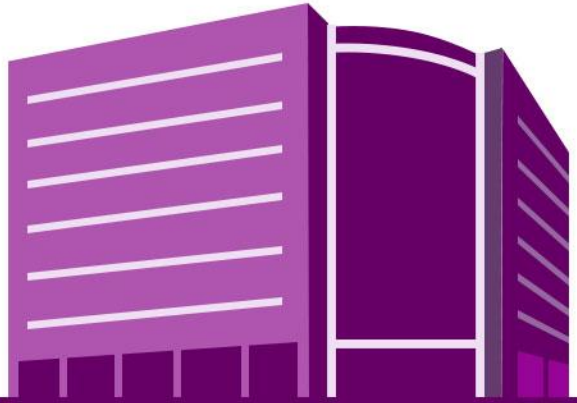
清華大学电子工程系
Department of Electronic Engineering, Tsinghua University

严谨 / 勤奋 / 求实 / 创新

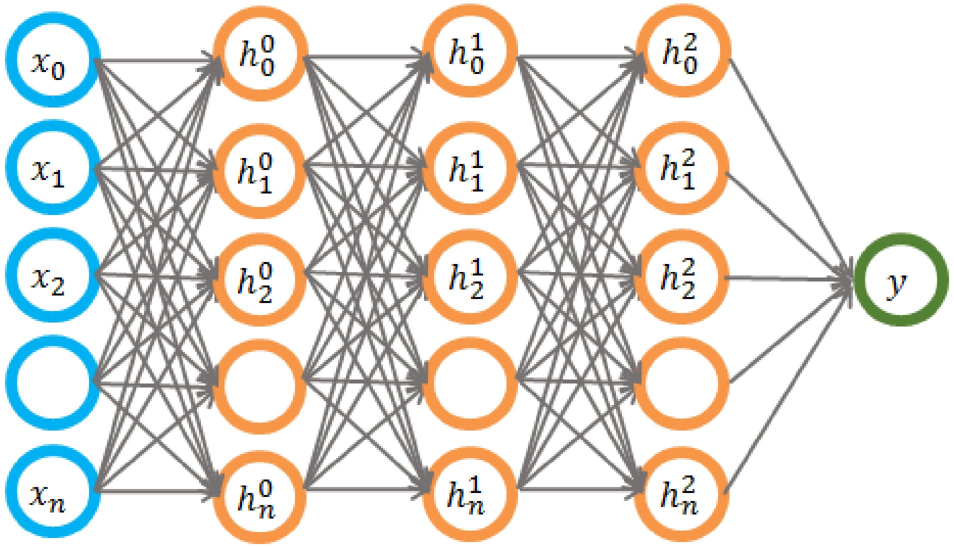
深度学习初始化与泛化性问题

清华大学电子工程系
段岳圻
duanyueqi@tsinghua.edu.cn



Deep Neural Networks

- Fully-connected NN
- Convolutional NN
- Recurrent NN
- Transformer networks



- Optimization
- Model initialization
- Generalization

Initialization

Parameter Initialization

- Convergence or not are sensitive to initial points
  - with some initial points being so unstable that the algorithm encounters numerical difficulties and fails altogether.
- Convergence speed and final training error depend on the initial point
- Initial points can affect the generalization as well.
  - Points of comparable cost can have wildly varying generalization error
- (Too) small initial weights lead to information loss in forward/back-propagation through the linear component of each layer
- Initial weights that are too large may, however, result in exploding values during forward propagation or back-propagation.

Basic Idea for Weight Initialization

- Variance changes across layers
  - $y = \sum_{i=1}^{n} w_i x_i$
  - $Var(y) = n\,Var(w)Var(x_i)$

- In order to remove this change, we need to ensure
  - $Var(w) \sim \frac{1}{n}$

- Random initialization
  - Gaussian or uniform distributions: $N\left(0, \frac{1}{n}\right)$ or $U\left(-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right)$

Commonly-used Initializations

- Xavier initialization
  - Consider different number of nodes in each layer
  - Consider the properties of sigmoid & tanh activation functions

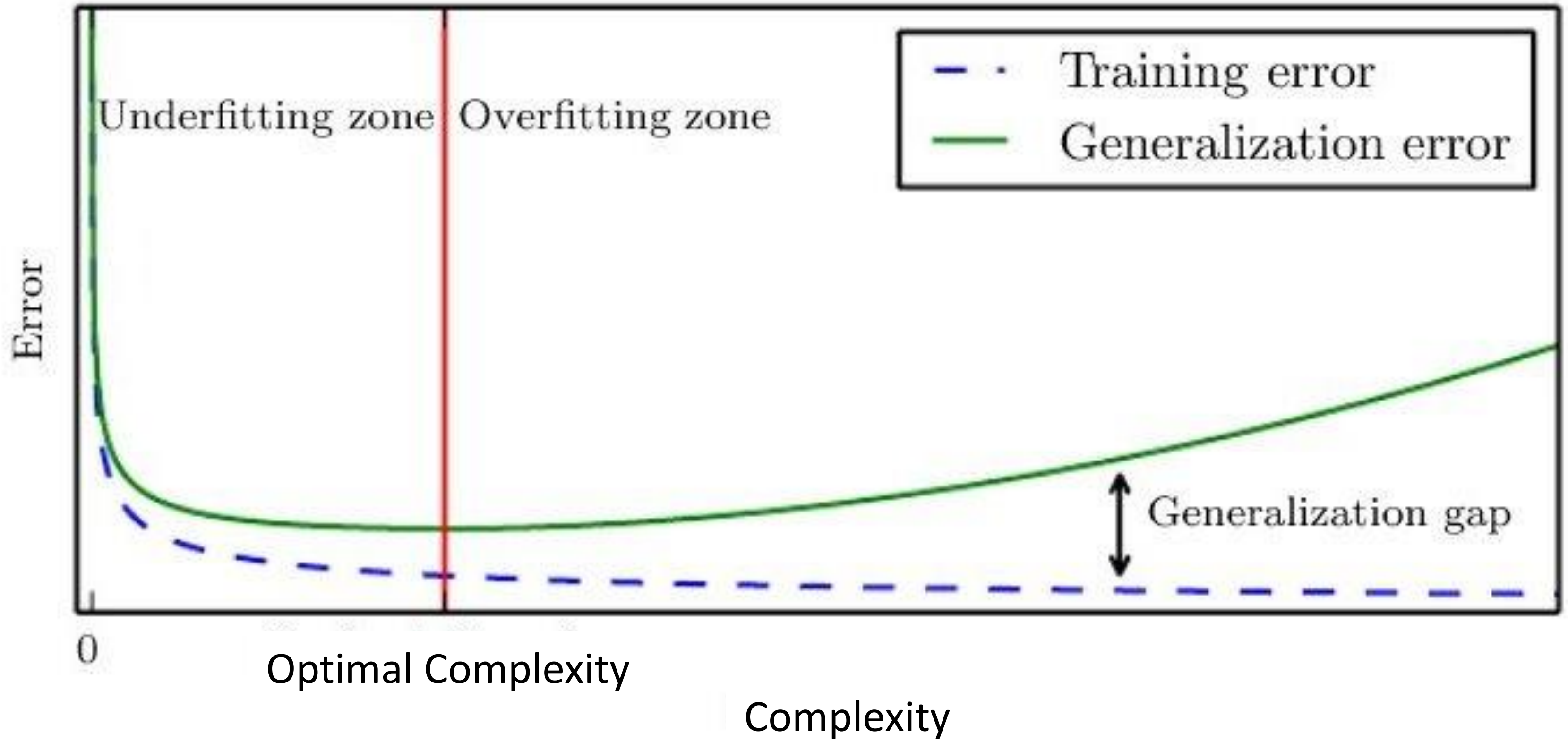
$$W \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_i+n_{i+1}}}, \frac{\sqrt{6}}{\sqrt{n_i+n_{i+1}}}\right]$$

- He initialization
  - Consider the properties of ReLU activation function

$$W \sim N(0, \frac{2}{n})$$

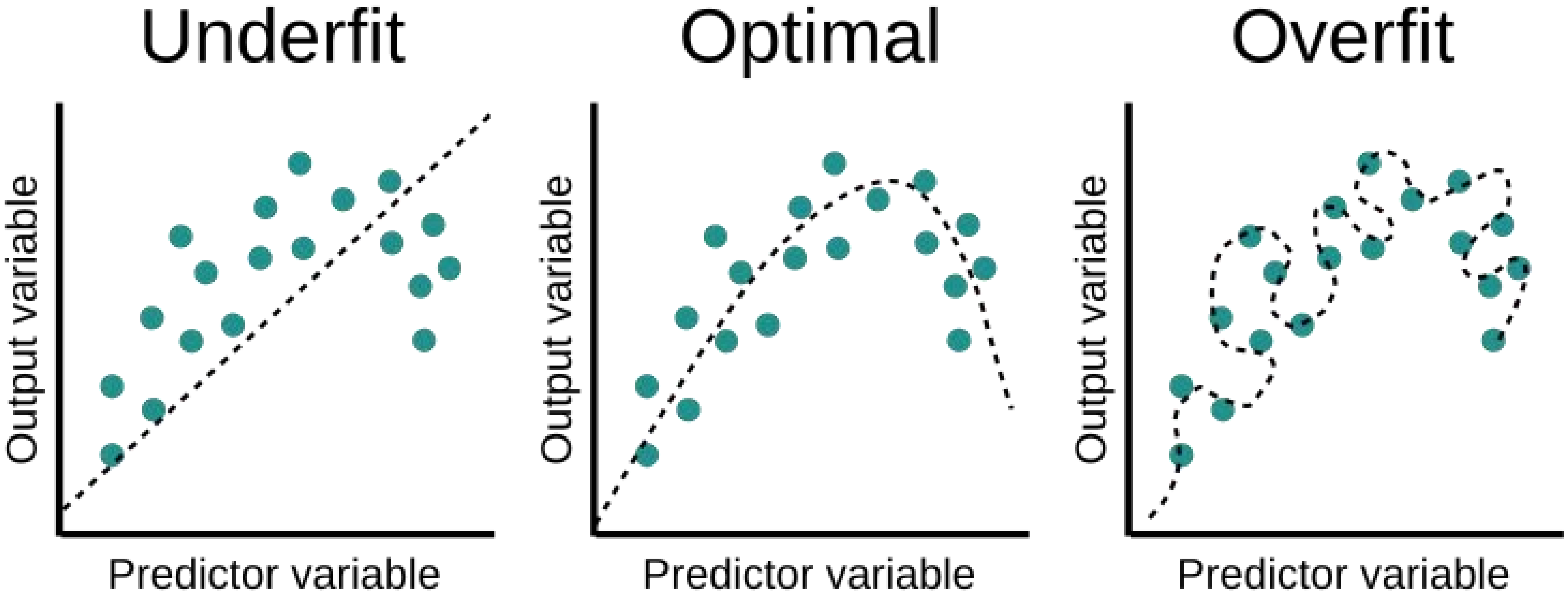
Generalization

Overfitting w.r.t Model Complexity



http://www.deeplearningbook.org/contents/ml.html

Overfitting w.r.t Model Complexity



Regularization

“Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.”

- Data
- DropOut
- Normalizations
- Weight decay
- Early stopping
- … …

Regularization

“Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.”

