Learning What and Where to Transfer

Yunhun Jang*[1,2], Hankook Lee*[1], Sung Ju Hwang[3,4,5], Jinwoo Shin[1,4,5]

[1] School of Electrical Engineering, KAIST
[2] OMNIOUS
[3] School of Computing, KAIST
[4] Graduate School of AI, KAIST
[5] AITRICS

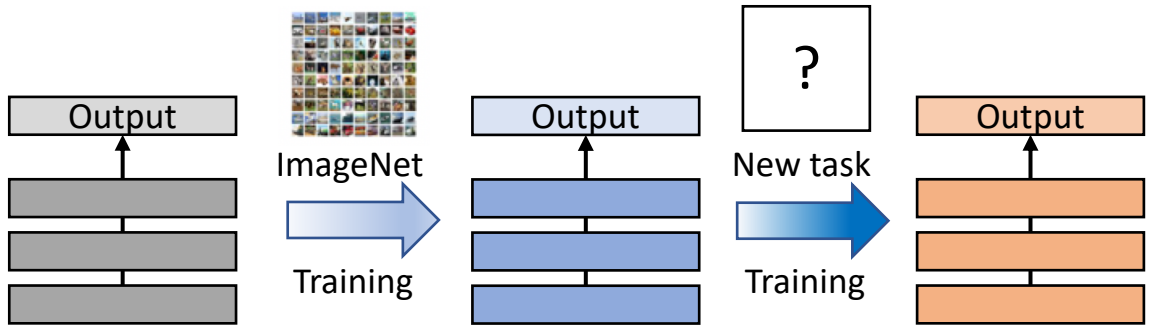
* Equal contribution

Transfer Learning

- DNNs require large labeled datasets to train
- *Transfer learning* is a popular method to mitigate the lack of samples
  - Improve the performance of a model *on a new task*
  - By utilizing the *knowledge* of pre-trained *source models*

Transfer Learning

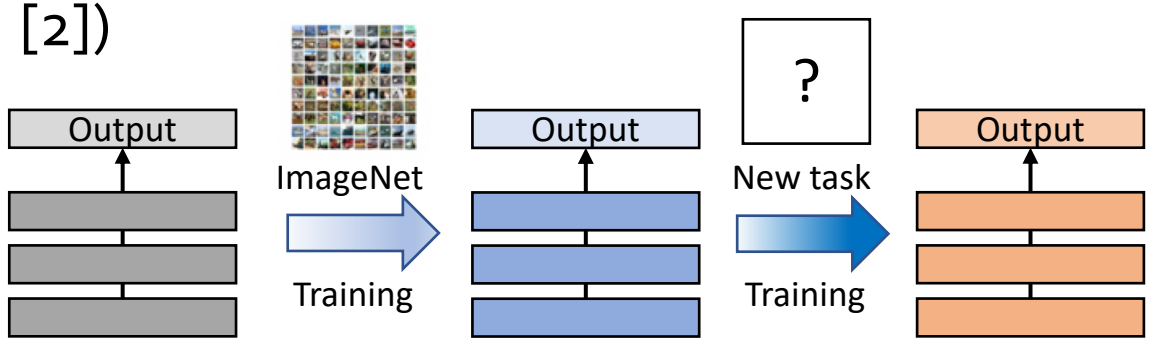
- DNNs require large labeled datasets to train
- *Transfer learning* is a popular method to mitigate the lack of samples
  - Improve the performance of a model *on a new task*
  - By utilizing the *knowledge* of pre-trained *source models*
- Limitations of previous methods
  - Require the same architecture between a source and target models (e.g., fine-tuning)



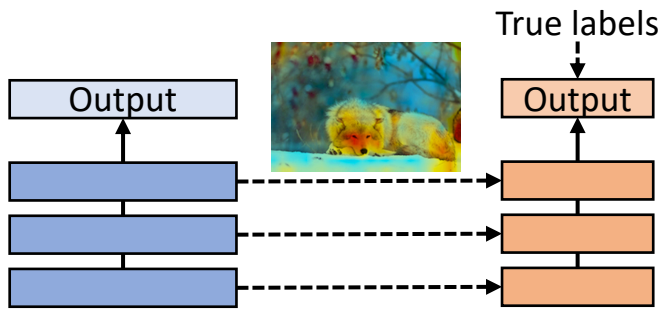
Pre-train and fine-tuning

Transfer Learning

- DNNs require large labeled datasets to train
- *Transfer learning* is a popular method to mitigate the lack of samples
  - Improve the performance of a model *on a new task*
  - By utilizing the *knowledge* of pre-trained *source models*
- Limitations of previous methods
  - Require the same architecture between a source and target models (e.g., fine-tuning)
  - Require exhaustive hand-crafted tuning (e.g., attention transfer [1], Jacobian matching [2])



Pre-train and fine-tuning



Attention transfer/Jacobian matching

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[2] Srinivas, S. and Fleuret, F. Knowledge transfer with Jaco- bian matching. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, 2018.

