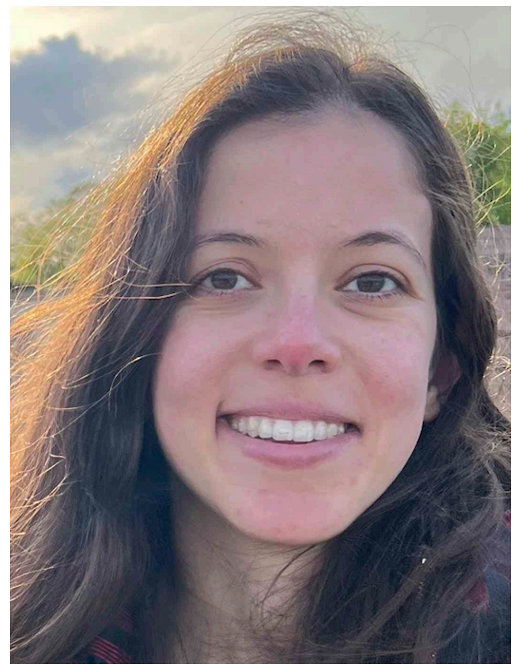
AdapterSoup: Weight Averaging to Improve Generalization of Pretrained Language Models



Alexandra Chronopoulou

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Matthew E. Peters

Ai2

Alexander Fraser

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Jesse Dodge

Ai2

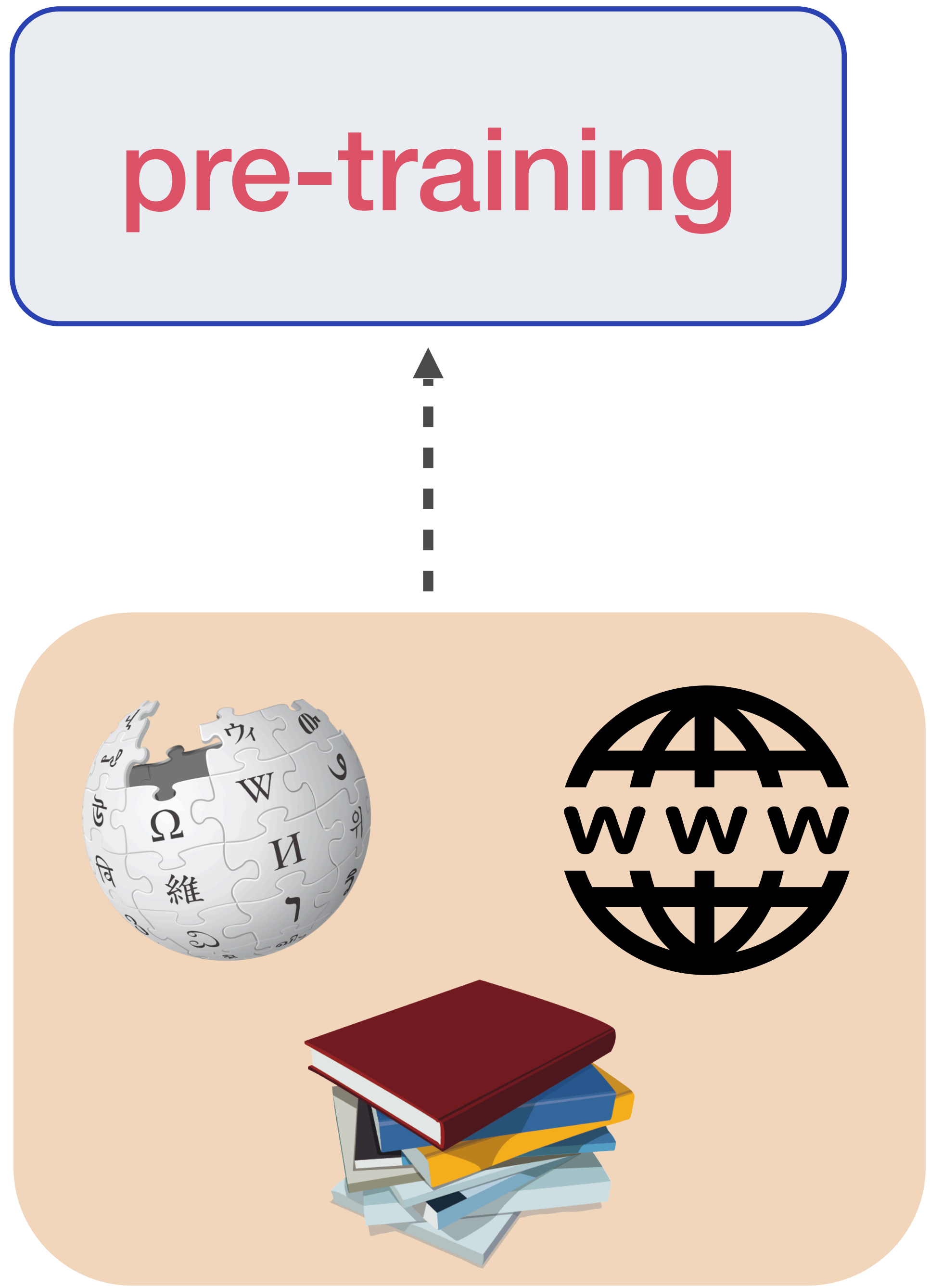
Presentation outline

- Motivation
- Proposed Approach
- Experiments
- Conclusion

- **Motivation**
- Proposed Approach
- Experiments
- Conclusion

Task: adapt a PLM to new domains

Task: adapt a PLM to new domains



Pretrain a model using data from heterogeneous domains

Prior work

- PLMs need adaptation to specific domains *(Han and Eisenstein, 2019; Lee et al., 2020; Gururangan et al., 2020)*

Prior work

- PLMs need adaptation to specific domains *(Han and Eisenstein, 2019; Lee et al., 2020; Gururangan et al., 2020)*

- Efficient adaptation with:
  - Domain-aware MoE *(Gururangan et al., 2022)*
  - Hierarchical domain adapters *(Chronopoulou et al., 2022)*
  - Prompt tuning *(Guo et al., 2022)*

Prior work

- PLMs need adaptation to specific domains *(Han and Eisenstein, 2019; Lee et al., 2020; Gururangan et al., 2020)*

- Efficient adaptation with:
  - Domain-aware MoE *(Gururangan et al., 2022)*
  - Hierarchical domain adapters *(Chronopoulou et al., 2022)*
  - Prompt tuning *(Guo et al., 2022)*

- Averaging the **weights** of fine-tuned models:
  - Model soups *(Wortsman et al., 2022)*
  - Fisher-weight averaging *(Matena and Raffel, 2022)*

Prior work

- PLMs need adaptation to specific domains *(Han and Eisenstein, 2019; Lee et al., 2020; Gururangan et al., 2020)*

