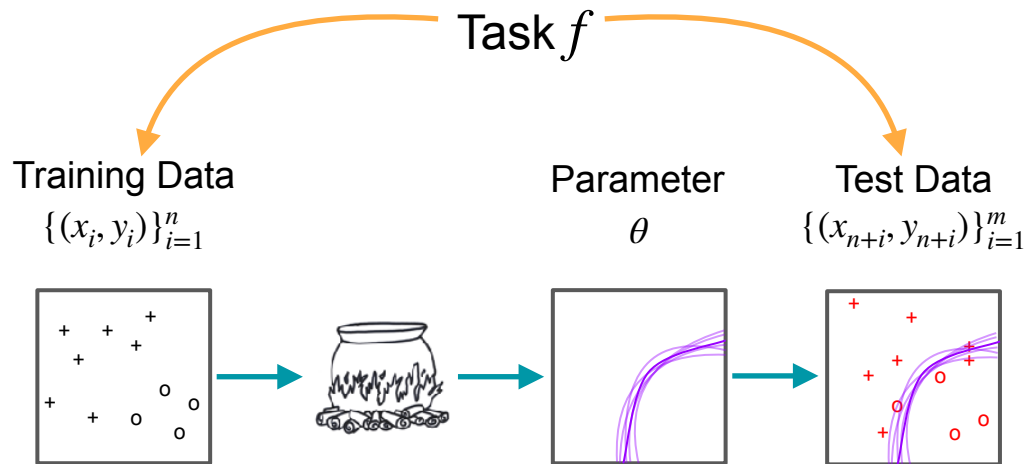[Kim et al ICLR 2019, Le et al NeurIPS BDL 2018]

Probabilistic Perspective on Meta-Learning

- Meta-learning as learning a stochastic process:
  - Meta-learning a prior over functions -> meta-learning inductive biases from meta-training set!
  - Base learner can be thought of as **amortized learning**.

- Uncertainties are important in meta-learning applications:
  - Active learning
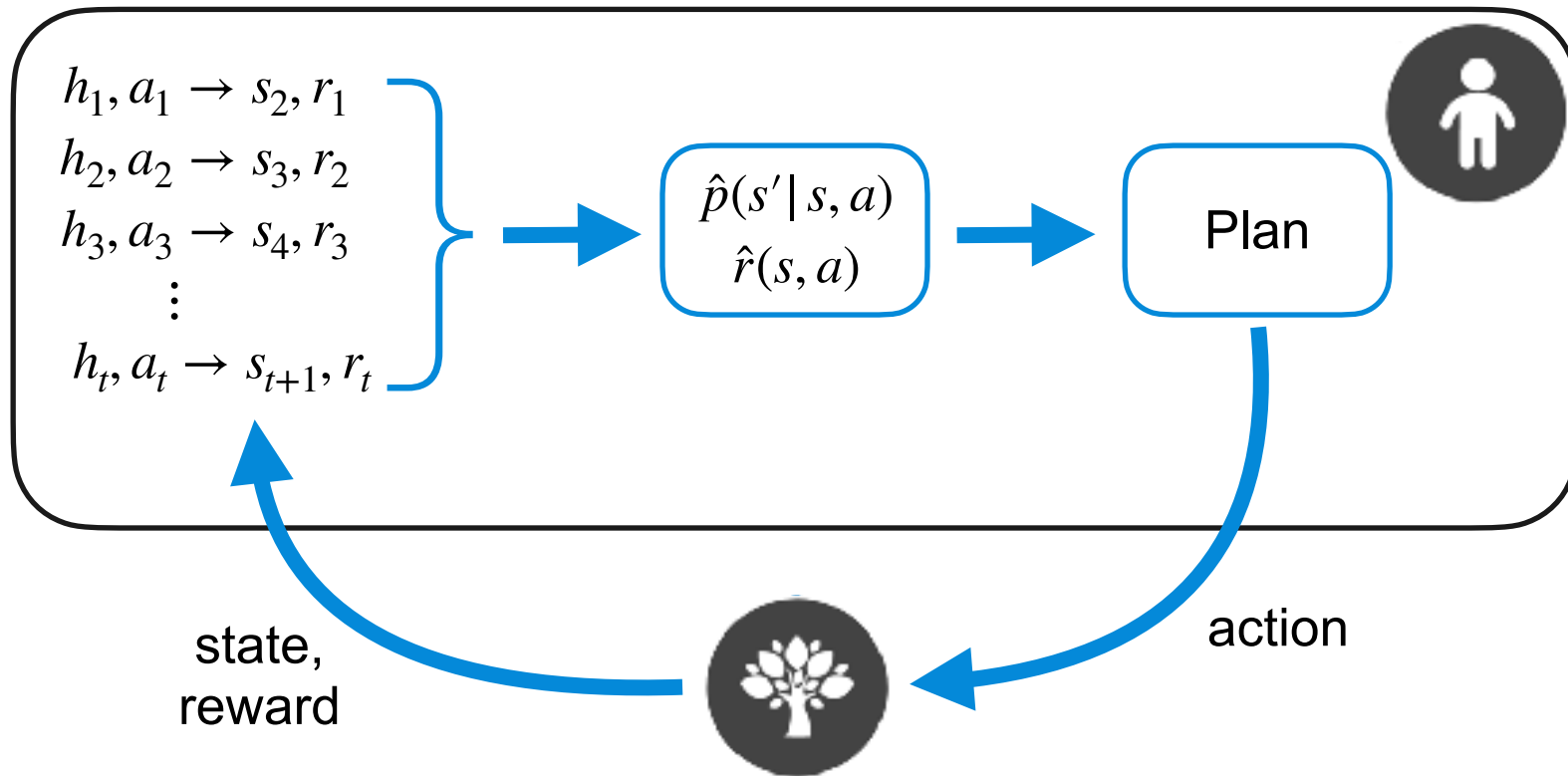 - Bayesian optimisation
  - Reinforcement learning

