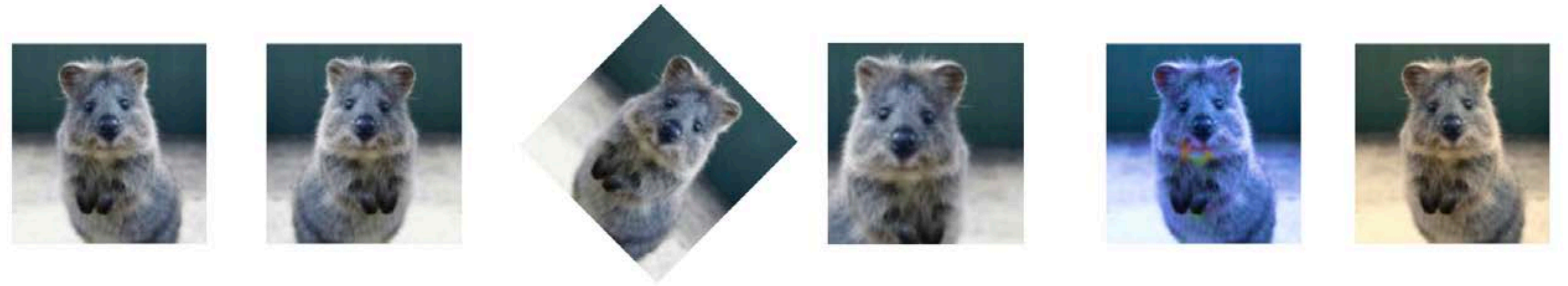
TRAINING FROM SMALL-SIZE DATA

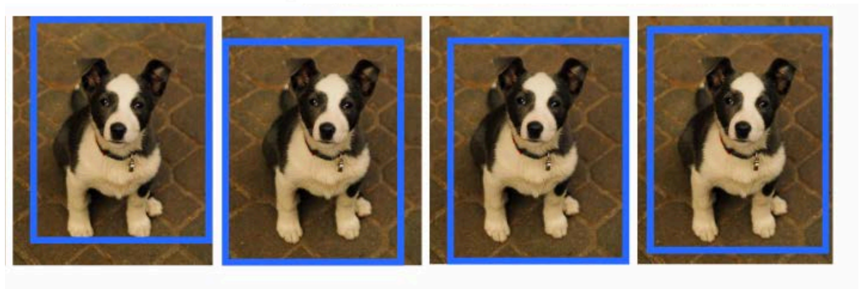
Outline

- Data augmentation
  - w/ Assignment 2
- Transfer learning
- Domain adaptation
- Semi-supervised learning
- Active learning / Uncertainty measure

Data augmentation: basics

- The more training samples are, the better the result will be
  - However, hard to get many samples with true labels
- We create 'new' samples from existing ones = data augmentation
- All sorts of image transformation that do not change 'contents'
  - Crop, horizontal/vertical flip
  - Geometric warp: scaling, sheer, etc.
  - Color changes



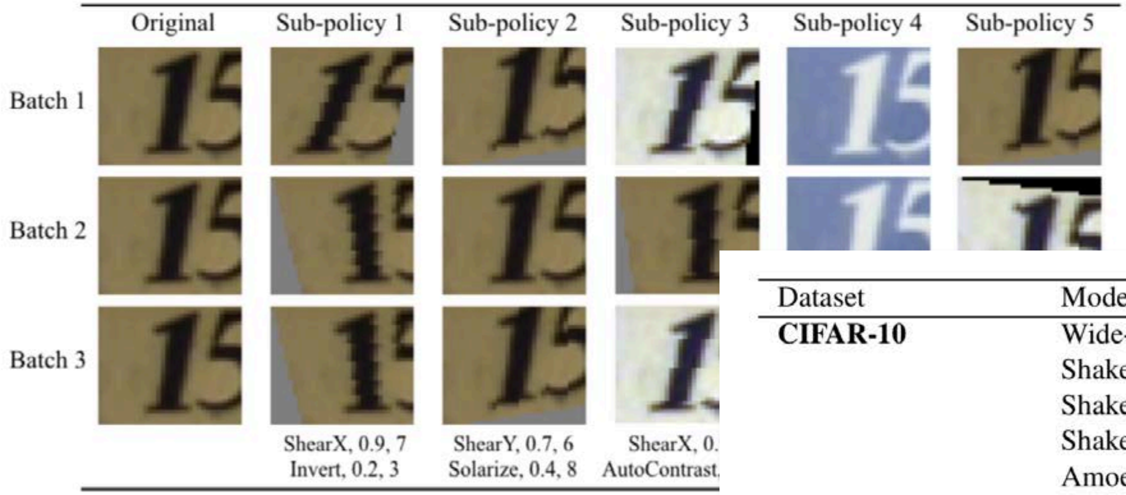


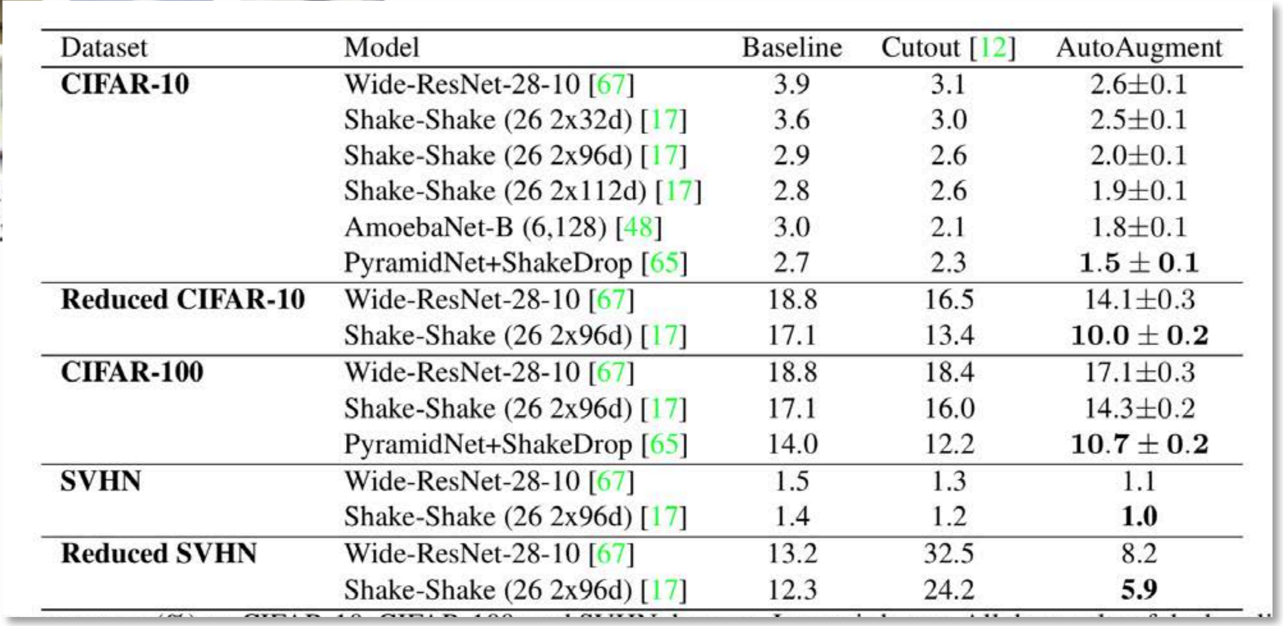
[Karianakis+2015]

AutoAugment

Cubuk+(Google Brain), AutoAugment: Learning Augmentation Strategies from Data, CVPR2019

- Search for the best combination of pre-defined image transformations; some of them have parameters need to be set
  - ShearX/Y, TranslateX/Y, Rotate, AutoContrast, Invert, Equalize, Solarize, Posterize, Contrast, Color, Brightness, Sharpness, Cutout, Sample Pairing
- Reinforcement learning is employed to do this search



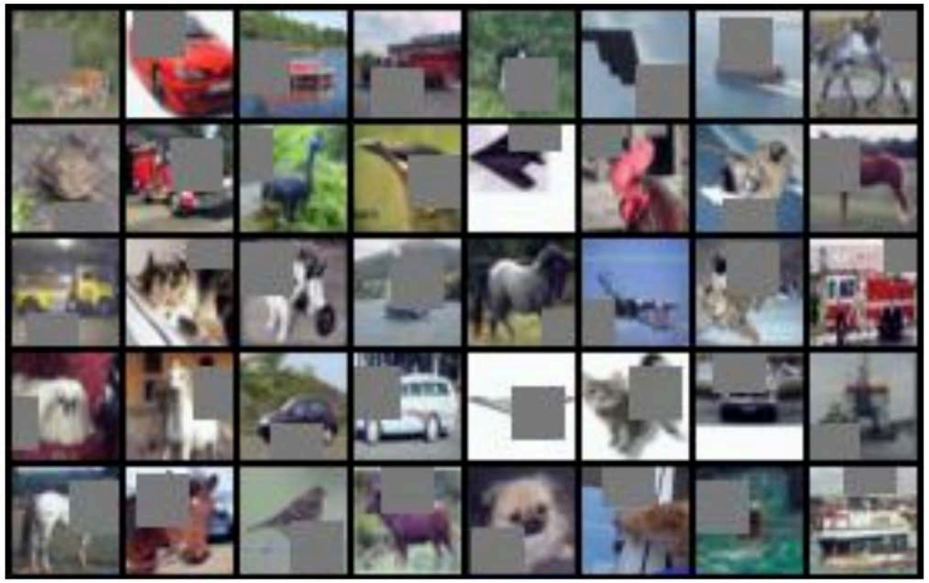


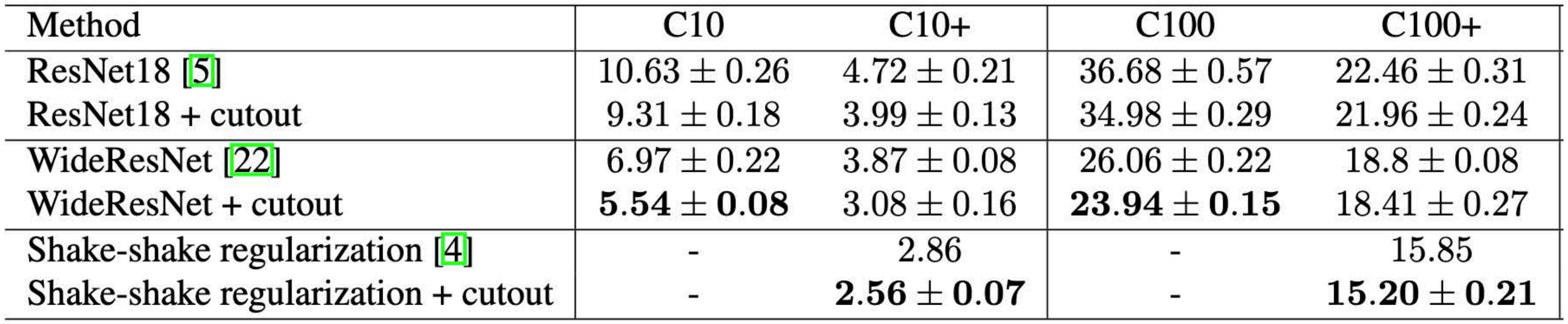
Dataset	Model	Baseline	Cutout [12]	AutoAugment
CIFAR-10	Wide-ResNet-28-10 [67]	3.9	3.1	2.6±0.1
	Shake-Shake (26 2x32d) [17]	3.6	3.0	2.5±0.1
	Shake-Shake (26 2x96d) [17]	2.9	2.6	2.0±0.1
	Shake-Shake (26 2x112d) [17]	2.8	2.6	1.9±0.1
	AmoebaNet-B (6,128) [48]	3.0	2.1	1.8±0.1
	PyramidNet+ShakeDrop [65]	2.7	2.3	**1.5 ± 0.1**
Reduced CIFAR-10	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	**10.0 ± 0.2**
CIFAR-100	Wide-ResNet-28-10 [67]	18.8	18.4	17.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	16.0	14.3±0.2
	PyramidNet+ShakeDrop [65]	14.0	12.2	**10.7 ± 0.2**
SVHN	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	**1.0**
Reduced SVHN	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	**5.9**

Cutout

Devries+, Improved Regularization of Convolutional Neural Networks with Cutout, arXiv2017

- Mask a part of the image with randomly generated gray square
- Some relation to dropout