- Data
- DropOut
- Normalizations
- Weight decay
- Early stopping
- … …

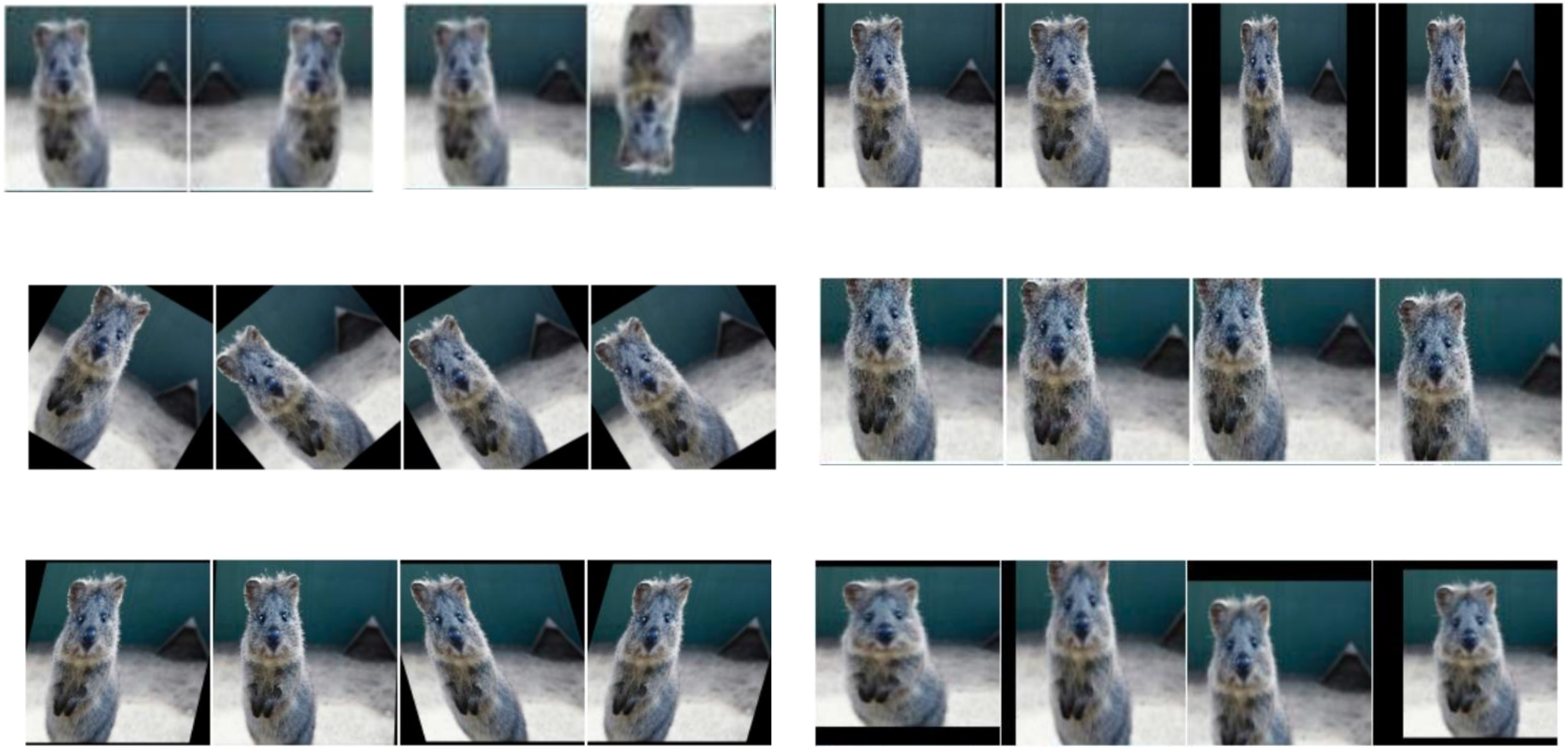
Data Augmentation

The best way to make a ML model generalize better is to train it on more data.

- Translating, rotation, scaling, randomly cropping
  - Very effective for object recognition
  - Be careful! ‘b’ v.s ‘d’, ‘6’ v.s ‘9’
- Noise injection
  - Inject into input data, e.g., adding Gaussian noise
  - Inject into hidden layers
  - Inject into output targets, label smoothing
- Leverage unlabeled data
  - Pretraining
  - Semi-supervised learning

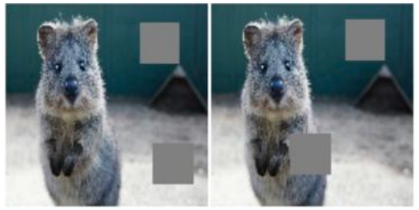
Data Augmentation with Single Data

- Geometric transformation



Data Augmentation with Single Data

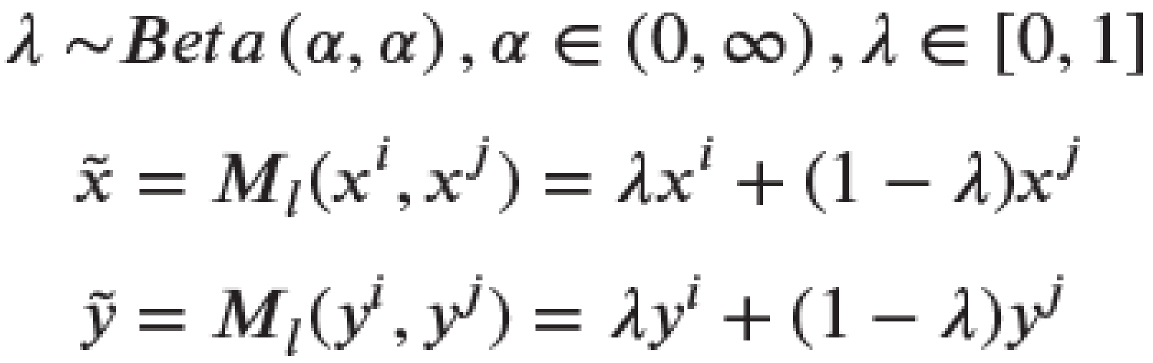
- Color transformation



Imgaug
地址：
https://github.com/aleju/imgaug

Data Augmentation with Multiple Data

- Mixup (linear interpolation)



$$\lambda \sim Beta(\alpha, \alpha), \alpha \in (0, \infty), \lambda \in [0, 1]$$

$$\tilde{x} = M_l(x^i, x^j) = \lambda x^i + (1 - \lambda)x^j$$

$$\tilde{y} = M_l(y^i, y^j) = \lambda y^i + (1 - \lambda)y^j$$

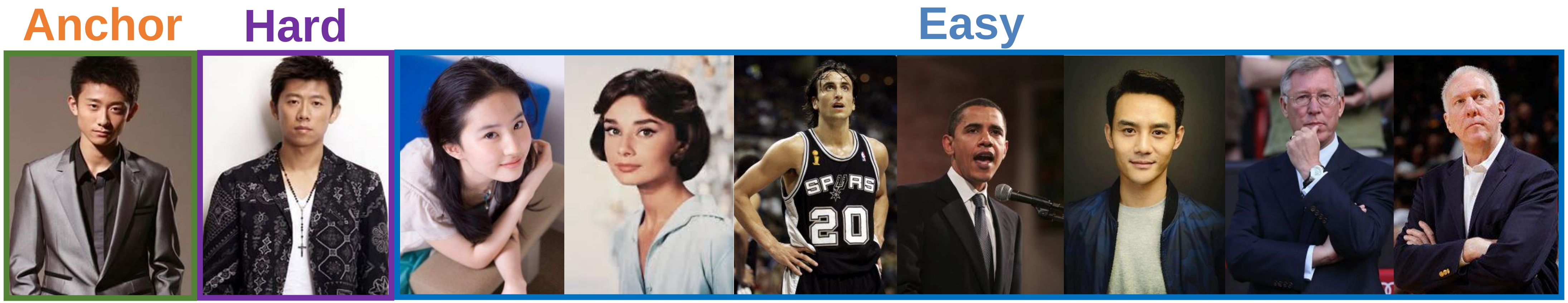
Better to use different labels

(xi,yi) and (xj,yj) are random samples from the training dataset

Zhang H, Cisse M, Dauphin Y N, et al. mixup: Beyond empirical risk minimization[J]. arXiv preprint arXiv:1710.09412, 2017. MIT

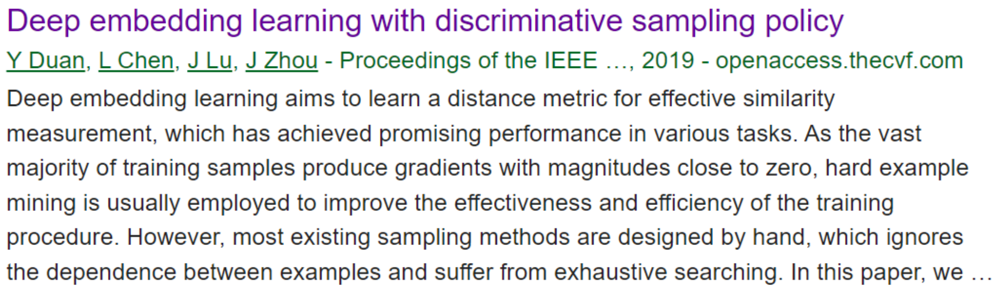
How to Select Training Samples?

- Hard negatives produce gradients with large magnitudes, while easy negatives are close to zero
- Hard negative mining for effective model training



Learning Discriminative Sampling Policy

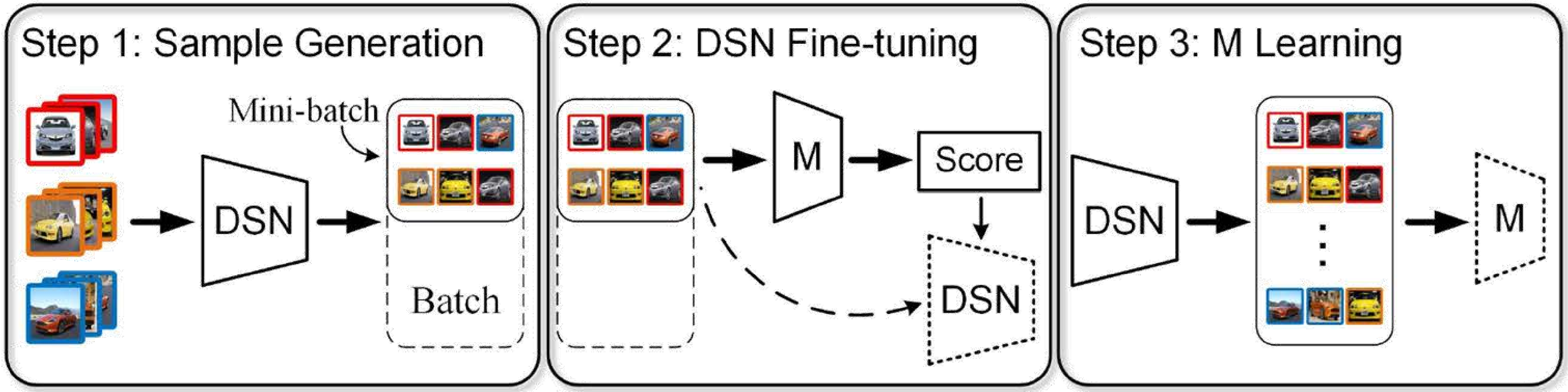
- Training a sampling network to select effective training data



