Investigating Meta-Learning Algorithms for Low-Resource Natural Language Understanding Tasks

Zi-Yi Dou, Keyi Yu, Antonios Anastasopoulos
Language Technologies Institute, Carnegie Mellon University

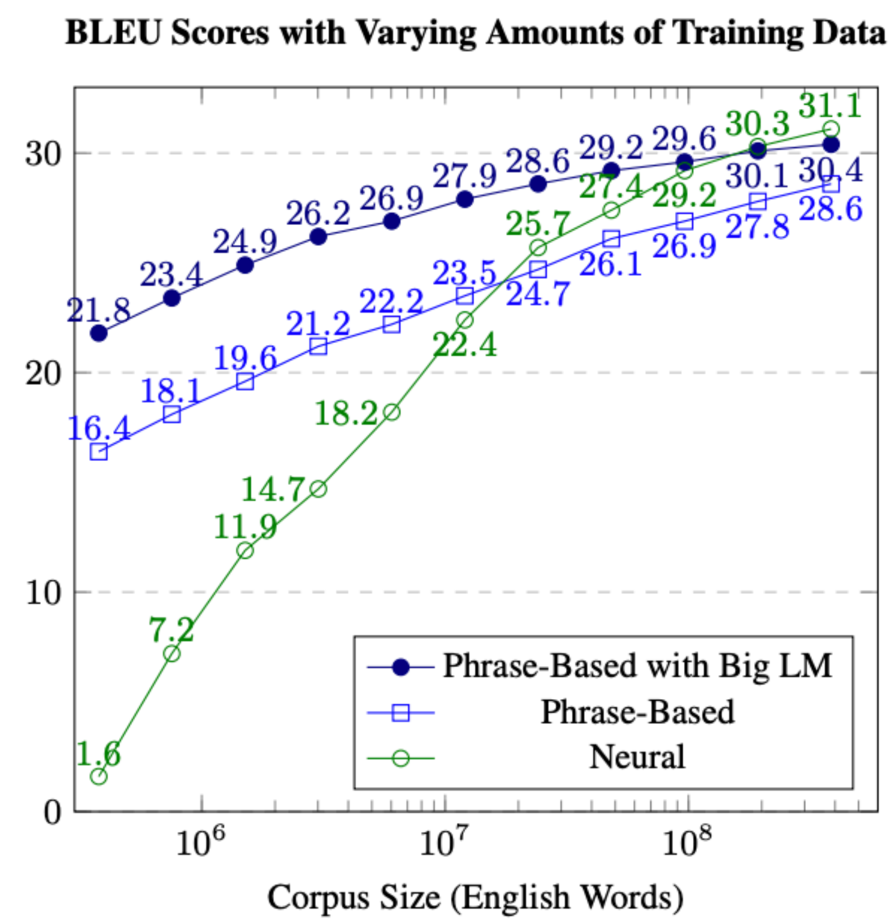
Introduction

Introduction

- Deep neural networks require large amounts of annotated data

Introduction

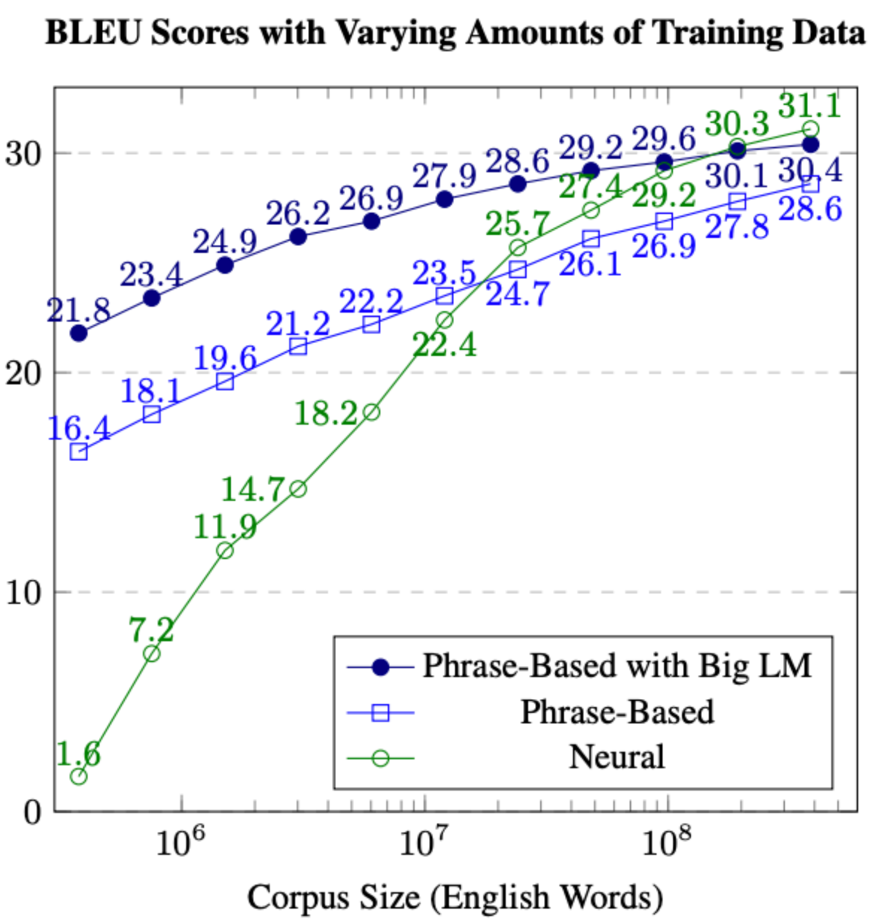
- Deep neural networks require large amounts of annotated data



(Koehn and Knowles, 2017)