Learning What/Where to Transfer

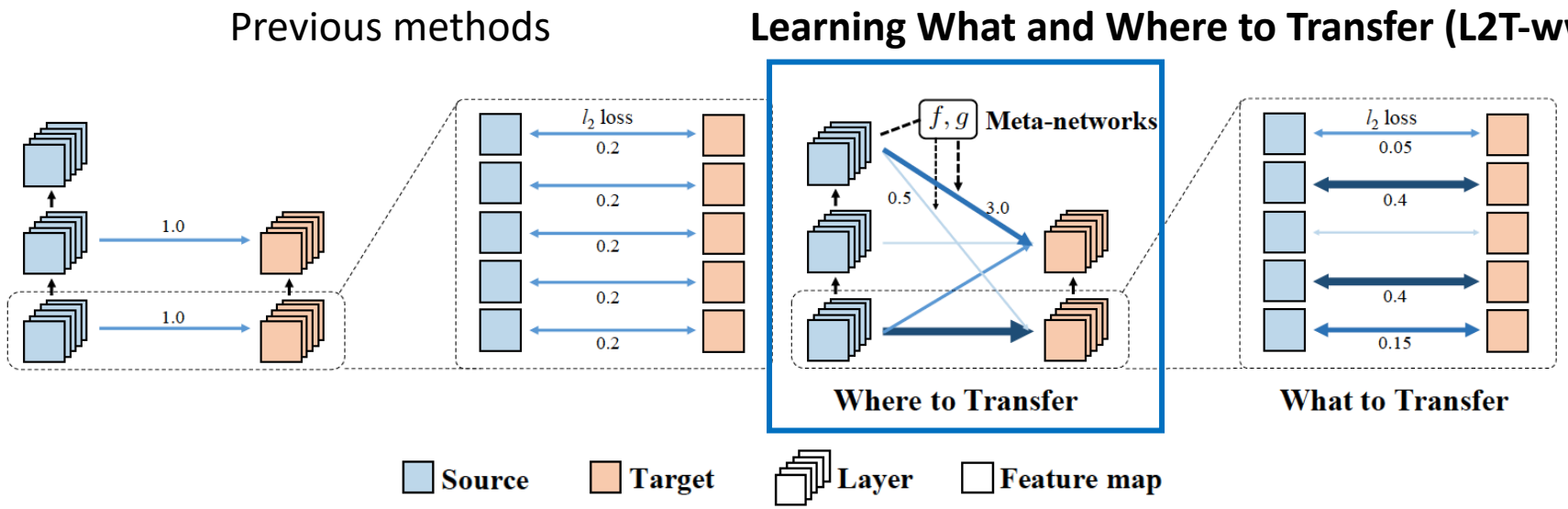
- Propose meta-networks f and g
  - *Learn the learning rules* to transfer the source knowledge

Learning What/Where to Transfer

- Propose meta-networks f and g
  - *Learn the learning rules* to transfer the source knowledge

Where to transfer

- A meta-network g decides useful pairs of source/target layers to transfer



Learning What/Where to Transfer

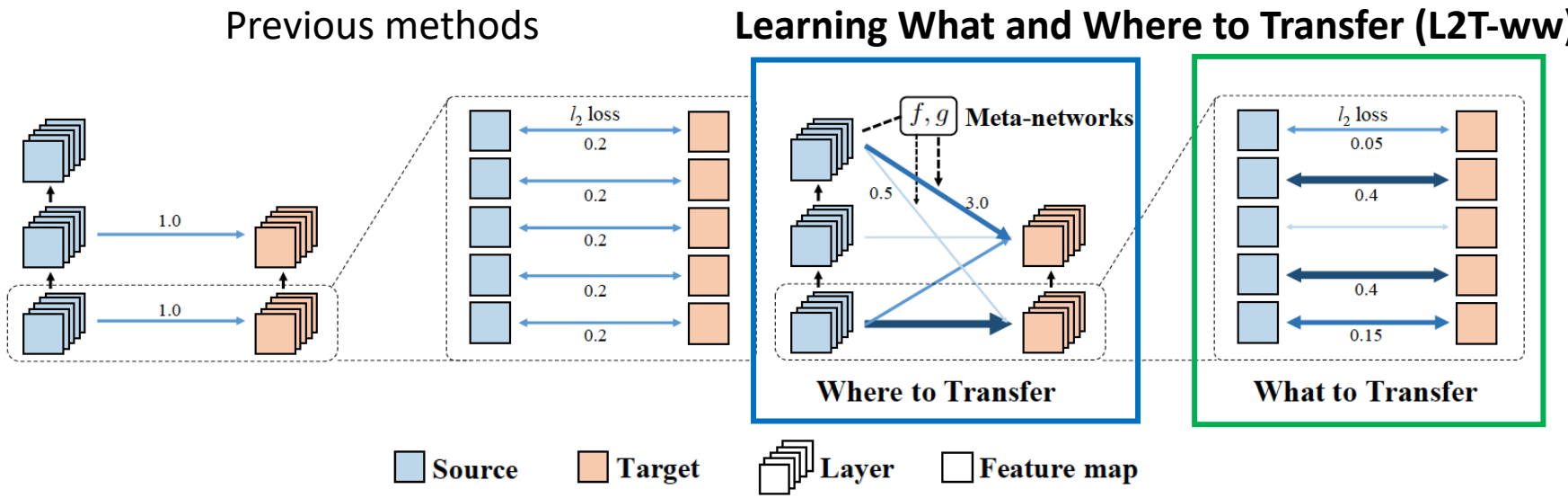
- Propose meta-networks f and g : Learning what/where to transfer (L2T-ww)
  - *Learn the learning rules* to transfer the source knowledge

Where to transfer

- A meta-network g decides useful pairs of source/target layers to transfer

What to transfer

- A meta-network f decides useful channels to transfer

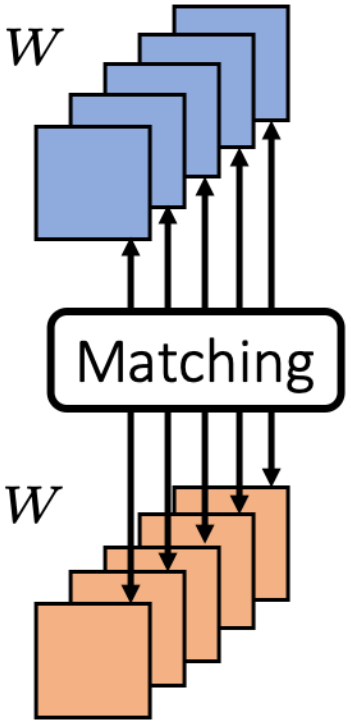


L2T-ww: Learning *What* to Transfer

- Transfer by making target features similar to those of source [3]