Deep embedding learning with discriminative sampling policy

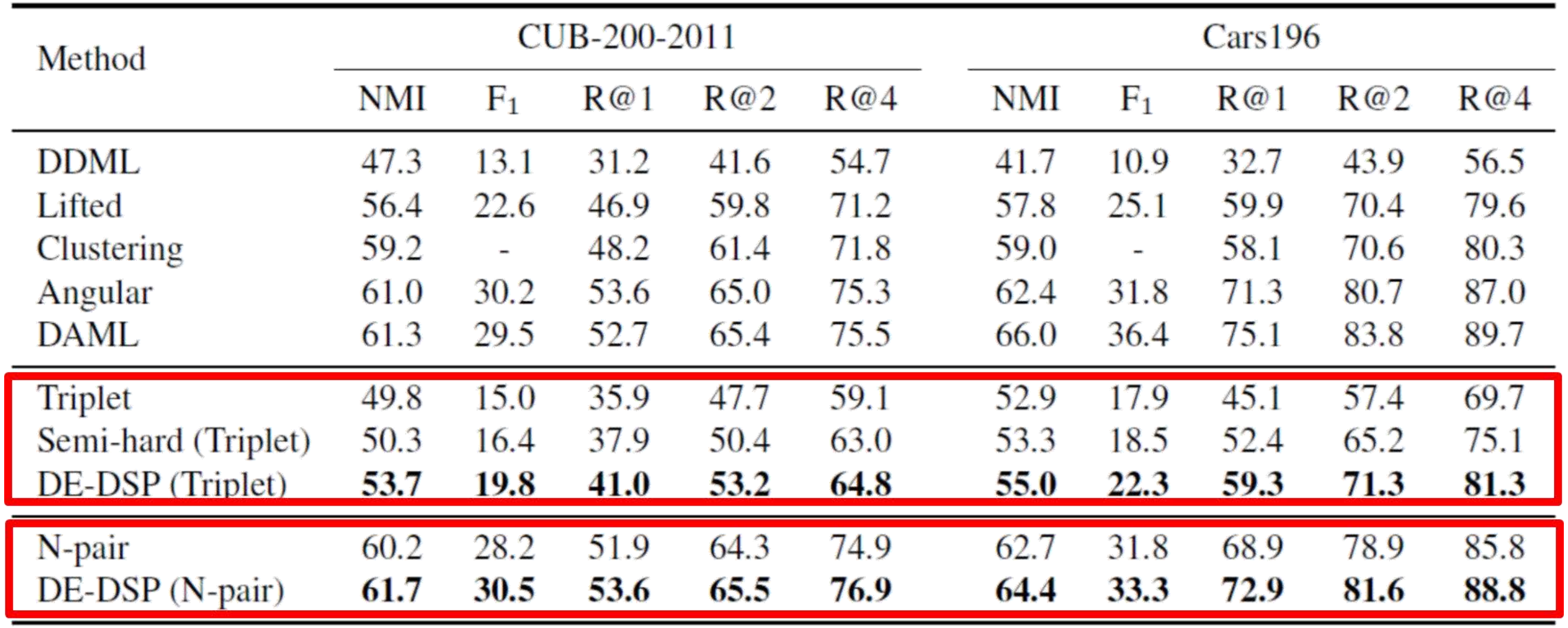
Y Duan, L Chen, J Lu, J Zhou - Proceedings of the IEEE …, 2019 - openaccess.thecvf.com

Deep embedding learning aims to learn a distance metric for effective similarity measurement, which has achieved promising performance in various tasks. As the vast majority of training samples produce gradients with magnitudes close to zero, hard example mining is usually employed to improve the effectiveness and efficiency of the training procedure. However, most existing sampling methods are designed by hand, which ignores the dependence between examples and suffer from exhaustive searching. In this paper, we …



Experiments

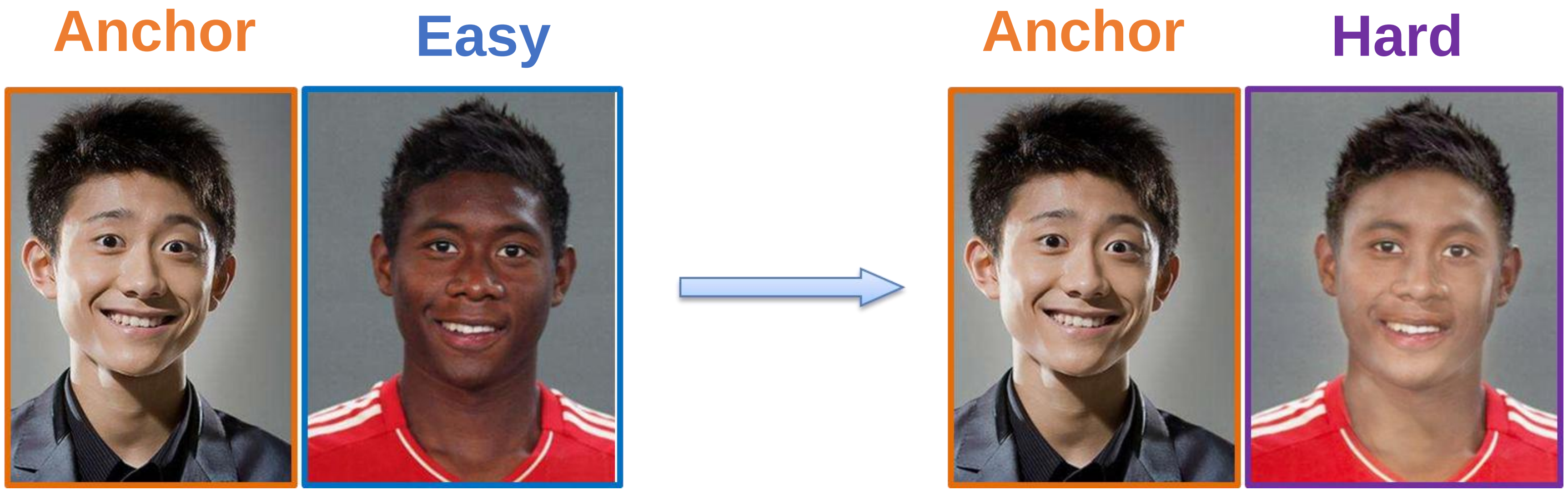
- Experiments on CUB-200-2011 and Cars196



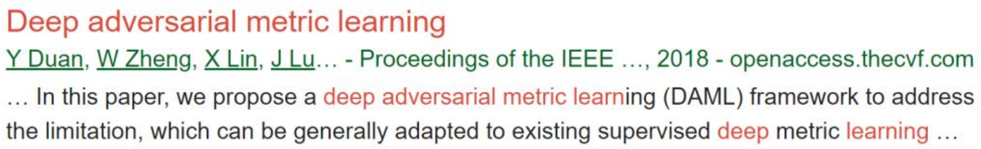
Method	CUB-200-2011					Cars196				
	NMI	F_1	R@1	R@2	R@4	NMI	F_1	R@1	R@2	R@4
DDML	47.3	13.1	31.2	41.6	54.7	41.7	10.9	32.7	43.9	56.5
Lifted	56.4	22.6	46.9	59.8	71.2	57.8	25.1	59.9	70.4	79.6
Clustering	59.2	-	48.2	61.4	71.8	59.0	-	58.1	70.6	80.3
Angular	61.0	30.2	53.6	65.0	75.3	62.4	31.8	71.3	80.7	87.0
DAML	61.3	29.5	52.7	65.4	75.5	66.0	36.4	75.1	83.8	89.7
Triplet	49.8	15.0	35.9	47.7	59.1	52.9	17.9	45.1	57.4	69.7
Semi-hard (Triplet)	50.3	16.4	37.9	50.4	63.0	53.3	18.5	52.4	65.2	75.1
DE-DSP (Triplet)	**53.7**	**19.8**	**41.0**	**53.2**	**64.8**	**55.0**	**22.3**	**59.3**	**71.3**	**81.3**
N-pair	60.2	28.2	51.9	64.3	74.9	62.7	31.8	68.9	78.9	85.8
DE-DSP (N-pair)	**61.7**	**30.5**	**53.6**	**65.5**	**76.9**	**64.4**	**33.3**	**72.9**	**81.6**	**88.8**

Not Enough Hard Samples?

- Potential to generate hard samples from easy samples



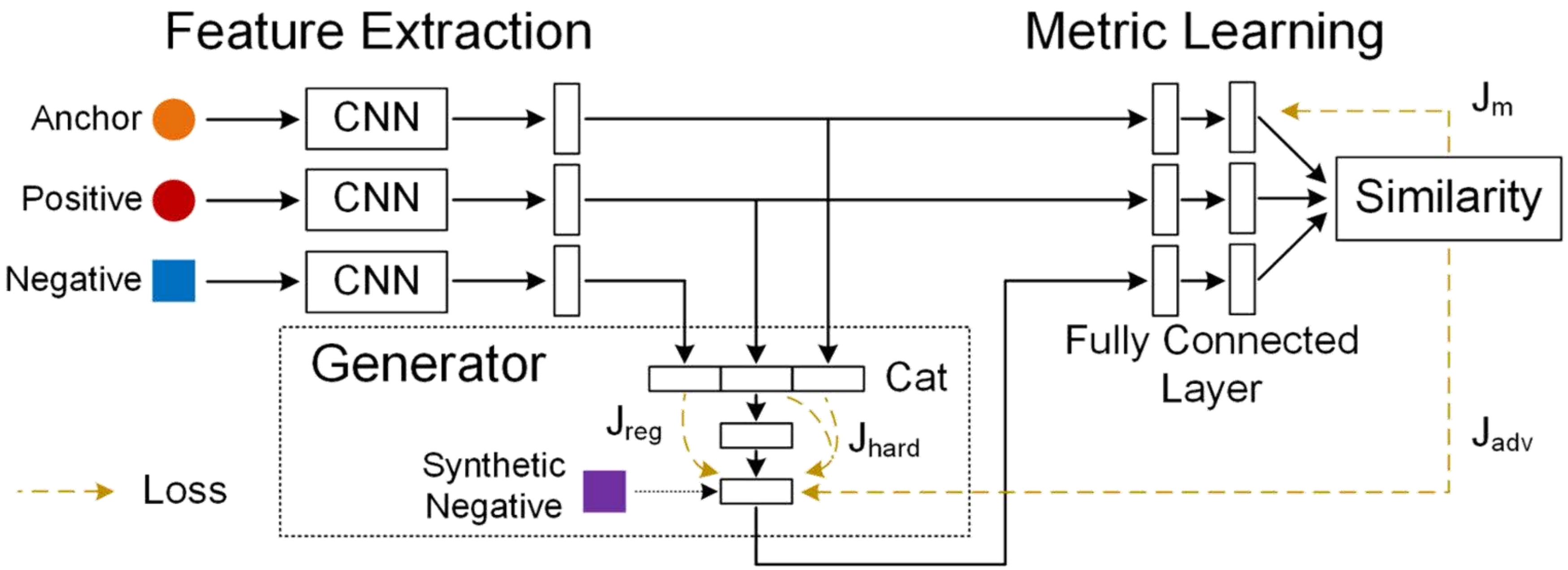
Training with Synthetic Data



Deep adversarial metric learning

Y Duan, W Zheng, X Lin, J Lu… - Proceedings of the IEEE …, 2018 - openaccess.thecvf.com

… In this paper, we propose a deep adversarial metric learning (DAML) framework to address the limitation, which can be generally adapted to existing supervised deep metric learning …

