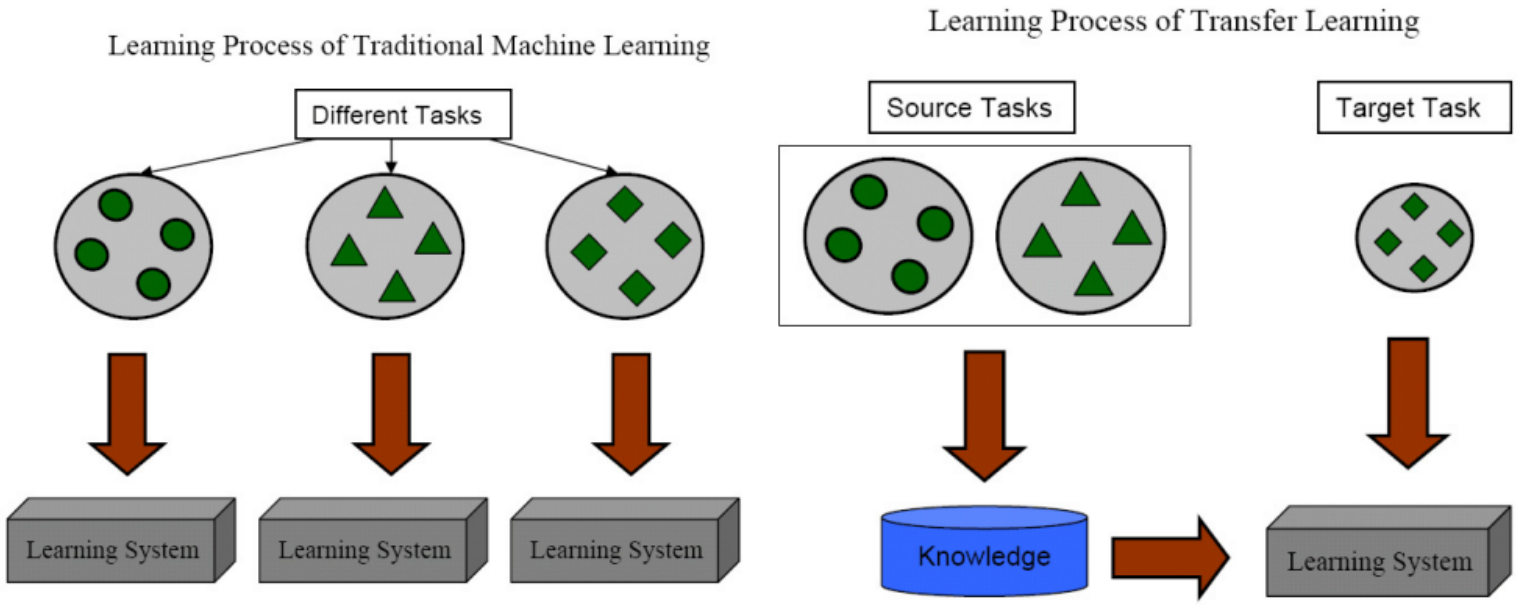
Deep Learning Optimization - Transfer Learning & Knowledge Distillation

March 23, 2022

Eunhyeok Park

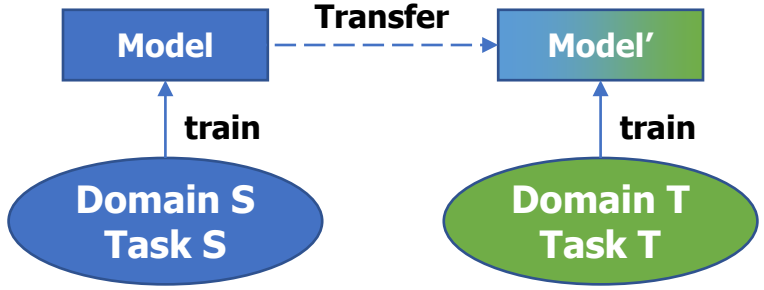
Transfer Learning - 1

- Transfer learning is a research topic of machine learning that tries to solve the target task by utilizing knowledge achieved from the source task



[A Survey on Transfer Learning, Pan]

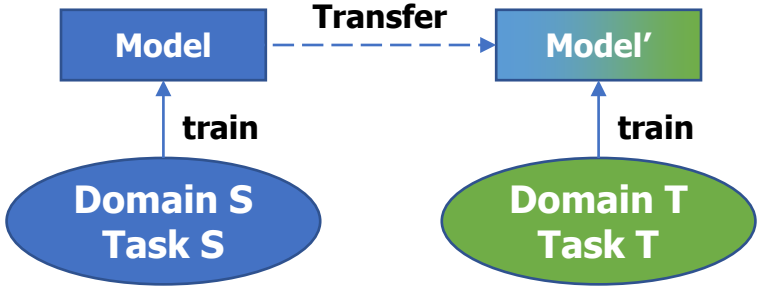
Transfer Learning - 2



- Notation and definition
  - **Domain** $\mathbf{D} = \{\boldsymbol{\mathcal{X}}, \mathbf{P}(\boldsymbol{X})\}$,
    $where\ \mathcal{X}$ is a feature space and a marginal probability distribution $P(X)$ for $X \in \mathcal{X}$
    - Image, spectrogram, item history…
  - **Task** $\mathbf{T} = \{\boldsymbol{\mathcal{Y}}, \mathbf{P}(\boldsymbol{y}|\boldsymbol{x})\}$,
    $where\ \mathcal{Y}$ is a label space and a conditional probability distribution $P(Y|X)$ for $Y \in \mathcal{Y}$
    - Annotated label, i.e. class label, bounding box position…
- Transfer learning?:
  - Given a source domain D_S and learning task T_S, a target domain D_T and learning task T_T, **transfer learning aims to help improve the learning of the target predictive function** $\boldsymbol{f_T}(\cdot)$ **in** $\boldsymbol{D_T}$ using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$

[A Survey on Transfer Learning, Pan]

Transfer Learning - 3



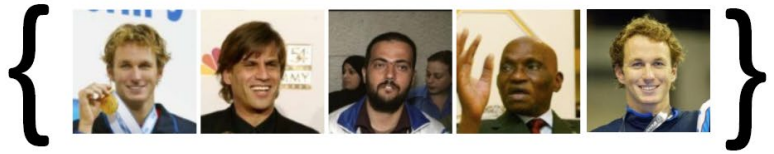
- $D_S \neq D_T, T_S = T_T$: **Domain adaptation**

0-9 digits classification
(e.g., 6, 2, 3, 8, 1, 0)

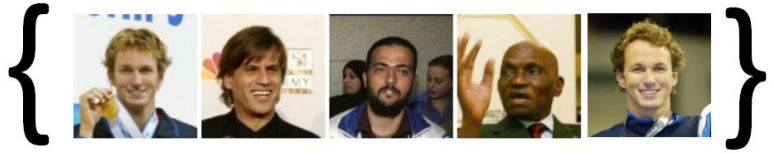
0-9 digits classification
(e.g., 0, 4, 8, 5, 3, 9)

$\mathcal{D}_S \neq \mathcal{D}_T$ $\mathcal{T}_S = \mathcal{T}_T$

- $D_S = D_T, T_S \neq T_t$: **Inductive transfer learning**



Age (e.g., 31, 49, 34, 50, 31)

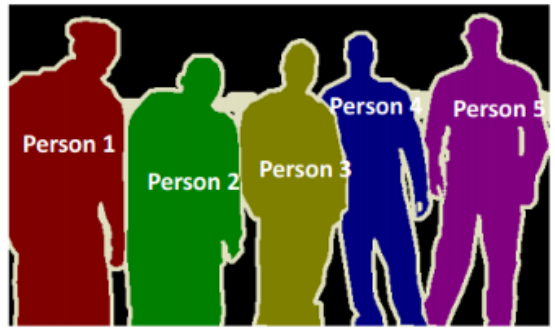


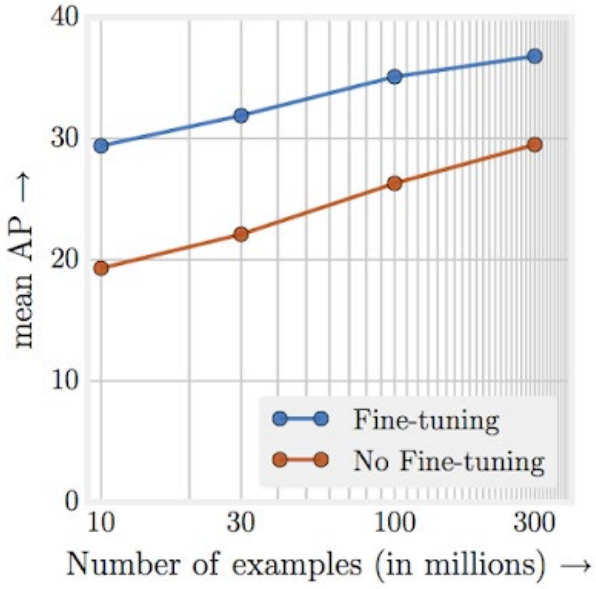
Person recognition
(e.g., John, Aaron, Adam, Will, John)

