

Bayesian Meta-Learning

CS 330

Course Reminders

Homework 3 due ~~Wednesday~~ Friday.

Homework 4 (optional) out today.

Plan for Today

Why be Bayesian?

Bayesian meta-learning approaches

- black-box approaches
- optimization-based approaches

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

Disclaimers

Bayesian meta-learning is an active area of research
(like most of the class content)

More questions than answers.

Recap: Properties of Meta-Learning Inner Loops

Properties of the inner loop
of the algorithm.

Expressive power

Algorithmic properties perspective

the ability for f to represent a range of learning procedures

Why? scalability, applicability to a range of domains

learned learning procedure will solve task with enough data

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

Consistency

↳ Statistical consistency.

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



Strong in Bayesian methods
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

How to evaluate Bayesian meta-learners.

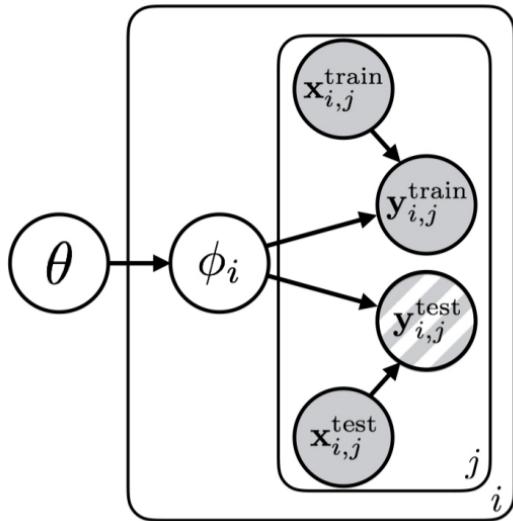
Multi-Task & Meta-Learning Principles

Training and testing must match.

Tasks must share “structure.”

What does “structure” mean?

statistical dependence on shared latent information θ



If you condition on that information,

- task parameters become independent

$$\text{i.e. } \phi_{i_1} \perp\!\!\!\perp \phi_{i_2} \mid \theta$$

and are not otherwise independent $\phi_{i_1} \perp\!\!\!\perp \phi_{i_2}$

- hence, you have a lower entropy

$$\text{i.e. } \mathcal{H}(p(\phi_i \mid \theta)) < \mathcal{H}(p(\phi_i))$$

- Task is from the same distrib. as ones used to train θ
- will reduce the uncertainty over ϕ .

- If tasks are completely independent, conditioning on the θ won't reduce the entropy of ϕ .

Memorization;
not using the
support set

Thought exercise #1: If you can identify θ (i.e. with meta-learning), when should learning ϕ_i be faster than learning from scratch?

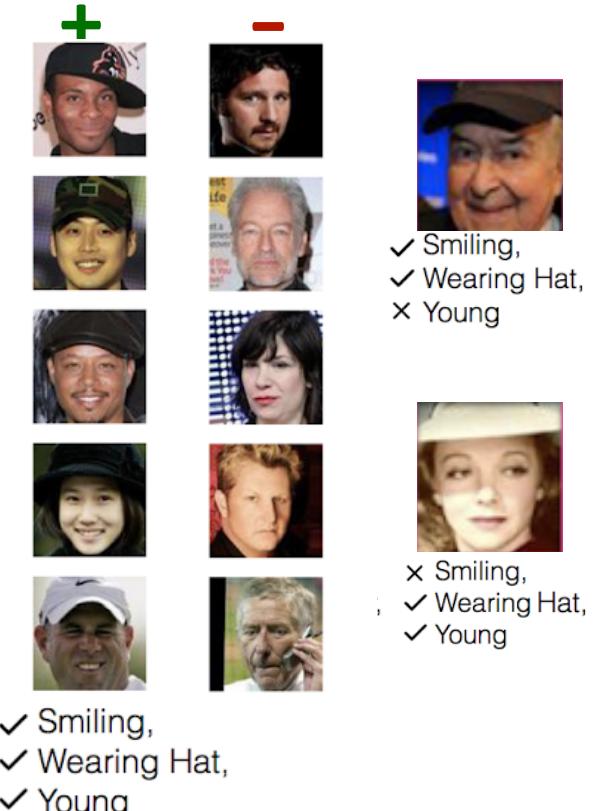
Thought exercise #2: what if $\mathcal{H}(p(\phi_i \mid \theta)) = 0 \quad \forall i?$

entropy ↓
Conditioned
on θ

θ tells you everything there is to know about the task-specific parameters (ϕ)

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

only gives you one set of ϕ_i , not a distribution of ϕ_i



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 $p(\phi_i | \mathcal{D}_i^{\text{tr}}, \theta)$

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?

