Bayesian Meta-Learning

CS 330

Course Reminders

Homework 3 due **Monday**.

Tutorial session **tomorrow 4:30 pm**

First guest lecture next Weds!

James Harrison on learned optimizers
(Google DeepMind)

Following up on some high-res feedback:

- I'll work on managing questions, repeating question when needed.

Recap: Amortized Variational Inference

A. Formulate **a lower bound** on the log likelihood objective.
B. Check **how tight** the bound is.
C. Variational inference -> *Amortized* variational inference
D. How to **optimize**

Recap: Amortized Variational Inference

A. Formulate **a lower bound** on the log likelihood objective.

$$\log p(x_i) \geq E_{z \sim q_i(z)}[\log p_\theta(x_i|z) + \log p(z)] + \mathcal{H}(q_i)$$

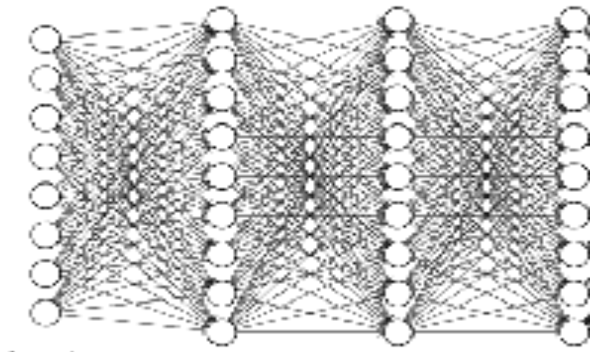
B. Check **how tight** the bound is.

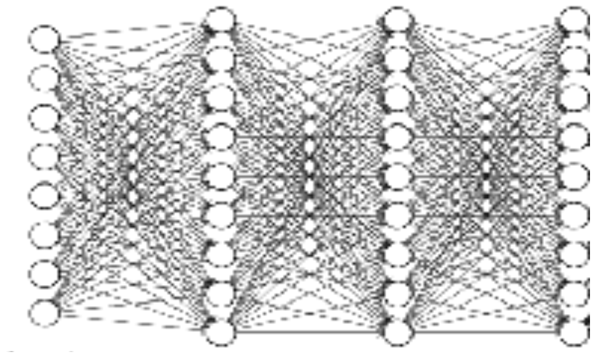
tight when $D_{\mathrm{KL}}(q_i(z) \| p(z|x_i))$ is 0

C. Variational inference -> *Amortized* variational inference

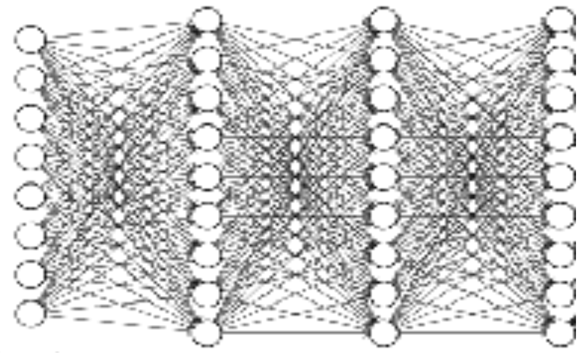
$$\text{what if we learn a } \textit{network } q_i(z) = q(z|x_i) \approx p(z|x_i)?$$

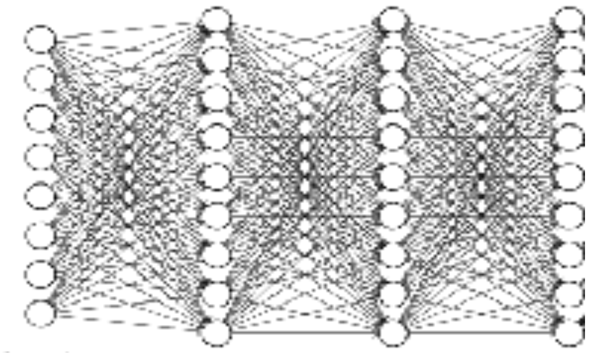
D. How to **optimize**

z  $p_\theta(x|z)$

x  $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$

Amortized variational inference

z  $p_\theta(x|z)$

x  $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$

$$\log p(x_i) \geq \overbrace{E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + \mathcal{H}(q_\phi(z|x_i))}^{\mathcal{L}(p_\theta(x_i|z),\, q_\phi(z|x_i))}$$

for each x_i (or mini-batch):

calculate $\nabla_\theta \mathcal{L}(p_\theta(x_i|z), q_\phi(z|x_i))$:

sample $z \sim q_\phi(z|x_i)$

$\nabla_\theta \mathcal{L} \approx \nabla_\theta \log p_\theta(x_i|z)$

$\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$

$\phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L}$

how do we calculate this?

Amortized variational inference

for each x_i (or mini-batch):

calculate $\nabla_\theta \mathcal{L}(p_\theta(x_i|z), q_\phi(z|x_i))$:

sample $z \sim q_\phi(z|x_i)$

$\nabla_\theta \mathcal{L} \approx \nabla_\theta \log p_\theta(x_i|z)$

$\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$

$\phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L}$

$$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$$

look up formula for entropy of a Gaussian

$$\mathcal{L}_i = \underbrace{E_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)]}_{J(\phi) = E_{z\sim q_\phi(z|x_i)}[r(x_i, z)]} + \mathcal{H}(q_\phi(z|x_i))$$

The reparameterization trick

$$J(\phi) = E_{z\sim q_\phi(z|x_i)}[r(x_i, z)]$$

$$= E_{\epsilon\sim\mathcal{N}(0,1)}[r(x_i, \mu_\phi(x_i) + \epsilon\sigma_\phi(x_i))]$$

$$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$$

$$z = \mu_\phi(x) + \epsilon\sigma_\phi(x)$$

estimating $\nabla_\phi J(\phi)$:

sample $\epsilon_1, \ldots, \epsilon_M$ from $\mathcal{N}(0,1)$ (a single sample works well!)

$$\nabla_\phi J(\phi) \approx \frac{1}{M}\sum_j \nabla_\phi r(x_i, \mu_\phi(x_i) + \epsilon_j\sigma_\phi(x_i))$$

$\epsilon \sim \mathcal{N}(0,1)$

independent of ϕ!

+ Very simple to implement
+ Low variance
- Only continuous latent variables

Discrete latent variables:

- vector quantization & straight-through estimator (“VQ-VAE”)
- policy gradients / “REINFORCE”

Another way to look at everything…

$$\mathcal{L}_i = E_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + \mathcal{H}(q_\phi(z|x_i))$$

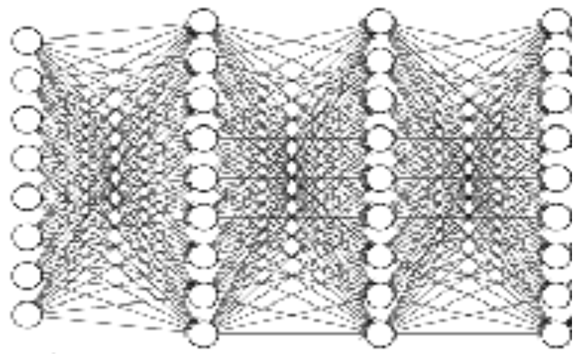
$$= E_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] + \underbrace{E_{z\sim q_\phi(z|x_i)}[\log p(z)] + \mathcal{H}(q_\phi(z|x_i))}_{-D_{\mathrm{KL}}(q_\phi(z|x_i)\|p(z))}$$

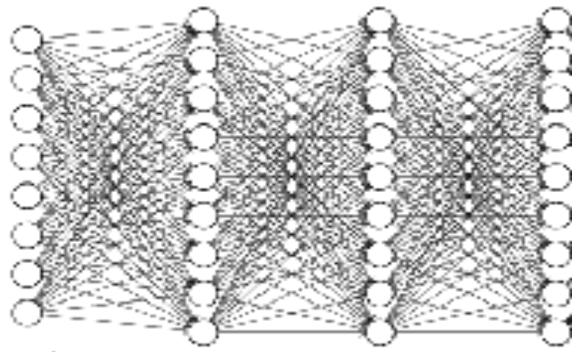
this has a convenient analytical form for Gaussians

$$= E_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{\mathrm{KL}}(q_\phi(z|x_i)\|p(z))$$

$$= E_{\epsilon\sim\mathcal{N}(0,1)}[\log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i))] - D_{\mathrm{KL}}(q_\phi(z|x_i)\|p(z))$$

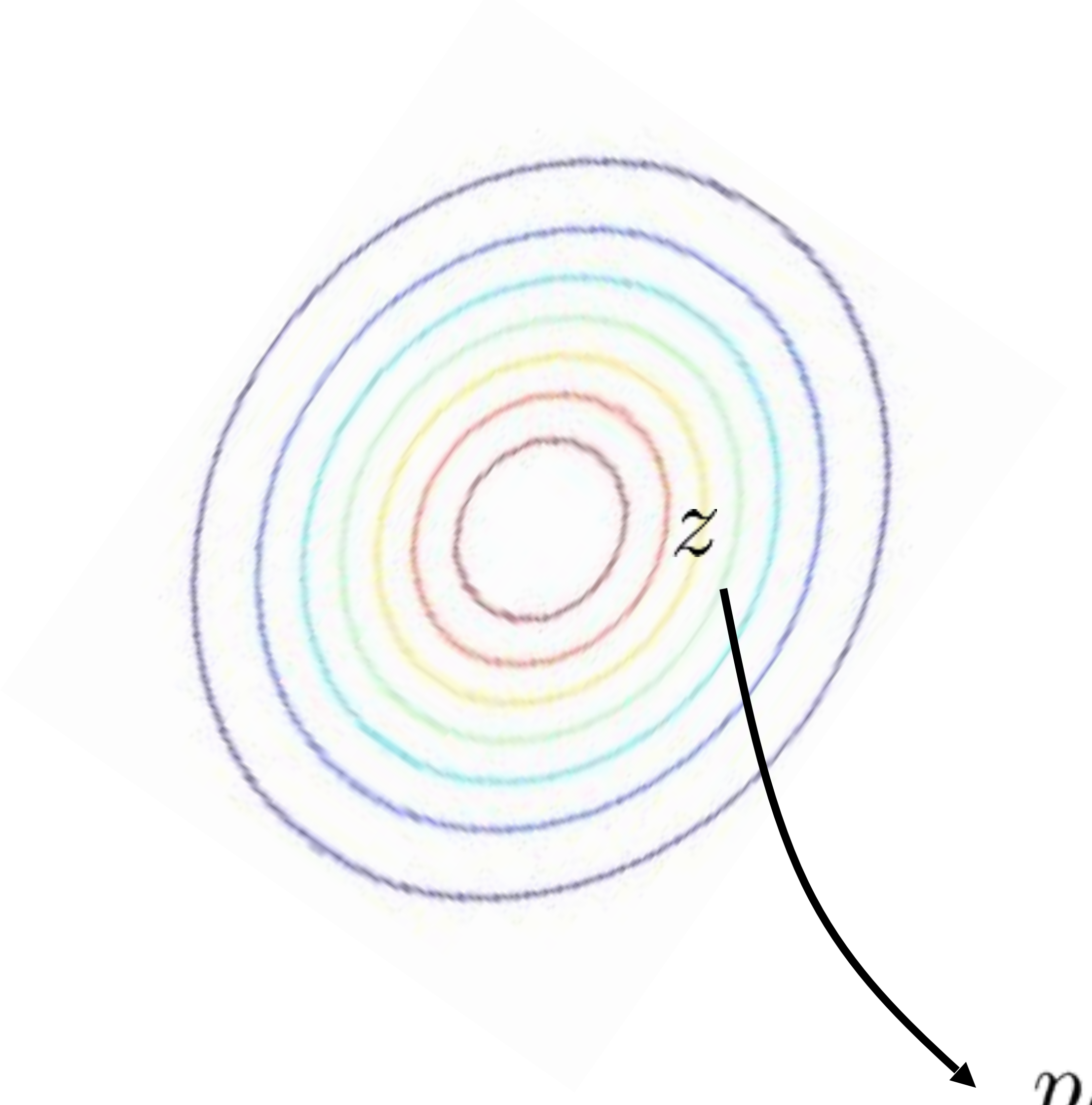
$$\approx \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i)) - D_{\mathrm{KL}}(q_\phi(z|x_i)\|p(z))$$

x_i  ϕ → $\mu_\phi(x_i)$, $\sigma_\phi(x_i)$ → $\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i) = z$ ← $\epsilon \sim \mathcal{N}(0,1)$