$\mathcal{D}_S = \mathcal{D}_T$ $\mathcal{T}_S \neq \mathcal{T}_T$

Importance of Transfer Learning

- Deep learning is a data-oriented machine learning technique
  - It could show remarkable performance in a variety of A.I fields, but only when large enough data is available
  - In real life, labeled data is expensive and may not be available
    - Ex) breath cancer image labeled by a doctor
    - Ex) pixel-wised labeled image for segmentation
    - Ex) a new, unseen problem with a few samples
- Transfer learning could be helpful to get high accuracy at the target task by exploiting the knowledge from the source task



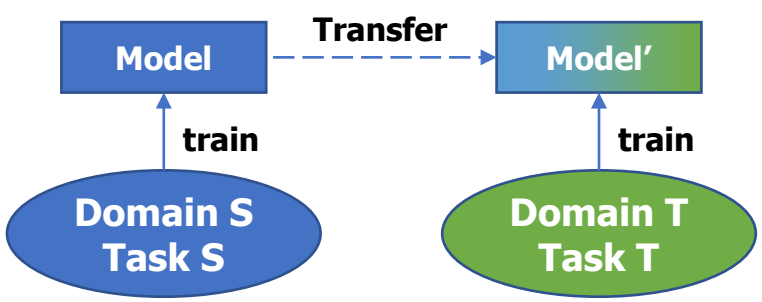


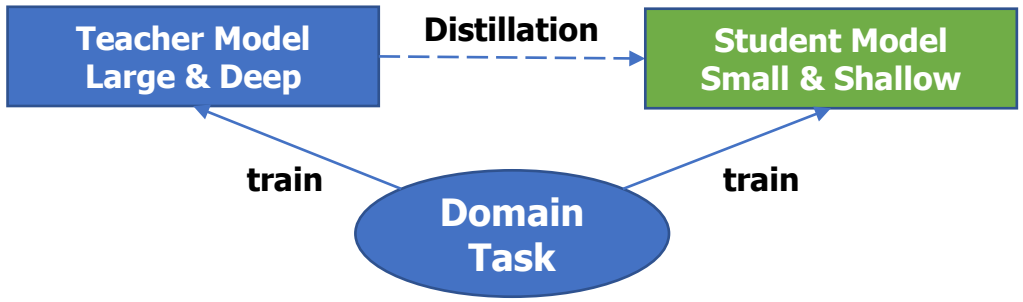
[https://ai.googleblog.com/2017/07/revisiting-unreasonable-effectiveness.html]

Transfer Learning vs Knowledge Distillation

- Transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$
  - Domain adaptation, multi-task learning, continual learning, etc.

- What if $D_S = D_T$ & $T_S = T_T$? → knowledge distillation
  - In general, use accurate and large model (teacher) and smaller model (student)
  - Knowledge distillation is used to increase the accuracy of a smaller model
  - Distill the teacher's feature map or output to the student





Analysis of Transfer Learning

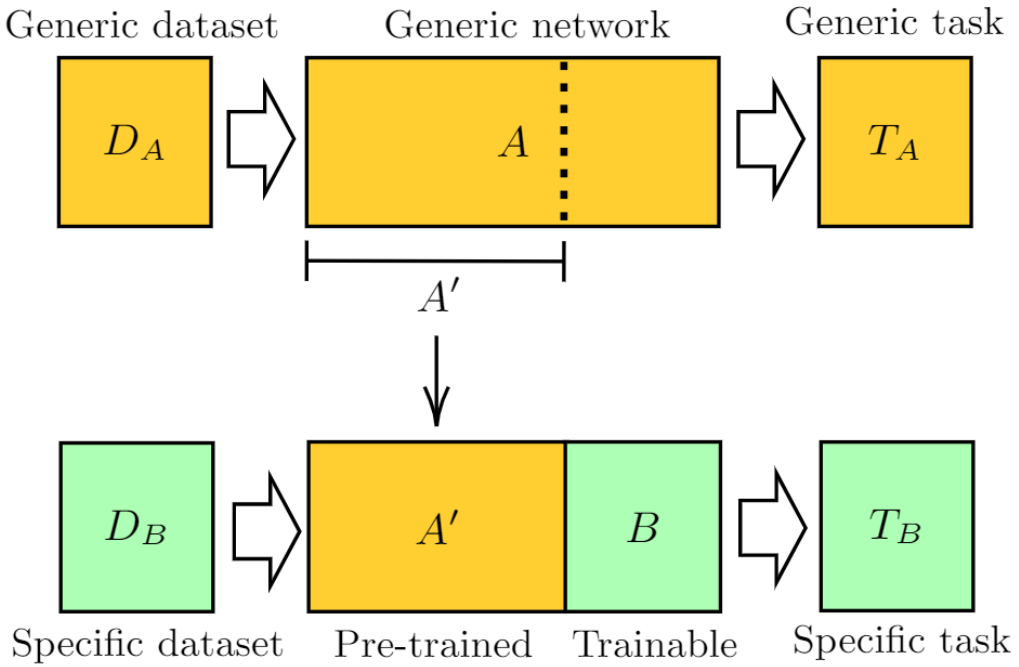
Pre-training & Transfer Learning

- Network pre-training for transfer learning
  - We train the backbone network using a large scale dataset, i.e., ImageNet
  - Learn generic feature extraction function
  - Trained weight will be reused as an initialization condition

- Why?
  - Insufficient training data for a new task
  - It is helpful to speed up the convergence

- How?
  - Replace end of networks and train the entire network or the new layers

- Is pre-training really helpful?



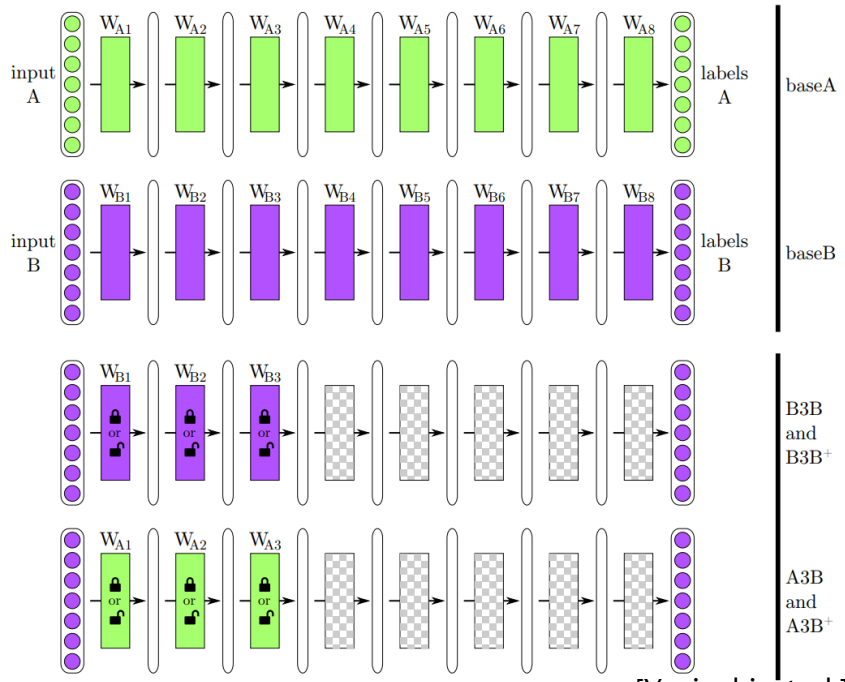
[Yosinski et al.]

