The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Spring 2023

ADVANCED TOPICS IN COMPUTER VISION

Atlas Wang
Assistant Professor, The University of Texas at Austin

Visual Informatics Group@UT Austin
https://vita-group.github.io/

Supervised Learning

Feature Space $\mathcal{X}$ **Label** Space $\mathcal{Y}$

Goal: Construct a **predictor** $f : \mathcal{X} \rightarrow \mathcal{Y}$ to minimize

$$R(f) \equiv \mathbb{E}_{XY}\left[\text{loss}(Y, f(X))\right]$$

Optimal predictor (Bayes Rule) depends on unknown P_{XY}, so instead *learn* a good prediction rule from training data $\{(X_i, Y_i)\}_{i=1}^{n} \overset{\text{iid}}{\sim} P_{XY}$ (unknown)

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)}\left[\log p(y|x)\right]$$

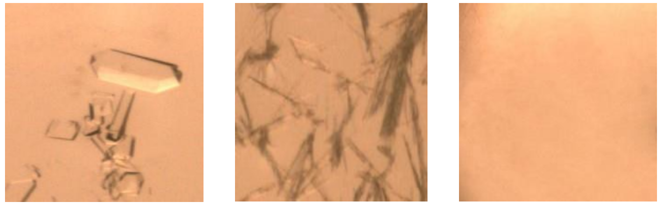
- ML has been largely focused on this …
- **But Lots** of other problem settings are coming up:
  - What if we ***also*** *have* unlabeled data?
  - What if we ***only*** *have* unlabeled data?
  - What if we *have poor-quality labels (e.g., coarse or potentially mistaken?)*
  - What if we have many datasets, but one somehow differing from another?
  - What if we only have one example, or a few per (new) class?
  - ……

And wait, there are more!

- Transfer Learning
- Semi-supervised learning
- One/Few-shot learning
- Un/Self-Supervised Learning
- Domain adaptation
- Meta-Learning
- Zero-shot learning
- Continual / Lifelong-learning
- Multi-modal learning
- Multi-task learning
- Active learning
- …

Setting	Source	Target	Shift Type
Semi-supervised	Single labeled	Single unlabeled	None
Domain Adaptation	Single labeled	Single unlabeled	Non-semantic
Domain Generalization	Multiple labeled	Unknown	Non-semantic
Cross-Task Transfer	Single labeled	Single unlabeled	Semantic
Few-Shot Learning	Single labeled	Single few-labeled	Semantic
Un/Self-Supervised	Single unlabeled	Many labeled	Both/Task

Particularly Meaningful for CV …



“Crystal” “Needle” “Empty”

Human expert/
Special equipment/
Experiment

“0” “1” “2” …

“Sports”
“News”
“Science”
…

Unlabeled data, X_i

Labeled data, Y_i

Cheap and abundant !

Expensive and scarce !

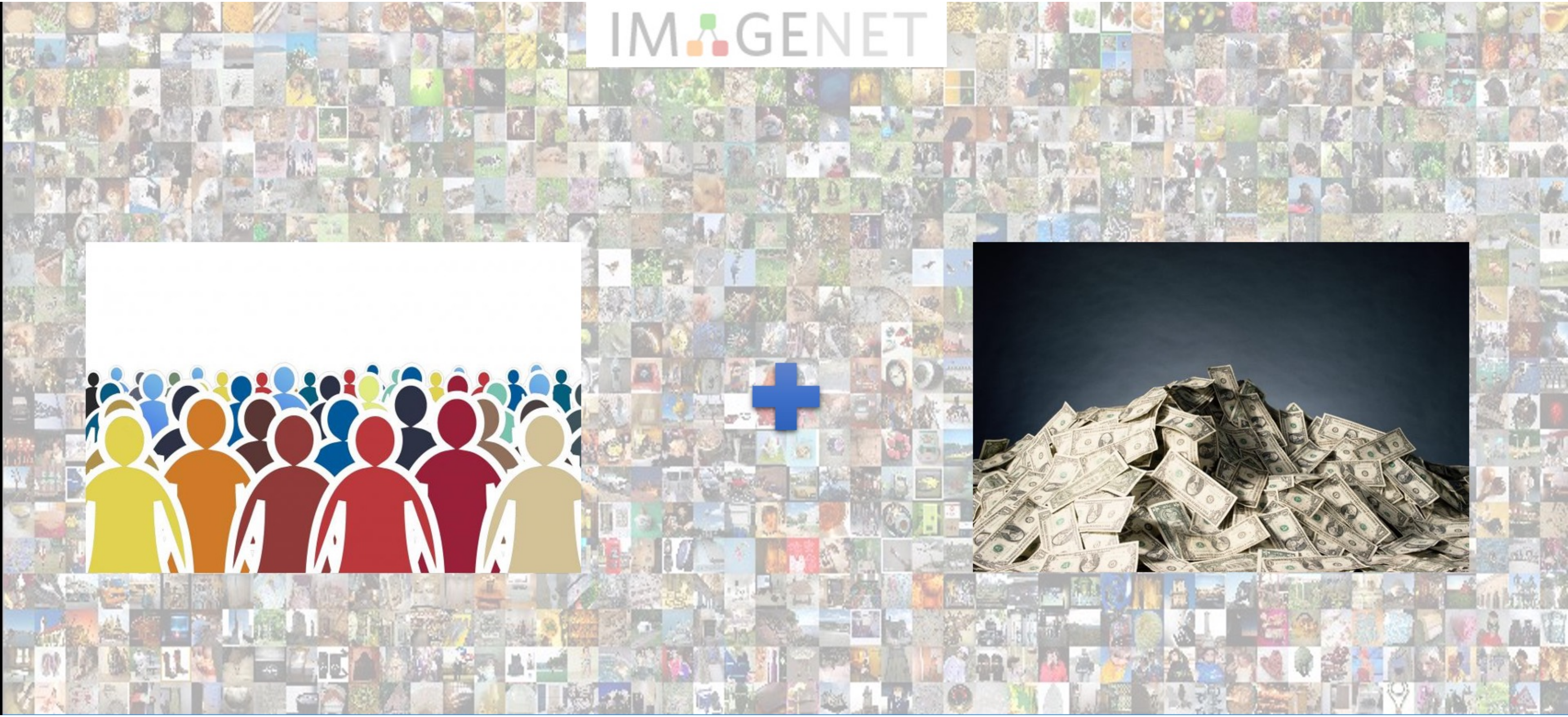
Particularly Meaningful for CV …

image-level labels | points | bounding boxes | scribbles | pixel-level labels

1s/class | 2.4s/instance | 10s/instance | 17s/instance | … | 78s/instance

Annotation time

Particularly Meaningful for CV …



A Whole Big Field! We try to cover a few …

- Semi-Supervised Learning
- Few-Shot Learning
- Active Learning
- Transfer and Multi-Task Learning
- Self-Supervised Learning

What is Semi-Supervised Learning?

Supervised Learning

$$(x, y) \sim p(x, y)$$

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)} [\log p(y|x)]$$

Semi-Su

D_U

- Training data: both labeled data (image, label) and unlabeled data (image)
- **Goal: Use unlabeled data to**
  ng
  led
  er

Cognitive science

Computational model of how humans learn from labeled and unlabeled data.

- concept learning in children: $\mathbf{x}$=animal, y=concept (e.g., dog)
- Daddy points to a brown animal and says "dog!"
- Children also observe animals by themselves

An Incomplete List of Methods ….

- **Confidence & Entropy – “no matter what, be confident”**
  - Pseudo labeling
  - Entropy minimization
  - Virtual Adversarial Training
- **Label Consistency – “label is robust to perturbations”**
  - Pseudo labeling, yet applying different sample augmentations
  - Temporal Ensembling, Mean Teacher …
- **Regularization**
  - Weight decay, Dropout …
  - Strong/unsupervised data augmentation: MixUp, CutOut, MixMatch …
- Co-Training / Self-Training / Pseudo Labeling / Noisy Student

Pseudo Labeling

Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks

Dong-Hyun Lee SAYIT78@GMAIL.COM
Nangman Computing, 117D Garden five Tools, Munjeong-dong Songpa-gu, Seoul, Korea

Abstract

We propose the simple and efficient method of semi-supervised learning for deep neural networks. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Labels*, just picking up the class which has the maximum predicted probability, are used as if they were true labels. This is in effect equivalent to *Entropy Regularization*. It favors a low-density separation between classes, a commonly assumed prior for semi-supervised learning. With Denoising Auto-Encoder and Dropout, this simple method outperforms conventional methods for semi-supervised learning with very small labeled data on the MNIST handwritten digit dataset.

and unsupervised tasks using same neural network simultaneously. In (Ranzato et al., 2008), the weights of each layer are trained by minimizing the combined loss function of an autoencoder and a classifier. In (Larochelle et al., 2008), *Discriminative Restricted Boltzmann Machines* model the joint distribution of an input vector and the target class. In (Weston et al., 2008), the weights of all layers are trained by minimizing the combined loss function of a global supervised task and a *Semi-Supervised Embedding* as a regularizer.

In this article we propose the simpler way of training neural network in a semi-supervised fashion. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Labels*, just picking up the class which has the maximum predicted probability every weights update, are used as if they were true la-

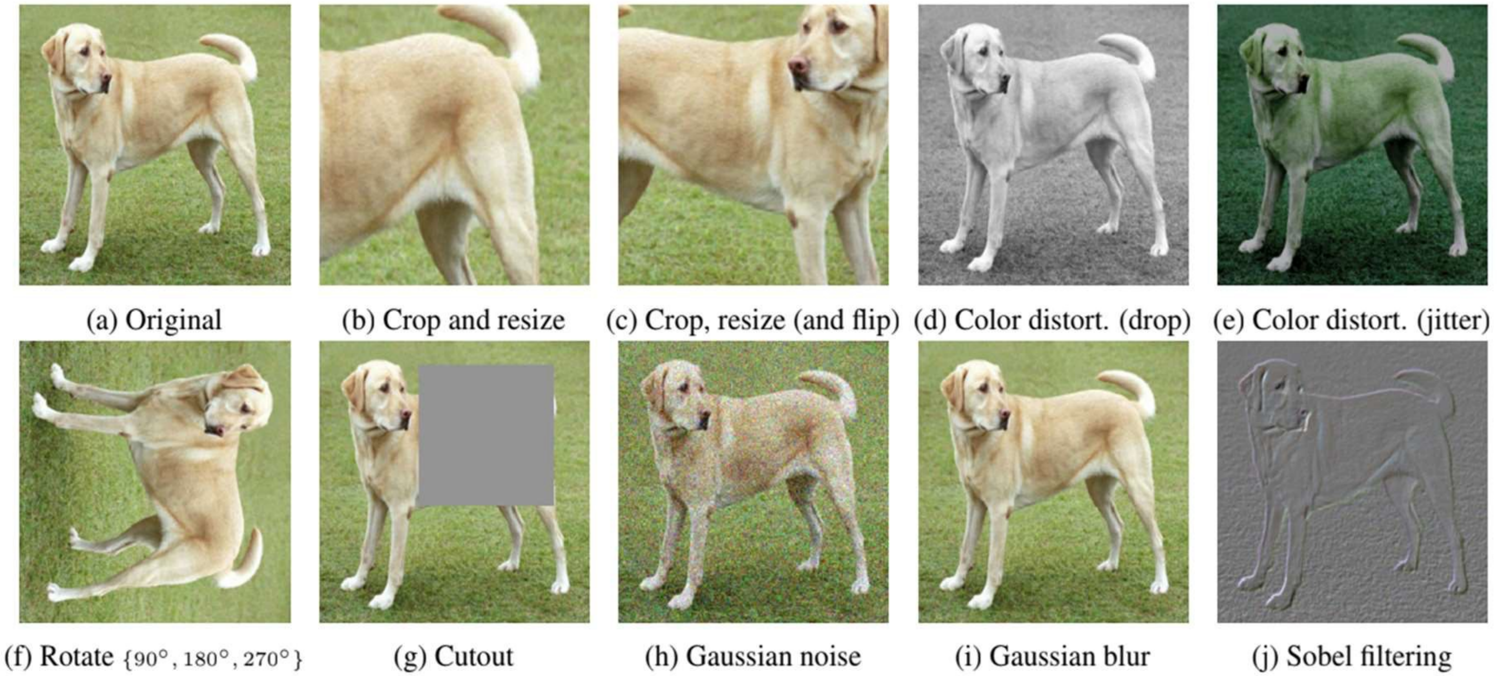
- **Simple idea:**
  - Train on labeled data
  - Make predictions on unlabeled data
  - Pick confident predictions, and add to training data
  - Can do end-to-end (no need to separate stages)

- **Issues:**
  - **"Under-confidence"** or flatness – "sharpening" by entropy minimization

$$\boxed{\operatorname{Sharpen}(p,T)_i := p_i^{\frac{1}{T}} \Bigg/ \sum_{j=1}^{L} p_j^{\frac{1}{T}}}$$

  - "**Overconfidence**"? – Need better uncertainty quantification

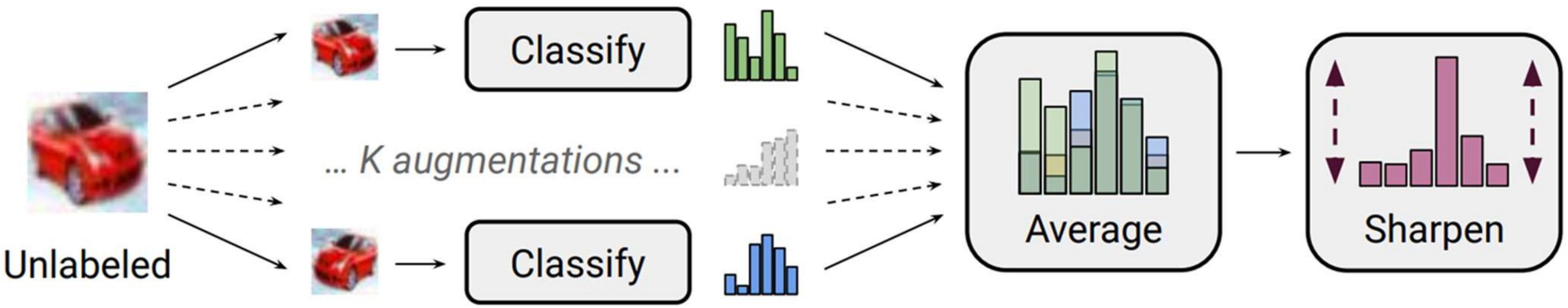
Label Consistency with Data Augmentations



Make sure that the logits are similar

We can either "ensemble" or "compare" them

MixMatch: A Holistic Approach for Semi-Supervised Learning



MixUp

$$
\begin{aligned}
\lambda &\sim \mathrm{Beta}(\alpha, \alpha) \\
\lambda' &= \max(\lambda, 1 - \lambda) \\
x' &= \lambda' x_1 + (1 - \lambda') x_2 \\
p' &= \lambda' p_1 + (1 - \lambda') p_2
\end{aligned}
$$