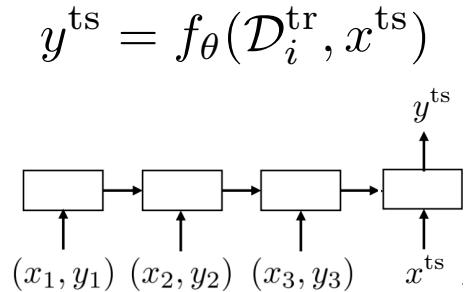
Bayesian meta-learning approaches

- black-box approaches
- optimization-based approaches

How to evaluate Bayesian meta-learners.

Meta-learning algorithms as computation graphs

Black-box



Optimization-based

$$\begin{aligned} y^{\text{ts}} &= f_{\text{MAML}}(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}}) \\ &= f_{\phi_i}(x^{\text{ts}}) \end{aligned}$$

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

$$\begin{aligned} y^{\text{ts}} &= f_{\text{PN}}(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}}) \\ &= \text{softmax}(-d(f_{\theta}(x^{\text{ts}}), \mathbf{c}_n)) \end{aligned}$$

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

Non-parametric

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.

outer loss of
the algorithm.

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 ?

$p(\phi|D_i^{\text{tr}})$, ~ apply max. likelihood?
→ to do this, we need ground truth labels for ϕ (we don't have it, we only have if for y)
→ No, we can't apply ML → $p(\phi|D_i^{\text{tr}})$ to get $p(\phi)$.



Not actual probabilities, most of the time overconfident.

does this for uncertainty over labels though.
eg) Hard to describe dists like $p(y|x, D_i^{\text{tr}})$
Q: Basis for measures of calibration?
→ Expected calibration error, Reliability diag. (psa)

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):

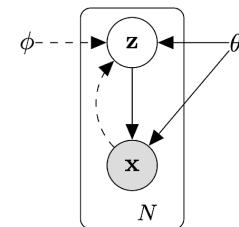
- approximate likelihood of latent variable model with variational lower bound



Bayesian ensembles (Lakshminarayanan et al. '17):

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

Fit multiple models on datasets using max. likelihood.
multiple particles. (Samples from the dist.).

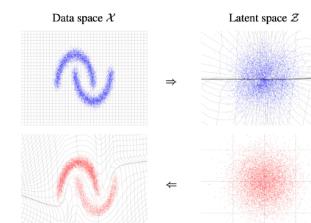


Bayesian neural networks (Blundell et al. '15):

- explicit Gaussian distribution over the space of network parameters

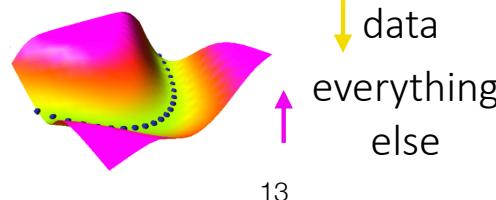
Normalizing Flows (Dinh et al. '16):

- invertible function from latent distribution to data distribution



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

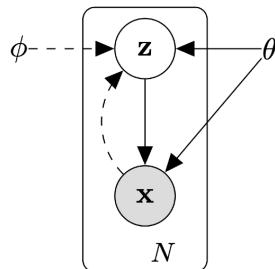
- estimate unnormalized density



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

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

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

p : model $p(x|z)$ represented w/ neural net,
 $p(z)$ represented as $\mathcal{N}(\mathbf{0}, \mathbf{I})$ model parameters θ ,
 $q(z|x)$: inference network, variational distribution variational parameters ϕ