Introduction

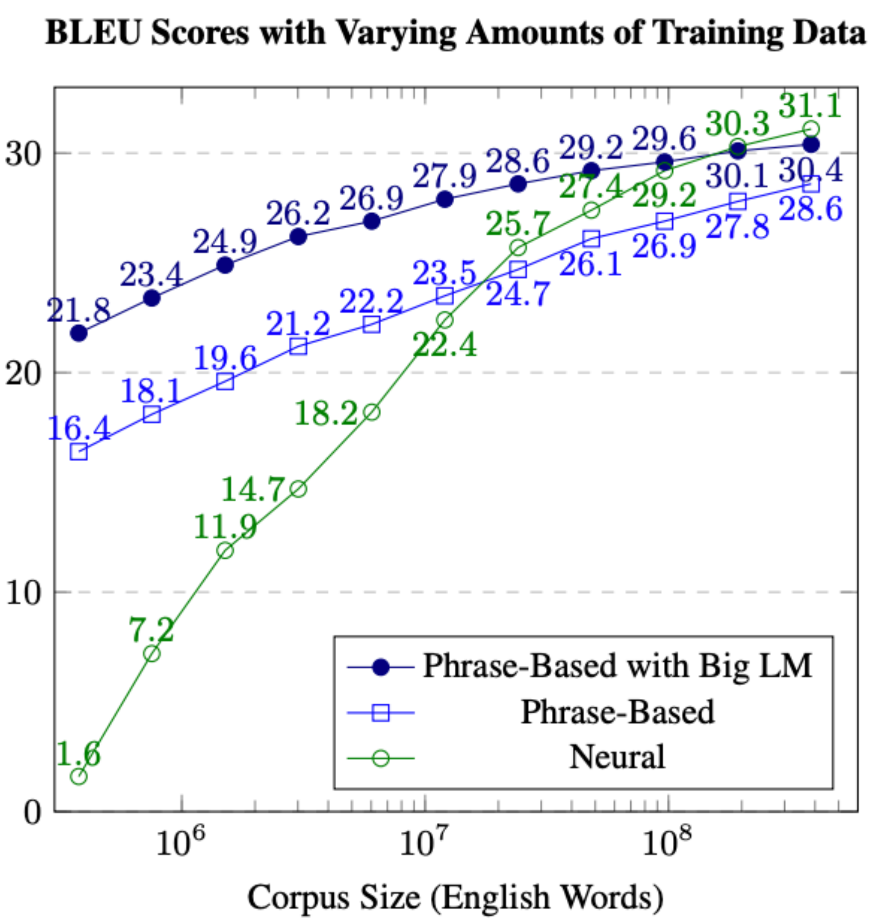
- Deep neural networks require large amounts of annotated data
- However, large amounts of labeled data do not always exist



(Koehn and Knowles, 2017)

Introduction

- Deep neural networks require large amounts of annotated data
- However, large amounts of labeled data do not always exist
- It is essential to develop ways to tackle the scarcity of fully-annotated data



(Koehn and Knowles, 2017)

Related Work

Related Work

- Transfer learning
  - Transfer knowledge from a related task
  - McCann et al. (2017), Peters et al. (2018), …

Related Work

- Transfer learning
  - Transfer knowledge from a related task
  - McCann et al. (2017), Peters et al. (2018), …
- Active learning
  - Actively query for labels of representative/informative examples
  - Zhang et al. (2017), Shen et al. (2018), …

Related Work

- Transfer learning
    - Transfer knowledge from a related task
    - McCann et al. (2017), Peters et al. (2018), …
- Active learning
    - Actively query for labels of representative/informative examples
    - Zhang et al. (2017), Shen et al. (2018), …
- Distant supervision
    - Utilize weakly labeled training data (labeled based on heuristics or rules)
    - Mintz et al. (2009), Luo et al. (2017), …

Related Work

- Transfer learning
    - Transfer knowledge from a related task
    - McCann et al. (2017), Peters et al. (2018), …
- Active learning
    - Actively query for labels of representative/informative examples
    - Zhang et al. (2017), Shen et al. (2018), …
- Distant supervision
    - Utilize weakly labeled training data (labeled based on heuristics or rules)
    - Mintz et al. (2009), Luo et al. (2017), …
- …

Our Goal

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations
- Previous work on text representation learning:

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations

- Previous work on text representation learning:

    - Language model pre-training

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations
- Previous work on text representation learning:
  - Language model pre-training
    - BERT (Devlin et al., 2019), …

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations

- Previous work on text representation learning:

    - Language model pre-training

        - BERT (Devlin et al., 2019), …

    - Multi-task learning

Our Goal

- Our goal is to do fast adaptation by learning robust and general text representations
- Previous work on text representation learning:
  - Language model pre-training
    - BERT (Devlin et al., 2019), …
  - Multi-task learning
    - MT-DNN (Liu et al., 2019), …

Meta-learning for NLU

Meta-learning for NLU

- Meta-learning, or learning to learn, tries to tackle the problem of *fast adaptation* on new training data

Meta-learning for NLU

- Meta-learning, or learning to learn, tries to tackle the problem of *fast adaptation* on new training data

- In this paper, we adapt several *optimization-based meta-learning algorithms* to NLU tasks

Meta-learning for NLU

- Meta-learning, or learning to learn, tries to tackle the problem of *fast adaptation* on new training data
- In this paper, we adapt several *optimization-based meta-learning algorithms* to NLU tasks
- We first adopt language model pre-training techniques to learn dense representations of texts, then continue to meta-learn robust representations

Training Procedure

Training Procedure

- Our methods can be divided into three stages:

Training Procedure

- Our methods can be divided into three stages:
    - 1. Pre-train the model parameters with unlabeled datasets

Training Procedure

- Our methods can be divided into three stages:
  - 1. Pre-train the model parameters with unlabeled datasets

Training Procedure

- Our methods can be divided into three stages:
  - 1. Pre-train the model parameters with unlabeled datasets 
  - 2. Meta-learn the parameters using MAML (Model-Agnostic Meta Learning) and its variants

Training Procedure

- Our methods can be divided into three stages:
  - 1. Pre-train the model parameters with unlabeled datasets 
  - 2. Meta-learn the parameters using MAML (Model-Agnostic Meta Learning) and its variants
  - 3. Finetune the parameters on the target task

Training Procedure

- Our methods can be divided into three stages:
  - 1. Pre-train the model parameters with unlabeled datasets

  - 2. Meta-learn the parameters using MAML (Model-Agnostic Meta Learning) and its variants
  - 3. Finetune the parameters on the target task

