



Regularization

Rowel Atienza, Ph.D.

University of the Philippines

github.com/roatienza

2023

Generalization

After training, the model should be validated using data never seen before: $\mathcal{D}_{test} = \{(\mathbf{x}_m, y_m)\}, \quad n = 1 \dots M$

The test scores called generalization performance are the ones reported

Issue: Overfitting and Memorization

It is possible that for a given training set, a model with sufficient complexity is able to memorize the input-output data

This leads to a problem called **overfitting**

Regularization

To prevent overfitting, a regularizer is used

Examples: weight penalty, noise injection, inductive bias on dataset, novel model architecture, dropout, data augmentation

Weight Penalty: Scaled Dot Product of Parameters

$$R_{emp}(f, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^n (y_n - f(\mathbf{x}_n, \boldsymbol{\theta}))^2 + \lambda \boldsymbol{\theta}^T \boldsymbol{\theta}$$

Tendency is to choose $\boldsymbol{\theta}$ that have small absolute values to reduce R_{emp}

Probabilistic Parameter Estimation

Maximum Likelihood Estimation (MLE)

Function of parameters that explain the data well:

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\mathcal{D}|\boldsymbol{\theta})$$

The likelihood of parameters $\boldsymbol{\theta}$ given that we observed data \mathcal{D}

Maximum because the loss function is minimized when $p(\mathcal{D}|\boldsymbol{\theta}) \rightarrow 1.0$

Log does not change the location of minima

Supervised Learning using Gaussian Distribution

$$\mathcal{D}_{train} = \{(\mathbf{x}_n, y_n)\}, \quad n = 1 \dots N$$

Target model: $p(\mathcal{D}|\boldsymbol{\theta}) \rightarrow p(\hat{y}_n|\mathbf{x}_n, \boldsymbol{\theta})$

Assume a Gaussian distribution with mean as a linear function $\mu = \mathbf{x}_n^T \boldsymbol{\theta}$ and with the parameter perturbed by a Gaussian noise with zero mean and variance σ^2 :

$$p(\hat{y}_n|\mathbf{x}_n, \boldsymbol{\theta}) = \mathcal{N}(\hat{y}_n|\mathbf{x}_n^T \boldsymbol{\theta}, \sigma^2)$$

Independent Identically Distributed (IID) Assumption

Independent : probabilities can be multiplied $p(a, b) = p(a)p(b)$

Identically Distributed : can share parameters

We can factorize the data distribution:

$$p(\mathcal{D}|\boldsymbol{\theta}) \propto p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = \prod_{n=1}^N p(\hat{y}_n|\mathbf{x}_n, \boldsymbol{\theta})$$

MLE

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^N \log p(\hat{y}_n|\mathbf{x}_n, \boldsymbol{\theta})$$

Since $\log ab = \log a + \log b$

If $p(\hat{y}_n | \mathbf{x}_n, \boldsymbol{\theta})$ is a Gaussian

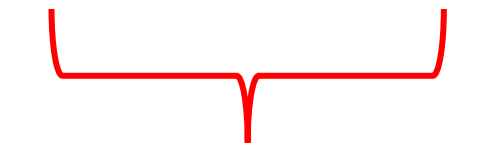
$$p(\mathcal{D} | \boldsymbol{\theta}) = p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) = \prod_{n=1}^N \mathcal{N}(\hat{y}_n | \mathbf{x}_n^T \boldsymbol{\theta}, \sigma^2)$$

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^N \log \mathcal{N}(\hat{y}_n | \mathbf{x}_n^T \boldsymbol{\theta}, \sigma^2)$$

If $p(\hat{y}_n | \mathbf{x}_n, \boldsymbol{\theta})$ is a Gaussian

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = - \sum_{n=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \left(\frac{(\hat{y}_n - \mathbf{x}_n^T \boldsymbol{\theta})^2}{2\sigma^2} \right)$$

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = \sum_{n=1}^N \left(\frac{(\hat{y}_n - \mathbf{x}_n^T \boldsymbol{\theta})^2}{2\sigma^2} \right) - \sum_{n=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}}$$



Squared Losses



Constant

Maximum A Posteriori (MAP) Principle

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

Since $p(\mathcal{D})$ is not a function of $\boldsymbol{\theta}$, it does not affect the optimization

$$\log p(\boldsymbol{\theta}|\mathcal{D}) = \log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$$

$$\operatorname{argmax}_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\mathcal{D}) = \operatorname{argmax}_{\boldsymbol{\theta}} (\log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}))$$

Maximum A Posteriori (MAP) Principle

$$\operatorname{argmax}_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\mathcal{D}) = \operatorname{argmin}_{\boldsymbol{\theta}} \underbrace{(-\log p(\mathcal{D}|\boldsymbol{\theta}))}_{\text{MLE}} - \underbrace{\log p(\boldsymbol{\theta})}_{\text{Regularizer}}$$

If we assume $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, then:

$$-\log p(\boldsymbol{\theta}) = \lambda \boldsymbol{\theta}^T \boldsymbol{\theta}$$

Capacity

Capacity - ability to fit a wide variety of functions

↓ Capacity → Underfitting: ↑ MSE(train) , ↑ MSE(test)

↑ Capacity → Overfitting: ↓ MSE(train) , ↑ MSE(test)

✓ Capacity → Optimal Fit: ↓ MSE(train) , ↓ MSE(test)