Yee Whye Teh

Uncertainty in Meta-Learning

Task f

Training Data
$\{(x_i, y_i)\}_{i=1}^{n}$

Parameter
θ

Test Data
$\{(x_{n+i}, y_{n+i})\}_{i=1}^{m}$
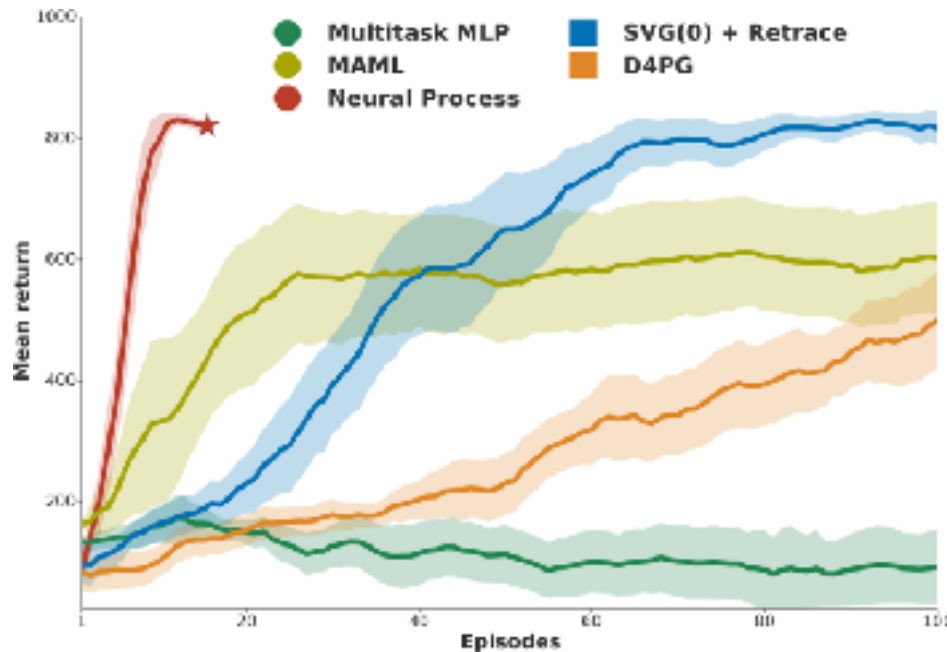
- Small training sets in meta-learning → uncertainty in task inference!
- Important for:
  - active learning,
  - Bayesian optimisation,
  - reinforcement learning

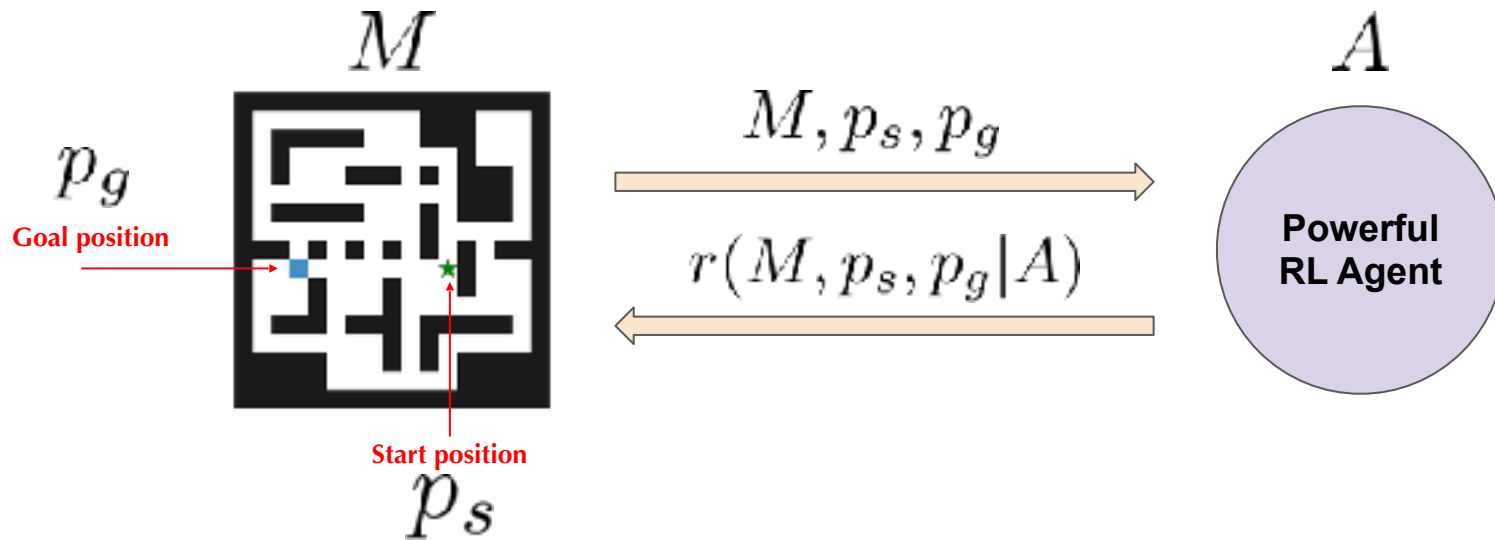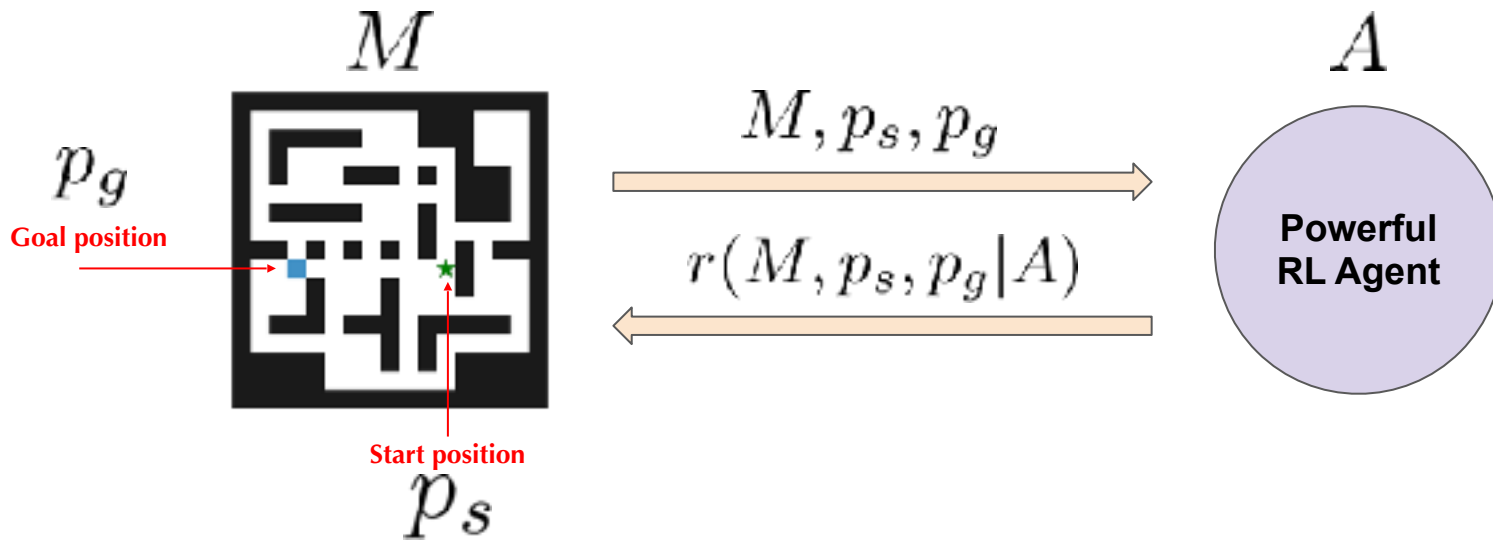Yee Whye Teh