 θ → $p_\theta(x_i|z)$

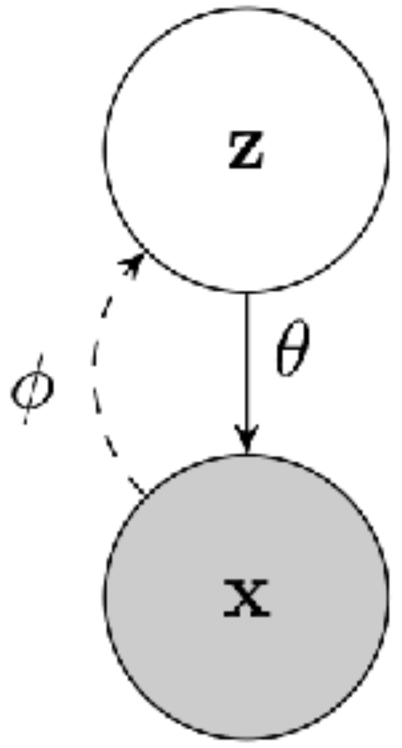
Example Models

The variational autoencoder



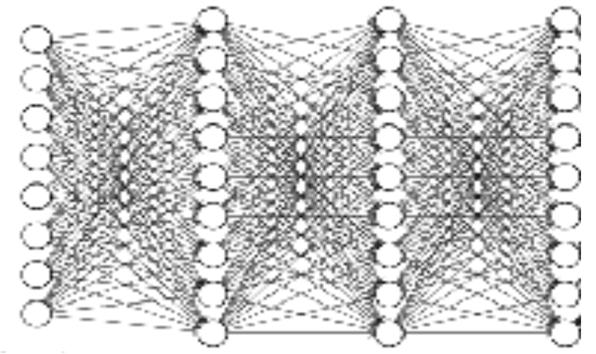
$p_\theta(x|z)$

$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$



$p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z))$

x_i



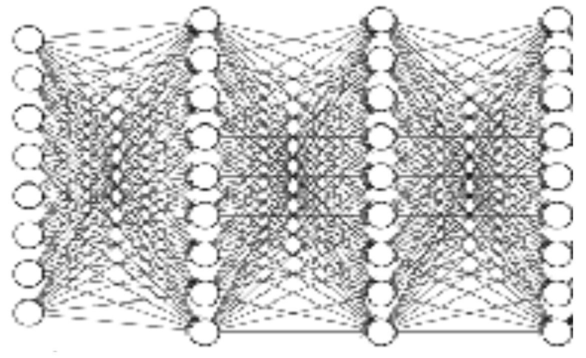
ϕ

$\mu_\phi(x_i)$

$\sigma_\phi(x_i)$

$\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i) = z$

$\epsilon \sim \mathcal{N}(0, 1)$

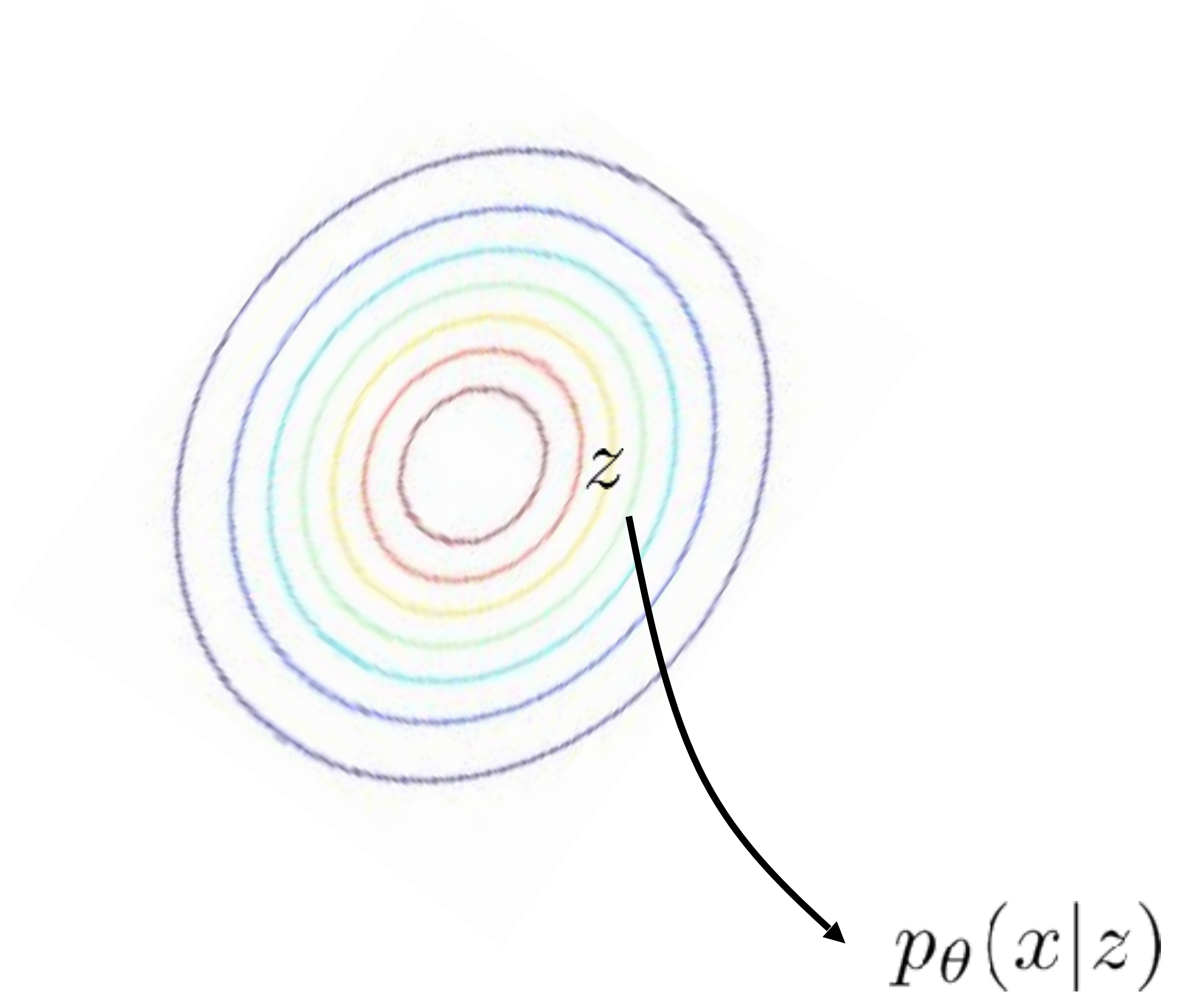


θ

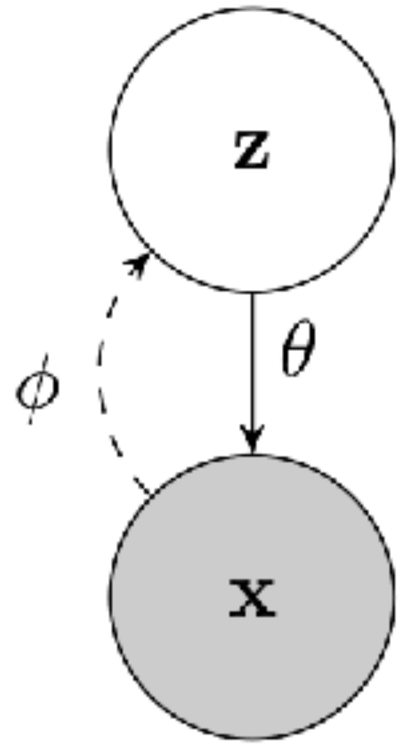
$\longrightarrow p_\theta(x|z)$

$$\max_{\theta,\phi} \frac{1}{N} \sum_i \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i)) - D_{\mathrm{KL}}(q_\phi(z|x_i) \| p(z))$$

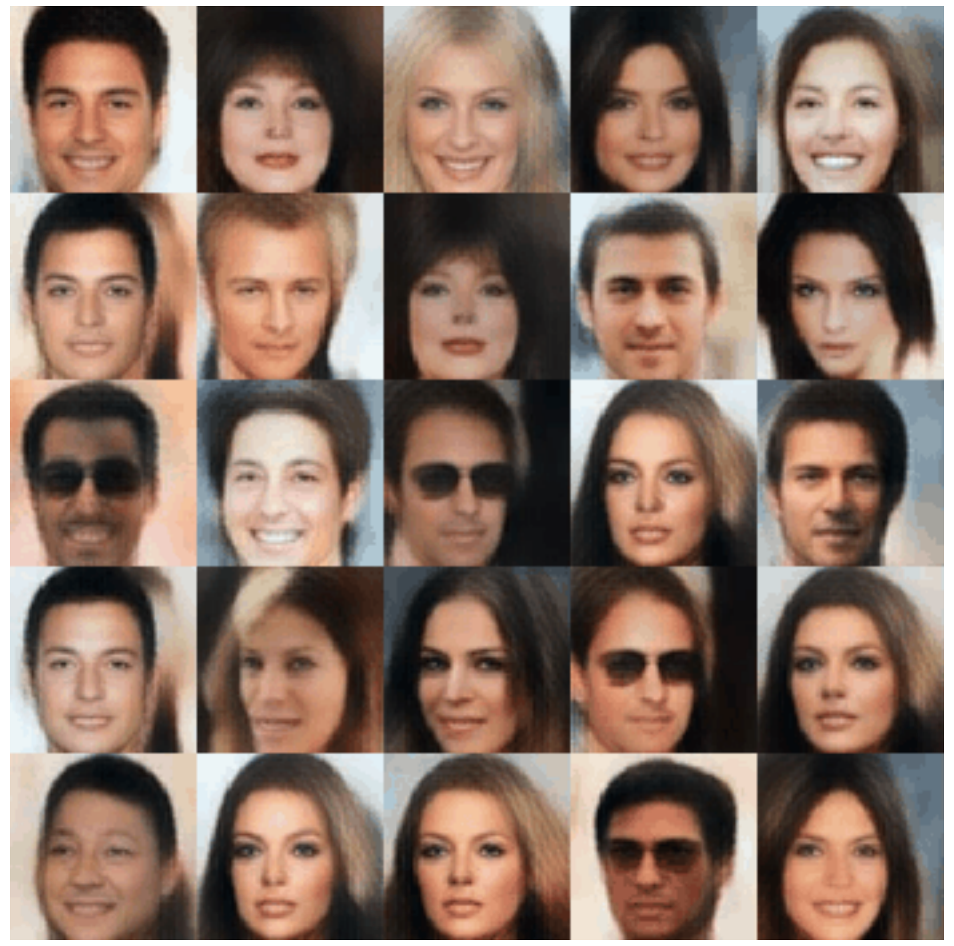
Using the variational autoencoder



$$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$$



$$p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z))$$



$$p(x) = \int p(x|z)p(z)dz$$

why does this work?

sampling:

$$z \sim p(z)$$

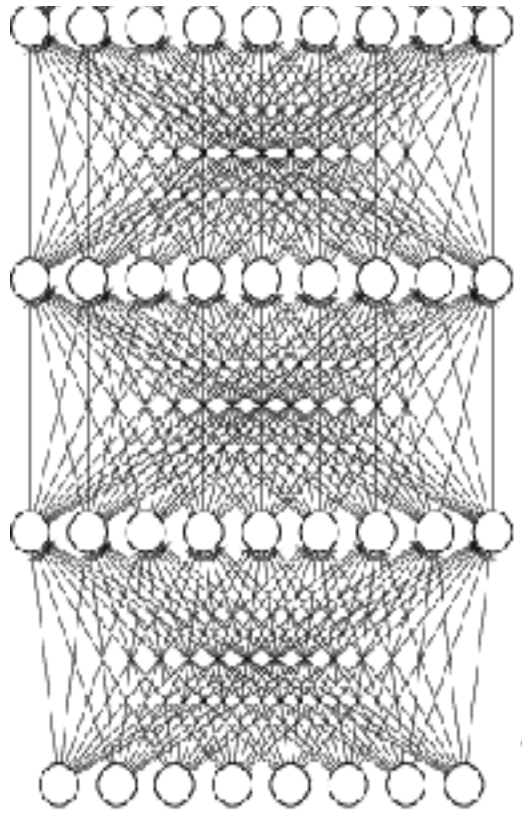
$$x \sim p(x|z)$$

$$\mathcal{L}_i = E_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{\mathrm{KL}}(q_\phi(z|x_i)\|p(z))$$