Sample Data

Input: x is a random number from -3 to +3

Output: $f(x) = y = x^2 + x + 1 + \lambda \sin \beta x$



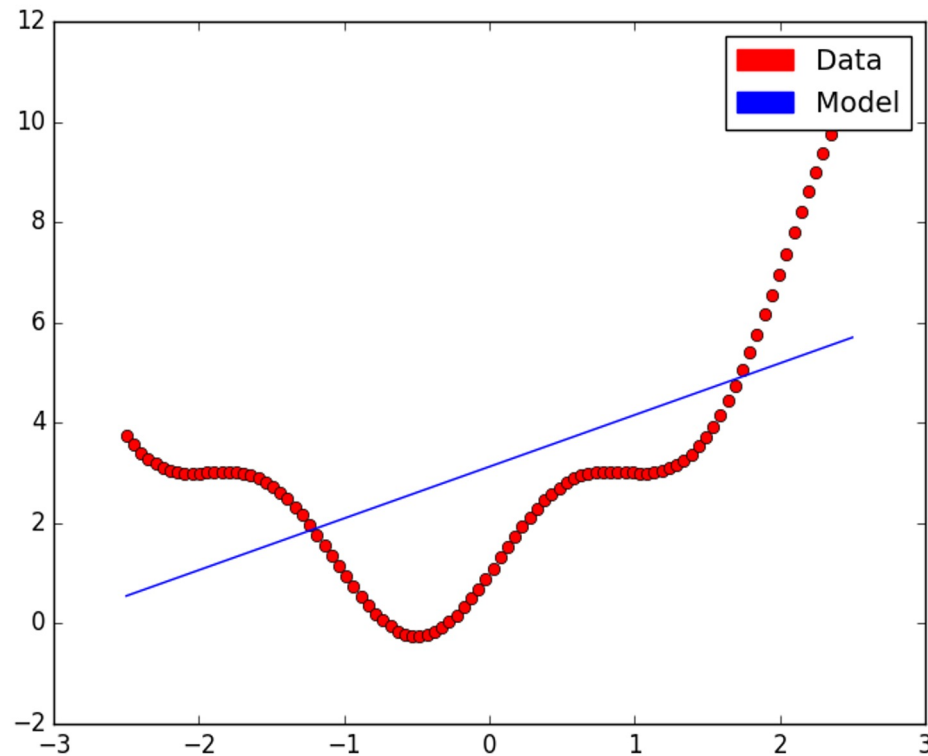
Noise

Model: Our model should be a 2nd degree polynomial

Suppose we can only see the data generated by $f(x)$

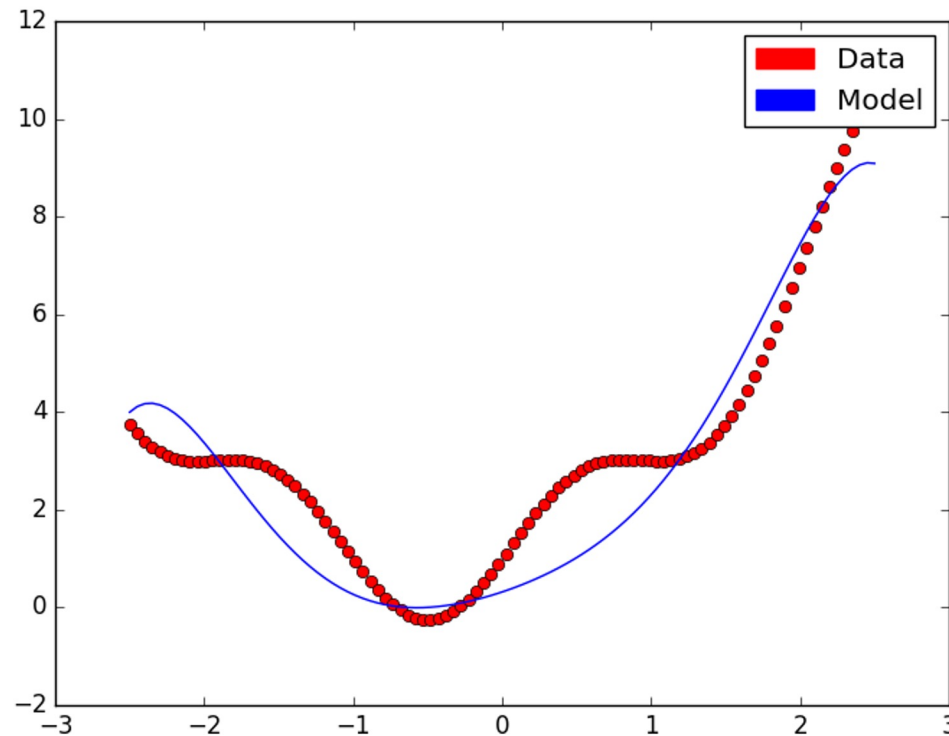
Model: $f(x) = y = \theta_1 x + \theta_0$

Underfitting: the model has both big train and test error



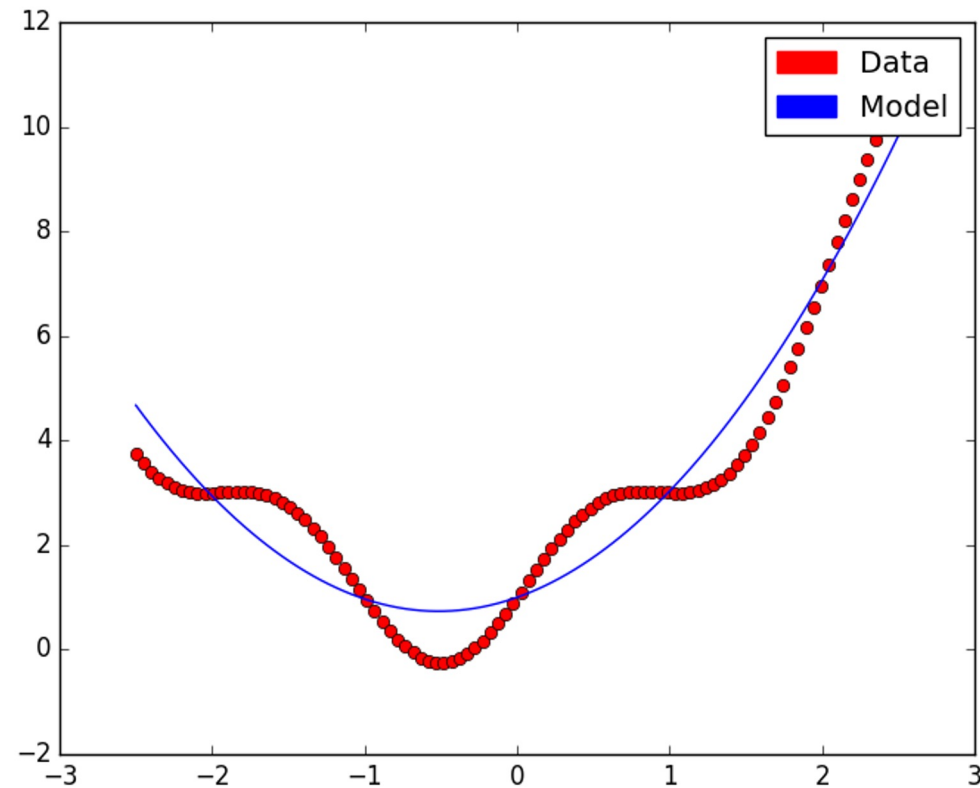
Model: $f(x) = y = \theta_6 x^6 + \dots + \theta_1 x + \theta_0$

Overfitting: the model has a small train error but has a big test error



Model: $f(x) = y = \theta_2 x^2 + \theta_1 x + \theta_0$

Optimal fit: the model has a small train and test errors



Data Augmentation

Increasing the amount of training data by creating fake data

Effective in computer vision (eg image of a dog rotated, resized, translated, etc) and speech



Original



Flipped



Rotated

Data Augmentation

Some image data are not amenable to geometric transformation as the meaning is altered

For example,

A large, dark blue handwritten digit 'b' in a cursive style.