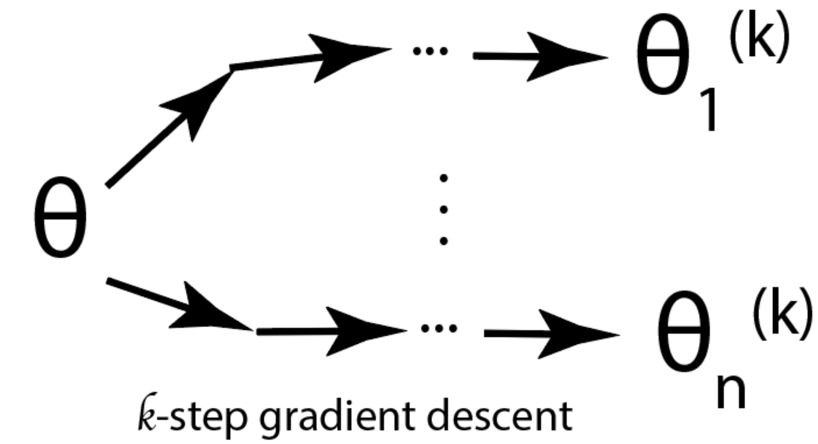
Meta-learning Stage

Meta-learning Stage

Sample batch of tasks $\{T_i\} \sim p(T)$
for all T_i **do**
  Compute adapted parameters $\theta_i^{(k)}$ with gradient descent.
end for

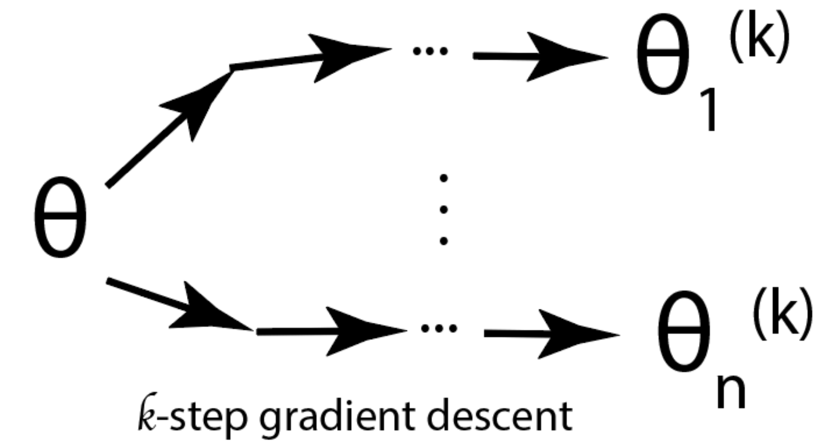
Meta-learning Stage

Sample batch of tasks $\{T_i\} \sim p(T)$
for all T_i **do**
  Compute adapted parameters $\theta_i^{(k)}$ with gradient descent.
end for



Inner loop

Meta-learning Stage

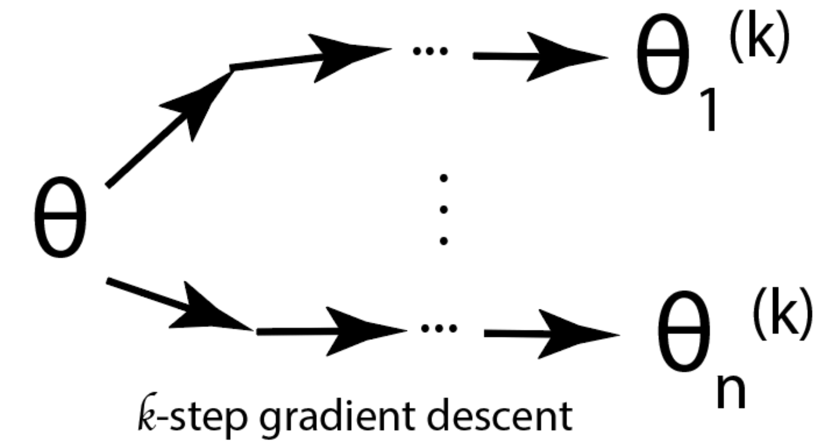


Inner loop

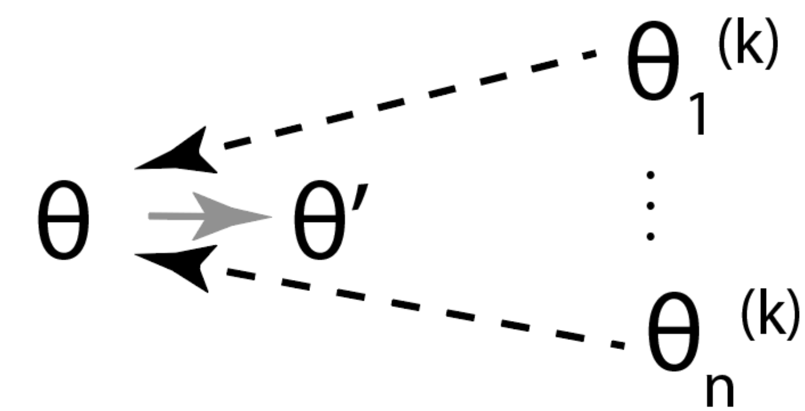
Sample batch of tasks $\{T_i\} \sim p(T)$
for all T_i **do**
  Compute adapted parameters $\theta_i^{(k)}$ with gradient descent.
end for
Update θ with $\theta = \text{MetaUpdate}(\theta; \{\theta_i^{(k)}\})$.

Meta-learning Stage

Sample batch of tasks $\{T_i\} \sim p(T)$
for all T_i **do**
  Compute adapted parameters $\theta_i^{(k)}$ with gradient descent.
end for
Update θ with $\theta = \text{MetaUpdate}(\theta; \{\theta_i^{(k)}\})$.