$S^m(x) \in \mathbb{R}^{C \times H \times W}$

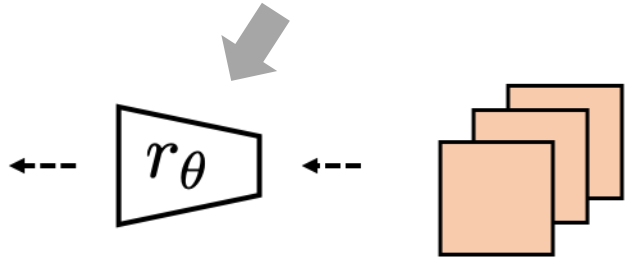
Feature maps of m^{th} source layer



Matching

$r_\theta(T^n(x)) \in \mathbb{R}^{C \times H \times W}$

Transformation for channel dimension matching (e.g., 1x1 conv)



$T^n(x)$

Feature maps of n^{th} target layer

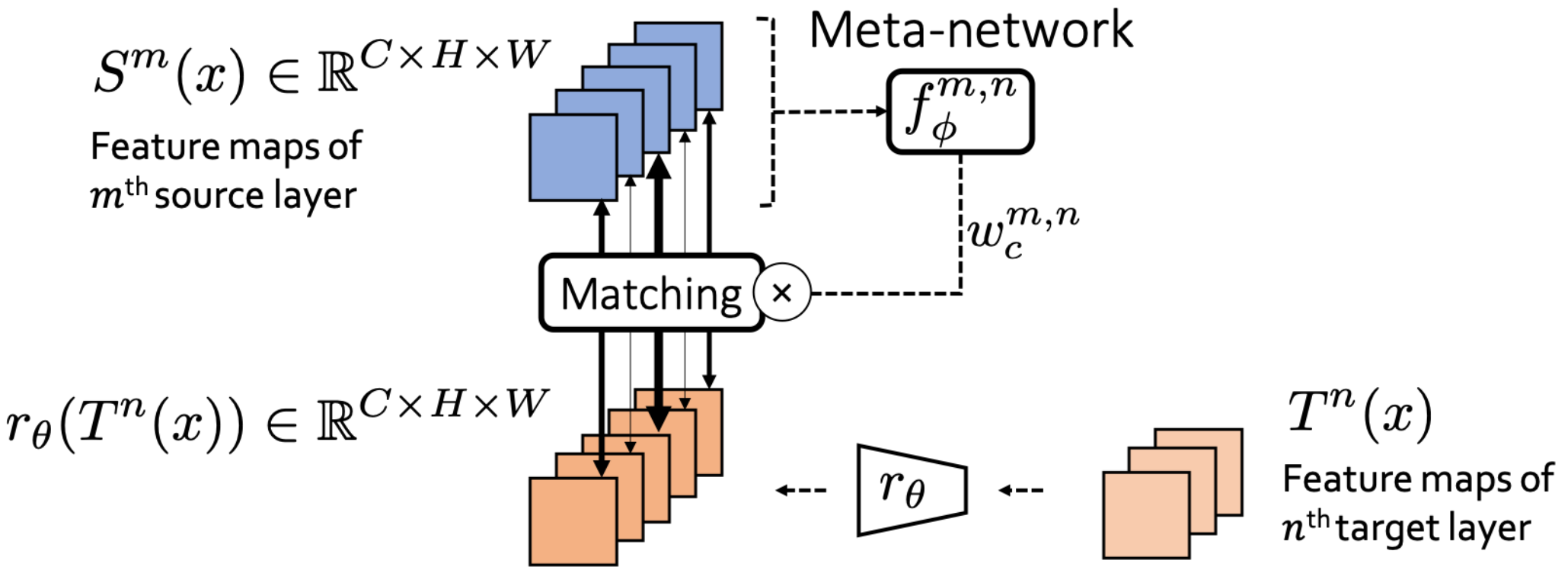
Feature Matching

$$\mathcal{L}_{\texttt{fm}}^{m,n}(\theta|x) = \frac{1}{CHW}\sum_{i,j}(r_\theta(T_\theta^n(x))_{c,i,j} - S^m(x)_{c,i,j})^2$$