Orig

A large, dark blue handwritten digit 'q' in a cursive style, which is a 90-degree clockwise rotation and a vertical flip of the original 'b'.

Rot/Flipped

A large, dark blue handwritten digit 'd' in a cursive style, which is a horizontal mirror image of the original 'b'.

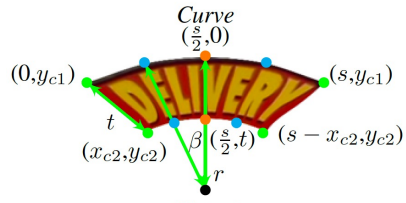
Mirrored

Data Augmentation

Other operations

1. Scaling
2. Translating
3. Minor distortion
4. Normalization
5. ZCA Whitening
6. etc

STRAug: Data Augmentation for STR



Warp



Geometry



Figure 6. Examples of images affected by *Noise* data augmentation.



Figure 7. Example images affected by *Blur* data augmentation.

Noise and Blur



Figure 8. Example images affected by *Weather* data augmentation.

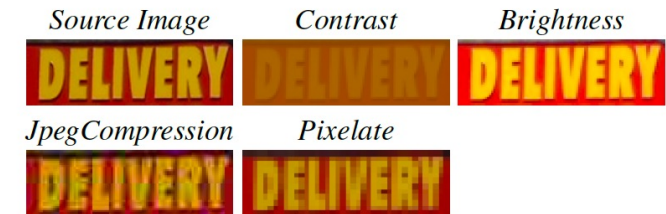


Figure 9. Example images affected by *Camera* data augmentation.



Figure 10. Example images affected by *Process* data augmentation.

Weather, Camera, Process

Atienza, Rowel. "Data Augmentation for Scene Text Recognition." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021.

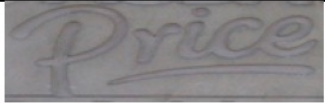





Model	Input Text Image	Baseline Prediction	+STRAug Prediction	%Acc Gain
CRNN [33]		Pyice	Price	1.30
R2AM [22]		OLDTOW	OLDTOWN	1.48
GCRNN [36]		TiMmES	Times	0.89
Rosetta [5]		eizu	eBizu	2.10
RARE [34]		Washi1	Washing	1.35
TRBA [3]		insi1d	inside	1.06

Figure 1. STRAug data augmentation significantly improves the overall accuracy of STR models especially on challenging input text images. We follow the evaluation protocol used in most STR models of case sensitive training and case insensitive validation.

STRAug on a Transformer-based STR

	Significant gains												
ViTSTR-Tiny	83.7	83.2	92.8	92.5	90.8	89.3	72.0	66.4	74.5	65.0	80.3	0.2	
ViTSTR-Tiny+Aug	85.1	85.0	93.4	93.2	90.9	89.7	74.7	68.9	78.3	74.2	82.1	0.1	
ViTSTR-Small	85.6	85.3	93.9	93.6	91.7	90.6	75.3	69.5	78.1	71.3	82.6	0.3	
ViTSTR-Small+Aug	86.6	87.3	94.2	94.2	92.1	91.2	77.9	71.7	81.4	77.9	84.2	0.1	
ViTSTR-Base	86.9	87.2	93.8	93.4	92.1	91.3	76.8	71.1	<u>80.0</u>	74.7	83.7	0.1	
ViTSTR-Base+Aug	88.4	87.7	94.7	94.3	93.2	92.4	78.5	72.6	81.8	81.3	85.2	0.1	

Atienza, Rowel. "Vision transformer for fast and efficient scene text recognition." *International Conference on Document Analysis and Recognition*. Springer, Cham, 2021.

Data Augmentation – Key Concepts

x_1 :



y_1 : bird

x_2 :



y_2 : cat

CutOut

x_1 :



Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. arXiv:1708.04552, 2017

y_1 : bird

MixUp

$$\tilde{x} = \lambda x_1 + (1 - \lambda)x_2$$

Mix of bird and cat

$$\tilde{y} = \lambda y_1 + (1 - \lambda)y_2$$

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. MixUp: Beyond empirical risk minimization. ICLR 2018.



CutMix

$$\tilde{x} = \mathbf{M} \odot x_1 + (\mathbf{1} - \mathbf{M})x_2$$

Mix of bird and cat

$$\tilde{y} = \lambda y_1 + (1 - \lambda)y_2$$

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. CVPR 2019.



Agreement Maximization (AgMax)

Force the model prediction to agree on 2 different augmentations of the same input

Atienza, Rowel. "Improving Model Generalization by Agreement of Learned Representations from Data Augmentation." Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2022.