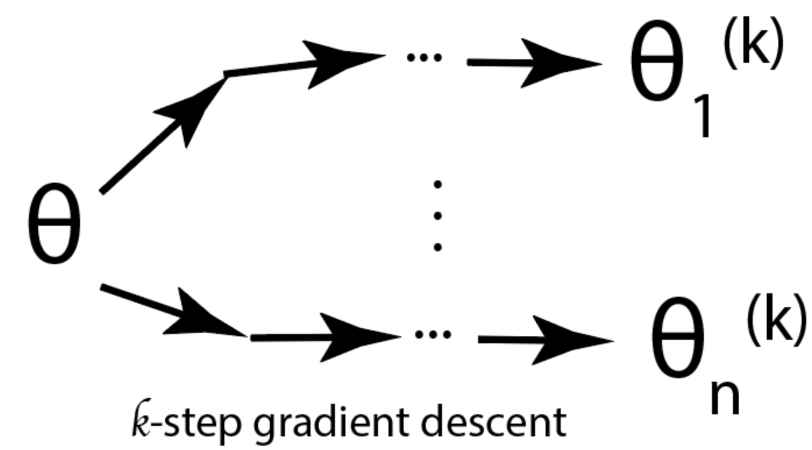
Inner loop



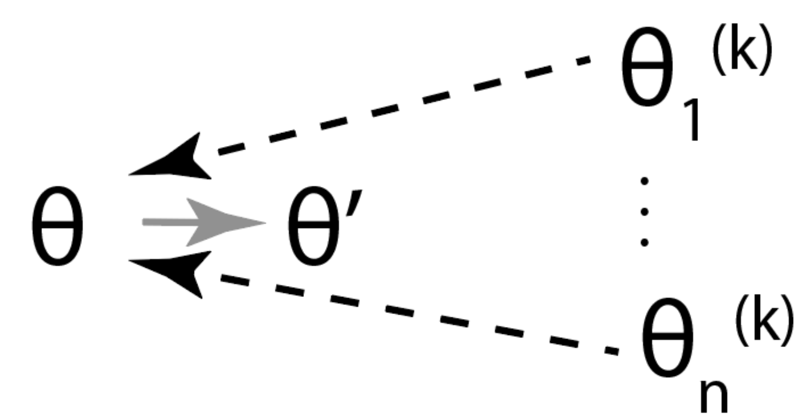
MetaUpdate

Meta-learning Stage

Sample batch of tasks $\{T_i\} \sim p(T)$
for all T_i **do**
  Compute adapted parameters $\theta_i^{(k)}$ with gradient descent.
end for
Update θ with $\theta = \text{MetaUpdate}(\theta; \{\theta_i^{(k)}\})$.



Inner loop



MetaUpdate

The meta-learning algorithms used in this paper just differ in the MetaUpdate step.

MAML

- Model-agnostic Meta Learning (MAML; Finn et al., 2017)
    - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

    - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

MAML

- Model-agnostic Meta Learning (MAML; Finn et al., 2017)

  - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

  - The MetaUpdate Step

$$\theta \xrightarrow[\text{update}]{\text{inner}} \theta_i^{(k)}$$

$$\theta = \theta - \beta \sum_{T_i} \nabla_\theta L_i(f_{\theta_i^{(k)}})$$

MAML

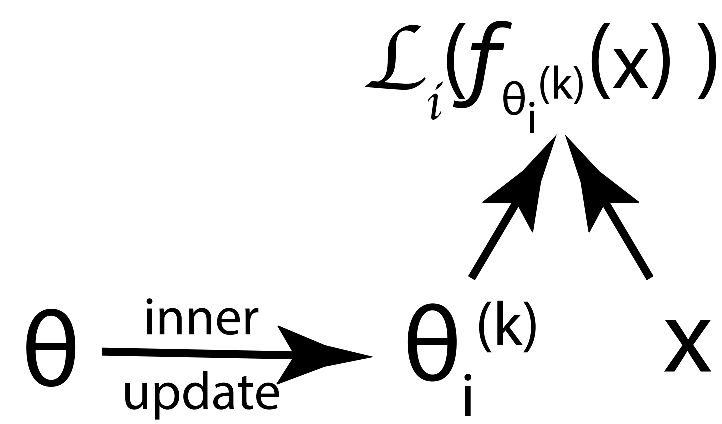
- Model-agnostic Meta Learning (MAML; Finn et al., 2017)

    - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

    - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$



MAML

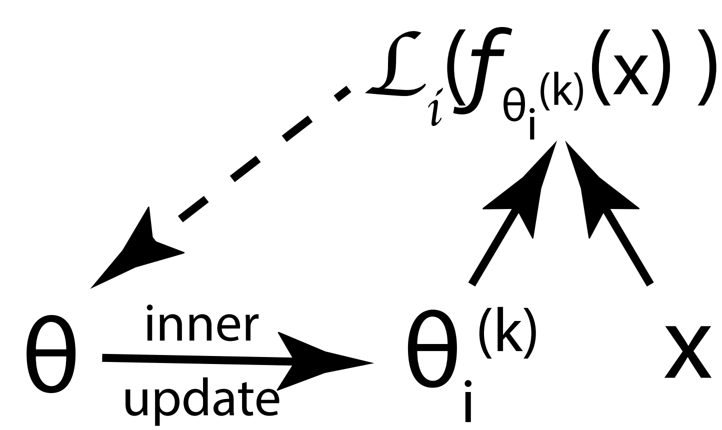
- Model-agnostic Meta Learning (MAML; Finn et al., 2017)

  - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_\theta L_i(f_{\theta_i^{(k)}})$$



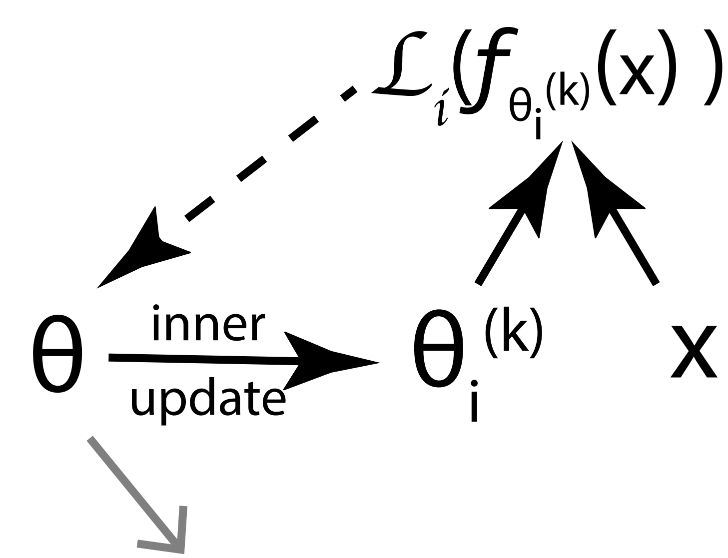
MAML

- Model-agnostic Meta Learning (MAML; Finn et al., 2017)
  - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_\theta L_i(f_{\theta_i^{(k)}})$$