[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

L2T-ww: Learning *What* to Transfer

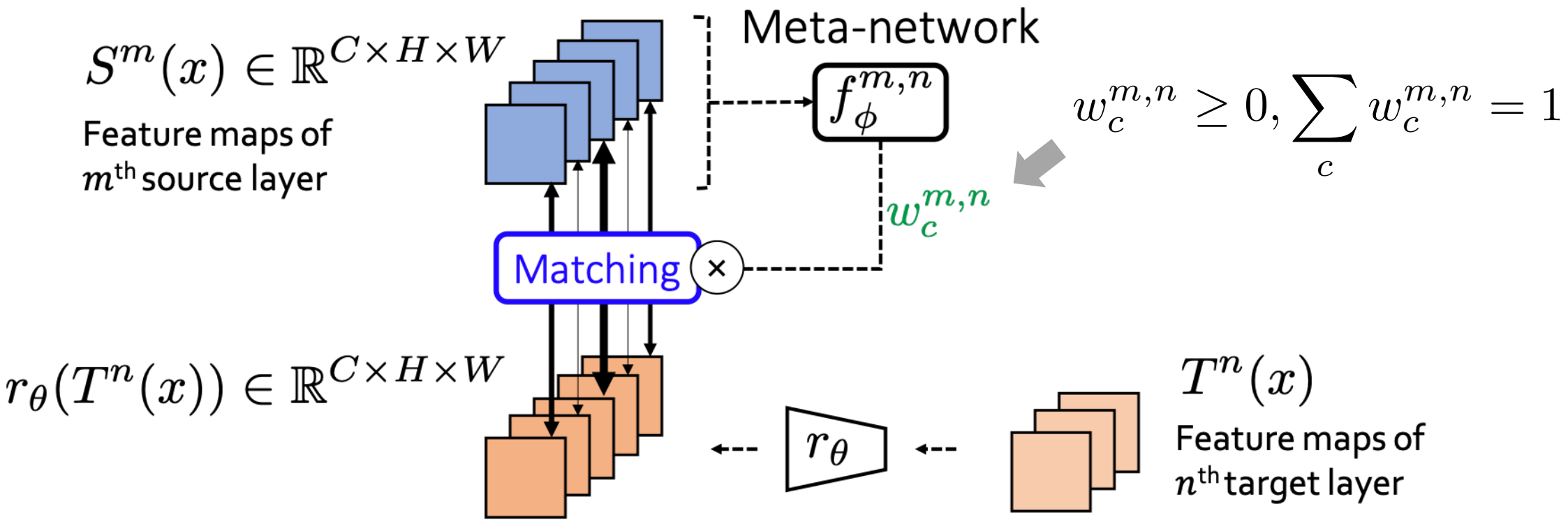
- Learn what to transfer



$$\mathcal{L}_{\texttt{wfm}}^{m,n}(\theta|x, w^{m,n}) = \frac{1}{HW} \sum_c w_c^{m,n} \sum_{i,j} (r_\theta(T_\theta^n(x))_{c,i,j} - S^m(x)_{c,i,j})^2$$

L2T-ww: Learning *What* to Transfer

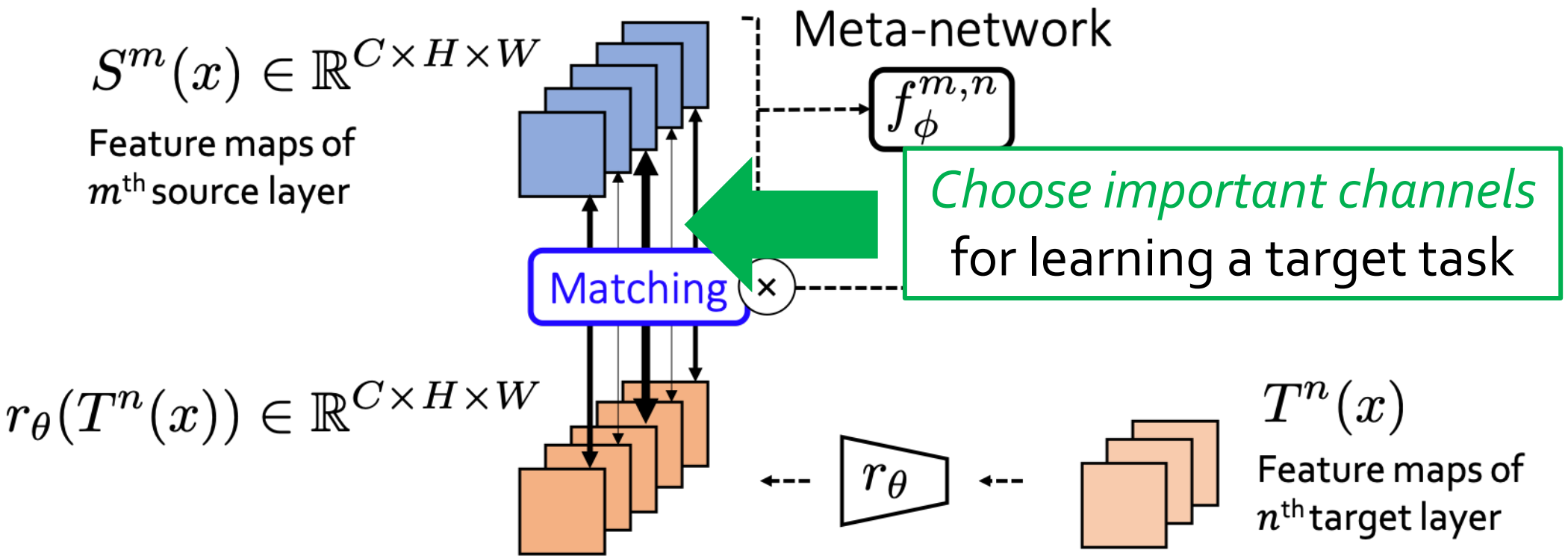
- Learn what to transfer



$$\mathcal{L}_{\texttt{wfm}}^{m,n}(\theta|x, w^{m,n}) = \frac{1}{HW}\sum_c w_c^{m,n}\sum_{i,j}\left(r_\theta(T_\theta^n(x))_{c,i,j} - S^m(x)_{c,i,j}\right)^2$$