Conditional models

$$\mathcal{L}_i = E_{z \sim q_\phi(z|x_i, y_i)}[\log p_\theta(y_i|x_i, z) + \log p(z|x_i)] + \mathcal{H}(q_\phi(z|x_i, y_i))$$

just like before, only now generating y_i
and *everything* is conditioned on x_i



$p(y|x, z)$

class 109
(brain coral)

$z \sim \mathcal{N}(0, \mathbf{I})$

$p(z)$

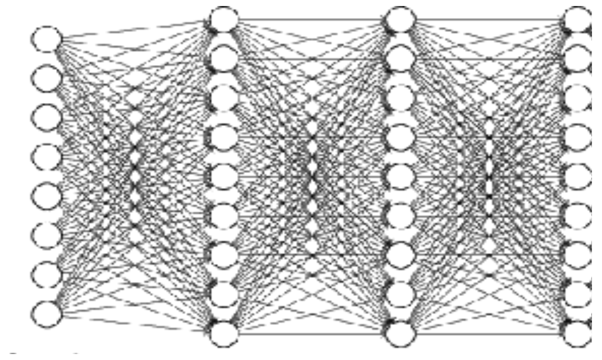
can *optionally* depend on x

at test time:

$z \sim p(z|x_i)$

$y \sim p(y|x_i, z)$

x_i
y_i



ϕ

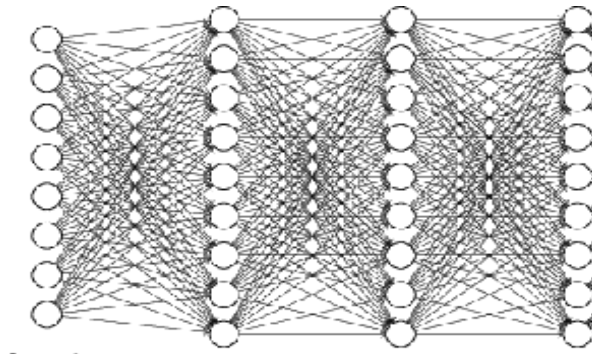
$\mu_\phi(x_i, y_i)$

$\sigma_\phi(x_i, y_i)$

$\mu_\phi + \epsilon\sigma_\phi = z$

$\epsilon \sim \mathcal{N}(0, 1)$

x_i



θ

$\longrightarrow p_\theta(y_i|x_i, z)$

Images from Razavi, van den Oord, Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. ‘19

Plan for (the rest of) today

Why be Bayesian?

Bayesian meta-learning approaches

- black-box approaches
- optimization-based approaches (time permitting)

How to evaluate Bayesian meta-learners.

Goals for by the end of lecture:

- Understand the interpretation of meta-learning as Bayesian inference
- Understand techniques for representing uncertainty over parameters, predictions

Recap: Properties of Meta-Learning Inner Loops

***Algorithmic properties* perspective**

Expressive power

the ability for f to represent a range of learning procedures

Why? scalability, applicability to a range of domains

Consistency

learned learning procedure will solve task with enough data

Why? reduce reliance on meta-training tasks, good OOD task performance

These properties are important for most applications!

Recap: Properties of Meta-Learning Inner Loops

***Algorithmic properties* perspective**

Expressive power

the ability for f to represent a range of learning procedures

Why? scalability, applicability to a range of domains

Consistency

learned learning procedure will solve task with enough data

Why? reduce reliance on meta-training tasks, good OOD task performance

Uncertainty awareness

ability to reason about ambiguity during learning

Why? active learning, calibrated uncertainty, RL principled Bayesian approaches

this lecture

Plan for Today

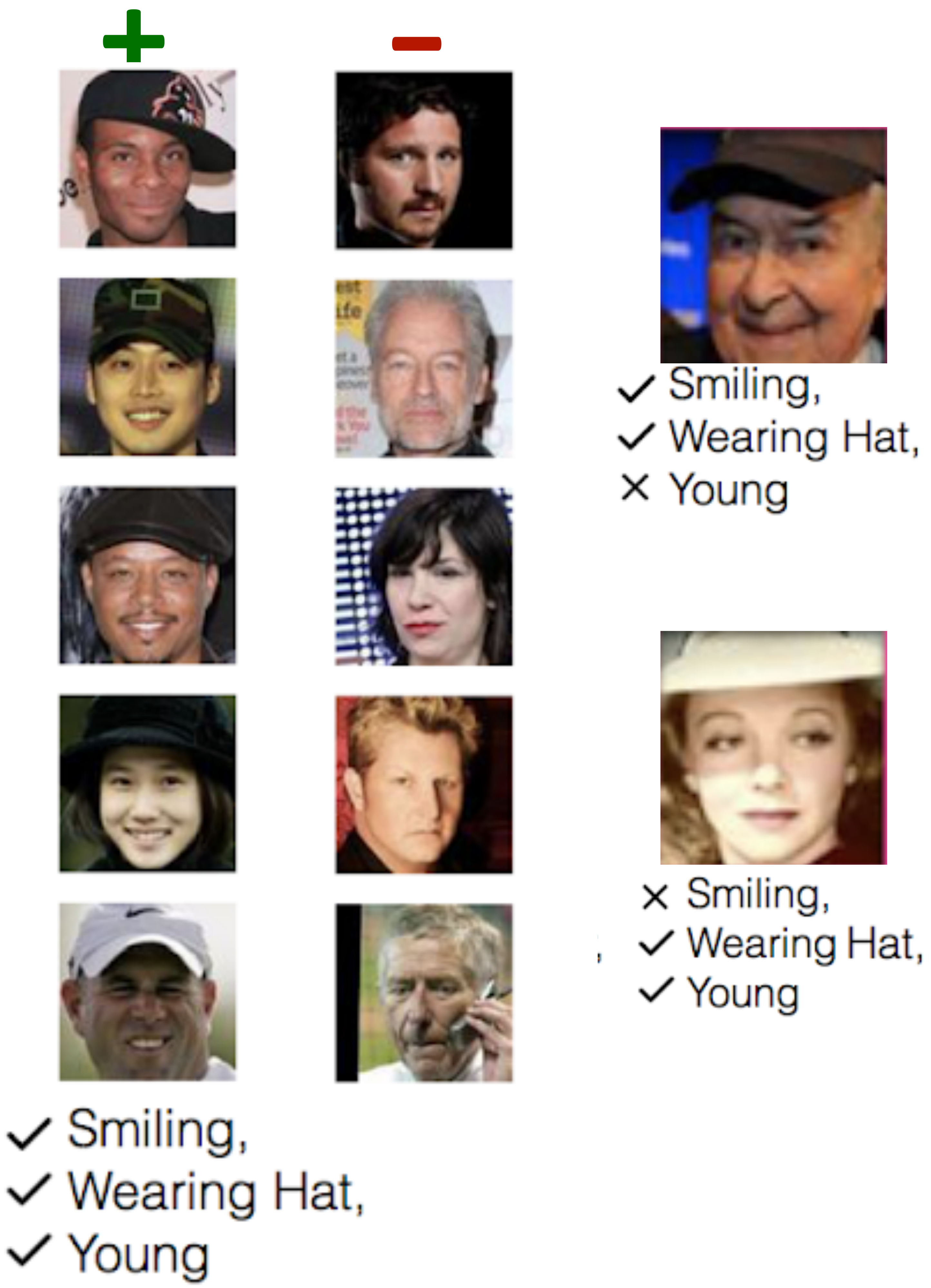
Why be Bayesian?

Bayesian meta-learning approaches

- black-box approaches
- optimization-based approaches (time permitting)

How to evaluate Bayesian meta-learners.

Recall parametric approaches: Use **deterministic** $p(\phi_i|\mathcal{D}_i^{\mathrm{tr}}, \theta)$ (i.e. a point estimate)



Why/when is this a problem?

Few-shot learning problems may be *ambiguous*.
(even with prior)

Can we learn to *generate hypotheses* about the underlying function?
i.e. sample from $p(\phi_i|\mathcal{D}_i^{\mathrm{tr}}, \theta)$

Important for:
- **safety-critical** few-shot learning (e.g. medical imaging)
- learning to **actively learn**
- learning to **explore** in meta-RL

Active learning w/ meta-learning: Woodward & Finn '16, Konyushkova et al. '17, Bachman et al. '17

Plan for Today

Why be Bayesian?

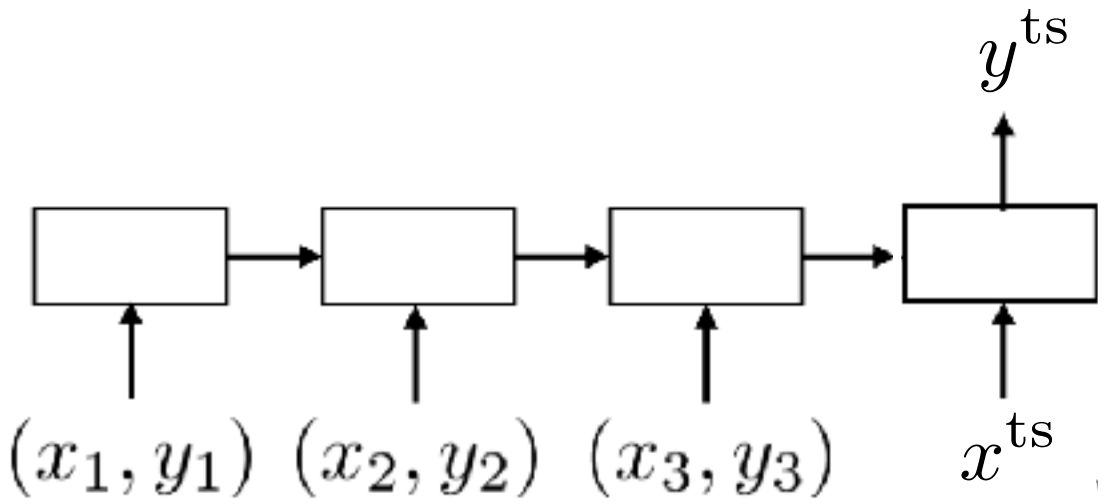
Bayesian meta-learning approaches
- black-box approaches
- optimization-based approaches (time permitting)

How to evaluate Bayesian meta-learners.

Meta-learning algorithms as computation graphs

Black-box

$$y^{\text{ts}} = f_\theta(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}})$$



Optimization-based

$$y^{\text{ts}} = f_{\text{MAML}}(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}})$$
$$= f_{\phi_i}(x^{\text{ts}})$$
$$\text{where } \phi_i = \theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}})$$

Non-parametric

$$y^{\text{ts}} = f_{\text{PN}}(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}})$$
$$= \text{softmax}(-d\left(f_\theta(x^{\text{ts}}), \mathbf{c}_n\right))$$
$$\text{where } \mathbf{c}_n = \frac{1}{K} \sum_{(x,y) \in \mathcal{D}_i^{\text{tr}}} \mathbb{1}(y = n) f_\theta(x)$$

Version 0: Let f output the parameters of a distribution over y^{ts}.

For example:

- probability values of discrete **categorical distribution**
- mean and variance of a **Gaussian**
- means, variances, and mixture weights of a **mixture of Gaussians**
- for multi-dimensional y^{ts}: parameters of a **sequence of distributions** (i.e. autoregressive model)

Then, optimize with maximum likelihood.

Version 0: Let f output the parameters of a distribution over y^{ts}.

For example:

- probability values of discrete **categorical distribution**
- mean and variance of a **Gaussian**
- means, variances, and mixture weights of a **mixture of Gaussians**
- for multi-dimensional y^{ts}: parameters of a **sequence of distributions** (i.e. autoregressive model)

Then, optimize with **maximum likelihood**.

Pros:
+ simple
+ can combine with variety of methods

Cons:
- can't reason about uncertainty over the underlying function [to determine how uncertainty across datapoints relate]
- limited class of distributions over y^{ts} can be expressed
- tends to produce poorly-calibrated uncertainty estimates

Thought exercise #4: Can you do the same maximum likelihood training for ϕ?