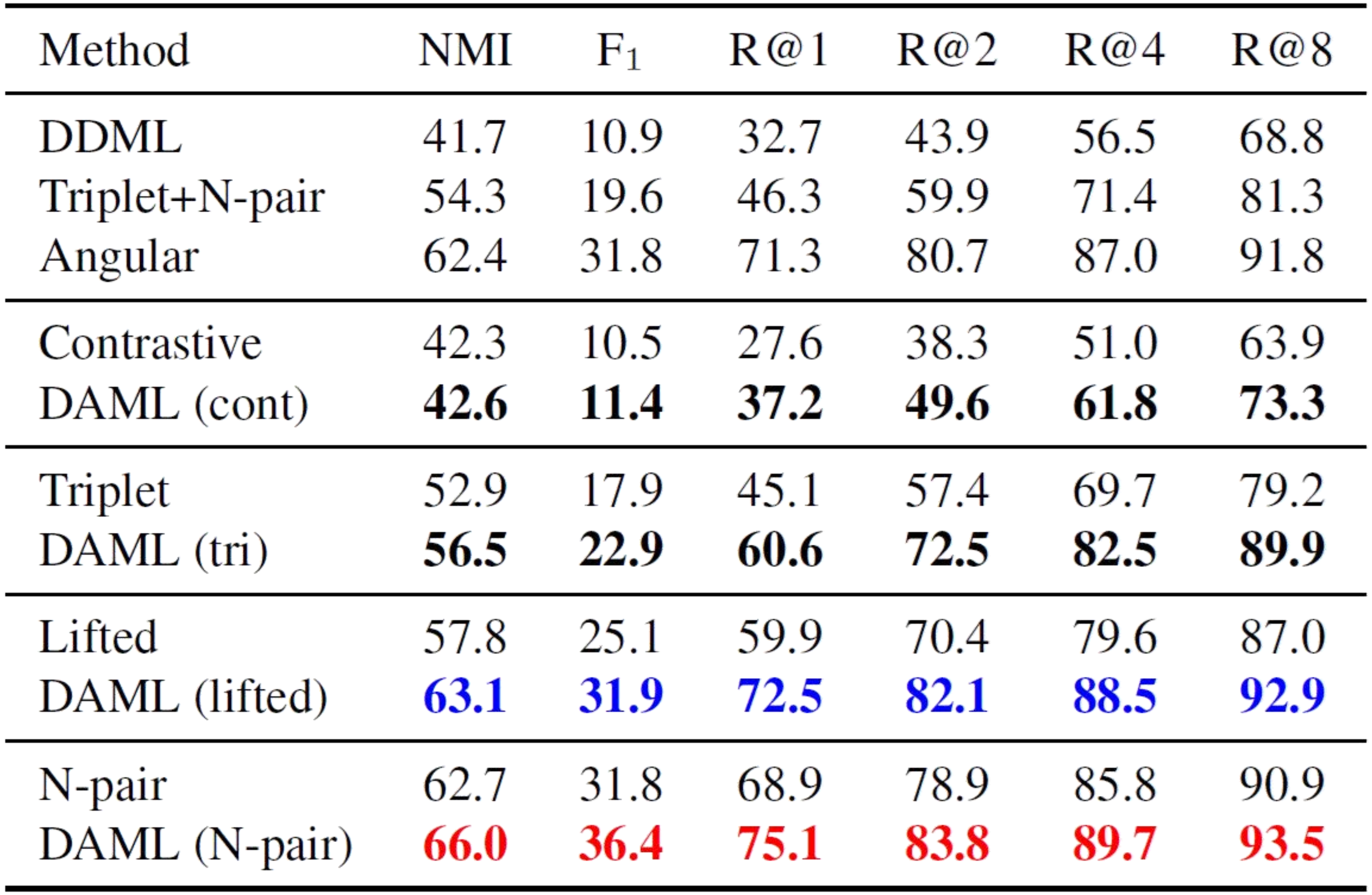


1. Learn to generate synthetic data

2. Adaptive to training process

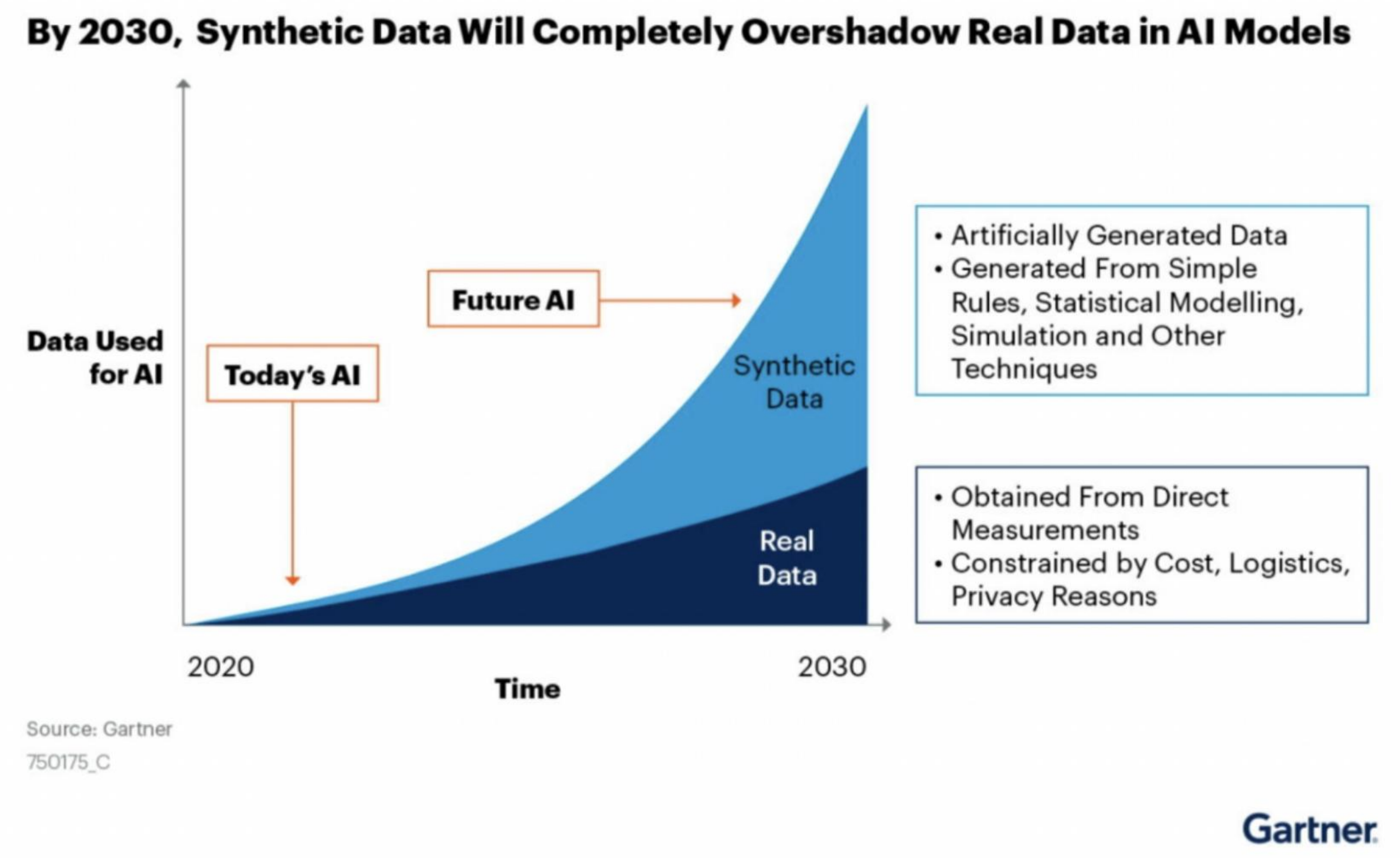
Experiments

- Experiments on Cars196



Method	NMI	F_1	R@1	R@2	R@4	R@8
DDML	41.7	10.9	32.7	43.9	56.5	68.8
Triplet+N-pair	54.3	19.6	46.3	59.9	71.4	81.3
Angular	62.4	31.8	71.3	80.7	87.0	91.8
Contrastive	42.3	10.5	27.6	38.3	51.0	63.9
DAML (cont)	**42.6**	**11.4**	**37.2**	**49.6**	**61.8**	**73.3**
Triplet	52.9	17.9	45.1	57.4	69.7	79.2
DAML (tri)	**56.5**	**22.9**	**60.6**	**72.5**	**82.5**	**89.9**
Lifted	57.8	25.1	59.9	70.4	79.6	87.0
DAML (lifted)	**63.1**	**31.9**	**72.5**	**82.1**	**88.5**	**92.9**
N-pair	62.7	31.8	68.9	78.9	85.8	90.9
DAML (N-pair)	**66.0**	**36.4**	**75.1**	**83.8**	**89.7**	**93.5**

Training with Synthetic Data



By 2030, Synthetic Data Will Completely Overshadow Real Data in AI Models

- Artificially Generated Data
- Generated From Simple Rules, Statistical Modelling, Simulation and Other Techniques

- Obtained From Direct Measurements
- Constrained by Cost, Logistics, Privacy Reasons

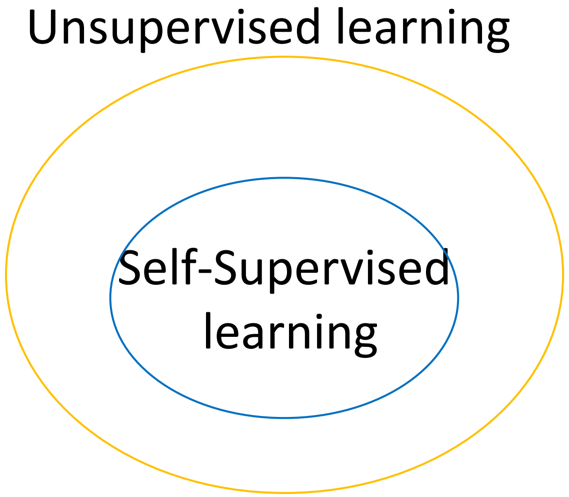
Source: Gartner
750175_C

Gartner: “Maverick Research: Forget about Your Real Data – Synthetic Data Is the Future of AI”, 24 June 2021.

Our latest work: https://liuff19.github.io/Make-Your-3D

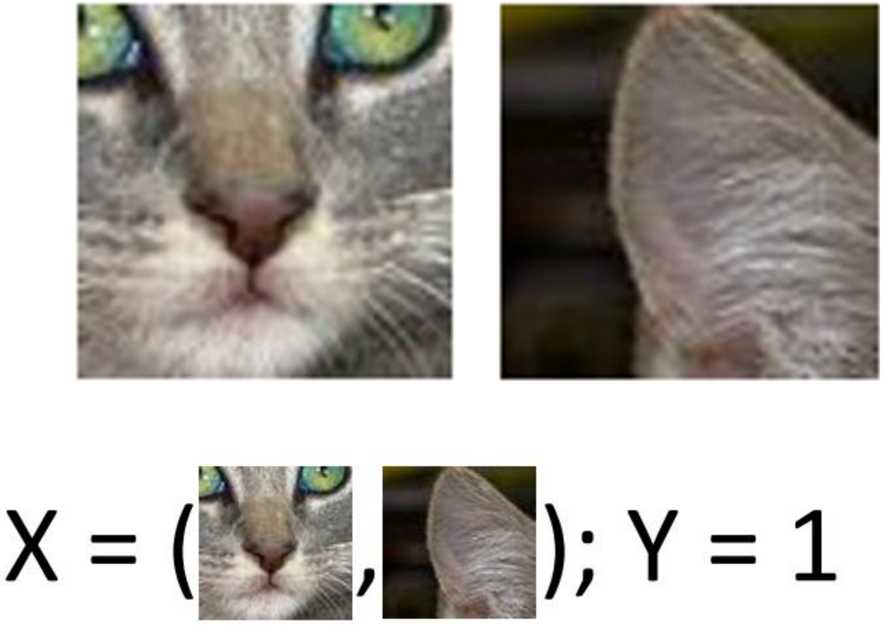
Unsupervised Learning

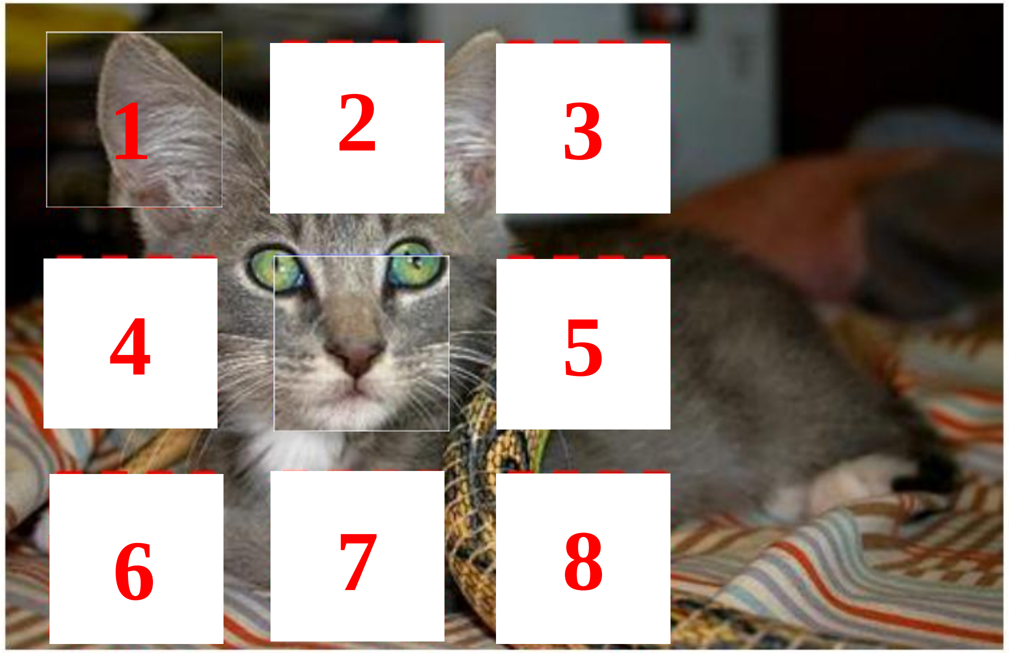
- Unsupervised learning: Training with unannotated data
- Self-supervised learning
  - Designing pretext task
  - Generating pseudo labels from data itself
  - Testing with/without finetuning