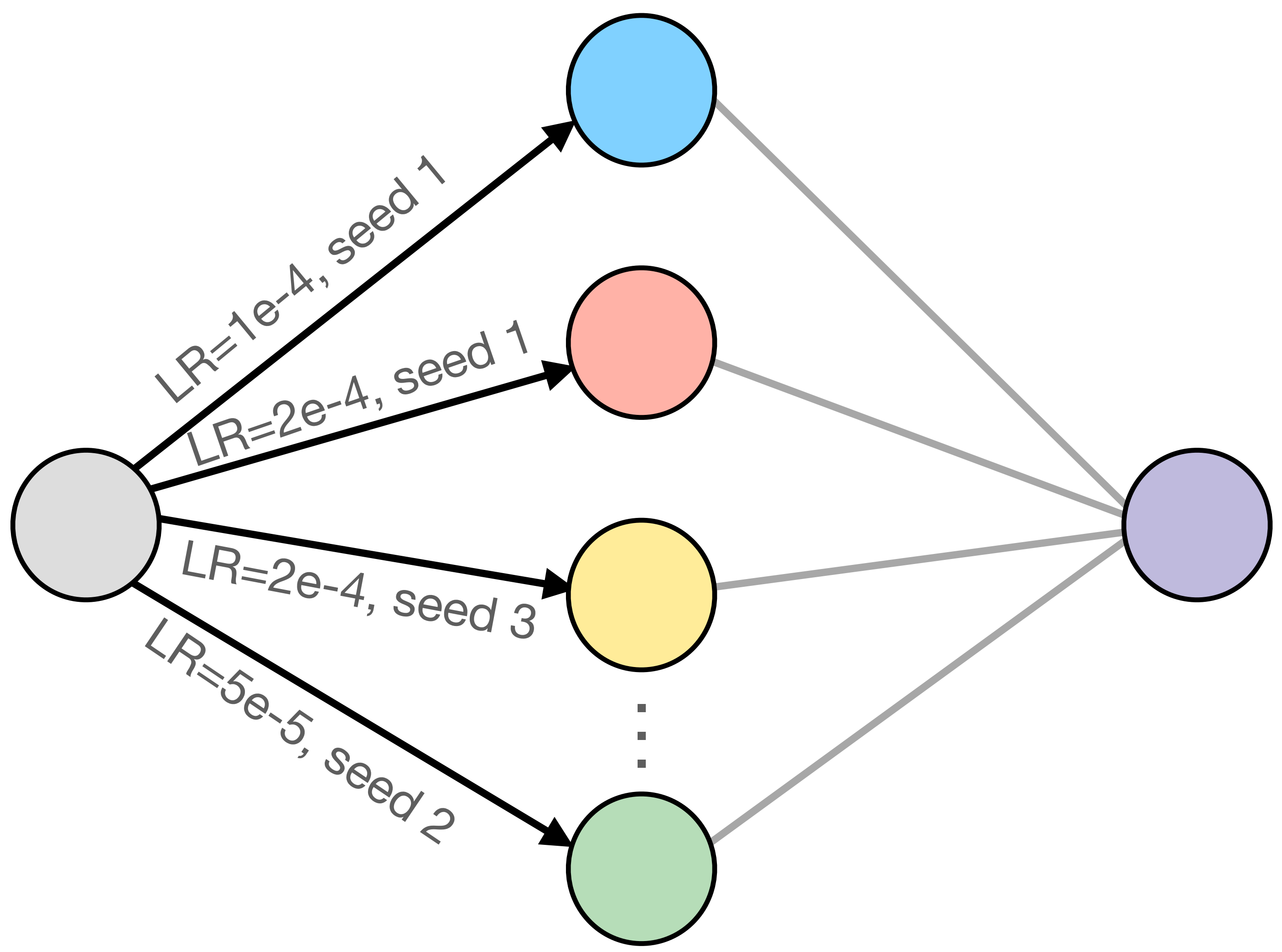
- Efficient adaptation with:
  - Domain-aware MoE *(Gururangan et al., 2022)*
  - Hierarchical domain adapters *(Chronopoulou et al., 2022)*
  - Prompt tuning *(Guo et al., 2022)*

- Averaging the **weights** of fine-tuned models:
  - Model soups *(Wortsman et al., 2022)*
  - Fisher-weight averaging *(Matena and Raffel, 2022)*

Weight-space averaging

Model soups

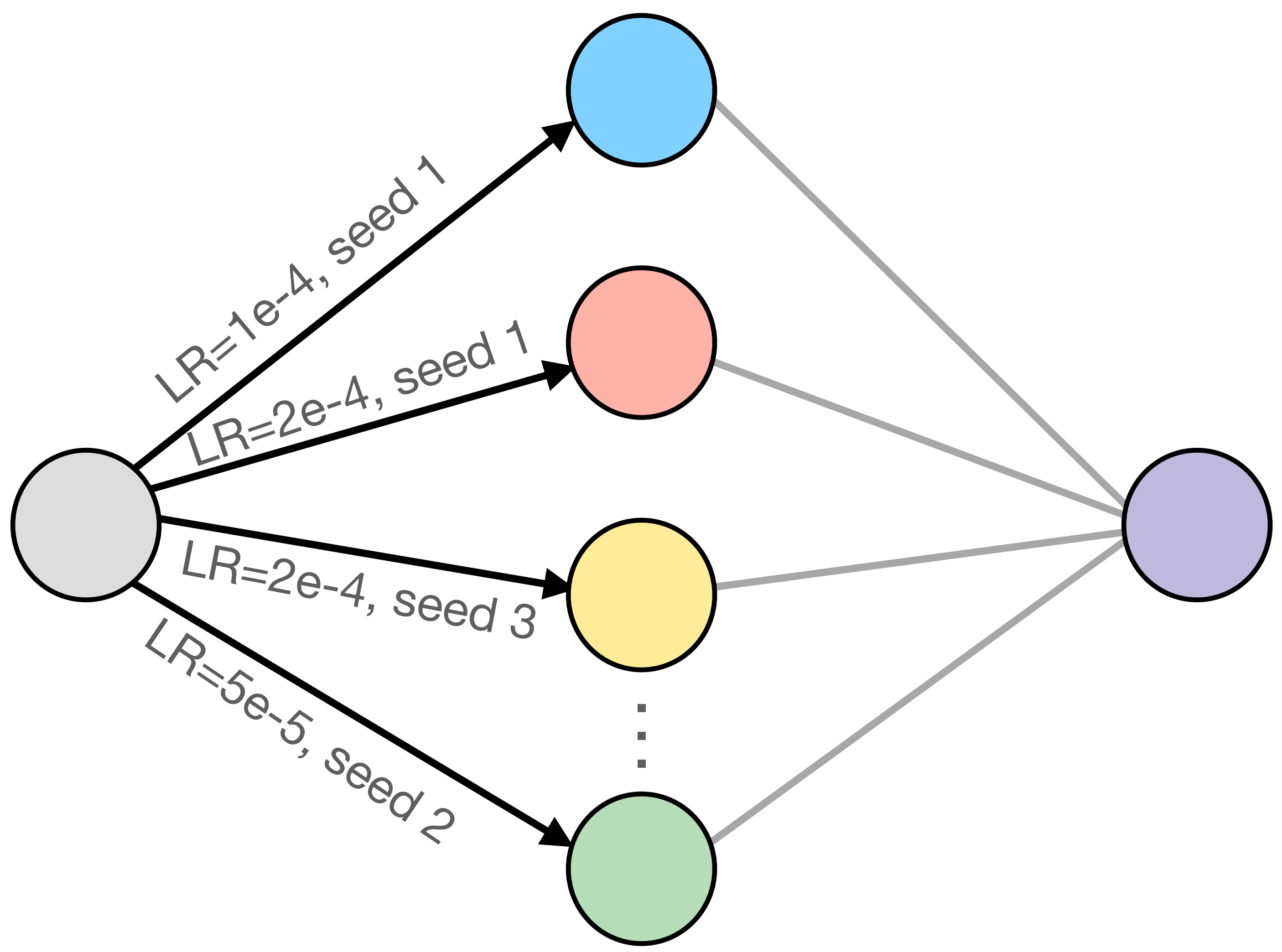
(Wortsman et al., 2022)



Weight-space averaging

Model soups

(Wortsman et al., 2022)



Fisher-weighted averaging

(Matena and Raffel, 2022)