L2T-ww: Learning *What* to Transfer

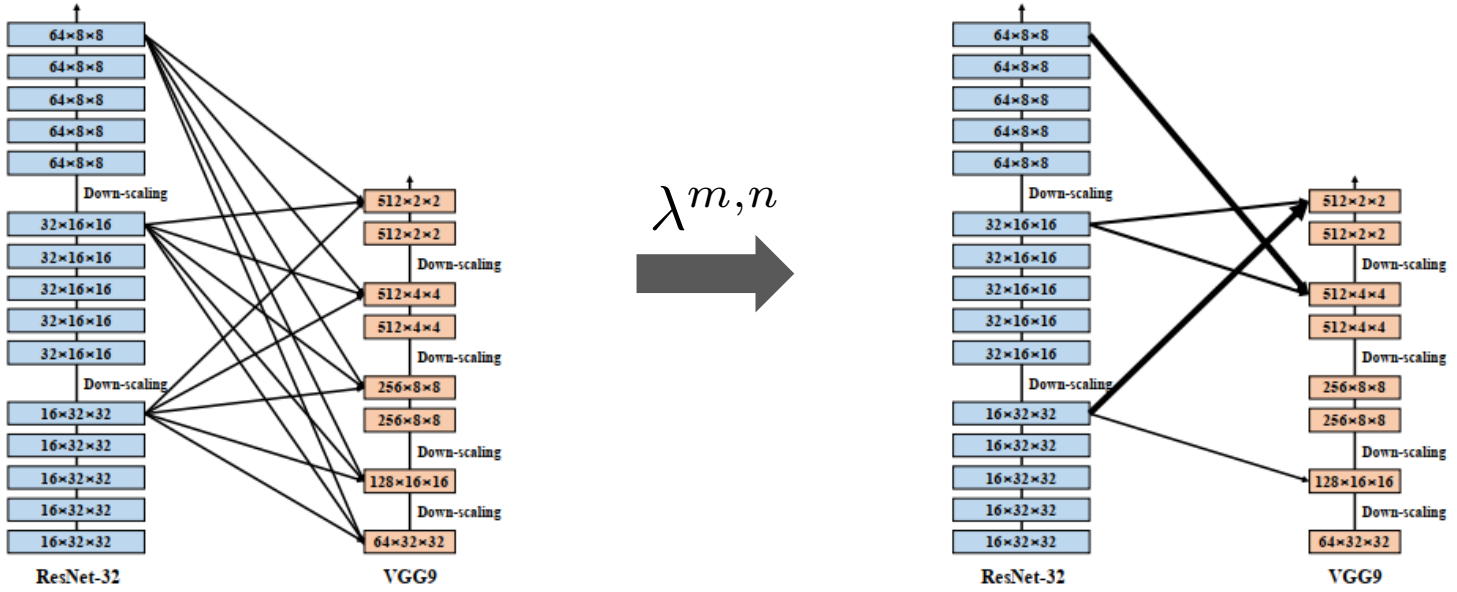
- Learn what to transfer



$$\mathcal{L}_{\texttt{wfm}}^{m,n}(\theta|x, w^{m,n}) = \frac{1}{HW} \sum_c w_c^{m,n} \sum_{i,j} \left(r_\theta(T_\theta^n(x))_{c,i,j} - S^m(x)_{c,i,j}\right)^2$$

L2T-ww: Learning *Where* to Transfer

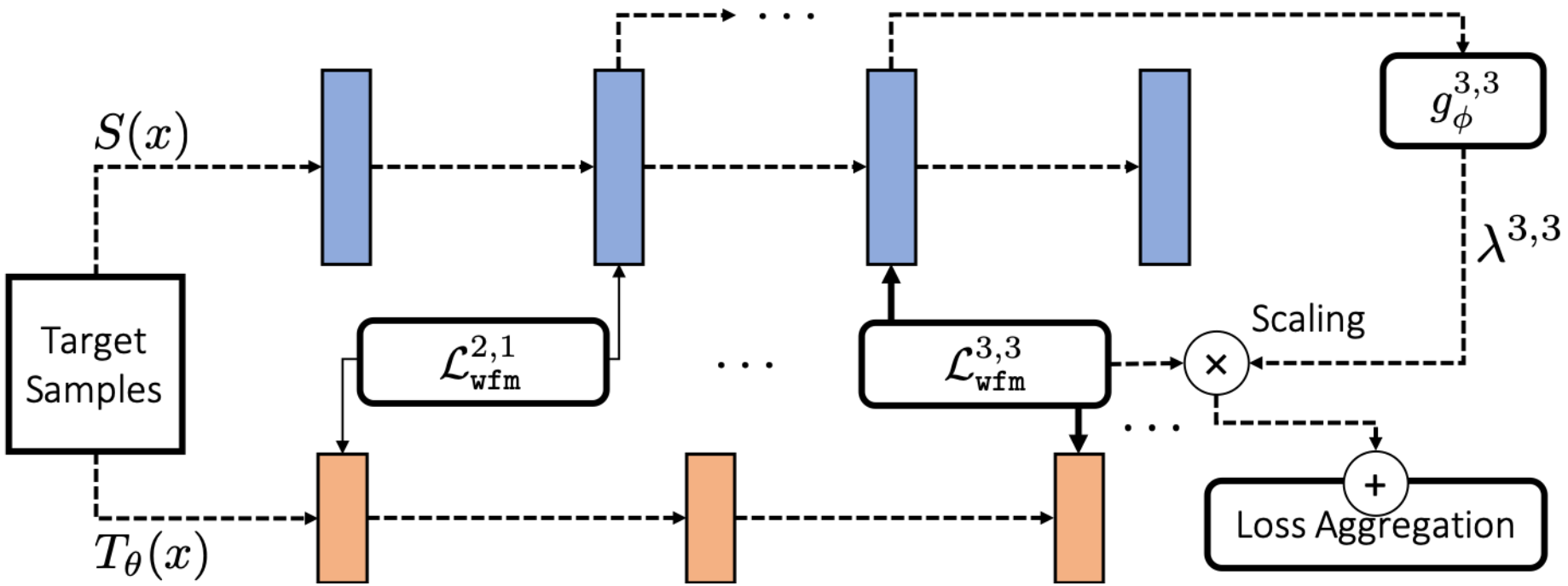
- Learn where to transfer



- Meta-networks choose important matching pairs to transfer
  - Given all possible candidate matching pairs $\mathcal{C}$