The Bayesian Deep Learning Toolbox

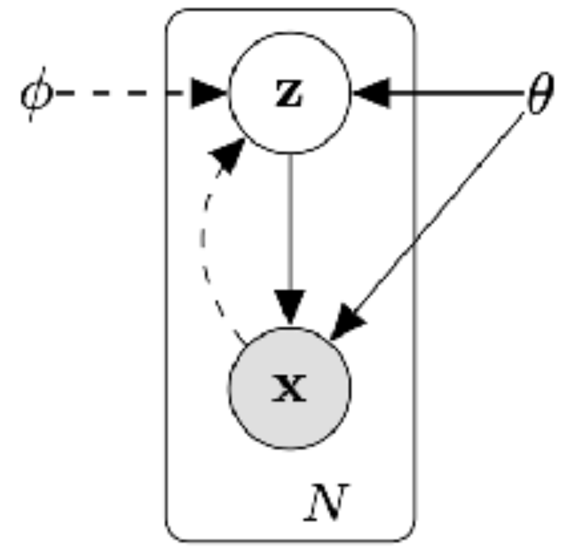
a broad one-slide overview

(CS 236 provides a thorough treatment)

Goal: represent distributions with neural networks

Latent variable models + variational inference (Kingma & Welling '13, Rezende et al. '14):

- approximate likelihood of latent variable model with variational lower bound

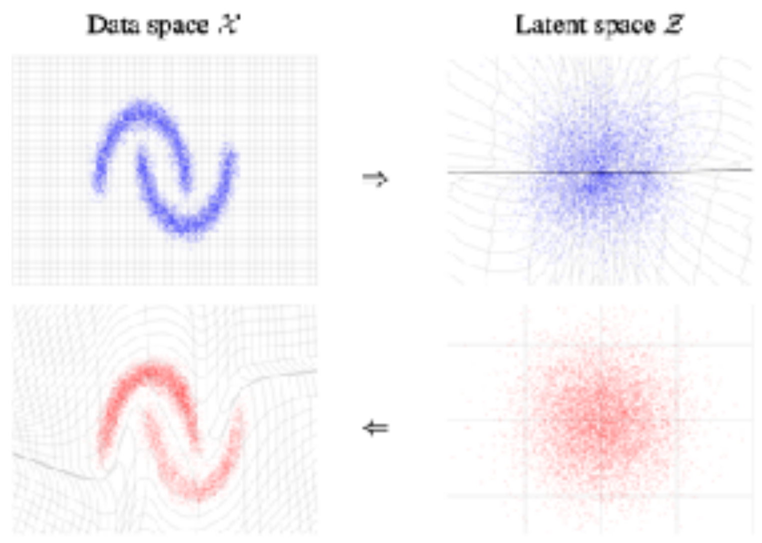


Bayesian ensembles (Lakshminarayanan et al. '17):

- particle-based representation: train separate models on bootstraps of the data

Bayesian neural networks (Blundell et al. '15):

- explicit distribution over the space of network parameters

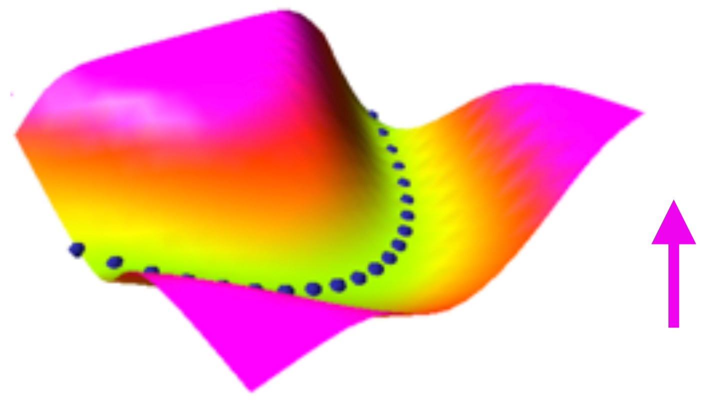


Normalizing Flows (Dinh et al. '16):

- invertible function from latent distribution to data distribution

Energy-based models & GANs (LeCun et al. '06, Goodfellow et al. '14):

- estimate unnormalized density



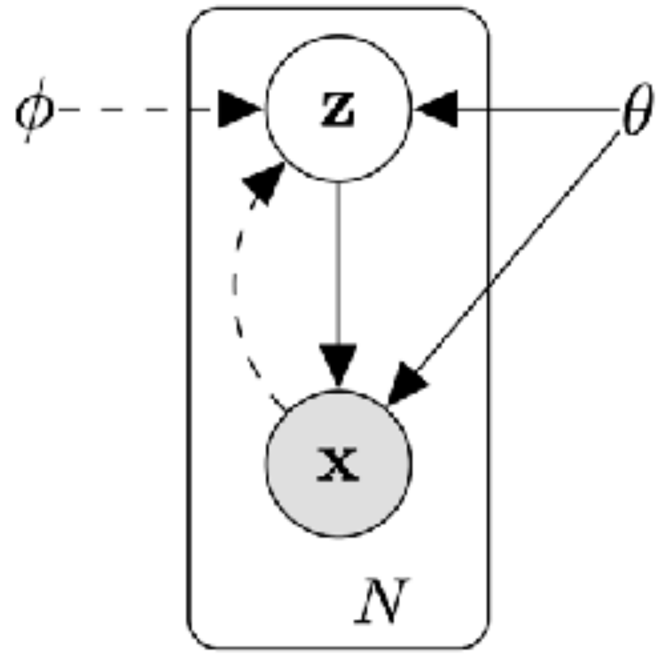
↓ data

↑ everything else

We'll see how we can leverage the first two.

The others could be useful in developing new methods.

Recap: The Variational Lower Bound



Observed variable x, latent variable z

ELBO: $\log p(x) \geq \mathbb{E}_{q(z|x)}\left[\log p(x,z)\right] + \mathscr{H}(q(z|x))$

Can also be written as: $= \mathbb{E}_{q(z|x)}\left[\log p(x|z)\right] - D_{KL}\left(q(z|x) \| p(z)\right)$

p: model
- $p(x|z)$ represented w/ neural net,
- $p(z)$ represented as $\mathcal{N}(\mathbf{0}, \mathbf{I})$

model parameters θ,
variational parameters ϕ

$q(z|x)$: inference network, variational distribution

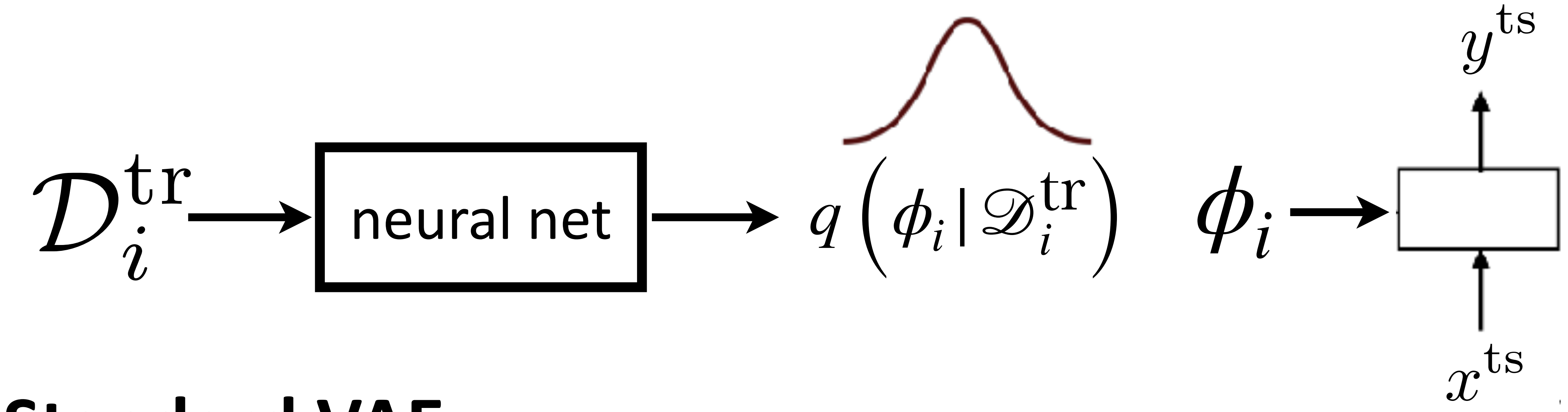
Problem: need to backprop through sampling
i.e. compute derivative of $\mathbb{E}_q$ w.r.t. q

Reparametrization trick For Gaussian $q(z|x)$:
$q(z|x) = \mu_q + \sigma_q \epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Can we use amortized variational inference for meta-learning?

Bayesian black-box meta-learning

with standard, deep variational inference



Standard VAE:

Observed variable x, latent variable z

ELBO: $\mathbb{E}_{q(z|x)}\left[\log p(x \mid z)\right] - D_{KL}\left(q(z \mid x) \| p(z)\right)$

p: model, represented by a neural net

q: inference network, variational distribution

Meta-learning:

Observed variable $\mathcal{D}$, latent variable ϕ

$$\max \mathbb{E}_{q(\phi)}\left[\log p(\mathcal{D} \mid \phi)\right] - D_{KL}\left(q(\phi) \| p(\phi)\right)$$

What should q condition on?

$$\max \mathbb{E}_{q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}\right)}\left[\log p(\mathcal{D} \mid \phi)\right] - D_{KL}\left(q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}\right) \| p(\phi)\right)$$

$$\max \mathbb{E}_{q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}\right)}\left[\log p\left(y^{\mathrm{ts}} \mid x^{\mathrm{ts}}, \phi\right)\right] - D_{KL}\left(q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}\right) \| p(\phi)\right)$$

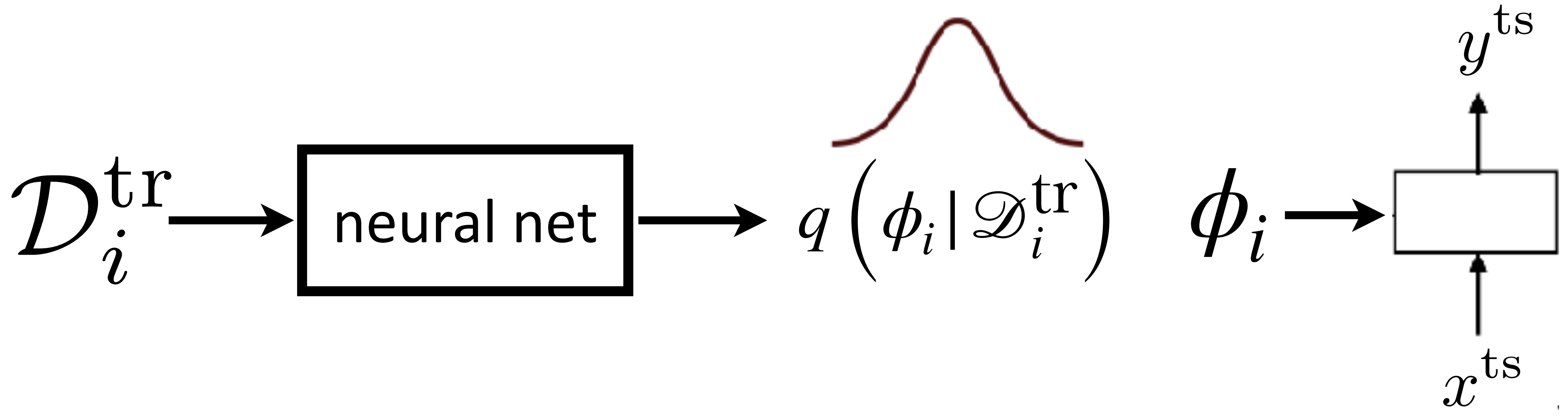
What about the meta-parameters θ?

$$\max_{\theta} \mathbb{E}_{q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}, \theta\right)}\left[\log p\left(y^{\mathrm{ts}} \mid x^{\mathrm{ts}}, \phi\right)\right] - D_{KL}\left(q\left(\phi \mid \mathcal{D}^{\mathrm{tr}}, \theta\right) \| p(\phi \mid \theta)\right)$$

Can also condition on θ here

Final objective (for completeness): $\max_{\theta} \mathbb{E}_{\mathcal{T}_i}\left[\mathbb{E}_{q\left(\phi_i \mid \mathcal{D}_i^{\mathrm{tr}}, \theta\right)}\left[\log p\left(y_i^{\mathrm{ts}} \mid x_i^{\mathrm{ts}}, \phi_i\right)\right] - D_{KL}\left(q\left(\phi_i \mid \mathcal{D}_i^{\mathrm{tr}}, \theta\right) \| p(\phi_i \mid \theta)\right)\right]$