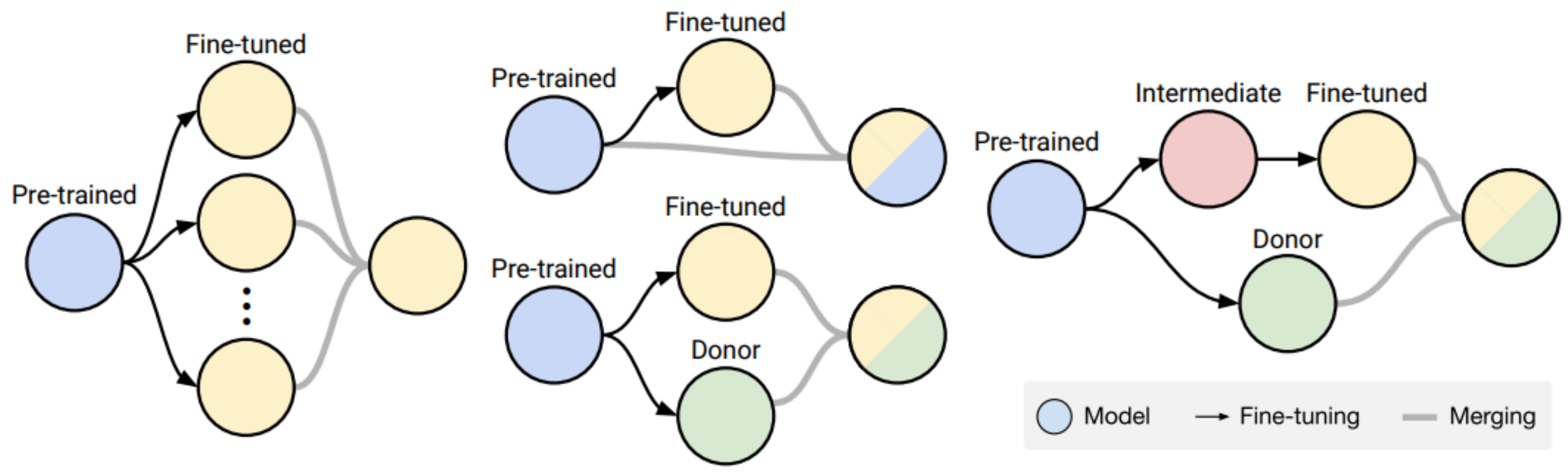
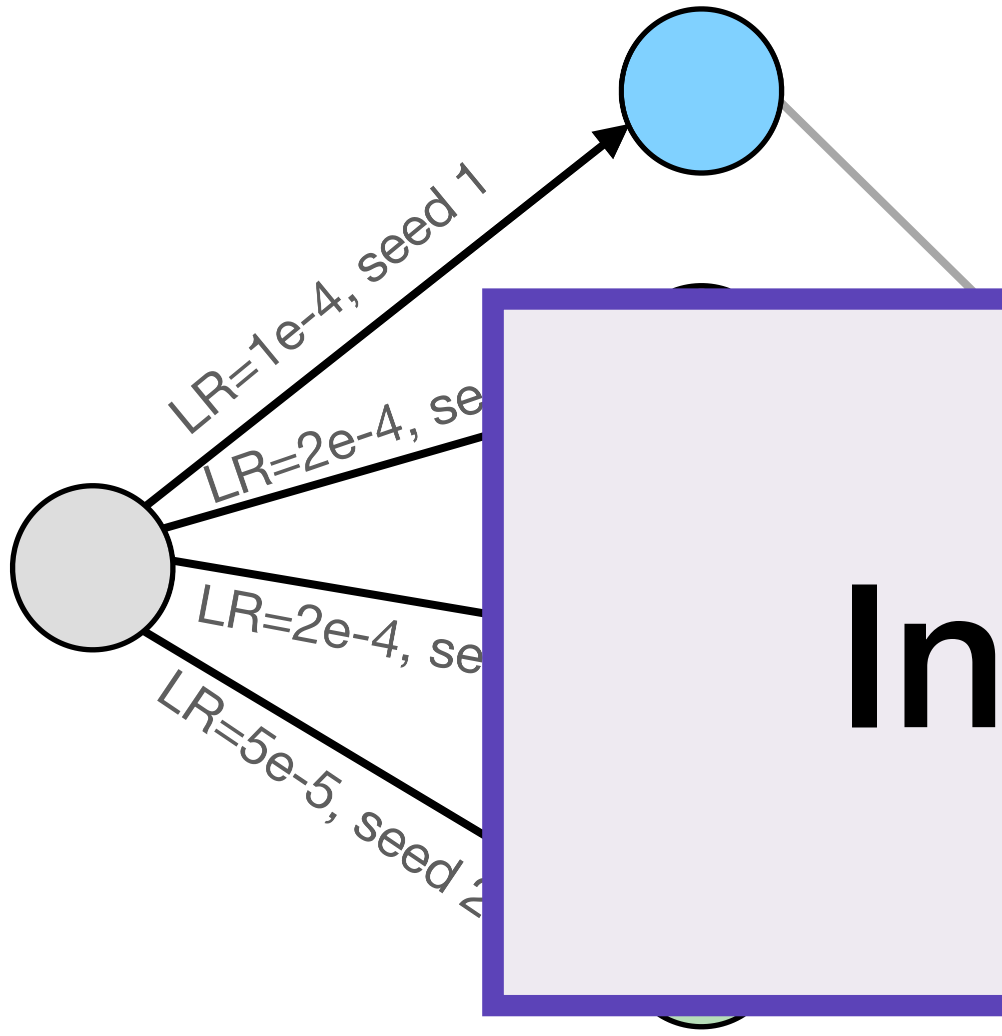


Fig. from Matena and Raffel (2022).

Weight-space averaging

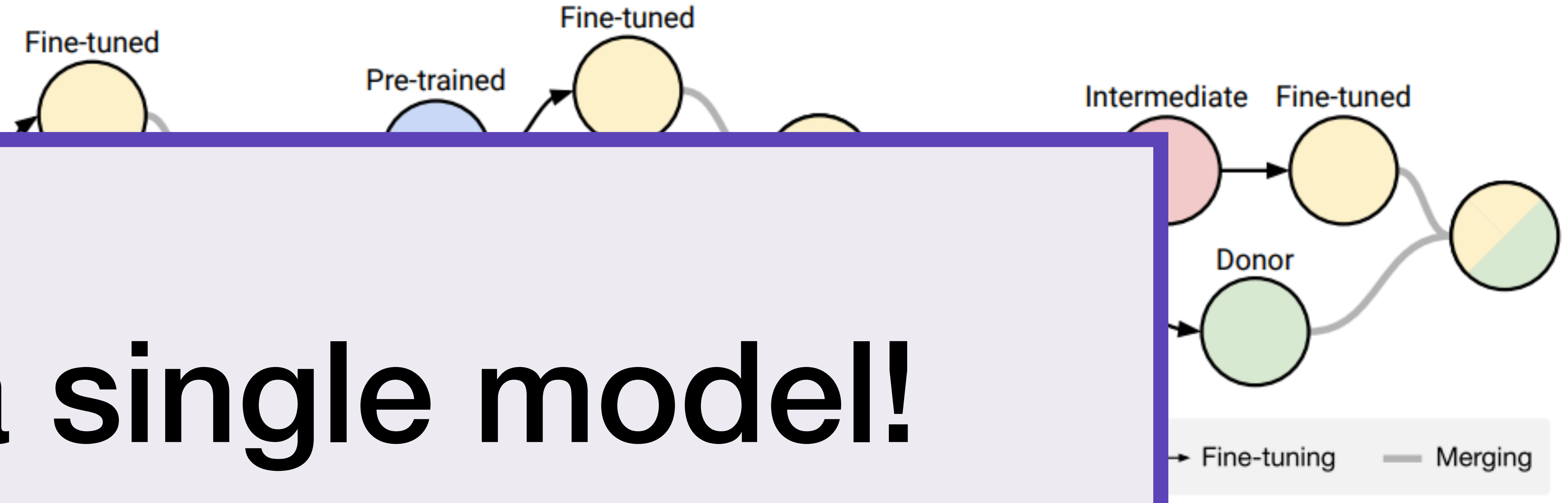
Model soups

(Wortsman et al., 2022)



Fisher-weighted averaging

(Matena and Raffel, 2022)



Inference cost of a single model!