L2T-ww: Learning *Where* to Transfer

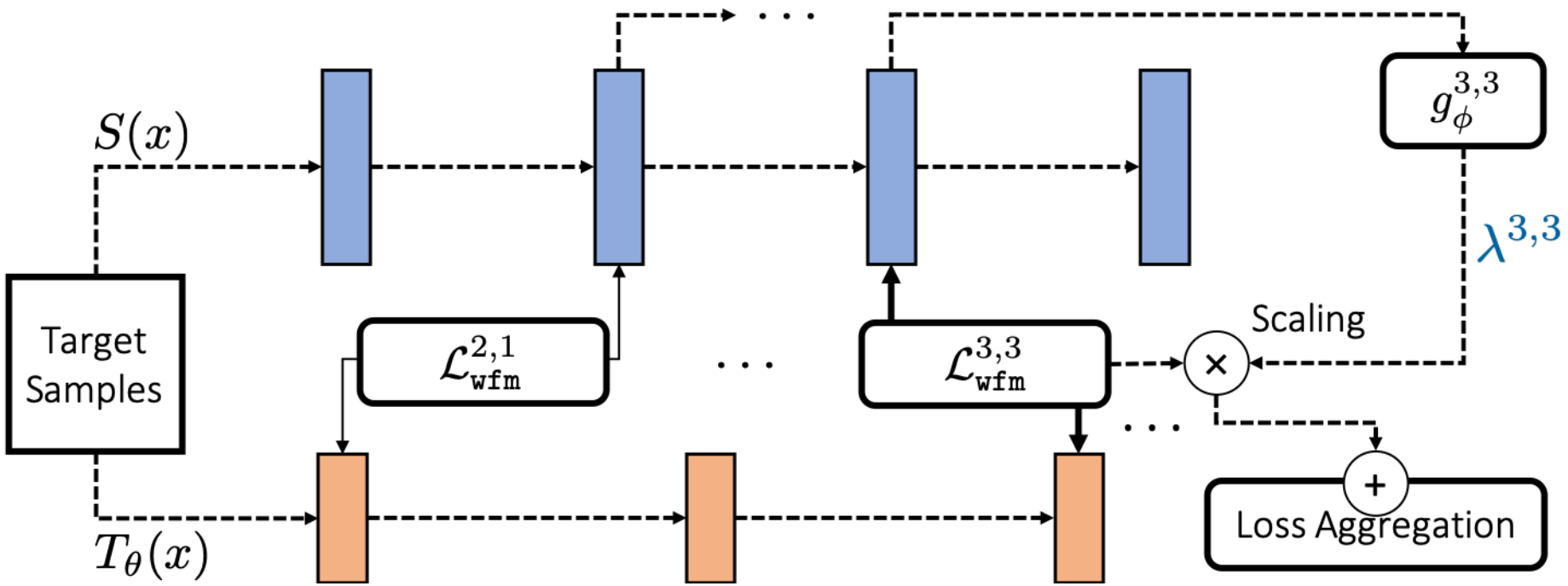
- Learn where to transfer



$$\mathcal{L}_{\texttt{wfm}}(\theta|x,\phi) = \sum_{(m,n)\in\mathcal{C}} \lambda^{m,n}\mathcal{L}_{\texttt{wfm}}^{m,n}(\theta|x,w^{m,n})$$

L2T-ww: Learning *Where* to Transfer

- Learn where to transfer



$$\mathcal{L}_{\texttt{wfm}}(\theta|x,\phi) = \sum_{(m,n)\in\mathcal{C}} \lambda^{m,n}\mathcal{L}_{\texttt{wfm}}^{m,n}(\theta|x,w^{m,n})$$

- Choose pairs of feature-matched layers among all the possible pairs

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- A popular bilevel scheme [4,5] for training meta-parameters ϕ:

[4] Colson, B., Marcotte, P., and Savard, G. An overview of bilevel optimization. *Annals of operations research*, 2007.
[5] Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- A popular bilevel scheme [4,5] for training meta-parameters ϕ:

> 1. *Training simulation*: for $t = 1, \ldots, T$,
>
> $$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{total}}(\theta_t|x_t, y_t, \phi)$$

[4] Colson, B., Marcotte, P., and Savard, G. An overview of bilevel optimization. *Annals of operations research*, 2007.
[5] Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- A popular bilevel scheme [4,5] for training meta-parameters ϕ:

> 1. *Training simulation*: for $t = 1, \ldots, T$,
>
> $$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{total}}(\theta_t|x_t, y_t, \phi)$$
>
> 2. *Evaluation*:
>
> $$\mathcal{L}_{\texttt{meta}}(\phi) = \mathcal{L}_{\texttt{org}}(\theta_{T+1}|x_{\texttt{val}}, y_{\texttt{val}})$$

[4] Colson, B., Marcotte, P., and Savard, G. An overview of bilevel optimization. *Annals of operations research*, 2007.
[5] Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- A popular bilevel scheme [4,5] for training meta-parameters ϕ:

> 1. *Training simulation*: for $t = 1, \ldots, T$,
>
> $$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{total}}(\theta_t|x_t, y_t, \phi)$$
>
> 2. *Evaluation*:
>
> $$\mathcal{L}_{\texttt{meta}}(\phi) = \mathcal{L}_{\texttt{org}}(\theta_{T+1}|x_{\texttt{val}}, y_{\texttt{val}})$$
>
> 3. Update ϕ based on $\nabla_\phi\mathcal{L}_{\texttt{meta}}(\phi)$ using second-order gradients

[4] Colson, B., Marcotte, P., and Savard, G. An overview of bilevel optimization. *Annals of operations research*, 2007.
[5] Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- A popular bilevel scheme [4,5] for training meta-parameters ϕ:

1. *Training simulation*: for $t = 1, \ldots, T$,
$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{total}}(\theta_t|x_t, y_t, \phi)$$
2. *Evaluation*:
$$\mathcal{L}_{\texttt{meta}}(\phi) = \mathcal{L}_{\texttt{org}}(\theta_{T+1}|x_{\texttt{val}}, y_{\texttt{val}})$$
3. Update ϕ based on $\nabla_\phi\mathcal{L}_{\texttt{meta}}(\phi)$ using second-order gradients

- The transfer loss $\mathcal{L}_{\texttt{wfm}}$ acts as a regularization
- A large number of steps T is required to obtain meaningful gradients
  - But it is time-consuming

[4] Colson, B., Marcotte, P., and Savard, G. An overview of bilevel optimization. *Annals of operations research*, 2007.
[5] Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- The proposed bilevel scheme for training meta-parameters ϕ:

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- The proposed bilevel scheme for training meta-parameters ϕ:

1. *Knowledge transfer*: for $t = 1, \ldots, T$,

$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{wfm}}(\theta_t|x, \phi)$$

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- The proposed bilevel scheme for training meta-parameters ϕ:

1. *Knowledge transfer*: for $t = 1, \ldots, T$,

$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{wfm}}(\theta_t|x, \phi)$$

2. *One-step adaption*:

$$\theta_{T+2} = \theta_{T+1} - \alpha\nabla_\theta\mathcal{L}_{\texttt{org}}(\theta_{T+1}|x, y)$$