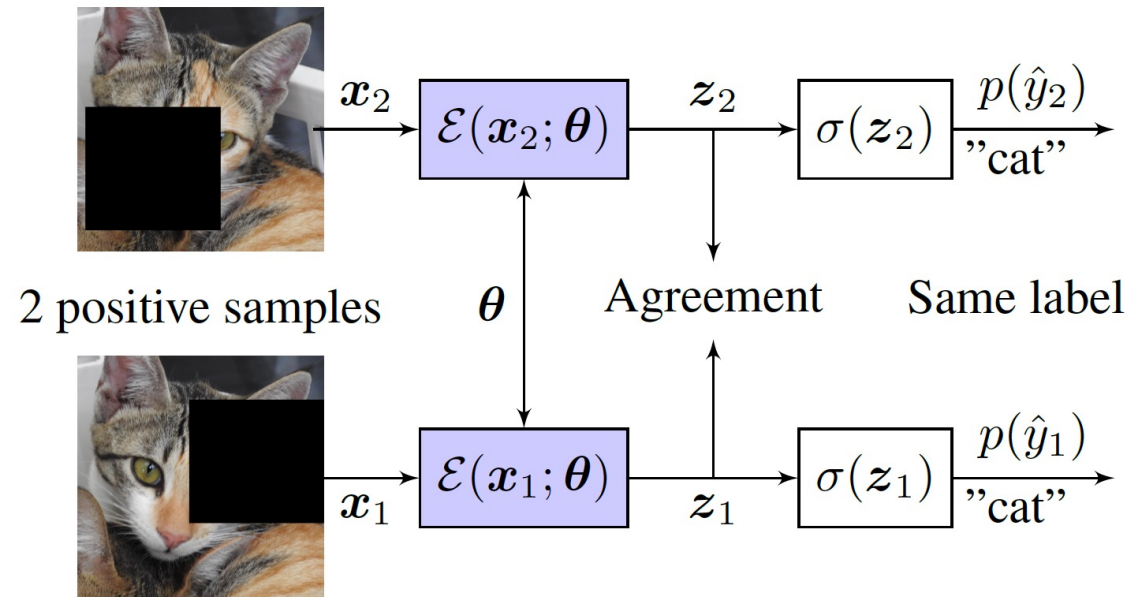
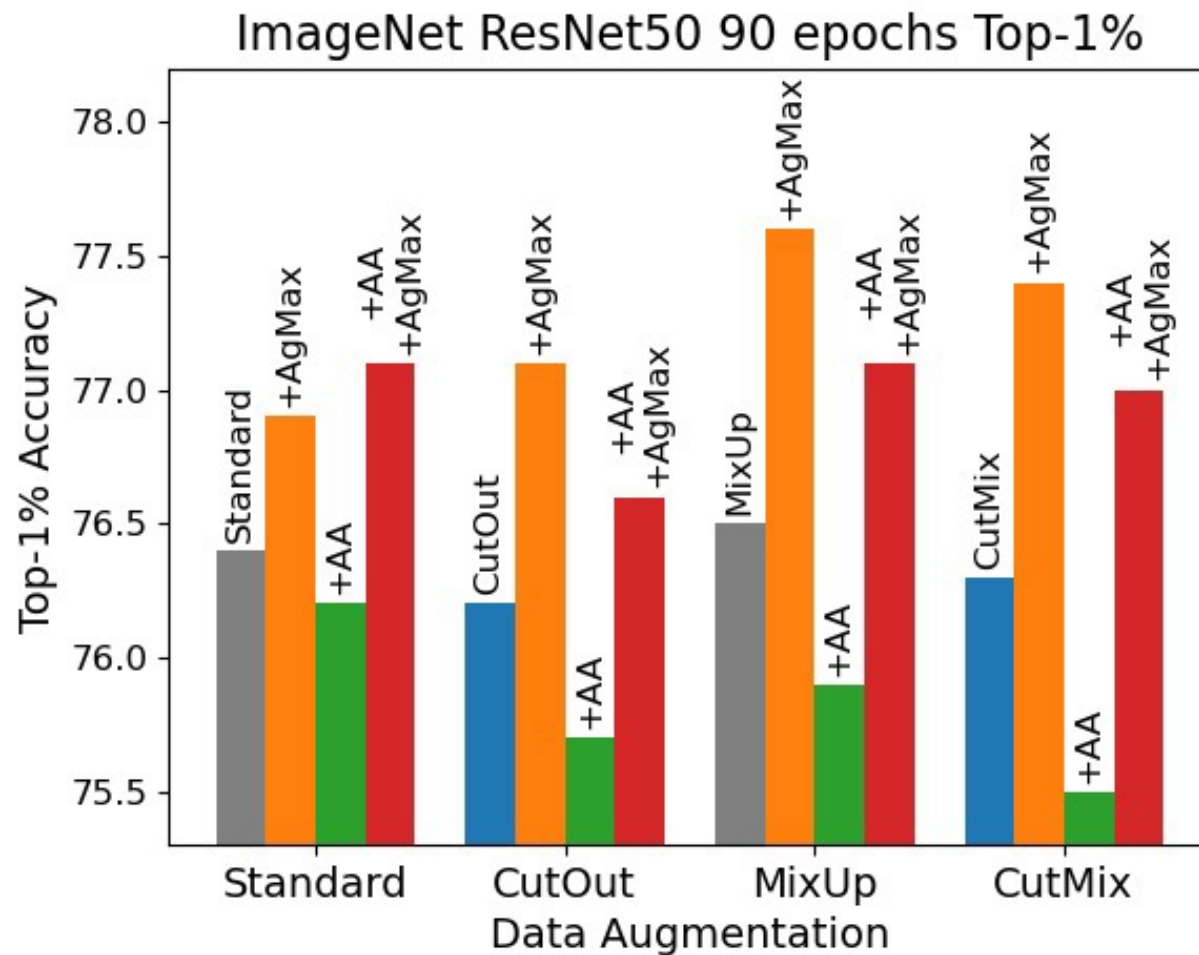


Figure 2. In supervised learning with *AgMax*, we impose an additional constraint that representations z_1 and z_2 must also agree. The two parallel models are just one and the same and share the same set of parameters θ . x_1 and x_2 are 2 positive samples with the same label.

AgMax on ImageNet1k using ResNet50



RandAugment

Policy for combining multiple data augmentations

Cubuk, Ekin D., et al. "Randaugment: Practical automated data augmentation with a reduced search space." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020.