Experiment Setup

- Split ImageNet dataset by half; about 500 classes for each set, A and B
  - Randomly-sampled split: A and B could have similar classes
  - Hierarchy-aware split: man-made (A) and natural (B) entities

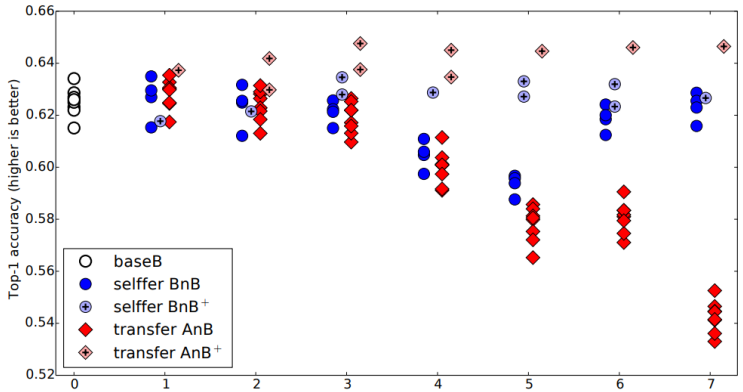
- Train two networks (8-layers) for each dataset respectably

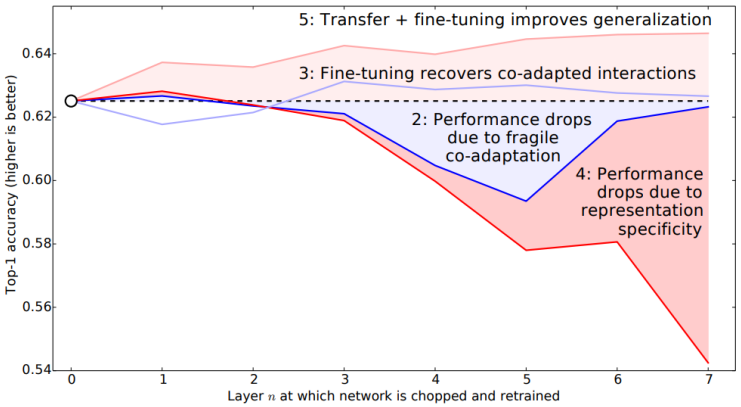
- Transfer the trained weight to the newer dataset
  - Notation: $n \in [1, 7]$
  - BnB(+)
    - Pretrained by B, then 1st to nth layers are transferred to B
    - Transferred weights are frozen (trained)
  - AnB(+)
    - Pretrained by A, then 1st to nth layers are transferred to B
    - Transferred weights are frozen (trained)



[Yosinski et al.]

Evaluation-BnB(+)



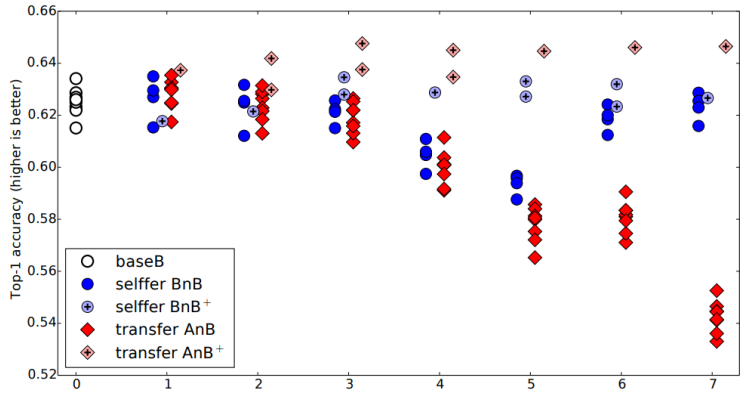


- BnB
  - Transferred weights are fixed, and only the initialized weights are trained
  - Performance drop when n = 4 or 5 due to “Fragile co-adaptation”
    - Feature interaction is complex or fragile. Thus the relationship couldn’t be learned without joint-training
    - Even though gradient descent is able to find a good local minimum, reaching to the global minimum is only possible when the layers are jointly trained

- BnB+
  - End-to-end training including the transferred layers
  - Comparable results in the original network

[Yosinski et al.]

Evaluation-AnB(+)



- AnB
  - Similar to BnB, but the learned domain is A
  - Worse than BnB
    - First couple of layers seem universal; there is no noticeable accuracy drop when $n \leq 2$
    - Near output layers (included in reused parts) are specific to the task. Thus, the near output layers of the existing network are specific to task A.
    - Without retraining, there is a large accuracy drop when $n > 2$, because the transferred weights are not useful for B
- AnB+
  - Transferred weights from A are also trained
  - Better results than baseB and BnB+
    - Pretrained weights contain additional functionality
    - Reused part (from task A) gives a good initial condition of training for task B

- AnB
  - Similar to BnB, but the learned domain is A
  - Worse than BnB
    - First couple of layers seem universal; there is no noticeable accuracy drop when $n \leq 2$
    - Near output layers (included in reused parts) are specific to the task. Thus, the near output layers of the existing network are specific to task A.
    - Without retraining, there is a large accuracy drop when $n > 2$, because the transferred weights are not useful for B
- AnB+
  - Transferred weights from A are also trained
  - Better results than baseB and BnB+
    - Pretrained weights contain additional functionality
    - Reused part (from task A) gives a good initial condition of training for task B

[Yosinski et al.]

Is Pre-training Always Essential?

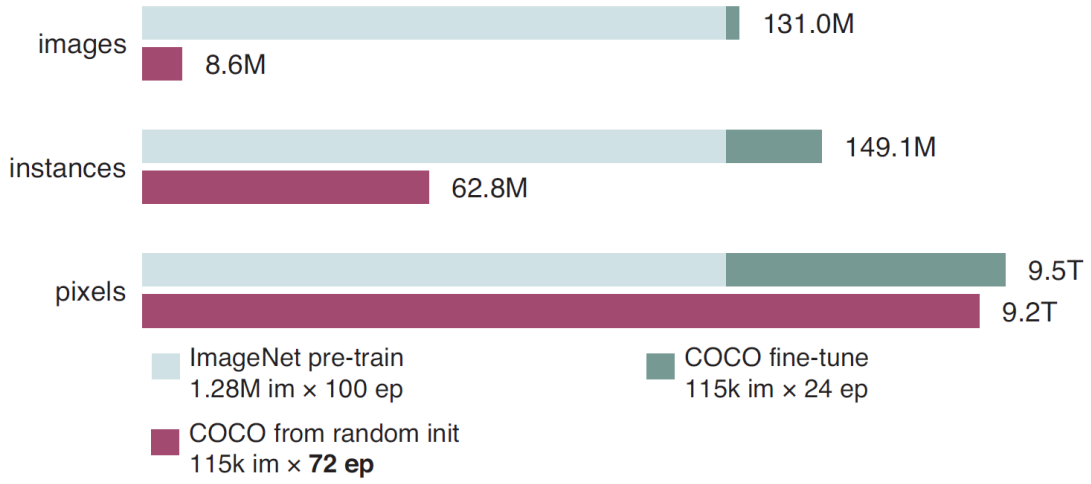
- ImageNet pre-training is often used for object detection
  - ImageNet dataset is large enough to prevent overfitting / learn good feature extractor
  - Many of pre-trained models are already existed
  - Minimize computation overhead with faster convergence

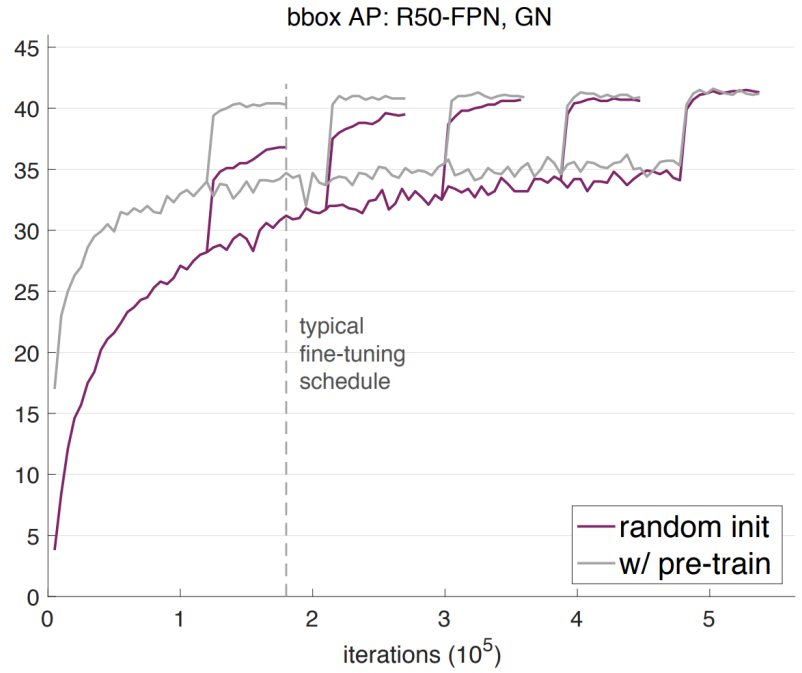
- There is big difference between classification & object detection
  - Global classification target
    vs localization-sensitive target

[He et al.]

Rethinking ImageNet Pre-training

- What if we train the model for object detection from scratch?
  - Use group normalization/synchronized BN for mitigating small batch issue
  - Train longer to learn both low- and high-level semantics
    - A large number of total samples (arguably in terms of pixels) are required

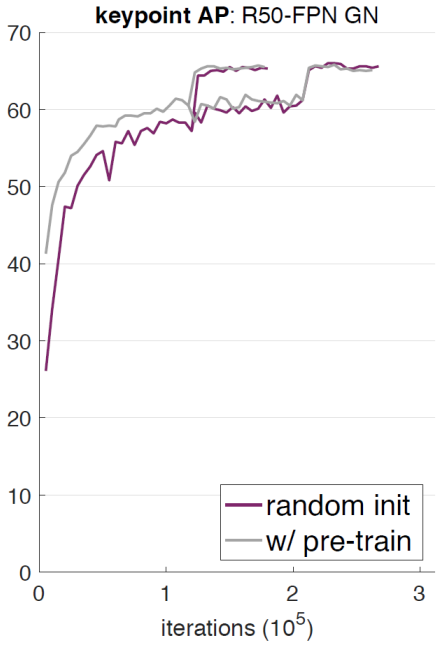




[He et al.]