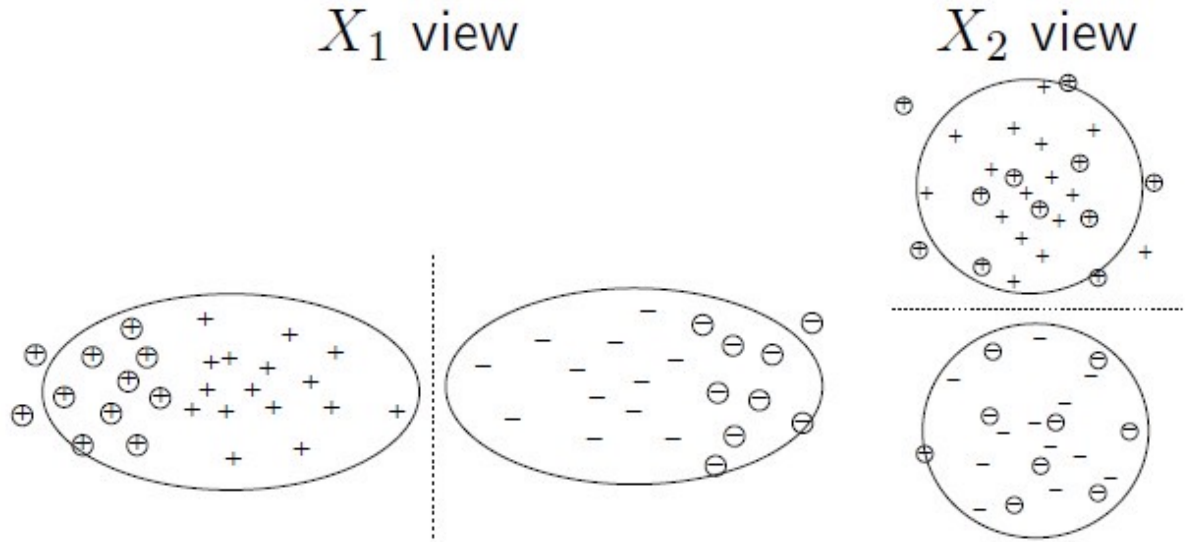
Algorithm 1 MixMatch takes a batch of labeled data $\mathcal{X}$ and a batch of unlabeled data $\mathcal{U}$ and produces a collection $\mathcal{X}'$ (resp. $\mathcal{U}'$) of processed labeled examples (resp. unlabeled with guessed labels).

```
Input: Batch of labeled examples and their one-hot labels X = ((x_b, p_b); b ∈ (1, ..., B)), batch of
   unlabeled examples U = (u_b; b ∈ (1, ..., B)), sharpening temperature T, number of augmentations K,
   Beta distribution parameter α for MixUp.
for b = 1 to B do
   x̂_b = Augment(x_b)   // Apply data augmentation to x_b
   for k = 1 to K do
      û_{b,k} = Augment(u_b)   // Apply k^th round of data augmentation to u_b
   end for
   q̄_b = (1/K) Σ_k p_model(y | û_{b,k}; θ)   // Compute average predictions across all augmentations of u_b
   q_b = Sharpen(q̄_b, T)   // Apply temperature sharpening to the average prediction (see eq. (7))
end for
X̂ = ((x̂_b, p_b); b ∈ (1, ..., B))   // Augmented labeled examples and their labels
Û = ((û_{b,k}, q_b); b ∈ (1, ..., B), k ∈ (1, ..., K))   // Augmented unlabeled examples, guessed labels
W = Shuffle(Concat(X̂, Û))   // Combine and shuffle labeled and unlabeled data
X' = (MixUp(X̂_i, W_i); i ∈ (1, ..., |X̂|))   // Apply MixUp to labeled data and entries from W
U' = (MixUp(Û_i, W_{i+|X̂|}); i ∈ (1, ..., |Û|))   // Apply MixUp to unlabeled data and the rest of W
return X', U'
```

“Co-Training”

Assumptions

- feature split $x = [x^{(1)}; x^{(2)}]$ exists
- $x^{(1)}$ or $x^{(2)}$ alone is sufficient to train a good classifier



“Co-Training”

- (Blum & Mitchell, 1998) (Mitchell, 1999) assumes that
  - features can be split into two sets;
  - each sub-feature set is sufficient to train a good classifier.
- Initially two separate classifiers are trained with the labeled data, on the two sub-feature sets respectively.
- Each classifier then classifies the unlabeled data, and “teaches” the other classifier with the few unlabeled examples (and the predicted labels) they feel most confident.
- Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats.

“Co-Training”

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$
each instance has two views $\mathbf{x}_i = [\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}]$,
and a learning speed k.

1. let $L_1 = L_2 = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$.
2. Repeat until unlabeled data is used up:
3. Train view-1 $f^{(1)}$ from L_1, view-2 $f^{(2)}$ from L_2.
4. Classify unlabeled data with $f^{(1)}$ and $f^{(2)}$ separately.
5. Add $f^{(1)}$'s top k most-confident predictions $(\mathbf{x}, f^{(1)}(\mathbf{x}))$ to L_2.
   Add $f^{(2)}$'s top k most-confident predictions $(\mathbf{x}, f^{(2)}(\mathbf{x}))$ to L_1.
   Remove these from the unlabeled data.

“Noisy Student”

Require: Labeled images $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ and unlabeled images $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_m\}$.

1: Learn teacher model θ_* which minimizes the cross entropy loss on labeled images

$$\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f^{noised}(x_i, \theta))$$

2: Use an unnoised teacher model to generate soft or hard pseudo labels for unlabeled images

$$\tilde{y}_i = f(\tilde{x}_i, \theta_*), \forall i = 1, \cdots, m$$

3: Learn student model θ'_* which minimizes the cross entropy loss on labeled images and unlabeled images with noise added to the student model

$$\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f^{noised}(x_i, \theta')) + \frac{1}{m}\sum_{i=1}^{m} \ell(\tilde{y}_i, f^{noised}(\tilde{x}_i, \theta'))$$

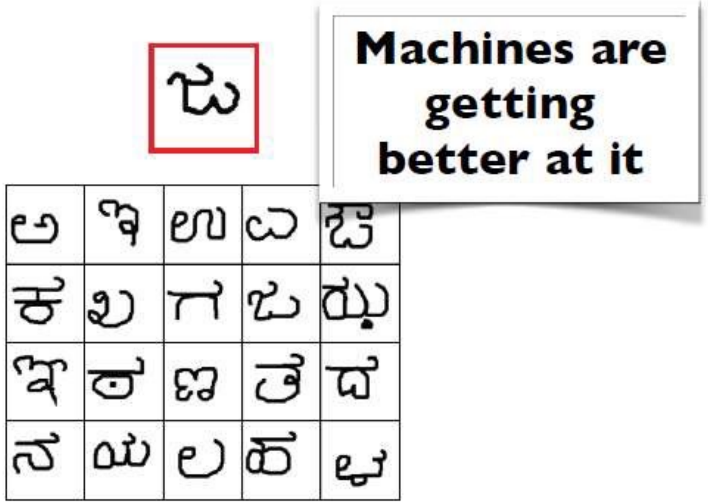
4: Iterative training: Use the student as a teacher and go back to step 2.

Method	# Params	Extra Data	Top-1 Acc.	Top-5 Acc.
ResNet-50 [23]	26M	-	76.0%	93.0%
ResNet-152 [23]	60M	-	77.8%	93.8%
DenseNet-264 [28]	34M	-	77.9%	93.9%
Inception-v3 [67]	24M	-	78.8%	94.4%
Xception [11]	23M	-	79.0%	94.5%
Inception-v4 [65]	48M	-	80.0%	95.0%
Inception-resnet-v2 [65]	56M	-	80.1%	95.1%
ResNeXt-101 [75]	84M	-	80.9%	95.6%
PolyNet [83]	92M	-	81.3%	95.8%
SENet [27]	146M	-	82.7%	96.2%
NASNet-A [86]	89M	-	82.7%	96.2%
AmoebaNet-A [54]	87M	-	82.8%	96.1%
PNASNet [39]	86M	-	82.9%	96.2%
AmoebaNet-C [13]	155M	-	83.5%	96.5%
GPipe [30]	557M	-	84.3%	97.0%
EfficientNet-B7 [69]	66M	-	85.0%	97.2%
EfficientNet-L2 [69]	480M	-	85.5%	97.5%
ResNet-50 Billion-scale [76]	26M	3.5B images labeled with tags	81.2%	96.0%
ResNeXt-101 Billion-scale [76]	193M		84.8%	-
ResNeXt-101 WSL [44]	829M		85.4%	97.6%
FixRes ResNeXt-101 WSL [71]	829M		86.4%	98.0%
Noisy Student (L2)	480M	300M unlabeled images	**87.4%**	**98.2%**

Few-Shot Learning

Human-level concept learning through probabilistic program induction

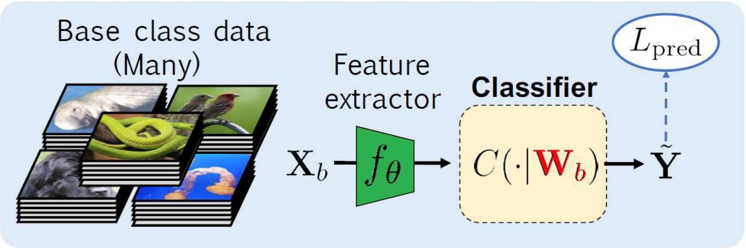
Brenden M. Lake,[1*] Ruslan Salakhutdinov,[2] Joshua B. Tenenbaum[3]



Normal Approach?

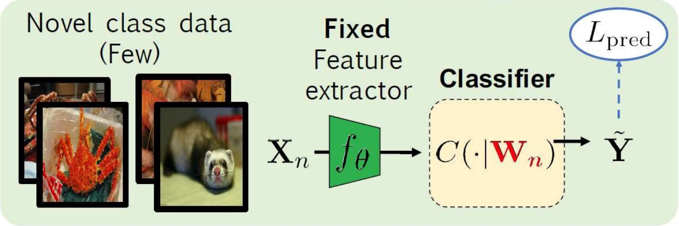
- **Do what we always do: Fine-tuning**
  - Train classifier on base classes

Training stage



  - Freeze features
  - Learn classifier weights for new classes using few amounts of labeled data (during "*query*" time!`

Fine-tuning stage



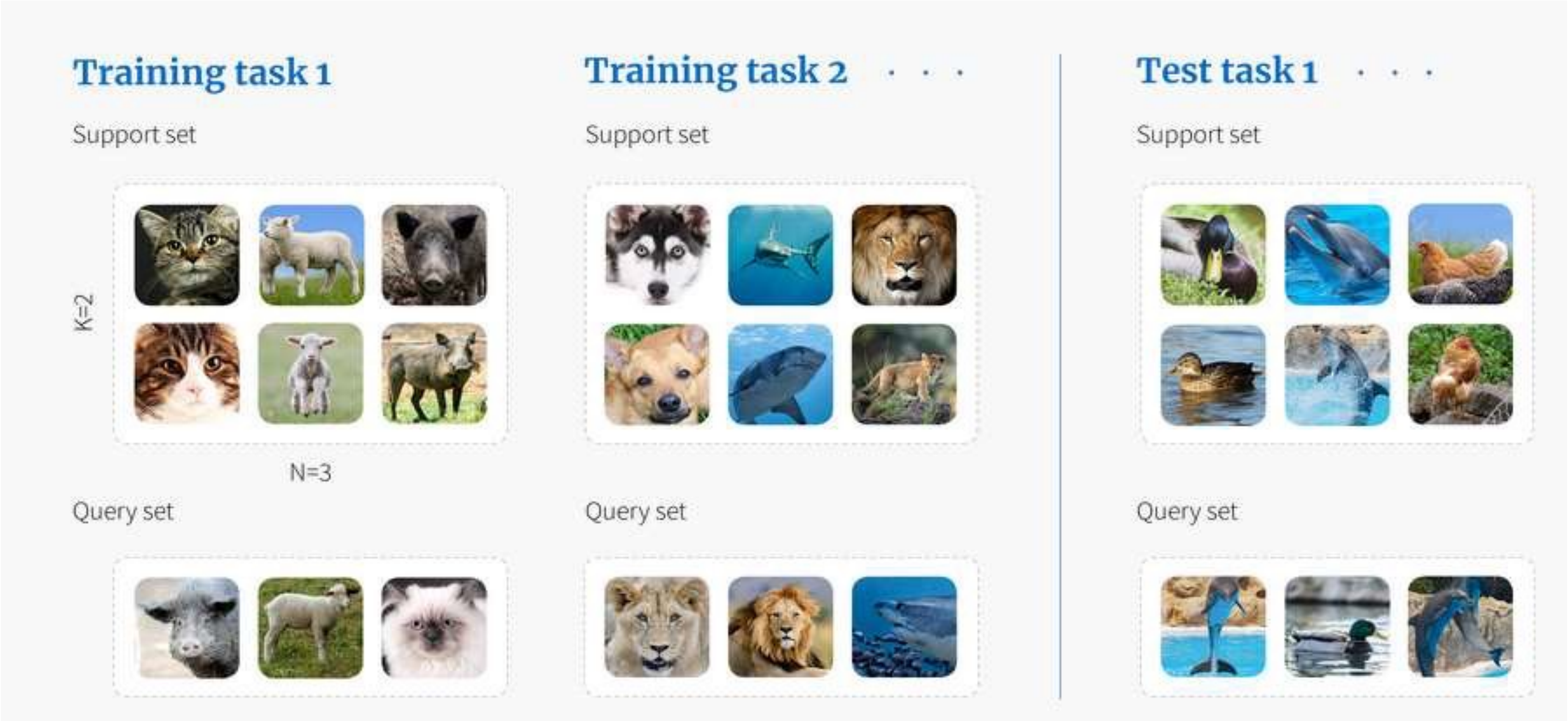
A Closer Look at Few-shot Classification, Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, Jia-Bin Huang

Cons?

- The training we do on the base classes does not factor the task into account
- No notion that we will be performing a bunch of N-way tests
- **Idea:** simulate what we will see during test time – and can do that many times!

Meta Learning Approach

- Set up a set of smaller tasks *during training* which simulates what we will be doing during testing

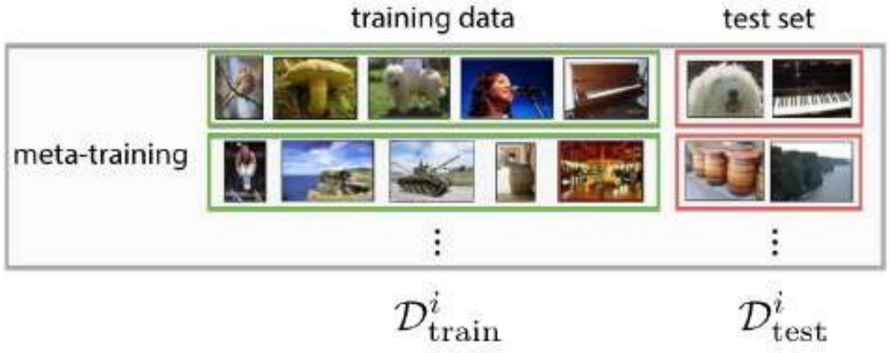


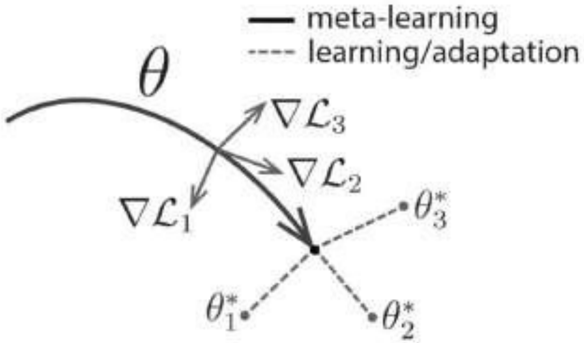
  – Can optionally pre-train features on held-out base classes (not typical)

- Testing stage is now the same, but with new classes

Model-Agnostic Meta-Learning (MAML)

a general recipe:





$$\theta \leftarrow \theta - \beta \sum_i \nabla_\theta \underbrace{\mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}^i_{\text{train}}), \mathcal{D}^i_{\text{test}})}_{\text{“meta-loss” for task } i}$$

* in general, can take more than one gradient step here

** we often use 4 – 10 steps

Chelsea Finn

Finn et al., “Model-Agnostic Meta-Learning”

Active Learning

From Education . . .

C. Bonwell and J. Eison [1]: In active learning, students participate in the process and students participate when they are doing something besides passively listening. It is a model of instruction or an education action that gives the responsibility of learning to learners themselves.

. . . to Machine Learning:

Settles [2, p.5]: Active learning systems attempt to overcome the labeling bottleneck by **asking queries in the form of unlabeled instances to be labeled by an oracle**. In this way, the active learner aims to achieve high accuracy using as few labeled instances as possible, thereby minimizing the cost of obtaining labeled data.

[1] Charles C. Bonwell and James A. Eison. Active learning: Creating excitement in the classroom. ASHE-ERIC Higher Education Report, 1, 1991.

[2] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison, Madison, Wisconsin, USA, 2009.

Active Learning

Setting

- Some information is costly (some not)
- Active learner controls selection process

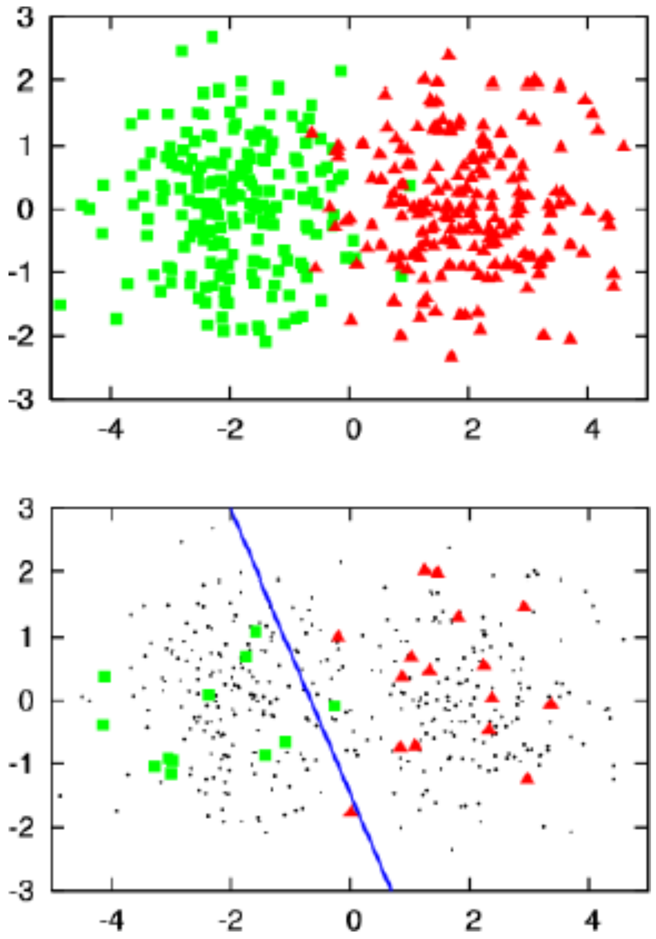
Objective

- Select the most valuable information
- Baseline: random selection

Historical Remarks

- Optimal experimental design
  - Valerii V. Fedorov. “Theory of Optimal Experiments Design”, Academic Press, 1972.
- Learning with queries/query synthesis
  - Dana Angluin. “Queries and concept learning”, Machine Learning, 2:319{342,1988.
- Selective sampling
  - David Cohn, L. Atlas, R. Ladner, M. El-Sharkawi, R. II Marks, M. Aggoune, and D. Park. “Training connectionist networks with queries and selective sampling”, In Advances in Neural Information Processing Systems (NIPS). Morgan Kaufmann, 1990.