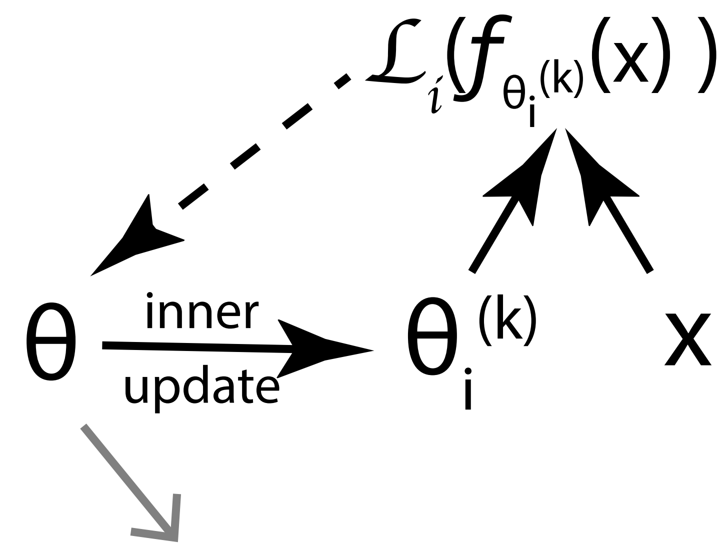
MAML

- Model-agnostic Meta Learning (MAML; Finn et al., 2017)
  - Objective function

$$\min_{\theta} \sum_{T_i \sim p(T)} L_i(f_{\theta_i^{(k)}})$$

  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_\theta L_i(f_{\theta_i^{(k)}})$$



Involve computing second-order derivatives

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
  - The MetaUpdate Step

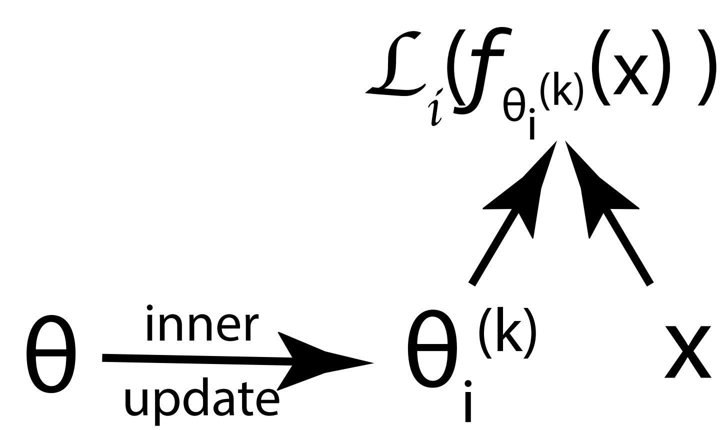
$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$

Reminder: MAML $$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
    - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



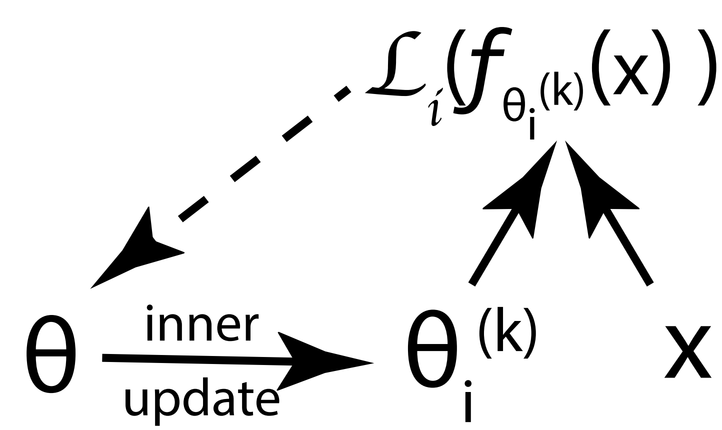
Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



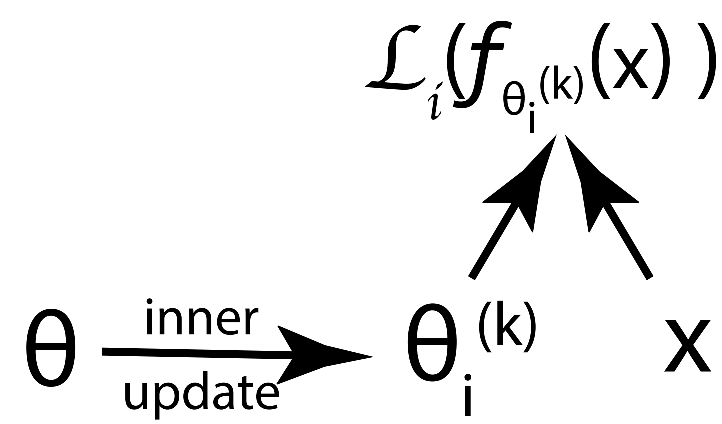
Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



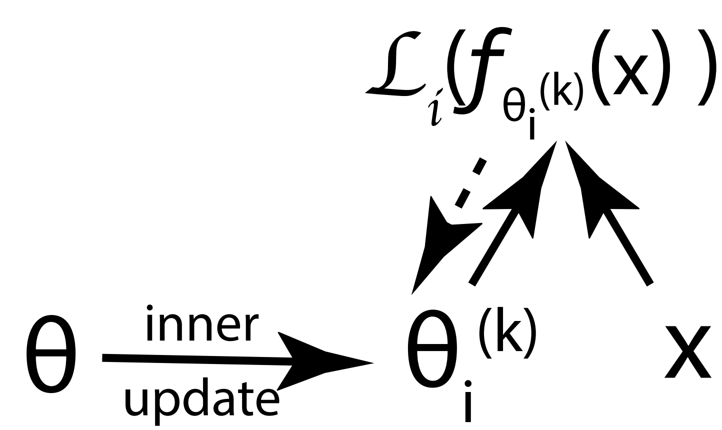
Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