Rethinking ImageNet Pre-training

- For the AP_{75}^{bbox} metric (using a high overlap threshold), training from scratch is better than fine-tuning by noticeable margins (1.0 or 0.8 AP)
- In Keypoint detection, the model trained from scratch can catch up more quickly
- ImageNet pre-training, which has little explicit localization information, does not help fine-grained localization



schedule		2×	3×	4×	5×	6×
R50	random init	36.8	39.5	40.6	40.7	41.3
	w/ pre-train	40.3	40.8	40.9	40.9	41.1
R101	random init	38.2	41.0	41.8	42.2	42.7
	w/ pre-train	41.8	42.3	42.3	41.9	42.2

Table 1. Object detection AP^{bbox} on COCO `val2017` of training schedules from 2× (180k iterations) to 6× (540k iterations). The model is Mask R-CNN with FPN and GN (Figures 1 and 3).

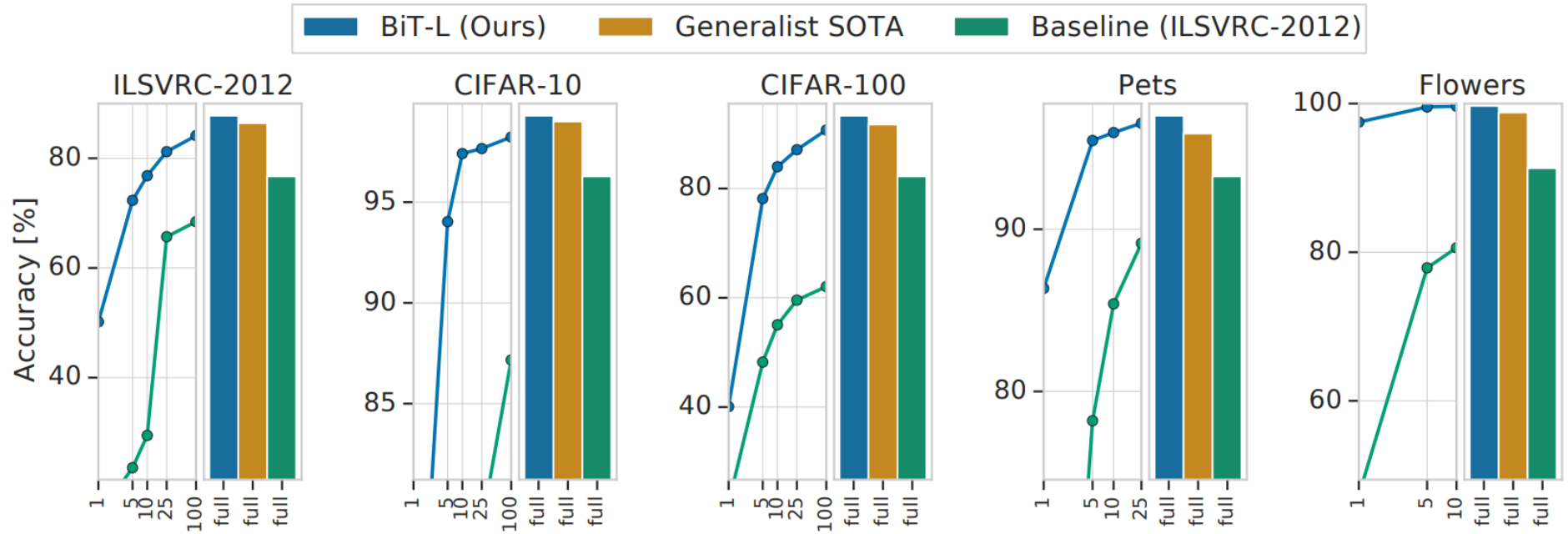
		AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
R50	random init	41.3	61.8	45.6	36.6	59.0	38.9
	w/ pre-train	41.1	61.7	44.6	36.4	58.5	38.7
	△	*+0.2*	*+0.1*	*+1.0*	*+0.2*	*+0.5*	*+0.2*
R101	random init	42.7	62.9	47.0	37.6	59.9	39.7
	w/ pre-train	42.3	62.6	46.2	37.2	59.7	39.7
	△	*+0.4*	*+0.3*	*+0.8*	*+0.4*	*+0.2*	*0.0*

Table 2. Training **from random initialization** *vs.* **with ImageNet pre-training** (Mask R-CNN with FPN and GN, Figures 1, 3), evaluated on COCO `val2017`. For each model, we show its results corresponding to the schedule (2 to 6×) that gives the best AP^{bbox}.

[He et al.]

Big Transfer (BiT)

- A simple paradigm of knowledge distillation
  - Pre-train on a large supervised source dataset and fine-tune the weights on the target task
  - Big Transfer (BiT)
    - Try to maximize the benefit of knowledge distillation by training the network with enormous amount of dataset



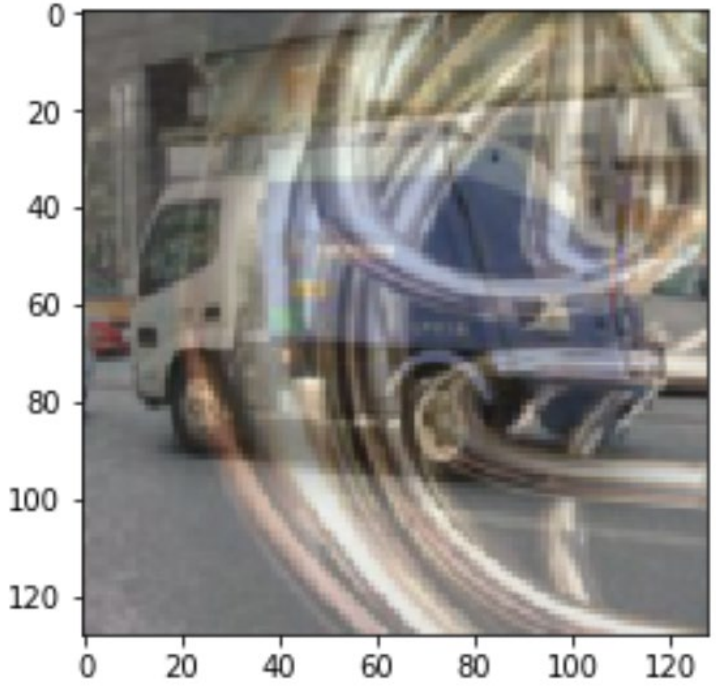
Pre-training Setup

- Three large datasets
  - BiT-S: ILSVRC2012 with 1.3M images, 1k classes
  - BiT-M ImageNet-21k with 14M images, 21k classes
  - BiT-L: JFT-300M with 300M images, 18291 classes
    - Automated labeling - ~20 % of the labels are noisy

- Group Normalization + Weight Standardization
  - Weakness of batch normalization: small per-gpu batch due to memory constraints
  - Ready for high-resolution transfer

Fine-tuning Setup

- Per-task hyper-parameter tuning is essential
  - Training schedule length
  - Resolution
  - Whether to use MixUp regularization



Results

Table 2: Improvement in accuracy when pre-training on the public ImageNet-21k dataset over the "standard" ILSVRC-2012. Both models are ResNet152x4.

	ILSVRC-2012	CIFAR-10	CIFAR-100	Pets	Flowers	VTAB-1k (19 tasks)
BiT-S (ILSVRC-2012)	81.30	97.51	86.21	93.97	89.89	66.87
BiT-M (ImageNet-21k)	85.39	98.91	92.17	94.46	99.30	70.64
Improvement	+4.09	+1.40	+5.96	+0.49	+9.41	+3.77

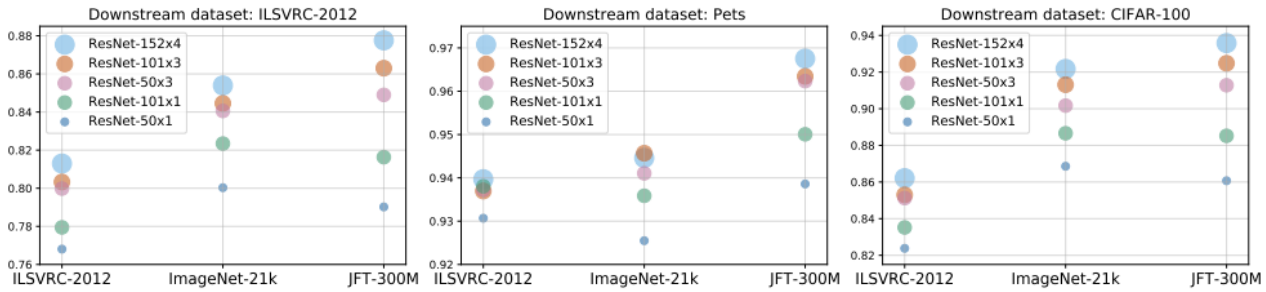


Fig. 5: Effect of upstream data (shown on the x-axis) and model size on downstream performance. Note that exclusively using more data or larger models may hurt performance; instead, both need to be increased in tandem.