Efficient Model-based Reinforcement Learning

$$h_1, a_1 \rightarrow s_2, r_1$$
$$h_2, a_2 \rightarrow s_3, r_2$$
$$h_3, a_3 \rightarrow s_4, r_3$$
$$\vdots$$
$$h_t, a_t \rightarrow s_{t+1}, r_t$$

$$\hat{p}(s' \mid s, a)$$
$$\hat{r}(s, a)$$

Plan

state, reward

action

Yee Whye Teh

Cart Pole

Yee Whye Teh

[Galashov et al 2019]

Adversarial Testing of RL Agents

M

p_g

Goal position

Start position

p_s

M, p_s, p_g

$r(M, p_s, p_g | A)$

A

Powerful RL Agent

Yee Whye Teh

[Galashov et al 2019]

Adversarial Testing of RL Agents

M

A

p_g

Goal position

M, p_s, p_g

$r(M, p_s, p_g | A)$

Powerful RL Agent
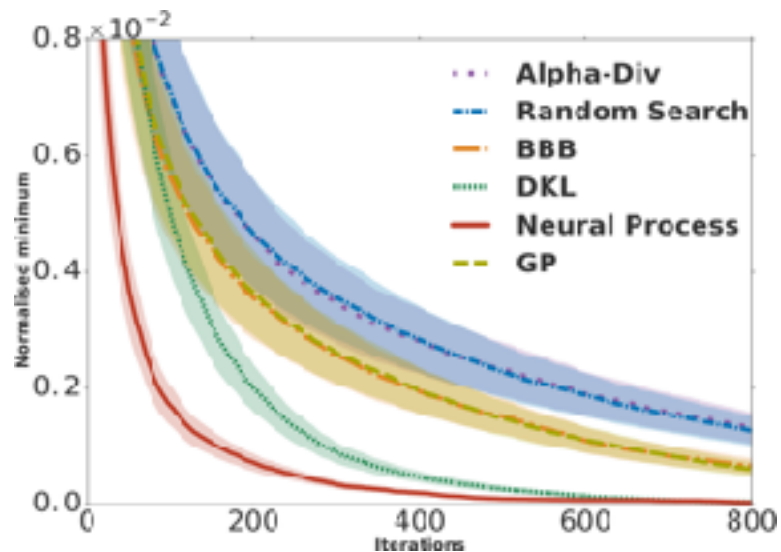
Start position

p_s

Bayesian Optimizaion
$$\min_{M, p_s, p_g} r(M, p_s, p_g | A)$$

$$(M, p_s, p_g, A) \sim p(\mathcal{T})$$ - training & holdout samples (agents, mazes, positions)

Yee Whye Teh

[Galashov et al 2019]

Adversarial Testing of RL Agents

Bayesian optimisation iterations
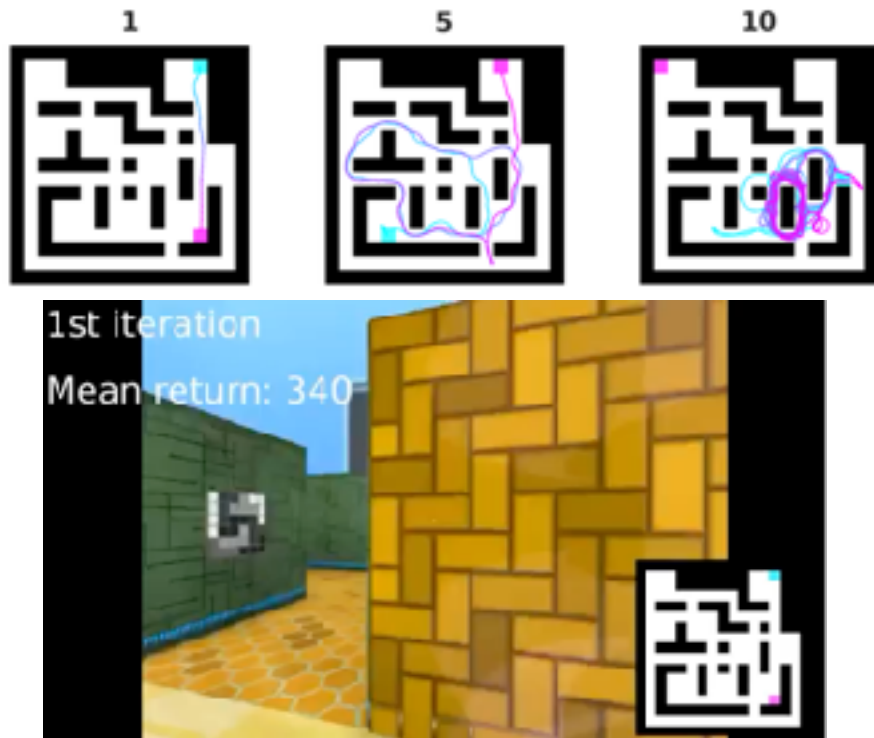
Yee Whye Teh

[Galashov et al 2019]

Meta-Learning: an idiosyncratic tutorial

- Optimisation perspective on meta-learning
  - Optimisation-based meta-learning
  - Black-box meta-learning

- Probabilistic perspective on meta-learning
  - Stochastic processes
  - Neural processes
  - Uncertainty in meta-learning

- **Probabilistic symmetries and neural architectures**

- Note: no meta reinforcement learning (meta-RL)

Yee Whye Teh

Permutation-Invariance in Neural Processes
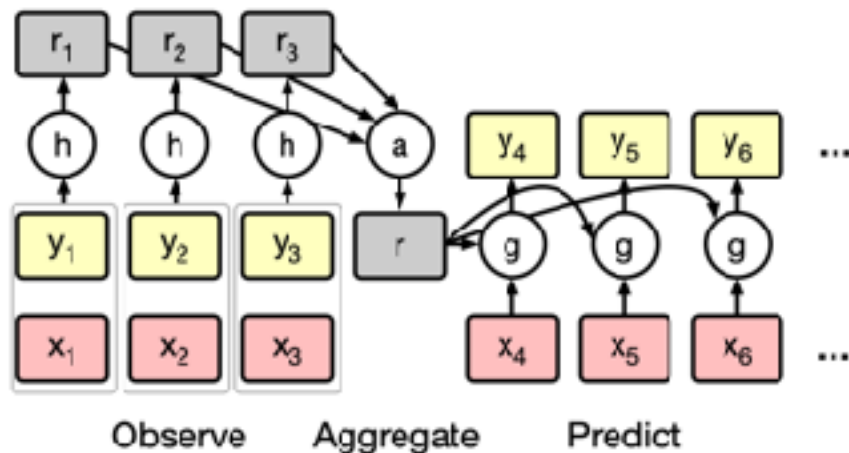
- Embed input/output pairs

$$(x_i, y_i) \mapsto f_\eta(x_i, y_i)$$

- Aggregate embeddings

$$\theta = \frac{1}{n} \sum_{i=1}^{n} f_\eta(x_i, y_i)$$

- Predict

$$p_\eta(y_{n+j} | x_{n+j}, (x_i, y_i)_{i=1}^{n}) = \mathcal{N}(y_{n+j}; g_\eta(\theta, x_{n+j}))$$