This work

- Can we average the weights of **independently trained adapters** to improve domain generalization of a PLM?

This work

- Can we average the weights of **independently trained adapters** to improve domain generalization of a PLM?

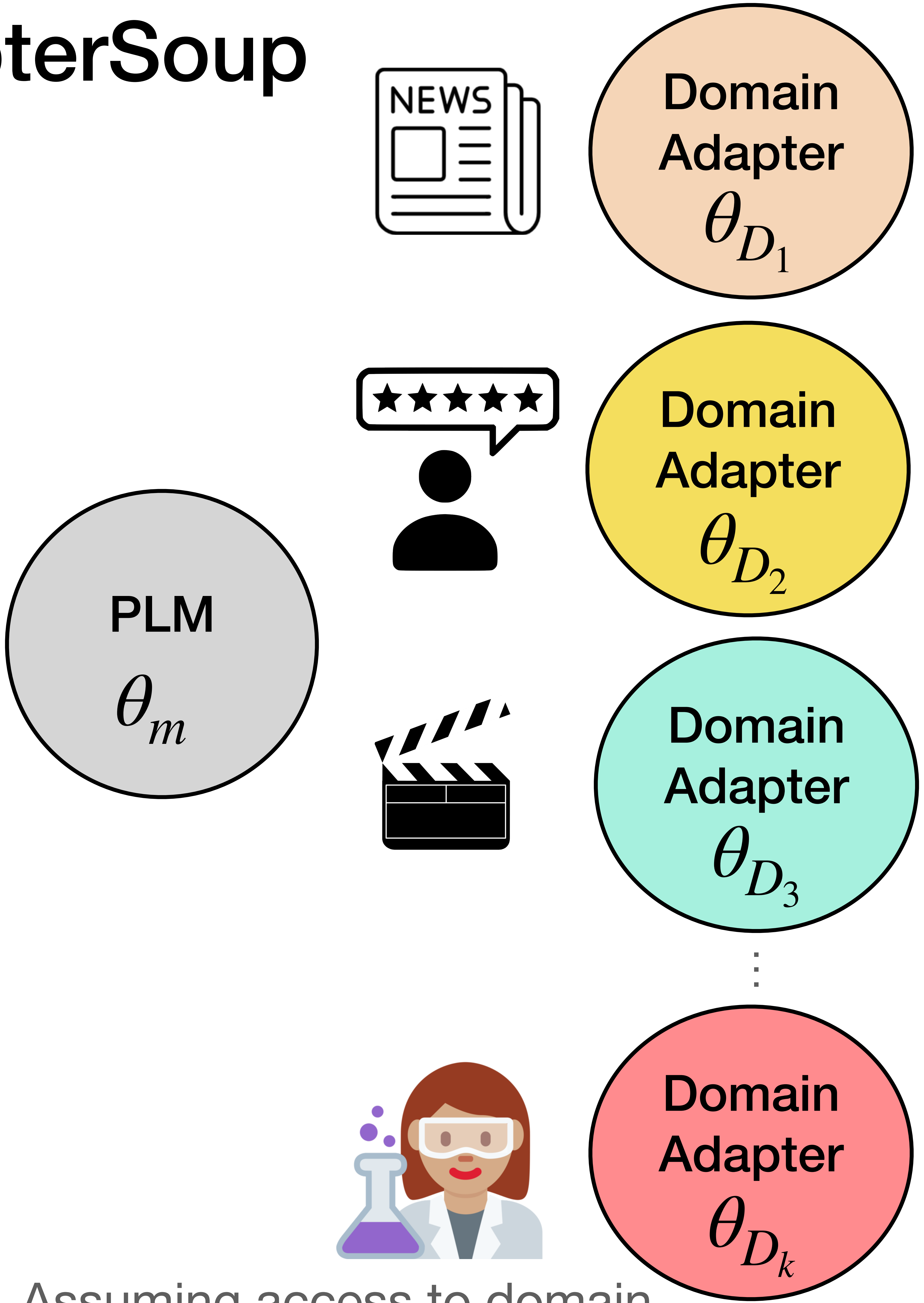
- How to select which adapters to combine?

- Motivation
- **Proposed Approach**
- Experiments
- Conclusion

AdapterSoup

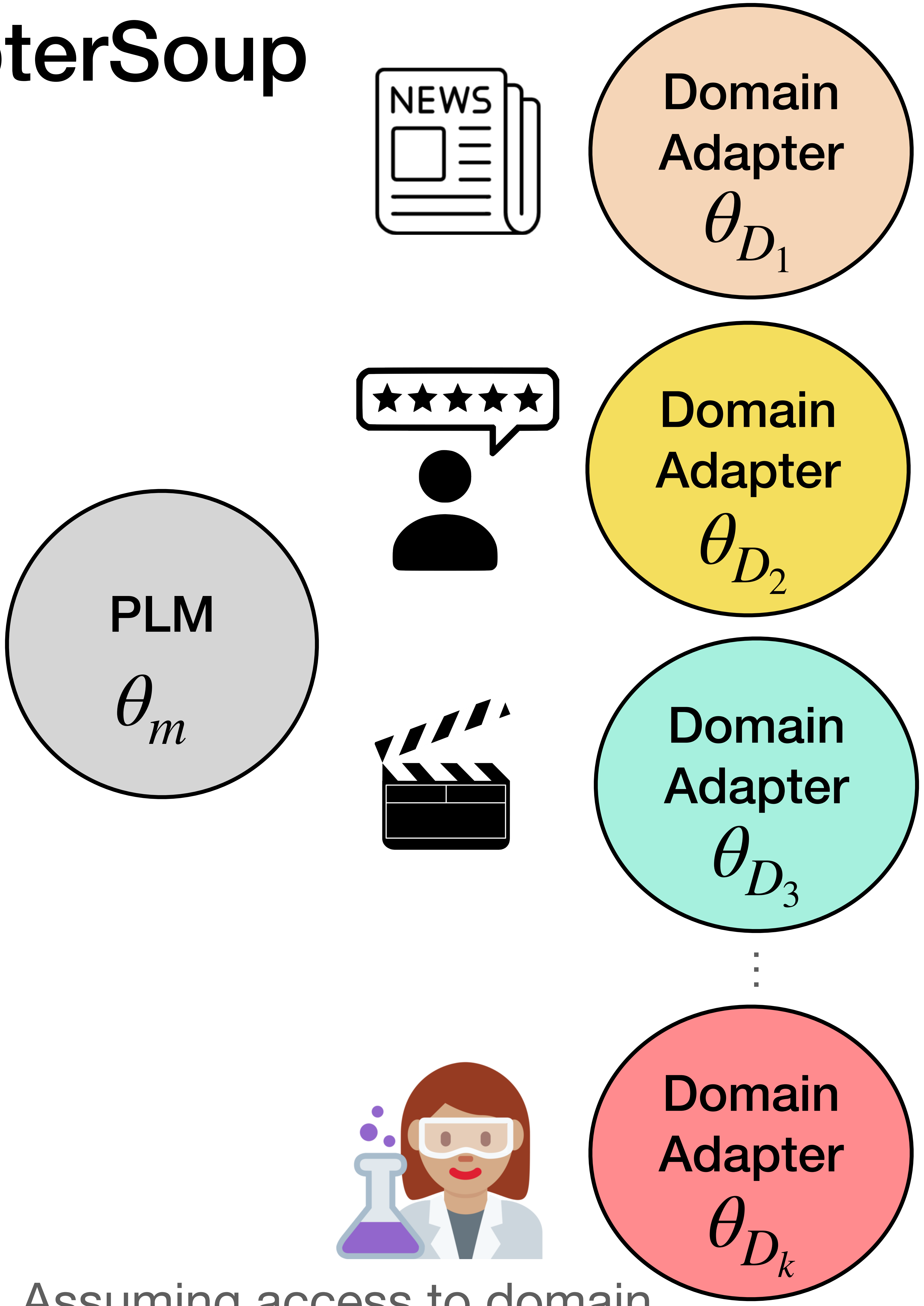
$$AdapterSoup(x) = f(x, \frac{1}{l}\sum_{i=1}^{l} \theta$$

AdapterSoup



Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

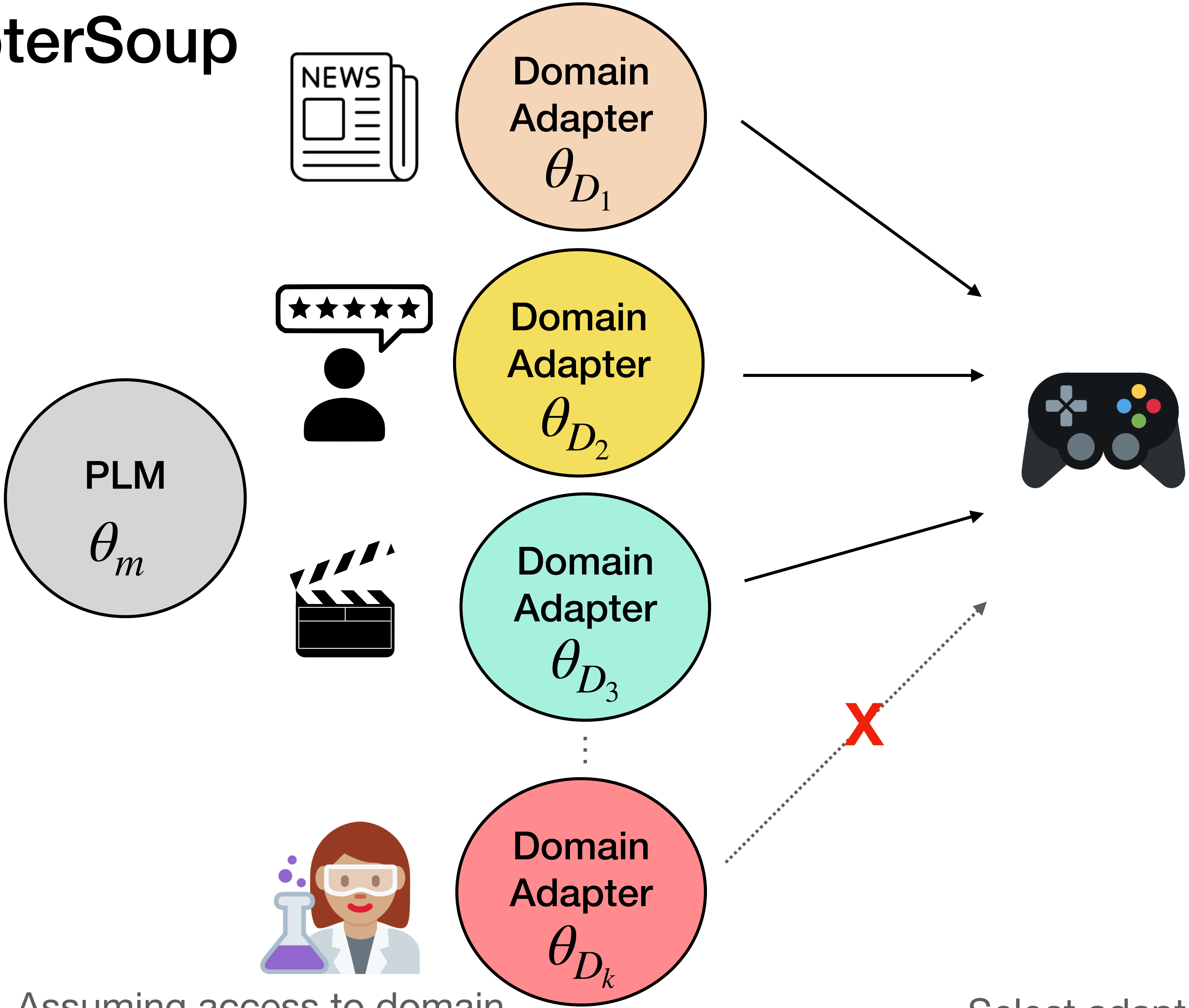
AdapterSoup



Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

Unseen new domain

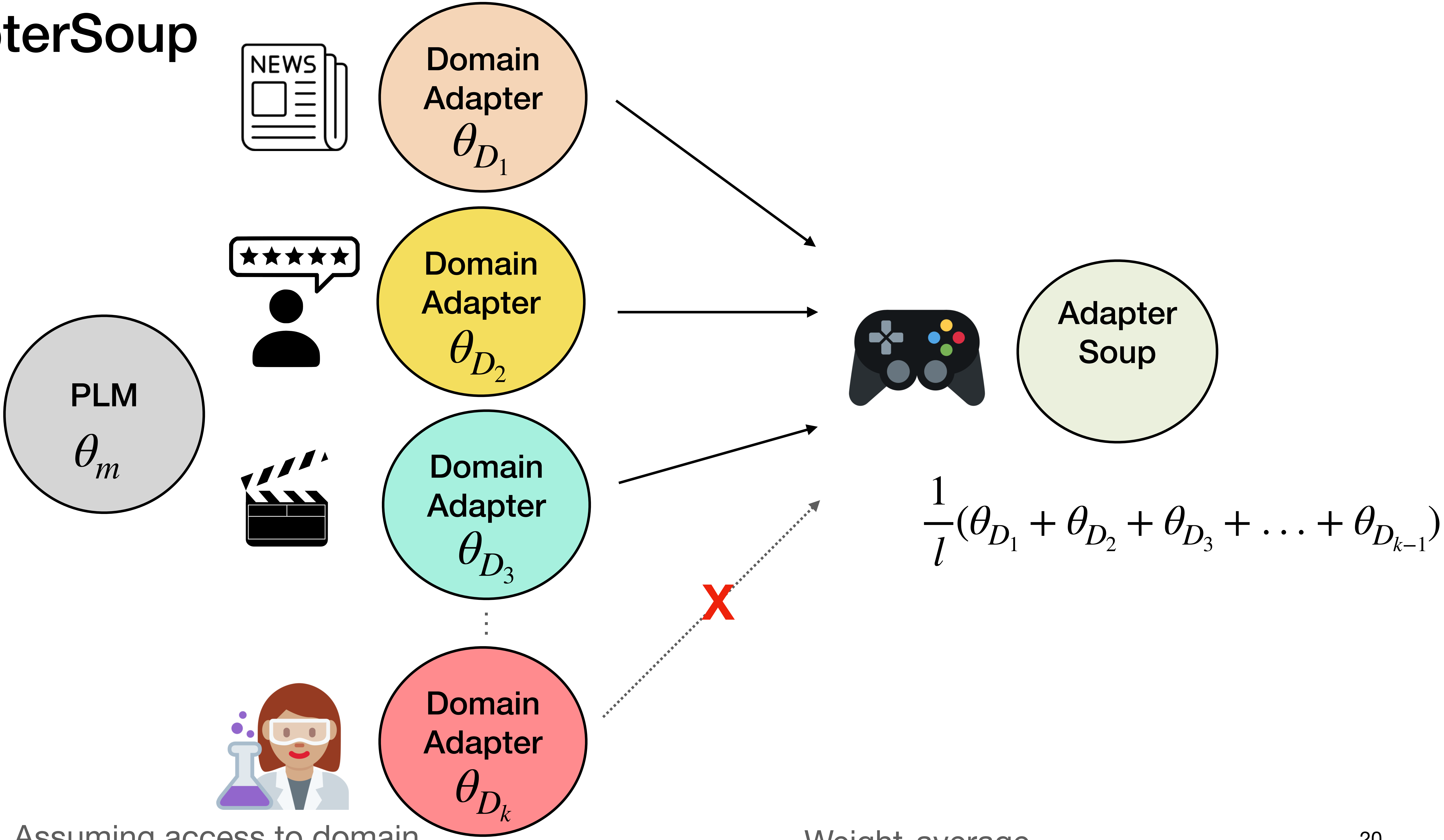