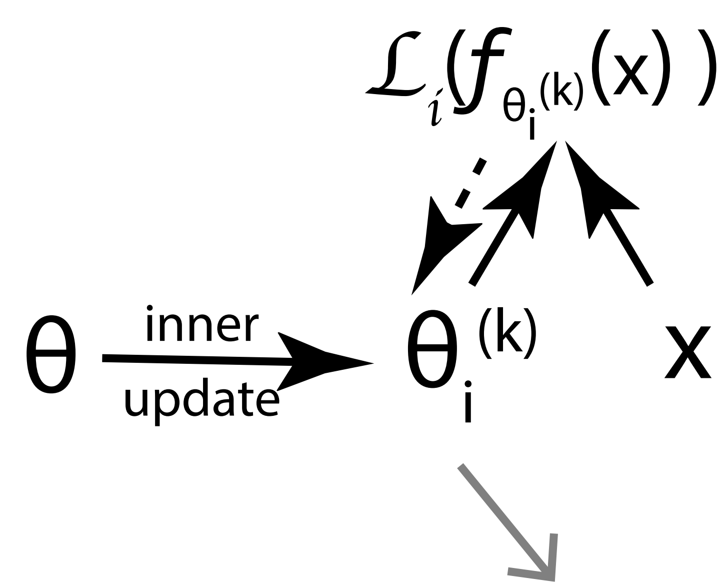
Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

First-order MAML

- Computing second-order derivatives can be computationally and memory intensive
- First-order MAML
  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

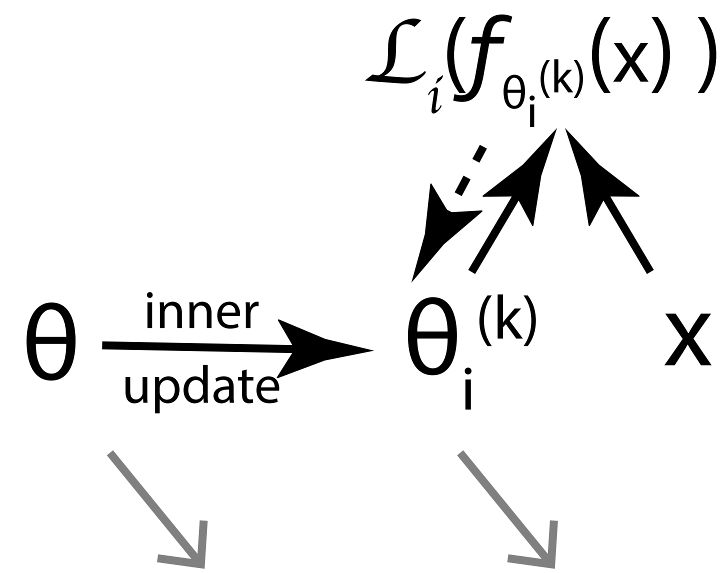
First-order MAML

- Computing second-order derivatives can be computationally and memory intensive

- First-order MAML

  - The MetaUpdate Step

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta_i^{(k)}} L_i(f_{\theta_i^{(k)}})$$



Reminder: MAML

$$\theta = \theta - \beta \sum_{T_i} \nabla_{\theta} L_i(f_{\theta_i^{(k)}})$$

Reptile

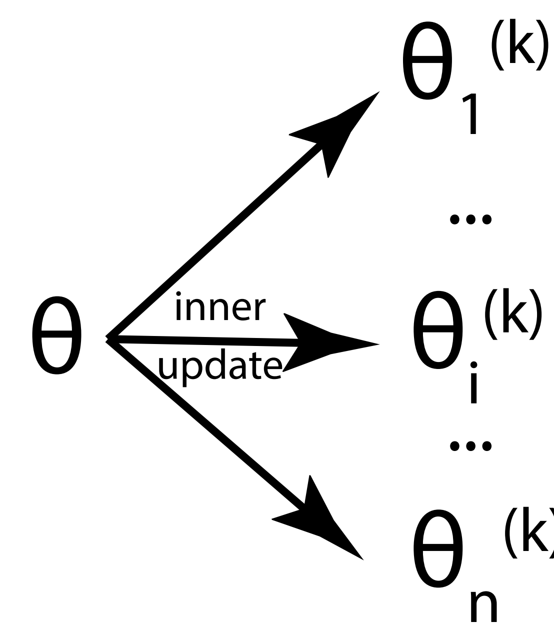
- Reptile (Nichol et al., 2018)
  - Another first-order algorithm
  - The MetaUpdate Step

$$\theta = \theta + \beta \frac{1}{|\{T_i\}|} \sum_{T_i} (\theta_i^{(k)} - \theta)$$

Similar to joint training

Reptile

- Reptile (Nichol et al., 2018)
  - Another first-order algorithm
  - The MetaUpdate Step

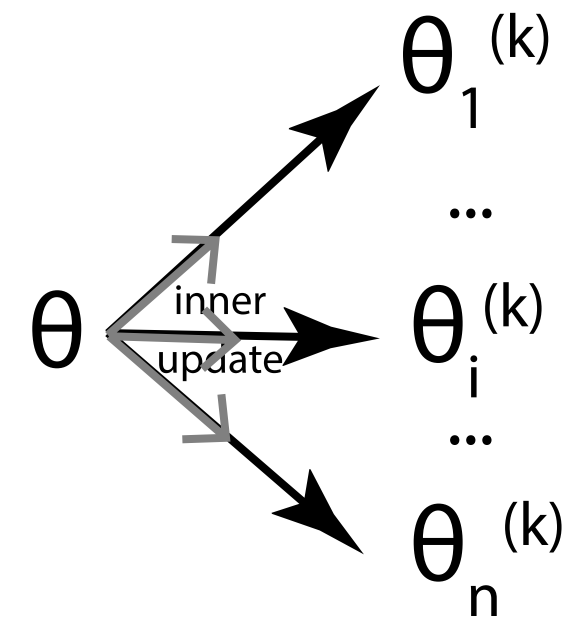


$$\theta = \theta + \beta \frac{1}{|\{T_i\}|} \sum_{T_i} (\theta_i^{(k)} - \theta)$$

Similar to joint training

Reptile

- Reptile (Nichol et al., 2018)
  - Another first-order algorithm
  - The MetaUpdate Step



$$\theta = \theta + \beta \frac{1}{|\{T_i\}|} \sum_{T_i} (\theta_i^{(k)} - \theta)$$

Similar to joint training

Training procedure

Training procedure

- A review of our training procedure:

Training procedure

- A review of our training procedure:
  - 1. Pre-train the model parameters with unlabeled datasets

Training procedure

- A review of our training procedure:
  - 1. Pre-train the model parameters with unlabeled datasets
  - 2. Meta-learn the parameters using MAML, First-order MAML or Reptile