Jigsaw

- Pretext task: Estimate the relative position of patches

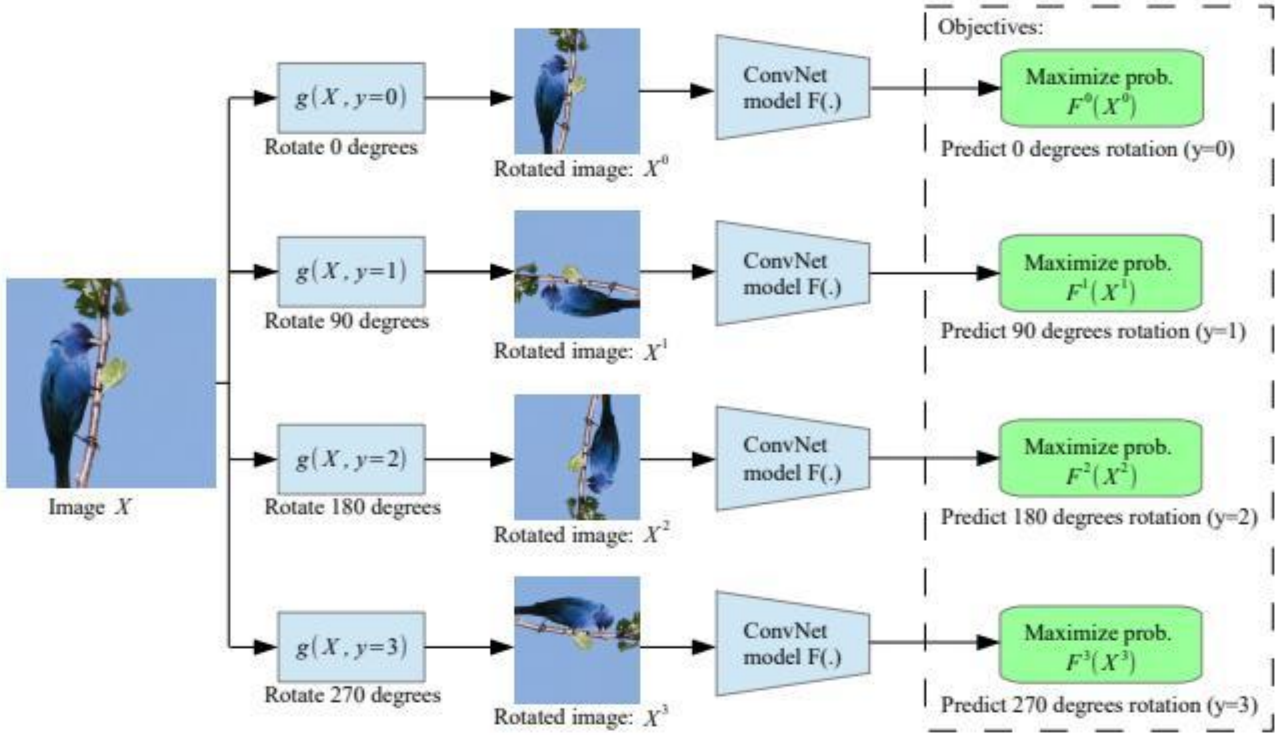




Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised Visual Representation Learning by Context Prediction. In *ICCV 2015*.

Rotation

- Pretext task: Estimate the angles

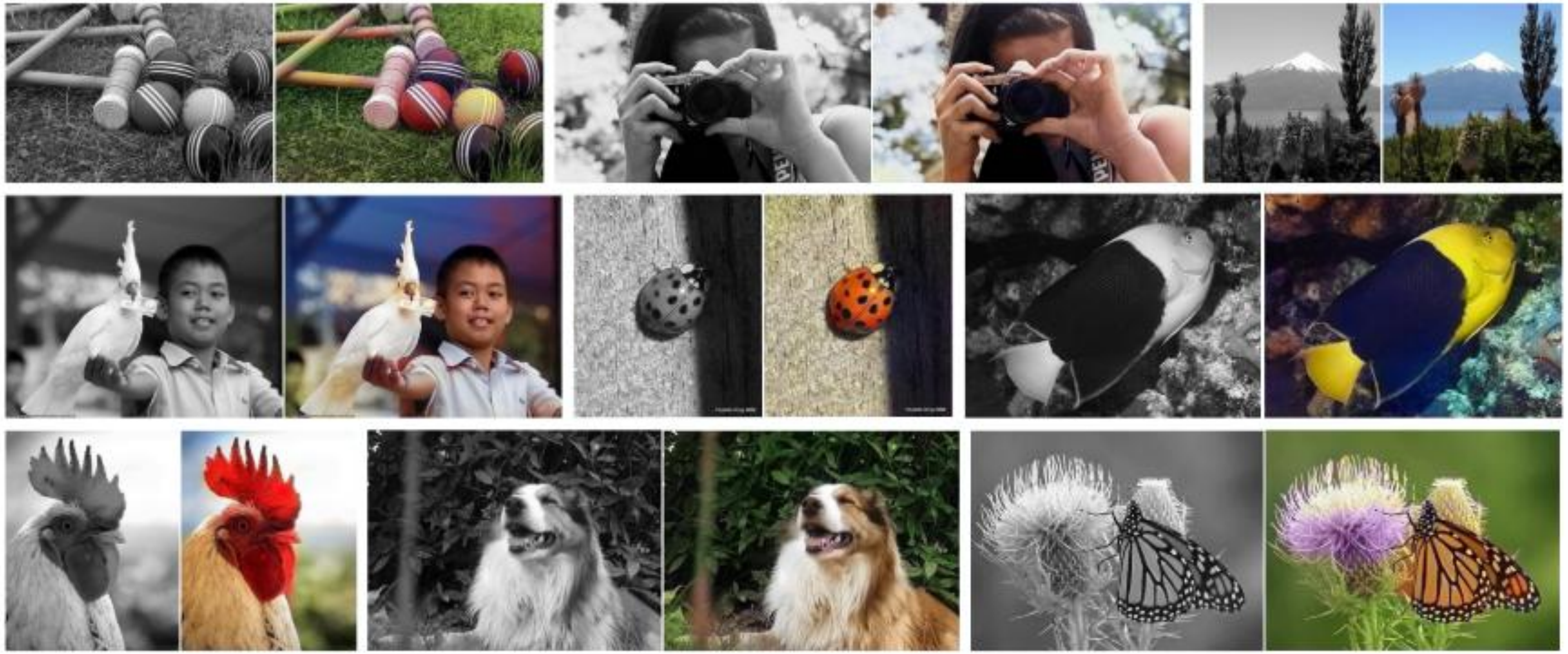


Gidaris, Spyros, Praveer Singh, and Nikos Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations." International Conference on Learning Representations. 2018.

Code: https://github.com/gidariss/FeatureLearningRotNet

Colorization

- Pretext task: Changing the images into gray ones, and estimate the color of the images



Zhang, Richard, Phillip Isola, and Alexei A. Efros. "Colorful image colorization." European conference on computer vision. Springer, Cham, 2016.

Summary of Self-supervised Learning

- Pros
  - Design pretext tasks to learn effective representations from unlabeled data
- Cons
  - Pretext tasks need human’s prior knowledge to design
  - Gap between pretext task and downstream task (classification, segmentation, detection…)
  - Learn unnecessary redundant information (e.g. colorization)

Contrastive Learning

1
ONE
DOLLAR
In God
we trust

Based on memories

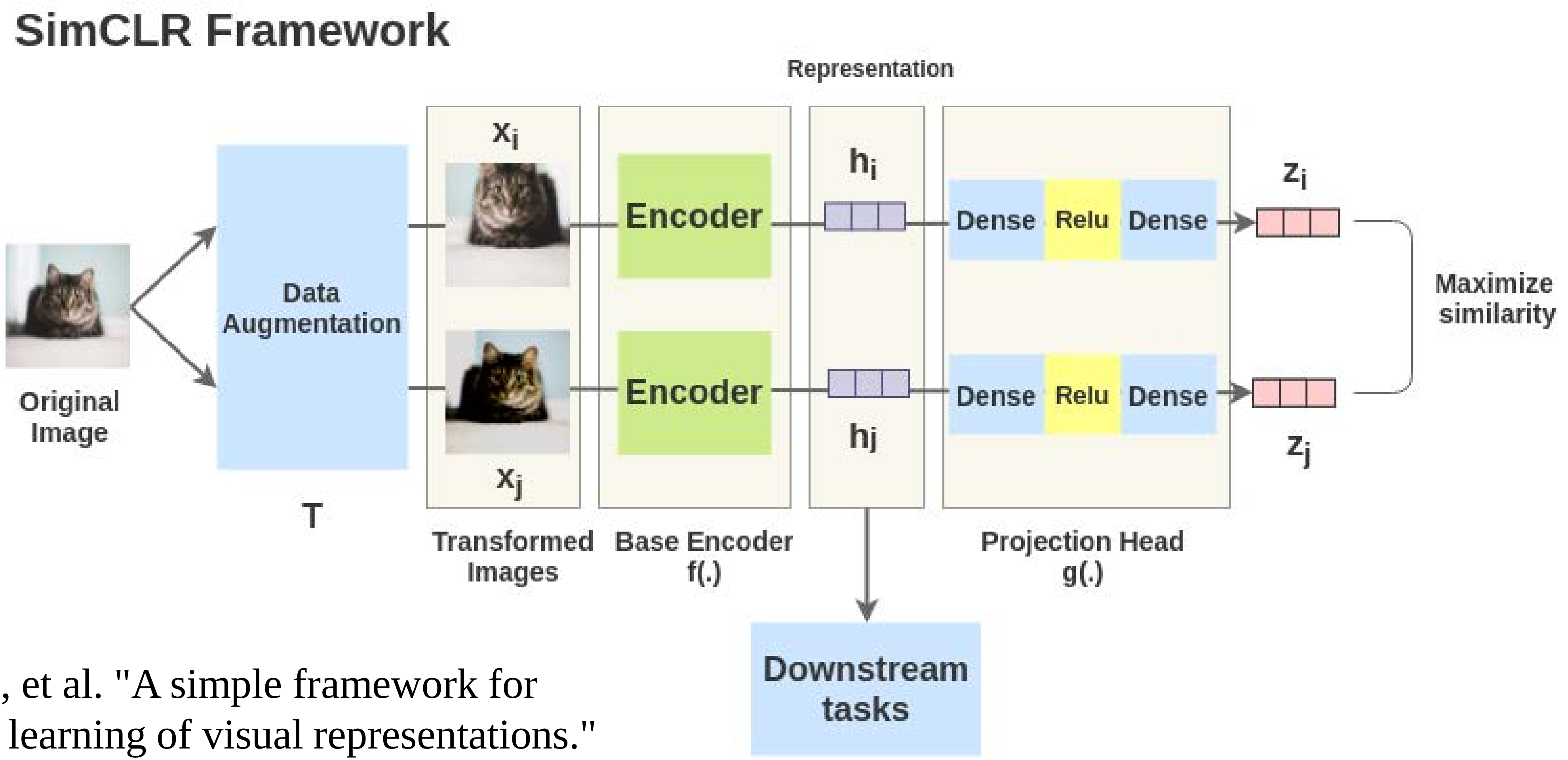
FEDERAL RESERVE NOTE
THE UNITED STATES OF AMERICA
THIS NOTE IS LEGAL TENDER FOR ALL DEBTS, PUBLIC AND PRIVATE
L54836042 D
WASHINGTON, D.C.
ONE
12
L
L54836042D
SERIES 2013
ONE DOLLAR
1