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- The proposed bilevel scheme for training meta-parameters ϕ:

1. *Knowledge transfer*: for $t = 1, \ldots, T$,
$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{wfm}}(\theta_t|x, \phi)$$
2. *One-step adaption*:
$$\theta_{T+2} = \theta_{T+1} - \alpha\nabla_\theta\mathcal{L}_{\texttt{org}}(\theta_{T+1}|x, y)$$
3. *Evaluation*:
$$\mathcal{L}_{\texttt{meta}}(\phi) = \mathcal{L}_{\texttt{org}}(\theta_{T+2}|x, y).$$
4. Update ϕ based on $\nabla_\phi\mathcal{L}_{\texttt{meta}}(\phi)$ using second-order gradients

L2T-ww: Training Meta-Networks

- Total loss for target model: $\mathcal{L}_{\texttt{total}}(\theta|x, y, \phi) = \mathcal{L}_{\texttt{org}}(\theta|x, y) + \beta\mathcal{L}_{\texttt{wfm}}(\theta|x, \phi)$
- The proposed bilevel scheme for training meta-parameters ϕ:

1. *Knowledge transfer*: for $t = 1, \ldots, T$,
$$\theta_{t+1} = \theta_t - \alpha\nabla_\theta\mathcal{L}_{\texttt{wfm}}(\theta_t|x, \phi)$$
2. *One-step adaption*:
$$\theta_{T+2} = \theta_{T+1} - \alpha\nabla_\theta\mathcal{L}_{\texttt{org}}(\theta_{T+1}|x, y)$$
3. *Evaluation*:
$$\mathcal{L}_{\texttt{meta}}(\phi) = \mathcal{L}_{\texttt{org}}(\theta_{T+2}|x, y).$$
4. Update ϕ based on $\nabla_\phi\mathcal{L}_{\texttt{meta}}(\phi)$ using second-order gradients

- Ours is effective for learning ϕ with a small number of steps T
- Ours learns θ and ϕ jointly without separate meta-learning phase

Experiments

- Learning what and where to transfer gives consistent improvements
  - Suggested method works well in various tasks and architectures

Source task	TinyImageNet		ImageNet			
Target task	CIFAR-100	STL-10	CUB200	MIT67	Stanford40	Stanford Dogs
Scratch	67.69±0.22	65.18±0.91	42.15±0.75	48.91±0.53	36.93±0.68	58.08±0.26
LwF[6]	69.23±0.09	68.64±0.58	45.52±0.66	53.73±2.14	39.73±1.63	66.33±0.45
AT[1] (one-to-one)	67.54±0.40	74.19±0.22	57.74±1.17	59.18±1.57	59.29±0.91	69.70±0.08
LwF[6]+AT[1] (one-to-one)	68.75±0.09	75.06±0.57	58.90±1.32	61.42±1.68	60.20±1.34	72.67±0.26
FM[3] (single)	69.40±0.67	75.00±0.34	47.60±0.31	55.15±0.93	42.93±1.48	66.05±0.76
FM[3] (one-to-one)	69.97±0.24	76.38±1.18	48.93±0.40	54.88±1.24	44.50±0.96	67.25±0.88
L2T-w (single)	70.27±0.09	74.35±0.92	51.95±0.83	60.41±0.37	46.25±3.66	69.16±0.70
L2T-w (one-to-one)	70.02±0.19	76.42±0.52	56.61±0.20	59.78±1.90	48.19±1.42	69.84±1.45
L2T-ww (all-to-all)	**70.96**±0.61	**78.31**±0.21	**65.05**±1.19	**64.85**±2.75	**63.08**±0.88	**78.08**±0.96

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Learning what and where to transfer gives consistent improvements
  - Suggested method works well in various tasks and architectures

Source task	TinyImageNet		ImageNet			
Target task	CIFAR-100	STL-10	CUB200	MIT67	Stanford40	Stanford Dogs
Scratch	67.69±0.22	65.18±0.91	42.15±0.75	48.91±0.53	36.93±0.68	58.08±0.26
LwF[6]	69.23±0.09	68.64±0.58	45.52±0.66	53.73±2.14	39.73±1.63	66.33±0.45
AT[1] (one-to-one)	67.54±0.40	74.19±0.22	57.74±1.17	59.18±1.57	59.29±0.91	69.70±0.08
LwF[6]+AT[1] (one-to-one)	68.75±0.09	75.06±0.57	58.90±1.32	61.42±1.68	60.20±1.34	72.67±0.26
FM[3] (single)	69.40±0.67	75.00±0.34	47.60±0.31	55.15±0.93	42.93±1.48	66.05±0.76
FM[3] (one-to-one)	69.97±0.24	76.38±1.18	48.93±0.40	54.88±1.24	44.50±0.96	67.25±0.88
L2T-w (single)	70.27±0.09	74.35±0.92	51.95±0.83	60.41±0.37	46.25±3.66	69.16±0.70
L2T-w (one-to-one)	70.02±0.19	76.42±0.52	56.61±0.20	59.78±1.90	48.19±1.42	69.84±1.45
L2T-ww (all-to-all)	**70.96**±0.61	**78.31**±0.21	**65.05**±1.19	**64.85**±2.75	**63.08**±0.88	**78.08**±0.96

Maximum **+15%** relative improvements

  - Learning what to transfer (channel importance) improves all the baselines

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Learning what and where to transfer gives consistent improvements
  - Suggested method works well in various tasks and architectures