Uncertainty sampling



Idea

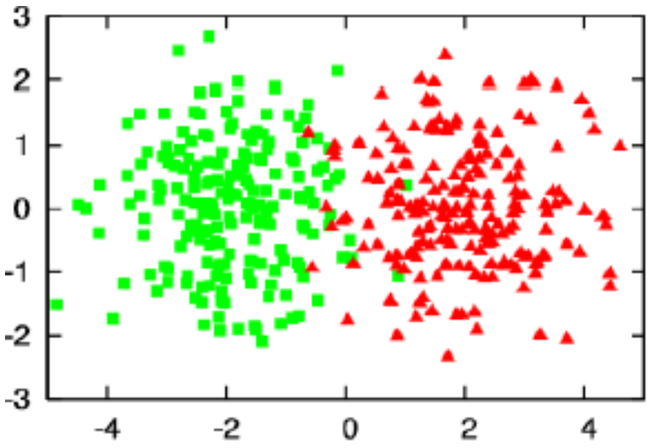
- Select those instances where we are least certain about the label

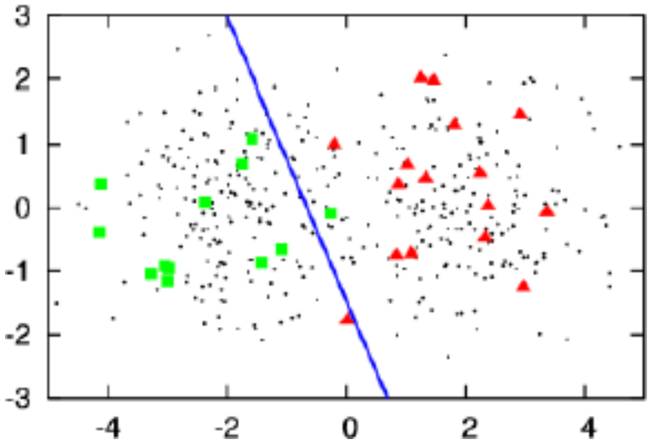
Approach

- 3 labels preselected
- Linear classifier
- Use distance to the decision boundary as uncertainty measure

"Training connectionist networks with queries and selective sampling".
David Cohn, L. Atlas, R. Ladner, M. El-Sharkawi, R. II Marks, M. Aggoune, and D. Park.
In Advances in Neural Information Processing Systems (NIPS). Morgan Kaufmann, 1990.

Uncertainty sampling



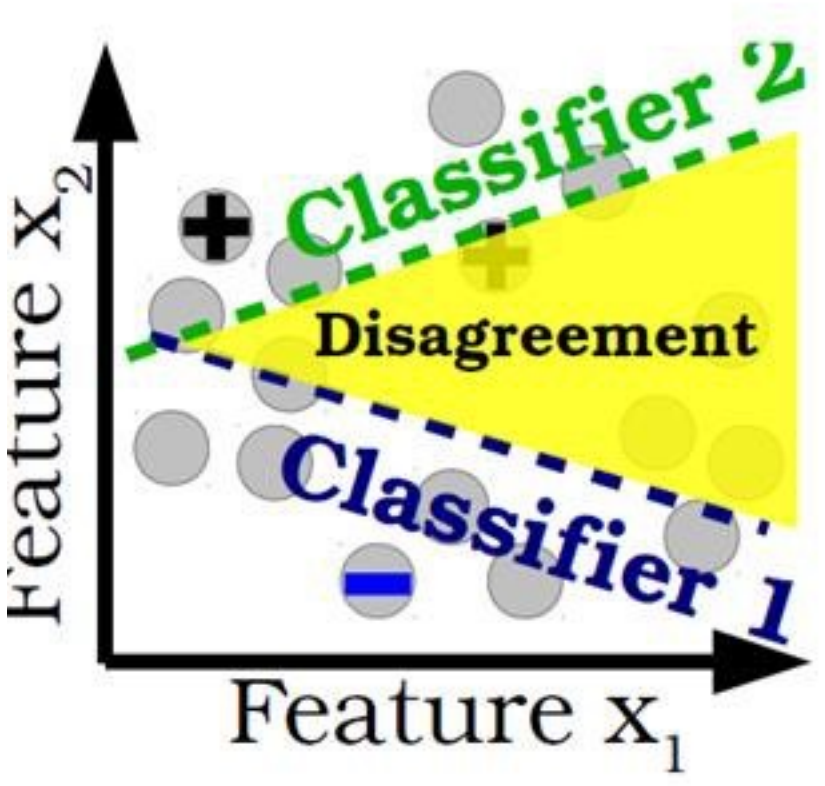


✚ easy to implement
✚ fast

-- no exploration (often combined with random sampling)
-- impact not considered (density weighted extensions exist)
-- problem with complex structures (performance can be even worse than random)

Pure exploitation, does not explore
Can get stuck in regions with high Bayesian error

Ensemble-based Sampling



Idea

Use disagreement between base classifiers

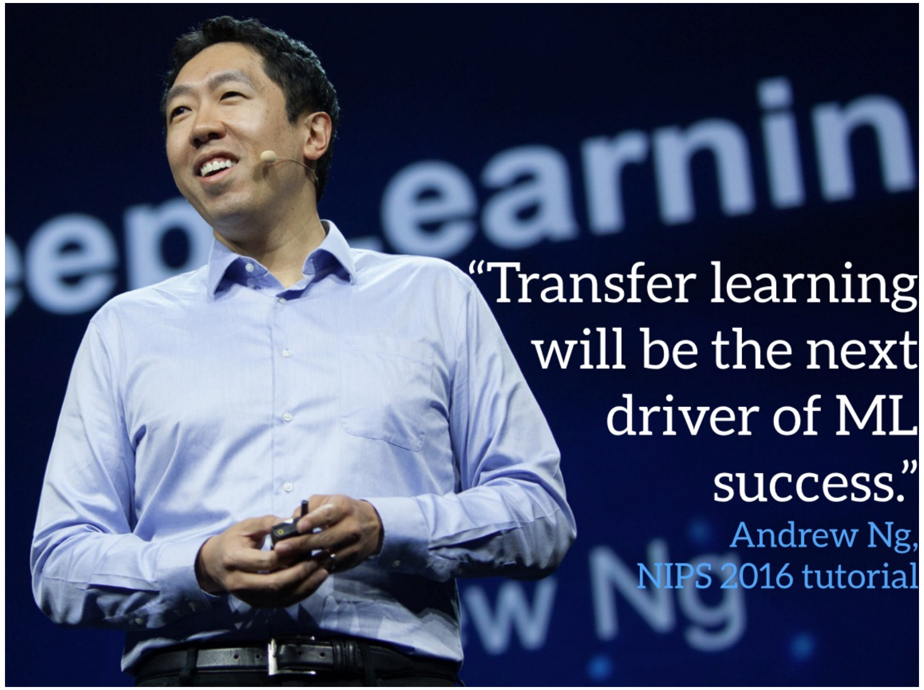
Approach

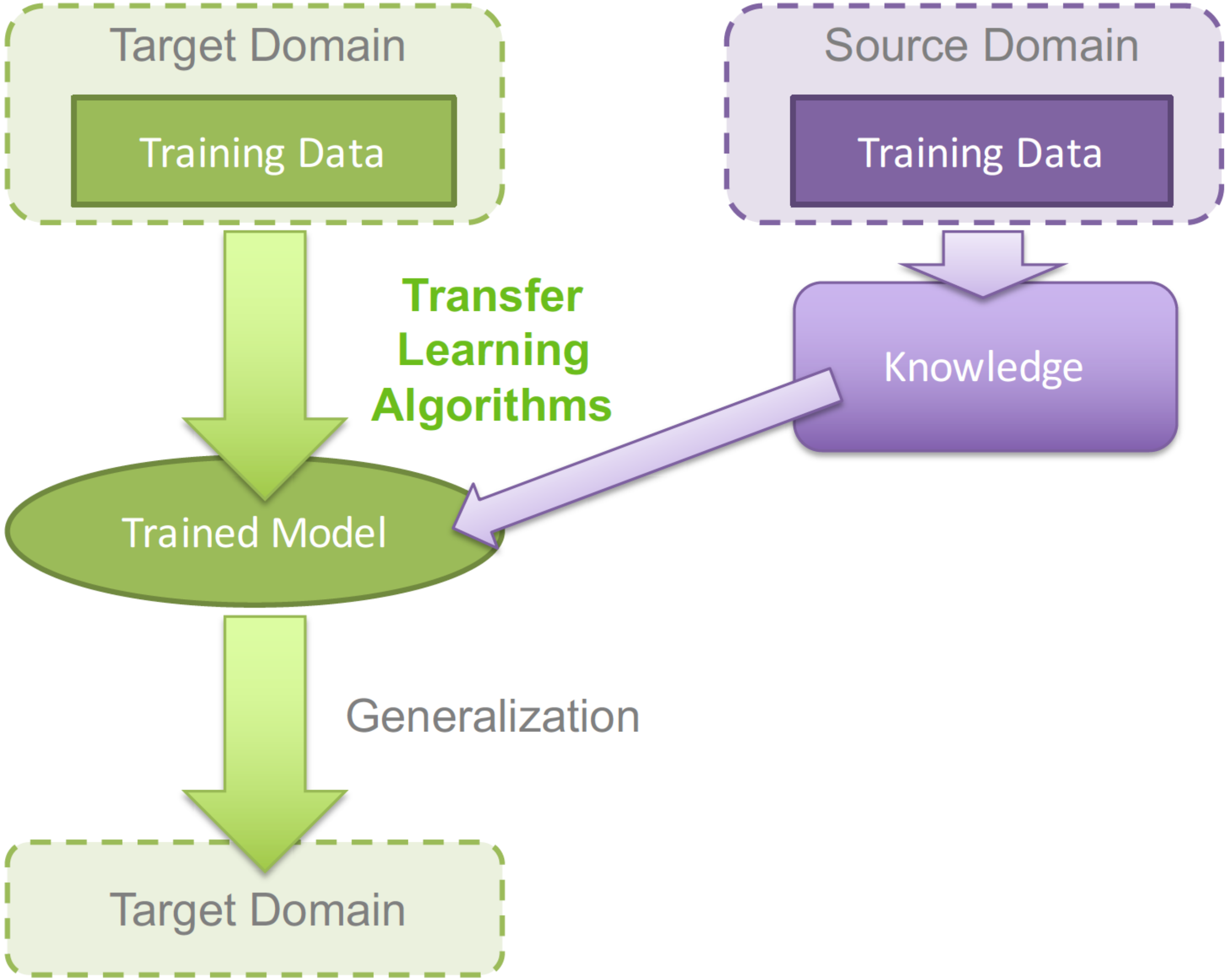
1. Get an initial set of labels
2. Split that set into (overlapping) subsets
3. On each subset, train a different base-classifier
4. Repeat until stop
5. On each unlabeled instance do
6. Apply all base-classifiers
7. Request label, if base-classifiers disagree
8. Update all base-classifiers
9. Go to step 4

"Query by committee", H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Fifth workshop on computational learning theory. Morgan Kaufmann, 1992.