Based on comparisons

- Learning from positive and negative comparisons
- How to build comparisons without labels?

$$\underline{\text{score}(f(x), f(x^+))} >> \underline{\text{score}(f(x), f(x^-))}$$

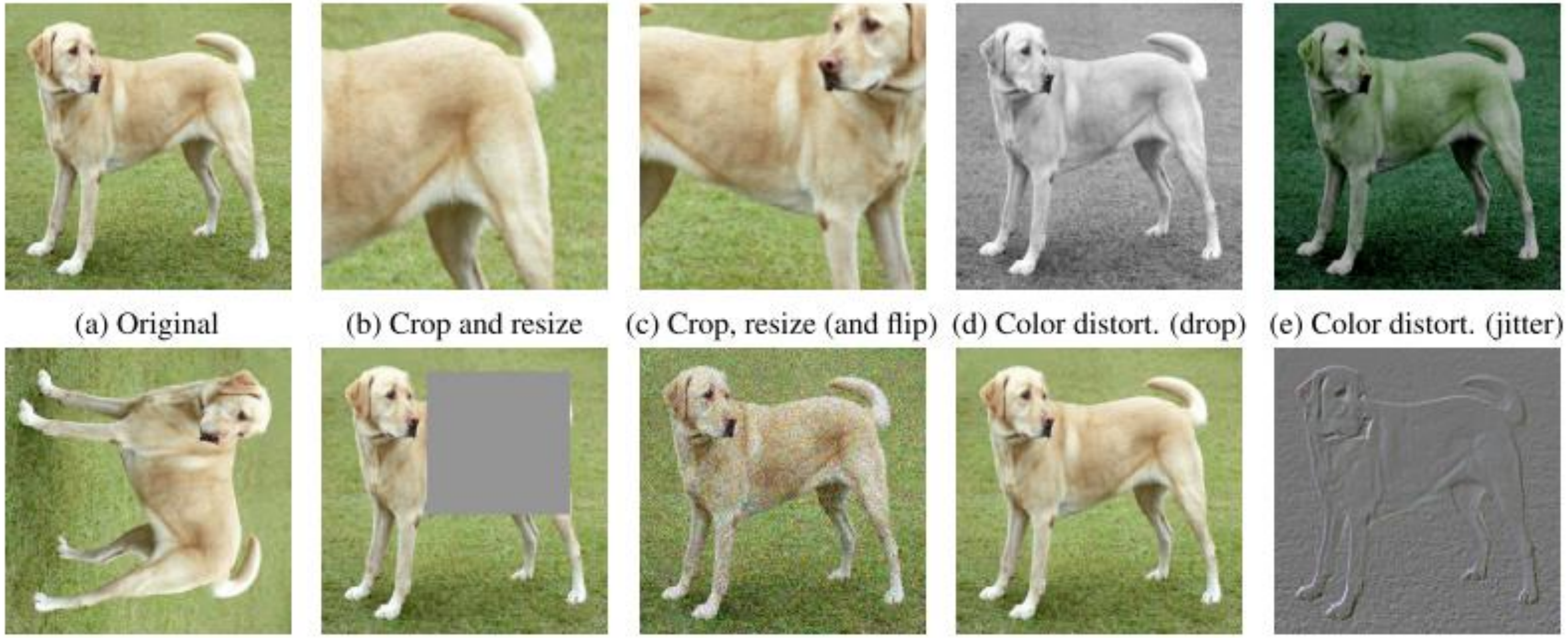
SimCLR



Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." International conference on machine learning. PMLR, 2020.

SimCLR

- How to construct positive and negative pairs?
    - Combination leads to better performance



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate {90°, 180°, 270°} (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

SimCLR

- Loss function

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

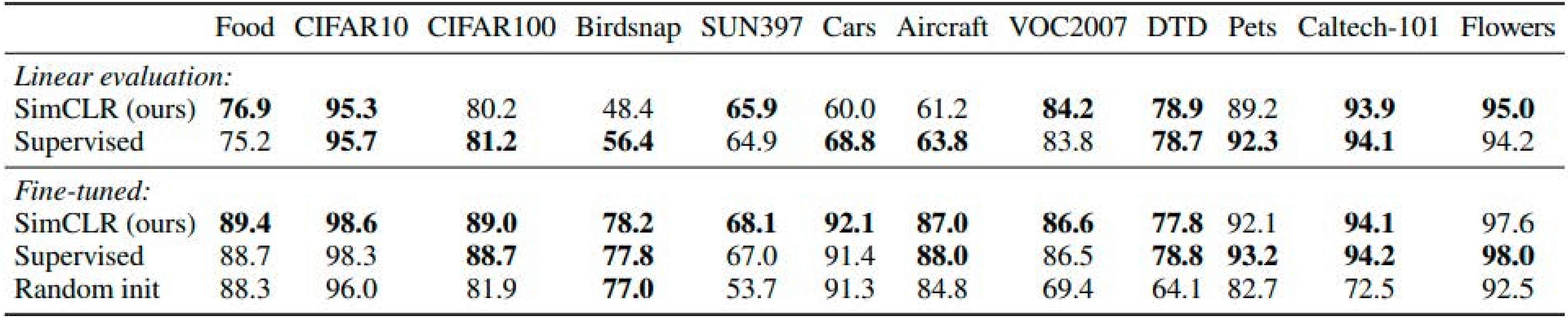
- Cosine similarity

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

- Temperature τ

Experiments

- Fine-grained classification: Birdsnap Cars Aircrafts Flowers
- Classification: Food CIFAR10 CIFAR100 Caltech-101
- Scene understanding: SUN397
- Texture: DTD; Object detection: VOC2007; Segmentation: PET



	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
SimCLR (ours)	**76.9**	**95.3**	80.2	48.4	**65.9**	60.0	61.2	**84.2**	**78.9**	89.2	**93.9**	**95.0**
Supervised	75.2	**95.7**	**81.2**	**56.4**	64.9	**68.8**	**63.8**	83.8	**78.7**	**92.3**	**94.1**	94.2
Fine-tuned:												
SimCLR (ours)	**89.4**	**98.6**	**89.0**	**78.2**	**68.1**	**92.1**	**87.0**	**86.6**	**77.8**	92.1	**94.1**	97.6
Supervised	88.7	98.3	**88.7**	**77.8**	67.0	91.4	**88.0**	86.5	**78.8**	**93.2**	**94.2**	**98.0**
Random init	88.3	96.0	81.9	**77.0**	53.7	91.3	84.8	69.4	64.1	82.7	72.5	92.5

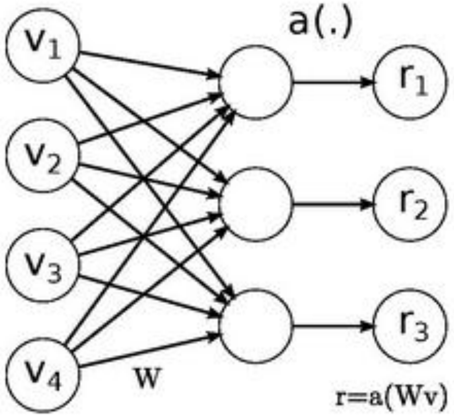
Regularization

"Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error."

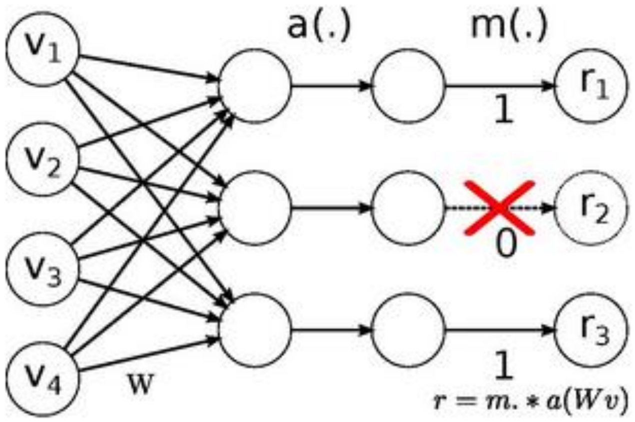
- Data
- DropOut
- Normalizations
- Weight decay
- Early stopping
- … …

DropOut/DropConnect

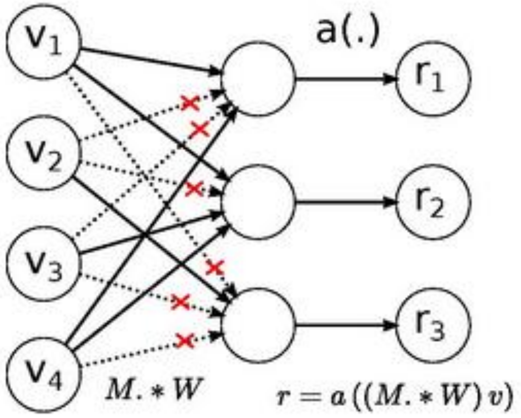
- Introduce noises to the training process
  - During training, multiply neuron output (or weight) by a random bit with probability of $(1-p)$
  - During test, multiply neuron out (or weight) by the expectation $(1-p)$



Original network



DropOut network



DropConnect network

Normalizations

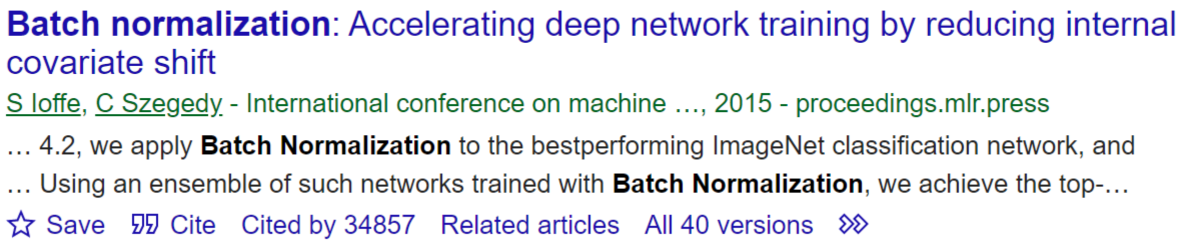
- Why normalization?
  - Speed up convergence: Normalization is a way to regulate the distribution of hidden nodes
    - The distribution of each layer’s inputs changes during training as the parameters of the previous layers change.
    - This will slow down the training by requiring lower learning rates and careful parameter initialization and makes it notoriously hard to train models with saturating nonlinearities.
  - Improve generalization: Normalization is also a way to restrict the possible value that hidden node or weight could take
    - Restricted functional class will lead to better generalization

Normalizations

$$h = \gamma \cdot \frac{x - \mu}{\sigma} + \beta$$

- Batch norm:
  - Vertical normalization of data (one dimension, multiple instances)
- Layer norm:
  - Horizontal normalization of data (one instance, multiple dimensions)
- Weight norm:
  - Normalization of weights