Problem: need to backprop through sampling

i.e. compute derivative of \mathbb{E}_q w.r.t. q

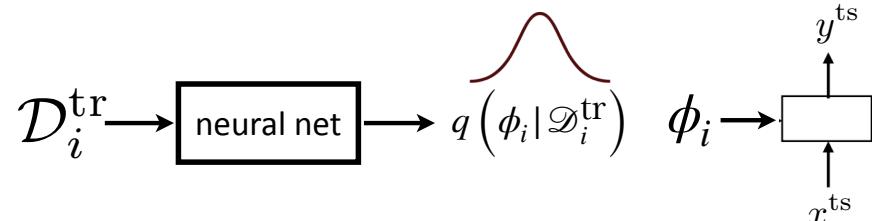
Reparametrization trick For Gaussian $q(z|x)$:

$$q(z|x) = \mu_q + \sigma_q \epsilon \quad \text{where } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

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

p : model, represented by a neural net

q : inference network, variational distribution

Meta-learning:



Observed variable \mathcal{D} , latent variable ϕ

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

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

What should q condition on?

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

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

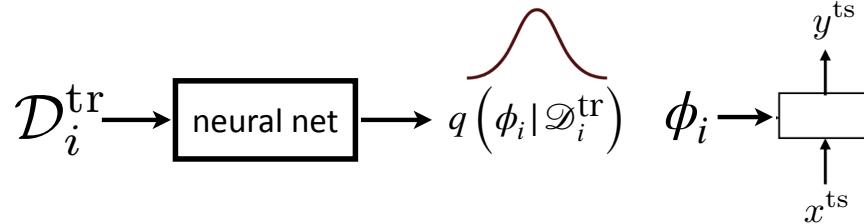
What about the meta-parameters θ ?

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



Can also condition on θ here

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



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

Pros:

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

Cons:

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

Plan for Today

Why be Bayesian?

Bayesian meta-learning approaches

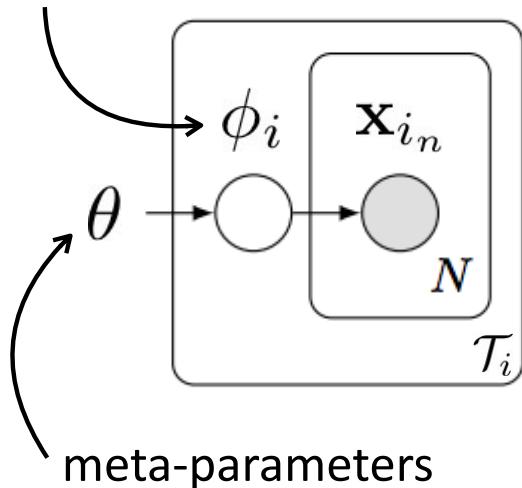
- black-box approaches
- **optimization-based approaches**

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

$$\begin{aligned} & \max_{\theta} \log \prod p(\mathcal{D}_i | \theta) \\ &= \log \prod_i^i \int p(\mathcal{D}_i | \phi_i) p(\phi_i | \theta) d\phi_i \quad (\text{empirical Bayes}) \\ &\approx \log \prod_i p(\mathcal{D}_i | \hat{\phi}_i) p(\hat{\phi}_i | \theta) \end{aligned}$$

MAP estimate

How to compute MAP estimate?

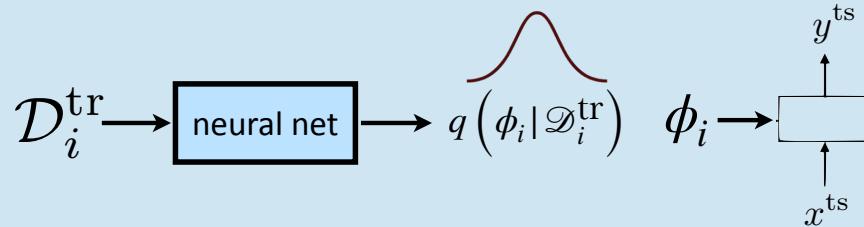
Gradient descent with early stopping = MAP inference under Gaussian prior with mean at initial parameters [Santos '96]
(exact in linear case, approximate in nonlinear case)

Provides a Bayesian interpretation of MAML.