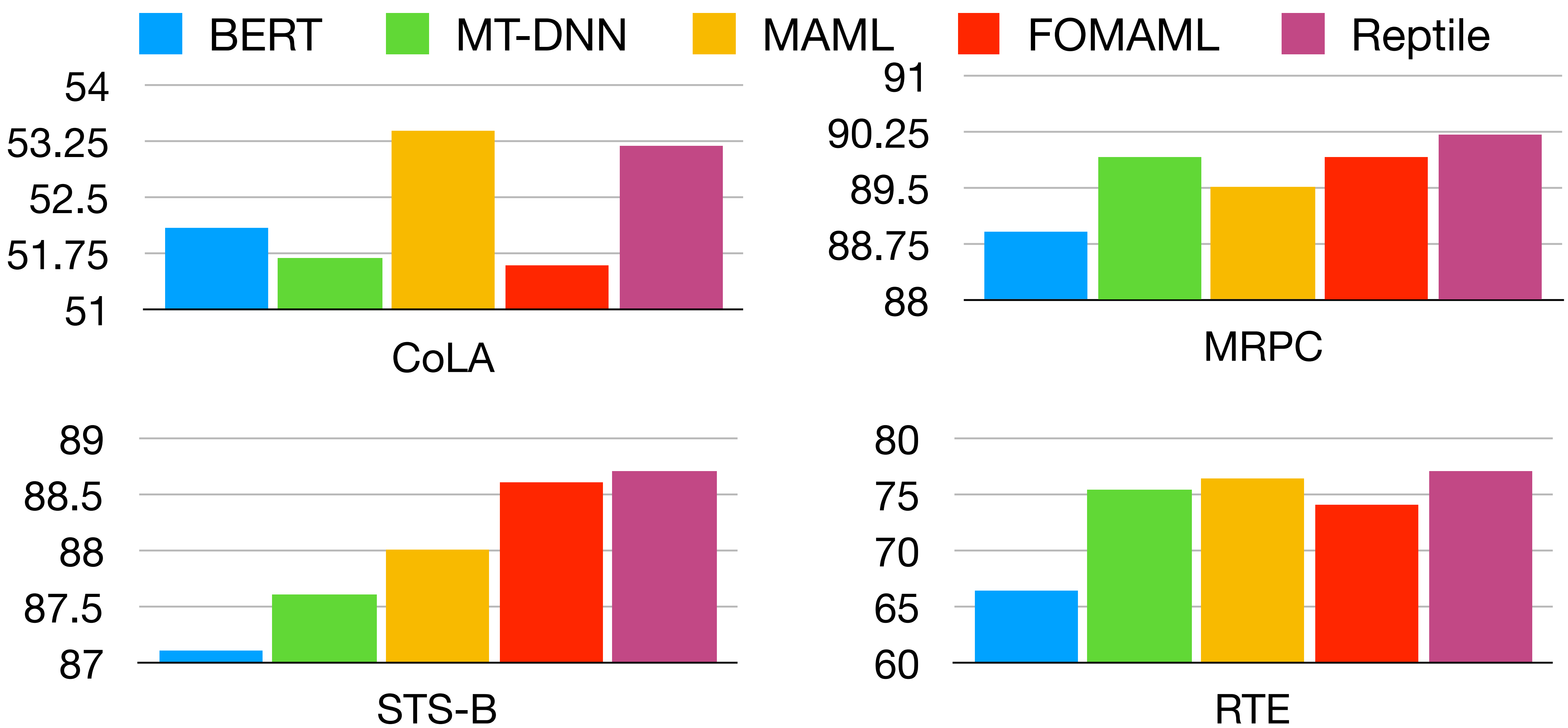
Training procedure

- A review of our training procedure:
  - 1. Pre-train the model parameters with unlabeled datasets 
  - 2. Meta-learn the parameters using MAML, First-order MAML or Reptile
  - 3. Finetune the parameters on the target task

Experiment

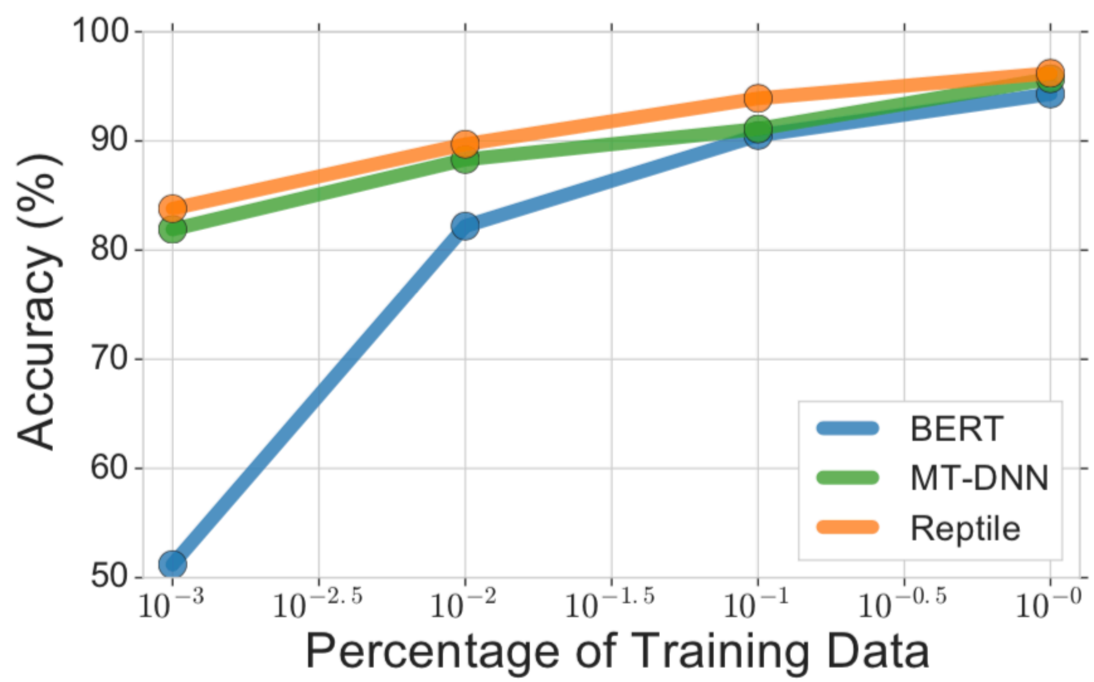
- Datasets
  - GLUE benchmark (Wang et al., 2019)
    - Auxiliary tasks: SST-2, QQP, MNLI, QNLI
    - Target Tasks: CoLA, MRPC, STS-B, RTE
  - SciTail dataset (Khot et al., 2018)
- Baselines
  - BERT (Devlin et al., 2019)
  - MT-DNN (Liu et al., 2019)