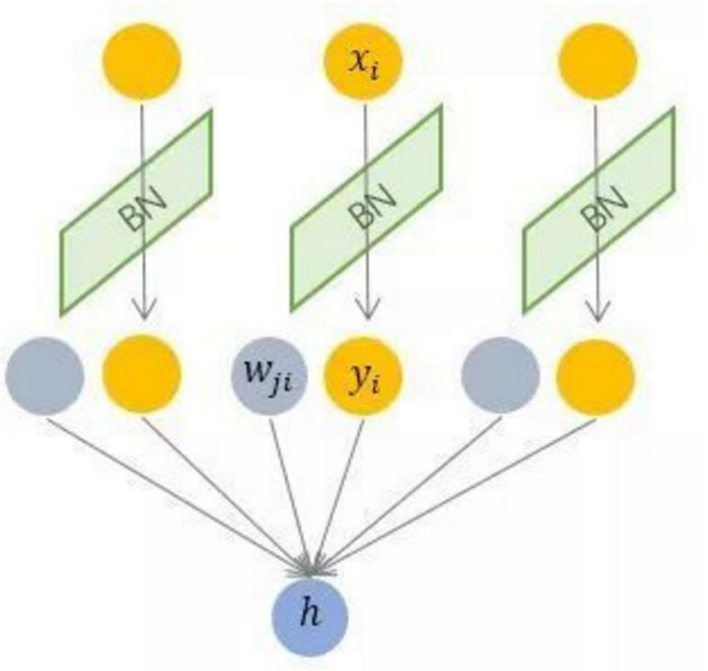
Batch Normalization



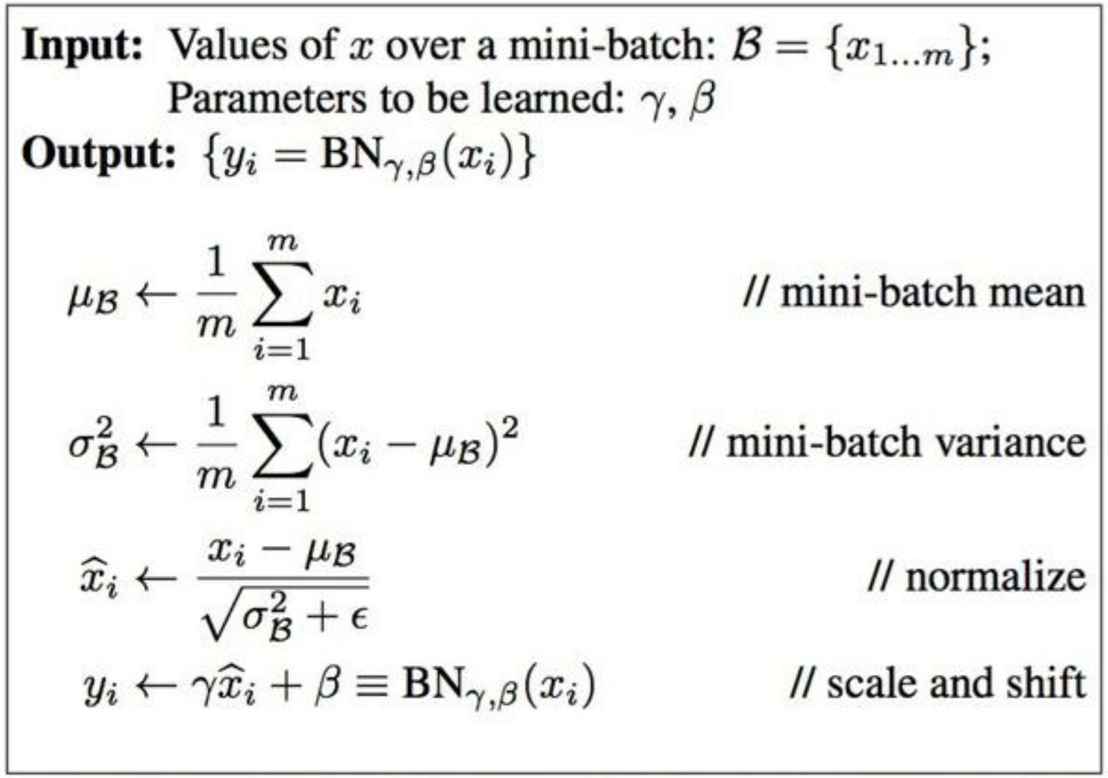
Batch normalization: Accelerating deep network training by reducing internal covariate shift

S Ioffe, C Szegedy - International conference on machine …, 2015 - proceedings.mlr.press

… 4.2, we apply **Batch Normalization** to the bestperforming ImageNet classification network, and … Using an ensemble of such networks trained with **Batch Normalization**, we achieve the top-…



Standardize the activations before nonlinear transformation inside a mini-batch



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

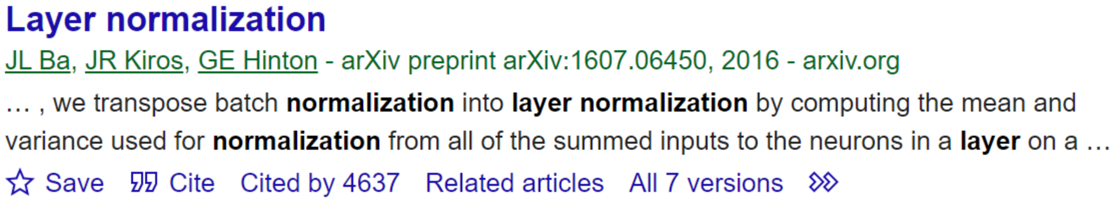
$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Training time: use mini-batch for normalization;
Test time: use global mean and standard deviation for normalization

Layer Normalization



Layer normalization

JL Ba, JR Kiros, GE Hinton - arXiv preprint arXiv:1607.06450, 2016 - arxiv.org

… , we transpose batch **normalization** into **layer normalization** by computing the mean and variance used for **normalization** from all of the summed inputs to the neurons in a **layer** on a …

- Normalization with respect to one single instance
- Normalization across the output of different nodes (dimensions) in the same layer
- May potentially restrict the expressiveness power of the neural networks model

$$\mu_i = \frac{1}{m}\sum_{j=1}^{m} x_{ij}$$

→ mean

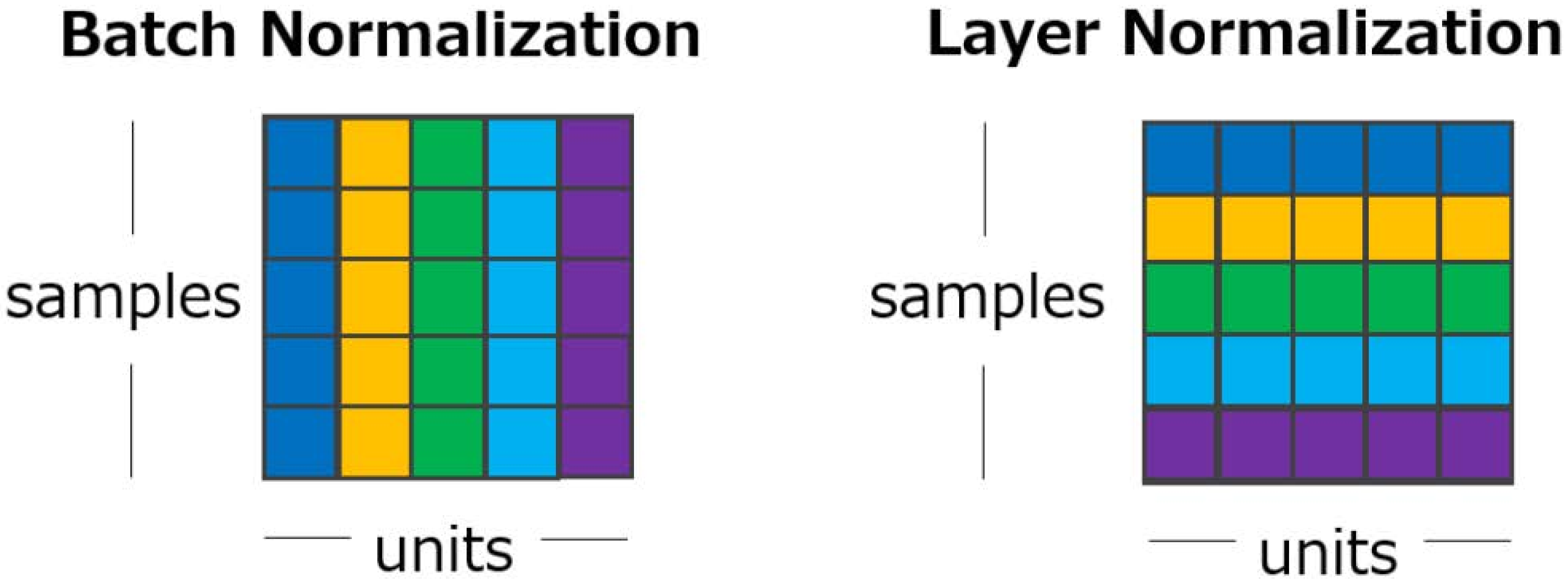
$$\sigma_i^2 = \frac{1}{m}\sum_{j=1}^{m} (x_{ij} - \mu_i)^2$$

→ variance

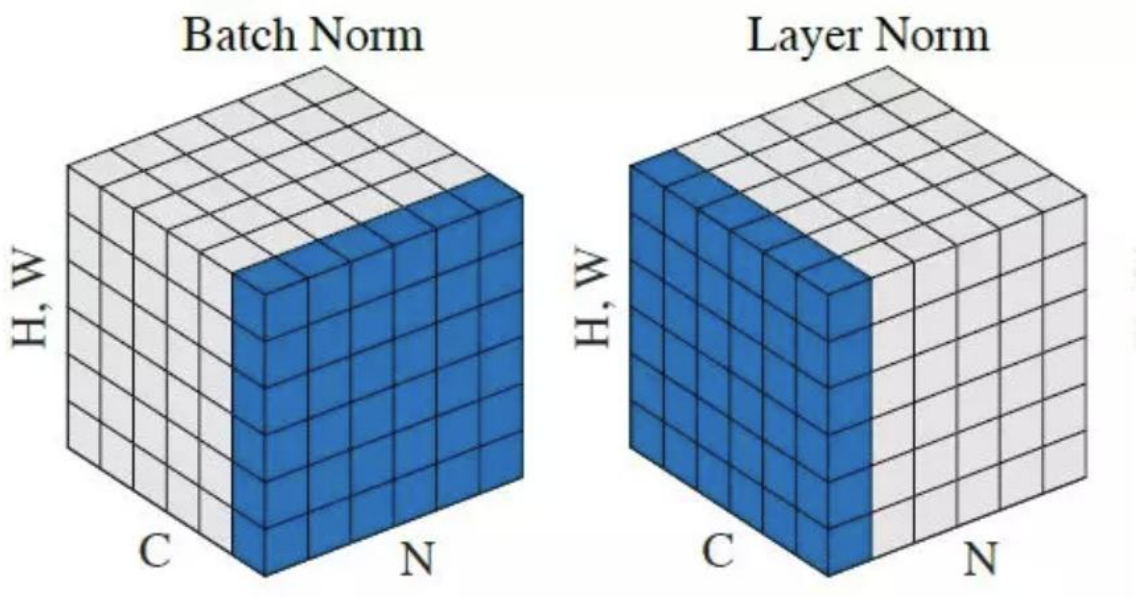
$$\hat{x}_{i,j} = \frac{x_{ij} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}$$