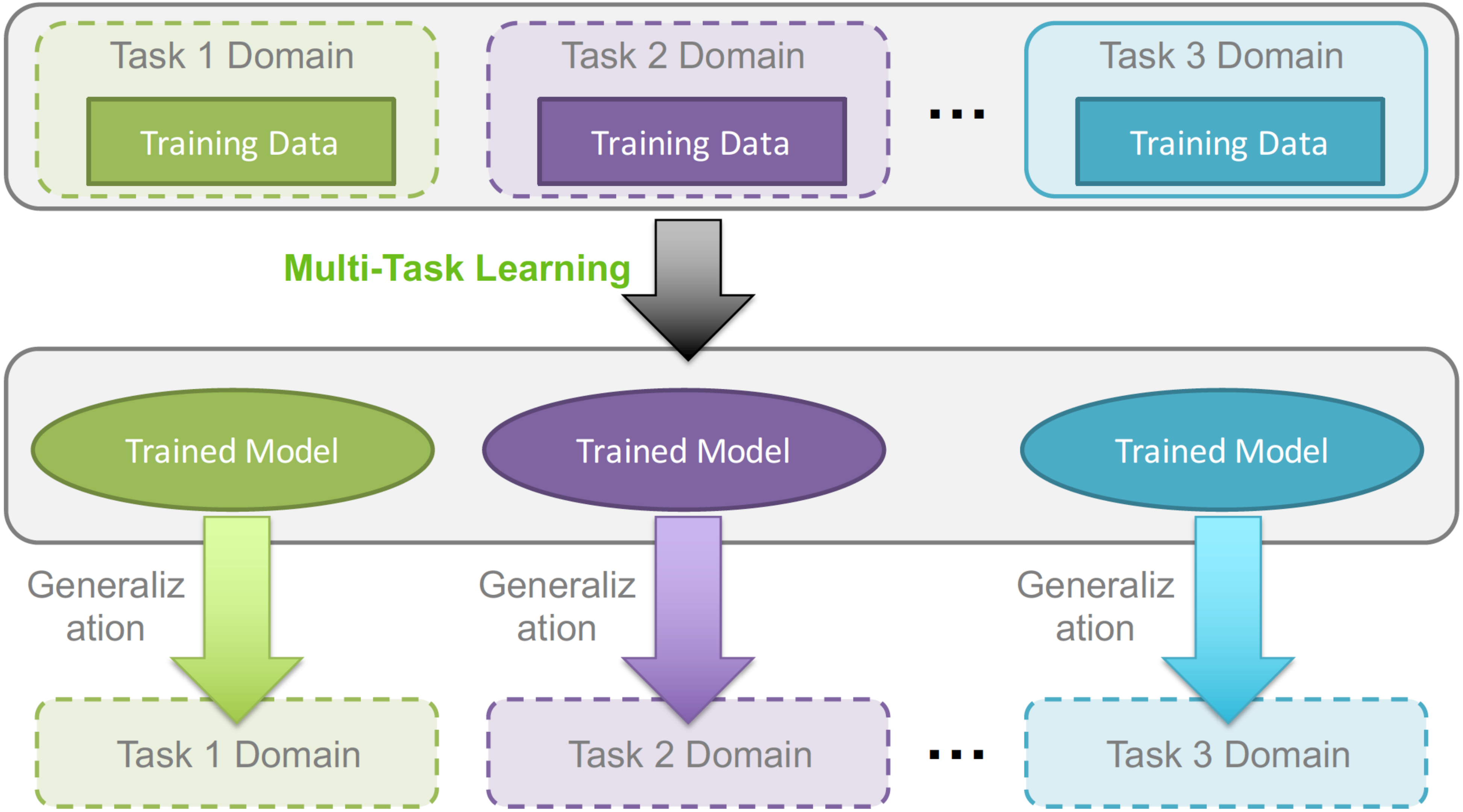
Transfer Learning

Improve Learning New Task
by Learned Task





Multi-Task Learning



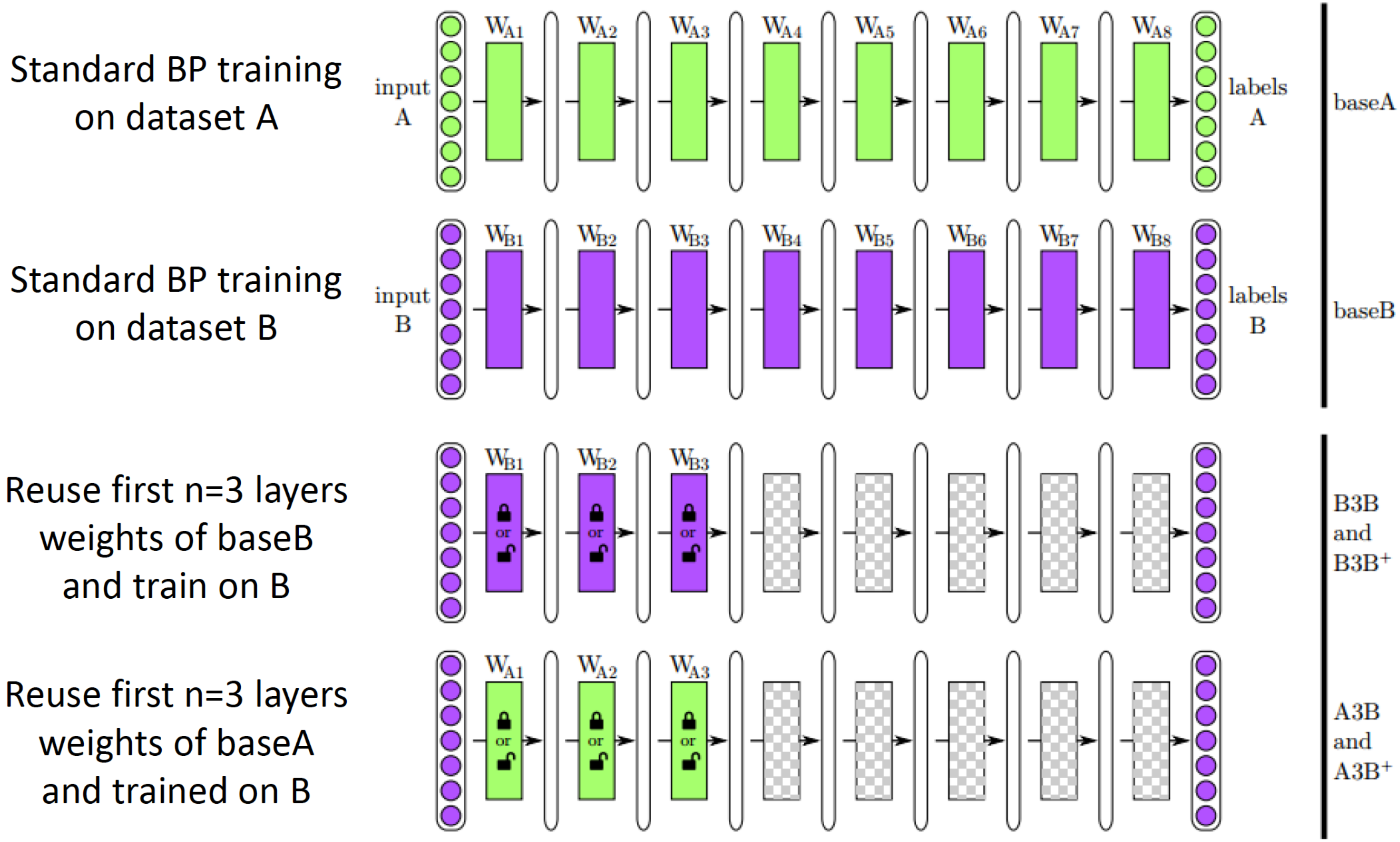
Transfer Learning: Main Solutions

- Instance (Data) Transfer
  - Reweight instances of target data according to source
  - Example: importance sampling; some “style-transfer” for data adaptation

- Feature Transfer
  - Mapping features of source and target data in a common space
  - Example: TCA; common pre-training + tuning methods in DL

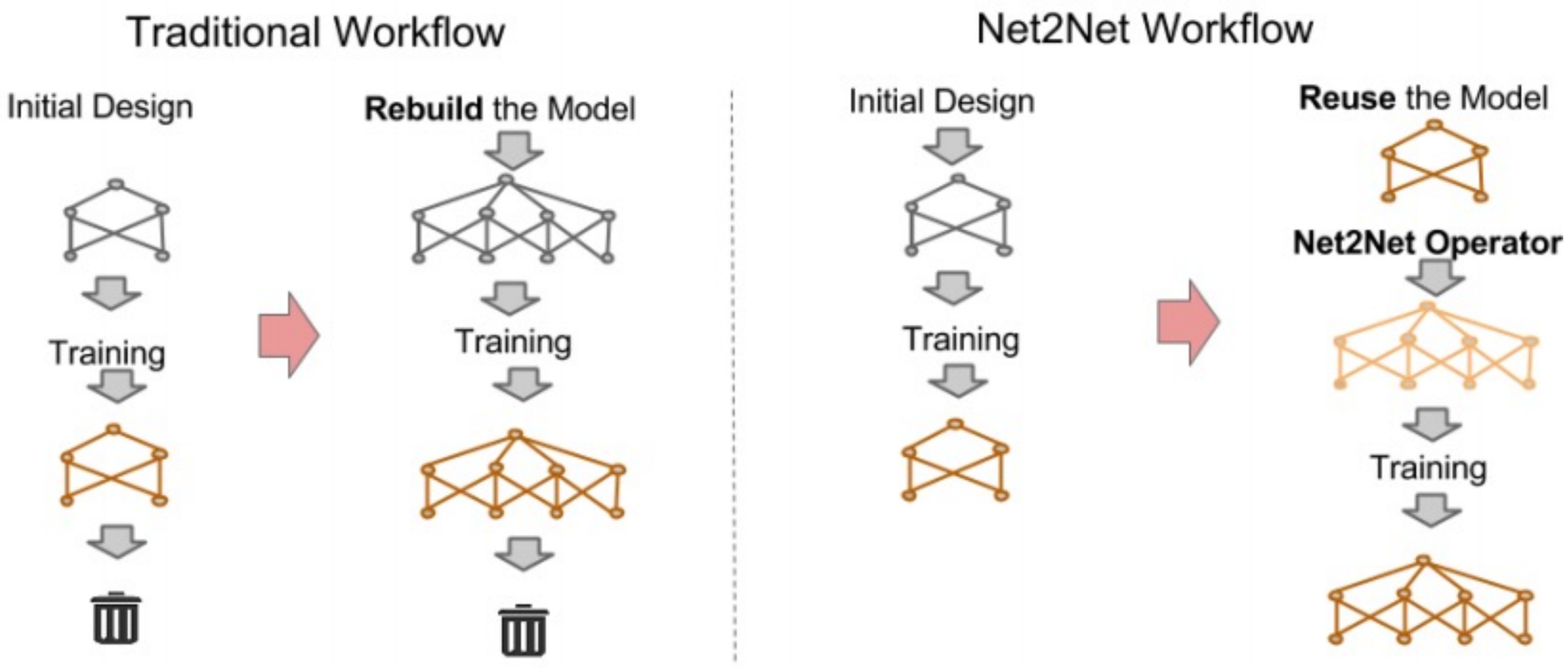
- Parameter Transfer
  - Learn target model parameters according to source model
  - Example: Multi-task learning; Net2Net

How transferable are deep learning features?



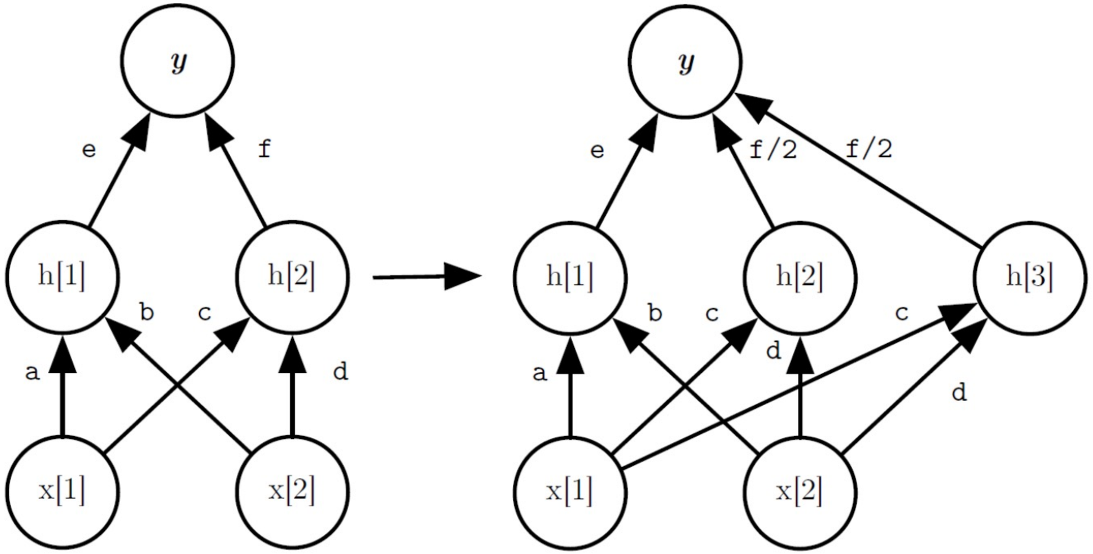
Net2Net Transfer

- Net2Net reuses information of an already trained deep model to speedup training of a new model (potentially different topology)

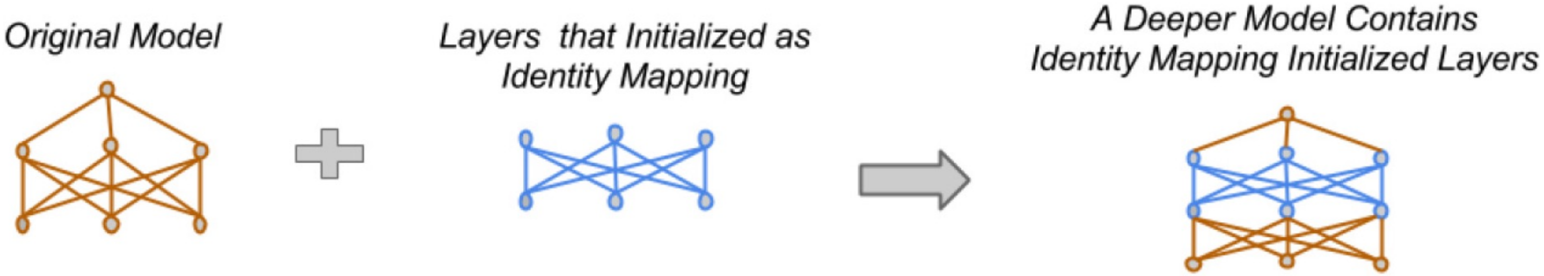


Net2Net Transfer

- Wider



- Deeper

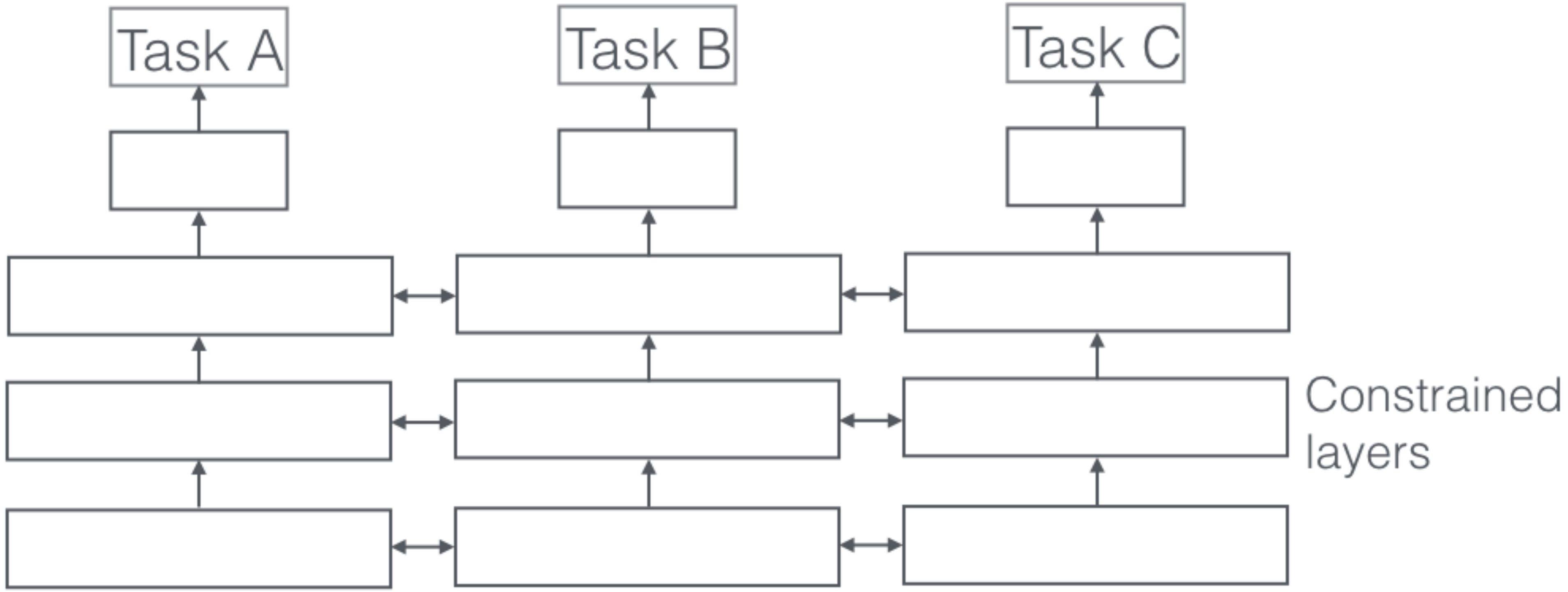


Multi-Task Learning: Main Solutions

- Direct Parameter Sharing (straightforward)
  - Examples: shared weights or activations in neural networks; shared parameters in Gaussian process

- Structural Regularization
  - Can be designed to incorporate various assumptions and domain knowledge
  - Can be trained using large-scale optimization algorithms on big data
  - The key is to design the regularization term that couples the tasks
  - Classical examples: group sparsity, low-rank, parameters grouping…

General Multi-Task Learning Schematic in DNNs



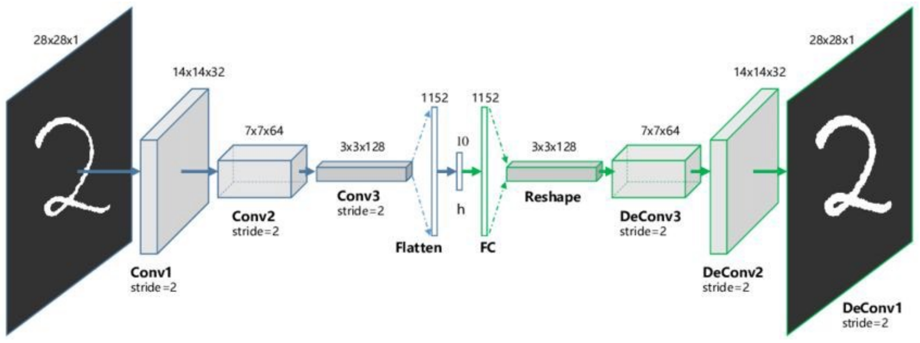
- Can often help tasks by fewer labels, due to knowledge sharing… (“**positive transfer**”)
- But can backfire some tasks during collaboration too, due to cross-task conflict… (“**negative transfer**”)

Now let's get ambitious: learning with NO Labels!!