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

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

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



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

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 $(\mu_\phi, \sigma_\phi^2)$, w.r.t. $\mathcal{D}_i^{\text{tr}}$

Pro: Running gradient descent at test time. **Con:** $p(\phi_i | \theta)$ 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



A more diverse 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!

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.
(or do gradient-based inference on **last layer only**)

Can we model non-Gaussian posterior over all parameters?

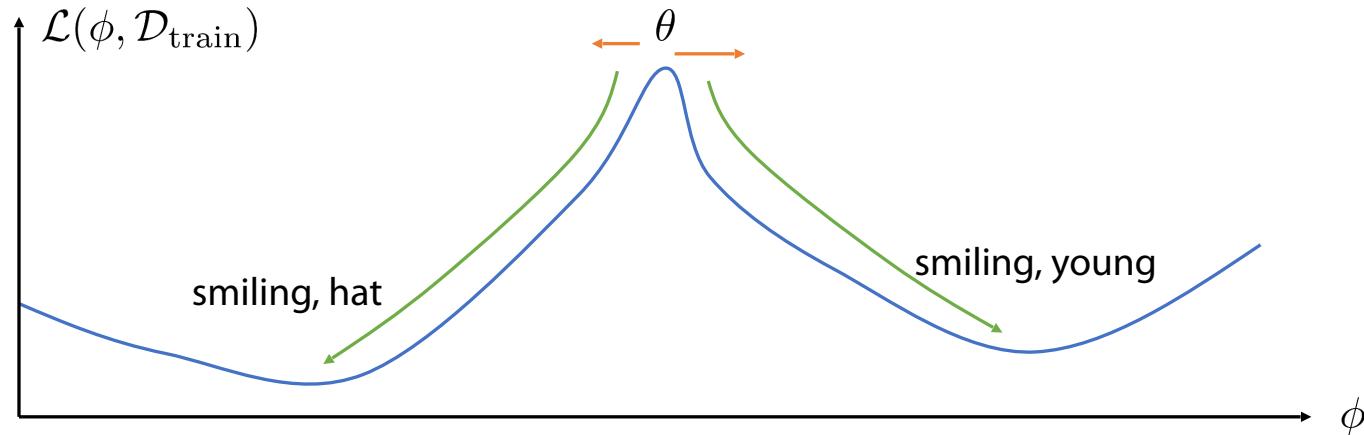
What about Bayesian **optimization-based** meta-learning?

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



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

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

- ✓ Smiling,
- ✓ Wearing Hat,
- ✓ Young

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

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

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

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

(not single parameter vector anymore)

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

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

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

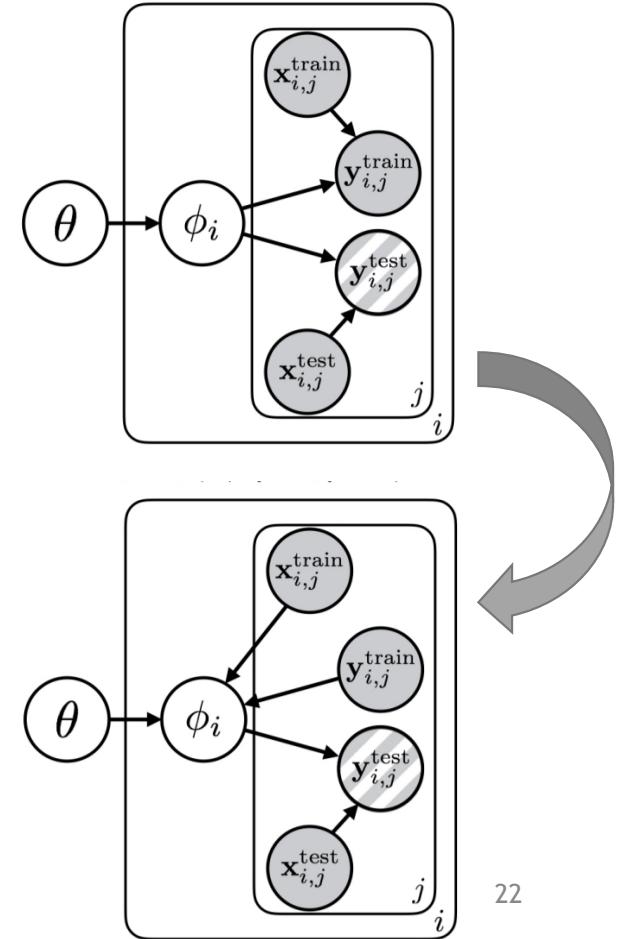
this is **extremely** crude

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**?