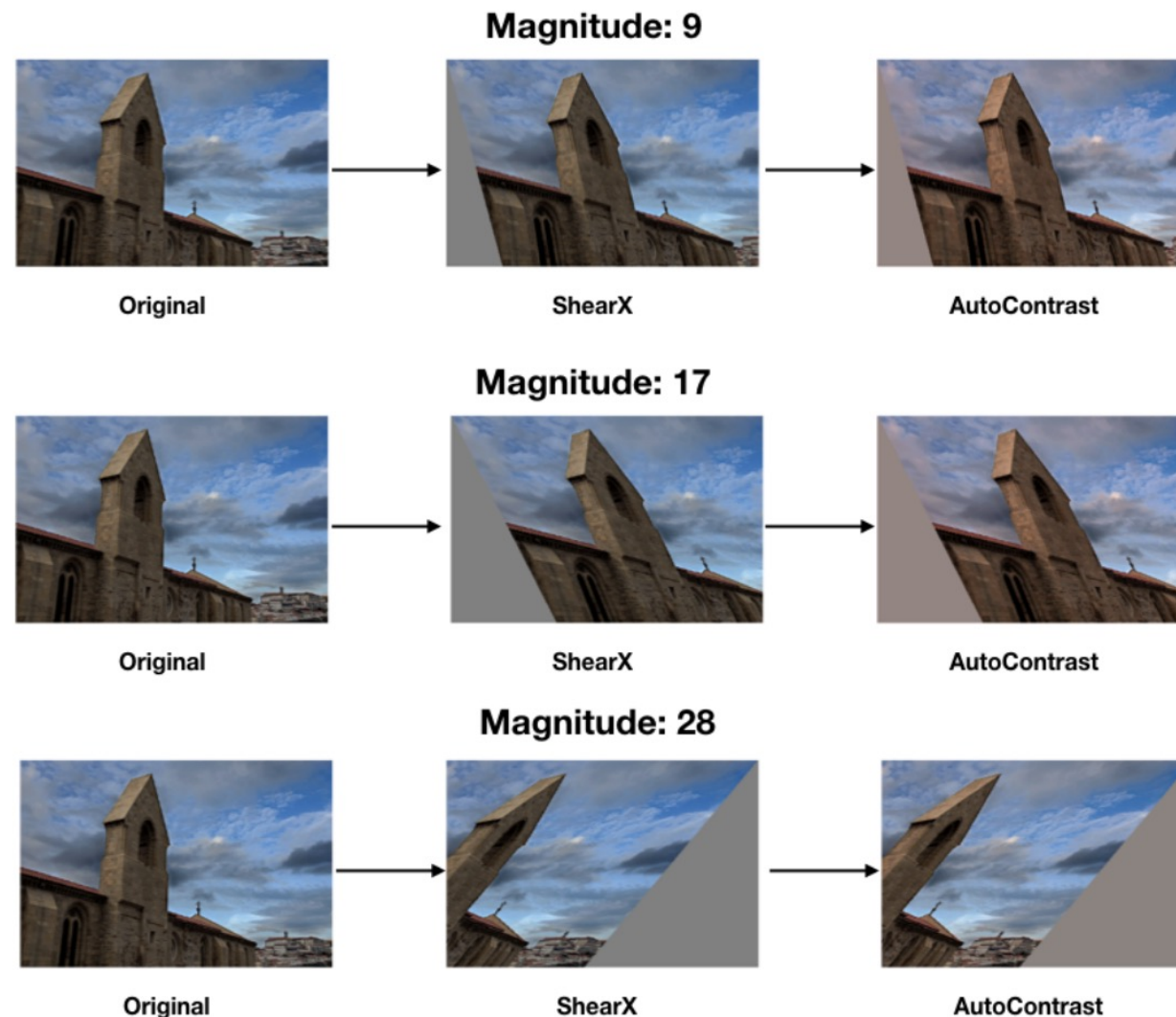


Figure 1. **Example images augmented by RandAugment.** In these examples $N=2$ and three magnitudes are shown corresponding to the optimal distortion magnitudes for ResNet-50, EfficientNet-B5 and EfficientNet-B7, respectively. As the distortion magnitude increases, the strength of the augmentation increases.