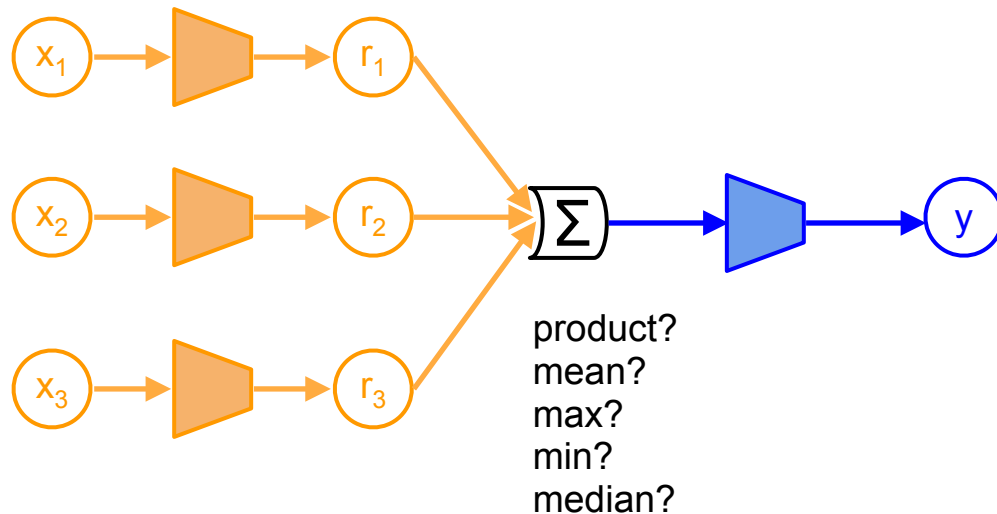Yee Whye Teh

Characterising Permutation-Invariant Functions

- Function $h : \mathcal{X}^n \to \mathcal{Y}$ is permutation-invariant,

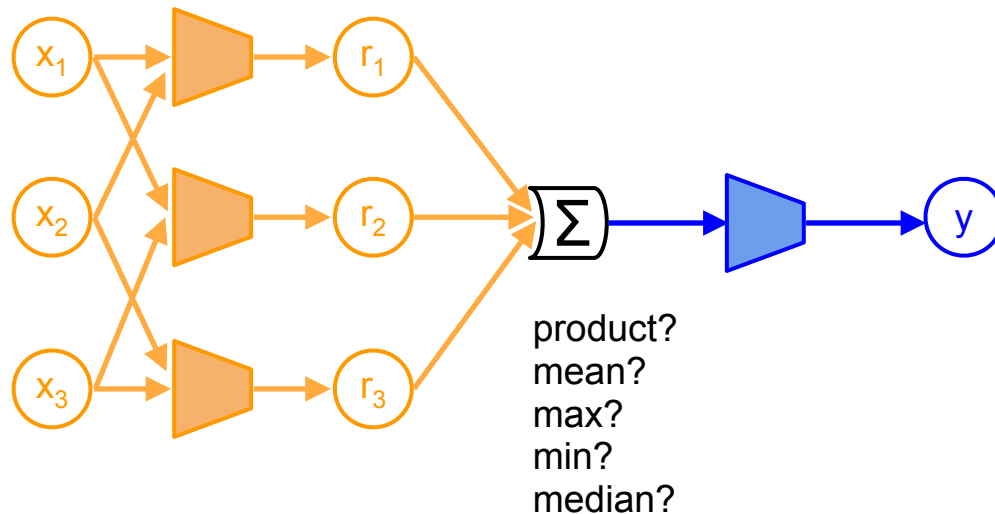$$h(\pi \cdot (x_1, \ldots, x_n)) = h(x_{\pi(1)}, \ldots, x_{\pi(n)})) = h(x_1, \ldots, x_n)$$

- Can we characterise the class of permutation-invariant functions?

- If we use neural networks to parameterise permutation-invariant functions, how should we choose the architecture?

- Given an architecture choice, can the neural network approximate well any arbitrary permutation-invariant function?

Yee Whye Teh

[Zaheer et al NeurIPS 2017, Bloem-Reddy & Teh JMLR 2020]

Characterising Permutation-Invariant Functions

product?
mean?
max?
min?
median?

Characterising Permutation-Invariant Functions

product?
mean?
max?
min?
median?

Yee Whye Teh

Functional Symmetry Properties

- Function $h : \mathcal{X}^n \rightarrow \mathcal{Y}^n$ is **permutation-equivariant**,

$$h(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$$

$$h(\pi \cdot (x_1, \ldots, x_n)) = \pi \cdot (y_1, \ldots, y_n) = \pi \cdot h(x_1, \ldots, x_n)$$

- **Group** G acting on input space $\mathcal{X}$ and output space $\mathcal{Y}$.
  - G**-invariant**:

$$h(g \cdot x) = h(x)$$

  - G**-equivariant**:

$$h(g \cdot x) = g \cdot h(x)$$

Yee Whye Teh

Probabilistic Symmetries

- A distribution P for a random sequence $\mathbf{X}_n = (X_1, \ldots, X_n)$ is **exchangeable** if

$$P(X_1, \ldots, X_n) = P(\pi \cdot (X_1, \ldots, X_n))$$

$$P(X_1 \in B_1, \ldots, X_n \in B_n) = P(X_{\pi(1)} \in B_1, \ldots, X_{\pi(n)} \in B_n))$$

- Exchangeability is permutation-invariance of P.

- $\mathbf{X}_\mathbb{N}$ is infinitely exchangeable if all length n prefixes are exchangeable.
- **de Finetti's Theorem**:

  $\mathbf{X}_\mathbb{N}$ is infinitely exchangeable $\Leftrightarrow X_i \mid Q \sim_{iid} Q$ for some random Q.

Yee Whye Teh

Probabilistic Symmetries for Conditional Distributions

- A conditional distribution $P(Y|X)$ is a stochastic relaxation for a function $Y = h(X)$.

- $P(Y|X)$ is G-invariant if:

$$P(Y|X) = P(Y|g \cdot X)$$

$$P(Y \in B | X \in A) = P(Y \in B | g \cdot X \in A)$$

- $P(Y|X)$ is G-equivariant if:

$$P(Y|X) = P(g \cdot Y | g \cdot X)$$

- Can we characterise the class of permutation-invariant conditional distributions?

Yee Whye Teh

Empirical Measure

- de Finetti's Theorem may fail for finitely exchangeable sequences.

