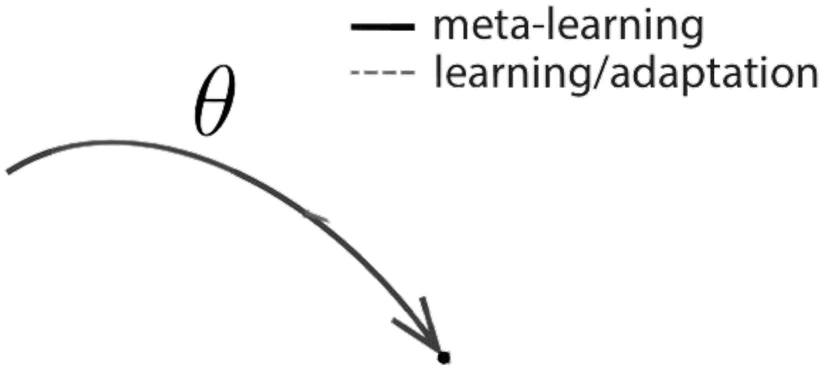
HyperMAML: Hyperinitializing MAML

Affaan Mustafa, Hamish Ivison, Jize Cao, Matthew James Bryan, Peng Zhang, Siting Li

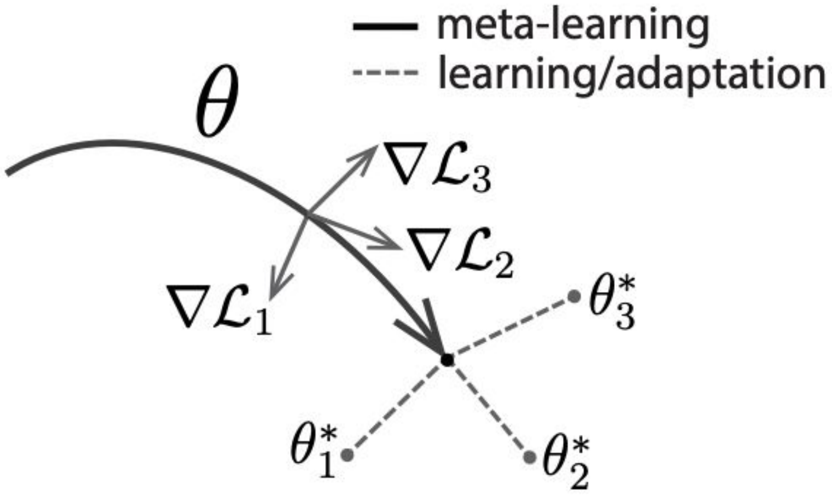
What is MAML?

Aim: Learn a model that can quickly adapt to a new task with little data. (in neural network terms, we are trying to find a good initialization!)



What is MAML?

Aim: Learn a model that can quickly adapt to a new task with little data. (in neural network terms, we are trying to find a good initialization!)



What is MAML?

Achieve this through bilevel optimization:

1. Sample a batch per task

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ~ p(T)
end while
```

What is MAML?

Achieve this through bilevel optimization:

1. Sample a batch per task
2. Compute one update wrt each task

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
    Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
  end for
end while

What is MAML?

Achieve this through bilevel optimization:

1. Sample a batch per task
2. Compute one update wrt each task
3. Update original network with gradient computed via adapted parameters.

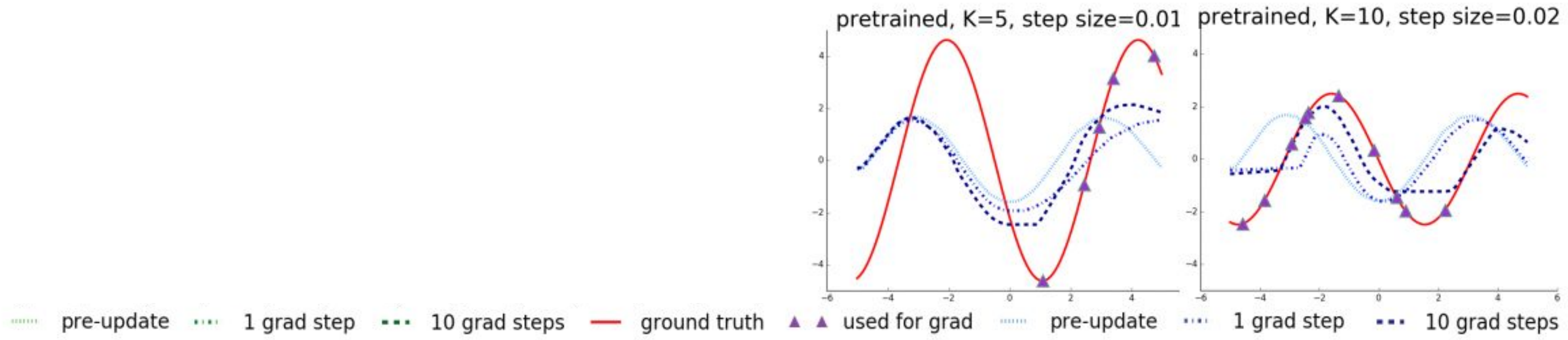
Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ~ p(T)
    for all T_i do
        Evaluate ∇_θ L_{T_i}(f_θ) with respect to K examples
        Compute adapted parameters with gradient descent: θ'_i = θ − α∇_θ L_{T_i}(f_θ)
    end for
    Update θ ← θ − β∇_θ Σ_{T_i~p(T)} L_{T_i}(f_{θ'_i})
end while
```