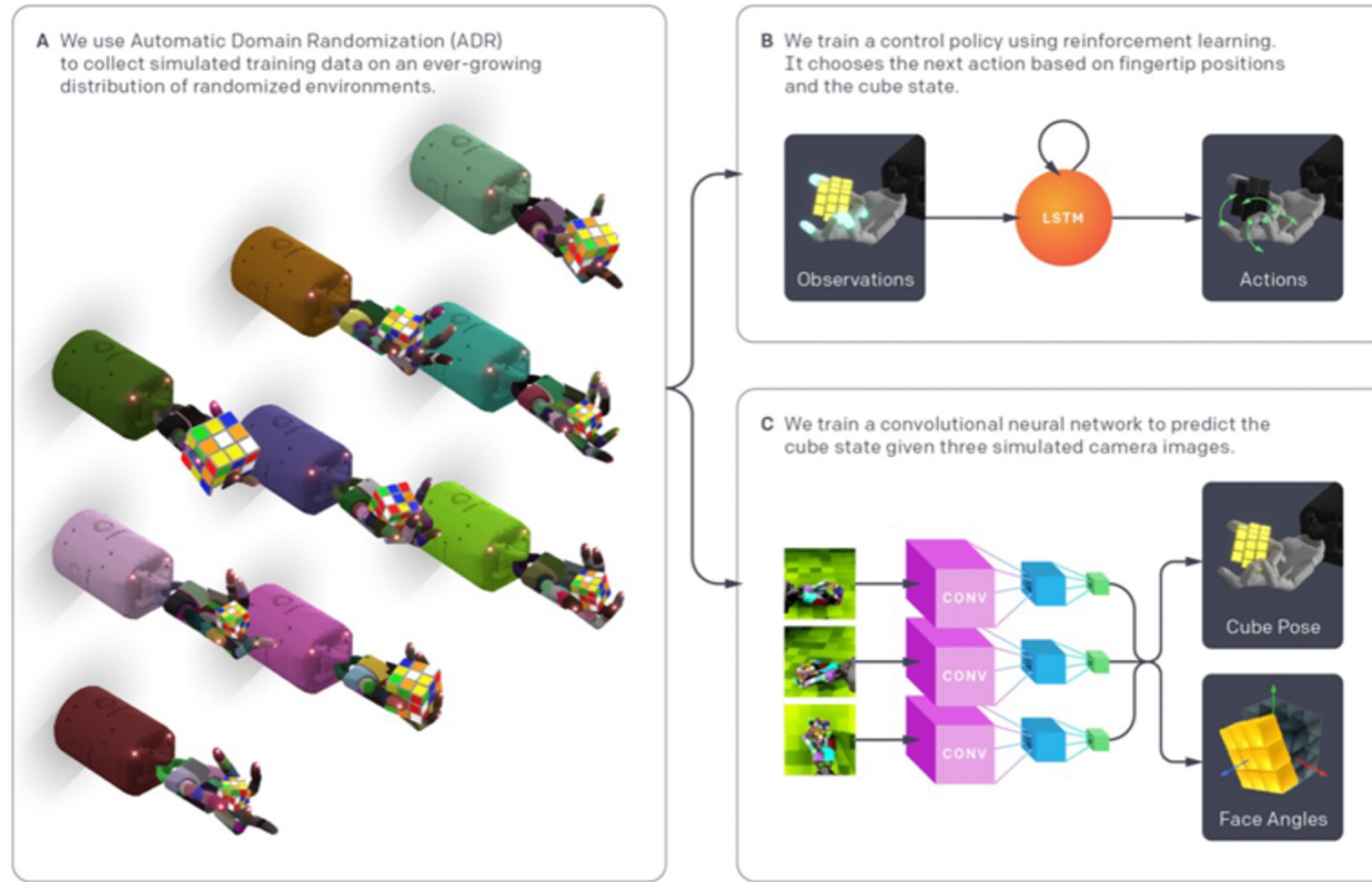
Data Augmentation

Other methods of Dataset Augmentation: Adding noise to input

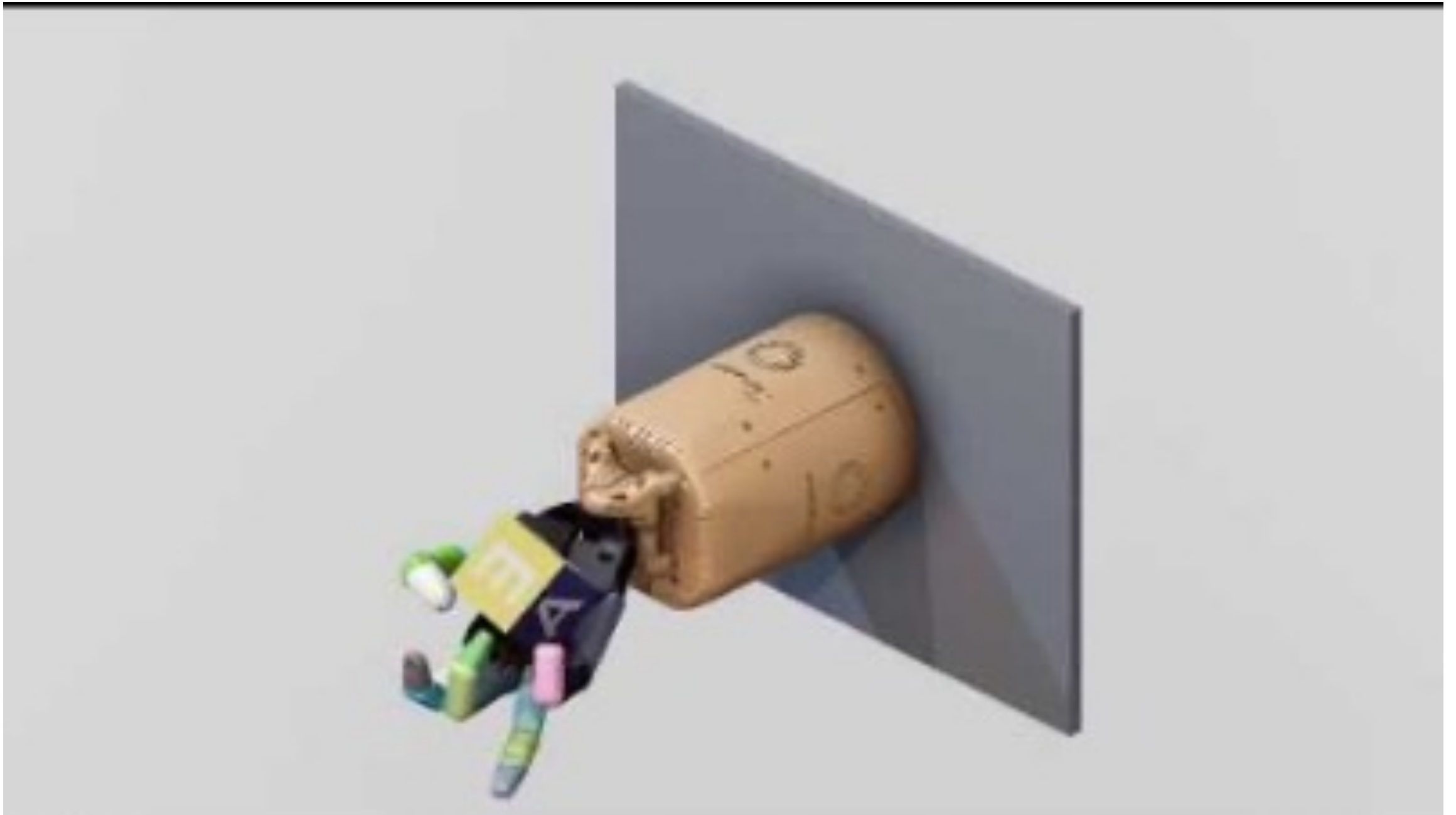
- Neural Networks are inherently not insensitive to noise
- Training a NN with a small amount of noise added to the data can make the network more robust

Automatic Domain Randomization (ADR)

Train in Simulation



Tobin, Josh, et al. "Domain randomization and generative models for robotic grasping." *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.



<https://youtu.be/FVl8OkNXiDk>



TWO MINUTE PAPERS

WITH KÁROLY ZSOLNAI FEHÉR (KZEF)

TRANSFERRING AI TO THE REAL WORLD

Disclaimer: we are not part of this research project, this is merely providing commentary on this work.

GOO: A Dataset for Gaze Object Prediction in Retail Environments

Tomas, Reyes, Dionido, Ty,
Mirando, Casimiro, Atienza, Guinto

University of the Philippines & Samsung R&D PH

CVPR Workshops 2021

Tomas, Henri, et al. "Goo: A dataset for gaze object prediction in retail environments." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

Noise as Weights Regularizer

Noise can also be added to weights

In Bayesian Approach, model weights are stochastic

Adding noise is one way to model stochasticity

For example, MSE: $L = (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y})$