Results

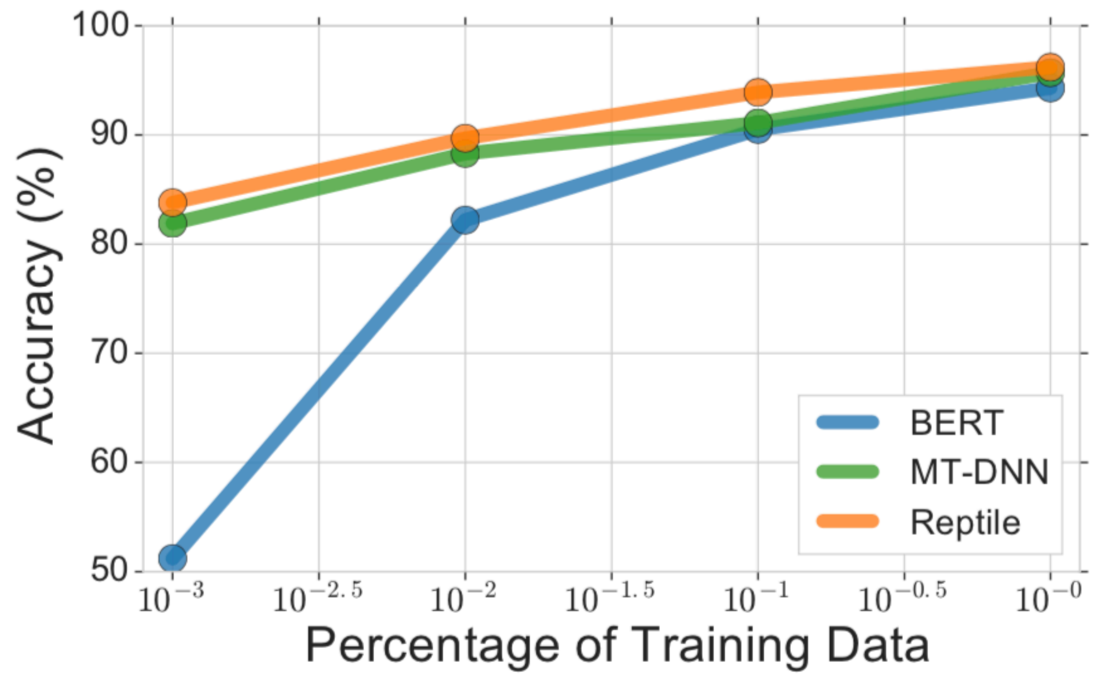


- Generally, the meta-learning algorithms achieve better performance than the baselines
- Reptile performs better than MAML and FOMAML

Fast Adaptation

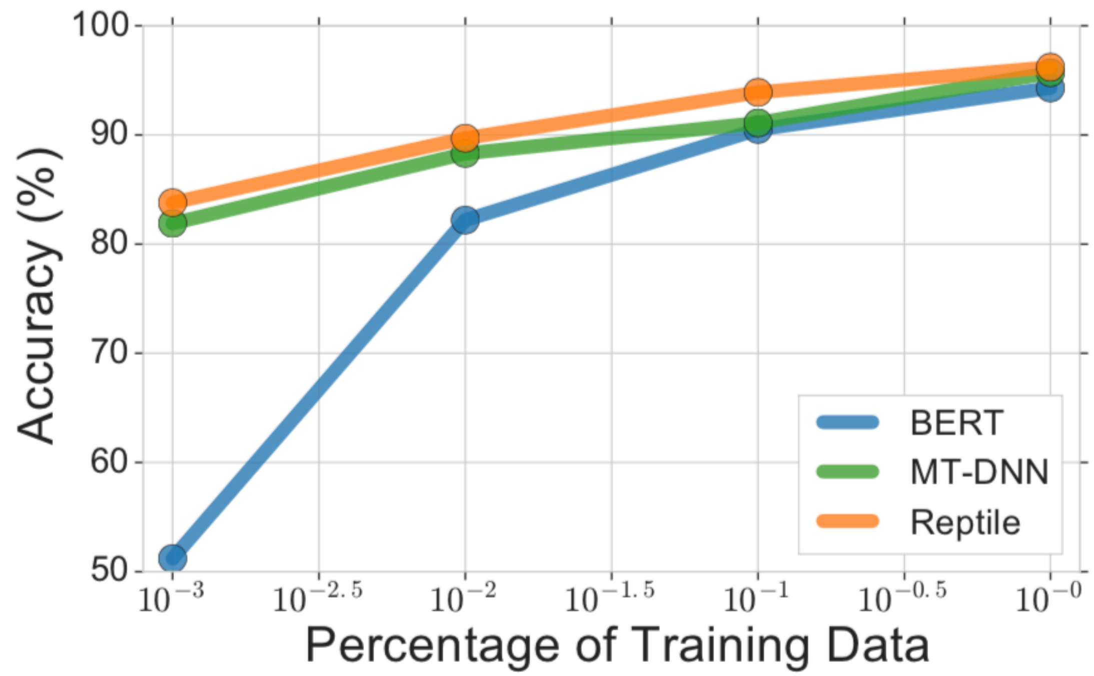


Fast Adaptation



- When adapted to a *completely new task* (SciTail), meta-learning algorithm (Reptile) outperforms MT-DNN and BERT with same amounts of training data

Fast Adaptation



- When adapted to a *completely new task* (SciTail), meta-learning algorithm (Reptile) outperforms MT-DNN and BERT with same amounts of training data

- The meta-learned representations can be adapted to new tasks more *efficiently* compared with other baselines

Conclusion

Conclusion

- In this paper, we adapt three optimization-based meta learning algorithms to natural language understanding tasks

Conclusion

- In this paper, we adapt three optimization-based meta learning algorithms to natural language understanding tasks
- We show the meta-learned representations can be adapted to new tasks more efficiently than other baselines

Conclusion

- In this paper, we adapt three optimization-based meta learning algorithms to natural language understanding tasks

- We show the meta-learned representations can be adapted to new tasks more efficiently than other baselines

- In the future, we want to take the performance of the adapted parameters into consideration during the meta-learning stage

Thank you!