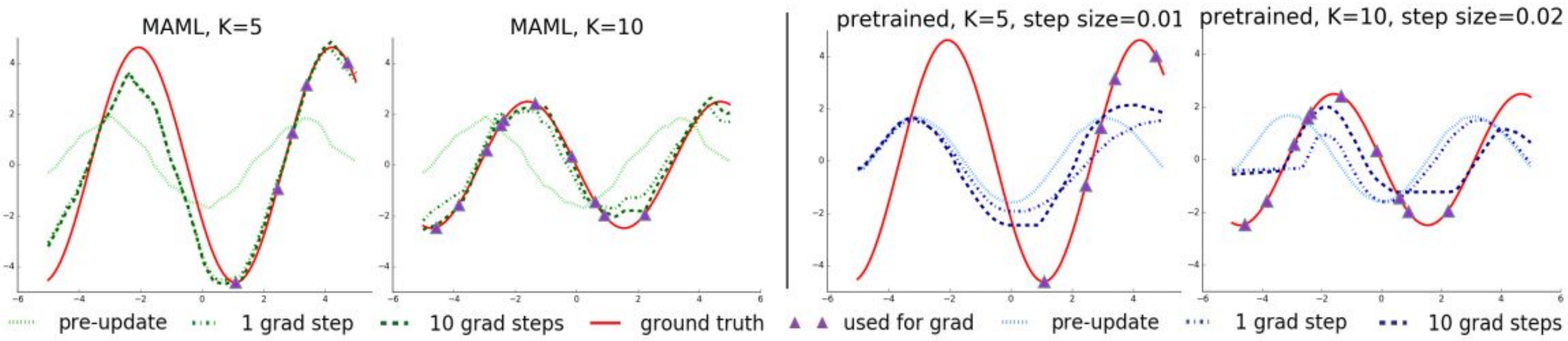
MAML Adapts better

Quick empirical validation: given two sine curves, how well can we adapt a model given some pretraining?