- The **empirical measure** of $\mathbf{X}_n$ is

$$\mathbb{M}_{\mathbf{X}_n}(\cdot) = \sum_{i=1}^{n} \delta_{X_i}(\cdot)$$

- The empirical measure is a **sufficient statistic**: P is exchangeable iff

$$P(\mathbf{X}_n \in \cdot \mid \mathbb{M}_{\mathbf{X}_n} = m) = \mathbb{U}_m(\cdot)$$

where $\mathbb{U}_m$ is the uniform distribution over all sequences $(x_1, \ldots, x_n)$ with empirical measure m.

Noise Outsourcing

- If X and Y are random variables in "nice" (e.g. Borel) spaces $\mathcal{X}$ and $\mathcal{Y}$, then there are a random variable $\eta \sim U[0,1]$ with $\eta \perp\!\!\!\perp X$ and a function $h : [0,1] \times \mathcal{X} \mapsto \mathcal{Y}$ such that

$$(X, Y) =_{a.s.} (X, h(\eta, X))$$

- Furthermore, if there is an **adequate statistic** $S(X)$ with $X \perp\!\!\!\perp Y \mid S(X)$, then

$$(X, Y) =_{a.s.} (X, h(\eta, S(X)))$$

Probabilistic Permutation-Invariance

- Now suppose we have random variables $\mathbf{X}_n$ and Y.
  - Y is conditionally permutation-invariant given $\mathbf{X}_n$.
  - $\mathbf{X}_n$ is marginally permutation-invariant (exchangeable).
- The empirical measure is a sufficient statistic for $\mathbf{X}_n$.

- It is also an adequate statistic for Y given $\mathbf{X}_n$:

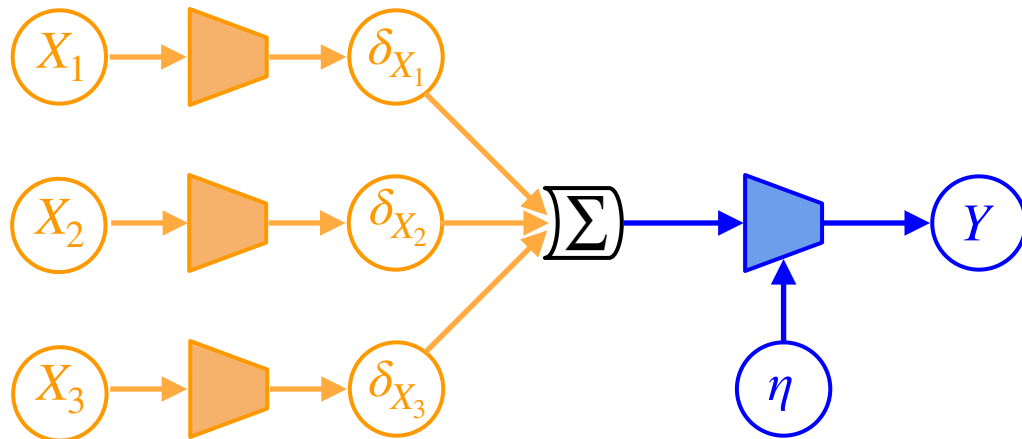$$P(Y \mid \mathbf{X}_n = \mathbf{x}_n) = P(Y \mid \mathbb{M}_{\mathbf{X}_n} = \mathbb{M}_{\mathbf{x}_n})$$

We have the conditional independence $\mathbf{X}_n \perp\!\!\!\perp Y \mid \mathbb{M}_{\mathbf{X}_n}$.
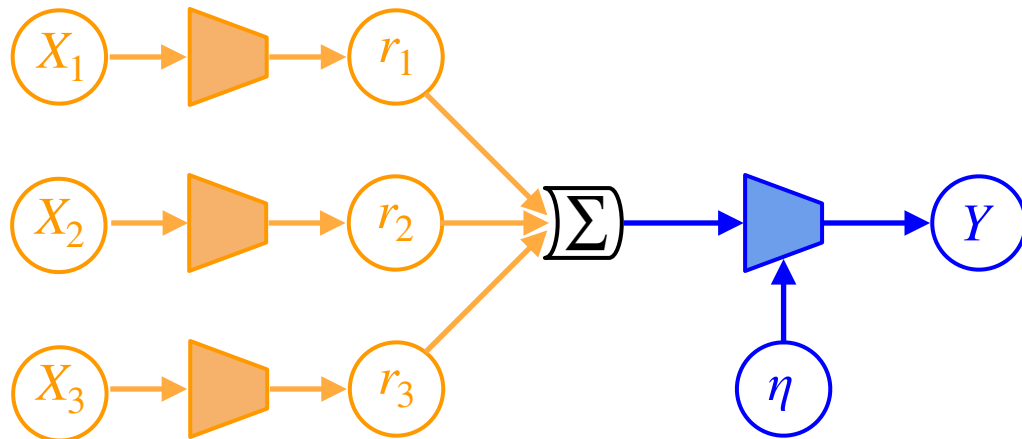
- Noise outsourcing…

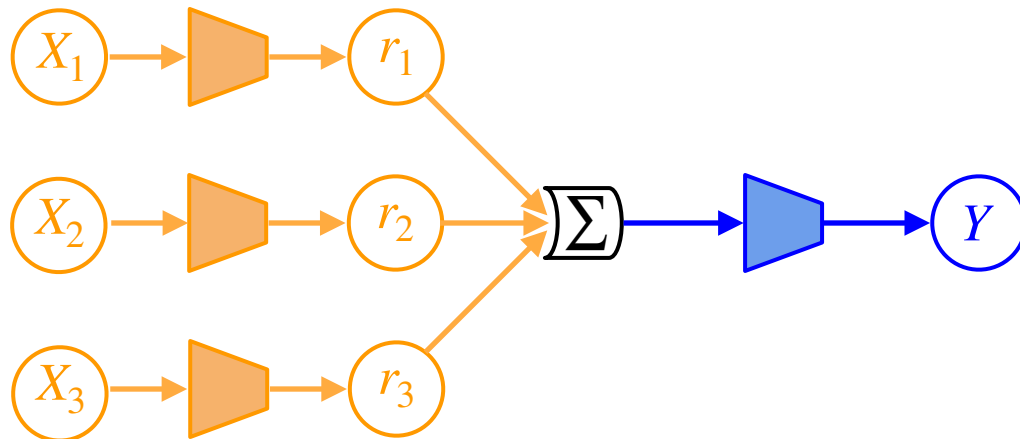$$(\mathbf{X}_n, Y) =_{a.s.} (\mathbf{X}_n, h(\eta, \mathbb{M}_{\mathbf{X}_n}))$$

Yee Whye Teh

Probabilistic Permutation-Invariance

Probabilistic Permutation-Invariance

Functional Permutation-Invariance

Probabilistic Permutation-Equivariance

- Now suppose we have random sequences $\mathbf{X}_n$ and $\mathbf{Y}_n$.
  - $\mathbf{Y}_n$ is conditionally permutation-equivariant given $\mathbf{X}_n$.
  - $\mathbf{X}_n$ is marginally permutation-invariant (exchangeable).