AdapterSoup



Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

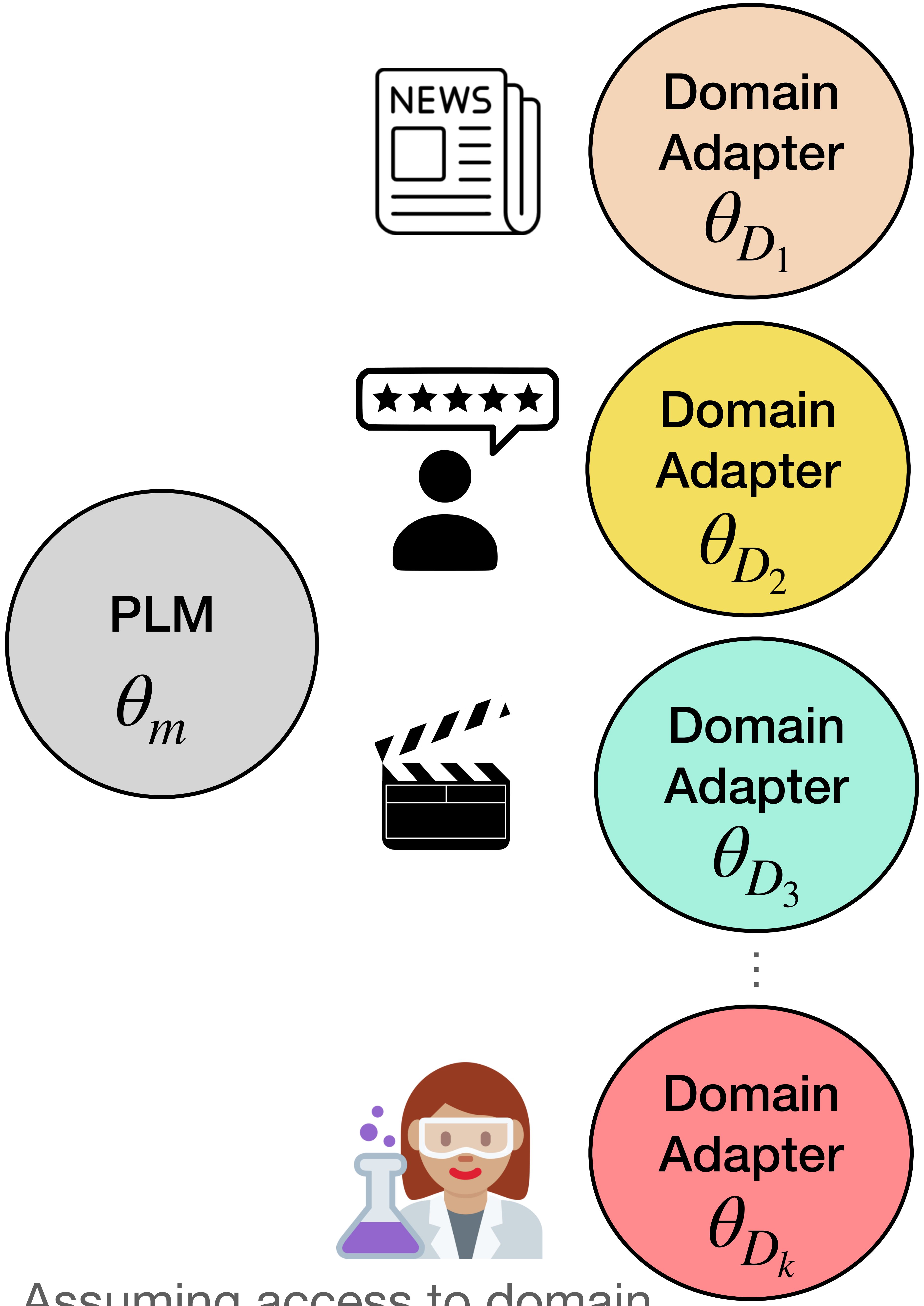
Select adapters for new domain

AdapterSoup



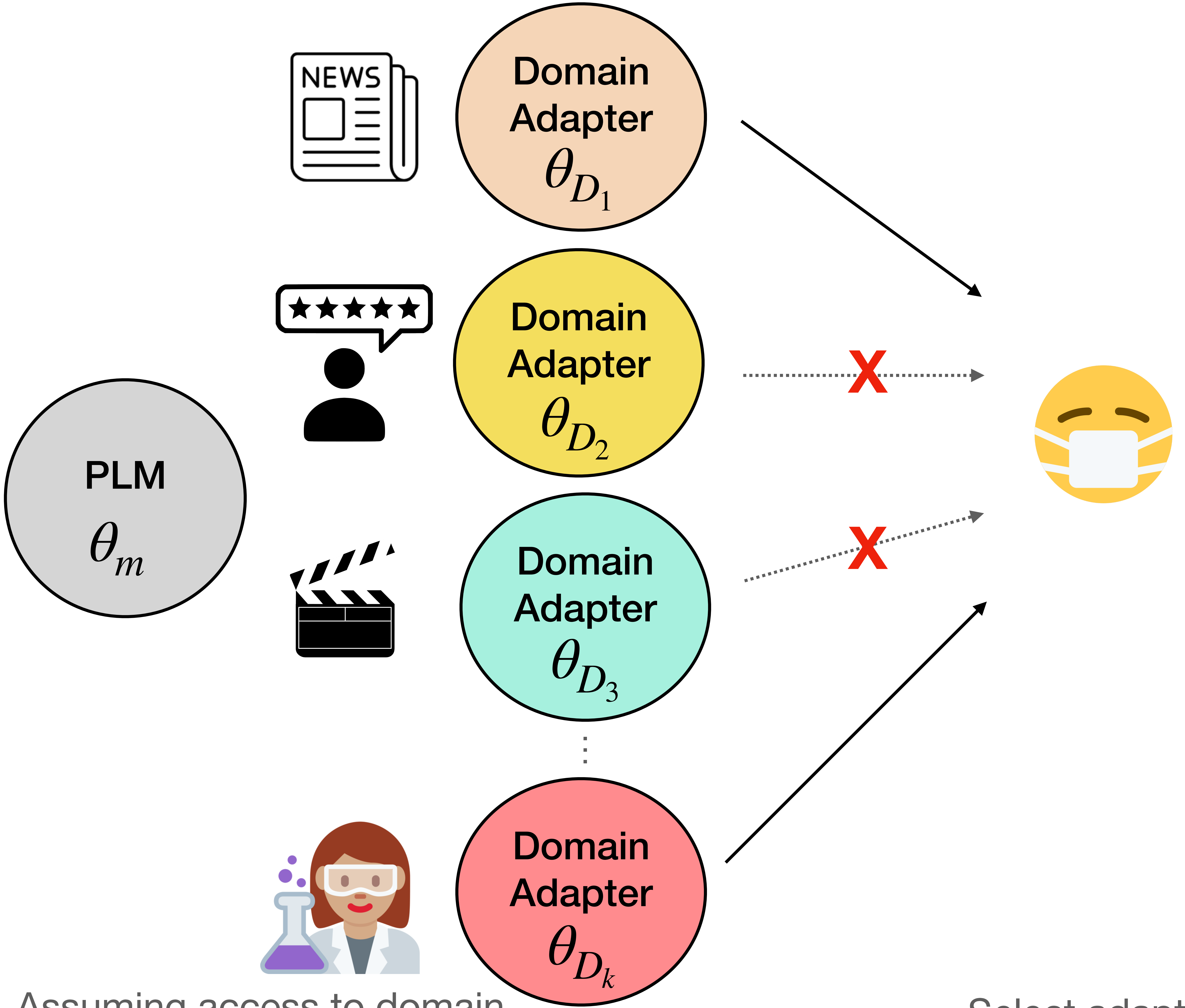
Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