Bayesian black-box meta-learning
with standard, deep variational inference



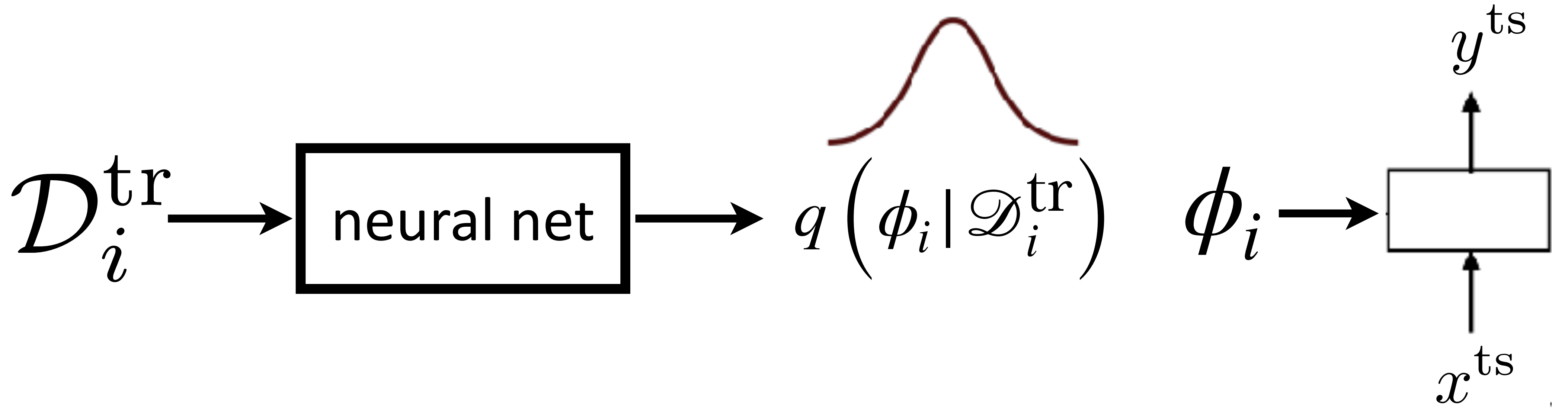
Meta-learning: Observed variable $\mathscr{D}$, latent variable ϕ

Final objective (for completeness): $\max_\theta \mathbb{E}_{\mathcal{T}_i}\left[\mathbb{E}_{q\left(\phi_i \mid \mathscr{D}_i^{\mathrm{tr}}, \theta\right)}\left[\log p\left(y_i^{\mathrm{ts}} \mid x_i^{\mathrm{ts}}, \phi_i\right)\right] - D_{KL}\left(q\left(\phi_i \mid \mathscr{D}_i^{\mathrm{tr}}, \theta\right) \| p(\phi_i \mid \theta)\right)\right]$

Question: Can you get non-Gaussian distributions over ϕ_i with this approach?

Bayesian black-box meta-learning

with standard, deep variational inference



$$\max_\theta \mathbb{E}_{\mathscr{T}_i}\left[\mathbb{E}_{q\left(\phi_i \mid \mathscr{D}_i^{\mathrm{tr}}, \theta\right)}\left[\log p\left(y_i^{\mathrm{ts}} \mid x_i^{\mathrm{ts}}, \phi_i\right)\right] - D_{KL}\left(q\left(\phi_i \mid \mathscr{D}_i^{\mathrm{tr}}, \theta\right) \| p(\phi_i \mid \theta)\right)\right]$$

Pros:

+ can represent non-Gaussian distributions over y^{ts}
+ produces distribution over functions

Cons:

- Can only represent Gaussian distributions $p(\phi_i \mid \theta)$

(okay when ϕ_i is latent vector)

Plan for Today

Why be Bayesian?

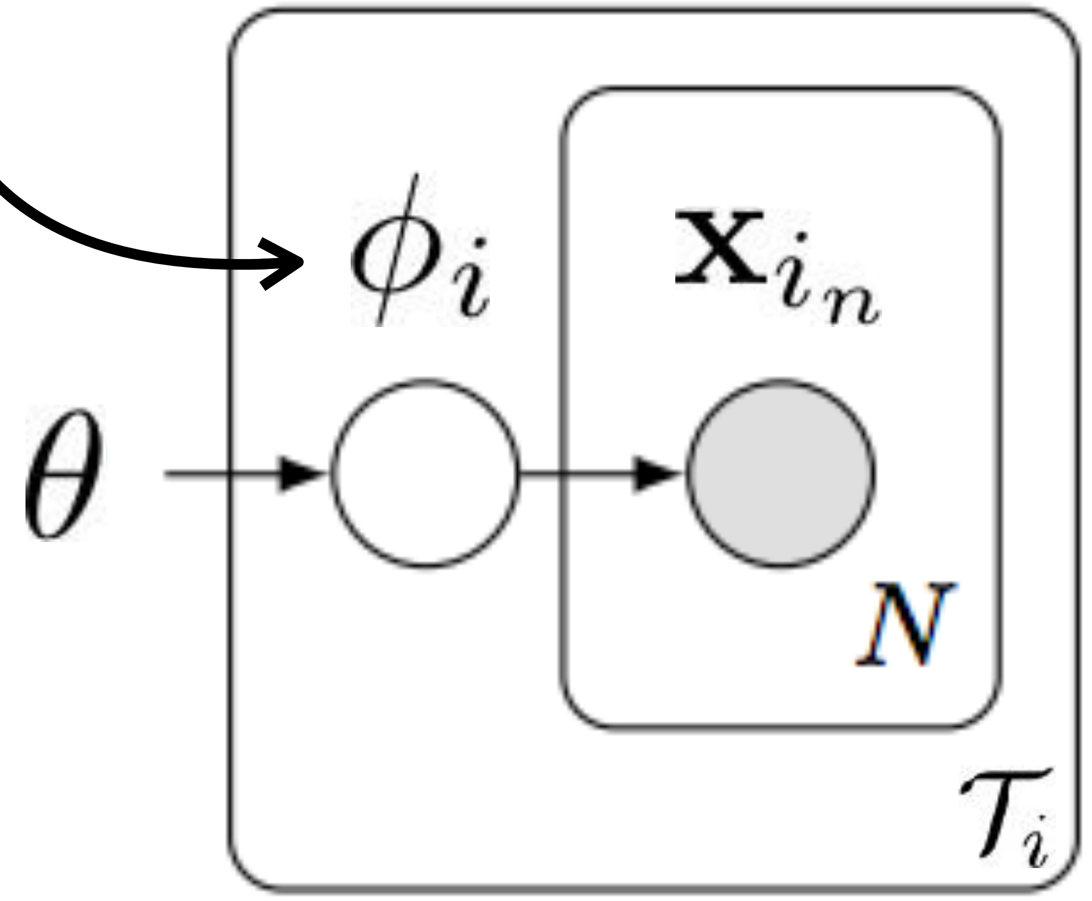
Bayesian meta-learning approaches
- black-box approaches
- **optimization-based approaches (time permitting)**

How to evaluate Bayesian meta-learners.

What about Bayesian **optimization-based** meta-learning?

Recasting Gradient-Based Meta-Learning as Hierarchical Bayes (Grant et al. ’18)

task-specific parameters



meta-parameters

$$\max_\theta \log \prod_i p(\mathcal{D}_i|\theta)$$

$$= \log \prod_i \int p(\mathcal{D}_i|\phi_i)p(\phi_i|\theta)d\phi_i \quad \text{(empirical Bayes)}$$

$$\approx \log \prod_i p(\mathcal{D}_i|\hat{\phi}_i)p(\hat{\phi}_i|\theta)$$

MAP estimate

How to compute MAP estimate?

Gradient descent with **early stopping** = **MAP inference** under **Gaussian prior** with mean at initial parameters [Santos ’96]

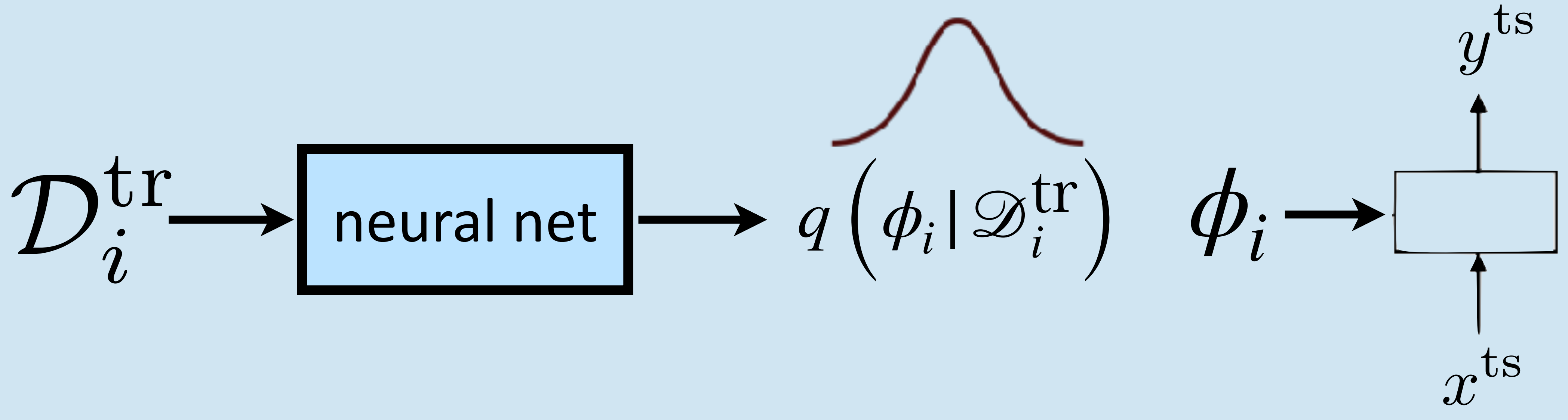
(exact in linear case, approximate in nonlinear case)

Provides a Bayesian interpretation of MAML.

But, we can’t **sample** from $p\left(\phi_i|\theta, \mathcal{D}_i^{\text{tr}}\right)$!

What about Bayesian **optimization-based** meta-learning?

Recall: Bayesian black-box meta-learning
with standard, deep variational inference



$$\max_{\theta} \mathbb{E}_{\mathcal{T}_i}\left[\mathbb{E}_{q\left(\phi_i \mid \mathcal{D}_i^{\mathrm{tr}}, \theta\right)}\left[\log p\left(y_i^{\mathrm{ts}} \mid x_i^{\mathrm{ts}}, \phi_i\right)\right] - D_{KL}\left(q\left(\phi_i \mid \mathcal{D}_i^{\mathrm{tr}}, \theta\right) \| p(\phi_i \mid \theta)\right)\right]$$

q: an arbitrary function

q can include a gradient operator!

Amortized Bayesian Meta-Learning
(Ravi & Beatson '19)

q corresponds to SGD on the mean & variance of neural network weights (μ_ϕ, σ_ϕ^2), w.r.t. $\mathcal{D}_i^{\mathrm{tr}}$

Pro: Running gradient descent at test time. Con: $p(\phi_i \mid \theta)$ modeled as a Gaussian.

Can we model **non-Gaussian** posterior?

What about Bayesian optimization-based meta-learning?

Can we use ensembles?

Kim et al. Bayesian MAML '18

An ensemble of mammals

Ensemble of MAMLs (EMAML)

Train M independent MAML models.

Won't work well if ensemble members are too similar.

Note: Can also use ensembles w/ black-box, non-parametric methods!

A more diverse ensemble of mammals

Stein Variational Gradient (BMAML)

Use stein variational gradient (SVGD) to push particles away from one another

$$\phi(\theta_t) = \frac{1}{M}\sum_{j=1}^{M}\left[k(\theta_t^j, \theta_t)\nabla_{\theta_t^j}\log p(\theta_t^j) + \underline{\nabla_{\theta_t^j} k(\theta_t^j, \theta_t)}\right]$$

Optimize for distribution of M particles to produce high likelihood.

$$\mathcal{L}_{\text{BFA}}(\Theta_\tau(\Theta_0); \mathcal{D}_\tau^{\text{val}}) = \log \underline{\left[\frac{1}{M}\sum_{m=1}^{M} p(\mathcal{D}_\tau^{\text{val}}|\theta_\tau^m)\right]}$$

Pros: Simple, tends to work well, non-Gaussian distributions.