	BiT-L	Generalist SOTA
ILSVRC-2012	**87.54 ± 0.02**	86.4 [57]
CIFAR-10	**99.37 ± 0.06**	99.0 [19]
CIFAR-100	**93.51 ± 0.08**	91.7 [55]
Pets	**96.62 ± 0.23**	95.9 [19]
Flowers	**99.63 ± 0.03**	98.8 [55]
VTAB (19 tasks)	**76.29 ± 1.70**	70.5 [58]

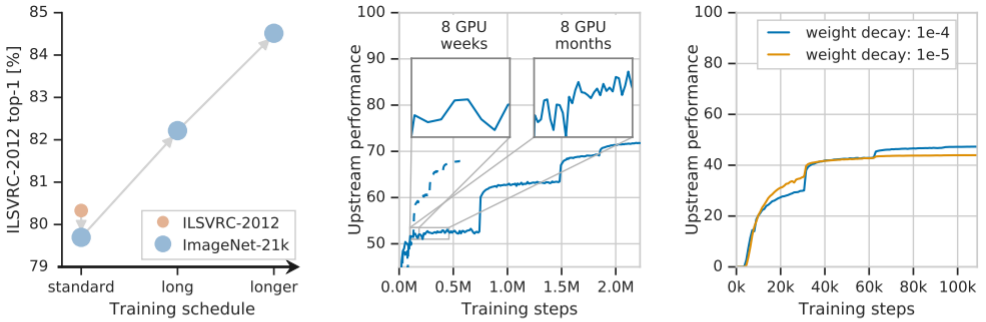


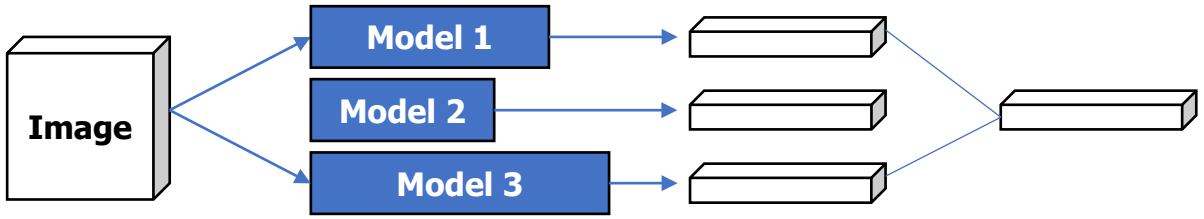
Fig. 7: **Left:** Applying the "standard" computational budget of ILSVRC-2012 to the larger ImageNet-21k seems detrimental. Only when we train longer (3x and 10x) do we see the benefits of training on the larger dataset. **Middle:** The learning progress of a ResNet-101x3 on JFT-300M seems to be flat even after 8 GPU-weeks, but after 8 GPU-months progress is clear. If one decays the learning rate too early (dashed curve), final performance is significantly worse. **Right**: Faster initial convergence with lower weight decay may trick the practitioner into selecting a sub-optimal value. Higher weight decay converges more slowly, but results in a better final model.

Basics of Knowledge Distillation

Motivation of Dark Knowledge -1

- Ensemble of models
  - The easiest way to extract a lot of knowledge
  - Make the models as different as possible to minimize the correlations
    - Different initializations / different architectures / different subsets of the training data
  - It is helpful to over-fit the individual models.

- Problem?
  - A test time, we average the predictions of all the models
  - A big ensemble is highly redundant.
  - We want to minimize the amount of computation and memory footprint.

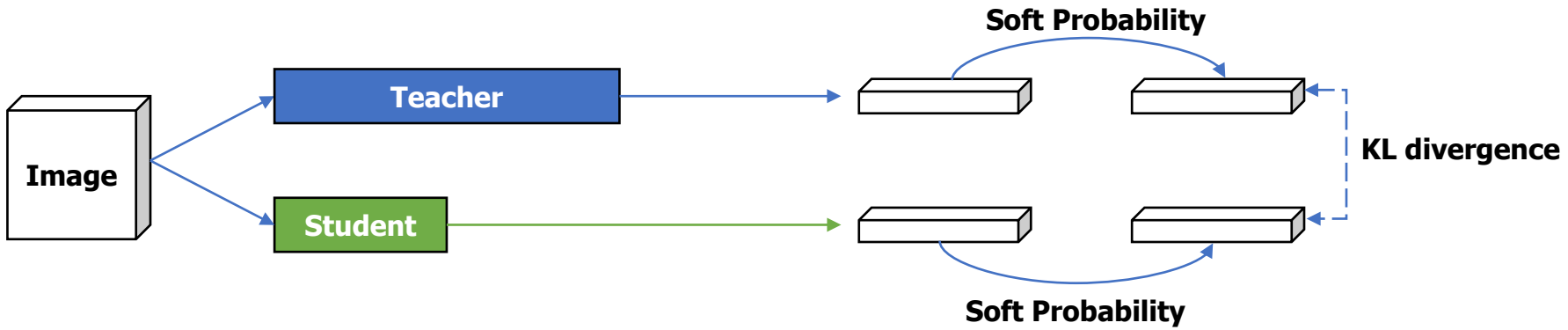


[Hinton, 2014]

Motivation of Dark Knowledge -2

- Main Idea
  - The ensemble implements a function from input to output.
  - Can we transfer the knowledge in the function into a single smaller model?

  - Know Distillation
    - Matching the output of source and target models
    - Introduce new loss function based on MSE loss or KL divergence between teacher output & student output in addition to the classification loss (cross-entropy loss)



[Hinton, 2014]

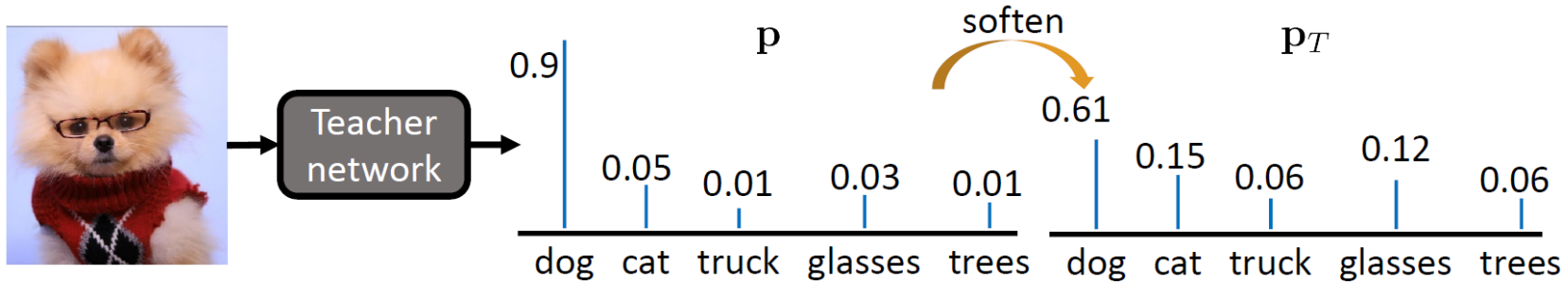
Soft Probability with Temperature

- Use temperature $T \geq 1$ to make a soft probability distribution

$$q_{i,T} = \frac{\exp\left(\frac{z_i}{T}\right)}{\Sigma_j \exp\left(\frac{z_j}{T}\right)},$$

  - z_i, q_i is i-th logit and probability, respectably

- Use the soft target in addition to the original classification loss
  - $L = (1-\alpha)L_{ce}(y, q) + \alpha T^2 L_{ce}(p_T, q_T)$
  - y, q and p are ground-truth labels, target model and source model outputs
  - Adopt scaling T^2 to match the relevance range of output

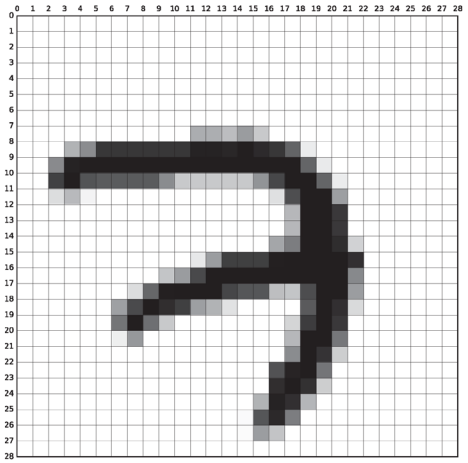


[Hinton, 2014]

MNIST Experiments

- Simple verification based on MLP-based MNIST classification network
  - Source model: 2 hidden layers MLP with 1200 hidden nodes
    - Accuracy: 99.33 %
  - Target model: 2 hidden layers MLP with 800 hidden nodes
    - Accuracy: 98.54 %
    - Accuracy w/ distillation: 99.26 %

  - # of computation reduction?