Suppose the weights are corrupted

$$\mathbf{w} + \text{noise} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

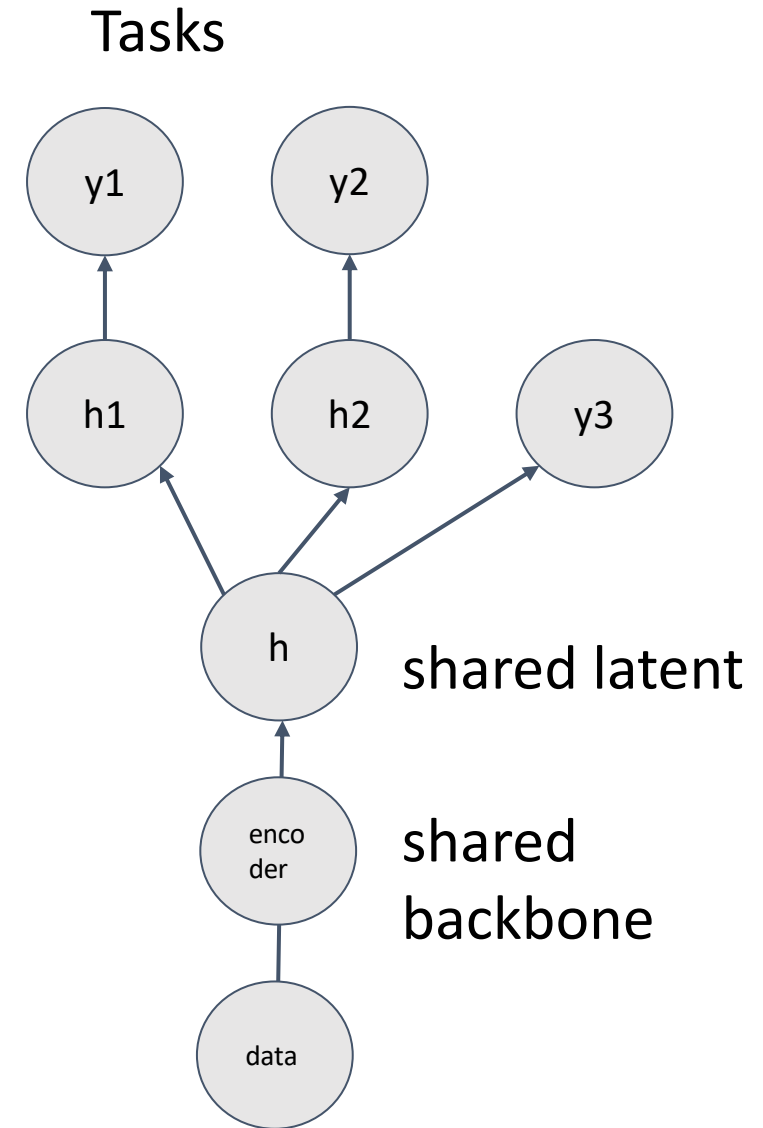
$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} - \text{noise}$$

The constraints drive the values of \mathbf{w} to a flat region such that a small perturbation will not alter the prediction

Multi-task Learning

Same data is used to learn multiple tasks

- This is a form of regularization since the network is subjected to soft constraints
- Shared parameters across multiple tasks
- In the factors that explain variations of observed data across multiple tasks, some are shared among multiple tasks
- y_1 may be classification, y_2 may be bounding box detection, y_3 may segmentation
- Example: Mask RCNN



Parameter Sharing

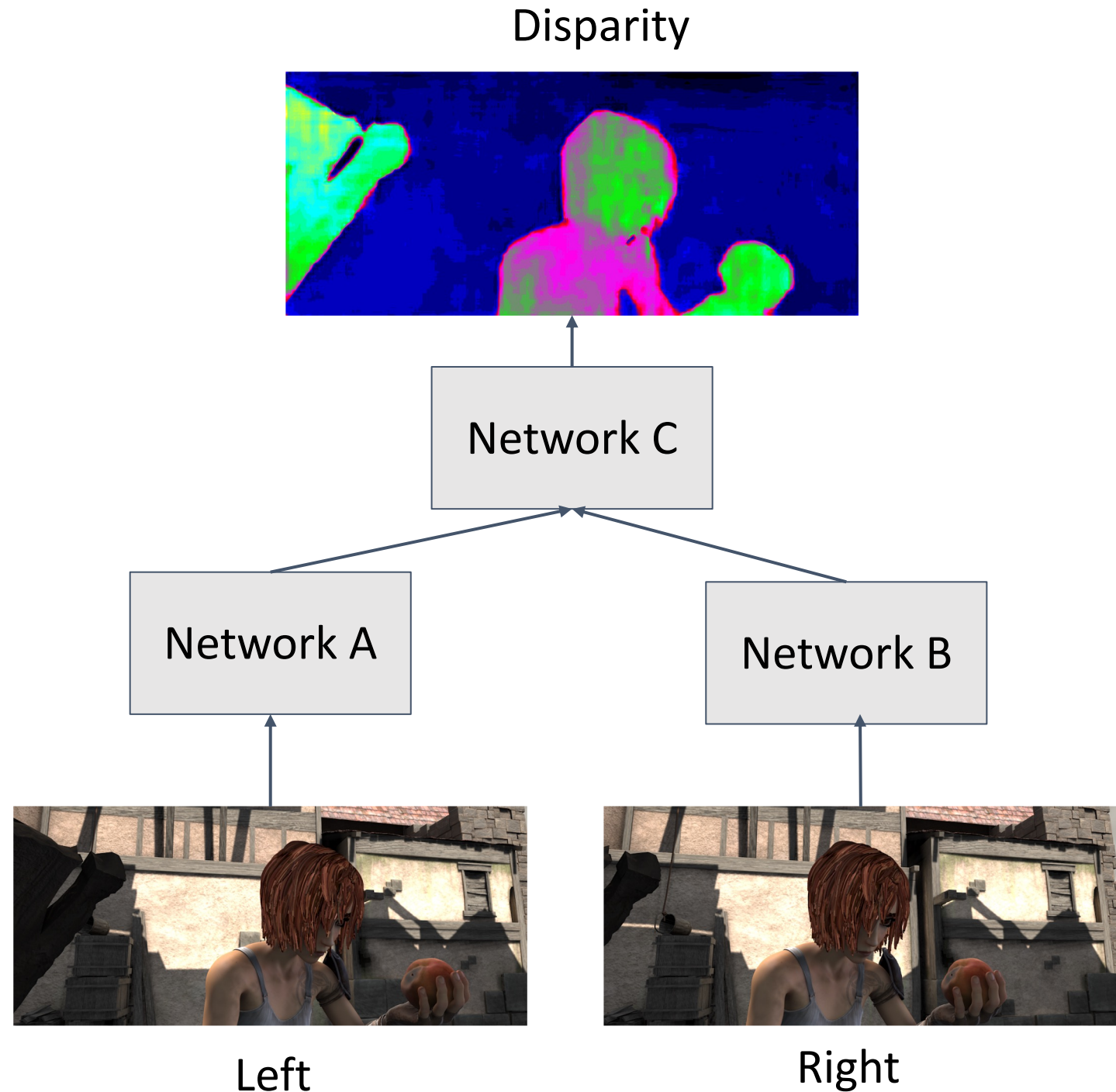
Two different networks doing similar tasks can share parameters

Very common in Siamese Networks

Example, Networks A and B share the same set of parameters in a Siamese Network to perform disparity estimation

Forces Networks A and B to be more robust by forcing them to work on different images from the same distribution

Atienza, Rowel. "Fast disparity estimation using dense networks." *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018.