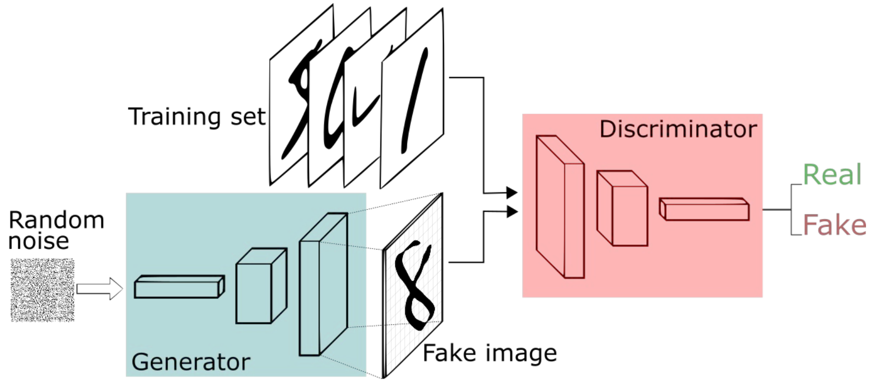
- ▶ **"Pure" Reinforcement Learning (cherry)**
  - ▶ The machine predicts a scalar reward given once in a while.
  - ▶ **A few bits for some samples**
- ▶ **Supervised Learning (icing)**
  - ▶ The machine predicts a category or a few numbers for each input
  - ▶ Predicting human-supplied data
  - ▶ **10→10,000 bits per sample**
- ▶ **Self-Supervised Learning (cake génoise)**
  - ▶ The machine predicts any part of its input for any observed part.
  - ▶ Predicts future frames in videos
  - ▶ **Millions of bits per sample**

First category of unsupervised learning

- Generative modeling
  - Generate or otherwise model pixels in the input space
  - Pixel-level generation is computationally expensive
  - Generating images of high-fidelity may not be necessary for representation learning



Autoencoder

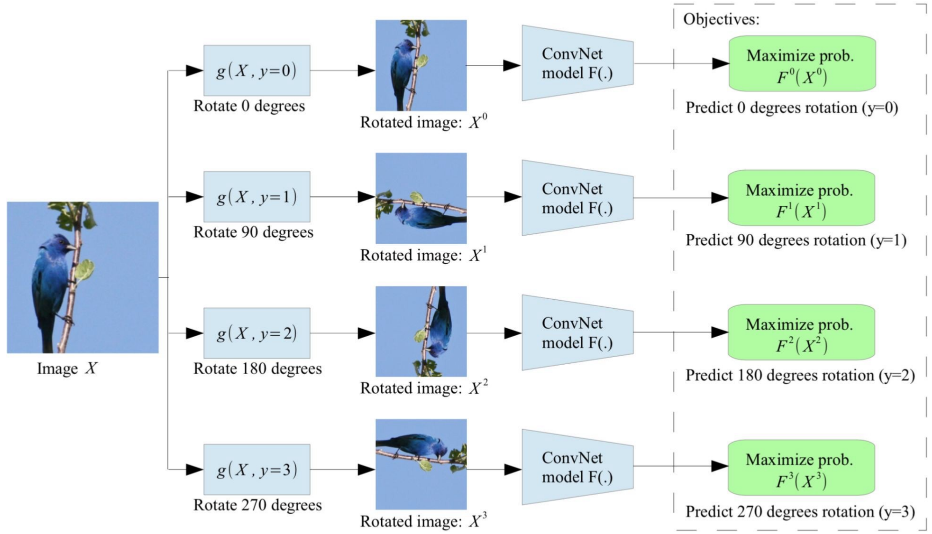


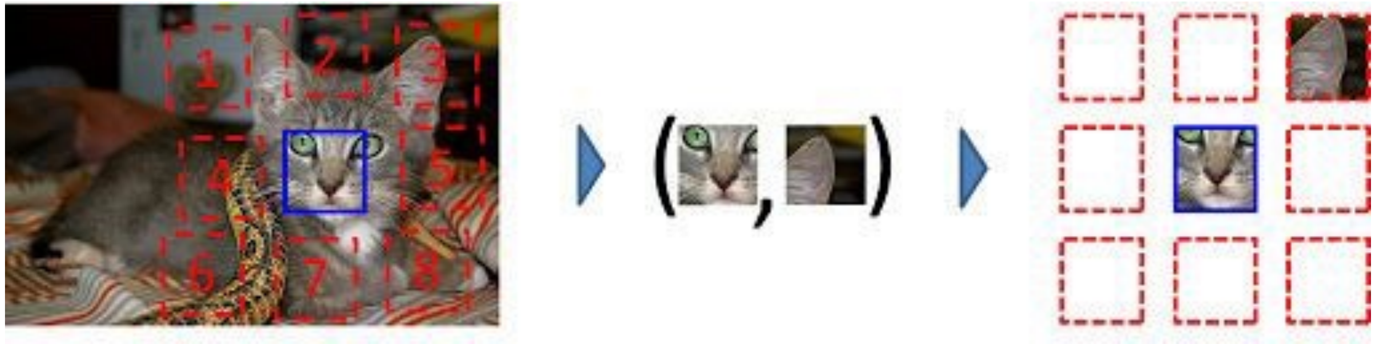
Generative Adversarial Nets

Image credit: Xifeng Guo, Thalles Silva.

Second category of unsupervised learning

- Discriminative modeling
  - Train networks to perform *pretext tasks* where both the inputs and labels are derived from an unlabeled dataset.
  - Heuristic-based pretext tasks: rotation prediction, relative patch location prediction, colorization, solving jigsaw puzzle.
  - Many heuristics seem ad-hoc and may be limiting.

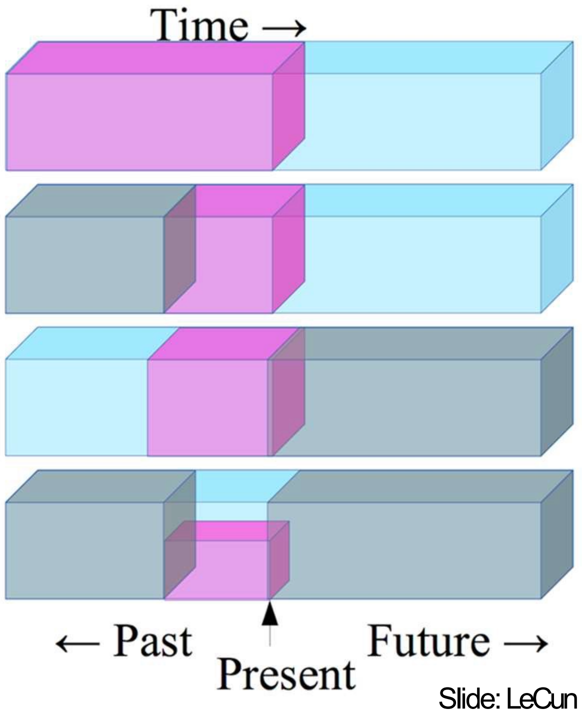




Images: [Gidaris et al 2018, Doersch et al 2015]

Motivation and Methodology

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- Pretend there is a part of the input you don't know and predict that.

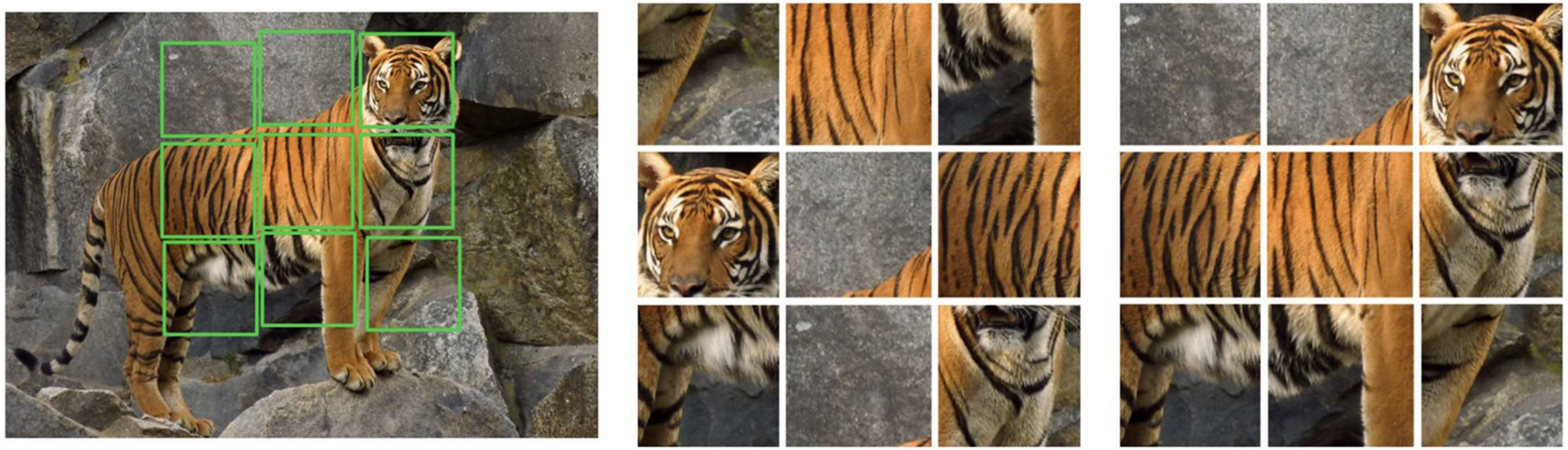


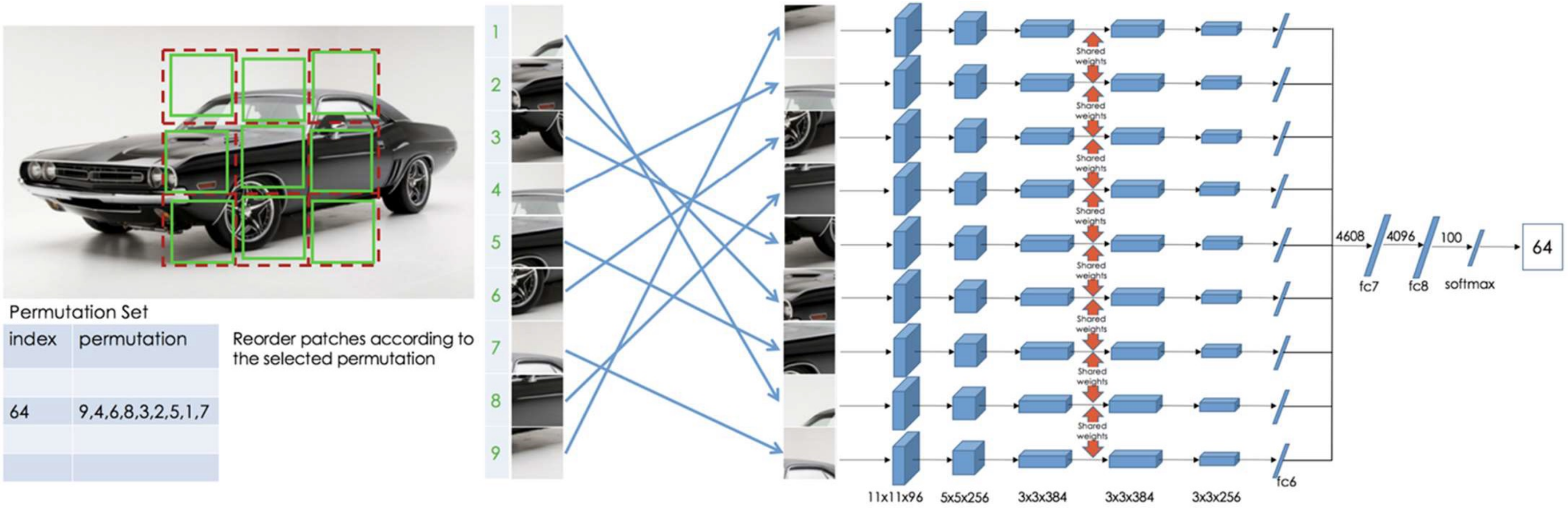
Slide: LeCun

Main Tasks in Use:

- Reconstruct from a corrupted (or partial) version
  - Denoising Autoencoder
  - In-painting
  - Colorization
- Visual common-sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - word2vec
  - Contrastive Predictive Coding (CPC)
  - **MoCO, simCLR …**

Example: Solving Jigsaw Puzzles





Simple Contrastive Learning (simCLR)

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]

- **Simple idea:** maximizing the agreement of representations under data transformation, using a contrastive loss in the latent/feature space
- **Super effective:** 10% relative improvement over previous SOTA (cpc v2), outperforms AlexNet with 100X fewer labels

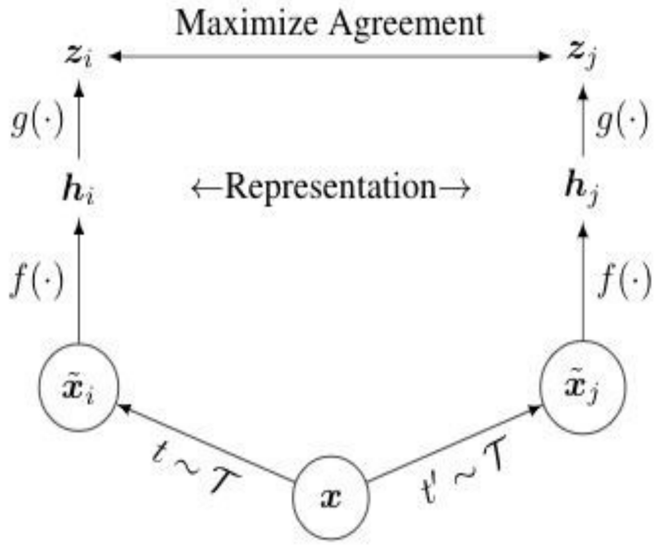
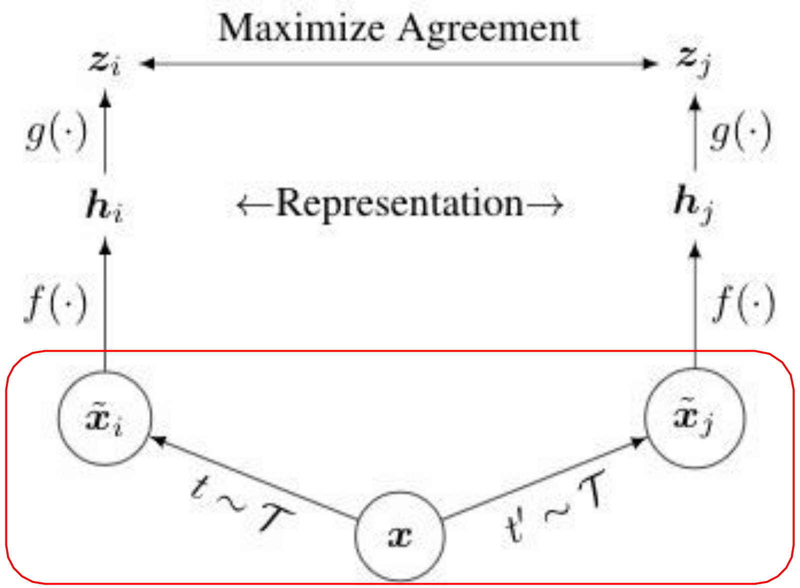


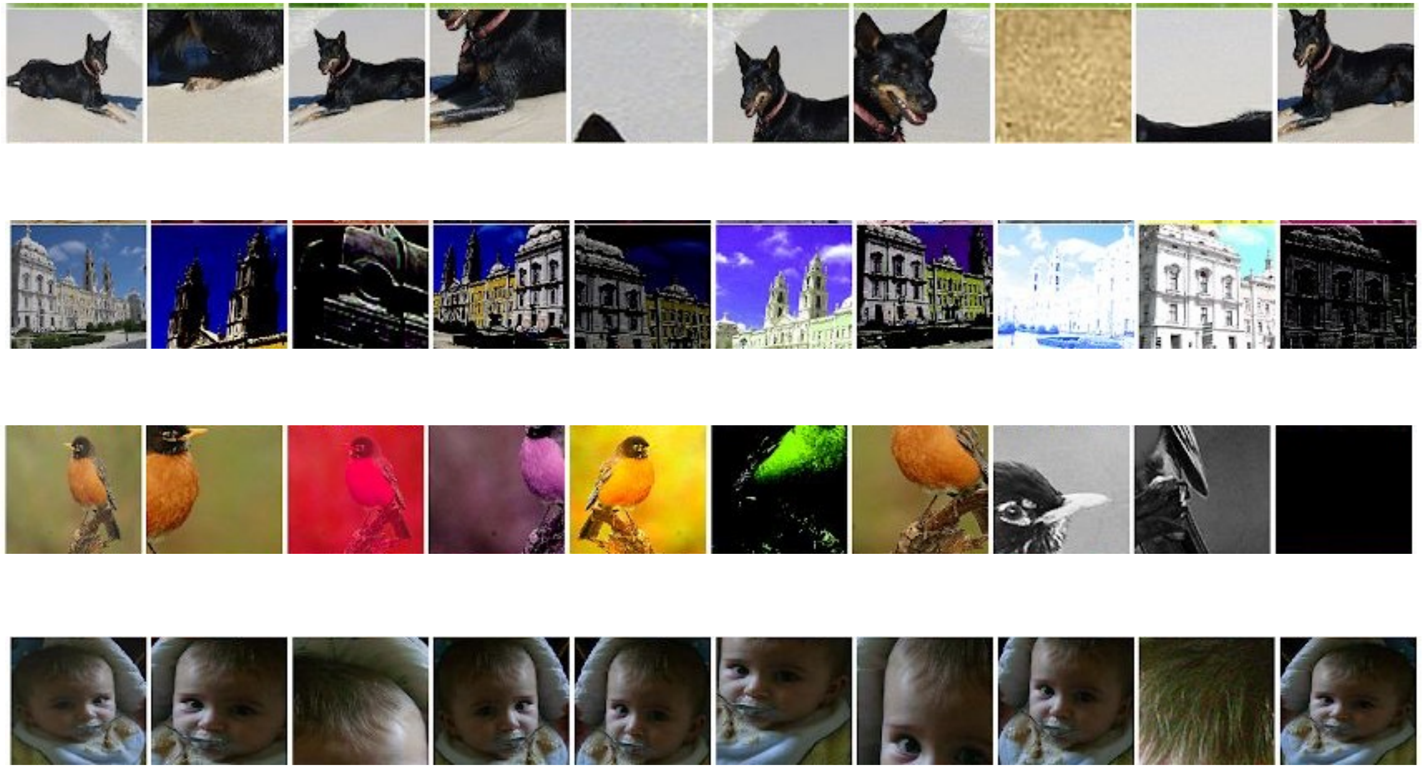
Figure 2. A framework for contrastive representation learning. Two separate stochastic data augmentations $t, t' \sim \mathcal{T}$ are applied to each example to obtain two correlated views. A base encoder network $f(\cdot)$ with a projection head $g(\cdot)$ is trained to maximize agreement in *latent representations* via a contrastive loss.