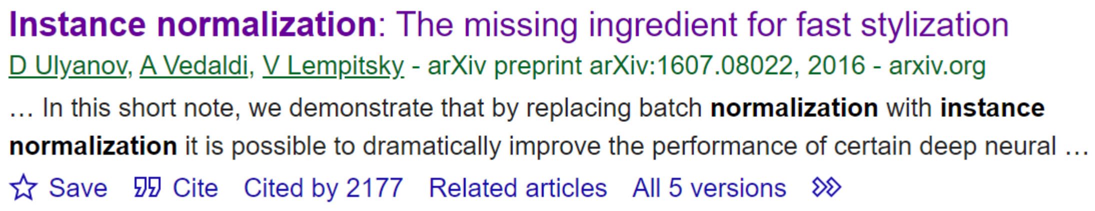
→ normalize

Batch Normalization vs. Layer Normalization



For Images





Instance normalization: The missing ingredient for fast stylization

D Ulyanov, A Vedaldi, V Lempitsky - arXiv preprint arXiv:1607.08022, 2016 - arxiv.org

… In this short note, we demonstrate that by replacing batch **normalization** with **instance normalization** it is possible to dramatically improve the performance of certain deep neural …

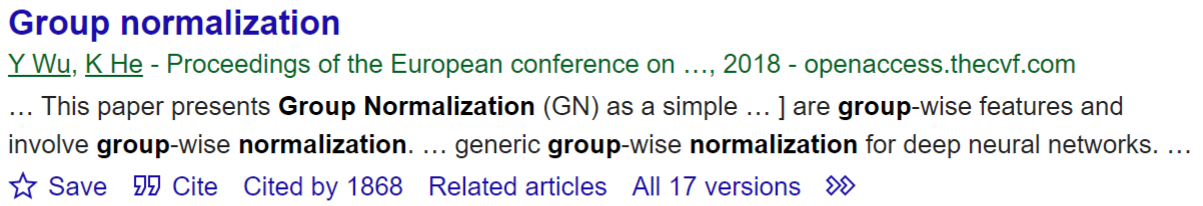
Instance Normalization

- Normalize across each channel
- k, j: index image height and width
- i: channel index (if the input is RGB, then it is a color channel)
- t: index of the image in the batch
- Used in style transfer with generative adversarial network

$$y_{tijk} = \frac{x_{tijk} - \mu_{ti}}{\sqrt{\sigma_{ti}^{2} + \epsilon}}$$

$$\mu_{ti} = \frac{1}{HW}\sum_{l=1}^{W}\sum_{m=1}^{H} x_{tilm}$$

$$\sigma^{2}_{ti} = \frac{1}{HW}\sum_{l=1}^{W}\sum_{m=1}^{H}(x_{tilm} - \mu_{ti})^{2}$$



Group normalization

Y Wu, K He - Proceedings of the European conference on …, 2018 - openaccess.thecvf.com

… This paper presents **Group Normalization** (GN) as a simple …] are **group**-wise features and involve **group**-wise **normalization**. … generic **group**-wise **normalization** for deep neural networks. …

Group Normalization

- Divide channels into groups
- Normalize inside each group and each sample
- Don’t depend on batch size

$$\mu_i = \frac{1}{m}\sum_{k\in S_i} x_k,$$

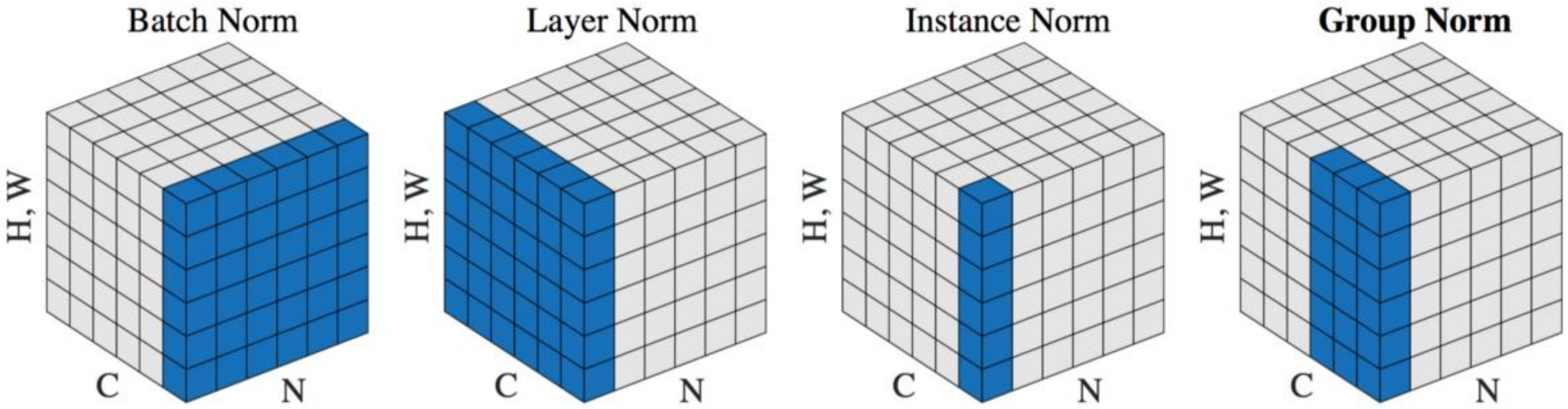
$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k\in S_i}(x_k - \mu_i)^2 + \epsilon}$$

$$S_i = \left\{k \mid K_N = i_N, \left\lfloor\frac{k_C}{C/G}\right\rfloor = \left\lfloor\frac{i_C}{C/G}\right\rfloor\right\}$$

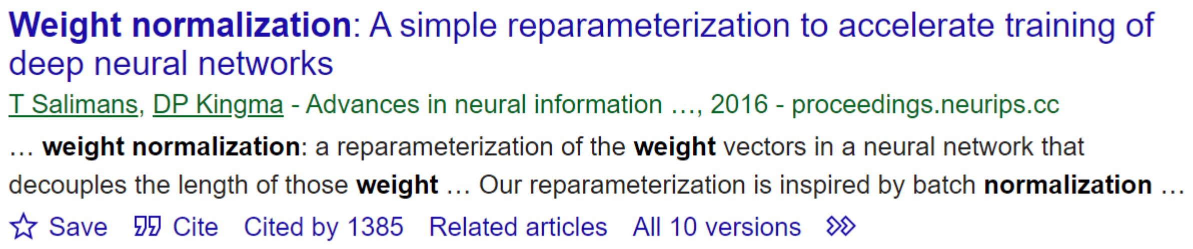
$$\hat{x}_i = \frac{1}{\sigma_i}(x_i - \mu_i)$$

$$y_i = \gamma\hat{x}_i + \beta$$

BN, LN, IN, GN


Batch Norm
H, W
C
N
Layer Norm
H, W
C
N
Instance Norm
H, W
C
N
Group Norm
H, W
C
N

Weight Normalization



Weight normalization: A simple reparameterization to accelerate training of deep neural networks

T Salimans, DP Kingma - Advances in neural information …, 2016 - proceedings.neurips.cc

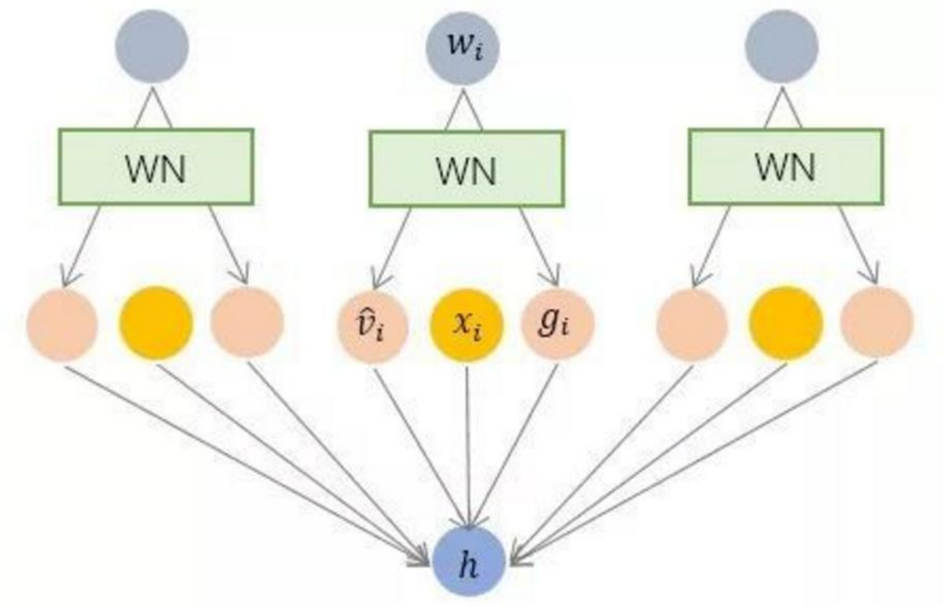
… **weight normalization**: a reparameterization of the **weight** vectors in a neural network that decouples the length of those **weight** … Our reparameterization is inspired by batch **normalization** …

☆ Save 99 Cite Cited by 1385 Related articles All 10 versions »

- Normalize the weight vector by its norm, to restrain the function class
- After normalization, the direction of the weight vector remains the same, but its norm changes to g.

$$\mathbf{w} = g \cdot \hat{\mathbf{v}} = g \cdot \frac{\mathbf{v}}{||\mathbf{v}||}$$

$$\sigma = ||\mathbf{v}||, \quad \mu = 0, \quad \mathbf{b} = 0$$

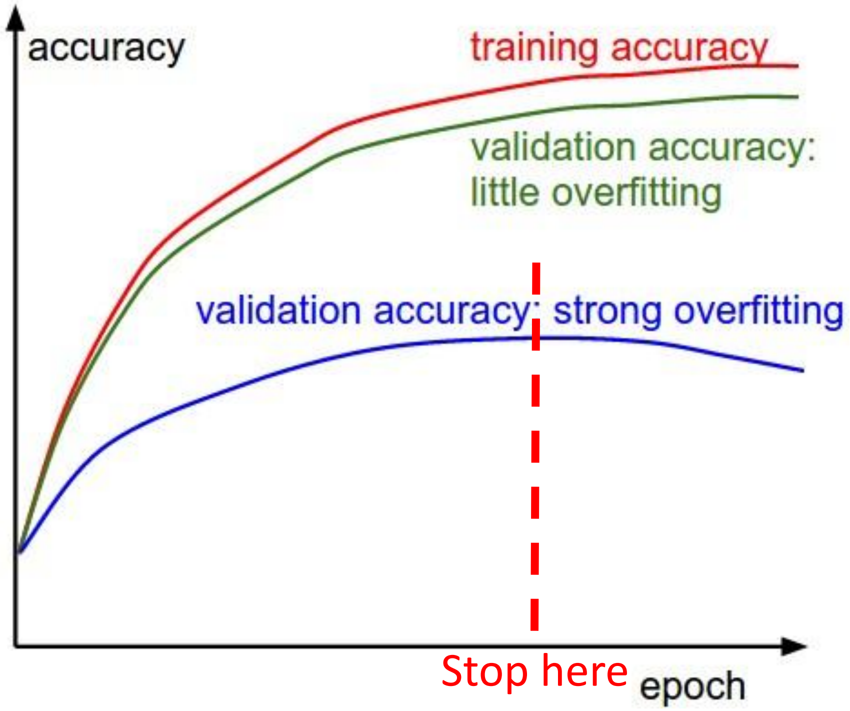


Other Regularization Strategies

- Weight decay (or L^2 parameter norm penalty)

$$J(\theta) = \frac{\alpha}{2}\theta^T\theta + L(\theta)$$
$$\Delta_\theta J(\theta) = \alpha\theta + \Delta_\theta L(\theta)$$
$$\theta \leftarrow \theta - \epsilon(\alpha\theta + \Delta_\theta L(\theta))$$
$$\theta \leftarrow (1 - \alpha\epsilon)\theta - \epsilon\Delta_\theta L(\theta)$$

- Early stopping
  - Terminate the training when the validation error does not drop for a certain iterations



Summary

- Fully connected neural networks
- Convolutional neural networks
- Recurrent neural networks
- Transformer networks

- Optimization: SGD
- Model Initialization
- Generalization