Early Stopping

Initially, validation error decreases as training error decreases

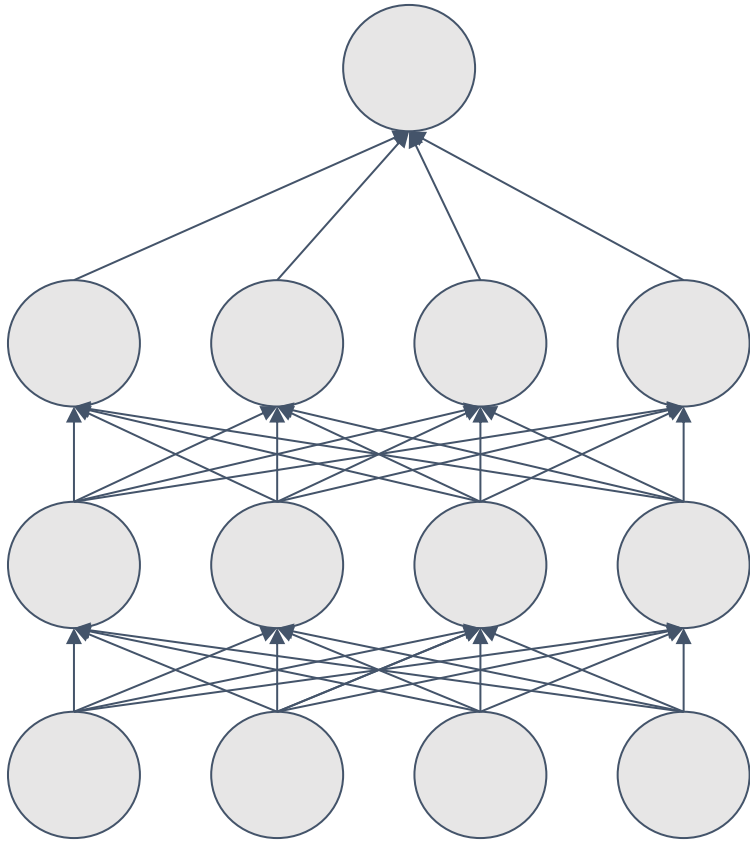
As training continues, the training error decreases while the validation error increases; network is overfitting

Early Stopping stops the training and saves the parameters when the validation error starts to increase

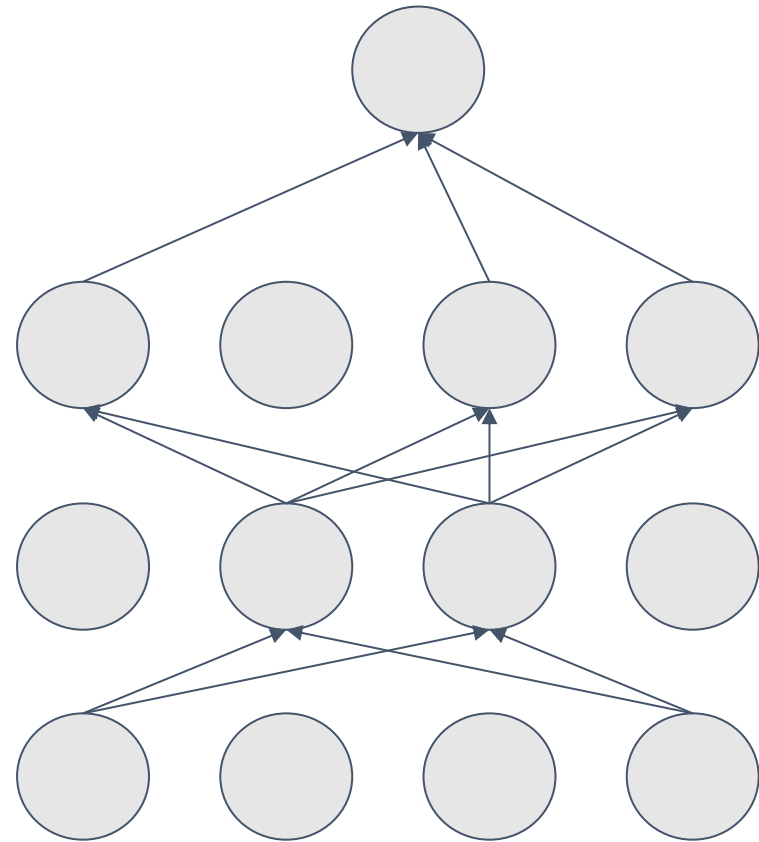
Early Stopping is a form of regularization since it limits the parameter space

Dropout

Strongly inspired by biological processes

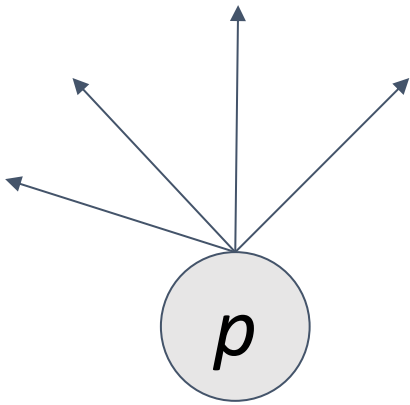


Standard Deep Neural
Network

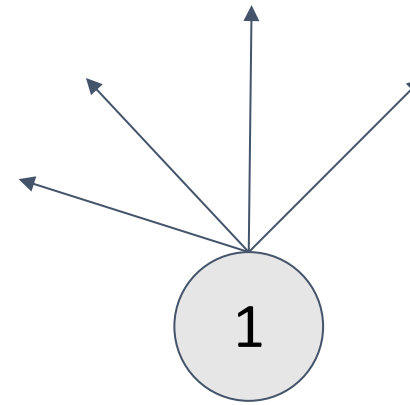


Standard Deep Neural Network
With Dropout

Dropout



During training, a neuron will be included in the network with probability p



During test, a neuron will always be included in the network

Cross-Validation: Use when there is a small dataset (100s to few 1000s)

For Example, 4-fold Validation



Reference

Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press, 2016, <http://www.deeplearningbook.org>

Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Srivastava, et. al. Journal of Machine Learning, 2014

Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong , "Mathematics for Machine Learning". Copyright 2020 by. Published by Cambridge University Press.

In Summary

Regularization prevents a model from overfitting

Regularization algorithms include L2/L1, data augmentation, noise injection, parameter sharing, dropout