Simple Contrastive Learning Contrast (simCLR)

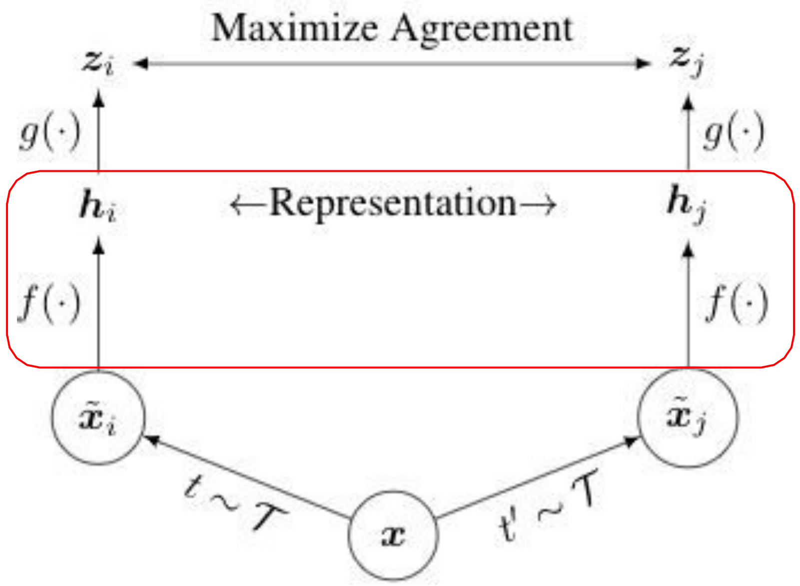


simCLR uses random crop and color distortion for augmentation.

Examples of augmentation applied to the left most images:

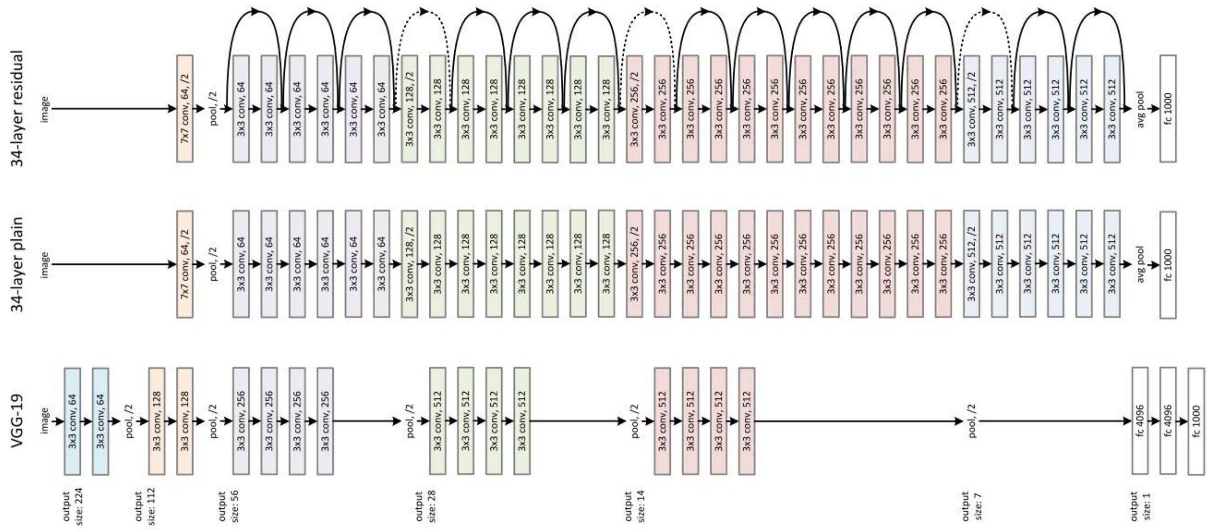


Simple Contrastive Learning Contrast (simCLR)

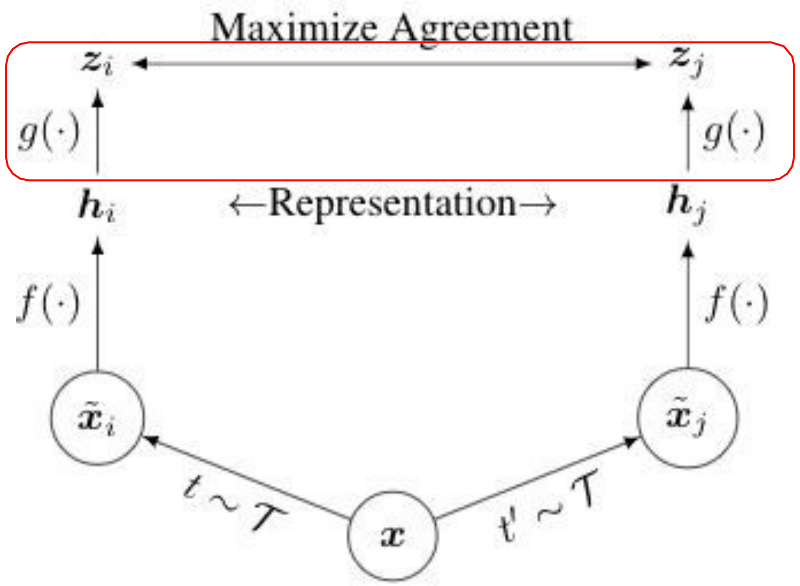


f(x) is the base network that computes internal representation.

Default simCLR uses (unconstrained) ResNet in this work.
However, it can be other networks.

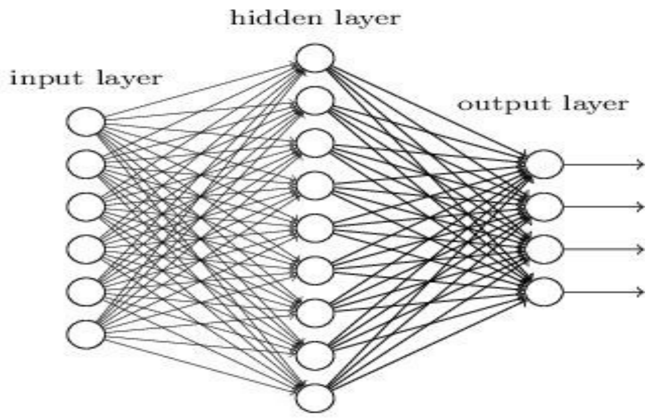


Simple Contrastive Learning Contrast (simCLR)



g(h) is a projection network that project representation to a latent space.

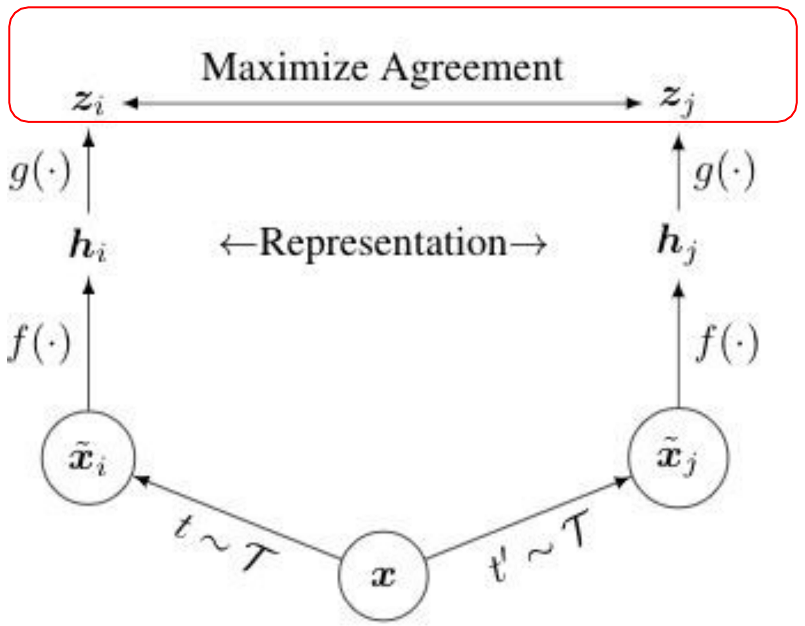
simCLR use a 2-layer non-linear MLP



Simple Contrastive Learning Contrast (simCLR)

In the h-representation space we do two things:

- **“Pull”** positive pairs closer together (two contrastive “views” generated from the same sample, only with different data augmentations
- “**Push”** negative pairs further away



Loss function (InfoNCE):

$$\text{Let } \mathrm{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$$

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

Original image | crop 1 | crop 2 | contrastive image

simCLR algorithm in pseudo code

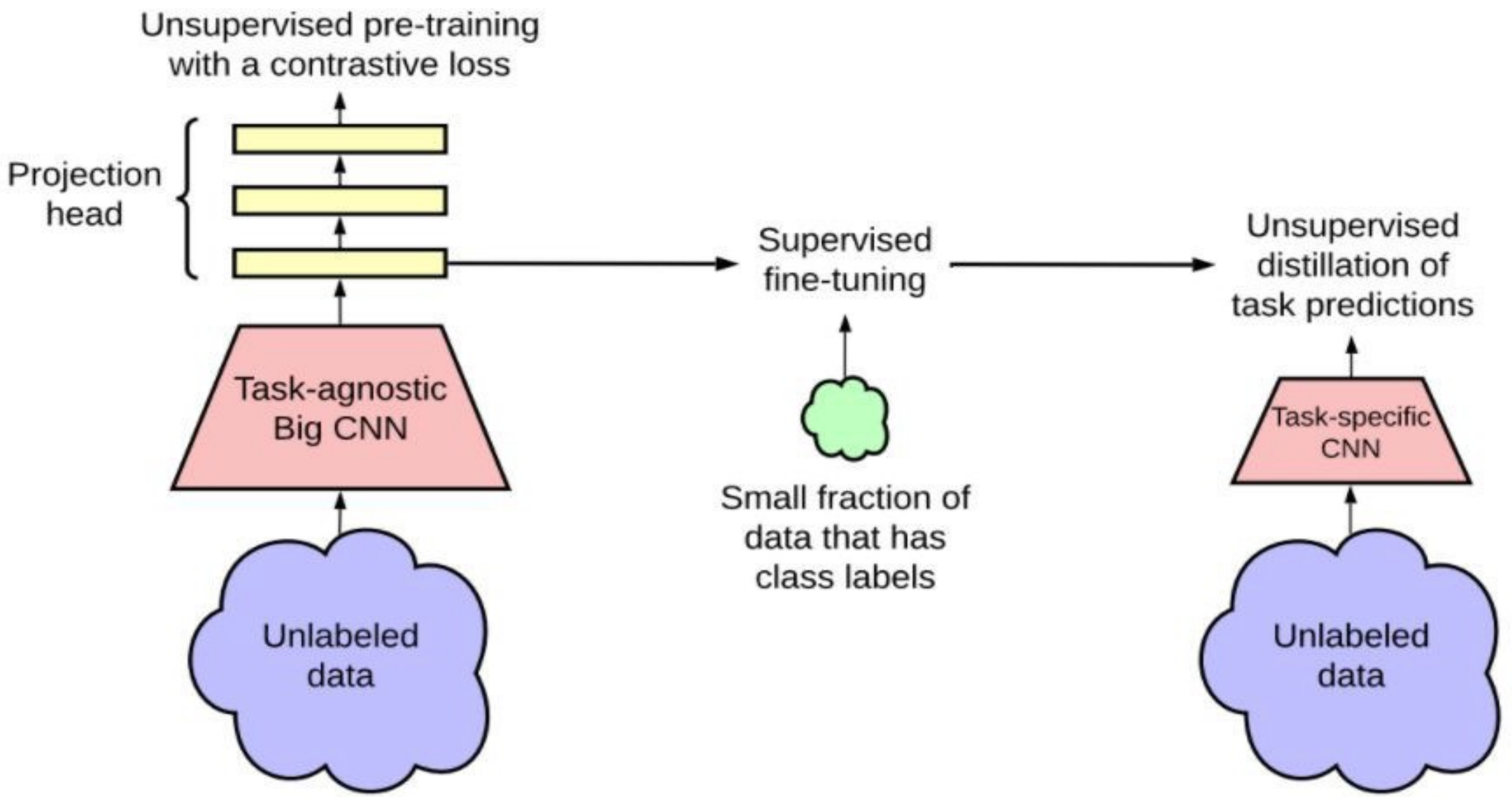
Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, temperature τ, form of f, g, $\mathcal{T}$.
for sampled mini-batch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\tau \|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i,j)$ **as** $-s_{i,j} + \log \sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network f

Take-home key points:

- Benefit from large batch sizes (at least, 1k-2k per minibatch)
- Composition of augmentations are crucial. Contrastive learning needs stronger data/color augmentation than supervised learning
- A nonlinear projection head improves the representation quality of the layer before it
- "Temperature hyperparameter" in the contrastive loss is very critical
- simCLR can immediately be used to few-shot, semi-supervised, and transfer learning
- Unsupervised contrastive learning benefits (more) from bigger models (simCLR v2)

simCLR as a strong semi-supervised learner



"Pre-train, Fine-tune, and Distill"

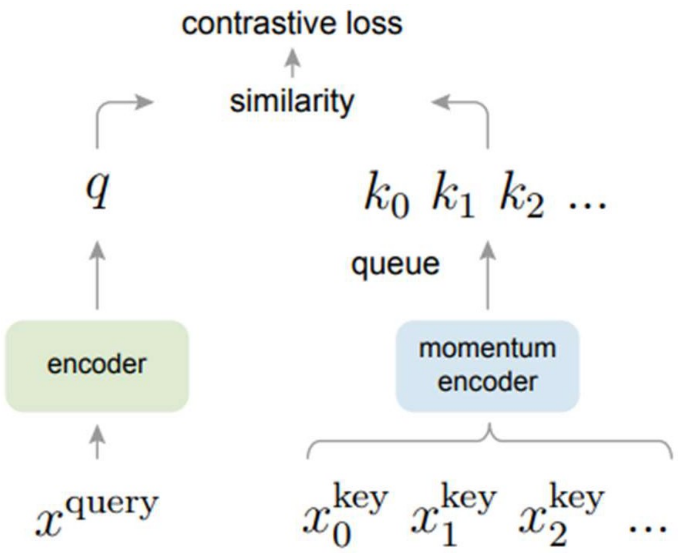
- **Surprise:** Bigger models are more label-efficient!
- Using pre-training + fine-tuning, "the fewer the labels, the bigger the model"

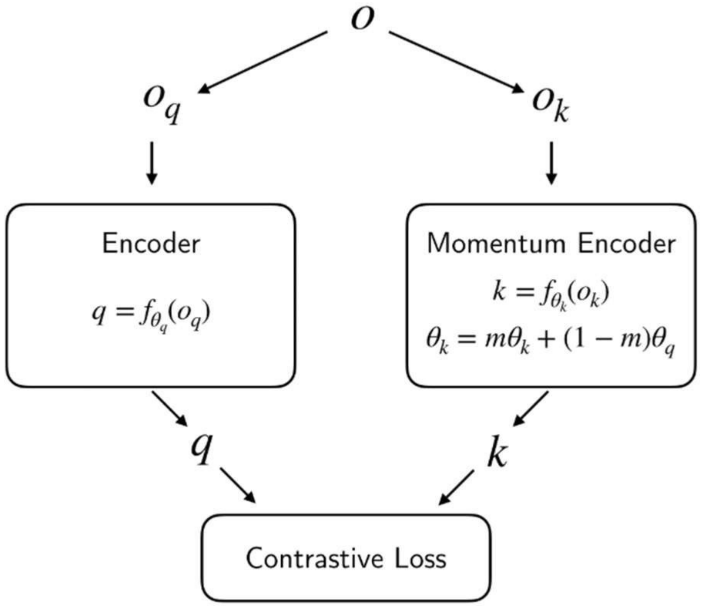
Momentum Contrast (MoCo)