Weight-average selected adapters



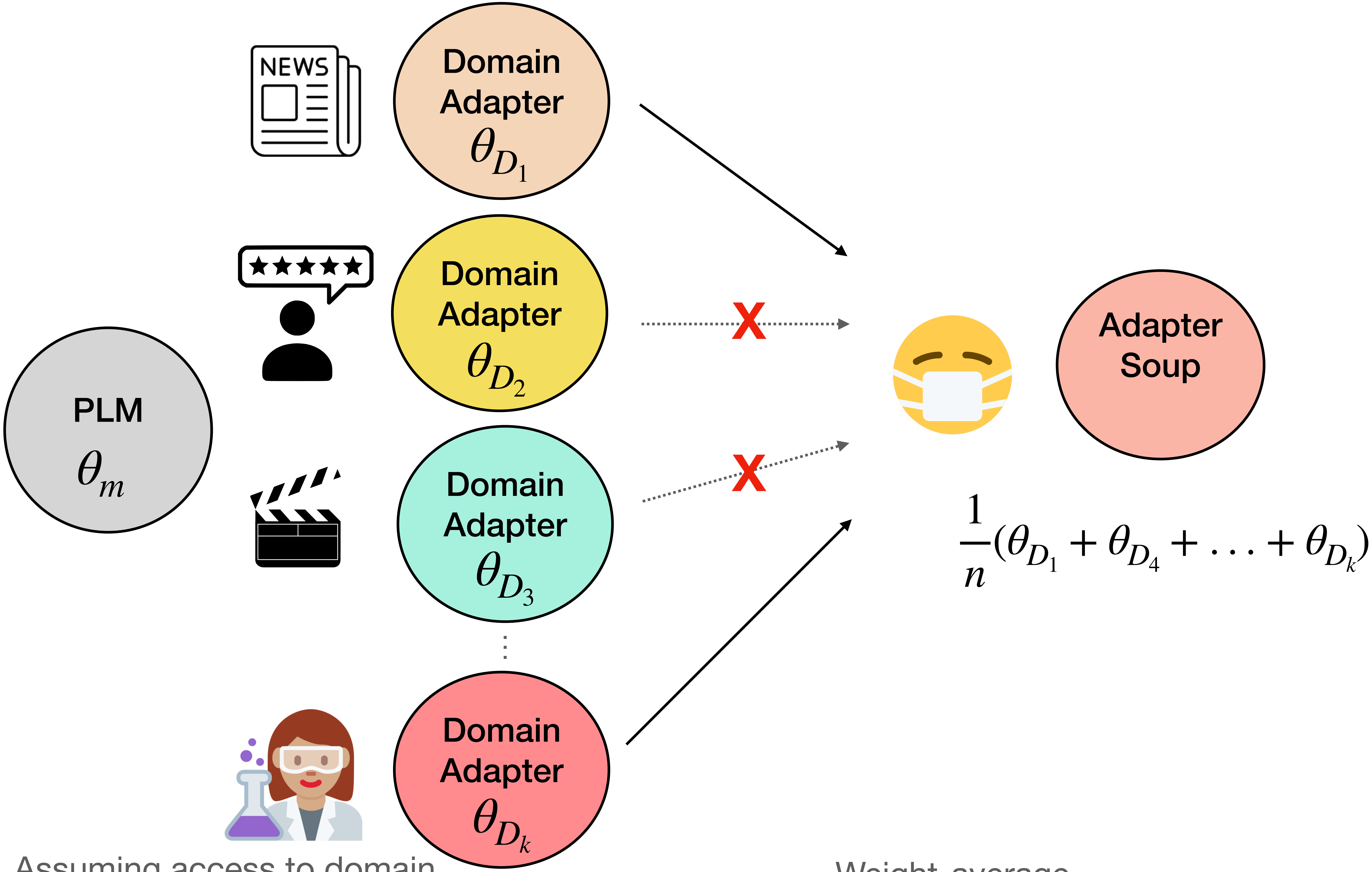
Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

Unseen new domain



Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

Select adapters for new domain



Assuming access to domain adapters $D_1, D_2, \ldots, D_k$

Weight-average selected adapters

How are the domain adapters selected?

How are the domain adapters selected?

Uniform average

- Weight-average all trained adapters

$$AdapterSoup(x) = f(x, \frac{1}{k}\sum_{i=1}^{k}\theta_i)$$

How are the domain adapters selected?

Uniform average

- Weight-average all trained adapters

$$AdapterSoup(x) = f(x, \frac{1}{k}\sum_{i=1}^{k}\theta_i) \longrightarrow$$

uniform soup *(Wortsman et al., 2022)*

How are the domain adapters selected?

Sentence similarity

- sentence-BERT to compute sentence embeddings *(Reimers and Gurevych, 2019)*

How are the domain adapters selected?

Sentence similarity

- sentence-BERT to compute sentence embeddings *(Reimers and Gurevych, 2019)*
- AdapterSoup in order of highest cosine sim.

How are the domain adapters selected?

Clustering

- Using PLM representations of our *k* training domains, we fit a GMM with *k* components *(Aharoni and Goldberg, 2020)*

How are the domain adapters selected?

Clustering

- Using PLM representations of our *k* training domains, we fit a GMM with *k* components *(Aharoni and Goldberg, 2020)*

- Weight-average adapters of training domains that are closest to new domain

- Motivation
- Proposed Approach
- **Experiments**
- Conclusion

Experimental setup

- PLM: GPT-2
- Adapters: bottleneck size 64
- Baselines:
    - GPT-2 (no further training)
    - single adapter selected using sentence similarity, clustering

Datasets

We use data from C4 (Raffel et al., 2020)

Training Domain	Train (Eval.) Tokens
dailymail.co.uk	25M (3M)
wired.com	18M (2M)
express.co.uk	16M (2M)
npr.org	25M (3M)
librarything.com	3M (500K)
instructables.com	25M (3M)
entrepreneur.com	16M (2M)
link.springer.com	28M (4M)
insiderpages.com	8M (1M)
ign.com	10M (1M)
eventbrite.com	11M (1M)
forums.macrumors.com	22M (3M)
androidheadlines.com	14M (2M)
glassdoor.com	4M (500K)
pcworld.com	14M (2M)
csmonitor.com	23M (3M)
lonelyplanet.com	6M (1M)
booking.com	30M (4M)
journals.plos.org	53M (6M)
frontiersin.org	38M (6M)
medium	22M (3M)

Novel Domain	Train (Eval.) Tokens
reuters.com	17M (2M)
techcrunch.com	13M (2M)
fastcompany.com	14M (2M)
nme.com	5M (1M)
fool.com	34M (4M)
inquisitr.com	13M (2M)
mashable.com	14M (2M)
tripadvisor.com	7M (1M)
ncbi.nlm.nih.gov	23M (3M)
yelp.com	68M (6M)