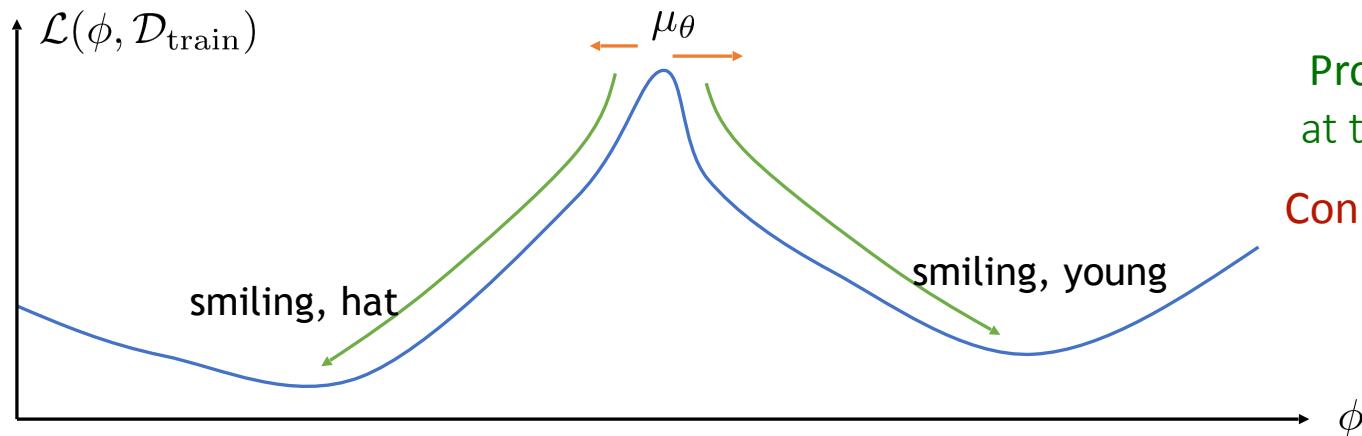
Finn*, Xu*, Levine. Probabilistic MAML '18

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

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

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.

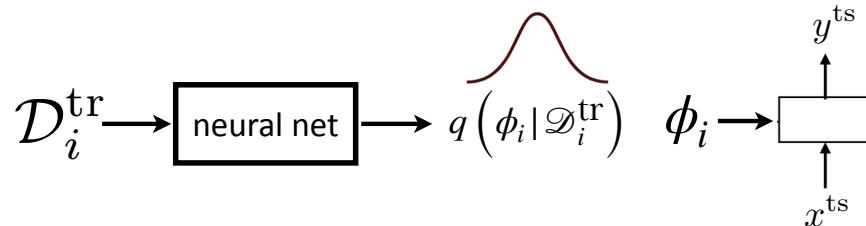
Methods Summary

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 $p(\phi_i | \theta)$
(okay when ϕ_i is latent vector)

Optimization-based approaches:

Amortized inference

Pro: Simple.

Con: $p(\phi_i | \theta)$ 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

How to evaluate Bayesian meta-learners.

How to evaluate a Bayesian meta-learner?

Use the standard benchmarks?

(i.e. Minilmagenet 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!