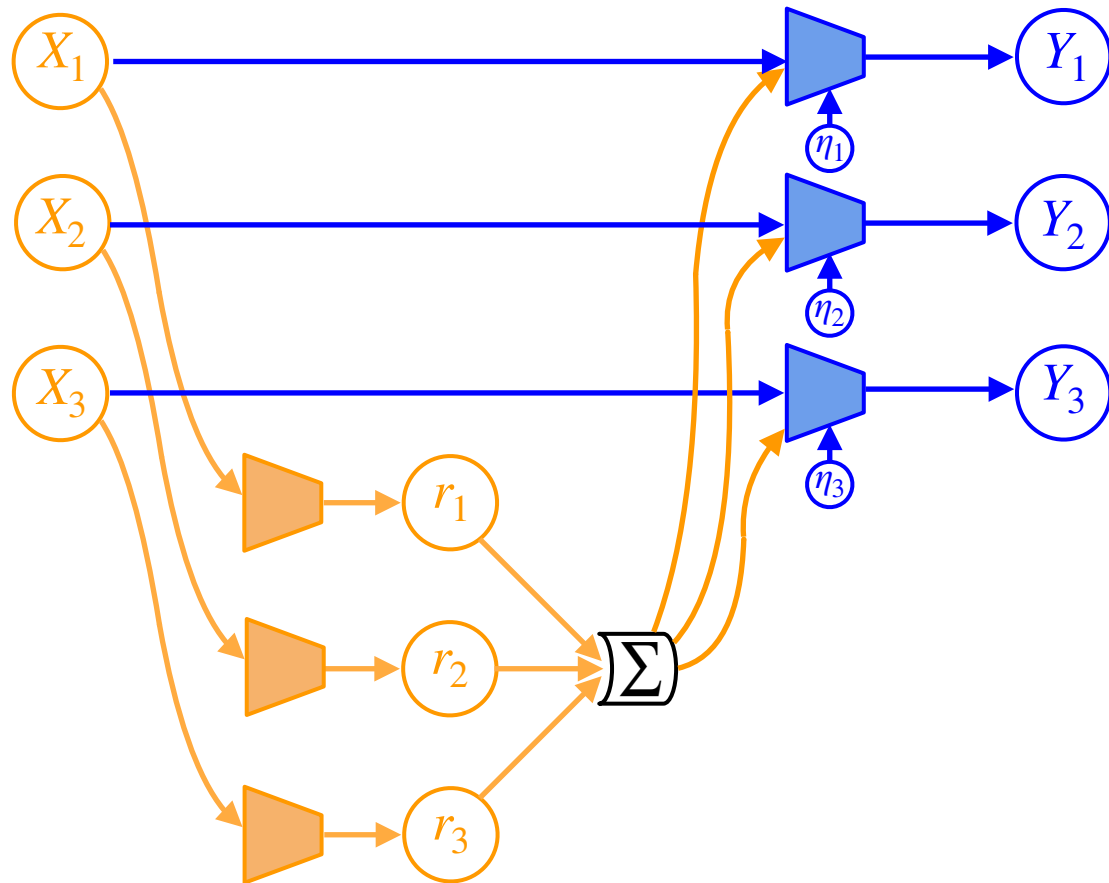- Also suppose that $Y_i \perp\!\!\!\perp \mathbf{Y}_n \backslash Y_i \,|\, \mathbf{X}_n$ for each i.