Method	C10	C10+	C100	C100+
ResNet18 [5]	10.63 ± 0.26	4.72 ± 0.21	36.68 ± 0.57	22.46 ± 0.31
ResNet18 + cutout	9.31 ± 0.18	3.99 ± 0.13	34.98 ± 0.29	21.96 ± 0.24
WideResNet [22]	6.97 ± 0.22	3.87 ± 0.08	26.06 ± 0.22	18.8 ± 0.08
WideResNet + cutout	$\mathbf{5.54 \pm 0.08}$	3.08 ± 0.16	$\mathbf{23.94 \pm 0.15}$	18.41 ± 0.27
Shake-shake regularization [4]	-	2.86	-	15.85
Shake-shake regularization + cutout	-	$\mathbf{2.56 \pm 0.07}$	-	$\mathbf{15.20 \pm 0.21}$

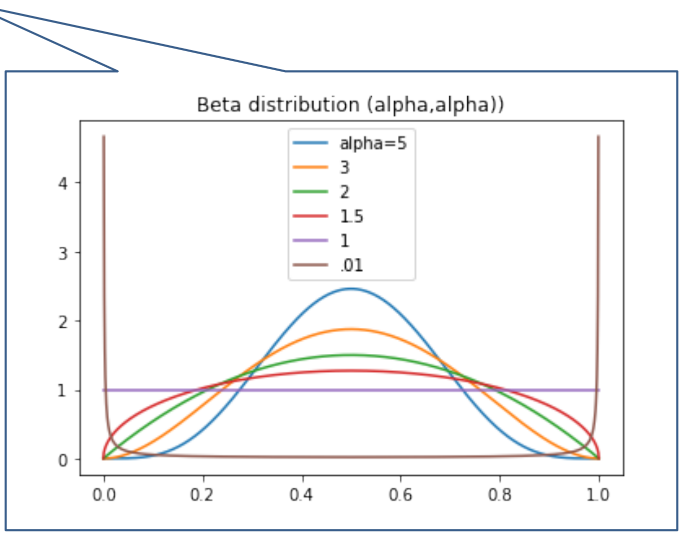
Mixup

Zhang+, mixup: Beyond empirical risk minimization, ICLR18

- Pick two samples w/ different labels and *mix up* them as follows
  - λ is a random number sampled from a beta distribution β(α,α)

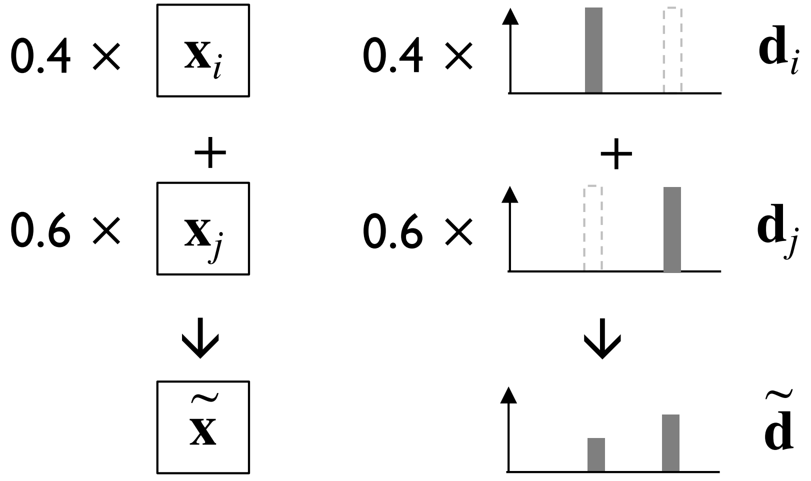
$$\tilde{\mathbf{x}} = \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j$$

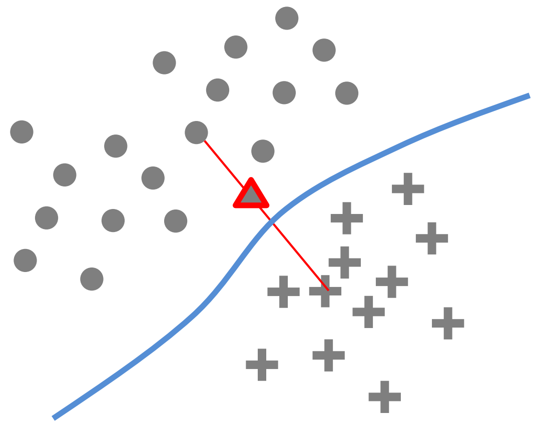
$$\tilde{\mathbf{d}} = \lambda \mathbf{d}_i + (1-\lambda)\mathbf{d}_j$$



Input:

Desired output:



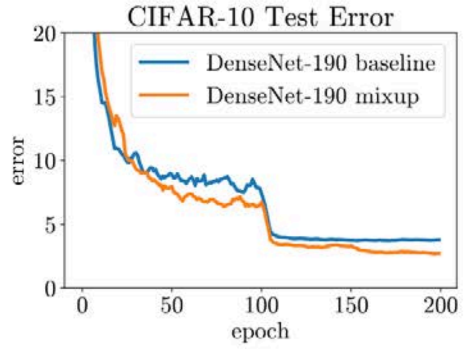


Mixup

Zhang+, mixup: Beyond empirical risk minimization, ICLR18

Dataset	Model	ERM	*mixup*
CIFAR-10	PreAct ResNet-18	5.6	**4.2**
	WideResNet-28-10	3.8	**2.7**
	DenseNet-BC-190	3.7	**2.7**
CIFAR-100	PreAct ResNet-18	25.6	**21.1**
	WideResNet-28-10	19.4	**17.5**
	DenseNet-BC-190	19.0	**16.8**

(a) Test errors for the CIFAR experiments.



(b) Test error evolution for the best ERM and *mixup* models.

Figure 3: Test errors for ERM and *mixup* on the CIFAR experiments.

Label corruption	Method	Test error		Training error	
		Best	Last	Real	Corrupted
20%	ERM	12.7	16.6	0.05	0.28
	ERM + dropout ($p = 0.7$)	8.8	10.4	5.26	83.55
	mixup ($\alpha = 8$)	**5.9**	6.4	2.27	86.32
	mixup + dropout ($\alpha = 4, p = 0.1$)	6.2	**6.2**	1.92	85.02
50%	ERM	18.8	44.6	0.26	0.64
	ERM + dropout ($p = 0.8$)	14.1	15.5	12.71	86.98
	mixup ($\alpha = 32$)	11.3	12.7	5.84	85.71
	mixup + dropout ($\alpha = 8, p = 0.3$)	**10.9**	**10.9**	7.56	87.90
80%	ERM	36.5	73.9	0.62	0.83
	ERM + dropout ($p = 0.8$)	30.9	35.1	29.84	86.37
	mixup ($\alpha = 32$)	25.3	30.9	18.92	85.44
	mixup + dropout ($\alpha = 8, p = 0.3$)	**24.0**	**24.8**	19.70	87.67

Table 2: Results on the corrupted label experiments for the best models.

Assignments 2

- Mission: Apply data augmentation to MNIST digit classification and analyze its effect
  - Send your submission to okatani@vision.is.tohoku.ac.jp by Nov. 18

- Minimum requirements:
  - Use at least 5 different augmentation methods using torchvision.transforms; you can find the definition of affine transformation here https://en.wikipedia.org/wiki/Affine_transformation
  - Choose a network and train it on 1,000 samples with each of the augmentation methods
  - Observe your results and explain what you have found
  - Don't forget to report the augmentation methods you tested

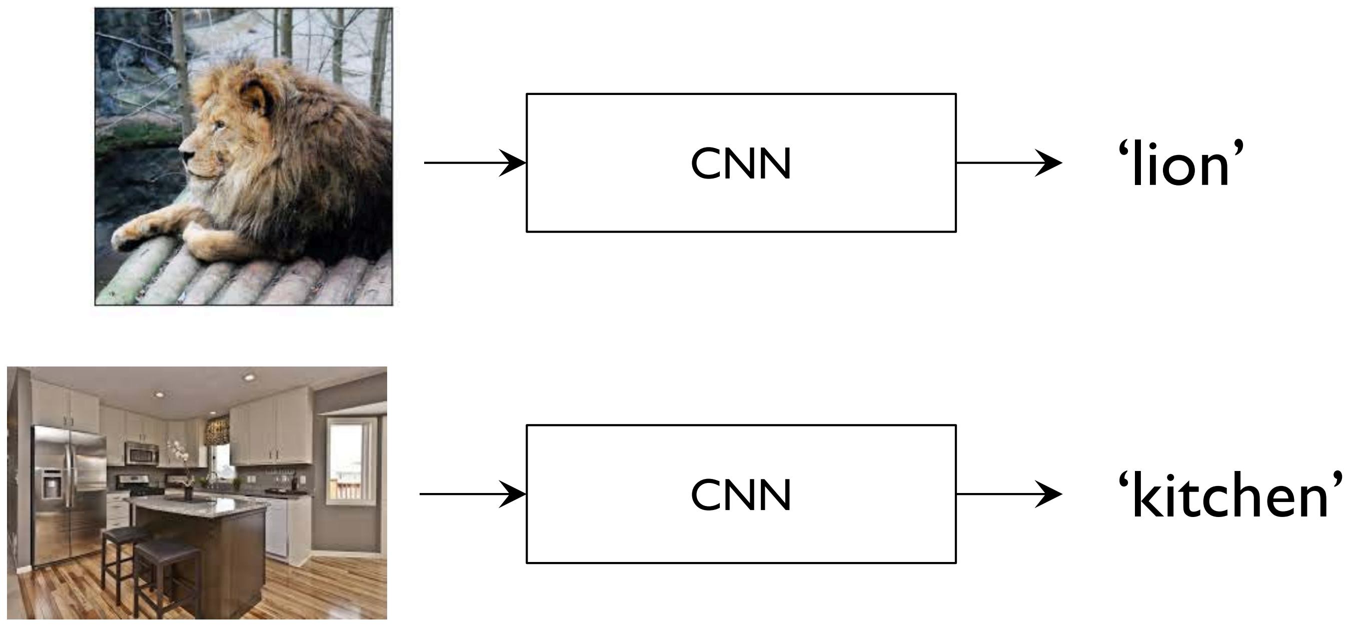
```
transform = transforms.Compose([transforms.RandomAffine((-10.0, 10.0)),
                                transforms.ToTensor(),
                                transforms.Normalize((0.1307,), (0.3081,))])
```

- Optional (5% additional score will be given if you accomplish this):
  - Analyze the effect of *Mixup* on MNIST; a sample code is here

https://drive.google.com/open?id=16FR37YVBEOIGoi0y25wD5ypf4elaZ7nb

Transfer Learning

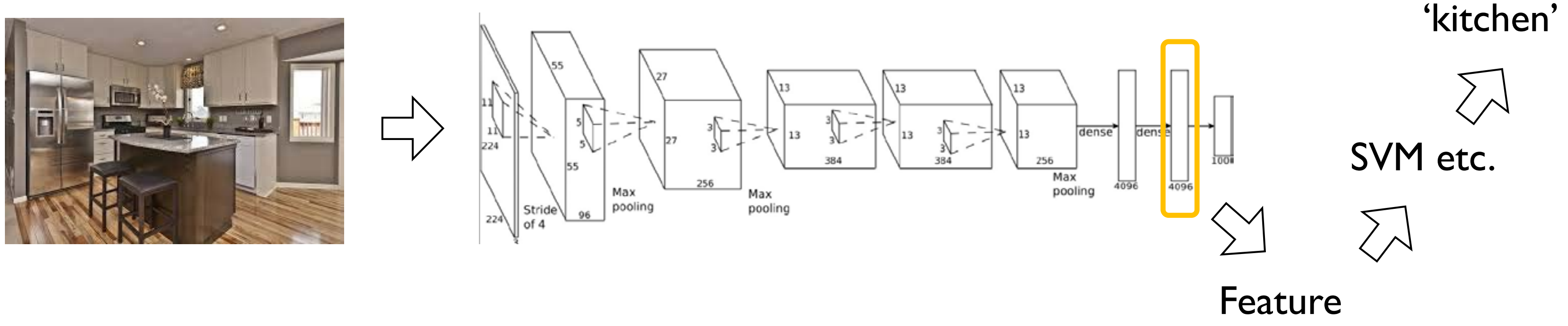
- Formal definition: *Applying knowledge gained in the process of solving one problem to a different but related problem*
- Known to be highly effective for neural networks
- They primary reason of using TL: train NNs on a task with a limited amount of training data
- Example: Object category recognition and scene category recognition