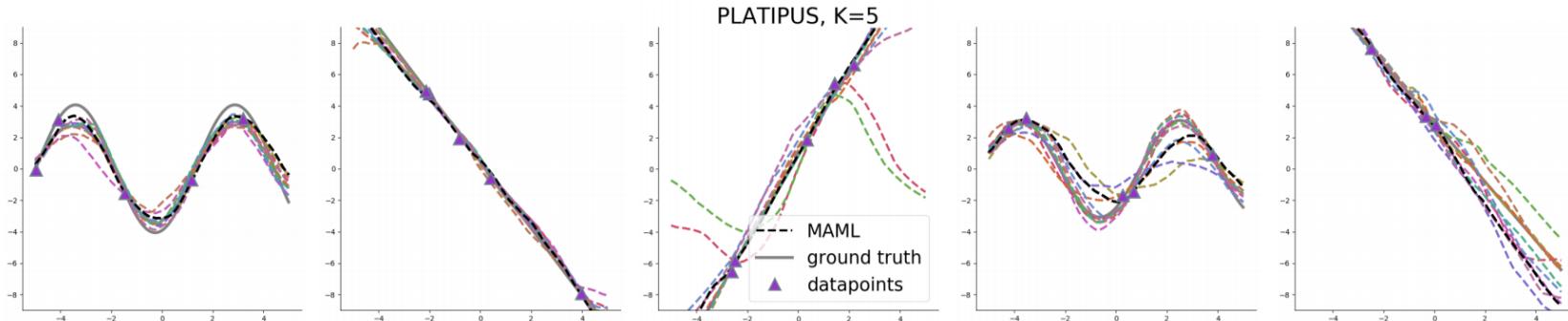
What are better problems & metrics?

It depends on the problem you care about!

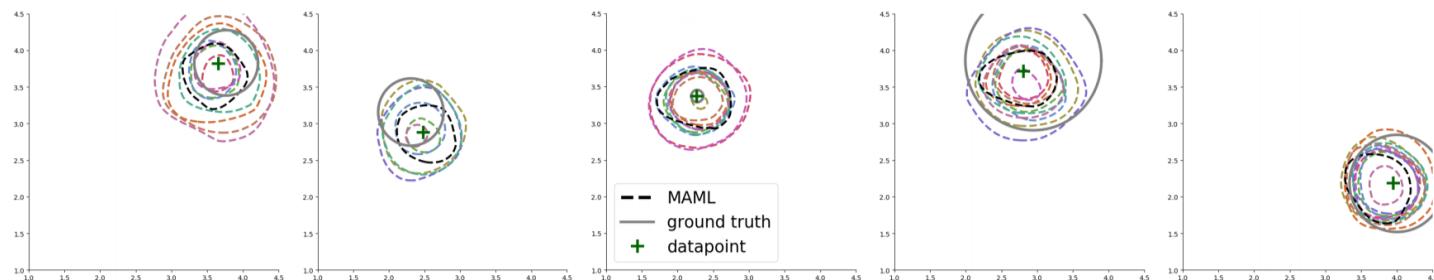
Qualitative Evaluation on Toy Problems with Ambiguity

(Finn*, Xu*, Levine, NeurIPS '18)

Ambiguous regression:



Ambiguous classification:



Evaluation on Ambiguous Generation Tasks

(Gordon et al., ICLR '19)



shot



C-VAE



VERSA



Ground Truth



shot



C-VAE



VERSA



Ground Truth



shot



C-VAE



VERSA



Ground Truth

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)



✓ Mouth Open
✓ Wearing Hat
✓ Young

✓ Mouth Open
✗ Wearing Hat
✓ Young

✓ Mouth Open
✓ Wearing Hat
✗ Young

✗ Mouth Open
✓ Wearing Hat
✗ Young

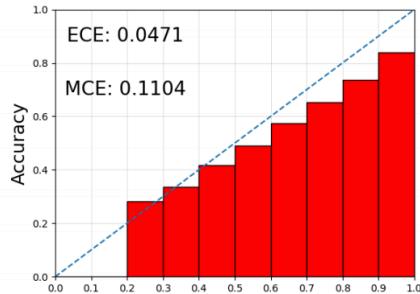
Ambiguous celebA (5-shot)			
	Accuracy	Coverage (max=3)	Average NLL
MAML	89.00 ± 1.78%	1.00 ± 0.0	0.73 ± 0.06
MAML + noise	84.3 ± 1.60 %	1.89 ± 0.04	0.68 ± 0.05
PLATIPUS (ours) (KL weight = 0.05)	88.34 ± 1.06 %	1.59 ± 0.03	0.67± 0.05
PLATIPUS (ours) (KL weight = 0.15)	87.8 ± 1.03 %	1.94 ± 0.04	0.56 ±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