- Then:
$$(\mathbf{X}_n, (Y_1, \ldots, Y_n)) =_{a.s.} (\mathbf{X}_n, (h(\eta_1, X_1, \mathbb{M}_{\mathbf{X}_n}), \ldots, h(\eta_n, X_n, \mathbb{M}_{\mathbf{X}_n}))$$
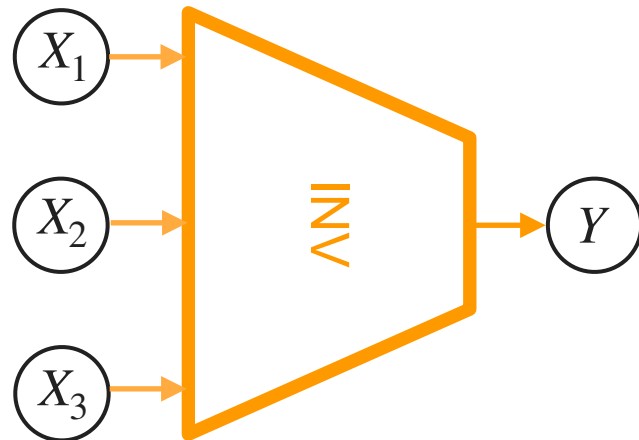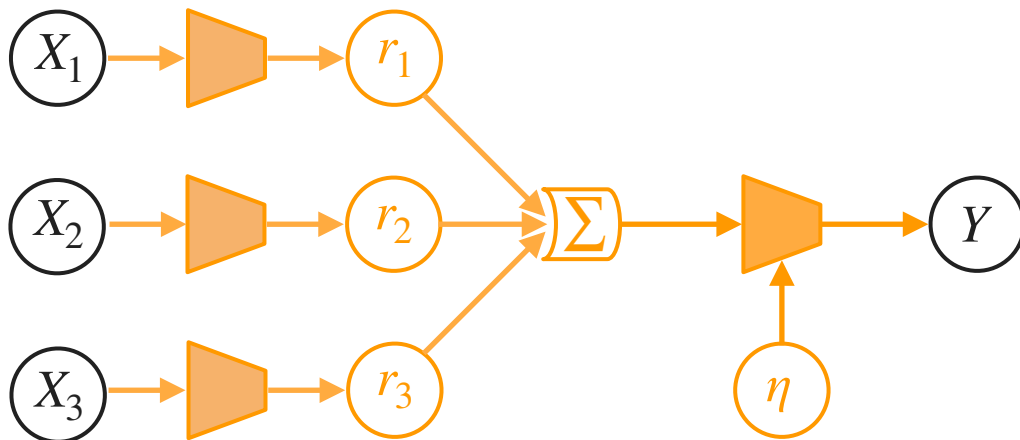
for outsourced noise (η_i) that are identical, mutually independent and independent of $\mathbf{X}_n$.
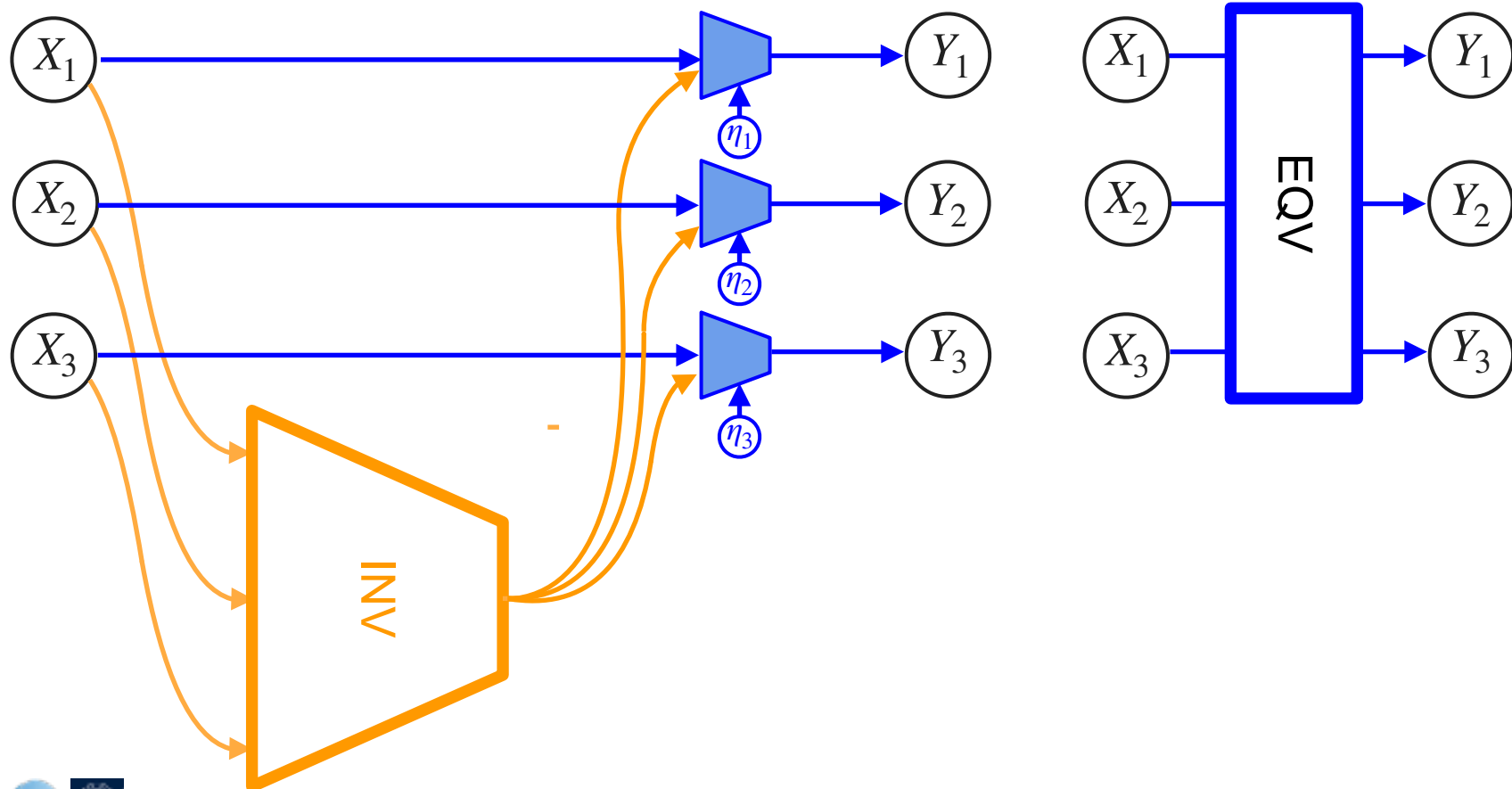
Yee Whye Teh
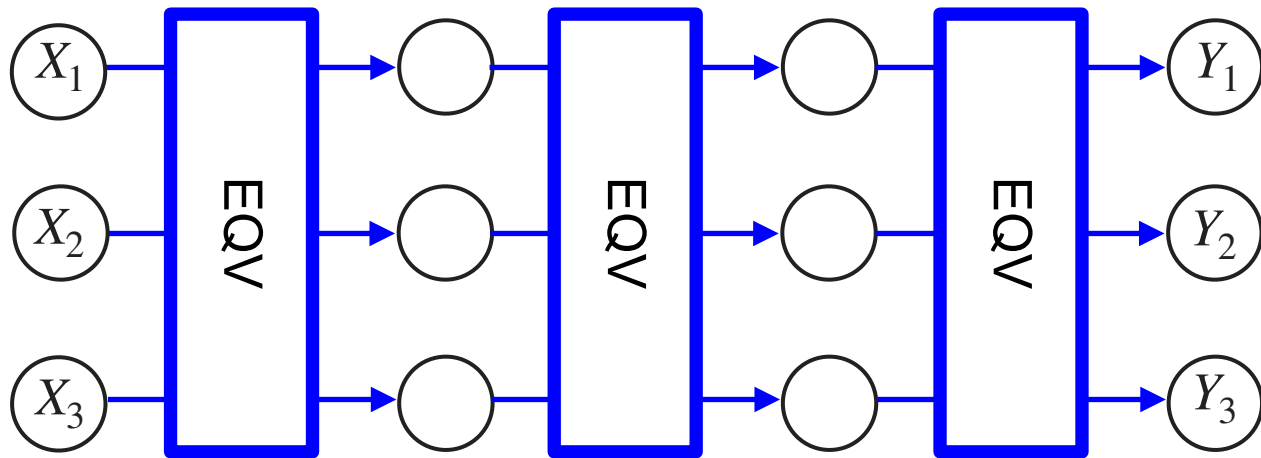
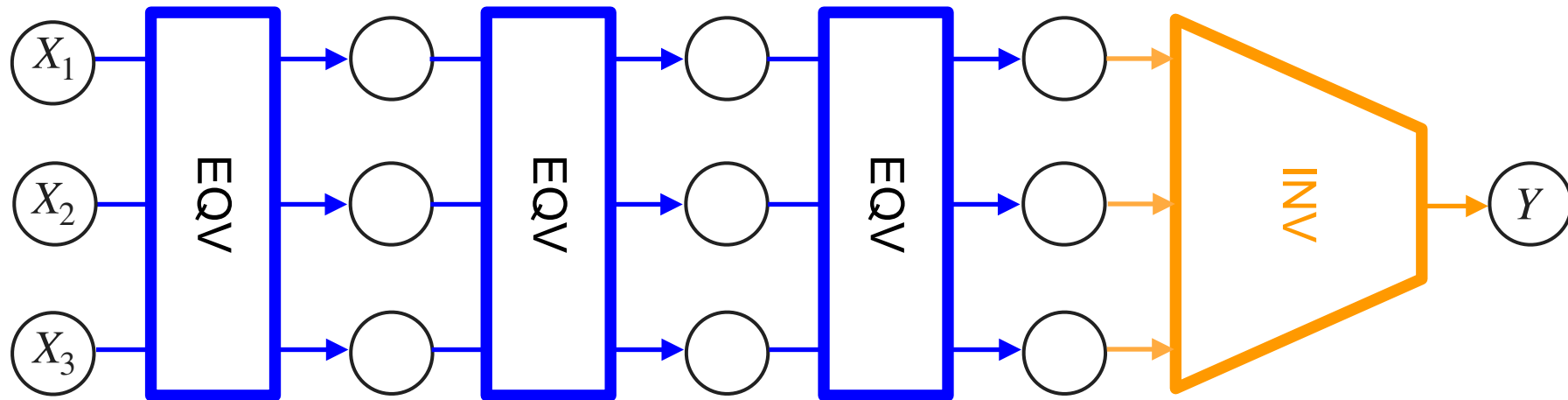# Probabilistic Permutation-Equivariance