Con: Need to maintain M model instances.

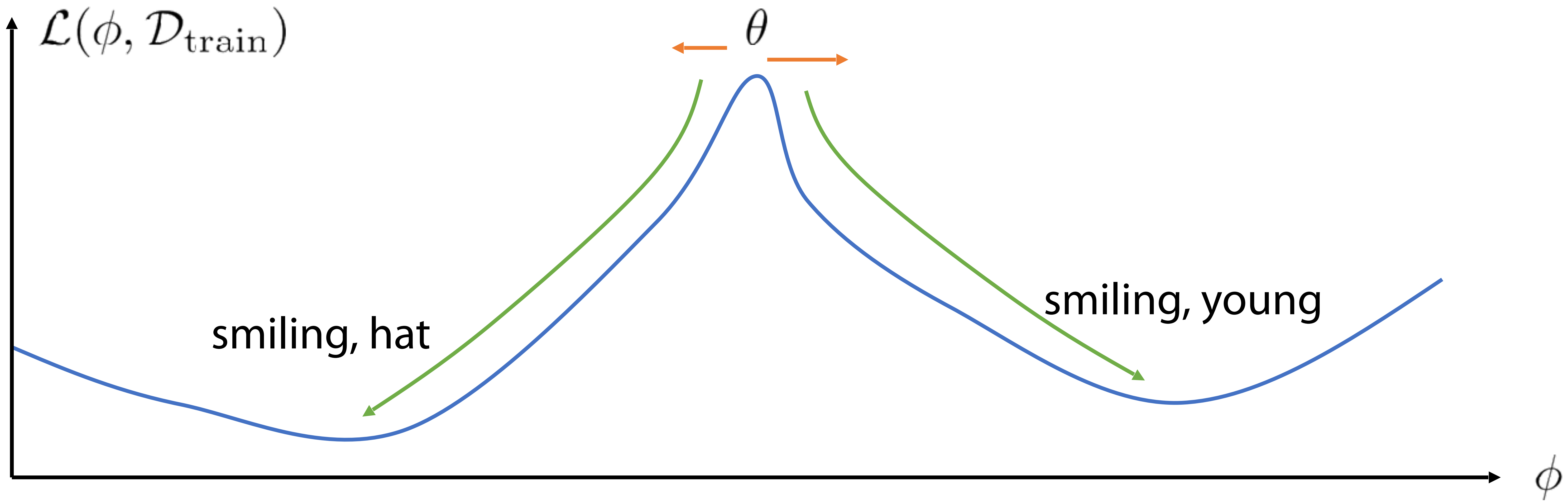
Can we model **non-Gaussian** posterior over **all parameters**?

What about Bayesian optimization-based meta-learning?

Sample parameter vectors with a procedure like Hamiltonian Monte Carlo?

Finn*, Xu*, Levine. Probabilistic MAML ‘18

Intuition: Learn a prior where a random kick can put us in different modes



$$\phi \leftarrow \theta + \epsilon$$

$$\phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L}(\phi, \mathcal{D}_{\text{train}})$$

What about Bayesian optimization-based meta-learning?

Sample parameter vectors with a procedure like Hamiltonian Monte Carlo?

Finn*, Xu*, Levine. Probabilistic MAML '18

$$\theta \sim p(\theta) = \mathcal{N}(\mu_\theta, \Sigma_\theta) \qquad \phi_i \sim p(\phi_i|\theta)$$

(not single parameter vector anymore)

$$\text{Goal: sample } \phi_i \sim p(\phi_i | x_i^{\text{train}}, y_i^{\text{train}}, \cancel{x_i^{\text{test}}})$$

$$p(\phi_i | x_i^{\text{train}}, y_i^{\text{train}}) \propto \int p(\theta) p(\phi_i|\theta) p(y_i^{\text{train}} | x_i^{\text{train}}, \phi_i) d\theta$$

$\Rightarrow$ this is completely intractable!

what if we knew $p(\phi_i | \theta, x_i^{\text{train}}, y_i^{\text{train}})$?

$\Rightarrow$ now sampling is easy! just use ancestral sampling!

key idea: $p(\phi_i | \theta, x_i^{\text{train}}, y_i^{\text{train}}) \approx \delta(\hat{\phi}_i)$

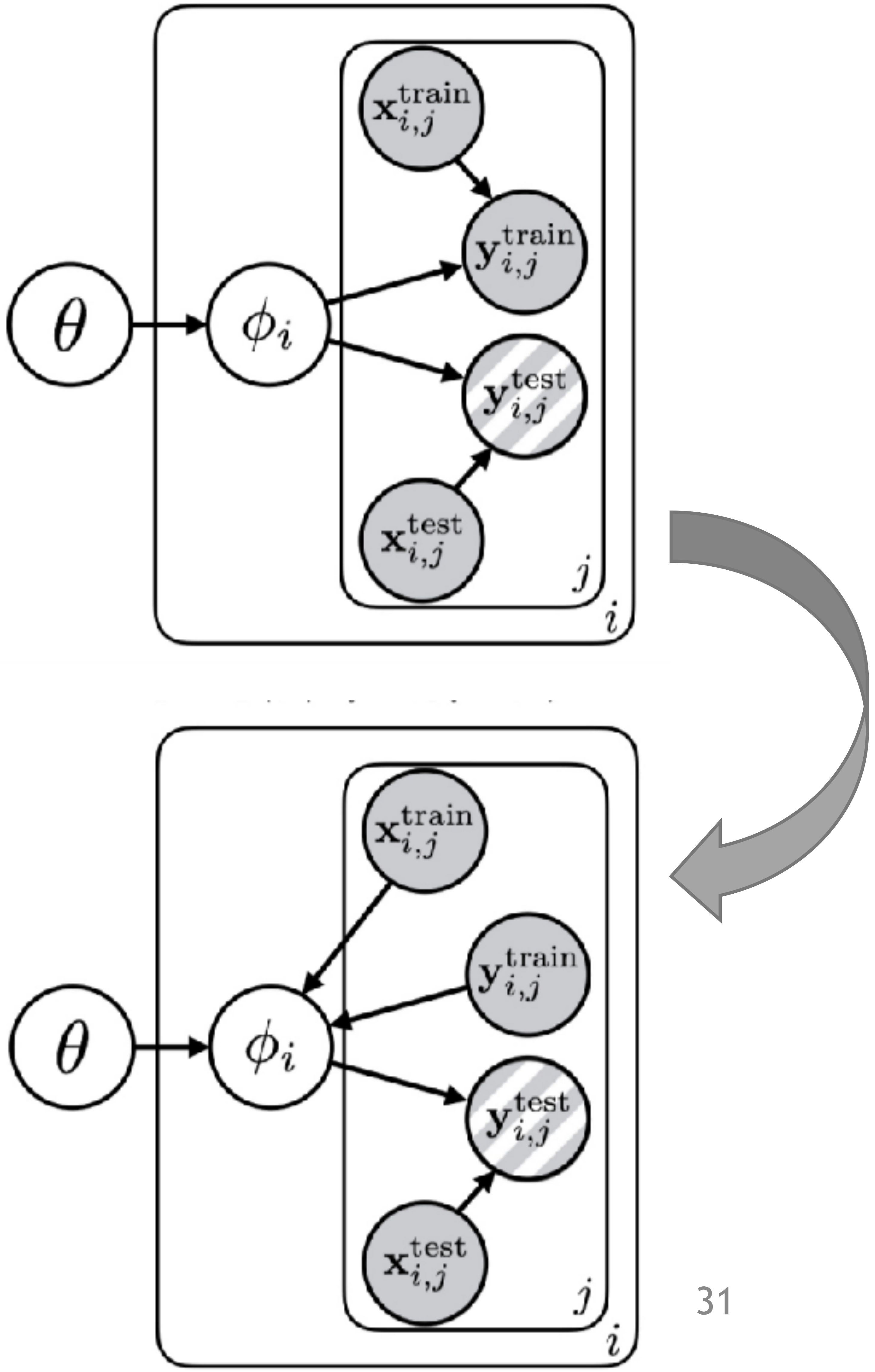
approximate with MAP

this is **extremely** crude

but **extremely** convenient!

$$\hat{\phi}_i \approx \theta + \alpha \nabla_\theta \log p(y_i^{\text{train}} | x_i^{\text{train}}, \theta)$$

(Santos '92, Grant et al. ICLR '18)



Training can be done with amortized variational inference.

What about Bayesian optimization-based meta-learning?

Sample parameter vectors with a procedure like **Hamiltonian Monte Carlo**?

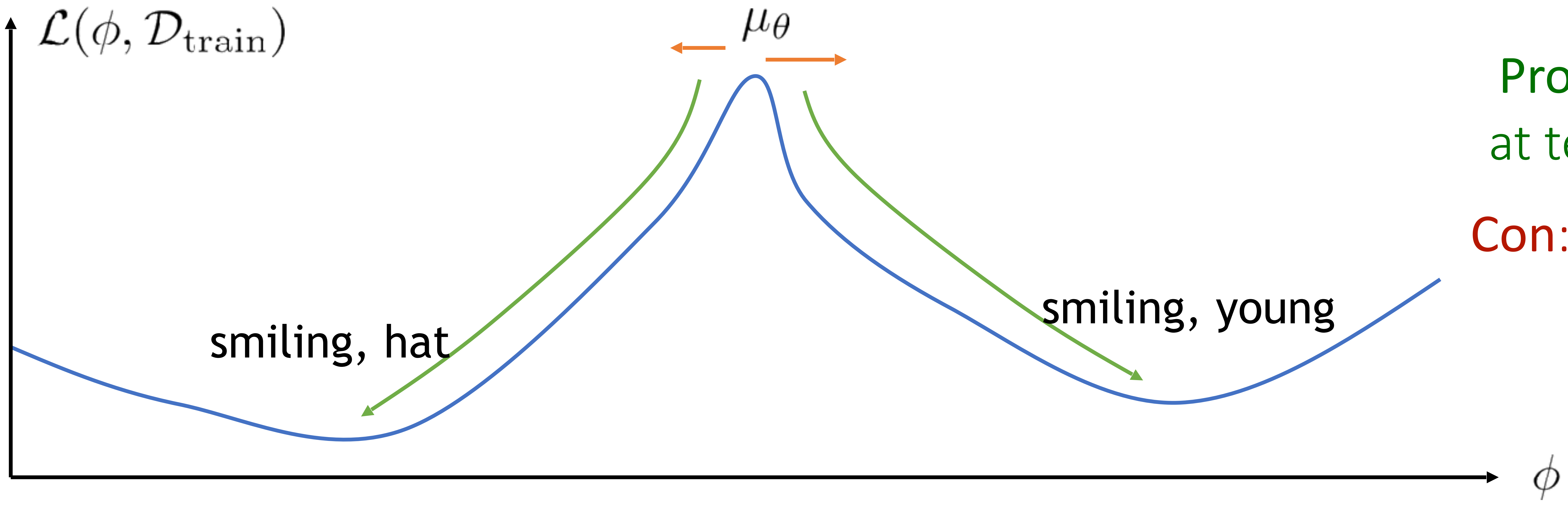
Finn*, Xu*, Levine. Probabilistic MAML '18

$$\theta \sim p(\theta) = \mathcal{N}(\mu_\theta, \Sigma_\theta)$$

$$\textbf{key idea: } p(\phi_i|\theta, x_i^{\text{train}}, y_i^{\text{train}}) \approx \delta(\hat{\phi}_i) \qquad \hat{\phi}_i \approx \theta + \alpha \nabla_\theta \log p(y_i^{\text{train}}|x_i^{\text{train}}, \theta)$$

What does ancestral sampling look like?

1. $\theta \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)$
2. $\phi_i \sim p(\phi_i|\theta, x_i^{\text{train}}, y_i^{\text{train}}) \approx \hat{\phi}_i = \theta + \alpha \nabla_\theta \log p(y_i^{\text{train}}|x_i^{\text{train}}, \theta)$



Pros: Non-Gaussian posterior, simple at test time, only one model instance.

Con: More complex training procedure.