Reliability Diagrams & Accuracy

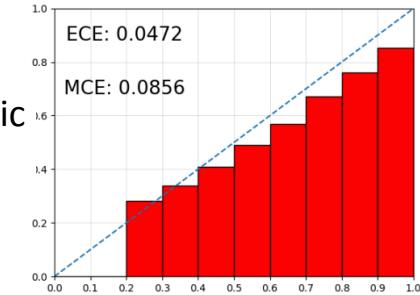
(Ravi & Beatson, ICLR '19)

miniImageNet: 1-shot, 5-class

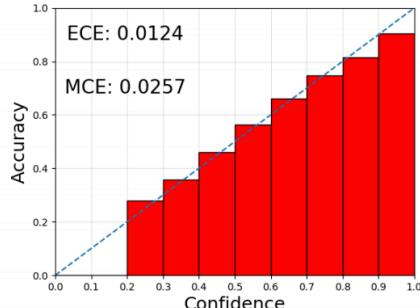
MAML



Probabilistic
MAML



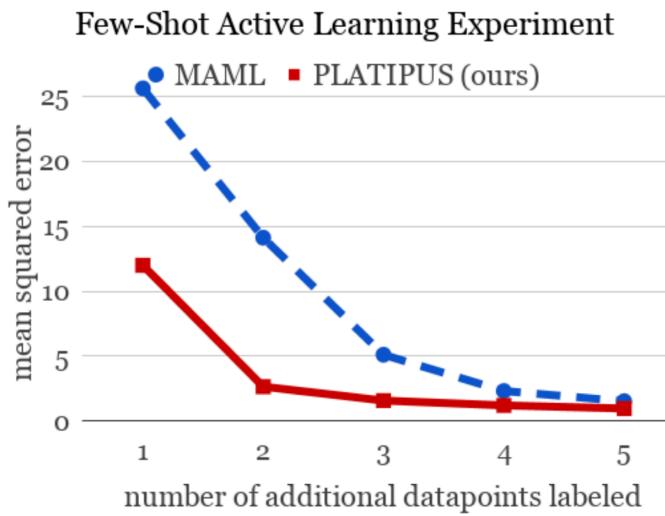
Ravi &
Beatson



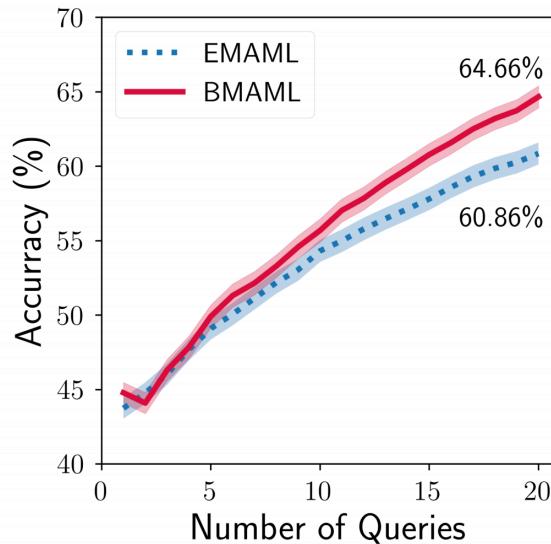
<i>miniImageNet</i>	1-shot, 5-class
MAML (ours)	47.0 ± 0.59
Prob. MAML (ours)	47.8 ± 0.61
Our Model	45.0 ± 0.60

Active Learning Evaluation

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



Kim et al. NeurIPS '18
MinilmageNet



Both experiments:

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

Expressive power

Algorithmic properties perspective

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

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: Domain adaptation & domain generalization

Following week: Lifelong learning & Hanie Sedghi guest lecture

Following week: Thanksgiving 

Course Reminders

Homework 3 due ~~Wednesday~~ Friday.

Homework 4 (optional) out today.