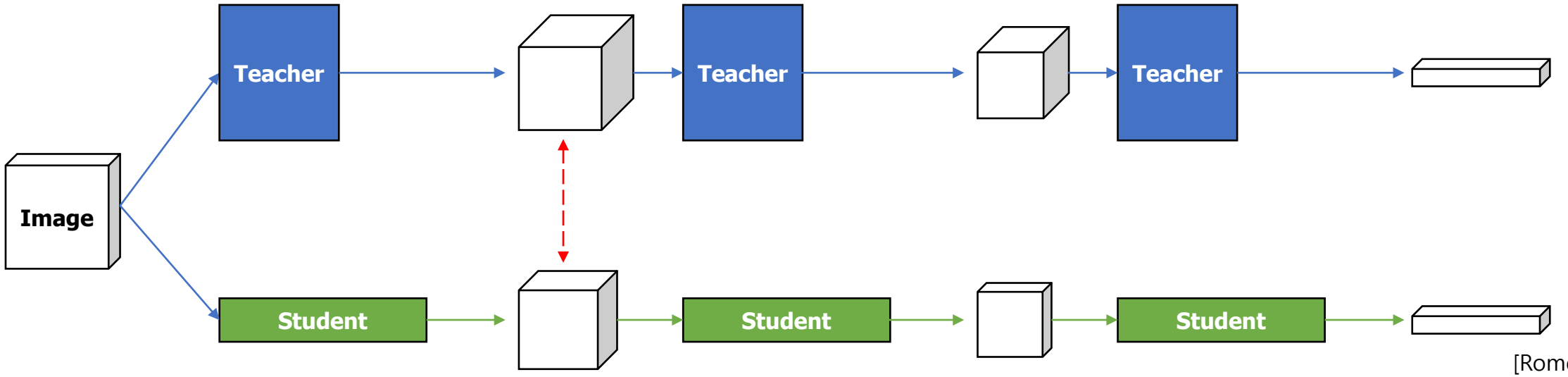


(**a**) MNIST sample belonging to the digit '7'.

(**b**) 100 samples from the MNIST training set.

Toward Deeper/Shallower Student

- Deeper network can give better accuracy
- Can deeper, but shallower student be trained?
  - In order to enhance feature map quality, intermediate knowledge distillation is required
  - Problem? feature-map size mismatch

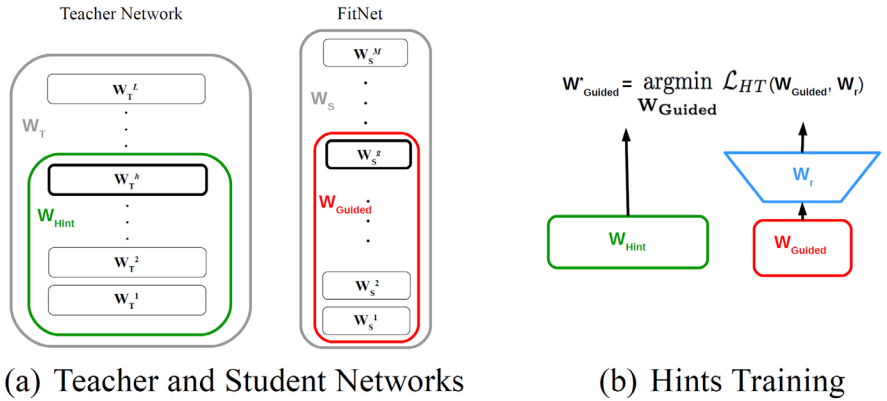


[Romero, 2015]

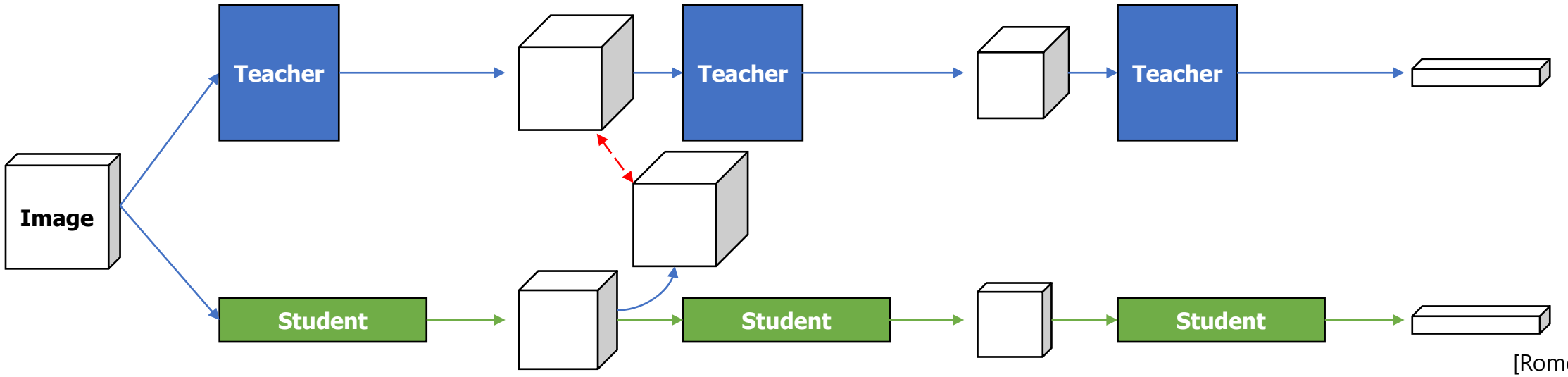
FitNet – 1

- Resolve feature size mismatch by introducing 1x1 convolution-based regressor

- FitNets training pipeline design
  - Step 1: Hint-based training
    - Train up to guided layer



(a) Teacher and Student Networks (b) Hints Training

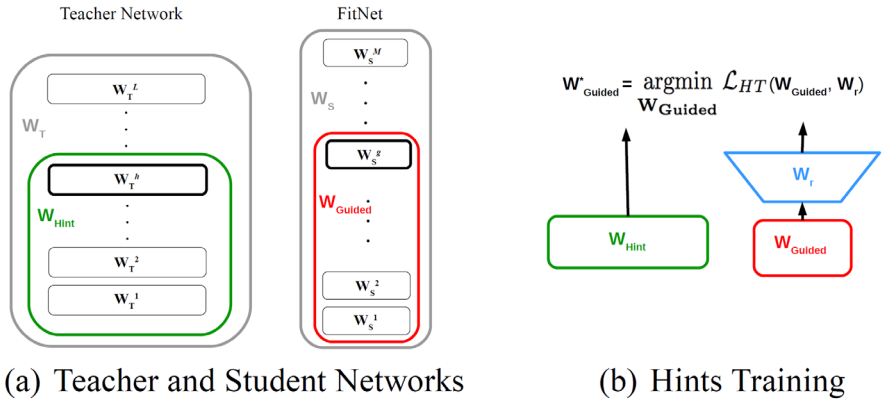


[Romero, 2015]

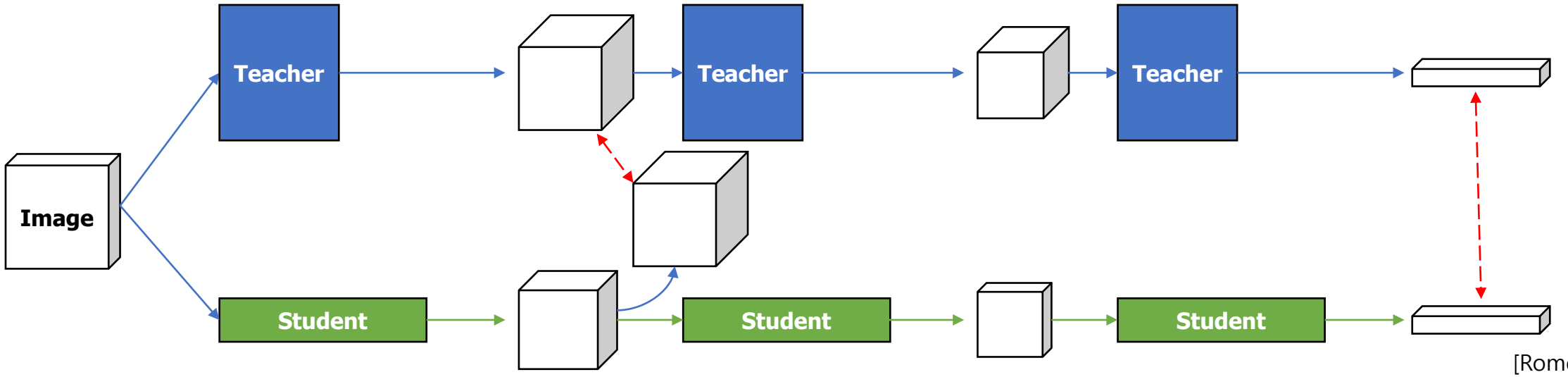
FitNet - 2

- Resolve feature size mismatch by introducing 1x1 convolution-based regressor

- FitNets training pipeline design
  - Step 1: Hint-based training
    - Train up to guided layer
  - Step 2: Knowledge distillation with soft probability



(a) Teacher and Student Networks (b) Hints Training



[Romero, 2015]

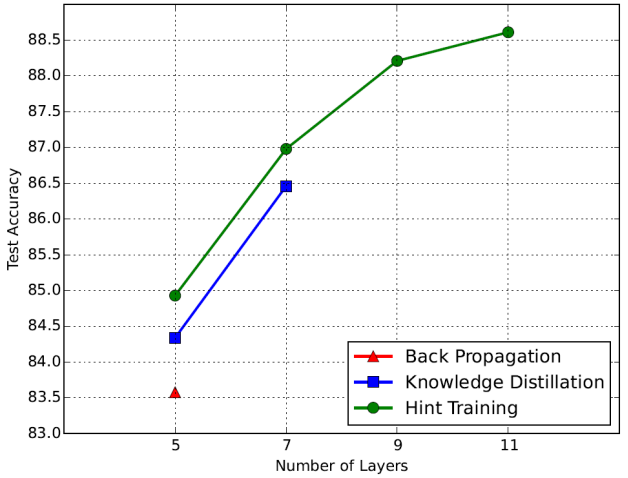
FitNet Results

- 17-layer student
  - 11-th layer is guided by 2nd layer

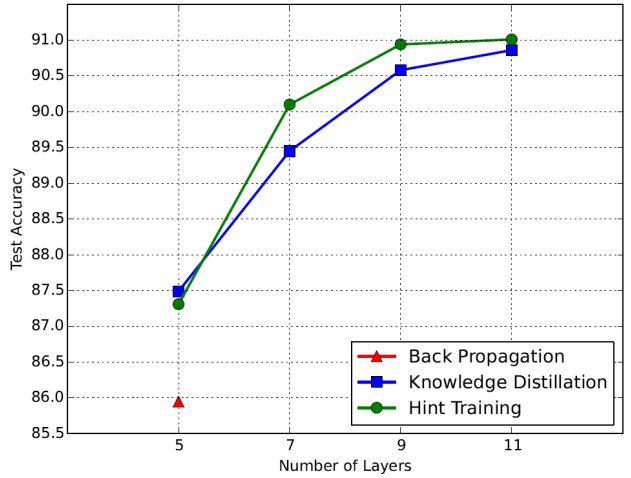
Algorithm	# params	Accuracy
Compression		
FitNet	~2.5M	**91.61%**
Teacher	~9M	90.18%
Mimic single	~54M	84.6%
Mimic single	~70M	84.9%
Mimic ensemble	~70M	85.8%
State-of-the-art methods		
Maxout		90.65%
Network in Network		91.2%
Deeply-Supervised Networks		**91.78%**
Deeply-Supervised Networks (19)		88.2%

Table 1: Accuracy on CIFAR-10

Hint-based training enables deeper and better students w/ computational budget



(a) 30M Multiplications



(b) 107M Multiplications

FitNet gives trade-off between accuracy and speed (on GPU)

Network	# layers	# params	# mult	Acc	Speed-up	Compression rate
Teacher	5	~9M	~725M	90.18%	1	1
FitNet 1	11	~250K	~30M	89.01%	**13.36**	**36**
FitNet 2	11	~862K	~108M	91.06%	4.64	10.44
FitNet 3	13	~1.6M	~392M	91.10%	1.37	5.62
FitNet 4	19	~2.5M	~382M	**91.61%**	1.52	3.60

Table 5: Accuracy/Speed Trade-off on CIFAR-10.

[Romero, 2015]

Advanced Methods of Knowledge Distillation

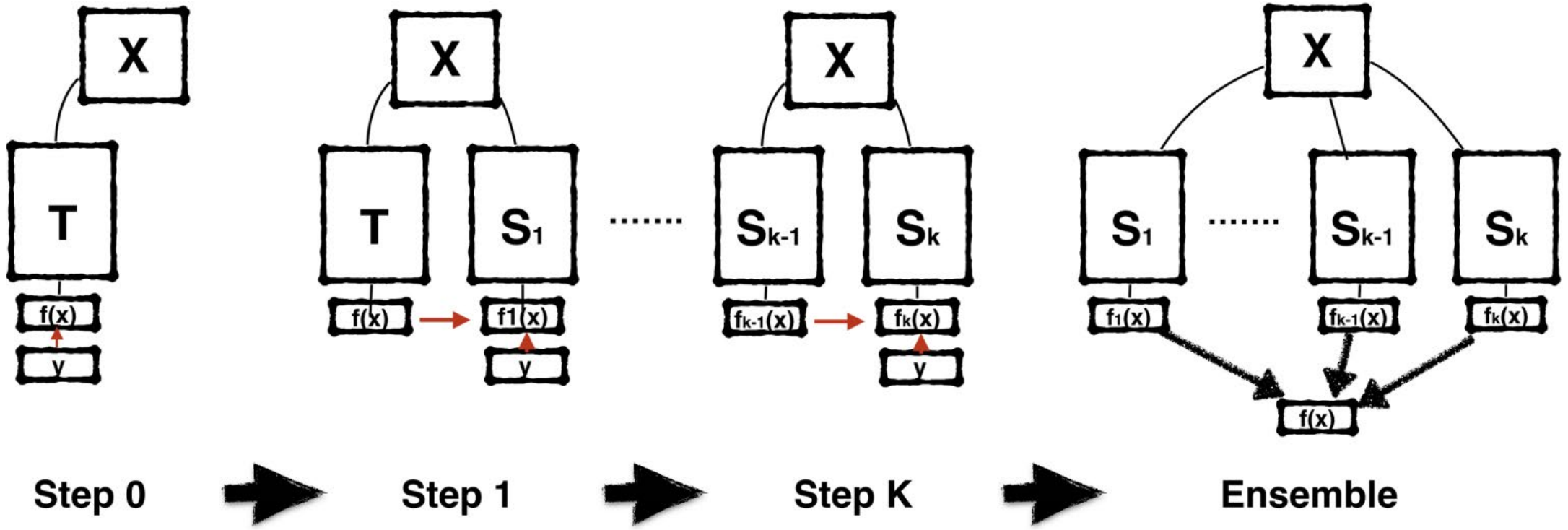
Self-Teaching Method

- There are multiple interpretation about knowledge distillation
  - Simplified ensemble of multiple networks
  - Regularization for the student network
  - Labeling noise mitigation

- Simple question – What if we use the teacher network as oneself?

Born Again Neural Networks

- $Loss = \min_{\theta_k} L\left(f\left(x, \arg\min_{\theta_{k-1}} L\left(y, f(x, \theta_{k-1})\right)\right), f(x, \theta_k)\right)$
  - Note that there is no cross-entropy loss for label



BAN Result

Table 2. **Test error on CIFAR-100** *Left Side:* DenseNet of different depth and growth factor and respective BAN student. BAN models are trained only with the teacher loss, BAN+L with both label and teacher loss. CWTM are trained with sample importance weighted label, the importance of the sample is determined by the max of the teacher's output. DKPP are trained only from teacher outputs with all the dimensions but the argmax permuted. *Right Side:* test error on CIFAR-100 sequence of BAN-DenseNet, and the BAN-ensembles resulting from the sequence. Each BAN in the sequence is trained from cross-entropy with respect to the model at its left. BAN and BAN-1 models are trained from Teacher but have different random seeds. We include the teacher as a member of the ensemble for Ens*3 for 80-120 since we did not train a BAN-3 for this configuration.

Network	Teacher	BAN	BAN+L	CWTM	DKPP	BAN-1	BAN-2	BAN-3	Ens*2	Ens*3
DenseNet-112-33	18.25	**16.95**	17.68	17.84	17.84	17.61	17.22	**16.59**	15.77	15.68
DenseNet-90-60	17.69	**16.69**	16.93	17.42	17.43	16.62	**16.44**	16.72	15.39	15.74
DenseNet-80-80	17.16	**16.36**	16.5	17.16	16.84	16.26	16.30	**15.5**	15.46	15.14
DenseNet-80-120	16.87	**16.00**	16.41	17.12	16.34	**16.13**	16.13	/	**15.13**	**14.9**

Motivation of Label Refinery

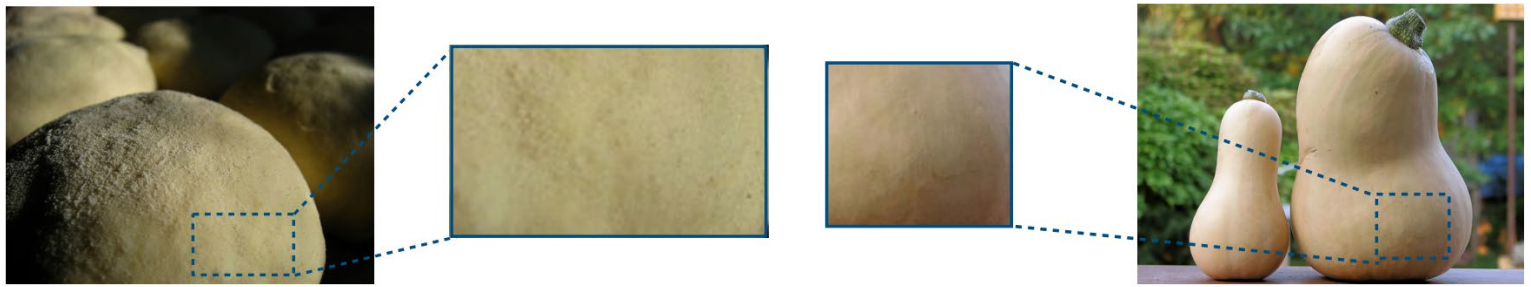
- 3 major challenges of current labeling principles and practices
  - Incompleteness
    - A natural image of a particular category will contain other object categories as well

(a) (b)

Fig. 2: Figure 2(a) shows a sample image from the "persian cat" category of ImageNet's training set. The standard technique to train modern state-of-the-art architectures is to crop patches as small as 8% area of the original image, and label them with the original image's label. This will often result in inaccurate labels for the augmented data. Figure 2(b) shows a sample crop of the original image where the "persian cat" is no longer in the crop. A trained ResNet-50 labels Figure 2(a) by "persian cat", and labels Figure 2(b) by "golf ball". We claim that using a model to generate labels for the patches results in more accurate labels and therefore more accurate models.

Motivation of Label Refinery

- 3 major challenges of current labeling principles and practices
  - Incompleteness
    - A natural image of a particular category will contain other object categories as well
  - Taxonomy Dependency:
    - Categories that are far from each other in the taxonomy structure can be very similar visually
  - Inconsistency
    - Strong augmentation, e.g. cropping, make the object of interest no longer visible

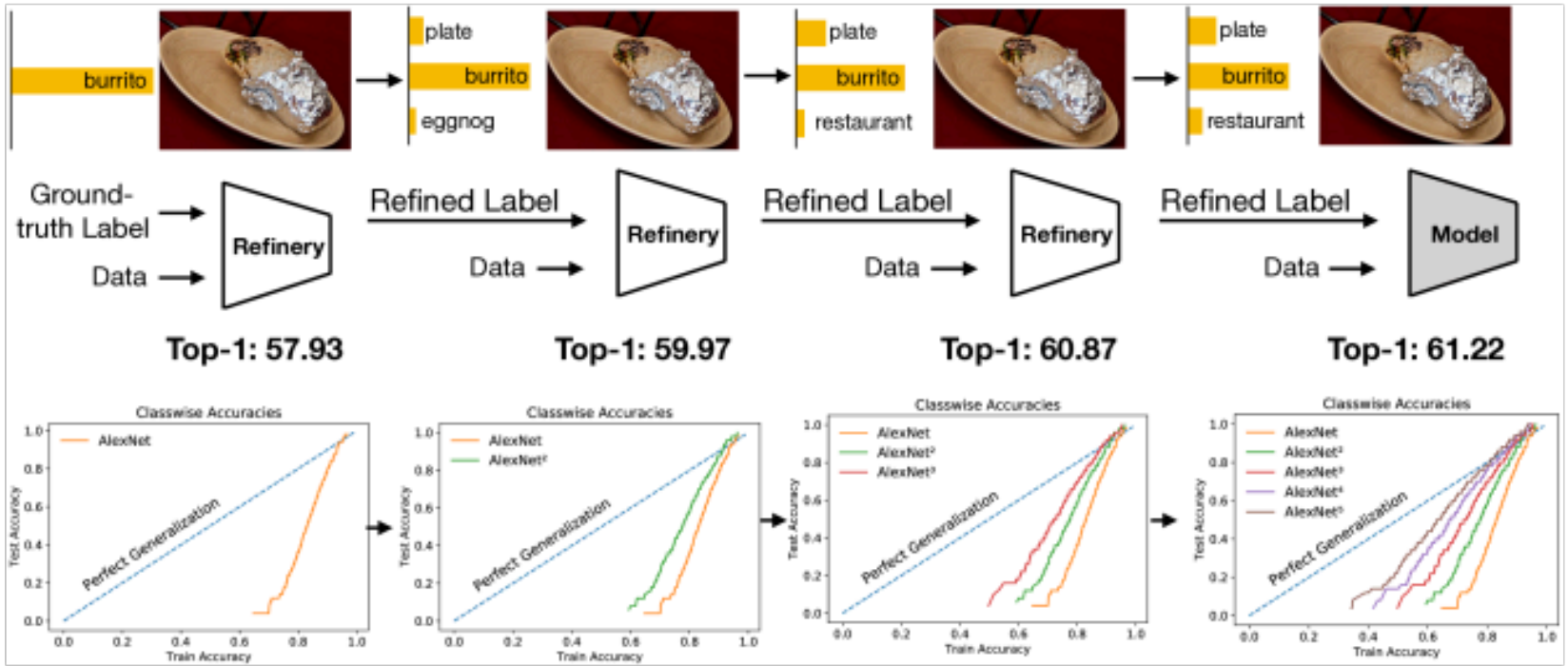


(a) (b)

Loss Function with Label Refinery

- Minimize KL-divergence between Label Refinery and target network

$$L_t(f(X_i)) = -\sum_c p_c^{t-1}(f(X_i)) \log \left(\frac{p_c^t(f(X_i))}{p_c^{t-1}(f(X_i))} \right) \tag{1}$$
$$= -\sum p_c^{t-1}(f(X_i)) \log p_c^t(f(X_i)) + \sum p_c^{t-1}(f(X_i)) \log p_c^{t-1}(f(X_i))$$



Label Refinery Results

Model	Top-1	Top-5
AlexNet	57.93	79.41
AlexNet^2	59.97	81.44
AlexNet^3	60.87	82.13
AlexNet^4	61.22	**82.56**
AlexNet^5	**61.37**	**82.56**

Model	Top-1	Top-5
ResNet50	75.7	92.81
ResNet50^2	**76.5**	**93.12**

Model	Top-1	Top-5
MobileNet	68.51	88.13
MobileNet^2	**69.52**	**88.7**

Model	Top-1	Top-5
VGG16	70.1	88.54
VGG16^2	71.85	90.07
VGG16^3	**72.49**	**90.76**

Model	Top-1	Top-5
VGG19	71.39	89.44
VGG19^2	72.66	90.75
VGG19^3	**73.32**	**91.30**

Model	Top-1	Top-5
Darknet19	70.6	89.13
Darknet19^2	72.74	90.73
Darknet19^3	**73.01**	**90.92**

Model	Paper Number		Our Impl.		Label Refinery	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
AlexNet [8]	59.3	81.8	57.93	79.41	**66.28**†	**86.13**†
MobileNet [28]	70.6	N/A	68.53	88.14	**73.39**	**91.07**
MobileNet0.75 [28]	68.4	N/A	65.93	86.28	**70.92**	**89.68**
MobileNet0.5 [28]	63.7	N/A	63.03	84.55	**66.66**†	**87.07**†
MobileNet0.25 [28]	50.6	N/A	50.65	74.42	**54.62**†	**77.92**†
ResNet-50 [5]	N/A	N/A	75.7	92.81	**76.5**	**93.12**
ResNet-34 [5]	N/A	N/A	73.39	91.32	**75.06**	**92.35**
ResNet-18 [5]	N/A	N/A	69.7	89.26	**72.52**	**90.73**
ResNetXnor-50 [32]	N/A	N/A	63.1	83.61	**70.34**	**89.18**
VGG16 [6]	73	91.2	70.1	88.54	**75**	**92.22**
VGG19 [6]	72.7	91	71.39	89.44	**75.46**	**92.52**
Darknet19 [33]	72.9	91.2	70.6	89.13	**74.47**	**91.94**

Label Refinery Results