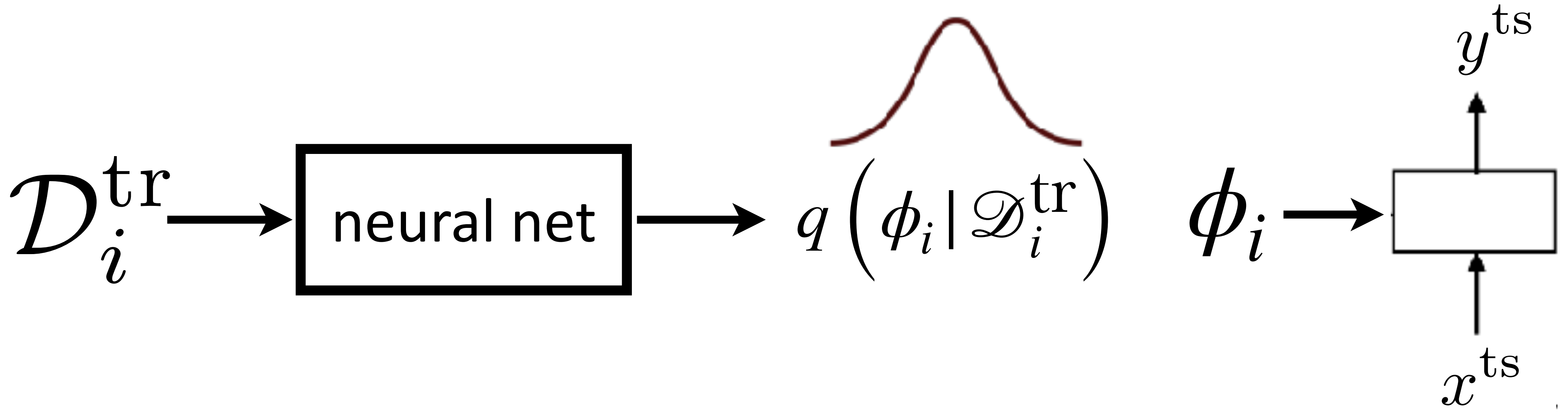
Methods Summary

Version 0: f outputs a distribution over y^{ts}.

Pros: simple, can combine with variety of methods

Cons: can't reason about uncertainty over the underlying function, limited class of distributions over y^{ts} can be expressed

Black box approaches: Use latent variable models + amortized variational inference



Pros: can represent non-Gaussian distributions over y^{ts}

Cons: Can only represent Gaussian distributions $p(\phi_i \mid \theta)$ (okay when ϕ_i is latent vector)

Optimization-based approaches:

Amortized inference

Pro: Simple.

Con: $p(\phi_i \mid \theta)$ modeled as a Gaussian.

Ensembles

Pros: Simple, tends to work well, non-Gaussian distributions.

Con: maintain M model instances. (or do inference on **last layer only**)

Hybrid inference

Pros: Non-Gaussian posterior, simple at test time, only one model instance.

Con: More complex training procedure.

Plan for Today

Why be Bayesian?

Bayesian meta-learning approaches
- black-box approaches
- optimization-based approaches (time permitting)

How to evaluate Bayesian meta-learners.

How to evaluate a Bayesian meta-learner?

Use the standard benchmarks?
(i.e. MiniImagenet accuracy)

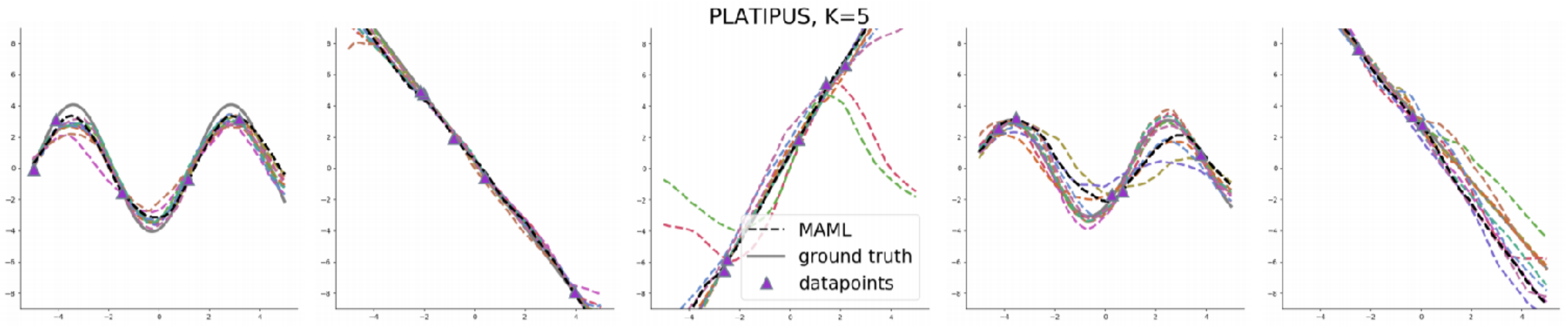
+ standardized
+ real images
+ good check that the approach didn’t break anything
- metrics like accuracy don't evaluate uncertainty
- tasks may not exhibit ambiguity
- uncertainty may not be useful on this dataset!

What are better problems & metrics?
It depends on the problem you care about!

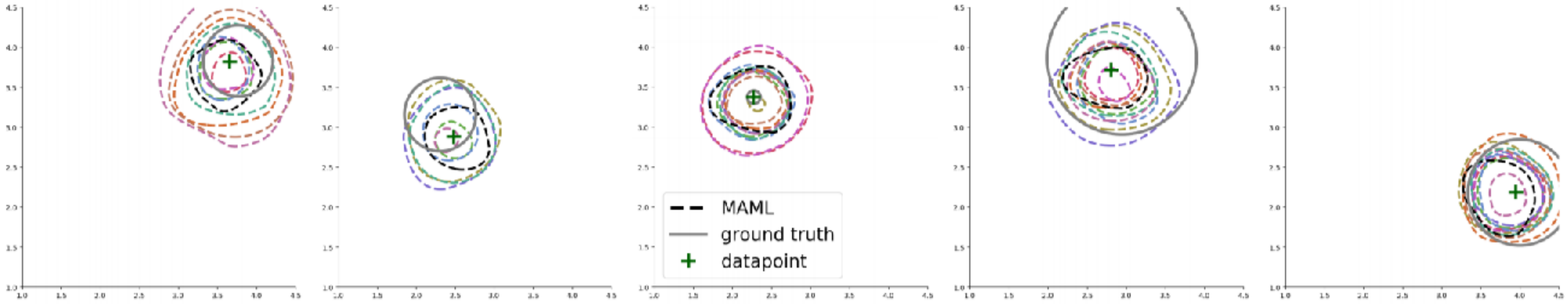
Qualitative Evaluation on Toy Problems with Ambiguity

(Finn*, Xu*, Levine, NeurIPS ’18)

Ambiguous regression:

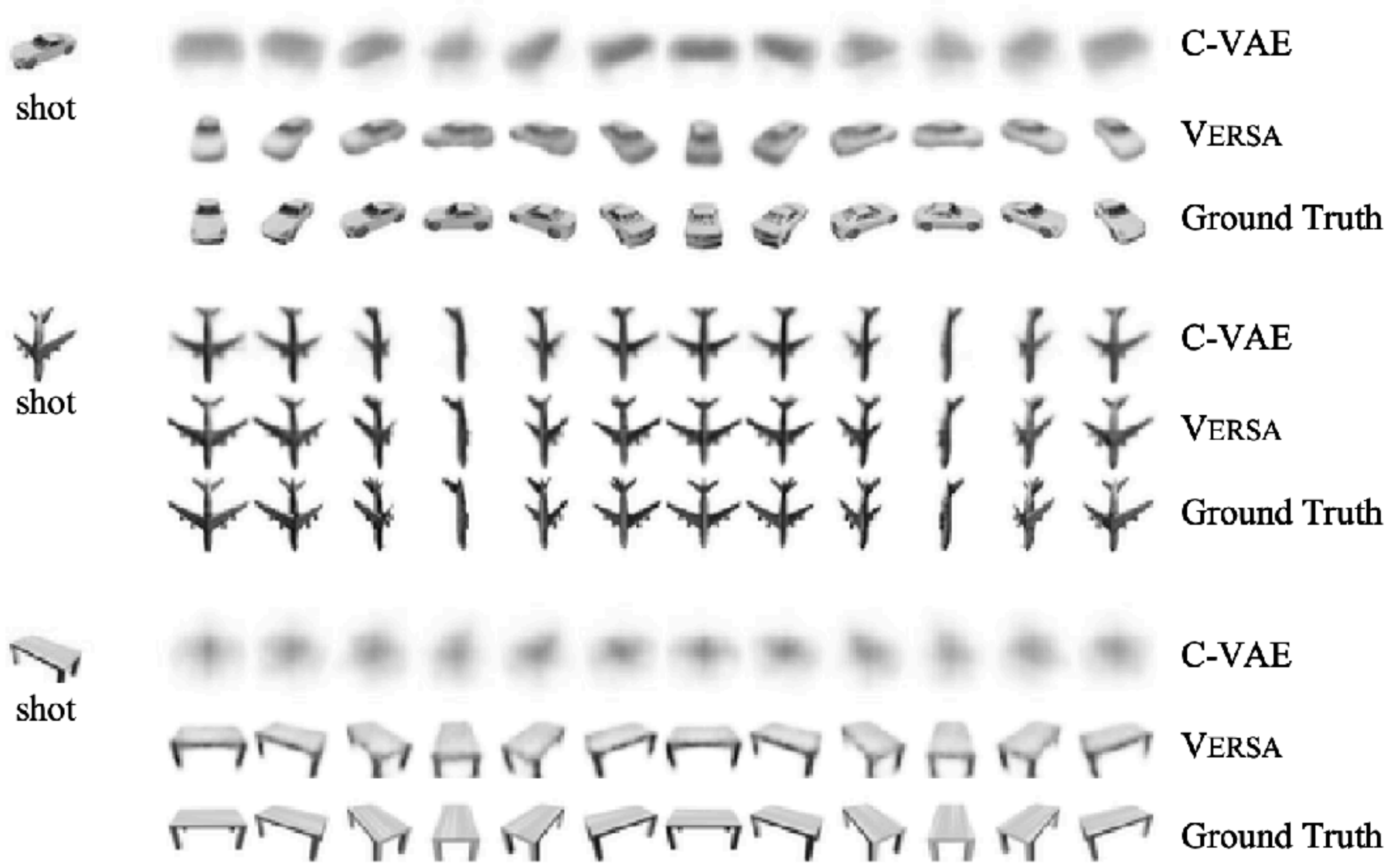


Ambiguous classification:



Evaluation on Ambiguous Generation Tasks

(Gordon et al., ICLR '19)



Model	MSE	SSIM
C-VAE 1-shot	0.0269	0.5705
VERSA 1-shot	0.0108	0.7893
VERSA 5-shot	0.0069	0.8483

Table 2: View reconstruction test results.