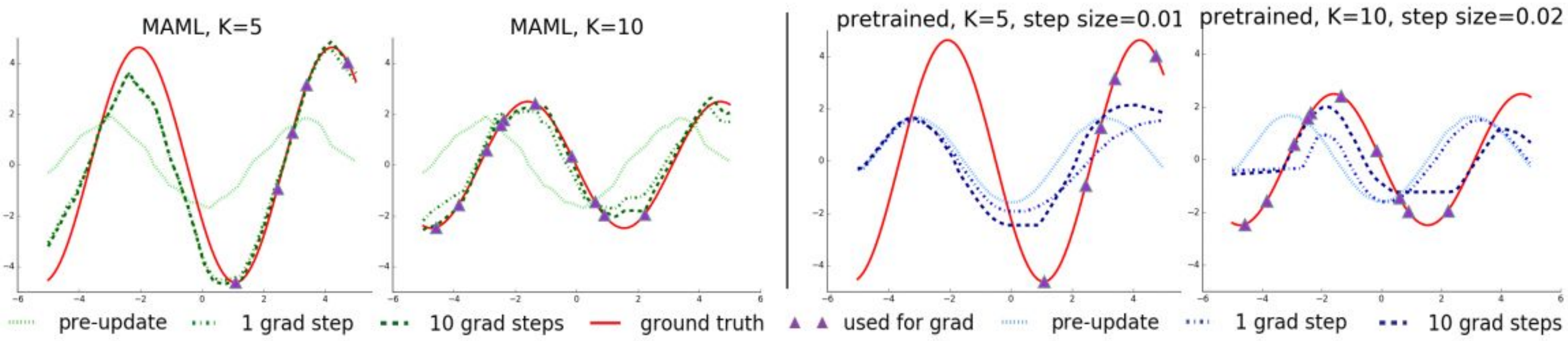
MAML Adapts better

Quick empirical validation: given two sine curves, how well can we adapt a model given some pretraining?



MAML Adapts better

Quick empirical validation: given two sine curves, how well can we adapt a model given some pretraining?

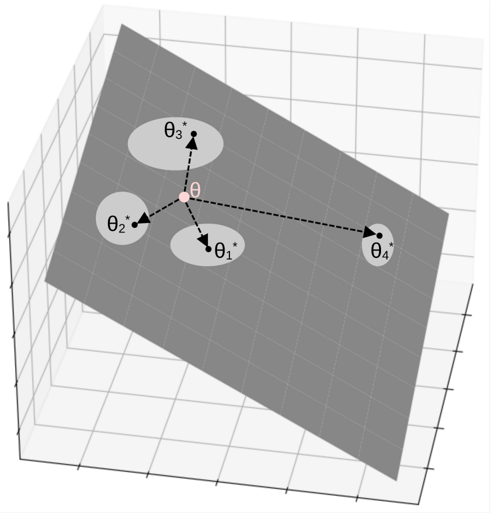


…And lots of experiments in the paper on various settings (few-shot learning, RL)

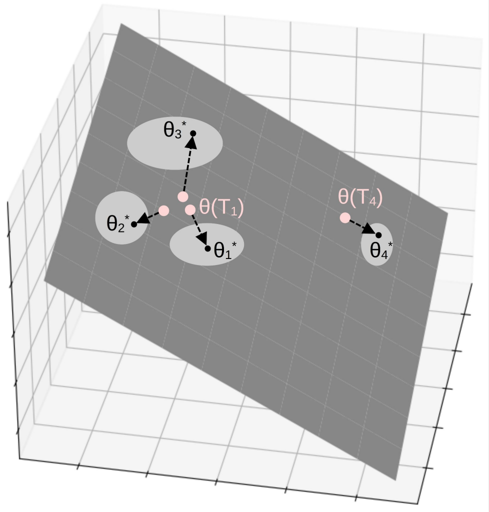
Motivation: better generalization across a semi-heterogeneous task set

Hypothesis:

- MAML "underfits": it attempts to identify a one-size-fits-all initial θ
- Initializing differently for each task may enable better performance on unseen tasks



MAML

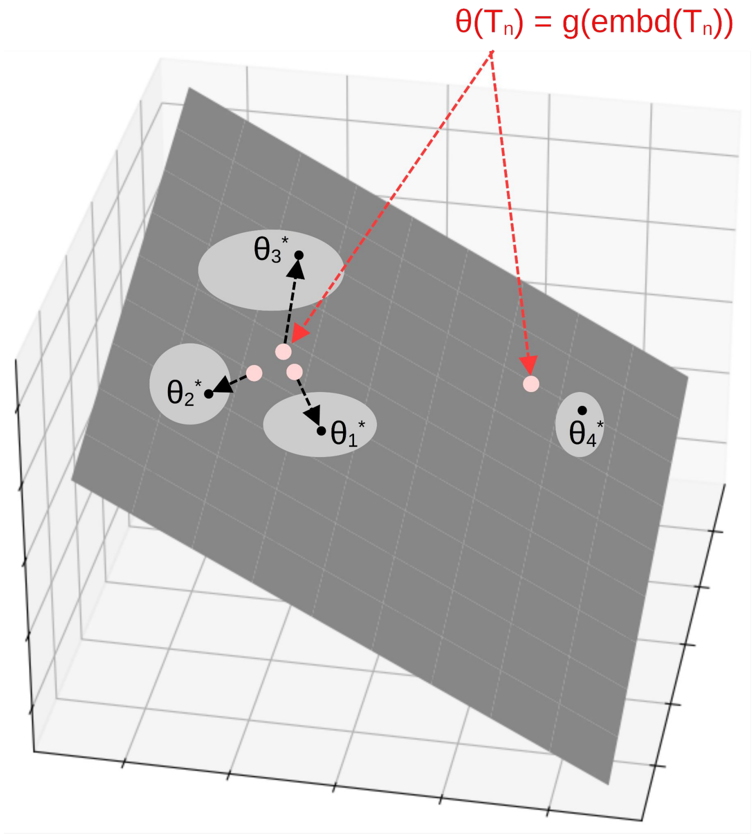


Better?

Proposal / Core Idea

- Learn θ generator g
- Input: embedded task description
- g trained using same process as MAML, but backpropagation continues backwards through θ and g.

<u>Hypothesis</u>: better scaling than a hypernetwork attempting to directly generate $\theta^{*}(T_n)$, and better training performance than MAML.



Algorithm-learning

Algorithm 2 Hyper MAML for Reinforcement Learning

Require: distribution over tasks $p(\mathcal{T})$, embedded task description $embd(\mathcal{T}_i)$
Require: step size for task-specific parameter β_1, step size for model generator (g_ϕ) β_2
  Initialize model generator g_ϕ
  while not done **do**
    Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
    for all $\mathcal{T}_i$ **do**
      Sample K trajectories $\mathcal{D}_i = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ using f_{θ_i} in $\mathcal{T}_i$, $\theta_i = g(embd(\mathcal{T}_i))$
      Evaluate $\nabla_{\theta_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i})$ using $\mathcal{D}_i$ and $\mathcal{T}_i$
      Compute adapted parameters with gradient descent:
      $\theta_i' = \theta_i - \beta_1 \nabla_{\theta_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i})$
      Sample trajectories $\mathcal{D}_i' = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ using $f_{\theta_i'}$ in $\mathcal{T}_i$
    end for
    Update model generator:
    $\phi \leftarrow \phi - \beta_2 \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \nabla_{\theta_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i}') \nabla_\phi g(embd(\mathcal{T}_i))$ using each D_i' and $\mathcal{L}_{\mathcal{T}_i}$
  end while

Algorithm-implementation

Test our hyper MAML on the new task T_j

- Require embedded task description $embd(T_j)$
- Generate task-specific initialization $\theta_j = g(embd(T_j))$
- Learning with K-shot

Environment settings

Task: continuous control environments in the rllab benchmark suite

Model: policy (neural network with two hidden layers of size 100, with ReLU nonlinearities)

Model generator: similar neural network with linear output layer

Gradient descent: REINFORCE with a manually tuned step size

Environment Settings

2D navigation:

s: current 2D position, a: clipped velocity, r: negative squared distance,

MAML: 500 meta-iterations, Hyper MAML: >500 iterations

Comparison of adaptation ability

1. Oracle given the test task and fine-tuning (upper bound)
2. Random initialization
3. Conventional pretraining one policy on all of the tasks
4. Original MAML (one-size-fits-all initial θ)

Toy examples

How to define the embedded task description T_i —- **Task Set specific**

- Object grab: a common part of the objectives / a good abstraction of the objectives' shape
- Power systems: classic loads/solar/wind curves (or corresponding predictions) to handle the time-varying physical quantities
- Natural language learning: a general context

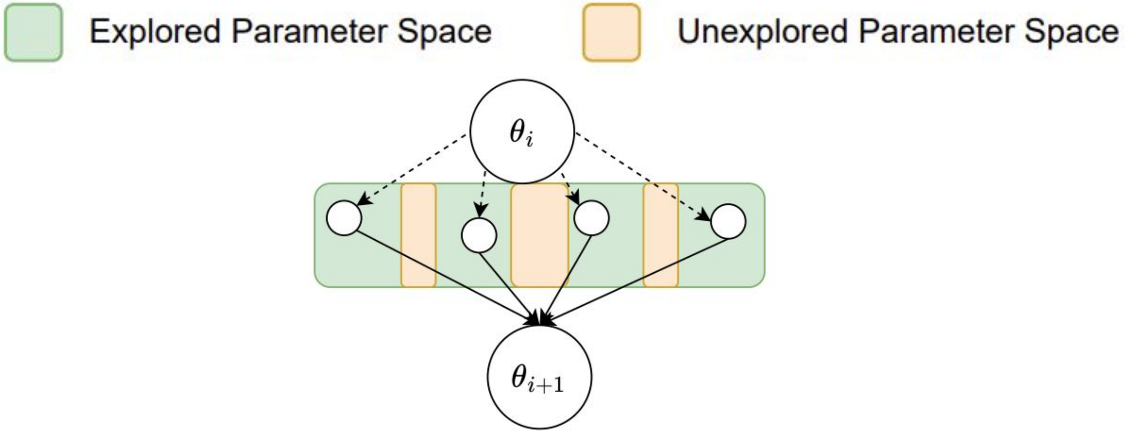
Potential Risks

- Computational Complexity
- Overfitting
- Non-Generalization
- Difficulty in Hyperparameter Tuning
- Scaling Challenges
- Embedding Quality

Downstream Impact

Ideally, suppose that the task set is finite, we can apply this to any current MAML paradigm (i.e: MAML-LLM)

- Green parts are data points sampled from four different tasks (QA, NLI, Paraphrase, Translation)
- Problem: Different tasks may have distinctive context knowledge (i.e: Translation versus QA)

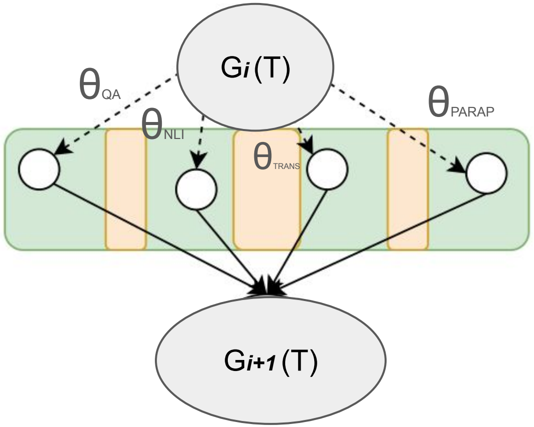


MAML-en-LLM

Downstream Impact

In our framework, instead of applying the same theta on all tasks, generate θ which is conditioning on task T.

- Potential Reliever: Tasks with distinctive context knowledge would have different theta, controlling by the generator. Meanwhile, the task with similar context knowledge would have a similar theta.



HyperMAML-en-LLM

Related Work

- Conditional Meta-Learning

  Latent Embedding Optimization (Rusu et al., 2019): Use task-specific starting point in a low-dimensional space, and perform MAML on it.

  MMAML (Vuorio et al., 2019): Replace the inner-loop gradient step on source tasks with task-specific parameter generation.

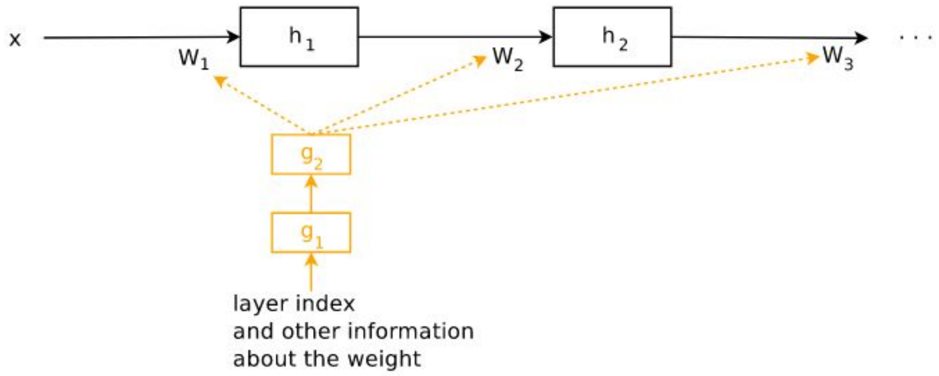
  Weighted MAML (Cai et al., 2020): Use weighted loss of source tasks which depends on target data.

  TASML (Wang et al., 2020): During adaptation on target task, sample the most similar sources tasks and further do MAML on them.

Related Work

- HyperNetworks / Meta Networks

  HyperNetworks (Ha et al., 2016): Use a small network (“hypernetwork") to generate the weights for a larger network (main network).



$$K^j = g(z^j), \quad \forall j = 1, ..., D$$

  Meta Networks (Munkhdalai et al., 2017): Use a meta learner with memory. Both z and g change with the target task information.

Related Work

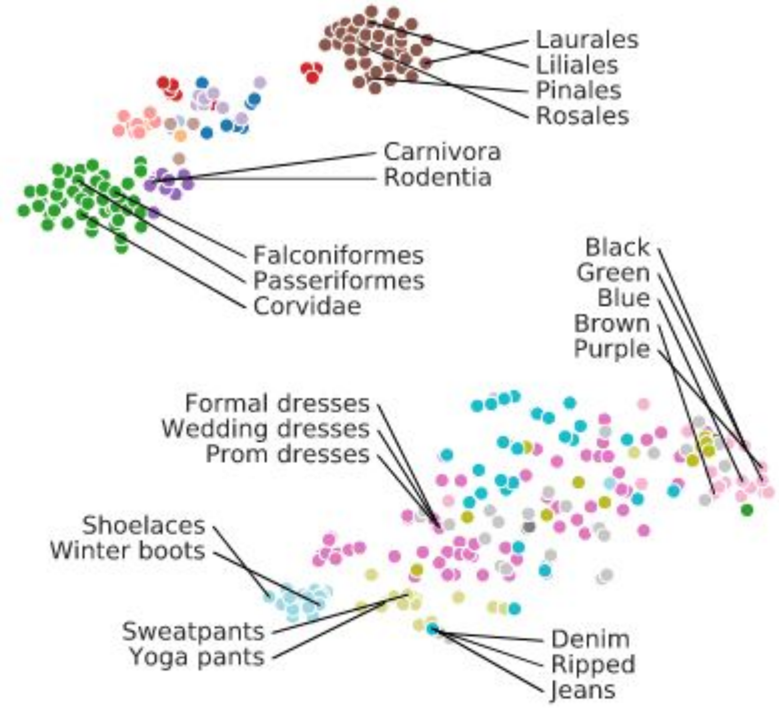
- Task Embedding for Meta-Learning

  Task2Vec (Achille et al., 2019):

  First obtain the task embeddings that contain task structure information.

  Then choose similar feature extractors for similar tasks.

Example: Image Classification



Task Embeddings

References

[1] Achille, Alessandro, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. “Task2Vec: Task Embedding for Meta-Learning.” arXiv, February 10, 2019. https://doi.org/10.48550/arXiv.1902.03545.

[2] Antoniou, Antreas, Harrison Edwards, and Amos Storkey. “How to Train Your MAML.” arXiv, March 5, 2019. https://doi.org/10.48550/arXiv.1810.09502.

[3] Cai, Diana, Rishit Sheth, Lester Mackey, and Nicolo Fusi. “Weighted Meta-Learning.” arXiv, March 20, 2020. https://doi.org/10.48550/arXiv.2003.09465.

[4] Finn, Chelsea, Pieter Abbeel, and Sergey Levine. “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.” arXiv, July 18, 2017. https://doi.org/10.48550/arXiv.1703.03400.

[5] Ha, David, Andrew Dai, and Quoc V. Le. “HyperNetworks.” arXiv, December 1, 2016. https://doi.org/10.48550/arXiv.1609.09106.

[6] Munkhdalai, Tsendsuren, and Hong Yu. “Meta Networks.” arXiv, June 8, 2017. https://doi.org/10.48550/arXiv.1703.00837.

[7] Rusu, Andrei A., Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. “Meta-Learning with Latent Embedding Optimization.” arXiv, March 26, 2019. https://doi.org/10.48550/arXiv.1807.05960.

[8] Vuorio, Risto, Shao-Hua Sun, Hexiang Hu, and Joseph J. Lim. “Multimodal Model-Agnostic Meta-Learning via Task-Aware Modulation.” arXiv, October 30, 2019. https://doi.org/10.48550/arXiv.1910.13616.

[9] Wang, Ruohan, Yiannis Demiris, and Carlo Ciliberto. “Structured Prediction for Conditional Meta-Learning.” In Advances in Neural Information Processing Systems, 33:2587–98. Curran Associates, Inc., 2020. https://proceedings.neurips.cc/paper/2020/hash/1b69ebedb522700034547abc5652ffac-Abstract.html.

[10] Sinha, Sanchit, et al. “MAML-En-LLM: Model Agnostic Meta-Training of LLMS for Improved in-Context Learning.” *arXiv.Org*, 19 May 2024, arxiv.org/abs/2405.11446.