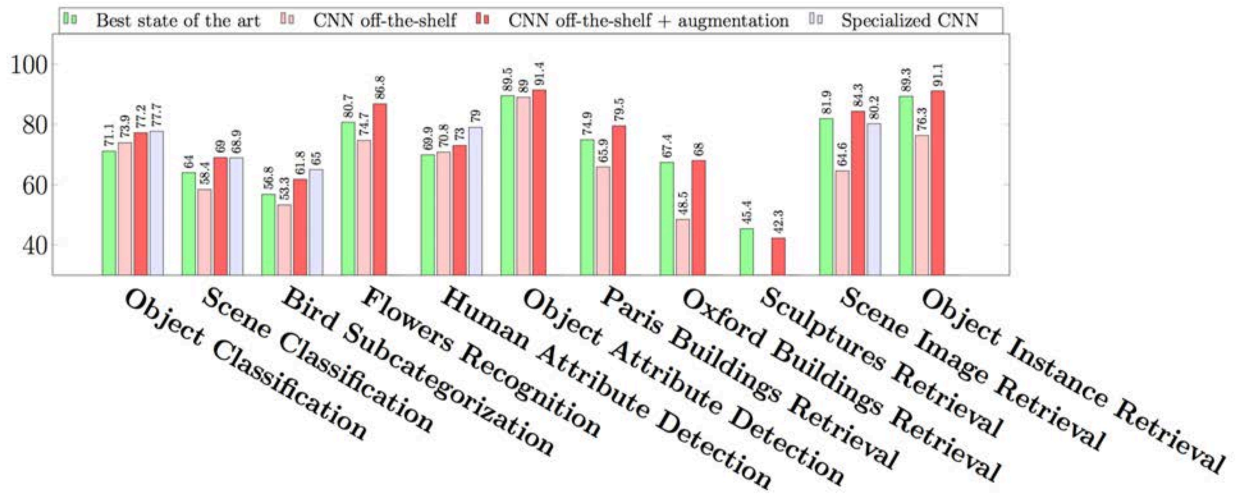


TL: Using layer activation as features

DeCAF [Denahue+13], CNN features off-the-shelf [Razavian+14]

- Using a CNN trained on one task as a `feature extractor'
  - Input an image into the CNN, extract activation (a vector) of one selected layer, and use it for classifying the input

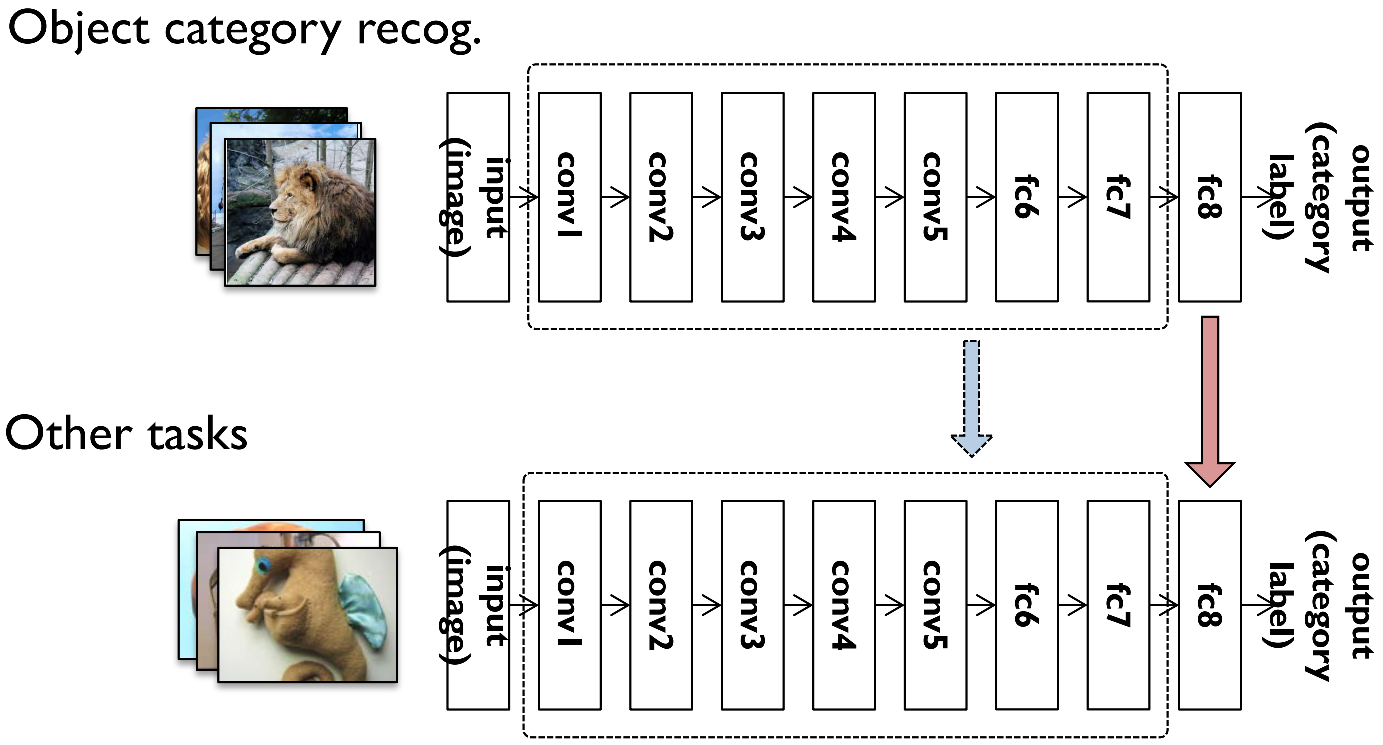




From Razavian+, CNN Features off-the-shelf: an Astounding Baseline for Recognition, 2014

TL: Fine-tuning of a pretrained CNN

- A network (i.e., its weights) trained on one task is utilized for another task by i) retraining the net on the new task
  - At least the last layer needs to be renovated
    - E.g., a different number of classes
  - Equivalent to initializing weights by the old task → Training is stabilized; weights are already close to the optimum for the new task

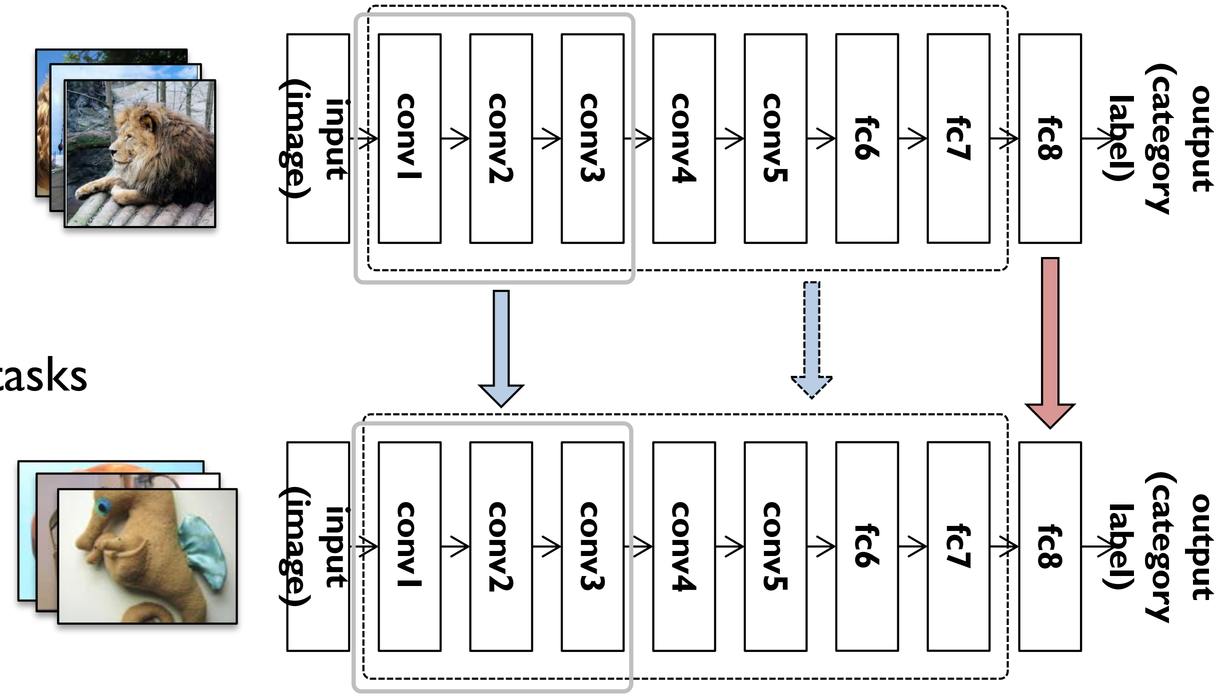


TL: Fine-tuning of a pretrained CNN

- A network (i.e., its weights) trained on one task is utilized for another task by ii) optimizing weights of only selected layers
  - Usually high layers are selected, because lower layers are expected to have learned fundamental features, which may be shared by other tasks
  - By selecting a few layers, the number of free parameters decreases → a small amount of data may be sufficient

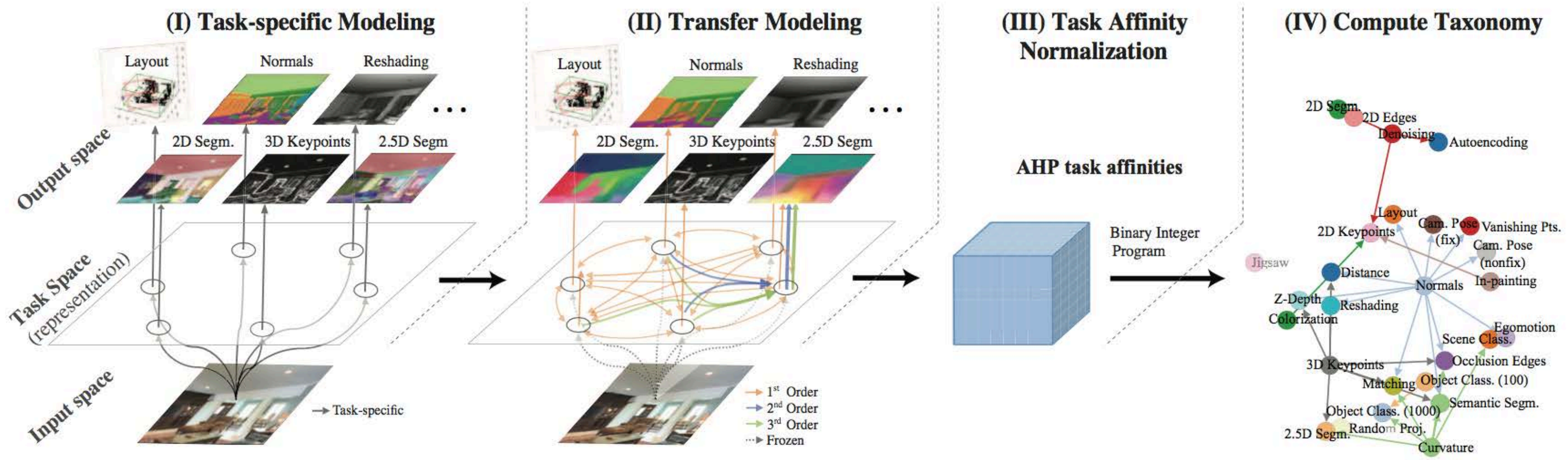
Object category recog.