Source task	TinyImageNet		ImageNet			
Target task	CIFAR-100	STL-10	CUB200	MIT67	Stanford40	Stanford Dogs
Scratch	67.69±0.22	65.18±0.91	42.15±0.75	48.91±0.53	36.93±0.68	58.08±0.26
LwF[6]	69.23±0.09	68.64±0.58	45.52±0.66	53.73±2.14	39.73±1.63	66.33±0.45
AT[1] (one-to-one)	67.54±0.40	74.19±0.22	57.74±1.17	59.18±1.57	59.29±0.91	69.70±0.08
LwF[6]+AT[1] (one-to-one)	68.75±0.09	75.06±0.57	58.90±1.32	61.42±1.68	60.20±1.34	72.67±0.26
FM[3] (single)	69.40±0.67	75.00±0.34	47.60±0.31	55.15±0.93	42.93±1.48	66.05±0.76
FM[3] (one-to-one)	69.97±0.24	76.38±1.18	48.93±0.40	54.88±1.24	44.50±0.96	67.25±0.88
L2T-w (single)	70.27±0.09	74.35±0.92	51.95±0.83	60.41±0.37	46.25±3.66	69.16±0.70
L2T-w (one-to-one)	70.02±0.19	76.42±0.52	56.61±0.20	59.78±1.90	48.19±1.42	69.84±1.45
L2T-ww (all-to-all)	**70.96**±0.61	**78.31**±0.21	**65.05**±1.19	**64.85**±2.75	**63.08**±0.88	**78.08**±0.96

Maximum **+15%** relative improvements

Maximum **+25%** relative improvements

  - Learning what to transfer (channel importance) improves all the baselines
  - Learning where to transfer (pair importance) gives more improvements on what to transfer

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments: Different *architectures*

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

+2.45% relative improvements

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments: Different *architectures, initialization*

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

+2.45% relative improvements

+2.72% relative improvements

[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments: Different *architectures*, *initialization* and *datasets*

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

+2.45% relative improvements

+2.72% relative improvements

+4.37% relative improvements

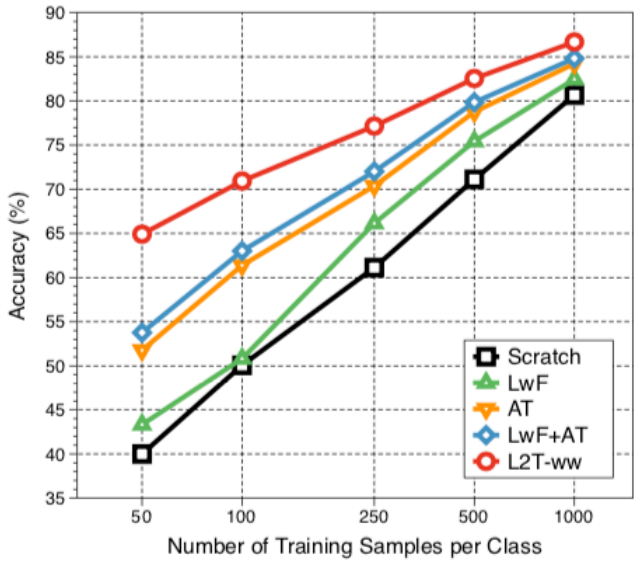
[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR* 2017
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments: Different *architectures*, *initialization* and *datasets*

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

- Limited data-regime experiments
  - Smaller the volume of the target dataset
    → More relative gain of ours
  - Ours efficiently boosts up the performance of a target model



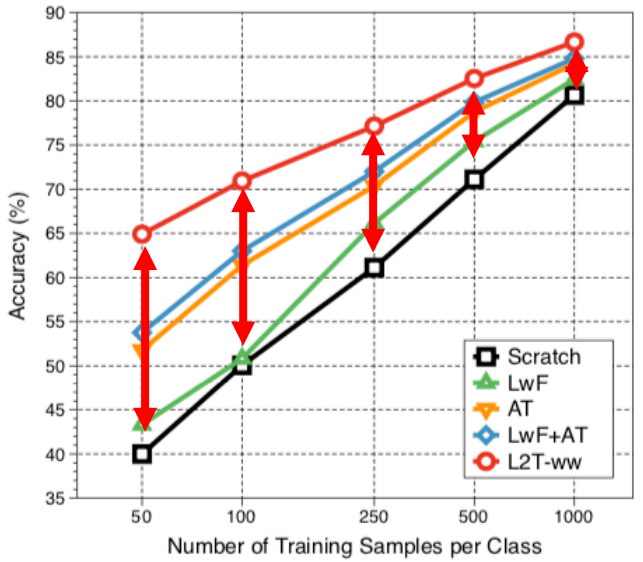
[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Experiments

- Multi-source experiments: Different *architectures*, *initialization* and *datasets*

First source	TinyImageNet (ResNet32)			
Second source	None	TinyImageNet (ResNet20)	TinyImageNet (ResNet32)	CIFAR-10 (ResNet32)
Scratch	65.18±0.91	65.18±0.91	65.18±0.91	65.18±0.91
LwF[6]	68.64±0.58	68.56±2.24	68.05±2.12	69.51±0.63
AT[1]	74.19±0.22	73.24±0.12	73.78±1.16	73.99±0.51
LwF[6]+AT[1]	75.06±0.57	74.72±0.46	74.77±0.30	74.41±1.51
FM[3] (single)	75.00±0.34	75.83±0.56	75.99±0.11	74.60±0.73
FM[3] (one-to-one)	76.38±1.18	77.45±0.48	77.69±0.79	77.15±0.41
L2T-ww (all-to-all)	**78.31**±0.21	**79.35**±0.41	**79.80**±0.52	**80.52**±0.29

- Limited data-regime experiments
  - Smaller the volume of the target dataset
    → More relative gain of ours
  - Ours efficiently boosts up the performance of a target model



[1] Zagoruyko, S. and Komodakis, N. Paying more attention to attention: Improving the performance of convolutional neural networks via
[3] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.
[6] Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Conclusion

- Meta-learning based transfer method
  - Selective transfer depending on a source and target task relation
  - Effective training scheme that learns meta-networks and target model jointly
  - Applicable between heterogeneous or/and multiple network architectures and tasks

Poster #186
Thursday Jun 13th 6:30 – 9:00 PM
@ Pacific Ballroom