Accuracy, Mode Coverage, & Likelihood on Ambiguous Tasks

(Finn*, Xu*, Levine, NeurIPS ’18)

Ambiguous celebA (5-shot)			
	Accuracy	Coverage (max=3)	Average NLL
MAML	**89.00 ± 1.78%**	1.00 ± 0.0	0.73 ± 0.06
MAML + noise	84.3 ± 1.60 %	1.89 ± 0.04	0.68 ± 0.05
PLATIPUS (ours) (KL weight = 0.05)	**88.34 ± 1.06 %**	1.59 ± 0.03	0.67± 0.05
PLATIPUS (ours) (KL weight = 0.15)	**87.8 ± 1.03 %**	**1.94 ± 0.04**	**0.56 ± 0.04**

Reliability Diagrams & Accuracy

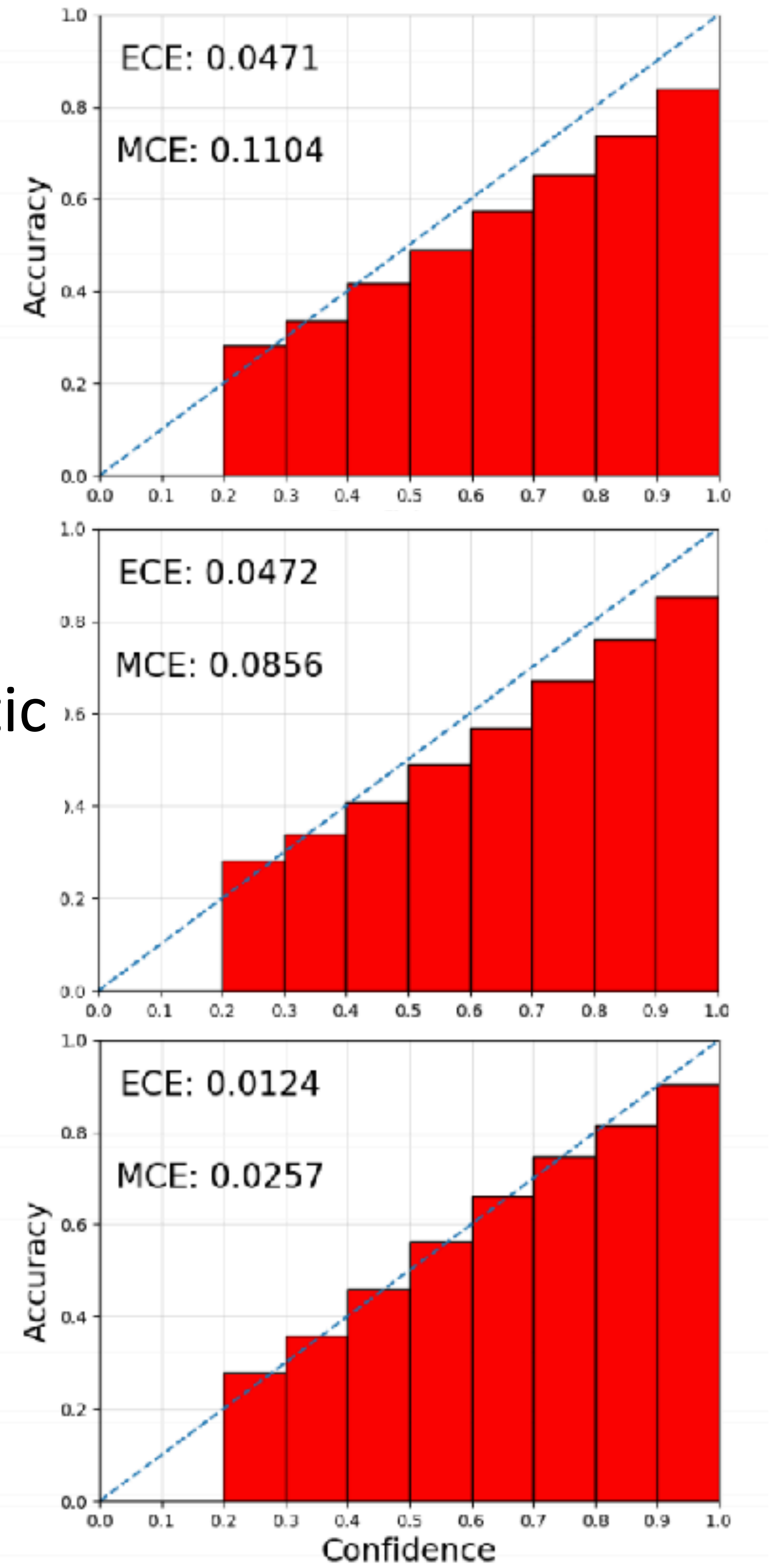
(Ravi & Beatson, ICLR '19)

*mini*ImageNet: 1-shot, 5-class

MAML

Probabilistic MAML

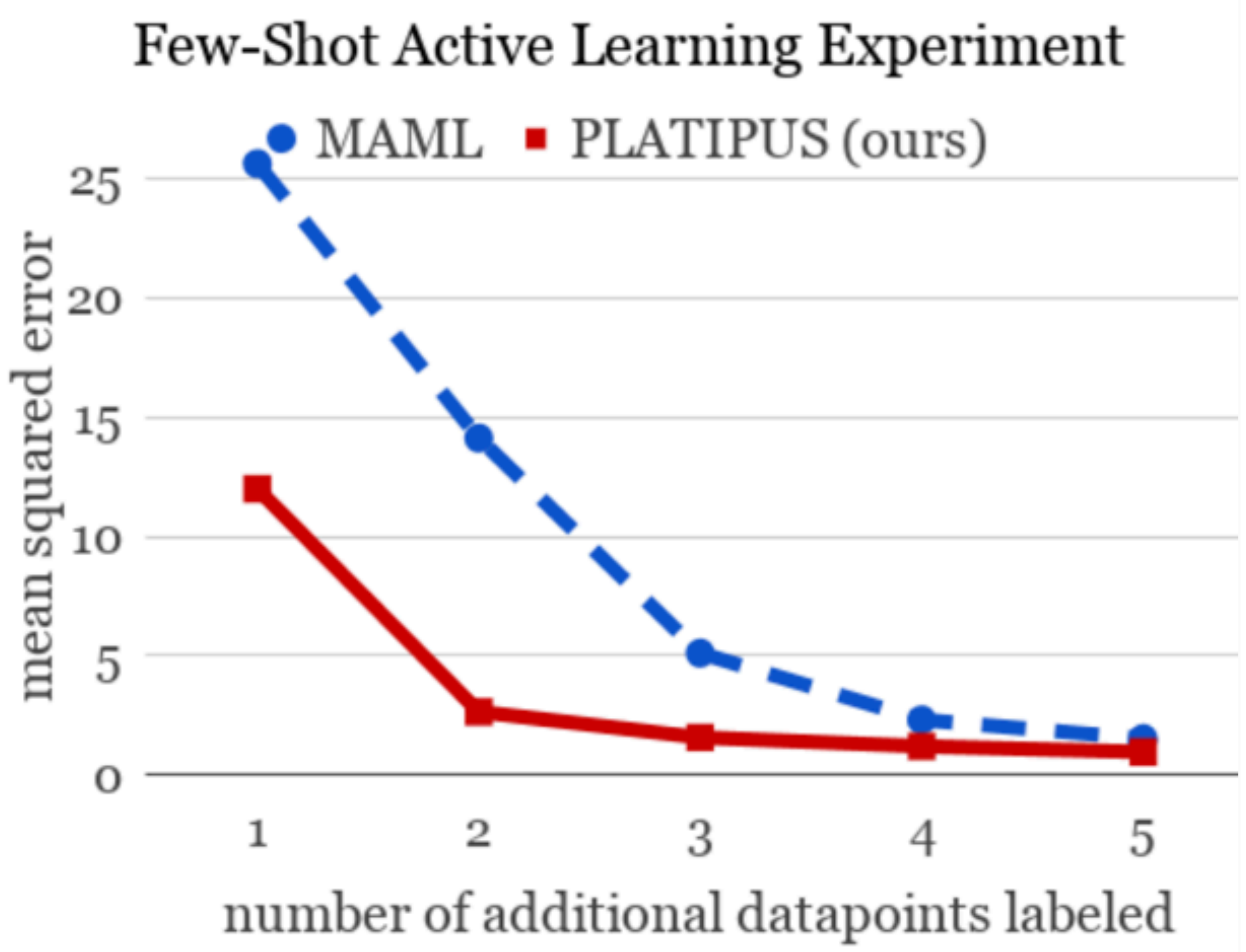
Ravi & Beatson



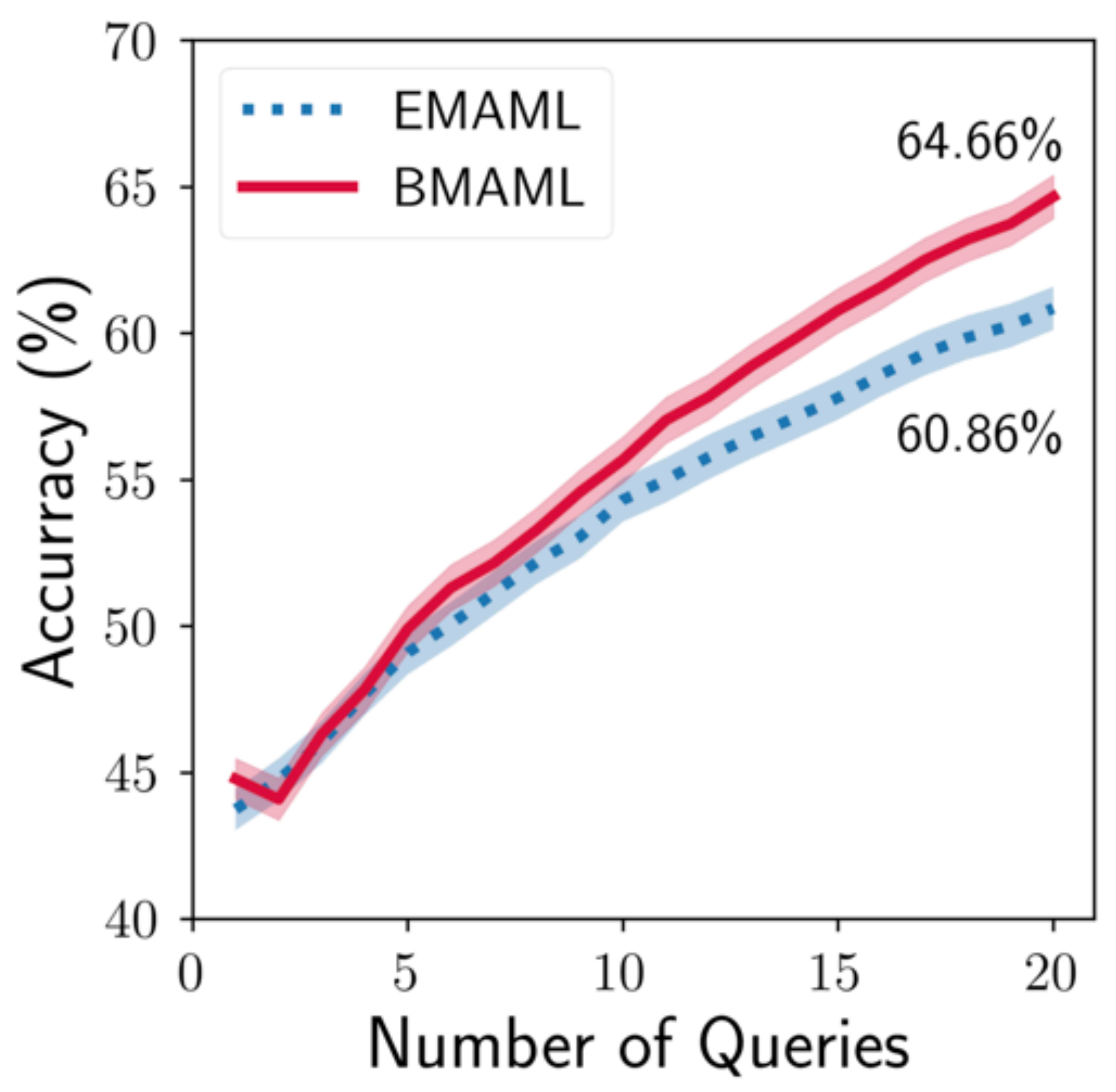
*mini*ImageNet	1-shot, 5-class
MAML (ours)	47.0 ± 0.59
Prob. MAML (ours)	47.8 ± 0.61
Our Model	45.0 ± 0.60

Active Learning Evaluation

Finn*, Xu*, Levine, NeurIPS '18
Sinusoid Regression



Kim et al. NeurIPS '18
MiniImageNet



Both experiments:

- Sequentially choose datapoint with **maximum predictive entropy** to be labeled
- Choose datapoint at random for non-Bayesian methods

Algorithmic properties **perspective**

Expressive power

the ability for f to represent a range of learning procedures

Why? scalability, applicability to a range of domains

Consistency

learned learning procedure will solve task with enough data

Why? reduce reliance on meta-training tasks, good OOD task performance

Uncertainty awareness

ability to reason about ambiguity during learning

Why? active learning, calibrated uncertainty, RL principled Bayesian approaches

Plan for Today

Why be Bayesian?

Bayesian meta-learning approaches

- black-box approaches
- optimization-based approaches (time permitting)

How to evaluate Bayesian meta-learners.

Goals for by the end of lecture:

- Understand the interpretation of meta-learning as Bayesian inference
- Understand techniques for representing uncertainty over parameters, predictions

Next Time

Next week: Large-scale meta-optimization
(incl. guest lecture on learned optimizers!)

Following week: Domain adaptation & lifelong learning

Following week: Thanksgiving 🦃🦃

Course Reminders

Homework 3 due **Monday**.

Tutorial session **tomorrow 4:30 pm**