Composing Invariant and Equivariant Modules

Composing Invariant and Equivariant Modules

Yee Whye Teh
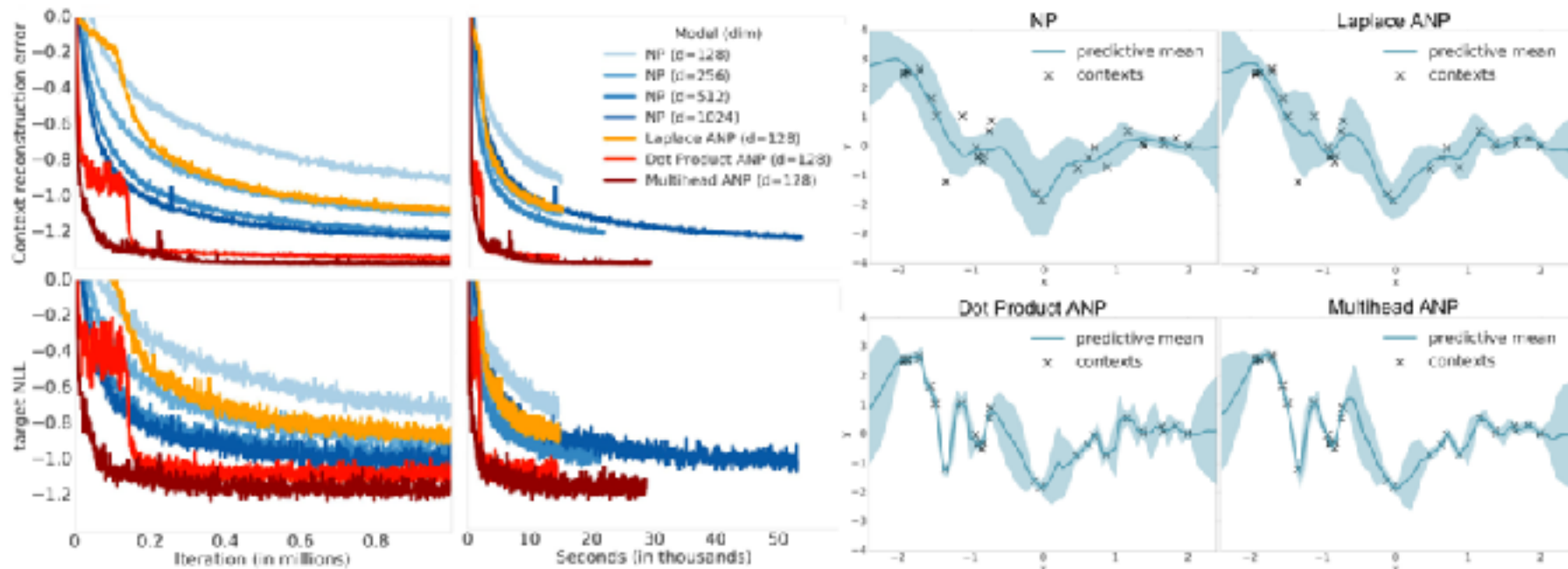
Composing Invariant and Equivariant Modules

Composing Invariant and Equivariant Modules

Yee Whye Teh

Attentive Neural Processes

Yee Whye Teh

[Kim et al ICLR 2019, Lee et al ICML 2019, Xu et al ICML 2020]

Maximal Invariant and Maximal Equivariant

- Let G be a compact group.

- A maximal invariant is a statistic $M : \mathcal{X} \mapsto \mathcal{S}$ such that

$$M(g \cdot x) = M(x) \, \forall g \in G, x \in \mathcal{X}$$

$$M(x_1) = M(x_2) \Rightarrow \exists g \in G : x_1 = g \cdot x_2$$

- A maximal equivariant $\tau : \mathcal{X} \mapsto G$ satisfies

$$\tau(g \cdot x) = g \cdot \tau(x)$$

Yee Whye Teh

Probabilistic and Functional Symmetries

- Let G be a compact group and X be marginally G-invariant.

- Let M be a maximal invariant, then

  Y is conditionally G-invariant given $X \Leftrightarrow (X, Y) =_{a.s.} (X, h(\eta, M(X)))$

  for outsourced noise η independent of X and a function h.

- If a maximal equivariant τ exists and $G_X \subset G_Y$ a.s., then

  Y is conditionally G-equivariant given $X \Leftrightarrow (X, Y) =_{a.s.} (X, h(\eta, X))$

  for outsourced noise η independent of X and a function h that is G-equivariant in its second argument.

Yee Whye Teh

Probabilistic Symmetries

- Tools from probabilistic symmetry, sufficiency and adequacy allowed us to answer questions about neural architectures under symmetry.
  - Framework extends to graph and array structured data with node exchangeability.
  - Continuous groups?
  - How to relax assumptions of conditional independence of outputs?

Meta-Learning: an idiosyncratic tutorial

- Optimisation perspective on meta-learning
  - Optimisation-based meta-learning
  - Black-box meta-learning

- Probabilistic perspective on meta-learning
  - Stochastic processes
  - Neural processes
  - Uncertainty in meta-learning

- Probabilistic symmetries and neural architectures

- Note: no meta reinforcement learning (meta-RL)

Yee Whye Teh

Some Recent Developments

- Recent Developments:
  - [Learning to Continually Learn]
  - [MetaFun: Meta-Learning with Iterative Functional Updates]
  - [Meta-Gradient Reinforcement Learning]
  - [Modular Meta-Learning with Shrinkage]
  - [Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML]
  - [Meta-Learning with Warped Gradient Descent]
- Meta-learning for continual learning
  - [Learning to Continually Learn]
  - [Meta-Learning Representations for Continual Learning]
  - [Task Agnostic Continual Learning via Meta Learning]
- Meta-learning theory
  - [Theoretical Models of Learning to Learn]
  - [A Model of Inductive Bias Learning]
  - [Provably Meta-Learning of Linear Representations]

Yee Whye Teh

Thank You!

- Collaborators, colleagues, mentors

- You!

http://csml.stats.ox.ac.uk/people/teh

Yee Whye Teh