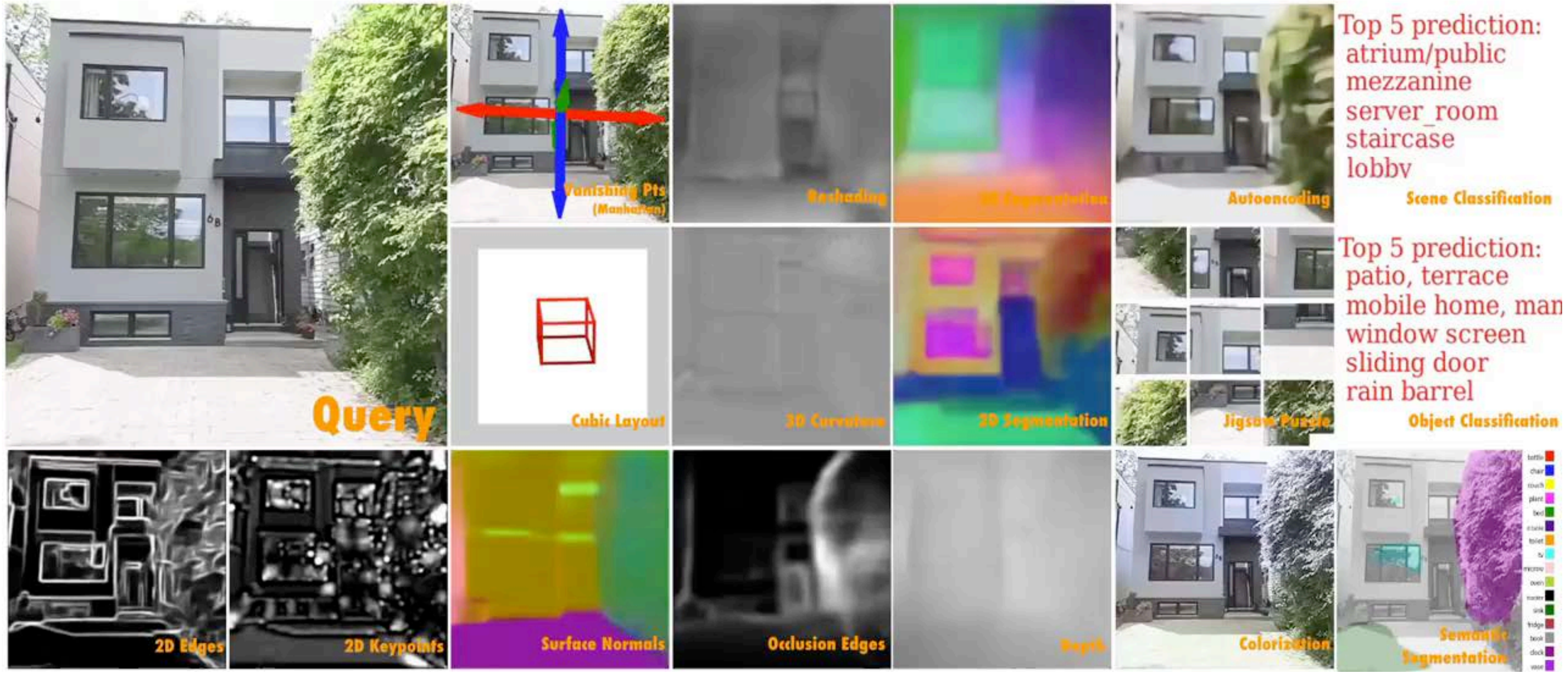
Other tasks



Transferability between tasks

Zamir+, Taskonomy: Disentangling Task Transfer Learning, CVPR2018

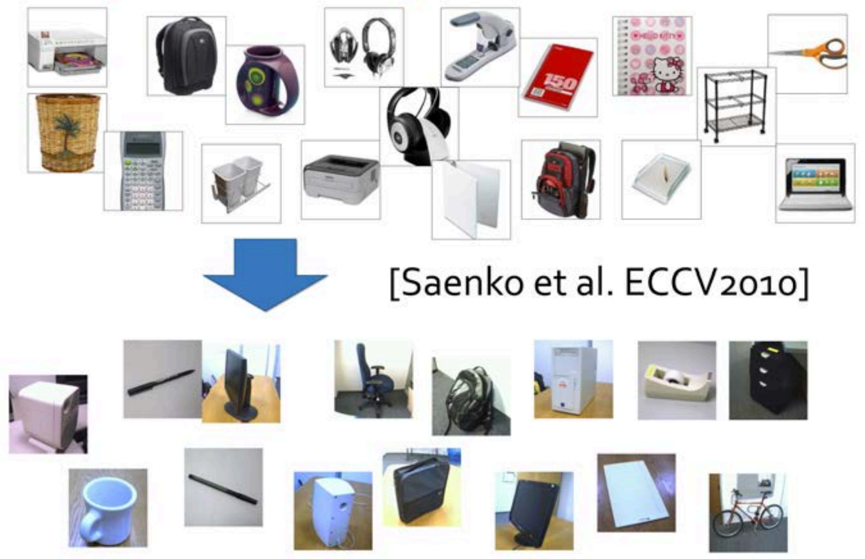




Domain shift

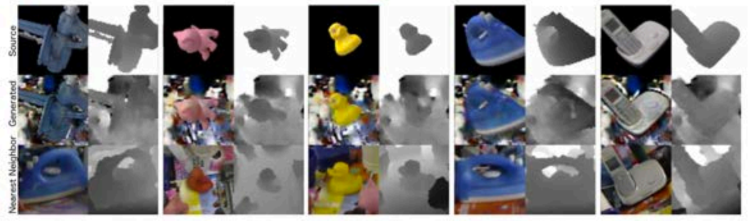
- Source domain and target domain
- Labeled samples are available in the source domain; e.g., synthetic data
- Unlabeled samples in the target domain
- Differences between the two domains matter, even if small

Example: Internet images -> Webcam sensor

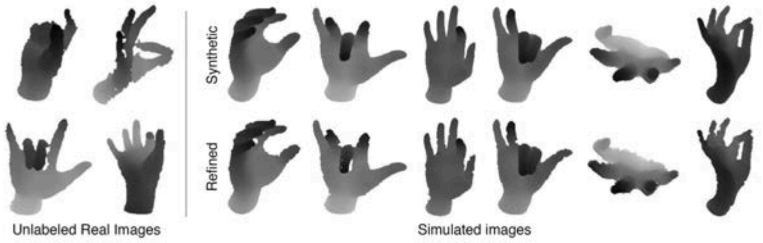


[Saenko et al. ECCV2010]

Adaptation via GAN image translation

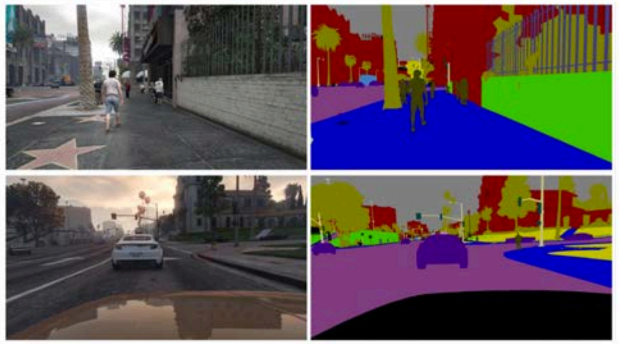


[Bousmalis et al. CVPR 2017]

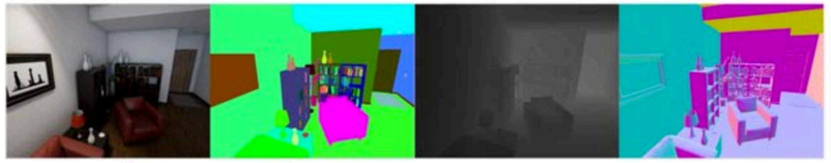


[Shrivastava et al. CVPR 2017]

Computer graphics (yet again) meets computer vision



[Richter et al. ECCV16]

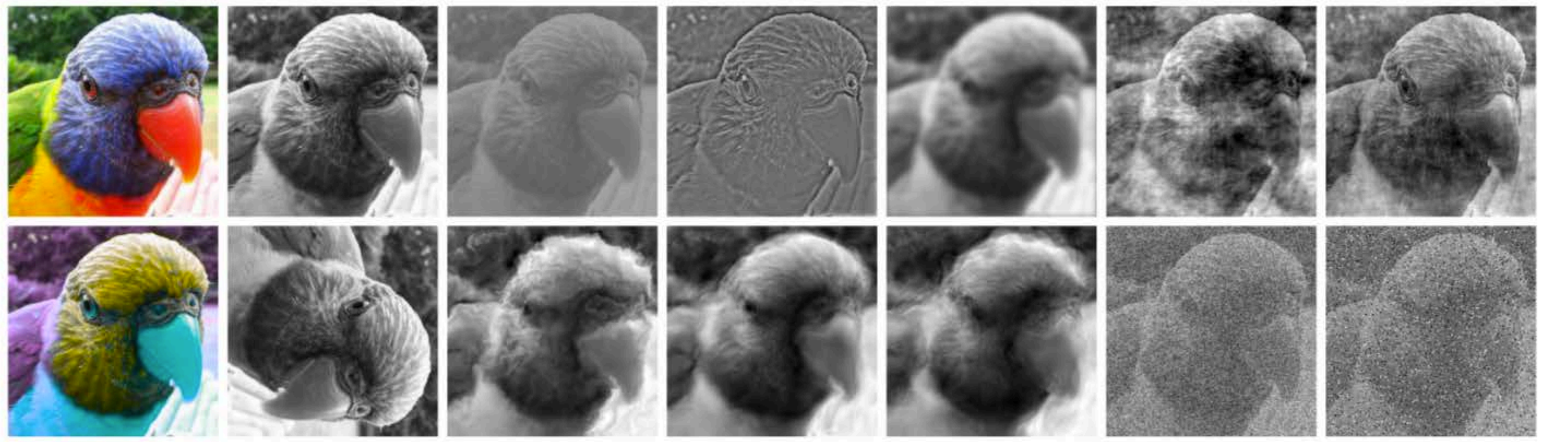


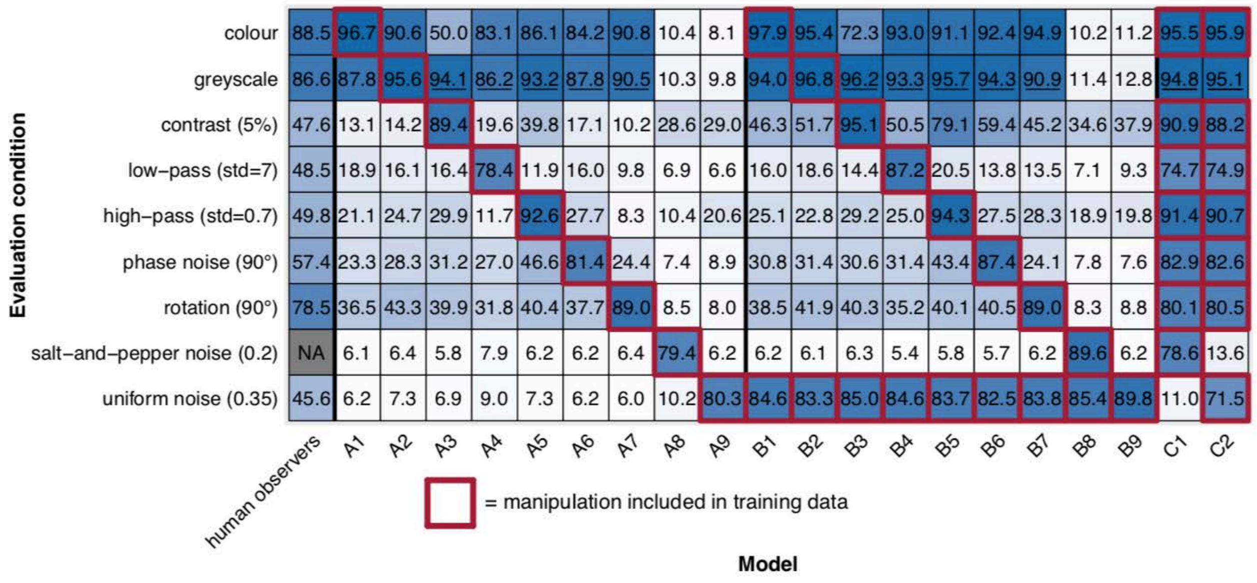
[Qiu et al. ECCVw16]

How large can the effect be?

Geirhos+(U Tubingen), Generalisation in humans and deep neural networks, NIPS2018

- Domains = noises or transformations applied on clean images
- CNNs trained on one domain are not effective on others





Evaluation condition	human observers	A1	A2	A3	A4	A5	A6	A7	A8	A9	B1	B2	B3	B4	B5	B6	B7	B8	B9	C1	C2
colour	88.5	96.7	90.6	50.0	83.1	86.1	84.2	90.8	10.4	8.1	97.9	95.4	72.3	93.0	91.1	92.4	94.9	10.2	11.2	95.5	95.9
greyscale	86.6	87.8	95.6	94.1	86.2	93.2	87.8	90.5	10.3	9.8	94.0	96.8	96.2	93.3	95.7	94.3	90.9	11.4	12.8	94.8	95.1
contrast (5%)	47.6	13.1	14.2	89.4	19.6	39.8	17.1	10.2	28.6	29.0	46.3	51.7	95.1	50.5	79.1	59.4	45.2	34.6	37.9	90.9	88.2
low–pass (std=7)	48.5	18.9	16.1	16.4	78.4	11.9	16.0	9.8	6.9	6.6	16.0	18.6	14.4	87.2	20.5	13.8	13.5	7.1	9.3	74.7	74.9
high–pass (std=0.7)	49.8	21.1	24.7	29.9	11.7	92.6	27.7	8.3	10.4	20.6	25.1	22.8	29.2	25.0	94.3	27.5	28.3	18.9	19.8	91.4	90.7
phase noise (90°)	57.4	23.3	28.3	31.2	27.0	46.6	81.4	24.4	7.4	8.9	30.8	31.4	30.6	31.4	43.4	87.4	24.1	7.8	7.6	82.9	82.6
rotation (90°)	78.5	36.5	43.3	39.9	31.8	40.4	37.7	89.0	8.5	8.0	38.5	41.9	40.3	35.2	40.1	40.5	89.0	8.3	8.8	80.1	80.5
salt–and–pepper noise (0.2)	NA	6.1	6.4	5.8	7.9	6.2	6.2	6.4	79.4	6.2	6.2	6.1	6.3	5.4	5.8	5.7	6.2	89.6	6.2	78.6	13.6
uniform noise (0.35)	45.6	6.2	7.3	6.9	9.0	7.3	6.2	6.0	10.2	80.3	84.6	83.3	85.0	84.6	83.7	82.5	83.8	85.4	89.8	11.0	71.5

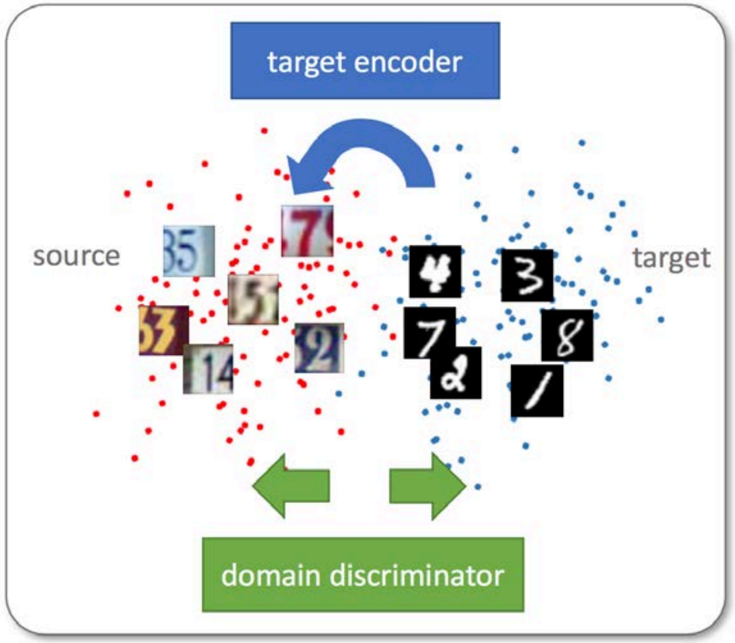
Model

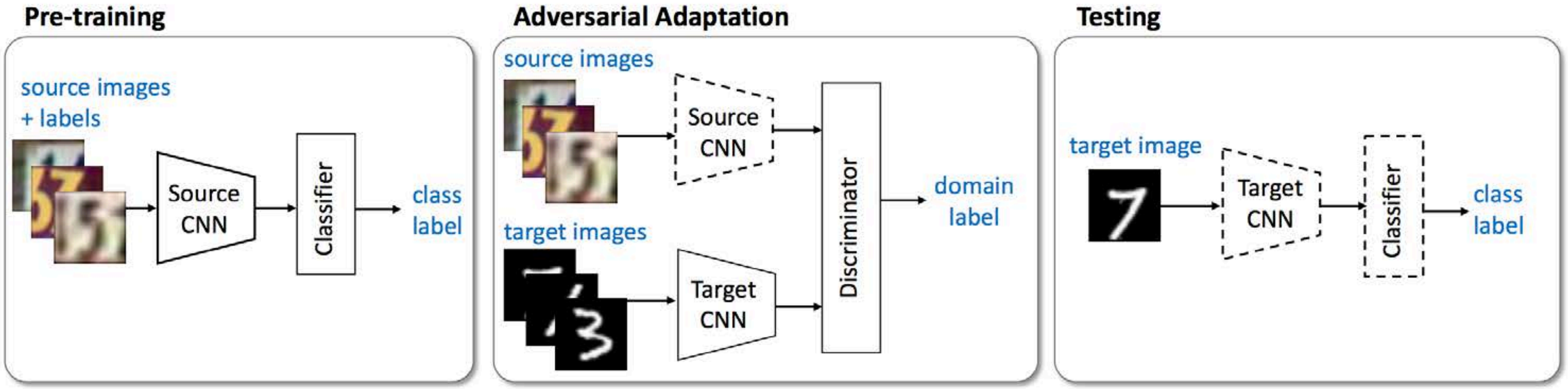
= manipulation included in training data

Domain adaptation: adversarial training

Tzeng+, Adversarial Discriminative Domain Adaptation, CVPR2017

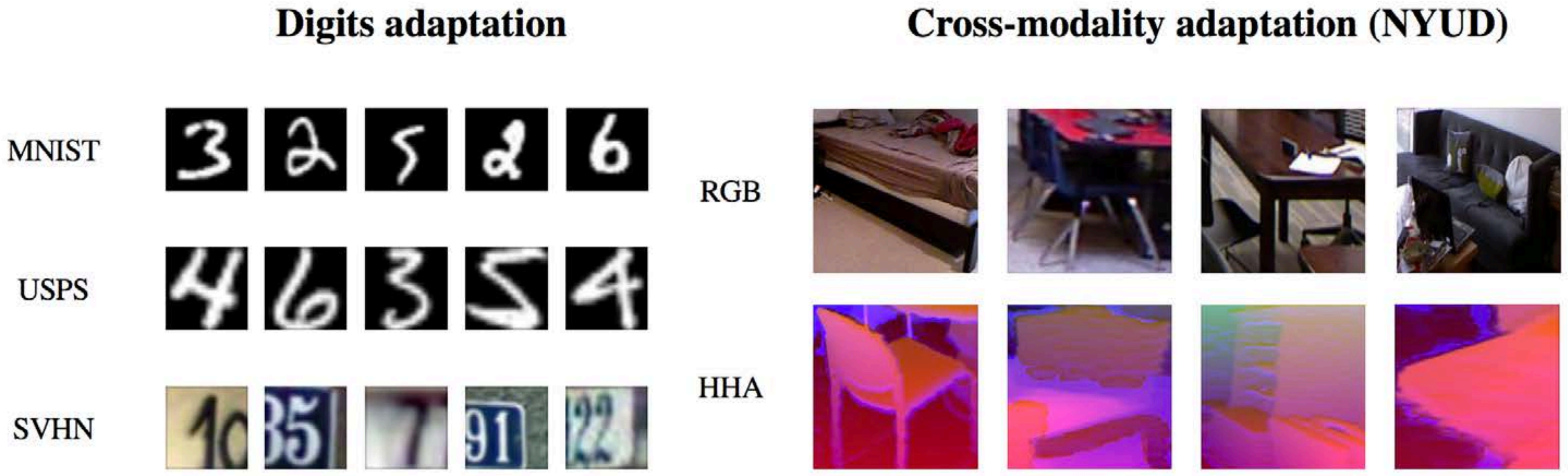
- Try to extract features from samples of target domain that are indistinguishable from those of source samples
  - Discriminator: trained so as to distinguish the domain from the feature
  - Target CNN: trained so as to yield indistinguishable feature from inputs
- Use the target CNN + the classifier to classify samples in the target domain

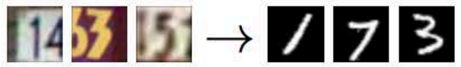




Domain adaptation: adversarial training

Tzeng+, Adversarial Discriminative Domain Adaptation, CVPR2017

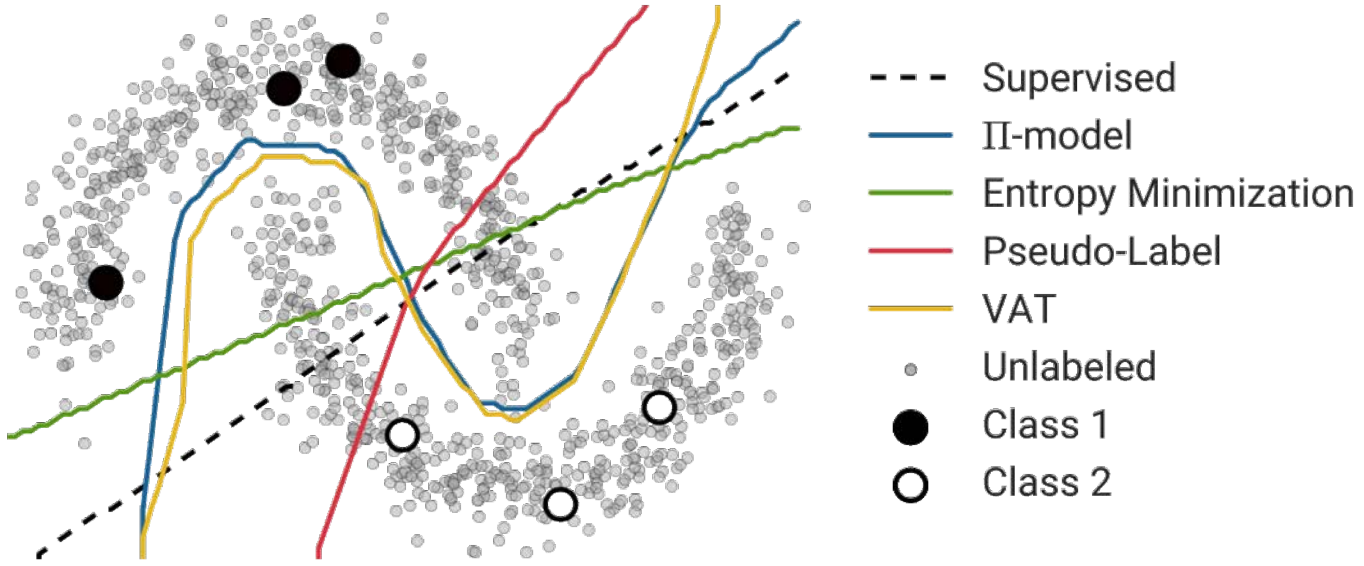


Method	MNIST → USPS	USPS → MNIST	SVHN → MNIST 
Source only	0.752 ± 0.016	0.571 ± 0.017	0.601 ± 0.011
Gradient reversal	0.771 ± 0.018	0.730 ± 0.020	0.739 [16]
Domain confusion	0.791 ± 0.005	0.665 ± 0.033	0.681 ± 0.003
CoGAN	0.912 ± 0.008	0.891 ± 0.008	did not converge
ADDA (Ours)	0.894 ± 0.002	0.901 ± 0.008	0.760 ± 0.018

Semi-supervised learning

Oliver+(Google Brain), Realistic Evaluation of Semi-Supervised Learning Algorithms, arXiv2018

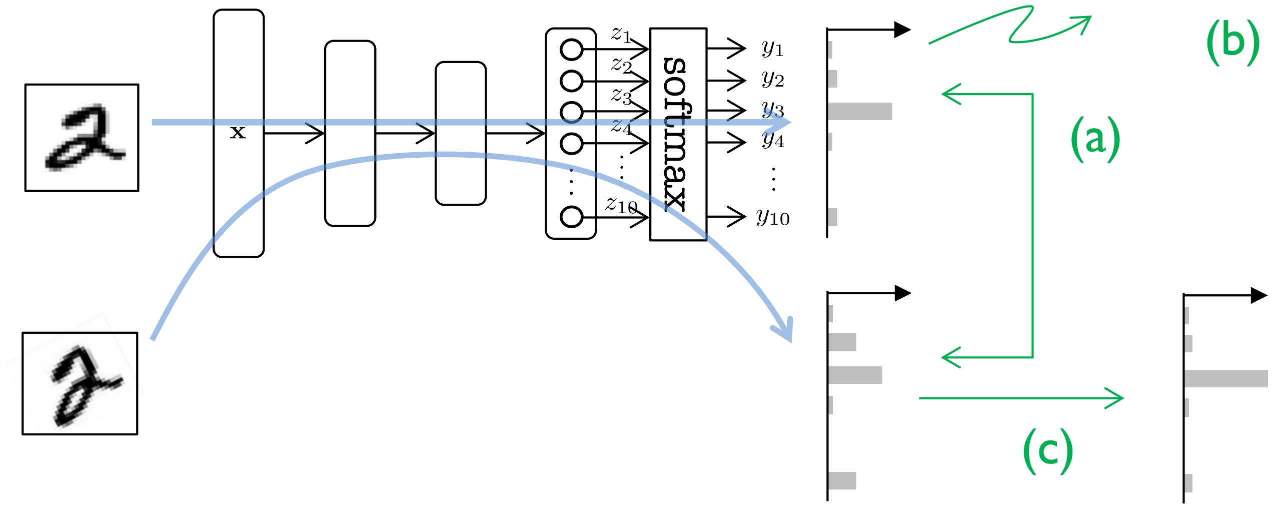
- Problem setting: We have a small amount of labeled samples and a large amount of unlabeled samples
  - Background: Annotating (giving labels to) samples is expensive
- How can we utilize the unlabeled samples?



Semi-supervised learning

Oliver+(Google Brain), Realistic Evaluation of Semi-Supervised Learning Algorithms, arXiv2018

- Consistency regularization … (a)
  - Apply transformations used in data augmentation to unlabeled samples and require the predictions to be consistent before/after the trans.
- Pseudo labeling … (b)
  - Regard the prediction for unlabeled samples with high confidence to be true prediction
- Entropy minimization … (c)
  - Require the distribution of class scores to be sharper

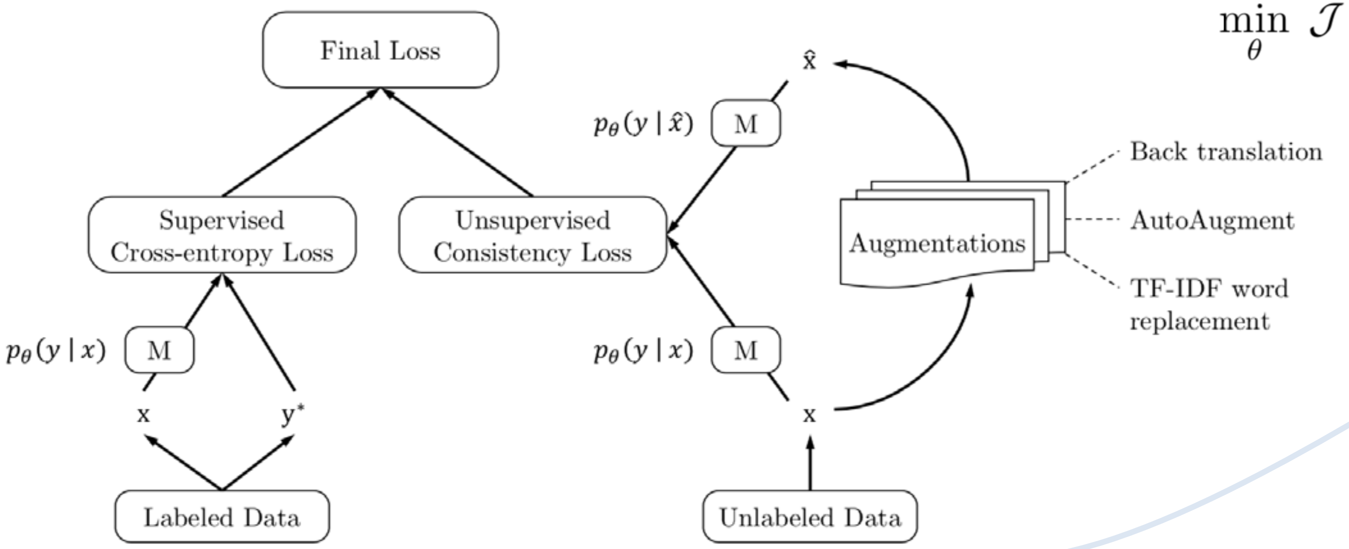


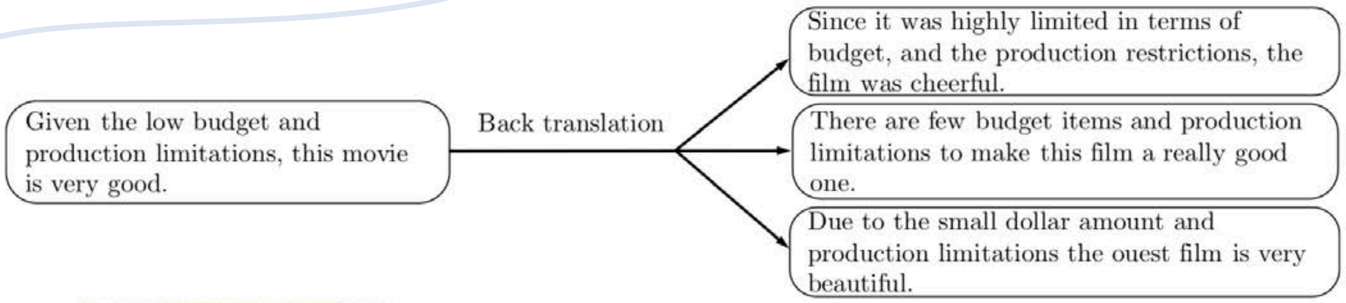
Unsupervised data augmentation (UDA)

Xie+(Google Brain), Unsupervised Data Augmentation, arXiv19.04

- Consistency regularization with all possible data augmentation methods

$$\min_{\theta} \mathcal{J} = \mathop{\mathbb{E}}_{x,y^* \in L} [p_\theta(y^* \mid x)] + \lambda \mathcal{J}_{\text{UDA}}(\theta)$$



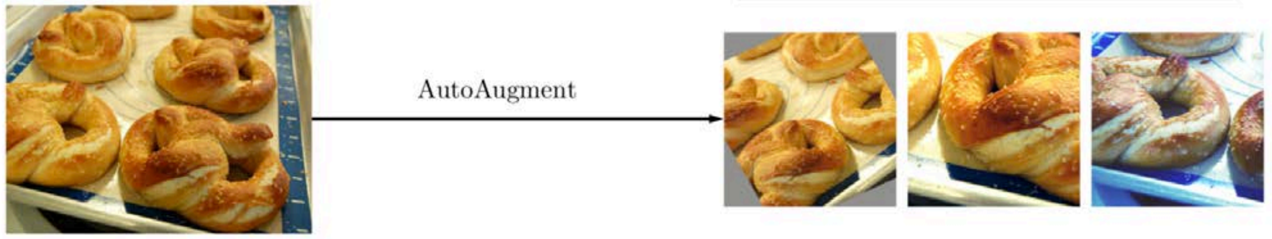


Given the low budget and production limitations, this movie is very good.

Since it was highly limited in terms of budget, and the production restrictions, the film was cheerful.

There are few budget items and production limitations to make this film a really good one.

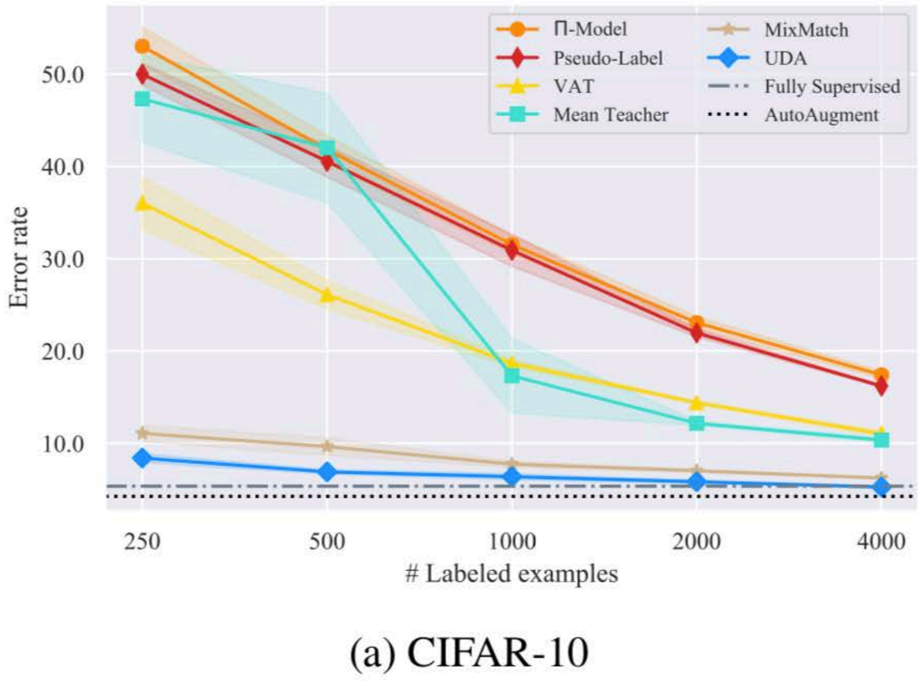
Due to the small dollar amount and production limitations the ouest film is very beautiful.



Unsupervised data augmentation (UDA)

Xie+(Google Brain), Unsupervised Data Augmentation, arXiv19.04

- Currently the best semi-supervised learning method
  - It surpasses the full-supervised training on SVHN



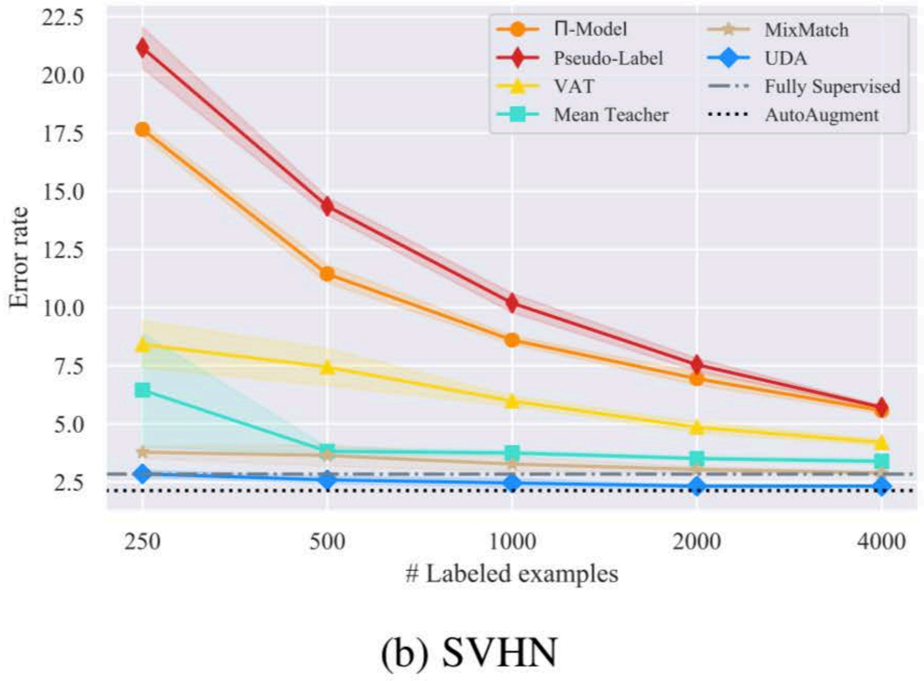


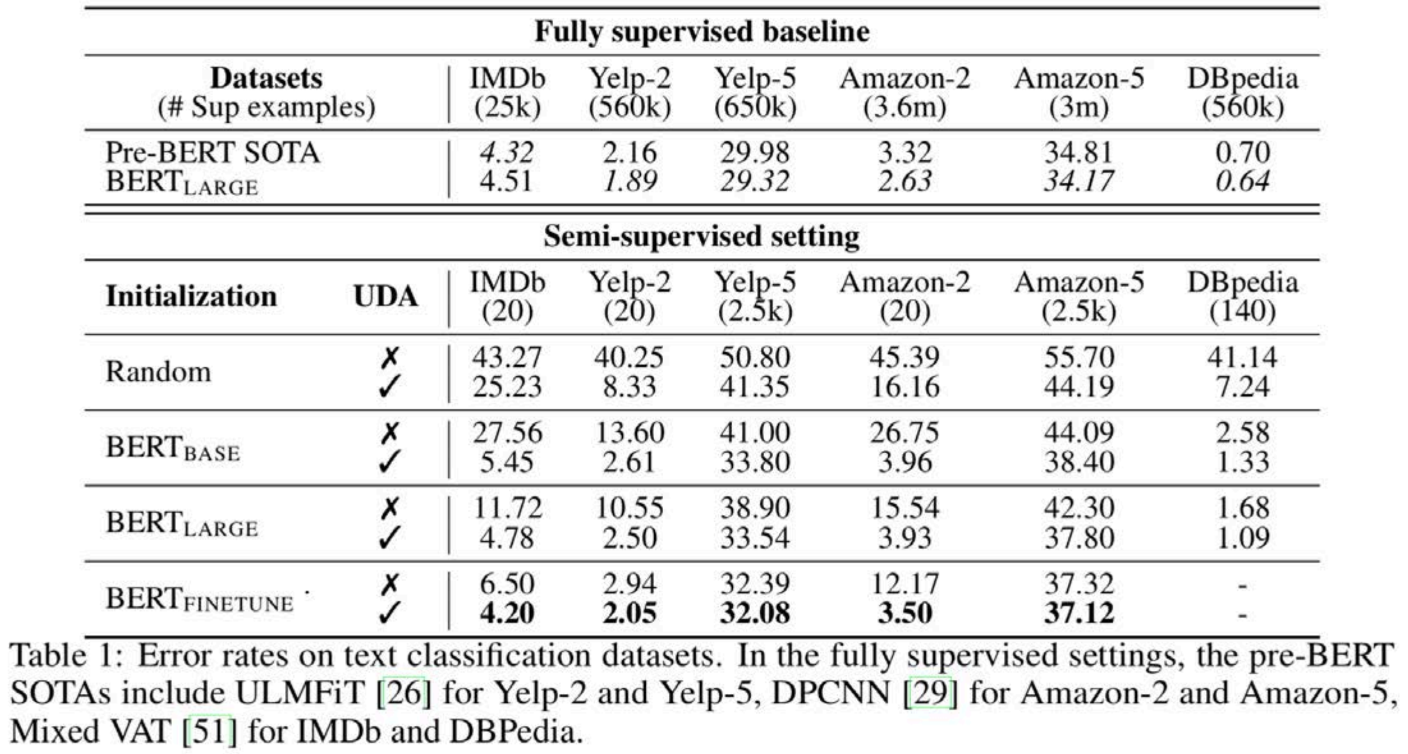
Figure 5: Comparison with semi-supervised learning methods on CIFAR-10 and SVHN with varied number of labeled examples. The performances of Π-Model, Pseudo-Label, VAT and Mean Teacher are reported in [3].

Unsupervised data augmentation (UDA)

Xie+(Google Brain), Unsupervised Data Augmentation, arXiv19.04

- It works well also for NLP tasks
  - Combined with a BERT pretrained model, it achieves SOTA-level performance with only 20 samples

E.g. Yelp "*They have the best happy hours, the food is good, and service is even better. When it is winter we become regulars.*" → 4 star

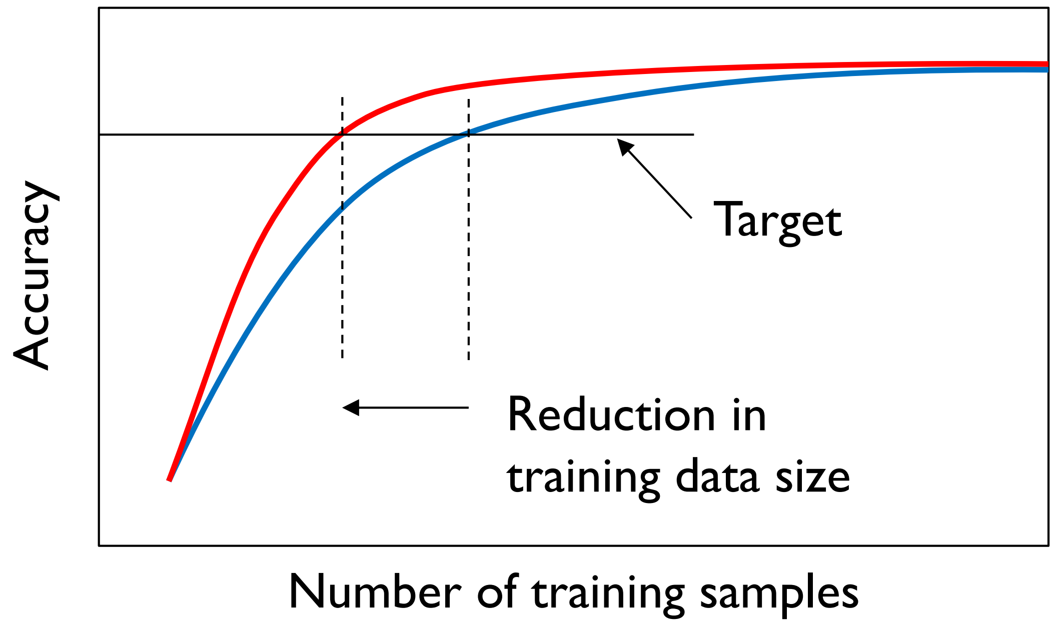


Fully supervised baseline							
Datasets (# Sup examples)		IMDb (25k)	Yelp-2 (560k)	Yelp-5 (650k)	Amazon-2 (3.6m)	Amazon-5 (3m)	DBpedia (560k)
Pre-BERT SOTA		*4.32*	2.16	29.98	3.32	34.81	0.70
$\text{BERT}_{\text{LARGE}}$		4.51	*1.89*	*29.32*	*2.63*	*34.17*	*0.64*
Semi-supervised setting							
Initialization	**UDA**	IMDb (20)	Yelp-2 (20)	Yelp-5 (2.5k)	Amazon-2 (20)	Amazon-5 (2.5k)	DBpedia (140)
Random	✗	43.27	40.25	50.80	45.39	55.70	41.14
	✓	25.23	8.33	41.35	16.16	44.19	7.24
$\text{BERT}_{\text{BASE}}$	✗	27.56	13.60	41.00	26.75	44.09	2.58
	✓	5.45	2.61	33.80	3.96	38.40	1.33
$\text{BERT}_{\text{LARGE}}$	✗	11.72	10.55	38.90	15.54	42.30	1.68
	✓	4.78	2.50	33.54	3.93	37.80	1.09
$\text{BERT}_{\text{FINETUNE}}$	✗	6.50	2.94	32.39	12.17	37.32	-
	✓	**4.20**	**2.05**	**32.08**	**3.50**	**37.12**	-

Table 1: Error rates on text classification datasets. In the fully supervised settings, the pre-BERT SOTAs include ULMFiT [26] for Yelp-2 and Yelp-5, DPCNN [29] for Amazon-2 and Amazon-5, Mixed VAT [51] for IMDb and DBPedia.

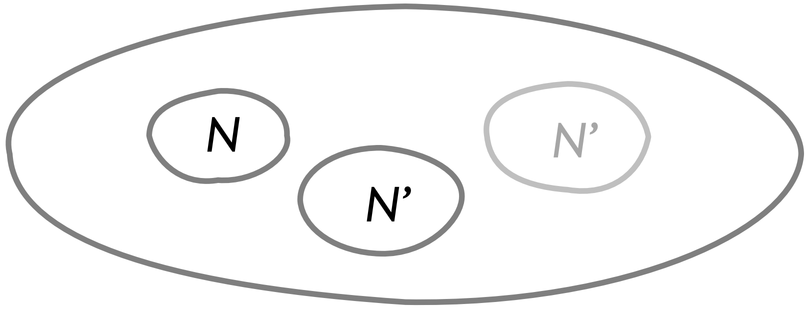
Active learning: Motivation and applications

- Goal: enable to select a fixed number of samples such that an NN achieves the maximum performance when trained on them
  - Maximum performance from fewest samples
  - Useful especially when we have a lot of unlabeled data but annotation cost is high
  - Or equivalently, we with to achieve a target accuracy with minimum annotation cost
- Which sample to choose: the priority computed by an *acquisition function*



Active learning: Outline

- The standard procedure is as follows:
  1. Train your net with initial *N* labeled samples
  2. Select *N'* samples (unlabeled) using the current net and give them labels
  3. Train your net using all the samples you currently have (from scratch)
  4. Go to 2



- Approaches [Settles, Active learning literature survey, 2010]
  - Based on *information theory*: Select samples providing the maximum information gain
  - Based on *uncertainty*: Select samples leading to the most uncertain prediction

Math of uncertainty

Prediction (e.g. class ID)

An input

Training data

$$\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D}) = \int \underbrace{\mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta})}_{Data} \underbrace{\mathrm{p}(\boldsymbol{\theta}|\mathcal{D})}_{Model} \, d\boldsymbol{\theta}$$

- Classification: p(y=c|x) is directly predicted
- Regression: y is predicted, not p(y|x)

Uncertainty of prediction given an input (Uncertainty due to input)

Uncertainty of model parameter dependent on training data (Uncertainty of model)

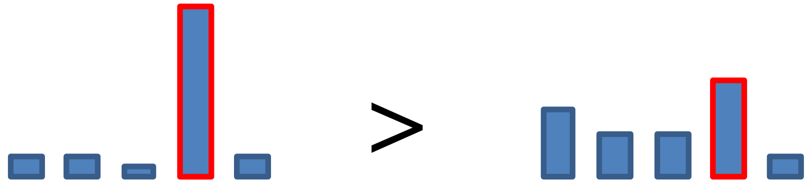
Approximated by an ensemble of models (trained from different initial values)

$$\mathrm{P}(\omega_c|\boldsymbol{x}^*, \mathcal{D}) \approx \frac{1}{M} \sum_{i=1}^{M} \mathrm{P}(\omega_c|\boldsymbol{x}^*, \boldsymbol{\theta}^{(i)}), \; \boldsymbol{\theta}^{(i)} \sim \mathrm{q}(\boldsymbol{\theta})$$

Metrics of uncertainty

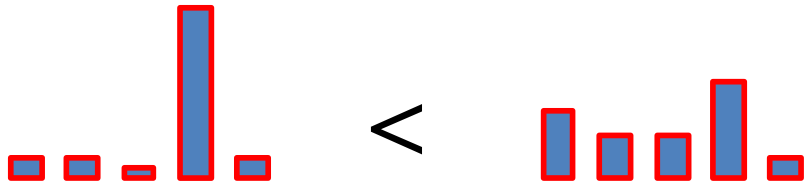
Single model

- Max-softmax
  - aka. *confidence*
  - Lower = more uncertain



- Entropy of softmax
  - Higher = more uncertain

$$H[p(\text{class} \mid \mathbf{x})] = -\sum_{k=1}^{K} p_k \log p_k$$



Ensemble models

- Max of *averaged* softmax
- Entropy of *averaged* softmax
- Variance of softmax
  - (in)consistency of model's prediction
- Variation Ratio (varR)
  - Lower = more uncertain

$$v := 1 - \frac{f_m}{T}$$

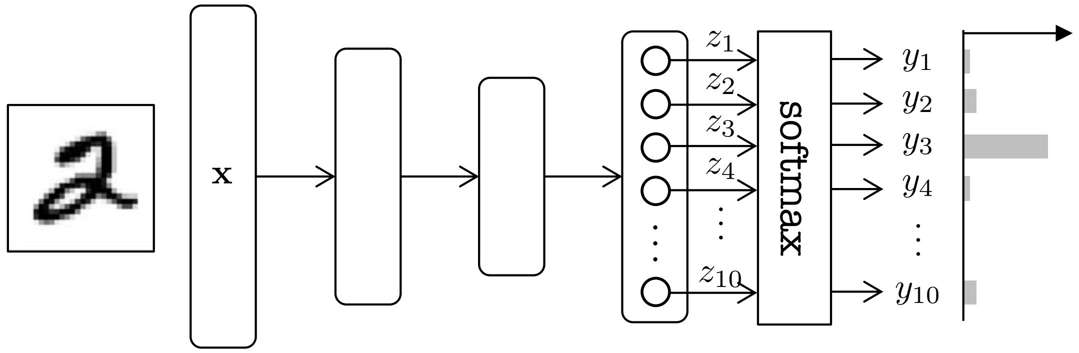
Num of models predicting differently from the majority

Num of ensembles

Predicted class			Predicted class	
3			3	
3			5	
2	1-1/6	>	2	1-3/6
3			3	
3			2	
3			3	

Softmax with temperature scaling

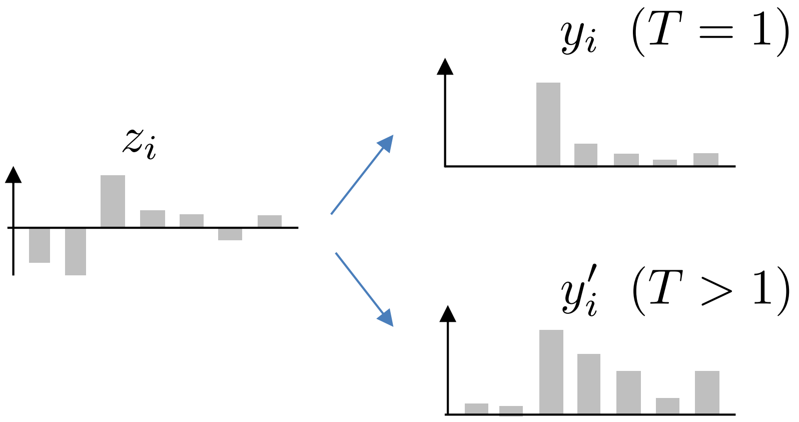
- Maximum of softmax outputs = *confidence*