Results

↓Perplexity shown

Method	10 Evaluation Domains										
	reuters	**techcrunch**	**fastco**	**nme**	**fool**	**inquisitr**	**mashable**	**tripadv**	**ncbi**	**yelp**	**Avg.**
GPT-2 (zero-shot)	21.5	27.7	27.9	28.2	23.8	22.4	27.1	40.4	20.7	36.2	27.6
Single Adapter Chosen Using:											
- Sentence similarity	18.9	22.0	22.0	23.1	22.9	18.4	25.3	37.0	18.2	49.4	24.4
- Clustering	17.6	22.4	24.0	21.1	23.3	18.7	23.6	37.7	18.2	44.3	24.0
AdapterSoup (Weight-space average):											
- Uniform	18.2	23.1	22.9	22.2	22.4	18.4	23.1	37.0	19.1	36.2	24.3
- Sentence similarity	17.6	22.0	**21.3**	**20.7**	**22.2**	18.4	22.4	36.2	**17.6**	35.2	23.4
- **Clustering**	**17.3**	**21.8**	**21.3**	21.1	**22.2**	**17.8**	**22.2**	**34.8**	**17.6**	**34.8**	**23.1**
Oracle											
- Best adapter per domain	17.6	22.0	21.5	21.1	22.9	17.8	22.2	37.0	18.2	35.9	23.6
- Clustering + 2 best	17.3	21.8	21.3	20.7	22.0	17.6	22.0	33.4	17.6	33.4	22.7
Hierarchy adapter	16.4	20.1	20.1	20.1	22.2	16.4	22.2	33.1	18.2	34.5	22.3

Results

↓ Perplexity shown

Method	reuters	techcrunch	fastco	nme	fool	inquisitr	mashable	tripadv	ncbi	yelp	Avg.
	10 Evaluation Domains										
GPT-2 (zero-shot)	21.5	27.7	27.9	28.2	23.8	22.4	27.1	40.4	20.7	36.2	27.6
Single Adapter Chosen Using:											
- Sentence similarity	18.9	22.0	22.0	23.1	22.9	18.4	25.3	37.0	18.2	49.4	24.4
- Clustering	17.6	22.4	24.0	21.1	23.3	18.7	23.6	37.7	18.2	44.3	24.0
AdapterSoup (Weight-space average):											
- Uniform	18.2	23.1	22.9	22.2	22.4	18.4	23.1	37.0	19.1	36.2	24.3
- Sentence similarity	17.6	22.0	**21.3**	**20.7**	**22.2**	18.4	22.4	36.2	**17.6**	35.2	23.4
- **Clustering**	**17.3**	**21.8**	**21.3**	21.1	**22.2**	**17.8**	**22.2**	**34.8**	**17.6**	**34.8**	**23.1**
Oracle											
- Best adapter per domain	17.6	22.0	21.5	21.1	22.9	17.8	22.2	37.0	18.2	35.9	23.6
- Clustering + 2 best	17.3	21.8	21.3	20.7	22.0	17.6	22.0	33.4	17.6	33.4	22.7
Hierarchy adapter	16.4	20.1	20.1	20.1	22.2	16.4	22.2	33.1	18.2	34.5	22.3

Using (almost any) AdapterSoup is preferable to GPT-2 without further training or to a single adapter

Results

↓ Perplexity shown

Method	reuters	techcrunch	fastco	nme	fool	inquisitr	mashable	tripadv	ncbi	yelp	Avg.
	10 Evaluation Domains										
GPT-2 (zero-shot)	21.5	27.7	27.9	28.2	23.8	22.4	27.1	40.4	20.7	36.2	27.6
Single Adapter Chosen Using:											
- Sentence similarity	18.9	22.0	22.0	23.1	22.9	18.4	25.3	37.0	18.2	49.4	24.4
- Clustering	17.6	22.4	24.0	21.1	23.3	18.7	23.6	37.7	18.2	44.3	24.0
AdapterSoup (Weight-space average):											
- Uniform	18.2	23.1	22.9	22.2	22.4	18.4	23.1	37.0	19.1	36.2	24.3
- Sentence similarity	17.6	22.0	**21.3**	**20.7**	**22.2**	18.4	22.4	36.2	**17.6**	35.2	23.4
- **Clustering**	**17.3**	**21.8**	**21.3**	21.1	**22.2**	**17.8**	**22.2**	**34.8**	**17.6**	**34.8**	**23.1**
Oracle											
- Best adapter per domain	17.6	22.0	21.5	21.1	22.9	17.8	22.2	37.0	18.2	35.9	23.6
- Clustering + 2 best	17.3	21.8	21.3	20.7	22.0	17.6	22.0	33.4	17.6	33.4	22.7
Hierarchy adapter	16.4	20.1	20.1	20.1	22.2	16.4	22.2	33.1	18.2	34.5	22.3

Hierarchy adapter: lower ppl but at a (much) higher training and inference cost

Results

↓Perplexity shown

Method	reuters	techcrunch	fastco	nme	fool	inquisitr	mashable	tripadv	ncbi	yelp	Avg.
	10 Evaluation Domains										
GPT-2 (zero-shot)	21.5	27.7	27.9	28.2	23.8	22.4	27.1	40.4	20.7	36.2	27.6
Single Adapter Chosen Using:											
- Sentence similarity	18.9	22.0	22.0	23.1	22.9	18.4	25.3	37.0	18.2	49.4	24.4
- Clustering	17.6	22.4	24.0	21.1	23.3	18.7	23.6	37.7	18.2	44.3	24.0
AdapterSoup (Weight-space average):											
- Uniform	18.2	23.1	22.9	22.2	22.4	18.4	23.1	37.0	19.1	36.2	24.3
- Sentence similarity	17.6	22.0	**21.3**	**20.7**	**22.2**	18.4	22.4	36.2	**17.6**	35.2	23.4
- **Clustering**	**17.3**	**21.8**	**21.3**	21.1	**22.2**	**17.8**	**22.2**	**34.8**	**17.6**	**34.8**	**23.1**
Oracle											
- Best adapter per domain	17.6	22.0	21.5	21.1	22.9	17.8	22.2	37.0	18.2	35.9	23.6
- Clustering + 2 best	17.3	21.8	21.3	20.7	22.0	17.6	22.0	33.4	17.6	33.4	22.7
Hierarchy adapter	16.4	20.1	20.1	20.1	22.2	16.4	22.2	33.1	18.2	34.5	22.3

AdapterSoup using clustering: best performance at the inference cost of a single adapter

Analysis

Novel Domain i	Sentence Sim.	Clustering
tripadvisor	booking insiderpages	booking insiderpages lonelyplanet
ncbi	journals frontiersin springer	journals frontiersin springer
reuters	csmonitor wired entrepreneur	dailymail express

Models selected using AdapterSoup with sentence similarity and clustering

Analysis

Novel Domain i	Sentence Sim.	Clustering
tripadvisor	booking insiderpages	booking insiderpages lonelyplanet
ncbi	journals frontiersin springer	journals frontiersin springer
reuters	csmonitor wired entrepreneur	dailymail express

Models selected using AdapterSoup with sentence similarity and clustering

- Tripadvisor & Ncbi: Both methods select almost same domains
- Reuters: good match with clustering, sentence sim. selects non-related domains

- Motivation
- Proposed Approach
- Experiments
- **Conclusion**

Key Takeaways

- AdapterSoup: weight-space averaging of selected adapters trained on top of a PLM to adapt to new domains
- Cost of a model at inference time
- Improves domain generalization of a PLM

Thanks!

paper: arxiv.org/abs/2302.07027

@alexandraxron

achron@cis.lmu.de