Momentum Contrast for Unsupervised Visual Representation Learning

Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

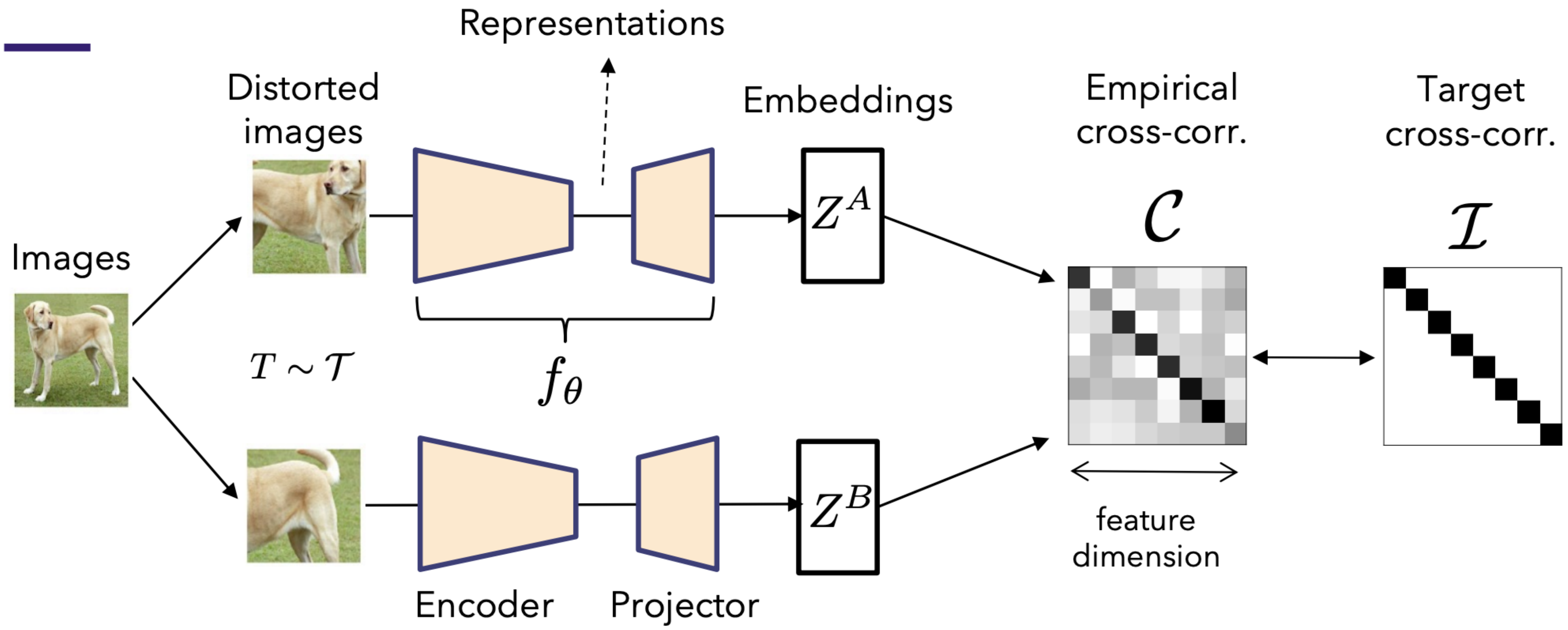
Facebook AI Research (FAIR)





$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

Barlow Twins: “Another Dimension” of Contrast



$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1-\mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} {\mathcal{C}_{ij}}^2}_{\text{redundancy reduction term}}$$

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}$$

VIC-Reg: A (more) Unified SSL Framework

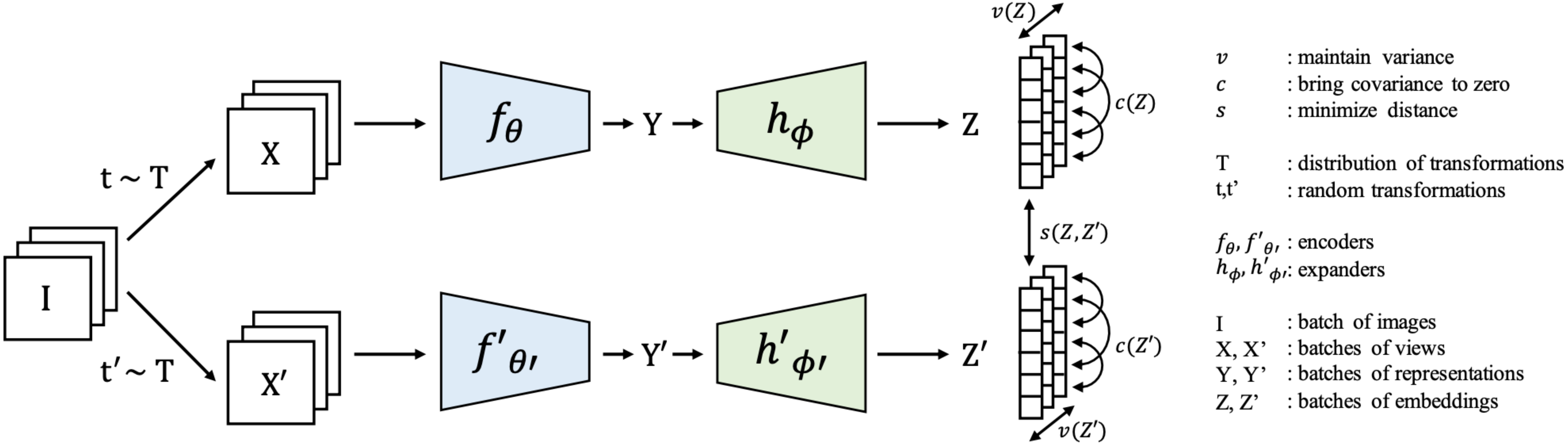
Promoted a lot by LeCun, etc.

… who argues *three essential things* constitute a good SSL loss:

- **Variance:** keeps the variance of each component of the representations (measured over a batch) above a threshold, to prevent cross-sample collapse. [contrastive learning, ”push” negative]
- **Invariance:** make the two similar representations as close to each other as possible [contrastive learning, ”pull” positive]
- **Covariance:** decorrelates the variables of one sample’s embedding and prevents an informational collapse in which the variables would vary together or be highly correlated. [barlow twins; non-existent in CL]

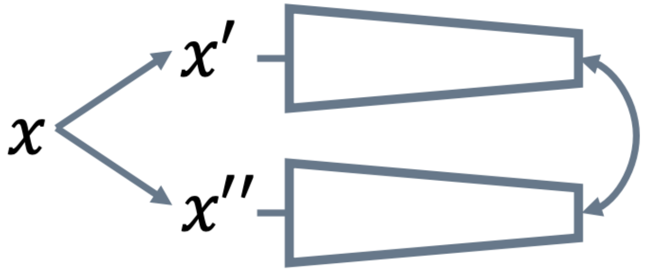
VIC-Reg (promoted a lot by LeCun, etc.)

- Joint embedding with variance, invariance and covariance regularization

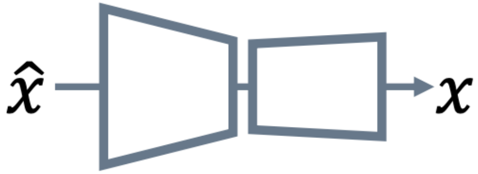


Beyond Contrast Learning: Masked Auto-Encoder (MAE)

- Contrastive / Siamese

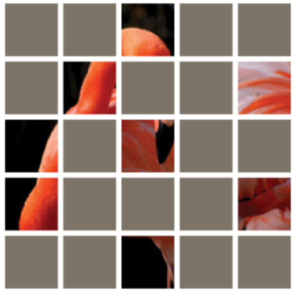


- Reconstructive / Auto-Encoding



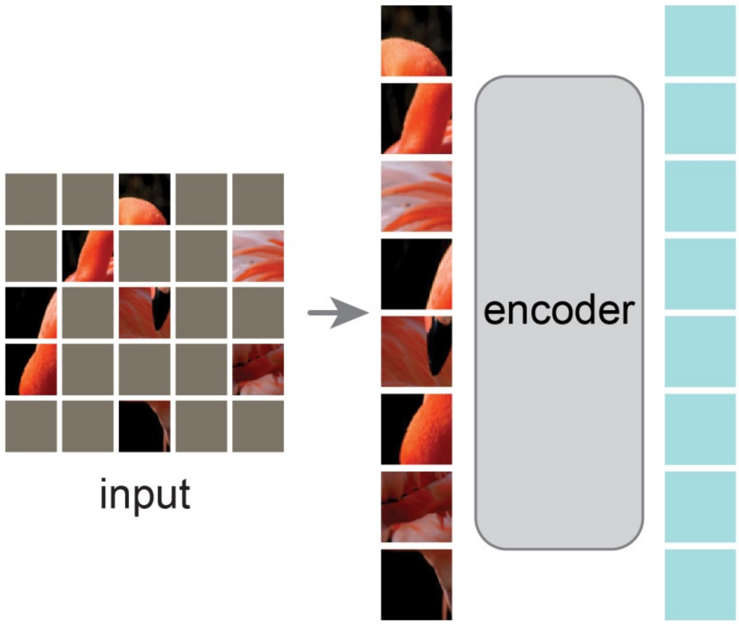
A more detailed tutorial: https://feichtenhofer.github.io/eccv2022-ssl-tutorial/Tutorial_files/slides/mae_tutorial_xinlei.pdf

How MAE works



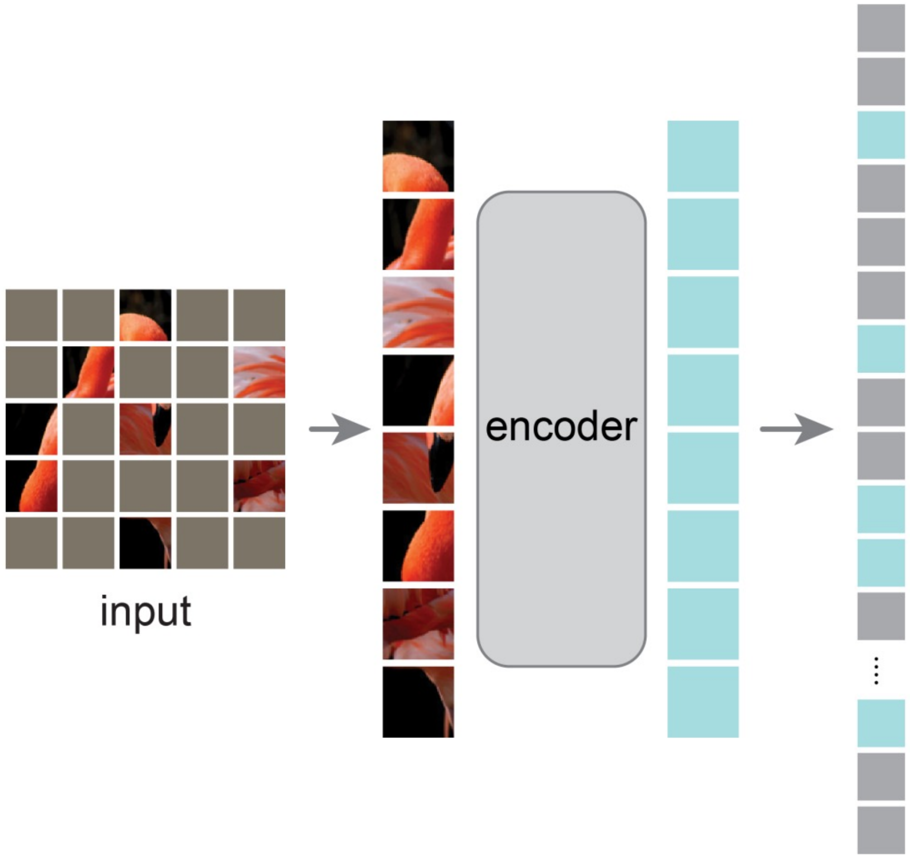
Random masking

How MAE works



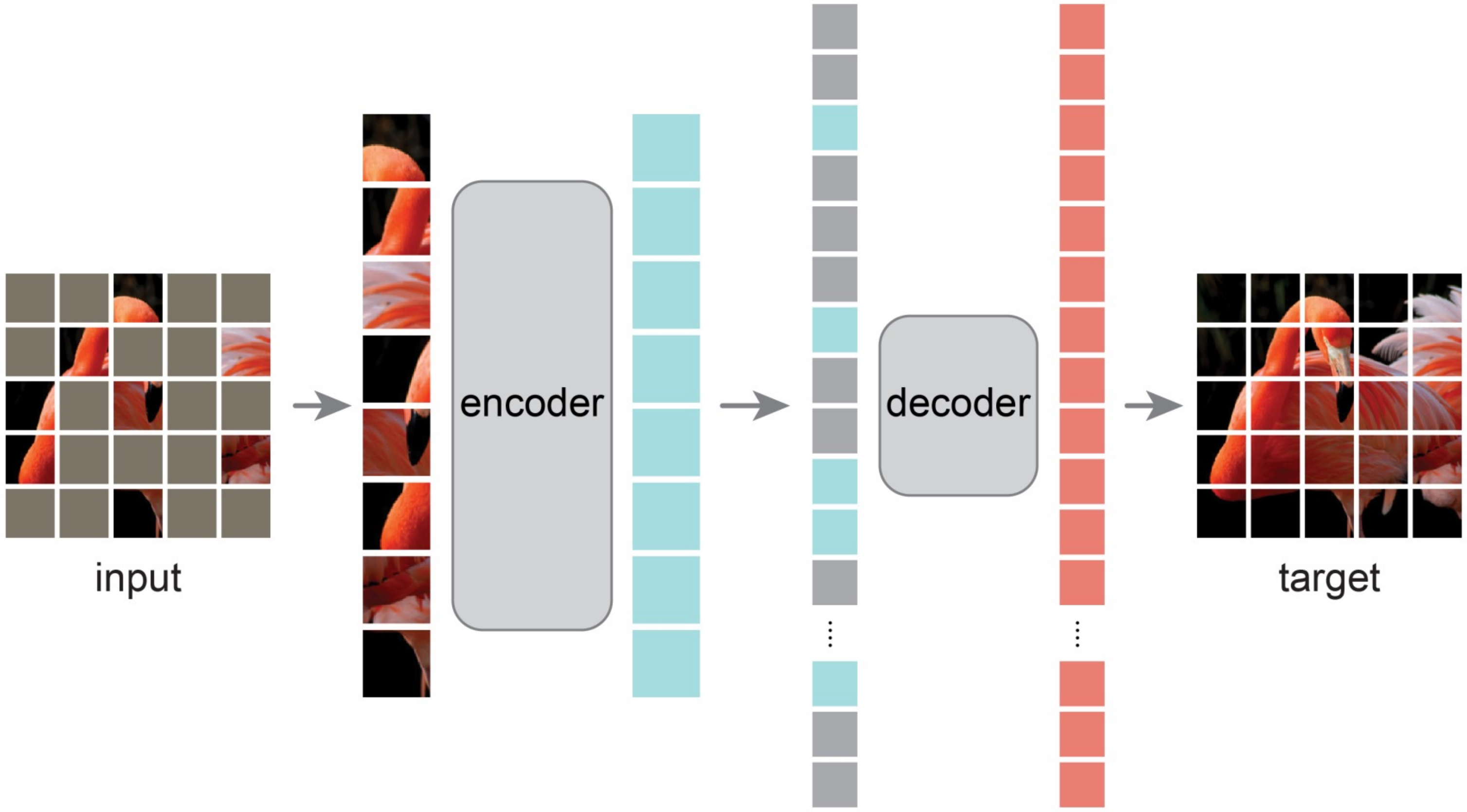
Encode visible patches

How MAE works



Add mask tokens

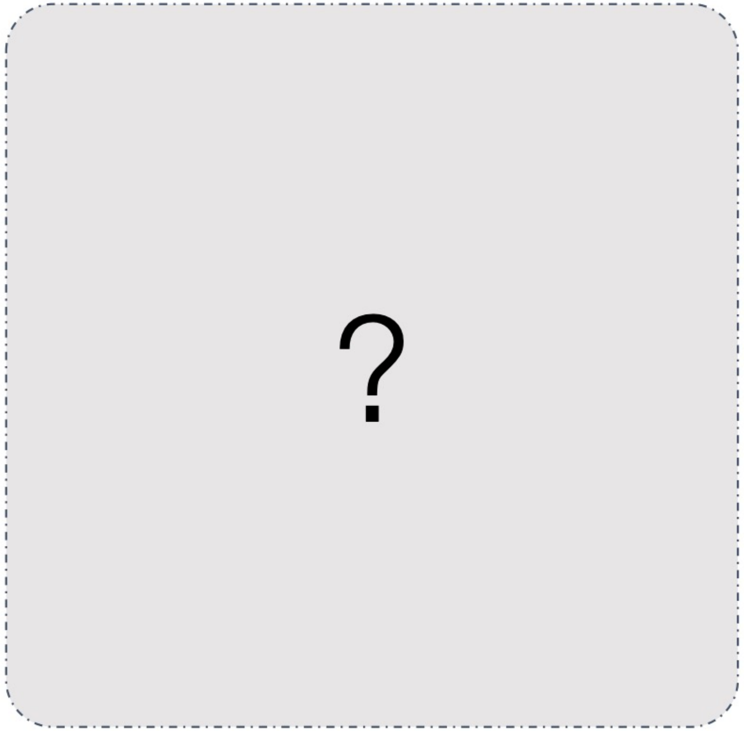
How MAE works



Reconstruct

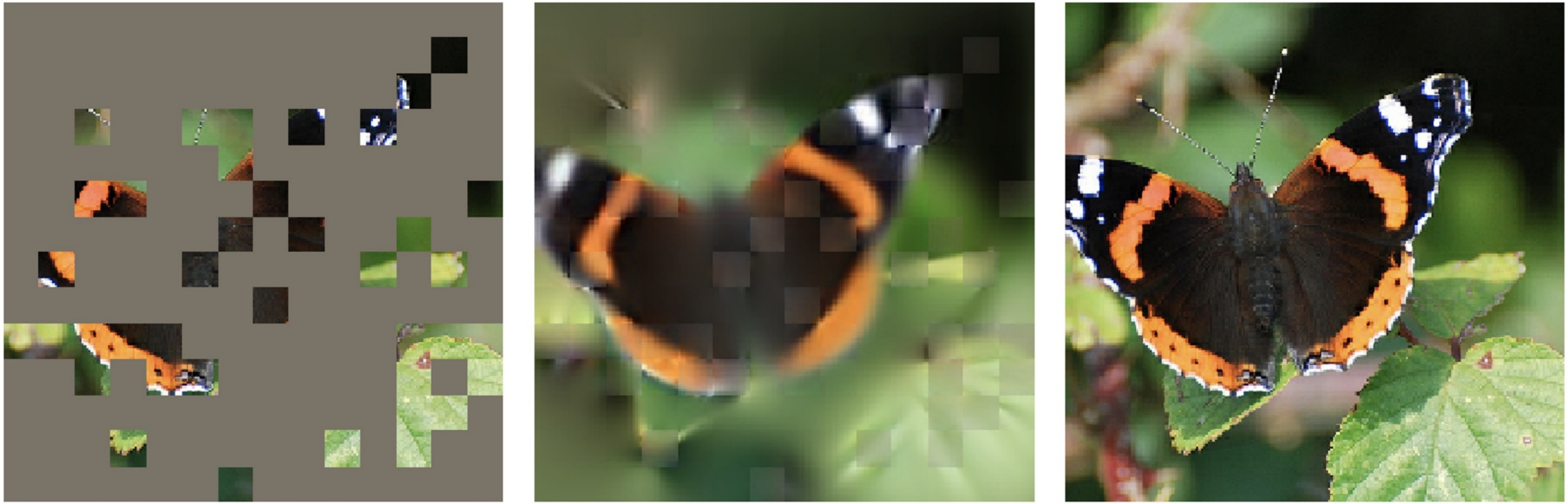
MAE works by Reconstruction

Masked input: 80%



You guess?