Prediction = the class yielding the maximum softmax

- Softmax w/ temperature scaling
  - Larger *T* makes the distribution more flat
  - The default is *T* = 1

$$y_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

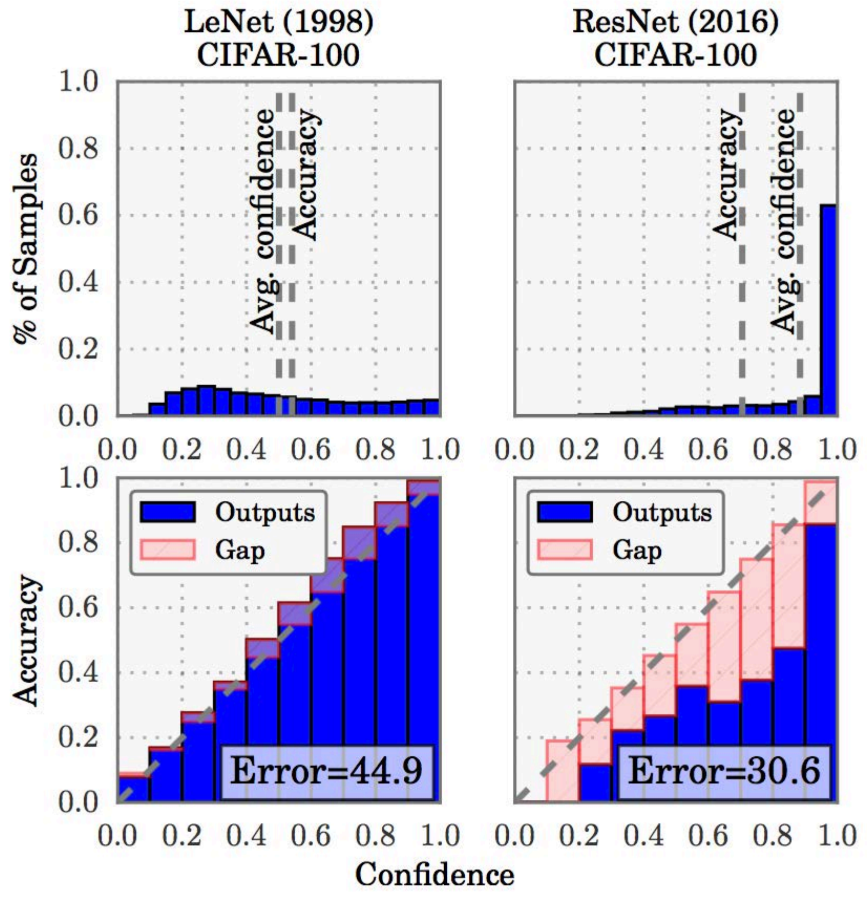


Calibration of softmax

Guo+, On Calibration of Modern Neural Networks, ICML2017

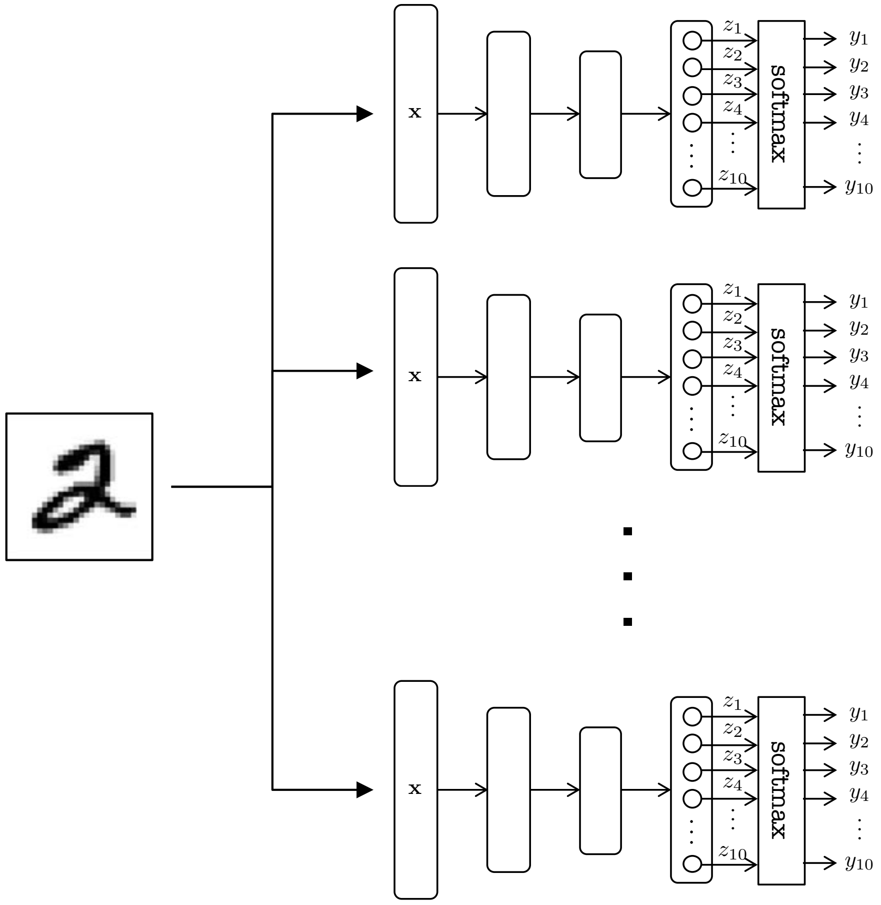
- Criticism for confidence: modern NNs tend to be overconfident
- Calibration: Adjust temperature so that confidence will be close to classification accuracy

E.g. Samples with confidence=0.5 should be classified accurately with probability = 0.5

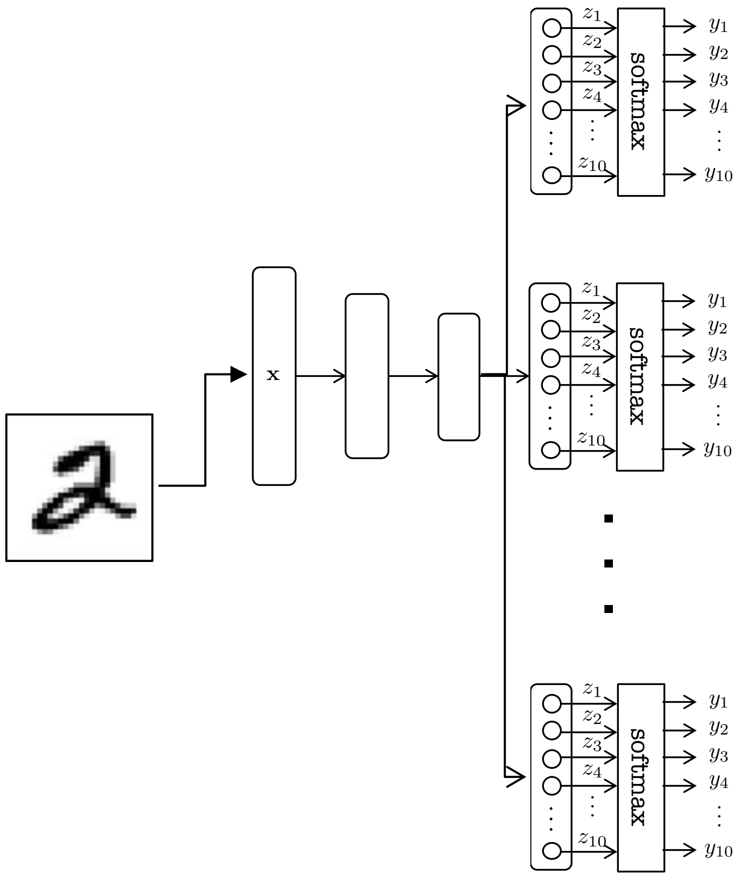


Model ensemble ≒ MC-dropout

- Ensemble of different models
  - The same network trained with different initial values



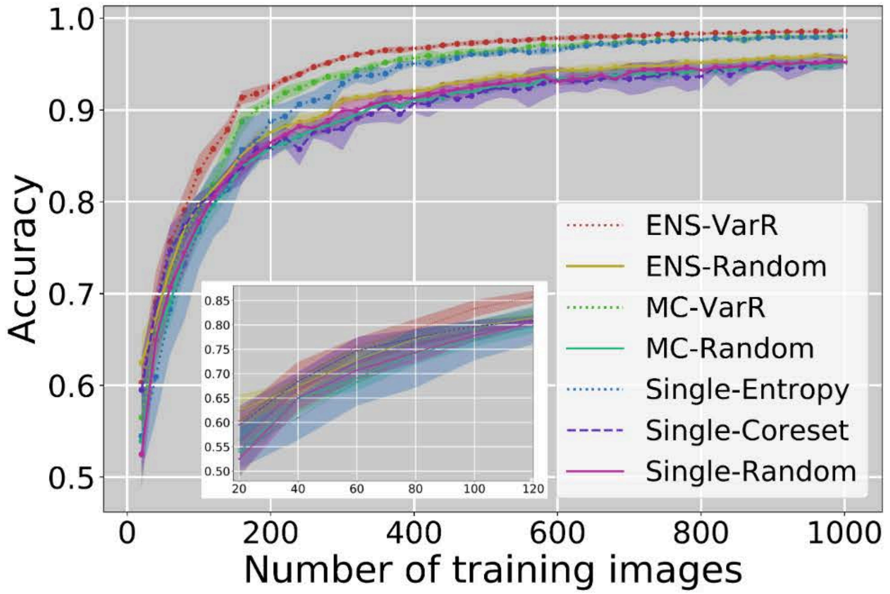
- MC-dropout
  - Dropout is used at test time



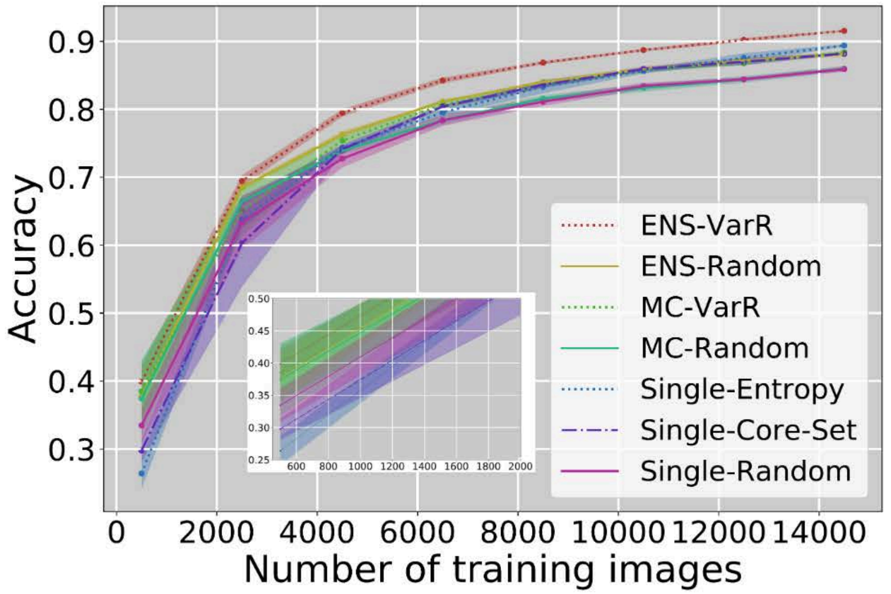
Active learning with an ensemble of networks

Beluch+, The power of ensembles for active learning in image classification, CVPR2018

- An ensemble of networks trained with different initial values
  - Acquisition function: varR (variarion ratio) $v := 1 - \frac{f_m}{T}$



(a) MNIST on S-CNN



(b) CIFAR-10 on DenseNet

Figure 1: Test accuracy over acquired images. We compare Variation Ratio for MC dropout and the ensemble (ENS) and softmax-entropy based acquisition for a single network. For all methods we also show performance under random acquisition. Shaded areas denote $\pm$ one standard deviation. (see text for details about the architectures used).

Outline

- Data augmentation
  - w/ Assignment 2
- Transfer learning
- Domain adaptation
- Semi-supervised learning
- Active learning / Uncertainty measure