MAE works by Reconstruction

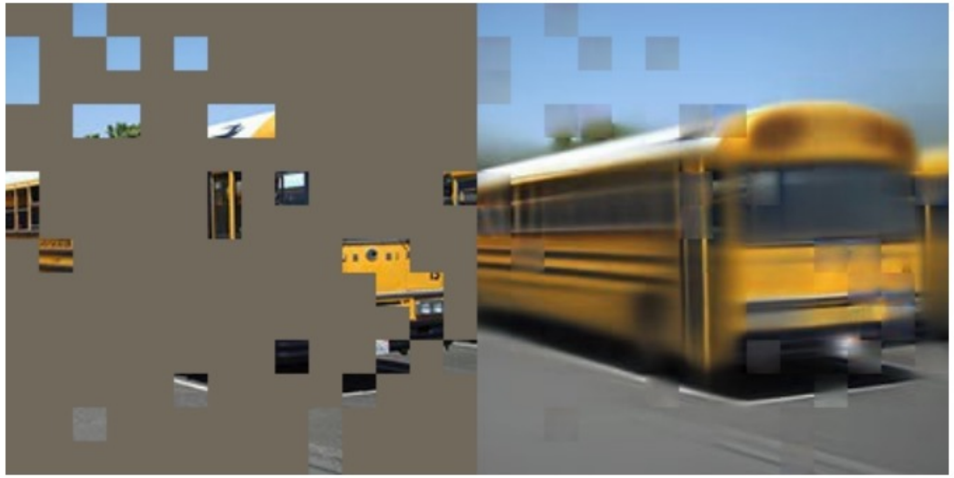


Masked input: 80% MAE's guess Ground truth

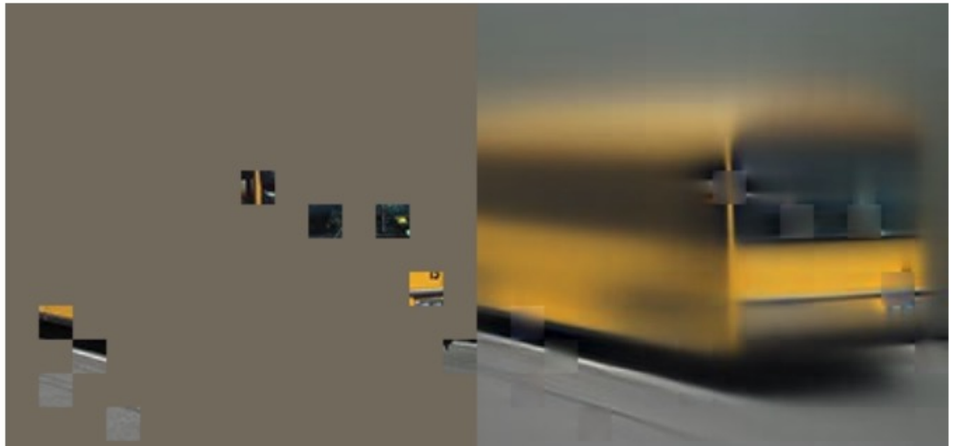
MAE Can Generalize

original

75% mask



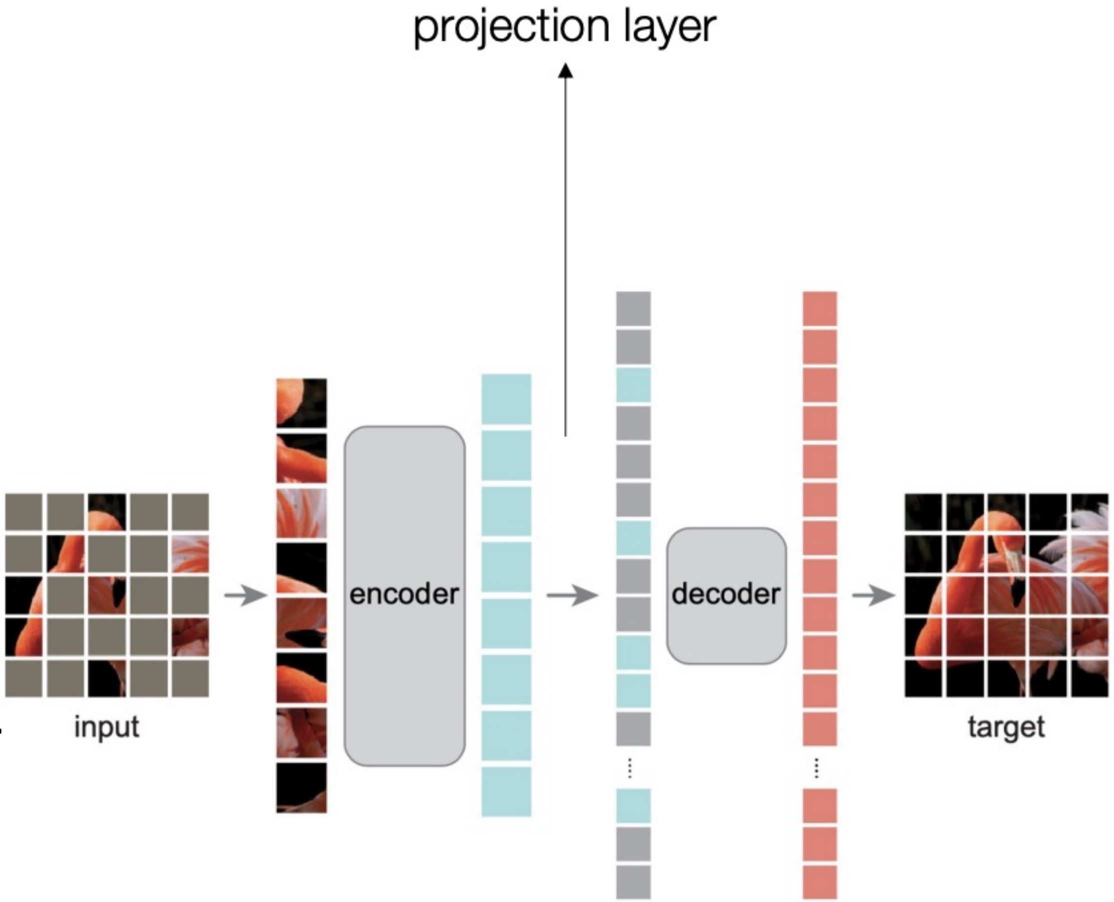
85% mask



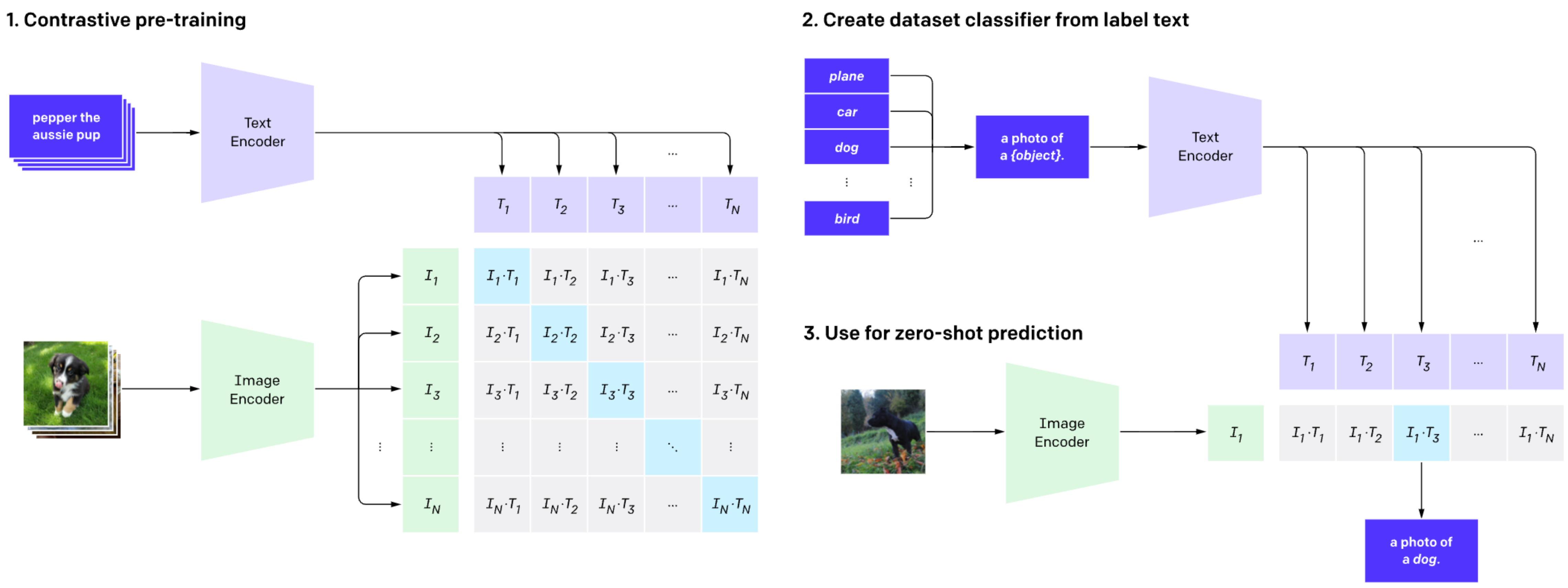
95% mask

MAE: More Take-Home Points

- BERT-like algorithm, **but with crucial design changes for vision**
  - BERT: 15% is enough
  - MAE: a **high ratio** of 75% - 80% is optimal
    - Very efficient when coupled with high mask ratio (75%)
  - MAR has large encoder on visible tokens
  - … + small decoder on all tokens
  - … + projection layer to connect the two
  - After pre-training, throw away the decoder
- Intriguing properties – better scalability
- work with minimal data augmentation



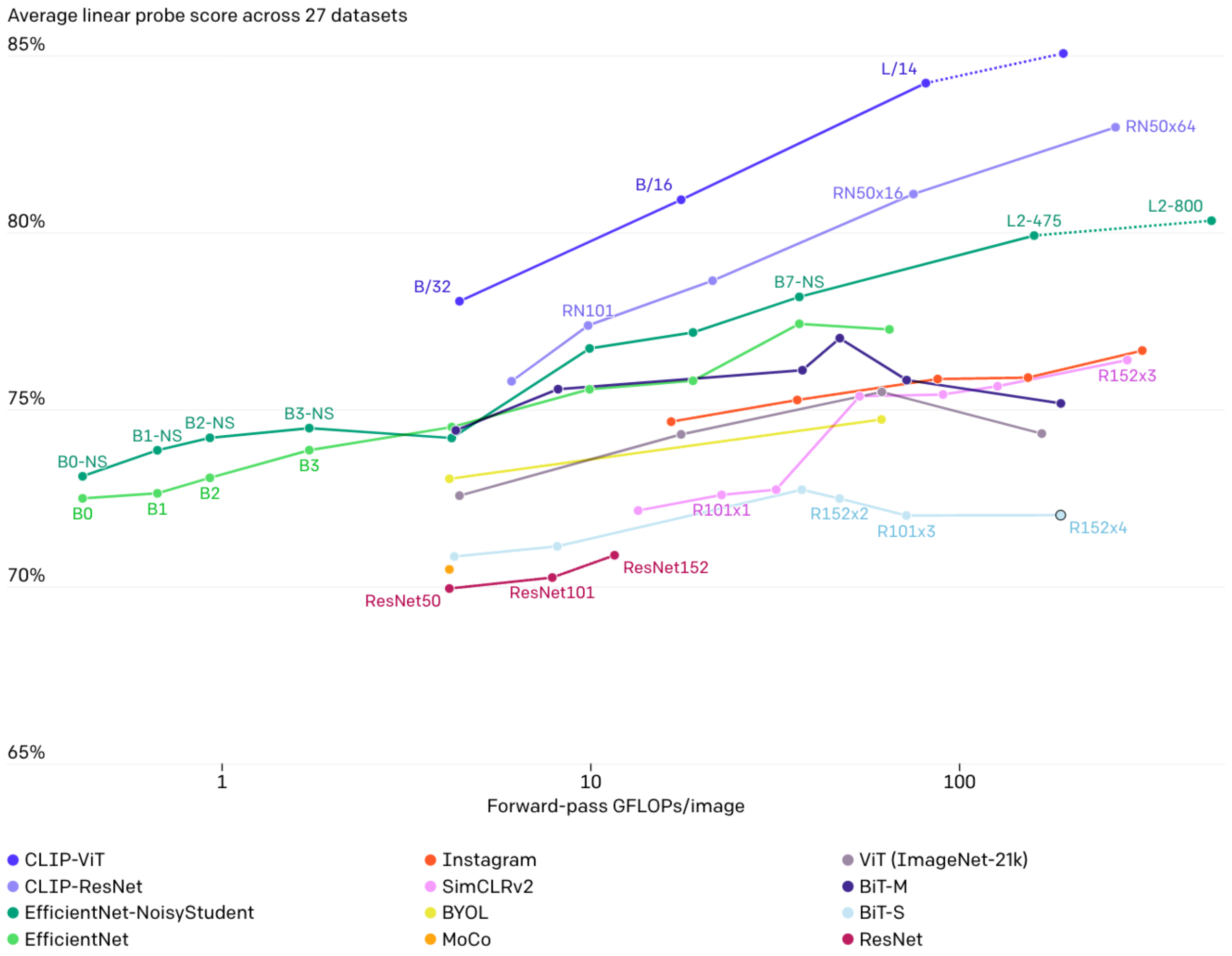
Contrastive Language-Image Pre-training (**CLIP**)



CLIP pre-trains an image encoder and a text encoder to predict which images were paired with which texts in our dataset. We then use this behavior to turn CLIP into a zero-shot classifier. We convert all of a dataset's classes into captions such as "a photo of a *dog*" and predict the class of the caption CLIP estimates best pairs with a given image.

https://openai.com/blog/clip/

CLIP is highly data-efficient, flexible and general



Across a suite of 27 datasets measuring tasks such as fine-grained object classification, OCR, activity recognition in videos, and geo-localization, we find that CLIP models learn more widely useful image representations. CLIP models are also more compute efficient than the models from 10 prior approaches that we compare with.

Some Limitations:

- struggles on abstract or systematic tasks
- struggles on very fine-grained classification
- sometimes sensitive to wording/phrasing, needing "prompt engineering"

https://openai.com/blog/clip/

General Message about Self-Supervised Learning

- MAE has won most CV downstream tasks (from 2D to 3D, sparse to dense)
- MoCo/SimCLR still own more competitive performance in the few-shot regime
- Maybe we should "hybrid"?
- Lots of open problems remain when/why an SSL representation works or not

The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering